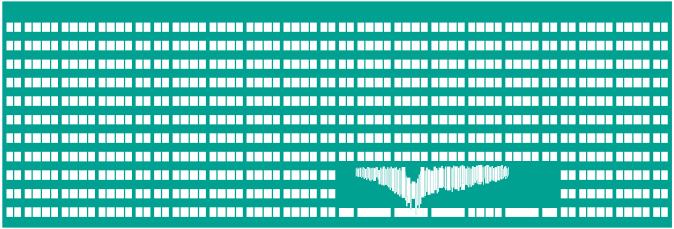
VŠB TECHNICKÁ UNIVERZITA OSTRAVA

VSB TECHNICAL UNIVERSITY OF OSTRAVA



www.vsb.cz

# Nice Application of Complete Graph Decomposition

Petr Kovář

joint work with

Dalibor Lukáš, Tereza Kovářová, Michal Kravčenko, Michal Merta

VSB – Technical University of Ostrava, Czech Republic

Seminár z teórie grafov,
11.3. 2021, Bratislava

# Outline of the talk

# Motivation

When solving real life problems

- large $n \times n$ matrix, $n \ldots$ millions
- numerical method

# Motivation

When solving real life problems

- large $n \times n$ matrix, $n$ . . . millions
- numerical method
- parallelize the computation
- $N$ processes, $N \times N$ blocks (submatrices) $B_{ij}$

$$\begin{pmatrix} B_{11} & B_{12} & \cdots & B_{1N} \\ \vdots & \vdots & \ddots & \vdots \\ B_{N1} & B_{N2} & \cdots & B_{NN} \end{pmatrix}$$

# Motivation

When solving real life problems

- large $n \times n$ matrix, $n$ ... millions
- numerical method
- parallelize the computation
- $N$ processes, $N \times N$ blocks (submatrices) $B_{ij}$

$$\begin{pmatrix} B_{11} & B_{12} & \cdots & B_{1N} \\ \vdots & \vdots & \ddots & \vdots \\ B_{N1} & B_{N2} & \cdots & B_{NN} \end{pmatrix}$$

We have a dense matrix!

# Motivation

When solving real life problems

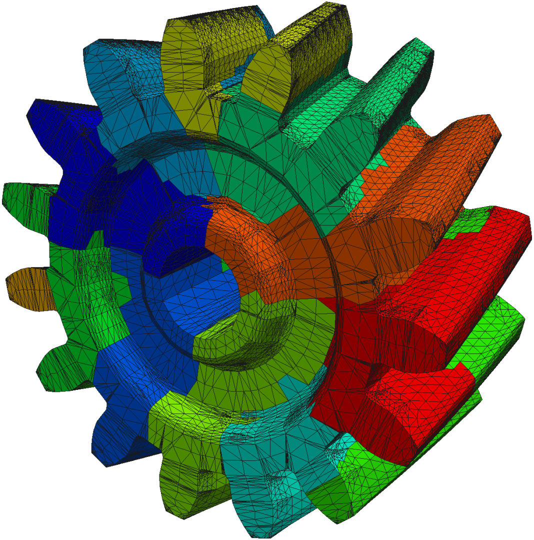
- large $n \times n$ matrix, $n \ldots$ millions
- numerical method
- parallelize the computation
- $N$ processes, $N \times N$ blocks (submatrices) $B_{ij}$

$$\begin{pmatrix} B_{11} & B_{12} & \cdots & B_{1N} \\ \vdots & \vdots & \ddots & \vdots \\ B_{N1} & B_{N2} & \cdots & B_{NN} \end{pmatrix}$$

We have a dense matrix!

- distribute $N$ blocks to each processor
- (geometrically) closely related blocks to the same processor

# FEM & BEM



Many points, split into N machines.

# Parallelization

Parallel machine without shared memory.

We prefer parallelization

- load balanced
- memory balanced

# Parallelization

Parallel machine without shared memory.

