Approximation Algorithms and Hardness for n-Pairs Shortest Paths and All-Nodes Shortest Cycles

1 st Harini Kamarthy Computer Science(of Aff.) University of North Texas (of Aff.) Denton, UnitedStates harinikamarthy@my.unt.edu

2 nd Vaishnavi Gunna Computer Science (of Aff.) University of North Texas(of Aff.) Denton, UnitedStates vaishnavigunna@my.unt.edu

***Abstract*— In this research, we delve into the study of approximability of the two interconnected problems on graphs: n-Pairs Shortest Paths (n-PSP) and All Node Shortest Cycles (ANSC). The n-PSP problem aims to identify the shortest path between a predetermined set of O(n) pairs, while ANSC focuses on finding a shortest cycle passing through every node. While the approximate version of n-PSP has been explored before, mostly in context of distance oracles, the problem of ANSC has predominantly been examined in terms of exact algorithms.**

With the wide range of techniques and the conditional lower limits that balance approximation ratio and running time, we give the thorough examination of the approximability of n-PSP and ANSC. Notably, our conditional lower bound findings show that, there exists no combinatorial method for an unweighted and undirected n-PSP that has an approximation ratio greater than the 1+1/k and runs in O(m2−2/(k+1)n1/(k+1)−ε) time under the combinatorial 4k-clique hypothesis. This almost matches an upper bound that Agarwal's 2014 finding predicted.

The approaches from the girth issue, distance oracles, approximate All Pair Shortest Paths (APSP), spanners, fault-tolerant, link-cut trees are just few of the methods used by our algorithms. For any constant > 0, one of the significant algorithmic results of our research is the creation of effective solutions for both n-PSP and ANSC, attaining the time complexity of O (m + n3/2+ ε) with an approximation factor of 2 + ε (and additive error dependent on ε). Our conditional lower bounds show that for any subquadratic-time combinatorial method, this approximation ratio is almost an optimal for n-PSP. To accomplish a time/accuracy trade-off, we also expand these algorithms for both n-PSP and ANSC to include near-linear time techniques.

By presenting a comprehensive analysis of approximability of n-PSP and ANSC, along with a range of algorithms and conditional lower bounds, this research contributes to the understanding of these fundamental graph problems. The results obtained pave the way for further advancements in solving these problems efficiently while considering the trade-offs between running time and approximation quality.

***Index Terms*—n-Pairs Shortest Paths, Approximation Algorithms, All Node Shortest Cycles, Distance Oracles, Combinatorial Algorithms, All Pair Shortest Paths, Girth Problem, Spanners, Approximation Ratio, Lower Bounds, Graph Algorithms**

I. INTRODUCTION

ANSC and the n-PSP are the two basic issues that arise when calculating distances in graphs. In n-PSP, set of vertices (si, ti) for $1 \leq i \leq O(n)$ are provided together with a network with n nodes and m edges. Finding shortest path between each si and its matching ti is the goal.

The objective of the second problem, known as ANSC, is to determine the shortest cycle that encompasses each vertex v in the network with m edges and n nodes. We seek to determine length of shortest cycle with each vertex v, designated as SC(v). We shall examine the essential similarities and differences between n-PSP and ANSC in this research.

II. MOTIVATION

We give some background knowledge to better comprehend n-PSP issue.

*1) **N-PSP Problem:*** The n-PSP has been of interest since the 1990s when Aingworth, Indyk, Chekuri, and Motwani achieved an additive 2-approximation algorithm with run time complexity of O˜(n2) [8]. However, subsequent research on this problem has been largely overshadowed by advancements in distance oracles [4] [7].

Although explicit studies on n-PSP have been scarce since the 90s, related problems concerning distances between pre-specified vertex pairs have been explored. For instance, pairwise distance preservers, pairwise spanners, and pairwise reachability preservers have been extensively studied [5] [9].

The motivation for studying n-PSP stems from the fact that computing all distances in graph, done in All-Pairs Shortest Paths(APSP) problem, could be time-consuming and unnecessary for many applications. Instead, researchers have investigated whether it is possible to compute specific distances more efficiently than computing all distances.

This query is tackled in a variety of ways. SSSP issue, which could be solved quicker than the APSP using Dijkstra's algorithm in runtime of O(m + n log n), is one method for computing all the distances from a single source. However, SSSP only calculates distances from one source.

