Extremal graph theory

Peter Hajnal

Bolyai Institute, University of Szeged, Hungary

Fall 2023

Reminder

Reminder

Reminder: Cliques

$K \subset V(G)$ is a clique in a graph G any two distinct vertices of G are connected.

Reminder

Reminder: Cliques

$K \subset V(G)$ is a clique in a graph G any two distinct vertices of G are connected.

$$\omega(G) = \max\{|K| : \ K \text{ is a clique}\}$$

Reminder

Reminder: Cliques

$K \subset V(G)$ is a clique in a graph G any two distinct vertices of G are connected.

$$\omega(G) = \max\{|K| : K \text{ is a clique}\}$$

Reminder: independent vertex sets

$F \subset V(G)$ is an independent vertex set if there is no edge $x = uv \in E(G)$ such that $u, v \in F$.

Reminder

Reminder: Cliques

$K \subset V(G)$ is a clique in a graph G any two distinct vertices of G are connected.

$$\omega(G) = \max\{|K| : \ K \text{ is a clique}\}$$

Reminder: independent vertex sets

$F \subset V(G)$ is an independent vertex set if there is no edge $x = uv \in E(G)$ such that $u, v \in F$.

$$\alpha(G) = \max\{|F| : \ F \text{ is independent}\}$$

Reminder

Reminder: Cliques

$K \subset V(G)$ is a clique in a graph G any two distinct vertices of G are connected.

$$\omega(G) = \max\{|K| : \ K \text{ is a clique}\}$$

Reminder: independent vertex sets

$F \subset V(G)$ is an independent vertex set if there is no edge $x = uv \in E(G)$ such that $u, v \in F$.

$$\alpha(G) = \max\{|F| : \ F \text{ is independent}\}$$

In this lecture we always assume that our graph is SIMPLE.

Algorithm finding a large independent set

Algorithm finding a large independent set

Greedy algorithm for finding a large independent set

Input: G simple graph

Algorithm finding a large independent set

Greedy algorithm for finding a large independent set

Input: G simple graph

Output: F, an independent vertex set

Algorithm finding a large independent set

Greedy algorithm for finding a large independent set

Input: G simple graph

Output: F, an independent vertex set

Inicialization:

Algorithm finding a large independent set

Greedy algorithm for finding a large independent set

Input: G simple graph

Output: F, an independent vertex set

Inicialization: $F := \emptyset$, $T := V(G)$.

Algorithm finding a large independent set

Greedy algorithm for finding a large independent set

Input: G simple graph

Output: F, an independent vertex set

Inicialization: $F := \emptyset$, $T := V(G)$.

// F is an independent set, that is only extended during the run.

Algorithm finding a large independent set

Greedy algorithm for finding a large independent set

Input: G simple graph

Output: F, an independent vertex set

Inicialization: $F := \emptyset$, $T := V(G)$.

// F is an independent set, that is only extended during the run. T is the set of surviving vertices, i.e. vertices that are not chosen, and not thrown away.

Algorithm finding a large independent set

Greedy algorithm for finding a large independent set

Input: G simple graph

Output: F, an independent vertex set

Inicialization: $F := \emptyset$, $T := V(G)$.

// F is an independent set, that is only extended during the run. T is the set of surviving vertices, i.e. vertices that are not chosen, and not thrown away.

While $T \neq \emptyset$ **Greedy extension:**

Algorithm finding a large independent set

Greedy algorithm for finding a large independent set

Input: G simple graph

Output: F, an independent vertex set

Inicialization: $F := \emptyset$, $T := V(G)$.

// F is an independent set, that is only extended during the run. T is the set of surviving vertices, i.e. vertices that are not chosen, and not thrown away.

While $T \neq \emptyset$ **Greedy extension:**

Take any vertex x from T.

Algorithm finding a large independent set

Greedy algorithm for finding a large independent set

Input: G simple graph

Output: F, an independent vertex set

Inicialization: $F := \emptyset$, $T := V(G)$.

// F is an independent set, that is only extended during the run. T is the set of surviving vertices, i.e. vertices that are not chosen, and not thrown away.

While $T \neq \emptyset$ **Greedy extension:**

Take any vertex x from T.

$F \leftarrow F \cup \{x\}$ // We extend F with x.

Algorithm finding a large independent set

Greedy algorithm for finding a large independent set

Input: G simple graph

Output: F, an independent vertex set

Inicialization: $F := \emptyset$, $T := V(G)$.

// F is an independent set, that is only extended during the run. T is the set of surviving vertices, i.e. vertices that are not chosen, and not thrown away.

While $T \neq \emptyset$ **Greedy extension:**

Take any vertex x from T.

$F \leftarrow F \cup \{x\}$ // We extend F with x.

$T \leftarrow T - \{x\} - N_T(x)$

Algorithm finding a large independent set

Greedy algorithm for finding a large independent set

Input: G simple graph

Output: F, an independent vertex set

Inicialization: $F := \emptyset$, $T := V(G)$.

// F is an independent set, that is only extended during the run. T is the set of surviving vertices, i.e. vertices that are not chosen, and not thrown away.

While $T \neq \emptyset$ **Greedy extension:**

Take any vertex x from T.

$F \leftarrow F \cup \{x\}$ // We extend F with x.

$T \leftarrow T - \{x\} - N_T(x)$ // After choosing x only the non-neighboring vertices of T will survive the extension.

The analysis of the algorithm

The analysis of the algorithm

Lemma

The size of the output of the greedy algorithm for finding large independent set is at least

$$\frac{|V(G)|}{D(G)+1}.$$

The analysis of the algorithm

Lemma

The size of the output of the greedy algorithm for finding large independent set is at least

$$\frac{|V(G)|}{D(G)+1}.$$

After each extension of F by a vertex, the set T is decreased by at most $D(G)+1$ vertices.

The analysis of the algorithm

Lemma

The size of the output of the greedy algorithm for finding large independent set is at least

$$\frac{|V(G)|}{D(G)+1}.$$

After each extension of F by a vertex, the set T is decreased by at most $D(G)+1$ vertices.

The size of the output is exactly the number of extension steps. During the extensions the original size of T, $|V(G)|$ drops to 0.

The analysis of the algorithm

Lemma

The size of the output of the greedy algorithm for finding large independent set is at least

$$\frac{|V(G)|}{D(G)+1}.$$

After each extension of F by a vertex, the set T is decreased by at most $D(G)+1$ vertices.

The size of the output is exactly the number of extension steps. During the extensions the original size of T, $|V(G)|$ drops to 0.

Hence we executed at least $|V(G)|/(D(G)+1)$ extension steps.

The essence of the algorithm

The essence of the algorithm

We say that at every extension step we bite/cut off a part of T.

The essence of the algorithm

We say that at every extension step we bite/cut off a part of T. We could give an upper bound on the size of each piece that we can bite/cut.

The essence of the algorithm

We say that at every extension step we bite/cut off a part of T. We could give an upper bound on the size of each piece that we can bite/cut. During the algorithm we consumed the whole vertex set.

The essence of the algorithm

We say that at every extension step we bite/cut off a part of T. We could give an upper bound on the size of each piece that we can bite/cut. During the algorithm we consumed the whole vertex set. Hence the number of cuts must be "large".

The essence of the algorithm

We say that at every extension step we bite/cut off a part of T. We could give an upper bound on the size of each piece that we can bite/cut. During the algorithm we consumed the whole vertex set. Hence the number of cuts must be "large".

The original algorithm choose an element x from T without any consideration.

The essence of the algorithm

We say that at every extension step we bite/cut off a part of T. We could give an upper bound on the size of each piece that we can bite/cut. During the algorithm we consumed the whole vertex set. Hence the number of cuts must be "large".

The original algorithm choose an element x from T without any consideration. An enhancement of our method can be if we choose a reasonable x.

The essence of the algorithm

We say that at every extension step we bite/cut off a part of T. We could give an upper bound on the size of each piece that we can bite/cut. During the algorithm we consumed the whole vertex set. Hence the number of cuts must be "large".

The original algorithm choose an element x from T without any consideration. An enhancement of our method can be if we choose a reasonable x. Our goal is to have as many bite/cuts as possible.

The essence of the algorithm

We say that at every extension step we bite/cut off a part of T. We could give an upper bound on the size of each piece that we can bite/cut. During the algorithm we consumed the whole vertex set. Hence the number of cuts must be "large".

The original algorithm choose an element x from T without any consideration. An enhancement of our method can be if we choose a reasonable x. Our goal is to have as many bite/cuts as possible. The natural choice is the vertex x that results in the smallest bite/cut.

Improved greedy algorithm

Improved greedy algorithm

Improved greedy algorithm

Improved greedy algorithm

Improved greedy algorithm

Inicialization:

Improved greedy algorithm

Improved greedy algorithm

Inicialization: $F := \emptyset$, $T := V(G)$.

Improved greedy algorithm

Improved greedy algorithm

Inicialization: $F := \emptyset$, $T := V(G)$.

// F is an independent set, that is only extended during the run.

Improved greedy algorithm

Improved greedy algorithm

Inicialization: $F := \emptyset$, $T := V(G)$.

