Expander Construction in VNC^1

Sam Buss[*1], Valentine Kabanets[† 2], Antonina Kolokolova[‡3], and Michal Koucký[§4]

1 **Department of Mathematics,University of California San Diego, La Jolla, USA**
sbuss@ucsd.edu
2 **School of Computing Science, Simon Fraser University, Burnaby, Canada**
kabanets@cs.sfu.ca
3 **Department of Computer Science, Memorial University of Newfoundland, St. John's, Canada**
kol@cs.mun.ca
4 **Computer Science Institute, Charles University, Prague, Czech Republic**
koucky@iuuk.mff.cuni.cz

Abstract

We give a combinatorial analysis (using edge expansion) of a variant of the iterative expander construction due to Reingold, Vadhan, and Wigderson [38], and show that this analysis can be formalized in the bounded arithmetic system VNC^1 (corresponding to the "NC^1 reasoning"). As a corollary, we prove the assumption made by Jeřábek [24] that a construction of certain bipartite expander graphs can be formalized in VNC^1. This in turn implies that every proof in Gentzen's sequent calculus LK of a monotone sequent can be simulated in the monotone version of LK (MLK) with only polynomial blowup in proof size, strengthening the quasipolynomial simulation result of Atserias, Galesi, and Pudlák [7].

1998 ACM Subject Classification F.4.1 Mathematical Logic, F.2.1 Numerical Algorithms and Problems, F.1.3 Complexity Measures and Classes

Keywords and phrases expander graphs, bounded arithmetic, alternating log time, sequent calculus, monotone propositional logic

Digital Object Identifier 10.4230/LIPIcs.ITCS.2017.31

1 Introduction

Expander graphs have become one of the most useful combinatorial objects in theoretical computer science, with many beautiful applications in computer science and mathematics [19], and responsible for several breakthroughs in computational complexity [37, 17]. These graphs have seemingly contradictory properties: sparseness and high connectivity. The high connectivity can be measured in a number of different, but essentially equivalent ways: vertex expansion (every small subset of vertices "expands", i.e., has a larger neighborhood), edge expansion (every small subset of vertices has many edges leaving the set), or fast mixing time (a random walk on a regular expander graph quickly converges to the uniform distribution on vertices).

* Supported in part by NSF grant CCF-1213151. Part of the work on VNC^1 was done while Buss was visiting the Chebychev Laboratory, St.Petersburg State University in Spring 2016, supported in part by Skolkovo Institute of Science and Technology.
† Supported in part by an NSERC Discovery grant. Part of the work was done while visiting UCSD.
‡ Supported in part by an NSERC Discovery grant. Part of the work was done while visiting UCSD.
§ The research leading to these results has received funding from the European Research Council under the European Unions Seventh Framework Programme (FP/2007-2013)/ERC Grant Agreement n. 616787.

The existence of expander graphs of constant degree can be argued nonconstructively using a simple probabilistic argument: for any constant $d \geq 3$, a random d-regular graph is almost surely an expander [33]. Constructing such graphs efficiently deterministically is much more difficult. The first explicit constructions were given by Margulis [27] and Gabber and Galil [18]. Lubotzky, Phillips, and Sarnak [25] gave a construction of expanders with particularly interesting properties, called Ramanujan graphs. All of these constructions are algebraic in nature: a graph is defined using a certain algebraic object (e.g., a group). Moreover, the analysis of correctness of the constructions is also algebraic. It relies on the algebraic notion of high connectivity called the *eigenvalue gap* and defined as follows. Consider the adjacency matrix of a given undirected d-regular graph, compute its eigenvalues, and order them according to the absolute value. It can be easily checked that d is the largest value. The difference between d and (the absolute value of) the second largest eigenvalue is the eigenvalue gap. The bigger this eigenvalue gap, the more connected the graph is. From this point of view, a d-regular expander is a graph with the eigenvalue gap at least $\Omega(d)$, i.e., the second largest eigenvalue should be at most some constant fraction of the degree.

A simpler, fully combinatorial construction of constant-degree expanders was given by Reingold, Vadhan, and Wigderson [38]. They started with constant-size expander graphs (which can found by brute-force search), and iteratively applied certain graph operations that increase the size of the graph while preserving its expansion property. This way, one can quickly construct an expander graph of any given size. While the construction of [38] is combinatorial, its analysis is still algebraic and is based on estimating the eigenvalue gap. Alon, Schwartz, and Shapira [4] gave a different construction of expanders, which combines algebraically constructed expanders of Alon and Roichman [3] with only two applications of a certain graph operation (replacement product), to obtain a constant-degree expander of arbitrary size. They also gave a fully combinatorial analysis of the replacement product operation they used in the second stage of the construction. Their full analysis, however, is still algebraic, as it relies on the algebraic construction and the eigenvalue gap analysis of [3]. In this respect, the situation in [4] is similar to that in [38] where the analysis of a related graph operation (zig-zag product) can be done in terms of min-entropy, while the analysis of the complete construction is still based on eigenvalues.

The focus of our paper is to give a construction of expanders with a simple (non-algebraic) analysis, where simplicity is measured in terms of the power of a system of bounded arithmetic needed to formalize the analysis. Informally, systems of bounded arithmetic are obtained by restricting the power of the standard first-order theory of Peano arithmetic. It is possible to devise systems of bounded arithmetic that correspond to systems of reasoning using only concepts from a given complexity class, e.g., P or NC^1. A natural question is: what is the weakest complexity class so that the existence of expander graphs can be proved using only the concepts of that complexity class?

The known expander constructions mentioned above can be formalized within a system of polytime reasoning, intuitively because eigenvalues and matrix determinants are known to be computable in polytime. Our main result is a construction of expanders that can be formalized within a system of NC^1 reasoning, VNC^1 (see below for a formal definition). As NC^1 algorithms are not known to compute the eigenvalues or determinant of a given matrix, any such formalization of an expander construction in VNC^1 must necessarily avoid the use of eigenvalues, and hence be "combinatorial" in that sense.

As expanders are used in a number of complexity-theoretic results, formalizing the expander construction within a weak system of bounded arithmetic is an important step in formalizing these complexity-theoretic results within the bounded arithmetic framework, which in turn may have other implications. For example, in proof complexity, we can use

our expander construction to argue that any Gentzen's sequent calculus LK proof (of a monotone sequent) can be simulated by a *monotone* LK (MLK) proof, with only polynomial blowup in proof size, improving upon the quasipolynomial simulation shown by Atserias, Galesi and Pudlák [7], and answering a question of Pudlák and Buss [36]. This simulation result follows by the work of Jeřábek [24] who proved the result under the assumption that a certain expander graph family can be proved to exist within a system of NC^1 reasoning. Our paper proves a strengthening of the assumption needed by Jeřábek.

1.1 Our results

Our main contribution is the analysis of one of the iterative expander constructions from [38], which we show to be formalizable in the bounded arithmetic system VNC^1 (of NC^1 reasoning). As in [38], the expander construction is *fully explicit* in the sense that that there is a deterministic polynomial-time algorithm that, given a vertex name v in binary and a number i, outputs the value of the rotation map $Rot(v,i) = (w,j)$, where w is the name of the ith neighbor of v in the graph, and j is the number such that v is the jth neighbor of w. Moreover, we show that there is an alternating linear-time algorithm that accepts exactly the triples of the form $\langle v, i, Rot(v,i)\rangle$; this kind of explicitness is what we will use to argue that the expander construction is formalizable in VNC^1.

▶ **Theorem 1** (Main result: Informal version). *The existence of an expander graph family can be proved using* NC^1 *reasoning only (within the system* VNC^1*).*

As our main application, building on Jeřábek [24] and Atserias, Galesi and Pudlák [7], we show that every proof in Gentzen's sequent calculus LK of a monotone sequent can be simulated by a monotone LK (MLK) proof (a sequent calculus proof in which all formulas are positive) with only polynomial blowup in size. This answers a question of Pudlák and Buss [36]. Previously, [7] showed such simulation with quasipolynomial blowup in proof size.

▶ **Theorem 2** (Main application). MLK *polynomially simulates* LK *on monotone sequents.*

It is easy to show that the intuitionistic propositional sequent calculus LJ polynomially simulates MLK (see Pudlák [34] and Bilková [8]); thus we get as an immediate corollary that propositional LJ polynomially simulates LK on monotone sequents, re-proving the result of Jeřábek [22, Theorem 3.9]. Many of the principles that have been considered in propositional proof complexity are expressed as monotone sequents, notably the pigeonhole principle and the clique-coloring tautologies. As these principles have polynomial size LK proofs [11], Theorem 2 implies that they also have polynomial size proofs in MLK as well as in propositional LJ. The prior best known results for the pigeonhole principle were the quasipolynomial size MLK proofs of Atserias, Galesi and Gavaldà [6].

It remains an open problem whether tree-like MLK can polynomially simulate MLK, equivalently whether tree-like MLK can polynomially simulate LK on monotone sequents. Note that [7] gives a quasipolynomial simulation.

Intuitively, to simulate an LK proof within MLK, one needs to construct (and prove correctness of) a monotone formula for the majority function. Such monotone formulas can be built using the classical AKS sorting networks [1]. Jeřábek [24] shows that the analysis of AKS sorting networks can be formalized within a certain system of NC^1 reasoning (slightly more powerful than VNC^1), under the assumption that the existence of expander graphs, with certain parameters, is also formalizable within the same system. Our Theorem 1 proves the assumption needed by Jeřábek (actually a slightly stronger version, as our proof of the existence of expanders is in the weaker system VNC^1), and so Theorem 2 immediately follows.

1.2 Relation to previous work

1.2.1 Expander constructions

The expander graph construction that we analyze is a variant of the iterative construction of expanders given in [38]. The idea is to start with a constant-size expander graph (found, say, by exhaustive search), and iteratively increase the size of the graph while keeping its expansion larger than some universal constant. The notion of expansion used by [38] is in terms of the eigenvalue gap. To analyze the expansion of the final graph, Reingold, Vadhan, and Wigderson [38] bound the effect of the graph operations they used (graph powering, graph tensoring, and zig-zag product) on the second largest eigenvalue of the adjacency matrix of the resulting graph. The analysis of graph powering (where an edge of the kth power of a graph G is a walk of length k in G) and graph tensoring (where an edge of the tensor product of G and H consists of a pair of edges, one from G and one from H) is immediate from the basic linear algebra. The analysis of the zig-zag product (a way to compose a graph G with a graph H so that the new graph has the degree of H) is technically the most difficult part of the algebraic analysis of the expander construction in [38].

In [4], a graph replacement product (closely related to the zig-zag product) is analyzed in terms of edge expansion, avoiding any mention of the eigenvalue gap. Since replacement product can be used instead of zig-zag product in an iterative expander construction along the lines of [38], this gives a combinatorial analysis of the part of the expander construction. In order to make the entire analysis combinatorial, it suffices to analyze graph powering and graph tensoring also in terms of edge expansion. This is exactly what we do in the present paper.

Our combinatorial analysis of graph tensoring, though subtle, is not very difficult. For the analysis to go through, it turns out necessary to work with graphs that have sufficiently many self-loops around every vertex (at least half the degree). On the other hand, graph powering is much more difficult to analyze combinatorially. Fortunately, here we were able to use the result of Mihail [28] who gave a combinatorial analysis of the mixing time of random walks on expanders in terms of edge expansion. (Interestingly, for her proof, she also had to work with graphs that have many self-loops around every vertex.) Finally, using Mihail's bounds, we are able to conclude the analysis of graph powering in terms of edge expansion, borrowing some ideas from [2].

1.2.2 Bounded arithmetic

There is a long history of formalizing complexity results in bounded arithmetic; indeed, this was one of the main motivations for the definitions of bounded arithmetic. First, bounded arithmetic theories can capture a range of complexity classes, from uniform AC^0 and uniform NC^1, to polynomial time, polynomial space and exponential time (see [10, 14]). Second, via the Paris-Wilkie or Cook translations, proofs in bounded arithmetic can be viewed as uniform families of propositional proofs. For this reason, a proof in bounded arithmetic can sometimes yield new propositional proofs.