The text discusses the concept of distance oracles, data structures that allow efficient querying of distances between vertices. It also points out that n-PSP problem is of special case where the pairs of vertices to compute distances for are known in advance [10]. The computation of arbitrary sets of distances may not be covered by this method as it only considers extremal distances.

The main contributions and results for the n-PSP problem are given in terms of approximation algorithms and conditional lower bounds. They also compare their results with existing algorithms and show that certain approximation factors are nearly tight.

The O(n) pre-specified vertex pairs scenario is addressed by the n-PSP problem, in contrast. The crucial question is whether n-PSP algorithms can outperform those that are already-existing distance oracles directly imply.

This study investigates the responses to this query in various contexts. For instance:

- We show that the n-PSP method developed from Agarwal's distance oracle [1] achieves approximately the optimum run time with combinatorial 4k-clique hypothesis in domain of (1 + 1/k) approximation.

- We describe n-PSP technique that beats the cutting-edge distance oracle put forward by Chechik and Zhang [16] for $(2 + \varepsilon, \beta)$-approximations.

In summary, this study investigates the n-PSP problem and its distinction from other distance computation problems. We analyze different approximation regimes and demonstrate algorithms that outperform existing distance oracles in terms of run time and approximation quality.

Let's see for ANSC problem in the same way.

2) ***ANSC Problem****:* Prior studies have also looked at the ANSC issue, which is the task of determining the smallest cycle that contains each vertex v in a given graph. With integer weights ranging from 1 to M, Yuster [12] described a randomized method for undirected networks that has a temporal complexity of O (Mn(+3)/2). After that, Sankowski and W egrzycki [11], as well as separately Agarwal and Ramachandran [2], established a deterministic method for unweighted directed networks with a time complexity of O (n). Another truncation from Replacement Paths issue to the weighted and directed ANSC was offered by Agarwal and Ramachandran [2]. The Replacement pathways issue involves locating alternate, shortest pathways through a given graph that do not pass over a certain edge. In order to maintain range of edge weights, Nechushtan and Chechik [6] modified the reduction by Agarwal and Ramachandran [2], which raised edge weights by factor of n.

While earlier research concentrated on the precise version of ANSC, to the best of our knowledge, we are the first to look at its approximability. We propose numerous algorithmic solutions to the ANSC problem in this study, including the following:

- We construct $(2+\varepsilon, \beta)$-approximation procedure, for ANSC with subquadratic run time.

- We present k-approximation algorithm for ANSC, obtaining the running time complexity which can be comparable to most known k-approximation girth approach.

These results contribute to the understanding of the ANSC problem and provide efficient approximation algorithms that balance running time and approximation quality.

A) Results for n-PSP and ANSC that are inferred from earlier work

We begin by establishing two crucial findings that link the n-PSP and ANSC issues. The appendix contains the evidence supporting these observations.

The first finding is that in weighted graphs, precise n-PSP is reduced to exact ANSC. The second finding is a conversion of directed graphs from ANSC to n-PSP that may be used in any approximation scenario with a finite approximation factor.

We may make use of the current approximate APSP methods to generate the approximation algorithms for n-PSP and the directed ANSC. However, because of the quantity of the APSP output, the running durations of these algorithms are necessarily at least Ω(n^2). By querying the appropriate pairs from the APSP output, we can approximate the both n-PSP and the directed ANSC with run time of O(T(n, m)), where T(n, m) is time complexity of an APSP algorithm.

Several algorithms for exact and approximate APSP are available. Exact algorithms for APSP include approaches based on matrix multiplication, with complexities such as O(nω) for undirected with unweighted graphs, whereas O(n^2.529) for directed with unweighted graphs. While there are numerous other approximation algorithms for APSP, our focus is on subquadratic time algorithms, and so, we will not be providing a detailed description of all these methods.

Using estimated techniques is an additional method to gain estimation techniques for the unsupervised n-PSP. FDOs. DOs are data structures that enable distance queries with specific approximation guarantees. By querying all input pairs, the same approximation ratio as the DO can be achieved for n-PSP with time complexity of O(n)·q(m, n), where q(m, n) is a query time of the DO.

