University of Wrocław
Department of Mathematics and Computer Science
Institute of Mathematics
specialty: theoretical mathematics

Jakub Tarnawski

A constant-factor approximation algorithm for the Asymmetric Traveling Salesman Problem on graphs with zero-one edge weights

Master's thesis
written under the supervision of
Prof. Andrzej Kisielewicz

Wrocław 2016

Abstract

The Traveling Salesman Problem is the task of finding a shortest cyclical tour which visits all vertices of a weighted graph. We consider approximation algorithms for ATSP – the directed version of the problem. There is a recent (Svensson 2015) constant-factor approximation algorithm for ATSP on unweighted directed graphs. The result is obtained through a reduction to a certain easier problem. In this thesis we solve this easier problem for a wider class of directed graphs – ones with all edge weights being 0 or 1. This also implies a better upper bound for the integrality gap of the Held-Karp relaxation related to this problem.

Contents

1 Introduction

The traveling salesman problem (TSP) – one of finding the shortest cyclical tour which visits all vertices in a graph with weights (costs) on edges[1] – is probably the most well-known question in combinatorial optimization. It is also a classical NP-complete problem.[2] This means that there is probably no exact algorithm which solves it in polynomial time. There are several methods of dealing with such hardness. One of the main approaches – especially in theoretical considerations – is by approximation algorithms, i.e., effective algorithms whose answer is provably never *much* worse than the optimum. We focus on such algorithms in this thesis. We will say that an algorithm for the traveling salesman problem is a β-approximation if its returned route is always at most β times more expensive than the optimum.[3]

Because the version of the problem where weights on edges are arbitrary and we require that each vertex be visited *exactly* once does not admit any nontrivial approximation algorithm (and is arguably unnatural), we instead consider a version where each vertex may be visited multiple times. We can equivalently say that each vertex is visited once, but we assume that the graph is complete and the edge weights satisfy the triangle inequality.

As a basic example, we can give a 2-approximation algorithm for the symmetric version of the problem (where the graph is undirected): find a minimum spanning tree and take its each edge twice. The subgraph so obtained is connected and Eulerian; its Eulerian cycle visits all vertices, so it is an admissible solution. Its cost is at most twice the cost of the minimum spanning tree. On the other hand, the optimum solution also contains some spanning tree, so its cost is at least that of the minimum spanning tree. Therefore our solution is at most 2 times worse than the optimum.

A better algorithm was given in 1976 by Christofides [Chr76]. It is almost as short and clean: the idea is to start with a minimum spanning tree, and then fix the degrees of the odd-degree vertices by connecting them with paths (pairing them up in an optimal way). One then shows that this is a 3/2-approximation for the symmetric case. Interestingly, this approximation factor remains the best known for the general case until today, and improving it is a major open problem in combinatorial optimization.

It was only recently that better algorithms were developed for the special case where the graph is also unweighted: in 2011, Oveis Gharan, Saberi and Singh [GSS11] gave a $(3/2 - \varepsilon)$-approximation, with ε of the order of 10^{-24}. A significant improvement was then obtained by Mömke and Svensson [MS11] (1.461), Mucha [Muc12] (1.444) and Sebő and Vygen [SV14] (1.4).

The asymmetric version (where the graph is directed, so that going from u to v can be much more expensive than from v to u), which we will focus on, appears to be significantly harder than the symmetric one. The oldest known nontrivial approximation algorithm for it was given by Frieze, Galbiati and Maffioli [FGM82] in 1982; it is a $(\log n)$-approximation[4]. An asymptotic improvement over this was achieved only in 2010 by Asadpour et al. [AGM$^+$10], who gave an $\mathcal{O}(\log n / \log\log n)$-approximation. Here, the major open problem is to show an algorithm with a constant approximation guarantee. On the hardness side, the best known result is that it is NP-hard to get a $\frac{75}{74}$-approximation [KLS15].[5]

Most modern approaches to both the symmetric and the asymmetric version involve solving a linear programming relaxation and then using its optimum solution to guide the algorithm. This linear program, called the Held-Karp relaxation, is such that its integral solutions correspond

[1] For all formal definitions, refer to Section 2. Section 1 serves as motivation and an informal overview.

[2] Formally, its decision version is NP-complete.

[3] Here β can be a constant or a function of the input.

[4] By convention, $n = |V|$ is the number of vertices in the graph.

[5] Formally, the following decision problem is NP-hard: given a threshold t and a graph which either has a tour of cost at most t or has no tour of cost at most $\frac{75}{74}t$, decide which one is the case.

to tours which visit all vertices. However, an optimum solution to the relaxation (which can be found in polynomial time) is not necessarily integral, and it will often have lower value than the integral optimum, thus providing only a lower bound. We are interested in a measure of quality of the relaxation called its *integrality gap*, which is the maximum ratio between values of the integral optimum and the fractional optimum. The best known lower bound for the integrality gap of the Held-Karp relaxation in the asymmetric case is 2 [CGK06], and this is believed to be close to the truth; however, the best known upper bound is $\mathcal{O}(\text{poly} \log \log n)$ by recent work of Anari and Oveis Gharan [AG15]. Note that this gives a polynomial-time algorithm which estimates the cost of the optimum tour within this factor (by just solving the linear program); however, it does not translate into an efficient way of actually finding a tour within this factor of the estimated value.

Before 2015, nothing was known about the special case of unweighted (directed) graphs. In a recent major development, Svensson [Sve15] gave a constant-approximation algorithm for this case. His argument consists of two components. The first is a black-box reduction to an easier problem called Local-Connectivity ATSP. In this problem, instead of providing a tour which connects the entire graph, one is only required to output a (not necessarily connected) Eulerian set of edges which crosses each cut induced by a given partitioning of the vertices. The trade-off is that instead of minimizing the cost of the tour, now one is forced to ensure that the cost of each connected component of the solution is within some factor of what the linear programming relaxation "pays" for this component (this property will be called *lightness*). The second component of the result is a *light* algorithm for unweighted graphs, which is rather short and natural.

In this thesis, we expand on that result and make progress towards the general problem by providing a *light* algorithm for the "next best" class of graphs, namely, directed graphs where every edge has weight 0 or 1 (Theorem 3.1). Using Svensson's reduction, this implies both a constant-factor approximation algorithm for ATSP on these graphs, and a constant upper bound on the integrality gap of the Held-Karp relaxation in this case (Theorem 3.2).

To further motivate our result (as in: why this class of weight functions is the natural next step), we note that, by Svensson's result, it is easy to obtain an $\mathcal{O}(w_{max}/w_{min})$-approximation in graphs where the maximum edge weight is w_{max} and the minimum is w_{min}. Therefore we are looking to deal with situations where this ratio is not bounded – and indeed, it is the worst when $w_{min} = 0$. We also remark that, for the Asymmetric Traveling Salesman *Path* Problem, there is a simple example of a graph where the Held-Karp relaxation has an integrality gap of 2 (the best known lower bound) with only weights 0 and 1 [FGS13, v2].[6]

This thesis is based on ideas obtained in joint work with Ola Svensson and László A. Végh. We also note that the result has already been generalized by Svensson, T. and Végh to any two different edge weights [STV16]; however, they employ a different method, and the constants in this thesis are lower.

The organization of the thesis is as follows. We start with formal definitions and a self-contained introduction to the problem in Section 2, where we also dicuss the main tools used in previous successful approaches and the new Local-Connectivity ATSP problem introduced by Svensson. In Sections 3 to 5 we present our result – a solution to that problem for graphs with zero-one weights – and its proof. Finally, we discuss open research directions in Section 6.

[6]The zero-one weight case was not considered for symmetric TSP, since there it immediately reduces to the unweighted case.

