Laboratoire d'Analyse et Modélisation de Systèmes pour l'Aide à la Décision
CNRS UMR 7243

CAHIER DU LAMSADE

303

Mars 2011

Exponential approximation schemata for some network design problems

N. Boria, N. Bourgeois, B. Escoffier, V. Th. Paschos

Exponential approximation schemata for some network design problems*

Nicolas Boria[1] Nicolas Bourgeois[1] Bruno Escoffier[1] Vangelis Th. Paschos[1,2]

[1]LAMSADE, CNRS and Université Paris-Dauphine, France
{boria,bourgeois,escoffier,paschos}@lamsade.dauphine.fr
[2]Institut Universitaire de France

March 1, 2011

Abstract

We study approximation of some well-known network design problems such as TRAVELING SALESMAN PROBLEM (for both minimization and maximization versions) and MIN STEINER TREE, by moderately exponential algorithms. The general goal of the issue of moderately exponential approximation is to catch-up on polynomial inapproximability, by providing algorithms achieving, with worst-case running times importantly smaller than those needed for exact computation, approximation ratios unachievable in polynomial time.

Keywords: Exponential algorithms; Approximation algorithms; Steiner tree; Traveling Salesman Problem

1 Introduction

Among network design problems, TRAVELING SALESMAN PROBLEM and MIN STEINER TREE have been extensively studied in combinatorial optimization, due to both their numerous practical applications and their theoretical interest. Among these works, many results deal with complexity and polytime approximation of these NP-hard problems. In particular, even in restricted versions, these problems are known to be APX-hard. Moreover, many results have been obtained in the paradigm the exact and/or parameterized computation for these problems. The goal of this paper is to explore their approximability in superpolynomial or moderately exponential time. Roughly speaking, if a given problem is solvable in time say $O^*(\gamma^n)$ but is NP-hard to approximate within some ratio r, we seek r-approximation algorithms with complexity - significantly - lower than $O^*(\gamma^n)$. This issue has already been considered for several other problems such as MINIMUM SET COVER [10, 6], MIN COLORING [5], MAX INDEPENDENT SET and MIN VERTEX COVER [4], MIN BANDWIDTH [11, 15], ... Similar issues arise in the field of FPT algorithms, where approximation notions have been introduced, for instance, in [12, 7].

Among the several natural questions occurring in this setting, two keep most of our attention in this work:

- The first one deals with ratios close to 1. For an APX-hard problem solvable to optimality in time $O^*(\gamma^n)$, can we find, for *any* $\epsilon > 0$, a $1+\epsilon$ (or $1-\epsilon$, if we handle maximization problems) approximation algorithm working in time $O^*(\gamma_\epsilon^n)$, where $\gamma_\epsilon < \gamma$?
- The second one deals with "worse" ratios. Given a polytime r-approximation algorithm, can we reach ratios better than r with "low" exponential running times, in particular ratios $r+\epsilon$ (or $r-\epsilon$) in time $O^*(\gamma_\epsilon^n)$ where $\gamma_\epsilon \to 1$ when $\epsilon \to 0$?

In both TRAVELING SALESMAN PROBLEM and MIN STEINER TREE, a complete undirected graph $G = (V, E)$ is given, together with a distance or cost $c(e) \geqslant 0$ for each edge $e \in E$. The instance is said to be metric

*Research partially supported by the French Agency for Research under the DEFIS program TODO, ANR-09-EMER-010.

if c satisfies the triangle inequality, i.e., $c(u,v) + c(v,w) \geqslant c(u,w)$ for any three vertices u, v, w. For MIN STEINER TREE, a subset $S \subseteq V$ of *terminal vertices* (or simply *terminals*) is given. A Steiner tree $T = (V', E')$ is a subgraph of G ($V' \subseteq V$ and $E' \subseteq E$) which is a tree spanning all the terminals, i.e., $S \subseteq V'$. The cost of the tree T is $c(T) = \sum_{e \in E'} c(e)$. The goal of the problem is to find a Steiner tree of minimum cost. Note that the instances are usually assumed to be metric since, otherwise, one can easily transform it into an equivalent metric one by replacing the cost of each edge $e = (u, v)$ by the cost of a shortest path between u and v. The problem is known to be NP-hard and even APX-hard [1]. The feasible solution consisting of computing a minimum spanning tree on the subset of terminals is a trivial 2-approximate solution, and several works managed to improve this ratio, up to the polytime approximation algorithm of [24] guaranteeing a ratio $1 + \ln(3)/2 < 1.55$. Dealing with exact computation, the problem is known to be solvable in $O^*(1.36^n)$ (and exponential space) and in $O^*(1.62^n)$ in polynomial space [13]. Moreover, the problem is fixed parameter tractable when the parameter is the number of terminal $k = |S|$: it is solvable in time $O^*((2+\epsilon)^k)$, for any $\epsilon > 0$, and exponential space [20]. In the case of bounded edge costs, these results can be improved: the problem is solvable in time $O^*(1.36^n)$ and polynomial space, and in time $O^*(2^k)$ and polynomial space [22].

A solution of TRAVELING SALESMAN PROBLEM is a tour (Hamiltonian cycle) Γ on the vertices. The cost of Γ is the sum of the cost of its edges. A lot of different versions of TRAVELING SALESMAN PROBLEM have been studied, including maximization and minimization versions dealing with graphs with general or metric costs. These four versions are NP-hard and even APX-hard, but have different behaviors in terms of polytime approximation. In terms of exact computation, it is well known that by a dynamic programming algorithm, all the mentioned versions of TRAVELING SALESMAN PROBLEM are solvable in time and space $O^*(2^n)$ [17]. Note that though this result has not been improved so far, very recently a major breakthrough has been obtained in [2] where the Hamiltonian cycle problem is solved in $O^*(1.66^n)$. Dealing with polynomial space, the best running time known so far for TRAVELING SALESMAN PROBLEM is $O^*(4^n n^{\log n})$ reached by the algorithm in [16] (see [3]).