Notably, none of the aforementioned methods directly produce ANSC subquadratic approximation algorithms in both the directed or undirected graphs. For the directed situation, there are no non-trivial distance oracles, however reductions from Girth lower limits do apply to ANSC. Furthermore, the k-Cycle Hypothesis demands a time complexity of m^(2−O(1)) for every Girth approximation better than a 2-approximation.

In summary, prior work offers insights into solving n-PSP and ANSC problems. Existing algorithms for exact APSP and approximate distance oracles can be leveraged to obtain approximation algorithms for n-PSP. However, these approaches do not directly provide subquadratic approximation algorithms for ANSC, and conditional lower bounds from the Girth problem apply to ANSC.

B) Our Results:

In our study, we focus on approximability of two fundamental graph problems: n-PSP and ANSC. With a variety of techniques and conditional lower limits that balance running time and approximation quality, we attempt to give a thorough understanding of these issues.

1)Main Hardness Assumptions

Before presenting our results, let us introduce the main hardness assumptions that underpin our lower bound proofs. These assumptions are well-established in the field of fine-grained complexity. While we employ multiple assumptions, we highlight two central ones in this summary.

The first hypothesis, known as the Combinatorial k-Clique Hypothesis, concerns hardness of finding k-cliques in the graphs. It asserts that there is no combinatorial algorithm that can detect a k-clique in a given graph in O(n^(k-ε)) time, in which n represents number of vertices and ε a positive constant.

The second hypothesis, referred to as the (k, r)-Hyperclique Hypothesis, focuses on the problem of detecting k-cliques in r-uniform hypergraphs. It states that there is no algorithm that can solve this problem in run time of O(n^(k-ε)), where n denotes number of vertices and ε a positive constant.

2)Hardness Results

We first presented the hardness results for both n-PSP and ANSC, demonstrating that solving these problems with high accuracy is computationally challenging. The idea of extremal versions of these issues is introduced, where the solution is either smallest or largest value among O(n) values returned. They mention cycle diameter as biggest number in an ANSC output and the cycle girth as length of smallest cycle on the graph.

For n-PSP, they establish the hardness result against (2-ε)-approximation by combinatorial algorithm. The running time required by such an algorithm is shown to be at least n^(3/2 - o(1)). To overcome this limitation, we introduce a broader interpretation of Combinatorial Dense Triangle Hypothesis, called Combinatorial k-Clique Hypothesis, which leads to stronger running time bounds under certain conditions.

Trade-off Lower Bounds: The paper investigates the trade-offs between the run time and approximation ratio for both combinatorial and non-combinatorial algorithms. It derives several corollaries from Theorem I.2, which provide specific lower bounds on the trade-off between time and accuracy in different scenarios.

The trade-off versus combinatorial algorithms that run more quickly but have greater approximation ratios is illustrated by Corollary I.1. Assuming Combinatorial k-Clique Hypothesis, it states that for k ≥ 4, no combinatorial algorithm for an unweighted and an undirected n-Pairs Minimum Distance can obtain an approximation ratio which is greater than the (3-4/k) in time complexity of m n (1/(k-2) - ε) or m + n^(k/(k-2) - ε).

Comparison with Independent Work: The paper also discusses the recent work by Abboud, Khoury, Bringmann, Zamir [3], who suggested for "cycle-removal" framework and achieved lower bounds for graph problems which are related to distances and girth. While their results focus on arbitrary large constant approximation ratios, the authors' results concentrate on approximation ratios closer to 1, such as 1 + 1/k - ε. The paper highlights the similarities and differences between their findings and the ones presented in this work.

