Insights into (k, ρ)-Shortcutting Algorithms

Alexander Leonhardt
Goethe University Frankfurt, Germany

Ulrich Meyer
Goethe University Frankfurt, Germany

Manuel Penschuck
Goethe University Frankfurt, Germany

Abstract

A graph is called a (k, ρ)-graph iff every node can reach ρ of its nearest neighbors in at most k hops. This property has proven useful in the analysis and design of parallel shortest-path algorithms [7, 13]. Any graph can be transformed into a (k, ρ)-graph by adding shortcuts. Formally, the (k, ρ)-Minimum-Shortcut-Problem ($k\rho$-MSP) asks to find an appropriate shortcut set of minimal cardinality.

We show that $k\rho$-MSP is $\mathcal{NP}$-complete in the practical regime of $k \geq 3$ and $\rho = \Theta(n^{\varepsilon})$ for $\varepsilon > 0$. With a related construction, we bound the approximation factor of known $k\rho$-MSP heuristics [7] from below and propose algorithmic countermeasures improving the approximation quality. Further, we describe an integer linear problem (ILP) that optimally solves $k\rho$-MSP. Finally, we compare the practical performance and quality of all algorithms empirically.

2012 ACM Subject Classification Mathematics of computing → Graph algorithms

Keywords and phrases Complexity, Approximation, Optimal algorithms, Parallel shortest path

Digital Object Identifier 10.4230/LIPIcs.ESA.2024.84

Related Version *The full version of this article is available on arXiv under the same name*: `https://arxiv.org/abs/2402.07771` [22]

Supplementary Material *Software*: `https://github.com/alleonhardt/k-rho-shortcutting` [21] archived at `swh:1:dir:8965d090c1d32ea024b1bb4b111329990a156b37`

Funding *Manuel Penschuck*: Funded by the Deutsche Forschungsgemeinschaft (DFG) – ME 2088/5-2 (FOR 2975 – Algorithms, Dynamics, and Information Flow in Networks).

Acknowledgements The authors thank the anonymous reviewers for their insightful comments which greatly improved this paper.

1 Introduction

Shortest path algorithms trace back to the very roots of computer science and are an integral part of the basic algorithmic toolbox. Consequently, a large body of literature has been devoted to shortest-path algorithms, dating back to at least the 1950s with the classical single-source shortest-path (SSSP) algorithms by Dijkstra [12], Bellman and Ford [5, 18].

Despite this immense attention, shortest-path algorithms remain notoriously hard to parallelize efficiently; one of the first parallel SSSP (PSSSP) algorithms is based on the Bellman-Ford algorithm. More efficient solutions often follow the *stepping framework* including Δ-stepping [24], radius-stepping [7] and ρ-stepping [13]. Although they outperform Bellman-Ford in practice, their theoretical guarantees on general graphs rarely reflect this.

Recently, Blelloch et al. [7] introduced the notion of (k, ρ)-graphs (Definition 3) which, roughly speaking, asserts that every node can reach its ρ nearest neighbors via a shortest-path with at most k edges. This notion provides just enough structure to derive new theoretical

bounds for stepping based algorithms [7, 13]. Crucially, any graph can be converted into a (k, ρ)-graph by introducing shortcuts. We refer to the problem of finding such a shortcut set of minimal cardinality as $k\rho$-MSP (Definition 5).

To the best of our knowledge, only two articles have considered (k, ρ)-graphs previously. Blelloch et al. [7] introduce the notion, propose two $k\rho$-MSP heuristics, and empirically study the number of edges added by them. Further, Dong et al. [13] rely on (k, ρ)-graphs to provide faster algorithms and new bounds without further investigating the notion itself.

Our contribution

We prove for the first time that $k\rho$-MSP is $\mathcal{NP}$-complete for *all* constant $k \geq 3$ and some $\rho = \Theta(n^{\varepsilon})$ in Section 3. As hub labeling, a related concept, is already known to be $\mathcal{NP}$-complete [9], we extend this result to $\rho < n - 1$ for an arbitrary constant k. This finding holds significant practical relevance as PSSSP algorithms for (k, ρ)-graphs crucially depend on small k and large ρ for their span bounds. As a by-product, we obtain non-trivial lower bounds on the approximation ratio of the best known $k\rho$-MSP heuristic in Section 4.

Based on the insights derived from the lower bounds, we propose several new preprocessing steps in Section 5 to improve the heuristic's approximation factor on several random graph models. In Section 6, we give an integer linear program (ILP) to solve $k\rho$-MSP optimally. Finally, we conduct an experimental evaluation that provides evidence suggesting a commonly found property in social network graphs is responsible for frequently inducing large approximation factors in existing heuristics. Our newly proposed heuristics are able to partially mitigate this issue and a final assessment on real world graphs shows that our techniques generalize well to a variety of graphs.

Related work

Although (k, ρ)-graphs themselves have not been studied extensively, a related concept of hub labeling (HL), initially introduced by Cohen et al. [9], has received significant attention. HL assigns every vertex $v \in V$ in a directed graph G a forward label $L_f(v)$ and a backward label $L_b(v)$. Frequently, the labels consist of a sequence of tuples, each of which hold a vertex and their respective distance from v, i.e., $(w, d(w, v)) \in L_b(v)$ and $(w, d(v, w)) \in L_f(v)$, where $d(u, v)$ is the weight of the shortest $u - v$-path. The vertices within $L_f(v) \cup L_b(v)$ are called hubs of v and are required to cover G in the sense that for any pair of vertices $x, y \in V$, $\exists v \in V : (v, d(x, v)) \in L_f(x) \wedge (v, d(v, y)) \in L_b(y)$ such that v is on a shortest $x - y$-path. It is straightforward to see that such a labeling can be used to build a shortcut set that transforms any graph into a $(2, n - 1)$-graph. A wide variety of work investigates HL and related works, both in theory and in practice [3, 2, 11, 17]. Although related, most techniques do not readily adapt to arbitrary choices of ρ and k.

2 Preliminaries

Let $G = (V, E)$ be an undirected graph with vertex set V and edge set E. A *weighted* graph has an additional weight function $W \colon E \to \mathbb{R}_{\geq 0}$, assigning each edge e the weight $w(e)$. In an unweighted graph, all edges have equal weight 1. We denote the *neighbors* of $u \in V$ as $N(u) = \{v \mid \{u, v\} \in E\}$ and let $N^+(u)$ refer to the union of $N(u)$ and $\{u\}$. Further, let $G[V']$ be the subgraph of G induced by the vertices V'.

For two nodes $u, v \in V$, let $d(u, v)$ be the *total weight of the shortest path* (by weight) connecting u to v. We also refer to $d(u, v)$ as *distance* from u to v. Further, let $\hat{d}(u, v)$ be the *hop distance* between u and v, which we define as the smallest number of edges among

all paths between u and v with a weight of $d(u,v)$. If no path exists between u and v, set $d(u,v) = \hat{d}(u,v) = \infty$. For the sake of brevity we denote a path with weight $d(u,v)$ as *shortest path*. When referring to a *shortest path with fewest hops* we denote a path with weight $d(u,v)$ and $\hat{d}(u,v)$ edges.

We first restate the two central properties, *k-radius* and *ρ-distance* according to [7]:

▶ **Definition 1.** *For $u \in V$, the* k-radius $\bar{r}_k(u)$ of u *is the distance to the closest node reachable from u, which is more than k hops away, i.e., $\bar{r}_k(u) = \min_{v \in V\colon\ \hat{d}(u,v) > k} d(u,v)$.*

▶ **Definition 2.** *For $u \in V$, the* ρ-nearest distance of u, *denoted by $r_\rho(u)$, is the distance from u to the ρ-th closest vertex to u.*

These allow us to define (k,ρ)-graphs, where each node can reach any of its ρ nearest neighbors in at most k hops. Observe that we deviate from [7] (which originally requires $r_\rho(v) \leq \bar{r}_k(v)$ for (k,ρ)-balls) to avoid ambiguity if multiple nodes lie at hop distance $k+1$.

