MASTER’S THESIS

NEW BOUNDS FOR APPROXIMATING

EXTREMAL DISTANCES IN

UNDIRECTED GRAPHS

Advisor:
Professor Roberto GROSSI

Candidate:
Massimo CAIRO

Academic Year 2014–2015
Abstract

In this thesis, we consider the problem of computing approximations of extremal distances in undirected graphs, namely the diameter, the radius and the eccentricity of every vertex. This is a basic problem in computational graph theory: a number of recent results attracted the interest of researchers. We present some of the algorithmic techniques recently discovered to address this problem and the conditional lower bounds on its complexity, with particular focus on the relation with combinatorial and graph-theoretic aspects.

This thesis includes some results obtained by the author in collaboration with professor Roberto Grossi (thesis advisor, University of Pisa) and professor Romeo Rizzi (University of Verona), including new algorithms, conditional impossibility results, and combinatorial insights that improve our understanding of the problem. Our original contributions resulted in a paper which will be presented at the conference ACM–SIAM SYMPOSIUM ON DISCRETE ALGORITHMS (SODA) in January 2016.
## Contents

**Introduction**  
4

1 Preliminaries  
1.1 Graphs basics .......................... 14
1.2 Extremal distances and approximations .......................... 16
1.3 Common tools .......................... 17

2 Lower bounds  
2.1 $k$-SAT and the SETH conjecture .......................... 20
2.2 Previous work .......................... 21
2.3 An improved construction .......................... 22
2.4 Lower bound on diameter and eccentricity .......................... 25

3 Approximation algorithms  
3.1 One-search approximations .......................... 31
3.2 Previous algorithms .......................... 33
3.3 New approximation bounds .......................... 37

4 Distance dominating sets  
4.1 Sampling $h$-dominating sets .......................... 49
4.2 Approximating the diameter .......................... 50
4.3 Lower bounds .......................... 52

Conclusions  
58

Bibliography  
59
Introduction

In this work we consider a basic problem in computational graph theory: given a graph, we want to efficiently provide an estimation of its extremal distances — the diameter, the radius, and the eccentricity of every vertex. We begin by introducing the concepts of graph and distance. Then we present the problem of extremal distances and review the previous work on the subject. Afterwards, we give an overview of our original contributions and present the structure of this thesis.

Graphs and algorithms. Graphs are mathematical structures consisting of a collection of vertices (or nodes) connected by edges (or arcs), which represent the topology of a network. Both vertices and edges can be augmented with additional information: for example, the label (or color) of a vertex, and the length (or weight) of an edge. Thanks to their flexibility, graphs are used extensively in Computer Science to model a great variety of different problems. To name only a few examples, they are used to represent the topology of computer and telecommunication networks, the flow of data and control in the analysis of computer programs, the relationship between individuals in a social network, and so on. Given the topology of a computer network, we may be interested in determining which computers can be reached via message forwarding (reachability or connectivity), which is the best way to do it (routing) and how many messages we can send at full capacity (maximum flow). In the analysis of control flow in computer programs, we want to know whether a statement is always executed before another (domination) as this allows for some optimization; in social networks, we can analyze the structure of the network to obtain various types of information about individuals and their relationship (graph mining). All these are examples of questions arising from applications which are naturally stated as problems on graphs.

Given the wide range of applications, it is crucial to design and implement efficient algorithms to solve problems on graphs. Indeed, instances of graphs arising from applications can be huge, reaching billions of nodes and edges, and thus require especially fast algorithms: as an emblematic example, consider the graph
representing the pages on the World Wide Web and their links. However, graphs are also extremely interesting from a purely mathematical point of view: graph theory is a flourishing field in mathematics and an active area of research, whose purpose is to discover combinatorial properties of graphs not necessarily related to algorithms. These two aspects — combinatorial and algorithmic — are deeply intermingled: understanding the combinatorial properties of graphs is at the basis of the development of efficient algorithms and, vice-versa, the search for efficient algorithms often leads to the discovery of new combinatorial properties. This characteristic makes the computational graph theory an extremely interesting field of research, which lays between Mathematics and Computer Science.

**Distances and shortest paths.** In many problems involving graphs we have a notion of *distance* between the nodes. Consider for example the road network of a city. We represent it as a graph, whose edges are (parts of) roads and vertices are intersection between roads. Also, each edge is associated with a value, indicating the length of the corresponding road. A natural question we may ask is how long we need to walk to go from one point in the network to another. Also, we may want to know which one is the best way to do it, that is, which one is the fastest route. When the starting and ending points are intersections (i.e. vertices), the fastest route is called a *shortest path*, and its length is the *distance* between the two vertices.

The simplest computation problem regarding distances is to output the value of the distance between two given vertices in a graph, and optionally a shortest path between the two. However, in many context we need to know the value of the distance (or a shortest path) between a single given vertex (the *source*) and all the other vertices in the graph. This version of the problem is called Single Source Shortest Paths (SSSP). Alternatively, we may ask for the distance (or shortest paths) between every pair of vertices. In this case, the problem is called All Pairs Shortest Paths (APSP).

The SSSP and APSP problems have been extensively studied in the past. In the special case where all the edges have length 1, the SSSP problem can be solved by a simple iterative procedure called Breadth First Search (BFS). We start by visiting the source and then, at each step, we visit all the nodes that can be reached with one edge from the previously visited nodes. The result is that at step $i$ we visit exactly those nodes at distance $i$ from the source, and when no other node is left we have computed all the desired distances. In the more general case of non-negative, integer or real edge lengths, an efficient algorithm has been conceived by Dijkstra, and published in 1959 in a note [12] which is one of the most cited papers in Computer Science. These two algorithms are a crucial tool for the design of most algorithm working with distances in graphs,
and they are very efficient: the BFS algorithm requires a running time which increases only linearly with the size of the graph (i.e. the number of edges and nodes), while the algorithm by Dijkstra, depending on the data structures used for its implementation, has a running time which is larger than linear by at most a logarithmic factor. Indicating by $n$ the number of nodes and by $m$ the number of edges, where we implicitly assume $m \geq n$, we say that the running time of both procedures is $\tilde{O}(m)$: that is, it is at most proportional to $m$, up to logarithmic factors as indicated by the $\tilde{O}$ notation.

Any algorithm designed for SSSP can be easily adapted to address APSP as well: it is sufficient to solve SSSP repeatedly, once for each node in the graph as source, so that the distance between every pair is computed. With this approach, we execute $n$ times an algorithm for SSSP, each time requiring $\tilde{O}(m)$ time, thus we obtain a total running time $\tilde{O}(nm)$. Several other solutions have been proposed addressing APSP directly: many of them improve the running time when the graph is dense, i.e. when $m \approx n^\delta$ for a large enough $\delta \in (1,2]$. We refer the reader to the literature [2, 4, 5, 14, 16, 22, 24, 25, 27, 30, 31]. Some of the proposed algorithms rely of fast matrix multiplication algorithms [8, 9, 26, 28], others exploit having integer weights and precompute the solution to smaller subproblems. To our knowledge, the only improvement in the case of sparse graphs, namely when $m = O(n)$, is an algorithm by Chan [5], which works on undirected, unweighted graphs and improves the $O(nm) = O(n^2)$ time by almost a logarithmic factor.

**Extremal distances.** In this work we focus on a specific problem related to distances. We are not interested in computing all the distances (either from a single given node, or in the whole graph), but only in computing extremal distances, namely, distances in the graph which attain an extremal — maximum or minimum — value.

We consider three definitions of extremal distances: the eccentricity, the diameter and the radius. The eccentricity of a node is defined as the maximum distance between that node and any other node, and it is a measure of how far away a node is with respect to the rest of the graph. The diameter is the maximum eccentricity in the graph, which is also the maximum value of the distance between any pair of nodes, while the radius is the minimum eccentricity. These measures have a specific meaning depending on which problem is being modeled. Consider a graph whose nodes are Web pages and edges are links between two pages: in this case, the diameter is the largest number of clicks necessary to go from any Web page to any other. Now consider a graph representing a road network of a city, where we want to place a facility (such as a fire station) so that the maximum distance from it is minimized. With the best possible choice of
location, the maximum distance from the facility to another node in the graph is the radius. Computing these extremal distances efficiently is an increasingly important problem, especially when dealing with the analysis of large networks.

The simplest way to compute the extremal distances is by solving the APSP problem. Indeed, by knowing the distance between every pair of nodes, one can easily find the extremal distances in a post-processing phase. This simple solution is often unpractical, as on large networks the $\tilde{O}(nm)$ cost of solving APSP becomes prohibitive. Also, even if a faster solution to APSP is found, with this approach we still need to pay at least a time proportional to $n^2$ to consider the distance between every pair of nodes. Solutions addressing directly the extremal distances have achieved polynomial speedup with respect to APSP in dense graphs only [29]. A recent work [5] presented an algorithm for the diameter which attains a logarithmic improvement over the $n^2$ cost in sparse graphs; however, to our knowledge no other significant improvement has been obtained on sparse graphs. Indeed, despite the diameter and the radius are just two numbers to output (and the eccentricities are only $n$ numbers), no algorithm is known to compute them in “truly” sub-quadratic time, that is, $O(n^{2-\varepsilon})$ time for some $\varepsilon > 0$, even when the graph is sparse. Some evidence has been recently provided showing that this bound could be inherent, as it is connected to other lower bounds in many, apparently unrelated, problems. For this reason, it seem unlikely that any algorithm exists to compute either the diameter, the radius, or the eccentricities of every node in $O(n^{2-\varepsilon})$ time.

In order to overcome this limitation, we can modify the original problem: one possible way is to restrict ourselves to specific classes of graphs, hence making further assumptions about the structure of the input. This approach leads to a vast area of research, due to the large number of graph classes whose properties could be exploited to design faster algorithm. However, this is outside of the scope of this work: for some examples of results in this direction, we refer the reader to the literature [10]. A different option is to give up on obtaining exact solutions, and look for approximations of the extremal distances: namely, measures close to the actual value of the diameter, the radius, or the eccentricity of a node.

**Approximation algorithms and lower bounds.** To obtain approximated extremal distances, we can start from the many approximated solutions for APSP achieving faster running times [7, 13]. However, this is of no help if we want to improve over the $\Omega(n^2)$ bound. A completely different approach allows us to obtain an approximation for the diameter and the radius given the solution of SSSP, for an arbitrary source. Specifically, it is known that the diameter is at most twice the radius, hence the eccentricity of an arbitrary node — which is computed implicitly by solving SSSP, and whose value by definition lies between
INTRODUCTION

the radius and the diameter — is an approximation of both the measures within a multiplicative factor 2 from the correct value.

In general, we evaluate the quality of an approximation by providing bounds to its error: specifically, we consider a multiplicative error factor \( \alpha \geq 1 \) and an additive error term \( \beta \geq 0 \). We give the precise meaning of the parameters \( \alpha \) and \( \beta \) in Chapter 1. The seminal work of Aingworth et al. [1] prepared the ground for the design of sub-quadratic time algorithms achieving multiplicative approximation factors \( \alpha \) strictly better than 2: in the last two years several results have been found in this direction and some of them are described in this work.

On the other hand, researchers have presented some evidence that it could be impossible to obtain approximations above a certain quality and below a certain running time: for example, it has been shown unlikely that any algorithm exists to approximate the diameter within a multiplicative factor \( \alpha < \frac{3}{2} \) in \( O(n^{2-\varepsilon}) \) time for some \( \varepsilon > 0 \). This and other lower bounds have been proved by assuming widely believed conjectures: it could be that they are not true, but this would prove the conjectures false yielding to many other unexpected consequences.

Overview of previous work. We briefly review some of the previous results on the problem of approximating the extremal distances. We consider algorithms for approximating either the diameter \( D \), the radius \( r \), or the eccentricities \( \epsilon(v) \) of every node \( v \) in the graph, with a multiplicative error \( \alpha \) and an additive error \( \beta \). Specifically, for the diameter approximation \( \tilde{D} \) it is required that \( \frac{1}{\alpha}D - \beta \leq \tilde{D} \leq D \), for the radius approximation \( \tilde{r} \) it is required that \( r \leq \tilde{r} \leq \alpha r + \beta \), and for the eccentricity approximation \( \tilde{\epsilon}_v \) for the node \( v \) it is required that \( \frac{1}{\alpha}\epsilon(v) - \beta \leq \tilde{\epsilon}_v \leq \epsilon(v) \). These requirements are recalled and explained in Section 1.2. To give a uniform treatment of the running time, we introduce a parameter \( \gamma \) and consider algorithms running in \( \tilde{O}(m^\gamma) \) expected time. Hence, for an algorithm running in \( \tilde{O}(m^a n^b) \) time, we will say that it has \( \gamma = a + b \): indeed, by assuming \( n \leq m \), we have \( \tilde{O}(m^a n^b) = \tilde{O}(m^a m^b) = \tilde{O}(m^{a+b}) \). We illustrate the previous results in terms of \( \alpha \), \( \beta \) and \( \gamma \).

The values of \( \alpha \) and \( \gamma \) for which an almost \( \alpha \)-approximation algorithm is known are reported as bullets in the upper envelopes of Fig. 1a,b. As discussed above, no algorithm is known that computes either the diameter, the radius or all the eccentricities in \( \tilde{O}(m^{2-\varepsilon}) \) time (which, for sparse graph, is also \( \tilde{O}(n^{2-\varepsilon}) \) time); on the other hand, we can solve APSP in \( \tilde{O}(mn) = \tilde{O}(m^2) \) time, and therefore compute all these three types of extremal distances with this running time. Hence, we have an algorithm with \( \alpha = 1 \), \( \beta = 0 \) and \( \gamma = 2 \) (bullets on the left in Fig. 1a,b). Also, as we will show later in detail, a single graph search yields \( \alpha = 2 \) for the diameter and the radius and \( \alpha = 3 \) for the eccentricities, without additive error, in \( \tilde{O}(m) \) time. Hence we have algorithms with \( \alpha = 2 \) for
the diameter and the radius, and $\alpha = 3$ for the eccentricities, all with $\beta = 0$ and $\gamma = 1$ (bullets on the right in Fig. 1a,b). Roditty and Vassilevska Williams [23] obtain $\alpha = \frac{3}{2}$, $\beta < 1$ and $\gamma = \frac{3}{2}$ for the diameter and the radius: they employ Las Vegas randomization to reduce the $\tilde{O}(m\sqrt{n} + n^2)$ time achieved in the seminal work by Aingworth, Chekuri, Indyk, and Motwani [1] to expected $\tilde{O}(m\sqrt{n})$ time (middle bullet in Fig. 1a). Chechik, Larkin, Roditty, Schoenebeck, Tarjan, and Williams [6] present new deterministic algorithms, addressing also eccentricities, and obtain $\alpha = 3/2$ for the diameter and the radius and $\alpha = 5/3$ for the eccentricities, all with $\beta = 0$ and $\gamma = 3/2$ (middle bullets in Fig. 1a,b).

Some lower bounds are known for the diameter. They are conditional under the Strong Exponential Time Hypothesis (SETH) of Impagliazzo, Paturi, and Zane [20] stating that for every $\varepsilon > 0$ there is an integer $k$ such that $k$-SAT cannot be solved in time $O(2^{(1-\varepsilon)n})$: any algorithm with $\alpha$, $\beta$ and $\gamma$ below these
hence it is impossible to get $\alpha < 3/2$ (shaded zone in Fig. 1a). With their construction, the possibility of $\alpha < 3/2$ with $\gamma < 2$ remains open if one allows for $\beta \geq 1$. Chechik et al. [6] reduce SAT to the problem of distinguishing between graphs of diameter $3(\ell + 1)$ and $4(\ell + 1)$, where $\ell \geq 0$ is a given parameter, showing that it is impossible to get $\alpha < 4/3$, $\beta = O(m^\delta)$, $\gamma < 2 - 2\delta$ for any $\delta \geq 0$ under SETH (darker shaded zone in Fig. 1a). However, as noted by the authors, this construction still leaves open the possibility of $4/3 \leq \alpha < 3/2$ with $\gamma < 2$ and $\beta \geq 1$ (lighter shaded zone in Fig. 1a).