2 Preliminaries

2.1 Graphs and tours

We assume that the reader is familiar with basic notions of graph theory. We consider a finite weighted directed graph $G = (V, E, w)$, where $w : E \to \mathbb{R}_+$. If every edge has weight 1, we say that the graph is unweighted. We use the terms "weight", "cost" and "value" interchangeably. Let us define the notion of *tour* using the standard characterization of Eulerian graphs:

Definition 2.1. *By a (traveling salesman)* tour *in G we denote a multiset $F \subseteq E$ of edges which is connected (i.e., the graph (V, F) is connected) and Eulerian (i.e., each vertex $v \in V$ has its out-degree and in-degree equal).*[7] *The cost of the tour is the quantity $w(F) = \sum_{e \in F} w(e)$.*

Note that vertices (and even edges) may be visited multiple times in a tour. We will assume that the input graph G is strongly connected (which is clearly equivalent to it having a tour). We can now define our main problem:

Definition 2.2. *The Asymmetric Traveling Salesman Problem (ATSP) is defined as follows. Given a strongly connected edge-weighted digraph $G = (V, E, w)$, the task is to find a tour F in G of minimum cost $w(F)$.*

Let us also introduce some additional notation:

- given subsets of vertices $S, T \subseteq V$, we denote the set of edges between them by $\delta(S, T)$, i.e., $\delta(S, T) = \{(v, w) \in E : v \in S, w \in T\}$,
- for a subset of vertices $S \subseteq V$, its outgoing edges are denoted by $\delta^+(S)$ and incoming edges by $\delta^-(S)$, i.e., $\delta^+(S) = \delta(S, V \setminus S)$, $\delta^-(S) = \delta(V \setminus S, S)$; internal edges are denoted by $E(S)$, i.e., $E(S) = \{(u, v) \in E : u, v \in S\}$,
- given a subset of edges $F \subseteq E$, its indicator vector is denoted by χ_F,
- we often shorten $\{v\}$ to v, as in $\delta^+(v)$,
- we call nonnegative vectors $x : E \to \mathbb{R}_+$ *flows*, and often write x_e rather than $x(e)$,
- given a function $f : A \to \mathbb{R}$ and a subset $B \subseteq A$ of its domain, we write $f(B)$ for $\sum_{x \in B} f(x)$ and $f|_B$ for the restriction of f to B,
- by the above, e.g. $x(\delta^+(S))$ denotes the sum of all x-flow going out of the set S, i.e., its fractional x-outdegree.

Definition 2.3. *We say that a polynomial-time algorithm ALG for ATSP is a β-approximation if for every input graph G we have $w(ALG(G)) \leq \beta \cdot w(OPT(G))$, where $ALG(G)$ is the tour output by the algorithm and $OPT(G)$ is a minimum-cost tour in G. We call $OPT(G)$ the (integral) optimum solution, and β – an approximation ratio of ALG.*

A constant-factor approximation algorithm is a β-approximation for a constant β.

[7] That is, to describe a tour, one only needs to specify how many times each edge is traversed; the order in which the edges appear in an Eulerian cycle is insignificant.

2.2 Integrality of circulations

We will find it useful to relax the notion of a tour to allow fractional solutions, since we can find these in polynomial time by solving a linear program called the Held-Karp relaxation, which we discuss in Section 2.3. Let us first introduce the fractional counterpart of a Eulerian multiset of edges – a circulation:

Definition 2.4. *We say that a flow $x : E \to \mathbb{R}_+$ is a* circulation *if every vertex $v \in V$ has its fractional outdegree and indegree equal, i.e., $x(\delta^+(v)) = x(\delta^-(v))$.*

Note that:

- integral circulations correspond to Eulerian multisets of edges,
- the constraint $x(\delta^+(v)) = x(\delta^-(v))$ is linear in the variables $\{x_e\}_{e \in E}$,
- if x is a circulation, it satisfies $x(\delta^+(S)) = x(\delta^-(S))$ for any subset of vertices $S \subseteq V$.[8]

The typical problem about circulations is to find a minimum-cost integral circulation satisfying certain integral lower and upper bounds (on vertices and/or edges). This problem is polynomial-time solvable, which can be seen as a consequence of the fact that *its polyhedron is integral*. We make this precise in the following theorem (cf. [Sch03, Corollary 13.10b and Note 11.6a]):

Theorem 2.5. *Consider a digraph $G = (V, E)$. Let $l, u : E \to \mathbb{Z}_+ \cup \{\infty\}$ and $L, U : V \to \mathbb{Z}_+ \cup \{\infty\}$ be some integral lower and upper bound functions on edges and vertices, respectively. Consider the polyhedron P corresponding to circulations in G satisfying these lower and upper bounds, that is,*

$$P = \{x \in \mathbb{R}^E : l(e) \le x_e \le u(e) \text{ for all } e \in E,\ L(v) \le x(\delta^+(v)) = x(\delta^-(v)) \le U(v) \text{ for all } v \in V\}.$$

Then the polyhedron P (possibly empty) is integral*, i.e., its each vertex $x \in \mathbb{R}^E$ is integral (i.e., $x \in \mathbb{Z}^E$).*

It is well-known that if a linear program is feasible and bounded, then the optimum is attained at a vertex of the polyhedron. Algorithmically, this fact together with Theorem 2.5 implies the following:

Corollary 2.6 (integrality of circulations). *Let G, l, u, L, U, P be as above and $w : E \to \mathbb{R}_+$ be a weight function. Given $x' \in P$ (i.e., x' is a fractional circulation satisfying the degree bounds), we can find an* integral *circulation $x \in P$ whose weight (with respect to w) is no more than that of x', i.e., $\sum_{e \in E} w_e x_e \le \sum_{e \in E} w_e x'_e$. This can be done in polynomial time.*

Proof. There are polynomial-time algorithms which can, given a cost vector $w \in \mathbb{R}^E$ and a description of a polyhedron $P \subseteq \mathbb{R}^E$, find a point $x \in P$ where the linear objective function $\sum_{e \in E} w_e x_e$ is minimized and x is a vertex of P – as long as P is bounded in the direction of the vector $-w$ [9] and nonempty. (See e.g. [LRS11, Theorem 2.1.6].) By Theorem 2.5, x is then integral. Also, its weight is no more than that of x' because x minimizes the weight. □

[8] To get this, sum up $x(\delta^+(v)) = x(\delta^-(v))$ over all $v \in S$.

[9] Which the circulation polyhedron P is, since $w \in \mathbb{R}^E_+$ and $P \subseteq \mathbb{R}^E_+$.

2.3 The Held-Karp relaxation and its integrality gap

We will introduce a fractional relaxation for ATSP similarly as for the minimum-cost circulation problem. Since ATSP is NP-hard, we do not expect to obtain an integral polyhedron in this way; still, we will be interested to know how "non-integral" it is. The motivation is that if there is an integral solution whose weight is within some factor β of the fractional optimum, then we could hope to find such a good integral solution by first solving for the fractional optimum and then "rounding" it to integrality in some way. Such rounding techniques have been very successful in the theory of approximation algorithms.

The task is to express the conditions of being a tour using linear constraints. Being Eulerian (or, in fractional terms, a circulation) is readily written as $x(\delta^+(v)) = x(\delta^-(v))$ for all $v \in V$ as before. As for connecting the entire graph, we can require every proper subset of vertices $S \subset V$ to have at least one outgoing edge with the following *subtour elimination constraints*: $x(\delta^+(S)) \geq 1$. Thus we arrive at:

Definition 2.7. *Given $G = (V, E, w)$, the* Held-Karp relaxation *for ATSP on G is the following linear program:*

$$\begin{aligned} &\textit{minimize} && \sum_{e \in E} w_e x_e \\ &\textit{subject to} && x(\delta^+(v)) = x(\delta^-(v)) && \textit{for all } v \in V, \\ & && x(\delta^+(S)) \geq 1 && \textit{for all } \emptyset \neq S \subsetneq V, \\ & && x_e \geq 0 && \textit{for all } e \in E. \end{aligned}$$

As a convention, we will always set $x^\star$ to be an optimum fractional solution to the relaxation. We denote its value $\sum_{e \in E} w_e x^\star_e$ by $w(x^\star)$ for brevity.

Our algorithms will begin by computing $x^\star$. Note that there are exponentially many constraints of the second kind; however, this linear program can still be solved in time polynomial in n [Sch03, Section 58.9]. It is again easy to see that integral solutions of the relaxation are exactly the tours of G; the value $w(x^\star)$ of $x^\star$ is thus a lower bound for the integral optimum (the minimum cost of a tour in G): $w(x^\star) \leq w(OPT(G))$.