▶ **Definition 3.** *Vertex $v \in V$ has a* (k,ρ)-ball *iff $r_\rho(v) < \bar{r}_k(v)$;*[1] *i.e., the ρ-closest node of v can be reached within k hops. Further, G is a (k,ρ)-graph iff every node has a (k,ρ)-ball.*

Arbitrary graphs can be transformed into (k,ρ)-graphs by adding shortcuts, i.e., edges that lower the hop distance $\hat{d}$ between nodes without changing their shortest path distance d:

▶ **Definition 4.** *For $G = (V,E)$, let $u, v \in V$ be different nodes with distance $D = d(u,v) < \infty$ and $\hat{d}(u,v) > 1$. A shortcut between u and v is a new edge $\{u,v\}$ with weight D.*

▶ **Definition 5.** *Given a weighted graph $G = (V,E)$, the* (k,ρ)-Minimum-Shortcut Problem *($k\rho$-MSP) asks for a minimum cardinality set S, s.t. $G' = (V, E \cup S)$ is a (k,ρ)-graph. Further let $k\rho\ell$-*MSP*, the decision variant of $k\rho$-MSP, ask whether $|S| \leq \ell$ exists.*

If clear from context, we use $k\rho$-MSP as a shorthand for its decision variant $k\rho\ell$-MSP. Finally, let $A \leq_p B$ denote a polynomial time reduction from problem A to B.

3 Complexity of $k\rho$-MSP

The Minimum-Shortcut Problem ($k\rho$-MSP) can be solved in polynomial time for $k = 1$, since each shortcut only affects its two endpoints. Hence, each node has to be connected to ρ of its nearest neighbors. In the following, we show that $k\rho$-MSP is $\mathcal{NP}$-complete for $k > 2$ and practical $\rho = \gamma$. Here γ is a parameter which refers to the number of leaf nodes needed by the pitchfork gadget (for an overview of the construction refer to Figure 1). Since it is trivial to verify a $k\rho$-MSP solution in polynomial time, $k\rho$-MSP is in $\mathcal{NP}$. For $\rho = n-1$ and some k, the $\mathcal{NP}$-completeness can be inferred by the work of Babenko et al. [3] and Cohen et al. [9]. Henceforth, we show that even when relaxing conditions on ρ such that $\rho < n-1$ the problem is $\mathcal{NP}$-complete. We focus on its hardness by establishing that the $\mathcal{NP}$-hard [19] Vertex Cover problem is polynomial-time reducible to $k\rho$-MSP.

Recall that, given an undirected graph $G = (V,E)$, Vertex Cover asks to find a minimal set of nodes $C \subseteq V$ for which at least one endpoint of every edge is in C. Since G is undirected and the transformation in Section 3.1 uses only unit edge weights, the arguably simplest graph model suffices to proof the hardness of $k\rho$-MSP.

[1] This definition requires that the distances between nodes within the graph are unique (excluding pairs of nodes (u,v) where u cannot reach v), if this is not given a consistent tiebreaker may reconcile the definition on these graphs.

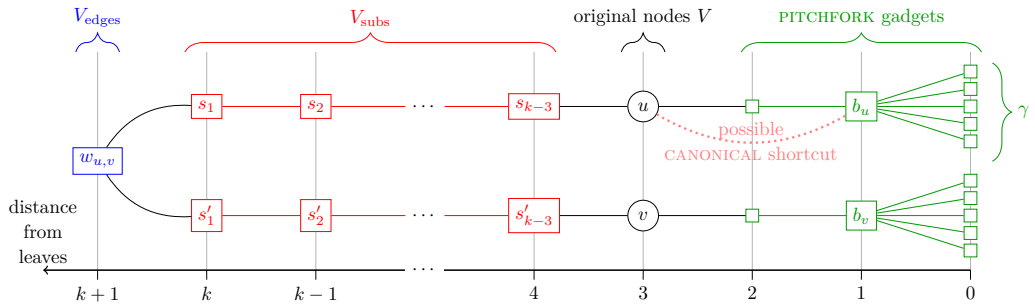


Figure 1 Transformation of edge $\{u, v\}$ for the input $G = (V, E)$. Each edge in E implies their own nodes in V_{edges} and V_{subs} (left); each original node in V has its own PITCHFORK-gadget (right). By construction $w_{u,v}$, is too far from the leaves of either PITCHFORK. By adding, e.g., the CANONICAL shortcut (see Definition 8) between u and b_u, the leaves become available for the (k, ρ)-ball of $w_{u,v}$.

▶ **Theorem 6.** *Let $k \geq 3$ be an integer constant, and $\rho = \Theta(n^{\varepsilon})$ for some constant $\varepsilon > 0$. Then, VERTEX COVER $\leq_p$ $k\rho$-MSP implying $k\rho$-MSP is $\mathcal{NP}$-hard.*

Proof. We transform a VERTEX COVER input $G = (V, E)$ into a $k\rho$-MSP graph G_T in Section 3.1. In Section 3.2, we show that the structure of G_T implies that there exist optimal so-called *canonical* $k\rho$-MSP solutions (see Corollary 11). Then we establish with Lemma 12 a bijection between shortcuts in canonical $k\rho$-MSP solutions to nodes in VERTEX COVER. Thus, any optimal solution for $k\rho$-MSP on G_T implies an optimal solution to VERTEX COVER. ◀

For simplicity, the proof only considers $\gamma \geq 7|V| + 6$. We refer to the full version of this paper [22], for the acclaimed range of γ. The extended proof is structurally similar to the present one but employs a recursive application of the PITCHFORK-gadgets described in Section 3.1.

3.1 Transforming Vertex Cover to $k\rho$-MSP

Let $G = (V, E)$ be an input graph for VERTEX COVER. For a fixed value for parameter $k \in [3, n)$, we transform G into G_T in two steps:

- We first obtain $G_P = (V_P, E_P)$ by replacing every edge $\{u, v\} \in E$ with its own copy of the following path template (i.e. the subgraph induced by the following nodes is a path):

$$(u, \underbrace{s_{k-3}, \ldots, s_1}_{k-3 \text{ subdivision}}, \underbrace{w_{u,v}}_{\text{named representative of edge}}, \underbrace{s'_1, \ldots, s'_{k-3}}_{k-3 \text{ subdivision}}, v)$$

  For $k = 3$, the template degenerates into $(u, w_{u,v}, v)$. As visualized in Figure 1, we conceptually partition the vertex set V_P into (i) the original nodes V, (ii) the edge nodes $V_{\text{edges}} = \{w_{u,v} \mid \{u, v\} \in E\}$, and (iii) the subdivision nodes V_{subs} with $|V_{\text{subs}}| = 2|E|(k-3)$. Observe that this transformation retains the degrees of the original nodes V and that all new nodes $V_P \setminus V$ have degree 2.
- We obtain the final graph $G_T = (V_T, E_T)$ from G_P by adding and connecting a copy of a so-called γ-PITCHFORK to each original node $v \in V$. A γ-PITCHFORK to the host node $v \in V$ is a $(\gamma+1)$-star graph where exactly one satellite has an additional edge to v. We denote the star's center of the gadget attached to node $v \in V$ as the *base node* b_v.

The γ-PITCHFORKs encode VERTEX COVER into the $k\rho$-MSP problem. The main idea is that all nodes but the ones in V_{edges} have a (k,ρ)-ball. The latter are too distant to the leaves of their respective γ-PITCHFORKs. For any edge node $w_{u,v} \in V_{\text{edges}}$, however, it suffices that the distance to the γ-PITCHFORK of either u or v is reduced with a single shortcut. This roughly corresponds to having at least one endpoint of each edge in a vertex cover.

