Using SAGE for Search in Graph Theory

Stephen G. Hartke

Department of Mathematics

University of Nebraska–Lincoln

www.math.unl.edu/∼shartke2

hartke@unl.edu

Joint work with Jamie Radcliffe

Searching in Graph Theory

Want to be able to do computations to

find or enumerate or classify graphs

(or graph-like structures) with a specified property.

Moore Graphs

Def. A Moore graph is Δ-regular, has $\Delta^2 + 1$ vertices,
has diameter 2, and is triangle- and C_4-free.

For which Δ do Moore graphs exist?

Moore Graphs

Def. A Moore graph is Δ-regular, has $\Delta^2 + 1$ vertices,

has diameter 2, and is triangle- and C_4-free.

For which Δ do Moore graphs exist?

Thm. (Hoffman-Singleton 60s)

Only for $\Delta = 2, 3, 7$ and *possibly* 57.

Graph Packing and Decomposition

Def. A packing of copies of G in a graph H

is a set of edge-disjoint subgraphs $\{H_1, H_2, \ldots, H_k\}$ of H

such that $H_i \cong G$.

Def. A decomposition of H into copies of G is

a packing of copies of G

such that every edge of H appears in a copy.

Example

Does K_n decompose into copies of a perfect matching?

Example

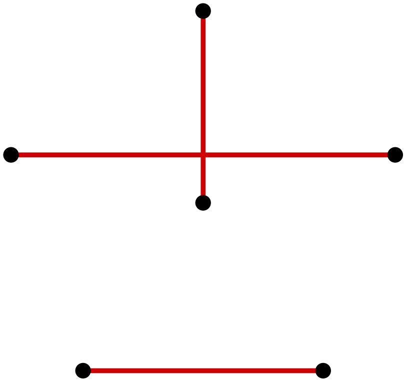
Does K_n decompose into copies of a perfect matching?

No, if n is odd: no perfect matchings.

Example

Does K_n decompose into copies of a perfect matching?

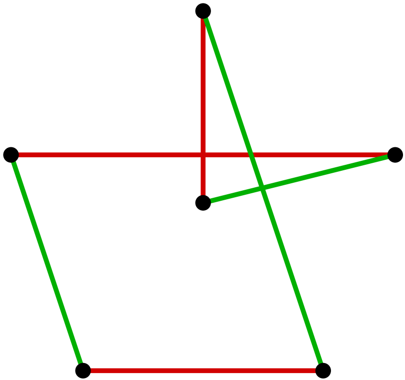
No, if n is odd: no perfect matchings. **Yes**, if n is even.



Example

Does K_n decompose into copies of a perfect matching?

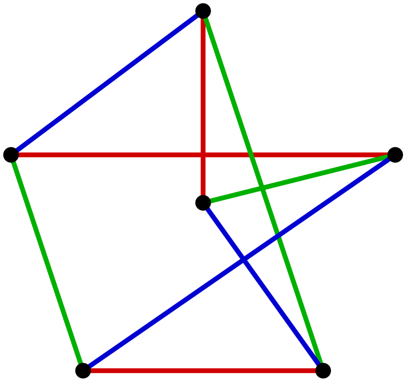
No, if n is odd: no perfect matchings. **Yes**, if n is even.



Example

Does K_n decompose into copies of a perfect matching?

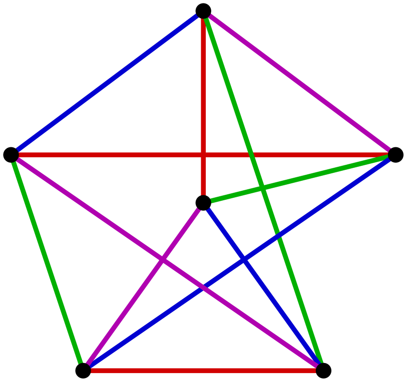
No, if n is odd: no perfect matchings. **Yes**, if n is even.



Example

Does K_n decompose into copies of a perfect matching?

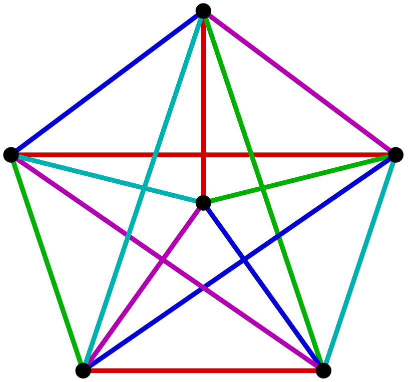
No, if n is odd: no perfect matchings. **Yes**, if n is even.