// F is an independent set, that is only extended during the run. T is the set of surviving vertices, i.e. vertices that are not chosen, and not thrown away.

Improved greedy algorithm

Improved greedy algorithm

Inicialization: $F := \emptyset$, $T := V(G)$.

// F is an independent set, that is only extended during the run. T is the set of surviving vertices, i.e. vertices that are not chosen, and not thrown away.

While $T \neq \emptyset$ **Greedy extension:**

Improved greedy algorithm

Improved greedy algorithm

Inicialization: $F := \emptyset$, $T := V(G)$.

// F is an independent set, that is only extended during the run. T is the set of surviving vertices, i.e. vertices that are not chosen, and not thrown away.

While $T \neq \emptyset$ **Greedy extension:**

Take a $x \in T = V(G|_T)$ of minimal degree in $G|_T$.

Improved greedy algorithm

Improved greedy algorithm

Inicialization: $F := \emptyset$, $T := V(G)$.

// F is an independent set, that is only extended during the run. T is the set of surviving vertices, i.e. vertices that are not chosen, and not thrown away.

While $T \neq \emptyset$ **Greedy extension:**

Take a $x \in T = V(G|_T)$ of minimal degree in $G|_T$.

$F \leftarrow F \cup \{x\}$,

Improved greedy algorithm

Improved greedy algorithm

Inicialization: $F := \emptyset$, $T := V(G)$.

// F is an independent set, that is only extended during the run. T is the set of surviving vertices, i.e. vertices that are not chosen, and not thrown away.

While $T \neq \emptyset$ **Greedy extension:**

Take a $x \in T = V(G|_T)$ of minimal degree in $G|_T$.

$F \leftarrow F \cup \{x\}$, // We extend F with x.

Improved greedy algorithm

Improved greedy algorithm

Inicialization: $F := \emptyset$, $T := V(G)$.

// F is an independent set, that is only extended during the run. T is the set of surviving vertices, i.e. vertices that are not chosen, and not thrown away.

While $T \neq \emptyset$ **Greedy extension:**

Take a $x \in T = V(G|_T)$ of minimal degree in $G|_T$.

$F \leftarrow F \cup \{x\}$, // We extend F with x.

$T \leftarrow T - \{x\} - N_T(x)$

Improved greedy algorithm

Improved greedy algorithm

Inicialization: $F := \emptyset$, $T := V(G)$.

// F is an independent set, that is only extended during the run. T is the set of surviving vertices, i.e. vertices that are not chosen, and not thrown away.

While $T \neq \emptyset$ **Greedy extension:**

Take a $x \in T = V(G|_T)$ of minimal degree in $G|_T$.

$F \leftarrow F \cup \{x\}$, // We extend F with x.

$T \leftarrow T - \{x\} - N_T(x)$ // After choosing x, the non-neighbors of x in T are thrown away.

The analysis of the improved algorithm

The analysis of the improved algorithm

Theorem (Turán's theorem, algorithmic form)

The size of the output of the improved greedy algorithm for finding large independent set is at least

$$\frac{|V(G)|}{\overline{d}(G)+1},$$

where $\overline{d}(G)$ denotes the average degree of G,
i.e. $\overline{d}(G)=\frac{\sum_{x\in V} d(x)}{|V|}=\frac{2|E|}{|V|}$.

Proof

Proof

Let G be any simple graph and run the improved greedy algorithm with input G.

Proof

Let G be any simple graph and run the improved greedy algorithm with input G.

Let x_i be the vertex chosen at the ith extension step.

Proof

Let G be any simple graph and run the improved greedy algorithm with input G.

Let x_i be the vertex chosen at the ith extension step.

Let H_i be the bite/cut after ith extension step.

Proof

Let G be any simple graph and run the improved greedy algorithm with input G.

Let x_i be the vertex chosen at the ith extension step.

Let H_i be the bite/cut after ith extension step. I.e.
$H_i = \{x_i\} \cup (N(x_i) \cap T)$.

Proof

Let G be any simple graph and run the improved greedy algorithm with input G.

Let x_i be the vertex chosen at the ith extension step.

Let H_i be the bite/cut after ith extension step. I.e. $H_i = \{x_i\} \cup (N(x_i) \cap T)$.

It is straight forward that $V(G) = H_1 \dot\cup H_2 \dot\cup \ldots \dot\cup H_\ell$, where ℓ is the number of extension steps, i.e. the size of the output.

Proof

Let G be any simple graph and run the improved greedy algorithm with input G.

Let x_i be the vertex chosen at the ith extension step.

Let H_i be the bite/cut after ith extension step. I.e. $H_i = \{x_i\} \cup (N(x_i) \cap T)$.

It is straight forward that $V(G) = H_1 \dot\cup H_2 \dot\cup \ldots \dot\cup H_\ell$, where ℓ is the number of extension steps, i.e. the size of the output.

Let $\mathcal{E} = \mathcal{E}(H_1, H_2, \ldots, H_\ell)$ be that simple graph where two vertices is connected if and only if (iff) they belong to the same bite/cut.

Proof (1st Observation)

Proof (1st Observation)

Observation

For each vertex x we have $d_G(x) \geq d_{\mathcal{E}}(x)$, specially

$$|E(\mathcal{E}(H_1, \ldots, H_\ell))| \leq |E(G)|.$$

Proof (1st Observation)

Observation

For each vertex x we have $d_G(x) \geq d_{\mathcal{E}}(x)$, specially

$$|E(\mathcal{E}(H_1, \ldots, H_\ell))| \leq |E(G)|.$$

Let x be a vertex from the ith bite/cut: $x \in H_i$.

Proof (1st Observation)

Observation

For each vertex x we have $d_G(x) \geq d_{\mathcal{E}}(x)$, specially

$$|E(\mathcal{E}(H_1, \ldots, H_\ell))| \leq |E(G)|.$$

Let x be a vertex from the ith bite/cut: $x \in H_i$. $d(x)$ counts edges incident to x.

Proof (1st Observation)

Observation

For each vertex x we have $d_G(x) \geq d_{\mathcal{E}}(x)$, specially

$$|E(\mathcal{E}(H_1, \ldots, H_\ell))| \leq |E(G)|.$$

Let x be a vertex from the ith bite/cut: $x \in H_i$. $d(x)$ counts edges incident to x. We divide these edge into two classes:

Proof (1st Observation)

Observation

For each vertex x we have $d_G(x) \geq d_{\mathcal{E}}(x)$, specially

$$|E(\mathcal{E}(H_1, \ldots, H_\ell))| \leq |E(G)|.$$

Let x be a vertex from the ith bite/cut: $x \in H_i$. $d(x)$ counts edges incident to x. We divide these edge into two classes:

(1) edges leading to $H_1 \cup H_2 \cup \ldots \cup H_{i-1}$ form x

Proof (1st Observation)

Observation

For each vertex x we have $d_G(x) \geq d_{\mathcal{E}}(x)$, specially

$$|E(\mathcal{E}(H_1, \ldots, H_\ell))| \leq |E(G)|.$$

Let x be a vertex from the ith bite/cut: $x \in H_i$. $d(x)$ counts edges incident to x. We divide these edge into two classes:

(1) edges leading to $H_1 \cup H_2 \cup \ldots \cup H_{i-1}$ form $x \to d^{\text{back}}(x)$,

Proof (1st Observation)

Observation

For each vertex x we have $d_G(x) \geq d_{\mathcal{E}}(x)$, specially

$$|E(\mathcal{E}(H_1, \ldots, H_\ell))| \leq |E(G)|.$$

Let x be a vertex from the ith bite/cut: $x \in H_i$. $d(x)$ counts edges incident to x. We divide these edge into two classes:

(1) edges leading to $H_1 \cup H_2 \cup \ldots \cup H_{i-1}$ form $x \to d^{\text{back}}(x)$,

(2) edges leading to $H_i \cup \ldots \cup H_\ell$ from x

Proof (1st Observation)

Observation

For each vertex x we have $d_G(x) \geq d_{\mathcal{E}}(x)$, specially

$$|E(\mathcal{E}(H_1, \ldots, H_\ell))| \leq |E(G)|.$$

Let x be a vertex from the ith bite/cut: $x \in H_i$. $d(x)$ counts edges incident to x. We divide these edge into two classes:

(1) edges leading to $H_1 \cup H_2 \cup \ldots \cup H_{i-1}$ form $x \to d^{\text{back}}(x)$,

(2) edges leading to $H_i \cup \ldots \cup H_\ell$ from $x \to d^{\text{forward}}(x)$.

Proof (1st Observation)

Observation

For each vertex x we have $d_G(x) \geq d_{\mathcal{E}}(x)$, specially

$$|E(\mathcal{E}(H_1, \dots, H_\ell))| \leq |E(G)|.$$

Let x be a vertex from the ith bite/cut: $x \in H_i$. $d(x)$ counts edges incident to x. We divide these edge into two classes:

(1) edges leading to $H_1 \cup H_2 \cup \dots \cup H_{i-1}$ form $x \to d^{\text{back}}(x)$,

(2) edges leading to $H_i \cup \dots \cup H_\ell$ from $x \to d^{\text{forward}}(x)$.

