On Optimal Preprocessing for Contraction Hierarchies

Nikola Milosavljević
Institute for Formal Methods in Computer Science
Universitätsstr. 38, D-70569 Stuttgart, Germany
milosavljevic@informatik.uni-stuttgart.de

ABSTRACT
For some graph classes, most notably real-world road networks, shortest path queries can be answered very efficiently if the graph is preprocessed into a contraction hierarchy. The preprocessing algorithm contracts nodes in some order, adding new edges (shortcuts) in the process. While preprocessing and query algorithm work for any contraction ordering, it is desirable to use one that produces as few shortcuts as possible.

It is known that the problem of minimizing the size (number of edges) of a given graph’s contraction hierarchy is APX-hard. Also, any graph can be processed into a contraction hierarchy with at most \( O(nh \log D) \) edges, where \( n \), \( D \), and \( h \) are the number of nodes, the diameter, and the highway dimension of the original graph, respectively.

In this paper we show that the \( O(nh \log D) \) bound is tight for a wide range of parameters \( n \), \( D \), and \( h \). We also show that planar graphs, despite having highway dimension \( \Omega(\sqrt{n}) \), can be preprocessed into a graph of size \( O(n \log n) \). Finally, we present a simpler proof of APX-hardness.

Categories and Subject Descriptors
F.2.2 [Analysis of Algorithms and Problem Complexity]: Nonnumerical Algorithms and Problems

General Terms
Theory

Keywords
Shortest paths, contraction hierarchies, planar graphs, APX-hardness

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
ACM SIGSPATIAL IWCTS’12, November 6, 2012. Redondo Beach, CA, USA.
Copyright (c) 2012 ACM ISBN 978-1-4503-1693-4/12/11...$15.00

1. INTRODUCTION

1.1 Contraction Hierarchies
The contraction hierarchy, introduced by Geisberger et al. [6], is a technique for fast computation of shortest paths in road networks. It exploits the fact that roads can be assigned importance or priority which is “nearly unimodal” along shortest paths. In other words, when traveling along a shortest path, the importance generally first increases, and then decreases, with very few deviations from this rule. For example, when going from New York to Los Angeles along a shortest path, the road priorities encountered along the path are: local → arterial → state → interstate → state → arterial → local, with relatively few local deviations from this pattern. Notice that the property is independent of scale: when traveling between two points in the same city, there may be no state or interstate roads, but the priorities are still “nearly unimodal” along the path.

The idea of contraction hierarchies is to add edges to the graph to turn “near unimodality” to “true unimodality” for all shortest paths. Moreover, it should be possible to transform any shortest path computed in the modified graph into a shortest path between the same endpoints in the original graph. Then one can compute a shortest path from \( s \) to \( t \) by a restricted Dijkstra algorithm that examines only the nodes that are either reachable from \( s \) via paths of non-decreasing importance, or can reach \( t \) via paths of non-increasing importance. This tends to significantly decrease the number of examined nodes and edges compared to the standard, unrestricted algorithm.

More formally, construction of a contraction hierarchy starts by fixing a total order, or ranking, on the nodes (not edges, as in the above informal description). The graph is modified in the process of node contraction. Nodes are contracted in the order of increasing ranks. Contracting node \( v \) consists of removing \( v \) from the graph (along with the edges incident to it), and directly connecting, by a new edge called a shortcut, any two (still uncontracted) neighbors of \( v \) whose distance is increased as a result. More precisely, if \( (u,v) \) and \( (v,w) \) are edges that form the only shortest path from \( u \) to \( w \), then introduce a new edge \( (u,w) \) whose length is equal to the length of the path. The output graph consists of the original graph and all shortcuts added in the contraction procedure.

It is not hard to see that the output graph satisfies the unimodality property. Note that a path is not unimodal if and only if it has a “local minimum”, i.e., an internal node ranked lower than both of its neighbors on the path. Roughly speak-
ing, once a node has been contracted, it cannot appear as a “local minimum” on a shortest path.

It is obvious that adding shortcuts preserves distances between nodes. Moreover, given a shortest path in the modified graph, it is trivial to compute a shortest path of the same length in the original graph. Simply replace any shortcut by a two-hop path that gave rise to it in the contraction process.

