Journal of Computer and System Sciences 136 (2023) 157–170

Contents lists available at ScienceDirect

Journal of Computer and System Sciences

journal homepage: www.elsevier.com/locate/jcss

From symmetry to asymmetry: Generalizing TSP approximations by parametrization

Lukas Behrendt [a], Katrin Casel [a,*], Tobias Friedrich [a], J.A. Gregor Lagodzinski [a], Alexander Löser [a], Marcus Wilhelm [b]

[a] *Hasso Plattner Institute, University of Potsdam, Germany*
[b] *Karlsruhe Institute of Technology, Germany*

A R T I C L E   I N F O

Article history:
Received 15 December 2021
Received in revised form 15 March 2023
Accepted 29 March 2023
Available online 12 April 2023

Keywords:
Parameterized approximation
Stability of approximation
TSP vs. ATSP

A B S T R A C T

We generalize the tree doubling and Christofides algorithm to parameterized approximations for ATSP (constant factor approximations that invest more runtime with respect to a chosen parameter). The parameters we consider are upper bounded by the number of *asymmetric distances*, which yields algorithms to efficiently compute good approximations for moderately asymmetric TSP instances. As generalization of the Christofides algorithm, we derive a parameterized 2.5-approximation, with the size of a vertex cover for the subgraph induced by the edges with asymmetric distances as parameter. Our generalization of tree doubling gives a parameterized 3-approximation, where the parameter is the minimum number of asymmetric distances in a minimum spanning arborescence. Further, we combine these with a notion of symmetry relaxation which allows to trade approximation guarantee for runtime. Since the parameters we consider are theoretically incomparable, we present experimental results showing that generalized tree doubling frequently outperforms generalized Christofides with respect to parameter size.

1. Introduction

The ubiquitous traveling salesman problem asks for a shortest round trip through a given set of cities. Its relation to the Hamiltonian cycle problem does not only imply NP-hardness, but also implies that efficient approximation is impossible for unrestricted instances, which is why distances are usually assumed to satisfy the triangle inequality. This restriction to *metric* instances is one of the most extensively studied problems in combinatorial optimization, yet its approximability prevails as an active research area. Despite the breakthrough by Svensson et al. [1], particularly the difference between symmetric and asymmetric distances remains rather poorly understood. In this paper we employ the tools of parameterized complexity as a new approach to explicitly study the effects of asymmetry on the approximability of the metric traveling salesman problem.

1.1. Motivation

Symmetric distance, meaning that traveling from A to B has the same cost as traveling from B to A, is certainly the most common assumption to the metric traveling salesman problem. In fact, it is so common that the name (metric) traveling

* Corresponding author.
E-mail addresses: katrin.casel@hu-berlin.de (K. Casel), tobias.friedrich@hpi.de (T. Friedrich), gregor.lagodzinski@hpi.de (J.A.G. Lagodzinski), marcus.wilhelm@kit.edu (M. Wilhelm).

https://doi.org/10.1016/j.jcss.2023.03.007
0022-0000/© 2023 Elsevier Inc. All rights reserved.

salesman problem (TSP) is usually associated with this symmetric version, while the more general case is explicitly referred to as *asymmetric* (ATSP).

It appears as if symmetry plays a vital role in view of approximations. For TSP it was known for over 40 years that a $\frac{3}{2}$-approximation is possible with the famous algorithm of Christofides [2] (or Christofides-Serdyukov, see [3]). Recently, Karlin et al. [4] showed a randomized approximation with an expected ratio of $\frac{3}{2} - \varepsilon$ for a small constant $\varepsilon > 0$. For ATSP, Svensson et al. [1] answered the longstanding open question for the existence of a constant-factor approximation in the affirmative. Although the current state of the art was very recently established by Traub and Vygen [5] with the ratio of $22+\varepsilon$, this still leaves a significant gap between the positive results for TSP and ATSP, whereas the best currently known lower bounds by Karpinski et al. [6] of $\frac{123}{122}$ for TSP and $\frac{75}{74}$ for ATSP do not indicate such a vast difference. This raises the question of how symmetry truly affects approximability.

The assumption of symmetry does not seem very natural. A study by Martínez Mori and Samaranayake [7] shows that road networks exhibit asymmetry even when only the lengths of the shortest paths are considered. Phenomena like road blocks, one-way streets and rush hours can result in unbounded violations of symmetry while the triangle inequality remains satisfied. In comparison, restricting to distances that satisfy the triangle inequality is a reasonable assumption in all scenarios where visiting cities more than once is acceptable. Finding a shortest tour that visits each city *at least* once translates to metric TSP by taking the shortest path metric, also called *metric closure*.

The *asymmetry factor* is the maximum ratio between the length of the shortest paths from A to B and B to A over all cities A, B. The investigation in [7] revealed that most asymmetries in road networks are insignificantly small. With these few but existing significant asymmetries in mind, we consider spending exponential time with respect to some measure of the degree of asymmetry. Our basic objective is to salvage the approximability of TSP for ATSP by allowing this increase in runtime. Formally, we give parameterized approximations (see e.g., [8]), which means a guaranteed performance ratio and a runtime of the form $poly(n)f(k)$, where f is an arbitrary function, n is the size of the instance and k is a measure for asymmetry. This approach aims to offer efficiency for instances of low asymmetry and to improve our understanding of the challenges asymmetric distances pose to the design of approximation algorithms.

1.2. Our results

We derive parameterized approximations based on the Christofides and the tree doubling algorithm with respective suitable parameters. Both parameters under study are bounded by the number of *asymmetric distances*, i.e., pairs of vertices (u, v) for which traveling from u to v is cheaper than traveling from v to u. Further, we combine these parameters with the asymmetry factor Δ [7] in the sense that we treat distances with asymmetry factor $\Delta \le \beta$ for some $\beta \ge 1$ as symmetric, which shrinks both parameters to consider only the more severe β*-asymmetries* (distance from v to u is at least β times the distance from u to v). In particular, we derive parameterized approximations with

- ratio $\frac{7}{4} + \frac{3}{4}\beta$ for parameter $k =$ size of a vertex cover for the subgraph induced by the β-asymmetric distances (*generalized Christofides*);
- ratio $2 + \beta$ for parameter $z =$ minimum number of β-asymmetric distances in a minimum spanning arborescence (*generalized tree doubling*).

For $\beta = 1$, we prove the ratio of 2.5 to be tight for generalized Christofides. The lack of such a tightness result and further observations lead us to conjecture that generalized tree doubling is actually a 2-approximation for $\beta = 1$. Since the two parameters k and z are theoretically incomparable, we conduct experiments which show that generalized tree doubling frequently outperforms generalized Christofides with respect to parameter size.

The paper is organized as follows. In Section 3 we generalize the Christofides algorithm. Our main result, the more elaborate generalized tree doubling algorithm, is presented in Section 4. In Section 5 we give the combination with the asymmetry factor and Section 6 describes our experimental results.

1.3. Related work

Conceptually, our approach can be seen as a study of *stability* with respect to asymmetry in the framework of *stability of approximation* by Böckenhauer et al. [9]. Probably the most extensively studied stability measure for (A)TSP is the β-triangle inequality, also called *parameterized* triangle inequality, for some $\beta > 0$, which refers to the requirement $c(u, v) \le \beta(c(u, w) + c(w, v))$ for all $u, v, w \in V$ with $u \ne v \ne w$. For ATSP with β-triangle inequality, the $\frac{1}{2(1-\beta)}$-approximation derived by Kowalik and Mucha [10] for $\beta \in (\frac{1}{2}, 1)$ improves upon a series of previous results [11–13] and is also known to be tight with respect to the cycle cover relaxation as lower bound. For TSP, the survey of Klasing and Mömke [14] gives a summary of the known results with β-triangle inequality.

Martínez Mori and Samaranayake [7] showed that the Christofides algorithm is $\frac{3}{2}$-stable with respect to the asymmetry factor, meaning that it can be used to compute a $\frac{3}{2}\Delta$-approximation for instances with asymmetry factor at most Δ.

So far, there are only a few parameterized approximations for (variations of) TSP. Marx et al. [15] consider ATSP on a restricted graph class called k-nearly-embeddable. They derive approximations where the ratio and the runtime depend on structural parameters of the given instance. A true parameterized approximation for a TSP type problem is given by Böckenhauer et al. in [16] for deadline TSP, a generalization of TSP where some cities have to be reached by the tour within a given deadline. They give a 2.5-approximation that requires exponential time only with respect to the number of cities with deadline.

Another interesting approach to invest moderate exponential time is given by Bonnet et al. in [17]. They derive a routine that allows to compute for any $r \leq n$ a $\log r$-approximation for ATSP that requires time $\mathcal{O}(2^{\frac{n}{r}} poly(n))$.

