Some graph products and their expansion properties

Andrew Brown, EPFL

Joint work with Amin Shokrollahi.
Introduction

- Graph products have recently been used to construct explicit families of expander graphs (The zig-zag product [RVW]).

- This is a recursive construction that uses graph products.

- Question: Can we, in a similar way, use products of codes to recursively construct explicit families of good binary codes?

- It turns out that the problem of finding good binary codes can be rephrased as finding Cayley graphs over $(\mathbb{F}_2^k, +)$ that are good expanders
Expander graphs

- Different ways to characterize expander graphs.
- The most intuitive is that any set of nodes must have many neighbors (combinatorics)
- There is also an algebraic characterization: Look at $\lambda(G)$, the second largest eigenvalue (in absolute value) of the normalized adjacency matrix of the graph.
- Smaller $\lambda(G)$ means better expansion
- A constant degree expander family is a family $\{G_i\}_i$ of $[n_i,d,\lambda_i]$-graphs with $\lim_{i \to \infty} n_i = \infty$ and $\lambda_i \leq \lambda$ for some fixed $\lambda < 1$.
- Random regular graphs are good expanders.
- Applications: Derandomization, cryptography, circuit complexity, topology, etc...
Code - Expander connection

- **Family of good codes:** A family \( \{ C_i \}_i \) of codes with parameters \([n_i, k_i, d_i]\), with \( k_i/n_i \leq R \) and \( d_i/n_i \leq \delta \) for some \( R, \delta < 1 \) (\( \lim_{i \to \infty} n_i = \infty \)).

- Different ways to relate expander graphs to error correcting codes:
  
- **Expander codes** (Sipser, Spielman). From a family of expander graphs, construct a family of good codes.

- Since there are known explicit constructions for the required expander families, this leads to explicit constructions of good codes.

- Codes described by their **Tanner graph**
Code - Expander connection

- **Cayley graph:** Given a group $G$ and a generating set $S$. We consider the graph with:
  - Nodes: elements of $G$
  - Edges: $g_1 \sim g_2 \iff \exists s \in S : g_2 = g_1 + s$.

- Take the $k \times n$ generator matrix of binary code $C$. It has rank $k$.

- So its $n$ columns generate $(\mathbb{F}_2^k, +)$. We let $G(C)$ be the **Cayley graph** of $(\mathbb{F}_2^k, +)$ with respect to this generating set.

- **Theorem.** The parameters are the following:
  \[
  [n, k, d]-\text{code} \rightarrow [2^k, n, 1 - \frac{2d}{n}]-\text{graph}
  \]

- So good codes lead to good expanders.
• **Recall:** We are looking to define code products
  • We have a correspondance:

  \[ \text{Code} \leftrightarrow \text{Cayley graph over } \mathbb{F}_2^k \]

• **Obvious idea:** What about applying the zig-zag to the Cayley graphs? **Problem:** The result is no longer an \( \mathbb{F}_2^k \)-Cayley graph.

• Need a graph product that preserves this property.

• A graph product that does this: **Tensor product**
Graph tensoring

- $A, B$ graphs with node sets
  $$[n_A] = \{1, \ldots, n_A\}$$
  $$[n_B] = \{1, \ldots, n_B\}$$

- $A \otimes B$:
  - Nodes: $[n_A] \times [n_B]$
  - Edges: $(a, b) \sim (a', b') \iff a \sim_A a' \quad b \sim_B b'$
Tensor product $A \otimes B$
copies of B
Graph tensoring

- Parameters $[n_A \cdot n_B, d_A \cdot d_B, \max(\lambda_A, \lambda_B)]$

- Increases the size of the graph (dimension): good
- Maintains the second eigenvalue (distance): good
- But also increases the degree a lot (length): bad

- Problem for codes: Degree increases too much ($\implies$ length of code increases faster than dimension).

- Idea: Remove some edges from $A \otimes B$ in a clever way.
Reducing the degree

- **Graph squaring:** $G^2$ has the same nodes as $G$, take all paths of length 2 as edges.

- August 2005: Rozenman and Vadhan presented a new operation **derandomized squaring** $\mathbb{S}$. This involves squaring a graph, and then *removing some edges* according to a second graph.

- Reduces degree at the cost of slightly worse expansion

- Can be seen as a projection of the zig-zag product

\[ A\mathbb{S}(C^2) = P[(A\boxdot C)^2] \]

- We wanted to remove edges from the tensor product (without losing too much expansion)

- We can use this idea to come up with **derandomized tensoring**: Take the tensor product of two graphs, and remove edges according to a third bi-partite graph.
Derandomized tensoring (1)

- $A$, $B$ graphs with node sets $[n_A]$, $[n_B]$, degrees $d_A, d_B$.

- Assume edge colorings
  \[
  \varphi_A : E(A) \to [d_A],
  \]
  and likewise $\varphi_B$.

- For a Cayley graph: 1 color $\leftrightarrow$ 1 generator

- Suppose we have a bipartite graph $C$ with $d_A$ left nodes, and $d_B$ right nodes.