3.2 Solving Vertex Cover

Let $G = (V, E)$ be the input graph for VERTEX COVER, $G_T = (V_T, E_T)$ the graph obtained from the transformation in Section 3.1, and $G_{\text{MSP}} = (V_T, E_T \cup S)$ where S is a minimal cardinality shortcut set to ensure G_{MSP} is a (k,ρ) graph. We begin by formally establishing the observation that exactly the nodes V_{edges} have no (k,ρ) ball in G_T:

▶ **Lemma 7.** *Fix $\gamma \geq 7|V| + 6$ and let $\rho = \gamma$. Then, $v \in V_T$ has a (k,ρ)-ball, iff $v \notin V_{edges}$.*

Proof. Observe that by construction of G_T and choice of γ, (i) the leaves of a single γ-pitchfork suffice to form a (k,ρ)-ball and (ii) any (k,ρ)-ball has to include some leaves of at least one γ-PITCHFORK. As illustrated in Figure 1, all nodes except V_{edges} have a hop distance of at most k to the leaves of their respective closest PITCHFORK. Contrary, any edge node $w_{u,v} \in V_{\text{edges}}$ requires $k+1$ hops to the leaves of the closest PITCHFORKs which are connected to nodes u and v, respectively. ◀

For the remainder of this section, we assume that $\rho = \gamma$ and $\gamma \geq 7|V| + 6$. Then let S be a minimal cardinality $k\rho$-MSP solution. We now categorize the shortcuts into CANONICAL and COMPLEX types, and show that any COMPLEX shortcut can be canonicalized:

▶ **Definition 8.** *We call a shortcut CANONICAL iff it connects a node in V to the base of its corresponding γ-PITCHFORK (see Figure 1). All other shortcuts are called COMPLEX.*

▶ **Observation 9.** *A CANONICAL shortcut at node $u \in V$ allows all nodes $w_{u,v} \in V_{edges}$ with $\{u,v\} \in E$ to form a (k,ρ)-ball.*

▶ **Lemma 10.** *Let S be an optimal solution for $k\rho$-MSP on G_T and $s_c \in S$ be an arbitrary COMPLEX shortcut. Then, the removal of s_c destroys exactly one (k,ρ) ball.*

Proof sketch. We defer the rigorous proof to the full version of this paper [22] and henceforth only give some intuition for the ideas of the proof. Roughly speaking, it establishes two central properties of COMPLEX shortcuts in a minimal shortcut set:

- No two COMPLEX shortcuts interact in a meaningful way, i.e. there is no shortcut s_1 that relies on a shortcut s_2 to close a (k,ρ) ball.
- The distance between any two edge nodes in V_{edges} is sufficiently large, such that no COMPLEX shortcut can affect more than one edge node $v \in V_{\text{edges}}$. ◀

▶ **Corollary 11.** *For any transformation G_T, there exists an optimal $k\rho$-MSP-solution S containing only CANONICAL shortcuts. We call such an S* canonical.

Proof. Let $s \in S$ be an arbitrary COMPLEX shortcut; if non exists, the claim follows. The removal of s destroys the (k,ρ)-ball of a unique $w_{u,v} \in V_{\text{edges}}$ due to Lemma 10. Thus we can replace s with the CANONICAL shortcut $s' = \{u, b_u\}$ (or – equivalently – $s' = \{v, b_v\}$). Observe that $S' = S \setminus \{s\} \cup \{s'\}$ has the same size $|S| = |S'|$, still turns G_T into a (k,ρ)-graph, but has one fewer COMPLEX shortcut. Finally recurse until S' is canonical. ◀

▶ **Lemma 12.** *An optimal solution set for $k\rho$-MSP on G_T can be transformed into a solution for* VERTEX COVER *on G.*

Proof. Let C be an optimal VERTEX COVER solution on G. Then, by Observation 9, we know that $S_c = \{\{v, b_v\} \mid v \in C\}$ consists only of CANONICAL shortcuts and turns G_T into a (k, ρ)-graph; i.e. any optimal $k\rho$-MSP solution S_c on G_T satisfies $|S_c| \leq |C|$.

Let S be an optimal solution of $k\rho$-MSP on G_T. Due to Corollary 11, assume without loss of generality that S is canonical. Then, $C = \{u \mid \{u, b_u\} \in S\}$ is a VERTEX COVER (see Observation 9). Since $|S| \leq |C|$, C is minimal. ◀

4 A lower bound on the approximation ratio of $k\rho$-DP

In Section 3, we show that computing a minimal shortcut set S to convert an arbitrary graph into a (k, ρ)-graph can be expensive. This has direct implications for preprocessing steps of algorithms based on (k, ρ)-graphs, which tend to favor large values of ρ. For instance, the depth[2] of RADIUS-STEPPING on a (k, ρ)-graph with small constant k is bounded by $\mathcal{O}(\frac{\log \rho}{\rho} \cdot n \log(n) L)$ where L is the maximal edge weight. [7]

Observe that any graph can be transformed into a (k, ρ)-graph by adding ρ shortcuts per node, i.e. $|S| \leq n\rho$ is trivial. Blelloch et al. [7] introduce two heuristics for smaller S: the greedy $k\rho$-GREEDY and $k\rho$-DP involving dynamic programming. The authors show that $k\rho$-GREEDY with $\rho = n - 1$, $k = 2$ suffers from an approximation ratio of at least $n - 5$. They demonstrate that, in practice, $k\rho$-DP yields smaller solutions than $k\rho$-GREEDY.

Given a graph $G = (V, E)$, the $k\rho$-DP heuristic [7] uses a dynamic program to compute the minimal number of shortcuts needed to form a (k, ρ)-ball around a node $s \in V$. While the solution is optimal for this node conditioned on a shortest path tree, $k\rho$-DP may not find a global minimal $|S|$, since the algorithm treats all (k, ρ)-balls independently (and in parallel). The conditioning is necessary since there might be several unique shortest path trees with fewest hops each requiring a different number of shortcuts to form a (k, ρ)-ball for the root vertex (refer to the full version for an example [22]).

For a fixed start node $s \in V$ and let T_s be a shortest path tree to the ρ-nearest neighbors of s, such that for all $v \in T_s$ the path between s and v in T_s has the least hops possible in G. $k\rho$-GREEDY works by greedily adding shortcuts along T_s, i.e. it adds shortcuts from s to all nodes in the i'th layer of T_s whenever $i \mod k \equiv 0$.

$k\rho$-DP on the other hand improves on that by determining the optimal placement of local shortcuts via dynamic programming: Let $N(u)$ and $N^+(u)$ denote the neighbors of u and the neighbors of u including u in T_s. If the height of T_s is at most k, s already has an (k, ρ)-ball; otherwise, we need to add shortcuts. Consider some node $u \in T_s$ with $u \neq s$, let p be its parent, and denote p's depth as $t = \hat{d}(s, p)$. Then, define $F(u, t)$ as the smallest number of shortcuts into the subtree (of T_s) rooted in u required to put u and its children within the (k, ρ)-ball of s:

$$F(u,t) = \min\left(\overbrace{1 + \sum_{w \in N^+(u)} F(w,1)}^{\substack{\text{add shortcut } \{s,u\}: \\ \text{depth of } u \text{ is now } 1}}, \overbrace{\sum_{w \in N^+(u)} F(w,t+1)}^{\substack{\text{no shortcut added to } u \\ \text{depth of } u \text{ remains } t+1}}\right) \quad \text{if } t < k \text{ else} \quad F(u,t) = \infty$$

Summing over the children of s, $\sum_{u \in N(s)} F(u, 0)$, yields the number of shortcuts on s. To the best of our knowledge, we now bound the solution quality of $k\rho$-DP for the first time:

[2] The depth of a parallel algorithm is the length of its critical path, bounding the runtime from below – even for an unbounded number of processors.

