Exploring small expander graphs

Maya Afshar, Alexis Enston, George Pirlea and Jas Semrl
University College London

***Abstract*—Expanders are graphs that are sparse, yet highly connected. They have a wide range of applications in networking, coding theory, cryptography, and complexity theory. Theoretical constructions of vertex expanders do exist, but are complex and not optimal in parameters. Through an exploration of small unique-neighbour bipartite expanders, we find that the absolute performance metric k increases as graphs become larger, but the size-adjusted performance metric $k/|L|$ decreases exponentially. Graphs with higher degree have better size-adjusted performance, but exhibit similar exponential decrease. By exhaustive search, we find that certain values of absolute performance k are unattainable for given size-degree pairs, and only increasing the degree makes them attainable. For the exploration, we develop algorithms to determine the performance-metric k for given graphs with $O(2^{|L|})$ worst-case performance and good average-case performance.**

Finally, we present two constructions (*iterative* and *geometric*) of unique-neighbour bipartite expanders based on guided random search, with performance metrics better than random graphs. The geometric construction exploits a novel correspondence between objects in n-dimensional spaces and bipartite graphs to obtain large graphs with good k.

I. Introduction

Expanders are graphs that are sparse, yet highly connected. Intuitively, they have $O(n)$ edges but have the connectivity properties of graphs with $O(n^2)$ edges. Surprisingly, such graphs do exist and are in fact common [1].

Furthermore, expanders have wide-ranging applications, and have been used to build robust communication networks [2], design efficient error-correcting codes [3], pseudorandom number generators [4] and hash functions [5], and prove important results in complexity theory [6].

Fundamentally, expansion is a graph connectivity property, and it can be defined in different ways, depending on the use for which the graph is intended. Robust communication networks, for example, often require *edge expansion* — every set of vertices needs to have many outgoing edges. Other applications, such as linear codes or routing algorithms, rely on *vertex expansion* — all sets of vertices need to have many external neighbours.

In this work, we only consider vertex expansion. In particular, we look at bipartite vertex expanders with the unique-neighbour property, as seen in definition I.1.

Intuitively, having unique-neighbours is desirable because their presence to a certain degree guarantees expansion. Consider the following: starting from an arbitrary set of right-vertices $S \subseteq R$, if we follow an edge into a unique-neighbour $v \in N(S)$, and then follow any other edge that goes out of v, we are *guaranteed* to reach a right-vertex outside of our original S. A k-unique neighbour expander provides this guarantee for *every* $S \subseteq R$ that has fewer than k neighbours.

Definition I.1 (k-unique neighbour bipartite expander)**.** *A bipartite graph $G = (L, R, E)$ is a k-unique neighbour expander iff:*

$$\forall S \subseteq R, |N(S)| < k, \exists v \in N(S), |N(v) \cap S| = 1$$

where N is the neighbour function:

$$N(S) = \{y \mid x \in S \text{ and } (x, y) \in E\}$$
$$N(x) = \{y \mid (x, y) \in E\}$$

Importantly, given a set S, determining which of its neighbours is an unique-neighbour can be done using only local information, i.e. using only knowledge of the edges coming out of S. This property of locality, for example, is crucial for the design of efficient routing algorithms.

Unique-neighbour expanders are desirable for many applications. Whilst theoretical constructions of expanders exist, they are complex and not optimal in parameters. In this project, we explore the space of small graphs in search of unique-neighbour expanders with a high value of k, hoping that insights at small scale might point towards methods to construct large expanders with good parameters. We find that the absolute performance metric k increases as graphs become larger, but the size-adjusted performance metric $k/|L|$ decreases exponentially, and this behaviour is exhibited regardless of the degree of the graph — which suggests that finding large graphs with high k and low d via random search is infeasible. The results of our exploration are described fully in section III. Efficient algorithms to determine the highest value of k a graph admits are crucial for the search.

Given a graph, we want to be able to determine the *largest* value of k for which the graph is a k-unique neighbour expander. Directly using definition I.1 yields an algorithm to find k by enumerating over all subsets of R. However, this is inefficient for most of the graphs we are interested in, which have $|R| > |L|$. We overcome this inefficiency by coming up with an alternative but equivalent definition, which we discuss in section IV.

Finally, we develop two constructions (*iterative* and *geometric*) of unique-neighbour bipartite expanders based on guided random search, which we discuss in sections V and VI, respectively. In particular, the geometric construction is based on a novel correspondence between objects in n-dimensional spaces and bipartite graphs, and performs a guided search on these objects. Both constructions generate graphs with better values of k than those exhibited by random graphs. These constructions are compared in section VII.

II. Related work

Throughout the years, many methods have been developed to *identify* and *construct* expanders.

Identification of expansion was originally suggested by Alon [7]. He claimed that randomly generating a graph and applying his method to that graph could give a much better result than the state of the art construction of that time (Margulis's construction [8]). Alon's theorem suggests that if the *second largest eigenvalue* of the adjacency matrix of a graph is close to zero, the graph has good expansion. A number of extensions were added later to the second largest eigenvalue method to improve its efficiency and applicability.

Czumaj and Sohler [9] proposed an algorithm that performs a number of random walks over the graph and counts the number of collisions of the walks as a measurement of expansion. This is a simple method, but can lead to false positive results, thus it is a non-deterministic method of proving expansion. Blum *et al.* [10] proved that deterministically identifying whether or not a graph is an expander is a coNP-complete problem.

The first expander graph construction algorithm was introduced by Margulis [8]. This work was later on expanded by Angluin [11] and Gabber and Galil [12]. These constructions are *algebraic constructions*, as they produce graphs from the structure of an algebraic group. Ajtai proposed a new recursive, *combinatorial* method [13]. Combinatorial constructions rely on operations that combine graphs (graph products), to produce large expanders from smaller expanders. After Ajtai, Reingold *et al.* developed the zig-zag product [14], an improved graph product that preserves expansion properties.

Relatively little work has been dedicated to studying unique-neighbour expansion, both in terms of quantifying such expansion and constructing unique-neighbour expanders.

