2-Matchings, the Traveling Salesman Problem, and the Subtour LP: A Proof of the Boyd-Carr Conjecture

Frans Schalekamp
Department of Mathematics, College of William and Mary, Williamsburg, Virginia 23185,
frans@wm.edu, franssschalekamp.com

David P. Williamson
School of Operations Research and Information Engineering, Cornell University, Ithaca, New York 14853,
dpw@cs.cornell.edu, www.davidpwilliamson.net/work

Anke van Zuylen
Department of Mathematics, College of William and Mary, Williamsburg, Virginia 23185, anke@wm.edu, ankevanzuylen.com


In this paper, we prove the Boyd-Carr conjecture. In the case that the support of a fractional 2-matching has no cut edge, we can further prove that an optimal 2-matching has cost at most 10/9 times the cost of the fractional 2-matching.

**Keywords:** traveling salesman problem; subtour elimination; linear programming; integrality gap; 2-matching; fractional 2-matching

**MSC2000 subject classification:** Primary: 90C05 linear programming, 90C27 combinatorial optimization, 05C70

**History:** Received June 7, 2012; revised December 29, 2012, April 2, 2013, and April 7, 2013. Published online in Articles in Advance August 1, 2013.

## 1. Introduction.

The traveling salesman problem (TSP) is the most famous problem in discrete optimization. Given a set of \( n \) cities and the costs \( c(i, j) \) of traveling from city \( i \) to city \( j \) for all \( i, j \), the goal of the problem is to find the least expensive tour that visits each city exactly once and returns to its starting point. An instance of the TSP is called symmetric if \( c(i, j) = c(j, i) \) for all \( i, j \); it is asymmetric otherwise. Costs obey the triangle inequality if \( c(i, j) \leq c(i, k) + c(k, j) \) for all \( i, j, k \). The TSP is known to be NP-hard, even in the case that instances are symmetric and obey the triangle inequality. From now on we consider only these instances unless otherwise stated.

Because of the NP-hardness of the traveling salesman problem, researchers have considered approximation algorithms for the problem. An \( \alpha \)-approximation algorithm for the traveling salesman problem is a polynomial-time algorithm that returns a tour of cost at most a factor of \( \alpha \) times the cost of an optimal tour. The best approximation algorithm currently known is a \( \frac{1}{2} \)-approximation algorithm given by Christofides in 1976 (Christofides [9]). Better approximation algorithms are known for special cases. Exciting progress has been made recently in the case of graph-TSP, in which costs \( c(i, j) \) are given by shortest path distances in an unweighted graph; we will discuss these results shortly. However, to date, Christofides’ algorithm has the best known performance guarantee for the general case.

There is a well-known, natural direction for making progress that has also defied improvement for nearly 30 years. The following linear programming (LP) relaxation of the traveling salesman problem was used by Dantzig et al. [10] in 1954. For simplicity of notation, we let \( G = (V, E) \) be a complete undirected graph on \( n \) nodes. We use \((i, j)\) to denote the (undirected) edge between \( i \) and \( j \). In the LP relaxation, we have a variable \( x(e) \) for all \( e = (i, j) \) that denotes whether we travel directly between cities \( i \) and \( j \) on our tour. Let \( c(e) = c(i, j) \) and let \( \delta(S) \) denote the set of all edges with exactly one endpoint in \( S \subseteq V \). Then the relaxation is

\[
\min \sum_{e \in E} c(e)x(e) \\
\text{subject to: } \sum_{e \in \delta(i)} x(e) = 2, \quad \forall i \in V,
\]

(1)
\[
\sum_{e \in \delta(S)} x(e) \geq 2, \quad \forall S \subset V, \quad 3 \leq |S| \leq |V| - 3
\]
(2)
\[
0 \leq x(e) \leq 1, \quad \forall e \in E.
\]
(3)

Constraints (1) are called the \textit{degree constraints}. Constraints (2) are sometimes called \textit{subtour elimination constraints} or sometimes just \textit{subtour constraints} since they prevent solutions in which there is a subtour of just the nodes in \(S\). As a result, the linear program is sometimes called the \textit{subtour LP}. It is known that the equality sign in the first set of constraints may be replaced by \(\geq\) in the case in which the costs obey the triangle inequality (Goemans and Bertsimas [15]; see also Williamson [27]).

The LP is known to give excellent lower bounds on TSP instances in practice, coming within a percentage or two of the length of the optimal tour (see, for instance, Johnson and McGeoch [16]). However, its theoretical worst case is not well understood. In 1980, Wolsey [28] showed that Christofides’ algorithm produces a solution whose value is at most \(\frac{3}{2}\) times the value of the subtour LP (also shown later by Shmoys and Williamson [26]).

This proves that the \textit{integrality gap} of the subtour LP is at most \(\frac{3}{2}\); the integrality gap is the worst-case ratio, taken over all instances of the problem, of the value of the optimal tour to the value of the subtour LP or the ratio of the optimal integer solution to the optimal fractional solution. The integrality gap of the LP is known to be at least \(\frac{4}{3}\) via a specific class of instances. However, no instance is known that has integrality gap worse than this, and it has been conjectured for some time that the integrality gap is at most \(\frac{4}{3}\) (see, for instance, Goemans [14]).

Stronger bounds on the integrality gap are known in the case of graph-TSP. If the graph is cubic and 3-connected, Gamarnik et al. [13] show an approximation algorithm with guarantee slightly better than \(\frac{\sqrt{3}}{2}\), and their result also implies an upper bound on the integrality gap of less than \(\frac{3}{2}\). Aggarwal et al. [1] improve the bound to \(\frac{3}{2}\) for these instances. Boyd et al. [8] and Mömke and Svensson [17] extend the bound of \(\frac{3}{2}\) to arbitrary cubic graphs and to subcubic graphs, respectively. Oveis Gharan et al. [21] show that without any assumptions on the underlying graph, graph-TSP can be approximated to within \(\frac{3}{2} - \varepsilon\) for a small constant \(\varepsilon > 0\), and this implies a bound on the integrality gap of \(\frac{3}{2} - \varepsilon\) for such instances as well. Mömke and Svensson [17] show that the integrality gap is at most 1.461, and Mucha [19] improved this result to \(\frac{13}{9}\). Sebő and Vygen [25] combine the ideas of Mömke and Svensson [17] with an algorithm based on a carefully chosen ear decomposition of the graph to show that the integrality gap for graph-TSP instances is at most \(\frac{\sqrt{2}}{2}\).

There is some evidence that the conjectured gap of \(\frac{4}{3}\) might be true. Benoit and Boyd [4] have shown via computational methods that the conjecture holds for \(n \leq 10\), and Boyd and Elliott-Magwood [7] have extended this to \(n \leq 12\). In a 1995 paper, Goemans [14] showed that adding any class of valid inequalities known at the time to the subtour LP could increase the value of the LP by at most \(\frac{4}{3}\); this is necessary for the conjecture to be true. Somewhat weaker evidence is as follows. A \textit{2-matching} is an integer solution to the subtour LP obeying only the degree constraints (1) and the bounds constraints (3).\footnote{We note that what we refer to here as 2-matchings are also sometimes called 2-factors.} A \textit{fractional 2-matching} is a 2-matching without the integrality constraints. Boyd and Carr [5] have shown that the integrality gap for the 2-matching problem is \(\frac{4}{3}\). Furthermore, Boyd and Carr [6] have shown that if the subtour LP solution is half-integral (that is, \(x(i, j) \in \{0, \frac{1}{2}, 1\}\) for all \(i, j \in V\)) and has a particular structure, then there is a tour of cost at most \(\frac{4}{3}\) times the value of the subtour LP.

Not only do we not know the integrality gap of the subtour LP, Boyd and Carr have observed that we don’t even know the worst-case ratio of the optimal 2-matching to the value of the subtour LP, which is surprising because 2-matchings are well understood and well characterized. They make the following conjecture.

\textbf{Conjecture 1 (Boyd and Carr [6]).} \textit{The worst-case ratio of an optimal 2-matching to an optimal solution to the subtour LP is \(\frac{10}{3}\).}

We remark that both the optimal subtour LP solution and the optimal 2-matching solution give lower bounds on the cost of the optimal TSP tour. The cost of an optimal 2-matching may be less than the optimal value of the subtour LP, since the 2-matching solution does not need to be connected. However, it is known that there are cases for which the cost of an optimal 2-matching is at least \(\frac{10}{9}\) times the optimal solution to the subtour LP; see Figure 1. Boyd and Carr have shown that the conjecture is true if the solution to the subtour LP has a very special structure: namely, all variables \(x(e) \in \{0, \frac{1}{2}, 1\}\), the cycles formed by the edges \(e\) with \(x(e) = \frac{1}{2}\) all have the same odd size \(k\), and the support is \((k - 1)\)-edge-connected.\footnote{In fact, they show in this case the optimal 2-matching has cost at most \((3k + 1)/(3k)\) times the cost of the optimal solution to the subtour LP.} In the general case, the only bound on this ratio that we know of is the Boyd and Carr bound on the integrality gap of 2-matchings; since the constraints of the subtour LP are a superset of the fractional 2-matching constraints, this implies the ratio is at most \(\frac{4}{3}\).
whether there is an optimal solution that obeys the degree constraints if the comb inequalities are added.

We show that we can simplify the Boyd and Carr result bounding the integrality gap for 2-matchings by \( \frac{4}{3} \); more precisely, he shows that if \( x \) is a feasible solution to \((\text{SUBT}')\), then \( \frac{10}{9} x \) is feasible for the LP obtained by adding comb inequalities to \((\text{SUBT}')\). It is known that adding a subset of the comb inequalities to the degree constraints \((1)\) and bounds \((3)\) gives the 2-matching polytope. This would imply the Boyd-Carr conjecture if it were known that there is an optimal solution that obeys the degree constraints when the comb inequalities are added to \((\text{SUBT}')\); as mentioned above, it can be shown that there is an optimal solution for \((\text{SUBT}')\) that obeys the degree constraints when the edge costs obey the triangle inequality. But we do not know whether there is an optimal solution that obeys the degree constraints if the comb inequalities are added.\(^3\)

The contribution of this paper is to improve our state of knowledge for the subtour LP by proving Conjecture 1.

We start by showing that in some cases the cost of an optimal 2-matching is at most \( \frac{10}{9} \) the cost of a fractional 2-matching; in particular, we show this is true whenever the support of the fractional 2-matching has no cut edge. This is not implied by Conjecture 1 since there exist fractional 2-matching solutions for which the support has no cut edge but that are not feasible solutions to the subtour LP (and this holds even if we restrict ourselves to fractional 2-matching solutions that consist of a single component); see Figure 2 for an example.

As the first step in this proof, we give a simplification of the Boyd and Carr result bounding the integrality gap for 2-matchings by \( \frac{4}{3} \). In the case that the support of an optimal fractional 2-matching has no cut edge, the proof becomes quite simple. The perfect matching polytope plays a crucial role in the proof: we use the matching edges to show us which edges to remove from the solution in addition to showing us which edges to add. We note that this idea was independently developed in the recent work of Mömke and Svensson [17], but also previously appeared in the reduction of the 2-matching polytope to the matching polytope; see, for instance, Schrijver [24, Section 30.7]. We also use a notion from Boyd and Carr [5] of a graphical 2-matching: in a graphical 2-matching, each node has degree either two or four; each edge has zero, one, or two copies; and each component has size at least three. Given the triangle inequality, we can shortcut any graphical 2-matching to a 2-matching of no greater cost.

To obtain our proof of the Boyd-Carr conjecture, we give a polyhedral formulation of the graphical 2-matching problem and use it to prove Conjecture 1. If \( x \) is a feasible solution for the subtour LP, then, roughly speaking, we show that \( \frac{10}{9} x \) is feasible for the graphical 2-matching polytope. Our previous results give us intuition for the precise mapping of variables that we need. Using the graphical 2-matching polytope allows us to overcome the issues with the degree constraints faced in trying to use Goemans’ results (Goemans [14]).

All the results above can be made algorithmic and have polynomial-time algorithms, though we do not explicitly determine running times.

\(^3\)To quote Goemans [14, p. 348]: “One might wonder whether the worst-case improvements remain unchanged when one adds the degree constraints \( x(\delta(i)) = 2 \) for all \( i \in V \) and restricts one’s attention to cost functions satisfying the triangle inequality. We believe so but have been unable to prove it. The result would follow immediately if one could prove that the degree constraints never affect the value of the relaxation when the cost function satisfies the triangle inequality.”
We conclude by posing a new conjecture: namely that the worst-case integrality gap of the subtour LP is achieved for solutions to the subtour LP that are fractional 2-matchings (that is, for instances such that adding the subtour constraints to the degree constraints and the bounds on the variables does not change the objective function value).

In a companion paper, Qian et al. [22] show that the proof of the Boyd-Carr conjecture can be used to help bound the integrality gap of the subtour LP for the 1,2-TSP. They show that the gap is at most $\frac{19}{15} \approx 1.267 < \frac{4}{3}$. They also give a proof that the cost of the optimal 2-matching is at most $\frac{10}{9}$ times the cost of a fractional 2-matching in the case that $c(i, j) \in \{1, 2\}$, which gives an alternative proof of the Boyd-Carr conjecture in this case.

Our paper is structured as follows. We introduce basic terms and notation in §2. In §3, we rederive the Boyd-Carr integrality gap for 2-matchings and show that the gap is at most $\frac{10}{9}$ in the case that the fractional 2-matching has no cut edge. In §4, we give the polytope for graphical 2-matchings and show how to use it to prove the Boyd-Carr conjecture. Finally, we close with our new conjecture in §5.

2. Preliminaries. We will work extensively with fractional 2-matchings; that is, basic optimal solutions $x$ to the LP (SUBT) with only constraints (1) and (3). For convenience we will abbreviate “fractional 2-matching” by F2M and “2-matching” by 2M. F2Ms have the following well-known structure (attributed to Balinski [3]). Each connected component of the support graph (that is, the edges $e$ for which $x(e) > 0$) either is a cycle on at least three nodes with $x(e) = 1$ for all edges $e$ in the cycle or consists of odd-sized cycles with $x(e) = \frac{1}{2}$ for all edges $e$ in the cycle connected by paths of edges $e$ with $x(e) = 1$ for each edge $e$ in the path (see Figure 3 for an example). We call the former components integer components and the latter fractional components. Many of our results focus on transforming an F2M into a 2M, in which all components are integer. For that reason, we will often focus solely on how to transform the fractional components into integer components. We then call the edges of fractional components for which $x(e) = \frac{1}{2}$ cycle edges and the edges for which $x(e) = 1$ path edges. Note that removing a cycle edge can never disconnect a fractional component. If removing a path edge disconnects a fractional component, we call it a cut edge. The associated path of the path edge we will call a cut path since every edge in it will be a cut edge. We will say that a fractional 2-matching is connected if it has a single component.

We will use a concept introduced by Boyd and Carr [5] of a graphical 2-matching (G2M). As stated above, in a graphical 2-matching, each node has degree either two or four; each edge has zero, one, or two copies; and each component has size at least three. Given the triangle inequality, we can shortcut any G2M to a 2M of no greater cost. Our techniques for transforming an F2M to a 2M actually find G2Ms.

Figure 2. Two fractional 2-matchings; the dotted edges have $x$-value $\frac{1}{2}$ and the solid edges have $x$-value 1. All edges in the support graph have cost 1; the cost between any other pair of nodes is equal to the cost of the shortest path in the support graph. The figure on the left is a fractional 2-matching with cut edge, showing that the worst case ratio of the 2-matching cost over the fractional 2-matching cost is at least $\frac{4}{3}$. The figure on the right is a fractional 2-matching for which the support has no cut edge but which is not a feasible solution to the subtour LP since the subtour constraint is violated for the set containing the three rightmost nodes.

