Monograph

Knowing All Optimal Solutions Does Not Help for TSP Reoptimization

Author(s):
Böckenhauer, Hans-Joachim; Hromkovič, Juraj; Sprock, Andreas

Publication Date:
2011

Permanent Link:
https://doi.org/10.3929/ethz-a-010887457

Rights / License:
In Copyright - Non-Commercial Use Permitted
Chapter 1

Knowing All Optimal Solutions Does Not Help for TSP Reoptimization

Hans-Joachim Böckenhauer, Juraj Hromkovič, and Andreas Sprock

Department of Computer Science, ETH Zurich, 8092 Zurich, Switzerland,
E-mail: {hjb,jhromkov,andreas.sprock}@inf.ethz.ch

The concept of reoptimization models the following question: Given an instance of an optimization problem together with an optimal solution, does this knowledge help for finding a high-quality solution for a locally modified instance?

We briefly survey the known results about reoptimization and consider a generalization of the model where not only one optimal solution is given, but the set of all optimal solutions for an instance is available for free. We use the traveling salesman problem as an example to prove that there exist problems for which this additional knowledge does not help at all for improving the approximability.

1.1. Introduction

In combinatorial optimization, we are traditionally concerned with finding optimal solutions to practically relevant input instances about which nothing is known in advance. Unfortunately, finding such optimal solutions is computationally hard in many cases, and thus we use different approaches like heuristics or approximation algorithms for computing good (but not necessarily optimal) feasible solutions. But in many cases even computing a solution of satisfactory approximation quality remains a hard task.

On the other hand, in many applications, it might help to consider not only isolated input instances, but to take into account some knowledge about similar instances, for example gathered from prior solution attempts to similar instances. As an example, consider the problem of finding an optimal train schedule (for some railway network, for some objective function, and under some constraints). If we succeeded in computing such an
optimal schedule, it is natural to expect that we can profit somehow from this schedule when some additional railway station is opened and we have to find a new optimal schedule.

The concept of reoptimization formally describes this approach: Given a problem instance together with an optimal (or approximate) solution for it, and a locally modified new problem instance, what can we say about the optimal solution for the new instance? Does the knowledge about the old optimal solution help when computing a solution for the new instance? How much does it help? How much can it improve the running time or the quality of the output?

In this context, we usually measure the quality of a solution in terms of the approximation ratio, i.e., the quotient of the value of the computed solution and the optimal value. More precisely, for a minimization problem $U$ and an algorithm $A$ for $U$, the approximation ratio $r_A(n)$ for inputs of size $n$ is defined as $\max\{A(I)/Opt(I) \mid I \text{ is an instance of } U \text{ of size } n\}$. If $r_A(n)$ can be bounded from above by a constant, we say that $A$ is a constant-factor approximation algorithm for $U$. The class $APX$ contains all problems for which a constant-factor approximation algorithm exists. A polynomial-time approximation scheme (PTAS) is an algorithm that, for a given $\varepsilon > 0$, computes a $(1 + \varepsilon)$-approximate solution in time polynomial in the input size (but possibly exponential in $1/\varepsilon$). $APX$-hard problems are the hardest problems inside $APX$. For them, no PTAS exists, unless $P = NP$. For a more detailed introduction to the theory of approximation algorithms, see, e.g., [23].

In this paper, we first give a brief overview of the known reoptimization results showing that the answers to these questions are very dependent on the optimization problem and the type of local modification. On the one hand, there exist $APX$-hard optimization problems where the knowledge of an optimal solution can help to improve the achievable approximation ratio on a locally modified instance, sometimes even to design a PTAS for the reoptimization variant of the problem. On the other hand, there exist problems for which the reoptimization variant is exactly as hard as the original problem.

As the main result of this paper, we prove an even stronger inapproximability result. It is well known that the traveling salesman problem (TSP), i.e., the problem of finding a shortest Hamiltonian tour in an edge-weighted complete graph, is not approximable with an approximation ratio of 2 in an $n$-vertex graph with arbitrary edge weights [1], unless $P = NP$. We now consider a reoptimization situation where we are given a TSP instance
Knowing All Optimal Solutions Does Not Help for TSP Reoptimization

1.2. Overview of Reoptimization Results

We consider the following reoptimization scenario: Given an instance of an optimization problem together with an optimal solution, we are supposed to compute a solution for a locally modified instance. Depending on the problem, one can think of many different local modifications. In graph problems, for instance, a single edge or vertex might be added or removed or the cost of a single edge might be changed. For an optimization problem $U$ and some local modification $lm$, we denote the resulting reoptimization problem by $lm$-Reopt-$U$.