Obviously $d(x) = d^{\text{back}}(x) + d^{\text{forward}}(x)$.

Proof (1st Observation)

Observation

For each vertex x we have $d_G(x) \geq d_{\mathcal{E}}(x)$, specially

$$|E(\mathcal{E}(H_1, \ldots, H_\ell))| \leq |E(G)|.$$

Let x be a vertex from the ith bite/cut: $x \in H_i$. $d(x)$ counts edges incident to x. We divide these edge into two classes:

(1) edges leading to $H_1 \cup H_2 \cup \ldots \cup H_{i-1}$ form $x \rightarrow d^{\text{back}}(x)$,

(2) edges leading to $H_i \cup \ldots \cup H_\ell$ from $x \rightarrow d^{\text{forward}}(x)$.

Obviously $d(x) = d^{\text{back}}(x) + d^{\text{forward}}(x)$.

We have $d_{\mathcal{E}}^{\text{back}}(x) = 0 \leq d_G^{\text{back}}(x)$, furthermore

$$d_{\mathcal{E}}^{\text{forward}}(x) = |H_i| - 1 = d_G^{\text{forward}}(x_i) \leq d_G^{\text{forward}}(x).$$

Proof (1st Observation)

Observation

For each vertex x we have $d_G(x) \geq d_{\mathcal{E}}(x)$, specially

$$|E(\mathcal{E}(H_1, \ldots, H_\ell))| \leq |E(G)|.$$

Let x be a vertex from the ith bite/cut: $x \in H_i$. $d(x)$ counts edges incident to x. We divide these edge into two classes:

(1) edges leading to $H_1 \cup H_2 \cup \ldots \cup H_{i-1}$ form $x \to d^{\text{back}}(x)$,

(2) edges leading to $H_i \cup \ldots \cup H_\ell$ from $x \to d^{\text{forward}}(x)$.

Obviously $d(x) = d^{\text{back}}(x) + d^{\text{forward}}(x)$.

We have $d_{\mathcal{E}}^{\text{back}}(x) = 0 \leq d_G^{\text{back}}(x)$, furthermore

$$d_{\mathcal{E}}^{\text{forward}}(x) = |H_i| - 1 = d_G^{\text{forward}}(x_i) \leq d_G^{\text{forward}}(x).$$

Hence the claim $d_{\mathcal{E}}(x) \leq d_G(x)$ is obvious.

Proof (2nd Observation)

Proof (2nd Observation)

Let $h_i = |H_i|$,

Proof (2nd Observation)

Let $h_i = |H_i|$, hence $\sum_{i=1}^{\ell} h_i = |V| =: n$.

Proof (2nd Observation)

Let $h_i = |H_i|$, hence $\sum_{i=1}^{\ell} h_i = |V| =: n$.

We have

$$|E(G)| \geq |E(\mathcal{E})| = \sum_{i=1}^{\ell} \binom{h_i}{2} \geq \ell \binom{|V|/\ell}{2},$$

where the last inequality is the Jensen inequality:

Proof (2nd Observation)

Let $h_i = |H_i|$, hence $\sum_{i=1}^{\ell} h_i = |V| =: n$.

We have

$$|E(G)| \geq |E(\mathcal{E})| = \sum_{i=1}^{\ell} \binom{h_i}{2} \geq \ell \binom{|V|/\ell}{2},$$

where the last inequality is the Jensen inequality: for the convex function $\binom{x}{2} = x(x-1)/2$.

Proof (2nd Observation)

Let $h_i = |H_i|$, hence $\sum_{i=1}^{\ell} h_i = |V| =: n$.

We have

$$|E(G)| \geq |E(\mathcal{E})| = \sum_{i=1}^{\ell} \binom{h_i}{2} \geq \ell \binom{|V|/\ell}{2},$$

where the last inequality is the Jensen inequality: for the convex function $\binom{x}{2} = x(x-1)/2$.

Observation

$$\mu_{n,\ell} := \min\{|E(\mathcal{E}(H_1, \ldots, H_\ell))| : \sum_{i=1}^{\ell} |H_i| = n\} \geq \ell \binom{n/\ell}{2}.$$

Proof (2nd Observation)

Let $h_i = |H_i|$, hence $\sum_{i=1}^{\ell} h_i = |V| =: n$.

We have

$$|E(G)| \geq |E(\mathcal{E})| = \sum_{i=1}^{\ell} \binom{h_i}{2} \geq \ell \binom{|V|/\ell}{2},$$

where the last inequality is the Jensen inequality: for the convex function $\binom{x}{2} = x(x-1)/2$.

Observation

$$\mu_{n,\ell} := \min\{|E(\mathcal{E}(H_1, \ldots, H_\ell))| : \sum_{i=1}^{\ell} |H_i| = n\} \geq \ell \binom{n/\ell}{2}.$$

The proof of the algorithmic form of Turán's theorem: The claim is a simple consequence of the two Observations and simple arithmetics.

A new Observation

A new Observation

We proved the following claim

A new Observation

We proved the following claim

A new Observation

We proved the following claim

Observation

If

$$|E(G)| < \mu_{n,\ell},$$

then the improved greedy algorithm executes at least $\ell + 1$ extension steps, i.e. the algorithm finds an independent set of size at least $\ell + 1$.

A new Observation

We proved the following claim

Observation

If

$$|E(G)| < \mu_{n,\ell},$$

then the improved greedy algorithm executes at least $\ell + 1$ extension steps, i.e. the algorithm finds an independent set of size at least $\ell + 1$.

During the analysis of the algorithm we estimated $\mu_{n,\ell}$ using the Jensen inequality.

A new Observation

We proved the following claim

Observation

If

$$|E(G)| < \mu_{n,\ell},$$

then the improved greedy algorithm executes at least $\ell + 1$ extension steps, i.e. the algorithm finds an independent set of size at least $\ell + 1$.

During the analysis of the algorithm we estimated $\mu_{n,\ell}$ using the Jensen inequality.

The Jensen inequality is sharp if we work with real numbers.

A new Observation

We proved the following claim

Observation

If

$$|E(G)| < \mu_{n,\ell},$$

then the improved greedy algorithm executes at least $\ell + 1$ extension steps, i.e. the algorithm finds an independent set of size at least $\ell + 1$.

During the analysis of the algorithm we estimated $\mu_{n,\ell}$ using the Jensen inequality.

The Jensen inequality is sharp if we work with real numbers. In our setting this is not true.

Balanced partitions

Definition

We consider a partition of a set of n elements into k parts. We say that the partition is balanced or its parts are almost "equal" iff one of (or all of) the following equivalent properties holds

(i) For each part O $|O| \in \left\{ \left\lfloor \frac{n}{k} \right\rfloor, \left\lceil \frac{n}{k} \right\rceil \right\}$.

(ii) For any two parts O and O' $||O| - |O'|| \leq 1$.

(iii) $n - k\left\lfloor \frac{n}{k} \right\rfloor$ parts have size $\left\lceil \frac{n}{k} \right\rceil$ and $k - \left(n - k\left\lfloor \frac{n}{k} \right\rfloor\right)$ parts have size $\left\lfloor \frac{n}{k} \right\rfloor$.

Balanced equivalence graphs, Turán graphs

Balanced equivalence graphs, Turán graphs

Definition: $\mathcal{E}_{n,k}$, the balanced equivalence graphs on n vertices with k components

Its vertex set is an n elements set, divided into k parts
$V = O_1 \dot\cup \ldots \dot\cup O_k$, of "almost equal" size.
Its edge set contains exactly those edges that are connecting two vertices from the same part.

Balanced equivalence graphs, Turán graphs

Definition: $\mathcal{E}_{n,k}$, the balanced equivalence graphs on n vertices with k components

Its vertex set is an n elements set, divided into k parts $V = O_1 \dot\cup \ldots \dot\cup O_k$, of "almost equal" size.
Its edge set contains exactly those edges that are connecting two vertices from the same part.

Definition: $T_{n,k}$ the Turán graph on n vertices with k parts

Its vertex set is an n elements set, divided into k parts $V = O_1 \dot\cup \ldots \dot\cup O_k$, of "almost equal" size.
Its edge set contains exactly those edges that are connecting two vertices from two different parts.

Balanced equivalence graphs, Turán graphs

Definition: $\mathcal{E}_{n,k}$, the balanced equivalence graphs on n vertices with k components

Its vertex set is an n elements set, divided into k parts
$V = O_1 \dot\cup \ldots \dot\cup O_k$, of "almost equal" size.
Its edge set contains exactly those edges that are connecting two vertices from the same part.