Figure 3. Illustration of a fractional 2-matching that has two components, one integer component and one fractional component. The dotted edges have $x$-value $\frac{1}{2}$ and will be called cycle edges, and the solid edges have $x$-value 1 and will be called path edges.
We will often need to find minimum-cost perfect matchings. By a result of Edmonds [11], the perfect matching polytope is defined by the following linear program (M):

\[
\begin{align*}
\min & \sum_{e \in E} c(e)x(e) \\
\text{subject to:} & \sum_{e \in \delta(i)} x(e) = 1, \quad \forall i \in V, \\
& \sum_{e \in \delta(S)} x(e) \geq 1, \quad \forall S \subset V, |S| \text{ odd}, \\
& x(e) \geq 0, \quad \forall e \in E.
\end{align*}
\]

(6) (7) (8)

3. 2-matching integrality gaps. In this section, we bound the cost of a G2M in terms of an F2M via combinatorial methods. We start by giving a proof of a result of Boyd and Carr [5] that there is a G2M of cost at most \( \frac{4}{3} \) the cost of an F2M. Our proof is somewhat simpler than theirs, but more importantly, it introduces the main ideas that we will need to obtain other results. We then show that if the F2M has no cut edges, we can improve the bound from \( \frac{4}{3} \) to \( \frac{10}{7} \). The main idea of this section is that given an F2M, we define a matching problem and compute a perfect matching. The perfect matching tells us how to modify the fractional components by either duplicating or removing edges so that we obtain a G2M. We then relate the cost of the perfect matching found to the F2M by providing a feasible solution to the perfect matching LP (M). We will need the following result of Naddef and Pulleyblank [20]; we give the proof since we will use some of its ideas later on.

**Lemma 1 (Naddef and Pulleyblank [20]).** Let \( G \) be a cubic, 2-edge-connected graph with edge costs \( c(e) \) for all \( e \in E \). Then there exists a perfect matching in \( G \) of cost at most \( \frac{1}{3} \sum_{e \in E} c(e) \).

**Proof.** The main idea is to show that \( x(e) = \frac{1}{3} \) is a feasible solution to the matching polytope (M). The lemma then follows because (M) has integer extreme points. Since \( G \) is cubic, \( |V| \) must be even, and \( \sum_{e \in \delta(i)} x(e) = 1 \). Now consider any \( S \subset V \) with \( |S| \) odd. Because \( G \) is cubic, it must be that \( |\delta(S)| \) is odd, and since \( G \) is 2-edge-connected, \( |\delta(S)| \geq 2 \). Therefore, \( |\delta(S)| \geq 3 \), and \( \sum_{e \in \delta(S)} x(e) \geq 1 \). □

We note that Lemma 1 also holds for cubic, 2-edge-connected multigraphs.

**Theorem 1.** There exists a G2M of cost at most \( \frac{4}{3} \) times the cost of an F2M if the F2M has no cut edge.

**Proof.** As described above, it is sufficient to focus on a single fractional component of the F2M. Let \( G \) be the support graph of this component.

To find the G2M, we find a minimum-cost perfect matching on the (multi)graph \( G' \) we obtain by replacing each path in \( G \) by a single edge, which we will call (at the risk of some confusion) a path edge. We set the cost of this edge to be the cost of the path in \( G \), and we set the cost of a cycle edge in \( G' \) to the negative of the cost of the cycle edge in \( G \). Note that \( G' \) is cubic and 2-edge-connected because the support graph \( G \) of the F2M has no cut edge.

Given a minimum-cost perfect matching in \( G' \), we construct a G2M in \( G \) by first including all paths from \( G \). If a path edge is in the matching in \( G' \), we double the path in \( G \). If a cycle edge is not in the matching in \( G' \), then we include the cycle edge in the G2M in \( G \); otherwise, we omit the cycle edge. See Figure 4 for an illustration.

We first show that this indeed defines a G2M: for each node, the degree is four if the perfect matching contains the path edge incident to the node (since in that case, the two cycle edges on the node cannot be in the perfect matching, and hence both are added to the G2M together with two copies of the path), and it is two otherwise (since one cycle edge is in the perfect matching, and hence only the other cycle edge and one copy of the path are added to the graphical 2-matching). Note that any connected component indeed has at least three nodes: Every connected component contains (the edges corresponding to) at least one path edge; now, either a path edge corresponds to a path of length at least 2, or the path edge corresponds to a single edge. In the latter case, at least two cycle edges incident to the endpoints of the path edge are also contained in the connected component: if the path edge is in the perfect matching, all four cycle edges incident to its endpoints are in the connected component; if the path edge is not in the perfect matching, the component will contain one cycle edge incident to each endpoint of the path edge. Finally, these cycle edges are distinct from the edge that is the path since \( G \) has no doubled edges.

We let \( C \) denote the sum of the costs of the cycle edges and \( P \) the cost of the paths. Note that the cost of the F2M solution is \( \frac{4}{3}C + P \). The cost of the G2M is equal to the cost of all edges in the support graph \( (P + C) \).
plus the cost of the perfect matching. Because $G'$ is cubic and 2-edge-connected, we can invoke Lemma 1 to show that the perfect matching has cost at most a third the cost of the edges in $G'$ or at most $\frac{1}{3}(P - C)$. Hence the cost of the G2M is at most

$$P + C + \frac{1}{3}(P - C) = \frac{4}{3}P + \frac{2}{3}C = \frac{4}{3} \left( P + \frac{1}{2}C \right)$$

or at most $\frac{4}{3}$ the cost of the F2M solution, as claimed. $\square$

The idea of using edges from a perfect matching to decide which edges to include in a matching and which edges to remove has also been used recently by Mömke and Svensson [17].

We now modify the proof of the theorem above so that the result extends to the case in which the F2M has cut edges.

**Theorem 2 (Boyd and Carr [5]).** There exists a G2M of cost at most $\frac{4}{3}$ times the cost of an F2M.

**Proof.** As described above, it is sufficient to focus on a single fractional component of the F2M, and we let $G$ be the support graph of this component.

We once again create a new graph $G'$ from $G$ so that we can later define a matching problem in $G'$. The matching will again show us how to create a G2M in $G$. We extend the previous construction to deal with the case when the support graph has cut paths. We introduce a gadget in $G'$ for each cut path in $G$, which replaces the cut path and its two endpoints. The other paths in $G$ are again replaced by single edges in $G'$ of cost equal to the cost of the path. Each cycle edge in $G$ is also in $G'$ with cost equal to the negative of its cost in $G$.

To introduce the cut-path gadget, we begin by using an idea of Boyd and Carr [5]; namely, we only need to consider three patterns to get an almost feasible graphical 2-matching on the cut path when we allow ourselves to increase the cost by a third compared to the F2M. Suppose the cut path has $l$ edges and $l + 1$ nodes, and let $k = \lceil l/3 \rceil$. We can remove every third edge and double the remaining edges to obtain groups of nodes that are 2-edge-connected, where we get $k$ groups of three nodes that are G2M components plus one group of $l - 3k + 1 \in \{0, 1, 2\}$ nodes. Alternatively, we could remove every third edge, starting from the first edge, and double the remaining edges, in which case the first group has one node; the next $k$ or $k - 1$ groups have three nodes; and the last group again has one, two, or three nodes. The final pattern removes every third edge, starting from the second edge, so that the first group has two nodes; the next $k$ or $k - 1$ groups have three nodes; and again the last group has one, two, or three nodes. Figure 5 illustrates the three patterns for $l = 9$. Note that if $l < 3$, then we can define the three patterns in a similar way. However, since there is no third edge (and no second edge if $l = 1$), the patterns obtained by removing the third (or second if $l = 1$) edge and doubling the remaining edges just result in doubling all the edges in the path. Hence, for $l = 1$, we get one pattern containing
no edges and two patterns containing the doubled path; for \( l = 2 \), we get one pattern containing the doubled path and two patterns containing two copies of, respectively, the first and second edge of the path.

To get a G2M that includes a certain pattern, we will ensure that if a group has size less than three, the G2M will include the two cycle edges incident to the first node (if the group is at the start of the pattern) or last node (if the group is at the end of the pattern).

We remark that for \( l \geq 2 \), there is exactly one pattern that starts with a group of size one, two, and three, and hence two patterns need the G2M to include two cycle edges incident to the first node of the cut path. On the other hand, there is also exactly one pattern that ends with a group of size one, two, and three (the length of the cut path determines which of the three patterns ends with a group of size three: it is the second pattern if \( l \pmod{3} = 0 \), the third pattern if \( l \pmod{3} = 1 \), and the first pattern if \( l \pmod{3} = 2 \)); hence there are also two patterns that need the G2M to include the two cycle edges incident to the last node of the cut path. If \( l = 1 \), there is one pattern that starts and ends with a group of size one; the other two patterns both start and end with a group of size two.

We are now ready to define the cut-path gadget. We replace each endpoint of the cut path in \( G \) by three dummy nodes connected by a path of length two in \( G' \); each of these new edges will have cost zero. We replace the cut path by three pattern edges. Each dummy node is connected to one pattern edge, while ensuring that the middle node is connected to the pattern edge corresponding to the pattern which does not need two cycle edges incident to the endpoint of the cut path (i.e., the pattern for which the group containing the endpoint has size three). We set the cost of a pattern edge to the cost of the edges in the corresponding pattern. See Figure 6 for an illustration of the gadget when \( l = 9 \). If \( l = 1 \), the patterns associated with the illustration in Figure 6 are chosen so that pattern 3 corresponds to not adding the edge of the cut path, and patterns 1 and 2 correspond to doubling the edge.

If we replace each cut path in \( G \) by a cut-path gadget in \( G' \), once again \( G' \) will be a cubic graph. It is not hard to check that \( G' \) is also 2-edge-connected because we have replaced the cut path in \( G \) with three pattern edges crossing the cut in \( G' \).

We argue that there is a minimum-cost perfect matching that uses exactly one pattern edge from each cut-path gadget. Note that we replace only the cut paths in \( G \) by a cut gadget in \( G' \), and once again we have replaced the cut path in \( G \) with three pattern edges crossing the cut in \( G' \).

We argue that there is a minimum-cost perfect matching that uses exactly one pattern edge from each cut-path gadget. Note that we replace only the cut paths in \( G \) by a cut gadget in \( G' \), which means that a perfect matching in \( G' \) contains an odd number of pattern edges in a gadget. If it contains three pattern edges, then we could find a matching of no greater cost by choosing only one pattern edge, namely, the pattern edge that is not incident to the middle node for the either one of its endpoints. Note that we can add two edges of cost 0 that connect the four nodes incident to the other two pattern edges to again have a perfect matching without increasing the cost.

Now we show how to obtain a G2M in \( G \) from the minimum-cost perfect matching in \( G' \). In the G2M we include all edges from \( G \) that are in paths that are not cut paths, the cycle edges in \( G \) that are not chosen by the perfect matching, duplicates of edges in non-cut paths in \( G \) that are chosen by the perfect matching, and the edges in a pattern if the corresponding pattern edge is in the perfect matching.

We argue that this set of edges is a G2M in \( G \). Note that if the perfect matching contains only the pattern edge incident to the middle node, then the two cycle edges that are adjacent to the gadget are also in the matching. Hence the corresponding endpoint in \( G \) of the cut path has no cycle edges incident to it in the G2M, but since the pattern edge is incident to the middle node, the corresponding pattern ensures that the node has degree two.
and is in a connected component of size three. If the perfect matching contains the pattern edge incident to a node other than the middle node, then neither of the two cycle edges that are adjacent to the gadget in \( G' \) are in the perfect matching. Hence the corresponding endpoint of the cut path in \( G \) has both of these cycle edges incident to it in the \( G' \) and zero or two edges from the pattern corresponding to the chosen pattern edge. Hence the node has degree two or four and it is in a connected component of size at least three.

As before, because \( G' \) is cubic and 2-edge-connected, we can apply Lemma 1 to bound the cost of the perfect matching in \( G' \). Let \( P_i \) be the cost of the paths in \( G \) that are not cut paths and \( P_c \) the cost of the cut paths in \( G \) so that the cost of the F2M is \( P_i + P_c \). Note that the cost of the three pattern edges in the gadget corresponding to a cut path sums up to four times the cost of the cut path. Thus the total cost of the edges in \( G' \) is \( P_i + 4P_c - C \). By Lemma 1, the cost of the perfect matching in \( G' \) is at most \( \frac{4}{3}(P_i + 4P_c - C) \), and, as we argued above, the perfect matching in \( G' \) contains at least one pattern edge corresponding to each cut path. The cost of the G2M corresponding to the minimum-cost perfect matching is therefore at most

\[
P_1 + C_1 + \frac{1}{3}(P_i + 4P_c - C) = \frac{4}{3}(P_i + P_c) = \frac{4}{3}(P_i + P_c + \frac{1}{2}C),
\]

as claimed. \( \square \)

We now show how to use the ideas behind the cut-path gadget to obtain a better G2M if no cut paths exist.

**Theorem 3.** If an F2M has no cut edge, then there exists a G2M of cost at most \( \frac{10}{9} \) times the cost of the F2M.

**Proof.** Once again we define a new graph \( G' \) from the support graph \( G \) of a fractional component of the optimal F2M. Each cycle edge in \( G \) is in \( G' \) with cost that is the negative of its cost in \( G \). Each path in \( G \) and its two endpoints are replaced by the cut-path gadget used in the proof of Theorem 2. The costs of the pattern edges in \( G' \) are slightly different than in the previous proof: we subtract the cost of the original path from the cost of the edges in the pattern to obtain the cost of the pattern edge. In other words, the cost of a pattern edge in \( G' \) is obtained by adding once the cost of the edges that appear twice in the pattern and subtracting the cost of the edges that do not appear in the pattern. Note that the sum of the costs of the three pattern edges in \( G' \) is equal to the cost of the original path in \( G \). Also, note that the sum of the costs of any two pattern edges in \( G' \) is nonnegative: an edge on the path contributes its cost either positively to one pattern and negatively to the other or positively to both patterns.

We first argue that there is a minimum-cost perfect matching that chooses either zero or one pattern edge in each cut-path gadget. Suppose the perfect matching contains two pattern edges in a gadget. Note that on both sides of the gadget, these pattern edges must be incident to the middle node; otherwise, some middle node is not matched. Hence the four endpoints of the two pattern edges are connected in \( G' \) by two edges of cost zero (i.e., by two vertical edges in the gadget as drawn in Figure 6). By the observation above, the cost of the two pattern edges is nonnegative, and so we can remove the two pattern edges from the matching and add the two edges of cost zero without increasing the cost of the matching. By the same argument, we can handle the case that the perfect matching contains three pattern edges from a gadget by choosing the pattern edge that is not incident to the middle node on both sides of the gadget and replacing the other two pattern edges in the matching by the cost zero edges that connect their endpoints.

Therefore, we can assume the perfect matching chooses either zero or one pattern edge in a gadget. If it chooses zero pattern edges, then we add the path from \( G \) to the G2M. Otherwise, the pattern corresponding to the chosen pattern edge is added to the G2M. We also add the cycle edges to the G2M corresponding to the cycle edges that are not in the perfect matching.

By almost the same arguments as before, the solution constructed is indeed a G2M. The only case not covered by previous arguments is the case in which zero pattern edges are chosen in \( G' \). Then it must be the case for each side of the pattern that one of the two adjacent cycle edges is chosen in \( G' \) and the other is not, so that one of the two cycle edges is included in the G2M and the other not. Since we include the path from \( G \) in the G2M if no pattern edges are chosen, each endpoint of the path will have degree two.