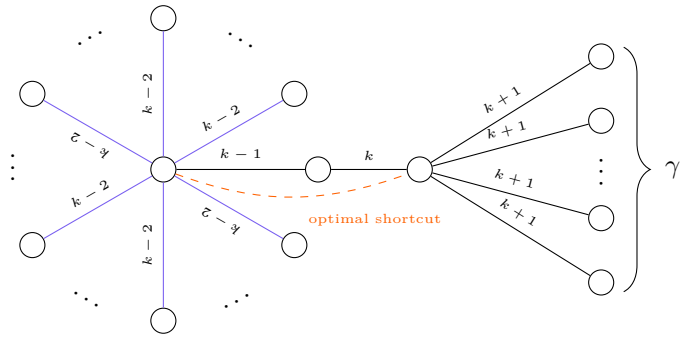


Figure 2 The perturbation of a star graph which induces the approximation factor mentioned in Theorem 13. Magenta edges depict an edge that is subdivided $k-3$ times, hence for $k=3$ it is a normal edge while for $k>3$ it is replaced by a path of length $k-3$ connecting the satellite nodes to the center. The label on top of each edge equals the hop distance of a satellite node after traversing the accompanied edge(s).

▶ **Theorem 13.** *The approximation ratio of $k\rho$-DP on a graph with $\Theta(n)$ nodes is at least $n-1$ and the approximation ratio of $k\rho$-Greedy is $\Omega(n^2)$ for any constant $k \geq 3$ and some parameter ρ.*

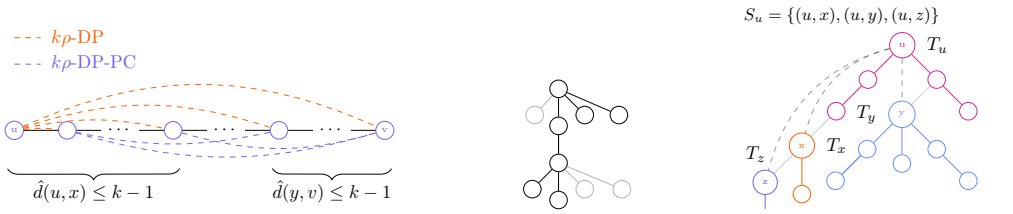
Proof. Consider a star graph S_n where each edge is replaced by a path of length $k-3$ as previously described in the reduction and add a single γ-pitchfork at the center node as depicted in Figure 2. In this graph only the satellite nodes do not form (k,ρ)-balls for some k and some ρ. Clearly, by Observation 9, a single shortcut at the only γ-pitchfork is enough to transform this graph into a (k,ρ)-graph. $k\rho$-DP will add $n-1$ shortcuts as the only shortcuts considered start at the nodes themselves, thus preventing it to shortcut the γ-pitchfork gadget as any node within, by definition, already forms a (k,ρ)-ball. Hence, the aforementioned construction yields a graph of order $n+(n-1)\cdot(k-3)+2+\gamma=\Theta(n)$ which can be transformed into a (k,ρ)-graph by a single shortcut while $k\rho$-DP adds $n-1$. On the same construction the greedy heuristic has an approximation factor of $\Omega(n^2)$, every satellite node will add $\gamma-(k-2+2\cdot n)=\Theta(n)$ shortcuts thus resulting in $\Omega(n^2)$ added shortcuts. ◀

5 The $k\rho$-DP-* family of heuristics

Based on the insights leading to the lower bounds of $k\rho$-DP, we introduce two simple preprocessing steps for $k\rho$-DP, namely $k\rho$-DP-PC (**p**air-short**c**utting) and $k\rho$-DP-SA (**s**et **a**lignment). These new heuristics address two separate issues introduced by a central property of $k\rho$-DP: All proposed shortcuts start at the source vertex on which $k\rho$-DP was invoked.

This property rules out any shortcuts starting from nodes which already form (k,ρ)-balls even if they are globally beneficial, such as canonical shortcuts within a γ-pitchfork gadget. We address this issue with $k\rho$-DP-PC. The new heuristic considers a larger set of shortcut candidates and introduces a global pooling phase to prune unpromising ones.

Additionaly we consider another variant, $k\rho$-DP-SA, that generates a set of local (minimal) shortcut candidates by perturbing a single solution derived from $k\rho$-DP to maximize the inter-set alignments.



(a) The additional shortcuts considered by $k\rho$-DP-PC in comparison to $k\rho$-DP for any pair of nodes (u, v) for which v is not within the (k, ρ)-ball of u.

(b) At most two *leaves* of any node in a shortest path tree with fewest hops are considered for the score.

(c) A part of the shortest path tree decomposition T'_u for u induced by the shortcut set S_u derived from $k\rho$-DP.

Figure 3 Schematics for the central components of $k\rho$-DP-PC (a), (b) and $k\rho$-DP-SA (c).

5.1 Pair shortCutting: $k\rho$-DP-PC

For every node pair (u, v) with $u \in V$, $v \in N_\rho(u)$, and $\hat{d}(u, v) > k$, consider a shortest path with fewest hops from u to v. Let $S_{u,v}$ be the set of node pairs on this path such that any tuple (x, y) in $S_{u,v}$ successfully asserts that a shortcut from x to y would move v into the (k, ρ)-ball of u:

$$S_{u,v} = \left\{(x, y) \mid \hat{d}(u, x) + 1 + \hat{d}(y, v) \leq k\right\} \tag{i}$$

Observe that a candidate $(x, y) \in S_{u,v}$ can appear in other contexts $S_{u',v'}$ as well. Hence, we rate its global relative importance as the sum of local scores. The local score of node u is the reciprocal of the number of vertices that u is still missing from its (k, ρ)-ball if only (x, y) were inserted. Then, we accept all candidates with scores exceeding $\mu + 3 \cdot \sigma$ where μ and σ denote the mean and standard deviation of all scores within the shared hash map. Finally, we run the original $k\rho$-DP heuristic on the resulting graph to form the remaining (k, ρ)-balls.

Notice that reducing ρ with respect to n reduces the number of interactions between distinct shortcuts. To preserve the possibility of finding exceptional shortcuts we only include shortcuts where at least k distinct vertices participated in its score and every of these distinct vertices has at least two nodes which are moved into their (k, ρ)-ball by the addition of the shortcut. Furthermore we use the concept of *important breadth* (Figure 3(b)) since two leaves suffice so that our heuristic will give a higher score to the parent of both leaves than to any leaf individually, further considered leaves only exaggerate the importance of this shortcut while overshadowing shortcuts which decrease the depth e.g., a γ-PITCHFORK gadget versus a path of length $2k + 1$. In addition to that the usage of the average and standard deviation allows us to select only exceptional shortcuts to increase the probability that such a shortcut improves the final solution.

For a thorough discussion of the runtime refer to the full version of this paper [22].

5.2 Set Alignment: $k\rho$-DP-SA

Given a graph $G = (V, E)$, let $S = S_1 \cup S_2 \cup \cdots \cup S_n$ be a shortcut set, where S_i denotes shortcuts starting in node i as computed by $k\rho$-DP. Then, for each node i, we can decompose its shortest path tree T_i as illustrated in Figure 3(c) into separate subtrees induced by S_i. For each $(i, x) \in S_i$ define the subtree $T_{i,x}$ rooted in x such that all leaves of $T_{i,x}$ are either leaves of T_i or the parents of a node y: $(i, y) \in S_i, y \neq x$. Now, let $A = (v_1 = i, v_2, \ldots, v_e = x)$ denote a shortest path with fewest hops in G connecting i to x and let

$A_S := \{v_z \mid v_z \in A \wedge (i, v_z) \in S_i\}$ be the set of nodes on this path which are the target of a shortcut from i. For every subtree $T_{i,x}$, where $\mathrm{depth}(T_{i,x}) < k-1$ we can construct a set $S'_{i,x,t} := (S_i \cup \{(t,x)\}) \setminus \{(i,x)\}$ where $t \in \{v_{z+\delta} \mid v_z \in A_S, 0 \le \delta < k-1-\mathrm{depth}(T_{i,x})\}$ which also constitutes a minimal shortcut set for node i.

Hence for any node i, we build the set $S_D := \bigcup_{x,t} S'_{i,x,t} \cup S_i$ of distinct shortcuts being part of an locally optimal solution for i and increase the score of such a shortcut by one in a concurrent shared hash map[3]. Any entry in it displaying a score of more than one signifies a shortcut that is in the intersection of at least two locally optimal solutions of unique nodes.