The concept of reoptimization was mentioned for the first time in a paper on the postoptimality analysis of a scheduling problem [2]. In postoptimality analysis, one studies the related question of how much a given instance of an optimization problem may be altered without changing the set of optimal solutions, see for example the paper by van Hoesel and Wagelmans [3].

Since then, reoptimization has been applied to various problems like the TSP [4–7], the Steiner tree problem [8–11], the shortest common superstring problem [12, 13], the knapsack problem [14], and several covering problems [15].

The results obtained in these papers show that the hardness of reoptimization problems varies a lot. Obviously, $lm$-Reopt-$U$ may be an easier problem than $U$, since we have access to an optimal solution for the original problem instance for free. Nevertheless, in most cases, the reoptimization variant of an $NP$-hard optimization problem remains $NP$-hard. This is due to the fact that, for most problems, it is possible to transform any given instance into a trivially solvable one by using a sequence of only polynomially many local modifications, see for example the paper by Böckenhauer et al. [16] for more details.

For some optimization problems, the reoptimization problem trivially admits a very good approximation since the old optimal solution itself is a good approximate solution for the new instance. For example, adding a single edge in an instance of a graph coloring problem can increase the cost of an optimal solution by at most one.

Let $\Delta$TSP denote the restriction of the TSP to complete edge-weighted
graphs where the edge-weights obey the triangle inequality, i.e., to instances $G = (V, E, c)$ with

$$c(\{u, v\}) \leq c(\{u, w\}) + c(\{w, v\})$$

for all $u, v, w \in V$. It is well known that the $\Delta$TSP is $\mathcal{APX}$-hard [17], and the best known approximation algorithm for it is due to Christofides [18] and achieves an approximation ratio of $3/2$. We consider the local modification of increasing the cost of a single edge in such a way that the triangle inequality is still satisfied. We denote the resulting reoptimization problem by Inc-Edge-Reopt-$\Delta$TSP. This reoptimization problem is $\mathcal{NP}$-hard [6, 19], but it admits an approximation algorithm which improves over the one from Christofides.

**Theorem 1.1** (Böckenhauer et al. [6, 19]). There exists a polynomial-time approximation algorithm for the Inc-Edge-Reopt-$\Delta$TSP with an approximation ratio of $7/5$.

The proof of this theorem is based on the following idea which can be used for several other reoptimization problems as well. The algorithm considers two possible solutions. One of these solutions is the given optimal solution for the old instance, possibly adapted slightly to become feasible for the new instance. The other solution is based on guessing (by exhaustive search) a small part of the new optimal solution which can be proven to have relatively high costs and to use some known approximation algorithm to approximate the rest of the new solution. Usually, it can be shown that the first solution is good if the local modification does not change the cost of the optimal solution too much, and the second solution can be proven to be good in the case that the value of the optimal solution changes a lot after applying the local modification.

For some problems, the concept of reoptimization can help even more in lowering the approximation ratio. There exist $\mathcal{APX}$-hard optimization problems for which some corresponding reoptimization problems admit a PTAS. As an example, we consider the Steiner tree problem on graphs with bounded edge weights. In the Steiner tree problem (STP), the input consists of a complete undirected graph with edge weights together with a subset of the vertices, called terminals. The goal is to find a minimum-weight subtree which contains all terminals (and may contain any of the non-terminal vertices). This problem is known to be $\mathcal{APX}$-hard [20], even if all edge costs are drawn from the set $\{1, 2, \ldots, r\}$ for some integer constant $r \geq 2$. By Inc-Term-Reopt-$r$-STP we denote the reoptimization variant of
the STP on graphs with edge costs from \{1, 2, \ldots, r\}, where a non-terminal vertex of the graph becomes a terminal. The Inc-Term-Reopt-\(r\)-STP is \(\mathcal{NP}\)-hard [16], but it admits a PTAS.

