2015 IEEE 56th Annual Symposium on Foundations of Computer Science

Approximating ATSP by Relaxing Connectivity

Ola Svensson
School of Computer and Communication Sciences
EPFL, Switzerland
Email: ola.svensson@epfl.ch

Abstract

The standard LP relaxation of the asymmetric traveling salesman problem has been conjectured to have a constant integrality gap in the metric case. We prove this conjecture when restricted to shortest path metrics of node-weighted digraphs. Our arguments are constructive and give a constant factor approximation algorithm for these metrics. We remark that the considered case is more general than the directed analog of the special case of the symmetric traveling salesman problem for which there were recent improvements on Christofides' algorithm.

The main idea of our approach is to first consider an easier problem obtained by significantly relaxing the general connectivity requirements into local connectivity conditions. For this relaxed problem, it is quite easy to give an algorithm with a guarantee of 3 on node-weighted shortest path metrics. More surprisingly, we then show that *any* algorithm (irrespective of the metric) for the relaxed problem can be turned into an algorithm for the asymmetric traveling salesman problem by only losing a small constant factor in the performance guarantee. This leaves open the intriguing task of designing a "good" algorithm for the relaxed problem on general metrics.

Keywords

approximation algorithms; asymmetric traveling salesman problem; combinatorial optimization;

I. Introduction

The traveling salesman problem is one of the most fundamental combinatorial optimization problems. Given a set V of n cities and a distance/weight function $w : V \times V \to \mathbb{R}^+$, it is the problem of finding a tour of minimum total weight that visits each city exactly once. There are two variants of this general definition: the *symmetric* traveling salesman problem (STSP) and the *asymmetric* traveling salesman problem (ATSP). In the symmetric version we assume $w(u, v) = w(v, u)$ for each pair $u, v \in V$ of cities; whereas we make no such assumption in the more general asymmetric traveling salesman problem.

In both versions, it is common to assume the triangle inequality and we shall do so in the rest of this paper. Recall that the triangle inequality says that for any triple i, j, k of cities, we have $w(i, j) + w(j, k) \geq w(i, k)$. In other words, it is not more expensive to take the direct path compared to a path that makes a detour. Another equivalent view of the triangle inequality is that, instead of insisting that each city is visited exactly once, we should find a tour that visits each city *at least* once. These assumptions are arguably natural in many, if not most, settings. They are also necessary in the following sense: any reasonable approximation algorithm (with approximation guarantee $O(\exp(n))$) for the traveling salesman problem without the triangle inequality would imply $P = NP$ because it would solve the problem of deciding Hamiltonicity.

Understanding the approximability of the symmetric and the asymmetric traveling salesman problem (where we have the triangle inequality) turns out to be a much more interesting and notorious problem. On the one hand, the strongest known inapproximability results, by Karpinski, Lampis, and Schmied [1], say that it is NP-hard to approximate STSP within a factor of $123/122$ and that it is NP-hard to approximate ATSP within a factor of $75/74$. On the other hand, the current best approximation algorithms are far from these guarantees, especially in the case of ATSP.

For the symmetric traveling salesman problem, Christofides' beautiful algorithm from 1976 sill achieves the best known approximation guarantee of 1.5 [2]. However, a recent series of papers [3], [4], [5], [6], broke this

0272-5428/15 $31.00 © 2015 IEEE
DOI 10.1109/FOCS.2015.10

barrier for the interesting special case of shortest path metrics of unweighted undirected graphs[1]. Specifically, Oveis Gharan, Saberi, and Singh [3] first gave an approximation guarantee of $1.5 - \epsilon$; Mömke and Svensson [4] proposed a different approach yielding a 1.461-approximation guarantee; Mucha [5] gave a tighter analysis of this algorithm; and Sebö and Vygen [6] significantly developed the approach to give the current best approximation guarantee of 1.4.

The interest in shortest path metrics has several motivations. It is a natural special case that seems to capture the difficulty of the problem: it remains APX-hard and the worst known integrality gap for the Held-Karp relaxation is of this type. Moreover, it has an attractive graph theoretic formulation: given an unweighted graph, find a shortest tour that visits each vertex at least once. This is the (unweighted) "graph" analog of STSP. Indeed, if allow the graph to be edge-weighted, this formulation is equivalent to STSP on general metrics. Let us also mention that the polynomial time approximation scheme for the symmetric traveling salesman problem on planar graphs was first obtained for the special case of unweighted graphs [7], i.e., when restricted to shortest path metrics of unweighted graphs, and then generalized to the case of edge-weights [8]. For STSP, it remains a major open problem whether the ideas in [3], [4], [5], [6] can be applied to general metrics. We further discuss this in Section VI.

The gap in our understanding is much larger for the asymmetric traveling salesman problem for which it remains a notorious open problem to design an algorithm with *any* constant approximation guarantee. This is a particularly intriguing as the standard linear programming relaxation, often referred to as the Held-Karp relaxation, is only known to have an integrality gap of at least 2 [9]. There are in general two available approaches for designing approximation algorithms for ATSP in the literature. The first approach is due to Frieze, Galbiati, and Maffiolo [10] who gave a $\log_2(n)$-approximation algorithm for ATSP already in 1982. Their basic idea is simple and elegant: a minimum weight cycle cover has weight at most that of an optimal tour and it will decrease the number of connected components by a factor of at least 2. Hence, if we repeat the selection of a minimum weight cycle cover $\log_2(n)$ times, we get a connected Eulerian graph which (by shortcutting) is a $\log_2(n)$-approximate tour. Although the above analysis is tight only in the case when almost all cycles in the cycle covers have length 2, it is highly non-trivial to refine the method to decrease the number of iterations. It was first in 2003 that Bläser [11] managed to give an approximation guarantee of $0.999 \log_2(n)$. This was improved shortly thereafter by Kaplan, Lewenstein, Shafrir and Sviridenko [12] who further developed this approach to obtain a $4/3 \log_3(n) \approx 0.84 \log_2(n)$-approximation algorithm; and later by Feige and Singh [13] who obtained an approximation guarantee of $2/3 \log_2(n)$.

A second approach was more recently proposed in an influential and beautiful paper by Asadpour, Goemans, Madry, Oveis Gharan, and Saberi [14] who gave an $O(\log n/ \log\log n)$-approximation algorithm for ATSP. Their approach is based on finding a so-called α-thin spanning tree which is a (unweighted) graph theoretic problem. Here, the parameter α is proportional to the approximation guarantee so $\alpha = O(\log n/ \log\log n)$ in [14]. Following their publication, Oveis Gharan and Saberi [15] gave an efficient algorithm for finding $O(1)$-thin spanning trees for planar and bounded genus graphs yielding a constant factor approximation algorithm for ATSP on these graph classes. Also, in a very recent major progress, Anari and Oveis Gharan [16] showed the existence of $O(\text{polylog} \log n)$-thin spanning trees for general instances. This implies a $O(\text{polylog} \log n)$ upper bound on the integrality gap of the Held-Karp relaxation. Hence, it gives an efficient so-called estimation algorithm for estimating the optimal value of a tour within a factor $O(\text{polylog} \log n)$ but, as their arguments are non-constructive, no approximation algorithm for finding a tour of matching guarantee. The result in [16] is based on developing and extending several advanced techniques. Notably, they rely on their extension [17] of the recent proof of the Kadison-Singer conjecture which was a major breakthrough by Marcus, Spielman, and Srivastava [18].

To summarize, the current best approximation algorithm has a guarantee of $O(\log n/ \log\log n)$ [14] and the

[1]The shortest path metric of a graph $G = (V, E)$ is defined as follows: the weight $w(u, v)$ between cities $u, v \in V$ equals the shortest path between u and v in G. If the graph is node-weighted $f : V \to \mathbb{R}^+$, the weight/length of an edge $\{u, v\} \in E$ is $f(u) + f(v)$.

best upper bound on the integrality gap of the Held-Karp relaxation is $O(\text{polylog}\log n)$ [16]. These two bounds are far away from the known inapproximability results [1] and from the lower bound of 2 on the integrality gap of the Held-Karp relaxation [9]. Moreover, there were no better approximation algorithms known in the case of shortest path metrics of unweighted digraphs for which there was recent progress in the undirected setting. In particular, it is not clear how to use the two available approaches mentioned above to get an improved approximation guarantee in this case: in the cycle cover approach, the main difficulty is to bound the number of iterations and, in the thin spanning tree approach, ATSP is reduced to an unweighted graph theoretic problem.

A. Our Results and Overview of Approach

We propose a new approach for approximating the asymmetric traveling salesman problem based on relaxing the global connectivity constraints into local connectivity conditions. We also use this approach to obtain the following result where we refer to ATSP on shortest path metrics of node-weighted digraphs as Node-Weighted ATSP.

Theorem I.1. *There is a constant approximation algorithm for Node-Weighted ATSP. Specifically, for Node-Weighted ATSP, the integrality gap of the Held-Karp relaxation is at most* 15 *and, for any* $\epsilon > 0$*, there is a polynomial time algorithm that finds a tour of weight at most* $(27+\epsilon)\,\mathrm{OPT}_{HK}$ *where* OPT_{HK} *denotes the optimal value of the Held-Karp relaxation.*

As further discussed in Section VI, the constants in the theorem can be slightly improved by specializing our general approach to the node-weighted case. However, it remains an interesting open problem to give a tight bound on the integrality gap.

Let us continue with a brief overview of our approach that is not restricted to the node-weighted version. It is illustrative to consider the following "naive" algorithm that actually was the starting point of this work:

1) Select a random cycle cover C using the Held-Karp relaxation.
   It is well known that one can sample such a cycle cover C of expected weight equal to the optimal value OPT_{HK} of the Held-Karp relaxation.
2) While there exist more than one component, add the lightest cycle (i.e., the cycle of smallest weight) that decreases the number of components.

It is clear that the above algorithm always returns a solution to ATSP: we start with a Eulerian graph[2] and the graph stays Eulerian during the execution of the while-loop which does not terminate until the graph is connected. This gives a tour that visits each vertex at least once and hence a solution to ATSP (using that we have the triangle-inequality). However, what is the weight of the obtained tour? First, as remarked above, we have that the expected weight of the cycle cover is OPT_{HK}. So if C contains $k = |C|$ cycles, we would expect that a cycle in C has weight OPT_{HK}/k (at least on average). Moreover, the number of cycles added in Step 2 is at most $k-1$ since each cycle decreases the number of components by at least one. Thus, if each cycle in Step 2 has weight at most the average weight OPT_{HK}/k of a cycle in C, we obtain a 2-approximate tour of weight at most $\mathrm{OPT}_{HK} + \frac{k-1}{k}\mathrm{OPT}_{HK} \leq 2\,\mathrm{OPT}_{HK}$.