Example

Does K_n decompose into copies of a perfect matching?

No, if n is odd: no perfect matchings. **Yes**, if n is even.



Decomposition Problems

Conj. (Ringel) If T is a tree with r edges,
then K_{2r+1} decomposes into copies of T.

Implied by the Graceful Labeling Conj by Rosa.

Decomposition Problems

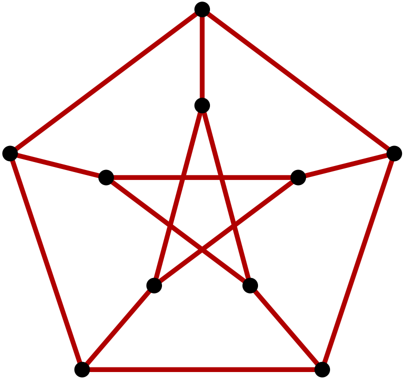
Conj. (Ringel) If T is a tree with r edges,
then K_{2r+1} decomposes into copies of T.

Implied by the Graceful Labeling Conj by Rosa.

Conj. (El-Zanati) If G is non-complete and has r edges,
then K_{2r+1} decomposes into copies of G.

Petersen Graph

Does K_{10} decompose into copies of the Petersen graph P?

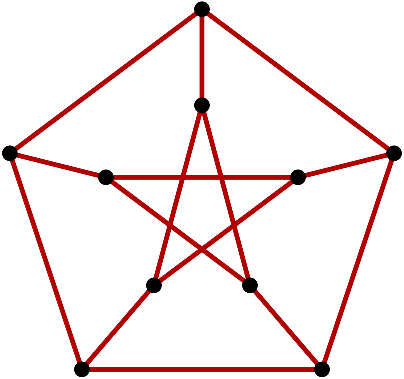


Petersen Graph

Does K_{10} decompose into copies of the Petersen graph P?

P is a Moore graph:

3-regular,
triangle- and C_4-free,
girth 5



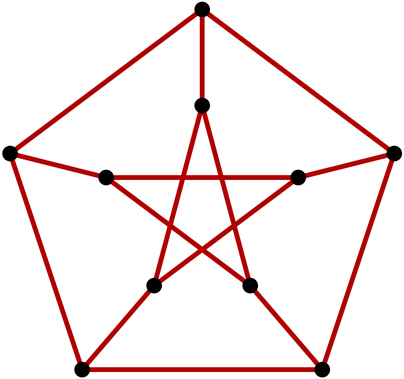
Petersen Graph

Does K_{10} decompose into copies of the Petersen graph P?

P is a Moore graph:

3-regular,
triangle- and C_4-free,
girth 5

P has 15 edges,
K_{10} has 45 edges.
$\Rightarrow$ 3 copies of P.



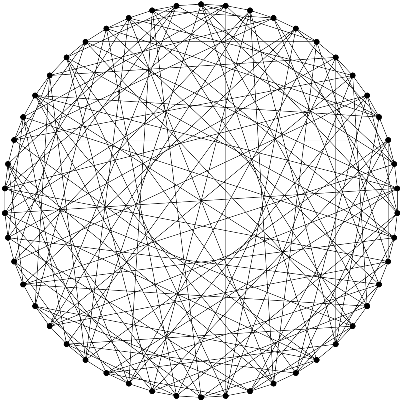
Petersen Graph

Does K_{10} decompose into copies of the Petersen graph P?

No. Elegant proof using eigenvalues by Fan and Schwenk.

Hoffman-Singleton Graph

Does K_{50} decomp into copies of Hoffman-Singleton graph HS?



picture by R.A. Slitherland

Hoffman-Singleton Graph

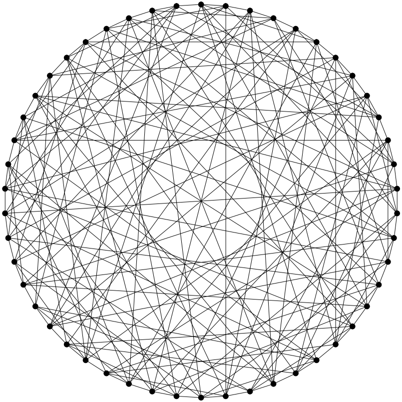
Does K_{50} decomp into copies of Hoffman-Singleton graph HS?