2. Preliminaries

Throughout the paper, instances of ATSP are always simple complete directed graphs denoted by $G = (V, A, c)$ with non-negative cost function c on A. For $u, v \in V$, (u, v) denotes the *arc* from u to v and $c(u, v)$ denotes its cost. For an arc $(u, v) \in A$ we call $(v, u) \in A$ the *opposite* arc. To refer to the connections between vertices without regarding any directedness, for an arc $(u, v) \in A$ and its opposite arc, we call $\{u, v\}$ an *arc-pair* or simply *link*. Links can be thought of like edges in an undirected graph. If the cost function c satisfies the triangle inequality, i.e., $c(u, v) \leq c(u, w) + c(w, v)$ for all $u, v, w \in V$, we call G *metric*. If the graph is not clear from context, we use $V[G]$ and $A[G]$ to denote the vertices and arcs of G, respectively.

In a not necessarily complete graph G', a *trail* is a sequence of vertices where each vertex is equal to or has an arc to its successor. A *path* is a trail containing no vertex twice. *Circuit* and *cycle* denote a trail and a path where the last vertex has an arc to the first vertex, respectively. We denote a trail by $v_1, \ldots, v_n$ and a circuit by $(v_1, \ldots, v_n)$. A *tour* of G' is a cycle that visits each vertex of G'.

If G is metric, every trail can be turned into a path visiting the same vertices via a *metric shortcut* without increasing the cost, where metric shortcut means removing multiple occurrences of each vertex. All tours in G are valid ATSP solutions, and we use $c^*(G)$ to denote the cost of an optimal solution for G.

For $G = (V, A, c)$ we denote the vertex-induced subgraph of $V' \subseteq V$ by $G[V']$, the arc-induced subgraph of $A' \subseteq A$ by $G[A']$ and also the link-induced subgraph of a set of links E by $G[E]$. Slightly abusing notation, $G[V']$ and $G[A']$ then also inherit the weights of G. Further, for a subgraph G' of G, we use $c(G')$ to denote the sum of all arc costs in G'. For these notions we observe the following properties.

Lemma 1. *Let G be a metric graph and $V' \subseteq V$. Then, $G[V']$ is metric as well.*

Proof. We give a proof by contradiction. Suppose that $G[V']$ is not metric. Then there exist three vertices $u, v, w \in V'$ for which $c(u, v) + c(v, w) < c(u, w)$. Since every vertex and arc of an induced subgraph exists in the original graph, the triangle u, v, w violates the triangle inequality in G. This contradicts our original assumption that G is metric. □

Lemma 2. *Let G be a metric graph and $V' \subseteq V$. Then, $c^*(G[V']) \leq c^*(G)$.*

Proof. We offer a constructive proof. Starting with an optimum solution τ for G we iteratively remove vertices $v \notin V'$ from τ until it contains only vertices from V'. Removing a vertex is equal to performing a metric shortcut and thus does not increase the cost of the whole tour. The claim follows. □

We also use one other transformation we call *minor*. Here, G' is a *minor* of G if there is a series of *contractions* which, starting from G, result in G'. A contraction of (u, v) replaces u and v with a single vertex uv and sets $c(w, uv) = \min\{c(w, u), c(w, v)\}$ and $c(uv, w) = \min\{c(u, w), c(v, w)\}$ for all $w \in V \setminus \{u, v\}$.

3. Generalized Christofides algorithm

The Christofides algorithm [2] is a polynomial approximation for TSP with performance ratio $\frac{3}{2}$. On instance G it first computes a minimum spanning tree T for G and then adds a minimum cost perfect matching M on the vertices V' of odd degree in T. The resulting subgraph is connected and each vertex has an even degree, so it is possible to compute a Eulerian cycle for it, which is a circuit of cost $c(T) + c(M)$ that visits all vertices. Metric shortcuts turn this circuit into a tour. Since taking every second edge in an optimal tour for $G[V']$ gives a perfect matching for the vertices of odd degree, the edges in M have a cost of at most $\frac{1}{2}c^*(G[V']) \leq \frac{1}{2}c^*(G)$. Together with the bound of $c^*(G)$ on the cost of T, this proves the approximation ratio of $\frac{3}{2}$.

Regarding ATSP, the most dire problem of this approach is that combining T and M to an Eulerian circuit is impossible if some arcs point in the wrong direction, and it is unclear how to restrict T and M accordingly while keeping the relation of their cost to the optimum value. Due to this conceptual problem, we use a reduction to a TSP instance for which the Christofides algorithm can be applied. Observe that brute-force guessing the correct set of asymmetries in an optimal solution does not seem like a helpful first step. It is not apparent how to efficiently compute (or approximate) a spanning tree with a fixed subset of arcs in such a way that when transforming this tree into a tour, the fixed arcs are only used in their intended direction. The design of our algorithm is instead based on a simple structural insight that allows the use of the Christofides algorithm on a symmetric subgraph.

We first explain an easier variant of the algorithm. The idea is to divide the graph into an asymmetric and a symmetric subgraph. For $G = (V, A, c)$ we define the set of *asymmetric links* by $E_a = \{\{u, v\} \mid u, v \in V, c(u, v) \neq c(v, u)\}$ and the sets of *asymmetric* and *symmetric vertices* by $V_a = \{v \in V \mid \{u, v\} \in E_a$ for some $u \in V\}$, and $V_s = V \setminus V_a$, respectively.

We define the asymmetric subgraph by $G[V_a \cup \{v\}]$, where v is an arbitrary but fixed vertex in V_s, and the symmetric one by $G[V_s]$. Note that tours through both subgraphs can be merged at the overlap in v and turned into a tour of the whole graph with metric shortcuts. Combining in this way an exact solution for $G[V_a \cup \{v\}]$ and a $\frac{3}{2}$-approximate solution for $G[V_s]$, computed by the Christofides algorithm, overall yields a parameterized $\frac{5}{2}$-approximation with parameter $|V_a|$.

To improve this, consider a vertex cover VC of $G[E_a]$. The complement of VC forms an independent set in $G[E_a]$, implying that G contains no asymmetric links between vertices in $V_s \cup (V_a \setminus VC)$. This can be exploited to consider the smaller structural parameter z, the size of a vertex cover in $G[E_a]$ and gives the following.

Theorem 3. *Metric ATSP can be $\frac{5}{2}$-approximated in $\mathcal{O}(n^3 + 2^z z^2)$ where z is the size of a minimum vertex cover of the subgraph induced by all asymmetric links.*

Proof. The approximation algorithm to prove the claimed result performs the following steps:

1. Compute a minimum vertex cover VC for $G[E_a]$ by some simple parameterized algorithm in $\mathcal{O}(|E_a| + 2^z z^2)$ (e.g. branching on the k^2-kernel as discussed in the introduction of [18] for "Bar Fight Prevention").
2. Pick an arbitrary vertex $v \in V \setminus VC$ and use the algorithm by Held and Karp [19] for ATSP on the subgraph $G' = G[VC \cup \{v\}]$ which computes an optimal solution τ' for this subgraph in $\mathcal{O}(2^z z^2)$.
3. Compute a $\frac{3}{2}$-approximate tour τ'' for $G'' = G[V[G] \setminus VC]$ with the Christofides algorithm.
4. Consider τ' by τ'' as vertex sequences each starting with v. Create a circuit that visits each vertex in G by concatenation of the sequences for τ' and τ''. Delete from the resulting circuit one occurrence of v to create a proper tour τ.

In step 3, the Christofides algorithm is used to compute an approximate tour on a subgraph. This is possible because the subgraph G'' consists of the complement of the vertex cover for the asymmetric links. Hence, the Christofides algorithm receives a symmetric subgraph as input. This yields τ'', a $\frac{3}{2}$-approximation for G'', and by Lemma 2 we obtain $c(\tau'') \leq \frac{3}{2} \cdot c^*(G)$.

In step 4 we make use of the fact that v appears both in τ' and τ'' to combine the two tours into a single circuit. This circuit uses every arc in τ' and τ'' exactly once, and visits every vertex visited by τ' or τ''.

Since τ' and τ'' span all vertices of G, the metric shortcut (the deletion of one occurrence of v in the tuple) to create τ yields a tour of G with cost at most

$$c(\tau) \leq c(\tau') + c(\tau'') \leq \left(1 + \frac{3}{2}\right) \cdot c^*(G). \tag{1}$$

Hence the claimed approximation ratio of $\frac{5}{2}$ follows. Lastly, the running time for the first two steps is dominated by computing τ' in $\mathcal{O}(2^z z^2)$, and the last steps are dominated by the min cost perfect matching for the Christofides algorithm, which can be estimated by $\mathcal{O}(n^3)$ [20]. □

Instead of exact algorithms for the vertex cover for $G[E_a]$ and the solution on $G[VC \cup \{v\}]$, we can also use approximations. A 2-approximation for vertex cover and the $\frac{2}{3}\log n$-approximation of Feige and Singh [21] on $G[VC \cup \{v\}]$ yields the following interesting result.