In this article, we study in Section 2 the possibility to get ratios arbitrary close to 1 with a running time better than the one of exact computation. We handle MIN STEINER TREE and some versions of TRAVELING SALESMAN PROBLEM. In both cases, the basic idea is to find a small part of the instance verifying some suitable properties, then to solve the instance on the remaining part and to build finally a global solution. In Section 3, we show how one can take advantage of the possible existence of some polytime r-approximation algorithm in order to reach interesting (though exponential) running times for ratios slightly better than r.

In what follows, given a graph $G = (V, E)$, we denote by n the number of its vertices ($|V| = n$) and by m the number of its edges ($|E| = m$).

2 Obtaining ratios arbitrarily close to 1

2.1 MIN STEINER TREE

Let (G, S, c) be an instance of MIN STEINER TREE where G is the input graph, $S \subseteq V$ is the set of terminals and c is the edge cost function. We denote $k = |S|$ the number of terminals. We will propose approximation algorithms that works in time smaller than exact ones. We will present the results according to the best algorithms known so far and given in [22]: when costs are bounded we call `ExactST-PARAM`(G, S) the algorithm that computes an optimum Steiner tree in time $O^*(2^k)$ and `ExactST`(G, S) the one with time complexity $O^*(1.36^n)$. Since we use as subroutines these algorithms, we also make the assumption that costs are bounded. However, it is easy to see that similar results would be obtained by considering other exact algorithms (valid on any edge costs and/or with other time complexity).

We first show Lemma 1. Finding subtrees (of a given tree) satisfying some properties has already been used to achieve interesting running times for MIN STEINER TREE in [13]. Here, we adapt this idea to our moderately exponential approximation setting by showing that there exists a subtree of an optimal solution of "small" cost but containing a "large" number of terminals. If T is a tree rooted at r and v is a vertex in T, then $T(v)$ is the subtree of T rooted at v (consisting of v and its descendants).

Lemma 1 *Let T be a Steiner tree (with k terminals). For any $p \leqslant k$, there exists a subtree $T' \subseteq T$ containing at least p and at most $3p$ terminals, and whose cost is at most $3c(T)p/k$.*

Proof. If $k \leqslant 3p$ then $T' = T$ works. Otherwise, root the tree at some vertex r and consider the following procedure.

1. Set $v \leftarrow r$.
2. While there exists a son u of v such that $T(u)$ contains at least $p+1$ terminals, set $v \leftarrow u$.
3. Now, there exists a subset of sons $u_1, \ldots, u_i$ such that the subtree rooted at v but restricted to the sons $u_1, \ldots, u_i$ contains at least $p+1$ and at most $2p+1$ terminals (or even at most $2p$ if v is not terminal). Remove this subtree from the tree (but keep v if v has other sons) and go back to step 1 if the remaining tree has more than $3p$ terminals.

In this way, we build a set of subtrees $T'_1, \ldots, T'_z$ whose edge sets $E_1, \ldots, E_z$ are a partition of the edges of T, and such that each T'_i, $i = 1, \ldots, z$, contains at most $2p+1$ and at least $p+1$ terminals, except for the last one that contains at least p and at most $3p$ terminals. Furthermore, a subtree T'_i has at most one vertex (its root) which might be common with some T'_j with $j > i$; hence, to each subtree, it can be associated at least p terminals with no overlap. Thus:

- $c(T) = \sum_{i=1}^{z} c(T'_j)$
- $k/(3p) \leqslant z \leqslant k/p$

and finally: $\exists j, c(T'_j) \leqslant c(T) \times \frac{3p}{k}$. □

Remark 1 *The following facts, that we will need later, hold:*

1. *There exists a subtree $T' \subseteq T$ containing at least p and at most $3p$ terminals, and whose cost is* at least $c(T)p/k$.
2. *Dealing with the total number of nodes n_T in the Steiner tree, for any $p \leqslant n_T$, there exists a subtree $T' \subseteq T$ containing p vertices and whose cost is at most $3c(T)p/n_T$.*

Item 2 is obtained using a similar proof as for Lemma 1. It gives the existence of such a subtree T' with at least p and at most $3p$ vertices, but we can remove vertices in T' until it contains exactly p vertices.

Based upon Lemma 1, we propose an approximation algorithm, called `SchemaST-PARAM`, which computes an $(1+\epsilon)$-approximate solution in time smaller than $O^*(2^k)$. This algorithm first identifies, thanks to Lemma 1, a part of the instance with a partial solution of small cost, contract the graph, apply an exact algorithm on the contracted graph, and patch the two partial solutions. `SchemaST-PARAM` works as follows:

1. For any subset $W \subset S$ of size at least $p = \epsilon k/3$ and at most ϵk run `ExactST-PARAM`(G, W). Let W_0 be the subset whose Steiner tree $T_0 =$ `ExactST-PARAM`(G, W_0) has minimum cost.
2. Build the contraction graph G': all the nodes from T_0 are replaced by a single node t_0. For any $u \in V \setminus T_0$, $c(t_0, u) = \min_{v \in T_0} c(v, u)$.
3. Compute $T_1 =$ `ExactST-PARAM`$(G', S \cup t_0 \setminus W_0)$.
4. Build $T_0 \cup T_1$, that means the subgraph of G that contains all the vertices and edges of T_0 and $T_1 \setminus t_0$ and for any edge (t_0, u) from T_1 contains the edge of minimum cost between u and T_0.
5. Return a spanning tree of $T_0 \cup T_1$.

Proposition 1 *When costs are bounded, for any $\epsilon \leqslant 1/5$, `SchemaST-PARAM` returns a $(1+\epsilon)$ approximation for* MIN STEINER TREE *with running time $O^*(2^{(1-\epsilon/3)k})$ and polynomial space.*

Proof. First, since t_0 is a terminal of T_1, $T_0 \cup T_1$ is connected, and it is clear that the computed tree contains all the terminals. Thus, the algorithm produces a Steiner tree of (G, S).