As indicated in the introduction, we will be interested in the quality (tightness) of this lower bound, i.e., the ratio between the integral optimum value and the value of $x^\star$.

Definition 2.8. *We define the integrality gap of the Held-Karp relaxation as a function of n to be the supremum, taken over all graphs $G = (V, E, w)$ of size n, of the ratio $w(OPT(G))/w(x^\star)$, where $x^\star$ is a Held-Karp optimum for G.*

Our relative lack of understanding of ATSP shows in the gap between the best known lower and upper bounds on the integrality gap of the Held-Karp relaxation: they are 2 [CGK06] and $\mathcal{O}(\text{poly} \log \log n)$ [AG15] respectively. Furthermore, the latter result is not constructive, i.e., it does not yield an approximation algorithm with the same ratio; the best known approximation ratio is $\mathcal{O}(\log n / \log \log n)$ [AGM+10].

Finally, since many modern algorithms utilize the "rounding" framework, their guarantees are stronger, in that they are with respect to the Held-Karp lower bound instead of the integral optimum. More precisely:

Definition 2.9. *We say that a polynomial-time algorithm ALG for ATSP is a* β-approximation with respect to the Held-Karp relaxation *if for every input graph G we have $w(ALG(G)) \leq \beta \cdot w(x^\star)$, where $ALG(G)$ is the tour output by the algorithm and $x^\star$ is a Held-Karp optimum for G.*

Of course, any β-approximation with respect to the Held-Karp relaxation is also a β-approximation. Conversely, most known β-approximations for ATSP (as well as for symmetric TSP) turn out to also be (or can be made to be) β-approximations with respect to the Held-Karp relaxation.

2.4 Local-Connectivity ATSP

In a recent major contribution, Svensson [Sve15] showed a constant-approximation algorithm for the important special case of unit weights (i.e., G is an unweighted digraph). This result is based on a reduction of ATSP to an easier problem called Local-Connectivity ATSP, where the global connectivity condition (that a tour should connect the entire graph) is somewhat relaxed. We will first define the problem and then discuss its relationship with ATSP.

Definition 2.10. *The Local-Connectivity ATSP problem is defined as follows. Given a (strongly connected weighted) digraph $G = (V, E, w)$ with an optimum Held-Karp solution $x^\star$ and a partitioning $V = V_1 \cup ... \cup V_k$ of its vertices, the task is to find a function* $\mathrm{lb} : V \to \mathbb{R}_+$ *and an Eulerian multiset of edges $F \subseteq E$ such that:*

- $\mathrm{lb}(V) \leq w(x^\star)$,
- *the set F crosses each cut induced by the partitioning, i.e., for each $i = 1, ..., k$ we have $|F \cap \delta^+(V_i)| \geq 1$,*
- *the function* lb *cannot depend on the partitioning $V = V_1 \cup ... \cup V_k$.*[10]

We say that a connected component $\widetilde{G}$ of the subgraph (V, F) is α-light[11] *if its weight satisfies the upper bound $w(\widetilde{G}) \leq \alpha \cdot \mathrm{lb}(\widetilde{G})$.*[12]

Finally, we say that a polynomial-time algorithm for Local-Connectivity ATSP is α-light if it always satisfies that each connected component $\widetilde{G}$ of the graph (V, F) is α-light.[13]

To explain the definition, we remark that:

- lb stands for "lower bound": since $\mathrm{lb}(V) \leq w(x^\star) \leq w(OPT(G))$, the function lb intuitively encodes some way to distribute between vertices the lower bound on the cost of any tour,
- the α-lightness condition is stronger than requiring $w(F) \leq \alpha \cdot lb(V) (\leq \alpha \cdot w(x^\star))$, which is what α-approximation (with respect to the Held-Karp relaxation) would entail,
- the connectivity requirement is relaxed with respect to ATSP: instead of having F connect the entire graph (i.e., cross all cuts), we only specify $k \leq n$ cuts which it must cross,
- there is no such thing as an optimum solution to Local-Connectivity ATSP, which the algorithm's output could be compared to; because of this, we do not use the term *approximation* but rather define *lightness*. One could say that a variant of approximation is built into the definition of the problem.

[10] One way to make this formal is to define solving Local-Connectivity ATSP as a two-stage process: first, the algorithm is only given G and must output lb; then it is given the partitioning $V = V_1 \cup ... \cup V_k$ and it must output F.

[11] Like the approximation ratio β, the parameter α can be a constant or a function of n.

[12] To shorten notation, $\widetilde{G}$ is used here to denote both its vertex set and its edge set, as in: $w(\widetilde{G}) = w(E(\widetilde{G})) = \sum_{e \in E(\widetilde{G})} w(e)$, $\mathrm{lb}(\widetilde{G}) = \mathrm{lb}(V(\widetilde{G})) = \sum_{v \in V(\widetilde{G})} \mathrm{lb}(v)$.

[13] Svensson [Sve15] in fact defines Local-Connectivity ATSP somewhat differently, with a fixed lb function (and a suggestion to consider others), but we need to make the definition more general.

We have kept saying that the new problem is no harder; we now back this claim up with a proof.

Fact 2.11. *If there exists an α-approximation for ATSP with respect to the Held-Karp relaxation [on some class of graphs], then there also exists an α-light algorithm for Local-Connectivity ATSP [on that class of graphs].*

Proof. We can define lb in any way such that $\mathrm{lb}(V) = w(x^\star)$ [14] and produce F using the α-approximation for ATSP.[15] Let us verify that this gives an α-light algorithm. Since F crosses all cuts, it also crosses those induced by any given partitioning. As for lightness, we have only one connected component $\widetilde{G} = (V, F)$, and $w(\widetilde{G}) = w(F) \leq \alpha \cdot w(x^\star) = \alpha \cdot \mathrm{lb}(V) = \alpha \cdot \mathrm{lb}(\widetilde{G})$.□

However, it turns out that the two problems are actually equivalent up to constant factors! The following is the main result of Svensson [Sve15]:

Theorem 2.12. *If there exists an α-light algorithm for Local-Connectivity ATSP [on some class of graphs], then:*

- *the integrality gap of the Held-Karp relaxation [on that class of graphs] is at most 5α,*[16]
- *for any $\varepsilon > 0$ there is a $(9+\varepsilon)\alpha$-approximation with respect to the Held-Karp relaxation for ATSP [on that class of graphs].*[17]

The reduction of Theorem 2.12 is general and works for any weight function. Unfortunately, we do not know if there is a constant-light algorithm for the general case. However, there is a rather natural 3-light algorithm for unit weights. We prove the following in Section 2.5:

Theorem 2.13. *There exists a 3-light algorithm for Local-Connectivity ATSP on unweighted graphs.*[18]

By Theorem 2.12, this immediately implies:

Theorem 2.14. *The integrality gap of the Held-Karp relaxation for ATSP on unweighted graphs is at most 15. Furthermore, for any $\varepsilon > 0$ there is a $(27+\varepsilon)$-approximation for ATSP on unweighted graphs (and it is with respect to the Held-Karp relaxation).*

2.5 A constant-light algorithm for unit weights

In this section we prove Theorem 2.13. Our algorithm for zero-one weights is an extension of this one, and it heavily builds on the techniques introduced here.

We give an overview of the approach first. The function $\mathrm{lb}(v)$ will be set to the value that the fractional optimum $x^\star$ "pays" for edges going out of v, which is equal to the fractional degree of v in $x^\star$ since the graph is unweighted. Our task is to round $x^\star$ to integrality while roughly respecting vertex degrees (which will guarantee lightness) and making sure that each cut induced by the given partitioning is still crossed (fractionally, $x^\star$ crosses each cut with at least 1 unit of flow). We will use integrality of circulations (see Corollary 2.6) to accomplish this. Unfortunately, we cannot use it directly by requiring a lower bound on the flow over a *set*

[14] E.g. we can put $w(x^\star)$ on one vertex and 0 on the rest, or we can set $\mathrm{lb}(v) = \sum_{e \in \delta^+(v)} w_e x^\star_e$.

[15] We disregard the partitioning given to us.

[16] This holds even if the α-light algorithm does not run in polynomial time.