HS is a Moore graph:

7-regular,

triangle- and C_4-free,

girth 5



picture by R.A. Slitherland

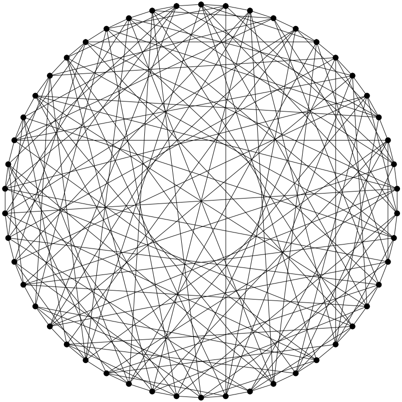
Hoffman-Singleton Graph

Does K_{50} decomp into copies of Hoffman-Singleton graph HS?

HS is a Moore graph:

7-regular,
triangle- and C_4-free,
girth 5

HS has 175 edges,
K_{50} has 1225 edges.
$\Rightarrow$ 7 copies of HS.



picture by R.A. Slitherland

Hoffman-Singleton Graph

Does K_{50} decomp into copies of Hoffman-Singleton graph HS?

Eigenvalue proof does not work.

Hoffman-Singleton Graph

Does K_{50} decomp into copies of Hoffman-Singleton graph HS?

Eigenvalue proof does not work.

How many copies of HS can be packed into K_{50}?

Hoffman-Singleton Graph

Does K_{50} decomp into copies of Hoffman-Singleton graph HS?

Eigenvalue proof does not work.

How many copies of HS can be packed into K_{50}?

Meszka and Siagiova can pack 5 copies using voltage graphs.

Hoffman-Singleton Graph

Does K_{50} decomp into copies of Hoffman-Singleton graph HS?

Eigenvalue proof does not work.

How many copies of HS can be packed into K_{50}?

Meszka and Siagiova can pack 5 copies using voltage graphs.

This is a finite problem. Can we solve it using computer search?

Integer Programming

An integer program IP is an optimization problem of the form

$$\begin{array}{rrl} \max & c^T x & \\ \text{subject to} & Ax & \leq b \\ & x & \geq 0 \\ & x & \in \mathbb{Z} \end{array}$$

Integer Programming

An integer program IP is an optimization problem of the form

$$\begin{array}{rrl} \max & c^T x & \\ \text{subject to} & Ax & \leq b \\ & x & \geq 0 \\ & x & \in \mathbb{Z} \end{array}$$

Linear programming relaxation drops the integral restriction.

LPs can be solved in polynomial time.

In practice, the best general method for attacking exponentially-sized problems.

Feasibility Testing

To search for a decomposition, we construct an IP such that

feasible solutions correspond to the desired decomposition.

Petersen Graph

$\forall$ edge e and copy i, indicator variables $x_e^i \in \{0, 1\}$
indicating whether edge e is in copy i.

Constraints:

partition: $\forall$ edge e, $\sum_i x_e^i = 1$

regular: $\forall$ copy $i, \forall v$, $\sum_{e \ni v} x_e^i = 3$

Petersen Graph

$\forall$ edge e and copy i, indicator variables $x_e^i \in \{0, 1\}$
indicating whether edge e is in copy i.

$\forall$ vertices u, v, w, copy i, ind vars $y^i_{u,v\to w} \in \{0, 1\}$
indicating whether w is a common nbr of u and v in copy i.

Constraints:

partition:	$\forall$ edge e,	$\sum_i x_e^i = 1$
regular:	$\forall$ copy $i, \forall v$,	$\sum_{e \ni v} x_e^i = 3$
Moore graph:	$\forall i, \forall u, v$,	$x^i_{\{u,v\}} + \sum_{w \neq u,v} y^i_{u,v\to w} = 1$

Petersen Graph

$\forall$ edge e and copy i, indicator variables $x_e^i \in \{0, 1\}$
indicating whether edge e is in copy i.

$\forall$ vertices u, v, w, copy i, ind vars $y^i_{u,v\to w} \in \{0, 1\}$
indicating whether w is a common nbr of u and v in copy i.