Define the upward graph of \( v \) be the subgraph induced by nodes that are either reachable from \( v \) via a shortest path of non-decreasing ranks, or can reach \( v \) via a shortest path of non-increasing ranks. The upward graph of \( v \) can be computed by a Dijkstra algorithm modified so that it relaxes only outgoing upward edges (i.e., outgoing edges from lower to higher rank) as well as incoming downward edges. Given \( s \) and \( t \), unimodality implies that the union of upward graphs of \( s \) and \( t \) contains a shortest path from \( s \) to \( t \). To compute one of them, it suffices to run the Dijkstra algorithm on the union of upward graphs. This query algorithm is exact, and extremely fast on many practical instances, because upward graphs tend to be small.

To summarize, preprocessing a graph into a contraction hierarchy consists of computing a node ranking and contracting nodes according to that ranking, adding shortcuts in the process. A query algorithm consists of computing the two upward graphs and running the Dijkstra algorithm on their union. Queries are exact, and correctness does not depend on the ranking.

For a good practical performance, however, it is essential that the ranking used in preprocessing reflect as closely as possible the “true” importance of nodes. Choosing a “wrong” ranking has two effects. First, too many shortcuts may be created in preprocessing, leading to a large space requirement, and a long preprocessing time. Second, the upward graphs used in the query algorithm may be too large, leading to a poor query time.

Optimizing both of these measures over all possible rankings has been studied in the past. In this paper we consider the first problem, i.e., minimizing the number of shortcuts, which we denote by MinCH from now on.

### 1.2 Related Work

Contraction hierarchies were introduced by Geisberger et al. [6]. Bauer et al. [3] proved NP-completeness of optimal preprocessing problems for several popular fast shortest path algorithms, including contraction hierarchies. For MinCH they proved APX-hardness by an approximation-preserving reduction from the vertex cover problem, implying that for any \( \varepsilon > 0 \), it is impossible to approximate MinCH within \( 1.3666 - \varepsilon \) in polynomial time, unless \( P = NP \) [4]. Their reduction is quite complicated, and in this paper we present a much simpler one.

Abraham et al. [2] introduced a structural parameter called highway dimension. Roughly speaking, the highway dimension is the worst-case (maximum) number of nodes required to hit all shortest paths contained in some region whose length is not too small compared to the size of the region. The authors argued that real-world road networks can be modeled by networks with small highway dimension. Indeed, when traveling between two opposite ends of a city, state, country, etc., one can identify a few points that one has to go through – corners of one or more arterial roads within a city, important highway interchanges etc. The authors proved that a network with \( n \) nodes, diameter \( D \), and highway dimension \( h \) can be preprocessed into a contraction hierarchy with \( O(nh \log D) \) edges. The associated algorithm for computing node ranking runs in exponential time, as it involves computing small shortest path covers (sets of nodes that hit all shortest paths), which is a set-cover-like problem. If polynomial-time preprocessing is required, their bound increases by a factor of \( O(\log h) \) [1]. In this paper we show that the analysis of their (exponential time) preprocessing algorithm is tight up to a constant factor.

### 2. CONTRIBUTIONS

We present a family of graphs, parameterized by number of nodes \( n \), diameter \( D \), and highway dimension \( h \), on which the exponential time preprocessing algorithm of Abraham et al. [2] may in the worst case produce \( \Theta(nh \log D) \) shortcuts. This shows that the analysis of the algorithm is tight up to a constant factor.

We also prove that any planar graph with \( n \) nodes can be preprocessed into a contraction hierarchy with \( O(n \log n) \) edges. This is actually an easy application of the generalized nested dissection method of Lipton, Rose, and Tarjan [7]. The latter was originally proposed for computing good Gaussian elimination orderings for matrices defined on planar graphs. Since planar graphs may have highway dimension of \( \Omega(\sqrt{n}) \) (e.g., a square grid with unit lengths [2]), it follows that the bound of Abraham et al. is not tight in the restricted case of planar graphs.

Finally, we give a simpler proof of APX-hardness of MinCH. Our reduction is from the feedback arc set problem (denoted by FAS in the rest of the paper). The inapproximability ratio is the same as in the original proof (i.e., the same as for the vertex cover problem).

The rest of the paper is organized as follows. In Section 3 we introduce terminology and notation. In Section 4 we prove the matching lower bound for the preprocessing algorithm of Abraham et al. In Section 5 we present the upper bound for planar graphs. In Section 6 we present the APX-hardness proof. Section 7 concludes the paper with some ideas for future work.