Corollary 4. *Metric ATSP can be $(\frac{2}{3}\log x + \frac{3}{2})$-approximated in polynomial time, where $x = \min(2z + 1, |V_a|)$, V_a is the set of asymmetric vertices and z is the size of a minimum vertex cover for the subgraph induced by all asymmetric links.*

This improves upon the approximation ratio of $\frac{2}{3}\log n$ if $\frac{x}{n} < 2^{-\frac{9}{4}}$, meaning that $G[VC \cup \{v\}]$ only contains a sufficiently small fraction of the vertices. We note that the result of Asadapour et al. [22] gives a polynomial $(8\log(z)/\log\log(z) + \frac{3}{2})$-approximation, which is asymptotically stronger but less suitable for the instances with small values of z we are interested in. Also, note that plugging in constant-factor approximations for ATSP (like the ones from Svensson et al. [1] or Traub and Vygen [5]) is of course possible, but theoretical analysis then only yields worse approximation ratios (specifically $+\frac{3}{2}$ to their respective ratios).

Further, note that one can also use any approximation for TSP (not just the Christofides algorithm) for the symmetric subgraph and obtain an $(\alpha + 1)$-approximation for ATSP from any α-approximation for TSP.

It remains to see if this approach can be improved. Aiming for a smaller parameter seems difficult as this would not split off a symmetric subgraph. Regarding a possible improvement of the ratio, one might hope to salvage the ratio of $\frac{3}{2}$ for TSP, obtained by the Christofides algorithm, for ATSP. However, such an improvement requires a different algorithmic strategy as

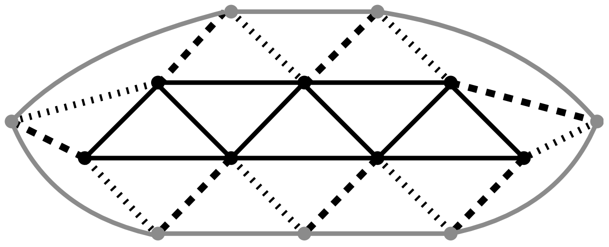

Fig. 1. G_k for $k = 7$: Black and gray links are symmetric with cost 2, dotted links are symmetric with cost 1. Dashed links are asymmetric, with cost 1 from gray to black vertex and cost 2 from black to gray vertex.

we can show that the ratio in Theorem 3 is asymptotically tight by giving a family of instances G_k for $k \in \mathbb{N}$, $k > 2$ such that the approximation ratio converges to 2.5 for increasing k. For an intuition of this construction, consider the instance described in Fig. 1. The black zig-zag pattern is the textbook example for the tightness of the Christofides algorithm. The idea is that the gray vertices build the minimum vertex cover such that the black zig-zag pattern becomes the symmetric instance. The gray cycle is then the asymmetric subgraph and solving it exactly yields a tour of cost $2k$. Together with the approximation on the symmetric subgraph, which converges to $3k$, this results in a tour of length $5k$, while the optimal tour takes the dotted and dashed links in the cheaper direction and has cost $2k$.

To formally define this family of instances, it is more convenient to describe the indicated non-complete graph and show that it can be turned into a complete graph with metric distances (i.e., a metric ATSP instance). To this end, we first define a generalized form of the triangle inequality for incomplete graphs, which we call *polygon inequality*. We then show that every graph which satisfies the polygon inequality can be transformed into a complete graph which satisfies the triangle inequality without modifying the existing arc costs.

Definition 5. In a directed graph $G = (V, A, c)$ with non-negative arc costs c, the polygon inequality holds if and only if for every arc $(u, v) \in A$ any path P between any two vertices $u, v \in V$ is at least as expensive as $c(u, v)$:

$$\forall v_1, \dots, v_j \in v \texttt{ with } (v_i, v_{i+1}) \in A, 1 \le i < j : \sum_{i=1}^{j-1} c(v_i, v_{i+1}) \ge c(v_1, v_j)$$

Note that in complete graphs, the polygon inequality and the triangle inequality are equivalent. We now show that any strongly connected graph where the polygon inequality holds, can be turned into a complete graph without changing the original arc costs.

Lemma 6. *Let G be a directed, strongly connected graph $G = (V, A, c)$ with non-negative arc costs c and with polygon inequality. Then, G can be turned into a metric ATSP instance $G' = (V, A', c')$ with $c(u, v) = c'(u, v)$ for all $(u, v) \in A$.*

Proof. Since the polygon inequality implies the triangle inequality in complete graphs, it suffices to show that we can make the graph complete without violating the polygon inequality. To do so, missing arcs are iteratively inserted and assigned the cost of the cheapest path between the connected vertices that existed in the graph prior to the insertion. We show that this never violates the polygon inequality by induction over the number of added arcs.

As the base case, we have $G' = G$ and know that the polygon inequality holds in G. For the induction step, we choose two arbitrary vertices u, v that are not yet connected by an arc (u, v) and compute a cheapest path P_{uv} from u to v. Since G is strongly connected, such a path P_{uv} always exists. We insert the arc (u, v) into G' with $c'(u, v)$ set to the cost of P_{uv} and show that this does not violate the polygon inequality. For the insertion to violate the inequality, one of two cases would have to apply:

1. There exists a path from u to v which has a cost less than (u, v). This cannot occur, as the cost of (u, v) was chosen equal to the cost of the cheapest path from u to v.
2. (u, v) is part of a path P_{xy} between two vertices x and y that has a lower cost than $c'(x, y)$.
   We show that this cannot occur as well. Already before the insertion of (u, v) there existed a path P'_{xy} from x to y, which can be obtained from P_{xy} by replacing (u, v) with a cheapest path from u to v and skipping loops. Since we chose $c'(u, v)$ to be the cost of a cheapest path from u to v, it follows that $c'(P'_{xy}) \le c'(P_{xy})$. By induction hypothesis, the polygon inequality holds for any path not containing (u, v), and thus the cost of P'_{xy} is at least $c'(x, y)$.

In both cases, the polygon inequality still holds after inserting the new arc. The claim follows.

Note that every cheapest path in G is still a cheapest path in G'. The reason is that for every new arc (u, v), G already contained a path $u, \dots, v$ with equal cost. □

With these tools, we can now nicely define our worst-case instance for Theorem 3.

Theorem 7. *There exists an infinite family of metric ATSP instances G_k, $k \geq 2$ with optimum value $2k$ such that the generalized Christofides algorithm produces a solution of cost $5k-3$.*

Proof. We construct a family of ATSP instances G_k, $k \geq 2$ as follows.

G_k consists of k vertices $g_1, \dots, g_k$ (which we call *gray vertices*) and k vertices $b_1, \dots, b_k$ (which we call *black vertices*). The gray vertices form a cycle $(g_1, \dots, g_k)$, which we denote by *gray cycle*. The black vertices also form a cycle $(b_1, \dots, b_k)$, which we denote by *black cycle*. All links in the gray and black cycles are symmetric and have a cost of 2. Furthermore, the black cycle has a series of additional internal arcs. These form a path $b_1, b_k, b_2, b_{k-1}, \dots, b_{\lceil k/2 \rceil}$, which alternates between vertices at the start and at the end of the cycle, and ends in a vertex in the middle. All links on this path are symmetric and have a cost of 2. Note that this path thus forms a minimum spanning tree of the subgraph induced by the black vertices, and that the start and end vertices of the path are as distant as possible: this constitutes a worst case instance for the Christofides algorithm.

In order to allow an optimum tour to alternate between the gray and black cycle, the gray cycle is connected to the black cycle with a number of asymmetric links: for $1 \leq i \leq k$ there is an arc (b_i, r_i) with cost 2 and an opposite arc with cost 1. Additionally, b_i is connected to r_{i+1} by a symmetric link of cost 1 for $1 \leq i \leq k$.

Then, we use Lemma 6 to turn $G_k = (V_k, A_k, c_k)$ into a complete graph. To this end, we have to show that the polygon inequality holds in G_k. Let P_{uv} be an arbitrary path from u to v with $u, v \in V_k$. There are two cases to consider:

1. P_{uv} consists of a single arc. In this case the path consists only of the arc (u, v). Thus, $c_k(P_{uv}) = c_k(u, v)$, which satisfies the polygon inequality.
2. P_{uv} consists of at least two arcs. The cheapest arc in G_k has cost 1, thus $c_k(P_{uv}) \geq 2$. The most expensive arc in G_k has cost 2, so $2 \geq c_k(u, v)$. As a result, $c_k(P_{uv}) \geq c_k(u, v)$, which satisfies the polygon inequality.

We use $G'_{=}(V'_k, A'_k, c'_k)$ to denote the metric ATSP instance created from G_k by Lemma 6 and observe the following properties of this instance. We claim that the cycle $(g_1, b_1, \dots, g_k, b_k)$, which alternates between the gray and black vertices, is an optimum tour for G'_k. Its cost is $2k$, as it contains only arcs with cost 1. Since there are no arcs with cost less than 1, the tour is an optimal solution.

The first step of the generalized Christofides algorithm is to find a minimum vertex cover on the graph induced by the asymmetric links. We claim that such a cover is given by the set of gray vertices which can be seen as follows.