The number of perturbations a single node i can construct of its shortcut set is bounded by $\mathcal{O}(|S_i| \cdot (k-1) \cdot \max_{A_S} |A_S|)$. By Lemma 14 this is again bounded by $\mathcal{O}(\rho \cdot \max_{A_S} |A_S|)$. This increases our span bounds as $\max_{A_S} |A_S| = \Omega(\frac{\rho}{k})$ for some graphs, e.g., line graphs. However, by only considering a subset of $\log \rho$ predecessors for any shortcut, we can bound the span to $\mathcal{O}(\rho \log \rho)$ and work to $\mathcal{O}(n\rho \log \rho)$; this matches the performance of $k\rho$-DP. Notice that in practice this limitation is often insignificant as many random graph classes including Gilbert random graphs have in expectation either bounded shortest paths lengths of $\mathcal{O}(\log n)$ [6] under reasonable assumptions on the edge weight distributions or at least a bounded diameter of $\mathcal{O}(\log(n)^{2/(3-\gamma)})$ [14] in the case of hyperbolic random graphs for $2 < \gamma < 3$.

▶ **Lemma 14.** *Let S_i be the shortcut set computed for node i by $k\rho$-DP. Then, $|S_i| < (\rho+1)/(k-1)$.*

Proof. Let T_i be the shortest path tree with fewest hops rooted in i containing its ρ-nearest neighbors. We decompose T_i into $L_0, L_1, L_2, ..., L_{k-1}$ where each is defined by $L_j := \left\{x \mid x \in T_i,\ \hat{d}(i,x) \equiv j \mod k\right\}$. Observe that $\sum_j |L_j| = |T_i|$, thus $\exists j : |L_j| \le \frac{|T_i|}{k} < \frac{\rho+1}{k-1}$. The claim follows since the set $S'_i = \{(i,x) \mid x \in L_j\}$ is discoverable by $k\rho$-DP and forms a (k,ρ)-ball for i. ◀

5.3 MinHash

Both presented heuristics can still introduce redundant shortcuts originating from different local contexts; for instance, consider partially overlapping shortcuts along a path. In the following, we propose a probabilistic approach based on MinHash[8] to detect synergetic shortcuts that partially overlap.

The omission of shortcuts displaying a high degree of overlap results in a marked redundancy reduction and in turn improves the solution size of our heuristics. For the missing details refer to the full version of this paper [22].

6 Optimal algorithm

In this section, we present – to the best of our knowledge – the first exact solver for $k\rho$-MSP. Since we established the problem to be $\mathcal{NP}$-hard in Section 3, there is little hope for a solution that is efficient on all instances. Hence, we propose an integer linear programming (ILP) formulation to harness the extensive research into efficient ILP solvers. Observe that this is an extension of the ILP provided by Gupta et al. [17] by allowing for a smaller ρ and in consequence we also need to account for a selection of the ρ'th nearest neighbors optimally.

[3] Since we only need to know if a shortcut is shared by at least two nodes there is no write congestion, hence $\mathcal{O}(1)$ in expectation.

Let $N_\rho(v) = \{u \in V \mid u \neq v,\ d(v,u) \leq r_\rho(v)\}$ be the set of vertices with distance at most the ρ-nearest neighbor, and define the shorthand notation $N_\rho^+(v) = N_\rho(v) \cup \{v\}$. Recall that $N_\rho(v)$ can contain more than ρ vertices in the case that there are several vertices with the same distance to v as the ρ-closest vertex. Define $w(u,v)$ as the weight of the edge $\{u,v\}$ if it exists or else the weight of a shortest path between u and v. Observe that in general $w(u,v)$ can exceed the shortest-path distance $d(u,v)$, that is if the edge (u,v) is not part of the shortest u-v-path. Finally, let S_{pot} be the set of possible shortcuts, i.e. $S_{pot} = \{(u,v) \mid u \in V,\ v \in N_\rho(u),\ (u,v) \notin E\}$.

We provide several variations of the ILP formulation for both the undirected *(U)* and the directed *(D)* case of $k\rho$-MSP. In contrast to the directed case, edges are bidirectionally usable in the undirected case, thus requiring a distinct formulation to encode the altered constraints. An explicit description of the formulation, correctness and a discussion on the encoding size can be found in the full version of this paper [22].

7 Experiments

In this section we compare the previously discussed algorithms on several random graph models. The ILP encoding is implemented in Python invoking Gurobi[4] 11.0.0. We implemented all heuristics (including those proposed by Blelloch et al. [7]) in the Rust programming language[5]. If not stated otherwise the following standard parameters and considerations apply to all experiments:

- We use several machines with AMD EPYC 7452 (64 threads) and 7702P (128 threads) CPUs, up to 512 GB RAM, *Ubuntu 22.04.3*, *Python 3.10.12* and *Rust 1.77.0*.
- We focus on the case $\rho = n-1$ as this confers two benefits:
  1. It is arguably the case with the most optimization potential.
  2. The difference in the observed solution is solely attributable to a superior choice of shortcuts since $N_\rho(v)$ is unique for every node v.
- Each ILP instance has a timeout of 1800 s after which it is deemed unsolvable.

We are heavily using random graph models as part of our evaluation as they have been found to emulate frequently found properties of real world networks while being mathematically tractable to a certain degree. The Gilbert model [15] functions as null model with almost no discernible structure. We denote it as $\mathcal{G}(n,p)$ where n is the number of nodes and p the independent probability of each edge to be present.

One prominent structural feature found in many observed networks are powerlaw degree distributions (i.e., a random node has degree X with $\mathbb{P}[X = x] \propto x^{-\gamma}$). [4] This is of special interest here, since our discussion in Section 4 suggests that such skewed distributions are hard instances for the existing heuristics. The hyperbolic random graph model [20] $\mathcal{H}(n,a,\gamma,T)$ with $\gamma = 3.0$, $a = 3$ and $T = 0$ is known to yield sparse graphs featuring powerlaw degree distributions and a non-vanishing local clustering coefficient [4]. Since the presence of a local community structure (e.g., short cycles etc.) may affect the shortcutting behavior, we also consider a Markov Chain randomization model [16] $\mathcal{MC}(G,s)$, where s refers to the number of switches per edge. It approximates a uniform sample from all simple graphs with prescribed degrees. Thus we use this model on previously generated hyperbolic random graphs with the same parameters as stated in Figures 5, 8, and 9 to quantify the impact

[4] `https://gurobi.com`
[5] Rust is a compiled language of comparable performance to C [10].

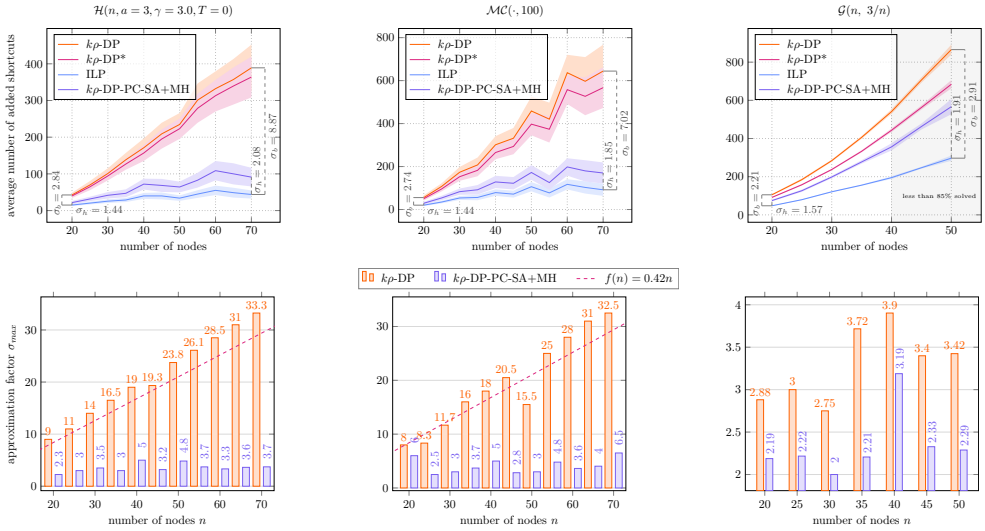


Figure 4 The comparison between the optimal algorithm and the heuristics for the $k\rho$-MSP problem for $k = 2$, $\rho = n - 1$ on several random graph classes for varying n. σ_h and σ_b describe the average approximation factor for $k\rho$-DP-PC-SA+MH and $k\rho$-DP respectively, while $\sigma_{max} = \max_i \sigma_i$ is the maximum approximation factor wittnessed on up to 50 sampled instances.