To argue about the cost of the minimum-cost perfect matching in \( G' \), we create a feasible solution for the matching linear program \( (M) \). To do this, for each pattern edge \( e \), we set \( x(e) = \frac{1}{5} \), and for every other edge \( e' \), we set \( x(e') = \frac{1}{2} \). See Figure 7 for an illustration. We will show this is a feasible solution in a moment. Let \( P \) be the cost of the path edges in the F2M and \( C \) the cost of the cycle edges so that the F2M has cost \( P + \frac{1}{2}C \). Since the sum of the cost of the pattern edges in a gadget is equal to the cost of the path, the cost of this solution for
Figure 7. Illustration of the feasible fractional solution for the matching linear program used in the proof of Theorem 3.

(M) is $\frac{1}{9}P - \frac{4}{9}C$, and there exists a perfect matching of cost at most this much. Thus the cost of the G2M is at most

$$P + C + \left(\frac{1}{9}P - \frac{4}{9}C\right) = \frac{10}{9}P + \frac{5}{9}C = \frac{10}{9} \left(P + \frac{1}{2}C\right),$$

as claimed.

To see that $x$ is a feasible solution for (M), consider any cut such that the number of nodes on each side of the cut is odd. If there exists a cycle from the F2M such that not all nodes in the gadgets for the nodes in the cycle are on the same side of the cut, then there are two edges crossing the cut with value $\frac{4}{9}$. Since $G'$ is cubic, if the cut has odd size, then the total number of edges crossing the cut is odd, and there must be at least one more edge in the cut with value at least $\frac{1}{2}$. Hence the total value on the edges crossing the cut is at least one. For any other cut, since there is no cut path in $G$, there are at least three gadgets crossing the cut in $G'$. Since each gadget contains three pattern edges, the value of the edges crossing the cut is again at least one. \qed

4. A polyhedral proof of the Boyd-Carr conjecture. We will generalize the result in Theorem 3 and show that the ratio between the cost of the optimal 2-matching and the subtour LP is at most $\frac{10}{9}$. In the combinatorial proofs of the previous section, we heavily used that F2Ms have a nice, simple structure, and, unfortunately, this does not hold for the subtour LP solution. We therefore turn to a polyhedral rather than a combinatorial proof. We derive a polyhedral description for graphical 2-matchings, and we then use this description to construct a feasible (fractional) G2M solution from any solution to the subtour LP of cost not more than $\frac{10}{9}$ times the value of the subtour LP. The manner in which the feasible G2M solution is defined based on a solution to (SUBT) is a generalization of the proof of Theorem 3.

We start by giving a polyhedral description of a generalization of 2-matching, where the node set consists of “mandatory nodes” ($V_{\text{man}}$) and “optional nodes” ($V_{\text{opt}}$). The former need to have degree two in the solution, whereas the latter can have degree zero or two. We will refer to this problem as the 2-Matching with Optional Nodes Problem (2MO).

**Theorem 4.** Let $G = (V_{\text{man}} \cup V_{\text{opt}}, E')$ be a 2MO instance. The convex hull of integer 2MO solutions is given by the following polytope:

\[
\sum_{e \in \delta(i)} y(e) = 2, \quad \forall i \in V_{\text{man}}, \tag{9}
\]

\[
\sum_{e \in \delta(i)} y(e) \leq 2, \quad \forall i \in V_{\text{opt}}, \tag{10}
\]

\[
\sum_{e \in \delta(S) \setminus F} y(e) + \sum_{e \in F} (1 - y(e)) \geq 1, \quad \forall S \subseteq V, \quad F \subseteq \delta(S), \quad F \text{ matching, } |F| \text{ odd}, \tag{11}
\]

\[
0 \leq y(e) \leq 1, \quad \forall e \in E. \tag{12}
\]
Aráoz et al. [2] show that the convex hull of disjoint sets of circuits of $G = (V, E)$ is given by constraints (11), constraints (12), and $\sum_{e \in b(i)} y(e) \leq 2, \forall i \in V$. Clearly, if we restrict the polytope by replacing some inequality constraints by equality constraints, we create no new extreme points; hence the extreme points of the polytope in Theorem 4 are integer 2MO solutions. For completeness, we give a direct proof of Theorem 4 by a reduction to a perfect matching problem in the appendix.

We now relate the graphical 2-matching (G2M) problem to the 2MO problem. Recall the definition of a G2M: (i) each node has degree either two or four; (ii) each edge has zero, one, or two copies; and (iii) each component has size at least three. We will (for the moment) relax the second condition so that each edge has at most three copies. We remind the reader that we use $x_{4i1j5}$ to denote the undirected edge between $i$ and $j$.

**Lemma 2.** We can reduce a G2M instance $G = (V, E)$ to a 2MO instance $G' = (V', E')$ as follows: Let $V_m = \{i_m: i \in V\}$, $V_o = \{i_o: i \in V\}$, $V' = V'_m \cup V'_o$, and $E' = \{(i_m, j_m): (i, j) \in E\} \cup \{(i_o, j_o): (i, j) \in E\}$. We add an edge $(i, j)$ to the (relaxed) G2M solution for each edge $(i_m, j_m)$, $(i_o, j_o)$, and $(i_m, j_o)$ that is in the associated 2MO solution.

**Proof.** Note that condition (i) for node $i$ directly follows from the degree constraints for nodes $i_m$ and $i_o$ in the reduction. Relaxed condition (ii) follows because for every edge in the G2M instance, there are three associated edges in the 2MO instance. Finally, since each node $i_m$ has degree two in the 2MO solution, there cannot be a component of size one. Suppose there is a component of size two. Then this must be an isolated doubled or quadrupled edge, say, $(i, j)$, because of the degree constraints. Clearly we can’t have a quadrupled edge since there are at most three copies of edge $(i, j)$ in the 2MO solution. We also can’t have an isolated doubled edge: in order for the edge to be isolated, we would need $(i_m, j_m)$ and $(i_m, j_o)$ to be in the 2MO solution. But then $j_o$ must have degree 2, and its second edge must be $(j_o, k_o)$ for some $k \neq i, j$, since there are no edges $(i_o, j_o)$ or $(j_o, j_o)$ in the 2MO instance. □

If the edges have nonnegative costs, we may assume without loss of generality that each edge appears at most twice in an optimal G2M solution: if any edge appears three times, we can remove two copies of it without affecting the parity of its endpoints, the connected components retain their sizes, and the cost cannot increase.

We will now use a solution $x$ to the subtour LP on $G = (V, E)$ to define a feasible solution $y$ to the 2MO instance $G' = (V', E')$ associated with the graphical 2-matching problem on $G$. If we treat every edge the same and set the $y$-value to $(1 - \alpha)x(i, j)$ on the edge between the mandatory copies of $i$ and $j$, then this implies we have to set the $y$-value to $\alpha x(i, j)$ on the edge from a mandatory to an optional copy. The following lemma states that taking $\alpha = \frac{1}{2}$ indeed defines a fractional solution to the 2MO instance corresponding to the G2M instance $G$ if $x$ is a feasible solution to the subtour LP on $G$.

**Lemma 3.** Given a graph $G = (V, E)$, let $x$ be a feasible solution to the subtour LP for $G$. Then the following solution is a feasible solution to the 2MO instance $G' = (V', E')$ associated with the graphical 2-matching instance given by $G$ for $\alpha = \frac{1}{2}$:

\[
y(i_m, j_m) = (1 - \alpha)x(i, j), \quad y(i_m, j_o) = \alpha x(i, j), \quad y(i_o, j_o) = \alpha x(i, j)
\]

for all $(i, j) \in E$.

Note that the cost of the constructed fractional 2MO solution is exactly $\frac{10}{9}$ times the cost of the solution of the subtour LP. Thus our result follows immediately from the lemma, Theorem 4, and Lemma 2.

**Corollary 1.** There exists a G2M of cost at most $\frac{10}{9}$ times the value of the subtour LP.

**Proof of Lemma 3.** We need to show that $y$ satisfies the Constraints (9)–(12) on $G'$, where $G'$ is defined as in Lemma 2. Constraints (9), (10), and (12) are obviously met, and we only need to show that constraints (11) are met. To this end, fix $S \subseteq V', F \subseteq \delta(S)$, where $F$ is a matching and $|F|$ is odd. For simplicity, for any set of edges $X \subseteq E'$, we define $y(X) = \sum_{e \in X} y(e)$. We need to show that

\[
y(\delta(S)) + |F| - 2y(F) \geq 1.
\]

(13)

We call a node $i \in V$ a split node if $|\{i_o, i_m\} \cap S| = 1$. We now consider three cases.

**Case 1** $F$ contains an edge $(i_o, j_m)$ for some $i_o \in V_o$ and $j_m \in V_m$, and $i$ is a split node. Since $i$ is a split node, exactly one of the edges $(i_o, j_m)$ and $(j'_m, i_m)$ crosses the cut for all $j' \in V$. If $(j'_m, i_m)$ crosses the cut, the contribution to the left-hand side of (13) is $(1 - \alpha)x(i, j')$ if $(j'_m, i_m) \notin F$ or $1 - (1 - \alpha)x(i, j')$ if $(j'_m, i_m) \in F$. Note that since $x(i, j') \leq 1$ and $\alpha \leq \frac{1}{2}$, the contribution is always at least $\alpha x(i, j')$. If $(i_o, j'_m)$ crosses the cut, the contribution to the left-hand side of (13) is either $\alpha x(i, j')$ if $(i_o, j'_m) \notin F$ or $1 - \alpha x(i, j')$ if $(i_o, j'_m) \in F$. Since $F$ is a matching, note that $(i_o, j'_m) \in F$ for only $j' = j$. 

Downloaded from informs.org by [141.212.43.70] on 27 February 2015, at 11:28. For personal use only, all rights reserved.
The contribution to the left-hand side of (13) by any other edge \( e \) is nonnegative since \( F \subseteq \delta(S) \) and \( y(e) \leq 1 \). We thus have that the left-hand side of (13) is at least \( \alpha \sum_{f \in V} x(i, j) + 1 - 2\alpha x(i, j) \geq 1 \), where the last inequality follows since \( \sum_{f \in V} x(i, j) = 2 \) and \( x(i, j) \leq 1 \).

Case 2 \( F \) contains an edge \((i_o, j_m)\) for some \( i_o \in V_o \) and \( j_m \in V_m \), and \( i \) is not a split node. In this case the edge \((j_m, i_m)\) must also cross the cut, and \((j_m, i_m) \notin F \) since \( F \) is a matching. Hence, the left-hand side of (13) is at least \( x(i, j) + 1 - 2\alpha x(i, j) \geq 1 \) for any \( \alpha \leq \frac{1}{4} \).

Case 3 It remains to consider the case when \( F \) only contains edges between mandatory nodes. Since \( F \) is not empty, it must be the case that \( S \) contains a node \( i_o \) for some \( i \in V \), and \( V \backslash S \) contains a node \( i'_o \) for some \( i' \in V \). Therefore, if we let \( R = \{i: i_o \in S\} \), then \( \emptyset \subseteq R \subseteq V \), and hence \( x(\delta(R)) \geq 2 \). Let \( H = \{(i, j) \in \delta(R): (i_m, j_m) \in F\} \). Since for each edge \((i, j) \in \delta(R)\), we know that \((i_m, j_m) \in \delta(S)\), we get that \( y(\delta(S)) \geq (1 - \alpha)x(\delta(R)) \).

Also, since \( F \) only contains edges between mandatory nodes, we have \( y(F) = (1 - \alpha)x(H) \). We can now conclude that the left-hand side of (13) is at least \((1 - \alpha)x(\delta(R)) + |F| - 2(1 - \alpha)x(H) \geq (1 - \alpha)2 + |F| - 2(1 - \alpha)x(H) \). If \( |F| = |H| = 1 \), then (13) holds since \( x(H) \leq 1 \).

The only case remaining is the case when \( F \) contains only edges between mandatory nodes and \( |F| \geq 3 \). We will prove the following claim.

**CLAIM 1.** \( S \subseteq V \) and let \( F \subseteq \delta(S) \cap \{(i, j): i, j \in V\} \) be a matching. Let \( H = \{(i, j) \in F: (i_m, j_m) \in F\} \). Then \( y(\delta(S)) \geq (1 + \alpha)x(H) \).

Using the claim, we have that \( y(\delta(S)) + |F| - 2y(F) \geq (1 + \alpha)x(H) + |F| - 2(1 - \alpha)x(H) = |F| - (1 - 3\alpha)x(H) \). For any \( \alpha \leq \frac{3}{4} \), this is at least \( 3\alpha|F| \) since \( x(e) \leq 1 \) and \( |H| = |F| \). Now, since we assume that \( |F| \geq 3 \), and we chose \( \alpha = \frac{3}{4} \), we have that \( 3\alpha|F| \geq 1 \).

It remains to prove the claim. Recall that \( i \) is a split node if \( \{(i, i_m) \cap S\} = 1 \). For an edge \((i_m, j_m) \in F\), either zero, one, or both of the nodes \( i, j \) are split nodes. We will refer to the edge \((i, j)\) as a Type \( k \) edge if \( k \) of the nodes \( i, j \) are split nodes for \( k \in \{0, 1, 2\} \). See Figure 8 for an illustration of the three possibilities. In the first case, all three edges in the gadget for edge \((i, j)\) are in \( \delta(S) \), and together they contribute \((1 + \alpha)x(i, j)\) to \( y(\delta(S)) \). To complete the proof, we will need to use that for a split node \( i \) and an arbitrary node \( j \in V \), exactly one of the edges \((i, i_m)\) and \((j_m, i_m)\) crosses the cut. More strongly, \( y(\delta(S)) \cap \{(i, i_m), (j, j_m), (i_m, j_m)\} \) is at least \( \alpha x(i, j) \) if \( i \) is a split node, and it is at least \( 2\alpha x(i, j) \) if both \( i \) and \( j \) are split nodes. Hence, we can use \( y(\delta(S)) \) to assign each split node \( i \) an amount \( \alpha \sum_{j} x(i, j) = 2\alpha \).

We now consider how much of \( y(\delta(S)) \) remains unassigned. For an edge \((i, j)\) of Type 0, neither \( i \) nor \( j \) is a split node, and hence \((1 + \alpha)x(i, j)\) remains unassigned. For an edge \((i, j)\) of Type 1, \( y(\delta(S)) \cap \{(i, j), (i_m, j_m), (i, j_m)\} = x(i, j) \), and since only \( i \) is a split node, \((1 - \alpha)x(i, j)\) remains unassigned. For an edge \((i, j)\) of Type 2, \( y(\delta(S)) \cap \{(i, j), (j_m, j_m), (i_m, j_m)\} = (1 - \alpha)x(i, j) \); since both \( i \) and \( j \) are split nodes, \((1 - 3\alpha)x(i, j)\) remains unassigned.

Let \( F_i \) be the number of edges in \( F \) of Type 0, 1, and 2, respectively, and let \( H = \{(i, j) \in F: (i_m, j_m) \in F\} \). Let \( n_i \) be the number of split nodes. Then we have argued that \( y(\delta(S)) \geq (1 + \alpha)x(H) + (1 - \alpha)x(H) + (1 - 3\alpha)x(H) + 2\alpha n_i \). Now, since \( F \subseteq \delta(S) \cap \{(i, j): i, j \in V\} \) is a matching, \( n_i \geq |H| + 2|H| \geq x(H) + 2x(H) \), so \( y(\delta(S)) \geq (1 + \alpha)x(H) \), as claimed. \( \square \)

5. Conjectures and conclusions.

I conjecture that there is no polynomial-time algorithm for the traveling salesman problem. My reasons are the same as for any mathematical conjecture: (1) It is a legitimate mathematical possibility, and (2) I do not know.