We prefer parallelization

- load balanced
- memory balanced

Computation of block $B_{ij}$, requires

- $i$-th and
- $j$-th parts of the geometry

# Parallelization

Parallel machine without shared memory.

We prefer parallelization

- load balanced
- memory balanced

Computation of block $B_{ij}$, requires

- $i$-th and
- $j$-th parts of the geometry

To each CPU as few different indices as possible.
(e.g. not all blocks from one row)

## Parallelization

Parallel machine without shared memory.

We prefer parallelization

- load balanced
- memory balanced

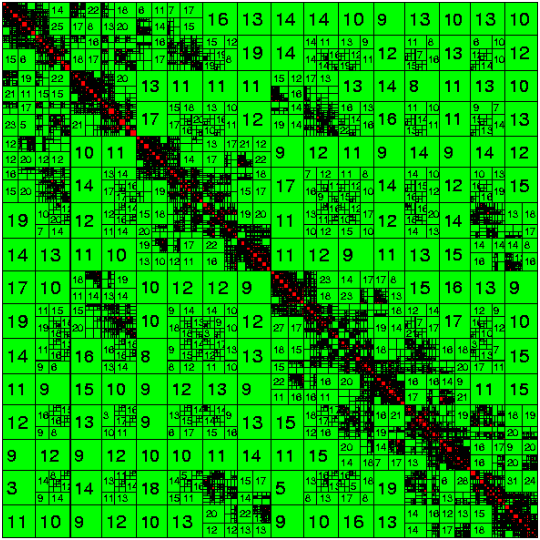
Computation of block $B_{ij}$, requires

- $i$-th and
- $j$-th parts of the geometry

To each CPU as few different indices as possible.
(e.g. not all blocks from one row)

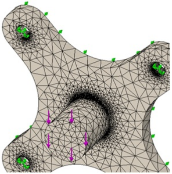
Not all $N^2$ blocks fit into the memory of one CPU!
(nor all $n^2$ elements of the matrix)

# Block matrix with numbers related to difficulty



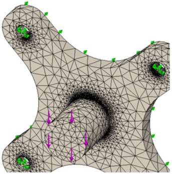
Question: which $N$ blocks on which CPU?

- one diagonal block (longest computation)
- $(N-1)$ non-diagonal blocks



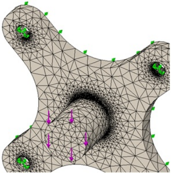
Question: which $N$ blocks on which CPU?

- one diagonal block (longest computation)
- $(N-1)$ non-diagonal blocks
- if $B_{ij}$ then also $B_{ji}$ (same CPU)
- if $B_{ij}$, $B_{ik}$, and $B_{lj}$ then also $B_{lk}$



Question: which $N$ blocks on which CPU?

- one diagonal block (longest computation)
- $(N-1)$ non-diagonal blocks
- if $B_{ij}$ then also $B_{ji}$ (same CPU)
- if $B_{ij}$, $B_{ik}$, and $B_{lj}$ then also $B_{lk}$

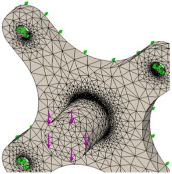


This translates into

- decomposing $K_N$ into $N$ subgraphs $G_1, G_2, \ldots, G_N$
- each with $(N-1)/2$ edges
- each with as few vertices as possible

Question: which $N$ blocks on which CPU?

- one diagonal block (longest computation)
- $(N-1)$ non-diagonal blocks
- if $B_{ij}$ then also $B_{ji}$ (same CPU)
- if $B_{ij}$, $B_{ik}$, and $B_{lj}$ then also $B_{lk}$



This translates into

- decomposing $K_N$ into $N$ subgraphs $G_1, G_2, \ldots, G_N$
- each with $(N-1)/2$ edges
- each with as few vertices as possible

First suppose $G_1, G_2, \ldots, G_N$ isomorphic to complete graph.