The complexity of step 1 of the algorithm is $O^*\left(\binom{k}{\epsilon k} 2^{\epsilon k}\right)$, while in step 3 we solve the problem on an instance with at most $k(1-\epsilon/3)$ terminals, hence in time $O^*\left(2^{k-k\epsilon/3}\right)$. But using Stirling's formula, we know that $\binom{k}{\epsilon k} = O^*\left(\frac{1}{(\epsilon^\epsilon(1-\epsilon)^{1-\epsilon})^k}\right)$. Since $\frac{2^\epsilon}{\epsilon^\epsilon(1-\epsilon)^{1-\epsilon}} \leqslant 2^{1-\epsilon/3}$ for $\epsilon \leqslant 1/5$ (actually, even for a little bit more than $1/5$), we have $\binom{k}{\epsilon k} 2^{\epsilon k} = O^*(2^{k-\epsilon k/3})$ for $\epsilon \leqslant 1/5$.

Let us now consider the approximation ratio achieved by `SchemaST-PARAM`. According to Lemma 1, and since we check among other subtrees the lightest subtree of the optimal, the cost of T_0 is at most $3c(\text{opt}(G,S))p/k = c(\text{opt}(G,S))\epsilon$. On the other hand, the optimum solution on the contraction graph cannot be heavier than the optimal solution in the initial instance, hence $c(T_0) \leqslant c(\text{opt}(G,S))$. Finally:

$$c\left(T_0 \cup T_1\right) \leqslant c\left(T_0\right) + c\left(T_1\right) \leqslant c\left(\text{opt}(G,S)\right)(1+\epsilon)$$

that completes the proof. □

We now show that this parameterized result allows the achievement of a similar result when we seek complexity results dealing with the total number of vertices. We will use the item 2 of Remark 1. We consider the following algorithm `Schema-ST`:

1. If $k \leqslant 3n/8$, then run `ExactST-PARAM`(G,S) and return the solution computed.
2. Otherwise:
   (a) For any subset $W \subset V$ of size $p = \epsilon n/8$ compute a minimum cost spanning tree on $G[W]$. Let W_0 be the subset whose spanning tree T_0 on W_0 has minimum cost.
   (b) Build the contraction graph G': all the nodes from W_0 are replaced by a single node t_0. For any $u \in V \setminus W_0$, $c(t_0,u) = \min_{v \in W_0} c(v,u)$.
   (c) Set $S' = S \cup t_0 \setminus W_0$ if W_0 contains a terminal vertex, otherwise set $S' = S \setminus W_0$. Compute $T_1 =$ `ExactST`(G',S').
   (d) Build $T_0 \cup T_1$, i.e., the subgraph of G that contains all the vertices and edges of T_0 and $T_1 \setminus t_0$ and for any edge (t_0,u) from T_1 (if any), $T_0 \cup T_1$ contains the edge of minimum cost between u and T_0.
   (e) Return a tree of $T_0 \cup T_1$ which spans all the terminals.

We denote by $\gamma \in [1.35, 1.36]$ a constant such `ExactST` works in time $O^*(\gamma^n)$.

Proposition 2 *When costs are bounded, for any $\epsilon \leqslant 3/5$ `Schema-ST` returns an $(1+\epsilon)$ approximation for* MIN STEINER TREE *with running time $O^*(\gamma^{(1-\epsilon/8)n}) \leqslant O^*(1.36^{(1-\epsilon/8)n})$ and polynomial space.*

Proof. In step 1 of `Schema-ST` ($k \leqslant 3n/8$), the returned solution is optimal and is obtained in time $O^*(2^k) = O^*(2^{3n/8}) = O^*(\gamma^{(1-\epsilon/8)n})$ for $\epsilon \leqslant 3/5$. Note that a slightly better result could be obtained using `SchemaST-PARAM` instead of the exact algorithm.

In step 2 ($k \geqslant 3n/8$), the algorithm obviously returns a feasible solution. Since computing a minimum cost spanning tree can be done in polynomial time, the running time of step 2a is $O^*(\binom{n}{\epsilon n/8})$, while in step 2c we solve the problem in an instance with $n(1-\epsilon/8)$ nodes, hence in time $O^*\left(\gamma^{n(1-\epsilon/8)}\right)$. But using Stirling's formula, we have $\binom{n}{\epsilon n/8} = O^*(\gamma^{n(1-\epsilon/8)})$ for $\epsilon \leqslant 3/5$.

Dealing with approximation ratio, note that in step 2 we have $k \geqslant 3n/8$, hence an optimum Steiner tree T contains n' vertices where $3n/8 \leqslant n' \leqslant n$. Using Remark 1, there exists a subtree of T with cost at most $\epsilon c(T)$ on $\epsilon n'/3$ vertices. Since $\epsilon n/8 \leqslant \epsilon n'/3$, by possibly removing vertices, there exists a subtree of T with cost at most $\epsilon c(T)$ on $\epsilon n/8$ vertices. Hence T_0 has cost at most $\epsilon c(T)$. As previously, the optimum solution on the contraction graph cannot be heavier than the optimal solution in the initial instance, hence $c(T_0) \leqslant c(T) = c(\text{opt}(G,S))$. Finally:

$$c\left(T_0 \cup T_1\right) \leqslant c\left(T_0\right) + c\left(T_1\right) \leqslant c(\text{opt}(G,S))(1+\epsilon)$$

and the proof of the proposition is completed. □

2.2 Traveling salesman problems

As mentioned in Section 1, TRAVELING SALESMAN PROBLEM (in both minimization and maximization versions) is solvable in time and space $O^*(2^n)$ by dynamic programming [17]. In polynomial space, the best running time known so far is $O^*(4^n n^{\log n})$ reached by the algorithm in [16] (see [3]). We present in this section two approximation algorithms `SchemaTSP-ES` and `SchemaTSP-PS` that provide for some versions of TRAVELING SALESMAN PROBLEM a $(1+\epsilon)$-approximate solution (or $(1-\epsilon)$ if we deal with a maximization

version). SchemaTSP-ES works in time $O^*(2^{(1-\Theta(\epsilon))n})$ and exponential space, and SchemaTSP-PS works in time $O^*(4^{(1-\Theta(\epsilon))n}n^{\log n})$ and polynomial space.