Definition: $T_{n,k}$ the Turán graph on n vertices with k parts

Its vertex set is an n elements set, divided into k parts
$V = O_1 \dot\cup \ldots \dot\cup O_k$, of "almost equal" size.
Its edge set contains exactly those edges that are connecting two vertices from two different parts.

$$T_{n,k} = \overline{\mathcal{E}_{n,k}}$$

Lemma

Lemma

Lemma

$$\mu_{n,\ell} = \min\{|E(\mathcal{E}(H_1, \dots, H_\ell))| : \sum_{i=1}^{\ell} |H_i| = n\} = |E(\mathcal{E}_{n,\ell})|.$$

Lemma

Lemma

$$\mu_{n,\ell} = \min\{|E(\mathcal{E}(H_1, \ldots, H_\ell))| : \sum_{i=1}^{\ell} |H_i| = n\} = |E(\mathcal{E}_{n,\ell})|.$$

Example

$n = 700$, $\ell = 200$

Lemma

Lemma

$$\mu_{n,\ell} = \min\{|E(\mathcal{E}(H_1, \ldots, H_\ell))| : \sum_{i=1}^{\ell} |H_i| = n\} = |E(\mathcal{E}_{n,\ell})|.$$

Example

$n = 700$, $\ell = 200$

Estimating $\mu_{n,\ell}$ using Jensen's inequality:

$$\mu_{n,\ell} \geq \ell \binom{n/\ell}{2} = 875.$$

Lemma

Lemma

$$\mu_{n,\ell} = \min\{|E(\mathcal{E}(H_1,\ldots,H_\ell))| : \sum_{i=1}^{\ell} |H_i| = n\} = |E(\mathcal{E}_{n,\ell})|.$$

Example

$n = 700$, $\ell = 200$

Estimating $\mu_{n,\ell}$ using Jensen's inequality:

$$\mu_{n,\ell} \geq \ell\binom{n/\ell}{2} = 875.$$

Applying Lemma:

$$\mu_{n,\ell} = |E(\mathcal{E}_{n,\ell})| = 900.$$

Proof of the Lemma

Proof of the Lemma

Assume that $\mathcal{E}$ is an equivalence graph, and the partition behind it is not balanced.

Proof of the Lemma

Assume that $\mathcal{E}$ is an equivalence graph, and the partition behind it is not balanced.

We can find two parts with sizes that differ from each other by more than 1.

Proof of the Lemma

Assume that $\mathcal{E}$ is an equivalence graph, and the partition behind it is not balanced.

We can find two parts with sizes that differ from each other by more than 1.

Modify $\mathcal{E}$ by modifying its underlying partition: Take the two parts from above and place one vertex from the bigger part into the smaller one.

Proof of the Lemma

Assume that $\mathcal{E}$ is an equivalence graph, and the partition behind it is not balanced.

We can find two parts with sizes that differ from each other by more than 1.

Modify $\mathcal{E}$ by modifying its underlying partition: Take the two parts from above and place one vertex from the bigger part into the smaller one. The other parts remain untouched.

Proof of the Lemma

Assume that $\mathcal{E}$ is an equivalence graph, and the partition behind it is not balanced.

We can find two parts with sizes that differ from each other by more than 1.

Modify $\mathcal{E}$ by modifying its underlying partition: Take the two parts from above and place one vertex from the bigger part into the smaller one. The other parts remain untouched.

Easy computation give us that this modification

Reformulation of the Theorem

Reformulation of the Theorem

Theorem

If G is a simple graph on n vertices and $|E(G)| < |E(\mathcal{E}_{n,\ell})|$, then the improved greedy algorithm will have at least $\ell + 1$ extension steps. Specially we have $\alpha(G) \geq \ell + 1$.

Reformulation of the Theorem

Theorem

If G is a simple graph on n vertices and $|E(G)| < |E(\mathcal{E}_{n,\ell})|$, then the improved greedy algorithm will have at least $\ell + 1$ extension steps. Specially we have $\alpha(G) \geq \ell + 1$.

Theorem

Let G be a simple graph on n vertices. If $\alpha(G) < k$, then $|E(G)| \geq |E(\overline{T}_{n,k-1})|$.

The reformulated theorem, complementary form

The reformulated theorem, complementary form

Theorem of Paul Turán

If G is a simple graph on n vertices and it doesn't contain a clique of size k then

$$|E(G)| \leq |E(T_{n,k-1})|.$$

The reformulated theorem, complementary form

Theorem of Paul Turán

If G is a simple graph on n vertices and it doesn't contain a clique of size k then

$$|E(G)| \leq |E(T_{n,k-1})|.$$

Theorem of Paul Turán

Let G be a simple graph on n vertices. If

$$|E(G)| > |E(T_{n,k-1})|,$$

then it must contain a clique of size k.

Turán's theorem, algorithmic version in complementary form

Turán's theorem, algorithmic version in complementary form

If we run the improved greedy algorithm for $\overline{G}$ then we will find a "large" vertex set, that forms a clique in G. The algorithm can be formulated in terms of the original graph.

Turán's theorem, algorithmic version in complementary form

If we run the improved greedy algorithm for $\overline{G}$ then we will find a "large" vertex set, that forms a clique in G. The algorithm can be formulated in terms of the original graph.

Exercise

Formulate an improved greedy algorithm for finding a "large" clique in the input graph.

Turán's theorem, algorithmic version in complementary form

If we run the improved greedy algorithm for $\overline{G}$ then we will find a "large" vertex set, that forms a clique in G. The algorithm can be formulated in terms of the original graph.

Exercise

Formulate an improved greedy algorithm for finding a "large" clique in the input graph. You can build on the ideas of our original algorithm, or you can take the above algorithm and eliminate the reference to the complementer graph from it.

Turán's theorem, algorithmic version in complementary form

If we run the improved greedy algorithm for $\overline{G}$ then we will find a "large" vertex set, that forms a clique in G. The algorithm can be formulated in terms of the original graph.

Exercise

Formulate an improved greedy algorithm for finding a "large" clique in the input graph. You can build on the ideas of our original algorithm, or you can take the above algorithm and eliminate the reference to the complementer graph from it.

The improved greedy algorithm for finding large clique in the case of $|E(G)| > |E(T_{n,k-1})|$ will find a clique of size at least k.

Theorem of Turán is sharp

Theorem of Turán is sharp

$T_{n,k}$ doesn't contain a clique of size $k+1$.

Theorem of Turán is sharp

$T_{n,k}$ doesn't contain a clique of size $k+1$.

1st argument (pigeon hole principle): Take any set L of size larger than the number of parts of our Turán graph. By pigeon hole principle we can find two vertices in L, that belong to the same part. By definition they are not connected, hence L is not a clique.

Theorem of Turán is sharp

$T_{n,k}$ doesn't contain a clique of size $k+1$.

1st argument (pigeon hole principle): Take any set L of size larger than the number of parts of our Turán graph. By pigeon hole principle we can find two vertices in L, that belong to the same part. By definition they are not connected, hence L is not a clique.

2nd argument (coloring): $T_{n,k}$ has a legal k-coloring. Hence $T_{n,k}$ cannot contain a subgraph R with $\chi(R) \geq k+1$. Hence $T_{n,k}$ doesn't contain a clique of size $k+1$, furthermore it doesn't contain any subgraph that is nor k-colorable.

Break

Generalizations

Generalizations

A special case of Turán's theorem consider a simple graph on n vertices doesn't have a clique of size 4 (or K_4 subgraph). The claim of the theorem is that G cannot have more than $|E(T_{n,3})|$ edges.

Generalizations

A special case of Turán's theorem consider a simple graph on n vertices doesn't have a clique of size 4 (or K_4 subgraph). The claim of the theorem is that G cannot have more than $|E(T_{n,3})|$ edges.

The condition can be formulated like "We consider simple graphs, bur K_4 is a forbidden subgraph".

Generalizations

A special case of Turán's theorem consider a simple graph on n vertices doesn't have a clique of size 4 (or K_4 subgraph). The claim of the theorem is that G cannot have more than $|E(T_{n,3})|$ edges.

The condition can be formulated like "We consider simple graphs, bur K_4 is a forbidden subgraph". K_4 van be considered as the graph of tetrahedron.

Generalizations

A special case of Turán's theorem consider a simple graph on n vertices doesn't have a clique of size 4 (or K_4 subgraph). The claim of the theorem is that G cannot have more than $|E(T_{n,3})|$ edges.

The condition can be formulated like "We consider simple graphs, bur K_4 is a forbidden subgraph". K_4 van be considered as the graph of tetrahedron. // Every polytope (bounded subset of $\mathbb{R}^n$, that is an intersection of finitely many closed halfspaces) has a graph. Its vertex set consists of the vertices of the polytope, it edge set is formed by the edges of the polytope.

Generalizations

A special case of Turán's theorem consider a simple graph on n vertices doesn't have a clique of size 4 (or K_4 subgraph). The claim of the theorem is that G cannot have more than $|E(T_{n,3})|$ edges.

The condition can be formulated like "We consider simple graphs, bur K_4 is a forbidden subgraph". K_4 van be considered as the graph of tetrahedron. // Every polytope (bounded subset of $\mathbb{R}^n$, that is an intersection of finitely many closed halfspaces) has a graph. Its vertex set consists of the vertices of the polytope, it edge set is formed by the edges of the polytope.

At the end of his research paper Turán asked the following question:

Generalizations

A special case of Turán's theorem consider a simple graph on n vertices doesn't have a clique of size 4 (or K_4 subgraph). The claim of the theorem is that G cannot have more than $|E(T_{n,3})|$ edges.