There has been considerable progress in formalizing advanced results from computational complexity in weak theories of bounded arithmetic; these include approximate counting, randomized computations, and Arthur-Merlin games [20, 21], Toda's theorem [12], and the PCP theorem [32]. The present paper continues this tradition by formalizing the construction of expander graphs in the weak fragment VNC^1 which corresponds to NC^1 computation.

There are a number of prior works which use bounded arithmetic to obtain upper bounds in proof complexity. A big advantage of using bounded arithmetic is that the proofs can

be considerably simplified. A classic example is the work by Paris and Wilkie [29] who showed that the proofs of the weak pigeonhole principle in $I\Delta_0$ constructed by [30] yield constant-depth, polynomial-size Frege proofs of the propositional translations of the weak pigeonhole principle (via the "Paris-Wilkie translation"). Lower-depth, quasipolynomial-size Frege proofs were later given by [26] via a proof of the weak pigeonhole principle in a different fragment of bounded arithmetic. Similarly, [35] gave proofs of Ramsey's theorem in S_2, and these translate into quasipolynomial-size, constant-depth Frege proofs. Recently, [12] used formalization of Toda's theorem in bounded arithmetic with modular counting quantifiers to show that constant-depth $\mathsf{AC}^0(p)$-proofs, for p a prime, can be translated into quasipolynomial size, depth-three propositional proofs, with formulas being Boolean combinations of mod p gates of small conjunctions. Another classic example is Cook's theorem that extended Frege proofs have polynomial size proofs of their partial consistency statements, which was established via provability in PV [15].

The present paper establishes a new result of this type via a Cook-style translation: together with earlier work of Jeřábek [23], our formalization of expander graphs in VNC^1 implies that the monotone propositional proof system MLK polynomially simulates the proof system LK. We will use the system VNC^1 defined by Cook and Morioka [16]. We conservatively extend VNC^1 to facilitate reasoning about the compositions of NC^1 functions, which allows us to simplify the formalization of our recursive expander construction.

1.2.2.1 Remainder of the paper

Section 2 contains basic definitions. Our expander construction is defined in Section 3. In Section 4, we present a construction of bipartite expanders needed by Jeřábek [24]. In Section 5, we show that the existence of our expander graphs is provable in VNC^1, thereby proving a formal version of Theorem 1. We derive Theorem 2 in Section 6. Section 7 contains concluding remarks. For space considerations, some proofs are only sketched and other proofs are omitted from this conference version; for the full version (with all missing proofs), please see [9].

2 Preliminaries

2.1 Notation

We consider undirected graphs, possibly with parallel edges and self-loops. For an undirected graph $G = (V, E)$ on n nodes, we usually associate the vertex set V with the set $[n] = \{1, 2, \ldots, n\}$, and denote an edge $i \sim j$ between nodes i and j as $\{i, j\} \in E$. In this notation, we also allow self-loops $\{i, i\} \in E$.

The adjacencies of a d-regular graph G are given via its rotation map Rot_G so that, for vertex v of G and an index $i \in [d]$, we have $Rot_G(v, i) = (w, j)$ if w is the ith neighbor of v, and v is the jth neighbor of w; so, in particular, the rotation map induces some fixed numbering of neighbors of a given vertex.

For an n-vertex graph G, its *adjacency matrix* is an $n \times n$ matrix A' whose (i, j)th entry contains the number of edges between vertices i and j in G. For d-regular graphs G, it will be more convenient for us to consider the *normalized adjacency matrix* defined as $\frac{1}{d} \cdot A'$. Note that the normalized adjacency matrix A of G is the probability transition matrix for a random walk on G. That is, if π is a probability distribution on vertices of G, then $A\pi$ is the probability distribution induced by one step of a random walk on G starting from a vertex distributed according to π. It is also easy to see that A^k is the normalized adjacency matrix of the graph G^k.

2.2 Expanders

For a graph $G = (V, E)$ and a subset $U \subseteq V$ of vertices, we denote by $\overline{U}$ the set $V \setminus U$, and by $E(U, \overline{U})$ the set of edges between U and $\overline{U}$. The *edge expansion* of a d-regular graph $G = (V, E)$ on n vertices is defined as

$$\min_{\emptyset \neq U \subset V,\ |U| \leq n/2} \frac{|E(U,\overline{U})|}{d \cdot |U|} = \min_{\emptyset \neq U \subset V} \frac{|E(U,\overline{U})|}{d \cdot \min\{|U|, |\overline{U}|\}}. \tag{1}$$

For a graph $G = (V, E)$ and a subset $U \subseteq V$ of vertices, we denote by $\Gamma_G(U)$ the set of all neighbors of U in G, i.e.,

$$\Gamma_G(U) = \{v \in V \mid \exists u \in U, \{u, v\} \in E\}.$$

We drop the subscript G if the graph G is understood from the context. We denote by $\Gamma^+(U)$ the set $\Gamma(U) \setminus U$ of new neighbors of U. The *vertex expansion* of a graph $G = (V, E)$ on n vertices is defined as

$$\min_{\emptyset \neq U \subset V, |U| \leq n/2} \frac{|\Gamma^+(U)|}{|U|}.$$

2.3 Bounded arithmetic theory VNC^1

A number of bounded arithmetic theories have been proposed for uniform NC^1: these include the theory $\mathrm{A}^{\log}$ of Clote and Takeuti [13], the theory AID of Arai [5], the theory VNC^1 of Cook and Morioka [16], and a reformulated version of VNC^1 by Cook and Nguyen [14]. Jeřábek [23] describes a theory VNC^1_* for NC^1 under a relaxed notion of uniformity for logarithmic depth circuits.

Cook and Morioka [16] define VNC^1 using tree recursion (*TreeRec*). Cook and Nguyen [14] give an equivalent definition of VNC^1 using the Boolean formula value problem. It is easier to formalize the expander graph construction with tree recursion, so we work with the version of VNC^1 as defined by Cook and Morioka [16].

The bounded arithmetic theory VNC^1 is an extension of the theory V^0 of bounded arithmetic; V^0 corresponds in power to AC^0. V^0 is a second-order (two-sorted) system of arithmetic, with two sorts of numbers (first-order objects) and strings (second order objects). Strings are viewed as members of $\{0,1\}^*$. The notation $X(i)$, where X is a string and $i \geq 0$ is a natural number, means the Boolean value of the i^{th} entry in string X. Sometimes $i{\in}X$ is written instead of $X(i)$. The constants 0 and 1 are number terms, and addition and multiplication are number functions. Another term of type number is string length $|X|$, defined to be the value of the largest element in X when viewed as a set plus 1. Addition and multiplication are defined for numbers only, and equality is defined both for numbers and strings. The axioms of V^0 consist of a finite set of "BASIC" open axioms defining simple properties of the constants, relation symbols and function symbols, plus Σ^B_0-Comprehension axioms

$$\Sigma^B_0\text{-COMP:} \quad \exists X{\leq}y\, \forall z{<}y\, (X(z) \leftrightarrow \varphi(z))$$

for any formula φ in Δ^B_0 not containing X as a free variable, but possibly containing free variables other than z. A Δ^B_0 formula is one in which all quantifiers are bounded and which contains no second-order quantifiers. We write $(\exists X{\leq}y)\psi$ for $\exists X((|X|{\leq}y) \wedge \psi)$.

Let $\varphi(i, \vec{x}, \vec{X})[p, q]$ and $\psi(i, \vec{x}, \vec{X})$ be Σ^B_0-formulas. The notation "$[p, q]$" indicates that p and q are propositional variables that may occur as atomic subformulas in φ. The Σ^B_0-*TreeRec*

property [16] is defined by the formula $B^{\varphi,\psi}(a,\vec{x},\vec{X},Z)$:

$$(\forall i{<}a)[(Z(a{+}i) \leftrightarrow \psi(i)) \wedge (Z(i) \leftrightarrow \varphi(i,\vec{x},\vec{X})[Z(2i{+}1), Z(2i{+}2)])].$$

For $i < a$, this states that $Z(i)$ is a Boolean function of the two values $Z(2i+1)$ and $Z(2i+2)$. Thus $Z(i)$ is computed by a circuit which is a formed as a binary tree with gate types specified by φ and input values specified by ψ. We can always assume w.l.o.g. that $a = 2^{|a|}-1$; we call this the "depth condition" and it means the binary tree is exactly balanced and of depth $|a|$. This tree is of course a fanin two Boolean circuit. The type of the i-th gate is determined by $\varphi(i,\vec{x},\vec{X})$ and thus is a Σ^B_0-property of i and the inputs $\vec{x}$ and $\vec{X}$.

The theory VNC^1 is defined as V^0 plus the Σ^B_0-*TreeRec* axioms $(\exists Z{\leq}2a)B^{\varphi,\psi}(a,\vec{x},\vec{X},Z)$ for all φ and ψ in Σ^B_0. The language of VNC^1 can be extended by adding a new relation symbol $R^{\varphi,\psi}(i,a,\vec{x},\vec{X})$ for every formula $B^{\varphi,\psi}$. The defining axioms for $R^{\varphi,\psi}$ are

$$B^{\varphi,\psi}(a,\vec{x},\vec{X},R^{\varphi,\psi}) \qquad \text{and} \qquad i{\geq}2a \rightarrow \neg R^{\varphi,\psi}(i,a,\vec{x},\vec{X}).$$

Note that the defining axioms uniquely specify all the values of $R^{\varphi,\psi}$, provably in VNC^1. Adding the predicate symbols $R^{\varphi,\psi}$ and their defining axioms to VNC^1 yields the theory $\mathsf{VNC}^1(\mathit{TreeRec})$.[1] As an extension by definitions, this theory is conservative over VNC^1. This means that VNC^1 and $\mathsf{VNC}^1(\mathit{TreeRec})$ can be used interchangeably. Indeed, any $\forall\Sigma^B_1(\mathit{TreeRec})$-formula which is provable in $\mathsf{VNC}^1(\mathit{TreeRec})$ can be translated naturally to an equivalent $\forall\Sigma^B_1$-formula which is VNC^1-provable. Thus, in Section 5, we may work in VNC^1 but still use the full power of $\mathsf{VNC}^1(\mathit{TreeRec})$.

A key property of VNC^1 is that it can Σ^B_1-define precisely the (uniform) NC^1 functions; this is discussed in Section 5.1.

2.4 LK and MLK proof systems

The system MLK of monotone reasoning in [7] is a variant of Gentzen's sequent calculus LK in which all formulas are positive. An LK proof is a list of sequents of the form $\varphi_1,\ldots,\varphi_n \rightarrow \psi_1,\ldots\psi_m$, interpreted as $\bigwedge_{i=1}^n \varphi_i \rightarrow \bigvee_{j=1}^m \psi_j$. The axioms are $\varphi \rightarrow \varphi$, $\Gamma \rightarrow 1$, and $0 \rightarrow \Gamma$, for an arbitrary list of formulas Γ. Let φ,ψ denote formulas and Γ,Δ lists of formulas. The main derivation rules of LK are as follows.

- **Left derivation:** $\dfrac{\varphi,\psi,\Gamma\rightarrow\Delta}{(\varphi\wedge\psi),\Gamma\rightarrow\Delta}$ $\quad$ $\dfrac{\varphi,\Gamma\rightarrow\Delta \qquad \psi,\Gamma'\rightarrow\Delta'}{(\varphi\vee\psi),\Gamma,\Gamma'\rightarrow\Delta,\Delta'}$ $\quad$ $\dfrac{\Gamma\rightarrow\varphi,\Delta}{\neg\varphi,\Gamma\rightarrow\Delta}$
- **Right derivation:** $\dfrac{\Gamma\rightarrow\Delta,\varphi,\psi}{\Gamma\rightarrow\Delta,(\varphi\vee\psi)}$ $\quad$ $\dfrac{\Gamma\rightarrow\Delta,\varphi \qquad \Gamma'\rightarrow\Delta',\psi}{\Gamma,\Gamma'\rightarrow\Delta,\Delta',(\varphi\wedge\psi)}$ $\quad$ $\dfrac{\varphi,\Gamma\rightarrow\Delta}{\Gamma\rightarrow\Delta,\neg\varphi}$
- **Cut rule:** $\dfrac{\Gamma\rightarrow\Delta,\varphi \qquad \varphi,\Gamma'\rightarrow\Delta'}{\Gamma,\Gamma'\rightarrow\Delta,\Delta'}$

Additionally, LK includes structural rules on both sides of a sequent such as weakening, contraction of duplicate formulas, and changing order of formulas on the same side. LK is equivalent in power to Frege systems, and tree-like LK is equivalent to LK, thus VNC^1 proofs translate into polynomial-size LK proofs [5, 16, 14].