### 3. PRELIMINARIES

In Section 1.1 computation of contraction hierarchy was described as iterative process, in terms of contracting nodes one by one. One can see that the final set of shortcuts can also be obtained by looking at any pair of nodes \( u, v \) and checking if they can be separated by removing all nodes whose rank is higher than at least one of \( u, v \). If that is the case, then the iterative process does not create an \((u, v)\), otherwise it does. Sometimes it will be helpful to have this alternative view in mind.

The length of edge \((x, y)\) is denoted by \( l(x, y) \). We write a path as sequences of nodes \((v_1, v_2, \ldots, v_i)\). Sometimes it will be convenient to repeat nodes, e.g., \((v_1, v_2, v_2, v_2, v_2, v_1, \ldots, v_1)\); this notation still refers to the same path. A path between nodes \( u \) and \( v \) is called a \( u-v \) path for short. The distance between \( u \) and \( v \), i.e., the minimum length of any \( u-v \) path, is denoted by \( d(u, v) \). The (closed) ball of radius \( r \) centered at \( u \), i.e., the set of nodes \( v \) such that \( d(u, v) \leq r \) is denoted by \( B(u, r) \). If a path contains a node from some set \( S \), we say that \( S \) covers the path.
4. LOWER BOUND

We begin with an overview of the results of Abraham et al. [2]. The highway dimension of a directed weighted graph is the smallest integer $h$ such that for all $r > 0$, and any ball $B$ of radius $4r$, there exists a set of at most $h$ nodes which covers any shortest path strictly longer than $r$ and completely contained in $B$. A set $S$ of nodes is an $(r,s)$-shortest path cover if any shortest path of length in the interval $(r,2r)$ contains a node in $S$, and any ball of radius $2r$ contains at most $s$ nodes in $S$. One can show that a network with highway dimension $h$ has an $(r,h)$-shortest path cover for all $r > 0$. The exponential-time preprocessing algorithm in [2] uses an exhaustive search to compute a $(2^h, h)$-shortest path cover $S_i$ for all $i \geq 0$. The node ranking is such that any node in $S_i$ is ranked lower than any node in $S_{i+1}$ for all $i \geq 0$; other than that, the ranking is arbitrary. The main result is that for this ranking, the preprocessing algorithm produces a contraction hierarchy of $O(nh \log D)$ shortcuts, where $n$ is the number of nodes, and $D$ is the network diameter.

The goal of this section is to show that this bound is tight for a wide range of parameters $n$, $D$, and $h$. To this end, we construct a family of tight examples (denoted by $T_{t,k}$) which cover any shortest path strictly longer than $r$ and completely contained in $B$. The family $T_{t,k}$ is parameterized by integers $t \geq 2$ and $k$. Figure 1 shows two shortest paths connecting the same pair of nodes.

**Figure 1:** The top-left figure shows $G_{2,2}$, with heavier edges representing $T_{2,2}$ (edge lengths are omitted). In the remaining figures heavier edges represent various shortest paths in $G_{2,2,2}$. The bottom row shows two shortest paths connecting the same pair of nodes.

Now we define graph $G_{t,k,q} = (V_{t,k,q}, E_{t,k,q})$. We start with $q$ copies of $G_{t,k}$, denoted by $G_{t,k,q} = (V_{t,k,q}, E_{t,k,q})$, $a = 1, 2, \ldots, q$. For an object (vertex, edge, set of vertices etc.) defined as follows. For all $u, v$ such that $u$ is an ancestor of $v$, there is an edge $(u, v)$ in $E_{t,k}$ whose length is $l(u, v) = 16^{-t}$, where $i = \lambda(u)$.

**Lemma 1.** Let $u, v \in V_{t,k}$, and let $w$ be their lowest common ancestor in $T$. The unique shortest $u-v$ path is $(u, v, w)$.

**Proof.** Let $\lambda(w) = i$. Clearly, any $u-v$ path in $T$ uses a node of height $i$ or more. By construction, if $(a, b) \in E_{t,k}$, then there is an $a-b$ path in $T$ which uses nodes whose height is in $[\lambda(a), \lambda(b)]$. It follows that any $u-v$ path in $G_{t,k}$ has to use a node of height $i$ or more, and hence also an edge of length $16^{-t}$.