**Theorem 1.2 (Böckenhauer et al. [16]).** Let \(r\) be an arbitrary positive integer, \(r \geq 2\). There exists a polynomial-time approximation scheme for the Inc-Term-Reopt-\(r\)-STP.

The proof of this theorem is based on the following idea. For a desired approximation ratio of \(1 + \varepsilon\), the algorithm computes the number \(m = r \cdot \lceil 1/\varepsilon \rceil\). If there are few terminals (i.e., less than \(m\)), then the optimal Steiner tree can be computed using the Dreyfus-Wagner algorithm [21] whose running time is exponential only in the number of terminals. Otherwise, in the case of many terminals, just adding one edge from the new terminal to the old optimal solution, if necessary, gives a solution which can be proven to be \((1 + \varepsilon)\)-approximative.

On the other hand, for some problems, reoptimization does not help at all. As an example, consider the TSP with arbitrary edge weights. As already mentioned above, it is well known that the TSP is not approximable within a polynomial approximation ratio [1]. This result generalizes to the Inc-Edge-Reopt-TSP as follows.

**Theorem 1.3 (Böckenhauer et al. [6, 19]).** It is \(\mathcal{NP}\)-hard to approximate the Inc-Edge-Reopt-TSP with an approximation ratio which is polynomial in the input size.

The proof of this theorem is based on a diamond graph construction similar to the one used by Papadimitriou and Steiglitz for constructing instances of the TSP that are hard for local search [22]. We will not go into detail here since we will use a similar technique in the next section to prove an even stronger inapproximability result.

### 1.3. Knowing All Optimal Solutions

In this section, we consider a generalization of the reoptimization model. We assume that we are not only given a single optimal solution for the old instance, but the set of all (possibly exponentially many) optimal solutions is available for free. We use the general TSP as an example to show that there exist optimization problems for which even this additional knowledge...
H.-J. Böckenhauer, J. Hromkovič, and A. Sprock

March 8, 2011 11:50 World Scientific Review Volume - 9in x 6in

does not help at all for improving the approximability. We start with a formal definition of this TSP reoptimization variant.

**Definition 1.1.** We define \textit{Inc-Edge-Reopt\textsubscript{ALL}-TSP} as the problem given by a complete graph \( G = (V, E) \) with two edge weight functions \( c_{\text{old}} \) and \( c_{\text{new}} \) such that \( (G, c_{\text{old}}) \) and \( (G, c_{\text{new}}) \) are both admissible inputs for the TSP and such that \( c_{\text{old}} \) and \( c_{\text{new}} \) coincide, except for one edge \( e_{\text{change}} \in E \) where \( c_{\text{old}}(e_{\text{change}}) < c_{\text{new}}(e_{\text{change}}) \). Moreover, we are given all optimal TSP solutions for \( (G, c_{\text{old}}) \). The goal is to compute an optimal TSP solution for \( (G, c_{\text{new}}) \).

We measure the size of an instance of \textit{Inc-Edge-Reopt\textsubscript{ALL}-TSP} by the size of \( (G, c_{\text{new}}) \) only. The size needed for representing all (possibly exponentially many) optimal solutions for \( (G, c_{\text{old}}) \) is not taken into account.

For proving the inapproximability, we give a reduction from the Hamiltonian cycle problem (HC) to \textit{Inc-Edge-Reopt\textsubscript{ALL}-TSP}. The HC problem is the problem to decide whether a given undirected unweighted graph \( G \) contains a Hamiltonian cycle or not.

For this reduction we employ a diamond graph construction similar to the construction of Papadimitriou and Steiglitz [22]. We start with the definition of the diamond graph.

**Definition 1.2.** The diamond graph \( D = (V, E) \) is a graph with 8 vertices and 9 edges with \( V = \{N, S, W, E, u, x, y, z\} \) and \( E = \{\{W, x\}, \{x, N\}, \{N, u\}, \{u, z\}, \{z, S\}, \{S, y\}, \{y, E\}, \{W, z\}, \{u, E\}\} \), see also Fig. 1.1.