In Monotone LK (MLK), all formulas in the proof are over the $\wedge,\vee$ basis with no $\neg$.

[1] Cook and Morika [16] call this theory $\mathsf{VNC}^1(\mathcal{L}_{\mathit{TreeRec}})$.

3 Constructing edge expanders

Here we define an iterative construction of a constant-degree edge expander family, and argue its edge expansion properties using simple combinatorial tools. The simplicity of the analysis will allow us (in Section 5) to formalize it within the system VNC^1. The construction is a variant of the iterative construction given by Reingold, Vadhan, and Wigderson [38], using the graph operations described next.

3.1 Graph operations

We define the graph operations that we will use to construct expanders.

Powering For a graph $G = (V, E)$ and an integer $k \geq 1$, the kth power G^k is the graph on vertices V where for each walk of length k from a vertex u to a vertex v in G there is an edge $u \sim v$ in G^k.

If Rot_G is the rotation map of G, then the rotation map of G^k is

$$Rot_{G^k}(v, (i_1, \ldots, i_k)) = (w, (j_k, \ldots, j_1)),$$

where w is the vertex reached from v in G by edges $i_1, \ldots, i_k$ using the rotation map Rot_G, and $(j_k, \ldots, j_1)$ describes the same sequence of edges in reverse order from w's point of view. For instance, $Rot_G(v, i_1) = (v', j_1)$ for some $v' \in V$, then $Rot_G(v', i_2) = (v'', j_2)$ for some v'', etc.

Tensor product For graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$, their tensor product $G_1 \otimes G_2$ is the graph on vertices $V_1 \times V_2$, where for every pair of edges $u \sim u'$ in G_1 and $v \sim v'$ in G_2 there is an edge $(u, v) \sim (u', v')$ in $G_1 \otimes G_2$.

If Rot_{G_1} and Rot_{G_2} are the rotation maps of G_1 and G_2, respectively, then the rotation map of $G_1 \otimes G_2$ is

$$Rot_{G_1 \otimes G_2}((v, w), (i, j)) = ((v', w'), (i', j')),$$

where $Rot_{G_1}(v, i) = (v', i')$ and $Rot_{G_2}(w, j) = (w', j')$.

Replacement product For a D-regular graph $G = (V, E)$ on n vertices and a d-regular graph $H = (V', E')$ on D vertices, the replacement product $G \circ H$ is a $2d$-regular graph on nD vertices $\{(v, i) \mid v \in V,\ 1 \leq i \leq D\}$. The graph $G \circ H$ has the edges $\{(v, i) \sim (v, j) \mid v \in V,\ i \sim j \in E'\}$ as well as, for every edge $v \sim w$ in G such that w is the ith neighbor of v, and v is the jth neighbor of w (i.e., $Rot_G(v, i) = (w, j)$), $G \circ H$ has d parallel edges between (v, i) and (w, j).

If Rot_G and Rot_H are the rotation maps of G and H, respectively, then the rotation map of the $G \circ H$ is

$$Rot_{G \circ H}((v, i), j) = \begin{cases} ((v, i'), j') \text{ for } Rot_H(i, j) = (i', j') & \text{if } j \leq d \\ ((w, i'), j) \text{ for } Rot_G(v, i) = (w, i') & \text{if } j > d. \end{cases}$$

Adding self-loops For a d-regular graph $G = (V, E)$, the graph $\bigcirc G$ is the $2d$-regular graph obtained from G by adding d parallel self-loops around every vertex of G; note that we count every self-loop around vertex v as one edge $v \sim v$.

If Rot_G is the rotation map of G, then the rotation map of $\bigcirc G$ is

$$Rot_{\bigcirc G}(v, i) = \begin{cases} Rot_G(v, i) & \text{if } i \leq d \\ (v, i) & \text{if } i > d. \end{cases}$$

3.2 Effect of graph operations on edge expansion

For the operation of adding self-loops, the following lemma is obvious.

▶ **Lemma 3** (Self-loops). *If G is a d-regular graph with edge expansion ϵ, then the graph $\bigcirc G$ is $2d$-regular with edge expansion $\epsilon/2$.*

▶ **Lemma 4** (Powering). *Let G be a d-regular graph with edge expansion ϵ. For every integer $k \geq 1$, the powered graph $(\bigcirc G)^k$ has edge expansion at least*

$$\frac{1}{2} \cdot \left(1 - \left(1 - \frac{\epsilon^2}{4}\right)^{k/2}\right).$$

Proof Sketch. Our analysis is done in two stages. First, we use the result of Mihail [28] showing that a random k-step walk on an edge expander $\bigcirc G$ quickly converges to the uniform distribution over the vertices of $\bigcirc G$. Then we show that such convergence to the uniform distribution implies good edge expansion of $(\bigcirc G)^k$, using some ideas from [2].

Mihail [28] gave a combinatorial proof of the following result showing the exponentially fast convergence of a random walk on a regular graph to the uniform distribution.

▶ **Claim 5** ([28]). *Let G be a d-regular graph with edge expansion ϵ. Let A be the normalized adjacency matrix of $G' = \bigcirc G$. Let π be any initial distribution on vertices of G', and let u be the uniform distribution on vertices of G'. Then*

$$\|A^k\pi - u\|^2 \leq (1 - (\epsilon^2/4))^k \cdot \|\pi - u\|^2.$$

Let $G' = \bigcirc G$, and let $G'' = (G')^k$. Next we relate the edge expansion of G'' to the mixing time of a k-step random walk on G'. Let u denote the uniform distribution on the vertices of G''. For a subset U of vertices of G'', we denote by u_U the probability distribution that is uniform over U, i.e., every vertex in U gets weight $1/|U|$, and every vertex outside of U gets weight 0. We denote by χ_U the characteristic vector of the set U (whose ith entry is 1 if $i \in U$, and is 0 otherwise). The following claim can be proved along the lines of [2].

▶ **Claim 6.** *Suppose $G'' = (V, E)$ is a regular graph on n vertices, with normalized adjacency matrix A such that for some $\delta > 0$ the following holds: for every subset $U \subset V$ of size at most $|V|/2$,*

$$\|Au_U - u\|^2 \leq \delta \cdot \|u_U - u\|^2.$$

Then G'' has edge expansion at least $(1 - \sqrt{\delta})/2$.

Now, by Claim 5, we get for the normalized adjacency matrix A of the graph $\bigcirc G$ and for every subset $U \subset V$ that

$$\|A^k u_U - u\|^2 \leq (1 - (\epsilon^2/4))^k \cdot \|u_U - u\|^2.$$

Applying Claim 6 concludes the proof of Lemma 4. ◀

▶ **Lemma 7** (Tensoring). *Let $G = (V_G, E_G)$ be a d_G-regular graph with $d_G/2$ self-loops at every vertex and $H = (V_H, E_H)$ be a d_H-regular graph with $d_H/2$ self-loops at every vertex. If G has edge expansion ϵ_G and H has edge expansion ϵ_H, then the tensor product graph $G \otimes H$ has edge expansion at least $\min\{\epsilon_G, \epsilon_H\}/50$.*

Proof Sketch. First, we give some intuition. Suppose G is a d_G-regular graph on n_G vertices, and H is d_H-regular graph on n_H vertices. As a "warm-up", consider the special case of a subset of vertices S of the tensor product $G \otimes H$ such that $S = A \times B$. Moreover, assume that $|B| < n_H/2$. Then at least $\epsilon_H d_H |B|$ edges are leaving the set B in graph H. Each of these edges paired up with an edge from A will be an edge leaving $A \times B$ in $G \otimes H$, yielding a total of at least $\epsilon_H d_H |B| d_G |A|$ edges leaving $A \times B$. After normalization (division by $d_G d_H |A||B|$), this yields edge expansion ϵ_H from the set S. In the case, B is larger than $n_H/2$, but A is smaller than $n_G/2$, we can use the edge expansion of A, to obtain the edge expansion at least ϵ_G from S.

For general sets S of vertices in $G \otimes H$, we consider the characteristic matrix of S, which is an $n_G \times n_H$ 0-1 matrix with (i,j)th entry being 1 iff $(i,j) \in S$. We then argue that it is possible to remove some rows or some columns of this matrix so that the resulting matrix has a constant fraction of 1's of the original matrix (i.e., we removed only a constant fraction of vertices from S), and either every row or every column has at most some constant fraction of 1's.

Suppose we have the former case (the other case is treated similarly). That is, we removed some rows of the characteristic matrix of S to obtain a new subset S' that has the form $\{a_1\} \times B_1 \cup \cdots \cup \{a_k\} \times B_k$, where $a_i \in V_G$ and $B_i \subset V_H$, and moreover, each $|B_i|$ is at most some constant fraction of n_H. Then for each B_i, we can use edge expansion of H to argue that ϵ_H fraction of edges from B_i are leaving B_i. Ideally, we would like then to argue that each such edge, when paired up with any edge from vertex a_i, will leave S'. This may not be true, however, as such an edge may go to some vertex in $\{a_j\} \times B_j$. To circumvent this problem, we use the assumption that both of our graphs G and H have many self-loops around every vertex (say, half of the degree). In that case, it is easy to argue that each edge leaving B_i in H, when paired up with any self-loop around a_i, yields an edge of $G \otimes H$ that leaves S. Since the number of self-loops around a_i is at least half the degree of G, this yields edge expansion at least $\epsilon_H/2$ from each set $\{a_i\} \times B_i$. Since S' is the union of the pairwise disjoint such sets, we get the edge expansion at least $\epsilon_H/2$ from S'. Finally, since S' contains a constant fraction of vertices from S, we conclude that the edge expansion from S is at least $\Omega(\epsilon_H)$. ◀

▶ **Lemma 8** (Replacement [4]). *Let $G = (V_G, E_G)$ be a D-regular graph on n vertices, and let $H = (V_H, E_H)$ be a d-regular graph on D vertices. If G has edge expansion ϵ_G and H has edge expansion ϵ_H, then $G \circ H$ has edge expansion at least $\epsilon_G^2 \epsilon_H/48$.*

Proof Sketch. The proof idea is to partition a given subset S of vertices of $G \circ H$ into n clusters $(\{a_1\} \times B_1) \cup \cdots \cup (\{a_n\} \times B_n)$, where each $a_i \in V_G$ and $B_i \subseteq V_H$. View the clusters where $|B_i|$ is at most some fraction of $|V_H|$ as light, and the remaining clusters as heavy. For every light cluster, one can use the expansion of H to lower-bound the expansion of B_i (within the copy of H associated with vertex a_i of G). If there are many vertices in light clusters, we get a good lower bound on the edge expansion of S. Otherwise, there are many vertices in heavy clusters. Using the expansion properties of G, one can argue in this case that there will be many edges between the set of vertices in heavy clusters and the vertices outside S. ◀

3.3 Construction

With the analysis of graph operations in hand, we can now present our iterative construction of edge expanders that will be shown formalizable in VNC[1]. Let G_0 be a $(2d)$-regular graph of constant size, where d is some constant. Let ϵ_0 be the edge expansion of G_0 such that

$\epsilon_0 \geq 1/1296$. Such a graph G_0 exists (by a counting argument) and can be found in constant time, using exhaustive search. Given G_0, we will define a bigger graph G_1 that is also $(2d)$-regular and has edge expansion at least $1/1296$. In general, given a $(2d)$-regular graph G_i with edge expansion at least $1/1296$, we define G_{i+1} as follows:

$$G_{i+1} \;=\; ((\bigcirc((\bigcirc G_i) \otimes (\bigcirc G_i)))^c) \circ H, \tag{2}$$

where c is some constant to be specified later, and H is a d-regular expander graph on $(2(4d)^2)^c$ vertices, with edge expansion at least $1/3$. Again, such a graph can be found using exhaustive search.