Approximation	Running time LB	Theorem	Hypothesis	Comments
$5/3-\varepsilon$	$m^{1+\delta-o(1)}$	[1]	Sparse Triangle	
$5/3-\varepsilon$	$n^{\omega-o(1)}$	[1]	Dense Triangle	
$5/3-\varepsilon$	$n^{\omega-o(1)}$	[1]	Simplicial Vertex	for n-Pairs Diameter
$2-\varepsilon$	$n^{3/2-o(1)}$	Thm I.1	Comb. BMM	
$2-\varepsilon$	$m+n^{2-o(1)}$	Cor I.1	Comb. 4-clique	**nearly matches Thm I.3**
$3-4/k-\varepsilon$	$m+n^{k/(k-2)-o(1)}$	Cor I.1	Comb. k-clique	
$1+1/k-\varepsilon$	$m^{2-2/(k+1)}n^{1/(k+1)-o(1)}$	Cor I.2	Comb. $4k$-clique	**nearly matches [15]**
$1+1/(k+0.5)-\varepsilon$	$m^{2-3/(k+2)}n^{2/(k+2)-o(1)}$	Cor I.3	Comb. $(4k+2)$-clique	**nearly matches [15]**
$3/2-\varepsilon$	$mn^{1/2-o(1)}$	Cor I.4	$(4,3)$-hyperclique	
$3-6/k-\varepsilon$	$m+n^{k/(k-2)-o(1)}$	Cor I.4	$(k,3)$-hyperclique	
any finite	$m^{2-o(1)}$	[1]	k-cycle	directed graphs

TABLE 1

n-PSP - Conditional Lower Bounds – Unweighted, Undirected Graphs. n-Pairs Minimum Distance

Approximation	Running time LB	Theorem	Hypothesis	Comments
$4/3-\varepsilon$	$m^{1+\delta-o(1)}$	[1]	Sparse Triangle	for Girth
$4/3-\varepsilon$	$n^{\omega-o(1)}$	[1]	Dense Triangle	for Girth
$7/5-\varepsilon$	$n^{\omega-o(1)}$	[1]	Simplicial Vertex	for Cycle Diameter
$3/2-\varepsilon$	$m^{4/3-o(1)}$	[1]	All-Edges Sparse Triangle	
$3/2-\varepsilon$	n^2	[1]	unconditional	

TABLE II

ANSC - Conditional Lower Bounds – Unweighted, undirected Graphs

3)Algorithmic Results

In both directed and undirected graphs, we look into approximation techniques for the n-PSP and ANSC issues. We explore the dependence between the number of edges (m) and nodes (n) in run times of the algorithms. The paper presents two categories of algorithms: those with multiplicative dependence and those with additive dependence on n and m.

i)n-PSP Problem:

For an n-PSP problem, paper focuses on the undirected case of n-PSP problem and presents the algorithmic results. It introduces an approximation algorithm that achieves a nearly 2k-2 approximation. However, the question of whether a k-approximation can be achieved with similar running time remains unanswered. (2+ε, β)-approximation solution for an n-PSP issue with a construction time of O~(m+n^(5/3+ε)) is implied by earlier work by Zhang and Chechik on distance oracles for unweighted and undirected graphs. The authors do provide a quicker technique, though, that completes the same (2+ε, β)-approximation in runtime of O~(n^(3/2+ε)+m. Notably, this runtime is polynomially quicker than the O(n^2+m) time complexity discussed above. The proposed algorithm's performance is nearly optimal, as it closely matches the lower bound derived from Corollary I.1.

ii)ANSC Problem:

For the ANSC problem, the paper presents algorithmic results for the both directed and undirected graphs. In the directed case, the authors generalize previous results from the Girth problem, achieving a (k+ε)-approximation algorithm for an ANSC with a running time complexity of O~(n^(1/k)log(M)), where M represents maximum edge weight.

In undirected graphs, the paper highlights the better approximation ratio achievable for ANSC compared to n-PSP. Specifically, (k+ε)-approximation algorithm for an ANSC is represented with a time complexity of O~(mn^(1/k)log(M)). It is worth noting that the dependence on m in this algorithm is expected, as executing Dijkstra's algorithm to completion is necessary for ANSC.

Attaining run times of type m+n^(2-ε) is the major goal of the algorithmic solutions. The "proof of concept" method used to introduce the paper which shows that, there is ANSC algorithm with the constant multiplicative and additive factors that runs with time complexity of O~(m+n^(2-1/6)). We then provide an enhanced approach that, in running time of O~(n^(15+ε)+m), achieves (2+ε, β)-approximation for ANSC. Main objective is to minimize the multiplicative approximation ratio to about 2, which corresponds to the n-PSP procedure.

III. Technical Overview Of Algorithms

We utilize a range of techniques from various problem domains to tackle the n-PSP and ANSC problems. These techniques include girth, all-pairs shortest paths (APSP), Spanners, distance oracles, link-cut trees, the simplicial vertex problem, fault-tolerant spanners.