Unfortunately, it seems hard to find a cycle cover C so that we can always connect it with light cycles. Instead, what we can do, is to first select a cycle cover C then add light cycles that decreases the number of components as long as possible. When there are no more light cycles to add, the vertices/cities are partitioned into $V_1, \ldots, V_k$ connected components. In order to make progress from this point, we would like to find a "light" Eulerian set F of edges that crosses the cuts $\{(V_i, \bar{V}_i) \mid i = 1, 2, \ldots, k\}$. We could then hope to add F to our solution and continue from there. It turns out that it is very important what "light" means in this context. For our arguments to work, we need that F is selected so that the weight of the edges in each component has weight at most α times what the linear programming solution "pays" for the vertices in that component. This is the intuition behind the definitions in Section III of Local-Connectivity ATSP and α-light algorithms for that problem. We

[2]Recall that a directed graph is Eulerian if the in-degree equals the out-degree of each vertex.

also need to be very careful in which way we add edges from light cycles and how to use the α-light algorithm for Local-Connectivity ATSP. In Section V, our algorithm will iteratively solve the Local-Connectivity ATSP and, in each iteration, it will add a carefully chosen subset of the found edges together with light cycles.

We remark that in Local-Connectivity ATSP we have relaxed the global connectivity properties of ATSP into local connectivity conditions that only say that we need to find a Eulerian set of edges that crosses at most $n = |V|$ cuts defined by a partitioning of the vertices. In spite of that, we are able to leverage the intuition above to obtain our main technical result:

Theorem (Simplified statement of Theorem V.1)**.** *The integrality gap of the Held-Karp relaxation is at most* 5α *if there exists an* α*-light algorithm* $\mathcal{A}$ *for Local-Connectivity ATSP. Moreover, for any* $\epsilon > 0$*, we can find a* $(9+\epsilon)\alpha$*-approximate tour in time polynomial in* $n, 1/\epsilon$*, and in the running time of* $\mathcal{A}$*.*

The proof of the above theorem (Section V) is based on generalizing and, as alluded to above, deviating from the above intuition in several ways. First, we start with a carefully chosen "Eulerian partition" which generalizes the role of the cycle cover C in Step 1 above. Second, both the iterative use of the α-light algorithm for Local-Connectivity ATSP and the way we add light cycles are done in a careful and dependent manner so as to be able to bound the total weight of the returned solution. Theorem I.1 follows from Theorem V.1 together with a 3-light algorithm for Node-Weighted Local-Connectivity ATSP. The 3-light algorithm, described in Section IV, is a rather simple application of classic theory of flows and circulations. We also remark that it is the only part of the paper that relies on having shortest path metrics of node-weighted digraphs.

Our work raises several natural questions. Perhaps the most immediate and intriguing question is whether there is a $O(1)$-light algorithm for Local-Connectivity ATSP on general metrics. We further elaborate on this and other related questions in Section VI.

II. Preliminaries

A. Basic Notation

Consider a directed graph $G = (V, E)$. For a subset $S \subseteq V$, we let $\delta^+(S) = \{(u, v) \in E \mid u \in S, v \notin S\}$ be the outgoing edges and we let $\delta^-(S) = \{(u, v) \in E \mid u \notin S, v \in S\}$ be the incoming edges of the cut defined by S. When considering a subset $E' \subseteq E$ of the edges, we denote the restrictions to that subset by $\delta^+_{E'}(S) = \delta^+(S) \cap E'$ and by $\delta^-_{E'}(S) = \delta^-(S) \cap E'$. We also let $\mathcal{C}(E') = \{\tilde{G}_1 = (\tilde{V}_1, E_1), \tilde{G}_2 = (\tilde{V}_2, \tilde{E}_2), \ldots, \tilde{G}_k = (\tilde{V}_k, \tilde{E}_k)\}$ denote the set of subgraphs corresponding to the k connected components of the graph (V, E'); the vertex set V will always be clear from the context. Here connected means that the subgraphs are connected if we undirect the edges.

When considering a function $f : U \to \mathbb{R}$, we let $f(X) = \sum_{x \in X} f(x)$ for $X \subseteq U$. For example, if G is edge weighted, i.e., there exists a function $w : E \to \mathbb{R}$, then $w(E')$ denotes the total weight of the edges in $E' \subseteq E$. Similarly, if G is node-weighted, then there exists a function $f : V \to \mathbb{R}$ and $f(S)$ denotes the total weight of the vertices in $S \subseteq V$. When talking about graphs, we shall slightly abuse notation and sometimes write $w(G)$ instead of $w(E)$ and $f(G)$ instead of $f(V)$ when it is clear from the context that w and f are functions on the edges and vertices. Finally, our *subsets of edges are multisets*, i.e., may contain the same edge several times. The set operators $\cup, \cap, \setminus$ are defined in the natural way. For example, $\{e_1, e_1, e_2\} \cup \{e_1, e_2\} = \{e_1, e_1, e_1, e_2, e_2\}$, $\{e_1, e_1, e_2\} \cap \{e_1, e_2\} = \{e_1, e_2\}$, and $\{e_1, e_1, e_2\} \setminus \{e_1, e_2\} = \{e_1\}$. Other sets, such as subsets of vertices, will always be simple sets without any multiplicities.

B. The (Node-Weighted) Asymmetric Traveling Salesman Problem

It will be convenient to define ATSP using the Eulerian point of view, i.e., we wish to find a tour that visits each vertex at least once. As already mentioned in the introduction, this definition is equivalent to that of visiting each city exactly once (in the metric completion) since we assume the triangle inequality.

ATSP

Given: An edge-weighted (strongly connected) digraph $G = (V, E, w : E \to \mathbb{R}^+)$.

Find: A connected Eulerian digraph $G' = (V, E')$ where E' is a multisubset of E that minimizes $w(E')$.

Similar to the recent progress on STSP, it is natural to consider special cases that are easier to argue about but at the same time capture the combinatorial structure of the problem. In particular, we shall consider the *Node-Weighted* ATSP, where we assume that there exists a weight function $f : V \to R^+$ on the vertices so that $w(u, v) = f(u)$. (Another equivalent definition, which also applies to undirected graphs, is to let the weight of an edge (u, v) be $f(u) + f(v)$. This is equivalent to the definition above, if scaled down by a factor of 2, since the solutions are Eulerian.)

Note that this generalizes ATSP on shortest path metrics of unweighted digraphs: that is the problem where f is the constant function. As a curiosity, we also note that the recent progress on STSP when restricted to shortest path metrics of unweighted graphs is not known to generalize to the node-weighted case. We raise this as an interesting open problem in Section VI.

III. ATSP WITH LOCAL CONNECTIVITY

In this section we define a seemingly easier problem than ATSP by relaxing the connectivity requirements. Consider an optimal solution x^* to $\mathrm{LP}(G)$. Its value, which is a lower bound on OPT, can be decomposed into a "lower bound" for each vertex v:

$$\sum_{e \in E} x^*_e w(e) = \sum_{v \in V} \underbrace{\sum_{e \in \delta^+(v)} x^*_e w(e)}_{\text{lower bound for } v}.$$

With this intuition, we let $\mathrm{lb} : V \to \mathbb{R}$ be the lower bound function defined by $\mathrm{lb}_{x^*,G}(v) = \sum_{e \in \delta^+(v)} x^*_e w(e)$. We simplify notation and write lb instead of $\mathrm{lb}_{x^*,G}$ as G will always be clear from the context and therefore also x^* (if the optimal solution to $\mathrm{LP}(G)$ is not unique then make an arbitrary but consistent choice). Note that $\mathrm{lb}(V)$ equals the value of the optimal solution to the Held-Karp relaxation.

Perhaps the main difficulty of ATSP is to satisfy the connectivity requirement, i.e., to select a Eulerian subset F of edges that connects the whole graph. We shall now relax this condition to obtain what we call *Local-Connectivity* ATSP:

Local-Connectivity ATSP

Given: An edge-weighted (strongly connected) digraph $G = (V, E, w)$ and a partitioning $V_1 \cup V_2 \cup \ldots \cup V_k$ of the vertices that satisfy: the graph induced by V_i is strongly connected for $i = 1, \ldots, k$.

Find: A Eulerian multisubset F of E such that

$$|\delta^+_F(V_i)| \geq 1 \text{ for } i = 1, 2, \ldots, k \qquad \text{and} \qquad \max_{\tilde{G} \in \mathcal{C}(F)} \frac{w(\tilde{G})}{\mathrm{lb}(\tilde{G})} \text{ is minimized.}$$

Recall that $\mathcal{C}(F)$ denotes the set of connected components of the graph (V, F). We remark that the restriction that each V_i should induce a strongly connected component is not necessary but it makes our proofs in Section IV easier.

We say that an algorithm for Local-Connectivity ATSP is *α-light* if it is guaranteed (over all instances) to find a solution F such that

$$\max_{\tilde{G} \in \mathcal{C}(F)} \frac{w(\tilde{G})}{\mathrm{lb}(\tilde{G})} \leq \alpha. \tag{1}$$

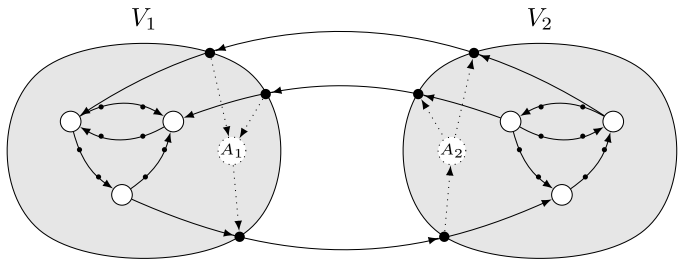

Figure 1. A depiction of the construction of the auxiliary graph G' (in the proof of Theorem IV.1): edges are subdivided, an auxiliary vertex A_i is added for each partition V_i, and A_i is "connected" to subdivisions of the edges in $\delta^+(V_i)$ and $\delta^-(V_i)$.

We also say that an algorithm is α-light on an ATSP instance $G = (V, E, w)$ if, for each partitioning $V_1 \cup \ldots \cup V_k$ of V (such that V_i induces a strongly connected graph), it returns a solution satisfying (1). We remark that we use the α-light terminology to avoid any ambiguities with the concept of approximation algorithms because an α-light algorithm does not compare its solution with respect to an optimal solution to the given instance of Local-Connectivity ATSP.