The tools for identifying expansion in a graph are not powerful enough to guarantee unique-neighbour expansion. Unique-neighbour expanders are a strict subset of vertex expanders [15], meaning not all vertex expanders are necessarily unique-neighbour expanders. Neither the second largest eigenvalue method, nor the random walk method can identify unique-neighbour expanders.

Alon and Capalbo [16] present a construction algorithm for slightly-unbalanced bipartite unique-neighbour expanders. However, this construction only produces graphs with $|L| = 21/22 \times |R|$, whereas some applications require $|R|$ to be significantly larger than $|L|$.

Capalbo *et al.* have shown that *lossless expanders* are also unique-neighbour expanders [17], although not necessarily good ones. Ta-Shma [18], Alon [16], and Guruswami [19] have each proposed constructions of lossless bipartite expanders, both in the balanced and unbalanced case, and Guruswami's construction produces graphs with parameters polynomially close to the optimum.

The definition of unique-neighbour expansion we use in this report, definition I.1, differs slightly from the usual definition in the literature, given below.

Definition II.1. *((α, ϵ)-Unique Neighbour Bipartite Expander) A bipartite graph $G = (L, R, E)$ is a (α, ϵ)-unique neighbour bipartite expander iff for some $\alpha, \epsilon > 0$:*

$$\forall S \subseteq R, |S| < \alpha|R|, |N_{unique}(S)| \geq \epsilon|S|$$

where

$$N_{unique}(S) = \{v \in N(S), |N(v) \cap S| = 1\}$$

While definition I.1 and definition II.1 differ, they are closely related. Our definition requires each set with a 'not too large' neighbour set (i.e. $|N(S)| < k$) to have at least one unique neighbour, whereas the standard definition measures the size of the originating set (i.e. $|S| < \alpha|R|$) and requires at least a constant fraction of unique neighbours. We motivate this difference by noting that our definition gives a direct measure of the kind of performance we are interested in, namely size of neighbour-sets that admit unique neighbours.

III. Random Graph Exploration

One part of our project was to explore the space of small graphs in search of expanders, with the hope that insights from small expanders might help us find a way to efficiently construct good expanders of larger size, where random search ceases to be effective. In the following, we describe our process for exploring small expander graphs and detail some key findings. We were particularly interested in answering the following questions:

1) How do the dimensions of the graph affect the expected value of k?
2) In regular graphs, how does the degree of the graph affect expected k?
3) Does the distribution of k differ between regular and biregular graphs respectively? In other words, does "more regularity" correlate with larger expected values of k?
4) Do small expanders with high values of k have any distinguishable features that we can discern and perhaps extrapolate to larger graphs?

Our method for answering these questions is straightforward: we generate large numbers of random graphs and for each of them compute the value of k. However, there are two issues with this method. Firstly, it is constrained by the speed of the algorithm we use to determine the value of k. Section IV details how we significantly improved the average-case performance of determining k for given graphs in order to make this exploration feasible for small graphs. Secondly, as the sizes of the graphs we explore increase, the number of possible graphs of that size grows *exponentially*. Thus, even if computing k were a constant-time operation, if we wanted to get statistically accurate expected values of k, we would have to generate hundreds of trillions of graphs. Despite these limitations, our straightforward method does give us answers, at least for small graphs.

If we keep the ratio $r = |R|/|L|$ and the degree d constant, we find that as we increase the size of the graph, the expected value of k also increases, albeit slightly. If we then start increasing the degree, whilst still keeping r constant, we find

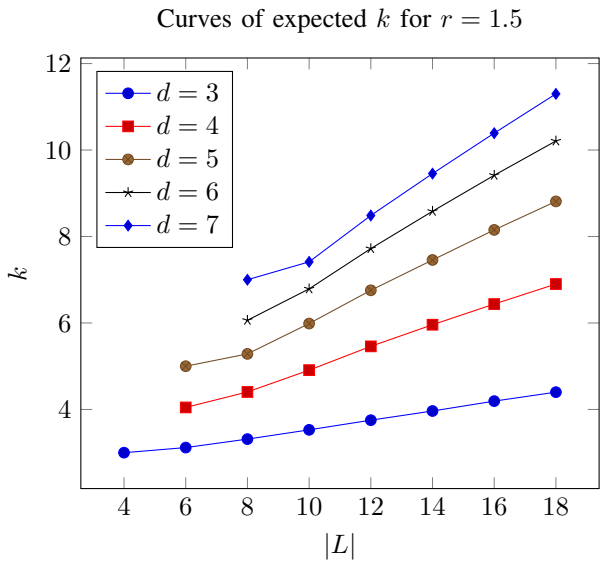


Fig. 1. The expected value of k increases as $|L|$ increases if we keep $r = |R|/|L|$. Note that curves with a higher d increase at a faster rate. Obtained by sampling 50,000 random d-regular bipartite graphs for each point on each curve.