First, it is easy to see that the exact algorithms mentioned above for TRAVELING SALESMAN PROBLEM can be adapted to work within the same running time and space for the optimum- (minimum- or maximum-) cost Hamitonian path problem when both endpoints are fixed. Denote by ExactHP-PS(G,s,t) and ExactHP-ES(G,s,t) two algorithms that compute an optimum-cost Hamiltonian path between s and t, respectively, in time $O^*(4^n n^{\log n})$ and polynomial space, and in time $O^*(2^n)$ and exponential space.

Let us first describe SchemaTSP-PS. It depends on a parameter p (the value of which depends on the version of TRAVELING SALESMAN PROBLEM) and uses ExactHP-PS(G,s,t) as a subroutine. SchemaTSP-PS works as follows:

1. For any subset $U \subset S$ of size p, run ExacstTSP-PS$(G[U],u,v)$ for any pair of vertices $(u,v) \in U \times U$. Let U_0, u^*, v^* be the subset and the vertices whose optimum-cost Hamiltonian path $\Gamma_0 =$ ExactHP-PS$(G[U_0],u^*,v^*)$ has optimum cost.
2. Fix $G' = G[(V \setminus U_0) \cup \{u^*, v^*\}]$.
3. Compute $\Gamma_1 =$ ExactHP-PS(G',u^*,v^*).
4. Return $\Gamma_0 \cup \Gamma_1$.

Algorithm SchemaTSP-ES is similar to SchemaTSP-PS up to the following modifications:

- In Step 1, use dynamic programming to compute for any subset $U \subset S$ of size p and for any two vertices $u, v \in U$ an optimal Hamiltonian path between u and v in $G[U]$.
- In Step 3, use ExactHP-ES(G,s,t) instead of ExactHP-PS(G,s,t).

Lemma 2 *The following properties hold:*

- *If $p \leqslant n/5$, SchemaTSP-ES produces an Hamiltonian cycle in time $O^*(2^{n-p})$ (and exponential space).*
- *If $p \leqslant n/4$, SchemaTSP-PS produces an Hamiltonian cycle in time $O^*(4^{n-p}n^{\log n})$ and polynomial space.*

Proof. Since Γ_0 and Γ_1 are Hamiltonian paths between u^* and v^* on two subgraphs that intersect only in the endpoints u^* and v^* of Γ_0 and Γ_1, $\Gamma_0 \cup \Gamma_1$ is a Hamiltonian cycle on G.

For SchemaTSP-ES, Step 1 works in time $O^*(\binom{n}{p})$ (for $p \leqslant n/2$). Indeed, note that we do not apply the exact exponential space algorithm on each subinstance of size p (that would lead to a complexity $O^*(\binom{n}{p}2^p)$) but by standard dynamic programming we compute for any subset U of size at most p and any vertices $u, v \in U$ an optimum solution in time $O^*(\sum_{i=1}^{p}\binom{n}{i}) = O^*(\binom{n}{p})$ for $p \leqslant n/2$. Step 3 takes time $O^*(2^{n-p})$. Using Stirling's formula, when $p \leqslant n/5$ (actually even for p a little bit greater than $n/5$), this leads to the fact that the global running time is then $O^*(2^{n-p})$.

The total running time of SchemaTSP-PS is $O^*\left(\binom{n}{p}4^p p^{\log p} + 4^{n-p}n^{\log n}\right)$. Using Stirling's formula, we get that $\binom{n}{p}4^p = O(4^{n-p})$ for $p \leqslant n/4$, hence the running time is $O^*(4^{n-p}n^{\log n})$ for $p \leqslant n/4$. □

Let Γ^* be an optimum solution for the problem dealt with.

Lemma 3 *If we deal with a minimization problem (resp., a maximization problem) then $c(\Gamma_0) \leqslant pc(\Gamma^*)/n$ (resp., $c(\Gamma_0) \geqslant pc(\Gamma^*)/n$).*

Proof. Since we try any possible subsets of vertices of size p, we try in particular all the p-subsequences of consecutive vertices from Γ^*; the lightest (resp., heaviest) of these subsequences has cost at most (resp., at least) $pc(\Gamma^*)/n$. □

Now, we use this algorithm to several versions of TRAVELING SALESMAN PROBLEM. The first one is the famous MIN METRIC TSP, where the costs satisfy the triangle inequality: $c(u,v) \leqslant c(u,x) + c(x,v)$ for any vertices u, v, x.

Proposition 3 *It is possible to compute a $(1+\epsilon)$-approximation for* MIN METRIC TSP*:*

- *in time $O^*(2^{(1-\epsilon/2)n})$, for any $\epsilon \leqslant 2/5$.*

- *in time* $O^*(4^{(1-\epsilon/2)n}n^{\log n})$ *and polynomial space, for any* $\epsilon \leqslant 1/2$.

Proof. Let $\epsilon \leqslant 1$, and run `SchemaTSP-ES` with $p = n\epsilon/2$. Then $p \leqslant n/5$ and thanks to Lemma 2 the running time is $O^*(2^{n-p}) = O^*(2^{(1-\epsilon/2)n})$.

Consider now an optimal solution Γ^*. By the triangle inequality, if Γ' is an optimal solution for TRAVELING SALESMAN PROBLEM on G', then $c(\Gamma') \leqslant c(\Gamma^*)$.

Let u' and v' be the predecessors of u^* and v^*, respectively, in Γ' (oriented arbitrarily). Then, removing from Γ' the edges (u', u^*), (v^*, v') and adding the edge (v', u') builds a Hamiltonian path in G' between u^* and v^* of cost $c(\Gamma') + c(v', u') - c(u', u^*) - c(v^*, v')$, hence:

$$c(\Gamma_1) \leqslant c(\Gamma') + c(v', u') - c(u', u^*) - c(v^*, v') \leqslant c(\Gamma') + c(u^*, v^*) \leqslant c(\Gamma') + c(\Gamma_0)$$

where the last inequalities follow from the triangle inequality. Then, using Lemma 3 we get:

$$c(\Gamma_0 \cup \Gamma_1) \leqslant c(\Gamma') + 2c(\Gamma_0) \leqslant c(\Gamma^*)\left(1 + \frac{2p}{n}\right) = c(\Gamma^*)(1+\epsilon)$$

For the result in polynomial space, the proof of the ratio is the same. The running time follows from Lemma 2 since, for $\epsilon \leqslant 1/2$, $p \leqslant n/4$. □

We now show that a similar result holds for the TRAVELING SALESMAN PROBLEM when costs are restricted to be integers between 1 and a fixed integer $k \geqslant 2$, both in MIN TSP-k and in MAX TSP-k. Note that these versions are APX-hard even for $k = 2$ [23].