Original contributions. In this work, we make three further steps in the study of the problem of approximating the extremal distances mentioned before. In particular, we obtain new bounds as illustrated in Fig. 2.

First, we describe a reduction from SAT to two problems on graphs: deciding whether the diameter is $3t$ or $2t$, and deciding whether the maximum eccentricity over a given subset of nodes $X \subseteq V$ is $5t$ or $3t$. Under SETH, both of them for $t > c \cdot m^\delta$ cannot be solved in $O(m^{2 - 2\delta - \epsilon})$ time for any $\epsilon > 0$, $c > 0$ and $0 \leq \delta < 1$: hence it is impossible to get $\alpha < 3/2$ for the diameter and $\alpha < 5/3$ for the eccentricities; with $\beta = O(m^\delta)$ and $\gamma < 2 - 2\delta$ for any $\delta \geq 0$ under SETH. In this way we tighten the known bounds for the diameter and the eccentricities (shaded zones in Fig. 2a,b). In particular, the recently achieved approximation factors $\alpha = 3/2$ for $D$ and $\alpha = 5/3$ for $\epsilon(v)$ cannot be improved in truly subquadratic time ($\gamma < 2$) even with an additive term $\beta = O(1)$ under SETH. This also indicates that the eccentricities are more difficult to approximate than the diameter.

Second, we present an algorithmic scheme that gives nontrivial approximations with exponent $\gamma = 1 + \epsilon$ arbitrarily close to the minimum 1. Specifically, for any integer $k \geq 0$, we present a randomized algorithm hinging on a novel iterative procedure that selects $O(n^{1/(k+1)})$ nodes from which to launch the graph searches. The properties of the selected nodes allow us to get $\alpha = 2 - 1/2^k$ for diameter and radius and $\alpha = 3 - 4/(2^k + 1)$ for eccentricities, with $\beta < 1$, in $\tilde{O}(mn^{1/2 + \epsilon})$ expected time (thus $\gamma = 1 + \frac{1}{k+1}$). For $k \geq 2$, this gives a family of previously unknown bounds (crosses in Fig. 2a, b), and approaches near-linear time as $k$ grows. For example, when $k = 2$ we obtain $\alpha = 7/4$ for diameter and radius and $\alpha = 11/5$ for the eccentricities, in $\tilde{O}(mn^{1/3})$ expected time. Looking at Fig. 1, we can observe that previous work focused on bounds for $\alpha$ in $[1, 3/2]$ (diameter and radius) and $[1, 5/3]$ (eccentricities); instead, our results explore for the first time the ranges $(3/2, 2]$ and $(5/3, 3]$, which are equally valuable for investigation. Also, approaching $\gamma \approx 1$ for larger values of $\alpha$ is crucial to analyze
massive networks. We obtain as special cases the bounds of the exact algorithms, for $k = 0$, and of Roditty and Vassilevska Williams [23] and Chechik et al. [6] on undirected graphs, for $k = 1$, hence giving a unified view comprising these algorithms.

Third, we study the size of distance $h$-dominating sets in undirected graphs of diameter $D$. An $h$-dominating set $X$ is a subset of the nodes at distance $\leq h$ from every other node in the graph. Our study is closely related to the possibility of fast diameter approximations, but could be of independent interest for other graph problems (e.g. facility location). We first show that an $h$-dominating set $X$ can be used to produce an upper bound $\overline{D}$ of the diameter $D$ in $\tilde{O}(|X| \cdot m)$ time, such that $D \leq \overline{D} \leq D + h$. Upper bounds are crucial for approximations, as any approximation algorithm with bounded error must (implicitly or explicitly) provide one: in particular an $\alpha$-multiplicative $\beta$-additive approximation $\tilde{D}$.
is equivalent to an upper bound $\mathcal{D}$ satisfying $D \leq \mathcal{D} \leq \alpha(D + \beta)$ by choosing $\mathcal{D} = \alpha(\hat{D} + \beta)$. A closer look at the algorithms in [1, 23], when applied to undirected graphs, shows that they implicitly rely on the existence of an $h$-dominating set of size $\tilde{O}(\sqrt{n})$ for $h = (D + 1)/2$. In our algorithmic scheme we explicitly find $h$-dominating sets of size $\tilde{O}(n^{1/(k+1)})$ for $h = \frac{2^{k-1}}{2k}(D + 1)$. To obtain faster approximation algorithms, one way is to find $h$-dominating sets of smaller size: thus, it is interesting to study the size of $X$ in the worst case, throwing a bridge between diameter approximation and extremal graph theory [3, 18]. We analyze the upper bounds given by our algorithms and show several lower bounds through explicit constructions. In particular, we consider the worst-case size of $h$-dominating sets for $h = \lambda(D + \delta)$, for some $0 \leq \lambda < 1$ and $\delta \geq 0$, because this is related to diameter approximations with $\alpha = \lambda + 1$ and $\beta = \lambda\delta/\alpha$. If there is
an $h$-dominating set $X$ of size $\Theta(n^\epsilon)$, then it is possible to use it to get $\gamma = 1 + \epsilon$. Our results are illustrated in Fig. 3. First, we rule out the possibility to get $\epsilon < 1$ for $\lambda < 1/2$ and $\delta = O(1)$. Second, we use a family of constructions $G^\ell_t$ to exclude $\epsilon < 1/\ell$ for $\lambda < 1 - \frac{2}{(\ell+1)+2}$ and $\delta \leq 1$, for any chosen $\ell \geq 2$. For some values of $\lambda$ the bounds obtained by our algorithms are optimal (up to logarithmic factors). For these values, a faster algorithm for diameter approximation with $\alpha = \lambda + 1$ would need new techniques to bound the diameter from above as it cannot rely on the existence of small $h$-dominating sets.

**Structure of this document.** We begin in Chapter 1 with some preliminaries: we define rigorously the concepts used throughout this work and introduce the tools at the basis of the techniques used to address the problem. In Chapter 2, we describe the previous results of impossibility, and develop a refined construction which gives more general conditional lower bounds. In Chapter 3, we present the previous algorithmic techniques to approximate the extremal distances, which are then generalized (only in the case of undirected graphs) to give our family of algorithms parametrized by $k$. In Chapter 4, we explore the connection between the approximation of the diameter and distance dominating sets, providing a general approach to approximate the diameter using distance dominating set. Then, we present explicit constructions of graphs which show that the smallest distance $h$-dominating set can be large (for some values of $h$ with respect to the diameter $D$) and give an inherent limit to the efficiency of this general approach.
Chapter 1

Preliminaries

In this chapter we introduce the main concepts used in this work.

1.1 Graphs basics

A graph $G = (V,E)$ is a structure $G$ consisting of a set of vertices $V$ and a set of edges $E \subseteq V \times V$. Each edge $e = (u,v) \in E$ connects the two vertices $u,v \in V$, called endpoints. In general, edges are ordered pairs and we say that the graph is directed. The graph is said undirected if for every $(u,v) \in E$ also $(v,u) \in E$, thus the order of the edge endpoints is not relevant. We consider finite graphs $G = (V,E)$, and denote the number of nodes by $n = |V|$ and the number of edges by $m = |E|$.

A walk $P$ on $G$ between two vertices $a, b \in V$ (also called endpoints) is a finite sequence of $\ell$ consecutive edges $e_1, \ldots, e_\ell$, where $e_i = (u_{i-1}, u_i)$ with $u_0 = a$ and $u_\ell = b$. It is indicated by

$$ P : \ u_0 \xrightarrow{e_1} u_1 \xrightarrow{e_2} \ldots \xrightarrow{e_\ell} u_\ell. $$

Each edge $(u,v) \in E$ is associated with a real non-negative weight $w(u,v) \geq 0$. In the unweighted case, we set $w(u,v) = 1$ for every edge $(u,v)$. In the weighted case, the value of $w(u,v)$ is arbitrary and provided along with the graph. The maximum edge weight is denoted by $M = \max_{(u,v) \in E} w(u,v)$. ($M = 1$ on unweighted graphs.) The length of a walk is sum of the weights of its edges. The distance $d(u,v)$ between two vertices $u, v \in V$ is the minimum length among the walks between $u$ and $v$. If there is no such walk, then $d(u,v) = \infty$. Notice that $d(u,u) = 0$ for every vertex $u \in V$, as there exists always an empty walk between $u$ and itself containing no edges. A walk $P : a \rightarrow \cdots \rightarrow b$, whose length is exactly the distance between its two endpoints $a$ and $b$, is called a shortest path and it is
indicated by 

\[ P : \quad a \sim b. \]

One of the main properties of the distance function is the triangular inequality, used extensively through this work.

**Lemma** (Triangular inequality). For any three vertices \( a, b, c \in V \), we have

\[ d(a, c) \leq d(a, b) + d(b, c). \]

**Proof.** If either \( d(a, b) = \infty \) or \( d(b, c) = \infty \) the statement is obvious. Otherwise, concatenate two shortest paths \( a \sim b \) and \( b \sim c \) obtaining a walk

\[ a \sim b \sim c \]

between \( a \) and \( c \) of length \( d(a, b) + d(b, c) \). As the value \( d(a, c) \) is the minimum length of such walks, we obtain the statement. \( \square \)

In undirected graphs, every walk from \( u \) to \( v \) can be reversed: there exists a walk from \( v \) to \( u \) containing the same edges, with swapped endpoints and in the reverse order. The reverse walk has the same length, hence in undirected graphs the distance function is *symmetric*:

\[ d(u, v) = d(v, u). \]

To simplify the notation, in the case of undirected graphs we extend the definition of \( d(\cdot, \cdot) \) to express the distance between a set of nodes and another node. Specifically, for a non-empty set \( S \subseteq V \) and a node \( v \in V \) we define

\[ d(S, v) = \min_{u \in S} d(u, v). \]

Notice that we can always take a node \( x \in S \) such that \( d(x, v) = d(S, v) \). The triangular inequality can be extended as follows.

**Lemma** (Triangular inequality with sets). For any node set \( S \subseteq V \) and nodes \( a, b \in V \), we have

\[ d(S, b) \leq d(S, a) + d(a, b). \]

**Proof.** Take \( x \in S \) such that \( d(x, a) = d(S, a) \). Observe that \( d(S, b) \leq d(x, b) \) by definition of distance from a set; hence

\[
\begin{align*}
    d(S, b) & \leq d(x, b) \\
    & \leq d(x, a) + d(a, b) \\
    &= d(S, a) + d(a, b)
\end{align*}
\]

where we applied the triangular inequality to show \( d(x, b) \leq d(x, a) + d(a, b) \). \( \square \)
1.2 Extremal distances and approximations

We now define the notions of eccentricity, diameter and radius on undirected graphs.

**Definition 1.** In an undirected graph $G = (V, E)$, the eccentricity $\epsilon(v)$ of a vertex $v \in V$ is the maximum distance between $v$ and any other node:

$$\epsilon(v) = \max_{u \in V} d(v, u).$$

**Definition 2.** The diameter $D(G)$ of an undirected graph $G = (V, E)$ is the maximum eccentricity:

$$D(G) = \max_{v \in V} \epsilon(v).$$

**Definition 3.** The radius $r(G)$ of $G$ is the minimum eccentricity:

$$r(G) = \min_{v \in V} \epsilon(v).$$

Notice that the diameter of an undirected graph is also the maximum distance between every pair of nodes, indeed:

$$D(G) = \max_{v \in V} \epsilon(v) = \max_{v \in V} \max_{u \in V} d(v, u) = \max_{u,v \in V} d(v, u).$$

Hence, it is possible to give an equivalent definition, which naturally extends also to directed graphs.

**Definition 4.** The diameter $D(G)$ of a directed or undirected graph $G = (V, E)$ is the maximum distance between every pair of nodes:

$$D(G) = \max_{u,v \in V} d(v, u).$$

The notions of eccentricity and radius extend to directed graphs in several, different ways. Also, there are other notions of diameter in the case of directed graphs. These notions are not taken into account in this work.

The following property of the eccentricity function $\epsilon(\cdot)$ is at the basis of most the approximation algorithms presented in this work.

**Lemma 1.** For any pair of nodes $u, v \in V$, we have $|\epsilon(u) - \epsilon(v)| \leq d(u, v)$.

**Proof.** Assume without loss of generality that $\epsilon(u) \geq \epsilon(v)$: we need to prove that $\epsilon(u) - \epsilon(v) \leq d(u, v)$.
or, equivalently, that

$$\epsilon(u) \leq d(u, v) + \epsilon(v).$$

Take $x \in V$ so that $\epsilon(u) = d(u, x)$. We have

$$d(u, x) \leq d(u, v) + d(v, x)$$

by triangular inequality and $d(v, x) \leq \epsilon(v)$ by definition of eccentricity. Hence:

$$\epsilon(u) = d(u, x) \leq d(u, v) + d(v, x) \leq d(u, v) + \epsilon(v)$$

as desired. \qed

We define rigorously the notion of *approximations* for the the diameter, the radius, and the eccentricity.

**Definition 5.** The values $\tilde{D}$, $\tilde{r}$ and $\tilde{\epsilon}_v$ are $\alpha$-multiplicative $\beta$-additive approximations of, respectively, the diameter $D$, the radius $r$, and the eccentricity $\epsilon(v)$ of a node $v \in V$ if

$$\frac{1}{\alpha}D - \beta \leq \tilde{D} \leq D$$

$$r \leq \tilde{r} \leq \alpha r + \beta$$

$$\frac{1}{\alpha}\epsilon(v) - \beta \leq \tilde{\epsilon}_v \leq \epsilon(v).$$

These approximation are one-sided, in the sense that they can be off from the correct value only on one side. Specifically, we require $\tilde{D} \leq D$ because the approximation of the diameter is usually obtained by finding two witness nodes $a$ and $b$ at distance $\tilde{D}$ from each other. We require $\tilde{\epsilon}_v \leq \epsilon(v)$ for the eccentricity, for a similar reason. Conversely, in the case of the radius the approximation is usually obtained by exhibiting a node with eccentricity $\tilde{r}$, hence we require $\tilde{r} \geq r$ instead.

We say that a value is an *almost* $\alpha$-approximation if it is and $\alpha$-multiplicative $\beta$-additive approximation for a constant $\beta > 0$.

### 1.3 Common tools

In this work we present both *deterministic* and *randomized* algorithms, working on an input graph $G$. For an introduction to randomized algorithms, we refer the reader to the literature [21]. We analyze the (expected) running time of the algorithms as a function of the number of nodes $n$ and the number of edges $m$.
of $G$. To express the asymptotic running time, we use the notations $O$, $o$, $\Omega$ and $\Theta$ whose meaning is assumed to be known to the reader. Sometimes we also use the notations $\tilde{O}$, $\tilde{\Omega}$ and $\tilde{\Theta}$, which hide poly-logarithmic factors.

We now introduce some tools used repeatedly in the rest of this work.

**Graph search.** In the algorithms described in this work we often solve the SSSP problem as a sub-routine. There are several different solutions for SSSP, but the most commonly used are the Breadth First Search (BFS) for unweighted graphs and the Dijkstra algorithm for weighted graphs [12]. Both these procedures work by visiting the nodes in order of distance from the source; the BFS algorithm requires $O(m)$ time, while the Dijkstra algorithm implemented with Fibonacci heaps [15] requires $O(m + n \log n)$ time.