1. The gray vertices form a vertex cover. This is equivalent to stating that the black vertices form an independent set, thus there are no asymmetric links between any of the black vertices. For this, recall that all links between black vertices are symmetric in G_k. Further, as can be seen in Fig. 1, for every pair of black vertices u, v there always is a cheapest path $P^*_{u,v} = u, \dots, v$ that uses only black vertices, and thus $c'_k(P^*_{u,v})$ equals $c'_k(P^*_{v,u})$. As a result, all links between black vertices in G'_k must be symmetric as well.
2. There exists no smaller vertex cover. Already in the underlying subgraph G_k, there is an asymmetric link between every pair of vertices b_i and r_i. In order to cover those edges in the asymmetric subgraph, at least one vertex per edge needs to be taken into the vertex cover, therefore there cannot be a vertex cover with fewer than k vertices.

So let this set of gray vertices be the vertex cover chosen by the generalized Christofides algorithm, and let w.l.o.g. b_1 be the vertex v chosen to be added to this set. Generalized Christofides then computes an exact solution for the asymmetric subgraph (induced by the vertices $g_1, \dots, g_k, b_1$), and uses the Christofides algorithm on the symmetric subgraph (induced by the black vertices, interpreted as undirected graph).

There is an optimum solution for the asymmetric subgraph which traverses almost the whole gray cycle once, only leaving once to pick up b_1. The cost of this solution is $2k$.

For Christofides, we assume an unlucky choice of the MST, namely the path we added to the black cycle in G_k. The path has a cost of $2k-2$. Since we need $k-1$ edges for a spanning tree and each edge between black vertices has at least cost 2, we know that our path is really an MST. The MST has only two vertices of odd degree: start and end vertex of the path, which have the biggest possible distance to each other (which is $\lfloor \frac{k}{2} \rfloor$). As a result, the edge chosen in the matching step has cost $2\lfloor \frac{k}{2} \rfloor$, which is at least $k-1$. The total cost of the solution returned by the Christofides algorithm is thus at least $2k-2+k-1$. Overall, this leads to a cost of at least $5k-3$ for the cost of the combined tour; note that metric shortcuts after combining the tours at b_1 do not have an effect on the cost. □

4. Generalized tree doubling algorithm

One other widely known approximation for TSP is the tree doubling algorithm. It computes a minimum spanning tree (MST) and doubles every edge in it to ensure the existence of an Eulerian circuit. Since the circuit uses every MST edge exactly twice, it is twice as expensive as the tree, which itself is at most as expensive as the optimum tour. Thus, transforming the circuit with metric shortcuts gives a 2-approximation. To adapt this approach to ATSP we use a *minimum spanning* *arborescence* (MSA) as the directed variant of an MST. (The notion *spanning arborescence* in this context denotes a rooted tree in a directed graph such that for each vertex v there is a unique path in the tree from the root to v. In particular, like a tree, an arborescence containing n vertices has exactly $n-1$ arcs.) Tree doubling then runs into trouble when the cost of an opposite arc is arbitrarily higher than the direction contained in the MSA. These arcs are the core of the problem and hence our basis to generalize the tree doubling algorithm.

Formally, we call $(u, v) \in A$ a *one-way arc* in $G = (V, A, c)$ if $c(u, v) < c(v, u)$. In a nutshell, our algorithm removes all one-way arcs from an MSA, computes a tour for each resulting connected component by an altered tree doubling routine and uses exponential time in the number of removed one-way arcs to connect these subtours to a solution for the whole graph. For a best runtime, we hence want to keep the number of one-way arcs in the starting MSA as small as possible. For our parametrization, we formally define k to be the minimum number of one-way arcs in an MSA for G.

At first glance, it might seem that finding an MSA with k one-way arcs is a difficult task. However this can be accomplished by searching for an MSA with an altered weight function and gives the following result.

Lemma 8. *Let G be a metric ATSP instance, then an MSA of G with a minimum number of one-way arcs can be computed in $\mathcal{O}(n^3)$.*

Proof. For $G = (V, A, c)$ let $A_o \subseteq A$ be the set of one-way arcs in G. Assume w.l.o.g. that c assigns integer weights (otherwise scaling transforms the weights into integers without altering the property of a spanning arborescence being an MSA). We define a new ATSP instance $G' = (V, A, c')$ by altering the weights in G as follows:

$$c'(e) = \begin{cases} |V|c(e) + 1 & \text{if } e \in A_o \\ |V|c(e) & \text{else} \end{cases}$$

Let $r \in V$ be any chosen root vertex. We claim that an MSA with root r for G' is an MSA with root r in G with a minimum number of one-way arcs (minimum among all MSAs with root r for G). Let L be an MSA with root r of G'.

Assume that L is not an MSA with root r of G, so there exists a spanning arborescence T with root r of G such that $c(L) > c(T)$. With the assumed integer weights, it follows that $c(L) \geq c(T) + 1$. By the definition of c', and the fact that each spanning arborescence of G contains at most $|V| - 1$ one-way arcs, it follows that $c'(T) \leq |V|c(T) + |V| - 1 \leq |V|(c(L) - 1) + |V| - 1 < c'(L)$, a contradiction to L being an MSA with root r for G'.

Hence L is an MSA with root r for G. Now let k be the number of one-way arcs in L, and assume that this is not the minimum among all MSAs with root r for G. Then there exists an MSA T for G with root r and with $k' < k$ one-way arcs. This gives $c'(T) = |V|c(T) + k' < |V|c(T) + k = c'(L)$ (observe that $c(L) = c(T)$ since these are both MSA for G), again a contradiction to the minimality of L for G'.

Lastly, an MSA for G' with fixed root r can be computed in $\mathcal{O}(n^2)$ (observe that we have a complete graph, so $|A| = |V|^2$), e.g., with the Chu–Liu/Edmonds algorithm [23] with Fibonacci heaps [24]. We do not care about a specific vertex r to be the root, actually we would rather pick it to minimize the number of one-way arcs, so we try all vertices in V as root and pick the one that minimizes the number of one-way arcs in the resulting MSA. This gives the total runtime in $O(n^3)$. □

With this best MSA, we can describe our generalized tree doubling algorithm. Let T be the MSA for G computed with Lemma 8, and let $T_1, \ldots, T_{k+1}$ be the connected components in the graph created by deleting all k one-way arcs from T. We construct a graph M by contracting each set of vertices $V[T_i]$ to one vertex v_i^M with our notion of contraction to a minor. This results in $V[M] = \{v_1^M, \ldots, v_{k+1}^M\}$ and for all $v_i^M, v_j^M \in V[M]$ with $i \neq j$, $c(v_i^M, v_j^M) = \min\left(\{c(t_i, t_j) \mid t_i \in V[T_i], t_j \in V[T_j]\}\right)$.

Lemma 9. *Let G be a metric ATSP instance and M be a minor of G, then $c^*(M) \leq c^*(G)$.*

Proof. Consider an arbitrary sequence of contractions of G that result in M. Map each vertex $v \in M$ to the subset of vertices $S_v \subseteq V[G]$ that v was contracted from. Let τ^* be an optimal tour for G. Arbitrarily pick for each such set S_v exactly one vertex r_v. Create a tour τ' from τ^* by shortcutting all vertices that were not picked to be some r_v. Note that since G is metric, τ' is at most as expensive as τ^*. Finally, let τ be the tour obtained from τ' renaming r_v to v, to formally obtain a tour in M.

Consider an arbitrary arc (x, y) in τ' and let $x = v_v$ and $y = v_w$. It follows by the construction of M that the arc from the vertex created from S_v to the vertex created from S_y in M has the cost $\min\{c(a, b) \mid a \in S_v, b \in S_w\} \leq c(x, y)$. Thus every arc in τ costs at most as much as its corresponding arc in τ', and we conclude that $c(\tau) \leq c(\tau') \leq c(\tau^*) = c^*(G)$. □

The idea for our generalized tree-doubling algorithm now is to extend an optimal tour τ' for M to a tour for G. Consider a vertex v_i^M in M (which corresponds to the component T_i) and assume w.l.o.g. that in τ' it is preceded by v_{i-1}^M and precedes v_{i+1}^M. Further, let $(v_{out}^{T_{i-1}}, v_{in}^{T_i})$ and $(v_{out}^{T_i}, v_{in}^{T_{i+1}})$ be the cheapest arc between T_{i-1} and T_i, and T_i and T_{i+1}, respectively. The goal is to find a path χ_i that starts in $v_{in}^{T_i}$, ends in $v_{out}^{T_i}$, and spans all vertices in T_i (formally a solution to

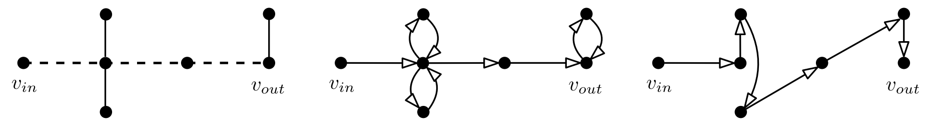

Fig. 2. Exemplary construction for a suitable path χ_i. Left: spanning tree with P_i dashed; Middle: trail through partially doubled edges; Right: resulting path.

s-t-path TSP for $G[T_i]$ with $s = v_{in}^{T_i}$ and $t = v_{out}^{T_i}$). Replacing v^{T_i} in τ' by χ_i for each i turns τ' into a tour. However, the cost of χ_i has to be bounded.