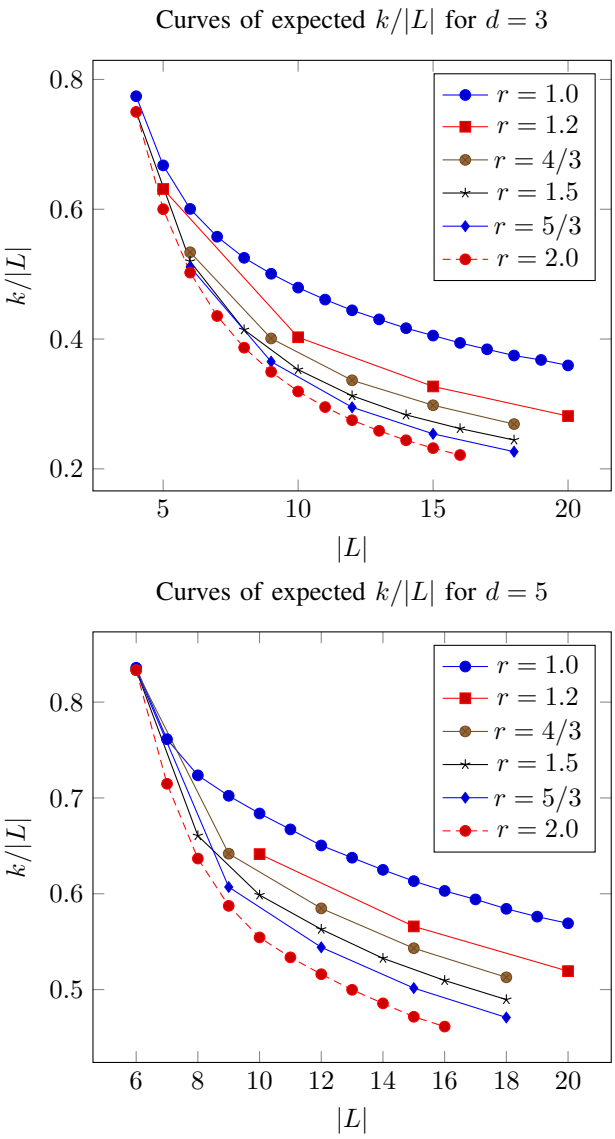


Fig. 2. Expected value of $k/|L|$ in terms of $|L|$ obtained by generating 50,000 random d-regular bipartite graphs with $r = |R|/|L|$ and $d \in \{3, 5\}$. Notice that the scale for the y-axis is different in the two plots — a higher degree moves all the curves upwards (better k), but preserves their shape.

that the rate of increase of the expected value of k increases as well. This is shown in figure 1, where all curves are increasing, and higher curves have steeper inclinations.

The fact that larger graphs have higher expected values of k may seem surprising, but it is in fact natural. Small graphs are prevented by their size from having large values of k. As the graph size increases, higher values of k become possible and start appearing in samples, slowly shifting the expected value upwards. Similarly, having a higher degree gives you the chance to exhibit higher values of k more often, which explains both the fact that curves shift upwards with higher d and that higher curves have steeper inclinations.

If, however, we look at the ratio between expected k and $|L|$, the picture is very different. Expected $k/|L|$ decreases exponentially as the size of the graph increases, as can be seen in figure 2. Moreover, the shape of the curves for $k/|L|$ is similar for different degrees if r is kept constant. This suggests that graphs with values of k close to $|L|$ are rare, and become exceedingly rare (and possibly non-existent) as we start looking at larger graphs.

The fact that higher degrees only shift the curve upwards, without significantly altering its shape (as seen in figure 3) indicates that there is little hope for obtaining large random graphs with very good k — if we want a random graph with k close to $|L|$, we either have to make d close to $|L|$ or generate a very, very large number of graphs and hope that we hit an extreme outlier. This can also be seen in figure 4, which shows the full distribution of k obtained from a sample of 50,000 random graphs of size 15 by 20 — graphs with high k become exceedingly rare (or possibly non-existent), and we can only obtain a higher k by increasing d.

We know for a fact that extreme outliers do not always exist. For example, we have exhaustively enumerated all graphs of degree $d = 3$ with $|L| = 6$ and $|R| = 12$, and found that $k = 4$ is the largest for graphs with these parameters. However, we know that graphs of the same size with $d = 4$ can have $k = 5$. This proves that some high values of k are not achievable for some small degrees, though we do not know the exact nature of the relationship between graph size, degree, and maximum attainable k or $k/|L|$. Plotting the maximum value of $k/|L|$ we obtain after generating 50,000 d-regular random graphs, as seen in figure 5, yields no discernible pattern. If we were to keep r fixed and change d, we would see the same curve-shifting behaviour seen in figure 3.

Interestingly, we find that biregular graphs have better distributions of k than regular graphs, and thus better expected values of k. However, they have the same maximum value of k in a sample of 50,000 random graphs. This is shown in figure 6. More structure and regularity within the graphs we generate tends to correlate with better expected values of k.

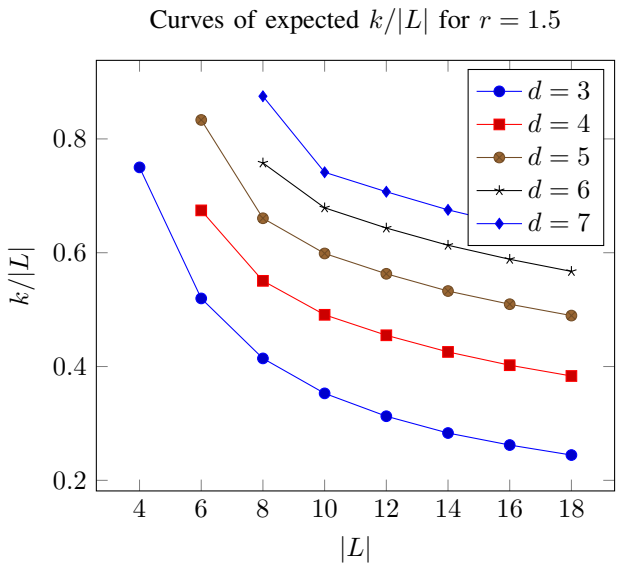


Fig. 3. Keeping the size of graph constant, increasing d shifts the $k/|L|$ curve upwards, i.e. towards better values of k.

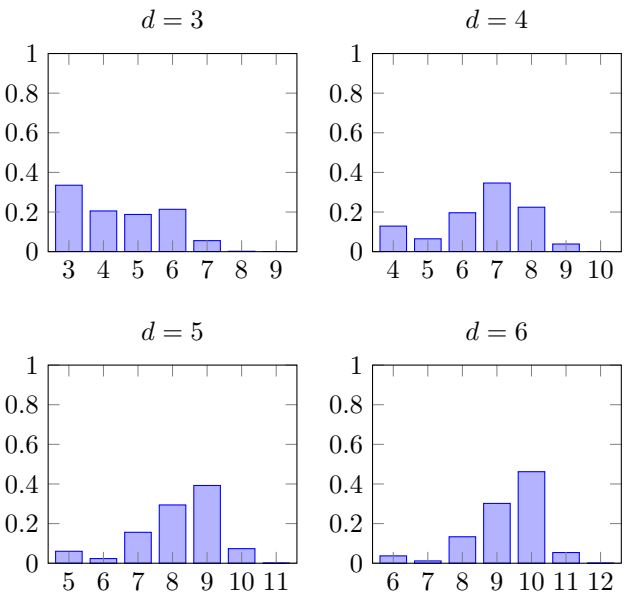


Fig. 4. Distribution of k obtained by generating 50,000 random d-regular bipartite graphs with $|L| = 15$ and $|R| = 20$. All values of k shown on the x-axis were found in the sample, but some of them appear only with very small probability (hence the seemingly non-existent bars).