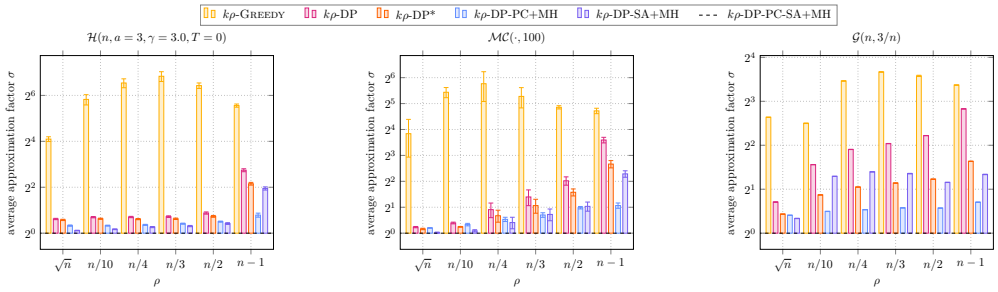


Figure 5 The average approximation factor of all heuristics compared to the baseline $k\rho$-DP-PC-SA+MH on various random graph classes for $n \leq 8000$ (the number of nodes fluctuates with the size of the largest connected component) and $k = 3$.

of the degree distribution mostly independently from other structural properties. Our final graph model is the random geometric graph model [25] denoted by $\mathcal{RGG}(n, r, d)$. This model also generates graphs with a local community structure but they lack the powerlaw degree distribution of the hyperbolic random graphs. Here r refers to the radius used to infer the connections between the random points and d to the dimension of the underlying geometric space.

7.1 Experimental evaluation

For our experiments we use four different heuristics:

1. $k\rho$-DP – The heuristic introduced by Blelloch et al. and the currently best known heuristic for this type of problem.
2. $k\rho$-Greedy – A faster heuristic which experimentally shows a worse performance compared to $k\rho$-DP.

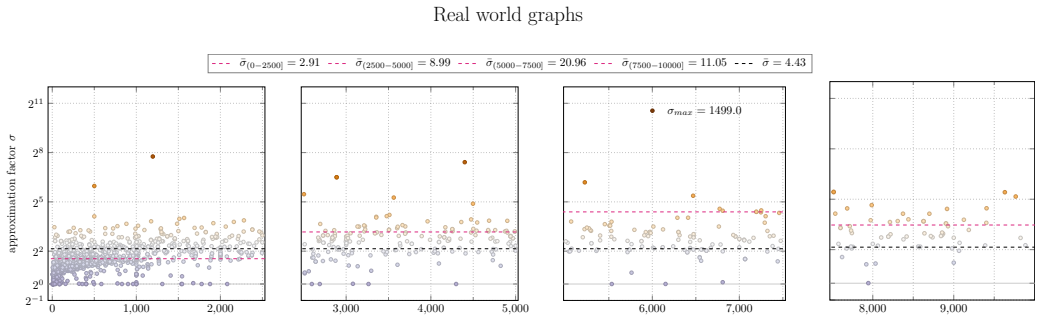


Figure 6 Approximation factor of $k\rho$-DP in relation to the baseline $k\rho$-DP-PC-SA+MH on all graphs from the network repository dataset [26] with up to 10000 nodes for $\rho = n - 1$ and $k = 3$.

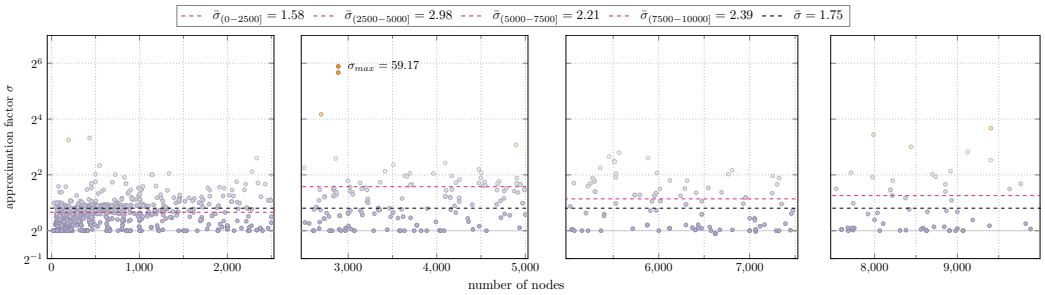


Figure 7 Approximation factor of $k\rho$-DP in relation to the baseline $k\rho$-DP-PC-SA+MH on all graphs from the network repository dataset [26] with up to 10000 nodes for $\rho = \lfloor \frac{n-1}{\log(n-1)} \rfloor$ and $k = 3$.

3. $k\rho$-DP* – Runs $k\rho$-DP in 10 consecutive phases. In each phase $\frac{n}{10}$ vertices are processed and shortcutted by $k\rho$-DP.
4. $k\rho$-DP-PC-SA+MH – A combination of the techniques described in Sections 5.1 and 5.2 with the additional usage of MinHashing.

Our construction in Section 4 to bound the approximation factor of $k\rho$-DP relies on carefully crafted graph structures. This raises the question, how the algorithm performs on "ordinary" networks. Thus our first experiment considers the approximation factor of $k\rho$-DP on $\mathcal{G}(n, p)$ graphs and two other random graph models. For each model and graph size, we sample 50 graphs and derive the average and maximum approximation factor.

For the $\mathcal{G}(n, 3/n)$ model, $k\rho$-DP displays a small approximation factor on average (less than three) in any case which is slowly growing with increasing graph order as depicted in Figure 4. In addition, the approximation factor is strongly concentrated around the average, suggesting that there are no commonly found exceptionally hard instances for $k\rho$-DP on $\mathcal{G}(n, 3/n)$. A similar behavior can be observed for random geometric graphs resulting in the same moderate improvement in the approximation factor for the newly introduced heuristic compared to $k\rho$-DP. The extended version [22] gives more details for this class of graphs.

Notice that our lower bound construction shares some similarities with graphs demonstrating a powerlaw degree distribution, namely the existence of high degree vertices which, if shortcutted just right, may complete many (k, ρ)-balls at once. Hence we assumed that these graphs are a hard input for $k\rho$-DP. Indeed our experiments show a quickly increasing average approximation factor for growing n for both random graph models with powerlaw degree distributions. For these two models the maximum approximation factor scales almost linearly with n. Unfortunately larger graphs are out of reach for our ILP formulation, thus preventing us to confirm this behavior on larger graphs.