Constraints:

partition:	$\forall$ edge e,	$\sum_i x_e^i = 1$
regular:	$\forall$ copy $i, \forall v$,	$\sum_{e \ni v} x_e^i = 3$
Moore graph:	$\forall i, \forall u, v$,	$x^i_{\{u,v\}} + \sum_{w \neq u,v} y^i_{u,v\to w} = 1$

These properties uniquely determine the Moore graphs.
Feasible solutions to the IP are the desired decompositions.

Feasibility Testing

To solve an IP, we use branch-and-bound:

Some variables are fixed to 0 or 1, and

the LP relaxation is tested for feasibility.

If infeasible, then can't have that partial decomposition.

Feasibility Testing

To solve an IP, we use branch-and-bound:

Some variables are fixed to 0 or 1, and

the LP relaxation is tested for feasibility.

If infeasible, then can't have that partial decomposition.

Problem! By just renaming vertices, get another solution.

Presence of multiple sols by symmetry

makes branch-and-bound slower!

Feasibility Testing

To solve an IP, we use branch-and-bound:

Some variables are fixed to 0 or 1, and

the LP relaxation is tested for feasibility.

If infeasible, then can't have that partial decomposition.

Problem! By just renaming vertices, get another solution.

Presence of multiple sols by symmetry

makes branch-and-bound slower!

Solution: Only check one rep from each equivalence class.

Canonical Representative

We pick a canonical rep from each equivalence class.

We test whether each partial decomp is a canonical rep.

Discard those that are not.

Canonical Representative

We pick a canonical rep from each equivalence class.

We test whether each partial decomp is a canonical rep.

Discard those that are not.

Determining the canonical rep is computationally expensive.

(closely related to graph isomorphism)

Canonical Rep Functions

Read and Faradžev: Represent partial packing as a matrix.

Define canonical rep as the max under lex order by rows.

Canonical Rep Functions

Read and Faradžev: Represent partial packing as a matrix.

Define canonical rep as the max under lex order by rows.

For the Petersen graph P:

$$\begin{bmatrix} - & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ - & - & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ - & - & - & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ - & - & - & - & 0 & 0 & 0 & 0 & 1 & 1 \\ - & - & - & - & - & 0 & 1 & 0 & 1 & 0 \\ - & - & - & - & - & - & 0 & 1 & 0 & 1 \\ - & - & - & - & - & - & - & 0 & 0 & 1 \\ - & - & - & - & - & - & - & - & 1 & 0 \\ - & - & - & - & - & - & - & - & - & 0 \\ - & - & - & - & - & - & - & - & - & - \end{bmatrix}$$

Canonical Rep Functions

Read and Faradžev: Represent partial packing as a matrix.

Define canonical rep as the max under lex order by rows.

Use same definition for a graph packing.

$$\begin{bmatrix}
- & 3 & 3 & 3 & 2 & 2 & 2 & 0 & 0 & 0 \\
- & - & 2 & 2 & 3 & 2 & 0 & 3 & 0 & 0 \\
- & - & - & 0 & 2 & 3 & 3 & 0 & 2 & 0 \\
- & - & - & - & 0 & 0 & 2 & 2 & 3 & 3 \\
- & - & - & - & - & 3 & 0 & 2 & 0 & 3 \\
- & - & - & - & - & - & 0 & 0 & 3 & 2 \\
- & - & - & - & - & - & - & 3 & 2 & 3 \\
- & - & - & - & - & - & - & - & 3 & 2 \\
- & - & - & - & - & - & - & - & - & 2 \\
- & - & - & - & - & - & - & - & - & -
\end{bmatrix}$$

Canonical Rep Functions

Read and Faradžev: Represent partial packing as a matrix.

Define canonical rep as the max under lex order by rows.

