TSP with Bounded Metrics: Approximation Hardness for the (1,2) Case

Lars Engebretsen[1,*] **and Marek Karpinski**[2,**]

[1] Department of Numerical Analysis and Computer Science

Royal Institute of Technology

SE-100 44 Stockholm

Email: enge@kth.se

[2] Department of Computer Science

University of Bonn

53117 Bonn

Email: marek@cs.uni-bonn.de

*Research partly performed while the author was visiting MIT with support from the Marcus Wallenberg Foundation and the Royal Swedish Academy of Sciences.

**Supported in part by DFG grant, DIMACS, and IST grant 14036 (RAND-APX).

Abstract

The general asymmetric TSP with triangle inequality is known to be approximable only within logarithmic factors. In this paper we study the asymmetric and symmetric TSP problems with *bounded metrics*, i.e., metrics where the distances are integers between one and some constant upper bound. In this case, the problem is known to be approximable within a constant factor. We prove that it is **NP**-hard to approximate the asymmetric TSP with distances one and two within $321/320 - \varepsilon$ and that it is **NP**-hard to approximate the symmetric TSP with distances one and two within $741/740 - \varepsilon$ for every constant $\varepsilon > 0$.

1 Introduction

A common special case of the Traveling Salesman Problem (TSP) is the *metric TSP*, where the distances between the cities satisfy the triangle inequality. The decision version of this special case was shown to be **NP**-complete by Karp [9], which means that we have little hope of computing exact solutions in polynomial time. Christofides [6] has constructed an elegant algorithm approximating the metric TSP within $3/2$, i.e., an algorithm that always produces a tour whose weight is at most a factor $3/2$ from the weight of the optimal tour. For the case when the distance function may be asymmetric, the best known algorithm approximates the solution within $O(\log n)$, where n is the number of cities [8]. As for lower bounds, the PCP Theorem [1] and a result due to Papadimitriou and Yannakakis [11] together imply that there exists some constant such that it is **NP**-hard to approximate TSP where the distances are constrained to be either one or two—note that such a distance function always satisfies the triangle inequality—within that constant. This hardness result was improved by Engebretsen [7], who proved that it is, for every constant $\varepsilon > 0$, **NP**-hard to approximate TSP with distances one and two within $2805/2804 - \varepsilon$ for the asymmetric and $5381/5380 - \varepsilon$ for the symmetric, respectively, version of the problem. Böckenhauer and Seibert [4] considered the symmetric TSP with distances one, two and three, and proved a lower bound of $3813/3812 - \varepsilon$. For a discussion of bounded metric TSP, see also Trevisan [12]. It appears that the metric TSP lacks the good definability properties which seem to be needed for proving strong inapproximability results. Therefore, any new insights into explicit lower bounds here are of considerable interest.

Papadimitriou and Vempala [10] recently announced stronger approximation hardness results for the asymmetric and symmetric versions of the TSP with graph metric, but left the case of TSP with *bounded metric* open. However, their original proof contained an error influencing the explicit constants. A new proof with the new constants of $117/116 - \varepsilon$ and $220/219 - \varepsilon$, respectively, was announced by Papadimitriou and Vempala in May 2002 (the latest version of the paper is available from URL http://www-math.mit.edu/~vempala/papers/tspinapprox.ps). Apart from being an interesting question on its own, it is conceivable that the special cases with bounded metric are easier to approximate than the cases when the distance between two points can grow with the number of cities in the instance. Indeed, the asymmetric TSP with distances bounded by B can be approximated within B by just picking any tour as the solution and the asymmetric TSP with distances one and two can be approximated within $4/3$ [3]. The symmetric version of the latter problem can be approximated within $7/6$ [11].

Definition 1.1. *The* Asymmetric Traveling Salesman Problem (ATSP) *is the following minimization problem: Given a collection of cities and a matrix whose entries are interpreted as the distance from a city to another, find the shortest tour starting and ending in the same city and visiting every city exactly once.*

Definition 1.2. (1,B)-ATSP *is the special case of ATSP where the entries in the distance matrix obey the triangle inequality and the off-diagonal entries in the distance matrix are integers between 1 and B.* (1,B)-TSP *is the special case of (1,B)-ATSP where the distance matrix is symmetric.*

In this paper, we prove that it is, for any constant $\varepsilon > 0$, **NP**-hard to approximate (1,2)-ATSP within $321/320 - \varepsilon$ (Corollary 2.2), and that it is, for any constant $\varepsilon > 0$, **NP**-hard to approximate (1,2)-TSP within $741/740 - \varepsilon$ (Corollary 3.1). This shows that the currently best known bounds for TSP with bounded metrics are, in some sense, not that far from the best currently known bounds for general TSP with triangle inequality. Specifically, the bounds for TSP with graph metric announced by Papadimitriou and Vempala in May 2002 can be written as $1 + \varepsilon$, where $\varepsilon \approx 0.01$ for the asymmetric TSP and $\varepsilon \approx 0.005$ for the symmetric TSP. We show, on the other hand, bounds for (1,B)-(A)TSP that are of the same form but with $\varepsilon \approx 0.003$ and $\varepsilon \approx 0.0013$, respectively.

By relaxing the requirement on the "boundedness" of the metric, i.e., by allowing some larger, but still constant, B in the (1,B)-(A)TSP problem, the actual constants in the approximation hardness results for TSP with bounded metrics can be made even closer to the constants obtained by Papadimitriou and Vempala. We elucidate on this matter in a sequel to this article.

The proofs of our approximation hardness results follow by reduction from the problem *Hybrid* introduced by Berman and Karpinski [2]. Another way to improve our bounds is therefore to establish stronger approximation hardness results for Hybrid. Some such progress has recently been reported by Chlebíková and Chlebík [5].

2 The approximation hardness of (1,2)-ATSP

As mentioned above, we prove our hardness results by reduction from the problem *Hybrid*, introduced by Berman and Karpinski [2] to prove hardness results for special cases of several combinatorial optimization problems where the number of occurrences of every variable is bounded by some constant. Essentially, Hybrid is the problem of maximizing, given a system of linear equations with special structure, the number of satisfied equations. The special structure of the linear equations in Hybrid is particularly well-suited for our reduction: The equations have either two or three unknowns and each variable occurs exactly three times in the instance.

The main idea in the reduction is the same as in earlier reductions [7, 11]; the reduction is *local* and *gadget based*. Specifically, each equation in the Hybrid instance is transformed into a certain subgraph of the TSP instance—a so called *gadget*. Different parts of the gadget correspond to the different variables participating in the equation. The gadgets are then linked together to form a circle. By the construction of the gadgets, there is a natural way to interpret a TSP tour in the resulting graph as an assignment to the variables in the Hybrid instance. To ensure that there is a certain connection between the length of the TSP tour and the number of equations satisfied by the corresponding assignment, the parts of the instance corresponding to the same variable are connected to each other in a certain way.

To obtain a good approximation hardness result, the gadgets must, loosely speaking, contain as few nodes as possible. On the other hand, the major challenge in the proof of correctness is to prove that *every* TSP tour in the resulting graph can be interpreted as an assignment to the variables in the Hybrid instance with the property that the number of satisfied equations is connected to the cost of the tour. Such connections are usually easier to establish when the gadgets contain more nodes. In this work, we are able to improve the approximation hardness constants by, firstly, observing that the Hybrid instances actually have even more structure than is explicitly stated by Berman and Karpinski [2] and, secondly, using gadgets with few nodes. This requires a fairly involved argument to establish that our reduction is correct.

2.1 The Hybrid problem and its connection to TSP

In their paper on approximation hardness of bounded occurrence instances of several combinatorial optimization problems, Berman and Karpinski [2] introduced the problem *Hybrid* and proved that it is hard to approximate.