Proposition 4 *It is possible to compute a* $(1+\epsilon)$*-approximation for* MIN TSP-k*:*

- *in time* $O^*(2^{(1-\epsilon/(k-1))n})$*, for any* $\epsilon \leqslant (k-1)/5$.
- *in time* $O^*(4^{(1-\epsilon/(k-1))n}n^{\log n})$ *and polynomial space, for any* $\epsilon \leqslant (k-1)/4$.

Proof. Let $\epsilon \leqslant (k-1)/5$, and run `SchemaTSP-ES` with $p = (\epsilon n - k)/(k-1)$. Then $p \leqslant \epsilon n/(k-1) \leqslant n/5$ and thanks to Lemma 2 the running time is $O^*(2^{n-p})$. Since $p \geqslant n\epsilon/(k-1) - 2$, we get the claimed running time.

Consider now an optimal solution Γ^*. This solution contains $2x \leqslant 2p$ edges with one endpoint in U_0 and one in $V \setminus U_0$. If we remove from Γ^* all the vertices in U_0 (and their adjacent edges), we get x paths (possibly consisting of one unique vertex) in $V \setminus U_0$. We build a Hamiltonian path on vertices in $V \setminus U_0$ by adding $x-1$ edges between these paths, and then by adding two more edges we get a Hamiltonian path Γ' between u^* and v^* in G'. To build Γ' from Γ^*, at least $2x$ edges have been removed, and $x+1$ edges have been added. Since costs are between 1 and k, and using the fact that $c(\Gamma^*) \geqslant n$, we get:

$$c(\Gamma_1) \leqslant c(\Gamma') \leqslant c(\Gamma^*) + (x+1)k - 2x \leqslant c(\Gamma^*)\left(1 + \frac{pk - 2p + k}{n}\right)$$

Using Lemma 3, we get:

$$c(\Gamma_0 \cup \Gamma_1) \leqslant c(\Gamma^*)\left(1 + \frac{pk - p + k}{n}\right) = c(\Gamma^*)(1+\epsilon)$$

For the result in polynomial space, the proof of the ratio is the same. The running time follows from Lemma 2 since for $\epsilon \leqslant (k-1)/4$, $p \leqslant n/4$. □

The case of MAX TSP-k is similar to the previous one.

Proposition 5 *It is possible to compute a* $(1-\epsilon)$*-approximation for* MAX TSP-k*:*

- *in time* $O^*(2^{(1-\epsilon/(2(k-1)))n})$*, for any* $\epsilon \leqslant 2(k-1)/5$.
- *in time* $O^*(4^{(1-\epsilon/(2(k-1)))n}n^{\log n})$ *and polynomial space, for any* $\epsilon \leqslant (k-1)/2$.

Proof. Let $\epsilon \leqslant 2(k-1)/5$, and run `SchemaTSP-ES` with $p = \epsilon n/(2(k-1))$. Then $p \leqslant n/5$ and thanks to Lemma 2 the running time is $O^*(2^{n-p})$ which is the claimed running time.

Consider an optimal solution Γ^*. This solution contains $2x \leqslant 2p$ edges with one endpoint in U_0 and one in $V \setminus U_0$ and contains y edges with both endpoints in U_0, with $x + y = p$. If we remove from Γ^*

all the vertices in U_0 we get x paths (possibly consisting of one unique vertex) in $V \setminus U_0$. We build a Hamiltonian path Γ' between u^* and v^* in G' by linking these paths, u^* and v^* by adding $x+1$ edges. In all, $2x+y$ edges of cost at most k have been removed while $x+1$ have been added, for a total cost difference of at most $(2x+y)k - x - 1 \leqslant 2pk - p$. Hence:

$$\begin{aligned} c(\Gamma_1) &\geqslant c(\Gamma') \geqslant c(\Gamma^*) - p(2k-1) \geqslant c(\Gamma^*)\left(1 - \frac{p(2k-1)}{n}\right) \\ c(\Gamma_0 \cup \Gamma_1) &\geqslant c(\Gamma^*)\left(1 - \frac{p(2k-1)}{n}\right) + \frac{c(\Gamma^*)\,p}{n} = c(\Gamma^*)(1-\epsilon) \end{aligned}$$

For the result in polynomial space, the proof of the ratio is the same. The running time follows from Lemma 2 since for $\epsilon \leqslant (k-1)/2$, $p \leqslant n/4$. □

3 Improving ratios obtained with polytime algorithms

In this section, we try to take advantage of the existence for MIN STEINER TREE and for some versions of TRAVELING SALESMAN PROBLEM of polytime approximation algorithms. If there exists a polytime r-approximation algorithm, we try to get ratios $r - \epsilon$ in time $O^*(\gamma_\epsilon^n)$ (or $O^*(\gamma_\epsilon^k)$ for MIN STEINER TREE) where $\gamma_\epsilon \to 1$ when $\epsilon \to 0$. A first idea is to modify the algorithms in the previous sections in order to use an approximation algorithm instead of an exact one when computing a solution on the big part of the instance. We will see that this simple idea works for MIN STEINER TREE. However, this generally does not lead to interesting results for TRAVELING SALESMAN PROBLEM because in many cases better ratios are known for TRAVELING SALESMAN PROBLEM than for minimum-cost Hamiltonian path when both endpoints are fixed.

3.1 MIN STEINER TREE

MIN STEINER TREE is approximable in polynomial time within ratio $r_{\text{st}} = 1 + \ln(3)/2$ [24]. Denote by `ApxST` the algorithm reaching this ratio. Recall that `ExactST-PARAM` returns an optimal solution in time $O^*(2^k)$ when costs are bounded. Let us consider the following algorithm, called `ExpApxST`.