- So there is a correspondance:
  | colors of $A$ $\leftrightarrow$ left nodes of $C$ |
  | colors of $B$ $\leftrightarrow$ right nodes of $C$ |

- $A \otimes_C B$: node set $[n_A] \times [n_B]$

- Edges: $(a, b) \sim (a', b') \iff a \sim_A a' \land b \sim_B b' \land \varphi_A(a, a') \sim_C \varphi_B(b, b')$
Derandomized tensoring (2)

• $A \otimes_C B$:
  - Number of nodes $= n_A \cdot n_B$
  - Degree $= |\text{Edges}(C)|$

• If $C$ is biregular of left and right degrees $\ell, r$:
  Degree $= d_A \cdot \ell = d_B \cdot r$.

• If $C$ is the complete bipartite graph then
  $A \otimes_C B = A \otimes B$.

• In terms of codes this involves appending certain columns from the two generator matrices.
Expansion properties

• What are the expansion properties of this product?

**Theorem.** Suppose without loss of generality that $\lambda_B \leq \lambda_A$. Suppose also that $C$ is biregular. Then

$$\lambda_{A \otimes CB} \leq \max \left( \lambda_A, \lambda_B, m(\lambda_A, \lambda_B, \lambda_C) \right),$$

where we let

\[
\begin{align*}
  f(a, b, c) &= ab + c\sqrt{(1 - a^2)(1 - b^2)}, \\
  g(b, c) &= \left(\frac{c^2}{b^2} - c^2 + 1\right)^{-1/2}, \\
  m(a, b, c) &= f\left(\min(a, g(b, c)), b, c\right).
\end{align*}
\]

• **Simpler case:** If $\lambda_A = \lambda_B$ then

$$\lambda_{A \otimes CB} \leq \max \left( \lambda_A, \lambda_A^2 + \lambda_C \cdot (1 - \lambda_A^2) \right)$$
Projection

- The analysis is done by viewing $A \otimes_C B$ as a projection of a larger graph.
Projection

- The analysis is done by viewing $A \otimes_C B$ as a projection of a larger graph.
Proof of Theorem

• We view our graph over $[n_A] \times [n_B]$ as projection of a graph over $[n_A] \times [n_B] \times [d_A + d_B]$. 

• Normal tensoring: $A \otimes B = \hat{A} \cdot \hat{B}$, where 
  \[
  \hat{A} = A \otimes \text{Id}(n_B) \\
  \hat{B} = \text{Id}(n_A) \otimes B
  \]

• Derandomized tensoring:
  \[
  A \otimes_C B = \text{Proj}[\hat{X} \cdot \hat{C} \cdot \hat{X}],
  \]
  where 
  * $\hat{X}$ depends on $A$ and $B$, 
  * $\hat{C} = \text{Id}(n_A n_B) \otimes C$. 

• Graph $\widehat{C}$
Graph $\tilde{X}$
Proof of Theorem

• **Lemma.** Let $S$ be the space

$$S = (1_{n_A})^\perp \otimes (1_{n_B})^\perp \otimes 1_d$$

The second eigenvalue of this projection is

$$\lambda\left(\text{Proj}[\hat{X}\hat{C}\hat{X}]\right) = \max_{x \in S} \frac{|\langle \hat{X}\hat{C}\hat{X} \cdot x, x \rangle|}{\langle x, x \rangle}.$$ 

• We decompose $S$ into

$$S = \left(1_{n_A}^\perp \otimes 1_{n_B}^\perp \otimes 1_d^\parallel\right) \oplus \left(1_{n_A}^\parallel \otimes 1_{n_B}^\perp \otimes 1_d^\parallel\right) \oplus \left(1_{n_A}^\perp \otimes 1_{n_B}^\perp \otimes 1_d^\parallel\right).$$

• Show that

$$x_1 \in S_1 \implies |\langle \hat{X}\hat{C}\hat{X} \cdot x, x \rangle| \leq \lambda_A \cdot \langle x, x \rangle$$

$$x_2 \in S_2 \implies |\langle \hat{X}\hat{C}\hat{X} \cdot x, x \rangle| \leq \lambda_B \cdot \langle x, x \rangle$$

$$x_3 \in S_3 \implies |\langle \hat{X}\hat{C}\hat{X} \cdot x, x \rangle| \leq m(\lambda_A, \lambda_B, \lambda_C) \cdot \langle x, x \rangle$$

• Deduce that if $x \in S$ then

$$\frac{|\langle \hat{X}\hat{C}\hat{X} \cdot x, x \rangle|}{\langle x, x \rangle} \leq \max\left(\lambda_A, \lambda_B, m(\lambda_A, \lambda_B, \lambda_C)\right)$$
Extensions

- This idea can be also be used to get a different analysis of the derandomized square

\[ \lambda(A \otimes C) \leq \lambda_A^2 + \lambda_C \cdot (1 - \lambda_A^2) \]

- We can also create a derandomized zig-zag product

\[ \lambda(A \odot C B) \leq \lambda_A + \lambda_B + \lambda_B^2 + \lambda_C \cdot (1 - \lambda_B^2) \]

smaller degree than the original zig-zag product, at the cost of slightly worse expansion.
Conclusion

- There is a coding theoretic motivation behind finding graph products with good expansion properties and small degree.

- We can define derandomized version of known products, decreasing the degree a lot while only slightly worsening the expansion.

- The analysis is done by looking at the product as a projection of a larger graph, whose adjacency matrix we can express easily.

- These tools can be used to obtain bounds the expansion of other graph products.