Definition 2.1. Hybrid *is the following maximization problem: Given a system of linear equations mod* 2 *containing* n *variables,* m_2 *equations with exactly two unknowns, and* m_3 *equations with exactly three unknowns, find an assignment to the variables that satisfies as many equations as possible.*

Theorem 2.1 [2]. *For any constant $\delta > 0$, there exists instances of Hybrid with 42ν variables, 60ν equations with exactly two variables, and 2ν equations with exactly three variables such that:*

1. *Each variable occurs exactly three times.*
2. *Either there is an assignment to the variables that leaves at most $\delta\nu$ equations unsatisfied, or else every assignment to the variables leaves at least $(1-\delta)\nu$ equations unsatisfied.*
3. *It is **NP**-hard to decide which of the two cases in item 2 above holds.*

Delving into the details of the Berman-Karpinski construction, it can be seen that every instance of Hybrid produced by it has an even more special structure: The equations containing three unknowns are of the form $x+y+z = \{0,1\}$; the number of such equations with right-hand side 0 is equal to the number of such equations with right-hand side 1. The equations containing two unknowns are all of the form $x_i + x_j = 0$. Moreover, the set of variables can be partitioned into classes with the property that for each class $\{x_1, x_2, \ldots, x_k\}$ of variables there are equations $x_i + x_{i+1} = 0$ $(1 \leq i < k)$ and one equation $x_k + x_1 = 0$.

By rewriting the latter equations mentioned above as $x_i + \bar{x}_{i+1} = 1$ $(1 \leq i < k)$ and $x_k + \bar{x}_1 = 1$, we have established the following corollary of Theorem 2.1:

Corollary 2.1. *There are instances of Hybrid with 42ν variables, 42ν equations of the form $x + \bar{y} = 1 \bmod 2$, 18ν equations of the form $x + y = 0 \bmod 2$, ν equations of the form $x + y + z = 0 \bmod 2$, and ν equations of the form $x + y + z = 1 \bmod 2$ such that:*

1. *Each variable occurs exactly three times, two times positively and one time negatively.*

2. *Either there is an assignment to the variables that leaves at most $\delta\nu$ equations unsatisfied, or else every assignment to the variables leaves at least $(1-\delta)\nu$ equations unsatisfied.*

3. *It is* ***NP****-hard to decide which of the two cases in item 2 above holds.*

To prove our hardness result for (1,2)-ATSP, we reduce instances of Hybrid having the form described in Corollary 2.1 to instances of (1,2)-ATSP:

Theorem 2.2. *Suppose that we are given an arbitrary instance of Hybrid with n variables, $m_{2,0}$ equations of the form $x + y = 0 \bmod 2$, $m_{2,1}$ equations of the form $x + \bar{y} = 1 \bmod 2$, $m_{3,0}$ equations of the form $x + y + z = 0 \bmod 2$, and $m_{3,1}$ equations of the form $x + y + z = 1 \bmod 2$ such that each variable occurs exactly three times, two times positively and one time negatively.*

Then it is possible to construct in polynomial time an instance of (1,2)-ATSP, with size polynomial in the size of the Hybrid instance, such that

1. *If there is an assignment to the variables in the Hybrid instance that leaves at most u equations unsatisfied, then there is a TSP tour of length $6n + m_{2,0} + m_{2,1} + 4m_{3,0} + 4m_{3,1} + u$.*

2. *From any TSP tour of length $6n+m_{2,0}+m_{2,1}+4m_{3,0}+4m_{3,1}+u$, it is possible to construct in polynomial time an assignment to the variables in the Hybrid instance that leaves at most u equations unsatisfied.*

The rest of this section is devoted to the proof of Theorem 2.2. Before turning to that, however, let us use the theorem to establish approximation hardness of (1,2)-ATSP:

Corollary 2.2. *For any constant $\varepsilon > 0$, it is* ***NP****-hard to approximate (1,2)-ATSP within $321/320 - \varepsilon$.*

Proof. Select $\delta > 0$ such that $(321 - \delta)/(320 + \delta) \geq 321/320 - \varepsilon$. From an instance of Hybrid with the structure described in Corollary 2.1, construct an instance of (1,2)-ATSP with the properties guaranteed by Theorem 2.2. Combining Theorem 2.2 with item 2 in Corollary 2.1 shows that the constructed (1,2)-ATSP instance either has a tour of length at most $6 \cdot 42v + 42v + 18v + 4v + 4v + \delta v = (320 + \delta)v$ or that every TSP tour has length at least $6 \cdot 42v + 42v + 18v + 4v + 4v + (1 - \delta)v = (321 - \delta)v$. Furthermore, item 3 in Corollary 2.1 states that it is **NP**-hard to distinguish those two cases. Therefore it is **NP**-hard to approximate (1,2)-ATSP within $(321 - \delta)/(320 + \delta) \geq 321/320 - \varepsilon$. ∎

2.2 Main ideas in the proof of Theorem 2.2

To describe a (1,2)-(A)TSP instance, it is enough to specify the edges of weight one. We do this by constructing a graph G and then let the (1,2)-(A)TSP instance have the nodes of G as cities. The distance between two cities u and v is defined to be one if (u, v) is an edge in G and two otherwise. To compute the weight of a tour, it is enough to study the parts of the tour traversing edges of G. In the asymmetric case G is a directed graph.

Definition 2.2. *We call a node where the tour leaves or enters G an* endpoint*. A node with the property that the tour both enters and leaves G in that particular node is called a* double endpoint *and counts as two endpoints.*

If c is the number of cities and $2e$ is the total number of endpoints, the weight of the tour is $c + e$ since every edge of weight two corresponds to two endpoints. Conversely, any tour of weight $c + e$ has exactly $2e$ endpoints.

On a high level, the (1,2)-ATSP instance in our reduction consists of a circle formed by *equation gadgets* representing the equations occurring in the corresponding instance of Hybrid. These equation gadgets are also connected through *consistency checkers*. We first show that any assignment satisfying all but u equations in the Hybrid instance can be transformed into a tour with exactly $2u$ endpoints. We then show that *any* TSP tour can be transformed by local transformations into another tour with equal or lower cost, and that it is possible to extract an assignment to the variables in the Hybrid instance from the way that this new tour traverses certain parts of TSP instance. This assignment satisfies all but at most $\lfloor e/2 \rfloor$ equations in the Hybrid instance, where e is the number of endpoints in the tour.

The proof of Theorem 2.2 now proceeds by first defining the gadgets and the consistency checkers, then defining the local transformations of an arbitrary TSP tour, and finally describing how an assignment can be found from the resulting tour.

2.3 Constructing a (1,2)-ATSP instance from Hybrid

The equation gadgets for equations of the form $x + y + z = \{0, 1\}$ are shown in Fig. 2; gadgets for equations of the form $x+y = 0$ and $x+\bar{y} = 1$ are shown in Fig. 3. The ticked edges in the gadgets correspond to the variables in the corresponding equation as indicated in the figures. The following properties of the gadgets can be checked by exhausting all possibilities:

Proposition 2.1. *There is a Hamiltonian path from A to B in the left gadget in Fig. 2 if and only if an even number of ticked edges is traversed and a Hamiltonian path from A to B in the right gadget in Fig. 2 if and only if an odd number of the ticked edges is traversed.*

There is a Hamiltonian path from A to B in the left gadget in Fig. 3 if and only if an even number of the ticked edges is traversed. There is a Hamiltonian path from A to B in the right gadget in Fig. 3 if and only if an odd number of the ticked edges is traversed.

The ticked edges corresponding to the same variable are joined together in a consistency checker. Specifically, the ticked edges are syntactic sugar for parts of the corresponding consistency checker. An entire consistency checker is shown in Fig. 4. A ticked edge in the equation gadgets shown in Fig. 2 corresponds to one of the three structures enclosed by a curve in Fig. 4. The correspondence is such that negated variables always correspond to the part enclosed by a dashed curve in Fig. 4—recall that each variable occurs one times negated and two times unnegated.

Note that there is no node between the two ticked edges in the gadget corresponding to equations of the form $x + y = 0$. Instead, the edge leaving the consistency checker corresponding to the first ticked edge is merged with the edge entering the consistency checker corresponding to the second ticked edge as shown in Fig. 5. This simplifies, and improves, our accounting procedure used to compute the actual approximation hardness constant.

The equation gadgets are hooked together in a circle in such a way that the node B in each gadget is identified with the node A in another gadget. The order of the gadgets is as follows: first all gadgets for equations of the form $x + y + z = 1$, then the gadgets for equations of the form $x + y + z = 0$, and finally the gadgets for equations containing two variables.

The connection between two gadgets corresponding to equations of the form $x+y+z = 1$ is "optimized" as indicated in Fig. 6. To the left, this figure shows the edges incident to B in one gadget and the edges leaving A in the other gadget; the bipartite graph on the right shows how this connection is actually implemented in our construction. This optimization improves the inapproximability factor slightly since the total number of nodes in the graph is reduced. Also the connection between the last gadget corresponding to an equation of the form $x + y + z = 1$ and the first gadget corresponding to an equation of the form $x+y+z = 0$ is optimized similarly. There is one node at A in the first gadget corresponding to an equation of the form $x + y + z = 1$; this node is shared with one gadget corresponding to an equation containing two variables.

