Flagmatic

Jakub Sliačan

University of Warwick

About...

Flagmatic | Home | 3-Graphs | Graphs | Oriented Graphs

Queen Mary University of London

Flagmatic

A tool for researchers in extremal graph theory.

Download Flagmatic | View Project on GitHub

Flagmatic is an application that uses flag algebras and semi-definite programming to find bounds on Turán density and related problems, using the method of Razborov.

Flagmatic is designed in a way that means it can be adapted to handle different kinds of problems. Currently, it can solve graph, oriented graph and 3-graph problems. In fact, it was originally created to solve 3-graph problems.

Flagmatic 2.0 is a reinvention of Flagmatic as a Sage package.

To download, please use the link above. Also, you can read the User's Guide.

You may need to download the Mac binary of CSDP. (More information about CSDP here.)

For Flagmatic 1.5, see the old website.

Problem type

Maximize induced density of a small H in a big F-free G.

Example

Maximize the density of  *in a* *-free G.*

Answer

$\phi($$) \leq 1/2$. *Complete balanced bipartite G:* $\phi($$) \geq 1/2$.

How?

Context:

$$\phi(\overset{\bullet}{\underset{\bullet}{|}}) = \lim_{n\to\infty} \max_{|G|=n} d(\overset{\bullet}{\underset{\bullet}{|}}, G)$$

Rewrite:

$$d(\overset{\bullet}{\underset{\bullet}{|}}, G) = \sum_{|F|=k} d(\overset{\bullet}{\underset{\bullet}{|}}, F) d(F, G)$$

Do not know $d(F, G)$, but $\sum_{|F|=k} d(F, G) = 1$.

Bound:

$$d(\overset{\bullet}{\underset{\bullet}{|}}, G) \leq \max_{|F|=k} d(\overset{\bullet}{\underset{\bullet}{|}}, F) \qquad \text{(poor)}$$

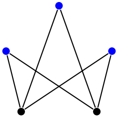
Need a better bound

The above bound is rarely sharp.

Example

$$\begin{aligned} d(\bullet\!\!-\!\!\bullet, G) &\leq \max_{|F|=3} d(\bullet\!\!-\!\!\bullet, F) \\ &= d(\bullet\!\!-\!\!\bullet, \wedge) = 2/3 \end{aligned}$$

Only sharp if every subgraph of G on 3 vertices is a $\wedge$. Impossible:



Account for subgraph overlaps

$G_{\color{red}\bullet}$ is G with one vertex red. Then $d(\overset{\bullet}{\underset{\color{red}\bullet}{|}}, G_{\color{red}\bullet})$ is the normalized degree of the red vertex.

1. $d(\overset{\bullet}{\underset{\color{red}\bullet}{|}}, G_{\color{red}\bullet})d(\overset{\bullet}{\underset{\color{red}\bullet}{|}}, G_{\color{red}\bullet})$ choosing two neighbours of ${\color{red}\bullet}$ (repetition allowed)
2. $d(\overset{\color{red}\bullet}{\bullet\wedge\bullet}, G_{\color{red}\bullet})$ choosing two neighbours of ${\color{red}\bullet}$ (repetition disallowed)

Negligible difference when G big. $\implies$ start with 1., switch to 2., uncolor (average over all choices of ${\color{red}\bullet}$ in G). Left with $\alpha d(\overset{\bullet}{\bullet\wedge\bullet}, G)$.

$$[\![d(\overset{\bullet}{\underset{\color{red}\bullet}{|}}, G_{\color{red}\bullet})d(\overset{\bullet}{\underset{\color{red}\bullet}{|}}, G_{\color{red}\bullet})]\!]_{\color{red}\bullet} \sim \frac{1}{3} d(\overset{\bullet}{\bullet\wedge\bullet}, G)$$

Manipulation

Vector $v = [d(\overset{\bullet}{\color{red}\bullet}, G_{\color{red}\bullet}), d(\overset{\bullet}{\underset{\color{red}\bullet}{|}}, G_{\color{red}\bullet})]$.

$$[\![vv^T]\!]_{\color{red}\bullet} \geq 0$$

Similarly, for every $A \succeq 0$,

$$[\![vAv^T]\!]_{\color{red}\bullet} \geq 0$$

$$\begin{aligned}
d(\overset{\bullet}{\underset{\bullet}{|}}, G) &= \sum_{|F|=3} d(\overset{\bullet}{\underset{\bullet}{|}}, F) d(F, G) \\
&\leq \sum_{|F|=3} d(\overset{\bullet}{\underset{\bullet}{|}}, F) d(F, G) + [\![vAv^T]\!] \qquad \text{with } A \succeq 0 \\
&= \sum_{|F|=3} \left(d(\overset{\bullet}{\underset{\bullet}{|}}, F) + c_F \right) d(F, G) \\
&\leq \max_{|F|=3} d(\overset{\bullet}{\underset{\bullet}{|}}, F) + c_F
\end{aligned}$$

Delegating tasks to the PC

Clearly, the proces was rather systematic. Need to know: density graphs, forbidden graphs. The rest can be done by the PC.

Optimization:

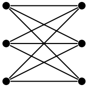
$$\min \gamma :$$

$d($  $, F) + c_F \leq \gamma$,for all F

$$A \succeq 0$$

Mantel in Flagmatic

Maximise  in a graph without copies of .
Recall $\phi($$) \leq 1/2$. Extremal graph is complete balanced bipartite:



In Flagmatic:

```
P = GraphProblem(3, forbid=3:121323, density=2:12)
P.solve_sdp()
```

Response:

```
Writing SDP input file...
Running SDP solver...
Returncode is 0. Objective value is 0.50000001.
Checking numerical bound...
```

New stuff

Add ingredients as you like (ingredient = graph inequality).

Example

Assume that the number or edges on randomly sampled 4 vertices from G is as in $\mathbb{G}(n, 1/2)$. Is it true that $p(H, G) = K_2^{e(H)}$ for all graphs H.

In Flagmatic:

```
P = GraphAssumptionsProblem(4, density=[(4:12233414, 8),
    (4:1223341424, 8), (4:121314232434, 24)])
P.add_assumptions(a1, a2, a3, a4, a5, a6, a7)
P.solve_sdp()
```

Where $a1$ is the assumption that  $= 1/2^{-6}$, $a2$ is the assumption that  $= \binom{6}{1} 2^{-6}$, and so on.

Some more examples

Looking for $\rho := \min\{$$+$$\}$. Known: $1/33 > \rho > 1/34.7858$.
Flagmatic gives $\rho > 1/34.26$.

```
P = GraphProblem(7, density=[(4:, 1), (4:121314232434,
    1)], minimize=True)
P.solve_sdp()
```