—Edmonds [12, p. 234]

We conclude our paper with a conjecture. We do so in the spirit of Jack Edmonds, quoted above; we do not know whether the conjecture is true or not, but we think that even a proof that the conjecture is false would be
interesting. Our conjecture says that when determining the integrality gap for the subtour LP, it suffices to look at instances that have an optimal solution that is an F2M, i.e., an extreme point of the fractional 2-matching polytope.

**Conjecture 2.** Let $\alpha_n$ be the integrality gap of the subtour LP on $n$ nodes. Then there exists an instance that has an optimal subtour LP solution that is an F2M and for which the optimal tour has cost $\alpha_n$ times the subtour LP cost.

Let us call cost functions $c$ such that there exists an optimal subtour LP solution that is an F2M fractional 2-matching costs for the subtour LP. We note that the family of instances that provide a lower bound of $\frac{4}{3}$ on the integrality gap is a family of fractional 2-matching costs.

We could make a similar conjecture for the ratio of the cost of the optimal 2-matching to the subtour LP, but by Corollary 1 and the example in Figure 1, we already know that the conjecture is true. However, its truth does not shed any light on the conjecture above.

Interestingly, we appear to know almost nothing about the consequences of Conjecture 2. Even for this very restricted set of cost functions, we do not know a better upper bound on the integrality gap of the subtour LP other than the bound of $\frac{4}{3}$. It would be very interesting to prove that for such costs the integrality gap is $\frac{4}{3}$.

Boyd and Carr [6] have shown this for fractional 2-matching costs in which all the cycles of the F2M have size 3. Note that the F2M consists of a single component and will not have a cut edge since it is feasible for the subtour LP. Therefore, their result also follows from the technique of Theorem 1 since the resulting graphical 2-matching is Eulerian if all cycles have size three (the graphical 2-matching may not be connected if there are cycles of size five).

In a companion paper, Qian et al. [22] show that if an analogous conjecture for edge costs $c(i, j) \in \{1, 2\}$ is true, then the integrality gap for 1,2-TSP is at most $\frac{9}{7}$. They conjecture that the integrality gap for the 1,2-TSP is at most $\frac{10}{7}$; it is known that it can be no smaller than $\frac{1}{3}$.

It would be very nice to show that if the analogous conjecture is true, then the integrality gap for 1,2-TSP is at most $\frac{10}{7}$. In recent work, Mömke and Svensson [18] show that if an analogous conjecture holds for graph-TSP instances, then the integrality gap for graph-TSP is at most $\frac{4}{3}$.

**Acknowledgments.** An extended abstract of this paper appeared in SODA 2012 Schalekamp et al. [23]. The authors thank Sylvia Boyd for useful and encouraging discussions; they also thank her for giving them pointers on her various results. Gyula Pap made some useful suggestions regarding the polyhedral formulation of graphical 2-matchings. The authors thank Lex Schrijver for pointing them to reference Arãoz et al. [2]. They thank anonymous referees for useful comments. This work was carried out while the second author was on sabbatical at TU Berlin, and the third author was a postdoctoral researcher at the Max-Planck-Institut für Informatik in Saarbrücken. The second author was supported in part by the Berlin Mathematical School, the Alexander von Humboldt Foundation, and NSF [Grant CCF-1115256].

**Appendix. Polyhedral description of 2MO.** We repeat Theorem 4 for the sake of completeness.

**Theorem 4.** Let $G = (V_{\text{man}} \cup V_{\text{opt}}, E)$ be a 2MO instance. The convex hull of integer 2MO solutions is given by the following polytope:

$$\sum_{e \in b(i)} x(e) = 2, \quad \forall i \in V_{\text{man}},$$

$$\sum_{e \in b(i)} x(e) \leq 2, \quad \forall i \in V_{\text{opt}},$$

$$\sum_{e \in b(h(S))} x(e) + \sum_{e \in F} (1 - x(e)) \geq 1, \quad \forall S \subseteq V, \quad F \subseteq \delta(S), \quad F \text{ matching, } |F| \text{ odd,}$$

$$0 \leq x(e) \leq 1, \quad \forall e \in E.$$

**Proof.** The proof that we present here is similar to the proof of the polyhedral description of the 2-matching polytope (Theorem 30.8) in Schrijver [24]. We will first show that any 2MO solution is contained in the polytope and next show that the extreme points of the polytope coincide with the 2MO solutions.

Constraints (14), (15), and (17) obviously hold for a 2MO solution. To show that constraint (16) is satisfied, we consider two cases:

- **Case 1** There is an $\bar{e} \in F$ with $x(\bar{e}) = 0$. This makes the left-hand side of constraint (16) at least 1 since $x(e) \geq 0$ for all $e$. 

• Case 2 $x(e) = 1$ for all $e \in F$. Since $|F|$ is odd, and each node is incident to an even number of edges in an 2MO solution, it follows that there has to be an edge in the solution in $\delta(S)$ that is not in $F$. So the constraint also holds in this case.

The polytope thus contains all 2MO solutions. We will now show that its extreme points coincide with 2MO solutions by reducing 2MO instances to matching instances for which perfect matchings correspond to 2MO solutions. We will show that any feasible point in the 2MO polytope corresponds to a feasible point in the perfect matching polytope. Because any point in the perfect matching polytope can be written as a convex combination of perfect matchings, this implies that any point in the 2MO polytope can be written as a convex combination of 2MO solutions, and therefore all extreme points of the 2MO polytope correspond to 2MO solutions.

Before we consider the reduction to perfect matchings, we will first show that adding constraint (16) for all $F \subseteq E$ of odd cardinality does not change the 2MO polytope. These additional constraints will be convenient when showing that a feasible point in the 2MO polytope is in the perfect matching polytope.

We prove this by induction on $|F|$. Consider $\bar{S}$ and $\bar{F} \subseteq \delta(\bar{S})$ so that $\bar{F}$ is not a matching; i.e., $|\bar{F} \cap \delta(i)| \geq 2$ for some $i \in V$. We consider three cases.

• Case 1 $|\bar{F} \cap \delta(i)| \geq 3$. Then

$$\sum_{e \in \delta(\bar{S}) \setminus \bar{F}} x(e) + \sum_{e \in \bar{F}} (1 - x(e)) \geq \sum_{e \in \bar{F} \cap \delta(i)} (1 - x(e))$$

$$\geq 3 - \sum_{e \in \bar{F} \cap \delta(i)} x(e)$$

$$\geq 3 - \sum_{e \in \delta(i)} x(e) \geq 3 - 2 = 1.$$

• Case 2 $|\bar{F} \cap \delta(i)| = 2$ and $i \in \bar{S}$. Let $F' = \bar{F} \setminus \delta(i)$ and let $S' = \bar{S} \setminus \{i\}$. Then

$$\sum_{e \in \delta(\bar{S}) \setminus \bar{F}} x(e) + \sum_{e \in \bar{F}} (1 - x(e)) \geq \sum_{e \in \delta(\bar{S}) \setminus F'} x(e) - \sum_{e \in \delta(i)} x(e) + \sum_{e \in \delta(i) \cap \bar{F}} x(e)$$

$$+ \sum_{e \in F'} (1 - x(e)) + \sum_{e \in \delta(i) \cap \bar{F}} (1 - x(e))$$

$$= \sum_{e \in \delta(\bar{S}) \setminus F'} x(e) + \sum_{e \in F'} (1 - x(e)) - \sum_{e \in \delta(i)} x(e) + 2.$$

By induction and the degree bound for $i$, this quantity is at least 1.

• Case 3 $|\bar{F} \cap \delta(i)| = 2$ and $i \notin \bar{S}$. Let $F' = \bar{F} \setminus \delta(i)$ as in the previous case but now let $S' = \bar{S} \cup \{i\}$. Then the same string of inequalities as in the previous case holds.

We now use the usual reduction from 2-matchings to matchings (see Theorem 30.7 in Schrijver [24], the notation of which we will also follow): for each node $j$ in the 2MO, there will be two nodes in the matching instance, $i'$ and $i''$. For each edge $e = (i, j)$ in the 2MO instance, there will be two nodes and five edges in the matching instance: nodes $p_{e,i}$ and $p_{e,j}$ and edges $(i', p_{e,i}), (i'', p_{e,i}), (p_{e,i}, p_{e,j}), (j', p_{e,j})$, and $(j'', p_{e,j})$. The only difference between the reduction from 2-matchings to matchings and the reduction from 2MO to matchings is that for optional nodes $i$ we also add an edge between nodes $i'$ and $i''$. An illustration of the reduction is given in Figure A.1, where the part of the matching instance is given that corresponds to an edge between a mandatory node $i$ and an optional node $j$.

Given a (fractional) solution $x$ to a 2MO instance, we define a solution $y$ to the corresponding matching instance as follows:

$$y(i', p_{e,i}) = y(i'', p_{e,i}) = \frac{1}{2} x(e) \quad \text{and} \quad y(p_{e,i}, p_{e,j}) = 1 - x(e) \quad \text{for all } e = (i, j) \in E,$$

and

$$y(i', i'') = 1 - \frac{1}{2} \sum_{e \in \delta(i)} x(e) \quad \text{for all } i \in V_{\text{opt}}.$$
We will now show that this solution is indeed in the perfect matching polytope given by the constraints (6)–(8) of the linear program (M) in §2 (where the variables are here called y instead of x). For nodes \( p_{e,i} \), the degree bound constraints (6) follow directly from the definition of \( y \) (there are three edges incident to \( p_{e,i} \), with y-values \( \frac{1}{2}x(e), \frac{1}{2}x(e), \) and \( 1-x(e) \), which sum to 1). For the other nodes, constraint (6) follows directly from the degree bound constraints (14) or (15) in the 2MO instance and the definition of \( y \). Constraints (8) follow directly from constraints (17).

We will now prove that constraints (7) also hold for all subsets of nodes of odd cardinality in our reduction. Let \( S' \) be such a subset. We consider four cases.

- **Case 1** \( \{i', i''\} \cap S' = \emptyset \) for some \( i \in V \). Note that we have edges \((i', p_{e,i})\) and \((i'', p_{e,i})\) in the reduction both of which have y-value \( \frac{1}{2}x(e) \) and of which exactly one will be in \( \delta(S') \). Furthermore, \((i', i'')\) is in \( \delta(S') \) if \( i \) is in \( V_{opt} \). Therefore \( \sum_{e \in \delta(S')} y(e) \geq \sum_{e \in \delta(i)} \frac{1}{2}x(e) = 1 \) if \( i \in V_{max} \) by the degree bound (14). Similarly \( \sum_{e \in \delta(S')} y(e) \geq \sum_{e \in \delta(i)} \frac{1}{2}x(e) + (1-\frac{1}{2} \sum_{e \in \delta(i)} x(e)) = 1 \) if \( i \in V_{opt} \) by the degree bound (15).

- **Case 2** For some \( e = (i, j) \in E, p_{e,i} \in S', p_{e,j} \notin S' \), and \( \{i', i''\} \cap S' = \emptyset \). Let \( p = p_{e,i} \). Then \( \sum_{e \in \delta(S')} y(e) \geq y(p, i') + y(p, i'') + y(p, p_{e,j}) = \frac{1}{2}x(e) + \frac{1}{2}x(e) + 1 - x(e) = 1 \).

- **Case 3** For some \( e = (i, j) \in E, p_{e,i} \in S', p_{e,j} \notin S' \), and \( \{j', j''\} \subseteq S' \). Let \( p = p_{e,j} \). Then \( \sum_{e \in \delta(S')} y(e) \geq y(p, j') + y(p, j'') + y(p, p_{e,i}) = \frac{1}{2}x(e) + \frac{1}{2}x(e) + 1 - x(e) = 1 \).

- **Case 4** We may now assume that \( S' \) is such that \( \{i', i''\} \cap S' \) is even for all \( i \in V \) and that \( \{i', i''\} \subseteq S' \) and \( \{j', j''\} \subseteq S' \) because otherwise we are in one of the previous cases. Define \( \tilde{S} = \{i \in V: i \in S' \text{ and } i'' \notin S' \} \) and \( \tilde{F} = \{e = (i, j) \in E: p_{e,i} \in S' \text{ and } p_{e,j} \notin S' \} \). Note that the previous argument implies \( \tilde{F} \subseteq \delta(\tilde{S}) \).

Consider \( e = (i, j) \in \delta(\tilde{S}) \) in the 2MO instance and assume without loss of generality that \( i \in \tilde{S} \). By the definition of \( \tilde{S} \), this means \( \{i', i''\} \cap S' = \emptyset \). We consider \( e \in \tilde{F} \) and \( e \notin \tilde{F} \) separately. First of all, assume \( e \in \tilde{F} \).

Since we are not in the previous cases, this means that \( p_{e,i} \in S' \text{ and } p_{e,j} \notin S' \). So for each such \( e \) in the 2MO instance, we have \( (p_{e,i}, p_{e,j}) \in \delta(S') \) in the matching instance, with a y-value of \( 1-x(e) \). Second, assume \( e \notin \tilde{F} \).

By the definition of \( \tilde{F} \), we know that either \( p_{e,i} \) and \( p_{e,j} \) are both in \( S' \) or both not in \( S' \). So for each such \( e \) in the 2MO instance, we have either \( \{(i', p_{e,i}), (i'', p_{e,i})\} \subseteq \delta(S') \) or \( \{(j', p_{e,j}), (j'', p_{e,j})\} \subseteq \delta(S') \) in the matching instance, each of which carries a total y-value of \( x(e) \).

We thus get \( \sum_{e \in \delta(S')} y(e) \geq \sum_{e \in \delta(S'), \tilde{F}} x(e) + \sum_{e \in \tilde{F}} (1-x(e)) \). We then note that the parity of \( |\tilde{F}| \) is equal to the parity of the number of nodes of the type \( p_{e,j} \) in \( S' \), which implies that the parity of \( |\tilde{F}| \) and \( |S'| \) are the same because the other nodes in \( S' \) appear in pairs. Thus since \( |S'| \) is odd, \( |\tilde{F}| \) is odd, and we have \( \sum_{e \in \delta(S')} y(e) \geq \sum_{e \in \delta(S'), \tilde{F}} x(e) + \sum_{e \in \tilde{F}} (1-x(e)) \geq 1 \) by the feasibility of \( x \) for constraints (16).

We conclude the proof by noting that a perfect matching in the constructed instance corresponds to the 2MO solution consisting of all edges \( e = (i, j) \) for which \( (p_{e,i}, p_{e,j}) \) is not in the perfect matching solution. □

### References


On the integrality gap of the subtour LP for the 1,2-TSP

Jiawei Qian · Frans Schalekamp · David P. Williamson · Anke van Zuylen

Abstract In this paper, we study the integrality gap of the subtour LP relaxation for the traveling salesman problem in the special case when all edge costs are either 1 or 2. For the general case of symmetric costs that obey triangle inequality, a famous conjecture is that the integrality gap is 4/3. Little progress towards resolving this conjecture has been made in 30 years. We conjecture that when all edge costs \( c_{ij} \in \{1, 2\} \), the integrality gap is 10/9. We show that this conjecture is true when the optimal subtour LP solution has a certain structure. Under a weaker assumption, which is an analog of a recent conjecture by Schalekamp et al., we show that the integrality gap is at most 7/6. When we do not make any assumptions on the structure of the optimal subtour LP
solution, we can show that integrality gap is at most $5/4$; this is the first bound on the integrality gap of the subtour LP strictly less than $4/3$ known for an interesting special case of the TSP. We show computationally that the integrality gap is at most $10/9$ for all instances with at most 12 cities.