[17] Its running time may depend on $1/\varepsilon$.

[18] This algorithm actually works for a broader class of weight functions (*node-weighted metrics*), but we ignore this here.

of vertices, so instead we emulate the set with a single new vertex. More precisely, we modify the graph and $x^\star$ as follows: for every component V_i of the partitioning $V = V_1 \cup ... \cup V_k$, we add a new auxiliary vertex A_i, take 1 unit of flow passing through V_i, and redirect it to pass through A_i. We scale down flow inside every component suitably so as to maintain that $x^\star$ is still a circulation. Now we can use integrality of circulations to select an arbitrary integral solution y to the circulation problem with vertex-degree bounds $1 \leq y(\delta^+(v)) \leq \lceil x^\star(\delta^+(v)) \rceil$ (which is feasible since the modified $x^\star$ satisfies these bounds). Note that we have $y(\delta^+(A_i)) = 1$ for each $i = 1, ..., k$. Finally, we remove the auxiliary vertices, which forces us to reroute one path inside each component V_i; this violates the degree bounds only by 1.

We proceed to describe the algorithm in more detail. First, we are given a strongly connected digraph $G = (V, E, w)$ (where w is identically 1) and a Held-Karp optimum $x^\star$ for it. We define the lb function as follows: $\mathrm{lb}(v) = x^\star(\delta^+(v))$. It satisfies $\mathrm{lb}(V) = w(x^\star)$.

Now we are given a partitioning $V = V_1 \cup ... \cup V_k$. Our task is to produce an Eulerian multiset of edges $F \subseteq E$ which is 3-light and crosses all the V_i-cuts (i.e., satisfies $|F \cap \delta^+(V_i)| \geq 1$ for $i = 1, ..., k$). To get lightness, we will ensure that

$$|F \cap \delta^+(v)| \leq \lceil x^\star(\delta^+(v)) \rceil + 1 \quad \text{for each } v \in V. \tag{1}$$

If (1) is satisfied, then F is 3-light. To see this, note that for each $v \in V$

$$\lceil x^\star(\delta^+(v)) \rceil + 1 \leq 3x^\star(\delta^+(v))$$

because $x^\star(\delta^+(v)) \geq 1$, and thus for any connected component $\widetilde{G}$ of (V, F) we have

$$w(\widetilde{G}) = \sum_{v \in \widetilde{G}} |F \cap \delta^+(v)| \leq \sum_{v \in \widetilde{G}} \left(\lceil x^\star(\delta^+(v)) \rceil + 1 \right) \leq 3 \sum_{v \in \widetilde{G}} x^\star(\delta^+(v)) = 3\,\mathrm{lb}(\widetilde{G}).$$

So our solution F should satisfy (1) and cross all V_i-cuts.

We begin by transforming G and $x^\star$ into a new graph G' and a circulation $y^\star$ on G'. This is done so as to reroute 1 unit of flow passing through V_i (i.e., a $1/x^\star(\delta^+(V_i))$ fraction of that flow) to pass through a new auxiliary vertex A_i; flow inside V_i is scaled down by a factor $1 - \frac{1}{x^\star(\delta^+(V_i))}$ to obtain a circulation. Formally, we proceed as follows:

- each edge $e = (u, v)$ – for u, v being in different components V_i – is replaced by adding vertices O_e, I_e and edges $(u, O_e), (O_e, I_e), (I_e, v)$,
- for each component V_i we add an auxiliary vertex A_i and edges (A_i, O_e) for every $e \in \delta^+(V_i)$ and (I_e, A_i) for every $e \in \delta^-(V_i)$,
- for each edge e inside a component V_i set $y^\star_e = x^\star_e \cdot \left(1 - \frac{1}{x^\star(\delta^+(V_i))}\right)$,
- for each edge $e = (u, v)$ where $u \in V_i$, $v \in V_j$, $i \neq j$ set:

$$\begin{aligned}
y^\star_{(u,O_e)} &= x^\star_e \cdot \left(1 - \frac{1}{x^\star(\delta^+(V_i))}\right), \\
y^\star_{(A_i,O_e)} &= \frac{x^\star_e}{x^\star(\delta^+(V_i))}, \\
y^\star_{(O_e,I_e)} &= x^\star_e, \\
y^\star_{(I_e,v)} &= x^\star_e \cdot \left(1 - \frac{1}{x^\star(\delta^+(V_j))}\right), \\
y^\star_{(I_e,A_j)} &= \frac{x^\star_e}{x^\star(\delta^+(V_j))}.
\end{aligned}$$

By construction, the flow $y^\star$ is a nonnegative circulation in G' satisfying $y^\star(\delta^+(A_i)) = 1$ for each i. It also has $y^\star(\delta^+(v)) = \left(1 - \frac{1}{x^\star(\delta^+(V_i))}\right) \cdot x^\star(\delta^+(v)) \leq x^\star(\delta^+(v))$ for any $v \in V_i$. Therefore $y^\star$ is a fractional circulation satisfying the following integral bounds on vertices: $y^\star(\delta^+(v)) \leq \lceil x^\star(\delta^+(v)) \rceil$ for all $v \in V$ and $y^\star(\delta^+(A_i)) = 1$ for all i. By Corollary 2.6, we can find in polynomial time an integral circulation y satisfying these bounds.

The next step is to map back from G' to G. We do so by removing the vertices A_i (along with adjacent edges) and contracting all vertices O_e and I_e. The integral circulation y is mapped to a multiset F' of edges of G in the natural way. (To be more precise, a boundary edge $e \in \delta^+(V_i)$ is included in F' if the edge (O_e, I_e) was included in y.) Now, F' is almost Eulerian, except that each auxiliary vertex A_i had one incoming and one outgoing edge in y, and now (in F') these edges have been mapped back to boundary edges of V_i in such a way that the outdegree-indegree balance has been violated for their endpoints in V_i. To fix this, we add one path P_i inside V_i between the two affected vertices;[19] the multiset of edges $F = F' \cup \bigcup_{i=1}^{k} P_i$ is Eulerian.

Why is F a good solution? It satisfies (1) because y satisfied $y(\delta^+(v)) \leq \lceil x^\star(\delta^+(v)) \rceil$ and adding the paths P_i can only increase the degree of any vertex by 1, since they are disjoint. To see that each V_i-cut is crossed, let (A_i, O_e) be the only outgoing edge of A_i in the support of y; by construction, $y(O_e, I_e) \geq 1$ and thus $e \in F' \cap \delta^+(V_i)$.

This finishes our description of the algorithm.

3 Our results

In this section we present our main result: a constant-light algorithm for Local-Connectivity ATSP on graphs with zero-one weights. Namely, we prove:

Theorem 3.1. *There exists a 42-light algorithm for Local-Connectivity ATSP on graphs with zero-one weights.*

By Theorem 2.12, this immediately implies:

Theorem 3.2. *The integrality gap of the Held-Karp relaxation for ATSP on graphs with zero-one weights is at most 210. Furthermore, for any $\varepsilon > 0$ there is a $(378 + \varepsilon)$-approximation for ATSP on graphs with zero-one weights (and it is with respect to the Held-Karp relaxation).*

The remainder of the text (Sections 3 to 5) is devoted to the proof of Theorem 3.1.

Let us first discuss the main idea. In the unweighted setting, the property $x^\star(\delta^+(v)) \geq 1$ implied that each vertex "paid" at least 1 unit of cost, and so increasing its degree by only a constant resulted in a constant-light algorithm. Here, this is no longer the case: a vertex can have arbitrarily small flow over 1-weight edges, so that it cannot afford even a single 1-weight edge in a light solution. This forces us to group vertices together, so that they can support a constant cost. It turns out that we can group them into cycles: since each cycle contains at least one 1-weight edge (otherwise it can just be contracted), a suitable modification to the definition of lb allows the cycles to play the role that single vertices play in the unit-weight algorithm.

We proceed to describe the algorithm in detail. Again, given a strongly connected digraph $G = (V, E, w)$ with weight function $w : E \to \{0, 1\}$ and a fractional optimum $x^\star$, we want to solve Local-Connectivity ATSP on it. We describe the algorithm as a series of steps.