To work uniformly on weighted and unweighted graphs, we use the term graph search to indicate the invocation of any of these algorithms to solve the SSSP problem from a specified source node. We denote by $C = C(n, m)$ the cost of this search, assuming $C = \tilde{O}(m)$ and $C = \Omega(m)$. In the case of directed graphs, we distinguish between forward and backward graph search: the former computed the distances from the source node, the latter computes the distances to the source node.

For a node $v \in V$ we denote by $N_\ell(v)$ the set containing the first $\ell$ visited nodes in a graph search from $v$; in other words, $N_\ell(v)$ contains the nearest $\ell$ nodes to $v$, breaking ties arbitrarily as done by the execution of a graph search.

**Multiple-source graph search.** Sometimes we are interested in computing the distance $d(S, v)$ from a node set $S \subseteq V$ to every node $v \in V$. Any algorithm for the SSSP problem (graph search) can be adapted to solve this task in $O(C)$ time: it suffices to modify the original graph, adding a dummy node $\pi$ connected to every node in $S$ by a directed edge of length 1, and then launching a graph search from $\pi$. The distance from $\pi$ to $v \in V$ in the modified graph is exactly $d(S, v) + 1$, and from this value we easily compute $d(S, v)$. We call the application of this technique a multiple-source graph search from the set $S$.

**Hitting set.** We need the following known result on uniform random sampling, also used in [1, 23]. In the rest of this work we use the expression with high probability in the sense of Lemma 2.

**Lemma 2.** Given a family $\mathcal{H}$ of at most $n$ sets each of size $\ell$, over a universe $U$ of size $L$, a random sampling of

$$\Theta\left(\frac{L}{\ell} \cdot \log n\right)$$
elements hits all the sets in $\mathcal{H}$ with high probability. That is, if $S$ is a set containing at least $\alpha \cdot \frac{L}{\ell} \log n$ elements sampled independently and uniformly at random, for a large enough $\alpha$, the probability that there is an $H \in \mathcal{H}$ disjoint from $S$ is at most $1/n^{\Omega(\alpha)}$.

Proof. For any $H \in \mathcal{H}$, the probability that an element $u \in U$ sampled uniformly at random is not in $H$ is $\frac{L - \ell}{L}$. If $S \subseteq U$ contains $s$ elements sampled independently and uniformly at random, then

$$\Pr[S \cap H = \emptyset] = \left(\frac{L - \ell}{L}\right)^s.$$

Let $p$ be the probability that for some $H \in \mathcal{H}$ we have $S \cap H = \emptyset$. By the union bound:

$$p \leq |\mathcal{H}| \cdot \left(\frac{L - \ell}{L}\right)^s = n \cdot \left(1 - \frac{\ell}{L}\right)^{s\ell/L} = O(n \cdot e^{-s\ell/L}).$$

If $s \geq \alpha \cdot \frac{L}{\ell} \cdot \log n$ for some constant $\alpha > 1$, then

$$p = O(n \cdot e^{-\alpha \log n}) = 1/n^{\Omega(\alpha)}.$$

$\square$
Chapter 2

Lower bounds

In this chapter we present the conditional results of impossibility for the approximation of the diameter and the eccentricities. We first introduce the SETH conjecture, which is assumed to be true in order to prove these negative results. Then, we briefly describe the previous results and proceed to illustrate our generalized and improved construction which yields to our lower bound.

2.1 $k$-SAT and the SETH conjecture

$k$-SAT is a standard problem defined as follows. We are given in input a boolean formula $\varphi$ over $n$ boolean variables $x_1, \ldots, x_n$, with the following structure. The formula $\varphi$ is a conjunction of $m$ clauses $\gamma_1, \ldots, \gamma_m$:

$$\varphi \equiv \gamma_1 \land \gamma_2 \land \cdots \land \gamma_m.$$  

Each clause $\gamma_i$ is in turn a disjunction of $k$ literals:

$$\gamma_i \equiv x_{i1}^1 \lor x_{i2}^2 \lor \cdots \lor x_{ik}^k.$$  

where each literal $x_{ij}^k$ is either a variable $x_j$ or its negation $\overline{x_j}$. We assume that the clauses $\gamma_1, \ldots, \gamma_m$ contain no duplicates. For example, consider the following formula, which is an instance of 3-SAT with $n = 4$ and $m = 5$:

$$\varphi \equiv (x_1 \lor x_2 \lor \overline{x_4}) \land (x_1 \lor \overline{x_3} \lor x_4) \land (\overline{x_1} \lor x_2 \lor x_3) \land (\overline{x_1} \lor \overline{x_3} \lor x_4).$$  

The $k$-SAT problems asks whether there exists a boolean assignment

$$p : \{x_1, \ldots, x_n\} \rightarrow \{\text{true, false}\}$$  

which satisfies the formula $\varphi$. Namely, we look for an assignment $p$ that makes the formula $\varphi$ evaluate to true by substituting each variable $x_j$ in $\varphi$ with the
boolean value \( p(x_j) \) assigned to it by \( p \). If such a \( p \) exists, then the answer to the problem is “yes”, otherwise it is “no”.

No algorithm is known for \( k \)-SAT with a running time polynomial in the number of variables \( n \). Indeed, the \( k \)-SAT problem is in the class of \( NP \)-complete problems: any polynomial solution for this problem implies that the two complexity classes \( P \) and \( NP \) are equal and therefore the existence of any polynomial algorithm for \( k \)-SAT is considered unlikely. Actually, there is an even stronger conjecture by Impagliazzo, Paturi, and Zane [20] about the complexity of \( k \)-SAT.

**Conjecture 1** (Strong Exponential Time Hypothesis — SETH). For every \( \gamma > 0 \), there exists an integer \( k \) such that there is no algorithm for \( k \)-SAT running in \( O(2^{(1-\gamma)n}) \) time.

The conjecture SETH has been used to prove (conditional) negative results for many problems, among which the approximation of extremal distances.

### 2.2 Previous work

Roditty and Vassilevska Williams [23] proved under SETH that it is impossible to approximate the diameter with an approximation factor \( \alpha < 3/2 \), without any additive term, in \( O(m^{2-\varepsilon}) \) time for some \( \varepsilon > 0 \). Indeed, any such approximation algorithm would be able to distinguish between graphs of diameter 2 and graphs of diameter 3, and they show that the latter problem requires \( m^{2-o(1)} \) time under SETH. To obtain this lower bound, they transform any instance \( \varphi \) of \( k \)-SAT into a graph \( G_{\varphi} \) which has diameter 2 or 3, the latter if and only if \( \varphi \) is satisfiable. This transformation is necessarily exponential, otherwise the polynomial solutions to the diameter problem would imply a polynomial algorithm for \( k \)-SAT. Indeed, the transformation works by splitting the \( n \) variables into two halves, and then enumerating all the (exponentially many) boolean assignments of each half of the variables. The graph \( G_{\varphi} \) is built in such a way that for each partial assignment \( p \) there is a node \( w_p \), and we have \( d(w_p, w_q) = 3 \) if and only if \( p \) and \( q \) can be combined to obtain a total assignment satisfying \( \varphi \). As the graph has size proportional to \( 2^{n/2} \), a truly sub-quadratic algorithm telling whether \( G_{\varphi} \) has diameter 2 or 3 has a running time proportional to \( (2^{n/2})^{2-\varepsilon} = 2^{(1-\varepsilon/2)n} \) and falsifies the conjecture SETH.

Chechik et al. [6] proposed a related construction that generates a graph of diameter either \( 3t \) or \( 4t \), according to the satisfiability of the \( k \)-SAT formula \( \varphi \), where \( t \geq 1 \) is an arbitrary parameter. Thanks to the parameter \( t \), they are
able to prove a lower bound even when there is an additive term $\beta \geq 1$. Indeed, with this construction, they prove that it is impossible under SETH to get for the diameter an approximation factor $\alpha < 4/3$ with additive term $\beta = O(m^{\delta})$ in $O(m^{2-2\delta-\epsilon})$ running time, for any $\delta \geq 0$ and $\epsilon > 0$. This is the first lower bound involving additive approximation; however, it is not fully satisfying, as it still allows for a multiplicative factor $4/3 < \alpha \leq 3/2$ with a constant additive term $\beta \geq 1$ in truly sub-quadratic time.

2.3 An improved construction

In this section we show how to improve the previous lower bounds in order to exclude any approximation with $\alpha < 3/2$ in sub-quadratic time, also if there is a constant additive term $\beta \geq 1$. To do this, we transform an instance $\varphi$ of $k$-SAT into an undirected, unweighted graph $G_\varphi$ (of exponential size) of diameter $2t$ or $3t$, where the latter occurs if and only if $\varphi$ is satisfiable. Our transformation is based on the construction of Roditty and Vassilevska Williams [23], which produces graphs of diameter 2 and 3.

The gadget graph $T^t[B]$. The first step in our construction is to define a gadget graph $T^t[B]$. For a given node set $B$ and $t \geq 1$, the gadget $T^t[B]$ is a graph $T = (V, E)$ on node set $V = B \cup Q$, where $Q$ is a set of additional private nodes. The purpose of this graph is to make the nodes in $B$ at distance exactly $t$ from each other. For $t = 1$, the gadget $T^1[B]$ is the complete graph on $B$, with no private node. Namely, $V = B$ and $E = B^2$. For $t = 2$, we introduce a private node $c \in Q$ and define $T^2[B]$ as the star with center in $c$ and tips in $B$. Namely, $V = B \cup \{c\}$ and

$$E = \{(u, c), (c, u) : u \in B\}.$$ 

To construct $T^{t+2}[B]$, we first introduce a distinct private node $u' \in Q$ for every $u \in B$. Then, we build the gadget

$$T^t[B'] = (B' \cup Q', E')$$

on the node set

$$B' = \{u' : u \in B\}$$

and add an edge $(u', u)$ for every $u \in B$. Specifically, we define

$$Q = B' \cup Q'$$

and

$$E = E' \cup \{(u', u) : u \in B\}.$$
The result is a star-like structure, illustrated in Fig. 2.1 where private nodes are colored black.

\[ u \in B, \quad u' \in Q \]

\[ u^{(z-1)} \]

\[ c \]

\[ u \in B, \quad u' \in Q \]

\[ u^{(z)} \]

Figure 2.1: Gadget \( T_t^1[B] \) for \(|B| = 5\) and \( t = 2z\) (left) or \( t = 2z + 1\) (right).

**Lemma 3.** The following properties hold:

- \( T_t^1[B] \) contains \( O(t \cdot |B|) \) nodes and \( O(t \cdot |B| + |B|^2) \) edges.
- For any \( a, b \in V \) we have \( d(a, b) \leq t \) with equality if and only if \( a, b \in B \) and \( a \neq b \).
- In \( T_t^{2z}[B] \), for any \( a \in V \) we have \( d(c, a) \leq z \) with equality if and only if \( u \in B \). The only shortest path between \( u, v \in B \) contains \( c \).

**Proof.** The statements are trivially true for \( T_t^1[B] \) and \( T_t^2[B] \). In the general case, they can be shown by induction noticing that any shortest path in \( T_t^{t+2}[B] \) is a shortest path in \( T_t^t[B'] \) possibly concatenated with an edge \((u, u')\) at each endpoint \( u \), if \( u \in B \).

**Construction of \( G^{\varphi}_{t} \).** Let \( \varphi \) be an instance of \( k\)-SAT on an even number \( d \) of variables \( D = \{x_1, \ldots, x_d\} \) and \( c \) clauses \( C = \{\gamma_1, \ldots, \gamma_c\} \).

We divide the variables in two sets \( D_1 \) and \( D_2 \), each of size \( d/2 \), and construct the sets \( P_1 \) and \( P_2 \), containing the \( 2^{d/2} \) partial assignments on the variables in \( D_1 \) and \( D_2 \), respectively. We say that a partial assignment \( p \in P_i \) satisfies a clause \( \gamma \in C \) if \( \gamma \) contains at least one literal evaluating to \( true \) under \( p \). We extend the set of clauses \( C \) to

\[ C^* = C \cup \{\delta_1, \delta_2\}, \]

where \( \delta_i \) is a dummy clause satisfied by the partial assignments \( p \in P_i \) only.\(^1\)

Then, we build the set

\[ P^* = P_1 \cup P_2 \cup \{\pi\}, \]

\(^1\)For example \( \delta_i = x \lor \pi \), where \( x \) is any variable in the group \( D_i \).
where $\pi$ is an empty partial assignment that does not satisfy any clause.

The graph $G_{\phi}^t = (V, E)$ is defined as follows. We start with a node set $U$ containing a node $u_\gamma$ for every clause $\gamma \in C^*$. Then, for each partial assignment $p \in P^*$ we introduce a new node $w_p$ and define the gadget

$$T_p = (V_p, E_p) = T^t[\{w_p\} \cup U_p],$$

where

$$U_p = \{u_\gamma \in U : \gamma \text{ is not satisfied by } p\}.$$ 

The graph $G_{\phi}^t$ is the union of the gadgets $T_p$: specifically,

$$V = \bigcup_{p \in P^*} V_p \quad \text{and} \quad E = \bigcup_{p \in P^*} E_p,$$

with $U \subseteq V$ as $V_\pi \supset U_\pi = U$ and $V_p \cap V_q \subseteq U$ for $p \neq q$ as private nodes of different gadgets are distinct.

**Lemma 4.** $G_{\phi}^t$ contains $t \cdot 2^{d/2+o(d)}$ edges and can be constructed in $t \cdot 2^{d/2+o(d)}$ time.

**Proof.** Observe that

$$|P^*| = 2 \cdot 2^{d/2} + 1 = 2^{d/2+o(d)}$$

and

$$|C^*| = O((2d)^k) = 2^{O(k \cdot \log d)} = 2^{o(d)}$$

as we do not have duplicated clauses and $k$ is constant. Computing $U_p$ for every $p \in P^*$ requires

$$O(|P^*| \cdot |C^*| \cdot k) = 2^{d/2+o(d)}$$

time. Each of the $|P^*|$ gadgets has

$$O(t \cdot |C^*| + |C^*|^2) = t \cdot 2^{o(d)}$$

edges by Lemma 3 and can be constructed trivially knowing $U_p$. Thus, the total time and size is at most $t \cdot 2^{d/2+o(d)}$.

**Lemma 5.** Any path in $G_{\phi}^t$ that does not contain a node in $U$ between its endpoints is also a path in $T_p$ for some $p \in P^*$ and in particular both its endpoints belong to $V_p$.

**Proof.** Otherwise, take two consecutive edges

$$x \rightarrow v \rightarrow y$$

along the path which belong to distinct gadgets $T_p$ and $T_q$. They must be incident to a node $v \in V_p \cap V_q \subseteq U$. 

\[\blacksquare\]
Lemma 6. For any two distinct non-private nodes
\[ a, b \in U \cup \{w_p : p \in P^*\}, \]
we have
\[ d(a, b) \geq t. \]

Proof. Take a shortest path \( a \leadsto b \) and apply Lemma 5. If it is a path on \( T_p \),
then \( d(a, b) = t \) by Lemma 3. Otherwise, it is of the form
\[ a \leadsto v \leadsto b \]
with \( v \in U \) distinct from \( a \) and \( b \), hence
\[ d(a, b) = d(a, v) + d(v, b) \geq 2t \]
by induction. \( \square \)

### 2.4 Lower bound on diameter and eccentricity

We described our construction \( G^\varphi_t \) for a given instance \( \varphi \) of \( k \)-SAT and we proved some of its properties. We now show how this construction gives us to obtain the claimed result of impossibility under SETH.