Such a path χ_i through T_i can be found by adapting the tree doubling algorithm as given in Algorithm 1. For this algorithm, we treat T_i as undirected and double all its edges that are not on the shortest path P_i from $v_{in}^{T_i}$ to $v_{out}^{T_i}$. The resulting graph contains an Eulerian trail from $v_{in}^{T_i}$ to $v_{out}^{T_i}$, which is turned into a path by metric shortcuts ensuring that $v_{in}^{T_i}$ and $v_{out}^{T_i}$ remain start and end vertex, see Fig. 2 for an example. Observe that we cannot use any of the better approximations for s-t-path TSP, such as [25], since the subgraph induced by the vertices in T_i is not necessarily completely symmetric. Further, even if this was possible, the only information we can use to compare the tour through T_i with the optimum for the whole graph G are the arcs from the MSA, which in the worst case always results in a ratio of 2.

Algorithm 1: Adjusted tree doubling for a spanning path of component T_i.

```
   Input: Symmetric tree $T_i$, vertices $v_{in}^{T_i}, v_{out}^{T_i} \in T_i$
   Output: A spanning path
1  $P_i \leftarrow$ the unique path from $v_{in}^{T_i}$ to $v_{out}^{T_i}$ in $T_i$
2  $T_i^M := T_i$
3  foreach edge $e$ in $T_i$ do
4      if $e \notin P_i$ then
5          double $e$ in $T_i^M$
6  $\pi_i \leftarrow$ an Eulerian trail from $v_{in}^{T_i}$ to $v_{out}^{T_i}$ in $T_i^M$
7  $\chi_i \leftarrow$ the metric shortcut of $\pi_i$
8  return $\chi_i$
```

For the cost of χ_i, note that it contains for each arc (u, v) in T_i at most both (u, v) and (v, u). Since there are no one-way arcs in T_i, any opposite arc is at most as expensive as the original arc in T_i. Consequently, the cost of χ_i is at most twice the cost of the arcs in T_i and the sum of all χ_i is at most $2c^*(G)$. In combination with the cost of at most $c^*(G)$ for τ', this yields the following result.

Theorem 10. *Metric ATSP can be 3-approximated in $\mathcal{O}(2^k \cdot k^2 + n^3)$, where k is the minimum number of one-way arcs in a minimum spanning arborescence.*

Proof. Consider the procedure described in Algorithm 2. We claim that first computing an MSA L with Lemma 8 and then

Algorithm 2: Generalized tree doubling algorithm.

```
1  Input: MSA $L = (V, A)$
2  $T_1, \dots, T_{k+1} \leftarrow$ components of $L[\{e \in A \mid e$ is not a one-way arc$\}]$
3  $M \leftarrow$ complete graph with vertices $v_1^M, \dots, v_{k+1}^M$
4  foreach $v_i^M, v_j^M \in V[M]$ with $i \neq j$ do
5      $c(v_i^M, v_j^M) \leftarrow \min\left(\{c(t_i, t_j) \mid t_i \in V[T_i], t_j \in V[T_j]\}\right)$
6  $\tau' \leftarrow$ an optimum tour of $M$
7  foreach subsequent $v_i^M, v_j^M \in \tau'$ do
8      $v_{out}^{T_i}, v_{in}^{T_j} \leftarrow \mathrm{argmin}\{(c(t_i, t_j)) \mid t_i \in V[T_i], t_j \in V[T_i]\}$
       /* Arc $(v_{out}^{T_i}, v_{in}^{T_j})$ thus has cost $c(v_i^M, v_j^M)$ */
9  foreach $i \in \{1, \dots, k+1\}$ do
10     $\chi_i \leftarrow$ path from $v_{in}^{T_i}$ to $v_{out}^{T_i}$ spanning $T_i$ computed by Algorithm 1
11 $\tau \leftarrow$ tour created from $\tau'$ by replacing $v_i^M$ by $\chi_i$ for each $i$
12 return $\tau$
```

running Algorithm 2 for this L yields a tour with the desired properties. Consider the computed solution $\tau = (\chi_1, \dots, \chi_{k+1})$. Since τ contains a path of each of the $k + 1$ components, τ visits each vertex exactly once and is thus a tour of G. In total, τ consists of the paths through each component in addition to the arcs $(v_{out}^{T_i}, v_{out}^{T_{i+1}})$ for $i \in [1, k + 1]$, which correspond to the arcs in an optimal solution for M. Since the subtours χ_i through the components T_i are computed by the adapted tree doubling algorithm, they satisfy the following inequality

$$\sum_{i=1}^{k+1} c(\chi_i) \le \sum_{i=1}^{k+1} 2 \cdot c(T_i) \le 2c^*(G). \tag{2}$$

Together with Lemma 9 it follows that the constructed tour is at most 3 times as expensive as the optimum tour for G.

The runtime of the algorithm is dominated by step 6 of Algorithm 2, where we use the dynamic programming algorithm by Held and Karp [19] that runs in $\mathcal{O}(2^k k^2)$. The runtime of all remaining steps is polynomial in n, where the most expensive part is the cubic runtime of Lemma 8. The overall runtime of the algorithm is therefore in $\mathcal{O}(n^3 + 2^k k^2)$. □

Contrary to the approach in Section 3, we cannot plug in some approximation for finding a good tour τ' for M to derive something like Corollary 4. Note that the minor M is not necessarily metric since contractions do not preserve the triangle inequality. Still, one might ask if M, as minor of a metric graph, has useful structural properties. However, the following result discourages such ideas.

Lemma 11. *Let G be a complete, directed graph with cost function c. Then, there exists a complete, metric graph $\hat{G}$ of which G is a minor.*

Proof. Let $G = (V, A, c)$ be a complete, directed graph. We construct a metric graph $\hat{G}$ as follows. For each vertex pair $\{u, v\} \in V \times V$ with $u \neq v$, we add the vertices u_v and v_u in $\hat{G}$. We create the arcs (u_v, v_u) and (v_u, u_v), and assign them the costs $c(u, v)$ and $c(v, u)$, respectively. Finally, we connect every vertex pair that is not yet connected with arcs in both directions with cost $m + 1$, where m is the maximum arc cost in G. Every vertex in $\hat{G}$ is incident to exactly one arc whose cost is not $m + 1$. Therefore, every triangle in $\hat{G}$ consists either of three arcs with cost $m + 1$, or of two arcs with cost $m + 1$ and one arc with cost less than $m + 1$. For all these triangles, the triangle inequality holds.

Every vertex u in G is represented by a set of vertices V_u in $\hat{G}$: $V_u = \{u_v \mid v \in V[G], v \neq u\}$. Similar to a vertex u having exactly one arc to every vertex v in G, a vertex set V_u has exactly one arc with cost $c(u, v) < m + 1$ to every vertex set V_v in $\hat{G}$, while all other arcs that begin in V_u have cost $m + 1$. In particular, for every arc (u, v) in G there is exactly one arc from V_u to V_v in $\hat{G}$ with the same cost.

In order to transform $\hat{G}$ into G we iteratively contract all the vertices in each set V_u. We show that during these contractions an arc with cost less than $m + 1$ is never deleted. When contracting two vertices $x, y \in V_u$, an arc can be deleted for two reasons:

1. The arc connects x and y. Then we do not care about its deletion since arcs within sets have cost $m + 1$.
2. The arc connects x and some vertex w, w.l.o.g. in the direction from x to w, and $c(y, w) \le c(x, w)$. The vertices x and y belong to the same vertex set, thus there is at most one arc (in each direction) with cost less than $m + 1$. Since $c(y, w) \le c(x, w)$, the arc (x, w) must have cost $m + 1$. The case of an arc from w to x is analogous.

After contracting all n vertex sets V_i, there are exactly n vertices (and thus $n^2 - n$ arcs) left. Since we did not delete any arc with cost less than $m + 1$ and G has $n^2 - n$ arcs as well, we know that there cannot be any arc with cost $m + 1$ left in the contracted graph.

In the following we show that the contractions of $\hat{G}$ result in G. Consider an arbitrary arc $(u, v) \in A(G)$. Let c_u be the vertex obtained by contracting all vertices in V_u. We know that all arcs from V_u to V_v have cost $m + 1$, except exactly one arc with cost $c(u, v)$. Since contractions always keep the cheaper arc, the cost of the remaining arc (c_u, c_v) also has cost $c(u, v)$. As a result, each c_u in the contracted graph can be considered as u in G. □

Computing a tour for M is related to the *generalized traveling salesman problem* (GTSP) which can be tracked back to publications of Henry-Labordère and Saksena [26,27]. Given a partition of the cities into r sets, GTSP asks for a minimum cost tour containing (at least) one vertex from each of the r sets. Unfortunately, there are no known efficient ways to solve or approximate GTSP. However, we observe that using an optimal GTSP tour for the vertex sets corresponding to $T_1, \ldots, T_{k+1}$ instead of the tour through M still yields a 3-approximation. In fact, this remains true even if we fix one arbitrary city for each set which can be seen as follows. Consider a metric ATSP instance G with minimum spanning arborescence T computed by Lemma 8. Let $T_1, \ldots, T_{k+1}$ be the connected components of the graph created from removing all one-way arcs from T. Pick arbitrarily one vertex v_i^t from T_i for each $i \in \{1, \ldots, k+1\}$ and consider as asymmetric subgraph to be brute-forced the graph $M' = G[\{v_1^t, \ldots, v_{k+1}^t\}]$. Note that as induced subgraph of G, M' is a metric instance of ATSP; in contrast to the graph M used for Theorem 10.