Lemma 2.1. *A graph constructed as described above from an instance of Hybrid with n variables, $m_{2,0}$ equations of the form $x + y = 0 \bmod 2$, $m_{2,1}$ equations of the form $x + \bar{y} = 1 \bmod 2$, $m_{3,0}$ equations of the form $x + y + z = 0 \bmod 2$, and $m_{3,1}$ equations of the form $x + y + z = 1 \bmod 2$ has in total $6n + m_{2,0} + m_{2,1} + 4m_{3,0} + 4m_{3,1}$ nodes.*

Proof. There is one consistency checker for every variable; each one of them contains six nodes. Not counting the nodes inside the consistency checkers, the gadgets for equations with two variables contain two nodes; both those nodes are shared between two gadgets. Hence each gadget corresponding to a two-variable equation contains, on average, one node.

Gadgets for equations of the form $x + y + z = 1$ as shown in Fig. 2 contain four nodes—except for the "leftmost" one which contains one extra node that is shared with another gadget. Similarly, gadgets for equations of the form $x + y + z = 0$ contain five nodes, two of which are shared between two gadgets—again except for the "leftmost" gadget which contains four nodes, one of which is shared with another gadget. Hence each gadget corresponding to a three-variable equation contains, on average, four nodes. ∎

2.4 Constructing a tour from an assignment

Consider an instance of Hybrid and an instance of (1,2)-ATSP constructed from it as described in § 2.3. Let π be an assignment to the variables in the Hybrid instance. We now describe a TSP tour corresponding to this assignment.

Consider the tour that 1) For each variable x traverses the consistency checker corresponding to x as shown in Fig. 7a if $\pi(x) = 0$ and as shown in Fig. 7b if

$\pi(x) = 1$. 2) For each equation gadget enters each equation gadget at A, takes the shortest possible way to B under the condition that the ticked edges are traversed as prescribed by the traversals of the consistency checkers described above, and then exits the equation gadget at B.

Such a tour has precisely two endpoints in each equation gadget corresponding to an unsatisfied equation and no endpoints elsewhere. (A slight technicality arises here, however, since the three ticked edges in a gadget corresponding to equations of the form $x + y + z = 0$ cannot be simultaneously traversed—that would result in a short cycle. Similarly, both edges in gadgets corresponding to equations of the form $x + \bar{y} = 1$ cannot be simultaneously traversed. We resolve these issues by defining the tour as shown in Figs. 8 and 9, thereby maintaining the property that the tour has two endpoints for each unsatisfied equation and no other endpoints.) The properties of the above construction can be summarized as follows:

Proposition 2.2. *Consider an instance of Hybrid and an instance of (1,2)-ATSP constructed from it as described in § 2.3. Let π be an assignment to the variables in the Hybrid instance that satisfies all but u equations. Then the tour constructed as described above has exactly $2u$ endpoints.*

2.5 Constructing an assignment from a tour

To construct an assignment from a given TSP tour, we consider how the tour behaves on the edges of the graph defining the TSP instance. The main idea in the construction is that if the tour traverses a consistency gadget as shown in Fig. 7a the corresponding variable should be given the value 0, and if the consistency gadget is traversed as shown in Fig. 7b the corresponding variable should be given the value 1. Complications arise, of course, from the fact that an arbitrary TSP tour may enter, or leave, a consistency checker somewhere in the middle. Such traversals cannot immediately be interpreted as an assignment to the corresponding variable.

Considering the equation gadgets, the ticked “edges” in Figs. 2 and 3 are not really edges since they correspond to parts of the corresponding consistency checker. Hence a TSP tour may leave or enter a ticked “edge” in the middle—we call such edges *semitraversed*. With slight abuse of notation, we also say that an occurrence of a literal is *traversed* if both of its connecting edges in the corresponding consistency checker are traversed, *untraversed* if none of its connecting edges are traversed, and *semitraversed* otherwise.

We resolve the problem of semitraversed occurrences by performing a sequence of local transformations of the given tour. These transformations convert an arbitrary TSP tour into a TSP tour with equal or lower cost that does not contain any semitraversed occurrences. From this resulting tour, an assignment can be constructed and it can be shown that every equation that is unsatisfied under this assignment can be associated with two unique endpoints in the TSP tour.

2.5.1 Obtaining structure inside consistency checkers

In the first phase, we first make all *bridges*, i.e., all pairs of undirected edges in the consistency checkers, traversed. Knowing that all bridges are traversed by the tour then makes it possible to prove results about further transformations of the tour.

Lemma 2.2. *Consider an instance of Hybrid and an instance of (1,2)-ATSP constructed from it as described in § 2.3. In such an instance, any TSP tour can be modified into a TSP tour that traverses both bridges in every consistency checker. Moreover, this transformation can be done in polynomial time and it does not increase the length of the tour.*

Proof. For every bridge, it can be seen by considering all possibilities exhaustively that any TSP tour that traverses some set E of the four connection edges can be modified into a tour with fewer endpoints that traverses the bridge and a subset of the edges in E. The less obvious cases are shown in Fig. 10. ■

Lemma 2.3. *Consider an instance of Hybrid and an instance of (1,2)-ATSP constructed from it as described in § 2.3. In such an instance, any TSP tour that traverses both bridges in every consistency checker can be modified into a TSP tour where the consistency checkers are traversed as shown in Figs. 7, 11, 12a–d, and 13. Moreover, this transformation can be done in polynomial time and it does not increase the length of the tour.*

Proof. The assumption that the TSP tour traverses both bridges in every consistency checker implies that the consistency checkers are traversed as shown in Figs. 7, 11, 12, and 13. Without increasing the number of endpoints in the tour, we can replace the traversals shown in Figs. 12e, g and i with the traversal shown in Fig. 12a; and the ones shown in Figs. 12f, h and j with the one shown in Fig. 12b. ■

2.5.2 Removing semitraversals

The transformations described in this section have the purpose of removing all semitraversals from the TSP tour. This is performed by a two-step procedure. First, we take care of variables x for which the negative occurrence of x is semitraversed. After this procedure, the only possible remaining semitraversals are on positive occurrences of variables. An exhaustive case analysis then shows that it is possible to get rid of also those semitraversals without increasing the total number of endpoints in the graph.

Lemma 2.4. *Consider an instance of Hybrid and an instance of (1,2)-ATSP constructed from it as described in § 2.3. In such an instance, any TSP tour that traverses the consistency checkers as shown in Figs. 7, 11, 12a–d, and 13 can be* *transformed into a tour where the consistency checkers are traversed as shown in Figs. 7, 11, and 13. Moreover, this transformation can be done in polynomial time and it does not increase the length of the tour.*

Proof. We need to prove that we can get rid of traversals shown in Figs. 12a–d. To this end, consider an arbitrary consistency checker traversed as shown in Fig. 12c. The corresponding variable y occurs negatively in some equation $x + \bar{y} = 1$. We claim that it is possible to modify the tour, without increasing the total number of endpoints, in such a way that the considered consistency checker is traversed either as in Fig. 7a or as in Fig. 11a. In particular, first suppose that the gadget is traversed as shown in Fig. 14a, i.e., that none of the two edges leading to node B is traversed. Then we can remove two endpoints inside the gadget by traversing the consistency checker for $\bar{y}$ as shown in Fig. 7a. This may introduce at most two endpoints elsewhere, so the net effect is that the total number of endpoints is not increased. Secondly, if there is one traversed edge leading to node B, the equation gadget must be traversed as shown in Fig. 14b. We can then change the traversal inside the gadget so that the upper edge leaving node A in Fig. 14b is traversed instead of the lower edge. This does not change the total number of endpoints in the graph and it makes the consistency checker for $\bar{y}$ traversed as in Fig. 11a.

The procedure described above can also be used to change traversals shown in Fig. 12a into traversals shown in Figs. 7a and 11c. A very similar procedure changes traversals shown in Fig. 12d into traversals shown in Figs. 7a and 11b. and traversals shown in Fig. 12b into traversals shown in Figs. 7a and 11c. ∎

Lemma 2.5. *Consider an instance of Hybrid and an instance of (1,2)-ATSP constructed from it as described in § 2.3. In such an instance, any TSP tour that traverses the consistency checkers as shown in Figs. 7, 11, and 13 can be transformed into a tour where the consistency checkers are traversed as shown in Figs. 7* *and 11. Moreover, this transformation can be done in polynomial time and it does not increase the length of the tour.*

Proof. First note that each semitraversed occurrence contains one endpoint. By making a semitraversed occurrence traversed, one endpoint is therefore removed from the consistency checker.