The condition can be formulated like "We consider simple graphs, bur K_4 is a forbidden subgraph". K_4 van be considered as the graph of tetrahedron. // Every polytope (bounded subset of $\mathbb{R}^n$, that is an intersection of finitely many closed halfspaces) has a graph. Its vertex set consists of the vertices of the polytope, it edge set is formed by the edges of the polytope.

At the end of his research paper Turán asked the following question: What happens if we forbid as subgraph the graph of another polytope. Proposing this question was crucial in the history of graph theory. It started a new line of research, that led to a new branch of combinatorics.

Generalizations

A special case of Turán's theorem consider a simple graph on n vertices doesn't have a clique of size 4 (or K_4 subgraph). The claim of the theorem is that G cannot have more than $|E(T_{n,3})|$ edges.

The condition can be formulated like "We consider simple graphs, bur K_4 is a forbidden subgraph". K_4 van be considered as the graph of tetrahedron. // Every polytope (bounded subset of $\mathbb{R}^n$, that is an intersection of finitely many closed halfspaces) has a graph. Its vertex set consists of the vertices of the polytope, it edge set is formed by the edges of the polytope.

At the end of his research paper Turán asked the following question: What happens if we forbid as subgraph the graph of another polytope. Proposing this question was crucial in the history of graph theory. It started a new line of research, that led to a new branch of combinatorics.

Extremal graph theory has been born.

The function ext

The function ext

Definition

$$ext(n; T) = \max\{|E(G)| : \ G \text{ is a simple graph on } n \text{ vertices}, \ T \not\subseteq G\}.$$

The function ext

Definition

$$ext(n;T) = \max\{|E(G)| : \ G \text{ is a simple graph on } n \text{ vertices}, \ T \not\subseteq G\}.$$

We refer to T as a forbidden subgraph. n is the size of the vertex set of the graph, we consider.

The function ext

Definition

$$ext(n;T) = \max\{|E(G)| : \ G \text{ is a simple graph on } n \text{ vertices}, \ T \not\subseteq G\}.$$

We refer to T as a forbidden subgraph. n is the size of the vertex set of the graph, we consider.

It will be very useful to introduce the following notation: $\mathcal{G}_n$ denotes the class of simple graphs on n vertices. So $G \in \mathcal{G}_n$ expresses that G is a simple graph on n vertices.

Turán-type questions

Turán-type questions

We can reformulate Turán's theorem again:

$$ext(n; K_k) = |E(T_{n,k-1})|.$$

Turán-type questions

We can reformulate Turán's theorem again:

$$ext(n; K_k) = |E(T_{n,k-1})|.$$

The problems related to $ext(n; T)$ are called Turán-type problem.

Turán-type questions

We can reformulate Turán's theorem again:

$$ext(n; K_k) = |E(T_{n,k-1})|.$$

The problems related to $ext(n; T)$ are called Turán-type problem.

Turán-type problems form only a small portion of extremal graph theory.

Turán-type questions

We can reformulate Turán's theorem again:

$$ext(n; K_k) = |E(T_{n,k-1})|.$$

The problems related to $ext(n; T)$ are called Turán-type problem.

Turán-type problems form only a small portion of extremal graph theory.

Extremal graph theory considers a special class of graphs and a graph parameter. We are interested what are the extremal values of this parameter if our graphs are chosen from the special class. Very often only the maximal, or only the minimal value has theoretical significance.

Open problems

Open problems

• We mentioned that the original paper of Paul Turán introduced some problems. It is interesting to note that the problem concerning the forbidden cube graph is still open, We do not know how many edges a $G \in \mathcal{G}_n$ can have if it doesn't contain as a subgraph a cube graph.

Open problems

• We mentioned that the original paper of Paul Turán introduced some problems. It is interesting to note that the problem concerning the forbidden cube graph is still open, We do not know how many edges a $G \in \mathcal{G}_n$ can have if it doesn't contain as a subgraph a cube graph.

• We also mention that the simplest Turán-type problem (when a triangle is forbidden) was know at the very beginning of the 20th century. Mantel proposed the claim as an exercises in a mathematical journal. It was only an interesting question.

Open problems

• We mentioned that the original paper of Paul Turán introduced some problems. It is interesting to note that the problem concerning the forbidden cube graph is still open, We do not know how many edges a $G \in \mathcal{G}_n$ can have if it doesn't contain as a subgraph a cube graph.

• We also mention that the simplest Turán-type problem (when a triangle is forbidden) was know at the very beginning of the 20th century. Mantel proposed the claim as an exercises in a mathematical journal. It was only an interesting question.

• Turán's publication was a really influential paper.

Open problems

• We mentioned that the original paper of Paul Turán introduced some problems. It is interesting to note that the problem concerning the forbidden cube graph is still open, We do not know how many edges a $G \in \mathcal{G}_n$ can have if it doesn't contain as a subgraph a cube graph.

• We also mention that the simplest Turán-type problem (when a triangle is forbidden) was know at the very beginning of the 20th century. Mantel proposed the claim as an exercises in a mathematical journal. It was only an interesting question.

• Turán's publication was a really influential paper.

• Turán's close colleague and friend, Paul Erdős immediately realized the importance of these type of questions. His earlier work already contained some sporadic extremal results. His work was one of the main driving force that shaped these questions and relating results into a theory.

Open problems

• We mentioned that the original paper of Paul Turán introduced some problems. It is interesting to note that the problem concerning the forbidden cube graph is still open, We do not know how many edges a $G \in \mathcal{G}_n$ can have if it doesn't contain as a subgraph a cube graph.

• We also mention that the simplest Turán-type problem (when a triangle is forbidden) was know at the very beginning of the 20th century. Mantel proposed the claim as an exercises in a mathematical journal. It was only an interesting question.

• Turán's publication was a really influential paper.

• Turán's close colleague and friend, Paul Erdős immediately realized the importance of these type of questions. His earlier work already contained some sporadic extremal results. His work was one of the main driving force that shaped these questions and relating results into a theory.

• One can assume that the forbidden subgraph T doesn't have isolated vertex. (Why?)

Break

The forbidden graph with one edge

The forbidden graph with one edge

Observation

Let I be the gaph containing two vertices and oone connecting edge. Then $ext(n; I) = 0$.

Forbidden graphs with two edges

Forbidden graphs with two edges

Observation

Let $\wedge$ be a simple graph on three vertices with two edges. Then $ext(n; \wedge) = \lfloor n/2 \rfloor$.

Forbidden graphs with two edges

Observation

Let $\wedge$ be a simple graph on three vertices with two edges. Then $ext(n; \wedge) = \lfloor n/2 \rfloor$.

Obervation

Let M_2 be a 1-regular graph on four vertices. Then $ext(n; M_2) = n - 1$, assuming $n \geq 4$.

Forbidden graphs with two edges

Observation

Let $\wedge$ be a simple graph on three vertices with two edges. Then $ext(n; \wedge) = \lfloor n/2 \rfloor$.

Obervation

Let M_2 be a 1-regular graph on four vertices. Then $ext(n; M_2) = n - 1$, assuming $n \geq 4$.

Proof: Easy.

A Corollary

A Corollary

Corollary

If $|E(T)| \geq 2$, then

$$ext(n; T) = \Omega(n).$$

A Corollary

Corollary

If $|E(T)| \geq 2$, then
$$ext(n; T) = \Omega(n).$$

The upper bounds is easy if T doesn't have a cycle.

Theorem on forbidden trees

Theorem on forbidden trees

Theorem

Let T be a forest (i.e. a graph without cycle; i.e. a graph with tree components). If $|E(T)| > 1$, then for suitable constants $\alpha_T, \beta_T > 0$

$$\alpha_T \cdot n \leq ext(n; T) \leq \beta_T \cdot n.$$

Hence the order of magnitude of $ext(n; T)$ is linear.

Theorem on forbidden trees

Theorem

Let T be a forest (i.e. a graph without cycle; i.e. a graph with tree components). If $|E(T)| > 1$, then for suitable constants $\alpha_T, \beta_T > 0$

$$\alpha_T \cdot n \le ext(n; T) \le \beta_T \cdot n.$$

Hence the order of magnitude of $ext(n; T)$ is linear.

The lower bound follows from the condition that the forbidden subgraph has at least two edges. This part of the claim is easy.

Lemma

Lemma

Notation

Let H be a graph. $\overline{d}(H)$ denotes the average degree of H, $\delta(H)$ denotes the minimal degree of H.

Lemma

Notation

Let H be a graph. $\overline{d}(H)$ denotes the average degree of H, $\delta(H)$ denotes the minimal degree of H.

Lemma

For $G \in \mathcal{G}_n$ we have a subgraph R ($R \subseteq G$) with

$$\delta(R) \geq \frac{\overline{d}(G)}{2}.$$

The proof of the Lemma

The proof of the Lemma

Algorithm

The proof of the Lemma

Algorithm

Input: G, simple graph. Output: R spanning subgraph with $\overline{d}/2$.

The proof of the Lemma

Algorithm

Input: G, simple graph. Output: R spanning subgraph with $\overline{d}/2$.

$A := G$

The proof of the Lemma

Algorithm

Input: G, simple graph. Output: R spanning subgraph with $\overline{d}/2$.