To turn an optimum tour τ' for M' into a solution for G, we replace each vertex v_i^t by a tour through the vertices in T_i. These tours are computed by tree doubling for T_i. Since M' is a subgraph of G, an optimal tour for it is at most as expensive as an optimal tour for G by Lemma 2. The cost of the total tour is hence at most $\gamma c^*(M') \le c^*(G)$ for the tour connecting the components, plus $2c^*(G)$ for the subtours for each T_i, which overall is a 3-approximate solution.

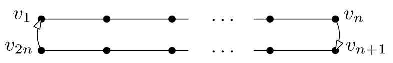

Fig. 3. The constructed graph that shows the tightness of the simplified parameterized 3-approximation.

To see that the ratio 3 for this simplified approach, which we refer to as *simplified parameterized 3-approximation*, is tight, consider for a fixed $n \in \mathbb{N}$ a cycle on $2n$ vertices where only the first and the $(n+1)$st links are asymmetric, see Fig. 3. Assign a cost of 1 to all arcs of the cycle. Let G be the metric closure of this graph and let T be an MSA for G that only contains one of the asymmetric links; observe that any MSA for G has to be built by removing exactly one link from the cycle, and one with a minimum number of one-way arcs hence deletes one of the asymmetric links. This results in two components, the path from v_1 to v_n and the path from v_{n+1} to v_{2n}. Pick v_n and v_{2n} as representatives for M'.

Trivially, (v_n, v_{2n}) is an optimal solution for M' and replacing v_n by the subtour $S_1 = (v_n, v_{n-1}, \dots, v_1)$, and v_{2n} by the subtour $S_2 = (v_{2n}, v_{2n-1} \dots, v_{n+1})$ results in a tour for G of cost $6n-4$. Since an optimal solution has cost $2n$, this shows that the ratio of 3 for the simplified parameterized 3-approximation is asymptotically tight.

Aside from the fact that we did not find a tight example for Theorem 10, seeing that the choice of any arbitrary vertex still yields a 3-approximation causes us to conjecture that our more sophisticated generalization of the tree doubling algorithm is in fact a 2-approximation. Proving such a ratio however requires an exploitable connection between the cost for the paths χ_i and the cost of τ'.

5. Trading approximation quality for runtime

In real life, we expect instances with many small asymmetries which have little impact but lead to relatively large parameter values. Therefore, ignoring asymmetric links where both directions have similar cost and trading some approximation quality for running time yields an intriguing perspective. As a formal way to describe moderate asymmetry, we use the asymmetry factor of Martínez Mori and Samaranayake [7] as introduced in Section 1.3. Since Δ is commonly used for the maximum degree, and we want to describe variable restrictions of the asymmetry factor, we use β instead. For $\beta \geq 1$ we call a link $\{u, v\}$ or arc (u, v) β*-symmetric* if $\frac{1}{\beta} \leq \frac{c(u,v)}{c(v,u)} \leq \beta$, otherwise it is called β*-asymmetric*. We show that our algorithms support a quality-runtime trade-off with respect to β.

5.1. Relaxed generalized Christofides algorithm

For a given β we modify the algorithm presented in Section 3 by treating every β-symmetric link as symmetric. This results in parametrization by the vertex cover of the subgraph induced by all β-asymmetric links. We denote this parameter by z_β. Since the β-symmetric subgraph is not completely symmetric, the Christofides algorithm cannot be directly used. Martínez Mori and Samaranayake [7] showed that it is $\frac{3}{2}$-stable by replacing every link with an undirected edge and assigning it the cost of the more expensive direction. Combined with the arguments used for Theorem 3, this gives a parameterized $(\frac{3}{2}\beta + 1)$-approximation for parameter z_β. This can be improved by turning the β-symmetric subgraph symmetric by assigning the cost of the cheaper direction. Although this may not yield a metric graph, it suffices that the original graph is metric to prove that the Christofides algorithm yields a good solution.

Theorem 12. *For any $\beta \geq 1$, metric ATSP can be $(\frac{3}{4}\beta + \frac{7}{4})$-approximated in $\mathcal{O}(n^3 + 2^{z_\beta} z_\beta^2)$ where z_β is the size of a minimum vertex cover of the subgraph induced by all β-asymmetric links.*

Proof. Let G be a β-symmetric metric graph and let $G_<$ be the undirected graph created from G by replacing each pair of arcs $(u, v), (v, u)$ with cost c_1 and c_2 in A by an undirected edge between u and v with cost $\min\{c_1, c_2\}$. Let T be a minimum spanning tree for G. Since all costs in $G_<$ are smaller or equal to the costs in G, the cost of T is at most the cost of a minimum spanning arborescence for G and hence at most $c^*(G)$.

Let $V' \subseteq V[G_<]$ be the set of vertices with odd degree in T. Since the original graph G is metric, a min-cost perfect matching for V' in G has a cost of at most $\frac{1}{2}c^*(G)$. Again, the cost of such a matching in $G_<$ is smaller or equal to its corresponding cost in G, hence a min-cost perfect matching M of V' in $G_<$ has cost at most $\frac{1}{2}c^*(G)$. Let W be an Eulerian circuit for the edges in $T \cup M$. Consider both directions to traverse W in G and the resulting cost of the tour. For each edge in W at most one of the corresponding two arcs in G has a larger cost than the undirected edge in $G_<$. In the worst case, all edges in W correspond to asymmetric links in G and have in one direction a cost of β times the cost of their counterparts in $G_<$. The better of the two traversal directions hence corresponds to a tour of cost at most $\frac{1}{2}(1+\beta)$ times the cost of W in $G_<$, which is at most $C(T \cup M) \leq \frac{3}{2} \cdot c^*(G)$. In the metric graph G, metric shortcuts can be used to turn W into a proper TSP tour without increasing the cost, which overall yields a $\frac{3}{4}(1+\beta)$-approximate solution.

Applying this $\frac{3}{4}(1+\beta)$-approximate solution for the β-symmetric subgraph G'' in step 3 of the algorithm used to prove Theorem 3 yields the claimed result. □

5.2. *Relaxed generalized tree doubling algorithm*

For the generalized tree doubling algorithm, we define a *β-one-way arc* as a one-way arc that is β-asymmetric. We denote by k_β the minimum number of β-one-way arcs in an MSA. Note that the strategy in Lemma 8 can also be used to find an MSA with k_β β-one-way arcs. The generalization of Theorem 10 is straightforward, instead of deleting all one-way arcs we only delete β-one-way arcs. This results in fewer components and a smaller graph M_β. The drawback is a change to the cost analysis: so far, we considered the component trees T_i to be symmetric. Now for every $(u, v) \in A[T_i]$, the opposite arc (v, u) can be up to β times as expensive. The adjusted tree doubling algorithm for the path through T_i uses every arc in T_i and its opposite arc at most once, which in total costs at most $(1+\beta)c^*(G)$. Combined with the cost of at most $c^*(G)$ for an optimum tour through M_β, this yields:

Theorem 13. *For any $\beta \geq 1$, metric ATSP can be $(2+\beta)$-approximated in $\mathcal{O}(2^{k_\beta} k_\beta^2 + n^3)$ where k_β is the minimum number of β-one-way arcs in an MSA.*

6. Experimental results

To test the practical viability of our algorithms, we implemented them in their relaxed form (see Section 5) to also observe their behavior when certain asymmetries are ignored. We evaluated on the asymmetric graphs from the TSPLIB collection [28], the standard benchmark for TSP solvers, and on a set of specific ATSP instances extracted from road networks by Rodríguez and Ruiz [29].

6.1. *Implementation details*

Our implementation (available on GitHub[1]) is written in *Python 3*, except for the vertex cover solver, which is written in *Java*. We used the Python library *NetworkX* [30] for graph manipulation, the C++ library *Lemon* [31] for computing MSAs, and *Concorde* [32] for solving TSP exactly. Since *Concorde* is a TSP solver, we transformed the ATSP instances into TSP instances with the transformation presented by Jonker and Volgenant [33,34].

We note that the runtime of our implementations is incomparable to state-of-the-art ATSP solvers. Among others, the reason is Python's inherently low performance and the inefficiency of solving ATSP with Concorde. However, this is of no importance for our evaluation of approximation ratio, parameter size, and the proof of concept.