![Diamond graph](image)

**Fig. 1.1.** Diamond graph

For our proof, we need the following lemma from Papadimitriou and Steiglitz [22], see also Hromkovič (2003) [23].
Lemma 1.1 (Papadimitriou and Steiglitz [22]). If the diamond graph $D$ is an induced subgraph of a graph $G$ with a Hamiltonian cycle $C$, then $C$ traverses $D$ in exactly one of the following two modes:

1. from north $N$ to south $S$ (or vice versa): $\ldots, N, x, W, z, u, E, y, S, \ldots$ (see Fig. 1.2(a)), or
2. from west $W$ to east $E$ (or vice versa): $\ldots, W, x, N, u, z, S, y, E, \ldots$ (see Fig. 1.2(b)).

That is, if a cycle $C$ enters the diamond from the north, it must leave it from the south; and similarly with respect to the east-west vertices.

![Fig. 1.2. (a) Traversal of a diamond in north-south direction (b) traversal of a diamond in west-east direction](image)

Now we are ready to formulate our main result.

Theorem 1.4. Under the assumption of $\mathcal{P} \neq \mathcal{NP}$, there does not exist any polynomial-time approximation algorithm with an approximation ratio of $2^n$ for $\text{Inc-Edge-Reopt}_{ALL}$-$\text{TSP}$, where $n$ is the number of vertices in the instance.

Proof. For proving this theorem, we give a reduction from the Hamiltonian cycle problem (HC) to the $\text{Inc-Edge-Reopt}_{ALL}$-$\text{TSP}$.

Let $G_{HC}$ be an input instance for HC with $V(G_{HC}) = \{v_1, \ldots, v_k\}$ and $E(G_{HC}) = \{e_1, \ldots, e_m\}$. To construct an instance of $\text{Inc-Edge-Reopt}_{ALL}$-$\text{TSP}$, we first build an unweighted graph $G_{TSP}$. As a first step, we substitute every vertex $v_i \in V(G_{HC}), i \in \{1, \ldots, k\}$, by a
diamond graph $D_i$ with $V(D_i) = \{N_i, S_i, W_i, E_i, u_i, x_i, y_i, z_i\}$ as shown in Fig. 1.1.

Secondly, we add edges from every S- and N-vertex of every diamond $D_i$ to every S- and N-vertex of all other diamonds $D_j, j \neq i$ (see Fig. 1.3(a)). Furthermore, for every edge $e_i = \{v_j, v_k\} \in E(G_{HC})$, the corresponding diamonds $D_j$ and $D_k$ are connected with two more edges: one between the W-vertex $W_j$ of $D_j$ and the E-vertex $E_k$ of $D_k$ and one edge between $W_k$ and $E_j$ (see Fig. 1.3(b)).

![Fig. 1.3. (a) North-south connections in the graph $G_{TSP}$ (b) west-east connections in the graph $G_{TSP}$ if $\{v_j, v_k\} \in E(G_{HC})$](image)

Now we transform $G_{TSP}$ into an instance $(G_{Reopt}, c_{old}, c_{new})$ of $Inc$-$Edge$-$Reopt\text{-}ALL$-$TSP$. Let $G_{Reopt} = (V,E) = K_{8k}$ be the complete
Knowing All Optimal Solutions Does Not Help for TSP Reoptimization

graph on $8k$ vertices. We define $c_{\text{old}}$ as follows: The edge $e_{\text{block}} = \{N_1, u_1\}$ plays a special role in our argumentation, we set $c_{\text{old}}(e_{\text{block}}) = 1 + \varepsilon$. For all other edges $e_j \in \{\{v_i, v_k\} | v_i, v_k \in V(G_{TSP}) \}$, $e_j \neq e_{\text{block}}$, we define

$$c_{\text{old}}(e_j) = \begin{cases} 1 & \text{if } e_j \in E(G_{TSP}) - \{e_{\text{block}}\} \\ M & \text{otherwise,} \end{cases}$$

where $M = 2^{9k}$.