1. For any subset $W \subset S$ of size at least $p = \epsilon k/3$ and at most ϵk:
   (a) Compute $T_0 =$`ExactST-PARAM`(G, W).
   (b) Build the contraction graph G': all the nodes from T_0 are replaced by a single node t_0. For any $u \in V \setminus T_0$, $c(t_0, u) = \min_{v \in T_0} c(v, u)$.
   (c) Compute $T_1 =$ `ApxST`$(G', S \cup t_0 \setminus W)$.
   (d) Build $T_0 \cup T_1$, that means the subgraph of G that contains all the vertices and edges of T and $T_1 \setminus t_0$ and for any edge (t_0, u) from T_1 contains the edge of minimum cost between u and T_0.
   (e) Consider a spanning tree T_W of $T_0 \cup T_1$.
2. Output the best among the solutions T_W computed.

Proposition 6 *When costs are bounded,* ***`ExpApxST`*** *returns a $(r_{\text{st}} - (r_{\text{st}} - 1)\epsilon/3)$-approximate solution in time $O^*(\binom{k}{\epsilon k} 2^{\epsilon k})$ and polynomial space.*

Proof. The running time of the algorithm is $O^*(\binom{k}{\epsilon k} 2^{\epsilon k}) = O^*\left((2^\epsilon/\epsilon^\epsilon(1-\epsilon)^{1-\epsilon})^k\right)$. Consider an optimum solution T. Thanks to Remark 1 after Lemma 1, there is a subtree $T' \subseteq T$ containing at least p and at most $3p$ terminals, and whose cost is *at least* $c(T)p/k$. When W is the set of terminals of T', we have $c(T_0) \leqslant c(T')$. But contracting T' in a single vertex in T gives a feasible solution of the contracted graph G', whose cost is $c(T) - c(T')$. It follows that `ApxST` returns a solution T_1 such that $c(T_1) \leqslant r_{\text{st}}(c(T) - c(T'))$. Finally, we get:

$$\begin{aligned} c(T_0 \cup T_1) &\leqslant c(T_0) + c(T_1) \leqslant c(T') + r_{\text{st}}(c(T) - c(T')) \\ &\leqslant r_{\text{st}} c(T) - (r_{\text{st}} - 1)\, c(T)\frac{p}{k} = \left(r_{\text{st}} - (r_{\text{st}} - 1)\,\frac{\epsilon}{3}\right) c(T) \end{aligned}$$

as claimed. □

Setting $\epsilon' = (r_{\text{st}} - 1)\epsilon/3$, this is a $(r_{\text{st}} - \epsilon')$ approximation in time $O^*(\gamma_{\epsilon'}^k)$, where $\gamma_{\epsilon'}$ goes to 1 when ϵ' goes to 0. For instance, this gives a 1.547-approximation ($r_{\text{st}} \sim 1.5493$) in time $O^*(1.08^k)$.

3.2 Traveling salesman problems

As mentioned before, the same simple idea does not lead to interesting results for traveling salesman problems since, in order to patch partial solutions, we need to solve minimum-cost Hamiltonian path problems, and in many cases better ratios are known for MIN TRAVELING SALESMAN PROBLEM than for minimum-cost Hamiltonian path when both endpoints are fixed. Indeed, the famous MIN METRIC TSP is approximable within ratio $3/2$ [9], but for the MIN METRIC HAMILTONIAN PATH (when both endpoints are fixed) the best known ratio achievable in polynomial time is $5/3$ [18]. For maximization versions, MAX TSP and MAX METRIC TSP are approximable with asymptotic ratios of $61/81$ and $17/20$, respectively, [18], while only a $1/2$-approximation algorithm is known for the corresponding versions of maximum-cost Hamiltonian path when both endpoints are fixed [21].

So, the idea is to use an approximation algorithm for TRAVELING SALESMAN PROBLEM to get a Hamiltonian cycle Γ_1 on G', and then to somehow combine Γ_0 and Γ_1 to get a Hamiltonian cycle on G.

Suppose that `PolyAPP` is a polytime r-approximation algorithm for a traveling salesman problem. We consider the following algorithm `MExpAPP-PS` that works for maximization versions.

- For any subset $U \subset S$ of size $p = \epsilon n$ and for any two vertices $u, v \in U$:
  - Compute $\Gamma_0 =$`ExactHP-PS`$(G[U], u, v)$.
  - Fix $G' = G[V \setminus U]$.
  - Compute $\Gamma_1 =$ `PolyAPP`(G').
  - Let Γ_1' be the path obtained from Γ_1 by removing the lightest edge (z, t) of Γ_1 and adding edges (u, z) and (t, v).
  - Consider the solution $\Gamma_0 \cup \Gamma_1'$.
- Return the best solution computed.

We also consider the exponential space version `MExpAPP-ES` where we first compute by dynamic programming for any subset $U \subset S$ of size $p = \epsilon n$ and for any two vertices $u, v \in U$, an optimum-cost Hamiltonian path between s and t in $G[U]$.

Proposition 7 *If `PolyAPP` is an r-approximation algorithm, then both `MExpAPP-PS` and `MExpAPP-ES` are $(r + \epsilon(1 - r) - O(1/n))$-approximation algorithms. `MExpAPP-PS` runs in time $O^*(\binom{n}{\epsilon n} 4^{\epsilon n} n^{\log n})$ and polynomial space. `MExpAPP-ES` runs in time $O^*(\binom{n}{\epsilon n})$ (and exponential space).*

Proof. It is easy to see that the running time of `MExpAPP-PS` is $O^*(\binom{n}{p} 4^p p^{\log p})$, while dynamic programming leads to a running time of $O^*(\binom{n}{p})$ for `MExpAPP-ES`.