6.2. *Experiments*

Many instances in the TSPLIB are not metric, thus our algorithms cannot be directly applied. We therefore computed the metric closure of each graph by setting the cost of each arc (u, v) to the cost of the shortest path from u to v and used this as input for our algorithm. The TSPLIB contains 19 asymmetric instances ranging from 17 up to 443 vertices with different underlying properties. The instances' names contain the number of vertices (e.g. *ftv33*) and similar names indicate similar properties. For example, all instances starting with *rbg* have relatively high symmetry and a high number of zero-cost arcs, which distinguishes them from the other instances. Table 1 depicts an overview of the instances and the characteristics of their metric closures. In particular, the metric closure of *br17* is completely symmetric, so we ignored it in our experiments.

For each TSPLIB instance we executed each algorithm five times with different values for β and recorded the value of the parameter as well as the approximation ratio. Starting with $\beta = 1$ (which corresponds to 100% of the asymmetric links), we raised the value of β each step, reducing the number of asymmetric links treated as asymmetric to a quarter of the previous experiment. Some instances include many zero-cost arcs, so there is no value of β ignoring those. We considered zero-cost arcs to have a small positive cost (set to 0.1) when calculating the asymmetry factor, thus treating links with a small additive error as symmetric in case of these otherwise undauntedly asymmetric one-way arcs of cost 0. Note that we did not alter the instance, but only used these additive errors for relaxation decisions. Finally, β was set to ∞, such that the graph is treated as completely symmetric. This results in the non-generalized versions of the tree doubling and Christofides algorithm. The results are shown in Table 2.

The second dataset contains 450 ATSP instances based on travel distances between random points sampled across different regions and cities in Spain. The graphs in this dataset have between 50 and 500 vertices. On average 98.8% of the links are asymmetric (std. dev. 1.08%) and no graph contains arcs of cost zero. Most links are however only slightly asymmetric: denoting by *asymmetry factor* the relative difference between the cost of a link's more expensive arc and its opposite arc, the mean asymmetry factor is 3.55% on average over all graphs (std. dev. 0.040%). The median asymmetry factor is 1.32% (std. dev. 1.56%) on average. There are however also links with large asymmetry factor. The highest asymmetry factor is 15.0 (std. dev. 58.8) on average. Overall this makes the graphs in the second dataset very relevant to the algorithms we present. Unfortunately, due to computational constraints and the size of the dataset and the graphs therein, we could only

[1] https://github.com/Blaidd-Drwg/atsp-approximation.

Table 1
An overview of the 19 asymmetric TSPLIB instances and the properties of their metric closures. The asymmetry factor of each vertex pair u, v is calculated by the formula $\max(\frac{c(u,v)}{c(v,u)}, \frac{c(v,u)}{c(u,v)})$ if there is no arc with cost zero between the two vertices and undefined otherwise. The median and maximum asymmetry factors were calculated ignoring undefined values.

instance name	symmetric links	median asymmetry factor	maximum asymmetry factor	zero-cost arcs
ft53	0%	2.04	23.04	None
ft70	0%	1.40	5.87	None
ftv170	6%	1.22	34.00	None
ftv33	6%	1.31	18.75	None
ftv35	5%	1.31	18.75	None
ftv38	6%	1.30	18.75	None
ftv44	5%	1.28	18.75	None
ftv47	3%	1.31	11.17	None
ftv55	5%	1.28	18.75	None
ftv64	4%	1.29	34.00	None
ftv70	4%	1.29	34.00	None
kro124p	0%	1.04	3.42	None
p43	63%	13.61	14.64	3%
rbg323	33%	3.00	20.00	47%
rbg358	50%	3.00	18.00	65%
rbg403	49%	2.50	12.00	68%
rbg443	49%	2.67	11.00	69%
ry48p	1%	1.04	3.63	None

Table 2
Experimental results on TSPLIB instances with percentage of asymmetric links that were treated as asymmetric shown in the column header. Each cell contains parameter value and approximation factor, separated by a slash (trivial parameter value 0 omitted in 0% column). Superiority in the sense of smaller parameter or better approximation ratio is highlighted with bold font.

	Generalized Christofides algorithm					Generalized tree doubling algorithm				
	100%	25%	6.25%	1.56%	0%	100%	25%	6.25%	1.56%	0%
ft53	53/**1.00**	29/1.54	13/1.70	6/1.69	**1.72**	**45**/1.08	**25**/**1.36**	**6**/**1.42**	**1**/**1.57**	1.97
ft70	69/1.02	34/1.24	12/1.26	7/1.41	**1.24**	**64**/1.02	**27**/**1.13**	**4**/**1.20**	**2**/**1.21**	1.28
ftv170	155/1.17	123/1.38	**97**/1.57	**64**/1.85	2.37	**108**/**1.14**	**107**/**1.14**	103/**1.21**	75/**1.46**	**1.81**
ftv33	29/**1.12**	19/1.45	11/**1.43**	5/1.56	**1.33**	**19**/1.34	**16**/**1.34**	11/1.44	**2**/**1.23**	1.50
ftv35	32/**1.07**	21/1.51	12/1.55	6/1.49	**1.38**	**23**/1.15	**17**/**1.23**	**11**/**1.47**	**2**/**1.28**	1.58
ftv38	33/**1.13**	23/1.38	12/**1.43**	7/1.47	**1.39**	**23**/1.24	**18**/**1.33**	12/1.54	**3**/**1.30**	1.62
ftv44	40/**1.09**	32/**1.38**	19/1.46	10/1.56	**1.54**	**32**/1.24	**25**/1.41	**18**/**1.41**	**7**/**1.50**	1.79
ftv47	44/**1.05**	32/1.47	19/1.66	13/1.65	1.66	**35**/1.09	**30**/**1.16**	19/**1.34**	**9**/**1.38**	**1.58**
ftv55	49/**1.13**	38/1.44	**23**/1.57	15/1.65	**1.84**	**37**/1.20	**32**/**1.26**	25/**1.34**	**12**/**1.58**	2.00
ftv64	57/1.11	46/1.46	**30**/1.66	18/1.73	1.72	**50**/**1.10**	**43**/**1.15**	31/**1.29**	**14**/**1.71**	**1.45**
ftv70	63/**1.11**	50/1.43	**32**/1.64	20/1.72	1.96	**53**/1.26	**47**/**1.14**	33/**1.21**	**16**/**1.57**	**1.51**
kro124p	99/1.11	86/1.30	65/1.36	40/1.41	**1.24**	**81**/**1.06**	**70**/**1.13**	**57**/**1.20**	**34**/**1.28**	1.37
p43	15/1.01	6/1.01	0/1.01	0/1.01	1.01	**0**/1.01	**0**/1.01	0/1.01	0/1.01	1.01
rbg323	**148**/**1.02**	59/**1.17**	43/**1.19**	18/1.30	1.34	235/1.09	**22**/1.27	**6**/1.27	**0**/1.30	**1.30**
rbg358	**108**/**1.01**	47/**1.13**	27/**1.15**	22/**1.14**	**1.18**	232/1.03	**39**/1.14	**18**/1.19	**13**/1.20	1.22
rbg403	125/**1.01**	41/**1.12**	11/1.26	11/1.26	**1.17**	**113**/1.05	**30**/1.14	**0**/**1.24**	**0**/**1.24**	1.24
rbg443	138/**1.00**	43/**1.14**	12/1.24	12/1.24	**1.15**	**127**/1.04	**32**/1.17	**0**/1.24	**0**/1.24	1.24
ry48p	47/1.20	37/1.40	23/1.46	11/1.47	**1.16**	**28**/**1.10**	**22**/**1.14**	**11**/**1.24**	**5**/**1.29**	1.21

determine the values of the parameters and not the cost of all optimal tours and the obtained approximation ratio. Fig. 4 presents the relative value of z and k for different values of β.

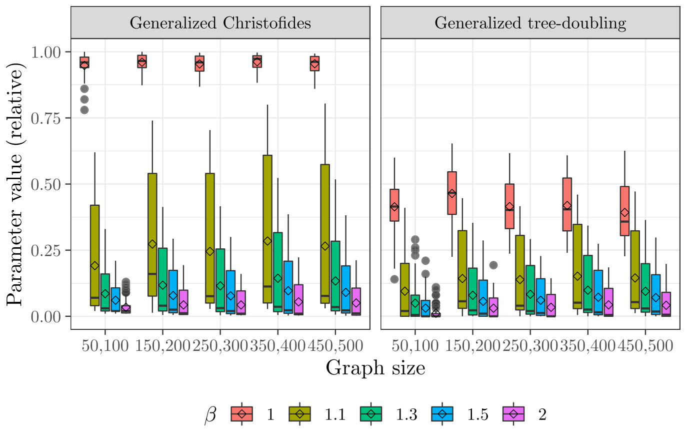

Fig. 4. Parameter values relative to graph size for generalized Christofides and tree doubling algorithms for different values of β. Each box spans the second and third quartile of the data and whiskers extend for 1.5 inter-quartile-ranges. The median is marked as a line, the mean as a rhombus and outliers as disks. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

6.3. Evaluation

First, we note that most graphs in the TSPLIB contain very little symmetry. This leads to large parameter values for $\beta = 1$, i.e., only some graphs with more than 10% symmetry have parameter values below 50% of the graph size. Still, we observe that the approximation factor is always far below the upper bound, never exceeding even 2.0. Also, we see that interpolating β to reduce the number of relevant asymmetric links produces a valuable trade-off between approximation quality and parameter value. Comparing both algorithms, we observe that on the majority of instances and values for β the generalized tree doubling algorithm produces smaller parameter values.