Since only positive occurrences of variables can be semitraversed according to the assumptions in the lemma, the only possibility to consider for gadgets corresponding to equations of the form $x + \overline{y} = 1$ is that x is semitraversed and $\overline{y}$ untraversed. In that case, however, we can remove two endpoints from the tour by making x traversed.

Gadgets corresponding to equations of the form $x+y = 0$ can contain either two semitraversed ticked edges or one semitraversed and one untraversed ticked edge since the two ticked edges are connected as shown in Fig. 5. In the former case, we make both semitraversed edges traversed and the edge from A to B untraversed by the tour; in the latter case we make the semitraversed edge untraversed and let the tour traverse the edge from A to B. It is easy to see that the resulting tours do not have more endpoints than the original tours.

In gadgets corresponding to equations of the form $x + y + z = 0$, the tour is modified as follows: If there are three semitraversed occurrences we modify the tour so that the gadget is traversed according to Fig. 8—since every semitraversed occurrence contains one endpoint that removes at least one endpoint. If there are two semitraversed occurrences and one traversed we again modify the tour so that the gadget is traversed according to Fig. 8—that does not increase the number of endpoints. If there are two semitraversed occurrences and one untraversed we make both semitraversed occurrences traversed and modify the tour on the equation gadget so that there is a Hamiltonian path from A to B—that removes two endpoints. For the remaining case, one semitraversed edge, an exhaustive case analysis shows that by changing the traversal of the equation gadget in such a way that there is an even number of traversed edges and a Hamiltonian path from A to B, the total number of endpoints is not increased.

In gadgets corresponding to equations of the form $x + y + z = 1$, we make all semitraversed occurrences traversed and then adjust the tour on the rest of the gadget in such a way that the total number of endpoints is minimized. If there are initially three semitraversed occurrences we remove at least two endpoints. If there are initially two semitraversed occurrences and one traversed, we remove two endpoints. If there are initially two semitraversed occurrences and one untraversed, we keep the number of endpoints constant. If there is initially one semitraversed occurrence and either two traversed or two untraversed, we remove two endpoints. Finally, if there is initially one semitraversed, one traversed and one untraversed occurrence, we keep the number of endpoints constant. ∎

2.5.3 Defining the assignment

By the local transformations described in the previous two subsections, we can assume that the consistency checkers are traversed as shown in Figs. 7 and 11, i.e., there are no semitraversed occurrences. Turning to the equation gadgets, this means that each ticked edge is either traversed or untraversed; there are no semi-traversed ticked edges. If we look at each equation locally, and assume that the variables participating in the equation are given assignments according to how the corresponding ticked edge is traversed—0 for untraversed edges; 1 for traversed edges—Proposition 2.1 states that there will be at least two endpoints in equation gadgets corresponding to unsatisfied equations. Hence, if all consistency checkers were traversed as shown in Fig. 7, we could assign values to variables according to the traversal of the consistency checkers and directly attribute two endpoints to every unsatisfied equation.

However, some consistency checkers may be traversed as shown in Fig. 11. Suppose that the consistency checker corresponding to some variable x is traversed as shown in Fig. 11a and suppose that we assign the value 1 to x. In the equation where x occurs negated and in one of the two equations where x occurs positively, the corresponding ticked edges then "announce" the correct value for x. In the remaining equation, though, the ticked edge corresponding to the second positive occurrence of x looks untraversed although x has been assigned the value 1. Since the ticked edges announces that $x = 0$ although in fact $x = 1$, the number of endpoints in this equation gadget could be zero even though the equation will not be satisfied by the assignment. But there are in this case two endpoints in the consistency checker for x; these two endpoints correspond precisely to the occurrence for which the consistency checker announces the wrong assignment. Announcing a wrong assignment in the worst case makes an equation gadget "think" that an equation is satisfied although it is not, but then the two endpoints that come with this erroneous announcement can pay for this unsatisfied equation.

Lemma 2.6. *Consider an instance of Hybrid and an instance of (1,2)-ATSP constructed from it as described in § 2.3. From any TSP tour with e endpoints that traverses the consistency checkers as shown in Figs. 7 and 11 it is possible to construct an assignment to the variables in the Hybrid instance with the property that at most $\lfloor e/2 \rfloor$ equations are left unsatisfied.*

Proof. The assignment is constructed as follows: Variables whose consistency checker is traversed as shown in Figs. 7a and 11c–d are given the value 0; all other variables are given the value 1.

Consider an arbitrary equation gadget. Since all consistency checkers are traversed as shown in Figs. 7 and 11, there are no semitraversed ticked edges. Under the assumption that each variable in the considered equation is given an assignment according to the traversal of the corresponding ticked edge in the considered equation gadget—the value 0 if the ticked edge is untraversed and the value 1 otherwise—there will be at least two endpoints in the gadget if the assignment does not satisfy the equation.

Consider now an arbitrary consistency checker. If it is traversed as shown in Fig. 11, there is one equation where the ticked edge is not traversed according to the assignment defined in the first paragraph of this proof. Hence it may happen that there is no endpoint in the corresponding equation gadget although the equation is in fact not satisfied under the assignment defined above. However, each consistency checker traversed as shown in in Fig. 11 contains at least two endpoints. To sum up, there is at least two distinct endpoints for each unsatisfied equation if the assignment is defined as in the first paragraph of this proof. ■

2.6 Proof of Theorem 2.2

Given an instance of Hybrid with the properties described in Theorem 2.2, an instance of (1,2)-ATSP is constructed as described in § 2.3. By Lemma 2.1, this instance has in total $6n + m_{2,0} + m_{2,1} + 4m_{3,0} + 4m_{3,1}$ cities.

If there is an assignment to the variables in the Hybrid instance that leaves at most u equations unsatisfied, it follows from Proposition 2.2 that the tour constructed from this assignment as described in § 2.4 has length $6n + m_{2,0} + m_{2,1} + 4m_{3,0} + 4m_{3,1} + u$.

Conversely, given a TSP tour of length $6n + m_{2,0} + m_{2,1} + 4m_{3,0} + 4m_{3,1} + u$, Lemmas 2.2–2.6 show that we can construct in polynomial time an assignment to the variables in the Hybrid instance that leaves at most u equations unsatisfied by first applying the transformations described in §§ 2.5.1 and 2.5.2 and then defining the assignment as described in § 2.5.3.

3 The hardness of (1,2)-TSP

It is possible to adapt the above construction for (1,2)-ATSP to prove a lower bound also for (1,2)-TSP, yielding the following result:

Theorem 3.1. *Suppose that we are given an arbitrary instance of Hybrid with n variables, $m_{2,0}$ equations of the form $x + y = 0 \bmod 2$, $m_{2,1}$ equations of the form $x + \bar{y} = 1 \bmod 2$, $m_{3,0}$ equations of the form $x + y + z = 0 \bmod 2$, and $m_{3,1}$ equations of the form $x + y + z = 1 \bmod 2$ such that each variable occurs exactly three times, two times positively and one time negatively.*

Then it is possible to construct in polynomial time an instance of (1,2)-TSP, with size polynomial in the size of the Hybrid instance, such that

1. *If there is an assignment to the variables in the Hybrid instance that leaves at most u equations unsatisfied, then there is a TSP tour of length $16n + m_{2,0} + m_{2,1} + 3m_{3,0} + 5m_{3,1} + u$.*
2. *From any TSP tour of length $16n + m_{2,0} + m_{2,1} + 3m_{3,0} + 5m_{3,1} + u$, it is possible to construct in polynomial time an assignment to the variables in the Hybrid instance that leaves at most u equations unsatisfied.*

Corollary 3.1. *For any constant $\varepsilon > 0$, it is* ***NP****-hard to approximate (1,2)-TSP within $741/740 - \varepsilon$.*

Proof. Select $\delta > 0$ such that $(741 - \delta)/(740 + \delta) \geq 741/740 - \varepsilon$. From an instance of Hybrid with the structure described in Corollary 2.1, construct an instance of (1,2)-TSP with the properties guaranteed by Theorem 3.1. Combining Theorem 3.1 with item 2 in Corollary 2.1 shows that the constructed (1,2)-ATSP instance either has a tour of length at most $16 \cdot 42\nu + 42\nu + 18\nu + 3\nu + 5\nu + \delta\nu = (740 + \delta)\nu$ or that every TSP tour has length at least $16 \cdot 42\nu + 42\nu + 18\nu + 3\nu + 5\nu + (1 - \delta)\nu = (741 - \delta)\nu$. Furthermore, item 3 in Corollary 2.1 states that it is **NP**-hard to distinguish those two cases. Therefore it is **NP**-hard to approximate (1,2)-TSP within $(741 - \delta)/(740 + \delta) \geq 741/740 - \varepsilon$. ■

The details of the construction leading to Theorem 3.1, as well as the proof of correctness, is very similar to the construction for the asymmetric case. Therefore, we describe most of the construction on a high level, delving into details only where the argument differs from the asymmetric case.