$A := G$

// A is the actual graph, initially it is G.

The proof of the Lemma

Algorithm

Input: G, simple graph. Output: R spanning subgraph with $\overline{d}/2$.

$A := G$

// A is the actual graph, initially it is G.

Until we find $x \in V(A)$: $d_A(x) < \frac{\overline{d}}{2}$ do

$$A \leftarrow A - x.$$

The proof of the Lemma

Algorithm

Input: G, simple graph. Output: R spanning subgraph with $\overline{d}/2$.

$A := G$

// A is the actual graph, initially it is G.

Until we find $x \in V(A)$: $d_A(x) < \frac{\overline{d}}{2}$ do

$$A \leftarrow A - x.$$

The Lemma is equivalent to the fact the the algorithm doesn't „consume" G completely.

The proof of the Lemma (continued)

The proof of the Lemma (continued)

Assume that during the algorithm all vertices are deleted.

The proof of the Lemma (continued)

Assume that during the algorithm all vertices are deleted.

Let π be the order of the vertices, that follows the deletions: $\pi : v_1, \ldots, v_n$, i.e. v_i is the vertex that is deleted in the ith step ($n = |V(G)|$).

The proof of the Lemma (continued)

Assume that during the algorithm all vertices are deleted.

Let π be the order of the vertices, that follows the deletions: $\pi : v_1, \ldots, v_n$, i.e. v_i is the vertex that is deleted in the ith step ($n = |V(G)|$).

$d_\pi^{\text{back}}(v)$ denotes the number of the neighbors of the vertex $v = v_i$ with a larger index than i.

The proof of the Lemma (continued)

Assume that during the algorithm all vertices are deleted.

Let π be the order of the vertices, that follows the deletions: $\pi : v_1, \ldots, v_n$, i.e. v_i is the vertex that is deleted in the ith step $(n = |V(G)|)$.

$d_\pi^{\text{back}}(v)$ denotes the number of the neighbors of the vertex $v = v_i$ with a larger index than i.

We know: For all vertices $v \in V$ $d_\pi^{\text{back}}(v) < \frac{\overline{d}}{2}$.

The proof of the Lemma (continued)

Assume that during the algorithm all vertices are deleted.

Let π be the order of the vertices, that follows the deletions: $\pi : v_1, \ldots, v_n$, i.e. v_i is the vertex that is deleted in the ith step ($n = |V(G)|$).

$d_\pi^{\text{back}}(v)$ denotes the number of the neighbors of the vertex $v = v_i$ with a larger index than i.

We know: For all vertices $v \in V$ $d_\pi^{\text{back}}(v) < \frac{\overline{d}}{2}$.

Observe that: $\sum d_\pi^{\text{back}}(v) = |E|$.

The proof of the Lemma (continued)

Assume that during the algorithm all vertices are deleted.

Let π be the order of the vertices, that follows the deletions: $\pi : v_1, \ldots, v_n$, i.e. v_i is the vertex that is deleted in the ith step ($n = |V(G)|$).

$d_\pi^{\text{back}}(v)$ denotes the number of the neighbors of the vertex $v = v_i$ with a larger index than i.

We know: For all vertices $v \in V$ $d_\pi^{\text{back}}(v) < \frac{\overline{d}}{2}$.

Observe that: $\sum d_\pi^{\text{back}}(v) = |E|$.

$|E|$ is $n\frac{\overline{d}}{2}$.

The proof of the Lemma (continued)

Assume that during the algorithm all vertices are deleted.

Let π be the order of the vertices, that follows the deletions: $\pi : v_1, \ldots, v_n$, i.e. v_i is the vertex that is deleted in the ith step ($n = |V(G)|$).

$d_\pi^{\text{back}}(v)$ denotes the number of the neighbors of the vertex $v = v_i$ with a larger index than i.

We know: For all vertices $v \in V$ $d_\pi^{\text{back}}(v) < \frac{\overline{d}}{2}$.

Observe that: $\sum d_\pi^{\text{back}}(v) = |E|$.

$|E|$ is $n\frac{\overline{d}}{2}$. A contradiction.

The proof of the Theorem

The proof of the Theorem

The Claim

If

$$\frac{\sum_i d_i}{n} \geq \frac{2|V(T)|n}{n} = 2|V(T)|,$$

then G contains a subgraph isomorphic to T.

The proof of the Theorem

The Claim

If
$$\frac{\sum_i d_i}{n} \geq \frac{2|V(T)|n}{n} = 2|V(T)|,$$
then G contains a subgraph isomorphic to T.

Using the Lemma we have a subgraph R, that satisfies $\delta(R) \geq |V(T)|$.

The proof of the Theorem

The Claim

If
$$\frac{\sum_i d_i}{n} \geq \frac{2|V(T)|n}{n} = 2|V(T)|,$$
then G contains a subgraph isomorphic to T.

Using the Lemma we have a subgraph R, that satisfies $\delta(R) \geq |V(T)|$.

We are looking for T in R.

The proof of the Theorem

The Claim

If

$$\frac{\sum_i d_i}{n} \geq \frac{2|V(T)|n}{n} = 2|V(T)|,$$

then G contains a subgraph isomorphic to T.

Using the Lemma we have a subgraph R, that satisfies $\delta(R) \geq |V(T)|$.

We are looking for T in R.

We think of T as a result of branching operations starting from an empty graph.

The proof of the Theorem

The Claim

If

$$\frac{\sum_i d_i}{n} \geq \frac{2|V(T)|n}{n} = 2|V(T)|,$$

then G contains a subgraph isomorphic to T.

Using the Lemma we have a subgraph R, that satisfies $\delta(R) \geq |V(T)|$.

We are looking for T in R.

We think of T as a result of branching operations starting from an empty graph. Let T_i be the graph during this construction with i edges.

The proof of the Theorem

The Claim

If
$$\frac{\sum_i d_i}{n} \geq \frac{2|V(T)|n}{n} = 2|V(T)|,$$
then G contains a subgraph isomorphic to T.

Using the Lemma we have a subgraph R, that satisfies $\delta(R) \geq |V(T)|$.

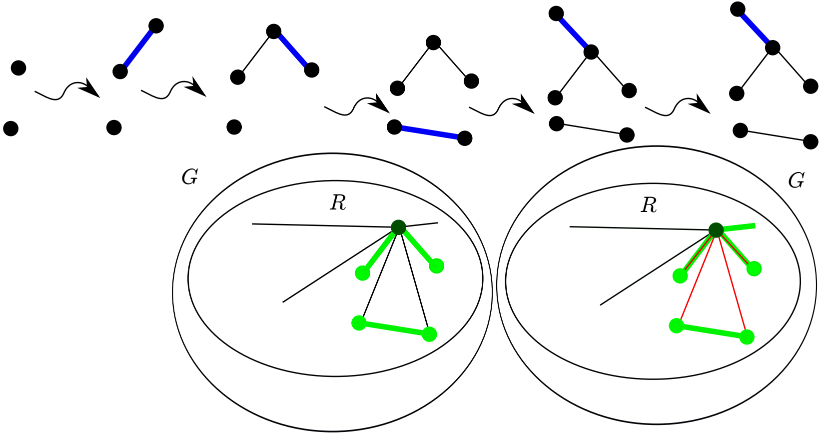
We are looking for T in R.

We think of T as a result of branching operations starting from an empty graph. Let T_i be the graph during this construction with i edges.

We proof by induction that T_i is a subgraph of R.

The proof on a Figure

The proof on a Figure



Break

A fundamental result

A fundamental result

Reminder: If the forbidden graph T is such that $\chi(T) = k$ then $ext(n; T) \geq |E(T_{n,k-1})|$.

A fundamental result

Reminder: If the forbidden graph T is such that $\chi(T) = k$ then $ext(n; T) \geq |E(T_{n,k-1})|$.

(Erdős—Stone, Erdős—Simonovits)

If T is such $\chi(T) = k \geq 2$, then $ext(n; T) = |E(T_{n,k-1})| + o(n^2)$.

A fundamental result

Reminder: If the forbidden graph T is such that $\chi(T) = k$ then $ext(n; T) \geq |E(T_{n,k-1})|$.

(Erdős—Stone, Erdős—Simonovits)

If T is such $\chi(T) = k \geq 2$, then $ext(n; T) = |E(T_{n,k-1})| + o(n^2)$.

(Erdős—Stone, Erdős—Simonovits)

A fundamental result

Reminder: If the forbidden graph T is such that $\chi(T) = k$ then $ext(n; T) \geq |E(T_{n,k-1})|$.

(Erdős—Stone, Erdős—Simonovits)

If T is such $\chi(T) = k \geq 2$, then $ext(n; T) = |E(T_{n,k-1})| + o(n^2)$.

(Erdős—Stone, Erdős—Simonovits)

(i) Let T be a forbidden subgraph with chromatic number $k \geq 3$ (i.e. $k - 1$ — the number of parts of the corresponding — is at least 2). Then $ext(n; T) = |E(T_{n,k-1})| + o(n^2)$ (In this case $o(n^2)$ is the remainder term in the formula).

A fundamental result

Reminder: If the forbidden graph T is such that $\chi(T) = k$ then $ext(n; T) \geq |E(T_{n,k-1})|$.