**Keywords**  Traveling salesman problem · Subtour elimination · Linear programming · Integrality gap

**Mathematics Subject Classification**  90C05 · 90C27 · 05C70

1 Introduction

The traveling salesman problem (TSP) is one of the most well studied problems in combinatorial optimization. Given a set of cities $\{1, 2, \ldots, n\}$, and distances $c(i, j)$ for traveling from city $i$ to $j$, the goal is to find a tour of minimum length that visits each city exactly once. An important special case of the TSP is the case when the distance forms a metric, i.e., $c(i, j) \leq c(i, k) + c(k, j)$ for all $i$, $j$, $k$, and all distances are symmetric, i.e., $c(i, j) = c(j, i)$ for all $i$, $j$. The symmetric TSP is known to be NP-hard, even if $c(i, j) \in \{1, 2\}$ for all $i$, $j$ [18]; note that such instances trivially obey the triangle inequality. Such instances are also known to be APX-hard; that is, there is no $\alpha$-approximation algorithm for the problem for some $\alpha > 1$ unless $P = NP$.

The metric TSP can be approximated to within a factor of $3/2$ using an algorithm by Christofides [7] from 1976. The algorithm combines a minimum spanning tree with a matching on the odd-degree nodes to get an Eulerian graph that can be shortcut to a tour; the analysis shows that the minimum spanning tree and the matching cost no more than the optimal tour and half the optimal tour respectively. Better results are known for several special cases, but, surprisingly, no progress has been made on approximating the general symmetric TSP in more than 30 years. A natural direction for trying to obtain better approximation algorithms is to use linear programming. The following linear programming relaxation of the traveling salesman problem was used by Dantzig et al. [8]. For simplicity of notation, we let $G = (V, E)$ be a complete undirected graph on $n$ nodes. In the LP relaxation, we have a variable $x(e)$ for all $e = (i, j)$ that denotes whether we travel directly between cities $i$ and $j$ on our tour. Let $c(e) = c(i, j)$, and let $\delta(S)$ denote the set of all edges with exactly one endpoint in $S \subseteq V$. Then the relaxation is

$$
\text{Min } \sum_{e \in E} c(e)x(e) \\
(\text{SUBT}) \quad \text{subject to: } \sum_{e \in \delta(i)} x(e) = 2, \quad \forall i \in V, \quad (1)
$$

$$
\sum_{e \in \delta(S)} x(e) \geq 2, \quad \forall S \subseteq V, \quad 3 \leq |S| \leq |V| - 3, \quad (2)
$$

$$
0 \leq x(e) \leq 1, \quad \forall e \in E. \quad (3)
$$

\copyright Springer
On the integrality gap of the subtour LP for the 1,2-TSP

The first set of constraints (1) are called the degree constraints. The second set of constraints (2) are sometimes called subtour elimination constraints or sometimes just subtour constraints, since they prevent solutions in which there is a subtour of just the nodes in S. As a result, the linear program is sometimes called the subtour LP. It has been shown by Wolsey [24] (and later Shmoys and Williamson [22]) that Christofides’ algorithm finds a tour of length at most \( \frac{3}{2} \) times the optimal value of the subtour LP; these proofs show that the minimum spanning tree and the matching on odd-degree nodes can be bounded above by the value of the subtour LP, and half the value of the subtour LP, respectively. This implies that the integrality gap, the worst case ratio of the length of an optimal tour divided by the optimal value of the LP, is at most \( \frac{3}{2} \). However, no examples are known that show that the integrality gap can be as large as \( \frac{3}{2} \); in fact, no examples are known for which the integrality gap is greater than \( \frac{4}{3} \). A well known conjecture states that the integrality gap is indeed \( \frac{4}{3} \); see (for example) Goemans [10].

Recently, progress has been made in several directions, both in improving the best approximation guarantee and in determining the exact integrality gap of the subtour LP for certain special cases of the symmetric TSP. In the graph-TSP, the costs \( c(i, j) \) are equal to the shortest path distance in an underlying unweighted graph. If the graph is cubic and 3-connected, Gamarnik et al. [9] show an approximation algorithm with guarantee slightly better than \( \frac{3}{2} \). Oveis Gharan et al. [17] show that the graph-TSP can be approximated to within \( \frac{3}{2} - \epsilon \) for a small constant \( \epsilon > 0 \) for all graphs. Boyd et al. [6], and Aggarwal et al. [1] independently give a \( \frac{4}{3} \)-approximation algorithm if the underlying graph is cubic. Mömke and Svensson [15] improve these results by giving a 1.461-approximation for the graph-TSP and an \( \frac{4}{3} \)-approximation algorithm if the underlying graph is subcubic. Mucha [16] improves the analysis of the Mömke–Svensson algorithm to a \( \frac{13}{9} \)-approximation algorithm, and Sebő and Vygen [21] combine the ideas of Mömke and Svensson [15] with an algorithm based on a carefully chosen ear decomposition of the graph to obtain a \( \frac{7}{5} \)-approximation algorithm. All of these \( \alpha \)-approximation algorithms imply a corresponding upper bound of \( \alpha \) on the integrality gap for the corresponding graph-TSP instances.

In Schalekamp et al. [20], three of the authors of this paper resolve a related conjecture. A 2-matching of a graph is a set of edges such that no edge appears twice and each node has degree two, i.e., it is an integer solution to the LP (\( \text{SUBT} \)) with only constraints (1) and (3). Note that a minimum-cost 2-matching thus provides a lower bound on the length of the optimal TSP tour. A minimum-cost 2-matching can be found in polynomial time using a reduction to a certain minimum-cost matching problem. Boyd and Carr [5] conjecture that the worst case ratio of the cost of a minimum-cost 2-matching and the optimal value of the subtour LP is at most \( \frac{10}{9} \). This conjecture was proved to be true by Schalekamp et al. and examples are known that show this result is tight.

Unlike the techniques used to obtain better results for the graph-TSP, the techniques of Schalekamp et al. work on general weighted instances that are symmetric and obey the triangle inequality. However, their results only apply to 2-matchings and it is not clear how to enforce global connectivity on the solution obtained by their method. A potential direction for progress on resolving the integrality gap for the subtour LP is a conjecture by Schalekamp et al. that the worst-case integrality gap is attained for instances for which the optimal subtour LP solution is a basic
solution to the linear program obtained by dropping the subtour elimination constraints.

In this paper, we turn our attention to the 1,2-TSP, where $c(i, j) \in \{1, 2\}$ for all $i, j$. Note that bounding the cost of enforcing connectivity is relatively easy in this class of problems, since we may connect any two components for an increase in cost of at most 2. Papadimitriou and Yannakakis [18] show how to approximate 1,2-TSP within a factor of $\frac{11}{9}$ by computing a minimum-cost 2-matching and merging its cycles into a tour. In addition, they show a ratio of $\frac{7}{6}$ if they start with a minimum-cost 2-matching that has no cycles of length 3. Bläser and Ram [4] improve this ratio and the best known approximation factor of $\frac{8}{7}$ is given by Berman and Karpinski [3].

We do not know a tight bound on the integrality gap of the subtour LP even in the case of the 1,2-TSP. As an upper bound, we appear to know only that the gap is at most $\frac{3}{2}$ via Wolsey’s result. There is an easy nine city example showing that the gap must be at least $\frac{10}{9}$; see Fig. 1. This example has been extended to a class of instances on $9k$ nodes for any positive integer $k$ by Williamson [23]. The contribution of this paper is to begin a study of the integrality gap of the 1,2-TSP, and to improve our state of knowledge for the subtour LP in this case. We prove an upper bound on the integrality gap for the subtour LP of $\frac{5}{4}$, which is the first bound on the integrality gap with value less than $\frac{3}{2}$ for a natural class of TSP instances. Under an analog of a conjecture of Schalekamp et al. [20], we show that the integrality gap is at most $\frac{7}{8}$, and with an additional assumption on the structure of the solution, we can improve this bound to $\frac{10}{9}$. We describe these results in more detail below.

All the known approximation algorithms since the initial work of Papadimitriou and Yannakakis [18] on the problem start by computing a minimum-cost 2-matching. However, the example of Fig. 1 shows that an optimal 2-matching can be as much as $\frac{10}{9}$ times the value of the subtour LP for the 1,2-TSP, so we cannot directly replace the bound on the optimal solution in these approximation algorithms with the subtour LP in the same way that Wolsey did with Christofides’ algorithm in the general case. Using the result of Schalekamp et al. [20] and a new lemma that relates part of the analysis of Papadimitriou and Yannakakis [18] to the subtour LP bound, we obtain a preliminary upper bound on the integrality gap of the subtour LP for the 1,2-TSP of $\frac{7}{9} \cdot \frac{10}{9} + \frac{4}{9} = \frac{106}{81} \approx 1.3086$.

To improve this upper bound to $\frac{5}{4}$, we first show stronger results in some cases. A fractional 2-matching is a basic optimal solution to the LP (SUBT) with only constraints.
(1) and (3). Schalekamp et al. [20] have conjectured that the worst-case integrality gap for the subtour LP is obtained when the optimal solution to the subtour LP is an extreme point of the fractional 2-matching polytope. We show that if this is the case for 1,2-TSP then we can find a tour of cost at most $\frac{2}{5}$ the cost of the fractional 2-matching, implying that the integrality gap is at most $\frac{7}{8}$ in these cases. Next, we show that if this optimal solution to the fractional 2-matching problem has a certain structure, then we can find a tour of cost at most $\frac{10}{9}$ times the cost of the fractional 2-matching, implying an upper bound on the integrality gap of $\frac{10}{9}$ for these cases. Figure 1 shows that this result is tight.

We then use the previous arguments to show that one can construct a tour of cost at most $\frac{5}{4}$ times the subtour LP value. To do this, we prove that we can assume without loss of generality that the optimal value of the subtour LP is less than $n + 1$, where $n$ denotes the number of nodes. Combined with a more careful analysis based on the results obtained before, we obtain our main result. The results above all lead to polynomial-time algorithms, though we do not state the exact running times.

Finally, we perform computational experiments to show that the integrality gap is at most $\frac{10}{9}$ for $n \leq 12$. We conjecture that the integrality gap is in fact exactly $\frac{10}{9}$.

We note that the upper bound on the integrality gap for general 1,2-TSP instances presented in this paper is stronger than the bound that appeared in a preliminary version of this paper [19] of $\frac{19}{15}$. In the time between publication of the preliminary version and the current revision, Mnich and Mömke [14] obtained an upper bound of $\frac{5}{4}$ on the integrality gap for 1,2-TSP instances that have the additional property of being “fractionally Hamiltonian”, which means that the optimal objective value of the subtour LP is equal to the number of nodes in the instance. In this version, using the same techniques as in the preliminary version, we show an unconditional upper bound on the integrality gap of $\frac{5}{4}$, and a bound of $\frac{26}{21}$ for fractionally Hamiltonian instances.

The remainder of this paper is structured as follows. Section 2 contains preliminaries and a first bound on the integrality gap for the 1,2-TSP. We show how to obtain stronger bounds if the optimal subtour LP solution is a fractional 2-matching in Sect. 3. In Sect. 4, we combine the arguments from the previous sections and show that the integrality gap without any assumptions on the structure of the subtour LP solution is at most $\frac{5}{4}$. We describe our computational experiments in Sect. 5. Finally, we close with a conjecture on the integrality gap of the subtour LP for the 1,2-TSP in Sect. 6.

2 Preliminaries and a first bound on the integrality gap

We will work extensively with 2-matchings and fractional 2-matchings; that is, extreme points $x$ of the LP (SUBT) with only constraints (1) and (3), where in the first case the solutions are required to be integer. For convenience we will abbreviate “fractional 2-matching” by F2M and “2-matching” by 2M. The basic solutions of the F2M polytope have the following well-known structure (attributed to Balinski [2]). Each connected component of the support graph (that is, the edges $e$ for which $x(e) > 0$) is either a cycle on at least three nodes with $x(e) = 1$ for all edges $e$ in the cycle, or consists of odd-sized cycles with $x(e) = \frac{1}{2}$ for all edges $e$ in the cycle connected by paths of edges $e$ with $x(e) = 1$ for each edge $e$ in the path (the center figure in Fig. 1 is an example). We call the former components integer components and the latter frac-
tional components. In a fractional component, we call a path of edges $e$ with $x(e) = 1$ a 1-path. The edges $e$ with $x(e) = \frac{1}{2}$ in cycles are called cycle edges. An F2M with a single component is called connected, and we call a component 2-connected if, in the graph induced by the component, the sum of the $x$-values on the edges crossing any cut is at least 2. We let $n$ denote the number of nodes in an instance.

As mentioned in the introduction, Schalekamp et al. [20] have shown the following.

**Theorem 1** (Schalekamp et al. [20]) If edge costs obey the triangle inequality, then the cost of an optimal 2-matching is at most $\frac{10}{9}$ times the value of the subtour LP. 

It is not hard to show that this immediately implies an upper bound of $\frac{13}{9}$ on the integrality gap of the subtour LP for the 1.2-TSP: we can just compute a minimum cost 2-matching at a cost of $\frac{10}{9}$ the value of the subtour LP, remove the most expensive edge from each cycle, which gives a collection of node-disjoint paths, and add edges of cost 2 to combine these paths into a tour. Each cycle has at least three edges; at worst, we remove an edge of cost 1 from each cycle and then need an edge of cost 2 to patch the paths into a tour. Thus the overall cost increases by at most $\frac{1}{3}n$, giving a tour, the cost of which can be bounded by $\frac{10}{9} + \frac{1}{3} = \frac{13}{9}$ times the value of the subtour LP, because $n$ is a lower bound on the value of the subtour LP.

The algorithm of Papadimitriou and Yannakakis [18] improves on this idea, by cleverly merging the cycles of the optimal 2M solution. We summarize the properties of their algorithm that we will use. First, observe that we can assume without loss of generality that the optimal 2M solution consists of a number of cycles with only edges of cost 1 (“pure” cycles) and at most one cycle which has one or more edges of cost 2 (the “non-pure” cycle), by deleting the edges of cost 2 and combining the resulting disjoint paths into a single cycle. Moreover, if $i$ is a node in the non-pure cycle which is incident on an edge of cost 2 in the cycle, then there can be no edge of cost 1 connecting $i$ to a node in a pure cycle (since otherwise, we can merge the non-pure cycle with a pure cycle without increasing the cost).

The Papadimitriou–Yannakakis algorithm solves the following bipartite matching problem: On one side we have a node for every pure cycle, and on the other side, we have a node for every node in the instance. There is an edge from pure cycle $C$ to node $i$, if $i \not\in C$ and there is an edge of cost 1 from $i$ to some node in $C$. Let $r$ be the number of pure cycles that are unmatched in a maximum cardinality bipartite matching. Papadimitriou and Yannakakis show how to “patch together” the matched cycles. We refer the reader to their original paper [18] for more details. The resulting cycles are then combined into a tour of cost at most

$$\frac{7}{9} \text{OPT}(2M) + \frac{4}{9}n + \frac{1}{3}r,$$

where $\text{OPT}(2M)$ is the cost of an optimal 2M solution.\(^1\)

\(^1\) In [18], $\text{OPT}(2M)$ is expressed as $n + k$, where $k$ is the number of edges of cost 2 in the optimal 2M solution. The number of unmatched pure cycles is denoted by $r_2$. The bound given by [18] is $n + k + \frac{7}{9}(n - n_2 - k) + r_2$, where $n_2$ is a quantity that is lower bounded by $3r_2$. Therefore, the bound in [18] can be upper bounded by $\frac{7}{9}(n + k) + \frac{7}{9}n + \frac{1}{3}r_2$. 

\(\odot\) Springer
We now show how to convert this bound into a bound in terms of the optimal value to $\text{SUBT}$.