An α-approximation algorithm for ATSP with respect to the Held-Karp relaxation is trivially an α-light algorithm for Local-Connectivity ATSP: output the same Eulerian subset F as the algorithm for ATSP. Since the set F connects the graph we have $\max_{\tilde{G} \in \mathcal{C}(F)} w(\tilde{G})/\operatorname{lb}(\tilde{G}) = w(F)/\operatorname{lb}(V) \le \alpha$. Moreover, Local-Connectivity ATSP seems like a significantly easier problem than ATSP as the Eulerian set of edges only needs to cross k cuts formed by a partitioning of the vertices. We substantiate this intuition by proving, in Section IV, that there exists a simple 3-approximation for Local-Connectivity ATSP on shortest path metrics of node-weighted graphs. We refer to this case as Node-Weighted Local-Connectivity ATSP. Perhaps more surprisingly, we show in Section V that any α-light algorithm for Local-Connectivity ATSP can be turned into an algorithm for ATSP with an approximation guarantee of 5α with respect to the same lower bound (from the Held-Karp relaxation).

Remark III.1. Our generic reduction from ATSP to Local-Connectivity ATSP (Theorem V.1) is robust with respect to the definition of lb and there are many possibilities to define such a lower bound. Another natural example is $\operatorname{lb}(v) = \sum_{e \in \delta^+(v) \cup \delta^-(v)} x^*_e w(e)/2$. In fact, in order to get a constant bound on the integrality gap of the Held-Karp relaxation, our results say that it is enough to find an $O(1)$-light algorithm for Local-Connectivity ATSP with respect to some nonnegative lb that only needs to satisfy that $\operatorname{lb}(V)$ is at most the value of the optimal solution to the LP. Even more generally, if $\operatorname{lb}(V)$ is at most the value of an optimal tour then our methods would give a similar approximation guarantee (but not with respect to the Held-Karp relaxation).

IV. Approximating Local-Connectivity ATSP

We give a simple 3-light algorithm for Node-Weighted Local-Connectivity ATSP. The proof is based on finding an integral circulation that sends flow across the cuts $\{(V_i, \bar{V}_i) : i = 1, 2, \ldots, k\}$ and, in addition, satisfies that the outgoing flow of each vertex $v \in V$ is at most $\lceil x^*(\delta^+(v)) \rceil + 1$ which in turn, by the assumptions on the metric, implies a 3-light algorithm.

Theorem IV.1. *There exists a polynomial time* 3*-light algorithm for Node-Weighted Local-Connectivity ATSP.*

Proof: Let $G = (V, E, w)$ and $V_1 \cup V_2 \cup \ldots \cup V_k$ be an instance of Local-Connectivity ATSP where $w : E \to \mathbb{R}^+$ is a node-weighted metric defined by $f : V \to \mathbb{R}^+$. Let also x^* be an optimal solution to $\operatorname{LP}(G)$. We prove the theorem by giving a polynomial time algorithm that finds a Eulerian multisubset F of E satisfying

$$|\delta^+_F(V_i)| \ge 1 \text{ for } i = 1, \ldots, k \quad \text{and} \quad |\delta^+_F(v)| \le \lceil x^*(\delta^+(v)) \rceil + 1 \text{ for } v \in V. \tag{2}$$

To see that this is sufficient, note that the Eulerian set F forms a solution to the Local-Connectivity ATSP instance because $|\delta^+_F(V_i)| \ge 1$ for $i = 1, \ldots, k$; and it is 3-light since, for each $\tilde{G} = (\tilde{V}, \tilde{E}) \in \mathcal{C}(F)$, we have (using that it is a node-weighted metric)

$$\frac{w(\tilde{G})}{\mathrm{lb}(\tilde{G})} = \frac{\sum_{v\in\tilde{V}} |\delta^+_{\tilde{E}}(v)| f(v)}{\sum_{v\in\tilde{V}} x^*(\delta^+(v)) f(v)} \le \frac{\sum_{v\in\tilde{V}} (\lceil x^*(\delta^+(v))\rceil + 1) f(v)}{\sum_{v\in\tilde{V}} x^*(\delta^+(v)) f(v)} \le 3.$$

The last inequality follows from $x^*(\delta^+(v)) \ge 1$ and therefore $\lceil x^*(\delta^+(v))\rceil + 1 \le 3x^*(\delta^+(v))$.

We proceed by describing a polynomial time algorithm for finding a Eulerian set F satisfying (2). We shall do so by finding a circulation in an auxiliary graph G' obtained from G as follows (see also Figure 1):

1) Replace each edge $e = (u, v)$ in G by adding vertices $\mathrm{O}_e, \mathrm{I}_e$ and edges $(u, \mathrm{O}_e), (\mathrm{O}_e, \mathrm{I}_e), (\mathrm{I}_e, v)$;
2) For each partition V_i, $i = 1, \ldots, k$, 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 $e \in \delta^-(V_i)$.

Recall that a circulation in G' is a vector y with a nonnegative value for each edge satisfying flow conservation: $y(\delta^+(v)) = y(\delta^-(v))$ for every vertex v. The following claim follows from the construction of G' together with basic properties of flows and circulations.

Claim IV.2. We can in polynomial time find an integral circulation y in G' satisfying:

$$y(\delta^+(A_i)) = 1 \text{ for } i = 1, \ldots, k \quad \text{and} \quad y(\delta^+(v)) \le \lceil x^*(\delta^+(v)\rceil \text{ for } v \in V.$$

Proof: We use the optimal solution x^* to $\mathrm{LP}(G)$ to define a fractional circulation y' in G' that satisfies the above degree bounds. As the vertex-degree bounds are integral, it follows from basic facts about flows that we can in polynomial time find an integral circulation y satisfying the same bounds (see e.g. Chapter 11 in [19]). Circulation y' is defined as follows:

1) for each edge $e = (u, v)$ in G with $u, v \in V_i$:

$$y'_{(u,\mathrm{O}_e)} = y'_{(\mathrm{O}_e,\mathrm{I}_e)} = y'_{(\mathrm{I}_e,v)} = x^*_{(u,v)} \left(1 - \frac{1}{x^*(\delta^+(V_i))}\right).$$

2) for each edge $e = (u, v)$ in G with $u \in V_i, v \in V_j$ where $i \ne j$:

$$y'_{(\mathrm{O}_e,\mathrm{I}_e)} = x^*_{(u,v)},$$
$$y'_{(A_i,\mathrm{O}_e)} = \frac{x^*_{(u,v)}}{x^*(\delta^+(V_i))}, \qquad y'_{(u,\mathrm{O}_e)} = x^*_{(u,v)} \left(1 - \frac{1}{x^*(\delta^+(V_i))}\right),$$
$$y'_{(\mathrm{I}_e,A_j)} = \frac{x^*_{(u,v)}}{x^*(\delta^+(V_j))}, \qquad y'_{(\mathrm{I}_e,v)} = x^*_{(u,v)} \left(1 - \frac{1}{x^*(\delta^+(V_j))}\right).$$

Basically, y' is defined so that a fraction $1/x^*(\delta^+(V_i))$ of the flow crossing the cut $(V_i, V \setminus V_i)$ goes through A_i. As $x^*(\delta^+(V_i)) \ge 1$ we have that y' is nonnegative. It is also immediate from the definition of y' that it satisfies flow conservation and the degree bounds of the claim: the in- and out-flow of a vertex $v \in V_i$ is $\left(1 - \frac{1}{x^*(\delta^+(V_i))}\right) x^*(\delta^+(v))$; the in- and out-flow of an auxiliary vertex A_i is 1 by design; and the in- and out-flow of O_e and I_e for $e = (u, v)$ is $(1 - 1/x^*(\delta^+(V_i)))x^*_e$ if $u, v \in V_i$ for some $i = 1, \ldots, k$ and x^*_e otherwise. As mentioned above, the existence of fractional circulation y' implies that we can also find, in polynomial time, an integral circulation y with the required properties. ■

Having found an integral circulation y as in the above claim, we now obtain the Eulerian subset F of edges. Initially, the set F contains $y_{(\mathrm{O}_e,\mathrm{I}_e)}$ multiplicities of each edge e in G. Note that with respect to this edge set, in each partition V_i, either all vertices in V_i are balanced (each vertex's in-degree equals its out-degree) or there exist exactly one vertex u so that $|\delta^+_F(u)| - |\delta^-_F(u)| = -1$ and one vertex v so that $|\delta^+_F(v)| - |\delta^-_F(v)| = 1$. Specifically, let u be the head of the unique edge e such that $y_{(\mathrm{I}_e,A_i)} = 1$ and let v be the tail of the unique edge e' so that $y_{(A_i,\mathrm{O}_{e'})} = 1$. If $u = v$ then all vertices in V_i are balanced. Otherwise u is so that $|\delta^+_F(u)| - |\delta^-_F(u)| = -1$

and v is so that $|\delta^+_F(v)| - |\delta^-_F(v)| = 1$. In that case, we add a simple path from u to v to make the in-degrees and out-degrees of these vertices balanced. As the graph induced by V_i is strongly connected, we can select the path so that it only visits vertices in V_i. Therefore, we only increase the degree of vertices in V_i by at most 1. Hence, after repeating this operation for each partition V_i, we have that F is a Eulerian subset of edges and $|\delta_F(\delta^+(v))| \leq y(\delta^+(v))+1 \leq \lceil x^*(\delta^+(v))\rceil+1$ for all $v \in V$. Finally, we have $|\delta^+_F(V_i)| \geq 1$ for each $i = 1, \dots, k$ because $y_{A_i,\mathrm{O}_e} = 1$ (and therefore $y_{\mathrm{O}_e,\mathrm{I}_e} \geq 1$) for one edge $e \in \delta^+(V_i)$. We have thus given a polynomial time algorithm that finds a Eulerian subset F satisfying the properties of (2), which, as discussed above, implies that it is a 3-light algorithm for Node-Weighted Local-Connectivity ATSP. ■

V. From Local to Global Connectivity

In this section, we prove that if there is an α-light algorithm for Local-Connectivity ATSP, then there exists an algorithm for ATSP with an approximation guarantee of $O(\alpha)$. The main theorem can be stated as follows.

Theorem V.1. *Let $\mathcal{A}$ be an algorithm for Local-Connectivity ATSP and consider an ATSP instance $G = (V, E, w)$. If $\mathcal{A}$ is α-light on G, there exists a tour of G with value at most $5\alpha \operatorname{lb}(V)$. Moreover, for any $\varepsilon > 0$, a tour of value at most $(9 + \varepsilon)\alpha \operatorname{lb}(V)$ can be found in time polynomial in the number $n = |V|$ of vertices, in $1/\varepsilon$, and in the running time of $\mathcal{A}$.*

Throughout this section, we let $G = (V, E, w)$ and $\mathcal{A}$ be fixed as in the statement of the theorem. The proof of the theorem is by giving an algorithm that uses $\mathcal{A}$ as a subroutine. We first give the non-polynomial algorithm in Section V-A (with the better guarantee) followed by Section V-B where we modify the arguments so that we also efficiently find a tour (with slightly worse guarantee).