# Graph theory formulation

Decompose $K_N$ into $N$ copies of $K_k$ – dense subgraphs.
Necessary condition: $N = k^2 - k + 1$.

# Graph theory formulation

Decompose $K_N$ into $N$ copies of $K_k$ – dense subgraphs.
Necessary condition: $N = k^2 - k + 1$.

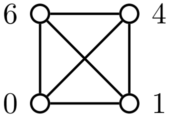
Particularly easy if the decomposition is cyclic:

## Definition (Graceful labeling)

Let $G$ be a graph with $m$ edges and a vertex labeling $\lambda : V(G) \to \{0, 1, \ldots, m\}$. The *length* of an edge $xy$ is

$$\ell(x, y) = \min\{|\lambda(x) - \lambda(y)|,\ 2m + 1 - |\lambda(x) - \lambda(y)|\}.$$

We call $f$ a *graceful* labeling if the set of edge lengths $\{\ell(x, y) : xy \in E(G)\} = \{1, 2, \ldots, m\}$.



# Graceful and $\rho$-labeling

The famous Graceful Tree Conjecture: “All trees graceful.”

## Graceful and $\rho$-labeling

The famous Graceful Tree Conjecture: "All trees graceful."

Other graphs are interesting as well

- $K_k$ is graceful iff $k \leq 4$

# Graceful and $\rho$-labeling

The famous Graceful Tree Conjecture: “All trees graceful.”

Other graphs are interesting as well

- $K_k$ is graceful iff $k \leq 4$
- unicyclic graphs

# Graceful and $\rho$-labeling

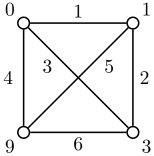
The famous Graceful Tree Conjecture: "All trees graceful."

Other graphs are interesting as well

- $K_k$ is graceful iff $k \leq 4$
- unicyclic graphs

**Definition ($\rho$-labeling)**

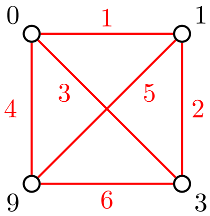
Let $G$ be a graph with $m$ edges nad a vertex labeling $f : V(G) \to \{0, 1, \dots, 2m\}$. The *length* of $xy$ is $\ell(x, y) = \min\{|\lambda(x) - \lambda(y)|,\ 2m + 1 - |\lambda(x) - \lambda(y)|\}$.
We say $\lambda$ is a *$\rho$-labeling* labeling if the set of edge lengths $\{\ell(x, y) : xy \in E(G)\} = \{1, 2, \dots, m\}$.



# $\rho$-labeling

**Theorem (Rosa 1967)**

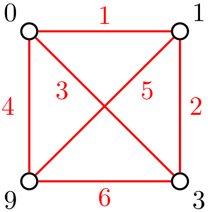
*A graph $G$ with $m$ edges allows a cyclic decomposition of $K_{2m+1}$ iff $G$ has a $\rho$-labeling.*



# $\rho$-labeling

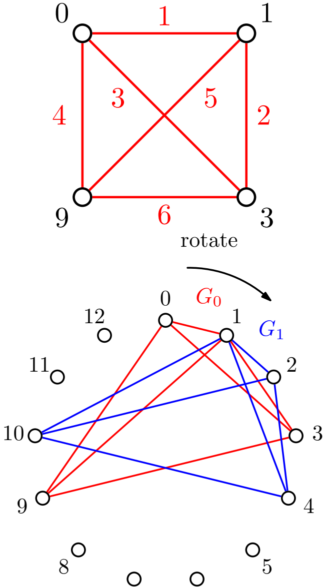
## Theorem (Rosa 1967)

*A graph $G$ with $m$ edges allows a cyclic decomposition of $K_{2m+1}$ iff $G$ has a $\rho$-labeling.*




- Not all complete graphs have a $\rho$-labeling,
- infinitely many complete graphs have.