Step 3.1. *We can assume that G contains no cycles of weight 0.*

[19] One can show that without loss of generality each V_i is strongly connected, i.e., there is a path inside V_i between any two vertices of V_i.

Otherwise we can contract them all – at the end, after we uncontract and need to fix the Eulerian property of our solution, we can route paths on these cycles arbitrarily. We can also add each 0-cycle to the solution.

Step 3.2. *Find a cycle cover $\mathcal{C}$ of cost at most $w(x^\star)$.*

To do this, first find an integral circulation with the constraint that the degree of each vertex is at least 1. (Since $x^\star$ is such a fractional circulation, by Corollary 2.6 we can also find an integral one with no higher cost.) The support of this integral circulation decomposes into connected components, which are connected Eulerian subgraphs. We will assume without loss of generality that they are, in fact, simple cycles, and let $\mathcal{C}$ be the set of these cycles. At the end we will explain how to lift this assumption (see Remark 5.6).

A crucial property is that, since we assumed that there are no 0-weight cycles, each cycle in G (in particular, each cycle in $\mathcal{C}$) has positive weight (i.e., it contains at least one 1-weight edge).

Step 3.3. *Define the lower bound* lb*:*

$$\overline{\mathrm{lb}}(v) = \sum_{e\in\delta^+(v)} x^\star_e w_e + \sum_{e\in\delta^+(v)\cap\mathcal{C}} w_e,$$
$$\mathrm{lb}(v) = \overline{\mathrm{lb}}(v)/2.$$

That is, $\overline{\mathrm{lb}}(v)$ is the weighted outdegree of v in $x^\star+\chi_{\mathcal{C}}$. We have $\overline{\mathrm{lb}}(V) \le w(x^\star)+w(\mathcal{C}) \le 2w(x^\star)$. The function lb is a version of $\overline{\mathrm{lb}}$ scaled down so that $\mathrm{lb}(V) \le w(x^\star)$.

Crucially, since each cycle C is of positive weight, we get $\overline{\mathrm{lb}}(C) \ge w(C) \ge 1$.[20]

Step 3.4. *Now we are given a partition $V = V_1 \cup ... \cup V_k$. We begin by adding $\mathcal{C}$ to our solution. We can assume that no cycle from $\mathcal{C}$ crosses any V_i-cut.*

If a cycle does cross such a cut, we can e.g. merge the components it connects. Therefore each component V_i is a union of some set of cycles from $\mathcal{C}$.

Now, to obtain lightness, we are going to require that each cycle be light. This is enough:

Lemma 3.3. *Let F be an Eulerian subgraph of G with $F \supseteq \mathcal{C}$. Suppose that for each cycle $C \in \mathcal{C}$ it satisfies*

$$w(F \cap (E(C) \cup \delta^+(C))) \le \alpha \cdot \overline{\mathrm{lb}}(C).$$

Then F is α-light.

Proof. Since $F \supseteq \mathcal{C}$, each connected component $\widetilde{G}$ of (V, F) is (vertex-wise) the disjoint union of a set of cycles $\mathcal{C}' \subseteq \mathcal{C}$, therefore

$$w(\widetilde{G}) = \sum_{C\in\mathcal{C}'} w(F \cap (E(C) \cup \delta^+(C))) \le \alpha \cdot \sum_{C\in\mathcal{C}'} \overline{\mathrm{lb}}(C) = \alpha \cdot \overline{\mathrm{lb}}(\widetilde{G}).$$

□

Broadly, we follow the same strategy as the 3-light algorithm for unweighted graphs: rounding $x^\star$ to an integral circulation while forcing flow over boundaries of the components V_i.

[20]Note that $\overline{\mathrm{lb}}(v)$ can be arbitrarily low, but $\overline{\mathrm{lb}}(C) \ge 1$ – intuitively, this is why we group vertices into cycles.

4 Components V_i with $x^\star(\delta^+(V_i)) = 1$

Let us first outline how we can deal with components V_i that have small $x^\star$-flow over their boundaries. The general algorithm presented in Section 5 can be seen as a generalization of this. We describe this simplified case first mostly for intuition.

Here, we proceed as in the unit-weight algorithm: we reroute all flow to and from V_i through an auxiliary vertex A_i. This means that after the rerouting, there is no flow inside V_i (since $x^\star(\delta^+(V_i)) = 1$), and after an integral circulation is obtained, we are left with routing a single path P through V_i.

Our procedure is as follows: temporarily contract each cycle $C \in \mathcal{C}$ inside V_i and route any simple path between the required pair of vertices (such a path exists because V_i is strongly connected). Then uncontract the cycles and route the path inside the cycles in the natural way. Denote the cycles that appear on the path $C_1, ..., C_k$, in order; for $j = 1, ..., k$ let P_j be the fragment of C_j that the path traverses, and let e_j be the edge through which C_j is exited. Then the path is $P_1, e_1, ..., P_k, e_k$ (where $e_k \in \delta^+(V_i)$ is the exiting edge). All cycles C_j are distinct. Since for each j, $\overline{\mathrm{lb}}(C_j) \geq w(C_j) \geq 1$, we get

$$w(P_j) + w(e_j) + w(C_j) \leq 1 + 2w(C_j) \leq 3 \cdot \overline{\mathrm{lb}}(C_j),$$

which proves that for each cycle C_j we have (denoting the returned Eulerian subgraph by F)

$$\sum_{e \in E(C_j) \cup \delta^+(C_j)} \chi_F(e) \cdot w(e) \leq 3 \cdot \overline{\mathrm{lb}}(C_j) = 6 \cdot \mathrm{lb}(C_j).$$

Of course, if a cycle $C \in \mathcal{C}$ does not appear on any such path, then it also satisfies this bound (even with 2 in place of 6). Overall, by Lemma 3.3, we get a 6-light algorithm for this case.

Intuitively, some properties of this path that we have used to account for it are:

- it does not use chords of the cycles (edges between vertices of the cycle that are not part of the cycle),
- each cycle is visited only once, so it can pay for the (only) edge which exits it.

In this way, we could get constant lightness as long as $x^\star(\delta^+(V_i)) \leq \mathcal{O}(1)$ (by rerouting all this flow through A_i). However, neither of these properties seems to be easy to satisfy in the general case, so we need to be more careful there.

5 The general case

We will again endeavor to get the lightness claim from looking locally at each cycle and using Lemma 3.3.

We begin as usual: by redirecting 1 unit of flow over the boundary of each component to an auxiliary vertex A_i. At the end, we will need to remove the extra vertices and thus reroute one path inside each component V_i. This will ensure that all V_i-cuts are crossed by the returned set of edges.

We describe the routine as a series of steps.

Step 5.1. *We produce a weighted graph G' together with a circulation $y^\star$ on it in the same way as in the unit-weight algorithm (see Section 2.5).*[21]

[21] Weights in G' are defined in the obvious way.

Note that we still have $V_1, ..., V_k \subseteq V(G')$, although the new vertices A_i, I_e, O_e are not in any component V_i. The flow $y^\star$ is a nonnegative circulation in G' satisfying $y^\star(\delta^+(A_i)) = 1$ for each i.

The main difference is that now we are going to intercept all inflow and outflow of each cycle $C \in \mathcal{C}$ by replacing the cycle with a certain simple structure, and then round the resulting flow in a suitable way using integrality of circulations – we will call this outer-level rounding. After we map the selected set of edges back to the original graph, we will need to complete the flow inside each C to be a circulation; we do so using a second, local rounding – we call this inner-level rounding. Our way of "contracting" the cycle is designed so as to let us argue that repairing the flow inside the cycle later is cheap enough.

To make this more vivid, we have the following lemma that deals with inner-level rounding:

Lemma 5.1. *Let C be a cycle in G, and let D be a multiset of boundary edges of C, i.e., $D \subseteq \delta^+(C) \cup \delta^-(C)$, such that $|D \cap \delta^+(C)| = |D \cap \delta^-(C)|$. For each $v \in C$ define*

$$r_v := y^\star(\delta(v, V \setminus C)) - y^\star(\delta(V \setminus C, v)) - \chi_D(\delta(v, V \setminus C)) + \chi_D(\delta(V \setminus C, v)).$$

Now suppose that for each contiguous segment of the cycle $C' \subseteq C$ we have $\left|\sum_{v \in C'} r_v\right| \leq 16$. Then there exists a multiset of edges $F_C \subseteq E(C)$ satisfying

$$|(F_C \cup D) \cap \delta^+(v)| = |(F_C \cup D) \cap \delta^-(v)|$$

for every vertex $v \in C$ and having cost at most $\sum_{e \in E(C)} y^\star_e w_e + 16 \cdot w(C)$. It can be found in polynomial time.