A. *Existence of a Good Tour*

Before describing the (non-polynomial) algorithm, we need to introduce the concept of Eulerian partition. We say that graphs $H_1 = (V_1, E_1), H_2 = (V_2, E_2), \dots, H_k = (V_k, E_k)$ form a *Eulerian partition* of G if the vertex sets $V_1, \dots, V_k$ form a partition of V and each H_i is a connected Eulerian graph where E_i is a multisubset of E. It is an *β-light Eulerian partition* if in addition

$$w(H_i) \leq \beta \cdot \operatorname{lb}(H_i) \qquad \text{for } i = 1, \dots, k.$$

Our goal is to find a 5α-light Eulerian partition that only consists of a single component, i.e., a 5α-approximate solution to the ATSP instance G with respect to the Held-Karp relaxation.

The idea of the algorithm is to start with a Eulerian partition and then iteratively merge/connect these connected components into a single connected component by adding (cheap) Eulerian subsets of edges. Note that, since we will only add Eulerian subsets, the algorithm always maintains that the connected components are Eulerian.

The *state of the algorithm* is described by a Eulerian multiset E^* that contains the multiplicities of the edges that the algorithm has picked.

Initialization: The algorithm starts with a 2α-light Eulerian partition $H_1^* = (V_1^*, E_1^*), \dots, H_k^* = (V_k^*, E_k^*)$ that maximizes the lexicographic order of

$$\langle \operatorname{lb}(H_1^*), \operatorname{lb}(H_2^*), \dots, \operatorname{lb}(H_k^*)\rangle. \tag{3}$$

As the lexicographic order is maximized, the Eulerian partitions are ordered so that $\operatorname{lb}(H_1^*) \geq \operatorname{lb}(H_2^*) \geq \cdots \geq \operatorname{lb}(H_k^*)$. For simplicity, we assume that these inequalities are strict (which is w.l.o.g. by breaking ties arbitrarily but consistently). The set E^* is initialized so that it contains the edges of the Eulerian partitions, i.e., $E^* = E_1^* \cup E_2^* \cup \cdots \cup E_k^*$.

During the execution of the algorithm we will also use the following concept. For a connected subgraph $\tilde{G} = (\tilde{V}, \tilde{E})$ of G, let $\operatorname{low}(\tilde{G})$ denote the Eulerian partition H_i^* of lowest index i that intersects $\tilde{G}$[3]. That is,

$$\operatorname{low}(\tilde{G}) = H^*_{\min\{i : V_i^* \cap \tilde{V} \neq \emptyset\}}.$$

[3]Equivalently, it is the set H_i^* maximizing $\operatorname{lb}(H_i^*)$ over all sets in the Eulerian partition that intersect $\tilde{G}$.

Note that after initialization, the connected components in $\mathcal{C}(E^*)$ are exactly the subgraphs $H_1^*, \ldots, H_k^*$. This means that $H_i^* = \mathrm{low}(\tilde{G})$ for exactly one component $\tilde{G} \in \mathcal{C}(E^*)$. Moreover, as the algorithm will only add edges, each H_i^* will be in at most one component throughout the execution.

Remark V.2. The main difference in the polynomial time algorithm is the initialization since we do not know how to find a 2α-light Eulerian partition that maximizes the lexicographic order in polynomial time. Indeed, it is consistent with our knowledge that 2α (even 2) is an upper bound on the integrality gap and, in that case, such an algorithm would always find a tour.

Remark V.3. For intuition, let us mention that the reason for starting with a Eulerian partition that maximizes the lexicographic order is that we will use the following properties to bound the weight of the total tour:

1) A connected Eulerian subgraph H of G with $w(H) \leq 2\alpha\,\mathrm{lb}(H)$ has $\mathrm{lb}(H) \leq \mathrm{lb}(\mathrm{low}(H))$.
2) For any disjoint connected Eulerian subgraphs $H_1, H_2, \ldots, H_\ell$ of G with $\mathrm{low}(H_j) = H_i^*$ and $w(H_j) \leq \alpha\,\mathrm{lb}(H_j)$ for $j = 1, \ldots, \ell$, we have

$$\sum_{j=1}^{\ell} \mathrm{lb}(H_j) \leq 2\,\mathrm{lb}(H_i^*).$$

These bounds will be used to bound the weight of the edges added in the merge procedure. Their proofs are easy and can be found in the analysis (see the proofs of Claim V.8 and Claim V.9).

Merge procedure: The algorithm repeats the following "merge procedure" until $\mathcal{C}(E^*)$ contains a single connected component. The components in $\mathcal{C}(E^*)$ partition the vertex set and each component is strongly connected as it is Eulerian (since E^* is a Eulerian subset of edges). The algorithm can therefore use $\mathcal{A}$ to find a Eulerian multisubset F of E such that

(i) $|\delta_F^+(\tilde{V})| \geq 1 \qquad$ for all $(\tilde{V}, \tilde{E}) \in \mathcal{C}(E^*)$; and
(ii) for each $\tilde{G} \in \mathcal{C}(F)$ we have $w(\tilde{G}) \leq \alpha\,\mathrm{lb}(\tilde{G})$.

Note that $\mathcal{A}$ is guaranteed to find such a set since it is assumed to be an α-light algorithm for Local-Connectivity ATSP on G. Furthermore, we may assume that no connected component in $\mathcal{C}(F)$ is completely contained in a connected component in $\mathcal{C}(E^*)$ (except for the trivial components formed by singletons). Indeed, the edges of such a component can safely be removed from F and we have a new (smaller) multiset that satisfies the above conditions. Having selected F, we now proceed to explain the "update phase":

U1: Let $X = \emptyset$.
U2: Select the component $\tilde{G} = (\tilde{V}, \tilde{E}) \in \mathcal{C}(E^* \cup F \cup X)$ that *minimizes* $\mathrm{lb}(\mathrm{low}(\tilde{G}))$.
U3: If there exists a cycle $C = (V_C, E_C)$ in G of weight $w(C) \leq \alpha\,\mathrm{lb}(\mathrm{low}(\tilde{G}))$ that connects $\tilde{G}$ to another component in $\mathcal{C}(E^* \cup F \cup X)$, then add E_C to X and repeat from Step U2.
U4: Otherwise, update E^* by adding the "new" edges in $\tilde{E}$, i.e., $E^* \leftarrow E^* \cup (\tilde{E} \cap F) \cup (\tilde{E} \cap X)$.

Some comments about the update of E^* are in order. We emphasize that we do *not* add all edges of $F \cup X$ to E^*. Instead, we only add those new edges that belong to the component $\tilde{G}$ selected in the final iteration of the update phase. As $\tilde{G}$ is a connected component in $\mathcal{C}(E^* \cup F \cup X)$, F and X are Eulerian subsets of edges, we have that E^* remains Eulerian after the update. This finishes the description of the merging procedure and the algorithm (see also the example below).

Example V.4. In Figure 2, we have that, at the start of a merging step, $\mathcal{C}(E^*)$ consists of 6 components containing $\{H_6^*, H_7^*, H_9^*, H_{10}^*\}, \{H_3^*\}, \{H_5^*, H_8^*\}, \{H_4^*\}, \{H_2^*\}$, and $\{H_1^*\}$. The blue (solid) cycles depict the connected Eulerian components of the edge set F. First, we set $X = \emptyset$ and the algorithm selects the component $\tilde{G}$ in $\mathcal{C}(E^* \cup F \cup X)$ that minimizes $\mathrm{lb}(\mathrm{low}(\tilde{G}))$ or, equivalently, that maximizes $\min\{i : H_i^* \text{ intersects } \tilde{G}\}$. In this example, it would be the left most of the three components in $\mathcal{C}(E^* \cup F)$ with $\mathrm{low}(\tilde{G}) = H_4^*$. The algorithm now tries to connect this component to another component by adding a cycle with weight at most $\alpha\,\mathrm{lb}(H_4^*)$. The red (dashed) cycle corresponds to such a cycle and its edge set is added to X. In the next iteration, the algorithm

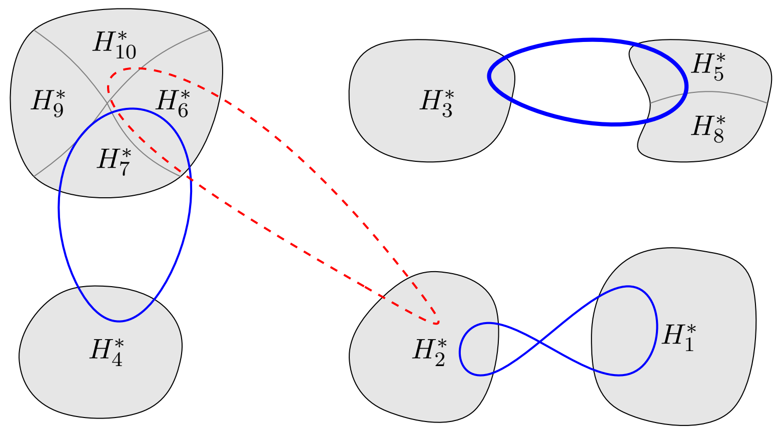

Figure 2. An illustration of the merge procedure. Blue (solid) cycles depict F and the red (dashed) cycle depicts X after one iteration of the update phase. The thick cycle represents the edges that this merge procedure would add to E^*.

considers the two components in $\mathcal{C}(E^* \cup F \cup X)$. The smallest component (with respect to $\mathrm{lb}(\mathrm{low}(\tilde{G}))$) is the one that contains H_3^*, H_5^*, and H_8^*. Now suppose that there is no cycle of weight at most $\alpha\,\mathrm{lb}(H_3^*)$ that connects this component to another component. Then the set E^* is updated by adding those edges of $F \cup X$ that belong to this component (depicted by the thick cycle).

1) Analysis: We start by showing that the algorithm terminates by arguing that the update phase terminates with fewer connected components and the merge procedure is therefore repeated at most $k \le n$ times.

Lemma V.5. *The update phase terminates in polynomial time and decreases the number of connected components in $\mathcal{C}(E^*)$.*

Proof: First, observe that each single step of the update phase can be implemented in polynomial time. The only nontrivial part is Step U3 which can be implemented as follows: for each edge $(u, v) \in \delta^+(\tilde{V})$ consider the cycle consisting of (u, v) and a shortest path from v to u. Moreover, the whole update phase terminates in polynomial time because each time the if-condition of Step U3 is satisfied, we add a cycle to X that decreases the number of connected components in $\mathcal{C}(E^* \cup F \cup X)$. The if-condition of Step U3 can therefore be satisfied at most $k \le n$ times.