3.1 Constructing a (1,2)-TSP instance from Hybrid

Given an instance Hybrid with n variables, $m_{2,0}$ equations of the form $x + y = 0 \bmod 2$, $m_{2,1}$ equations of the form $x + \bar{y} = 1 \bmod 2$, $m_{3,0}$ equations of the form $x + y + z = 0 \bmod 2$, and $m_{3,1}$ equations of the form $x + y + z = 1 \bmod 2$, the corresponding instance of (1,2)-TSP is constructed as described below:

The equation gadgets for equations of the form $x + y + z = \{0, 1\}$ are shown in Fig. 15; gadgets for equations of the form $x + y = 0$ and $x + \bar{y} = 1$ are shown in Fig. 16. The ticked edges in the gadgets correspond to the variables in the corresponding equation as indicated in the figures.

The ticked edges corresponding to the same variable are joined together in a consistency checker as shown in Fig. 17. The correspondence is such that negated variables always correspond to the part enclosed by a dashed curve in Fig. 17—recall that each variable occurs one times negated and two times unnegated.

As in the asymmetric case, there is no node between the two ticked edges in the gadget corresponding to equations of the form $x + y = 0$. Instead, the edge leaving the consistency checker corresponding to the first ticked edge is merged with the edge entering the consistency checker corresponding to the second ticked edge as shown in Fig. 18. Similarly, there is no node in the center of the gadget for equations of the form $x + y + z = 0$. Instead, the consistency checkers are joined as shown in Fig. 19.

The equation gadgets are hooked together in a circle in such a way that node B in each gadget is identified with node A in another gadget. With an argument similar to the proof of Lemma 2.1, it can be seen that the instance produced as described above has $16n + m_{2,0} + m_{2,1} + 3m_{3,0} + 5m_{3,1}$ cities.

3.2 Constructing a tour from an assignment

Consider an instance of Hybrid and an instance of (1,2)-TSP constructed from it as described in § 3.1. Let π be an assignment to the variables in the Hybrid instance. We now describe a TSP tour corresponding to this assignment.

Consider the tour that 1) For each variable x traverses the consistency checker corresponding to x as shown in Fig. 20a if $\pi(x) = 0$ and as shown in Fig. 20b if $\pi(x) = 1$. 2) For each equation gadget enters each equation gadget at node A, takes the shortest possible way to B under the condition that the ticked edges are traversed as prescribed by the traversals of the consistency checkers described above, and then exits the equation gadget at node B.

It can be seen by case analysis that such a tour has precisely two endpoints in each equation gadget corresponding to an unsatisfied equation and no endpoints elsewhere. (As in the asymmetric case, slight technicalities arise here since the three ticked edges in a gadget corresponding to equations of the form $x + y + z = 0$ cannot be simultaneously traversed, nor can the two ticked edges in gadgets corresponding to equations of the form $x + \overline{y} = 1$. These technicalities are resolved in the same way as in the asymmetric case.)

3.3 Constructing an assignment from a tour

As in the asymmetric case, it remains to show that *any* TSP tour with e endpoints in a (1,2)-TSP instance constructed from a Hybrid instance as described in § 3.1

can be associated with an assignment to the variables in the Hybrid instance and that this assignment satisfies all but at most $\lfloor e/2 \rfloor$ equations.

The proof of this fact follows in exactly the same way as in the asymmetric case. The only additional complication follows from that fact that some consistency checkers have two connection edges on one side due to the gadgets corresponding to equations of the form $x + y + z = 0$ (Fig. 19). However, any tour that traverses two connection edges on some consistency checker can be transformed into a tour without this property by a simple local transformation as indicated in Fig. 22. Having established this, it can be seen by a case analysis that any tour can be transformed into a tour that traverses all bridges and does not have more endpoints than the original tour in precisely the same way as indicated in the proof of Lemma 2.2 and Fig. 20. The remaining transformations described in §§ 2.5.1 and 2.5.2 can be straightforwardly adapted to the symmetric case since they only work with the connection edges of the consistency checkers. Having transformed the tour, the assignment to the variables in the Hybrid instance is defined as follows: Variables whose consistency checker is traversed as shown in Figs. 20a and 21c–d are given the value 0; all other variables are given the value 1. It can then be seen in the same way as in § 2.5.3 that this assignment has the properties required by Theorem 3.1.

References

1. Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Márió Szegedy. Proof verification and the hardness of approximation problems. *Journal of the ACM*, 45(3):501–555, May 1998.

2. Piotr Berman and Marek Karpinski. On some tighter inapproximability results. In Jiří Wiedermann, Peter van Emde Boas, and Mogens Nielsen, editors, *Proceedings of 26th International Colloquium on Automata, Languages and Programming*, volume 1644 of *Lecture Notes in Computer Science*, pages 200–209. Springer-Verlag, Prague, 11–15 July 1999.

3. Markus Bläser and Bodo Siebert. Computing cycle covers without short cycles. In *Proceedings of the 9th Annual European Symposium on Algorithms*, Lecture Notes in Computer Science. Springer-Verlag, Århus, 28–31 August 2001.

4. Hans-Joachim Böckenhauer and Sebastian Seibert. Improved lower bounds on the approximability of the traveling salesman problem. *RAIRO Theoretical Informatics and Applications*, 34(3):213–255, 2000.

5. Janka Chlebíková and Miroslav Chlebík. Approximation hardness for small occurrence instances of NP-hard problems. Technical Report TR02-073, Electronic Colloquium on Computational Complexity, December 2002.

6. Nicos Christofides. Worst-case analysis of a new heuristic for the traveling salesman problem. Technical Report CS-93-13, Graduate School of Industrial Administration, Carnegie Mellon University, Pittsburgh, 1976.

7. Lars Engebretsen. An explicit lower bound for TSP with distances one and two. *Algorithmica*, 35(4):301–319, 2003.

8. Heim Kaplan, Moshe Lewenstein, Nira Shafrir, and Maxim Sviridenko. Approximation algorithms for asymmetric TSP by decomposing directed regular multigraphs. In *44th Annual Symposium on Foundations of Computer Science*, pages 56–65. IEEE, Cambridge, Massachusetts, 11–14 October 2003.

9. Richard M. Karp. Reducibility among combinatorial problems. In Raymond E. Miller and James W. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103. Plenum Press, New York, 1972.

10. Christos H. Papadimitriou and Santosh Vempala. On the approximability of the traveling salesman problem. In *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing*, pages 126–133. Portland, Oregon, 21–23 May 2000.

11. Christos H. Papadimitriou and Mihalis Yannakakis. The traveling salesman problem with distances one and two. *Mathematics of Operations Research*, 18(1):1–11, February 1993.

12. Luca Trevisan. When Hamming meets Euclid: The approximability of geometric TSP and MST. *SIAM Journal on Computing*, 30(2):475–485, 2000.

List of Figures

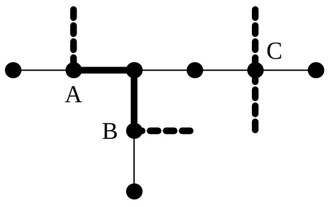
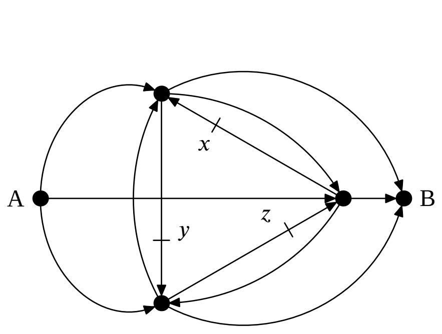


Figure 1. The above figure contains two partial tours—one entering the graph at A and leaving at B, and one both entering and leaving at C. The nodes A and B are endpoints and C is a double endpoint. The dashed parts of the tour denotes parts where the tour traverses edges with weight two.