# Example



# Equivalent formulation

## Definition (perfect difference sets)

A set of integers $\{a_1, a_2, \ldots, a_k\} \subseteq [0, N]$ such that every nonzero residue modulo $N$ can be uniquely expressed in the form $a_i - a_j$.

# Equivalent formulation

## Definition (perfect difference sets)

A set of integers $\{a_1, a_2, \ldots, a_k\} \subseteq [0, N]$ such that every nonzero residue modulo $N$ can be uniquely expressed in the form $a_i - a_j$.

## Example

$\{0, 1, 4, 6\}$ is a perfect difference set for $m = 6$:
$1 = 1 - 0$, $2 = 6 - 4$, $3 = 3 - 1$, $4 = 4 - 0$, $5 = 6 - 1$, $6 = 6 - 0$.

# Equivalent formulation

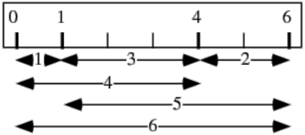
## Definition (perfect difference sets)

A set of integers $\{a_1, a_2, \ldots, a_k\} \subseteq [0, N]$ such that every nonzero residue modulo $N$ can be uniquely expressed in the form $a_i - a_j$.

## Example

$\{0, 1, 4, 6\}$ is a perfect difference set for $m = 6$:
$1 = 1 - 0,\ 2 = 6 - 4,\ 3 = 3 - 1,\ 4 = 4 - 0,\ 5 = 6 - 1,\ 6 = 6 - 0.$

*Perfect ruler* (Guy 1994) has $k$ distinct marks s.t. any distance $1, 2, 3, 4, \ldots, N$ can be measured. E.g. 0, 1 ,4, 6



## Example (Continued...)

Based on the perfect difference set $\{0, 1, 4, 6\}$ for $m = 6$ we can decompose $K_{2m+1} = K_{13}$ cyclically.

## Example (Continued...)

Based on the perfect difference set $\{0, 1, 4, 6\}$ for $m = 6$ we can decompose $K_{2m+1} = K_{13}$ cyclically.

- label vertices of $K_4$ by $0, 1, 4, 6$
- find a cyclic decomposition of $K_{13}$

## Example (Continued. . . )

Based on the perfect difference set $\{0, 1, 4, 6\}$ for $m = 6$ we can decompose $K_{2m+1} = K_{13}$ cyclically.

- label vertices of $K_4$ by $0, 1, 4, 6$
- find a cyclic decomposition of $K_{13}$

A 13 by 13 block matrix to 13 machines, so that:

- 1 computation of diagonal block + 12 non-diagonal blocks to each process
- 4 rows (and 4 columns) of the geometry to each process

## Example (Continued. . . )

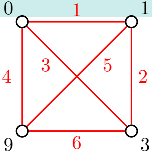
Based on the perfect difference set $\{0, 1, 4, 6\}$ for $m = 6$ we can decompose $K_{2m+1} = K_{13}$ cyclically.

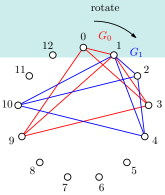
- label vertices of $K_4$ by $0, 1, 4, 6$
- find a cyclic decomposition of $K_{13}$

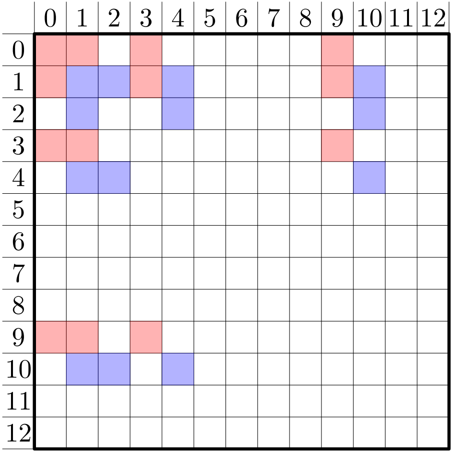
A 13 by 13 block matrix to 13 machines, so that:

- 1 computation of diagonal block + 12 non-diagonal blocks to each process
- 4 rows (and 4 columns) of the geometry to each process
- the higher $N$ ($N = 13$) the better ratio

# Example


0
1
9
3
1
2
3
4
5
6


rotate
$G_0$
$G_1$
0
1
2
3
4
5
6
7
8
9
10
11
12


0 1 2 3 4 5 6 7 8 9 10 11 12
0
1
2
3
4
5
6
7
8
9
10
11
12

# Yet another formulation

Decomposing a complete graph into complete subgraphs – this seems to be a design theory problem.