▶ **Theorem 9.** *There is a constant c such that the graph $G_{i+1} = (V_{i+1}, E_{i+1})$ defined from $G_i = (V_i, E_i)$ as above has the following parameters:*

- $|V_{i+1}| = |V_i|^2 \cdot D$, *where* $D = (2(4d)^2)^c$,
- *the degree of G_{i+1} is $2d$,*
- *the edge expansion of G_{i+1} is at least $1/1296$.*

Proof. The bounds on the size and the degree of G_{i+1} follow easily from the definitions of the graph operations used to define G_{i+1} from G_i. Let $\epsilon \geq 1/1296$ be the edge expansion of G_i. First, by Lemma 3, the edge expansion of $\bigcirc G_i$ is at least $\epsilon/2$. By Lemma 7, the edge expansion of $G' = (\bigcirc G_i) \otimes (\bigcirc G_i)$ is at least $\epsilon' = \epsilon/100$. By Lemma 4, the kth power of the graph $\bigcirc G'$ has edge expansion at least

$$\frac{1}{2} \cdot \left(1 - \left(1 - \frac{\epsilon^2}{40000}\right)^{k/2}\right).$$

Choose k to be a sufficiently large constant c so that the edge expansion of the cth power of our graph, as given by the formula above, is at least $1/3$. Finally, by Lemma 8, the edge expansion of the graph G_{i+1} is at least $(1/3)^3/48 = 1/1296$. This completes the proof. ◀

We give also a modified construction of expanders that allows explicit constructions of edge expanders $\tilde{G}_i = (\tilde{V}_i, \tilde{E}_i)$ with $|\tilde{V}_i| = 2^i$, and more generally of edge expanders on exactly M vertices for arbitrary $M \geq 1$.

Let c be a constant. Choose the constant d to be a sufficiently large power of two, $d = 2^\ell$, so that there is a d-regular graph H on $(2(4d)^2)^c$ vertices with edge expansion at least $1/3$ and so that for all $i \leq c\ell + 7$, there are $2d$-regular graphs $\tilde{G}_i$ on 2^i vertices with edge expansion at least $1/1296$. These graphs H and $\tilde{G}_i$ can be found by exhaustive search. We construct expander graphs $\tilde{G}_i$ with edge expansion $\geq 1/1296$. For $i > 2c\ell + 7$, let $i' = \lfloor (i - 2c\ell - 5)/2 \rfloor$ and $i'' = \lceil (i - 2c\ell - 5)/2 \rceil$, so $i = i' + i'' + 2c\ell + 5$. Define

$$\tilde{G}_i \;=\; ((\bigcirc((\bigcirc \tilde{G}_{i'}) \otimes (\bigcirc \tilde{G}_{i''})))^c) \circ H. \tag{3}$$

▶ **Theorem 10.** *There is a constant c such that the graph $\tilde{G}_i = (\tilde{V}_i, \tilde{E}_i)$ defined as above has the following parameters:*

- $|\tilde{V}_i| = 2^i$,
- *the degree of $\tilde{G}_i$ is $2d$,*
- *the edge expansion of $\tilde{G}_i$ is at least $1/1296$.*

Now that we have constructed edge expanders of sizes 2^i, it is easy to obtain an edge expander $\tilde{G}$ of a given *arbitrary size* M. For this, choose i so that $2^{i-1} < M \leq 2^i$. Partition the vertices of $\tilde{G}_i$ into M disjoint subsets each of size 1 or 2. Define the graph $\tilde{G}$ by collapsing each of these subsets of vertices of $\tilde{G}_i$ into a single vertex of $\tilde{G}$, and inheriting the edges from

the all of the nodes in the subset. It is easy to see the degree of $\tilde{G}$ is at most $4d$; by adding extra self-loops, we get a new graph that is $4d$-regular. It is also easy to show that this new graph has expansion at least $\epsilon/2$ where $\epsilon = 1/1296$.

We get the following.

▶ **Theorem 11.** *Fix constants c and d as above. There is a family of $4d$-regular expanders $\tilde{G}$ on M nodes, for any $M \geq 1$, with edge expansion at least $1/2592$.*

Moreover, there is a deterministic polynomial-time algorithm that, given the name of a vertex v (in binary) of $\tilde{G}$ and an index $i \in [2d]$, outputs the value $Rot_{\tilde{G}}(v, i)$. Furthermore, there is an alternating linear time algorithm which accepts the graph of $\tilde{G}$; namely, it accepts exactly the triples of the form $\langle v, i, Rot_{\tilde{G}}(v, i)\rangle$.

It may be unexpected that we discuss alternating linear time, but the point is that this is what we need for the formalization of our arguments in the bounded arithmetic theory VNC^1 in Section 5. For that, the important thing is the computational complexity of Rot_{G_k} as a function of the size $|V_k|$ of the graph, whereas Theorem 11 expresses runtimes in terms of the size of the name of the vertex. But, the alternating linear time algorithm of Theorem 11 will be viewed as an alternating logarithmic time algorithm for purposes of formalization in VNC^1. (In the same setting, the polynomial time algorithm would be a polylogarithmic time algorithm, and it is open whether such algorithms can in general be formalized in VNC^1.)

4 Constructing bipartite vertex expanders

Jeřábek [24] needs the existence of certain bipartite vertex expanders to formalize the AKS sorting networks in VNC^1_*. We define these graphs next. Recall that, for a set S of nodes in a graph G, $\Gamma(S)$ denotes the set of all neighbors of vertices in S.

Given constants $\alpha \in (0, 1)$ and $A > 1$, a *bipartite (α, A) vertex expander* is a bipartite graph $G = (L \cup R, E)$, where $|L| = |R| = m$, such that

1. the degree of G is at most A, and
2. for all $\ell \leq m$, every set $S \subseteq [m]$ of vertices in either partition with $|S| \geq \alpha\ell$ has $|\Gamma(S)| \geq (1 - \alpha)\ell$.

That is, for every set of vertices of size at least $\alpha\ell$ in one partition, there are at least $(1-\alpha)\ell$ neighbors in the other partition.

The assumption required by [24] is:

> For $\alpha = 1/600$, there exist a constant A and a parameter-free NC^1_* function $G(m)$ such that VNC^1_* proves "$\forall m \in \mathbb{N}$, $G(m)$ is an (α, A) bipartite vertex expander on $m + m$ vertices".

We will argue that such bipartite vertex expanders can be efficiently obtained from our edge expanders defined above.

▶ **Theorem 12.** *For any constant $0 < \alpha < 1$, there exist a constant $A \geq 1$ and an efficient (uniform NC^1) algorithm that, for every $m \in \mathbb{N}$, computes the rotation map of an (α, A) bipartite vertex expander on $m + m$ vertices.*

Proof. We use the edge expander $\tilde{G}$ constructed in Theorem 11 with $M = m$. The graph $\tilde{G} = (\tilde{V}, \tilde{E})$ has $|V| = m$, degree $4d$, and expansion at least $\epsilon/2$, where $\epsilon = 1/1296$. Starting with $\tilde{G}$, we will

1. Convert the edge expander $\tilde{G}$ into a vertex expander, and
2. Turn the latter vertex expander into a bipartite (α, A) vertex expander on $m+m$ vertices.

1. Getting a vertex expander from an edge expander: Let $G = (V, E)$ be the graph $\tilde{G}$ on m nodes constructed above, but with a self-loop added to every node. So G has constant degree $4d + 1$.

We have for every set $S \subseteq V$ of size $|S| \leq m/2$ that at least $\epsilon(2d)|S|$ edges are leaving S in $\tilde{G}$. As the degree of $\tilde{G}$ is $4d$, we conclude that the neighborhood $\Gamma(S)$ of S in G contains at least $\epsilon \cdot (2d) \cdot |S|/(4d) = \epsilon' \cdot |S|$ distinct nodes from $\overline{S}$, where $\epsilon' = \epsilon/2$. As G has self-loops around every node, we get

$$|\Gamma(S)| \geq (1 + \epsilon') \cdot |S|, \tag{4}$$

for every subset S of G with $|S| \leq m/2$.

Consider the power graph G^i, for any $i \geq 1$. Applying Eq. (4) inductively, we get for every subset S of G^i with $|S| \leq m/2$, and for every $i \geq 1$ that

$$|\Gamma_{G^i}(S)| \geq \min\{m/2, (1 + \epsilon')^i \cdot |S|\}. \tag{5}$$

Now let S be a subset of V of size $|S| \geq m/2$. We have $|\Gamma^+(S)| \geq \epsilon' \cdot |\overline{S}|$, where $\Gamma^+(S) = \Gamma(S) \cap \overline{S}$ is the set of new neighbors of S. It follows that

$$|\overline{\Gamma(S)}| \leq (1 - \epsilon') \cdot |\overline{S}|. \tag{6}$$

Applying Eq. (6) inductively, we get for every $i \geq 1$, and for every subset S of V of size $|S| \geq m/2$ that

$$|\overline{\Gamma_{G^i}(S)}| \leq (1 - \epsilon')^i \cdot |\overline{S}|. \tag{7}$$

▶ **Claim 13.** *There exists a constant $t' = t'(\alpha, \epsilon')$ such that, for every $\ell \leq m$ and every set S of $G^{t'}$ with $|S| \geq \alpha\ell$, we have $|\Gamma_{G^{t'}}(S)| \geq (1 - \alpha)\ell$.*

Proof of Claim 13. Consider two cases: $\ell \leq m/2$, and $\ell > m/2$. If $\ell \leq m/2$, then by Eq. (5) we get for $t_1 = \lceil \log_{1+\epsilon'}(1/\alpha) \rceil$ that

$$|\Gamma_{G^{t_1}}(S)| \;\geq\; \min\{m/2, (1 + \epsilon')^{t_1} \cdot \alpha\ell\} \;\geq\; \min\{m/2, \ell\} \;=\; \ell.$$

If $\ell > m/2$, then $|\overline{S}| \leq m - \alpha\ell < (1 - (\alpha/2)) \cdot m < m$. For $t_2 = \lceil (\log 1/\alpha)/(\log 1/(1 - \epsilon')) \rceil$, we get

$$|\overline{\Gamma_{G^{t_2}}(S)}| \;\leq\; (1 - \epsilon')^{t_2} \cdot m \;\leq\; \alpha \cdot m,$$

and hence, $|\Gamma_{G^{t_2}}(S)| \geq (1-\alpha)m \geq (1-\alpha)\ell$. Taking $t' = \max\{t_1, t_2\}$ concludes the proof. ◀

2. Getting a bipartite vertex expander: Let $G^{t'}$ be the vertex expander defined above. Observe that it has m nodes, and has constant degree $A = (4d + 1)^{t'}$. We turn this graph into a bipartite graph by taking two copies of the vertices of $G^{t'}$, denoted by L and R, connecting nodes $i \in L$ and $j \in R$ by an edge iff $\{i, j\}$ is an edge of $G^{t'}$. Claim 13 implies that the resulting graph is an (α, A) vertex expander.