#### Diameter.

We start by analyzing the diameter of our construction \( G^\varphi_t \).

Lemma 7. \( G^\varphi_t \) has diameter either \( 2t \) or \( 3t \). It is \( 3t \) if and only if \( \varphi \) is satisfiable.

Proof. We first show that \( \varphi \) is satisfiable if and only if there exist two disjoint sets \( U_p \) and \( U_q \) for some \( p, q \in P^* \). Observe that the sets \( U_p \) and \( U_q \) are disjoint if and only if every clause in \( C^* = C \cup \{\delta_1, \delta_2\} \) is satisfied by either \( p \) or \( q \). The dummy clauses \( \delta_1 \) and \( \delta_2 \) are both satisfied if and only if \( p \) and \( q \) belong respectively to \( P_1 \) and \( P_2 \), and thus form a valid total assignment \( p \cup q \). Moreover, \( p \cup q \) satisfies \( \varphi \) if and only if each clauses in \( C \) is satisfied by either \( p \) or \( q \) (as the clauses are disjunctive). As any total assignment satisfying \( \varphi \) can be written as \( p \cup q \) for \( p \in P_1 \subseteq P^* \) and \( q \in P_2 \subseteq P^* \), the claim is proven.

The following four facts prove the statement.

(a) We have
\[ d(w_p, w_q) \geq 2t \]
for any \( p \neq q \).
Take any shortest path from $w_p$ to $w_q$. By Lemma 5 it contains a node $v \in U$, since $w_p$ and $w_q$ belong to distinct gadgets. Thus
\[ d(w_p, w_q) = d(w_p, v) + d(v, w_q) \geq 2t \]
by Lemma 6.

(b) If $d(a, b) > 2t$ for some nodes $a$ and $b$, then some sets $U_p$ and $U_q$ are disjoint. Take $p, q \in P^*$ such that $a \in V_p$ and $b \in V_q$. If $U_p$ and $U_q$ are not disjoint there is a node $z \in U_p \cap U_q \subseteq V_p \cap V_q$.

As $a, z \in V_p$ and $z, b \in V_q$ we get
\[ d(a, b) \leq d(a, z) + d(z, b) \leq 2t \]
by Lemma 3.

(c) If $U_p$ and $U_q$ are disjoint, then
\[ d(w_p, w_q) \geq 3t. \]

Notice that $p \neq q$ as $U_p$ and $U_q$ are non-empty (they contain $u_{\delta_1}$ or $u_{\delta_2}$). Any shortest path from $w_p$ to $w_q$ contains a node $v \in U$ by point (a) above. If there is no other $v' \in U$ on the path, then $w_p, v \in V_p$ and $v, w_q \in V_q$ by Lemma 5, and in particular
\[ v \in V_p \cap V_q \cap U \subseteq U_p \cap U_q \]
contradicting our assumption. So, the path is of the form
\[ w_p \leadsto v \leadsto v' \leadsto w_q \]
with distinct $v, v' \in U$. Hence
\[ d(w_p, w_q) = d(w_p, v) + d(v, v') + d(v', w_q) \geq 3t \]
by Lemma 6.
(d) For any two nodes \( a \) and \( b \), we have
\[
d(a, b) \leq 3t.
\]

Take \( p, q \in P^* \) such that \( a \in V_p \) and \( b \in V_q \). Consider a path
\[
a \sim u_{\delta(p)} \sim u_{\delta(q)} \sim b
\]
where \( \delta(s) \) is the clause \( \delta_i \) not satisfied by \( s \). Since
\[
a, u_{\delta(p)} \in V_p, \\
u_{\delta(p)}, u_{\delta(q)} \in V_{\pi}, \\
u_{\delta(q)}, b \in V_q
\]
we have
\[
d(a, b) = d(a, u_{\delta(p)}) + d(u_{\delta(p)}, u_{\delta(q)}) + d(u_{\delta(q)}, b) \leq 3t
\]
by Lemma 3.

Using Lemma 7, we prove our lower bound for the approximation of the diameter.

**Theorem 1.** Under SETH, there is no algorithm running in
\[
O(m^{2-2\delta-\Omega(1)})
\]
time and distinguishing for \( t > c \cdot m^\delta \) between diameter-3\( t \) and diameter-2\( t \) graphs, for any \( c > 0 \) and \( 0 \leq \delta < 1 \).

**Proof.** Suppose by contradiction to have a procedure \( A \) distinguishing between diameter-3\( t \) and diameter-2\( t \) graphs for \( t \geq c \cdot m^\delta \) in
\[
O(m^{2-2\delta-\Omega(1)}) = O(m^{(1-\delta)(2-\epsilon)})
\]
time for some \( \epsilon > 0 \), \( c > 0 \) and \( 0 \leq \delta < 1 \). Fix \( \gamma = \epsilon/4 \): under SETH there is a \( k \) such that there is no algorithms for \( k \)-SAT running in \( O(2^{(1-\gamma)d}) \) time.

Consider an instance \( \varphi \) of \( k \)-SAT in \( d \)-variables. By Lemma 4, for some
\[
\mu = 2^{d/2+o(d)}
\]
the graph \( G_\varphi \) contains \( m \leq \mu t \) edges and can be constructed in \( O(\mu t) \) time. As \( \delta < 1 \), we can pick
\[
t = \Theta(\mu^{\frac{1}{1-\delta}})
\]
such that
\[ t > c \cdot \mu^\delta t^\delta \geq c \cdot m^\delta \]
and
\[ m \leq \mu t = O(\mu^{1 + \delta/\mu}) = O(\mu^{1/\mu}). \]

Notice that for any input graph with \( m = \mu t \) edges, the procedure \( A \) runs in
\[ O(m^{(1-\delta)(2-\varepsilon)}) = O(\mu^{2-\varepsilon}) \]
time but takes \( \Omega(\mu t) \) time to read the graph, hence we can assume
\[ \mu^{2-\varepsilon} = \Omega(\mu t). \]

To solve \( \varphi \), we construct the graph \( G_{\varphi}^t \), which by Lemma 7 has diameter \( 2t \) or \( 3t \) according to the satisfiability of \( \varphi \), then we apply the procedure \( A \) to distinguish the two cases (as we picked \( t > c \cdot m^\delta \)). The procedure \( A \) runs in
\[ O(m^{(1-\delta)(2-\varepsilon)}) = O(\mu^{2-\varepsilon}) \]
time where \( \mu^{2-\varepsilon} = \Omega(\mu t) \) dominates the \( O(\mu t) \) cost of constructing the graph. Thus, the total time is
\[ O(\mu^{2-\varepsilon}) = 2^{(2-\varepsilon)d/2+O(d)} = 2^{(1-\varepsilon/2)d+O(d)} = 2^{(1-2\gamma)d+O(d)} = O(2^{(1-\gamma)d}) \]
contradicting SETH.

**Corollary 1.** Under SETH, there is no algorithm giving an \((3/2-\varepsilon)\)-multiplicative \( O(m^\delta) \)-additive approximation of the diameter in \( O(m^{2-2\delta-\Theta(1)}) \) time, for any \( \varepsilon > 0 \) and \( 0 \leq \delta < 1 \). In particular, there is no \( O(m^{2-\Theta(1)}) \) time algorithm giving an almost \((3/2-\varepsilon)\)-approximation.

**Proof.** Consider a procedure that gives a \((3/2-\varepsilon)\)-multiplicative \( c \cdot m^\delta \)-additive approximation \( \tilde{D} \) of the diameter \( D \). Take a small enough\(^2 \gamma > 0 \) such that \( \tilde{D} \) satisfies
\[ \tilde{D} \geq \frac{2}{3} D + \gamma D - c \cdot m^\delta. \]

For \( c' = c/(3\gamma) \) and
\[ t > c' \cdot m^\delta = \frac{c \cdot m^\delta}{3\gamma}, \]

\(^2\)For approximation factor \( \frac{3}{2} - \varepsilon \), choose \( \gamma = \frac{4\varepsilon}{3(3-2\varepsilon)}. \)
our procedure distinguishes diameter-2t and diameter-3t graphs: in the first case \( \tilde{D} \leq D = 2t \) while in the second case

\[
\tilde{D} \geq 2t + \gamma \cdot 3t - c \cdot m^\delta \\
> 2t + \frac{c \cdot m^\delta}{3\gamma} - c \cdot m^\delta \\
= 2t + c \cdot m^\delta - c \cdot m^\delta = 2t
\]

Hence, by Theorem 1 it cannot run in \( O(m^{2-2\delta-\Omega(1)}) \) time.

**Eccentricities.** We now prove a lower bound for the eccentricities. For even \( t = 2z \), all the gadgets \( T_p \) appearing in \( G_{2z}^\varphi \) have a center, say \( c_p \in V_p \). We are interested in the maximum eccentricity among these centers.

**Lemma 8.** In \( G_{2z}^\varphi \) the maximum eccentricity

\[
\bar{\epsilon} = \max_{p \in P^*} \epsilon(c_p)
\]

among the centers \( c_p \) of all the gadgets \( T_p \) is either \( 3z \) or \( 5z \). It is \( 5z \) if and only if \( \varphi \) is satisfiable.

**Proof.** By Lemma 7 it is sufficient to prove \( \bar{\epsilon} = D - z \).

We first prove \( \epsilon(c_p) \leq D - z \) for any \( p \in P^* \). For nodes \( v \in V_p \), we have

\[
d(c_p, v) \leq z \leq D - z
\]

by Lemma 3. For \( v \not\in V_p \), any shortest path from \( w_p \) to \( v \) has to pass through \( c_p \). Indeed, consider the first node \( u \in U \) on the shortest path, which must exist by Lemma 5: the sub-path \( w_p \leadsto u \) contains \( c_p \) by Lemma 3. Writing the path as

\[
w_p \leadsto c_p \leadsto v,
\]

we obtain

\[
d(c_p, v) = d(w_p, v) - d(w_p, c_p) \leq D - z
\]

as \( d(w_p, v) \leq D \) and \( d(w_p, c_p) = z \) by Lemma 3.

Now we show \( \epsilon(c_p) \geq D - z \) for some \( p \in P^* \). Take a diametral node \( v \) so that \( \epsilon(v) = D \) and \( p \in P^* \) such that \( v \in V_p \): we have \( d(c_p, v) \leq z \) by Lemma 3 and

\[
\epsilon(c_p) \geq \epsilon(v) - d(c_p, v) \geq D - z
\]

by Lemma 1.
**Theorem 2.** Under SETH, there is no algorithm running in \(O(m^{2-2\delta-\Omega(1)})\) time which decides for \(z > c \cdot m^\delta\) whether the maximum eccentricity over a given subset of nodes is \(5z\) or \(3z\), for any \(c > 0\) and \(0 \leq \delta < 1\).

**Proof.** We adapt the proof of Theorem 1. Notice that we used our hypothetical procedure distinguishing diameter-3\(t\) and diameter-2\(t\) graphs only on instances of the form \(G^t_r\). On such instances having \(t = 2z\), this is equivalent to telling whether \(\bar{\epsilon} = \max_{p \in P^*} \{\epsilon(c_p)\}\) is \(5z\) or \(3z\) by Lemma 8. Thus, any procedure performing the latter in \(O(m^{2-2\delta-\Omega(1)})\) time yields to the same contradiction. □

**Corollary 2.** Under SETH, there is no algorithm giving a \((\frac{5}{3} - \varepsilon)\)-multiplicative \(O(m^\delta)\)-additive approximation of all the eccentricities in a graph in \(O(m^{2-2\delta-\Omega(1)})\) time, for any \(\varepsilon > 0\) and \(0 \leq \delta < 1\). In particular, there is no \(O(m^{2-\Omega(1)})\) time algorithm giving an almost \((\frac{5}{3} - \varepsilon)\)-approximation.

**Proof.** Following the proof of 1, we take a small enough\(^3\) \(\gamma > 0\) such that the eccentricity estimation \(\hat{\epsilon}_v\) satisfies

\[
\hat{\epsilon}_v \geq \frac{3}{5}\epsilon(v) + \gamma\epsilon(v) - c \cdot m^\delta.
\]

For \(c' = c/(5\gamma)\) and

\[
z > c' \cdot m^\delta = \frac{c \cdot m^\delta}{5\gamma},
\]

our procedure can decide whether the maximum eccentricity over a given subset of nodes \(X\) is \(3z\) or \(5z\). In the first case \(\hat{\epsilon}_v \leq \epsilon(v) \leq 3z\) for any \(v \in X\); in the second case, for some \(v \in X\) we have \(\epsilon(v) = 5z\) and

\[
\hat{\epsilon}_v \geq 3z + \gamma \cdot 5z - c \cdot m^\delta
\]

\[
> 3z + 5\gamma \cdot \frac{c \cdot m^\delta}{5\gamma} - c \cdot m^\delta
\]

\[
= 3z + c \cdot m^\delta - c \cdot m^\delta = 3z.
\]

By Theorem 2, no procedure that performs this task can run in \(O(m^{2-2\delta-\Omega(1)})\) time.

**Remark.** Under SETH, Theorem 1 excludes the possibility of a truly sub-quadratic algorithm that takes a graph of constant diameter \(D = 3t\) and produces an approximation \(\hat{D} > \frac{2}{3}D = 2t\). This result is surprisingly tight: when \(D\) is constant but not divisible by 3, an \(O(m^{2-\delta})\)-time algorithm that produces a value \(\hat{D} > \frac{2}{3}D\) is possible, as shown by Roditty and Vassilevska Williams [23].

\(^3\)For approximation factor \(\frac{5}{3} - \varepsilon\), choose \(\gamma = \frac{9\varepsilon}{\delta(3-3\varepsilon)}\).
Chapter 3

Approximation algorithms

In this chapter we present the previous algorithms that approximate the extremal distances efficiently; then, we present our novel algorithm which gives a family of new bounds for undirected graphs.

3.1 One-search approximations

There is a simple, well-known algorithm that is able to produce very fast approximations of the diameter, the radius and all the eccentricities on undirected graphs. It is sufficient to choose any node \( x \in V \) and perform a graph search from \( x \). During the search, we compute \( d(x, v) \) for every \( v \in V \) and, in particular, the eccentricity \( \epsilon(x) \). These values can be used to obtain 2-approximations of the diameter and the radius, and 3-approximations of all the eccentricities in the graph. The main idea behind these approximation is to use the known distances in the graph to give an upper bound to other distances in the graph. This is achieved exploiting the triangle inequality.

For the diameter and the radius, it is sufficient to return \( \tilde{D} = \tilde{r} = \epsilon(x) \) as estimate for both the measures. In Lemma 9, we show how the triangular inequality works to provide the desired upper bounds.

**Lemma 9.** The value \( \epsilon(x) \) for any \( x \in V \) is a 2-approximation of the diameter \( D \) and the radius \( r \), without additive error.

**Proof.** First, we have \( r \leq \epsilon(x) \leq D \) by definition of radius and diameter. Hence, we need to prove \( \epsilon(x) \geq D/2 \) and \( \epsilon(x) \leq 2r \).

We employ the triangular inequality to give an upper bound of the diameter \( D \).
Take two diametral nodes $a, b \in V$ such that $d(a, b) = D$. We get:

$$
D = d(a, b) \\
\leq d(a, x) + d(x, b) \\
\leq \epsilon(x) + \epsilon(x) \leq 2\epsilon(x).
$$

In the case of the radius, we do the opposite. Take a central node $c \in V$ so that $\epsilon(c) = r$ and a node $y \in V$ such that $d(x, y) = \epsilon(x)$. We get

$$
\epsilon(x) = d(x, y) \\
\leq d(x, c) + d(c, y) \\
\leq \epsilon(c) + \epsilon(c) \leq 2r
$$

concluding our proof.