Despite our initial hopes, manually examining small graphs with high k yields no discernible features that could be extrapolated to obtain larger graphs with high k. Nonetheless, our exploration of small graphs has helped us uncover some interesting facts about unique-neighbour expanders:

1) Larger graphs (larger $|L|$ and fixed r) have higher expected values of k, but lower expected values of $k/|L|$.
2) Increasing the degree moves the $k/|L|$ curve upwards and shifts distributions towards higher values of k.
3) Increasing the degree not only increases the expected value of k, but also increases the rate of increase of the expected value of k.
4) Biregular graphs have better distributions of k than regular graphs, but not better maximum values of k.
5) Some high values of k are not attainable for graphs with low degrees, but we do not know the exact relationship.

IV. Computing k

We explored two different approaches for computing k for given graphs, based around subsets of either L or R. They have different time complexities, and the best method to use depends on the shape of the graph.

A. R-subset checker

The value of k can be computed by enumerating all possible subsets of R. This method is suited for graphs where $|L|$ and $|R|$ are of similar size.

The value of k returned is the same no matter which order the sets are enumerated in. In addition to this, some subsets can be skipped whilst checking. To do so, a breadth-first search can be done on the subsets of R in order to enumerate them in increasing size. Once a set $F \subseteq R$ without a unique neighbour has been found, only sets where $|N(S)| < |N(F)|$ must be checked.

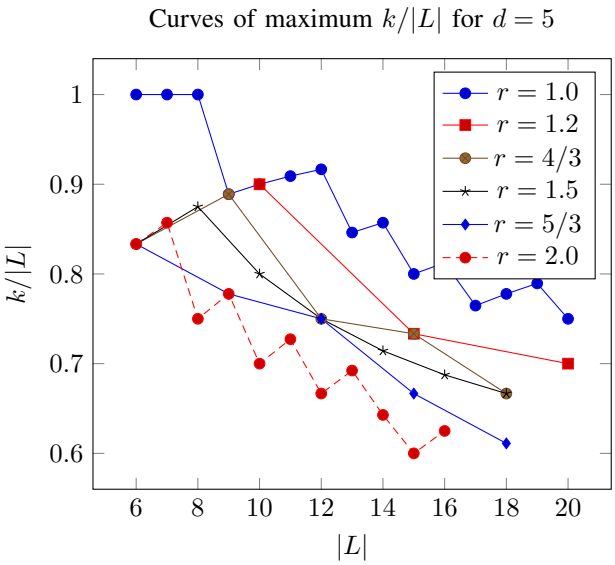


Fig. 5. Maximum value of $k/|L|$ in terms of $|L|$ obtained by generating 50,000 random d-regular bipartite graphs with $r = |R|/|L|$. No pattern can be discerned. Note that higher values of k might exist for graphs of these sizes — our search is not exhaustive, and we cannot tell whether we reached the optimum in 50,000 samples.

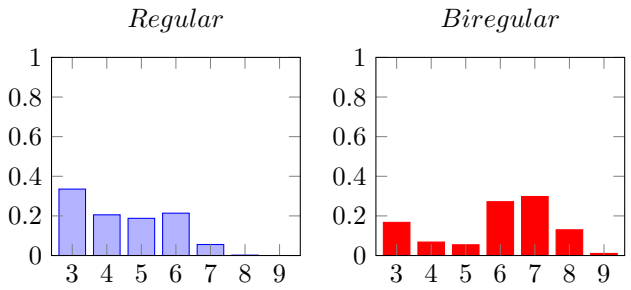


Fig. 6. Distribution of k obtained by generating 50,000 random d-regular and respectively, biregular bipartite graphs with $|L| = 15$ and $|R| = 20$. All values of k shown on the x-axis were found in the sample. Notice that biregularity shifts the distribution towards higher values of k.

```
1: function R-ComputeK(G = (L, R, E))
2:     k ← ∞
3:     for all S ⊆ R do
4:         if (|N(S)| < k) ∧ ¬(∃v ∈ N(S), |N(v) ∩ S| = 1)
   then
5:             k ← |N(S)|
6:         end if
7:     end for
8:     return k
9: end function
```

The worst case run-time for this algorithm is $O(2^{|R|})$. More specifically, if enumerating the sets in order of increasing size, the time taken on average depends on the vertex expansion of the graph itself and the value of k:

$$\sum_{i=2}^{n} \left(\frac{|R|!}{i!(|R|-i)!} \times i \times |L| \right)$$

where $n = \max |N(S)|, S \subseteq L, |S| \leq k$.

B. L-subset checker

Definition I.1 can be redefined in terms of subsets of L:

Definition IV.1 (k-unique neighbour bipartite expander)**.** *A bipartite graph $G = (L, R, E)$ is a k-unique neighbour bipartite expander iff:*

$$\forall S \subseteq L, |S| < k,$$
$$(\exists v \in S, |N(v) \cap (N(S) \setminus N(L \setminus S))| = 1)$$
$$\vee (N(S) \subseteq N(L \setminus S))$$

This can be used to compute k by enumerating all subsets S of L, where $|S| < k$. This method is suited for graphs where $|R|$ is much larger than $|L|$.

This differs from the R-subset checker in that it only checks the largest set of right-neighbours for a given $S \subseteq L$ rather than all of them. It will give the same value for k as the R-subset checker, since every subset smaller than k is checked, whilst avoiding unnecessary computations involving larger sets. Also, it only checks the largest neighbour set in R instead of all of the neighbour-sets for a given subset of L, which further improves the runtime.

```
1: function L-ComputeK(G = (L, R, E))
2:     k ← 1
3:     repeat
4:         for all S ⊆ L of size |S| = k do
5:             C = N(S) \ N(L \ S)
6:             if (C ≠ ∅) ∧ ¬(∃v ∈ S, |N(v) ∩ C| = 1) then
7:                 return k
8:             end if
9:         end for
10:        k ← k + 1
11:    until k > |L|
12:    return ∞
13: end function
```

The worst-case runtime for this is $O(2^{|L|})$. More specifically, since k is usually much smaller than $|L|$, on average it will take:

$$\sum_{i=1}^{k} \left(\frac{|L|!}{i!(|L|-i)!} \times |R| \times |L| \right)$$

C. Estimation of k

We attempted to estimate k by modifying the two algorithms to sample subsets of R and L respectively, instead of checking all of the necessary subsets. Whilst this method can produce an upper bound for k very quickly, it is very far from the actual value. For example, when sampling 10% of the search space, the bound for k found was on average 2.3 times larger than the actual value.