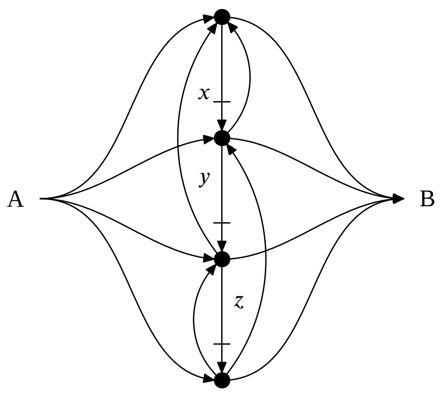


Figure 2. The gadget for equations of the form $x + y + z = 0$ (left) and $x + y + z = 1$ (right). There is a Hamiltonian path from A to B only if an even (left) or odd (right) number of the ticked edges is traversed.

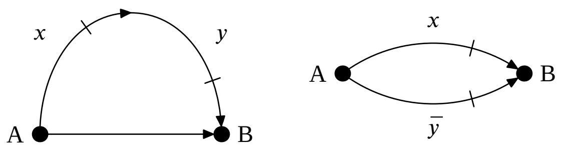


Figure 3. The gadget for equations of the form $x + y = 0$ (left) and $x + \overline{y} = 1$ (right). There is a Hamiltonian path from A to B only if an even (left) or odd (right) number of the ticked edges is traversed.

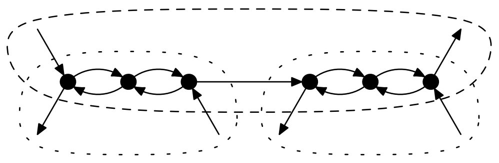


Figure 4. The gadget used to connect the ticked edges that correspond to the same variable x. The ticked edges corresponding to the two positive occurrences of x are represented by the parts enclosed in the dotted curves and the ticked edge corresponding to $\bar{x}$ is represented by the part enclosed in the dashed curve.

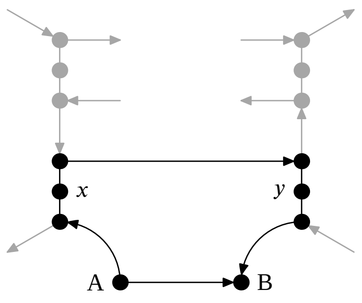


Figure 5. A more detailed view of the gadget for equations of the form $x + y = 0$. In this figure the ticked edges have been expanded to show the consistency checkers. The black edges correspond to the gadget shown in Fig. 3

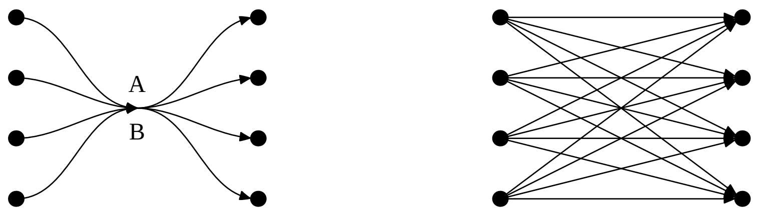


Figure 6. The cost of the gadgets for equations of the form $x + y + z = 1$ is lowered by the above transformation. The figure to the left shows the connection between two such gadgets as it is obtained by joining B in one gadget as shown in Fig. 2 with A in another such gadget. The figure to the right shows how this connection is actually implemented.

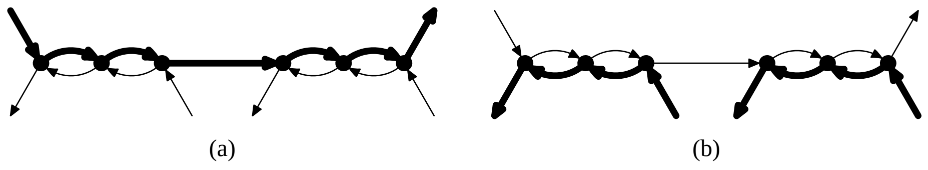


Figure 7. The figure above shows the “intended” traversals of the consistency checkers. The traversal (a) is to be interpreted as assigning 0 to the corresponding variable; traversal (b) as assigning 1.

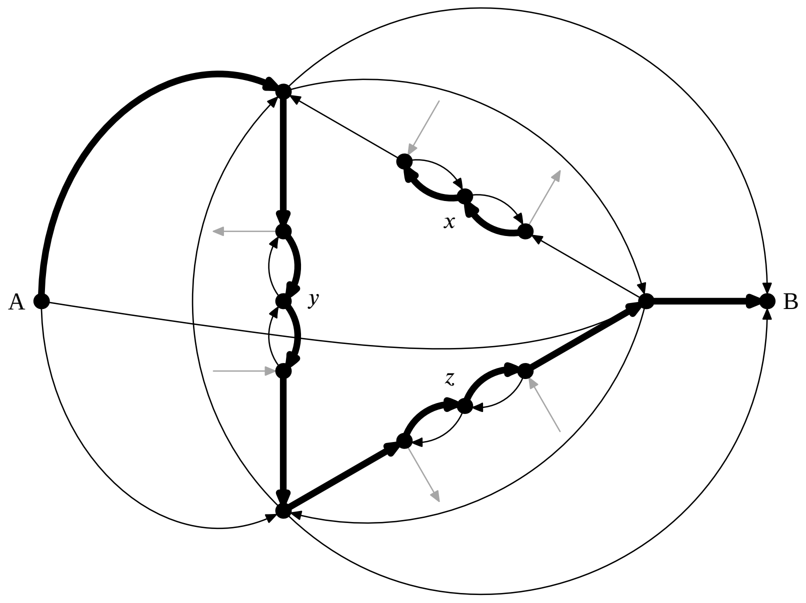


Figure 8. A more detailed view of how the tour corresponding to an assignment π such that $\pi(x) = \pi(y) = \pi(z) = 1$ traverses the gadget for equations of the form $x + y + z = 0$. In this figure the ticked edges have been expanded to show the consistency checkers. The black edges correspond to the gadget shown in Fig. 2. Note that the tour has two endpoints in the consistency checker corresponding to x.

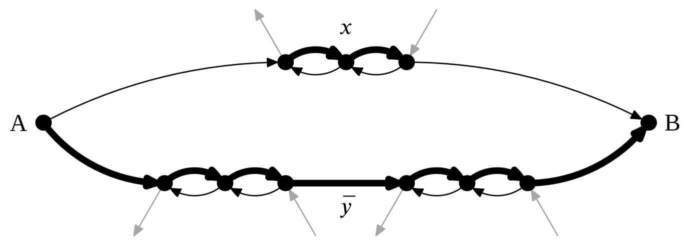


Figure 9. A more detailed view of how the tour corresponding to an assignment π such that $\pi(x) = 1$ and $\pi(y) = 0$ traverses the gadget for equations of the form $x + \overline{y} = 1$. In this figure the ticked edges have been expanded to show the consistency checkers. The black edges correspond to the gadget shown in Fig. 3. Note that the tour has two endpoints in the consistency checker corresponding to x.

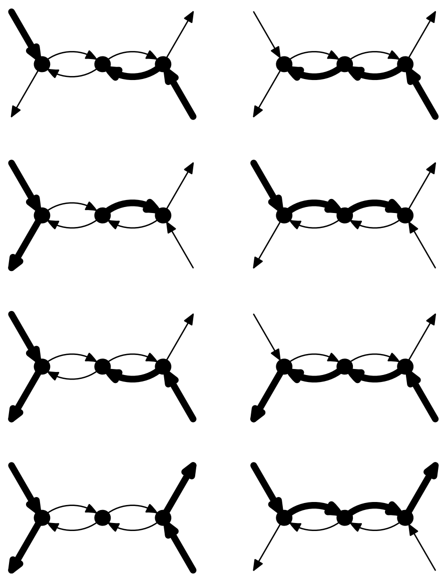


Figure 10. It is possible to change the traversals in the left column into the traversals in the right column without increasing the total number of endpoints in the graph.