In the case of the eccentricity of any node $v \in V$, we return as estimate

$$
\tilde{\epsilon}_v = \max \left\{ \frac{d(x, v)}{\epsilon(x) - d(x, v)} \right\}. \tag{3.1}
$$

Here the proof of approximation quality is slightly more complex.

**Lemma 10.** For any node $v \in V$, the value $\tilde{\epsilon}_v$ defined in Equation (3.1) (for any $x \in V$) is a 3-approximation of the eccentricity of $v$ without additive error. Namely,

$$
\frac{1}{3} \epsilon(v) \leq \tilde{\epsilon}_v \leq \epsilon(v).
$$

**Proof.** We prove the two inequalities separately.

First, $\epsilon(v) \geq d(x, v)$ by definition of eccentricity and $\epsilon(v) \geq \epsilon(x) - d(x, v)$ thanks to Lemma 1. Hence

$$
\tilde{\epsilon}_v = \max \{ d(x, v), \epsilon(x) - d(x, v) \} \leq \epsilon(v).
$$

Second, we have $\tilde{\epsilon}_v \geq \epsilon(x) - d(x, v)$ and $d(x, v) \leq \tilde{\epsilon}_v$ by construction, and $\epsilon(x) \geq \epsilon(v) - d(x, v)$ again by Lemma 1. Hence

$$
\tilde{\epsilon}_v \geq \epsilon(x) - d(x, v) \\
\geq \epsilon(v) - d(x, v) - d(x, v) \\
\geq \epsilon(v) - 2\tilde{\epsilon}_v
$$

from where we get $\tilde{\epsilon}_v \geq \epsilon(v)/3$. \qed
3.2 Previous algorithms

A seminal paper by Aingworth, Chekuri, Indyk, and Motwani [1] presented some new ideas, which are at the basis of the following development of approximation algorithms for extremal distances. First of all, they present an algorithm able to distinguish between graphs of diameter 4 and graphs of diameter 2 in sub-quadratic $\tilde{O}(m\sqrt{n})$ time. Then, basing on this algorithm, they devise an almost $3/2$-approximation algorithm for the diameter running in $\tilde{O}(m\sqrt{n} + n^2)$ time, and a 2-additive approximation algorithm for APSP running in $O(n^{2.5}\sqrt{\log n})$ time. Here we describe only the algorithm for distinguishing between diameter 4 and diameter 2.

The intuition. The algorithm of Aingworth et al. works by launching several graph searches, starting from a subset of the nodes in the graph. They are able to select $\tilde{O}(\sqrt{n})$ nodes from where to launch the search, in such a way that if the diameter is 4 then at least one of the performed searches finds a pair of nodes at distance 3. By finding two nodes at distance 3, we exclude the possibility that the diameter is 2 and we return 4 as the answer.

Assume for the moment to have an undirected graph in input. The main idea to select the nodes, is to treat differently the nodes which have high degree and the nodes which have low degree. Let $N(v)$ denote the neighbors of a node $v \in V$. If there is a node $w$ such that $|N(w)|$ is small (low degree), then we can afford to launch a graph search from each $u \in N(w)$. Otherwise, the sets $N(u)$ for $u \in V$ are all large, and for this reason there is by Lemma 2 a small set $S$ hitting all of them; in this case, we launch the searches from $S$ instead. We will see later how the properties of $N(w)$ or $S$ guarantee that we find at least one node $v$ of eccentricity 3 or more.

The algorithm. We now describe the algorithm in the general case of directed graph. Let $\overrightarrow{N}(u)$ denote the out-neighbors of the node $u \in V$, namely:

$$\overrightarrow{N}(u) = \{v \in V \mid (u, v) \in E\}.$$ 

We distinguish two cases: either there is a node $w \in V$ with $|\overrightarrow{N}(w)| \leq \sqrt{n\log n}$, or we have $|\overrightarrow{N}(u)| > \sqrt{n\log n}$ for every node $u \in V$. In the first case, we run a forward graph search from $w$ and a backward graph search from each $u \in \overrightarrow{N}(w)$. In the second case, we first find an hitting set $S \subseteq V$ of size $\Theta(\sqrt{n\log n})$, such that

$$S \cap \overrightarrow{N}(u) \neq \emptyset$$
for every $u \in V$. This can be done via random sampling by Lemma 2; however, in this case it can be also done deterministically (we refer the reader to the original paper [1]). Then, we run a forward graph search from each $v \in S$.

The whole procedure is summarized in Algorithm 3.1.

**Algorithm 3.1** Diameter 2-vs-4 (adapted from Aingworth et al. [1]).

**Input:** Directed graph $G = (V, E)$ of diameter either 2 or 4.

**Output:** The diameter $D(G)$, which is either 2 or 4.

1. Compute the value $|\overrightarrow{N}(v)|$ for every node $v \in V$, and execute one of the following two cases:
   
   **Case 1.** There is a node $w \in V$ with $|\overrightarrow{N}(w)| \leq \sqrt{n \log n}$
   
   (a) Run a forward graph search from $w$.
   
   (b) Run a backward graph search from each $v \in \overrightarrow{N}(w) \cup \{w\}$.

   **Case 2.** For every node $u \in V$ we have $|\overrightarrow{N}(u)| > \sqrt{n \log n}$
   
   (a) Find a set $S \subseteq V$ of size $\Theta(\sqrt{n \log n})$ hitting $\overrightarrow{N}(u)$ for every $u \in V$.
   
   (b) Run a forward graph search from each $v \in S$.

2. If any graph search has depth 3 or more, return 4, otherwise return 2.

We now analyze the algorithm.

**Lemma 11.** Algorithm 3.1 runs in

$$O(\sqrt{n \log n} \cdot C) = \tilde{O}(m\sqrt{n})$$

time.

**Proof.** In Step 1, checking whether to execute Case 1 or Case 2 takes only linear time. In Case 2, finding the hitting set $S$ also requires linear time. In both Case 1 and Case 2, we perform $O(\sqrt{n \log n})$ graph searches, yielding a total running time $O(\sqrt{n \log n} \cdot C)$.

**Lemma 12.** Algorithm 3.1 is correct.
Proof. If the diameter is 2, then clearly no graph search can find a distance of 3 or more, so the algorithm answers correctly. Hence, assume that the diameter is 4 and, by contradiction, that all the graph searches have depth at most 2. Take \( a, b \in V \) such that \( d(a, b) = 4 \).

We analyze the two cases separately.

**Case 1.** There is a node \( w \in V \) with \( |\overset{\rightarrow}{N}(w)| \leq \sqrt{n \log n} \).

As we perform a forward graph search from \( w \), we have \( d(w, b) \leq 2 \) by hypothesis. Hence, there is node \( c \) (on a shortest path from \( w \) to \( b \)) such that \( d(w, c) \leq 1 \) and \( d(c, b) \leq 1 \). Notice that \( c \in \overset{\rightarrow}{N}(w) \cup \{w\} \) as \( d(w, c) \leq 1 \), so we perform a backward graph search from \( c \) and, in particular, \( d(a, c) \leq 2 \) by hypothesis. By the triangular inequality, we get

\[
d(a, b) \leq d(a, c) + d(c, b) \leq 2 + 1 = 3
\]

which contradicts our hypothesis.

**Case 2.** For every node \( v \in V \) we have \( |\overset{\rightarrow}{N}(v)| > \sqrt{n \log n} \).

The set \( S \) hits every set \( \overset{\rightarrow}{N}(v) \) and, in particular, there is a node \( c \in S \) with \( d(a, c) \leq 1 \). As we perform a graph search from \( c \in S \), we have \( d(c, b) \leq 2 \) by hypothesis. Again, by triangular inequality:

\[
d(a, b) \leq d(a, c) + d(c, b) \leq 1 + 2 = 3
\]

which is absurd. \( \square \)

**A 3/2-approximation algorithm.** The algorithm is generalized to produce an almost 3/2-approximation of the diameter: we sketch the main idea here. First, instead of considering the neighbors \( N(u) \) and checking if they are few or many, we take the nearest \( \ell \approx \sqrt{n} \) neighbors \( N_\ell(u) \) for every node \( u \). Second, we find a set \( S \) of size \( \tilde{O}(\sqrt{n}) \) hitting all the sets \( N_\ell(u) \), and we select the node \( w \) such that the nodes \( N_\ell(w) \) reach the farthest distance from \( w \). Finally, we execute the graph searches prescribed by both cases, from each node \( v \in N_\ell(w) \) and from each node \( v \in S \), and we return the maximum distance found. This leads to an almost 3/2-approximation of the diameter; however, we incur in an extra term \( n^2 \) in the running time required to build the sets \( N_\ell(u) \) for every node \( u \in V \).

Roditty and Vassilevska Williams [23] showed a randomized algorithm based on the ideas of Aingworth et al. [1] which removes the extra \( n^2 \) term in the running time, and gives an almost 3/2-approximation in \( \tilde{O}(m\sqrt{n}) \) expected time. This algorithm works by first constructing \( S \subseteq V \) via random sampling, and then finding a node \( w \in V \): specifically, the node \( w \) is the farthest node from \( S \); the
rest of the algorithm is similar to that by Aingworth et al. and is illustrated in Algorithm 3.2.

**Algorithm 3.2** Diameter 3/2-approximation (adapted from [23]).

**Input:** Directed graph $G = (V, E)$.

**Output:** An approximation of the diameter $\hat{D}$.

1. Let $q = \sqrt{n/\log n}$.

2. Repeat:
   
   (a) Sample $\Theta(q \cdot \log n)$ nodes uniformly at random.
   
   Let $S \subseteq V$ be the set of sampled nodes.
   
   (b) Run a multiple-source backward graph search from $S$.
   
   Let $w$ be the last visited node.
   
   (c) Run a forward graph search from $w$.
   
   Let $W$ be the set containing the first $\ell = \lceil n/q \rceil$ visited nodes.

   until $W \cap S \neq \emptyset$.

3. Run a forward graph search from each $v \in S$ and a backward graph search from each $v \in W$.

4. Return the maximum distance found, namely:

\[
\hat{D} = \max \left\{ \max_{u \in V} d(w, u), \max_{u \in S} d(v, u), \max_{u \in W} d(u, v) \right\}.
\]

We refer the reader to the original papers [1, 23] for a rigorous analysis of these algorithm. In the next section, we show and analyze a new algorithm, which in the case of undirected graphs generalizes the above algorithms and gives a sequence of new bounds for different values of the approximation factor $\alpha$. 
CHAPTER 3. APPROXIMATION ALGORITHMS

3.3 New approximation bounds

The idea behind the approximation algorithms by Aingworth et al. and Roditty and Vassilevska Williams is to look for a small set \( W = N_\ell(w) \), for some \( \ell > 0 \), which contains nodes at large distance \( h \) from \( w \). In general, this set may not exist in a graph for a given distance \( h \): this happens if every node \( v \) has more than \( \ell \) nodes at distance \( h \). Still, in this case there is a useful hitting set \( S \) of size \( \tilde{O}(n/\ell) \) at distance \( h \) from every node \( v \). In the algorithms of Aingworth et al. and of Roditty and Vassilevska Williams, this idea is exploited for \( \ell \approx \sqrt{n} \) to obtain sets \( W \) and \( S \) of size \( \tilde{O}(\ell) = \tilde{O}(n/\ell) = \tilde{O}(\sqrt{n}) \).

The intuition. We introduce a novel machinery that applies this idea iteratively to obtain smaller and smaller \( W_i = N_\ell^i(w_i) \) for \( i = 0, 1, \ldots, k \). At each step, we are given a small set \( W_i \) containing the nodes at a certain distance \( h \) from \( w_i \) and look for a smaller set \( W_{i+1} \) containing the nodes at a certain distance \( h' \) from \( w_{i+1} \) (this will be formalized next). Suppose that a suitable set \( W_{i+1} \) does not exist in the graph, for a certain size \( \ell_{i+1} \) and distance \( h' \): this means that every node \( v \) has more than \( \ell_{i+1} \) nodes at distance \( h' \). In this case, one would like to exhibit a small hitting set \( S_i \) at bounded distance from every node in the graph; we also want \( S_i \subseteq W_i \) to exploit the fact that \( W_i \) was small in the first place.

We first observe that if \( S_i \) is close to every node in \( W_i \), then by transitivity it is close to the rest of the nodes (as \( h \) is large). Unfortunately, there is no guarantee that a small \( S_i \subseteq W_i \) hits the neighbors \( N_{\ell_{i+1}}(v) \) at distance \( < h' \) from every \( v \in W_i \), as some of the neighbors could fall outside \( W_i \). However, we observe that it is possible to hit the neighbors of inner nodes, those distant at least \( h' \) from the outside of \( W_i \).

In the algorithm of Roditty and Vassilevska Williams, the node \( w \) used to generate \( W = N_\ell(w) \) was selected as the farthest from \( S \). In our iterative procedure, we select the node \( w_{i+1} \) far from both \( S_i \) and \( V \setminus W_i \): with this trick, we exploit the fact that \( w_{i+1} \) is an inner node and manage to guarantee either that \( W_{i+1} = N_{\ell_{i+1}}(w_{i+1}) \) contains all the nodes at distance \( < h' \) for \( w_{i+1} \) or that \( S_i \) is at distance \( h' \) from every node in \( W_i \).

Sampling procedure. We now give a rigorous description of our approximation algorithm, parametrized by a integer constant \( k \geq 0 \). We start with a sampling phase, whose output is a sequence of nodes \( w_1, \ldots, w_k \in V \) and a sequence of node sets \( S_0, \ldots, S_k \subseteq V \).

Let \( q = (n/\log n)^{1+\epsilon} \). (This choice minimizes \( |S_0 \cup \cdots \cup S_k| \) to \( \tilde{O}(n^{1+\epsilon}) \) as shown in Lemma 15.) We begin by setting \( W_0 = V \) and \( \ell_0 = |W_0| = n \); then, we perform \( k \) iterations, numbered from 0 to \( k - 1 \). Iteration \( i \) receives as input the
CHAPTER 3. APPROXIMATION ALGORITHMS

node set \( W_i \) of size \( \ell_i = |W_i| \), and produces the set \( S_i \subseteq W_i \), the node \( w_{i+1} \) and a new node set \( W_{i+1} \) of smaller size \( \ell_{i+1} = \lceil \ell_i/q \rceil \). This set is used by the next iteration if \( i + 1 < k \), and it is used to define \( S_k = W_k \) when \( i + 1 = k \).

Specifically, we construct \( S_i \) by sampling uniformly at random \( \Theta(q \cdot \log n) = \Theta(\ell_i/\ell_{i+1} \cdot \log n) \)

nodes from \( W_i \) as required by Lemma 2. Then, we compute the distance from

\[ Z_i = (V \setminus W_i) \cup S_i \]
to every node \( u \in V \) (with a multiple-source graph search as described in Section 1.3) and select a node \( w_{i+1} \in V \) that maximizes this distance. Finally, we perform a graph search from \( w_{i+1} \) and construct the set

\[ W_{i+1} = N_{\ell_{i+1}}(w_{i+1}) \]

In Lemma 13, we show that

\[ W_{i+1} \cap Z_i \neq \emptyset \]

with high probability. This property is crucial for our approximation bound. To obtain a Las Vegas\(^1\) algorithm, we check the condition \( W_{i+1} \cap Z_i \neq \emptyset \) at the end of iteration \( i \): if it does not hold, we repeat the iteration \( i \) resampling \( S_i \). The whole sampling procedure is summarized in Algorithm 3.3.