**Lemma 1** Let $r$ be the number of pure cycles that are unmatched in a maximum cardinality bipartite matching instance defined by Papadimitriou and Yannakakis. Then

$$\text{OPT}(\text{SUBT}) \geq n + r.$$  

**Proof** We note that for a bipartite matching instance, the size of the minimum cardinality vertex cover is equal to the size of the maximum matching. We use this fact to construct a feasible dual solution to the subtour LP that has value $n + r$. Let $\mathcal{E}_M, V_M$ be the pure cycles and nodes (in the original graph), for which the corresponding nodes in the bipartite matching instance are in the minimum cardinality vertex cover. The dual of the subtour LP $(\text{SUBT})$ is

$$\text{Max} \ 2 \sum_{S \subseteq V} y(S) + 2 \sum_{i \in V} y(i) - \sum_{e \in E} z(e)$$

$$(D) \text{ subject to:} \quad \sum_{S \subseteq V : e \in \delta(S)} y(S) + y(i) + y(j) - z(e) \leq c(e), \quad \forall e = (i, j),$$

$$y(S) \geq 0, \quad \forall S \subset V, \ 3 \leq |S| \leq n - 3,$$

$$z(e) \geq 0, \quad \forall e \in E.$$  

We set $z(e) = 0$ for each $e \in E$, and we set $y(i) = \frac{1}{2}$ for each $i \in V \setminus V_M$. For a pure cycle on a set of nodes $C$, we set $y(C) = \frac{1}{2}$, if the cycle is not in $\mathcal{E}_M$. The dual objective for this solution is exactly $n + r$: its value is $n$ plus the number of pure cycles minus the size of the vertex cover, or $n$ plus the number of pure cycles minus the size of the matching, since the vertex cover has the same size as the matching. Thus it is the same as $n$ plus the number of pure cycles not in the matching, or $n + r$.

It remains to show that the dual constructed is feasible. Define the load on an edge $e = (i, j)$ of solution $(y, z)$ to be $\sum_{S \subseteq V : e \in \delta(S)} y(S) + y(i) + y(j) - z(e)$. For any edge $e = (i, j)$ of cost 1 inside a cycle of the 2M, the load on the edge is at most 1, since the only potentially non-zero dual variables loading the edge are the dual variables $y(i)$ and $y(j)$. For an edge $(i, j)$ where $i \in C$ and $j \notin C$, the load is $y(i) + y(j) + y(C) + y(C') \leq 2$. Suppose $(i, j)$ has cost 1, and the cycles $C$ and $C'$ are both pure cycles. Then the edge occurs twice in the bipartite matching instance (namely, once going from $i$ to $C$ and once going from $j$ to $C'$) and hence the dual of at least two of the four objects $i, j, C$ and $C'$ has been reduced to 0. The total load on edge $(i, j)$ is thus at most 1. Now, suppose $C'$ is the non-pure cycle, then $y_{C'} = 0$, since we only increased the dual variables for the pure cycles. Moreover, at least one endpoint of the $(j, C)$ edge in the bipartite matching instance must be in the vertex cover, so the load on edge $(i, j)$ is again at most 1. \qed

Note that, combined with (4) and Theorem 1, Lemma 1 implies that the cost of the tour is at most $\frac{7}{2} \cdot \frac{10}{7} \text{OPT}(\text{SUBT}) + \frac{4}{7} \text{OPT}(\text{SUBT}) = \frac{106}{7} \text{OPT}(\text{SUBT})$. This bound obtained on the integrality gap seems rather weak, as the best known lower bound.
on the integrality gap is only \( \frac{10}{9} \). Schalekamp et al. [20] have conjectured that the integrality gap (or worst-case ratio) of the subtour LP occurs when the solution to the subtour LP is a fractional 2-matching.

**Conjecture 1** (Schalekamp et al. [20]) Let \( \alpha_n \) be the integrality gap of the subtour LP on \( n \) vertices. Then there exists an instance which has an optimal subtour LP solution that is an F2M and for which the optimal tour has cost at least \( \alpha_n \) times the subtour LP cost.

In the next section, we show that we can obtain better bounds on the integrality gap of the subtour LP in the case that the optimal solution is a fractional 2-matching. In Sect. 4, we then show how to combine Lemma 1 with the bounds in the next section to obtain an upper bound of \( \frac{5}{4} \) on the integrality gap.

### 3 Better bounds if the optimal solution is an F2M

If the optimal solution to the subtour LP is a fractional 2-matching, then a natural approach to obtaining a good tour is to start with the edges with \( x \)-value 1, and add as many edges of cost 1 and \( x \)-value \( \frac{1}{2} \) as possible, without creating a cycle on a subset of the nodes. More precisely, we will propose an algorithm that creates an acyclic spanning subgraph \((V, T)\) where all nodes have degree one or two. We will refer to an acyclic spanning subgraph in which all nodes have degree one or two as a partial tour. A partial tour can be extended to a tour by adding \( d/2 \) edges of cost 2, where \( d \) is the number of degree 1 nodes. The cost of the tour is \( c(T) + d \), where \( c(T) = \sum_{e \in T} c(e) \).

We will use the following lemma.

**Lemma 2** Let \( G = (V, T) \) be a partial tour. Let \( A \) be a set of edges not in \( T \) that form an odd cycle or a path on \( V' \subset V \), where the nodes in \( V' \) have degree one in \( T \). We can find \( A' \subset A \) such that \((V, T \cup A') \) is a partial tour, and

- \( |A'| \geq \frac{1}{3}|A| \) if \( A \) is a cycle,
- \( |A'| \geq \frac{1}{3}(|A| - 1) \) if \( A \) is a path,

We postpone the proof of the lemma and first prove the implication for the bound on the integrality gap if the optimal subtour LP solution is a fractional 2-matching.

**Theorem 2** There exists a tour of cost at most \( \frac{7}{6} \) times the cost of a connected F2M solution if \( c(i, j) \in \{1, 2\} \) for all \( i, j \).

**Proof** Let \( P = \{e \in E : x(e) = 1\} \) (the edges in the 1-paths of \( x \)). We will start the algorithm with \( T = P \). Let \( R = \{e \in E : x(e) = \frac{1}{2} \text{ and } c(e) = 1\} \) (the edges of cost 1 in the cycles of \( x \)). Note that the connected components of the graph \((V, R)\) consist of paths and odd cycles. The main idea is that we consider these components one by one, and use Lemma 2 to show that we can add a large number of the edges of each path and cycle, where we keep as an invariant that \( T \) is a partial tour. Note that by Lemma 2, the number of edges added from each path or cycle \( A \) is at least \(|A|/3\), except for the paths for which \(|A| \equiv 1 \pmod{3}\). Let \( P_1 \) be this set of paths. We would like to claim that we add a third of the edges on average from each component,
and we therefore preprocess the paths in $\mathcal{P}_1$, where we add one edge (either the first or last edge from each path in $\mathcal{P}_1$) to $T$ if this is possible without creating a cycle in $T$, and if so, we remove this edge and its neighboring edge in $R$ (if any) from $R$. After the preprocessing, we use Lemma 2 to process each of the components in $(V, R)$.

We call a path $A$ in $\mathcal{P}_1$ “eared” if the 1-paths that are incident on the first and last node of the path are such that they go between two neighboring nodes of $A$. Without loss of generality we will assume for each path in $\mathcal{P}_1$ that is not eared, that the first edge on the path does not form a cycle with a 1-path. It is not hard to see that we can add the first edge from at least half of the paths in $\mathcal{P}_1$ that are not eared: Suppose we simply add the first edge from each path in $\mathcal{P}_1$ that is not eared to $T$, regardless of whether it makes a cycle in $T$ or not. Then, each cycle in $T$ contains at least two edges that came from a path in $\mathcal{P}_1$, and hence, removing one such edge per cycle leaves a partial tour $T$ that contains the first edge from at least half of the paths that are not eared in $\mathcal{P}_1$.

After preprocessing the paths in $\mathcal{P}_1$, we iterate through the connected components in $(V, R)$ and add edges to $T$ while maintaining that $T$ is a partial tour. By Lemma 2, the number of edges added from each path or cycle $A$ is at least $|A|/3$, except for the paths in $\mathcal{P}_1$. We now consider two cases for the paths in $\mathcal{P}_1$, depending on whether we added an edge from the path to $T$ in the preprocessing step or not. Note that for a path $A$ in $\mathcal{P}_1$ for which we added an edge to $T$ in the preprocessing step, $R$ contains a path of $|A| - 2$ edges after the preprocessing step, and by Lemma 2, we add at least $(|A| - 2 - 1)/3$ of these to $T$. Together with the edge added in the preprocessing step, we thus add at least $1 + (|A| - 2 - 1)/3 = |A|/3$ edges. For a path in $\mathcal{P}_1$ for which we did not add an edge to $T$ in the preprocessing stage, we add at least $(|A| - 1)/3$ edges. Now, recall that a path $A$ in $\mathcal{P}_1$ has $|A| \equiv 1 \pmod{3}$, and that the number of edges added is an integer, so in the first case, the number of edges added is at least $|A|/3 + 2/3$ and in the second case it is $|A|/3 - 1/3$. Let $z$ be the number of eared paths in $\mathcal{P}_1$. Then, the number of paths in $\mathcal{P}_1$ that are in the second case is at most $z$ plus the number of paths in $\mathcal{P}_1$ that fall in the first case. Hence, the total number of edges from $R$ that were added to $T$ can be lower bounded by $|R|/3 - z/3$. We now give an upper bound on the number of nodes of degree one in $T$.

Let $k$ be the number of cycle nodes in $x$, i.e. $k = \#\{i \in V : x(i, j) = 1/2 \text{ for some } j \in V\}$, and let $p$ be the number of cycle edges of cost 2 in $x$, i.e. $p = \#\{e \in E : x(e) = 1/2 \text{ and } c(e) = 2\}$. Note that $(V, R)$ contains $p$ paths (which may have zero edges) on the cycle nodes, and hence $p \geq z$. Initially, when $T$ contains only the edges in the 1-paths, all $k$ nodes have degree one, and there are $k - p$ edges in $R$. We argued that we added at least $|R|/3 - z/3 = 1/2k - 1/3p - 1/3z$ edges to $T$. Each edge reduces the number of nodes of degree one by two, and hence, the number of nodes of degree one at the end of the algorithm is at most $k - 2(1/2k - 1/3p - 1/3z) = 1/3k + 1/3p + 1/3z$. Recall that $c(P)$ denotes the cost of the 1-paths, and the total cost of $T$ at the end of the algorithm is at most $c(P) + 1/3k - 1/3p - 1/3z$. Since at most $1/3k + 1/3p + 1/3z$ nodes have degree one in $T$, we can extend $T$ into a tour of cost at most $c(P) + 1/3k + 1/3p + 1/3z$.

The cost of the solution $x$ can be expressed as $c(P) + 1/2k + 1/2p$. Note that each 1-path connects two cycle nodes, hence $c(P) \geq 1/2k$. Moreover, an eared path $A$ is incident to one (if $|A| = 1$) or two (if $|A| > 1$) 1-paths of length two, since the support graph of $x$ is simple. Therefore we can lower bound $c(P)$ by $1/2k + z$. Therefore,
\[ \frac{7}{6} (c(P) + \frac{1}{2}k + \frac{1}{2}p) \geq c(P) + \frac{1}{12}k + \frac{1}{6}z + \frac{7}{12}p \geq c(P) + \frac{2}{3}k + \frac{1}{3}z + \frac{1}{2}p, \]
where \( p \geq z \) is used in the last inequality.

**Proof of Lemma 2** The basic idea behind the proof of the lemma is the following: We go through the edges of \( A \) in order, and try to add them to \( T \) if this does not create a cycle or node of degree three in \( T \). If we cannot add an edge, we simply skip the edge and continue to the next edge. Since the edges in \( T \) form a collection of disjoint paths and each node in \( A \) has degree one in \( T \), we can always add either the first edge or the second edge of \( A \): if the first edge cannot be added, then adding it to \( T \) must create a cycle, and since the edges in \( T \) form a collection of node disjoint paths, adding the second edge of the path or cycle to \( T \) cannot create a cycle. Similarly, we need to skip at most two edges between two edges that are successfully added to \( T \): first, an edge is skipped because otherwise we create a node of degree three in \( T \), and if a second edge is skipped, then this must be because adding that edge to \( T \) would create a cycle. But then, adding the next edge on the path cannot create a cycle in \( T \).

To lower bound the number of edges from we can add from each path or cycle \( A \), we partition the edges into groups of two or three consecutive edges. For a path \( A \), the first group contains the first two edges, and each subsequent group contains the next three edges. The final group contains the last zero, one or two edges of the path. For each group except the last group, at least one edge is added to \( T \). Hence, we can conclude that we can add at least \((|A| - 4)/3\) from the groups of size three, and \(1\) for the first group, for a total of \((|A| - 1)/3\) edges, where \(|A|\) denotes the number of edges in \( A \). For a cycle \( A \), we need to be slightly more careful, since the argument that we can add at least one edge from the last group of size three does not hold if the very first edge was added to \( T \) (since it may be the case that the first and third edge of the group cannot be added without creating a node of degree three, and the second edge of the group cannot be added without creating a cycle). Therefore, we let the first group contain two consecutive edges, where the second edge is the edge that was the first to be added to \( T \). By the same argument as for the path, we can thus conclude that we can add at least \((|A| - 1)/3\) edges.

We now show that by being a little more careful, we can in fact add \(|A|/3\) edges if \( A \) is a cycle. Note that the number of nodes in \( A \) is odd, and hence there must be some \( j \) such that the path in \( T \) that starts in \( u_j \) ends in some node \( v \notin A \). We claim that if we consider the edges in \( A \) starting with either edge \{\( u_{j-1}, u_j \)\} or edge \{\( u_j, u_{j+1} \)\}, we are guaranteed that for at least one of these starting points, we can add both the first and the third edge to \( T \).

Clearly, neither \{\( u_{j-1}, u_j \)\} nor \{\( u_j, u_{j+1} \)\} can create a cycle if we add it to \( T \). So suppose that \( T \cup \{u_{j-1}, u_j\} \cup \{u_j, u_{j+1}\} \) contains a cycle. This cycle does not contain the node \( u_j \), because the path in \( T \) that starts in \( u_j \) ends in some node \( v \notin C \). Hence \( T \) contains a path that starts in \( u_{j+1} \) and ends in \( u_{j+2} \). But then \( T \cup \{u_j, u_{j+1}\} \cup \{u_{j+2}, u_{j+3}\} \) does not have a cycle, since if it did, \( T \) must have a path starting in \( u_{j+2} \) and ending in \( u_{j+3} \) which is only possible if \( u_{j+1} = u_{j+3} \). Since the number of nodes in \( A \) is at least three, this is not possible.

We remark that the ratio of \( \frac{7}{6} \) in Theorem 2 is achieved if every 1-path contains just one edge of cost 1, and all cycle edges have cost 1. However, in such a case, we could find another optimal F2M solution of the same cost, which has fewer cycle edges: If
On the integrality gap of the subtour LP for the 1,2-TSP

we have a 1-path of cost 1 with endpoints in two different odd cycles of edges with $x(e) = \frac{1}{2}$, we can obtain the alternative solution by removing the 1-path, and increasing the $x$-value on the four cycle edges incident on its endpoints to 1, and then alternating between setting the $x$-value to 0 and 1 around the cycles. Now, since the cycles are odd, the degree constraints are again satisfied. The objective value does not increase because we only change the $x$-value on edges of cost 1. For a 1-path of cost 1 with endpoints in the same odd cycle, the cycle gives us two paths between the endpoints, one of odd length and one of even length. We can alternate increasing and decreasing the $x$-value by $\frac{1}{2}$ on the odd-length path and finally decrease the $x$-value for the 1-path to $\frac{1}{2}$, to obtain a new F2M solution of the same cost with fewer cycle edges. We note that these modifications may increase the number of components of the F2M solution.