Let Γ^* be an optimum solution for the problem dealt. Consider all the n sequences $(u', u, \ldots, v, v')$ of $p + 2$ consecutive vertices in Γ^* (oriented arbitrarily), and let us denote by $\Gamma^*_{u,v}$ the path from u to v in Γ^*. Then:

$$\begin{aligned} \sum_u c(\Gamma^*_{u,v}) &= pc(\Gamma^*) \\ \sum_u (c(u', u) + c(v, v')) &= 2c(\Gamma^*) \end{aligned}$$

Then, by an average based argument, for any nonnegative μ, ν, there exist u, v such that:

$$\mu c\left(\Gamma^*_{u,v}\right) - \nu\left(c\left(u', u\right) + c\left(v', v\right)\right) \geqslant \mu \frac{p}{n} c\left(\Gamma^*\right) - \nu \frac{2}{n} c\left(\Gamma^*\right) = \left(\epsilon\mu - 2\frac{\nu}{n}\right) c\left(\Gamma^*\right)$$

Note that $\nu \geqslant 0$ for n large enough ($n \geqslant 1/(1 - \epsilon)$). We consider in the sequel these vertices u and v for the following values of μ and ν:

$$\begin{aligned} \mu &= 1 - r\left(1 - \frac{1}{(1-\epsilon)n}\right) \\ \nu &= r\left(1 - \frac{1}{(1-\epsilon)n}\right) \end{aligned}$$

Obviously $c(\Gamma^*) = c(\Gamma^*_{u,v}) + c(\Gamma^*_{v,u})$. If U_0 denotes the vertices in $\Gamma^*_{u,v}$, we consider the solution built by the `MExpAPP-PS` (or `MExpAPP-ES`) for (U_0, u, v). Of course, $c(\Gamma_0) = c(\Gamma^*_{u,v})$. Moreover, since we have removed the lightest edge in Γ_1 (among the $(1-\epsilon)n$ edges in Γ_1), we have:

$$c\left(\Gamma'_1\right) \geqslant \left(1 - \frac{1}{(1-\epsilon)n}\right) c\left(\Gamma_1\right) \geqslant \left(1 - \frac{1}{(1-\epsilon)n}\right) rc\left(\Gamma^*_1\right) = \nu c\left(\Gamma^*_1\right)$$

where $c(\Gamma^*_1)$ is an optimum-cost Hamiltonian cycle in G'. Finally the returned solution S satisfies:

$$c(S) \geqslant c\left(\Gamma_0\right) + c\left(\Gamma'_1\right) \geqslant c\left(\Gamma^*_{u,v}\right) + \nu c\left(\Gamma^*_1\right)$$

Now, removing edges (u, u') and (v, v') and adding (u', v') in $\Gamma^*_{v,u}$ results in a Hamiltonian cycle in G', hence:

$$c\left(\Gamma^*_1\right) \geqslant c\left(\Gamma^*_{v,u}\right) - c\left(u, u'\right) - c\left(v, v'\right) = c\left(\Gamma^*\right) - c\left(\Gamma^*_{u,v}\right) - c\left(u, u'\right) - c\left(v, v'\right)$$

Finally:

$$\begin{aligned} c(S) &\geqslant \nu c\left(\Gamma^*\right) + \mu c\left(\Gamma^*_{u,v}\right) - \nu\left(c\left(u', u\right) + c\left(v', v\right)\right) \\ &\geqslant \nu c\left(\Gamma^*\right) + \left(\epsilon\mu - \frac{2\nu}{n}\right) c\left(\Gamma^*\right) \quad = \quad \left(r + \epsilon(1-r) - O(1/n)\right) c\left(\Gamma^*\right) \end{aligned}$$

that completes the proof. □

Apply Proposition 7 using the results of [8]: MAX TSP and MAX METRIC TSP are approximable in polynomial time with (asymptotic) ratios 61/81 and 17/20 respectively. In Tables 1 and 2 numerical results derived from Proposition 7 are given.

Ratio	0.753	0.76	0.77	0.8	0.85
Running time (exponential space)	Polynomial	1.14^n	1.29^n	1.63^n	1.96^n
Running time (polynomial space)	Polynomial	1.42^n	1.65^n	2.12^n	3.37^n

Table 1: Approximation of MAX TSP.

Ratio	0.85	0.86	0.87	0.9
Running time (exponential space)	Polynomial	1.28^n	1.49^n	1.89^n
Running time (polynomial space)	Polynomial	1.42^n	1.79^n	3.00^n

Table 2: Approximation of MAX METRIC TSP.

We conclude this section by handling MIN ASYMMETRIC TSP, the directed version of the (symmetric) MIN METRIC TSP. A $(1 + \log(n))$-approximation algorithm has been proposed in [14] (recently improved down to $0.842 \log(n)$ in [19]). We show that the algorithm of [14] can be straightforwardly adapted to get interesting tradeoffs between running time and approximation.

Proposition 8 *For any integer $k \geqslant 1$, it is possible to compute a $(1+k)$-approximation of* MIN ASYMMETRIC TSP*:*

- *in time $O^*(2^{n/k})$ and exponential space*
- *in time $O^*(4^{n/k}\left(\frac{n}{k}\right)^{\log n})$ and polynomial space*

Proof. The algorithm in [14] works as follows. Starting from $G_0 = G$, it computes in polynomial time a minimum-cost 2-factor C_0 of the graph, i.e., a minimum-cost collection of cycles such that each vertex is in exactly one cycle. Obviously, an optimum-cost Hamiltonian cycle Γ^*_0 in G_0 is a particular 2-factor, hence $c(\Gamma^*_0) \geqslant c(C_0)$. If C_0 contains only one cycle, we are done; otherwise, we choose one arbitrary vertex in each cycle of C_0, and build the subgraph G_1 of G_0 induced by these vertices. We iterate the same process, and get a 2-factor C_1 of cost $c(C_1) \leqslant c(\Gamma^*_1)$, where Γ^*_1 is an optimum-cost Hamiltonian cycle on G_1. The process ends at some step t, when C_t has only one cycle.