## Definition

A *block design (BIBD)* is a collection $B$ of $b$ subsets (called blocks) of a finite set $X$ of $v$ elements such that any element of $X$ is contained in the same number $r$ of blocks, every block has the same number $k$ of elements, and each pair of distinct elements appear together in the same number $\lambda$ of blocks.
A symmetric BIBDs are (also known as 2-designs) are denoted as $2-(v,k,\lambda)$ designs has $b=v$. ($b \geq v$ by Fisher's inequality.)

# Yet another formulation

Decomposing a complete graph into complete subgraphs – this seems to be a design theory problem.

## Definition

A *block design (BIBD)* is a collection $B$ of $b$ subsets (called blocks) of a finite set $X$ of $v$ elements such that any element of $X$ is contained in the same number $r$ of blocks, every block has the same number $k$ of elements, and each pair of distinct elements appear together in the same number $\lambda$ of blocks.
A symmetric BIBDs are (also known as 2-designs) are denoted as $2-(v,k,\lambda)$ designs has $b=v$. ($b \geq v$ by Fisher's inequality.)

In our case:

- $k = k$
- $v = N = k^2 - k + 1$
- $b = N$ (symmetric)
- $\lambda = 1$

# Constructions

Sufficient condition: $k-1$ is a prime power.

# Constructions

Sufficient condition: $k - 1$ is a prime power.

**Theorem (Singer 1934)**

*A perfect difference set with $k$ elements exists if $k - 1$ is a prime power.*

# Constructions

Sufficient condition: $k-1$ is a prime power.

**Theorem (Singer 1934)**

*A perfect difference set with $k$ elements exists if $k-1$ is a prime power.*

However, $K_7$ does not decompose $K_{43}$.
Neither $K_{11}$ does not decompose $K_{111}$.
...

# Constructions

Sufficient condition: $k-1$ is a prime power.

**Theorem (Singer 1934)**

*A perfect difference set with $k$ elements exists if $k-1$ is a prime power.*

However, $K_7$ does not decompose $K_{43}$.
Neither $K_{11}$ does not decompose $K_{111}$.
...

**Theorem (Hartke, Östergøard, Bryant, El-Zanati 2009)**

*There exists no $(K_6-e)$-decomposition of $K_{29}$.*

# Constructions

Sufficient condition: $k-1$ is a prime power.

**Theorem (Singer 1934)**

*A perfect difference set with $k$ elements exists if $k-1$ is a prime power.*

However, $K_7$ does not decompose $K_{43}$.
Neither $K_{11}$ does not decompose $K_{111}$.
...

**Theorem (Hartke, Östergøard, Bryant, El-Zanati 2009)**

*There exists no $(K_6 - e)$-decomposition of $K_{29}$.*

Overview for $k < 100$ on web (Baumert):
`http://www.ccrwest.org/diffsets/diff_sets/baumert.html`

# Main result (2012/2015)

# Main result (2012/2015)

...is the nice application!

# Main result (2012/2015)

. . . is the nice application!

Construction for certain (not all) values.

# Main result (2012/2015)

...is the nice application!

Construction for certain (not all) values.

It has been implemented and successfully tested:
Fast BEM matrices of size $n$ up to millions, distributed to hundreds of nodes $N$.

# Main result (2012/2015)

. . . is the nice application!