**Lemma 13.** In Algorithm 3.3, the until \( W_{i+1} \cap Z_i \neq \emptyset \) condition holds with high probability.

**Proof.** Consider the family of sets

\[ \mathcal{H} = \{ N_{\ell_{i+1}}(u) : u \in V \text{ and } N_{\ell_{i+1}}(u) \subseteq W_i \} \]
on the universe \( W_i \). We apply Lemma 2 to show that \( S_i \) hits every set in \( \mathcal{H} \) with high probability. Indeed, we have \( |\mathcal{H}| \leq n \), \( |W_i| = \ell_i \), each set in the family \( N_{\ell_{i+1}}(u) \) has size \( \ell_i \), and \( S_i \) contains

\[ \Theta(q \cdot \log n) = \Theta(\ell_i/\ell_{i+1} \cdot \log n) \]

nodes sampled uniformly at random from \( W_i \).

If \( W_{i+1} \not\subseteq W_i \) then

\[ W_{i+1} \cap Z_i \supseteq W_{i+1} \cap (V \setminus W_i) = W_{i+1} \setminus W_i \neq \emptyset. \]

\(^1\)A Las Vegas randomized algorithm always gives the correct answer, but its running time is a random variable (with bounded expected value) [21].
Algorithm 3.3 Sampling procedure.

**Input:** Undirected graph \( G = (V, E) \) and a constant integer parameter \( k \geq 0 \).

**Output:** Nodes \( w_1, \ldots, w_k \in V \) and node sets \( S_0, \ldots, S_k \subseteq V \).

1. Let \( W_0 = V \), \( \ell_0 = n \) and \( q = (n/\log n)^{1/k+1} \).

2. For each \( i = 0, 1, \ldots, k - 1 \):
   Repeat:
   (a) Sample \( \Theta(q \cdot \log n) \) nodes uniformly at random from \( W_i \).
   Let \( S_i \) be the set of sampled nodes.
   (b) Run a multiple-source graph search from \( Z_i = (V \setminus W_i) \cup S_i \).
   Let \( w_{i+1} \) be the last visited node.
   (c) Perform a graph search from \( w_{i+1} \).
   Let \( W_{i+1} \) be the set containing the first \( \ell_{i+1} = \lceil \ell_i/q \rceil \) visited nodes.
   until \( W_{i+1} \cap Z_i \neq \emptyset \).

3. Set \( S_k = W_k \). (Also, define \( Z_k = (V \setminus W_k) \cup S_k = V \) for uniformity.)

Otherwise, \( W_{i+1} = N_{\ell_{i+1}}(w_{i+1}) \in \mathcal{H} \), hence, with high probability, \( S_i \) hits \( W_{i+1} \) and
\[
W_{i+1} \cap Z_i \supseteq W_{i+1} \cap S_i \neq \emptyset.
\]

**Lemma 14.** Algorithm 3.3 runs in \( O(C) \) expected time.

**Proof.** Steps 1 and 3 take at most linear time. In each execution of the repeat-until loop in step 2, the time spent in sampling \( S_i \) and checking the termination condition is dominated by the \( O(C) \) cost of substeps 2b and 2c. By Lemma 13, the repeat-until loop for iteration \( i \) is executed a constant number of times in expectation. As we perform \( k \) iteration for constant \( k \), the total expected cost is \( O(C) \). \( \square \)

**Lemma 15.** Algorithm 3.3 returns sets \( S_0, \ldots, S_k \) with
\[
|S_0 \cup \cdots \cup S_k| = O(n^{1/k+1} \log^{1/k+1} n).
\]
CHAPTER 3. APPROXIMATION ALGORITHMS

Proof. Each set $S_0, \ldots, S_{k-1}$ is of size
\[
\Theta(q \cdot \log n) = O((n/\log n)^{\frac{1}{k+1}} \cdot \log n) = O(n^{\frac{1}{k+1}} \log^{\frac{k}{k+1}} n)
\]
by construction. Notice that $\ell_i = O(n/q^i)$, thus
\[
|S_k| = |W_k| = \ell_k = O(n/q^k) = O(n/(n/\log n)^{\frac{k}{k+1}}) = O(n^{\frac{1}{k+1}} \log^{\frac{k}{k+1}} n).
\]

For any $0 \leq i \leq k$, let $h_i$ be the maximum distance from $Z_i$ to any node in the graph, where $h_k = 0$ by construction as $Z_k = V$. The set $W_{i+1}$ contains all the nodes at distance $< h_i$ from $w_{i+1}$. (See fact (c) in the next proof.) We relate the distances $d(S_i, v)$ with the values $h_i$. For $i = 0$, we have
\[
Z_0 = (V \setminus W_0) \cup S_0 = S_0
\]
as $W_0 = V$, thus $d(S_0, v) \leq h_0$. For $i \geq 1$, we obtain the following.

Lemma 16. For $1 \leq i \leq k$ and any node $v \in V$, we have either
\[
d(S_i, v) \leq h_i
\]
or
\[
d(S_i, v) \leq d(w_i, v) - h_{i-1} + 2h_i + M.
\]

Proof. We will use the following facts.

(a) For any node $u \in V$ either
\[
d(S_i, u) \leq h_i
\]
or
\[
d(V \setminus W_i, u) \leq h_i,
\]
as $h_i$ is the maximum distance from $Z_i = (V \setminus W_i) \cup S_i$ to any node.

(b) If $u \in W_i$ then
\[
d(u, w_i) \leq d(V \setminus W_i, w_i),
\]
as $W_i$ comprises the $\ell_i$ nearest nodes to $w_i$.

(c) We have
\[
d(V \setminus W_i, w_i) \geq h_{i-1}.
\]
Take a node $z \in W_i \cap Z_{i-1}$ (as it exists due to the termination condition of the repeat-until loop). By (b),
\[
d(V \setminus W_i, w_i) \geq d(z, w_i) \geq d(Z_{i-1}, w_i) = h_{i-1}
\]
as $w_i$ is the farthest node from $Z_{i-1}$.
Fix a node \( v \in V \). Consider a shortest path \( P \) from \( w_i \) to \( v \). Let \( a \) be the last node on \( P \) such that \( d(S_i, a) \leq h_i \). The node \( a \) exists since \( d(S_i, w_i) \leq h_i \). Indeed, taking any node \( u \in S_i \subseteq W_i \), we have

\[
d(S_i, w_i) \leq d(u, w_i) \leq d(V \setminus W_i, w_i)
\]

by (b): this implies that \( d(S_i, w_i) \leq h_i \) by (a).

If \( a = v \), then \( d(S_i, v) \leq h_i \) and the statement is proven. Otherwise, take the node \( b \) which follows \( a \) on \( P \), so the path \( P \) is of the form

\[
w_i \to a \to b \to v
\]

with \( d(S_i, b) > h_i \). By (a),

\[
d(V \setminus W_i, b) \leq h_i.
\]

As

\[
d(V \setminus W_i, w_i) \geq h_{i-1}
\]

by (c), then

\[
d(w_i, b) \geq h_{i-1} - h_i
\]

by triangle inequality. Writing \( P \) as \( w_i \to b \to v \) we get

\[
d(b, v) = d(w_i, v) - d(w_i, b) \leq d(w_i, v) - (h_{i-1} - h_i).
\]

We can bound \( d(S_i, v) \) as

\[
d(S_i, v) \leq d(S_i, a) + w(a, b) + d(b, v) \\
\leq h_i + M + d(w_i, v) - (h_{i-1} - h_i) \\
= d(w_i, v) - h_{i-1} + 2h_i + M
\]

and the statement is proven.

The next lemma combines the inequalities obtained at each iteration \( i \).

**Lemma 17.** Let \( v \in V \) be any node. For some \( 0 \leq \bar{i} \leq k \), we have

\[
d(S_{\bar{i}}, v) \leq h_{\bar{i}} \leq (2^{k-\bar{i}} - 1)(\Delta + M),
\]

where

\[
\Delta = \max_{1 \leq i \leq k} \{d(w_i, v) - d(S_i, v)\}.
\]
Proof. As $h_0$ is the maximum distance from $Z_0 = S_0$, we have $d(S_0, v) \leq h_0$. Let $\bar{i} \in \{0, \ldots, k\}$ be the maximum index $i$ such that $d(S_i, v) \leq h_i$.

We show

$$h_i \leq (2^{k-i} - 1)(\Delta + M)$$

for $\bar{i} \leq i \leq k$, proving the lemma.

Start with $i = k$: we have

$$(2^{k-i} - 1)(\Delta + M) = (2^{k-k} - 1)(\Delta + M) = (1 - 1)(\Delta + M) = 0$$

and $h_k = 0$ by construction.

Now suppose inductively $h_i \leq (2^{k-i} - 1)(\Delta + M)$ for some $\bar{i} < i \leq k$. We need to show

$$h_{i-1} \leq (2^{k-(i-1)} - 1)(\Delta + M).$$

By the assumption $i > \bar{i}$, it cannot be $d(S_i, v) \leq h_i$ (as $\bar{i}$ is the maximum index satisfying this condition), so it must be that

$$d(S_i, v) \leq d(w_i, v) - h_i + 2h_i + M$$

by Lemma 16. Therefore

$$h_{i-1} \leq 2h_i + d(w_i, v) - d(S_i, v) + M$$
$$\leq 2h_i + \Delta + M$$
$$\leq 2 \cdot (2^{k-i} - 1)(\Delta + M) + \Delta + M$$
$$= (2^{k-(i-1)} - 1)(\Delta + M)$$

completing our inductive proof.

Approximations. The sampling procedure just described is at the core of our approximation algorithm: we exploit the nodes $w_1, \ldots, w_k$ and the sets $S_0, \ldots, S_k$ to produce our estimations of the extremal distances. Notice that during the sampling we launch a graph search from $w_i$ for every $i = 1, \ldots, k$, so we already know the values $d(w_i, v)$ for every node $v \in V$. After the sampling, we execute the most costly part of the whole algorithm: we perform a graph search from each node $x \in S_0, \ldots, S_k$ and compute the value $d(x, v)$ for every $v \in V$. Notice that, in particular, we have obtained $\epsilon(w_i)$ for every $i = 1, \ldots, k$ and $\epsilon(x)$ for every $x \in S_0, \ldots, S_k$.

At the end, we use these values to compute the approximated values $\tilde{D}$ for the diameter, $\tilde{r}$ for the radius and $\tilde{\epsilon}_v$ for the eccentricity of $v \in V$, defined as
follows. For the diameter, we return the maximum eccentricity found during the graph searches:

\[
\hat{D} = \max \left\{ \max_{1 \leq i \leq k} \epsilon(w_i), \max_{x \in S_0 \cup \cdots \cup S_k} \epsilon(x) \right\}.
\]

For the radius, we return instead the minimum eccentricity, but in this case we only need to consider the nodes in \(S_0, \ldots, S_k\):

\[
\hat{r} = \min_{x \in S_0 \cup \cdots \cup S_k} \epsilon(x).
\]

For the eccentricities of a node \(v \in V\), the estimation \(\hat{\epsilon}_v\) can be obtained in two ways:

- from the distance between a node \(x\) and \(v\) computed during the searches, or
- from the eccentricity of a node \(x\) close to \(v\), after subtracting \(d(x, v)\) to avoid over-estimating.

The approximation \(\hat{\epsilon}_v\) is indeed defined as:

\[
\hat{\epsilon}_v = \max \left\{ \max_{1 \leq i \leq k} d(w_i, v), \max_{x \in S_0} d(x, v), \max_{x \in S_1 \cup \cdots \cup S_k} \epsilon(x) - d(x, v) \right\}.
\]

The whole approximation procedure is summarized in Algorithm 3.4.

**Lemma 18.** Algorithm 3.4 runs in

\(O(n^{\frac{1}{k+1}} \log^{\frac{1}{k+1}} n \cdot C) = \tilde{O}(mn^{\frac{1}{k+1}})\)

expected time.

**Proof.** Step 1 runs in \(O(k \cdot C)\) expected time by Lemma 14. Step 2 performs \(|S_0 \cup \cdots \cup S_k|\) graph searches and by Lemma 15 runs in

\(O(n^{\frac{1}{k+1}} \log^{\frac{k}{k+1}} n \cdot C)\)

time. Finally, step 3 does not increase the asymptotic time complexity as the values \(\hat{D}, \hat{r}\) and \(\hat{\epsilon}_v\) can be be computed during the previous steps at no extra cost.

**Lemma 19.** Algorithm 3.4 returns an estimation \(\hat{D}\) of the diameter \(D\) such that

\[
\frac{2^k}{2^{k+1} - 1} D - \frac{2^k - 1}{2^{k+1} - 1} M \leq \hat{D} \leq D.
\]
Algorithm 3.4 Approximation of the diameter, the radius and all the eccentricities.

**Input:** Undirected graph \( G = (V, E) \) and an integer \( k \geq 0 \).

**Output:** Estimations \( \tilde{D} \) (diameter), \( \tilde{r} \) (radius) and \( \tilde{\epsilon}_v \) (eccentricity of every node \( v \in V \)).

1. Execute Algorithm 3.3 and obtain \( w_1, \ldots, w_k \in V \) and \( S_0, \ldots, S_k \subseteq V \).
2. Run a graph search from every node \( x \in S_0, \ldots, S_k \).
3. Return:

\[
\tilde{D} = \max \left\{ \max_{1 \leq i \leq k} \epsilon(w_i), \max_{x \in S_0 \cup \ldots \cup S_k} \epsilon(x) \right\}
\]
\[
\tilde{r} = \min_{x \in S_0 \cup \ldots \cup S_k} \epsilon(x)
\]
\[
\tilde{\epsilon}_v = \max \left\{ \max_{1 \leq i \leq k} d(w_i, v), \max_{x \in S_0} d(x, v), \max_{x \in S_1 \cup \ldots \cup S_k} \epsilon(x) - d(x, v) \right\}
\]

**Proof.** Recall the definition of \( \tilde{D} \):

\[
\tilde{D} = \max \left\{ \max_{1 \leq i \leq k} \epsilon(w_i), \max_{x \in S_0 \cup \ldots \cup S_k} \epsilon(x) \right\}
\]

Note that \( \tilde{D} = \epsilon(u) \) for some \( u \in V \), hence \( \tilde{D} = \epsilon(u) \leq D \) by definition of diameter. We now prove the other inequality: start by taking \( v \in V \) such that \( \epsilon(v) = D \). For any \( i \), we have

\[
d(w_i, v) \leq \epsilon(w_i) \leq \tilde{D}
\]

and

\[
d(S_i, v) \geq D - \tilde{D}.
\]

To show the latter inequality, take \( x \in S_i \) such that \( d(x, v) = d(S_i, v) \): we have

\[
\tilde{D} \geq \epsilon(x) \\
\geq \epsilon(v) - d(x, v) \\
\geq D - d(S_i, v)
\]
CHAPTER 3. APPROXIMATION ALGORITHMS

by Lemma 1.
Now apply Lemma 17 to \( v \), where by the facts shown above
\[
\Delta = \max_{1 \leq i \leq k} \{d(w_i, v) - d(S_i, v)\} \\
\leq \tilde{D} - (D - \tilde{D}) \\
= 2\tilde{D} - D
\]
and \( d(S_i, v) \geq D - \tilde{D} \). We get
\[
D - \tilde{D} \leq (2^{k-1} - 1)(\Delta + M) \\
\leq (2^k - 1)(\Delta + M) \\
\leq (2^k - 1)(2\tilde{D} - D + M),
\]
hence
\[
\tilde{D} \geq \frac{2^k}{2^{k+1} - 1}D + \frac{2^k - 1}{2^{k+1} - 1}M.
\]

Remark. In the special case \( k = 1 \), our algorithm behaves identically to the algorithm by Roditty and Vassilevska Williams when restricted to undirected graphs. With our analysis we obtain
\[
\tilde{D} \geq \frac{2^k}{2^{k+1} - 1}D - \frac{2^k - 1}{2^{k+1} - 1}M
\]
\[
= \frac{2^1}{2^2 - 1}D - \frac{2^1 - 1}{2^2 - 1}M \\
= \frac{2}{3}D - \frac{1}{3}M.
\]
For weighted graphs, this shows a smaller additive term \( \frac{1}{3}M \) instead of the term \( M \) stated in the original paper [23].