Let us explain the statement. The multiset D is the set of boundary edges of C obtained from the outer-level rounding (which is going to replace $y^\star$ on these boundary edges). We are going to repair the degree balance of vertices in C by adding a suitable flow g to $y^\star|_{E(C)}$, and the quantities r_v are vertex demands for g. If their "prefix sums" are bounded, then we can obtain g by using only the cycle edges, and putting only a constant flow on each of them. Therefore what we want from the outer-level rounding is a multiset D which satisfies the condition of Lemma 5.1 and which is not much more expensive than $y^\star|_{\delta^+(C) \cup \delta^-(C)}$.

And actually, we will satisfy the prefix sum condition of Lemma 5.1 separately for inflow and outflow. That is:

Fact 5.2. *In the setting of Lemma 5.1, for each $v \in C$ define*

$$r^+_v := y^\star(\delta(v, V \setminus C)) - \chi_D(\delta(v, V \setminus C)),$$
$$r^-_v := y^\star(\delta(V \setminus C, v)) - \chi_D(\delta(V \setminus C, v)).$$

If for each contiguous segment C' of C we have $\left|\sum_{v \in C'} r^+_v\right|, \left|\sum_{v \in C'} r^-_v\right| \leq 8$, then $\left|\sum_{v \in C'} r_v\right| \leq 16$.

Proof. Use $r_v = r^+_v - r^-_v$ and $|a - b| \leq |a| + |b|$. □

Proof of Lemma 5.1. Let $v_1, ..., v_\ell$ be all vertices on the cycle C, in order (we will write $v_{\ell+1} = v_1$). First note that since $y^\star(\delta^+(C)) = y^\star(\delta^-(C))$ and $\chi_D(\delta^+(C)) = \chi_D(\delta^-(C))$, we have $r_{v_1} + ... + r_{v_\ell} = \sum_{v \in C} r_v = 0$. Define $M = -\min_{i=1,...,\ell}(r_{v_1} + ... + r_{v_i})$; note that $M \geq 0$ and for each $i = 1, ..., \ell$, $r_{v_1} + ... + r_{v_i} + M \geq 0$. We define the added flow g on the cycle edges as follows: $g_{(v_i, v_{i+1})} := r_{v_1} + ... + r_{v_i} + M$. Then:

- for each e, $0 \leq g_e \leq 16$ by the prefix sum assumption (note that $g_e = \left|r_{v_{i'}} + ... + r_{v_i}\right|$ for some i', i),

- the flow g satisfies the demands r, i.e., for $i = 1, ..., \ell$, $g(\delta^+(v_i)) - g(\delta^-(v_i)) = r_{v_i}$.

Now consider the flow $f := y^\star|_{E(C)} + g$ on $E(C)$.

Claim 1. *For each $v \in C$, $f(\delta^+(v)) - f(\delta^-(v)) = \chi_D(\delta^-(v)) - \chi_D(\delta^+(v))$.*

To simplify notation in the proof of the claim, we fix $v \in C$ and define, for any flow h, $\mathrm{bal}(h) := h(\delta^+(v)) - h(\delta^-(v))$. The operator bal is linear. We want to show that $\mathrm{bal}(f + \chi_D) = 0$. First note that

$$r_v = y^\star(\delta(v, V \setminus C)) - y^\star(\delta(V \setminus C, v)) - \mathrm{bal}(\chi_D)$$

because edges in D only enter/exit v from/to outside of C; and since $\mathrm{bal}(y^\star) = 0$, we have

$$y^\star(\delta(v, V \setminus C)) + y^\star(\delta(v, C - v)) = y^\star(\delta(V \setminus C, v)) + y^\star(\delta(C - v, v))$$

and so

$$r_v = -y^\star(\delta(v, C - v)) + y^\star(\delta(C - v, v)) - \mathrm{bal}(\chi_D) = -\mathrm{bal}(y^\star|_{E(C)}) - \mathrm{bal}(\chi_D).$$

Hence, since $\mathrm{bal}(g) = r_v$,

$$\mathrm{bal}(f + \chi_D) = \mathrm{bal}(y^\star|_{E(C)}) + \mathrm{bal}(g) + \mathrm{bal}(\chi_D) = 0,$$

which proves Claim 1.

Claim 2. *The cost of f is at most $\sum_{e \in E(C)} y^\star_e w_e + 16 \cdot w(C)$.*

This is easy to see: the first summand is the cost of $y^\star|_{E(C)}$, and the cost of g is at most $16 \cdot w(C)$ because $g_e \leq 16$ for all cycle edges e (and $g_e = 0$ for other edges).

To finish, we use (a straightforward extension of) integrality of circulations: since there exists a nonnegative fractional flow f of at most this cost satisfying vertex demands $\chi_D(\delta^-(v)) - \chi_D(\delta^+(v))$ (which are integral), there also exists a nonnegative integral circulation of at most this cost satisfying these demands, and thus we get the sought multiset of edges F_C. (This can be done in polynomial time.) □

Now we should come up with an outer-level rounding scheme which satisfies the prefix sum condition of Fact 5.2 and is not too expensive. We describe how to deal with outgoing edges; the incoming ones are dealt with identically. We will, roughly speaking, group all outflow of each $C \in \mathcal{C}$ into groups of size 1 unit, and choose 1 edge from each such group. To satisfy the prefix sum condition, the groups of edges should be more or less contiguous on the cycle; to be cheap, we should group edges of similar cost together.

Step 5.2. *For each cycle $C \in \mathcal{C}$ do the following. Let $v_1, ..., v_\ell$ be all vertices on the cycle C, in order. For each distinct edge cost c, we form a sequence of* groups *as follows. Order all outgoing edges in any way so that for $i < i'$, all edges going out of v_i come before all edges going out of $v_{i'}$. Begin with one empty group. For every edge e in this order do the following, with U being the last-added group of edges:*

- *if $\sum_{f \in U} y^\star_f + y^\star_e < 1$, then add e to U,*
- *if $\sum_{f \in U} y^\star_f + y^\star_e = 1$, then add e to U and form a new empty group,*
- *if $\sum_{f \in U} y^\star_f + y^\star_e > 1$, then replace e by two new parallel edges e' and e'', the first with $y^\star_{e'} = 1 - \sum_{f \in U} y^\star_f$ and the second with $y^\star_{e''} = y^\star_e - y^\star_{e'}$, add e' to U, form a new empty group, and recurse on e''.*

In this way we obtain a partition of $\delta^+(C)$ into a collection $\mathcal{U}$ of groups such that:

- for each $U \in \mathcal{U}$ save at most two (one per distinct edge cost), $\sum_{f\in U} y^\star_f = 1$ (and $\sum_{f\in U} y^\star_f \in [0,1)$ for the remaining ones),
- for each $i = 1, ..., \ell$, there are at most two $U \in \mathcal{U}$ (one per distinct edge cost) whose set of tails $\{\text{tail}(e) : e \in U\}$ intersects both the set $\{v_1, ..., v_i\}$ and its complement $\{v_{i+1}, ..., v_\ell\}$,
- each $U \in \mathcal{U}$ contains edges of one cost.

Let us call the $U \in \mathcal{U}$ with $\sum_{f\in U} y^\star_f \in [0,1)$ *special*. By the first property above, there are at most two special U (one per distinct edge cost). This gives us:

Lemma 5.3. *Let D be any subset of $\delta^+(C)$ such that for each $U \in \mathcal{U}$,*

- $|D \cap U| = 1$ *if U is not special,*
- $|D \cap U| \in \{0, 1\}$ *if U is special.*

Recall that $r^+_v = y^\star(\delta(v, V \setminus C)) - \chi_D(\delta(v, V \setminus C))$. Then:

- *for each contiguous fragment C' of C we have $\left|\sum_{v\in C'} r^+_v\right| \le 8$, and*
- $\sum_{e\in D} w(e) \le \sum_{e\in\delta^+(C)} y^\star_e w(e) + 2$.