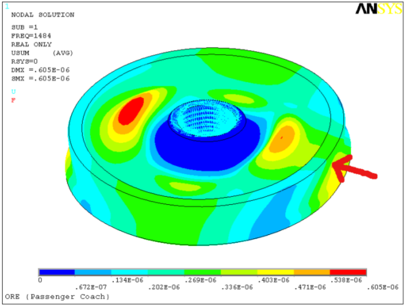
Construction for certain (not all) values.

It has been implemented and successfully tested:
Fast BEM matrices of size $n$ up to millions, distributed to hundreds of nodes $N$.

| n | N=31, k=6 | N=91, k=10 | N=133,k=12 |
|---|---|---|---|
| 12 288 | 175 MB, 1 s | 200 MB, 1 s | 207 MB, 1 s |
| 196 608 | 353 MB, 53 s | 280 MB, 25 s | 276 MB, 18 s |
| 786 432 | 999 MB, 294 s | 570 MB, 110 s | 535 MB, 99 s |
| 3 145 728 | | | 1911 MB, 596 s |

Format: average memory [MB], CPU time per process [s]

Motivation: railway wheel noise elimination by profiling




Courtesy of J. Szweda, Department of Mechanics, VŠB – TUO

# Additional constructions

Distribution of processes to $N$ CPUs needs not to be balanced (not isomorphic subgraphs).

Typical cluster has $2^t$ cores, e.g. $128$.

# Additional constructions

Distribution of processes to $N$ CPUs needs not to be balanced (not isomorphic subgraphs).

Typical cluster has $2^t$ cores, e.g. $128$. We know how to decompose $K_{133}$ into 133 subgraphs $K_{12}$.
Remove 5 rows and columns:

- preferably from different $K_{12}$ subgraphs
- obtain some $K_{12}$, $K_{11}$, $K_{10}$, maybe a few smaller
- or obtain many $K_{12}$, $K_{11}$, one $K_7$

# Additional constructions

Distribution of processes to $N$ CPUs needs not to be balanced (not isomorphic subgraphs).

Typical cluster has $2^t$ cores, e.g. $128$. We know how to decompose $K_{133}$ into 133 subgraphs $K_{12}$.
Remove 5 rows and columns:

- preferably from different $K_{12}$ subgraphs
- obtain some $K_{12}$, $K_{11}$, $K_{10}$, maybe a few smaller
- or obtain many $K_{12}$, $K_{11}$, one $K_7$

Or for 2014 cores decompose $K_{993}$ into 933 copies of $K_{31}$.

# Additional constructions

Distribution of processes to $N$ CPUs needs not to be balanced (not isomorphic subgraphs).

Typical cluster has $2^t$ cores, e.g. $128$. We know how to decompose $K_{133}$ into 133 subgraphs $K_{12}$.
Remove 5 rows and columns:

- preferably from different $K_{12}$ subgraphs
- obtain some $K_{12}$, $K_{11}$, $K_{10}$, maybe a few smaller
- or obtain many $K_{12}$, $K_{11}$, one $K_7$

Or for 2014 cores decompose $K_{993}$ into 933 copies of $K_{31}$.

A graph theorists say: That's cheating! Yet, it works. . .

# Additional constructions

Distribution of processes to $N$ CPUs needs not to be balanced (not isomorphic subgraphs).

Typical cluster has $2^t$ cores, e.g. $128$. We know how to decompose $K_{133}$ into 133 subgraphs $K_{12}$.
Remove 5 rows and columns:

- preferably from different $K_{12}$ subgraphs
- obtain some $K_{12}$, $K_{11}$, $K_{10}$, maybe a few smaller
- or obtain many $K_{12}$, $K_{11}$, one $K_7$

Or for 2014 cores decompose $K_{993}$ into 933 copies of $K_{31}$.

A graph theorists say: That's cheating! Yet, it works...

Computational time depends on the largest graph:
preferably few small and a many large dense graphs (complete or "almost" complete graphs).

