More efficient and flexible Flag-Algebras coming from polynomial optimization

Daniel Brosch

SIAM AG21, 20 August 2021

Tilburg University

Example problem: Triangle free graphs

Maximize the edge density of a graph while avoiding triangles, as the number of vertices n approaches infinity.

density of K_2 in G

$$\mathrm{ex}(K_2; \triangle) := \lim_{n\to\infty} \max_{G} K_2$$

s.t. G is $\triangle$-free graph, $|V(G)| = n$.

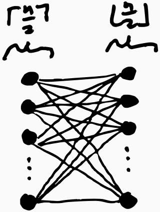
maximize

avoid

Lower bound for the edge density in triangle free graphs

We can determine a lower bound by constructing a sequence of triangle free graphs.

$$K_{\lceil \frac{n}{2} \rceil, \lfloor \frac{n}{2} \rfloor} =$$

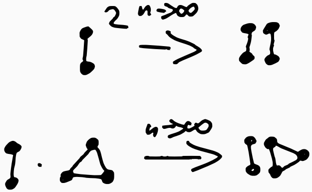


$$\Rightarrow \mathrm{ex}(\,|\,;\triangle) \geq \lim_{n\to\infty} \frac{\lceil \frac{n}{2} \rceil \cdot \lfloor \frac{n}{2} \rfloor}{\binom{n}{2}} = \frac{1}{2}.$$

But how can we determine an upper bound?

Flag-Algebras [Razborov 2007]

What happens when we multiply two densities?

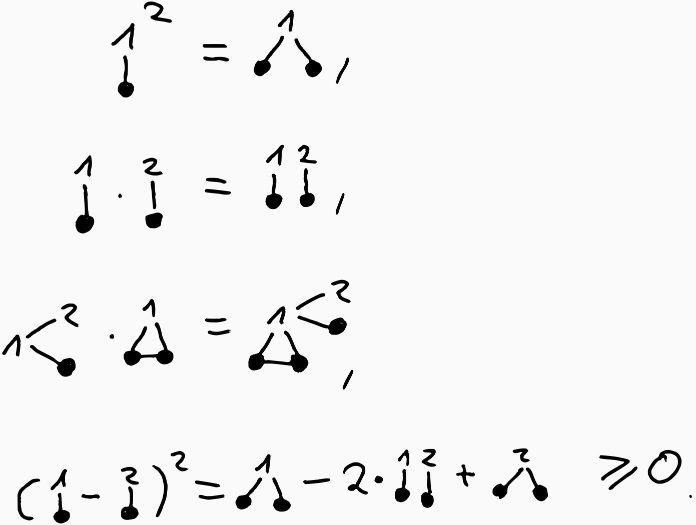


In the limit we simply glue together the two graphs!

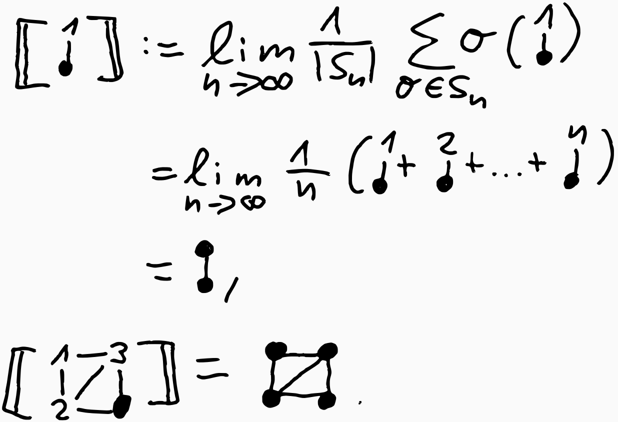
The Flag-Algebra of graphs

Extend this action to partially labelled graphs ("Flags") and extend to a vectorspace over the reals to obtain the Flag-Algebra of graphs.

Flag-Algebras: Examples



We can unlabel a Flag by symmetrization.