The technical overview here highlights several key aspects, including the collection of conditional lower bounds and the development of approximation algorithms.

A. *Conditional Lower Bounds:*

We develop conditional lower limits for fundamental issues including k-cycle, triangle detection, k-clique based on conventional hardness assumptions. Reductions from these difficulties allow for the achievement of these lower limitations. The transformation of Combinatorial k-Clique Hypothesis into the n-Pairs Minimum Distance issue is particularly significant. This reduction not only gives tight constraints that are in line with some of the techniques created for the low-approximation domain, but also reduces bounds for approximation ratios larger than 2. The scientists believe that this significant decrease will lead to other findings on fine-grained hardness for problems associated with it.

Approximation	Running time	Theorem	Comments
$(2+\varepsilon, f(\varepsilon))$	$m+n^{3/2+\varepsilon}$	Thm I.3	$\varepsilon > 0$, some function f, **nearly matches Cor I.1**
$2k-2$	$mn^{1/k}$	[1]	integer $k \geq 2$
$(2k-1)\cdot(2k-2)$	$m+n^{1+2/k}$	[1]	integer $k \geq 2$
2	$m \cdot n^{(1+\omega)/8}$	[1]	for ST-Shortest Paths, $\lvert S\rvert, \lvert T\rvert = O(\sqrt{n})$

TABLE III

n-PSP -- Approximation algorithms - undirected unweighted graphs

Approximation	Running time	Theorem	Comments
$2k+1+\varepsilon$	mn^{α_k}	[1]	directed graphs, α_k solves $\alpha_k(1+\alpha_k)^{k-1} = 1-\alpha_k$
$2+\varepsilon$	$mn^{1/2}$	[1]	directed graphs, **nearly matches Thm 5.1 of [30]**
$k+\varepsilon$	$mn^{1/k}$	Thm I.4	$\varepsilon > 0$ and integer $k \geq 2$
nearly $1+1/(k-1)$	$m^{2-2/k}n^{1/k}$	[1]	integer $k \geq 2$
$(6,1)$	$m+n^{2-1/6}$	Thm I.5	
$(2+\varepsilon, \beta)$	$m+n^{1.5+\varepsilon}$	Thm I.6	$\varepsilon > 0$ and β is a function of ε
$(k^2, k^3 2^{k+1})$	$m+n^{1+2/k}$	[1]	integer $k \geq 2$

TABLE IV

ANSC -- Approximation algorithms - undirected unweighted graphs

To illustrate the reduction, consider the case of k=4. We will construct a 5-partite graph G', consisting of vertex partitions V12, V23, V34, V41, and V'12. The Vertices in V12 and V'12 are indexed by vertex pairs (v1, v2) from input graph G, while the other partitions are similarly indexed according to the subscripts. Edges are added between adjacent partitions based on the existence of 3-cliques in G. The reduction demonstrates that if G contains a 4-clique, then there exists length-4 path in G' between certain vertices. Conversely, if G does not contain 4-clique, the shortest path from (v1, v2) in V12 to (v1, v2) in V'12 must have a length of at least 8. Using this reduction, an algorithm for n-Pairs Minimum Distance problem with better than 2-approximation ratio and running time complexity of m*n^(1/2-ε) would imply the ability in solving the 4-clique problem in a contradicting time complexity.

We extend this reduction to larger values of k and provide combinatorial argument to identify vertices participating in k-clique when there exists the path that is too short. So, this argument allows to establish lower bounds with increasing approximation ratios as k grows, including the capture of hypercliques. Despite having somewhat weaker exponents than combinatorial lower limits based on the (r, k)-hyperclique hypothesis, they nonetheless aid in grasping the complexity of the issue.

Algorithmic Results:

Paper presents a collection of approximation algorithms for the both ANSC and n-PSP, considering the undirected and the directed graphs. Algorithms aim to achieve the time complexities of form O~(m + n^(2-ε)), allowing algorithms for thick networks to run in almost linear time.

For n-PSP undirected graphs, the paper introduces a nearly 2k-2 approximation algorithm. Additionally, a faster algorithm with (2+ε, β)-approximation is presented, demonstrating improved running time compared to previous algorithms.