We now consider the local modification of changing the cost of one edge in $c_{\text{old}}$ leading to the new TSP instance $(G_{\text{Reopt}}, c_{\text{new}})$. For this, we change the cost of the edge $e_{\text{change}} = \{W_1, z_1\}$ in the diamond $D_1$ from $c_{\text{old}}(e_{\text{change}}) = 1$ to $c_{\text{new}}(e_{\text{change}}) = M$.

In addition to $(G_{\text{Reopt}}, c_{\text{old}}, c_{\text{new}})$, we have to specify the set of all optimal solutions for $(G_{\text{Reopt}}, c_{\text{old}})$. These are all the $2^k \cdot (k - 1)!$ many Hamiltonian tours which traverse every diamond in north-south or south-north direction.

In the graph $G_{TSP}$, independently from the original graph $G_{HC}$, there exist $2^k \cdot (k - 1)!$ many Hamiltonian cycles traversing all diamonds in north-south or south-north direction: The order of the diamonds can be varied arbitrarily, yielding $(k - 1)!$ different possibilities, and all diamonds can be traversed either in north-south or south-north direction which leads to $(2^k \cdot (k - 1)!)$ different Hamiltonian tours in north-south direction overall. Any of these Hamiltonian cycles leads to an optimal solution in $(G_{\text{Reopt}}, c_{\text{old}})$ with cost of $8 \cdot k - 1 + M$ because, for all edges $e_i \neq e_{\text{block}}$, $e_i \in E(G_{TSP})$, $c_{\text{old}}(e_i) = 1$ holds and the edge $e_{\text{block}}$ is not traversed by a run through the diamond $D_1$ in north-south direction.

Furthermore, if and only if there exists a Hamiltonian cycle in $G_{HC}$, there also exist one or more (exactly as many as in $G_{HC}$) Hamiltonian cycles in $G_{TSP}$ that traverse every diamond in west-east direction. These Hamiltonian cycles lead to second-best solutions in $G_{\text{Reopt}}$ of cost $8 \cdot k + \varepsilon$ because every traversal of the diamond $D_1$ in west-east direction has to use the edge $e_{\text{block}}$. Note that these west-east solutions do not use the edge $e_{\text{change}}$. A Hamiltonian cycle in $G_{\text{Reopt}}$ traversing some diamonds in north-south direction and some other in west-east direction use at least two edges of cost $M$ due to the construction, leading to a cost of at least $8 \cdot k - 2 + 2 \cdot M$.

By increasing the edge cost of $e_{\text{change}}$ from $c_{\text{old}}(e_{\text{change}}) = 1$ to $c_{\text{new}}(e_{\text{change}}) = M$, all optimal solutions in $(G_{\text{Reopt}}, c_{\text{old}})$ get a cost of $8 \cdot k - 1 + M$ in $(G_{\text{Reopt}}, c_{\text{new}})$. Therefore, in $(G_{\text{Reopt}}, c_{\text{new}})$ there exists an optimal solution with cost of $8 \cdot k + \varepsilon$ if and only if there is a Hamiltonian
cycles in $G_{HC}$. Otherwise, the old optimal solutions in $(G_{Reopt}, c_{old})$ stay optimal in $(G_{Reopt}, c_{new})$.

Thus, an approximation algorithm with an approximation ratio smaller than $(M + 8 \cdot k - 1)/(8 \cdot k + \varepsilon)$ would solve the HC problem. Due to $M = 2^{8k}$, we have

$$\frac{M + 8 \cdot k - 1}{8 \cdot k + \varepsilon} > 2^{8k}$$

for almost all values of $k$. Since the constructed graph $G_{Reopt}$ has $8k$ vertices, the claim follows. \qed

1.4. Conclusion

We have surveyed some of the known results about reoptimization and we have generalized the classical reoptimization model by allowing our algorithm to access not only one optimal solution for the old instance, but all optimal solutions. We have seen that the general TSP is an example of a problem where this extra knowledge does not help at all. Regarding future work, it seems to be worthwhile to consider this generalization of the reoptimization in greater detail and to investigate, how much the knowledge of all old optimal solutions can help for other reoptimization problems.

Acknowledgments

This work was partially supported by SNF grant 200021-121745/1.

References

Knowing All Optimal Solutions Does Not Help for TSP Reoptimization


H.-J. Böckenhauer, J. Hromkovič, and A. Sprock