V. Iterative Construction

The L-subset checker in IV-B can be used as the basis of a construction which produces a graph with a given k by starting from any graph G and removing vertices from R. This can be achieved in similar time to computing k, and is of complexity $O(2^{|L|})$.

```
1: function SetK(G = (L, R, E), k)
2:     t ← 1
3:     repeat
4:         for all S ⊆ L of size |S| = t do
5:             C = N(S) \ N(L \ S)
6:             while (C ≠ ∅) ∧ ¬(∃v ∈ S, |N(v) ∩ C| = 1)
   do
7:                 delete v ∈ C from R and C
8:             end while
9:         end for
10:        t ← t + 1
11:    until t = k
12: end function
```

On line 7, a choice must be made as to which vertex to delete. This choice will affect how large the resulting graph will be. We implemented the following heuristic based on the observation that graphs which are more regular have higher values of k, as discussed in section III. This removes the vertex in C which has the largest similarity to other vertices across the entire graph.

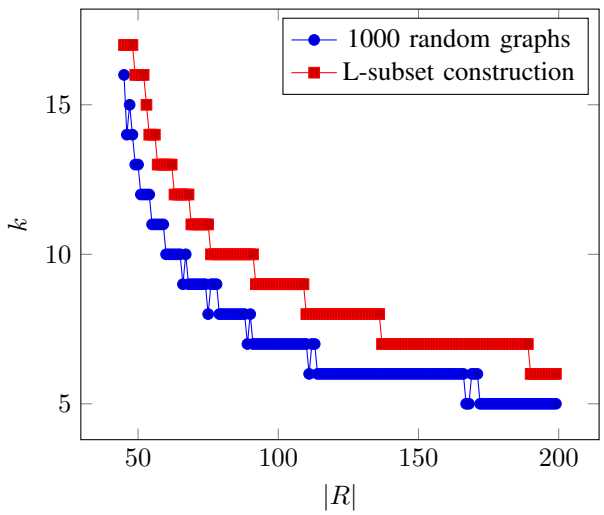
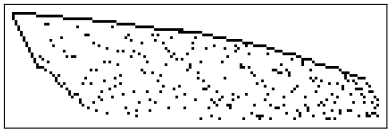


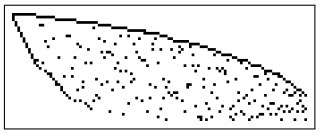
Fig. 7. Values of k for graphs with $|L| = 40$

1: **for all** $v \in C$ **do**
2: $v_c \leftarrow 0$
3: **for all** $w \in N(v)$ **do**
4: $v_c \leftarrow v_c + |N(w)|$
5: **end for**
6: **end for**
7: delete v with max v_c from R and C

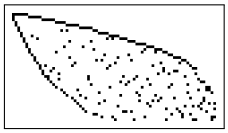
This construction produces better values of k than our baseline of choosing the best of n random graphs. Figure 7 shows the results of this construction starting on the largest possible graph with $k = 4$, i.e. $\forall v \forall w((N(v) = N(w)) \Rightarrow (v = w))$

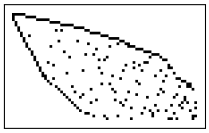
The following images shows the graph in row-column form during the stages of this construction. The figures progress to look more similar to the identity matrix which intuitively makes sense, since the identity matrix is the perfect unique-vertex expander. However, no structure that can be extrapolated to larger graphs can be otherwise discerned.

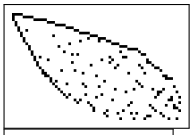

$k = 7$

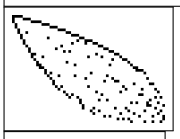

$k = 8$

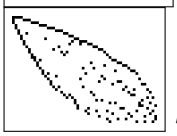

$k = 9$


$k = 10$


$k = 11$


$k = 12$


$k = 13$


$k = 14$

VI. Geometric Construction

In addition to the previous iterative construction, we propose a geometric construction based on a mapping of objects in n-dimensional spaces into bipartite graphs. This method can generate 3-regular graphs with a pre-determined k-value.

A. Space Definition

We start by defining our pre-mapped geometric space which will be later mapped into a bipartite graph.

Definition VI.1 (Pre-Mapped Space)**.** *A Pre-Mapped Space $\mathcal{S}(P, A, T)$ is a 3-tuple consisting of a set of points P, a set of arcs $A \subseteq P^2$ and a set of triangles $T \subseteq \{(p_0, p_1, p_2) \in P^3$ such that $((p_0, p_1), (p_1, p_2), (p_2, p_0)) \in A^3\}$.*

These spaces can be divided into subspaces. We define a subspace as follows.

Definition VI.2 (Pre-mapped Subspace)**.** *A space $\mathcal{S}'(P', A', T')$ is a subspace of $\mathcal{S}(P, A, T)$ if and only if $P' \subseteq P$ and $A' \subseteq A$ and $T' \subseteq T$.*

Some subspaces are connected triangular polyhedrons, which are three or more dimensional polyhedrons with only triangular 2-dimensional faces. They are defined as follows:

Definition VI.3 (Connected Triangular Polyhedron)**.** *A connected triangular polyhedron in a space $\mathcal{S}(P, A, T)$ is a space $\mathcal{S}'(P', A', T')$, which is a subspace of $\mathcal{S}$ and for which it holds that $\forall(a_1, a_2) \in A' : \exists(p_1, p_2) \in (P')^2$ such that $p_1 \neq p_2 \wedge ((a_1, a_2, p_1), (a_1, a_2, p_2)) \in (T')^2$*

Some examples of three-dimensional polyhedrons are shown in Figure 8. In every space, there may exist zero, one or many smallest polyhedra. They are defined as follows:

Definition VI.4 (The Smallest Polyhedron)**.** *The smallest polyhedron in a space $\mathcal{S}$ is the subspace with the lowest number of arcs which is a connected triangular polyhedron.*

B. Space Mapping

We propose the following mapping from $\mathcal{S}$ to a bipartite graph. This constructs a graph $G = (L, R, E)$ where each element of L represents an arc from A, each element of R represents a triangle from T and each edge in E signifies if an arc from the left set is in the triangle from the right set.