Regarding ANSC, the paper provides approximation algorithms for the both directed and undirected graphs. Algorithms for directed graphs yield various approximation ratios, highlighting the distinction between ANSC and n-PSP in this setting. For undirected graphs, a (k+ε)-approximation algorithm is presented with a time complexity of O~(m*n^(1/k)*log(M)). Furthermore, paper introduces algorithms with time complexities of form O~(m + n^(2-ε)) to ANSC, improving upon earlier "proof of concept" algorithms.

B. *Approximation Algorithms:*

i) CYCLEESTIMATIONDIJKSTRA data structure for ANSC:

Common methods include running the Dijkstra's algorithm, among random sample of vertices and truncated version of Dijkstra's algorithm among the larger selection of set of vertices to approximate the distances and estimate the girth in graphs. The scenario is more complicated for the ANSC problem, though. Estimate of the shortest cycle (SC) for each vertex on the cycle must be updated when a cycle is found using Dijkstra's algorithm for ANSC. To address this, the authors provide CYCLEESTIMATIONDIJKSTRA, a data structure that combines the strength of link-cut trees and another version of

Dijkstra's algorithm. All vertices from this data structure are effectively kept up to date with pertinent cycle information.

ANSC warm-up method uses CYCLEESTIMATIONDIJKSTRA's data structure along with conventional sampling and other modified Dijkstra techniques to provide 2-approximation in running time of O~(m*n^(1/2)).

ii) Time/accuracy trade-off for a better running time:

Paper focuses on algorithms that can achieve improved running times, even if they result in worse than the 2-approximation ratios. For an n-PSP, it is observed that 2-approximation algorithm with the O~(m*n^(1/2)) run time could be integrated into basic case of Thorup-Zwick distance oracles.

We use the undirected setting to simplify several portions of the technique even though detecting cycles in undirected graphs poses difficulties do not present in directed networks. Interestingly, compared to the directed situation, this reduction produces superior limits for the undirected scenario. The labeling approach is employed in conjunction with an induction in the directed case, but we find that, this induction is not efficient for the undirected graphs. Its removal leads to algorithm with better approximation factor and run time.

iii) Time/accuracy trade-off for a better approximation ratio time:

In pursuit of obtaining better than a 2-approximation while accepting slower running times than O~(m*n^(1/2)), we draw the inspiration from an algorithm by Baek Tejs Knudsen, Dahlgaard for approximating the girth of graph. They adapted almost similar algorithm from Girth to ANSC, with only differences being use of the edge sampling in place of the vertex sampling and integration of CYCLEESTIMATIONDIJKSTRA's data structure.

n-PSP: We provide a brief explanation of (2+ε, β)-approximation approach for an n-PSP in running time of O~(m + n^(3/2)), for any constant ε > 0. (1+ε, β) spanner with dimension of n^(1+ε) is computed for O(n^(3/2)) edge subgraph that contains the incident edges of all the possible vertices with degree of no more than n^(1/2). Then, this spanner is subjected to O(m*sqrt(n))-time 2-approximation method.

2-approximation Algorithm for n-PSP:

```
In [1]: import heapq
        import sys

        def approximate_n_path_shortest_path(graph, source, target):
            num_vertices = len(graph)
            visited = set()
            distances = [sys.maxsize] * num_vertices
            parent = [-1] * num_vertices

            # Step 1: Initialize visited set and priority queue
            visited.add(source)
            distances[source] = 0
            pq = [(0, source)]

            # Step 2: Dijkstra's algorithm
            while pq:
                dist, current_vertex = heapq.heappop(pq)

                # Step 3: Check if target vertex is reached
                if current_vertex == target:
                    break

                # Step 3 (continued): Explore neighbors
                for neighbor in graph[current_vertex]:
                    edge_weight = 1  # Assuming unweighted graph
                    new_dist = dist + edge_weight
                    if new_dist < distances[neighbor]:
                        distances[neighbor] = new_dist
                        parent[neighbor] = current_vertex
                        heapq.heappush(pq, (new_dist, neighbor))
                        visited.add(neighbor)

            # Step 4: Check if target vertex is reachable
            if parent[target] == -1:
                return None  # No path from source to target

            # Step 5: Construct approximate shortest path
            path = []
            current_vertex = target
            while current_vertex != source:
                path.append(current_vertex)
                current_vertex = parent[current_vertex]
            path.append(source)

            # Step 6: Return the approximate shortest path
            return list(reversed(path))

        # Example usage
        graph = [
            [1, 2],     # Neighbors of vertex 0
            [0, 2, 3], # Neighbors of vertex 1
            [0, 1, 3], # Neighbors of vertex 2
            [1, 2, 4], # Neighbors of vertex 3
            [3]         # Neighbors of vertex 4
        ]
        source = 0
        target = 4

        result = approximate_n_path_shortest_path(graph, source, target)
        if result is not None:
            print("Approximate Shortest Path:", result)
        else:
            print("No path from source to target.")

        Approximate Shortest Path: [0, 1, 3, 4]
```