If $u$ and $v$ are an ancestor-descendent pair in $T$, then $\lambda(u), \lambda(v) < i$. Therefore, any $u-v$ path contains an internal node of height $i$, and hence also at least two edges of length $16^{-i}$. Furthermore, $(u, w), (v, w) \in E_{t,k}$, and $l(u, w) = l(v, w) = 16^{-t} = (u, w, v)$ is a shortest $u-v$ path. Since edge lengths are positive, all shortest $u-v$ paths are of the form $(u, w', v)$, where $\lambda(w) = i$. By construction, $w$ is the only node of height $i$ which is adjacent to both $u$ and $v$, so $(u, w, v)$ is in fact unique.□

Now we define graph $G_{t,k,q} = (V_{t,k,q}, E_{t,k,q})$. We start with a copy of $G_{t,k}$, denoted by $G_{t,k,q} = (V_{t,k,q}, E_{t,k,q})$, $a = 1, 2, \ldots, q$. For an object (vertex, edge, set of vertices etc.) defined as follows. For all $u, v$ such that $u$ is an ancestor of $v$, there is an edge $(u, v)$ in $E_{t,k}$ whose length is $l(u, v) = 16^{-t}$, where $i = \lambda(u)$.

**Lemma 2.** Let $u^{(a)}, v^{(a)}$ be arbitrary nodes in $V_{t,k,q}$. Let $w$ be the lowest common ancestor of $u$ and $v$. The shortest $u^{(a)}-v^{(b)}$ paths are

- $(u^{(a)}, u^{(b)}), (v^{(a)}, v^{(b)})$, if $\lambda(u) \leq \lambda(v)$, and/or
- $(u^{(a)}, w^{(a)}), (v^{(a)}, w^{(b)})$, if $\lambda(w) \leq \lambda(u)$.

Note that there are two shortest paths if and only if $\lambda(u) = \lambda(v)$, and $a \neq b$ (Figure 1, bottom row).

**Proof.** Any $u^{(a)}-v^{(b)}$ path of length $l$ can be naturally mapped to a $u-v$ path in $G_{t,k}$ of length at most $l$ (by ignoring steps between different copies of $G_{t,k}$). By Lemma 1, the shortest $u-v$ path in $G_{t,k}$ is $p^* = (u, v, w)$. Since $p^*$ is unique, and edges of $G_{t,k}$ have length at least 1, it follows that any $u^{(a)}-v^{(b)}$ path that maps to $p^*$ has length at least $l(p^*) + 1$. On the other hand, the two $u^{(a)}-v^{(b)}$ paths given in the statement have length at most $l(p^*) + \varepsilon$. Therefore, any shortest $u^{(a)}-v^{(b)}$ path maps to $p^*$.

It is easy to see that the optimal path transitions from $G_{t,k}$ to $G_{t,k}$ or vice versa only once, since any extra transitions between copies of $G_{t,k}$ could be "shortcut". That transition should be done via shortest possible edge. The candidates are $(u^{(a)}, u^{(b)})$ and $(u^{(a)}, v^{(b)})$. This is because $w$ cannot have the smallest height among $u, v, w$, unless $w$
is equal to $u$ or $v$, in which case $(u^{(a)}, u^{(b)})$ coincides with $(u^{(a)}, u^{(b)})$ or $(u^{(a)}, v^{(b)})$. The statement follows from these observations.

For $i \geq 0$, define $S_i$ as the union of sets $\{u^{(j)} \mid j = 1, 2, \ldots, q\}$ over all nodes $v$ in $T$ such that $\lambda(v) = i$.

**Lemma 3.** For all $i \geq 1$, $S_i$ is a $(2^{4i-4}, q)$-shortest path cover, and also a $(2^{2i-3}, q)$-shortest path cover.

**Proof.** First we prove that any shortest path with length in $[2^{4i-4}, 2^{4i-2}] = [16^{-1}, 4 \cdot 16^{-1}]$ contains a node in $S_i$. Consider a shortest path between $u^{(a)}$ and $v^{(b)}$ whose length is in this interval. Let $w$ be the least common ancestor of $u$ and $v$, and let $\lambda(w) = j$. By Lemma 2, $d(u^{(a)}, v^{(b)}) \in [0, \varepsilon] \cup [16^{-1}, 16^{-1} + \varepsilon] \cup [2 \cdot 16^{-1}, 2 \cdot 16^{-1} + \varepsilon]$. Since $\varepsilon$ is very small, and $d(u^{(a)}, v^{(b)}) \in [16^{-1}, 4 \cdot 16^{-1}]$, it follows that $j = i$. Since $\lambda(w) = j$, we have $w^{(a)}, w^{(b)} \in S_i$. By Lemma 2, the path contains $w^{(a)}$ or $w^{(b)}$, so it contains a node in $S_i$.