This mapping can be performed in $O(n^3)$ time, where n is the size of P.

```
function MAP(S(P, A, T))
    L, R, E ← ∅, ∅, ∅
    G ← (L ∪ R, E)
    for all (p0, p1, p2) ∈ T do
        L' ← {(p0, p1), (p1, p2), (p0, p2)}
        R' ← {(p0, p1, p2)}
        R ← R ∪ R'
        L ← L ∪ R'
        E ← L ∪ (L' × R')
    end for
    return G
end function
```

Theorem VI.5. *The graph obtained by the mapping space $\mathcal{S}$ is a k-expander if and only if the smallest polyhedron in $\mathcal{S}$ has k arcs.*

Proof. According to Definition I.1, the graph obtained is a k-expander if and only if a set $S \subseteq R$ with no unique neighbours has $N(S)$ of size k. Take any such set S. It maps to a collection of triangles T'. Those triangles are made up entirely of arcs that map to $N(S)$, call it A'. Name the set of all points needed to construct these edges and triangles P'. These sets together form a subspace $\mathcal{S}'(P', A', T')$. We must prove that $\mathcal{S}'$ is in fact a polyhedron with k arcs and that it is the smallest polyhedron in $\mathcal{S}$.

We start by proving that $\mathcal{S}'$ is a polyhedron with k arcs. For any element in $N(S)$ there exist at least two elements in S of which it is a neighbour. This means that in $\mathcal{S}'$, for every arc a in A', there exist at least two unique triangles in T', each of which consists of the two points in a and a unique third point in P'. This, according to Definition VI.3, means that $\mathcal{S}'$ is a polyhedron. The number of arcs in $\mathcal{S}'$ is $|A'| = |N(S)| = k$.

We prove the second part by contradiction. Assume $\mathcal{S}'$ is not the smallest polyhedron. Therefore there exists $\mathcal{S}^*(P^*, A^*, T^*)$ which is a subspace of $\mathcal{S}$ and a polyhedron with strictly fewer arcs than $\mathcal{S}'$. T^* will map to a set $S^* \subseteq R$ with the neighbouring set of $N(S^*)$ which only consists of mappings from the arcs in A^*. This means that $N(S^*) < k$. It is also true, according to Definition VI.3, that each arc in A^* neighbours at least two triangles in T^*, which implies that each element of $N(S^*)$ neighbours at least two elements in S^*. Therefore, we have found a subset $S^* \subseteq R$ such that $|N(S^*)| < k$ and no neighbour of S^* is a unique neighbour. This means that the bipartite graph obtained from the mapping has a k-value strictly lower than its k-value, which is a contradiction. Therefore, our assumption must be wrong and $\mathcal{S}'$ is the smallest polyhedron in space $\mathcal{S}$.

By combining part one and two of this proof, we can conclude that the graph obtained by the mapping the space $\mathcal{S}$ is a k-expander if and only if the smallest polyhedron in $\mathcal{S}$ has k arcs. □

This means that if we manage to construct a space with a good ratio of triangles to arcs and a large smallest polyhedron, we can map it in polynomial time to a good unique neighbour bipartite expander.

C. Constructions of Pre-Mapped Spaces with k=6

To obtain a 3-regular graph which has $k = 6$, we create a space where the smallest triangular polyhedron has 6 edges. The only such polyhedron is a tetrahedron with 4 triangles (See figure 8). This means that the graph produced by our mapping would be of size 6×4. However, we can combine many polyhedrons to obtain shapes with a better ratio $r = |R|/|L|$. One way of doing this is to construct n-dimensional spaces in which the objects form an n-simplex – an n dimensional equivalent of a tetrahedron. The construction of such a shape is straightforward. A vertex diagram of an n-simplex is a complete graph with $n + 1$ nodes. Therefore, we have to introduce $n+1$ points, connect all pairs of those points with an arc and add a triangle to all triplets of points. This takes $O(n^3)$ time, where n is the number of dimensions in the space we are constructing. The smallest subspace that forms a polyhedron of each of those will always be a tetrahedron – the k will remain at 6. The number of triangles and edges does increase, but the ratio between the two improves as shown in Table I.

TABLE I
SIZE OF GRAPH WITH SIZE K WHEN CONSTRUCTED FROM AN n-SIMPLEX

n dimensions	$\|A\|$ AND $\|L\|$	$\|T\|$ and $\|R\|$
3	6	4
4	10	10
5	15	20
6	21	35
7	28	56

D. Increasing k

We propose a method of increasing the value of k of a pre-mapped space $\mathcal{S}$ by removing certain triangles. Firstly the space is analysed to find any subspaces $\mathcal{S}'$ which are a polyhedron with less than 15 arcs. Figure 8 shows these polyhedra. We then remove one triangle in each such subspace from $\mathcal{S}$. This will result in the smallest polyhedron in $\mathcal{S}$ having at least 15 edges, and k being greater than 15. We propose a set of algorithms that start with a space where the smallest polyhedron has k edges, and produces a space where the smallest polyhedron has $k + 3$ edges.

a) Increasing k from 6 to 9: To increase k from 6 to 9, we must eliminate a triangle from all subspaces that are tetrahedrons. We can do that by following the procedure bellow:

```
function K9(S(P, A, T))
    for all (p1, p2, p3, p4) ∈ R^4 do
        A' ← {p1, p2, p3, p4}^2
        T' ← {p1, p2, p3, p4}^3
        if A' ⊆ A ∧ T' ⊆ T then
            T ← T − {RANDOM(T')}
        end if
    end for
    return S(P, A, T)
end function
```

For this algorithm we choose a random triangle to remove. This space modification can be achieved in $O(n^4)$ time, where n is the number of points in P.