2-approximation Algorithm for n-PSP

ANSC: To address the challenges posed by the lack of direct guarantees on cycle lengths in spanners, the authors propose the use of fault-tolerant spanners, which provide a solution for ANSC by incorporating fault tolerance. 1-fault-tolerant k-spanner introduced as a subgraph that satisfies certain distance conditions. Specifically, for each pair of vertices u and v, and any edge e, and if d is distance between u, v in G\e (the graph G with edge e removed), then the distance dH(u, v) in the fault-tolerant spanner H satisfies d ≤ dH(u, v) ≤ kd.

2-approximation algorithm for ANSC:

```
In [2]: import random

        def approximate_ANSC(graph):
            num_vertices = len(graph)
            visited = set()
            stack = []

            def dfs(vertex):
                visited.add(vertex)
                stack.append(vertex)

                while stack:
                    current_vertex = stack.pop()

                    for neighbor in graph[current_vertex]:
                        if neighbor not in visited:
                            visited.add(neighbor)
                            stack.append(neighbor)

            # Step 2-5: Perform DFS traversal from randomly chosen unvisited vertices
            while len(visited) < num_vertices:
                unvisited_vertices = set(range(num_vertices)) - visited
                random_vertex = random.choice(list(unvisited_vertices))
                dfs(random_vertex)
```

```python
    # Step 6: Perform reverse DFS traversal to identify strongly connected components
    visited.clear()
    strongly_connected_components = 0

    while stack:
        current_vertex = stack.pop()

        if current_vertex not in visited:
            dfs(current_vertex)
            strongly_connected_components += 1

    # Step 7: Check the number of strongly connected components
    return strongly_connected_components == 1

# Example usage
graph = [
    [1, 2],     # Neighbors of vertex 0
    [0, 2, 3], # Neighbors of vertex 1
    [0, 1, 3], # Neighbors of vertex 2
    [1, 2, 4], # Neighbors of vertex 3
    [3]         # Neighbors of vertex 4
]

result = approximate_ANSC(graph)
if result:
    print("The graph is all-node strongly connected.")
else:
    print("The graph is not all-node strongly connected.")
```

```
The graph is not all-node strongly connected.
```

2-approximation Algorithm for ANSC

The aim then becomes to achieve better multiplicative approximation factor. To accomplish this, the authors utilize 1-fault-tolerant k-spanners for larger values of k, enabling approximation algorithms with running times close to linear. The Algorithm is employed for estimating small cycles of same size. For larger cycles, instead of depending solely on fault-tolerant spanners, composition of spanners suggested is employed in conjunction with the observations mentioned earlier in this section.

IV. Conclusion

In conclusion, this research paper delves into the study of approximability of two interconnected problems on graphs: n-PSP and ANSC. The paper provides an extensive investigation into the approximability of these problems, offering a diverse range of algorithms and conditional lower bounds which trade-off between approximation ratio and running time.

For n-PSP, the paper presents algorithms that achieve near-linear time complexity and approximation ratios of 2 + ε. These algorithms utilize methods from girth problem, spanners, distance oracles. Conditional lower bounds which are established for n-PSP demonstrate the inherent difficulty of achieving better approximation ratios under certain combinatorial hypotheses.

Regarding ANSC, the paper introduces algorithmic results for both directed and undirected graphs. The algorithms for directed graphs utilize concepts from the Girth problem and achieve approximation ratios of (k + ε) with efficient running times. For undirected graphs, the paper presents novel approaches that combine distance estimation techniques, link-cut trees, and fault-tolerant spanners. These algorithms achieve approximation ratios of (k + ε) and near-linear runtime complexity of O~(m + n^(2-ε)).