(Erdős—Stone, Erdős—Simonovits)

If T is such $\chi(T) = k \geq 2$, then $ext(n; T) = |E(T_{n,k-1})| + o(n^2)$.

(Erdős—Stone, Erdős—Simonovits)

(i) Let T be a forbidden subgraph with chromatic number $k \geq 3$ (i.e. $k - 1$ — the number of parts of the corresponding — is at least 2). Then $ext(n; T) = |E(T_{n,k-1})| + o(n^2)$ (In this case $o(n^2)$ is the remainder term in the formula).

(ii) Let T be a forbidden non-empty, bipartite graph, i.e. $\chi(T) = 2$. Then $ext(n; T) = o(n^2)$ (i.e. the formal remainder term is the main term now).

A reformulation of the theorem

A reformulation of the theorem

Let us given $2 \geq k \in \mathbb{N}$.

A reformulation of the theorem

Let us given $2 \geq k \in \mathbb{N}$. Let $\varepsilon > 0$ be an arbitrary small real number.

A reformulation of the theorem

Let us given $2 \geq k \in \mathbb{N}$. Let $\varepsilon > 0$ be an arbitrary small real number. Let S an arbitrary natural number.

A reformulation of the theorem

Let us given $2 \geq k \in \mathbb{N}$. Let $\varepsilon > 0$ be an arbitrary small real number. Let S an arbitrary natural number. Let $G \in \mathcal{G}_n$ be a graph, with "many" edges:

$$|E(T_{n,k-1})| + \varepsilon \cdot n^2 = \frac{1}{2}\left(1 - \frac{1}{k-1}\right) n^2 + \varepsilon \cdot n^2.$$

A reformulation of the theorem

Let us given $2 \geq k \in \mathbb{N}$. Let $\varepsilon > 0$ be an arbitrary small real number. Let S an arbitrary natural number. Let $G \in \mathcal{G}_n$ be a graph, with "many" edges:

$$|E(T_{n,k-1})| + \varepsilon \cdot n^2 = \frac{1}{2}\left(1 - \frac{1}{k-1}\right) n^2 + \varepsilon \cdot n^2.$$

In this case G contains $K_{S,S,\dots,S} = K_{k\times S}$, assuming that n is large enough.

A reformulation of the theorem

Let us given $2 \geq k \in \mathbb{N}$. Let $\varepsilon > 0$ be an arbitrary small real number. Let S an arbitrary natural number. Let $G \in \mathcal{G}_n$ be a graph, with "many" edges:

$$|E(T_{n,k-1})| + \varepsilon \cdot n^2 = \frac{1}{2}\left(1 - \frac{1}{k-1}\right) n^2 + \varepsilon \cdot n^2.$$

In this case G contains $K_{S,S,\dots,S} = K_{k\times S}$, assuming that n is large enough. $K_{S,S,\dots,S} = K_{k\times S}$ is a complete k-partite graph, with parts of size S.

The proof: The first steps

The proof: The first steps

Lemma

For $G \in \mathcal{G}_n$ $|E(G)| \geq \delta\binom{n}{2}$ (the average degrre is at least $\delta(n-1)$). The G has a subgraph R, with all degrees at least $\delta(|V(R)|-1)$.

The proof: The first steps

Lemma

For $G \in \mathcal{G}_n$ $|E(G)| \geq \delta\binom{n}{2}$ (the average degrre is at least $\delta(n-1)$). The G has a subgraph R, with all degrees at least $\delta(|V(R)|-1)$.

Algorithm

The proof: The first steps

Lemma

For $G \in \mathcal{G}_n$ $|E(G)| \geq \delta\binom{n}{2}$ (the average degrre is at least $\delta(n-1)$). The G has a subgraph R, with all degrees at least $\delta(|V(R)|-1)$.

Algorithm

// G-ről feltesszük, hogy átlag foka $\delta(|V(G)|-1)$

The proof: The first steps

Lemma

For $G \in \mathcal{G}_n$ $|E(G)| \geq \delta\binom{n}{2}$ (the average degrre is at least $\delta(n-1)$). The G has a subgraph R, with all degrees at least $\delta(|V(R)|-1)$.

Algorithm

// G-ről feltesszük, hogy átlag foka $\delta(|V(G)|-1)$

$A := G$

The proof: The first steps

Lemma

For $G \in \mathcal{G}_n$ $|E(G)| \geq \delta\binom{n}{2}$ (the average degrre is at least $\delta(n-1)$). The G has a subgraph R, with all degrees at least $\delta(|V(R)|-1)$.

Algorithm

// G-ről feltesszük, hogy átlag foka $\delta(|V(G)|-1)$

$A := G$

// A is the actual graph, initially it is G.

The proof: The first steps

Lemma

For $G \in \mathcal{G}_n$ $|E(G)| \geq \delta\binom{n}{2}$ (the average degrre is at least $\delta(n-1)$). The G has a subgraph R, with all degrees at least $\delta(|V(R)|-1)$.

Algorithm

// G-ről feltesszük, hogy átlag foka $\delta(|V(G)|-1)$

$A := G$

// A is the actual graph, initially it is G.

Until we find $x \in V(A)$ such that $d_A(x) < \delta(|V(A)|-1)$

$$A \leftarrow A - x.$$

The proof: The first steps

Lemma

For $G \in \mathcal{G}_n$ $|E(G)| \geq \delta\binom{n}{2}$ (the average degrre is at least $\delta(n-1)$). The G has a subgraph R, with all degrees at least $\delta(|V(R)|-1)$.

Algorithm

// G-ről feltesszük, hogy átlag foka $\delta(|V(G)|-1)$

$A := G$

// A is the actual graph, initially it is G.

Until we find $x \in V(A)$ such that $d_A(x) < \delta(|V(A)|-1)$

$$A \leftarrow A - x.$$

The Lemma is equivalent to the fact that the algorithm do not delete all vertices.

The proof of the Lemma

The proof of the Lemma

Assume that during the algorithm all vertices are deleted.

The proof of the Lemma

Assume that during the algorithm all vertices are deleted.

In the ith step we delete fewer than $\delta(n-i)$ edges.

The proof of the Lemma

Assume that during the algorithm all vertices are deleted.

In the ith step we delete fewer than $\delta(n-i)$ edges.

During the process we delete fewer tham

$$\delta(n-1)+\delta(n-2)+\ldots+\delta\cdot 2+\delta\cdot 1=\delta\binom{n}{2}$$

edges.

The proof of the Lemma

Assume that during the algorithm all vertices are deleted.

In the ith step we delete fewer than $\delta(n-i)$ edges.

During the process we delete fewer tham

$$\delta(n-1)+\delta(n-2)+\ldots+\delta\cdot 2+\delta\cdot 1=\delta\binom{n}{2}$$

edges. Contradiction, since the number of edges in G is $\delta\binom{n}{2}$.

Extended Lemma

Extended Lemma

Extended Lemma

$\varepsilon_0 > 0$ is a small real, and N is a big natural number. Let G be a large enough graph ($|V(G)| := n > \nu(N, \varepsilon_0)$) with average degree at least $\delta \cdot (n-1)$.

Extended Lemma

Extended Lemma

$\varepsilon_0 > 0$ is a small real, and N is a big natural number. Let G be a large enough graph ($|V(G)| := n > \nu(N, \varepsilon_0)$) with average degree at least $\delta \cdot (n-1)$.

Then we have a subgraph R that satisfies

$$(\delta - \varepsilon_0)(|V(R)| - 1)$$

and $|V(R)| \geq N$.

The proof of the Extended Lemma

The proof of the Extended Lemma

Let $A = G$ be the actual graph. Until we find a vertex $x \in V(A)$ that has degree less than $(\delta - \varepsilon_0)|V(A)|$ delete x ($A \leftarrow A - \{x\}$).

The proof of the Extended Lemma

Let $A = G$ be the actual graph. Until we find a vertex $x \in V(A)$ that has degree less than $(\delta - \varepsilon_0)|V(A)|$ delete x ($A \leftarrow A - \{x\}$).

The goal is to prove that the process halts when we have more than N vertices.

The proof of the Extended Lemma

Let $A = G$ be the actual graph. Until we find a vertex $x \in V(A)$ that has degree less than $(\delta - \varepsilon_0)|V(A)|$ delete x $(A \leftarrow A - \{x\})$.

The goal is to prove that the process halts when we have more than N vertices.

If this is not the case then the number of deleted edges and the number of remainder edges is at most

$$(\delta - \varepsilon_0)(n-1) + (\delta - \varepsilon_0)(n-2) + \ldots + (\delta - \varepsilon_0)(N+1) + \binom{N}{2}$$
$$= (\delta - \varepsilon_0)\binom{n}{2} + (1 - \delta + \varepsilon_0)\binom{N}{2}.$$

The number of edges in G is at least $\delta\binom{n}{2}$.

The proof of the Extended Lemma

Let $A = G$ be the actual graph. Until we find a vertex $x \in V(A)$ that has degree less than $(\delta - \varepsilon_0)|V(A)|$ delete x ($A \leftarrow A - \{x\}$).