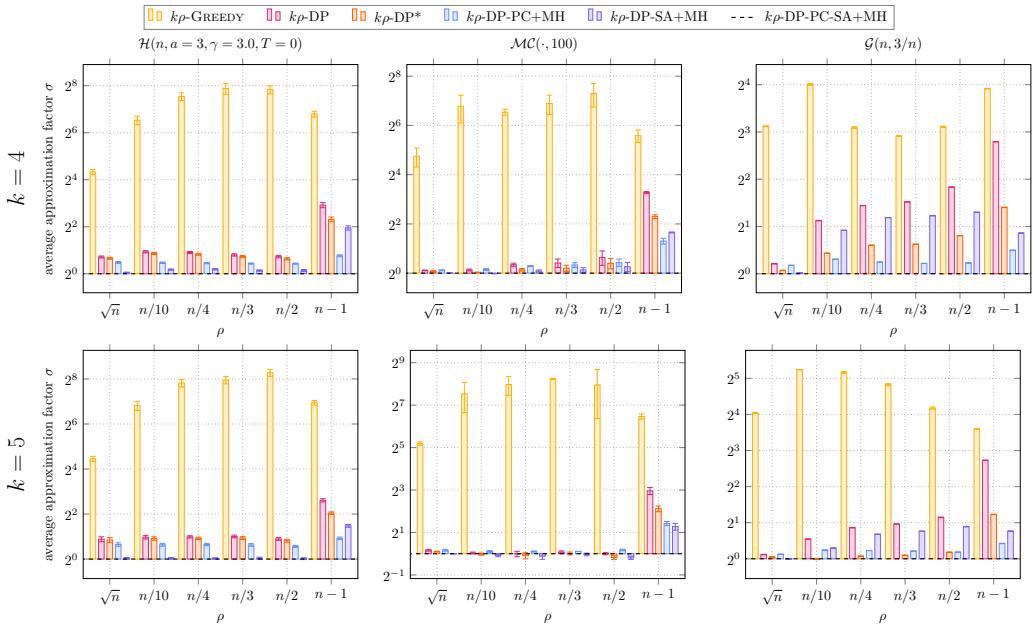


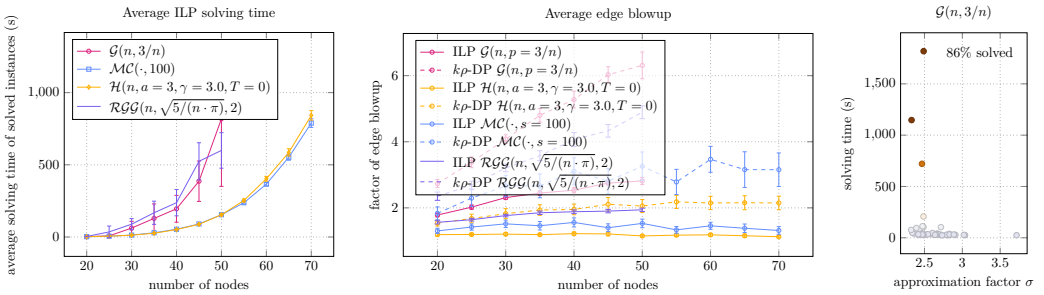
Figure 8 Shows the average approximation factor of all heuristics compared to the baseline $k\rho$-DP-PC-SA+MH on various random graph classes for $n \leq 8000$ (the number of nodes fluctuates with the size of the largest connected component).

Although the gap between $k\rho$-DP-PC-SA+MH and $k\rho$-DP on these graphs is larger than for Gilbert and random geometric graphs the gap does not scale with increasing graph size (see Figure 5). In addition Figures 5 and 8 show that the advantage of $k\rho$-DP-PC-SA+MH is highly dependent on a large ρ. This is unsurprising as our heuristic depends on synergies at a global scale which dramatically decrease for falling values of ρ. Still our new heuristic manages to improve upon the good performance of $k\rho$-DP by deriving solutions which are on average smaller by a factor of 1 to 8, depending on the concrete values of ρ and the underlying random graph model.

Observe that our heuristic is a multi-stage algorithm as it first invokes $k\rho$-DP-PC, then $k\rho$-DP-SA and finally $k\rho$-DP (i.e., each stage observes previously inserted shortcuts). This alone provides our heuristic with a considerable advantage compared to $k\rho$-DP which works completely oblivious of the other computed shortcuts. $k\rho$-DP* evidently simulates a continuously growing globally known shortcut set and allows us to further discern the root of better performance of our heuristic. As reported in Figure 4, $k\rho$-DP* barely improves over $k\rho$-DP on graphs with powerlaw degree distributions, while it performs significantly better on $\mathcal{G}(n, p)$. We attribute these observation to the fact, that multiple stages do not help to find and place globally good shortcuts as needed to solve powerlaw degree instance; yet they uncover "accidental" synergies in the large number of shortcuts needed for Gilbert graphs.

7.2 Measuring solution quality and speedup

As depicted in Figure 9(a), the average run time of our ILP solver grows dramatically in the graph size n in particular for Gilbert graphs and random geometric graphs. While we can solve all instances in the reported parameter regime for all other models within the time budget of 1800 s, the fraction of solved instances on Gilbert and random geometric graphs shrinks with increasing n, refer to the extended version for more details [22]. Thus we



(a) The average time for the ILP solver to solve the random graph instances with respect to their order.

(b) The average edge blowup i.e. the ratio of edges in the shortcutted graph to the edges in the input graph $|E \cup S|/|E|$ where S is the generated shortcut set.

(c) Solving time for the ILP as a function of the approximation factor σ for $n = 35$.

Figure 9 Miscellaneous results including average running time, average factor of edge increases and solvability on Gilbert random graphs.

visually highlight the regime with only partial results in Figure 4 to clearly mark potentially biased results. Despite various preliminary tests, no association between the solving time and the approximation factor could be found in this regime as depicted in Figure 9(c). Hence this reduces the probability that the unsolved instances induce a systemic bias on the investigated measures due to an association to the solving time.

For an input graph $G = (V, E)$ and a computed shortcut set S define the *edge blowup factor* as the relative increase $|E \cup S|/|E|$ of edges. Recall that a main use case for (k, ρ)-graphs are sharper theoretical bounds for stepping based PSSSP algorithms. The blowup factor directly affects the work of these algorithms, at least multiplicatively.

Surprisingly, for random graphs with powerlaw degree distributions, the ILP solution displays a small edge blowup factor of approximately 1.4 which in addition seems to decrease for increasing graph sizes. This suggests that the additional work for PSSSP algorithms induced by the (k, ρ) transformations are, on average, asymptotically irrelevant on these practically relevant graph classes. In comparison $k\rho$-DP initially sets out with an edge blowup factor of less than two for both aforementioned models which in line with previous results, subsequently displays an increasing trend on graphs of higher order. Contrasting this are the results for Gilbert random graphs, here both the heuristic and the ILP algorithm persistently increase their edge blowup factor for larger graphs.

In the spirit of the main application of our heuristic being faster parallel shortest path algorithms we provide a basic parallelized implementation of the aforementioned heuristics[6]. Our experiments indicate that the algorithms themselves are highly amenable to parallelization; more details can be found in the extended version [22]. Our experiments confirm that $k\rho$-Greedy is the fastest of the heuristics but it suffers from a large approximation factor and thus induces a larger overhead for most applications on the resulting graph. The other heuristics are sequentially around 2-7 times slower than $k\rho$-Greedy but provide a smaller shortcut set with the same guarantees as $k\rho$-Greedy.

[6] Neither the final shortcut traversal nor the edge insertions into the graph are parallelized yet.

7.3 Performance on real world graphs

In contrast to Blelloch et al. [7] we consider networks from the network repository [26], instead of graphs from the SNAP datasets [23]. On account of our different parameter regime for k and ρ, testing the heuristics (even the original ones), is prohibitively expensive for the graphs originally used by them. Nonetheless to quantify the robustness of our new heuristic, we consider observed networks from the network repository [26], including protein interactions, social, citation and neurological networks. We test the heuristics on 2748 graphs from [26] (those with less than 10000 nodes) and filter out those graphs which are already (k, ρ)-graphs. The results are split into four separate plots where each displays a specific range of n. We again use $k\rho$-DP-PC-SA+MH as baseline as illustrated in Figures 6 and 7. On the investigated graphs $k\rho$-DP-PC-SA+MH either matches the performance of $k\rho$-DP or improves upon it by up to several orders of magnitude. This culminates into a maximum observed approximation factor of 1499.0 for $\rho = n - 1$, i.e., on average for each edge added by $k\rho$-DP-PC-SA+MH, $k\rho$-DP adds 1499. When reducing ρ to $\lfloor \frac{n-1}{\log(n-1)} \rfloor$ we can still observe a maximum approximation factor of 59.17. Including exceptional instances, the average approximation factor over all tested graphs is $\bar{\sigma} = 4.43$ and $\bar{\sigma} = 1.75$ for $\rho = n - 1$ and $\rho = \lfloor \frac{n-1}{\log(n-1)} \rfloor$ respectively. This provides strong empirical evidence that our new heuristic is robust on practical instances for differing values of ρ.