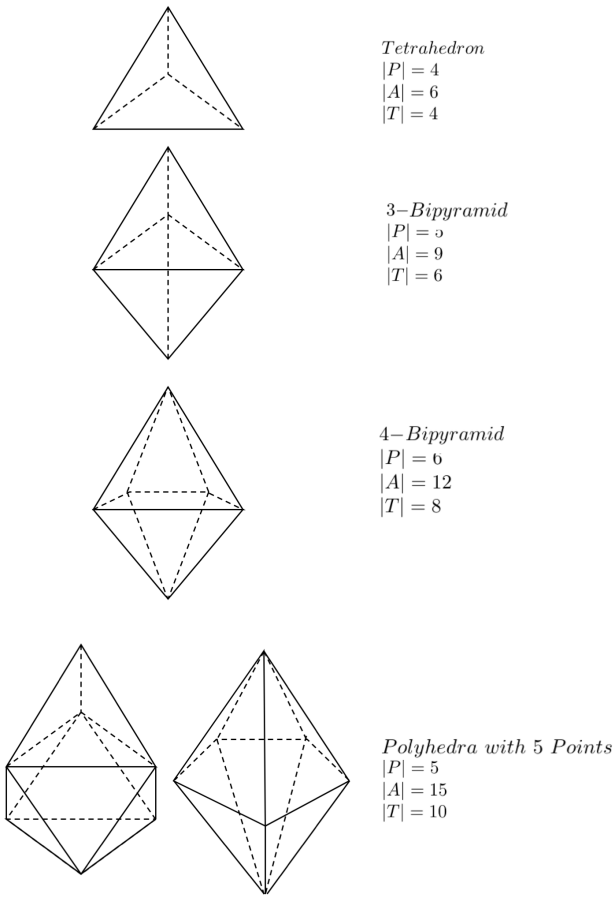


Fig. 8. Polyhedra with 15 or fewer arcs

b) Increasing k from 9 to 12: Now, we must eliminate all 3-bipyramids from the space. This can be achieved by noting that the subspace, in order to be a 3-bipyramid, must contain a set of three points that do not have a triangle in the middle. The procedure goes as follows:

function K12($\mathcal{S}(P, A, T)$)
  for all $(p_1, p_2, p_3) \in P^3 - T$ **do**
    for all $(p_4, p_5) \in (P - \{p_1, p_2, p_3\})^2$ **do**
      $A' \leftarrow \{p_1, p_2, p_3, p_4, p_5\}^2$
      $T' \leftarrow \{p_1, p_2, p_3\}^2 \times \{p_4, p_5\}$
      if $A' \subseteq A \wedge T' \subseteq T$ **then**
        $T \leftarrow T - \{\text{RANDOM}(T')\}$
      end if
    end for
  end for
  return $\mathcal{S}(P, A, T)$
end function

Again we choose a random triangle to remove. This space modification can be achieved in $O(n^5)$ time, where n is the number of points in P; however, its average case is much faster.

c) Increasing k from 12 to 15: By examining the 4-bipyramid, one can see that it is symmetric over all axes in a Cartesian coordinate system. This helps us reduce the average case by using the procedure outlined with the following pseudocode:

function K15($\mathcal{S}(P, A, T)$)
  for all $((p_1, p_2), (p_3, p_4), (p_5, p_6)) \in \left(P^2\right)^3$ **do**
    $T' \leftarrow \{p_1, p_2\} \times \{p_3, p_4\} \times \{p_5, p_6\}$
    $A' \leftarrow \{p_1, p_2\} \times \{p_3, p_4, p_5, p_6\}$
    $A' \leftarrow A' \cup \{p_3, p_4\} \times \{p_5, p_6\}$
    if $A' \subseteq A \wedge T' \subseteq T$ **then**
      $T \leftarrow T - \{\text{RANDOM}(T')\}$
    end if
  end for
  return $\mathcal{S}(P, A, T)$
end function

Again we choose a random triangle to remove. This procedure has the worst case runtime of $O(n^6)$ time, where n is the number of points in P. The average case is reduced if the edge and triangle checking is performed during the selection of pairs of points, rather than after.

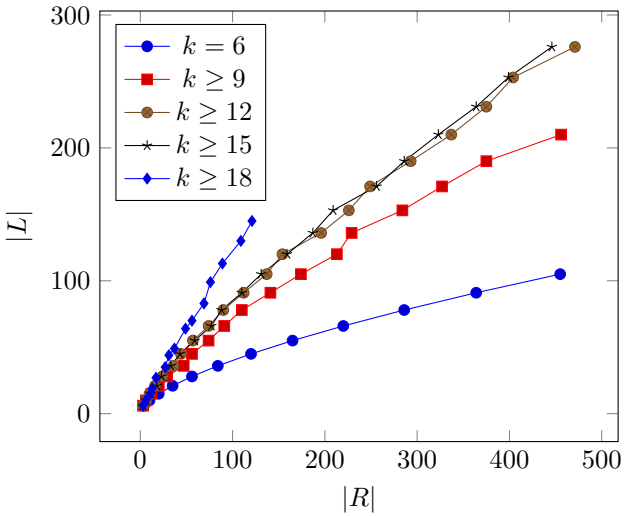


Fig. 9. Values of $|L|$ and $|R|$ for graphs constructed with different k values

E. Implementation

We combined these procedures to firstly generate an n-simplex, and then modify it by removing triangles as detailed previously. This results in graphs with a good ratio $r = |R|/|L|$, with a value of k of at least 15. This can run within realistic time constraints even for graphs with $|R| \approx 400$. We have also implemented the elimination of polyhedra with 15 arcs to even further increase k, however, those did not produce graphs with a good ratio r. Figure 9 shows the graph sizes obtained using this method.

VII. Analysis and Discussion

In this section we analyse the results of our exploration and evaluate our constructions.