Finally, the explicitness of this construction of (α, A) vertex expanders can be argued similarly to the case of the edge expanders of Theorem 11: we trace the construction of $G^{t'}$ to get an efficient (uniform NC^1) algorithm for computing the rotation map of the corresponding bipartite (α, A) expander on $m + m$ vertices. ◀

5 Formalizing the construction in bounded arithmetic

This section discusses the formalization of the expander graph construction in the theory VNC^1 of bounded arithmetic. A high-level description of how we formalize the expander graph construction in VNC^1 is as follows:

1. The first step is to establish (in Section 5.4) that VNC^1 can define the operations of graph powering, replacement product, and tensoring. From this it follows that VNC^1 can carry out the definition of G_{i+1} from G_i, for the graphs G_i defined in Section 3. Similarly, VNC^1 can carry out the construction of $\tilde{G}_i$ from $\tilde{G}_{i'}$ and $\tilde{G}_{i''}$ as in (3).
2. For the second step, we wish to use induction on t to prove the existence of the graph G_t for suitable t. However, since VNC^1 does not support induction on Σ^B_1-formulas, we cannot use the usual induction axioms for VNC^1. Instead, we exploit the fact that the graph G_{i+1} has size quadratic in the size of G_i, namely $|G_{i+1}| = \Theta(|G_i|^2)$. This large growth rate allows us to use Σ^B_1-induction to prove the existence of G_t for arbitrary (first-order) integers t. For this, Theorems 16 and 17 of Section 5.3 prove that the needed induction principle is provable in VNC^1. The intuition is that the computational content of the induction axioms corresponds to composing logarithmic depth circuits, and that since the G_i's are growing quadratically, arbitrary composition of logarithmic depth circuits for the G_i's yields a circuit which is still of only logarithmic depth.
   The same Σ^B_1-induction will also be used to prove the existence of the graphs $\tilde{G}_i$, exploiting the fact that the size of $\tilde{G}_i$ is quadratic in the sizes of $\tilde{G}_{i'}$ and $\tilde{G}_{i''}$.
3. Theorems 16 and 17 give the needed induction principle for handling compositions of circuits, but more work is needed for VNC^1 to formalize the iterated composition of circuits. What we mean by "iterated composition" of circuits is that there are multiple circuits (about $|(|t|)|$ many circuits) which are arranged with the outputs of one circuit feeding into the inputs of the next circuit. To formalize this circuit composition in VNC^1, we need to modify Cook and Morioka's definition [16] of *TreeRec* tree recursion in VNC^1. The problem with the *TreeRec* form of tree recursion is that the second order inputs to a circuit defined by tree recursion are not used at the input gates of the circuit, but rather are used throughout the circuit, indeed potentially at every gate in the circuit. To fix this, Section 5.2 introduces a modified version of tree recursion, called *TreeRec*$'$, which allows the use of second order inputs $X_0(i)$ only as input values. This allows composition of circuits using the inputs X_0 for the iteratively computed values. The *TreeRec*$'$ tree recursion and the new induction principle of Section 5.3 then suffice to define G_t by using recursively the definition of G_{i+1} from G_i.
4. The fourth step is to prove the expansion properties of G_{i+1} follow from those of G_i. Or, more precisely, proving that if G_{i+1} does not have the desired edge expansion then G_i also does not.
5. The fifth step is to use induction on t to prove the expansion properties for G_t. This is done in Theorem 21; its proof again utilizes the induction principle introduced in Section 5.3. This shows that VNC^1 can prove the existence of expander graphs.
6. The sixth, and final step, is to note that the proof of Theorem 12 can be carried out in VNC^1, so VNC^1 proves the existence of bipartite vertex expanders.

This proof is given below. We start by proving some useful properties of VNC^1 in Sections 5.1–5.3. We show in Section 5.4 that VNC^1 can express relevant graph properties. Section 5.5 shows that the edge expansion properties of our graph operations can be proved within VNC^1.

5.1 Defining NC^1 functions within VNC^1

Cook and Morioka [16, Lemma 13] show that $\mathsf{VNC}^1(\mathit{TreeRec})$ can prove the $\Sigma^B_0(\mathit{TreeRec})$-COMP axioms. They then define the FNC^1 functions F by using $\Sigma^B_0(\mathit{TreeRec})$-formulas $\varphi(i,\vec{x},\vec{X})$ and terms $t(\vec{x},\vec{X})$ and defining the string $F(\vec{x},\vec{X})$ by[2]

$$F(\vec{x},\vec{X})(j) \;\leftrightarrow\; j{<}t(\vec{x},\vec{X}) \wedge \varphi(j,\vec{x},\vec{X}). \tag{8}$$

They also show that the Σ^B_1-definable functions of VNC^1 are precisely the FNC^1 functions [16, Theorem 17]. Recall that a Σ^B_1-definition is given by VNC^1 proof of $(\exists!Y)\varphi(\vec{x},\vec{X},Y)$ where $\varphi \in \Sigma^B_1$; this serves as a definition of the string function $\vec{x},\vec{X} \mapsto Y$.

The definition of FNC^1 functions using (8) is equivalent to the usual definition of the FNC^1 functions as the functions whose bit graphs are computable in U_{E^*}-uniform NC^1, or equivalently are computable in $\mathsf{ALogTime}$. Those functions are computed by a family $\{C_n\}_n$ of fanin ≤ 2 Boolean circuits, taking inputs of length n and having depth $O(\log n)$. The U_{E^*}-uniformity condition was defined by Ruzzo [40] and means that the circuits C_n are described by two functions $g(i,n)$ and $p(i,w,n)$ which are computable in the linear time hierarchy (equivalently, they have Σ^B_0 graphs). The first function $g(i,n)$ returns the type of gate i in C_n. The second function $p(i,w,n)$ takes as input also a $w \in \{0,1\}^*$: the bits of w describe a path in the circuit starting at gate i and following successively the first or second input to gates according to the bits of w. The value of $p(i,w,n)$ is the index of the gate reached by following this path specified by w starting from gate i in C_n. The functions g and p are in the linear time hierarchy; however, since they have inputs of length $O(\log n)$, they run in time $O(\log n)$ using a constant number of alternations. For more details, see [40].

We will need to carefully analyze the effect of composing FNC^1 functions; for this reason it is important that the existence of U^*_E-uniform NC^1 circuits for FNC^1 functions can be proved by the theory VNC^1. This follows from Theorem 15 below.

5.2 A modified tree recursion

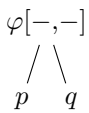
TreeRec acts like a fanin two, Boolean circuit where the internal gate types are given by $\varphi = \varphi(i,\vec{x},\vec{X})$. A disadvantage of this definition of *TreeRec* is that the side parameters $\vec{X}$ can be used unrestrictedly by the Σ^B_0-formulas φ and ψ. The formula $\varphi(i,\vec{x},\vec{X})$ defines the type of gate number i when the circuit has $\vec{x},\vec{X}$ as inputs. Likewise, $\psi(i,\vec{x},\vec{X})$ defines the True/False value of the i-th input. This differs from the usual conventions of having a circuit have fixed gate types, and having the inputs affect only the values of input gates. It also makes it difficult to define the notion of composing circuits, with the outputs of one family of circuits serving as the inputs to another circuit.

We define a new formulation of tree recursion called *TreeRec′* to address this problem. In a *TreeRec′* definition, one of the second order inputs, X_0, will serve as an "ordinary" input to the circuit, with the values $X_0(j)$ specifying the True/False values on inputs to the circuit. The other second order inputs, $\vec{X}'$, can be used to define gate types similarly as is done by *TreeRec*. This allows recursive computations on the value X_0 to be formalized with composition of circuits.

We assume X_0 is one of the side string parameters $\vec{X}$, so $\vec{X}$ is $X_0,\vec{X}'$. We modify the definition of *TreeRec* so that the values $X_0(i)$ are used only as inputs to the *TreeRec* circuit,

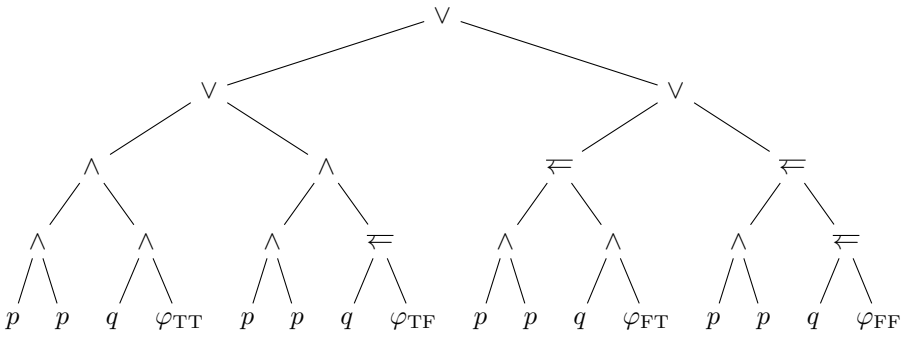
[2] This definition of FNC^1 is same as what Cook and Morioka [16] call "the function symbols in $\mathsf{VNC}^1(\mathsf{FNC}^1)$". We use just "$\mathsf{FNC}^1$" to keep the notation less cumbersome.

and are not used to determine the gate types; in particular, X_0 is not used by φ. The basic construction for the definition of *TreeRec*′ is that a single gate in a *TreeRec* circuit, of gate type $\varphi[-,-]$:



(9)

is replaced by a small tree of binary gates



(10)

Here the binary gate $r \Leftarrow s$ is $\neg r \wedge s$; and the values φ_{pq} are the truth values of $\varphi(i, a, \vec{x}, \vec{X})[p, q]$. By inspection, the circuit is depth four and fanin two: the top $\vee$ gate branches on the value of p; the next two $\vee$ gates branch on q. The last two levels select the correct value of φ_{pq}, for $p =$ T,F and $q =$ T,F based on the values of p and q. In other words, the circuit (10) implements a "lookup table", using the values p and q to select the appropriate value φ_{pq}. Assuming that the four values of φ_{pq} are correctly computed, the effect of replacing the binary gates (9) with the circuits (10) gives a circuit of depth $4|a|$ computing the same result as the original *TreeRec* circuit of depth $|a|$.

We wish to replace the four leaf nodes of (10) labelled φ_{pq} with Boolean circuits which have as inputs only the values $X_0(i)$. Since φ is Σ^B_0-formula, such circuits can easily be described by a polynomial time function of $i, \vec{x}, \vec{X}'$. These circuits are formed by applying the Paris-Wilkie transformation to φ, namely by replacing bounded quantifiers in φ with conjunctions and disjunctions, and hardcoding the values of $\vec{x}$ and $\vec{X}'$ (but not X_0) as constants. The result is that each leaf φ_{pq} of the circuit (10) can be replaced by a fanin two circuit which (a) has as inputs only $X_0(j)$'s and constants, (b) is size $\leq q(|a|, |\vec{x}|)$ and depth $\leq |q(|a|, |\vec{x}|)|$ for some polynomial q, and (c) there is a Σ^B_0-definable number function $f(p, q, i, a, \vec{x}, \vec{X})$ of VNC^1 which outputs a succinct description of the circuit. VNC^1 is able to straightforwardly define f and prove all these properties.

With this construction in hand, we define a modified version of tree recursion:

▶ **Definition 14.** Let $\varphi(i, a, \vec{x}, y_0, \vec{X}')[p, q]$ be a Σ^B_0-formula and let $k(i, a, \vec{x}, y_0, \vec{y}')$ be a Σ^B_0-definable number function. The Σ^B_0-*TreeRec*′ property for φ and k is given by:

$$B^{\varphi,k}(a, \vec{x}, X_0, \vec{X}', Z) = (\forall i{<}a)[\,(Z(i) \leftrightarrow \varphi(i, a, \vec{x}, |X_0|, \vec{X}')[Z(2i{+}1), Z(2i{+}2)]) \\ \wedge\ (Z(a+i) \leftrightarrow X_0(k(i, a, \vec{x}, |X_0|, |\vec{X}'|)))\,].$$

The *defining axioms for the predicate symbols* $R^{\varphi,k}(i, a, \vec{x}, X_0, \vec{X}')$ are the formulas

$$B^{\varphi,k}(a, \vec{x}, X_0, \vec{X}', R^{\varphi,k}) \qquad \text{and} \qquad i{\geq}2a \rightarrow \neg R^{\varphi,k}(i, a, \vec{x}, X_0, \vec{X}'). \tag{11}$$

Note that the gate type depends only on $|X_0|$, not on the values of $X_0(\cdot)$. VNC^1 proves that (11) uniquely specifies all values of $R^{\varphi,k}$. Furthermore, it is not hard to see that VNC^1 proves the existence of string objects satisfying the conditions of (11). Thus, we may conservatively extend $\mathsf{VNC}^1(\mathit{TreeRec})$ by adding all these predicate symbols along with their defining axioms. The resulting theory is called $\mathsf{VNC}^1(\mathit{TreeRec}, \mathit{TreeRec}')$.