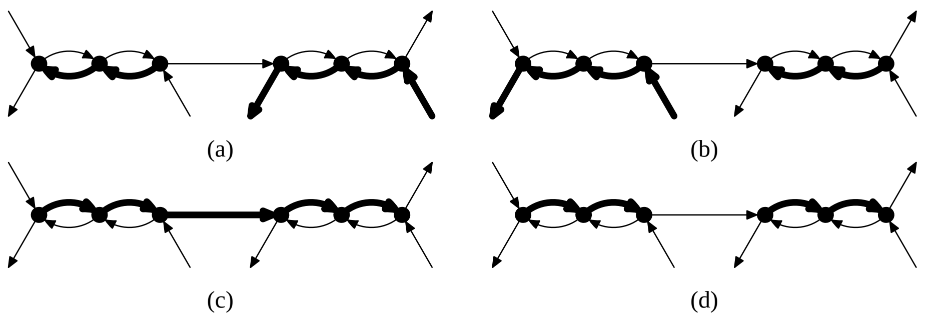


Figure 11. The traversals shown above may still be present in the tour after the "normalization" described in Lemmas 2.2–2.5.

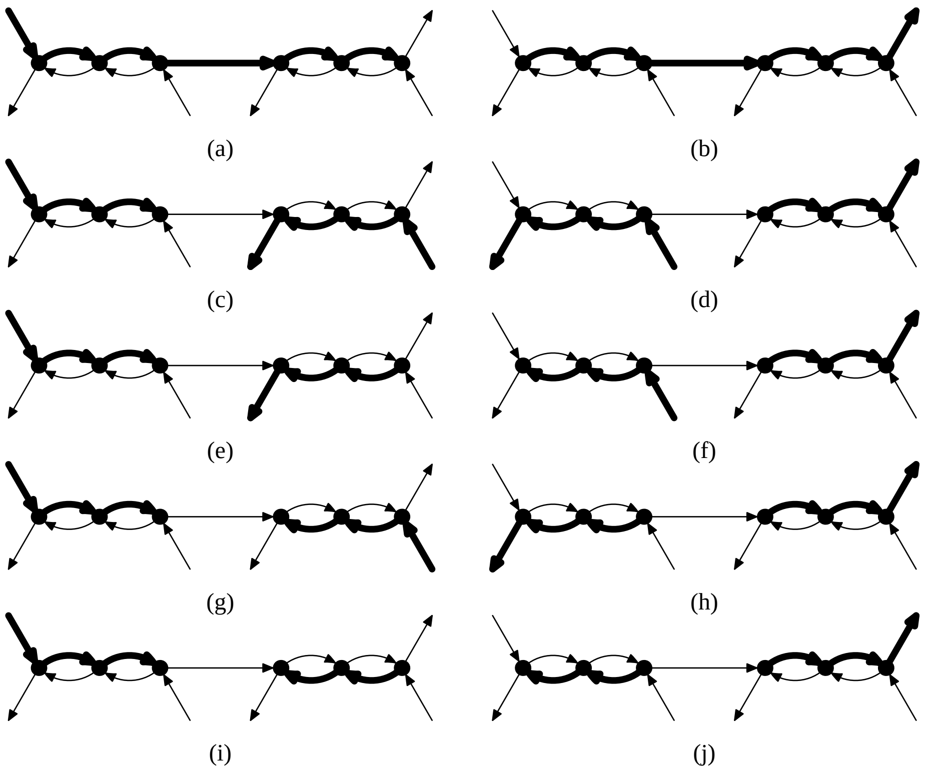

(a) (b) (c) (d) (e) (f) (g) (h) (i) (j)

Figure 12. If the negative occurrence in the consistency checker is semitraversed, the checker has to be traversed as shown above.

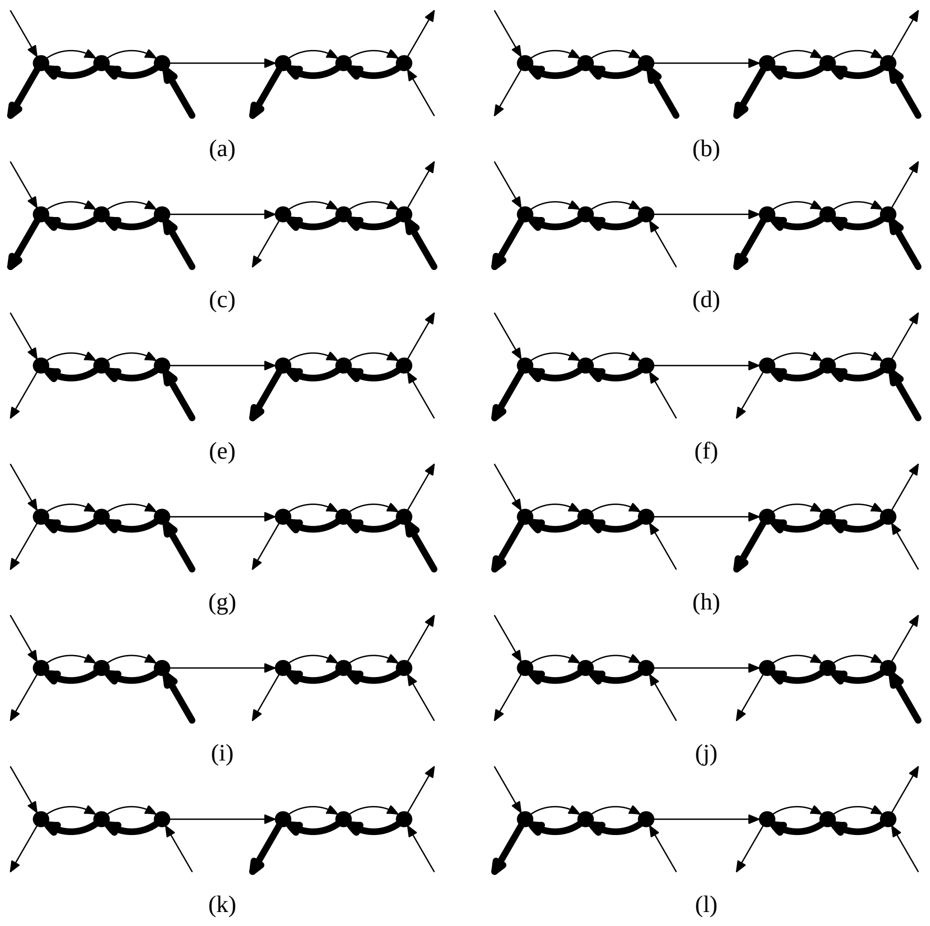
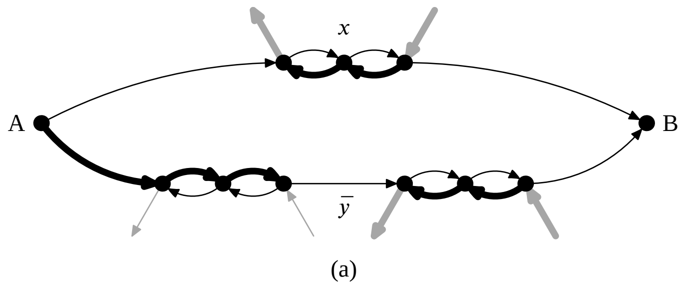


Figure 13. If there is at least one semitraversed occurrence in the consistency checker but the upper level is untraversed, the checker has to be traversed as shown above.



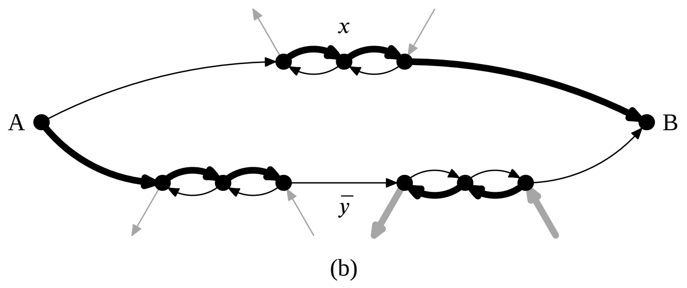
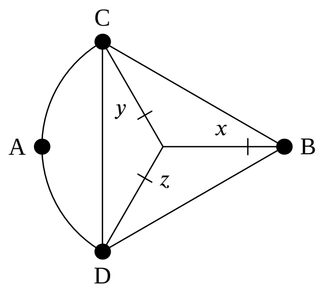


Figure 14. A gadget for equations of the form $x + \bar{y} = 1$ where the variable gadget corresponding to $\bar{y}$ is traversed as shown in Fig. 12c.