Better: order by copy 2 first, then copy 1.

$$\begin{bmatrix}
- & 2 & 2 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\
- & - & 0 & 0 & 2 & 2 & 0 & 0 & 0 & 0 \\
- & - & - & 0 & 0 & 0 & 2 & 2 & 0 & 0 \\
- & - & - & - & 0 & 0 & 0 & 0 & 2 & 2 \\
- & - & - & - & - & 0 & 2 & 0 & 2 & 0 \\
- & - & - & - & - & - & 0 & 2 & 0 & 2 \\
- & - & - & - & - & - & - & 0 & 0 & 2 \\
- & - & - & - & - & - & - & - & 2 & 0 \\
- & - & - & - & - & - & - & - & - & 0 \\
- & - & - & - & - & - & - & - & - & -
\end{bmatrix},
\begin{bmatrix}
- & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\
- & - & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\
- & - & - & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\
- & - & - & - & 0 & 1 & 0 & 1 & 0 & 0 \\
- & - & - & - & - & 0 & 0 & 0 & 0 & 1 \\
- & - & - & - & - & - & 0 & 0 & 1 & 0 \\
- & - & - & - & - & - & - & 0 & 1 & 0 \\
- & - & - & - & - & - & - & - & 0 & 1 \\
- & - & - & - & - & - & - & - & - & 1 \\
- & - & - & - & - & - & - & - & - & -
\end{bmatrix}$$

Allows us to fix the first copy.

Implementation

Originally implemented in C using the COIN LP library.

COIN: COmputational INfrastructure for Operations Research

`www.coin-or.org`

Open-source (but CPL), sponsored by IBM and ppl at Lehigh

OSI: Open Solver Interface, for calling LP and IP solvers

Implementation

Originally implemented in C using the COIN LP library.

COIN: COmputational INfrastructure for Operations Research

Problems: C code was unwieldy for experimentation.

Wanted to use GAP for group calculations.

Implementation

Originally implemented in C using the COIN LP library.

COIN: COmputational INfrastructure for Operations Research

Problems: C code was unwieldy for experimentation.

Wanted to use GAP for group calculations.

Solution: Python and SAGE!

Python and SAGE

Pros:

- Python is a readable, expressive high-level language.
  Great for experimentation!
- SAGE unites many open programs in a coherent way.
- SAGE is free—cost and freedom.
- SAGE has its own useful math code (ie, NICE).
- SAGE is documented.
- PyCOIN Python interface to COIN generated by SWIG.

Python and SAGE

Cons:

- New language to learn.
- Missing from GAP interface: Stabilizer, CosetRep.
- PyCOIN has memory leaks.
- Python is slower than straight C.

Computational Results

Reimplemented in Python and SAGE over Thanksgiving 2007.

Run on a Pentium IV 3.4GHz Linux PC.

For Petersen graph, IP has 1215 vars, 3124 constraints

Without canonical rep func, takes 30 mins and 776 nodes.

With canonical rep func, takes a few secs and 66 nodes.

Computational Results

Reimplemented in Python and SAGE over Thanksgiving 2007.

Run on a Pentium IV 3.4GHz Linux PC.

For Petersen graph, IP has 1215 vars, 3124 constraints

Without canonical rep func, takes 30 mins and 776 nodes.

With canonical rep func, takes a few secs and 66 nodes.

For Hoffman-Singleton graph,

IP has 361374 vars, 2529618 constraints

Work focusing now on refining the implementation details.

Other Tools?

McKay has a powerful canonical rep function in nauty.

Implemented as NICE in SAGE by Miller.

To use it for search, need algo described yesterday by Miller.

Other Tools?

McKay has a powerful canonical rep function in nauty.

Implemented as NICE in SAGE by Miller.

To use it for search, need algo described yesterday by Miller.

Edge-labeled version of NICE?

Other Tools?

McKay has a powerful canonical rep function in nauty.

Implemented as NICE in SAGE by Miller.

To use it for search, need algo described yesterday by Miller.

- Edge-labeled version of NICE?

- Use of cvxopt and GLPK?

Future Work

Resolve the Hoffman-Singleton decomp of K_{50}

Use method for other decomposition problems:

Conj of El-Zanati: If G is non-complete and has r edges, then K_{2r+1} decomposes into copies of G.

Smallest unknown case: $G = K_6 - e$.

Future Work

Resolve the Hoffman-Singleton decomp of K_{50}

Use method for other decomposition problems:

> Conj of El-Zanati: If G is non-complete and has r edges,
> then K_{2r+1} decomposes into copies of G.
>
> Smallest unknown case: $G = K_6 - e$.

Use in proofs?

Using SAGE for Search in Graph Theory

Stephen G. Hartke

Department of Mathematics

University of Nebraska–Lincoln

www.math.unl.edu/∼shartke2

hartke@unl.edu

Joint work with Jamie Radcliffe