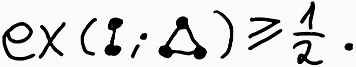
Flag-Sums of Squares

As with polynomial optimization, sums of squares of linear combinations of graph densities are nonnegative

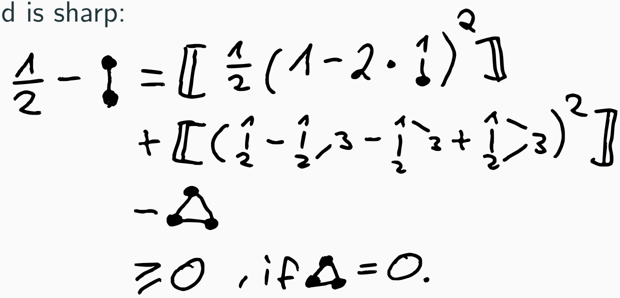
$$\sum_i \left(\sum_j c_{ij} d(G_{ij}) \right)^2 \geq 0,$$

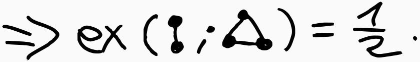
and can be optimized over by semidefinite programming.

We saw earlier that



This bound is sharp:



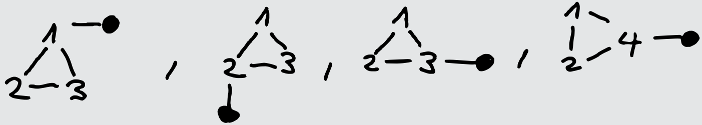


Problem 1: Efficiency

Flag-SOS can be solved using semidefinite programming relaxations, but the hierarchies grow very quickly!

But: The hierarchies have some symmetries!

Example for symmetries between Flags

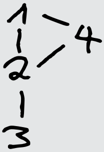


are all different Flags, i.e. have their own rows and columns in the SDP.

In order to exploit the symmetry, we first rewrite the (non-limit) problems as polynomial optimization problems.

Graphs as monomials

We can describe graphs by monomials in binary variables x_{ij} with $i < j$, which correspond to edges.



$$\triangleq x_{12} x_{14} x_{23} x_{24} .$$

Exploiting the symmetry: Back to polynomial optimization

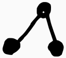
Graph densities as symmetric polynomials

Graph densities (and their linear combinations) are exactly the fully symmetric polynomials according to the action

$$\sigma(x_{ij}) = x_{\sigma(i)\sigma(j)},$$

for $\sigma \in S_n$.

$$= \lim_{n\to\infty} \frac{1}{\binom{n}{2}} \sum_{i<j} x_{ij},$$



$$= \lim_{n\to\infty} \frac{1}{n(n-1)(n-2)} \sum_{\substack{i,j,k \\ \text{different}}} x_{ij} x_{jk}$$

Symmetry reduction

The symmetry was partially exploited by Raymond, Saunderson, Singh and Thomas in 2017, and it was shown that the reduced hierarchies converge to the usual Flag-SOS hierarchies.

We fully exploited the symmetry to obtain more efficient, but equivalent hierarchies.

Symmetry reduction: main idea

- Semidefinite programming is convex.
- Convex combinations of optimal solutions are optimal solutions.
- There exists a symmetric (= invariant) optimal solution (by averaging over the symmetry).
- The set of invariant matrices forms a matrix algebra.
- These can be block-diagonalized by Artin-Wedderburn theory (which was specialised to symmetry reduction for polynomial optimization by Gatermann and Parrilo).

A very common approach for S_n symmetry reduction

Permutation modules M^λ are very well understood S_n-modules, given by partitions λ. If we can find an isomorphism between permutation modules and the underlying S_n-module (here the polynomial ring with the action of S_n), we can easily determine the block-diagonalization.

Here, such an isomorphism does not exist. But we can decompose the polynomial ring into quotients of permutation modules:

$$\mathbb{R}[X_{ij}] \simeq \bigoplus_{\text{Graphs } G \text{ up to isomorphism}} M^{\lambda(G)}/F(G),$$

where $F(G)$ is a subgroup of $\mathrm{Aut}(G)$, and acts on $M^{\lambda(G)}$ by permuting rows.

Understanding quotients of permutation modules

First main result

We found an efficient algorithm to decompose quotients of the form

$$M^\lambda / F$$

into irreducible Specht modules. This can then be used to symmetry reduce a wide variety of problems with S_n symmetry.

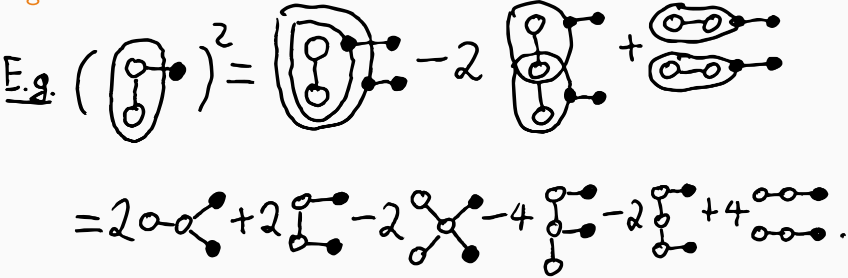
Already found a different application to the crossing number of the complete bipartite graph (joint work with **Sven Polak**).