The main advantage of $\mathit{TreeRec}'$ definitions is that they can explicitly give U_{E^*}-uniform log-depth circuits. For this, we assume that X_0 is the only second-order input (so $\vec{X}'$ is missing). We also assume that $a = s(\vec{x}, |X_0|)$ for some V^0-term s. It is usually convenient to assume in addition that each $x_i < |X_0|^{O(1)}$, so that we can think of $|X_0|$ as the size of the input (up to a polynomial); in fact, often $\vec{x}$ is missing, so the only first-order input is $|X_0|$. The other condition needed for U_{E^*}-uniformity is that there must be a linear time hierarchy algorithm (i.e., a Σ^B_0 formula) determining the extended connection language for the connectivity of gates in the circuit. Since the circuit is formed as a binary tree, with a natural numbering system for gates, the extended connection language of the circuit is trivial. Specifically, suppose $w \in \{0,1\}$ is a string of bits and i is a gate. Interpret bits "0" and "1" as selecting the first or second input to a gate, and let w specify a path starting at gate i, and traversing inputs according to the bits of w. The gate at the end of this path is gate i' where i' has binary representation obtained by concatenating the binary representation of i and the string w. The type of gate i can be defined with a Σ^B_0-formula using the Σ^B_0-formula φ and the Σ^B_0-defined function k. Thus, with the assumptions stated above, a $\mathit{TreeRec}'$ definition defines a U_{E^*}-uniform circuit.

The next theorem states that every $\Sigma^B_0(\mathit{TreeRec})$-property has log-depth, fanin two, Boolean circuits in the form used by $\mathit{TreeRec}'$.

▶ **Theorem 15.** *Let $\chi(\vec{x}, X_0, \vec{X}')$ be a $\Sigma^B_0(\mathit{TreeRec})$-formula. Then there are a Σ^B_0-formula $\varphi(i, a, \vec{x}, y_0, \vec{X}')$, a Σ^B_0-defined function $k(i, a, \vec{x}, y_0, |\vec{y}'|)$, and a V^0-term $s(\vec{x}, |X_0|, |\vec{X}'|)$ so that $\mathsf{VNC}^1(\mathit{TreeRec}, \mathit{TreeRec}')$ proves*

$$\chi(\vec{x}, X_0, \vec{X}') \;\leftrightarrow\; R^{\varphi,k}(0, s(\vec{x}, |X_0|, |\vec{X}'|), \vec{x}, X_0, \vec{X}').$$

$\Sigma^B_0(\mathit{TreeRec})$-properties may involve composing multiple *TreeRec* predicates with built-in function symbols, then combining them with Boolean operations and first-order quantifiers. Theorem 15 states that any such property χ may expressed as a $\mathit{TreeRec}'$: the advantage is that this gives an explicit NC^1 representation of χ; namely in terms of logarithmic depth Boolean circuits. "Logarithmic" means as a function of the values $\vec{x}$ and of the sizes $|X_0|, |\vec{X}'|$ of the second order inputs X_0, X'.

5.3 A conservation result

We prove the closure of VNC^1 under a rule of inference based on a "telescoping" iteration. This turns out to be exactly what is needed for the formalization of the expander graph construction inside VNC^1. We write $\sqrt{a}$ for the greatest integer at most $\sqrt{a}$.

▶ **Theorem 16.** *Suppose $\chi(X)$ is a Σ^B_0-formula containing only X free, and let $\psi(a)$ be $(\exists X{\leq}a)\chi(X)$. Also suppose VNC^1 proves*

$$(\forall a)(\psi(a) \to \psi(\sqrt{a})). \tag{12}$$

Then VNC^1 proves $\psi(a) \to \psi(1)$, and thus also proves $\chi(Y) \to (\exists X{\leq}\,1)\chi(X))$.

Theorem 16 used a descending induction; a similar theorem holds also for ascending induction:

▶ **Theorem 17.** *Suppose $\varphi(X)$ is a Σ_0^B-formula containing only X free. Also suppose* VNC^1 *proves*

$$\varphi(Y) \to (\exists X)(|X| \geq |Y|^2 \wedge \varphi(Y)).$$

Then VNC^1 *proves* $(\exists Y)\varphi(Y) \to (\forall x)(\exists X)(|X| > x \wedge \varphi(X))$.

5.4 Expressing expander graph properties in VNC^1

We now discuss how VNC^1 can express properties about graphs, adjacency matrices, expansion properties, and graph constructions such as powering, tensor product and replacement product. A graph G on n vertices will be encoded in VNC^1 as a string object (a second order object). Here n is a number (a first-order object), and the intent is to represent G in terms of its adjacency matrix. The (i,j)-th entry of the adjacency matrix is the number of edges between vertices i and j. It is represented by a three-place second order predicate $A(i,j,k)$ where $A(i,j,k)$ is true when there are exactly k edges between i and j. (Strictly speaking, we should write $A(\langle i,j,k\rangle)$, but we suppress this notation.) Each i,j,k is a number (a first order object); it will be important that we always have $k < p(n)$ for some fixed polynomial p, since then k is Σ_0^B-definable from A,i,j, and we can write $k = A(i,j)$ for the value of k.

Row vectors and column vectors (containing numbers) are likewise representable by strings, with $A(i,k)$ meaning that the i-th entry of the vector is equal to k.

With these conventions it is easy for VNC^1 to Σ_0^B- or Σ_1^B-define many properties of the graph G encoded as above. We illustrate this with several examples.

- For $u < n$, the set of edges containing vertex u can be defined as the set

  $$E(\{u\}) \;=\; \{\langle u,v,k\rangle : (\exists k' \leq p(n))(k < k' \wedge A(u,v,k'))\}.$$

  Note this allows for multiedges. The degree of v is $|E(\{v\})|$ and can be Σ_0^B-defined with the *Numones* function. G has degree d if each $u \in [n]$ has degree d. There will always be a polynomial upper bound $p(n)$ on the degree.
- For $U \subset [n]$, the set $E(U,\overline{U})$ is defined similarly as

  $$E(U,\overline{U}) \;=\; \{\langle i,j,k\rangle : i \in U \wedge j \notin U \wedge (\exists k' \leq p(n))(k < k' \wedge A(i,j,k'))\}.$$

- Rational numbers p/q are represented by pairs of integers (p,q) (not necessarily in reduced form). The usual ordering $p/q < p'/q'$ is of course definable by $pq' < p'q$, where $q, q' > 0$. Pairs of rational numbers may be added or multiplied or divided as usual.
  The proof of the Cauchy-Schwarz theorem, and more generally our proofs of expansion properties, argue about sums of vectors of rational numbers. VNC^1 can define summations of vectors of integers [14], but it is not clear whether it can define summations of vectors of arbitrary rational numbers. This will be handled in our VNC^1 proofs by clearing the denominators so that we can argue about summations of integers instead of about summations of rational numbers. In our applications, the least common multiple of the denominators will be easily computed, making it easy to clear the denominators.
- The edge expansion of a degree d graph G can thus be defined by as in equation (1) with $V = [n]$. This, however, is not a Σ_1^B-definition, since it requires minimizing over all subsets $U \subset [n]$. Instead we can define the property "G has edge expansion $> p/q$" as

  $$(\forall U{<}n)\Big(0 < |U| \leq \frac{n}{2} \to \frac{|E(U,\overline{U})|}{d \cdot |U|} > \frac{p}{q}\Big).$$

  This is a Π_1^B-condition. Recall that "$(\forall U{<}n)$" is quantifying over all subsets of $[n]$.

- A rotation map is encoded by a second order object $Rot(u, i, v, j)$ with the meaning that the i-th edge of u is the same as the j-th edge of v. We can relate the rotation map Rot and the adjacency matrix A by letting the i-th edge from u to v be the edge $\langle u, v, k\rangle$ such that

$$|\{\langle u, i', v, j\rangle : Rot(u, i', v, j) \wedge i' < i\}| \;=\; k$$

  Furthermore, the adjacency matrix A is Σ^B_1-definable in terms of Rot, since $A(u, v) = k$ holds exactly when there are exactly k values $\langle i, j\rangle$ such that $Rot(u, i, v, j)$. Since v, j are uniquely determined by u, i, we also use the notation $Rot(u, i) = (v, j)$.
  It is also possible to Σ^B_1-define a canonical rotation map as a function of the adjacency matrix.

Graph operations are also readily defined by VNC^1:

- To add self-loops to convert a d-regular G to a $2d$-regular G', define the adjacency matrix $A'(u, v, k)$ as

$$(u \neq v \wedge A(u, v, k)) \vee (u = v \wedge (\exists k' \leq d)(A(u, v, k') \wedge (k = k' + d))).$$

- (Graph Powering.) Let $k > 1$ be fixed. VNC^1 can Σ^B_1-define the graph power G^k from G as follows. We write $\langle i_1, \ldots, i_k\rangle$ for an efficient sequence coding so that each $\langle i_1, \ldots, i_k\rangle$ is represented by an integer $< d^k$. Then $Rot(u, \langle i_1, \ldots, i_k\rangle) = (v, \langle j_1, \ldots, j_k\rangle)$ holds iff

$$(\exists \langle u_0, \ldots, u_k\rangle)[u_0{=}u \wedge u_k{=}v \wedge \bigwedge\nolimits_{s=1}^{k} (Rot(u_{s-1}, i_s) = (u_s, j_{k-s+1}))].$$

  Since k is fixed and each $u_i < n$, the quantifier is a bounded number quantifier.
- Similar arguments give Σ^B_1-definitions of Tensor Product and Replacement Product. The constructions are straightforward and we leave the details to the reader.

These constructions, along with Theorem 17, allow VNC^1 to prove the existence of the graphs G_i as defined by (2). Fix constants d and c, and fix a $(2d)$-regular G_0 with edge expansion ϵ_0. Also, fix a rotation map $Rot_0 = Rot_{G_0}$ for G_0. Given G_i and Rot_i, for $i \geq 0$, VNC^1 can prove the existence of G_{i+1} satisfying (2) along with the existence of Rot_{i+1}. Furthermore, by Theorem 17, VNC^1 can prove the existence of a second-order object encoding a sequence of graphs and rotation maps

$$(G_0, Rot_0),\ (G_1, Rot_1),\ (G_2, Rot_2),\ \ldots,\ (G_{|a|}, Rot_{|a|}), \tag{13}$$

so each G_{i+1} and associated rotation map Rot_{i+1} is obtained from G_i and Rot_i by Equation (2). Letting the constant $D = 2(4d)^2)^c$ as before, each G_i has $(|V_0| \cdot 4D)^{2^i}/D$ many vertices, provably in VNC^1. (See Theorem 9.) The size of G_{i+1} is greater than the square of the size of G_i; indeed, $|V_{i+1}| = D \cdot |V_i|^2$. Therefore, Theorem 17 applies, to show that VNC^1 can Σ^B_1-define the sequence (13) as function of a, and hence can Σ^B_1-define $G_{|a|}$ and $Rot_{|a|}$ as functions of a.

Similar, only slightly more complicated, arguments allow VNC^1 to prove the existence of the graphs $\tilde{G}_i$ as defined by (3). Now i can be an arbitrary first-order (integer) value $i = a$, not just a length $|a|$. Fix appropriate constants $d = 2^\ell$ and c, and for $i \leq 2c\ell{+}8$, fix graphs $\tilde{G}_i$ with edge expansion $\geq 1/1296$ and their rotation maps Rot_i. Using induction on Σ^B_0-formulas, VNC^1 proves the existence of a sequence of values $k_0, \ldots, k_s$ such that $k_0 = a$ and each $k_{i+1} = \lfloor (k_i{-}2c\ell{-}5)/2 \rfloor$, and such that s is the first value where $k_s < 2c\ell{+}7$. Given both $\tilde{G}_{k_{i+1}}$ and $\tilde{G}_{k_{i+1}+1}$ and their rotation maps $Rot_{k_{i+1}}$ and $Rot_{k_{i+1}+1}$, and using

the definition (3), VNC^1 can prove the existence of both $\tilde{G}_{k_i}$ and $\tilde{G}_{k_i+1}$ and their rotation maps. Furthermore, the sizes of $\tilde{G}_{k_i}$ and $\tilde{G}_{k_i+1}$ are both greater than the square of the size of $\tilde{G}_{k_{i+1}+1}$. Therefore, by Theorem 17 again, VNC^1 can prove the existence of a second-order object encoding a sequence of pairs of graphs and rotation maps:

$$(\tilde{G}_{k_s}, Rot_{k_s}, \tilde{G}_{k_s+1}, Rot_{k_s+1}), (\tilde{G}_{k_{s-1}}, Rot_{k_{s-1}}, \tilde{G}_{k_{s-1}+1}, Rot_{k_{s-1}+1}), \ldots \\ (\tilde{G}_{k_0}, Rot_{k_0}, \tilde{G}_{k_0+1}, Rot_{k_0+1}), \tag{14}$$

with successive pairs of expander graphs obtained via (3). Since $k_0 = a$, this shows that VNC^1 can Σ^B_1-define $\tilde{G}_a$ and Rot_a as functions of a.