# Graphs similar to complete graphs.

## Graphs similar to complete graphs.

If it has a $\rho$-labeling, then it can be used.

## Graphs similar to complete graphs.

If it has a $\rho$-labeling, then it can be used.

- all graphs with at most 11 edges have a $\rho$-labeling
- many classes of sparse graphs

## Graphs similar to complete graphs.

If it has a $\rho$-labeling, then it can be used.

- all graphs with at most 11 edges have a $\rho$-labeling
- many classes of sparse graphs

By hand up to $n = 31$:

- $\rho$-labeling of a graph with $(n-1)/2$ edges
- exceptions $n = 28, 29, \ldots$

## Graphs similar to complete graphs.

If it has a $\rho$-labeling, then it can be used.

- all graphs with at most 11 edges have a $\rho$-labeling
- many classes of sparse graphs

By hand up to $n = 31$:

- $\rho$-labeling of a graph with $(n-1)/2$ edges
- exceptions $n = 28, 29, \ldots$

...difficult (almost) as for complete graphs

## Graphs similar to complete graphs.

If it has a $\rho$-labeling, then it can be used.

- all graphs with at most 11 edges have a $\rho$-labeling
- many classes of sparse graphs

By hand up to $n = 31$:

- $\rho$-labeling of a graph with $(n-1)/2$ edges
- exceptions $n = 28, 29, \ldots$

...difficult (almost) as for complete graphs

**Decomposition needs not to be cyclic**

$K_7 - K_{3,3}$ decomposes $K_{25}$, yet no $\rho$-labeling.
reference?

## Subgraphs need not to be isomorphic

- for odd $N$ subgraphs can be isomorphic (exceptions!)
- for even $N$ subgraphs cannot be isomorphic (parity) (simple graph not, though isomorphic digraphs exist)

# Subgraphs need not to be isomorphic

- for odd $N$ subgraphs can be isomorphic (exceptions!)
- for even $N$ subgraphs cannot be isomorphic (parity)
  (simple graph not, though isomorphic digraphs exist)

## Lemma

*Let $r, s$ be odd. If $G$ decomposes $K_r$ into $r$ copies and $H$ decomposes $G[\overline{K_s}]$ into $s$ copies, then a dense graph $X$ on $|H|$ vertices decomposes $K_{rs}$ into $rs$ copies of $X$.*

# Subgraphs need not to be isomorphic

- for odd $N$ subgraphs can be isomorphic (exceptions!)
- for even $N$ subgraphs cannot be isomorphic (parity) (simple graph not, though isomorphic digraphs exist)

## Lemma

*Let $r, s$ be odd. If $G$ decomposes $K_r$ into $r$ copies and $H$ decomposes $G[\overline{K_s}]$ into $s$ copies, then a dense graph $X$ on $|H|$ vertices decomposes $K_{rs}$ into $rs$ copies of $X$.*

## Theorem

*If $r = p^2 - p + 1$ and $s = q^2 - q + 1$, then we can decompose $K_{rs}$ into $rs$ dense (for small $s$) isomorphic subgraphs on $pq$ vertices.*

## Example

Decompose $K_{147}$ $(147 = 7 \cdot 21)$ into 147 isomorphic subgraphs on 15 vertices.

# Subgraphs need not to be isomorphic

- for odd $N$ subgraphs can be isomorphic (exceptions!)
- for even $N$ subgraphs cannot be isomorphic (parity)
  (simple graph not, though isomorphic digraphs exist)

## Lemma

*Let $r, s$ be odd. If $G$ decomposes $K_r$ into $r$ copies and $H$ decomposes $G[\overline{K_s}]$ into $s$ copies, then a dense graph $X$ on $|H|$ vertices decomposes $K_{rs}$ into $rs$ copies of $X$.*

## Theorem

*If $r = p^2 - p + 1$ and $s = q^2 - q + 1$, then we can decompose $K_{rs}$ into $rs$ dense (for small $s$) isomorphic subgraphs on $pq$ vertices.*

## Example

Decompose $K_{147}$ $(147 = 7 \cdot 21)$ into 147 isomorphic subgraphs on 15 vertices. (Theoretical optimum: 13 vertices.)