Lemma 20. Algorithm 3.4 returns \( \tilde{r} \) such that
\[
r \leq \tilde{r} \leq \frac{2^{k+1} - 1}{2^k}r + \frac{2^k - 1}{2^k}M.
\]

Proof. By definition
\[
\tilde{r} = \min_{x \in S_0 \cup \ldots \cup S_k} \epsilon(x) \geq r.
\]
We now prove the other inequality. Take \( v \) such that \( \epsilon(v) = r \). For every \( i \), we have
\[
d(w_i, v) \leq \epsilon(v) = r
\]
and 
\[ d(S_i, v) \geq \hat{r} - r. \]
To show the latter inequality, take \( x \in S_i \) such that 
\[ d(S_i, v) = d(x, v), \]
we have 
\[ \tilde{r} \leq \epsilon(x) \leq \epsilon(v) + d(x, v) = r + d(S_i, v). \]

Apply Lemma 17 to \( v \), where 
\[ \Delta = \max_{1 \leq i \leq k} \{d(w_i, v) - d(S_i, v)\} \leq r - (\hat{r} - r) \leq 2r - \hat{r}, \]
and \( d(S_i, v) \geq \hat{r} - r \). We get 
\[ \hat{r} - r \leq (2^{k-i} - 1)(\Delta + M) \leq (2^k - 1)(\Delta + M) \leq (2^k - 1)(2r - \hat{r} + M), \]

hence 
\[ \hat{r} \leq \frac{2^{k+1} - 1}{2^k} r + \frac{2^k - 1}{2^k} M. \]

\[ \Box \]

**Lemma 21.** Algorithm 3.4 returns an estimation \( \tilde{\epsilon}_v \) of the eccentricity \( \epsilon(v) \) such that 
\[ \frac{2^k + 1}{3 \cdot 2^k - 1} \epsilon(v) - \frac{2^k - 1}{3 \cdot 2^k - 1} M \leq \tilde{\epsilon}_v \leq \epsilon(v). \]

*Proof.* Consider any node \( v \in V \); the value \( \tilde{\epsilon}_v \) is defined as:

\[ \tilde{\epsilon}_v = \max \left\{ \max_{1 \leq i \leq k} d(w_i, v), \max_{x \in S_0} d(x, v), \max_{x \in S_1 \cup \ldots \cup S_k} \epsilon(x) - d(x, v) \right\} \]

Notice that the value \( \tilde{\epsilon}_v \) is obtained either as \( d(u, v) \) or as \( \epsilon(u) - d(u, v) \) for some node \( u \in V \). In the first case, we have \( \tilde{\epsilon}_v = d(u, v) \leq \epsilon(v) \) by definition of eccentricity. In the second case, we have \( \epsilon(u) \leq \epsilon(v) + d(u, v) \) by Lemma 1, hence again:

\[ \tilde{\epsilon}_v = \epsilon(u) - d(u, v) \leq \epsilon(v) + d(u, v) - d(u, v) = \epsilon(v). \]
We now prove the other inequality. For every $i$, we have
\[ d(w_i, v) \leq \tilde{\epsilon}_v \]
and
\[ d(S_i, v) \geq \frac{\epsilon(v) - \tilde{\epsilon}_v}{2}. \]

To prove the latter inequality, take $x \in S_i$ such that $d(x, v) = d(S_i, v)$: we have
\[
\tilde{\epsilon}_v \geq \epsilon(x) - d(x, v) \\
\geq \epsilon(v) - d(x, v) - d(x, v) \\
= \epsilon(v) - 2d(S_i, v)
\]
by Lemma 1, hence $d(S_i, v) \geq \frac{\epsilon(v) - \epsilon_v}{2}$.

Apply Lemma 17 to the node $v$, where
\[
\Delta = \max_{1 \leq i \leq k} \{d(w_i, v) - d(S_i, v)\}
\]
\[
\leq \tilde{\epsilon}_v - \frac{\epsilon(v) - \epsilon_v}{2} = \frac{3\tilde{\epsilon}_v - \epsilon(v)}{2}
\]
and
\[ d(S_i, v) \geq \frac{\epsilon(v) - \epsilon_v}{2}. \]

We get
\[
\frac{\epsilon(v) - \tilde{\epsilon}_v}{2} \leq h_{\bar{i}}
\]
\[
\leq (2^{k-\bar{i}} - 1)(\Delta + M)
\]
\[
\leq (2^{k} - 1)(\Delta + M)
\]
\[
\leq (2^{k} - 1) \left( \frac{3\tilde{\epsilon}_v - \epsilon(v)}{2} + M \right).
\]

This inequality already gives us a lower bound on $\epsilon_v$. However, we obtain a better
bound treating the case $\bar{i} = 0$ separately.

When $\bar{i} = 0$, we have
\[ h_{\bar{i}} = h_0 \leq (2^k - 1)(\Delta + M). \]

Take a node $u$ such that $\epsilon(v) = d(v, u)$. As $h_0$ is the maximum distance from $Z_0 = S_0$, we have $d(S_0, u) \leq h_0$. Now take $x \in S_0$ such that $d(x, u) = d(S_0, u) \leq h_0$; we have
\[
\epsilon(v) = d(v, u) \\
\leq d(v, x) + d(x, u) \\
\leq \tilde{\epsilon}_v + h_0
\]
and

\[ \epsilon(v) - \bar{\epsilon}_v \leq h_0 \leq (2^k - 1)(\Delta + M). \]

When \( \bar{i} \geq 1 \) instead, we obtain

\[ \frac{\epsilon(v) - \bar{\epsilon}_v}{2} \leq (2^{k-\bar{i}} - 1)(\Delta + M) \]
\[ \leq (2^{k-1} - 1)(\Delta + M) \]
\[ \leq \frac{1}{2}(2^k - 1)(\Delta + M). \]

In either case, we have

\[ \epsilon(v) - \bar{\epsilon}_v \leq (2^k - 1)(\Delta + M) \]
\[ \leq (2^k - 1)\left(\frac{3\bar{\epsilon}_v - \epsilon(v)}{2} + M\right) \]

hence

\[ \bar{\epsilon}_v \geq \frac{2^k + 1}{3 \cdot 2^k - 1} \epsilon(v) - \frac{2^k - 1}{3 \cdot 2^k - 1} M. \]

Lemma 19, Lemma 20 and Lemma 21 show that the values \( \bar{D}, \bar{r} \) and \( \bar{\epsilon}_v \) are respectively \((2 - \frac{1}{2^k})\)-approximations of the diameter and the radius and \((3 - \frac{1}{2^{k+1}})\)-approximations of the eccentricities. They are obtained in \( O(n^{1/k} \log^{1/k} n \cdot C) \) expected time by Lemma 18. We summarize these results in the following theorem.

**Theorem 3.** For any \( k \geq 0 \), there is an algorithm that gives an almost \((2 - \frac{1}{2^k})\)-approximation of the diameter and the radius and an almost \((3 - \frac{1}{2^{k+1}})\)-approximation of all the eccentricities of an undirected graph in

\[ O(n^{1/k} \log^{1/k} n \cdot C) = \tilde{O}(mn^{1/k}) \]

expected time, where \( C = \tilde{O}(m) \) is the cost of a graph search. The additive terms are \( \frac{2^k - 1}{2^{k+1} - 1} M \) for diameter, \( \frac{2^k - 1}{2^{k+1} - 1} M \) for the radius and \( \frac{2^k - 1}{3 \cdot 2^{k-1}} M \) for the eccentricities, where \( M \) is the maximum weight of the edges.

**Corollary 3.** For arbitrarily small \( \varepsilon > 0 \), there is \( \delta > 0 \) such that in \( O(mn^{\varepsilon}) \) expected time it is possible to give an almost \((2 - \delta)\)-approximation of the diameter and the radius and an almost \((3 - \delta)\)-approximation of all the eccentricities of an undirected graph with additive error \(< M\).
Chapter 4

Distance dominating sets

In this chapter we explore the relation between distance $h$-dominating sets and the efficient approximation of the diameter. The notion of distance $h$-dominating sets [18, 19] is given in Definition 6. In the following we omit the word “distance” and simply say “$h$-dominating set”. First, we first observe that our sampling procedure described in Section 3.3 is able to produce $h$-dominating of small size, for some values of $h$ related to the diameter $D$. Second, we present a general technique to exploit $h$-dominating in order to obtain an approximation of the diameter. Third, we present some explicit constructions, which for some values of $D$ and $h$ produce families of graphs of diameter $D$, where every $h$-dominating set is asymptotically large: this constructions give an inherent limit to the efficiency achievable employing our general approach.

**Definition 6.** In a graph $G = (V, E)$, a subset of the nodes $X \subseteq V$ is a distance $h$-dominating set if

$$d(X, u) \leq h$$

for every node $u \in V$.

### 4.1 Sampling $h$-dominating sets

We start by showing that the algorithm presented in Section 3.3 relies on $h$-dominating sets.

**Lemma 22.** The set

$$X = S_0 \cup \cdots \cup S_k$$

generated by Algorithm 3.3 is a $\frac{2^k - 1}{2^k} (D + M)$-dominating set of size

$$|X| = O(n^{\frac{1}{1+\epsilon}} \log^{\frac{1}{1+\epsilon}} n).$$
CHAPTER 4. DISTANCE DOMINATING SETS

Proof. Let $v \in V$ be any node. By Lemma 17, we have

$$d(X, v) \leq d(S_i, v) \leq (2^{k-i} - 1)(\Delta + M)$$

where

$$\Delta = \max_{1 \leq i \leq k} \{d(w_i, v) - d(S_i, v)\},$$

$$\leq D - d(X, v)$$

as $d(w_i, v) \leq D$ by definition of diameter and $d(S_i, v) \geq d(X, v)$ as $S_i \subseteq X$. Thus,

$$d(X, v) \leq (2^k - 1)(D - d(X, v) + M)$$

from which we get

$$d(X, v) \leq \frac{2^k - 1}{2^k} (D + M).$$

The bound $|X| = O(n^{\frac{k}{k+1}} \log^{\frac{k}{k+1}} n)$ is given by Lemma 15.

This observation implicitly leads to the following purely combinatorial result.

**Theorem 4.** Every undirected unweighted graph admits a $2^{k-1}(D+1)$-dominating set of size $\tilde{O}(n^{\frac{1}{k+1}})$, for every integer $k \geq 0$.

**Proof.** It is a consequence of Lemma 22.

This is shown in Fig. 3, for $\lambda = \frac{2^k-1}{2^k}$, $\delta \leq 1$ and $\epsilon = \frac{1}{k+1}$.

### 4.2 Approximating the diameter

We now show our general approach to obtain an estimation of the diameter from an $h$-dominating set. First, we produce an upper bound $\overline{D}$ as described next.

**Proposition 1.** Given a $h$-dominating set $X \subseteq V$, it is possible to produce an upper bound $\overline{D}$ of the diameter $D$ such that

$$D \leq \overline{D} \leq D + h$$

in $O(|X| \cdot C)$ time.
**Chapter 4. Distance Dominating Sets**

**Proof.** Compute the value

\[ h_0 = \max_{u \in V} d(X, u) \]

with a multiple-source graph search from \( X \), taking \( O(C) \) time. As \( X \) is a \( h \)-dominating set, we have \( h_0 \leq h \).\(^1\) Then, compute

\[ D_0 = \max_{x \in X} \epsilon(x) \leq D \]

in \( O(|X| \cdot C) \) time running a graph search from each \( x \in X \). Return the value

\[ \overline{D} = D_0 + h_0 \leq D + h. \]

To prove that \( D \leq \overline{D} \), take \( v \in V \) and \( x \in X \) such that \( \epsilon(v) = D \) and \( d(x, v) = d(X, v) \leq h_0 \). By Lemma 1, we have \( \epsilon(v) \leq \epsilon(x) + d(x, v) \). Hence we obtain

\[
D = \epsilon(v) \\
\leq \epsilon(x) + d(x, v) \\
\leq D_0 + h_0 = \overline{D}
\]

completing our proof. \( \square \)

If \( h = \lambda(D + \delta) \) for some known constants \( \lambda \) and \( \delta \), then the upper bound \( \overline{D} \) obtained with Proposition 1 can be transformed into a \((\lambda + 1)\)-multiplicative additive approximation \( \tilde{D} \). It is sufficient to choose as estimation of the diameter the value

\[
\tilde{D} = \frac{\overline{D} - \lambda \delta}{\lambda + 1}.
\]

Indeed, we have

\[
\tilde{D} = \frac{\overline{D} - \lambda \delta}{\lambda + 1} \\
\leq \frac{D + h - \lambda \delta}{\lambda + 1} \\
= \frac{D + \lambda(D + \delta) - \lambda \delta}{\lambda + 1} \\
= \frac{(\lambda + 1)D}{\lambda + 1} = D
\]

and

\[
\tilde{D} \geq \frac{D - \lambda \delta}{\lambda + 1} \\
= \frac{1}{\lambda + 1}D - \frac{\lambda \delta}{\lambda + 1}
\]

\(^1\)In fact, here we discover that \( X \) is an \( h_0 \)-dominating set, but in general it can be \( h_0 < h \).
as required by the definition of $\alpha$-multiplicative $\beta$-additive approximation. In particular, applying Proposition 1 to the $\frac{2^{k-1}}{2^k}(D+M)$-dominating set $X$ produced by Algorithm 3.3 in $O(C)$ expected time (by Lemma 22), we obtain an alternative proof of our bounds for diameter approximation. Indeed, the multiplicative factor is

$$\lambda + 1 = \frac{2^k - 1}{2^k} + 1 = \frac{2^k - 1 + 2^k}{2^k} = 2 - \frac{1}{2^k}$$

and the additive term is

$$\frac{\lambda \delta}{\lambda + 1} = \frac{2^{k-1} M}{2^k - 1 + 2^k} = \frac{2^k - 1}{2^k - 1 + 2^k} M = \frac{2^k - 1}{2^k + 1 - 1} M.$$

Finally, the total expected time is

$$O(C) + O(|X| \cdot C) = O(C + n^{\frac{1}{k+1}} \log^{\frac{k}{k+1}} n \cdot C) = O(n^{\frac{1}{k+1}} \log^{\frac{k}{k+1}} n \cdot C) = \tilde{O}(mn^{\frac{1}{k+1}})$$

matching the bounds stated in Lemma 18 and Lemma 19.

### 4.3 Lower bounds

We presented a general algorithmic approach to approximate the diameter by finding small-size $h$-dominating sets. To better understand the properties of this approach and its limitations, it is natural to provide lower bounds on the size of these sets in the worst case and in relation to the diameter $D$.

The specific case $h = 1$ and $D = 2$ has been already studied in the literature since a distance 1-dominating set is a classical dominating set. Desormeaux et al. [11] prove that in undirected graphs of diameter 2 the smallest 1-dominating set has size $\Theta(\sqrt{n \log n})$ in the worst case.\(^2\) We provide several lower bounds, with focus on the coefficients $\lambda$ and $\delta$ that relate $h$ and $D$ as $h = \lambda(D + \delta)$. Our bounds are illustrated as shaded zones in Fig. 3.