We proceed by proving that at termination the update phase decreases the number of connected components in $\mathcal{C}(E^*)$. Consider when the algorithm reaches Step U4. In that case it has selected a component $\tilde{G} = (\tilde{V}, \tilde{E}) \in \mathcal{C}(E^* \cup F \cup X)$. Note that $\tilde{G} \notin \mathcal{C}(E^*)$ because the edge set F crosses each cut defined by the vertex sets of the connected components in $\mathcal{C}(E^*)$. Therefore when the algorithm updates E^* by adding all the edges $(F \cup X) \cap \tilde{E}$ it decreases the number of components in $\mathcal{C}(E^*)$ by at least one. ■

To analyze the performance guarantee we shall split our analysis into two parts. Note that when one execution of the merge procedure terminates (Step U4) we add edge set $(F \cap \tilde{E}) \cup (X \cap \tilde{E})$ to our solution. We shall analyze the contribution of these two sets $F \cap \tilde{E}$ and $X \cap \tilde{E}$ separately. More formally, suppose that the algorithm does T repetitions of the merge procedure. Let $\tilde{G}_1 = (\tilde{V}_1, \tilde{E}_1), \tilde{G}_2 = (\tilde{V}_2, \tilde{E}_2), \ldots, \tilde{G}_T = (\tilde{V}_T, \tilde{E}_T)$, $F_1, F_2, \ldots, F_T$, and $X_1, X_2, \ldots, X_T$ denote the selected components, the edge set F, and the edge set X, respectively, at the end of each repetition. To simplify notation, we denote the edges added to E^* in the t:th repetition by $\tilde{F}_t = F_t \cap \tilde{E}_t$ and $\tilde{X}_t = X_t \cap \tilde{E}_t$.

With this notation, we proceed to bound the total weight of the solution by

$$\underbrace{w\left(\cup_{t=1}^{T}\tilde{F}_t\right)}_{\leq 2\alpha\,\mathrm{lb}(V) \text{ by Lemma V.7}} + \underbrace{w\left(\cup_{t=1}^{T}\tilde{X}_t\right)}_{\leq \alpha\,\mathrm{lb}(V) \text{ by Lemma V.6}} + \sum_{i=1}^{k} w(H_i^*) \leq 5\alpha\,\mathrm{lb}(V) \text{ as claimed in Theorem V.1.}$$

Here we used that $\sum_{i=1}^{k} w(H_i^*) \leq 2\alpha\,\mathrm{lb}(V)$ since $H_1^*, \ldots, H_k^*$ is a 2α-light Eulerian partition. It remains to prove Lemmas V.6 and V.7.

Lemma V.6. *We have* $w\left(\cup_{t=1}^{T}\tilde{X}_t\right) \leq \alpha\,\mathrm{lb}(V)$.

Proof: Note that $\tilde{X}_t$ consists of a subset of the cycles added to X_t in Step U3 of the update phase. Specifically, those cycles contained in the connected component $\tilde{G}_t$ selected at Step U2 in the last iteration of the update phase during the t:th repetition of the merge procedure. We can therefore decompose $\cup_{t=1}^{T}\tilde{X}_t$ into cycles $C_1 = (V_1, E_1), C_2 = (V_2, E_2), \ldots, C_c = (V_c, E_c)$ indexed in the order they were added by the algorithm. When C_j was selected in Step U3 of the update phase, it satisfied the following two properties:

(i) it connected the component $\tilde{G}$ selected in Step U2 with at least one other component $\tilde{G}'$ such that $\mathrm{lb}(\mathrm{low}(\tilde{G}')) > \mathrm{lb}(\mathrm{low}(\tilde{G}))$; and
(ii) it had weight $w(C_j)$ at most $\alpha\,\mathrm{lb}(\mathrm{low}(\tilde{G}))$.

In this case, we say that C_j marks $\mathrm{low}(\tilde{G})$.

We claim that at most one cycle in $C_1, C_2, \ldots, C_c$ marks each $H_1^*, H_2^*, \ldots, H_k^*$. To see this, consider the first cycle C_j that marks H_i^* (if any). By (i) above, when C_j was added, it connected two components $\tilde{G}$ and $\tilde{G}'$ such that $\mathrm{lb}(\mathrm{low}(\tilde{G}')) > \mathrm{lb}(\mathrm{low}(\tilde{G}))$ where $\mathrm{low}(\tilde{G}) = H_i^*$. As the algorithm only adds edges, $\tilde{G}$ and $\tilde{G}'$ will remain connected throughout the execution of the algorithm. Therefore, by the definition of low and by the fact that $\mathrm{lb}(\mathrm{low}(\tilde{G}')) > \mathrm{lb}(\mathrm{low}(\tilde{G}))$, we have that a component $\tilde{G}''$ appearing later in the algorithm always has $\mathrm{low}(\tilde{G}'') \neq H_i^*$. Hence, no other cycle marks H_i^*.

The bound now follows from that at most one cycle marks each H_i^* and such a cycle has weight at most $\alpha\,\mathrm{lb}(H_i^*)$. ■

We complete the analysis of the performance guarantee with the following lemma.

Lemma V.7. *We have* $w\left(\cup_{t=1}^{T}\tilde{F}_t\right) \leq 2\alpha\,\mathrm{lb}(V)$.

Proof: Consider the t:th repetition of the merge procedure. The edge set $\tilde{F}_t$ is Eulerian but not necessarily connected. Let $\mathcal{F}^t$ denote the set of the Eulerian subgraphs corresponding to the connected components in $\mathcal{C}(\tilde{F}_t)$ where we disregard the trivial components that only consist of a single vertex. Further, partition $\mathcal{F}^t$ into $\mathcal{F}_1^t, \mathcal{F}_2^t, \ldots, \mathcal{F}_k^t$ where $\mathcal{F}_i^t$ contains those Eulerian subgraphs in $\mathcal{F}^t$ that intersect H_i^* and do not intersect any of the subgraphs $H_1^*, H_2^*, \ldots, H_{i-1}^*$. That is,

$$\mathcal{F}_i^t = \{H \in \mathcal{F}^t : \mathrm{low}(H) = H_i^*\}.$$

Note that the total weight of $\tilde{F}_t$, $w(\tilde{F}_t)$, equals $w(\mathcal{F}^t) = \sum_{i=1}^{k} w(\mathcal{F}_i^t)$. We bound the weight of $\mathcal{F}^t$ by considering each $\mathcal{F}_i^t$ separately. We start by two simple claims that follow from that each $H \in \mathcal{F}^t$ satisfies $w(H) \leq \alpha\,\mathrm{lb}(H)$ (since $\mathcal{A}$ is an α-light algorithm) and the choice of $H_1^*, \ldots, H_k^*$ to maximize the lexicographic order of (3). We remark that the proofs of the following claims are the only arguments that use the fact that the lexicographic order was maximized.

Claim V.8. For $H \in \mathcal{F}_i^t$, we have $\mathrm{lb}(H) \leq \mathrm{lb}(\mathrm{low}(H)) = \mathrm{lb}(H_i^*)$.

Proof: Inequality $\mathrm{lb}(H) > \mathrm{lb}(H_i^*)$ together with the fact that $w(H) \leq \alpha\,\mathrm{lb}(H) \leq 2\alpha\,\mathrm{lb}(H)$ would contradict that $H_1^*, \ldots, H_k^*$ was chosen to maximize the lexicographic order of (3). Indeed, in that case, a 2α-light Eulerian partition of higher lexicographic order would be $H_1^*, H_2^*, \ldots, H_{i-1}^*, H$ and the remaining vertices (as trivial singleton components) that do not belong to any of these Eulerian subgraphs. ■

Claim V.9. We have $\mathrm{lb}(\mathcal{F}_i^t) \le 2\,\mathrm{lb}(H_i^*)$.

Proof: Suppose toward contradiction that $\mathrm{lb}(\mathcal{F}_i^t) > 2\,\mathrm{lb}(H_i^*)$. Let $\mathcal{F}_i^t = \{H_1, H_2, \dots, H_\ell\}$ and define H^* to be the Eulerian graph obtained by taking the union of the graphs H_i^* and $H_1, \dots, H_\ell$. Consider the Eulerian partition $H_1^*, \dots, H_{i-1}^*, H^*$ and the remaining vertices (as trivial singleton components) that do not belong to any of these Eulerian subgraphs. We have $\mathrm{lb}(H^*) > \mathrm{lb}(H_i^*)$ and therefore the lexicographic value of this Eulerian partition is larger than the lexicographic value of $H_1^*, \dots, H_k^*$. This is a contradiction if it is also a 2α-light Eulerian partition, i.e., if $\frac{w(H^*)}{\mathrm{lb}(H^*)} \le 2\alpha$.

Therefore, we must have $w(H^*) > 2\alpha\,\mathrm{lb}(H^*)$. By the facts that $w(H_j) \le \alpha\,\mathrm{lb}(H_j)$ (since $\mathcal{A}$ is an α-light algorithm) and that $H_1^*, \dots, H_k^*$ is a 2α-light Eulerian partition,

$$w(H^*) = w(H_i^*) + \sum_{j=1}^{\ell} w(H_j) \le 2\alpha\,\mathrm{lb}(H_i^*) + \sum_{j=1}^{\ell} \alpha\,\mathrm{lb}(H_j) \quad \text{and} \quad \mathrm{lb}(H^*) \ge \sum_{j=1}^{\ell} \mathrm{lb}(H_j).$$

These inequalities together with $w(H^*) > 2\alpha\,\mathrm{lb}(H^*)$ imply $\mathrm{lb}(\mathcal{F}_i^t) = \sum_{j=1}^{\ell} \mathrm{lb}(H_j) \le 2\,\mathrm{lb}(H_i^*)$. ■

Using the above claim, we can write $w\left(\cup_{t=1}^{T} \tilde{F}_t\right)$ as

$$\sum_{t=1}^{T}\sum_{i=1}^{k} w(\mathcal{F}_i^t) \le \alpha \sum_{t=1}^{T}\sum_{i=1}^{k} \mathrm{lb}(\mathcal{F}_i^t) = \alpha \sum_{i=1}^{k} \sum_{t:\mathcal{F}_i^t \neq \emptyset} \mathrm{lb}(\mathcal{F}_i^t) \le 2\alpha \sum_{i=1}^{k} \sum_{t:\mathcal{F}_i^t \neq \emptyset} \mathrm{lb}(H_i^*).$$

We complete the proof of the lemma by using Claim V.8 to prove that $\mathcal{F}_i^t$ is non-empty for at most one repetition t of the merge procedure. Suppose toward contradiction that there exist $1 \le t_0 < t_1 \le T$ so that both $\mathcal{F}_i^{t_0} \neq \emptyset$ and $\mathcal{F}_i^{t_1} \neq \emptyset$. In the t_0:th repetition of the merge procedure, H_i^* is contained in the subgraph $\tilde{G}_{t_0}$ since otherwise no edges incident to H_i^* would have been added to E^*. Therefore $\mathrm{lb}(\mathrm{low}(\tilde{G}_{t_0})) \ge \mathrm{lb}(H_i^*)$. Now consider a Eulerian subgraph $H \in \mathcal{F}_i^{t_1}$. First, we cannot have that H is contained in the component $\tilde{G}_{t_0}$ since each (nontrivial) component of F is assumed to not be contained in any component of $\mathcal{C}(E^*)$. Second, by Claim V.8, we have $w(H) \le \alpha\,\mathrm{lb}(H) \le \alpha\,\mathrm{lb}(H_i^*)$.