It is immediate from the definition of G_i, using induction on i, that VNC^1 proves that each G_i has degree $2d$ (for the appropriate value of d). Likewise VNC^1 proves that each $\tilde{G}_i$ has degree $2d$. It is more difficult to prove that VNC^1 proves G_i and $\tilde{G}_i$ have the edge expansion properties of Theorems 9 and 10. This is discussed in the next sections.

5.5 Formalizing edge expansion properties in VNC^1

We prove that the graph operations can be analyzed in VNC^1. For $\emptyset \neq U \subsetneq V$, we denote by $\textit{edge-exp}_G(U)$ the *edge expansion* ratio defined as follows:

$$\textit{edge-exp}_G(U) \;=\; \frac{|E(U, \overline{U})|}{d \cdot \min\{|U|, |\overline{U}|\}}.$$

▶ **Lemma 18.** *Let k be even.* VNC^1 *proves the following: Suppose G^k is the graph power of G as defined in Section 5.4, and V is the common vertex set of G and G^k. Then*

$$(\exists U)[U \subset V \wedge \textit{edge-exp}_{(\bigcirc G^k)}(U) < [\tfrac{1}{2}\big(1 - \big(1 - \tfrac{\epsilon^2}{4}\big)^{k/2}\big)] \rightarrow (\exists U)[U \subset V \wedge \textit{edge-exp}_G(U) < \epsilon].$$

▶ **Lemma 19.** VNC^1 *proves the following: Let $G = (V_G, E_G)$ be a d_G-regular graph with $d_G/2$ self-loops at every vertex and $H = (V_H, E_H)$ be a d_H-regular graph with $d_H/2$ self-loops at every vertex. Let $\epsilon = \min\{\epsilon_G, \epsilon_H\}$. Then,*

$$(\exists U)[U \subset (V_G \otimes V_H) \wedge \textit{edge-exp}_{G \otimes H}(U) < \epsilon/50] \\ \rightarrow\; (\exists U)[U \subset V_G \wedge \textit{edge-exp}_G(U) < \epsilon_G] \vee (\exists U)[U \subset V_H \wedge \textit{edge-exp}_H(U) < \epsilon_H].$$

▶ **Lemma 20.** VNC^1 *proves the following: Let $G = (V_G, E_G)$ be a D-regular graph on n vertices, and let $H = (V_H, E_H)$ be a d-regular graph on D vertices. Let $\epsilon = \epsilon_G^2 \epsilon_H/48$, and let $V_{G \circ H}$ denote the vertices of $G \circ H$. Then,*

$$(\exists U)[U \subset V_{G \circ H} \wedge \textit{edge-exp}_{G \circ H}(U) < \epsilon] \\ \rightarrow\; (\exists U)[U \subset V_G \wedge \textit{edge-exp}_G(U) < \epsilon_G] \vee (\exists U)[U \subset V_H \wedge \textit{edge-exp}_H(U) < \epsilon_H].$$

Finally, the arguments in Section 3.3 also formalize in VNC^1 to combine Lemmas 18-20 to prove the existence of expander graphs. For this, we need to formulate the arguments so as to apply Theorem 16. We first how how to prove the existence of the edge expanders G_i in VNC^1. To talk about the edge expansion of G_i, we encode a subset U of V_i using a string Y of length exactly $|V_i| + 1 = (|V_0| \cdot 4D)^{2^i}/D + 1$, by letting $Y = U \cup \{|V_i|\}$. It follows from the discussion at the end of Section 5.4 that VNC^1 can Σ^B_1-define G_i as a function of $|V_i|$, hence as a function of Y.

Let $A(Y)$ express the conditions that (a) $|Y| = |V_i| + 1$ for some i, and (b) Y encodes a subset U of V_i such that $\textit{edge-exp}_{G_i}(U) < 1/1296$. The (contrapositive of the) argument in Section 3.3, formalized in VNC^1, shows that the following is VNC^1 provable:

$$(\exists Y \leq a) A(Y) \rightarrow (\exists Y \leq \sqrt{a}) A(Y). \tag{15}$$

For $i = 0$, this uses the fact that G_0 has edge expansion $\geq 1/1296$, and since G_0 is a constant graph, this can be checked by a enumerating all of the finitely many subsets.

Applying Theorem 16 to (15) gives that VNC^1 proves

$$(\exists Y{\leq}a)A(Y) \to (\exists Y{\leq}1)A(Y).$$

There are only four possible Y's with $|Y| \leq 1$. The righthand side, $(\exists Y{\leq}1)A(Y)$, is a false Σ^B_0-formula asserting a finite property. Hence, VNC^1 can trivially disprove $(\exists Y{\leq}1)A(Y)$ by direct evaluation. Therefore, VNC^1 proves $\neg(\exists Y)A(Y)$, i.e., can prove that any V_i must be an expander. This completes the proof of the following.

▶ **Theorem 21.** *There is a constant d so that VNC^1 proves the existence of arbitrarily large, degree $2d$ graphs with edge expansion $\geq 1/1296$. Namely, VNC^1 proves*

$$(\forall a)(\exists V, E)[|V| \geq a \wedge (V, E) \text{ is a degree } 2d \text{ graph} \\ \wedge(\forall U)(U \subseteq V \to \textit{edge-exp}_{(V,E)}(U) \geq 1/1296)].$$

In fact, there is a Σ^B_1-definable function G of VNC^1 so that that VNC^1 proves

$$(\forall a)[G(a) \text{ is a degree } 2d \text{ graph } G(a) = (V, E) \text{ with } |V| \geq a \\ \wedge(\forall U)(U \subseteq V \to \textit{edge-exp}_{(V,E)}(U) \geq 1/1296)].$$

VNC^1 can also prove the existence of edge expander graphs of aribitrary size.

▶ **Theorem 22.** *There is a constant $d = 2^\ell$ and a Σ^B_1-definable function G of VNC^1 so that VNC^1 proves*

$$(\forall a)[G(a) \text{ is a 4d-regular graph } G(a) = (V(a), E(a)) \text{ with } |V| = a \\ \wedge(\forall U)(U \subseteq V \to \textit{edge-exp}_{(V,E)}(U) \geq 1/(2 \cdot 1296))].$$

Proof. Pick appropriate constant values for d and c. VNC^1 starts by proving the existence of $\tilde{G}_i$ for the least i such that $2^i \geq a$. VNC^1 can prove the existence of the sequence $k_0, \ldots, k_s$ with $k_0 = i$, and each $k_{i+1} = \lfloor (k_i - 2c\ell - 5) \rfloor$ and s the first value with $k_s < 2c\ell + 7$. In addition, by Section 5.4, VNC^1 can prove the existence of second-order objects encoding edge expanders $\tilde{G}_j = (\tilde{V}_j, \tilde{E}_j)$ for every value $j = k_i$ or $j = k_i + 1$ with $i \leq s$. Recall that $|\tilde{V}_j| = 2^j$. Let $A(Y)$ express the condition that for some $i \leq s$, either (a) $|Y| = 2^{k_i} + 1$ and Y encodes a subset U of $\tilde{V}_{k_i}$ such that $\textit{edge-exp}_{\tilde{G}_{k_i}}(U) < 1/1296$, or (b) $|Y| = 2^{k_i+1} + 1$ and Y encodes a subset U of $\tilde{V}_{k_i+1}$ such that $\textit{edge-exp}_{\tilde{G}_{k_i+1}}(U) < 1/1296$. The (contrapositive) of the argument in Section 3.3, now shows that

$$(\exists Y{\leq}a)A(Y) \to (\exists Y{\leq}\sqrt{a})A(Y).$$

is VNC^1-provable. Applying Theorem 16 gives that VNC^1 proves

$$(\exists Y{\leq}a)A(Y) \to (\exists Y{\leq}1)A(Y).$$

Therefore, VNC^1 proves $\neg(\exists Y{\leq}a)A(Y)$, i.e., it proves the edge expansion properties for arbitrary Y, and hence the edge expansion properties of $\tilde{G}_i$. ◀

Finally, VNC^1 can also formalize the argument given in Section 4 to construct bipartite vertex expanders. The only new proof ingredient is the use of logarithms to define t_1 and t_2 in the proof of Claim 13. VNC^1 can define rational approximations to logarithms; here we need only integers t_1 and t_2 such that $(1 + \epsilon')^{t_1} \geq 1/\alpha$ and $(1 - \epsilon')^{t_2} \leq \alpha$. Since ϵ' is

small, these values can be estimated as $|\lceil 1/\alpha \rceil|/\epsilon'$. Actually, in the argument for Section 4, we have $\alpha = 1/600$ and $\epsilon' = \epsilon/D'$ are fixed constants; hence t_1 and t_2 are constants as well. Finally, at the very end of the proof of Theorem 12, we have $A = (D'(2d)+1)^{\max\{t_1,t_2\}}$, where $t' = \max\{t_1, t_2\}$. Thus A is also a constant. Here it is important that t' is constant, or at least is not too large, so that t' can be used as an exponent.

Thus we have proved the following theorem.

▶ **Theorem 23.** VNC^1 *proves Theorem 12 for any constant* α*. Namely, for any fixed rational* $0 < \alpha < 1$*, there exists an* $A > 0$ *and a* Σ^B_1*-defined function* $F(m)$ *of* VNC^1 *so that the following holds:* VNC^1 *proves that for all* m*,* $F(m)$ *equals the rotation map* Rot_G *of an* (α, A) *bipartite vertex expander graph* G *on* $m+m$ *vertices.*

As VNC^1 is a subtheory of VNC^1_*, Theorem 23 is stronger than the assumption needed by Jeřábek [23].

6 Application to monotone sequent calculus

In [36], Pudlák and Buss introduced a proof system for reasoning with monotone formulas, motivated by strong lower bounds results for monotone circuits, and posed the question whether similar difference in complexity holds in the propositional proof system setting. More specifically, they formulated monotone sequent calculus and asked whether any non-monotone proof of a monotone sequent can be replaced by a monotone proof at most polynomially larger. In [34], Pudlák further investigated this question, focusing in particular on the pigeonhole principle. There, he discussed the need to formalize properties of monotone counting formulas such as AKS sorting networks of [1], and asked whether there are small proofs of basic properties of counting formulas.

The pigeonhole principle was shown to have polynomial-size monotone sequent calculus proofs by Atserias, Galesi and Gavaldá in [6]; this paper was the first to use the name MLK for this system. The same paper also gave quasipolynomial-size proofs of basic counting principles. Building upon the latter result, Atserias, Galesi and Pudlák [7] show that, in contrast to monotone circuit classes, monotone proof systems are nearly as powerful as non-monotone ones: polynomial-size non-monotone proofs can be simulated by monotone ones of quasipolynomial size. The quasipolynomial blowup is introduced in the [6] proofs of certain properties of threshold formulas.

To prove that every LK proof can be converted into an MLK proof of quasipolynomial size, [7] use monotone threshold formulas to eliminate negated variables. A *threshold formula* $TH^n_k(x_1, \ldots, x_n)$ asserts that at least k variables x_i are 1. The standard inductive definition builds TH^n_k as a disjunction of $TH^{n/2}_i(x_1, \ldots, x_{n/2}) \wedge TH^{n/2}_j(x_{n/2+1}, \ldots, x_n)$ for all pairs $i, j \leq n/2$ such that $i + j \geq k$. This definition yields quasipolynomial size formulas TH^n_k, and thus gives only quasipolynomial size LK proofs of properties of TH^n_k. If LK is polynomially bounded, then so is MLK (as in this case properties of threshold functions would have polynomial-size LK proofs). More generally, they use the following lemma based on results from [6]:

▶ **Lemma 24** ([7, Lemma 6])**.** *Let* TH^n_k *be a polynomial-size monotone threshold formula. Then* MLK *polynomially simulates* LK *on monotone sequents, provided there are polynomial-size* LK *proofs of the following sequents:*

1. $TH^n_k(x_1, \ldots, x_n) \to$ *and* $\to TH^n_0(x_1, \ldots, x_n)$ *for every* n *and* $k > n$.
2. $TH^n_k(x_1, \ldots, x_i/0, \ldots, x_n) \to TH^n_{k+1}(x_1, \ldots, x_i/1, \ldots, x_n)$ *for all* n*,* k *and* i *such that* $0 \leq k, i \leq n$.