7.4 Discussion

Our heuristics empirically show a good performance on real world networks and on various random graph models. Still, this performance is conditioned on the specific parameter regimes we tested for k and ρ. As can be seen in multiple figures, our heuristics depend on a large ρ to detect synergies between shortcut sets and their performance quickly degrades for a lower value of ρ. We would expect that our algorithms perform on par or worse than Blelloch et al. heuristics for small values of ρ and k, as at least $k\rho$-DP-PC can add unnecessary shortcuts. Although, this is a clear limitation for our heuristics, a large value of ρ and a small value of k is crucially needed to obtain sharper bounds for stepping based PSSSP algorithms [13]. Hence, the parameter regimes investigated are at least of theoretical relevance and further experiments could uncover their practical importance as well[7].

8 Conclusion

We showed new results regarding the complexity of $k\rho$-MSP and provided important theoretical insights into the lower bounds of the approximation factor of existing heuristics. These results allowed us to derive a new heuristic built upon the work of Blelloch et al. [7]. Experimentally, our heuristic showed good performance on random graph models as well as a wide variety of real world graphs. As such, our contributions in this work are two-pronged: We contributed novel theoretical insights on (k, ρ)-shortcutting algorithms, as well as a practical contribution which empirically showed the value of our developed heuristic on simulated and real-world data.

[7] Hub-Labeling ($k = 2, \rho = n - 1$) also led to a practical class of algorithms which are among the fastest ones for this kind of problems [1, 11].

References

1 Ittai Abraham, Daniel Delling, Andrew V. Goldberg, and Renato F. Werneck. Hierarchical hub labelings for shortest paths. In *Algorithms – ESA 2012*, pages 24–35, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

2 Haris Angelidakis, Yury Makarychev, and Vsevolod Oparin. Algorithmic and hardness results for the hub labeling problem. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '17, pages 1442–1461, USA, 2017. Society for Industrial and Applied Mathematics.

3 Maxim Babenko, Andrew V. Goldberg, Haim Kaplan, Ruslan Savchenko, and Mathias Weller. On the complexity of hub labeling (extended abstract). In *Mathematical Foundations of Computer Science 2015*, pages 62–74, Berlin, Heidelberg, 2015. Springer Berlin Heidelberg.

4 Albert-László Barabási. Network science book. *Network Science*, 625, 2014.

5 Richard Bellman. On a routing problem. *Quarterly of Applied Mathematics*, 16(1):87–90, 1958. URL: http://www.jstor.org/stable/43634538.

6 Shankar Bhamidi, Remco Van der Hofstad, and Gerard Hooghiemstra. First passage percolation on the erdős–rényi random graph. *Combinatorics, Probability and Computing*, 20(5):683–707, 2011.

7 Guy E. Blelloch, Yan Gu, Yihan Sun, and Kanat Tangwongsan. Parallel Shortest Paths Using Radius Stepping. In *Proceedings of the 28th ACM Symposium on Parallelism in Algorithms and Architectures*, SPAA '16, pages 443–454, New York, NY, USA, 2016. Association for Computing Machinery. doi:10.1145/2935764.2935765.

8 Andrei Z. Broder. On the Resemblance and Containment of Documents. In *Proceedings of the Compression and Complexity of Sequences 1997*, SEQUENCES '97, page 21, USA, 1997. IEEE Computer Society.

9 Edith Cohen, Eran Halperin, Haim Kaplan, and Uri Zwick. Reachability and distance queries via 2-hop labels. *SIAM Journal on Computing*, 32(5):1338–1355, 2003. doi:10.1137/S0097539702403098.

10 Manuel Costanzo, Enzo Rucci, Marcelo Naiouf, and Armando De Giusti. Performance vs Programming Effort between Rust and C on Multicore Architectures: Case Study in N-Body. In *2021 XLVII Latin American Computing Conference (CLEI)*, pages 1–10, 2021. doi:10.1109/CLEI53233.2021.9640225.

11 Daniel Delling, Andrew V. Goldberg, Ruslan Savchenko, and Renato F. Werneck. Hub labels: Theory and practice. In *Proceedings of the 13th International Symposium on Experimental Algorithms - Volume 8504*, pages 259–270, Berlin, Heidelberg, 2014. Springer-Verlag. doi:10.1007/978-3-319-07959-2_22.

12 Edsger W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1):269–271, 1959.

13 Xiaojun Dong, Yan Gu, Yihan Sun, and Yunming Zhang. Efficient stepping algorithms and implementations for parallel shortest paths. In *Proceedings of the 33rd ACM Symposium on Parallelism in Algorithms and Architectures*, pages 184–197, 2021.

14 Tobias Friedrich and Anton Krohmer. On the diameter of hyperbolic random graphs. *SIAM Journal on Discrete Mathematics*, 32(2):1314–1334, 2018. doi:10.1137/17M1123961.

15 Edgar N. Gilbert. Random Graphs. *The Annals of Mathematical Statistics*, 30:1141–1144, 1959. URL: http://www.jstor.org/stable/2237458.

16 Christos Gkantsidis, Milena Mihail, and Ellen W. Zegura. The Markov Chain Simulation Method for Generating Connected Power Law Random Graphs. In *Proceedings of the Fifth Workshop on Algorithm Engineering and Experiments, Baltimore, MD, USA, January 11, 2003*, pages 16–25. SIAM, 2003.

17 Siddharth Gupta, Adrian Kosowski, and Laurent Viennot. Exploiting Hopsets: Improved Distance Oracles for Graphs of Constant Highway Dimension and Beyond. In *46th International Colloquium on Automata, Languages, and Programming (ICALP 2019)*, volume 132, pages

143:1–143:15, Dagstuhl, Germany, 2019. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ICALP.2019.143.

18 Lester R. Ford Jr. *Network Flow Theory*. RAND Corporation, Santa Monica, CA, 1956.

19 Richard M. Karp. Reducibility Among Combinatorial Problems. In *Complexity of Computer Computations*, The IBM Research Symposia Series, pages 85–103. Plenum Press, New York, 1972.

20 Dmitri Krioukov, Fragkiskos Papadopoulos, Maksim Kitsak, Amin Vahdat, and Marián Boguñá. Hyperbolic geometry of complex networks. *Phys. Rev. E*, 82:036106, September 2010. doi:10.1103/PhysRevE.82.036106.

21 Alexander Leonhardt, Ulrich Meyer, and Manuel Penschuck. K-Rho-Shortcutting Heuristics. Software, swhId: swh:1:dir:8965d090c1d32ea024b1bb4b111329990a156b37 (visited on 2024-08-09). URL: https://github.com/alleonhardt/k-rho-shortcutting.

22 Alexander Leonhardt, Ulrich Meyer, and Manuel Penschuck. Insights into (k, ρ)-shortcutting algorithms, 2024. arXiv:2402.07771.

23 Jure Leskovec and Andrej Krevl. SNAP Datasets: Stanford large network dataset collection. http://snap.stanford.edu/data, June 2014.

24 Ulrich Meyer and Peter Sanders. Δ-Stepping: A Parallel Single Source Shortest Path Algorithm. In *Algorithms — ESA' 98*, pages 393–404, Berlin, Heidelberg, 1998. Springer Berlin Heidelberg.

25 Mathew Penrose. *Random Geometric Graphs*. Oxford University Press, May 2003. doi:10.1093/acprof:oso/9780198506263.001.0001.

26 Ryan A. Rossi and Nesreen K. Ahmed. The network data repository with interactive graph analytics and visualization. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015. URL: http://networkrepository.com.