Next we bound the number of nodes of $S_i$ in any open ball $B$ of radius $r \in (16^{-1} + \varepsilon, 16^{-1}]$. In particular, the interval includes $2^h = 16^i$. Suppose that $B$ is centered at some arbitrary $u^{(a)} \in V_{i,k,q}$. Let $v^{(b)}$ be some node in $S_i$. By definition of $S_i$, $\lambda(v) = i$, and $b$ can be arbitrary. Let $w$ be the lowest common ancestor of $u$ and $v$. If $v$ is not an ancestor of $u$, then $\lambda(w) \geq i+1$. By Lemma 2, $d(u^{(a)}, v^{(b)}) \geq 16^i$, so $v^{(b)} \notin B$. If $v$ is an ancestor of $u$, then $\lambda(w) = i$. By Lemma 2, $d(u^{(a)}, v^{(b)}) \leq 16^{-1} + \varepsilon$, so $v^{(b)} \in B$. Since there is at most one ancestor of $u$ with height $i$, and $q$ possible choices for $b$, we conclude that $|B \cap S_i| \leq q$.

**Lemma 4.** The $i$th level of $G_{i,k,q}$ is exactly $q$.

**Proof.** Let $w$ be the root of $T$, and let $4r = 16^{-1} + \varepsilon$. Clearly, $V_{i,k,q} \subseteq B(u^{(a)}, 4r)$ for $a = 1, 2, \ldots, q$. Let $u$ and $v$ be leaves of $T$ whose lowest common ancestor is $w$. We have that $d(u^{(a)}, v^{(b)}) = 2 \cdot 16^{h-k} - r$ for all $a = 1, 2, \ldots, q$. The corresponding shortest $u^{(a)}$-$v^{(b)}$ paths $u^{(a)}, v^{(b)}$ are vertex disjoint, so they cannot be covered by less than $q$ nodes. This implies that the highway dimension is at least $q$.

To upper-bound the highway dimension, consider an arbitrary ball $B = B(u^{(a)}, 4r)$, where $u^{(a)} \in V_{i,k,q}$. We will prove that all shortest path of length more than $r$ completely contained in $B$ can be covered by at most $q$ nodes in $B$.

Let $\lambda(u) = i$, and let $16^{-1} - 4r < 16^k$ for some $j \geq 1$. If $j \leq i-1$, then $4r < 16^{-1}$ so $B(u^{(a)}, 4r)$ contains only $u^{(a)}$, $b = 1, 2, \ldots, q$. Clearly, all shortest paths between these $q$ nodes can be covered by the nodes themselves.

Now assume that $j \geq i$. Let $v$ be the highest ancestor of $u$ such that $d(v, u) \leq 4r$ (in $T$, of course). Since $16^{-1} \leq 4r < 16^k$, $\lambda(v) = j$. Note that it is possible that $v = u$, which happens if $j = i$.

Let $x^{(a)} \in B(u^{a}, 4r)$. The goal is to show that $x$ is a descendant of $v$. If this is not the case, then the lowest common ancestor of $x$ and $v$ has height $j+1$ or more. Since $u$ is a descendant of $v$, the lowest common ancestor of $x$ and $u$ has height $j+1$ or more. By Lemma 2, $d(x^{(a)}, u^{(a)}) \geq 16^i > 4r$, which contradicts $x^{(a)} \in B$.

Consider an arbitrary shortest $x^{(a)}$-$y^{(b)}$ path of length more than $r$ and completely contained in $B$. We saw above that $v$ is an ancestor of both $x$ and $y$. We claim that $v$ is actually the lowest common ancestor of $x$ and $y$. If this is not the case, then the lowest common ancestor of $x$ and $y$ has height $j-1$ or less. By Lemma 2, $d(x^{(a)}, y^{(b)}) \leq 2 \cdot 16^{-1} - r < 16^i \leq 4r = r$, a contradiction.