This motivates the following definition. We call an F2M solution canonical, if all edges in the support have cost 1 and all 1-paths contain at least two edges. If a canonical F2M solution is connected, we can improve the analysis in Theorem 2 to show the following.

**Theorem 3** There exists a tour of cost at most $\frac{10}{9}$ times the cost of a connected canonical F2M solution if $c(i, j) \in \{1, 2\}$ for all $i, j$.

**Proof** We adapt the final paragraph of the proof of Theorem 2. As before, the cost of the tour is at most $c(P) + \frac{2}{3}k + \frac{1}{3}p + \frac{1}{3}z$. However, since all cycle edges have cost 1, $p = 0$ and $z = 0$. The cost of the tour is thus at most $c(P) + \frac{2}{3}k$.

The cost of the F2M solution is $c(P) + \frac{1}{2}k$. Since each cycle node is the endpoint of a 1-path and vice versa, the number of 1-paths is $k/2$. By the fact that $x$ is canonical, each of these 1-paths has cost at least two, so we get that $c(P) \geq k$. The proof is concluded by noting that then $\frac{10}{9} (c(P) + \frac{1}{2}k) \geq c(P) + \frac{1}{9}k + \frac{10}{9} \cdot \frac{1}{2}k = c(P) + \frac{2}{3}k$. 

\[ \square \]

### 4 An upper bound of $\frac{5}{3}$ on the integrality gap

We now show how to use the results in the previous two sections to obtain an upper bound of $\frac{5}{3}$ on the integrality gap for the general case. In addition, we show that if all edges in the support of the optimal subtour LP solution have cost 1, then the integrality gap is at most $\frac{26}{21}$.

We will bound the integrality gap of the solution obtained by the Papadimitriou–Yannakakis algorithm, by (i) bounding the difference between the cost of the 2M and the subtour LP, and (ii) bounding the difference between the 2M solution and the tour constructed from it by the Papadimitriou–Yannakakis algorithm.

As in the Papadimitriou–Yannakakis algorithm described in Sect. 2, we call a cycle in a 2M a “pure” cycle if all its edges have cost 1, and a “non-pure” cycle otherwise. The idea behind this section is to show that the quantity in (i) can be “charged” to the nodes in the non-pure cycle only, and that the quantity in (ii) can be “charged” mainly to the nodes in the pure cycles.

We first state the following lemma, which formalizes the second statement.
Lemma 3 If \( \text{OPT}(\text{SUBT}) < n + 1 \), then the difference between the cost of the 2M used and the tour constructed by the Papadimitriou–Yannakakis algorithm can be upper bounded by \( an_{\text{pure}} + \beta(n_{\text{non-pure}} - \ell) \), where \( n_{\text{pure}} \) is the number of nodes in pure cycles in the 2M, \( n_{\text{non-pure}} \) is the number of nodes in the non-pure cycle, and \( \ell \) is the number of edges of cost 2 in the non-pure cycle, for any values of \( \alpha, \beta \) so that \( 9\alpha \geq 2 \) and \( 3\alpha + 2\beta \geq 1 \).

Note that Lemma 1 and the assumption that \( \text{OPT}(\text{SUBT}) < n + 1 \) imply that the Papadimitriou–Yannakakis algorithm finds a bipartite matching that matches all the pure cycles. A careful look at the analysis of Papadimitriou and Yannakakis [18] then shows that their algorithm finds a tour which satisfies the lemma. The details basically follow the analysis of Papadimitriou and Yannakakis, and are therefore postponed to Appendix 1.

The key observation in this section is that we can indeed restrict our attention to instances with \( \text{OPT}(\text{SUBT}) < n + 1 \), the requirement of Lemma 3.

Lemma 4 The worst-case integrality gap is attained on an instance with subtour LP value less than \( n + 1 \), where \( n \) is the number of nodes in the instance.

Proof Consider an instance \( I \) on \( n \) nodes for which the ratio between the length of the optimal tour and the subtour LP value is \( \gamma \), and suppose \( \text{OPT}(\text{SUBT}) = n + k \) for some \( k \geq 1 \). We construct an instance \( I' \) with \( n' = n + 1 \) nodes, for which the ratio between the length of the optimal tour and the subtour LP value is at least \( \gamma \), and the optimal value of the subtour LP is at most \( n + k = n' + k - 1 \). Repeatedly applying this procedure proves the lemma.

If \( \text{OPT}(\text{SUBT}) = n + k \), then the subtour LP solution on \( I \) has a total \( x \)-value of \( k \) on edges of cost 2, since the objective value is equal to \( \sum_{e \in E} x(e) + \sum_{e \in E: c(e) = 2} x(e) \), and \( \sum_{e \in E} x(e) = \frac{1}{2} \sum_{v \in V} \sum_{e \in \delta(v)} x(e) = n \). We fix an optimal subtour solution \( x \), and we construct \( I' \) from \( I \), by adding one node \( i \), and adding edges \((i, j)\) of cost 1, for every \( j \) in \( I \) that is incident on an edge \( e \) with \( c(e) = 2 \) and \( x(e) > 0 \). All other edges incident on \( i \) get cost 2. Note that the optimal tour on \( I' \) has length at least the length of the optimal tour on \( I \), since we can take a tour on \( I' \) and shortcut \( i \) to obtain a tour on \( I \). On the other hand, we can use \( x \) to define a feasible solution on \( I' \), by “rerouting” one unit in total from edges \( e = (j, k) \) with \( c(e) = 2 \) to the edges \((j, i)\) and \((i, k)\). Since the cost of this solution on \( I' \) is the same as the cost of \( x \), the ratio between the length of the optimal tour and the subtour LP value has not decreased.

Remark 1 We note that the proof of Lemma 4 implies that to compute integrality gaps or approximation guarantees, we may assume without loss of generality that an instance has an optimal subtour LP value of at most \( n + 1 \), where \( n \) is the number of nodes in the instance. If this does not hold, we may add nodes as in the proof of Lemma 4 without increasing \( \text{OPT}(\text{SUBT}) \), and a tour of cost \( C \) on the extended instance can be shortcut to a tour on the original instance of cost at most \( C \).

Theorem 4 The integrality gap of the subtour LP is at most \( \frac{5}{4} \) for the 1,2-TSP, and it is at most \( \frac{26}{17} \) for 1,2-TSP instances for which \( \text{OPT}(\text{SUBT}) < n + \frac{1}{2} \), where \( n \) is the number of nodes in the instance.
Proof

By Lemma 4, we can assume without loss of generality that $OPT(SUBT) < n + 1$. To compute a tour, we first drop the subtour elimination constraints and find an optimal F2M solution. Since the F2M problem is a relaxation of the subtour LP, and it is half-integral, its objective value is either $n + \frac{1}{2}$ or $n$. Furthermore, note that in order to bound the integrality gap, we may assume that every edge that is not in the support of the optimal F2M solution has cost 2.

We first consider the case $OPT(SUBT) < n + \frac{1}{2}$, in which case the optimal F2M solution has objective value $n$. Since all edges in the support of the F2M solution have cost 1, we may assume by the arguments preceding Theorem 3 that all 1-paths contain at least two edges of cost 1; in other words, we may assume the components of the F2M solution are canonical. By applying Theorem 3 we convert each fractional component of the F2M solution into a cycle on the nodes in the component.

Note that each cycle that is the result of applying Theorem 3 contains at least one edge that is not in the optimal F2M solution, and thus it has at least one edge of cost 2. By the observation of Papadimitriou and Yannakakis [18], we may merge these into a single non-pure cycle. The integer components of the F2M solution are pure cycles, since the support of the F2M solution only contains edges of cost 1. We let $n_{\text{pure}}$ be the number of nodes in the pure cycles (or, equivalently, in the integer components of the F2M solution), and let $n_{\text{non-pure}}$ be the number of nodes in the non-pure cycle (or, equivalently, the number of nodes in the fractional components of the F2M solution). Let $\ell$ be the number of cost-2 edges in the computed 2-matching. Then the cost of the computed 2-matching is $n + \ell = n_{\text{pure}} + n_{\text{non-pure}} + \ell$, by the assumption that the optimal F2M solution has objective value $n$. By Theorem 3, $n_{\text{non-pure}} + \ell \leq \frac{10}{9} n_{\text{non-pure}}$, i.e., $\ell \leq \frac{1}{9} n_{\text{non-pure}}$.

If we apply the Papadimitriou–Yannakakis algorithm to this 2-matching, this increases the cost by at most $an_{\text{pure}} + \beta(n_{\text{non-pure}} - \ell)$, provided that $9\alpha \geq 2$ and $3\alpha + 2\beta \geq 1$ by Lemma 3. Choosing $\alpha = \frac{5}{21}$, $\beta = \frac{4}{7}$, we thus find that the total cost of the tour is at most $n + \ell + \frac{5}{21} n_{\text{pure}} + \frac{4}{7} n_{\text{non-pure}} - \frac{1}{7} \ell \leq n + \frac{5}{21} n_{\text{pure}} + (\frac{5}{7} + \frac{4}{9} \cdot \frac{1}{9}) n_{\text{non-pure}} = (1 + \frac{5}{21}) n$, where we used the fact that $\ell \leq \frac{1}{9} n_{\text{non-pure}}$.

If $n + \frac{1}{2} \leq OPT(SUBT) < n + 1$, the optimal F2M solution has cost at most $n + \frac{1}{2}$. We temporarily decrease the cost of the unique cost-2 edge in the F2M to 1, and follow the same procedure as above, to find a 2-matching. Let $n_{\text{non-pure}}$ be the number of nodes in the non-pure cycle, and note that $n_{\text{non-pure}}$ is at least 9, since a fractional component of a canonical F2M solution contains at least two odd cycles, containing at least six nodes, and at least three 1-paths, containing at least one additional node each.

Let the cost of the computed 2-matching (with respect to the true costs) be $n + \ell$; in other words, the procedure from Theorem 3 added $\ell - 1$ edges of cost 2 in addition to the single cost-2 edge in the F2M. By Theorem 3, $\ell - 1 \leq \frac{1}{9} n_{\text{non-pure}}$. As in the case when $OPT(SUBT) < n + \frac{1}{2}$, we apply the Papadimitriou–Yannakakis algorithm to this 2-matching, and by Lemma 3 this increases the cost by at most $an_{\text{pure}} + \beta(n_{\text{non-pure}} - \ell)$. We now choose $\alpha = \frac{4}{9}$, $\beta = \frac{1}{8}$, to get that the total cost of the tour is at most $n + \ell + \frac{4}{9} n_{\text{pure}} + \frac{1}{8} n_{\text{non-pure}} - \frac{1}{8} \ell = n + \frac{4}{9} n_{\text{pure}} + \frac{9}{8} n_{\text{non-pure}} + \frac{7}{8} (\ell - 1) + \frac{7}{8}$. Now, recall that $\ell - 1 \leq \frac{1}{9} n_{\text{non-pure}}$ and that $n_{\text{non-pure}} \geq 9$, and thus $\frac{7}{8} \leq \frac{5}{8} + \frac{1}{4} \cdot \frac{1}{9} n_{\text{non-pure}}$. Hence,
we can upper bound the cost of the tour by \( n + \frac{1}{4}n_{\text{pure}} + \left( \frac{9}{8} + \frac{7}{9} + \frac{1}{4} \cdot \frac{1}{9} \right)n_{\text{non-pure}} + \frac{5}{8} = \frac{5}{4}(n + \frac{1}{2}) \leq \frac{5}{4}OPT(\text{SUBT}). \)

**Remark 2** The bound of \( \frac{5}{4} \) in Theorem 4 may be marginally improved by a more careful analysis of small instances. It appears that in order to decrease the bound to \( \frac{10}{9} \), or even \( \frac{11}{9} \), more substantial new ideas are needed, however.

**Remark 3** The results in this section yield a polynomial time algorithm for finding a tour of cost at most \( \frac{5}{4}OPT(\text{SUBT}) \): first, solve the F2M problem, and use the procedure from the proof of Lemma 4 to ensure that \( OPT(\text{SUBT}) < n + 1 \) by adding nodes if necessary. Then, apply the procedure from the proof of Theorem 4 to find a tour of cost at most \( \frac{5}{4}OPT(\text{SUBT}) \). Finally, shortcut the nodes added by the procedure from Lemma 4 to get a tour for the original instance.

### 5 Computational results

In the case of the 1,2-TSP, for a fixed \( n \) we can generate all instances as follows. For each value of \( n \), we first generate all nonisomorphic graphs on \( n \) nodes using the software package NAUTY [13]. We let the cost of edges be one for all edges in \( G \) and let the cost of all other edges be two. Then each of the generated graph \( G \) gives us an instance of 1,2-TSP problem with \( n \) nodes, and this covers all instances of the 1,2-TSP for size \( n \) up to isomorphism.

In fact, we can do slightly better by only generating biconnected graphs. We say that a graph \( G = (V, E) \) is **biconnected** if it is connected and there is no vertex \( v \in V \) such that removing \( v \) disconnects the graph; such a vertex \( v \) is a **cut vertex**. It is possible to show that the subtour LP value is at least \( n + 1 \) if \( G \) is not biconnected, hence, by Lemma 4 it suffices to consider biconnected graphs. However, the proof of Lemma 4 involves adding additional new nodes (perhaps many of them). Using a similar technique to the one in the proof of Lemma 4, one can show that given a graph on \( n \) vertices, there is a biconnected graph on at most \( n + 2 \) vertices that has no better ratio of optimal tour to subtour LP value. In Appendix 2 we prove two lemmas that imply the following corollary.

**Corollary 1** Let \( G = (V, E) \) be the graph of cost 1 edges in a 1,2-TSP instance. Then if \( G = (V, E) \) is not biconnected, there exists a biconnected \( G' = (V', E') \) with \( |V'| \leq |V| + 2 \) such that \( OPT(G)/SUBT(G) \leq OPT(G')/SUBT(G') \).

For each instance of size \( n \), we solve the subtour LP and the corresponding integer program using CPLEX 12.1 [12] and a Macintosh laptop computer with dual core 2GHz processor and 1GB of memory. It is known that the integrality gap is 1 for \( n \leq 5 \), so we only consider problems of size \( n \geq 6 \). The results are summarized in Table 1. For \( n = 11 \), the number of nonisomorphic biconnected graphs is nearly a billion and thus too large to consider, so we turn to another approach. For \( n = 11 \) and \( n = 12 \), we use the fact that we know a lower bound on the integrality gap of \( \frac{n+1}{n} \), namely for the instances depicted in Fig. 2. The claimed lower bounds on the integrality gap for these instances follow readily from the integrality gap for the example in Fig. 1.
Table 1 The subtour LP integrality gap for 1,2-TSP for $6 \leq n \leq 12$, where the ratio for $6 \leq n \leq 10$ is only on biconnected graphs

<table>
<thead>
<tr>
<th>$n$</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subtour IP/LP ratio</td>
<td>8/7.5</td>
<td>8/7.5</td>
<td>9/8.5</td>
<td>10/9</td>
<td>11/10</td>
<td>12/11</td>
<td>13/12</td>
</tr>
<tr>
<td>No. of graphs</td>
<td>56</td>
<td>468</td>
<td>7,123</td>
<td>194,066</td>
<td>9,743,542</td>
<td>900,969,091</td>
<td>–</td>
</tr>
</tbody>
</table>

The second row shows the number of nonisomorphic biconnected graphs for $6 \leq n \leq 11$

![Fig. 2](image-url) Illustration of the instances with integrality gap at least $\frac{12}{11}$ for $n = 11$ (without the grey node) and $\frac{13}{12}$ for $n = 12$ (with the grey node) for the 1,2-TSP. All edges of cost 1 are shown.