The union of all the cycles in all the 2-factors $C_0, \ldots, C_t$ is a strongly connected graph where all vertices have even in-degree and even out-degree, so by taking shortcuts, using the triangle inequality we get a Hamiltonian cycle Γ such that: $c(\Gamma) \leqslant \sum_{i=0}^{t} c(C_i) \leqslant \sum_{i=0}^{t} c(\Gamma_i^*)$. Also, by the triangle inequality, $c(\Gamma_i^*) \leqslant c(\Gamma_0^*)$; hence, $c(\Gamma) \leqslant (1+t)c(\Gamma_0^*)$.

Now, since each cycle has at least 2 vertices, the number n_i of vertices in G_i satisfies $n_i \leqslant \lfloor n_{i-1}/2 \rfloor$, and this leads to $n_i \leqslant n_0/2^i$.

Here, instead of repeating this process until C_i has only one cycle, we repeat it k steps (of course unless the algorithm has stopped before), and then we compute an exact solution on G_k in time $O^*(2^{n_k})$. Since $n_k \leqslant n_0/2^k = n/2^k$, the claimed running time follows. Furthermore:

$$c(\Gamma) \leqslant \sum_{i=0}^{k-1} c\left(C_i\right) + c\left(\Gamma_k^*\right) \leqslant kc\left(\Gamma_0^*\right) + c\left(\Gamma_k^*\right) = (1+k)c\left(\Gamma_0^*\right)$$

The time bound follows from the fact that the optimum-cost asymmetric Hamiltonian path is solvable in time $O^*(2^n)$ (and exponential space), or in time $O^*(4^n n^{\log n})$ and polynomial space. □

References

[1] M. W. Bern and P. E. Plassmann. The Steiner problem with edge lengths 1 and 2. *Information Processing Letters*, 32(4):171–176, 1989.

[2] A. Björklund. Determinant sums for undirected Hamiltonicity. In *Proc. FOCS*, pages 173–182. IEEE Computer Society, 2010.

[3] A. Björklund and T. Husfeldt. Exact algorithms for exact satisfiability and number of perfect matchings. *Algorithmica*, 52(2):226–249, 2008.

[4] N. Bourgeois, B. Escoffier, and V. Th. Paschos. Efficient approximation of combinatorial problems by moderately exponential algorithms. In *Proc. WADS*, volume 5664 of *Lecture Notes in Computer Science*, pages 507–518. Springer, 2009.

[5] N. Bourgeois, B. Escoffier, and V. Th. Paschos. Efficient approximation of MIN COLORING by moderately exponential algorithms. *Information Processing Letters*, 109(16):950–954, 2009.

[6] N. Bourgeois, B. Escoffier, and V. Th. Paschos. Efficient approximation of MIN SET COVER by moderately exponential algorithms. *Theoretical Computer Science*, 410(21-23):2184–2195, 2009.

[7] L. Cai and X. Huang. Fixed-parameter approximation: conceptual framework and approximability results. In *Proc. IWPEC*, volume 4169 of *Lecture Notes in Computer Science*, pages 96–108. Springer, 2006.

[8] Z.-Z. Chen, Y. Okamoto, and L. Wang. Improved deterministic approximation algorithms for MAX TSP. *Information Processing Letters*, 95(2):333–342, 2005.

[9] N. Christofides. Worst-case analysis of a new heuristic for the traveling salesman problem. Technical report 338, Grad. School of Industrial Administration, CMU, 1976.

[10] M. Cygan, L. Kowalik, and M. Wykurz. Exponential-time approximation of weighted set cover. *Information Processing Letters*, 109(16):957–961, 2009.

[11] M. Cygan and M. Pilipczuk. Exact and approximate bandwidth. *Theoretical Computer Science*, 411(40-42):3701–3713, 2010.

[12] R. G. Downey, M. R. Fellows, and C. McCartin. Parameterized approximation problems. In *Proc. IWPEC*, volume 4169 of *Lecture Notes in Computer Science*, pages 121–129. Springer, 2006.

[13] F. V. Fomin, F. Grandoni, and D. Kratsch. Faster Steiner tree computation in polynomial-space. In *Proc. ESA*, volume 5193 of *Lecture Notes in Computer Science*, pages 430–441. Springer, 2008.

[14] A. Frieze, G. Galbiati, and F. Maffioli. On the worst-case performance of some algorithms for the asymmetric traveling salesman problem. *Networks*, 12:23–39, 1982.

[15] M. Fürer, S. Gaspers, and S. P. Kasiviswanathan. An exponential time 2-approximation algorithm for bandwidth. In *Proc. IWPEC*, volume 5917 of *Lecture Notes in Computer Science*, pages 173–184. Springer, 2009.

[16] Y. Gurevich and S. Shelah. Expected computation time for Hamiltonian path problem. *SIAM Journal of Computing*, 16(3):486–502, 1987.

[17] M. Held and R. M. Karp. A dynamic programming approach to sequencing problems. *SIAM Journal*, 10(1):196–210, 1962.

[18] J.A. Hoogeveen. Analysis of christofides' heuristic: Some paths are more difficult than cycles. *Operations Research Letters*, 10:291–295, 1991.

[19] H. Kaplan, M. Lewenstein, N. Shafrir, and M. Sviridenko. Approximation algorithms for asymmetric TSP by decomposing directed regular multigraphs. *Journal of the ACM*, 52(4):602–626, 2005.

[20] D. Mölle, S. Richter, and P. Rossmanith. A faster algorithm for the Steiner tree problem. In *Proc. STACS*, volume 3884 of *Lecture Notes in Computer Science*, pages 561–570. Springer, 2006.

[21] J. Monnot. Approximation algorithms for the maximum hamiltonian path problem with specified endpoint(s). *European Journal of Operational Research*, 161(3):721–735, 2005.

[22] J. Nederlof. Fast polynomial-space algorithms using Möbius inversion: improving on Steiner tree and related problems. In *Proc. ICALP*, volume 5555 of *Lecture Notes in Computer Science*, pages 713–725. Springer, 2009.

[23] C. H. Papadimitriou and M. Yannakakis. The traveling salesman problem with distances one and two. *Mathematics of Operations Research*, 18:1–11, 1993.

[24] G. Robins and A. Zelikovsky. Improved Steiner tree approximation in graphs. In *Proc. SODA*, pages 770–779, 2000.