Now, since $v$ is the lowest common ancestor of $x$ and $y$, it follows by Lemma 2 that $v^{(a)}$ or $v^{(b)}$ are on the shortest $x^{(a)}$-$y^{(b)}$ path. Hence, nodes $v^{(a)}, b = 1, 2, \ldots, q$ cover all required paths.

**Theorem 1.** For any $h, d, n$ there is a graph $G = (V, E)$ with highway dimension $h$, diameter $\Theta(D)$, and $|V| \geq n$, such that the preprocessing algorithm of Abraham et al. [2] produces a graph with $\Theta(|V|/\log D)$ edges.

**Proof.** We will prove that $G_{i,k,q}$ satisfies the requirements for a particular choice of $t, k, q$.

The diameter of $G_{i,k,q}$ is $2 \cdot 16^{-1} + \varepsilon$. Its number of nodes is $|V_{i,k,q}| = q|V_{i,k}| = q \sum_{i=0}^{k-1} \omega^{k-i} = \Theta(q^k)$, where we used the fact that $t \geq 2$.

We bound the number of shortcuts that are created in any ordering where $S_i$ is contracted before $S_j$ for $j > i$. Consider path $(u^{(b)}, v^{(a)}, u^{(a)}, v^{(b)})$, where $w$ is the lowest common ancestor of $u$ and $v$ (cf. Lemma 2). If $u = v$, then $u^{(b)} = v^{(a)} = u^{(a)}$, so we have a zero-or-one-hop path, which can be shortcut. From now on we assume that $u \neq v$. If $v$ is not an ancestor of $u$, then $\lambda(u) > \lambda(v)$. Thus $w^{(a)}$ is contracted after $v^{(a)}$, and the path is not shortcut. From now we assume that $v$ is an ancestor of $u$, i.e., that $w = v$. If $a = b$, then we have a one-hop path $(u^{(b)}, v^{(a)})$, which cannot be shortcut. Finally, if $a \neq b$, then the path is shortcut if and only if $u^{(a)}$ is contracted before $v^{(b)}$.

We summarize the above discussion: $p_1 = (u^{(b)}, u^{(a)}, u^{(a)}, v^{(a)})$ is shortcut if and only if $a \neq b, v$ is a proper ancestor of $u$, and $u^{(a)}$ is contracted before $v^{(b)}$. Exchanging the roles of $a$ and $b$, we get that $p_2 = (u^{(a)}, u^{(b)}, v^{(a)}, v^{(b)})$ is shortcut if and only if $a \neq b, v$ is a proper ancestor of $u$ and $u^{(b)}$ is contracted before $v^{(a)}$. Note that $p_1$ is a shortcut path if and only if $p_2$ is, because $G_{i,k,q}$ is symmetric with respect to re-indexing copies of $G_{i,k}$. We conclude that either $p_1$ or $p_2$, but not both, are shortcut.

The above analysis holds for every node $u$, its proper ancestor $v$, and every pair $a, b$ such that $a \neq b$. Thus the total number of shortcuts is

$$\sum_{i=0}^{k-1} \frac{|V_{i,k,q}|}{|V_{i,k}|} \cdot (k-i) \cdot \frac{\omega^{k-i} - 1}{2} \cdot \frac{\omega^{i-1}}{2}$$

split by height

$$\quad \frac{\omega^{i-1}}{2}$$

with height $i$ proper ancestors $v$ of $u$ pairs $a, b$ of $a \neq b$ one of two paths shortcut

$$= \Theta(q^k t^2) \cdot \Theta(qk|V_{i,k,q}|).$$

Set $k = \lfloor \log_D q \rfloor$; with this, the diameter is $\Theta(D)$. Set $q = h$. Choose $t$ large enough so that $|V_{i,k,q}| = \Theta(h^t)$ exceeds $n$. By Lemma 4, the highway dimension of $G_{i,k,q}$ is exactly $h$. By Lemma 3, $S_i$ is a $(2^{3h-3}, h)$- and $(2^{h-2}, h)$-shortest path cover. Finally, the number of shortcuts created by contracting $S_i$ before $S_j$ for $j < i$, is $\Theta(h|V_{i,k,q}|)$ as $D$, which is required. □
5. PLANAR GRAPHS