This can also be observed on the instances of the second dataset, which we consider to be more representative of realistic inputs. We want to highlight that the parameters are significantly smaller than the size of the input graphs even for small values of β. E.g., for the generalized tree doubling algorithm with $\beta = 1.1$ the median relative parameter value over all instances is 0.045. It also seems that the relative size of the parameters is stable for different input sizes.

These results underline the practicality of our approach, especially with regards to the parameter values obtained by choosing a suitable β.

CRediT authorship contribution statement

Lukas Behrendt: Formal analysis, Investigation, Methodology, Software, Writing – original draft. **Katrin Casel:** Conceptualization, Investigation, Supervision, Validation, Writing – review & editing. **Tobias Friedrich:** Project administration, Resources, Supervision. **J.A. Gregor Lagodzinski:** Conceptualization, Investigation, Supervision, Validation, Writing – review & editing. **Alexander Löser:** Formal analysis, Investigation, Methodology, Software, Writing – original draft. **Marcus Wilhelm:** Formal analysis, Investigation, Methodology, Software, Writing – original draft.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data is publicly available, corresponding links are given in the paper.

Acknowledgments

We thank René van Bevern and the reviewers of FCT for their helpful comments and suggestions for improvement.

References

[1] O. Svensson, J. Tarnawski, L.A. Végh, A constant-factor approximation algorithm for the asymmetric traveling salesman problem, J. ACM 67 (6) (2020) 37, https://doi.org/10.1145/3424306.
[2] N. Christofides, Worst-case analysis of a new heuristic for the travelling salesman problem, Technical Report 388, Graduate School of Industrial Administration, Carnegie Mellon University, 1976.

[3] R. van Bevern, V.A. Slugina, A historical note on the 3/2-approximation algorithm for the metric traveling salesman problem, Hist. Math. 53 (2020) 118–127, https://www.sciencedirect.com/science/article/pii/S0315086020300240.

[4] A. Karlin, N. Klein, S.O. Gharan, A (slightly) improved approximation algorithm for metric TSP, in: Proc. of STOC'21, ACM, 2021, pp. 32–45, https://doi.org/10.1145/3406325.3451009.

[5] V. Traub, J. Vygen, An improved approximation algorithm for ATSP, in: Proc. of STOC'20, ACM, 2020, pp. 1–13, https://doi.org/10.1145/3357713.3384233.

[6] M. Karpinski, M. Lampis, R. Schmied, New inapproximability bounds for TSP, J. Comput. Syst. Sci. 81 (8) (2015) 1665–1677, https://doi.org/10.1016/j.jcss.2015.06.003.

[7] J.C. Martínez Mori, S. Samaranayake, Bounded asymmetry in road networks, Sci. Rep. 9 (11951) (2019), https://doi.org/10.1038/s41598-019-48463-z.

[8] D. Marx, Parameterized complexity and approximation algorithms, Comput. J. 51 (1) (2008) 60–78, https://doi.org/10.1093/comjnl/bxm048.

[9] H. Böckenhauer, J. Hromkovic, R. Klasing, S. Seibert, W. Unger, Towards the notion of stability of approximation for hard optimization tasks and the traveling salesman problem, Theor. Comput. Sci. 285 (1) (2002) 3–24, https://doi.org/10.1016/S0304-3975(01)00287-0.

[10] L. Kowalik, M. Mucha, Two approximation algorithms for ATSP with strengthened triangle inequality, in: Proc. of WADS'09, Springer, 2009, pp. 471–482, https://doi.org/10.1007/978-3-642-03367-4_41.

[11] L.S. Chandran, L.S. Ram, Approximations for ATSP with parametrized triangle inequality, in: Proc. of STACS'02, Springer, 2002, pp. 227–237, https://doi.org/10.1007/3-540-45841-7_18.

[12] M. Bläser, B. Manthey, J. Sgall, An improved approximation algorithm for the asymmetric TSP with strengthened triangle inequality, J. Discret. Algorithms 4 (4) (2006) 623–632, https://doi.org/10.1016/j.jda.2005.07.004.

[13] T. Zhang, W. Li, J. Li, An improved approximation algorithm for the ATSP with parameterized triangle inequality, J. Algorithms 64 (2–3) (2009) 74–78, https://doi.org/10.1016/j.jalgor.2008.10.002.

[14] R. Klasing, T. Mömke, A modern view on stability of approximation, in: Adventures Between Lower Bounds and Higher Altitudes - Essays Dedicated to Juraj Hromkovič on the Occasion of His 60th Birthday, in: LNCS, vol. 11011, Springer, 2018, pp. 393–408, https://doi.org/10.1007/978-3-319-98355-4_22.

[15] D. Marx, A. Salmasi, A. Sidiropoulos, Constant-factor approximations for asymmetric TSP on nearly-embeddable graphs, in: Proc. of APPROX/RANDOM'16, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2016, 16, https://doi.org/10.4230/LIPIcs.APPROX-RANDOM.2016.16.

[16] H. Böckenhauer, J. Hromkovic, J. Kneis, J. Kupke, The parameterized approximability of TSP with deadlines, Theor. Comput. Sci. 41 (3) (2007) 431–444, https://doi.org/10.1007/s00224-007-1347-x.

[17] É. Bonnet, M. Lampis, V.T. Paschos, Time-approximation trade-offs for inapproximable problems, J. Comput. Syst. Sci. 92 (2018) 171–180, https://doi.org/10.1016/j.jcss.2017.09.009.

[18] M. Cygan, F. Fomin, L. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, S. Saurabh, Parameterized Algorithms, Springer, 2015, https://doi.org/10.1007/978-3-319-21275-3.

[19] M. Held, R.M. Karp, A dynamic programming approach to sequencing problems, J. Soc. Ind. Appl. Math. 10 (1) (1962) 196–210.

[20] M.-Y. Kao, Encyclopedia of Algorithms, Springer Science & Business Media, 2008.

[21] U. Feige, M. Singh, Improved approximation ratios for traveling salesperson tours and paths in directed graphs, in: Proc. of APPROX'07, Springer, 2007, pp. 104–118, https://doi.org/10.1007/978-3-540-74208-1_8.

[22] A. Asadpour, M.X. Goemans, A. Madry, S.O. Gharan, A. Saberi, An O(log n/log log n)-approximation algorithm for the asymmetric traveling salesman problem, Oper. Res. 65 (4) (2017) 1043–1061, https://doi.org/10.1287/opre.2017.1603.

[23] J. Edmonds, Optimum branchings, J. Res. Natl. Bur. Stand. B, Math. Math. Phys. 71B (4) (1967) 233.

[24] H.N. Gabow, Z. Galil, T.H. Spencer, R.E. Tarjan, Efficient algorithms for finding minimum spanning trees in undirected and directed graphs, Combinatorica 6 (2) (1986) 109–122, https://doi.org/10.1007/BF02579168.

[25] J.A. Hoogeveen, Analysis of Christofides' heuristic: some paths are more difficult than cycles, Oper. Res. Lett. 10 (5) (1991) 291–295, https://doi.org/10.1016/0167-6377(91)90016-I.

[26] A.L. Henry-Labordère, The record balancing problem: a dynamic programming solution of a generalized traveling salesman problem, RAIRO B-2 (1969) 43–49.

[27] J.P. Saksena, Mathematical model of scheduling clients through welfare agencies, Comput. Oper. Res. 8 (1970) 185–200.

[28] G. Reinelt, TSPLIB—a traveling salesman problem library, INFORMS J. Comput. 3 (4) (1991) 376–384, https://doi.org/10.1287/ijoc.3.4.376.

[29] A. Rodríguez, R. Ruiz, The effect of the asymmetry of road transportation networks on the traveling salesman problem, Comput. Oper. Res. 39 (7) (2012) 1566–1576, https://doi.org/10.1016/j.cor.2011.09.005.

[30] A. Hagberg, D. Schult, P. Swart, Exploring network structure, dynamics, and function using NetworkX, in: Proc. of the 7th Python in Science Conference, 2008, pp. 11–15, http://conference.scipy.org/proceedings/SciPy2008/paper_2/.

[31] B. Dezső, A. Jüttner, P. Kovács, LEMON – an open source C++ graph template library, Electron. Notes Theor. Comput. Sci. 264 (5) (2011) 23–45, https://doi.org/10.1016/j.entcs.2011.06.003.

[32] D. Applegate, R. Bixby, W. Cook, V. Chvátal, On the solution of traveling salesman problems, Doc. Math. (1998) 645–656.

[33] R. Jonker, T. Volgenant, Transforming asymmetric into symmetric traveling salesman problems, Oper. Res. Lett. 2 (4) (1983) 161–163.

[34] R. Jonker, T. Volgenant, Transforming asymmetric into symmetric traveling salesman problems: erratum, Oper. Res. Lett. 5 (4) (1986) 215–216.