We first obtain a lower bound from the gadget graph $T^k[B]$ defined in Section 2.3.

\(^2\)Desormeaux et al. [11] consider total dominating sets. On graphs without isolated nodes, a dominating set of size $t$ can be transformed into a total dominating set of size $2t$ [19], hence their $\Omega(\sqrt{n \log n})$ lower bound still holds.
Chapter 4. Distance Dominating Sets

Theorem 5. For every $D \geq 1$ there exists a family of undirected, unweighted graphs of diameter $D$ and number of nodes $n \to \infty$ where any $h$-dominating set for

$$h = \lceil D/2 - 1 \rceil$$

has size $\Theta(n)$.

Proof. Consider the gadget graph $T^t[B]$ on a set $B$ of size $s \to \infty$. The number of nodes in is $n = \Theta(s)$ and the diameter is $D = t$ by Lemma 3. Consider any $h$-dominating set $X$ for $h = \lfloor t/2 - 1 \rfloor$: it must contain at least one node per branch, in order to be at distance $\leq h$ from each tip $u \in B$. Hence, $|X| \geq s = \Theta(n)$. □

Corollary 4. For every $\lambda < 1/2$ and $\delta = O(1)$, there exists a family of graphs of diameter $D$ and number of nodes $n \to \infty$ where any $h$-dominating set for $h = \lambda(D + \delta)$ has size $\Theta(n)$.

Proof. For $\lambda < 1/2$ and $\delta = O(1)$, we can pick a large enough $D$ such that

$$h = \lambda(D + \delta) \leq \lceil D/2 - 1 \rceil.$$  □

We now provide a family of constructions for every integer $\ell \geq 2$. Each construction for a fixed value of $\ell$ produces a family of graphs $G_t^\ell$ parametrized by an integer $t \geq \ell$. The graph $G_t^\ell$ contains $n = t^{\ell}$ nodes and has diameter

$$D = \frac{\ell(\ell + 1)}{2}.$$  

Moreover, we prove that every $h$-dominating set for $h < D$ is of size $\Omega(t)$.

Construction $G_t^2$. For simplicity, we first describe the construction in the special case $\ell = 2$. We then generalize the construction for other values of $\ell$ and rigorously prove our claims.

The node set of $G_t^2$ contains the pairs of natural numbers

$$xy \in \{1, \ldots, t\}^2.$$  

For any node $ab \in \{1, \ldots, t\}^2$, there is a “write” edge

$$ab \xrightarrow{w} xb$$

for any $x \in \{1, \ldots, t\}$, and a “swap” edge

$$ab \xrightarrow{s} ba.$$
Note that for any edge $xy \to x'y'$ there is an edge $x'y' \to xy$, hence the graph is undirected. The diameter is at most 3 as for any two nodes $ab$ and $xy$ there is a path

$$ab \xrightarrow{W} ya \xrightarrow{S} ay \xrightarrow{W} xy.$$ 

Any path starting from a node $ab \in \{1, \ldots, t\}^2$ matches the following pattern

$$\varepsilon \varepsilon \rightarrow \delta \varepsilon \rightarrow \delta \varepsilon | \varepsilon \delta \rightarrow \delta \delta \rightarrow \cdots$$

where $\varepsilon := a|b$ and $\delta := 1|\ldots|t$. Hence, the nodes at distance 2 (or less) from any node $ab$ match the pattern $\delta \varepsilon | \varepsilon \delta$, thus their number is $O(t)$. This implies that any $2$-dominating set of $G_n^2$ needs $\Omega(t) = \Omega(\sqrt{n})$ nodes to reach all the $n = t^2$ nodes in the graph.

**A family of constructions $G_i^\ell$.** We now fully describe our family of constructions (thus a family of families of graphs). For a given $\ell \geq 2$, we define the graphs $G_i^\ell$ parametrized by $t \geq \ell$. The node set of $G_i^\ell$ contains the sequences of $\ell$ coordinates over the natural numbers $\{1, \ldots, t\}$:

$$x_1 \ldots x_\ell \in \{1, \ldots, t\}^\ell.$$ 

For a node $a_1 \cdots a_\ell$ and any $x \in \{1, \ldots, t\}$, we have a “write” edge that writes the first coordinate:

$$a_1 a_2 \cdots a_\ell \xrightarrow{W} xa_2 \cdots a_\ell.$$ 

Then, for $i \in \{1, \ldots, \ell - 1\}$, we have an edge which swaps the coordinates $i$ and $i + 1$, thus for a node $a_1 \cdots a_\ell$ we have

$$a_1 \cdots a_i a_{i+1} \cdots a_\ell \xrightarrow{S_i} a_1 \cdots a_{i+1} a_i \cdots a_\ell.$$ 

The resulting graph is clearly undirected.

**Lemma 23.** The diameter of $G_i^\ell$ is at most

$$\frac{\ell(\ell + 1)}{2}.$$ 

**Proof.** Let $a_1 \cdots a_\ell$ and $x_1 \cdots x_\ell$ be any two nodes. For every $j \in \{1, \ldots, \ell\}$ there is a path

$$x_{\ell-j+2} \cdots x_\ell a_j \cdots a_\ell \xrightarrow{S_{j-1}} \cdots x_{\ell-j+2} \cdots x_\ell a_j \cdots a_\ell \xrightarrow{W} x_{\ell-j+1} \cdots x_\ell a_j a_{j+1} \cdots a_\ell$$

of length $j$. Concatenating these paths we get

$$a_1 \cdots a_\ell \xrightarrow{\cdots} x_1 a_2 \cdots a_\ell \xrightarrow{\cdots} x_{\ell-j+2} \cdots x_\ell a_j \cdots a_\ell \xrightarrow{\cdots} x_1 \cdots x_\ell.$$
which is a path from \( a_1 \ldots a_\ell \) to \( x_1 \ldots x_\ell \) of length

\[
\sum_{j=1}^{\ell} j = \frac{\ell(\ell + 1)}{2}
\]

Lemma 24. In the graph \( G_\ell^t \), the nodes at distance \( \leq p \) from any given node are \( O(t^q) \), where \( q \) is the largest integer such that

\[
\frac{q(q+1)}{2} \leq p.
\]

Proof. Let \( \bar{a} = a_1 \ldots a_\ell \) be a given node. Consider any node \( \bar{x} = x_1 \ldots x_\ell \) and define \( b_i = 0 \) if \( x_i \in \{a_1, \ldots, a_\ell\} \) and \( b_i = 1 \) otherwise. We define the size of \( \bar{x} \) as

\[
s(\bar{x}) = \sum_{i=1}^{\ell} b_i.
\]

We give a weight \( i \) to the \( i \)-th coordinate, and define the potential of \( \bar{x} \) as

\[
p(\bar{x}) = \sum_{i=1}^{\ell} i \cdot b_i.
\]

The following facts hold.

1. \( p(\bar{x}) \leq d(\bar{a}, \bar{x}) \).

   By induction, \( p(\bar{a}) = 0 \) and \( \bar{x} \rightarrow \bar{y} \) implies \( p(\bar{y}) \leq p(\bar{x}) + 1 \). Indeed, if \( \bar{x} \xrightarrow{w} \bar{y} \), then only \( b_1 \) can change and it has weight 1. If \( \bar{x} \xrightarrow{s_i} \bar{y} \) instead, then the values of \( b_i \) and \( b_{i+1} \) are swapped (and the others do not change) and their weights differ by one unit only.

2. \( s(\bar{x}) \leq q \), where \( q \) is the largest integer such that \( \frac{q(q+1)}{2} \leq p(\bar{x}) \).

   Fixed a maximum potential \( p \), the largest size \( q \) is obtained by the greedy choice \( b_1, \ldots, b_q = 1 \) and \( b_{q+1}, \ldots, b_\ell = 0 \), where the potential is

\[
\sum_{j=1}^{q} j = \frac{q(q+1)}{2} \leq p.
\]
Thus, the nodes at distance at most \( p \) from a given node \( \bar{a} \) have size at most \( q \), and their number is at most
\[
\sum_{s=0}^{q} \binom{\ell}{s} \cdot (t - \ell)^s \cdot \ell^{\ell - s} = O(t^n).
\]

Indeed, the value \( s \) varies over the possible values of \( s(\bar{x}) \); the term \( \binom{\ell}{s} \) comes from the choice of which \( b_i \) are set to 1; the terms \( (t - \ell)^s \) and \( \ell^{\ell - s} \) come from the choice of the values for the coordinates \( x_i \) where \( b_i = 1 \) and \( b_i = 0 \), respectively.

**Theorem 6.** For every \( \ell \geq 2 \) there exists a family of undirected, unweighted graphs of diameter
\[
D = \frac{\ell(\ell + 1)}{2}
\]
and number of nodes \( n \to \infty \) where any \( h \)-dominating set for \( h < D \) has size \( \Omega(n^{1/\ell}) \).

**Proof.** Consider the graphs \( G_t^{\ell} \) for \( t \to \infty \). First, the number of nodes is \( n = t^\ell \) by construction. Second, the diameter is at most \( \frac{\ell(\ell + 1)}{2} \) by Lemma 23. Third, by Lemma 24 there are at most \( O(t^{\ell - 1}) \) nodes at distance strictly less than \( \frac{\ell(\ell + 1)}{2} \) from any given node. For large enough \( t \), this number is less than the total number of nodes \( t^\ell \): hence there are some nodes in the graph at distance at least \( \frac{\ell(\ell + 1)}{2} \) from any given node, and in particular the diameter is exactly \( D = \frac{\ell(\ell + 1)}{2} \).

Since the nodes at distance \( h < D \) from any given node are only \( O(t^{\ell - 1}) \), we need at least \( \Omega(t) \) different nodes to reach at distance \( h \) all the \( t^\ell \) nodes in the graph. Hence, any \( h \)-dominating set for \( h < D \) has size \( \Omega(t) = \Omega(n^{1/\ell}) \).

**Corollary 5.** For \( \ell \geq 2 \) and
\[
\lambda < 1 - \frac{2}{\ell(\ell + 1) + 2},
\]
there exists a family of graphs of diameter \( D \) and number of nodes \( n \to \infty \) where any \( h \)-dominating set for \( h = \lambda(D + 1) \) has size \( \Omega(n^{1/\ell}) \).

**Proof.** Observe that
\[
\lambda < 1 - \frac{2}{\ell(\ell + 1) + 2}
= \frac{\ell(\ell + 1)}{\ell(\ell + 1) + 2}
= \frac{\ell(\ell + 1)}{\ell(\ell + 1) + 2}
= \frac{\ell(\ell + 1)}{2} + \frac{1}{2} + \frac{1}{2}
= \frac{D}{D + 1}
\]
and
\[
\begin{align*}
h &= \lambda D + \lambda \\
&< \frac{D}{D + 1} D + \frac{D}{D + 1} \\
&= (D + 1) \frac{D}{D + 1} = D.
\end{align*}
\]

Hence, the family $G_t^\ell$ is such that every $h$-dominating set has size $\Omega(t) = \Omega(n^{1/t})$. \qed
Conclusions

In this work we presented in a unified setting some of the recent results on the problem of approximating extremal distances in undirected graphs, and we developed novel techniques that yield new bounds on the possibility of efficient algorithms. On one hand, we tightened the lower bounds deriving from the SETH conjecture; on the other hand, we obtained a family of new upper bounds through algorithms based on an iterative sampling procedure.

To obtain our lower bounds, we employed a construction parametrized by an integer \( t \geq 1 \) that transforms an instance of \( k \)-SAT into an undirected unweighted graph, which has diameter \( 3t \) if the instance is satisfiable and \( 2t \) otherwise. Similarly, for \( t = 2z \), the maximum eccentricity in a specific subset of the nodes is \( 5z \) if the instance is satisfiable and \( 3z \) otherwise. With this reduction we show that if there is a truly subquadratic-time algorithm that gives an \( \alpha \)-approximation of the diameter for \( \alpha < 3/2 \), or an \( \alpha \)-approximation of all the eccentricities for \( \alpha < 5/3 \), then the widely believed conjecture SETH is false. Thanks to the arbitrariness of the parameter \( t \), this bound holds even if there is a constant additive term \( \beta = O(1) \). If the additive term \( \beta \) is not constant but grows as \( \beta \sim m^\delta \) for \( 0 \leq \delta < 1 \), where \( m \) is the number of edges, then it is impossible to get \( O(m^{2-2\delta-\epsilon}) \) time. Our construction captures and unifies the previous bounds for both multiplicative and additive approximations of the diameter, and also gives a tight bound for the approximation of the eccentricities.

Our new upper bounds are obtained through a family of randomized algorithms parametrized by a constant \( k \geq 0 \), which run in \( O(mn^{1/k+1}) \) expected time and give an \( \alpha \)-approximations of the diameter and the radius, for \( \alpha = 2 - 1/2^k \) and of all the eccentricities for \( \alpha = 3 - 4/(2^k+1) \). At the core of these algorithms there is a sampling procedure which selects \( \sqrt[2k+1]{n} \) nodes in the graph: we obtain our approximations by launching a sequence of graph searches from each of these nodes. The crucial tool in this sampling procedure is an iterative process, which at each step receives in input a zone in the graph with few nodes but large distances, and gives either a small hitting set, near to all the nodes in the zone and transitively to the rest of the graph, or a new zone with fewer nodes to use for the next iteration. For \( k \geq 2 \), this technique yields new approximation bounds,
performing a smaller number of graph searches.

We have shown that our sampling procedure finds $h$-dominating sets of small size, where $h$ is a constant fraction of the diameter. Indeed, in the case of the diameter, our algorithm can be thought of as an instance of a general approach, that exploits small $h$-dominating sets to efficiently provide upper bounds to the diameter, and hence an estimation of its value. This discovery leads to new questions concerning the smallest possible size of $h$-dominating sets in the worst case, in graphs of a fixed diameter $D$: indeed, the minimum size of $h$-dominating sets is an inherent lower bound to the efficiency of any algorithm hinging on this general approach. For some values of $h$ and $D$, we proved that our algorithm obtains $h$-dominating sets of optimal size, up to logarithmic factors. For other values of $h$ and $D$, the upper bounds implicitly given by our algorithm and the lower bound arising from our explicit constructions are still separated by a polynomial gap: we leave the study of these bounds as an open problem.

The connection between $h$-dominating sets and the approximation of the diameter is a clear example of the close relationship between purely graph-theoretic problems and the development of efficient algorithms on graphs. In this case, the search for new algorithms for the approximation of extremal distance leads to the discovery of a more general combinatorial property, regarding the existence of small $h$-dominating sets. Finding $h$-dominating sets of small size can be useful also in other contexts: for example, the nodes in an $h$-dominating set can be good places where to place facilities, as there is always at least one of them close to any given point in the graph. With our work, we highlighted a previously unknown connection between the existence of small $h$-dominating sets and the value of the diameter $D$: however, this connection still has to be studied in order to obtain tighter and more general bounds.

Some interesting questions concern directed graphs. While previous work on diameter approximation did not distinguish between directed and undirected graphs, sharing the same algorithmic techniques, here our results are tailored for undirected graphs (except for our lower bound on the diameter which also holds for directed graphs). It is important to establish whether our algorithms have a counterpart working on directed graphs as well, or if they constitute a separation point between directed and undirected graphs for what concerns the approximation of the diameter.

Finally, we ask ourselves whether our new techniques could be applied also to other problems, starting from the other variants of extremal distances which have been defined, especially on directed graphs, the approximation of APSP, and related problems on graphs.
Bibliography