We then check whether this is the worst integrality gap for each vertex of subtour LP. A list of non-isomorphic vertices of the subtour LP is available for $n = 6$ to 12 at Sylvia Boyd’s website [http://www.site.uottawa.ca/~sylvia/subtourvertices](http://www.site.uottawa.ca/~sylvia/subtourvertices). In order to check whether the lower bound on the integrality gap is tight, we solve the following integer programming problem for each vertex $x$ of the polytope for $n = 11$ and $n = 12$, where now the costs $c(e)$ are the decision variables, and $x$ is fixed:

\[
\text{Max } z - \alpha_n \sum_{e \in E} c(e)x(e)
\]

subject to:

\[
\sum_{e \in T} c(e) \geq z, \quad \forall \text{ tours } T,
\]

\[
c(e) \in \{1, 2\}, \quad \forall e \in E.
\]

Note that $\alpha_n$ is the lower bound on the integrality gap for instances of $n$ nodes. If the objective is nonpositive for all of the vertices of the subtour LP, then we know that $\alpha_n$ is the integrality gap for a particular value of $n$.

Since the number of non-isomorphic tours of $n$ nodes is $(n - 1)!/2$, the number of constraints is too large for CPLEX for $n = 11$ or 12. We overcome this difficulty by first solving the problem with only tours that have at least $n - 1$ edges in the support graph of the vertex $x$, and repeatedly adding additional violated tours. We find that the worst case integrality gap for $n = 11$ is $\frac{12}{11}$ and for $n = 12$ is $\frac{13}{12}$.

We can now observe that our overall computation leads to a bound of $\frac{10}{9}$ on the integrality gap for instances of the 1,2-TSP with $n \leq 12$. Suppose the worst-case integrality gap for these instances is attained for an instance with $k$ vertices. If $k \leq 8$, then we know that there is a biconnected graph on at most 10 vertices with no better integrality gap, and we have determined the worst-case ratio for all biconnected graphs.
6 Conjectures and conclusions

As stated in the introduction, we conjecture the following.

**Conjecture 2** The integrality gap of the subtour LP for the 1,2-TSP is \( \frac{10}{9} \).

Schalekamp et al. [20] have conjectured that to determine the integrality gap for the subtour LP, we can restrict ourselves to considering instances, which have an optimal solution that is an extreme point of the F2M polytope.

We have shown in Theorem 2 that if an analogous conjecture is true for 1,2-TSP, then the integrality gap for 1,2-TSP is at most \( \frac{7}{6} \); it would be nice to show that if the analogous conjecture is true for 1,2-TSP then the integrality gap is at most \( \frac{10}{9} \).

Finally, we remark that the integrality gap of the linear program obtained by adding the constraints

\[
\sum_{e \in \delta(S) \setminus F} x(e) + \sum_{e \in F} (1 - x(e)) \geq 1 \quad \forall S \subseteq V, \ F \subseteq \delta(S), \ |F| \text{ odd},
\]

to (\( \text{SUBT} \)) is at most \( \frac{11}{9} \) by (4) and Lemma 1, since the 2M polytope is described by these additional constraints plus the degree constraints. It is an interesting question whether the analysis of Berman and Karpinski [3] can also be expressed in terms of the optimal value of this stronger LP.

**Acknowledgments** We thank Sylvia Boyd for useful and encouraging discussions. We thank two anonymous referees for helpful comments and suggestions.

**Appendix 1: Proof of Lemma 3**

We now prove Lemma 3 from Sect. 4.

**Lemma 5** If \( \text{OPT}(\text{SUBT}) < n + 1 \), then the difference between the cost of the 2M used and the tour constructed by the Papadimitriou–Yannakakis algorithm can be upper bounded by \( \alpha n_{\text{pure}} + \beta (n_{\text{non-pure}} - \ell) \), where \( n_{\text{pure}} \) is the number of nodes in pure cycles in the 2M, \( n_{\text{non-pure}} \) is the number of nodes in the non-pure cycle, and \( \ell \) is the number of edges of cost 2 in the non-pure cycle, for any values of \( \alpha, \beta \) so that \( 9\alpha \geq 2 \) and \( 3\alpha + 2\beta \geq 1 \).

**Proof** Recall from Sect. 2 that the Papadimitriou–Yannakakis algorithm starts by finding a maximum cardinality bipartite matching in a graph which has a node for
each pure cycle on one side, and a node for each node in the instance on the other side. There is an edge \((C, i)\) if \(i \not\in C\), and there exists some node \(j\) in \(C\) such that \((i, j)\) is an edge of cost 1.

In Lemma 1, we show that \(OPT(SUBT) \geq n + r\), where \(r\) is the number of pure cycles that are not matched in the maximum cardinality bipartite matching. Hence, the assumption that \(OPT(SUBT) < n + 1\) implies that all the pure cycles are matched. In order to show that this implies the lemma, we will repeat some key parts of the algorithm and analysis of Papadimitriou and Yannakakis.

Consider the directed graph \(F = (\mathcal{C}, A)\) which has a node for every cycle in the instance of the 1,2-TSP, and an arc \((C, C')\) if the maximum cardinality bipartite matching contains an edge from cycle \(C\) to a node \(i\) in cycle \(C'\). Each node in \(F\) that corresponds to a pure cycle has outdegree 1, and the non-pure cycle (if it exists) has outdegree 0. Papadimitriou and Yannakakis show how to find a spanning subgraph of \(F'\) of \(F\) such that each nontrivial component is an in-tree of depth one or a path of length two. The only possible trivial component is the node that corresponds to the non-pure cycle. Since the non-pure cycle has outdegree 0, it can only occur in a nontrivial component as the root of an in-tree, or as the endpoint of a path of length two. It turns out that the latter does not happen in the construction described by Papadimitriou and Yannakakis, but even if it did, we could just remove the last edge in the length-two path to obtain one in-tree of depth one and one trivial component containing the non-pure cycle. Hence, we may assume the non-pure cycle only occurs in a nontrivial component as the root of an in-tree.

Papadimitriou and Yannakakis now merge the cycles in one component of \(F'\) into a single cycle containing at least one edge of cost 2 as follows: If the component is an in-tree of depth one, let \(C\) be the cycle corresponding to the root, let \(C_1, \ldots, C_m\) be the remaining cycles in the component, and let \(v_i\) be the node in \(C\) such that \((C_i, v_i)\) was in the bipartite matching. We consider the nodes in \(C\) in clockwise order, starting from a node \(v \neq v_i\) for \(i = 1, \ldots, m\) if such a node exists, and an arbitrary node \(v\) otherwise. If we encounter two adjacent nodes \(v_i, v_j\) in \(\{v_1, \ldots, v_m\}\), then we merge the corresponding cycles \(C_i\) and \(C_j\) with \(C\) according to (a) in Fig. 3. Otherwise, if the current node is \(v_j\) but its clockwise neighbor is not or if its clockwise neighbor is the first node \(v\), then we merge \(C_j\) with \(C\) as in (b) in Fig. 3. Finally, if the component is a path of length two, we merge the three cycles as in (c) in Fig. 3. Note that each cycle in the resulting graph contains at least one edge of cost 2, and hence we can find a tour of the same cost by removing the edges of cost 2, and arbitrarily connecting the resulting paths into a tour.

We now show that the number of edges of cost 2 that are added by merging cycles according to Fig. 3 can be upper bounded by \(\alpha n_{\text{pure}} + \beta (n_{\text{non-pure}} - \ell)\), provided that \(\alpha\) and \(\beta\) are so that \(9\alpha \geq 2\) and \(3\alpha + 2\beta \geq 1\).

We say a node is involved in a merging if it is either a node in one of the cycles that are fully drawn in Fig. 3, or if it is node \(v\) or \(v_i\) in subfigure (b). Note that each node is involved in at most one merging. Recall that the non-pure cycle can only occur as the root of a 1-tree of depth one or as a trivial component in \(F'\), and hence, only the partially drawn cycle in (a) and (b) is (potentially) a non-pure cycle.

We now examine each of the cases (a), (b) and (c) in Fig. 3 in turn. In Fig. 3a one edge of cost 2 is added and we can charge this edge to the (at least) 6 nodes from pure
cycles involved in this merging, as long as $6\alpha \geq 1$. This is indeed the case, because we have the stronger requirement that $9\alpha \geq 2$. In (b), again, one edge of cost 2 is added, and we can charge the edge to the (at least) three nodes of the pure cycle involved in the merging and the 2 nodes of the (potentially) non-pure cycle involved in the merging, as long as $3\alpha + 2\beta \geq 1$ (in case the 2 nodes were part of the non-pure cycle), and $5\alpha \geq 1$ (in case the 2 nodes were part of a pure cycle). Finally, in Fig. 3c, two edges of cost 2 are added; we can charge the two edges to the (at least) nine nodes from pure cycles involved in the merging as long as $9\alpha \geq 2$.

Hence, we have shown that difference in cost between the tour and the 2M can be charged to the nodes, in such a way that each node is charged at most once, and a node in a pure cycle is charged at most $\alpha$ and a node in a non-pure cycle is charged at most $\beta$.

Finally, we remark that a node in a non-pure cycle is charged only in case (b). Now, if $(v_i, v)$ in Fig. 3b is an edge of cost 2, then there is no need to charge any nodes, since the cost after merging is the same as before the merge. Hence, if we direct all edges of the non-pure cycle in clockwise direction, then the head of the edges of cost 2 is never charged. The total change to the nodes in the non-pure cycle is therefore at most $\beta(n_{\text{non-pure}} - \ell)$.

\[\square\]

Appendix 2: Proof of Corollary 1

We now show that the worst-case integrality gap for the subtour LP for the 1,2-TSP can be found on graphs of cost 1 edges that are biconnected, as stated in Corollary 1 in Sect. 5. Let $OPT(G)$ and $SUBT(G)$ be the cost of the optimal tour and the value of
Define a solution $x$.

**Lemma 6** Let $G = (V, E)$ be the graph of cost 1 edges in a 1,2-TSP instance. Then if $G = (V, E)$ is not connected, there exists a connected graph $G' = (V', E')$ with $|V'| = |V| + 1$ such that $\text{OPT}(G)/\text{SUBT}(G) \leq \text{OPT}(G')/\text{SUBT}(G')$.

**Proof** Suppose $G$ has more than one connected component. We create $G' = (V', E')$ by adding a new vertex $i^*$ to the graph, and adding edges from all $j \in V$ to $i^*$ so that $V' = V \cup \{i^*\}$ and $E' = E \cup \{(i^*, j) : j \in V\}$. Given a tour of $G'$, we can easily produce a tour of $G$ of no greater cost by shortcutting $i^*$, so that $\text{OPT}(G) \leq \text{OPT}(G')$. Let $x$ be an optimal solution to the subtour LP for the graph $G$. We now define a solution $x'$ for $G'$, where $x'_{ij} = x_{ij}$ if $i$ and $j$ are in the same connected component of $G$, while if $i$ and $j$ are in different connected components of $G$, then we set $x'_{ij} = 0$, $x'_{i^*j} = x_{ij}$, and $x'_{ij} = x_{ij}$. It is easy to see that the cost of $x'$ is the same as that of $x$. We now argue that there is some solution $x''$ feasible for the subtour LP on $G'$ such that its cost is no greater, so that $\text{SUBT}(G') \leq \text{SUBT}(G)$. It is clear that the bounds constraints (3) are satisfied for $x'$ and the degree constraints (1) are satisfied for $x'$ for all $i \in V$; however, the degree constraint for $i^*$ may not be satisfied. Since for any component $C \subseteq V$ of $G$, $x'(\delta(C)) \geq 2$, it is clear that $x'(\delta(i^*)) \geq 2$, but it may be the case that $x'(\delta(i^*)) > 2$. For the subtour constraints (2), consider any $S \subseteq V'$, $S \neq \emptyset$, such that $i^* \not\in S$. Then $x'(\delta(S)) \geq x'(\delta(S)) \geq 2$, and for any $S \subseteq V'$ with $i^* \in S$, $S \neq \{i^*\}$, $x'(\delta(S)) = x'(\delta(V' - S)) \geq 2$ by the previous argument. Finally, Goemans and Bertsimas [11] have shown (see also Williamson [23]) that if edge costs obey the triangle inequality, and there is some solution $x'$ to the subtour LP in which degree constraints are exceeded but all other constraints are met, then there is another feasible solution $x''$ of no greater cost in which all constraints are satisfied. Hence we have that $\text{SUBT}(G') \leq \text{SUBT}(G)$. Thus we have that $\text{OPT}(G)/\text{SUBT}(G) \leq \text{OPT}(G')/\text{SUBT}(G')$. \hfill $\square$

**Lemma 7** Let $G = (V, E)$ be the graph of cost 1 edges in a 1,2-TSP instance. Then if $G = (V, E)$ is connected but not biconnected, there exists a biconnected $G' = (V', E')$ with $|V'| = |V| + 1$ such that $\text{OPT}(G)/\text{SUBT}(G) \leq \text{OPT}(G')/\text{SUBT}(G')$.

**Proof** By hypothesis we assume that the graph $G = (V, E)$ is connected. Let $i_1, \ldots, i_k$ be all the cut vertices of $G$, and let $C_1, \ldots, C_\ell$ be all the connected components formed when these vertices are removed, so that $C_1, \ldots, C_\ell, \{i_1\}, \ldots, \{i_k\}$ form a partition of $V$. We create a new graph $G' = (V', E')$ by adding a new vertex $i^*$, and adding edges from $i^*$ to each vertex in $C_1 \cup \cdots \cup C_\ell$, so that $V' = V \cup \{i^*\}$ and $E' = E \cup \{(i^*, j) : j \in C_p \text{ for some } p\}$. We note that $G'$ is biconnected. As before, we have $\text{OPT}(G) \leq \text{OPT}(G')$ since given a tour of $G'$ we can shortcut $i^*$ to get a tour of $G$. Let $x$ be an optimal subtour LP solution for graph $G$. We now argue, as we did in the proof of Lemma 6, that we can create an $x'$ that costs no more than $x$ such that all the subtour and bounds constraints are obeyed, and all degree constraints are either met or exceeded; this will imply that $\text{SUBT}(G') \leq \text{SUBT}(G)$, and complete the proof. Suppose without loss of generality that removing cut vertex $i_1$ creates components $C_1$ and $C = C_2 \cup \cdots \cup C_\ell \cup \{i_2\} \cup \cdots \cup \{i_k\}$, so that $C_1, \{i_1\}$,
and $C$ partition $V$. We set $x'_{ij} = 0$ and $x'_{i*} = x'_{i+j} = x_{ij}$ if $i \in C_1$ and $j \in C$; $x'_{ij} = x_{ij}$ otherwise. If $i \in C_1$ and $j \in C$, then $(i, j) \notin E$ since $i_1$ is a cut vertex, so the cost of $x'$ is no more than that of $x$. The arguments that all constraints are satisfied except for the degree constraint on $i^*$ follow as in the proof of Lemma 6. We now must argue that $x'(*) \geq 2$. To do this, we show that $\sum_{i \in C_1, j \in C} x_{ij} \geq 1$. Since $x(\delta(i_1)) = 2$, it must be the case that either $\sum_{j \in C} x_{i_1j} \leq 1$ or $\sum_{j \in C} x_{i_1j} \leq 1$; without loss of generality we assume the former is true. Then since $x(\delta(C_1 \cup \{i_1\})) \geq 2$, and $x(\delta(C_1 \cup \{i_1\})) = \sum_{j \in C} x_{i_1j} + \sum_{i \in C_1, j \in C} x_{ij}$, it follows that $\sum_{i \in C_1, j \in C} x_{ij} \geq 1$, and the proof is complete.

\[\square\]

References

On the integrality gap of the subtour LP for the 1,2-TSP