In short, H is a Eulerian subgraph that connects $\tilde{G}_{t_0}$ to another component and it has weight at most $\alpha\,\mathrm{lb}(\mathrm{low}(\tilde{G}_{t_0}))$. As H is Eulerian, it can be decomposed into cycles. One of these cycles, say C, connects $\tilde{G}_{t_0}$ to another component and

$$w(C) \le w(H) \le \alpha\,\mathrm{lb}(H_i^*) \le \alpha\,\mathrm{lb}(\mathrm{low}(\tilde{G}_{t_0})). \tag{4}$$

In other words, there exists a cycle C that, in the t_0:th repetition of the merge procedure, satisfied the if-condition of Step U3, which contradicts the fact that Step U4 was reached when component $\tilde{G}_{t_0}$ was selected. ■

B. Polynomial Time Algorithm

In this section we describe how to modify the arguments in Section V-A to obtain an algorithm that runs in time polynomial in the number n of vertices, in $1/\varepsilon$, and in the running time of $\mathcal{A}$.

By Lemma V.5, the update phase can be implemented in polynomial time in n. Therefore, the merge procedure described in Section V-A runs in time polynomial in n and in the running time of $\mathcal{A}$. The problem is the initialization: as mentioned in Remark V.2, it seems difficult to find a polynomial time algorithm for finding a 2α-light Eulerian partition $H_1^*, \dots, H_k^*$ that maximizes the lexicographic order of

$$\langle \mathrm{lb}(H_1^*), \mathrm{lb}(H_2^*), \dots, \mathrm{lb}(H_k^*) \rangle.$$

We overcome this obstacle by first identifying the properties that we actually use from selecting the Eulerian partition as above. We then show that we can obtain a Eulerian partition that satisfies these properties in polynomial time.

As mentioned in the analysis in Section V-A, the only place where we use that the Eulerian partition maximizes the lexicographic order of (3) is in the proof of Lemma V.7. Specifically, it is used in the proofs of Claims V.8

and V.9. Instead of proving these claims, we shall simply concentrate on finding a Eulerian partition that satisfies a relaxed variant of them (formalized in the lemma below, see Condition (5)). The claimed polynomial time algorithm is then obtained by first proving that a slight modification of the merge procedure returns a tour of value at most $(9\alpha + 2\varepsilon)\,\mathrm{lb}(V)$ if Condition (5) holds, and then we show that a Eulerian partition satisfying this condition can be found in time polynomial in n and in the running time of $\mathcal{A}$. We start by describing the modification to the merge procedure.

Modified merge procedure: The only modification to the merge procedure in Section V-A is that we change the update phase by relaxing the condition of the if-statement in Step U3 from $w(C) \leq \alpha\,\mathrm{lb}(\mathrm{low}(\tilde{G}))$ to $w(C) \leq \alpha(3\,\mathrm{lb}(\mathrm{low}(\tilde{G})) + \varepsilon\,\mathrm{lb}(V)/n)$. In other words, Step U3 is replaced by

U3': If there exists a cycle $C = (V_C, E_C)$ in G of weight $w(C) \leq \alpha(3\,\mathrm{lb}(\mathrm{low}(\tilde{G})) + \varepsilon\,\mathrm{lb}(V)/n)$ that connects $\tilde{G}$ to another component in $\mathcal{C}(E^* \cup F \cup X)$, then add E_C to X and repeat from Step U2.

Clearly the modified merge procedure still runs in time polynomial in n and in the running time of $\mathcal{A}$. Moreover, we show that if Condition (5) holds then the returned tour will have weight $O(\alpha)$. Recall from Section V-A that $\tilde{F}_t$ denotes the subset of F and $\tilde{X}_t$ denotes the subset of X that were added in the t:th repetition of the (modified) merge procedure. Furthermore, we define (as in the previous section) $\mathcal{F}_i^t = \{H \in \mathcal{C}(\tilde{F}_t) : \mathrm{low}(H) = H_i^*$ and H is a nontrivial component, i.e., H contains more than one vertex$\}$.

Lemma V.10. *Assume that the algorithm is initialized with a 3α-light Eulerian partition $H_1^*, H_2^*, \ldots, H_k^*$ so that, in each repetition t of the modified merge procedure, we add a subset $\tilde{F}_t$ such that*

$$\mathrm{lb}(\mathcal{F}_i^t) \leq 3\,\mathrm{lb}(H_i^*) + \frac{\varepsilon\,\mathrm{lb}(V)}{n} \qquad \textit{for } i = 1, 2, \ldots, k. \tag{5}$$

Then the returned tour has weight at most $(9 + 2\epsilon)\alpha\,\mathrm{lb}(V)$.

Let us comment on the above statement before giving its proof. The reason that we use a 3α-light Eulerian partition (instead of one that is 2α-light) is that it leads to a better constant when balancing the parameters. We also remark that (5) is a relaxation of the bound of Claim V.9 from $\mathrm{lb}(\mathcal{F}_i^t) < 2\,\mathrm{lb}(H_i^*)$ to $\mathrm{lb}(\mathcal{F}_i^t) \leq 3\,\mathrm{lb}(H_i^*) + \varepsilon\,\mathrm{lb}(V)/n$; and it also implies a relaxed version of Claim V.8: from $\mathrm{lb}(H) \leq \mathrm{lb}(H_i^*)$ to $\mathrm{lb}(H) \leq 3\,\mathrm{lb}(H_i^*) + \varepsilon\,\mathrm{lb}(V)/n$. It is because of this relaxed bound that we modified the if-condition of the update phase (by relaxing it by the same amount) which will be apparent in the proof.

Proof: As in the analysis of the performance guarantee in Section V-A, we can write the weight of the returned tour as

$$w\left(\cup_{t=1}^T \tilde{F}_t\right) + w\left(\cup_{t=1}^T \tilde{X}_t\right) + \sum_{i=1}^k w(H_i^*).$$

To bound $w\left(\cup_{t=1}^T \tilde{X}_t\right)$, we observe that proof of Lemma V.6 generalizes verbatim except that the weight of a cycle C that marks H_i^* is now bounded by $\alpha(3\,\mathrm{lb}(H_i^*) + \varepsilon\,\mathrm{lb}(V)/n)$ instead of by $\alpha\,\mathrm{lb}(H_i^*)$ (because of the relaxation of the bound in the if-condition of the update phase). Hence, $w\left(\cup_{t=1}^T \tilde{X}_t\right) \leq \sum_{i=1}^k \alpha(3\,\mathrm{lb}(H_i^*) + \varepsilon\,\mathrm{lb}(V)/n) \leq (3 + \epsilon)\alpha\,\mathrm{lb}(V)$ because $k \leq n$.

We proceed to bound $w\left(\cup_{t=1}^T \tilde{F}_t\right)$. Using the same arguments as in the proof of Lemma V.7,

$$w\left(\cup_{t=1}^T \tilde{F}_t\right) \leq \alpha \sum_{i=1}^k \sum_{t:\mathcal{F}_i^t \neq \emptyset} \mathrm{lb}(\mathcal{F}_i^t) \leq \alpha \sum_{i=1}^k \sum_{t:\mathcal{F}_i^t \neq \emptyset} \left(3\,\mathrm{lb}(H_i^*) + \varepsilon\,\mathrm{lb}(V)/n\right)$$

where, for the last inequality, we used the assumption of the lemma. Now we apply exactly the same arguments as in the end of the proof of Lemma V.7 to prove that $\mathcal{F}_i^t$ is non-empty for at most one repetition t of the merge procedure. The only difference, is that (4) should be replaced by

$$w(C) \leq w(H) \leq \alpha(3\,\mathrm{lb}(H_i^*) + \varepsilon\,\mathrm{lb}(V)/n) \leq \alpha(3\,\mathrm{lb}(\mathrm{low}(\tilde{G}_{t_0})) + \varepsilon\,\mathrm{lb}(V)/n)$$

(because (5) can be seen as a relaxed version of Claim V.8). However, as we also updated the bound in the if-condition, the argument that C would satisfy the if-condition of Step U3' is still valid. Hence, we conclude that $\mathcal{F}_i^t$ is non-empty in at most one repetition and therefore

$$w\left(\cup_{t=1}^{T}\tilde{F}_t\right) \leq \alpha \sum_{i=1}^{k} \sum_{t:\mathcal{F}_i^t \neq \emptyset} (3\,\mathrm{lb}(H_i^*) + \varepsilon\,\mathrm{lb}(V)/n) \leq (3+\varepsilon)\alpha\,\mathrm{lb}(V).$$

By the above bounds and since $H_1^*, H_2^*, \ldots, H_k^*$ is a 3α-light Eulerian partition, we have that the weight of the returned tour is

$$\begin{aligned} w\left(\cup_{t=1}^{T}\tilde{F}_t\right) + w\left(\cup_{t=1}^{T}\tilde{X}_t\right) + \sum_{i=1}^{k} w(H_i^*) &\leq (3+\varepsilon)\alpha\,\mathrm{lb}(V) + (3+\varepsilon)\alpha\,\mathrm{lb}(V) + 3\alpha\,\mathrm{lb}(V) \\ &= (9+2\varepsilon)\alpha\,\mathrm{lb}(V). \end{aligned}$$

■

Finding a good Eulerian partition in polynomial time: By the above lemma, it is sufficient to find a 3α-light Eulerian partition so that Condition (5) holds during the execution of the modified merge procedure. However, how can we do it in polynomial time? We do as follows. First, we select the trivial 3α-light Eulerian partition where each subgraph is only a single vertex. Then we run the modified merge procedure and, in each repetition, we verify that Condition (5) holds. Note that this condition is easy to verify in time polynomial in n. If it holds until we return a tour, then we know by Lemma V.10 that the tour has weight at most $(9+2\varepsilon)\alpha\,\mathrm{lb}(V)$. If it does not hold during one repetition, then we will restart the algorithm with a new 3α-light Eulerian partition that we find using the following lemma. We continue in this manner until the merge procedure executes without violating Condition (5) and therefore it returns a tour of weight at most $(9\alpha + 2\varepsilon)\,\mathrm{lb}(V)$.