By combining these algorithmic results and lower bounds, this research contributes to the understanding of approximability of the both n-PSP and ANSC, shedding light on their fundamental graph properties. The paper paves the way for further advancements in solving these problems efficiently while considering the trade-offs between running time and approximation quality. The comprehensive analysis and diverse set of techniques presented in research provide the solid foundation for future research in field of graph algorithms.

V. Future Work

Based on the report's findings and conclusions, there are several potential avenues for future work and further research in the field of n-PSP and ANSC problems. Here are some possible next steps and actions that could be planned:

1. *Refining and optimizing approximation algorithms:* The report presents several approximation algorithms for n-PSP and ANSC with different runtime complexities and approximation ratios. Further work could focus on refining these algorithms to improve their approximation guarantees or reduce their running times. This could involve exploring new algorithmic techniques or adapting existing ones from related domains.

2. *Developing faster algorithms with better approximation ratios:* While the report provides algorithms with near-linear running times, achieving a better multiplicative approximation ratio remains an open question. Future work could aim to design algorithms with improved approximation guarantees while maintaining efficient running times.

3. *Investigating the impact of additional graph properties:* The report primarily focuses on unweighted undirected graphs, but extending the study to weighted graphs or directed graphs could yield valuable insights. Exploring the impact of additional graph properties, such as edge weights, graph connectivity, or specific graph structures, could lead to new algorithmic techniques or uncover further lower bounds.

4. *Examining practical applications and real-world scenarios:* While the report contributes theoretical advancements in the approximability of n-PSP and ANSC, further research could investigate their practical applications in real-world scenarios. Understanding the specific contexts in which these problems arise and analyzing the implications of the algorithmic results in practical settings could guide the development of efficient and effective solutions for real-world graph-related challenges.

5. *Investigating parallel and distributed algorithms:* Given the increasing importance of parallel and distributed computing, future work could explore design and analysis of parallel or distributed algorithms for ANSC and n-PSP. Leveraging parallel and distributed computing techniques could potentially lead to significant speedup and improved scalability, especially for large-scale graphs.

6. *Extending the analysis to other graph problems:* The report establishes connections between n-PSP, ANSC, and other graph problems such as distance oracles, girth, and cycle detection. Future work could further explore these connections and investigate the relationships between n-PSP, ANSC, and other fundamental graph problems.

By focusing on these future directions, researchers can continue to advance the understanding of n-PSP and ANSC problems, discover novel algorithmic techniques, and explore their practical applications, ultimately leading to more efficient and effective solutions in graph theory and related fields.

VI. References

[1] Agarwal, R. (2014). The Space-Stretch-Time Tradeoff in Distance Oracles. *European Symposium on Algorithms*, 11.

[2] Agarwal, U. (2018). Fine-grained complexity for sparse graphs. *ACM*, 14.

[3] Amir Abboud, K. B. (2022). Hardness of approximation in p via short cycle removal: cycle detection, distance oracles, and beyond. *ACM*, 13.

[4] Baruch Awerbuch, B. B. (1998). Near-Linear Time Construction of Sparse Neighborhood Covers. *SIAM*, 14.

[5] Bodwin, A. A. (2018). Reachability Preservers: New Extremal Bounds and Approximation Algorithms. *SIAM*, 18.

[6] Chechik, S., & Nechushtan, M. (2020). Simplifying and Unifying Replacement Paths Algorithms in Weighted Directed Graphs. *Dagstuhl Research Online Publication Server*, 29.

[7] Cohen, E. (1998). Fast Algorithms for Constructing t-Spanners and Paths with Stretch t. *SIAM*, 25.

[8] D. Aingworth, C. C. (1999). Fast Estimation of Diameter and Shortest Paths (Without Matrix Multiplication). *SIAM*, 28.

[9] Elkin, D. C. (2006). Sparse Sourcewise and Pairwise Distance Preservers. *SIAM*, 35.

[10] Liam Roditty, V. V. (2013). Fast approximation algorithms for the diameter and radius of sparse graphs. *ACM*, 9.

[11] Węgrzycki, P. S. (2018). Improved Distance Queries and Cycle Counting by Frobenius Normal Form. *Theory of Computing Systems* , 18.

[12] Yuster, R. (2011). A shortest cycle for each vertex of a graph. *ScienceDirect*, 5.