Reduced Flag-SOS

Second main result

We determined a fully symmetry reduced Flag-SOS hierarchy, which is equivalent, but more efficient than the usual hierarchies.

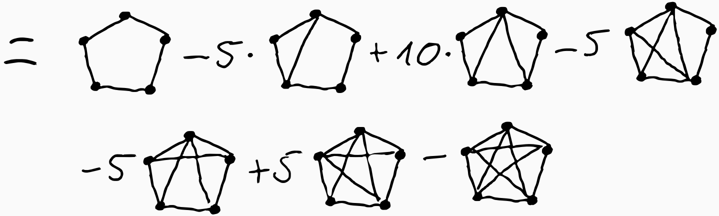
Here vertices are not explicitly labeled, but instead grouped together.



Prioritizing small graphs

In practice one often has problems with small, dense graphs, for example when working with induced subgraph densities.

Induced C_5 density



Prioritizing small graphs

We can determine different symmetry reduced hierarchies with only small graphs by:

- Partially breaking symmetry for $k \ll n$:

$$S_k \times S_{n-k}.$$

- Fully block-diagonalizing the small side and only considering the trivial isotypic component of the bigger side:

$$\left(\bigoplus_i V_i^{S_k}\right) \times V_0^{S_{n-k}}.$$

- Applying a Moebius transformation on the small part to create additional orthogonal relations.

This results in a much sparser hierarchy only involving small, but dense graphs.

Flag-SOS software

We implemented a Julia library that implements the reduced Flag-SOS hierarchies:

- Fully reduced limit hierarchies.
- Support for extensions to different Flag-Algebras, such as permutations, directed graphs, hypergraphs, point order types,. . .
- Can also generate hierarchies for fixed finite n or variable finite n (with polynomial coefficients in the second variant).

Problem 2: Flexibility

Flag-SOS are not useful for so-called degenerate problems. For example,

$$\mathrm{ex}(K_2 ; C_4) = 0.$$

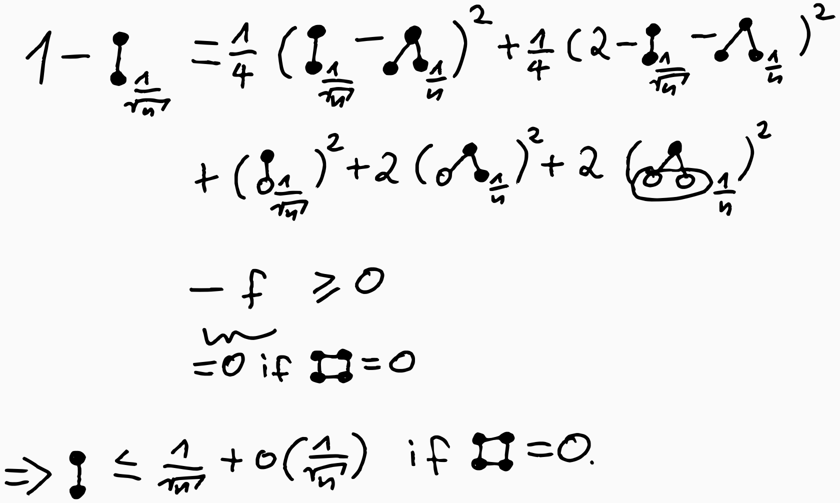
Here the edge density approaches zero as n grows. Thus we are instead interested in the rate at which the densities approach zero.

Solving degenerate problems

One approach one can find in the literature is to construct a sequence of SOS-certificates for each n.

Instead, we can determine different limit hierarchies by rescaling variables depending on n. This allows for a single compact certificate for the rate of densities in the limit.

Degenerate problem certificate example



(This bound is sharp in the limit.)

Conclusion

- Fully symmetry reduced Flag-SOS hierarchies for both sparse and dense graphs
- Extendable Flag-SOS Julia package implementing the hierarchies
- Theory and algorithms for quotients of permutation modules
- New kind of limit hierarchies for degenerate problems

Preprint and Julia package should be online soon!