In this section we present a simple argument proving that a contraction hierarchy built from a planar graph $G=(V,E)$ has $O(n \log n)$ edges, where $n=|V|$. Let $H$ be an undirected graph obtained by ignoring edge directions in $G$ (and eliminating parallel edges). Consider the following modified preprocessing algorithm running on $H$. Fix a node ordering, and contract nodes one by one in this ordering, this time ignoring the shortest path condition. In other words, when contracting node $v$, create a new undirected edge between any two uncontracted neighbors of $v$ that are not already adjacent. Alternatively, the modified preprocessing creates an edge between any two nodes connected by a (not necessarily shortest) path whose all internal nodes are contracted before any of the endpoints. Based on this alternative view, it is not hard to see that the number of edges in the contraction hierarchy built from $G$ using the original preprocessing algorithm is at most twice the number of edges in $H$ after the modified preprocessing. Lipton, Rose, and Tarjan [7] proved that the latter is bounded by $O(n \log n)$, and that the associated contraction ordering can be computed in $O(n \log n)$ time.

6. HARDNESS

We prove that MinCH is APX-hard using a reduction which is considerably simpler than the one in [3]. We present an approximation-preserving reduction from the feedback arc set problem (FAS) to MinCH. FAS is the problem of deciding, for a given directed graph $G$ and integer $q$, if there exist at most $q$ edges in $G$ such that every directed cycle in $G$ contains at least one of these edges.

Let $G=(V,E)$ be an arbitrary undirected weighted directed graph. We construct an instance of MinCH, which is a weighted directed graph $H$, as follows. We start with the original graph, building undirected edges of length 1. For each $v \in V$, add one new node $w_v$, and a directed edge $(v,w_v)$ of length 1. Let $W = \{w_v \mid v \in V\}$. For any $v \in V$ and $w \in W$ such that the distance from $v$ to $w$ is 3 or more, add a shortcut from $v$ to $w$ of length 3.

Consider a contraction order in which all nodes in $V$ are contracted before any of the nodes in $W$. Note that no nontrivial (i.e., with more than one edge) shortest path uses a shortcut of length 3 that we added from $V$ to $W$. Hence these shortcuts have no effect on the shortcuts created while contracting $V$. The latter are in one-to-one correspondence with the shortest paths of length 2 from $V$ to $W$ whose second node is contracted before the first. In other words, they are in one-to-one correspondence with the edges $(u,v) \in E$ such that $v$ is contracted before $u$. Therefore, minimizing the number of shortcuts from $V$ to $W$ (over all contraction orderings of this kind) is equivalent to computing the minimum feedback arc set in $G$.

Of course, the goal is not to minimize the number of shortcuts from $V$ to $W$, but the number of all shortcuts. To handle this, let $k$ be a large enough integer to be determined later. Instead of adding only one node $w$ for each $v \in V$, we add $k$ nodes, and add a directed edge from $v$ to each one of them. As before, let $W$ be the set of nodes thus added. The rest of the construction is the same.

In the modified instance, the minimum number of shortcuts is exactly $k$ times the cardinality of the minimum feedback arc set of $G$. For $k$ large enough (compared to $|V|^2$, say), the number of shortcuts created within $V$ becomes insignificant.

We need to remove one more restriction, the one that all nodes of $V$ are contracted before any nodes of $W$. To that end, we add a new node $x$, and a set of 3-hop paths connecting nodes in $W$ to $x$. For each $w \in W$, there are $2l$ 3-hop paths starting at $w$ and ending at $x$. The 2W/|l| paths do not share any internal vertices. The costs of the 2W/|l| edges incident to $W$ are 3. The costs of the other 4W/|l| edges are 1. We denote by $Z$ the set of newly added nodes, i.e., $Z$ consists of $x$ and the internal nodes off all paths.

Now we prove that this last modification does its job.

**Lemma 5.** If some vertex in $W$ is contracted before some vertex in $V$, then at least $l$ shortcuts are created.

**Proof.** Assume that some vertex in $W$ is contracted while there are still uncontracted vertices in $V$. Let $v$ be such a vertex that is contracted first.

Consider the time when $x$ is contracted, and restrict attention only to the paths between $x$ and $w$ (i.e., disregard the paths between $x$ and other vertices in $W$). Let $z_1$ and $z_2$ be two uncontracted neighbors of $x$, if they exist. The path $(z_1, x, z_2)$ costs 2, while any other path costs at least 6, so the shortcut is created. Hence if at the time when $x$ is contracted it has at least $l+1$ uncontracted neighbors, then at least $(l+1)/2 \geq l$ shortcuts are created, and we are done. From now on we assume that this is not the case.