Proof. For the first part, it is enough to show that for all $i = 1, ..., \ell$, $\left|r^+_{v_1} + ... + r^+_{v_i}\right| \le 4$. Note that

$$\begin{aligned} r^+_{v_1} + ... + r^+_{v_i} &= y^\star(\delta(\{v_1, ..., v_i\}, V \setminus C)) - \chi_D(\delta(\{v_1, ..., v_i\}, V \setminus C)) \\ &= \sum_{U\in\mathcal{U}} \left[y^\star(U \cap \delta(\{v_1, ..., v_i\}, V \setminus C)) - \chi_D(U \cap \delta(\{v_1, ..., v_i\}, V \setminus C))\right]. \end{aligned}$$

Now, the term in the square bracket is 0 unless U is special or its set of tails intersects both $\{v_1, ..., v_i\}$ and $\{v_{i+1}, ..., v_\ell\}$.[22] But by the first and second properties above, there are at most four such U, and for any U, the term in the parenthesis is in $[-1, 1]$. This proves the claim.

For the second part, we argue that only special groups U can bring imbalance to the quantity

[22] Otherwise, either $\{\text{tail}(e) : e \in U\} \subseteq \{v_1, ..., v_i\}$, in which case the term is equal to $y^\star(U) - \chi_D(U) = 1 - 1 = 0$, or $\{\text{tail}(e) : e \in U\} \cap \{v_1, ..., v_i\} = \emptyset$, in which case the term is equal to $0 - 0 = 0$.

$\sum_{e\in D} w(e) - \sum_{e\in\delta^+(C)} y^\star_e w(e)$, and each of them brings at most 1 unit of imbalance. Formally,

$$\begin{aligned}
\sum_{e\in D} w(e) - \sum_{e\in\delta^+(C)} y^\star_e w(e) &= \sum_{e\in\delta^+(C)} (\chi_D(e)w(e) - y^\star_e w(e)) \\
&= \sum_{c\,:\,\text{edge cost}} c\cdot\left[\sum_{\substack{e\in\delta^+(C):\\ w(e)=c}} (\chi_D(e) - y^\star_e)\right] \\
&= \sum_{c\,:\,\text{edge cost}} c\cdot\left[\sum_{\substack{U\in\mathcal{U}:\\ U \text{ has edges of cost } c}} (\chi_D(U) - y^\star(U))\right] \\
&= \sum_{c\,:\,\text{edge cost}} c\cdot\left[\sum_{\substack{U\in\mathcal{U}:\\ U \text{ has edges of cost } c\\ \text{and } U \text{ is special}}} (\chi_D(U) - y^\star(U))\right] \\
&\le \sum_{c\,:\,\text{edge cost}} c\cdot 1 \\
&\le 2
\end{aligned}$$

since there is at most one special U for any c. □

The situation for incoming edges is exactly analogous.

Step 5.3. *For each cycle $C \in \mathcal{C}$, let $\mathcal{U}_C$ be the collection of groups obtained as above from outgoing edges, and let $\mathcal{V}_C$ be the collection of groups of incoming edges (same as above, replacing outgoing with incoming, tail with head, δ^+ with δ^- etc.).*

And we have the following analogue of Lemma 5.3.

Lemma 5.4. *Let D be any subset of $\delta^-(C)$ such that for each $U \in \mathcal{V}_C$,*

- *$|D \cap U| = 1$ if U is not special,*
- *$|D \cap U| \in \{0, 1\}$ if U is special.*

Recall that $r^-_v = y^\star(\delta(V \setminus C, v)) - \chi_D(\delta(V \setminus C, v))$. Then for each contiguous fragment C' of C we have $\left|\sum_{v\in C'} r^-_v\right| \le 8$.[23] □

So how do we select one edge per group? Basically, we redirect all edges in a group to a new auxiliary vertex (one per group), and then connect all the incoming-group vertices with outgoing-group vertices in a rather arbitrary manner.

Step 5.4. *We construct a digraph G'', a circulation y' on G'', and a one-to-one mapping b from a subset of $E(G')$ to $E(G'')$* [24] *as follows:*

- *for each outgoing group $U \in \bigcup_{C\in\mathcal{C}} \mathcal{U}_C$ we create a vertex Out_U,*

[23] We do not care about the cost here since we need only bound it for outgoing edges.
[24] The mapping b is only used for formalities.

- *for each incoming group* $U' \in \bigcup_{C \in \mathcal{C}} \mathcal{V}_C$ *we create a vertex* $In_{U'}$,
- *for each component* V_i *we create a vertex* A_i,
- *for each* $e \in G$ *going between two different components* V_i*, we create vertices* I_e *and* O_e,
- *for each edge* $e = (u, v) \in E(G')$*:*
  - *if* u *and* v *lie on the same cycle* $C \in \mathcal{C}$*: do nothing (i.e., don't include the edge in* G''*),*
  - *if* u *and* v *are in the same component* V_i *but on different cycles* C, C'*, then redirect both endpoints of the edge, i.e.: let* $U \in \mathcal{U}_C$ *be the outgoing group of* e*, and let* $U' \in \mathcal{V}_{C'}$ *be the incoming group of* e*; create an edge* $e' = (Out_U, In_{U'})$ *in* G''*, put* $y'_{e'} = y^\star_e$*, and put* $b(e) = e'$,
  - *if* e *is of the form* $(O_{e'}, I_{e'})$ *or* $(I_{e'}, A_i)$ *or* $(A_i, O_{e'})$ *(for some* $e' \in G$ *and* i*), then copy* e *to* G'' *verbatim (i.e., create an edge* e *in* G''*, put* $y'_e = y^\star_e$ *and* $b(e) = e$*),*
  - *if* $u = I_{e'}$ *(for some* $e' \in G$*) and* v *belongs to a cycle* $C' \in \mathcal{C}$*, then redirect one endpoint of the edge, i.e.: let* $U' \in \mathcal{V}_{C'}$ *be the incoming group of* e*; create an edge* $e'' = (I_{e'}, In_{U'})$ *in* G''*, put* $y'_{e''} = y^\star_e$*, and* $b(e) = e''$,
  - *if* $v = O_{e'}$ *(for some* $e' \in G$*) and* u *belongs to a cycle* $C \in \mathcal{C}$*, then redirect one endpoint of the edge, i.e.: let* $U \in \mathcal{U}_C$ *be the outgoing group of* e*; create an edge* $e'' = (Out_U, O_{e'})$ *in* G''*, put* $y'_{e''} = y^\star_e$*, and* $b(e) = e''$,
- *for each cycle* $C \in \mathcal{C}$*:*
  - *create a vertex* Cyc_C,
  - *for each outgoing group* $U \in \mathcal{U}_C$*, create an edge* $e' = (Cyc_C, Out_U)$ *with* $y'_{e'} = \sum_{f \in U} y^\star_f$,
  - *for each incoming group* $U' \in \mathcal{V}_C$*, create an edge* $e' = (In_{U'}, Cyc_C)$ *with* $y'_{e'} = \sum_{f \in U'} y^\star_f$.

It is clear from the construction that y' is a circulation in G'' (for vertices In_U and Out_U the inflow and outflow are both equal to $\sum_{f \in U} y^\star_f$, and for Cyc_C they are equal to $y^\star(\delta^+(C)) = y^\star(\delta^-(C))$). Moreover it clearly satisfies:

Claim 3. *For each outgoing or incoming group* U*, if* U *is not special, then* $y'(\delta^+(v_U)) = y'(\delta^-(v_U)) = 1$ *where* $v_U = In_U$ *or* Out_U*, and if* U *is special, then this quantity is in* $[0, 1]$*. And for each* i *we have* $y'(\delta^+(A_i)) = 1$*.*

Therefore by integrality of circulations, we can also find such an integral circulation.