Lemma V.11. *Suppose that repetition t of the (modified) merge procedure violates Condition* (5) *when run starting from a 3α-light Eulerian partition $H_1^*, H_2^*, \ldots, H_k^*$. Then we can, in time polynomial in n, find a new 3α-light Eulerian partition $\hat{H}_1^*, \hat{H}_2^*, \ldots, \hat{H}_{\hat{k}}^*$ so that*

$$\sum_{j=1}^{\hat{k}} \mathrm{lb}(\hat{H}_j^*)^2 - \sum_{j=1}^{k} \mathrm{lb}(H_j^*)^2 \geq \frac{\varepsilon^2}{3n^2}\,\mathrm{lb}(V)^2. \tag{6}$$

Note that the above lemma implies that we will reinitialize (in polynomial time) the Eulerian partition at most $3n^2/\varepsilon^2$ times because any Eulerian partition $H_1^*, \ldots, H_k^*$ has $\sum_{i=1}^{k} \mathrm{lb}(H_i^*)^2 \leq \mathrm{lb}(V)^2$. As each execution of the merge procedure takes time polynomial in n and in the running time of $\mathcal{A}$, we can therefore find a tour of weight at most $(9+2\varepsilon)\alpha\,\mathrm{lb}(V) = (9+\varepsilon')\alpha\,\mathrm{lb}(V)$ in the time claimed in Theorem V.1, i.e., polynomial in n, $1/\varepsilon'$, and in the running time of $\mathcal{A}$. It remains to prove the lemma.

Proof: Since the t:th repetition of the merge procedure violates Condition (5), there is an $1 \leq i \leq k$ such that

$$\mathrm{lb}(\mathcal{F}_i^t) > 3\,\mathrm{lb}(H_i^*) + \frac{\varepsilon}{n}\,\mathrm{lb}(V).$$

We shall use this fact to construct a new 3α-light Eulerian partition consisting of a new Eulerian subgraph H^* together with a subset of $\{H_1^*, H_2^*, \ldots, H_k^*\}$ containing those subgraphs that do not intersect H^* and finally the vertices (as trivial singleton components) that do not belong to any of these Eulerian subgraphs. We need to define the Eulerian subgraph H^*. Let $I \subseteq \{1, 2, \ldots, k\}$ be the indices of those Eulerian subgraphs of $H_1^*, \ldots, H_k^*$ that intersect the vertices in $\mathcal{F}_i^t$. Note that, by definition, we have $i \in I$ and $j \geq i$ for all $j \in I$. We shall construct the graph H^* iteratively. Initially, we let H^* be the connected Eulerian subgraph obtained by taking the union of $\mathcal{F}_i^t$ and H_i^*. This is a connected Eulerian subgraph as each Eulerian subgraph in $\mathcal{F}_i^t$ intersects H_i^* and H_i^* is a connected Eulerian subgraph.

The careful reader can observe that up to now H^* is defined in the same way as in the proof of Claim V.9. However, in order to satisfy (6) we shall add more of the Eulerian subgraphs in $\{H_j^*\}_{j\in I}$ to H^*. Specifically, we would like to add $\{H_j^*\}_{j\in I'}$, where $I' \subseteq I \setminus \{i\}$ is selected so as to maximize $\mathrm{lb}(H^*)$ (because we wish to increase the "potential" in (6)) subject to that $w(H^*) \le 3\alpha\,\mathrm{lb}(H^*)$ (because the new Eulerian partition should be 3α-light).

To see that $w(H^*) \le 3\alpha\,\mathrm{lb}(H^*)$ implies that the new Eulerian partition is 3α-light, recall that the new Eulerian partition consists of H^*, the Eulerian subgraphs $\{H_j^*\}_{j\notin I}$, and the vertices that do not belong to any of these Eulerian subgraphs. By the definition of I, no H_j^* with $j \notin I$ intersects H^*. As $H_1^*, \ldots, H_k^*$ are disjoint, it follows that the new Eulerian partition consists of disjoint subgraphs. Moreover, each H_j^* satisfies $w(H_j^*) \le 3\alpha\,\mathrm{lb}(H_j^*)$ since the Eulerian partition we started with is 3α-light. Hence, the new Eulerian partition is 3α-light if $w(H^*) \le 3\alpha\,\mathrm{lb}(H^*)$. Inequality (7) is thus a sufficient condition for the new Eulerian partition to be 3α-light. We remark that the condition trivially holds for $I' = \emptyset$ because $\mathrm{lb}(\mathcal{F}_i^t) > 3\,\mathrm{lb}(H_i^*) + \varepsilon\,\mathrm{lb}(V)/n$.

Claim V.12. We have $w(H^*) \le 3\alpha\,\mathrm{lb}(H^*)$ if

$$\sum_{j\in I'} \mathrm{lb}(H_j^* \cap \mathcal{F}_i^t) \le \frac{2}{3}\,\mathrm{lb}(\mathcal{F}_i^t) - \mathrm{lb}(H_i^* \cap \mathcal{F}_i^t). \tag{7}$$

Proof: We have

$$w(H^*) = w(\mathcal{F}_i^t) + w(H_i^*) + \sum_{j\in I'} w(H_j^*) \le \alpha\,\mathrm{lb}(\mathcal{F}_i^t) + 3\alpha\,\mathrm{lb}(H_i^*) + 3\alpha \sum_{j\in I'} \mathrm{lb}(H_j^*),$$

where the inequality follows from that $\mathcal{F}_i^t$ was selected by the α-light algorithm $\mathcal{A}$ and $H_1^*, \ldots, H_k^*$ is a 3α-light Eulerian partition. Moreover,

$$\mathrm{lb}(H^*) = \mathrm{lb}(\mathcal{F}_i^t) + \mathrm{lb}(H_i^* \setminus \mathcal{F}_i^t) + \sum_{j\in I'} \mathrm{lb}(H_j^* \setminus \mathcal{F}_i^t).$$

Hence, we have, by rearranging terms and using $\mathrm{lb}(H_j^*) - \mathrm{lb}(H_j^* \setminus \mathcal{F}_i^t) = \mathrm{lb}(H_j^* \cap \mathcal{F}_i^t)$, that $w(H^*) \le 3\alpha\,\mathrm{lb}(H^*)$ holds if

$$3\alpha\,\mathrm{lb}(H_i^* \cap \mathcal{F}_i^t) + 3\alpha \sum_{j\in I'} \mathrm{lb}(H_j^* \cap \mathcal{F}_i^t) \le 2\alpha\,\mathrm{lb}(\mathcal{F}_i^t).$$

The above can be simplified to

$$\sum_{j\in I'} \mathrm{lb}(H_j^* \cap \mathcal{F}_i^t) \le 2\,\mathrm{lb}(\mathcal{F}_i^t)/3 - \mathrm{lb}(H_i^* \cap \mathcal{F}_i^t).$$

■

From the above discussion, we wish to find a subset $I' \subseteq I \setminus \{i\}$ that satisfies (7) and maximizes

$$\mathrm{lb}(H^*) = \mathrm{lb}(\mathcal{F}_i^t) + \mathrm{lb}(H_i^* \setminus \mathcal{F}_i^t) + \sum_{j\in I'} \mathrm{lb}(H_i^* \setminus \mathcal{F}_i^t),$$

where only the last term depends on the selection of I'. We interpret this as a knapsack problem that, for each $j \in I \setminus \{i\}$, has an item of size $s_j = \mathrm{lb}(H_j^* \cap \mathcal{F}_i^t)$ and profit $p_j = \mathrm{lb}(H_j^* \setminus \mathcal{F}_i^t)$; the capacity U of the knapsack is $\frac{2}{3}\,\mathrm{lb}(\mathcal{F}_i^t) - \mathrm{lb}(H_i^* \cap \mathcal{F}_i^t)$, i.e., the right-hand-side of (7). We solve this knapsack problem and obtain I' as follows:

1) Find an optimal extreme point solution z^* to the standard linear programming relaxation of the knapsack problem:

$$\begin{aligned} \text{maximize} \quad & \sum_{j\in I\setminus\{i\}} z_j p_j \\ \text{subject to} \quad & \sum_{j\in I\setminus\{i\}} z_j s_j \le U, \\ & 0 \le z_j \le 1 \qquad \text{for all } j \in I \setminus \{i\}. \end{aligned}$$

2) As the above relaxation has only one constraint (apart from the boundary constraints), the extreme point z^* has at most one variable with a fractional value. We obtain an integral solution (i.e., a packing) by simply dropping the fractionally packed item. That is, we let $I' = \{j \in I \setminus \{i\} : z^*_j = 1\}$.

The running time of the above procedure is dominated by the time it takes to solve the linear program. This can be done very efficiently by solving the fractional knapsack problem with the greedy algorithm (or, for the purpose here, use any general polynomial time algorithm for linear programming). We can therefore obtain I' and the new Eulerian partition in time polynomial in $|I| \leq n$ as stated in lemma.

It remains to prove (6). Let us first bound the profit of our "knapsack solution" I'.

Claim V.13. We have $\sum_{j\in I'} \mathrm{lb}(H^*_j \setminus \mathcal{F}^t_i) \geq \frac{1}{3}\sum_{j\in I\setminus i} \mathrm{lb}(H^*_j \setminus \mathcal{F}^t_i) - \mathrm{lb}(H^*_i)$.

Proof: By definition,

$$\sum_{j\in I'} \mathrm{lb}(H^*_j \setminus \mathcal{F}^t_i) = \sum_{j\in I\setminus\{i\}: z^*_j = 1} p_j \geq \sum_{j\in I\setminus\{i\}} z^*_j p_j - \max_{j\in I\setminus\{i\}} p_j,$$

where we used that at most one item is fractionally packed in z^*. As $j \geq i$ for all $j \in I$, $\max_{j\in I\setminus\{i\}} p_j = \max_{j\in I\setminus\{i\}} \mathrm{lb}(H^*_j \setminus \mathcal{F}^t_i) \leq \mathrm{lb}(H^*_i)$. To complete the proof of the claim, it is thus sufficient to prove that $z'_j = 1/3$ for all $j \in I \setminus \{i\}$ is a feasible solution to the LP relaxation of the knapsack problem. Indeed, by the optimality of z^*, we then have $\sum_{j\in I\setminus\{i\}} z^*_j p_j \geq \sum_{j\in I\setminus\{i\}} z'_j p_j = \frac{1}{3}\sum_{j\in I\setminus\{i\}} \mathrm{lb}(H^*_j \setminus \mathcal{F}^t_i)$.