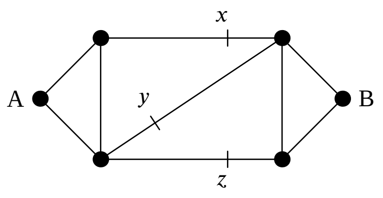


Figure 15. The gadget for equations of the form $x + y + z = 0$ (left) and $x + y + z = 1$ (right). There is a Hamiltonian path from A to B only if an even (left) or odd (right) number of ticked edges is traversed.

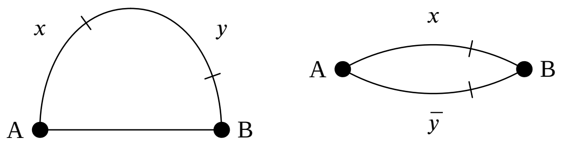


Figure 16. The gadget for equations of the form $x + y = 0$ (left) and $x + \overline{y} = 1$ (right). There is a Hamiltonian path from A to B only if an even (left) or odd (right) number of the ticked edges is traversed.

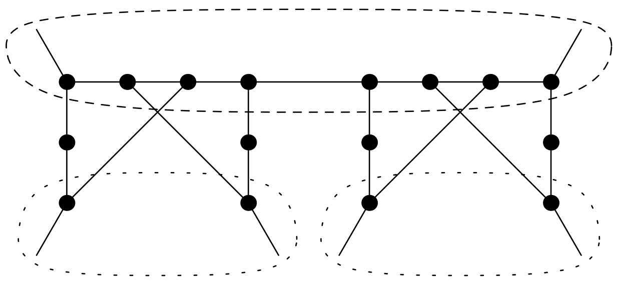


Figure 17. The gadget used to connect the ticked edges that correspond to the same variable x. The ticked edges corresponding to the two positive occurrences of x are represented by the parts enclosed in the dotted curves and the ticked edge corresponding to the negative occurrence is represented by the part enclosed in the dashed curve.

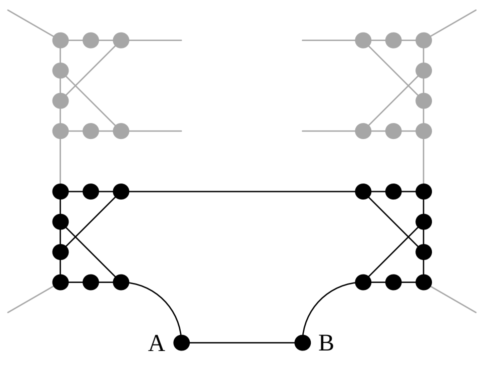


Figure 18. A more detailed view of the gadget for equations of the form $x + y = 0$. In this figure the ticked edges have been expanded to show the consistency checkers. The black edges correspond to the gadget shown in Fig. 16

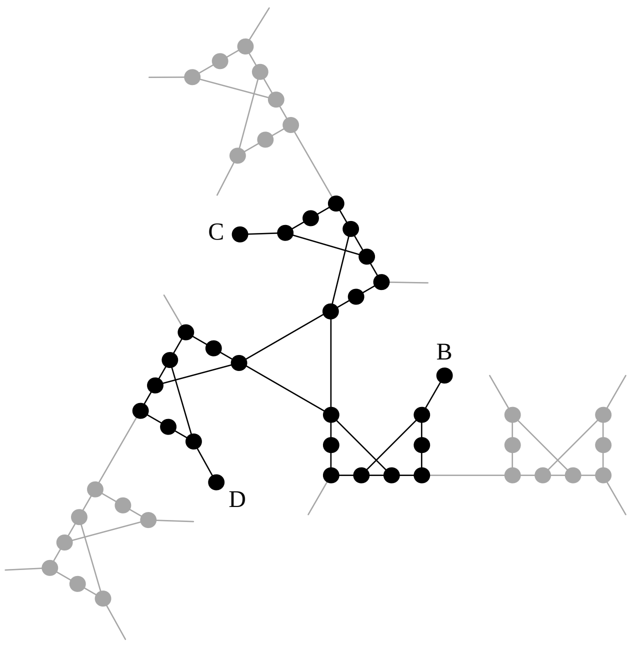


Figure 19. A more detailed view of the gadget for equations of the form $x+y+z=0$. The figure shows how the three variable gadgets meet in the center of the gadget. The black edges above correspond to the ticked edges in Fig. 15 and the three labeled nodes above are the same as the corresponding nodes in Fig. 15.

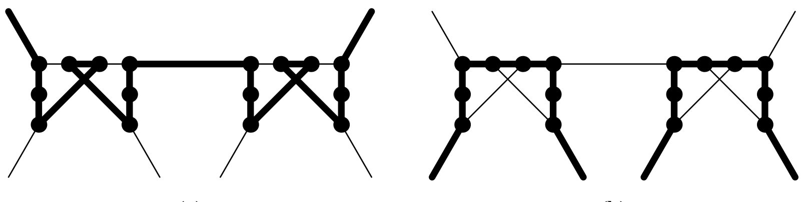


Figure 20. The figure above shows the "intended" traversals of the consistency checkers. The traversal (a) is to be interpreted as assigning 0 to the corresponding variable; traversal (b) as assigning 1.

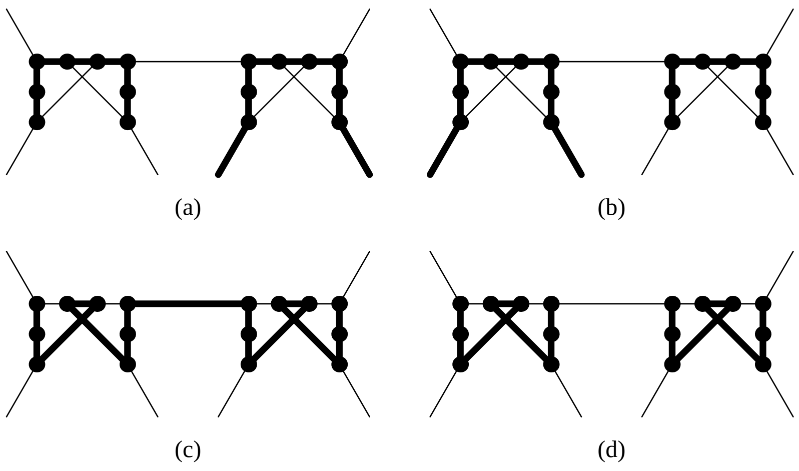


Figure 21. If there are no semitraversed occurrences in the consistency checker it may still be traversed as shown above.

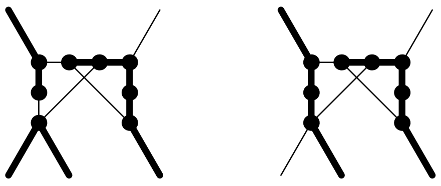


Figure 22. Some consistency checkers have double connection edges at one point, see also Fig. 19. By local transformations according to the above pattern we can assume that at most one of the double edges are traversed.

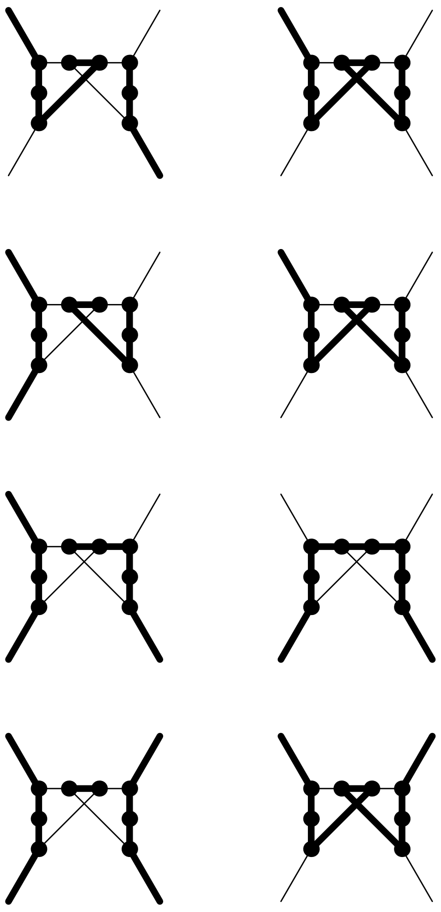


Figure 23. It is possible to change the traversals in the left column into the traversals in the right column without increasing the total number of endpoints in the graph.