Such polynomial-size monotone threshold formulas can be built using the classic construction of monotone log-depth sorting networks by Ajtai, Komlós and Szemerédi [1], known as AKS sorting networks. A sorting network can be thought of as a circuit with n outputs gates, which contain the values of the input gates in sorted order. That is, the k^{th} output of a sorting network is 0 iff there are at least k 0s among inputs to the network. The construction of AKS sorting networks is fairly involved; see [31, 41] for expositions. At the end of the paper, Atserias et al. note that replacing their threshold formulas with monotone NC^1 sorting networks of Ajtai, Komlós and Szemerédi would remove the blowup and allow for *polynomial-size* simulation, provided the relevant properties can be proven with NC^1 reasoning (not necessarily monotone).

Jeřábek [24] has shown just that, under the assumption that bipartite expanders graphs with appropriate parameters can be constructed, and their properties proven in NC^1 reasoning. More precisely, Jeřábek [24] has shown that AKS sorting networks (Paterson's [31] variant) are indeed formalizable in a theory VNC^1_* of NC^1 reasoning, under the assumption of the existence of a family of bipartite expanders provable in VNC^1_* (with parameters as in Claim 13). The theory VNC^1_* is somewhat stronger than VNC^1 that we use, in that it can evaluate and reason about less uniform families of log-depth circuits; however, proofs in VNC^1_* still translate into polynomial-size LK proofs [23]. Thus, Jeřábek obtains the following result:

▶ **Theorem 25** ([24, Theorem 5.5]). *Suppose that there exists a constant D and a parameter-free NC^1_* function $G(m)$ such that VNC^1_* proves that for all numbers m, $G(m)$ is a $\langle 1/600, D\rangle$ bipartite $m+m$ expander. Then* MLK *polynomially simulates* LK *on monotone sequents.*

The construction in Theorem 12 gives expanders with the appropriate parameters, and Theorem 23 shows that it can be done in VNC^1 (and thus VNC^1_*). As this proves the assumption of Theorem 25, we immediately get the following corollary.

▶ **Theorem 26** (Main application). MLK *polynomially simulates* LK *on monotone sequents.*

7 Conclusions and open problems

From the point of view of bounded reverse mathematics, the area that tries to pinpoint the minimal reasoning power needed to prove mathematical theorems, it is very interesting to understand what is the complexity of reasoning required to prove properties of expander graphs, and thus what is the complexity of reasoning in expander-based proofs such as the known proofs of $\mathsf{SL} = \mathsf{L}$ [37, 39]. This paper makes a step in this direction by showing that an expander construction can be formalized within the system VNC^1.

A number of open questions remain. Can we formalize expanders in a weaker theory than VNC^1, e.g., the system of TC^0 reasoning? Can Reingold's result that undirected graph connectivity is in deterministic logspace [37] be formalized in the system of logspace reasoning? The analysis of graph powering given in this paper and the analysis of replacement product given in [4] are not strong enough to achieve that goal.

Finally, as was already asked by [24], can the AKS construction of expanders be modified to yield U_{E^*}-uniform sorting networks?

Acknowledgements. We want to thank Denis Thérien and Pascal Tesson for inviting V.K., A.K., and M.K. to the 2007 McGill Complexity Workshop in Barbados, where this paper was initiated. V.K. and A.K. also wish to thank Josh Buresh-Oppenheim, Shlomo Hoory, and Rahul Santhanam for our many discussions on expander graphs. V.K. and A.K. are particularly thankful to Russell Impagliazzo for inviting them to spend a semester at UCSD in

the spring of 2016, where this work was finally completed. S.B. thanks Amir Akbar Tabatabai and Raheleh Jalali for useful discussions on VNC[1], and Rosalie Iemhoff and Anupam Das for discussions on intuitionistic logic. We also thank Anupam Das for his comments on our paper, and Albert Atserias for clarifying to us the history of the MLK proof system. We are especially grateful to Emil Jeřábek for carefully reading our manuscript and pointing out some errors in the early versions.

References

1 Miklós Ajtai, Janós Komlós, and Endre Szemerédi. An $O(n \log n)$ sorting network. In *Proceedings of the Fifteenth Annual ACM Symposium on Theory of Computing*, pages 1–9. Association for Computing Machinery, 1983.

2 Noga Alon and Fan R.K. Chung. Explicit construction of linear sized tolerant networks. *Discrete Mathematics*, 72:15–19, 1988.

3 Noga Alon and Yuval Roichman. Random Cayley graphs and expanders. *Random Structures and Algorithms*, 5:271–284, 1994.

4 Noga Alon, Oded Schwartz, and Asaf Shapira. An elementary construction of constant-degree expanders. *Comb. Probab. Comput.*, 17(3):319–327, May 2008. doi:10.1017/S0963548307008851.

5 Toshiyasu Arai. A bounded arithmetic AID for Frege systems. *Annals of Pure and Applied Logic*, 103:155–199, 2000.

6 Albert Atserias, Nicola Galesi, and Ricard Gavaldá. Monotone proofs of the pigeon hole principle. *Mathematical Logic Quarterly*, 47(4):461–474, 2001.

7 Albert Atserias, Nicola Galesi, and Pavel Pudlák. Monotone simulations of non-monotone proofs. *Journal of Computer and System Sciences*, 65(4):626–638, 2002. doi:10.1016/S0022-0000(02)00020-X.

8 Marta Bílková. Monotone sequent calculus and resolution. *Commentationes Mathematicae Universitatis Carolinae*, 42:575–582, 2001.

9 Sam Buss, Valentine Kabanets, Antonina Kolokolova, and Michal Koucký. Expander construction in VNC[1]. *Electronic Colloquium on Computational Complexity (ECCC)*, TR16-144, 2016. URL: http://eccc.hpi-web.de/report/2016/144/.

10 Samuel R. Buss. *Bounded Arithmetic*. Bibliopolis, 1986. Revision of 1985 Princeton University Ph.D. thesis.

11 Samuel R. Buss. Polynomial size proofs of the propositional pigeonhole principle. *Journal of Symbolic Logic*, 52:916–927, 1987.

12 Samuel R. Buss, Leszek Aleksander Kołodziejczyk, and Konrad Zdanowski. Collapsing modular counting in bounded arithmetic and constant depth propositional proofs. *Transactions of the AMS*, 367:7517–7563, 2015.

13 Peter Clote and Gaisi Takeuti. Bounded arithmetics for NC, ALOGTIME, L and NL. *Annals of Pure and Applied Logic*, 56:73–117, 1992.

14 Stephen Cook and Phuong Nguyen. *Logical Foundations of Proof Complexity*. Cambridge University Press, New York, NY, USA, 1st edition, 2010.

15 Stephen A. Cook. Feasibly constructive proofs and the propositional calculus. In *Proceedings of the Seventh Annual ACM Symposium on Theory of Computing*, pages 83–97, 1975.

16 Stephen A. Cook and Tsuyoshi Morioka. Quantified propositional calculus and a second-order theory for NC[1]. *Archive for Mathematical Logic*, 44:711–749, 2005.

17 Irit Dinur. The PCP theorem by gap amplification. *J. ACM*, 54(3), June 2007. doi:10.1145/1236457.1236459.

18 Ofer Gabber and Zvi Galil. Explicit construction of linear sized superconcentrators. *Journal of Computer and System Sciences*, 22:407–420, 1981.

19 Shlomo Hoory, Nathan Linial, and Avi Wigderson. Expander graphs and their applications. *Bulletin of the American Mathematical Society*, 43(4):439–561, 2006.

20 Emil Jeřábek. Approximate counting in bounded arithmetic. *Journal of Symbolic Logic*, 72(3):959–993, 2007.

21 Emil Jeřábek. Approximate counting by hashing in bounded arithmetic. *Journal of Symbolic Logic*, 74(3):829–860, 2009.

22 Emil Jeřábek. Substitution frege and extended frege proof systems in non-classical logics. *Annals of Pure and Applied Logic*, 159(1):1 – 48, 2009. doi:10.1016/j.apal.2008.10.005.

23 Emil Jeřábek. On theories of bounded arithmetic for NC^1. *Annals of Pure and Applied Logic*, 162(4):322–340, 2011.

24 Emil Jeřábek. A sorting network in bounded arithmetic. *Annals of Pure and Applied Logic*, 162(4):341–355, 2011.

25 Alexander Lubotzky, Ralph Phillips, and Peter Sarnak. Ramanujan graphs. *Combinatorica*, 8(3):261–277, 1988.

26 Alexis Maciel, Toniann Pitassi, and Alan R. Woods. A new proof of the weak pigeonhole principle. *Journal of Computer and System Sciences*, 64(4):843–872, 2002.

27 Grigory Margulis. Explicit constructions of expanders. *Problems of Information Transmission*, pages 71–80, 1973.

28 Milena Mihail. Conductance and convergence of Markov chains: A combinatorial treatment of expanders. In *Proceedings of the Thirtieth Annual IEEE Symposium on Foundations of Computer Science*, pages 526–531, 1989.

29 Jeff B. Paris and Alex J. Wilkie. Δ_0 sets and induction. In W. Guzicki, W. Marek, A. Pelc, and C. Rauszer, editors, *Open Days in Model Theory and Set Theory*, pages 237–248, 1981.

30 Jeff B. Paris, Alex J. Wilkie, and A. R. Woods. Provability of the pigeonhole principle and the existence of infinitely many primes. *Journal of Symbolic Logic*, 53:1235–1244, 1988.

31 M. S. Paterson. Improved sorting networks with O(log N) depth. *Algorithmica*, 5(1-4):75–92, 1990. doi:10.1007/BF01840378.

32 Jan Pich. Logical strength of complexity theory and a formalization of the PCP theorem in bounded arithmetic. *Logical Methods in Computer Science*, 11(2:8):1–38, 2015.

33 Mark Pinsker. On the complexity of a concentrator. In *Proceedings of the Seventh Annual Teletraffic Conference*, pages 1–4, 1973.

34 P. Pudlak. On the complexity of the propositional calculus. In S. Barry Cooper and John K.Editors Truss, editors, *Sets and Proofs:*, London Mathematical Society Lecture Note Series, page 197–218. Cambridge University Press, Jun 1999.

35 Pavel Pudlák. Ramsey's theorem in bounded arithmetic. In *Computer Science Logic, Lecture Notes in Computer Science #553*, pages 308–312. Springer-Verlag, 1992.

36 Pavel Pudlák and Samuel R Buss. How to lie without being (easily) convicted and the lengths of proofs in propositional calculus. In *International Workshop on Computer Science Logic*, pages 151–162. Springer, 1994.

37 Omer Reingold. Undirected connectivity in log-space. *J. ACM*, 55(4):17:1–17:24, September 2008. doi:10.1145/1391289.1391291.

38 Omer Reingold, Salil Vadhan, and Avi Wigderson. Entropy waves, the zig-zag graph product, and new constant-degree expanders. *Annals of Mathematics*, 155(1):157–187, 2002.

39 Eyal Rozenman and Salil P. Vadhan. Derandomized squaring of graphs. In *Approximation, Randomization and Combinatorial Optimization, Algorithms and Techniques, 8th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems, APPROX 2005 and 9th InternationalWorkshop on Randomization and Computation, RANDOM 2005, Berkeley, CA, USA, August 22-24, 2005, Proceedings*, pages 436–447, 2005.

40 Walter L. Ruzzo. On uniform circuit complexity. *Journal of Computer and System Sciences*, 22:365–383, 1981.

41 Joel Seiferas. Sorting networks of logarithmic depth, further simplified. *Algorithmica (New York)*, 53(3):374–384, 2009. doi:10.1007/s00453-007-9025-6.