We found that while larger graphs have a higher expected value of k, the expected value of $k/|L|$ drops exponentially with the size of the graph. Moreover, this exponential decrease

in expected $k/|L|$ is exhibited no matter the degree of the graphs we generate. This suggests that random search is not a feasible method for finding large graphs with low degree and values of k close to $|L|$. However, we don't know the exact nature of the relationship between graph size, degree and maximum value of k. It might be the case, for example, that the maximum values of k we found for small graph sizes after 50,000 samples are indeed the maxima for these sizes. If that were true, extrapolating from what we have seen in the small-scale would suggest that maximum $k/|L|$ goes towards 0 as $|L|$ increases (if the degree is kept constant). In other words, it would suggest that large graphs with low d and k close to $|L|$ do not exist, rather than simply being exceedingly rare. Nonetheless, some insights from the exploration, such as the fact that more regularity correlated with better expected k, might be helpful in designing constructions.

We have had relative success in speeding-up the naive computation of the highest-admitted value of k for a given graph. Introducing definition IV.1 meant that we could implement the L-subset checker, which checks left-sets in increasing order of size and terminates as soon as it finds a counterexample. This property of early termination significantly improves the average-case running time of the checker (as most graphs have relatively small k), and indeed, is what made our exploration of random graphs feasible. By contrast, our attempt to estimate k rather than exactly compute it did not produce useful results, as estimation only provides a broad upper bound on the actual value of k for the given graph.

From the constructions we developed, our iterative construction gives the best results, and can construct k-expander graphs in similar time to our method of computing k. Both of these algorithms have an exponential runtime, which means they are only applicable to small-sized graphs. Our geometric construction is also limited to generating graphs of relatively small k (with a maximum of $|R| = 400$, $|L| = 250$, $d = 3$ and $k = 15$). However, both constructions exhibit better behaviour than random graphs at least for some sizes.

VIII. Conclusion and Future Work

In this work, we carried out an exploration of small k-unique-neighbour expanders, gaining some important insights about their properties, and we developed two constructions which exhibit better performance metrics than random graphs.

However, k-unique-neighbour expansion is a relatively unexplored area, and there is much work still to be done. For example, the mapping between n-dimensional spaces and bipartite graphs is novel. Further exploration in the area of multi-dimensional spaces with good triangle to edge ratio is needed. It might uncover significantly better, potentially even explicit constructions, as well as the theoretical bounds on k-values that can be obtained from such a space mapping. Furthermore, while our exploration has helped us better understand *small* unique-neighbour expanders, more theoretical work is needed to understand how *large* unique-neighbour expanders behave. Even some more practical issues, such as determining which complexity class the problem of computing k for a given graph belongs to, are still unresolved. Usual expansion is proven to be coNP-complete [10]. Is the same true for unique-neighbour expansion? We do not know. Further research into these issues might provide us with ways to find better unique-neighbour expanders for our various applications.

References

[1] M. Pinsker, "On the complexity of a concentrator," in *7th International Telegraffic Conference*, vol. 4, 1973, pp. 1–318.

[2] N. Pippenger, "Self-routing superconcentrators," *Journal of Computer and System Sciences*, vol. 52, no. 1, pp. 53–60, 1996.

[3] M. Sipser and D. A. Spielman, "Expander codes," *IEEE Transactions on Information Theory*, vol. 42, no. 6, pp. 1710–1722, 1996.

[4] R. Impagliazzo, N. Nisan, and A. Wigderson, "Pseudorandomness for network algorithms." ACM Press, 1994, pp. 356–364. [Online]. Available: http://portal.acm.org/citation.cfm?doid=195058.195190

[5] D. X. Charles, K. E. Lauter, and E. Z. Goren, "Cryptographic Hash Functions from Expander Graphs," *Journal of Cryptology*, vol. 22, no. 1, pp. 93–113, Jan. 2009. [Online]. Available: http://link.springer.com/10.1007/s00145-007-9002-x

[6] I. Dinur, "The PCP theorem by gap amplification." ACM Press, 2006, p. 241. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1132516.1132553

[7] N. Alon, "Eigenvalues and expanders," *Combinatorica*, vol. 6, no. 2, pp. 83–96, 1986.

[8] G. Margulis, "Explicit constructions of concentrators," *Problems of Information Transmission*, no. 9.4, pp. 325—-332, 1973.

[9] A. Czumaj and C. Sohler, "Testing expansion in bounded-degree graphs," *Combinatorics, Probability and Computing*, vol. 19, no. 5-6, pp. 693–709, 2010.

[10] M. Blum, R. M. Karp, O. Vornberger, C. H. Papadimitriu, and M. Yannakakis, "The complexity of testing whether a graph is a superconcentrator," *Information Processing Letters*, vol. 13, no. 4-5, pp. 164–167, 1981.

[11] D. Angluin, "A note on a construction of margulis," *Information Processing Letters*, vol. 8, no. 1, pp. 17–19, 1979.

[12] O. Gabber and Z. Galil, "Explicit constructions of linear-sized superconcentrators," *Journal of Computer and System Sciences*, vol. 22, no. 3, pp. 407–420, 1981.

[13] M. Ajtai, "Recursive construction for 3-regular expanders," *Combinatorica*, vol. 14, no. 4, pp. 379–416, 1994.

[14] O. Reingold, S. Vadhan, and A. Wigderson, "Entropy waves, the zig-zag graph product, and new constant-degree expanders," *Annals of mathematics*, pp. 157–187, 2002.

[15] F. Hlásek, "Bounds on existence of odd and unique expanders," 2016.

[16] N. Alon and M. Capalbo, "Explicit unique-neighbor expanders," in *Foundations of Computer Science, 2002. Proceedings. The 43rd Annual IEEE Symposium on*. IEEE, 2002, pp. 73–79.

[17] M. Capalbo, O. Reingold, S. Vadhan, and A. Wigderson, "Randomness conductors and constant-degree lossless expanders," in *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*. ACM, 2002, pp. 659–668.

[18] A. Ta-Shma, C. Umans, and D. Zuckerman, "Loss-less condensers, unbalanced expanders, and extractors," in *Proceedings of the thirty-third annual ACM symposium on Theory of computing*. ACM, 2001, pp. 143–152.

[19] V. Guruswami, C. Umans, and S. Vadhan, "Unbalanced expanders and randomness extractors from parvaresh–vardy codes," *Journal of the ACM (JACM)*, vol. 56, no. 4, p. 20, 2009.