We have that z' is a feasible solution because

$$\frac{1}{3}\sum_{j\in I\setminus\{i\}} \mathrm{lb}(H^*_j \cap \mathcal{F}^t_i) \leq \frac{1}{3}\mathrm{lb}(\mathcal{F}^t_i) \leq \left(\frac{2}{3} - \frac{\mathrm{lb}(H^*_i \cap \mathcal{F}^t_i)}{\mathrm{lb}(\mathcal{F}^t_i)}\right)\mathrm{lb}(\mathcal{F}^t_i) = U,$$

where the first inequality follows from that the subgraphs $\{H^*_j\}_{j\in I}$ are disjoint and the second inequality follows from that $\mathrm{lb}(H^*_i)/\,\mathrm{lb}(\mathcal{F}^t_i) \leq 1/3$. ■

We finish the proof of the lemma by using the above claim to show the increase of the "potential" function as stated in (6). By the definition of the new Eulerian partition (it contains $\{H^*_j\}_{j\notin I}$), we have that the increase is at least

$$\mathrm{lb}(H^*)^2 - \sum_{j\in I} \mathrm{lb}(H^*_j)^2.$$

Let us concentrate on the first term:

$$\begin{aligned}\mathrm{lb}(H^*)^2 &= \left(\mathrm{lb}(\mathcal{F}^t_i) + \mathrm{lb}(H^*_i \setminus \mathcal{F}^t_i) + \sum_{j\in I'} \mathrm{lb}(H^*_j \setminus \mathcal{F}^t_i)\right)^2 \\ &\geq \mathrm{lb}(\mathcal{F}^t_i)\left(\mathrm{lb}(\mathcal{F}^t_i) + \mathrm{lb}(H^*_i \setminus \mathcal{F}^t_i) + \sum_{j\in I'} \mathrm{lb}(H^*_j \setminus \mathcal{F}^t_i)\right).\end{aligned}$$

By Claim V.13, we have that the expression inside the parenthesis is at least

$$\begin{aligned}\mathrm{lb}(\mathcal{F}^t_i) + \mathrm{lb}(H^*_i \setminus \mathcal{F}^t_i) + &\frac{1}{3}\sum_{j\in I\setminus\{i\}} \mathrm{lb}(H^*_j \setminus \mathcal{F}^t_i) - \mathrm{lb}(H^*_i) \\ &\geq \mathrm{lb}(\mathcal{F}^t_i) + \frac{1}{3}\sum_{j\in I} \mathrm{lb}(H^*_j \setminus \mathcal{F}^t_i) - \mathrm{lb}(H^*_i).\end{aligned}$$

By using $\mathrm{lb}(H^*_i) \leq \mathrm{lb}(\mathcal{F}^t_i)/3$, we can further lower bound this expression by

$$\frac{1}{3}\mathrm{lb}(\mathcal{F}^t_i) + \frac{1}{3}\left(\mathrm{lb}(\mathcal{F}^t_i) + \sum_{j\in I} \mathrm{lb}(H^*_j \setminus \mathcal{F}^t_i)\right) = \frac{1}{3}\mathrm{lb}(\mathcal{F}^t_i) + \frac{1}{3}\sum_{j\in I} \mathrm{lb}(H^*_j).$$

Finally, as $\mathrm{lb}(\mathcal{F}_i^t) \geq \varepsilon\,\mathrm{lb}(V)/n$, $\mathrm{lb}(\mathcal{F}_i^t) \geq 3\,\mathrm{lb}(H_i^*)$, and $\mathrm{lb}(H_j^*) \leq \mathrm{lb}(H_i^*)$ for all $j \in I$, we have

$$
\begin{aligned}
\mathrm{lb}(H^*)^2 - \sum_{j\in I}\mathrm{lb}(H_j^*)^2 &\geq \mathrm{lb}(H^*)^2 - \mathrm{lb}(H_i^*)\sum_{j\in I}\mathrm{lb}(H_j^*) \\
&\geq \mathrm{lb}(\mathcal{F}_i^t)\left(\frac{1}{3}\,\mathrm{lb}(\mathcal{F}_i^t) + \frac{1}{3}\sum_{j\in I}\mathrm{lb}(H_j^*)\right) - \mathrm{lb}(H_i^*)\sum_{j\in I}\mathrm{lb}(H_j^*) \\
&\geq \frac{\mathrm{lb}(\mathcal{F}_i^t)^2}{3} + \frac{\mathrm{lb}(\mathcal{F}_i^t)}{3}\sum_{j\in I}\mathrm{lb}(H_j^*) - \mathrm{lb}(H_i^*)\sum_{j\in I}\mathrm{lb}(H_j^*) \\
&\geq \frac{\varepsilon^2}{3n^2}\,\mathrm{lb}(V)^2
\end{aligned}
$$

which completes the proof of Lemma V.11. ■

VI. Discussion and Open Problems

We gave a new approach for approximating the asymmetric traveling salesman problem. It is based on relaxing the global connectivity requirements into local connectivity conditions, which is formalized as Local-Connectivity ATSP. We showed a rather easy 3-light algorithm for Local-Connectivity ATSP on shortest path metrics of node-weighted graphs. This yields via our generic reduction a constant factor approximation algorithm for Node-Weighted ATSP. However, we do not know any $O(1)$-light algorithm for Local-Connectivity ATSP on general metrics and, motivated by our generic reduction, we raise the following intriguing question:

Open Question VI.1. Is there a $O(1)$-light algorithm for Local-Connectivity ATSP on general metrics?

We note that there is great flexibility in the exact choice of the lower bound lb as noted in Remark III.1. A further generalization of our approach is to interpret it as a primal-dual approach. Specifically, it might be useful to interpret the lower bound as a feasible solution of the dual of the Held-Karp relaxation: the lower bound is then not only defined over the vertices but over all cuts in the graph. We do not know if any of these generalizations are useful at this point and it may be that there is a nice $O(1)$-light algorithm for Local-Connectivity ATSP without changing the definition of lb.

By specializing the generic reduction to Node-Weighted ATSP, it is possible to improve our bounds slightly for this case. Specifically, one can exploit the fact that a cycle C always has $w(C) \leq \mathrm{lb}(C)$ in these metrics. This allows one to change the bound in Step U3 of the update phase to be $w(C) \leq \mathrm{lb}(\mathrm{low}(\tilde{G}))$ instead of $w(C) \leq \alpha\,\mathrm{lb}(\mathrm{low}(\tilde{G}))$, which in turn improves the upper bound on the integrality gap of the Held-Karp relaxation to $4 \cdot \alpha + 1 = 13$ (since $\alpha = 3$ for node-weighted metrics). That said, we do not see how to make a significant improvement in the guarantee and it would be very interesting with a tight analysis of the integrality gap of the Held-Karp relaxation for Node-Weighted ATSP. We believe that such a result would also be very interesting even if we restrict ourselves to shortest path metrics of unweighted graphs.

Finally, let us remark that the recent progress for STSP on shortest path metrics of unweighted graphs is not known to extend to node-weighted graphs, i.e., Node-Weighted STSP. Is it possible to give a $(1.5 - \varepsilon)$-approximation algorithm for Node-Weighted STSP for some constant $\varepsilon > 0$? We think that this is a very natural question that lies in between the now fairly well understood STSP on shortest path metrics of unweighted graphs and STSP on general metrics (i.e., edge-weighted instead of node-weighted graphs).

Acknowledgment

The author is very grateful to László Végh, Johan Håstad, and Hyung-Chan An for inspiring discussions and valuable comments that influenced this work. We also thank Jakub Tarnawski and Jens Vygen for useful feedback on the manuscript.

This research is supported by ERC Starting Grant 335288-OptApprox.

References

[1] M. Karpinski, M. Lampis, and R. Schmied, "New inapproximability bounds for TSP," in *Algorithms and Computation - 24th International Symposium, ISAAC 2013*, 2013, pp. 568–578.

[2] N. Christofides, "Worst-case analysis of a new heuristic for the travelling salesman problem," Graduate School of Industrial Administration, CMU, Tech. Rep., 1976.

[3] S. O. Gharan, A. Saberi, and M. Singh, "A randomized rounding approach to the traveling salesman problem," in *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011*, 2011, pp. 550–559.

[4] T. Mömke and O. Svensson, "Approximating graphic TSP by matchings," in *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011*, 2011, pp. 560–569.

[5] M. Mucha, "13/9-approximation for graphic TSP," in *29th International Symposium on Theoretical Aspects of Computer Science, STACS 2012*, 2012, pp. 30–41.

[6] A. Sebö and J. 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*, vol. 34, no. 5, pp. 597–629, 2014.

[7] M. Grigni, E. Koutsoupias, and C. H. Papadimitriou, "An approximation scheme for planar graph TSP," in *36th Annual Symposium on Foundations of Computer Science, FOCS 1995*, 1995, pp. 640–645.

[8] S. Arora, M. Grigni, D. R. Karger, P. N. Klein, and A. Woloszyn, "A polynomial-time approximation scheme for weighted planar graph TSP," in *Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 1998*, 1998, pp. 33–41.

[9] M. Charikar, M. X. Goemans, and H. J. Karloff, "On the integrality ratio for the asymmetric traveling salesman problem," *Math. Oper. Res.*, vol. 31, no. 2, pp. 245–252, 2006.

[10] A. M. Frieze, G. Galbiati, and F. Maffioli, "On the worst-case performance of some algorithms for the asymmetric traveling salesman problem," *Networks*, vol. 12, no. 1, pp. 23–39, 1982.

[11] M. Bläser, "A new approximation algorithm for the asymmetric TSP with triangle inequality," *ACM Transactions on Algorithms*, vol. 4, no. 4, 2008.

[12] H. Kaplan, M. Lewenstein, N. Shafrir, and M. Sviridenko, "Approximation algorithms for asymmetric TSP by decomposing directed regular multigraphs," *J. ACM*, vol. 52, no. 4, pp. 602–626, 2005.

[13] U. Feige and M. Singh, "Improved approximation ratios for traveling salesperson tours and paths in directed graphs," in *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, 10th International Workshop, APPROX 2007, and 11th International Workshop, RANDOM 2007*, 2007, pp. 104–118.

[14] A. Asadpour, M. X. Goemans, A. Madry, S. O. Gharan, and A. 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*, 2010, pp. 379–389.

[15] S. O. Gharan and A. Saberi, "The asymmetric traveling salesman problem on graphs with bounded genus," in *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2011*, 2011, pp. 967–975.

[16] N. Anari and S. O. Gharan, "Effective-resistance-reducing flows, spectrally thin trees, and asymmetric TSP," *CoRR*, vol. abs/1411.4613, 2014.

[17] ——, "The kadison-singer problem for strongly rayleigh measures and applications to asymmetric TSP," *CoRR*, vol. abs/1412.1143, 2014.

[18] A. Marcus, D. A. Spielman, and N. Srivastava, "Interlacing families II: Mixed characteristic polynomials and the kadison-singer problem," 2013.

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