## If $N$ is even...

If $q$ is even, we decompose $K_{pq}$ into $pq$ subgraphs $H_1, H_2, \ldots, H_{pq}$ (not isomorphic).

# If $N$ is even...

If $q$ is even, we decompose $K_{pq}$ into $pq$ subgraphs $H_1, H_2, \ldots, H_{pq}$ (not isomorphic).

## Lemma

*Let $p$ be odd and $q$ even. If $G$ decomposes $K_p$ into $p$ copies and $H$ decomposes $G[\overline{K_q}]$ into $q$ copies, then $K_{pq}$ can be decomposed into $pq$ dense subgraphs $X_1, X_2, \ldots, X_{pq}$ each on $|H|$ vertices.*

# If $N$ is even...

If $q$ is even, we decompose $K_{pq}$ into $pq$ subgraphs $H_1, H_2, \ldots, H_{pq}$ (not isomorphic).

**Lemma**

*Let $p$ be odd and $q$ even. If $G$ decomposes $K_p$ into $p$ copies and $H$ decomposes $G[\overline{K_q}]$ into $q$ copies, then $K_{pq}$ can be decomposed into $pq$ dense subgraphs $X_1, X_2, \ldots, X_{pq}$ each on $|H|$ vertices.*

Leads to a recursive construction.
Isolated values (case by case).

# Further approximations

Up to $N = 1000$ ($k \simeq 40$):

- constructing dense graphs using a greedy computer search

# Further approximations

Up to $N = 1000$ ($k \simeq 40$):

- constructing dense graphs using a greedy computer search
- corresponds to using more CPUs, not $N$ but $N' > N$
- not balanced for CPU (nor memory) load

Roughly $N' \doteq \frac{7}{5}N$.

# Further approximations

Up to $N = 1000$ ($k \simeq 40$):

- constructing dense graphs using a greedy computer search
- corresponds to using more CPUs, not $N$ but $N' > N$
- not balanced for CPU (nor memory) load

Roughly $N' \doteq \frac{7}{5}N$.

We beat this by theoretical constructions, though only for certain values – using more memory not more CPUs.
E.g. decomposing $K_{559}$ into 559 isomorphic subgraphs each on 28 vertices.

# Further approximations

Up to $N = 1000$ ($k \simeq 40$):

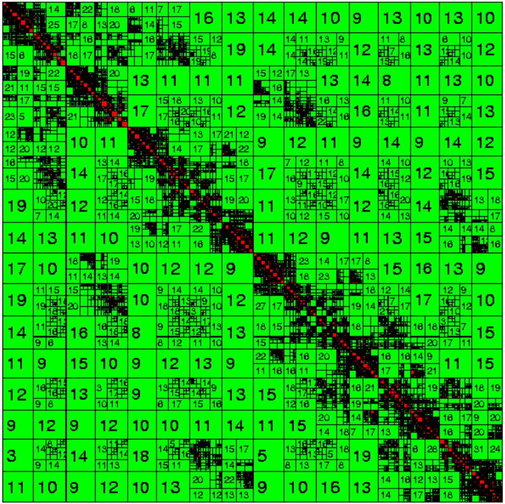
- constructing dense graphs using a greedy computer search
- corresponds to using more CPUs, not $N$ but $N' > N$
- not balanced for CPU (nor memory) load

Roughly $N' \doteq \frac{7}{5}N$.

We beat this by theoretical constructions, though only for certain values – using more memory not more CPUs.
E.g. decomposing $K_{559}$ into 559 isomorphic subgraphs each on 28 vertices.

Theoretical optimum: 25 vertices.

# Optimization?



. . . balance the sums among the subgraphs.

Thank you for your attention.