Consider the time when $w$ is contracted. Note that all vertices in $V$ are within 2 hops from $w$ in the original graph. Let $v$ be an uncontracted vertex in $V$ that is closest to $w$ by hop count. That is, if there is a neighbor of $w$ in $V$ that is uncontracted, make $v$ that neighbor, otherwise make $v$ any uncontracted node in $V$. Let $z$ be an uncontracted neighbor of $w$ in $Z$, if one exists.

Note that in the original graph there is either a path $(v,w,z)$ or a path $(v,z',w,z)$, where $z'$ is a contracted node in $V$. Recall that $v$ and $z$ are uncontracted, and $w$ is about to be contracted. Also, the length of these paths is at most $3 + 3 = 6$. The length of any path from $v$ to $z$ that does not use $w$ is at least $1 + 1 + 1 + 3 + 2 = 9$, hence a shortcut is created. Hence if at the time when $w$ is contracted it has at least $l$ uncontracted neighbors in $Z$, we are done. From now on we assume that this is not the case.

Consider the time when $x$ or $w$ is contracted, whichever is earlier. By assumption, at least $l$ paths between $x$ and $W$ have an internal vertex which is already contracted. Fix one such path, and consider the first vertex on that path that was contracted. Both of its neighbors were uncontracted at the time. Furthermore, the path through the contracted node costs at most 4, while any alternative path costs at least 6. Hence exactly one shortcut is created, or at least $l$ for all paths.

Therefore, if $l$ is large enough compared to $|E| + |V|^2$, no node from $W$ will be contracted before any node of $V$ in any optimal contraction order. After contraction of $V$, we claim that one can contract $W$ without adding new shortcuts. The only possible shortcuts are those that connect vertices in $Z$. For any potential shortcut, a path through $W$ costs at least 6, but there is a path through $Z$ which costs 4, so no shortcut is created. Then we are left with a tree, which can clearly be contracted without adding new shortcuts by always contracting a leaf.

The following lemma summarizes the previous discussion.
Lemma 6.

\[ k \text{OPT}_{\text{FAS}}(G) \leq \text{OPT}_{\text{MinCH}}(H) \leq k \text{OPT}_{\text{FAS}}(G) + |V|^2. \]

The following is the main result of this section.

**Theorem 2.** If there is an \( \alpha \)-approximation algorithm for MinCH (\( \alpha > 1 \)), then there is an \( \alpha \)-approximation algorithm for FAS.

**Proof.** Let \( A \) be an \( \alpha \)-approximation algorithm for MinCH. Then an \( \alpha \)-approximation algorithm for FAS is as follows. First test if the input graph \( G \) is acyclic. If so, return 0 and terminate. Otherwise, generate \( H \) as described above, with \( k = \alpha |V|^2 \) large enough (but polynomial in the size of \( G \)). Invoke \( A \) on it, obtaining an \( \alpha \)-approximate minimal number of shortcuts \( s \). Return \( \lfloor s/k \rfloor \). By Lemma 6

\[ \lfloor s/k \rfloor \leq \frac{s}{k} \leq \frac{\alpha \text{OPT}_{\text{MinCH}}(H)}{k} \leq \alpha \text{OPT}_{\text{FAS}}(G) + \frac{\alpha |V|^2}{k} \leq \alpha \text{OPT}_{\text{FAS}}(G) + 1. \]

\( \square \)

7. **CONCLUSION AND FUTURE WORK**

In our view, the main result of this paper is the fact that planar graphs can be preprocessed into small contraction hierarchies despite having large highway dimension. On the other hand, according to [2], contraction hierarchies do not perform well on some planar graphs. A possible explanation could be that a typical search space size (i.e., the number of nodes visited by the query algorithm) is large. This aspect deserves to be investigated further.

It would also be interesting to see if the approach used to derive the upper bound for planar graphs can be ported to graphs that are more similar to realistic road networks. A potential candidate are multiscale-dispersed graphs proposed in [5] as a model for road networks.

Finally, we note that our preprocessing algorithm for planar graphs is completely oblivious to the shortest path structure. Thus it is likely outperformed in practice by existing fine-tuned heuristics for computing node ranks. This is confirmed by our preliminary experimental comparisons with greedy strategies, i.e., repeatedly picking one or more contractions that minimize the increase in the edge count.

8. **REFERENCES**