Step 5.5. *Find an integral circulation* z *in* G'' *which satisfies the conditions of Claim 3. Then map it back to a multiset* Z *of edges of* G' *in the natural way. Formally, we use the map* b*, i.e., define* $\chi_Z(e) := z_{b(e)}$ *(or* 0 *if* b *is not defined for* e*).*

The conditions of Claim 3 translate to Z and G' as follows: for each outgoing or incoming group U, if U is not special, then $|Z \cap U| = 1$, and if U is special, then $|Z \cap U| \in \{0, 1\}$. And each auxiliary vertex A_i has exactly one incoming and one outgoing edge in Z.

This means that for each cycle $C \in \mathcal{C}$, the set $D^+ := Z \cap \delta^+(C)$ satisfies the conditions of Lemma 5.3. Same applies to the set $D^- := Z \cap \delta^-(C)$ and Lemma 5.4. We have $|D^+| = |D^-|$ since z is a circulation and this size is the degree of Cyc_C in it. Thus by Fact 5.2 and Lemma 5.1 (applied to $D := D^+ \cup D^-$) we can:

Step 5.6. *Using Lemma 5.1 obtain, for each cycle $C \in \mathcal{C}$, a multiset of edges $F_C \subseteq E(C)$ such that the multiset $F' := Z \cup \bigcup_{C \in \mathcal{C}} F_C$ is Eulerian.*

And finally:

Step 5.7. *Remove the vertices A_i, and reroute one path per component V_i as in Section 4.*[25] *Return the Eulerian subset $F := F' \cup P \cup \mathcal{C}$ of G, where P is the union of the newly added paths.*

We are now left only with bounding the cost of F. Recall that we need to show that the condition of Lemma 3.3 holds. Since we proved this for $P \cup \mathcal{C}$ in Section 4, we only concentrate on F' here.

We can bound the cost of internal edges for any $C \in \mathcal{C}$ by Lemma 5.1:

$$\sum_{e \in E(C)} \chi_{F'}(e) w(e) = \sum_{e \in E(C)} \chi_{F_C}(e) w(e) \leq \sum_{e \in E(C)} y^\star_e w(e) + 16 \cdot w(C).$$

For boundary edges, by Lemma 5.3 (with $D^+ = Z \cap \delta^+(C) = F' \cap \delta^+(C)$) we have:

$$\sum_{e \in \delta^+(C)} \chi_{F'}(e) w(e) = \sum_{e \in D^+} w(e) \leq \sum_{e \in \delta^+(C)} y^\star_e w(e) + 2.$$

Summing these two, we get

$$\begin{aligned}
\sum_{e \in E(C) \cup \delta^+(C)} \chi_{F'}(e) w(e) &\leq \sum_{e \in E(C) \cup \delta^+(C)} y^\star_e w_e + 2 + 16 \cdot w(C) \\
&\leq 18 \cdot \left[\sum_{e \in E(C) \cup \delta^+(C)} y^\star_e w(e) + w(C) \right] \\
&\leq 18 \cdot \overline{\mathrm{lb}}(C) \\
&= 36 \cdot \mathrm{lb}(C),
\end{aligned}$$

where the second inequality follows from $w(C) \geq 1$. Recall from Section 4 that $P \cup \mathcal{C}$ adds another $6 \cdot \mathrm{lb}(C)$ to this. In total, by Lemma 3.3, our algorithm is 42-light. This ends the proof of Theorem 3.1.

Remark 5.5. *Note that the full algorithm can be seen as a generalization of the algorithm of Section 4, since if $x^\star(\delta^+(V_i)) = 1$, then $y^\star(\delta^+(V_i)) = 0$ and nothing else happens in V_i until a single path is rerouted in Step 5.7.*

Remark 5.6. *So what if a cycle $C \in \mathcal{C}$ is not actually a simple cycle, but only an Eulerian subgraph? Then we can (after Step 3.2) unwind it by creating a new artifical vertex v' every time a vertex v appears on the cycle not for the first time. All chords and boundary edges of C should still point to the original vertex; the new vertices will have only one incoming and one outgoing edge. Once we receive the partition $V = V_1 \cup \ldots \cup V_k$, we suitably modify V_i to make it still include the unwound C. Note that $x^\star$ is probably no longer a Held-Karp solution for the modified graph, but we actually only use that it is a circulation with $x^\star(\delta^+(V_i)) \geq 1$ for each i, which it still satisfies. Next, we run the algorithm. After it finishes, we rewind the cycle onto the original Eulerian subgraph and map the returned edges in F in the natural way.*

[25] And then contract all vertices I_e and O_e, thus reverting to the graph G.

6 Further directions

The next logical step would seem be to obtain a constant-factor approximation algorithm for graphs with any two different edge weights. At the time of writing, this has already been done by Svensson, T. and Végh [STV16]. However, it is not clear how to extend their techniques (or those presented in this thesis) to get a constant-factor approximation even for three weights, and this would be an interesting development. A further extension could be an $\mathcal{O}(\beta(k))$-approximation for k different edge weights (for some β being only a function of k). Finally, the big remaining question is to understand the power of the Held-Karp relaxation for ATSP and see if it can provide a constant guarantee in the general case. Although it would be arguably even more curious to obtain a good algorithm which does *not* use this linear programming relaxation...

References

[AG15] Nima Anari and Shayan Oveis Gharan. Effective-resistance-reducing flows and asymmetric TSP. In *Proceedings of the 56th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*. IEEE, 2015.

[AGM+10] Arash Asadpour, Michel X. Goemans, Aleksander Madry, Shayan Oveis Gharan, and Amin Saberi. An O(log n/ log log n)-approximation algorithm for the asymmetric traveling salesman problem. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2010*, pages 379–389, 2010.

[CGK06] Moses Charikar, Michel X. Goemans, and Howard J. Karloff. On the integrality ratio for the asymmetric traveling salesman problem. *Math. Oper. Res.*, 31(2):245–252, 2006.

[Chr76] Nicos Christofides. Worst-case analysis of a new heuristic for the travelling salesman problem. Technical report, DTIC Document, 1976.

[FGM82] Alan M. Frieze, Giulia Galbiati, and Francesco Maffioli. On the worst-case performance of some algorithms for the asymmetric traveling salesman problem. *Networks*, 12(1):23–39, 1982.

[FGS13] Zachary Friggstad, Anupam Gupta, and Mohit Singh. An improved integrality gap for asymmetric TSP paths. *CoRR*, abs/1302.3145, 2013. URL: `http://arxiv.org/abs/1302.3145`.

[GSS11] Shayan Oveis Gharan, Amin Saberi, and Mohit Singh. A randomized rounding approach to the traveling salesman problem. In *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011*, pages 550–559, 2011.

[KLS15] Marek Karpinski, Michael Lampis, and Richard Schmied. New inapproximability bounds for TSP. *J. Comput. Syst. Sci.*, 81(8):1665–1677, 2015.

[LRS11] Lap Chi Lau, Ramamoorthi Ravi, and Mohit Singh. *Iterative methods in combinatorial optimization*, volume 46. Cambridge University Press, 2011.

[MS11] Tobias Mömke and Ola Svensson. Approximating graphic TSP by matchings. In *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011*, pages 560–569, 2011.

[Muc12] Marcin Mucha. 13/9-approximation for graphic TSP. In *29th International Symposium on Theoretical Aspects of Computer Science, STACS 2012*, pages 30–41, 2012.

[Sch03] Alexander Schrijver. *Combinatorial Optimization - Polyhedra and Efficiency*. Springer-Verlag, Berlin, 2003.

[STV16] Ola Svensson, Jakub Tarnawski, and László A. Végh. Constant factor approximation for ATSP with two edge weights. In *Integer Programming and Combinatorial Optimization (IPCO)*, Lecture Notes in Computer Science. Springer, 2016. URL: `http://arxiv.org/abs/1511.07038`.

[SV14] András Sebő and Jens Vygen. Shorter tours by nicer ears: 7/5-approximation for the graph-TSP, 3/2 for the path version, and 4/3 for two-edge-connected subgraphs. *Combinatorica*, 34(5):597–629, 2014.

[Sve15] Ola Svensson. Approximating ATSP by relaxing connectivity. In *Proceedings of the 56th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2015. URL: `http://arxiv.org/abs/1502.02051`.