The goal is to prove that the process halts when we have more than N vertices.

If this is not the case then the number of deleted edges and the number of remainder edges is at most

$$(\delta - \varepsilon_0)(n-1) + (\delta - \varepsilon_0)(n-2) + \ldots + (\delta - \varepsilon_0)(N+1) + \binom{N}{2}$$
$$= (\delta - \varepsilon_0)\binom{n}{2} + (1 - \delta + \varepsilon_0)\binom{N}{2}.$$

The number of edges in G is at least $\delta\binom{n}{2}$.

If n is "large", then this is a contradiction.

The new Claim

The new Claim

The new Claim

Let $\varepsilon > 0$ be an arbitrary real. Let $G \in \mathcal{G}_n$ be a graph with minimal degree at least

$$\left(1 - \frac{1}{k} + \frac{\varepsilon}{2}\right)(n-1).$$

The new Claim

The new Claim

Let $\varepsilon > 0$ be an arbitrary real. Let $G \in \mathcal{G}_n$ be a graph with minimal degree at least

$$\left(1 - \frac{1}{k} + \frac{\varepsilon}{2}\right)(n-1).$$

Them for any $s \in \mathbb{N}$ a large enough G contains $K_{(k+1)\times s}$ subgraph.

The proof of the new Claim

The proof of the new Claim

The proof is induction on k.

The proof of the new Claim

The proof is induction on k. We are going to find a sequence of subgraphs.

The proof of the new Claim

The proof is induction on k. We are going to find a sequence of subgraphs. The scheme is

$$K_{1\times s_1} \to K_{2\times s_2} \to K_{3\times s_3} \to \ldots \to K_{(k-1)\times s_{k-1}} \to K_{k\times s_k}.$$

The proof of the new Claim

The proof is induction on k. We are going to find a sequence of subgraphs. The scheme is

$$K_{1\times s_1} \to K_{2\times s_2} \to K_{3\times s_3} \to \ldots \to K_{(k-1)\times s_{k-1}} \to K_{k\times s_k}.$$

The final s_k parameter is the vaule s in the Claim.

The proof of the new Claim

The proof is induction on k. We are going to find a sequence of subgraphs. The scheme is

$$K_{1\times s_1} \to K_{2\times s_2} \to K_{3\times s_3} \to \ldots \to K_{(k-1)\times s_{k-1}} \to K_{k\times s_k}.$$

The final s_k parameter is the vaule s in the Claim. The s_i values will satisfy that $s_i \gg s_{i+1}$.

The proof of the new Claim

The proof is induction on k. We are going to find a sequence of subgraphs. The scheme is

$$K_{1\times s_1} \to K_{2\times s_2} \to K_{3\times s_3} \to \ldots \to K_{(k-1)\times s_{k-1}} \to K_{k\times s_k}.$$

The final s_k parameter is the vaule s in the Claim. The s_i values will satisfy that $s_i \gg s_{i+1}$.

• The start of the induction is obvious: $K_{1\times s_1}$ is an empty graph on s_1 verices and n is large enough.

The proof of the new Claim

The proof is induction on k. We are going to find a sequence of subgraphs. The scheme is

$$K_{1\times s_1} \to K_{2\times s_2} \to K_{3\times s_3} \to \ldots \to K_{(k-1)\times s_{k-1}} \to K_{k\times s_k}.$$

The final s_k parameter is the vaule s in the Claim. The s_i values will satisfy that $s_i \gg s_{i+1}$.

• The start of the induction is obvious: $K_{1\times s_1}$ is an empty graph on s_1 verices and n is large enough.

• For the induction step we need to prove that for any s we can find a large $S = S(s)$, that in a large enough G satisfying the assumption on the minimal degree if $K_{\ell\times S}$ is a subgraph, then a subgraph, $K_{(\ell+1)\times s}$ is guaranteed too.

Proof (continued)

Proof (continued)

Let F be the set of vertices, where we have the subgraph $K_{\ell\times S}$ ($|F| = \ell S$, F divided into ℓ disjoint subsets of size S, the parts of F).

Proof (continued)

Let F be the set of vertices, where we have the subgraph $K_{\ell\times S}$ ($|F| = \ell S$, F divided into ℓ disjoint subsets of size S, the parts of F).

The vertices of $\overline{F} = V(G) - F$ are classified as good (J is the set of them) and bad (R is the set of them). Hence $\overline{F} = J\dot{\cup}R$.

Proof (continued)

Let F be the set of vertices, where we have the subgraph $K_{\ell\times S}$ ($|F| = \ell S$, F divided into ℓ disjoint subsets of size S, the parts of F).

The vertices of $\overline{F} = V(G) - F$ are classified as good (J is the set of them) and bad (R is the set of them). Hence $\overline{F} = J\dot{\cup}R$.

$x \in J/x$ is good iff x has at least s neighbors on each part of F.

Proof (continued)

Proof (continued)

If $|J| > (s-1)\binom{S}{s}^{\ell}$, we are done: each good vertex has a type:

Proof (continued)

If $|J| > (s-1)\binom{S}{s}^{\ell}$, we are done: each good vertex has a type: s (arbitrary) neighbors in all ℓ parts.

Proof (continued)

If $|J| > (s-1)\binom{S}{s}^{\ell}$, we are done: each good vertex has a type: s (arbitrary) neighbors in all ℓ parts.

$\binom{S}{s}^{\ell}$ is the number of possible types.

Proof (continued)

If $|J| > (s-1)\binom{S}{s}^{\ell}$, we are done: each good vertex has a type: s (arbitrary) neighbors in all ℓ parts.

$\binom{S}{s}^{\ell}$ is the number of possible types.

The size of J is so big that the pigeon hole principle guaratees that we have at least s good vertices with the same type.

Proof (continued)

If $|J| > (s-1)\binom{S}{s}^{\ell}$, we are done: each good vertex has a type: s (arbitrary) neighbors in all ℓ parts.

$\binom{S}{s}^{\ell}$ is the number of possible types.

The size of J is so big that the pigeon hole principle guaratees that we have at least s good vertices with the same type.

We take s good vertices with the same type. We also take the $\ell \cdot s$ vertices corresponding the common type. These $(\ell+1)s$ vertices span a $K_{(\ell+1)\times s}$ subgraph.

Proof: Missing edges between F and $\overline{F}$ I

Proof: Missing edges between F and $\overline{F}$ I

We have a lower bound on each degree,

Proof: Missing edges between F and $\overline{F}$ I

We have a lower bound on each degree, specially in each vertex in F is NOT connected to at most

$$\left(\frac{1}{k}-\frac{\varepsilon}{2}\right)(n-1)$$

other vertex.

Proof: Missing edges between F and $\overline{F}$ I

We have a lower bound on each degree, specially in each vertex in F is NOT connected to at most

$$\left(\frac{1}{k}-\frac{\varepsilon}{2}\right)(n-1)$$

other vertex.

So between F and $\overline{F}$ at most

$$|F|\cdot\left(\frac{1}{k}-\frac{\varepsilon}{2}\right)(n-1)=\ell S\cdot\left(\frac{1}{k}-\frac{\varepsilon}{2}\right)(n-1)\leq S\cdot\left(1-\frac{\varepsilon\ell}{2}\right)n$$

edges are missing.

Proof: Missing edges between F and $\overline{F}$ II

Proof: Missing edges between F and $\overline{F}$ II

On the other hand each vertex in R has at most s

Proof: Missing edges between F and $\overline{F}$ II

On the other hand each vertex in R has at most s

So every vertex in R is not connected to at least $S - s$ vertices in F. The number of missing edges is at least

$$|R|(S - s) = (n - |F| - |J|)(S - s) = (n - \ell S - |J|)(S - s).$$

Proof: Missing edges between F and $\overline{F}$ I+II

Proof: Missing edges between F and $\overline{F}$ I+II

$$(n-\ell S-|J|)(S-s)\leq S\cdot\left(1-\frac{\varepsilon\ell}{2}\right)n.$$

Proof: Missing edges between F and $\overline{F}$ I+II

$$(n-\ell S-|J|)(S-s)\leq S\cdot\left(1-\frac{\varepsilon\ell}{2}\right)n.$$

After rearrangement

$$(n-\ell S)(S-s)-S\cdot\left(1-\frac{\varepsilon\ell}{2}\right)n\leq (S-s)|J|.$$

Proof: Missing edges between F and $\overline{F}$ I+II

$$(n-\ell S-|J|)(S-s)\leq S\cdot\left(1-\frac{\varepsilon\ell}{2}\right)n.$$

After rearrangement

$$(n-\ell S)(S-s)-S\cdot\left(1-\frac{\varepsilon\ell}{2}\right)n\leq (S-s)|J|.$$

Hence

$$\left(\frac{\varepsilon}{2}\ell S-s\right)n-\ell S(S-s)\leq (S-s)|J|,$$

$$\frac{\varepsilon\ell S-2s}{2(S-s)}\cdot n-\ell S\leq |J|.$$

Proof: The end

Proof: The end

We are given ε, ℓ, s and we got to choose the value S.

Proof: The end

We are given ε, ℓ, s and we got to choose the value S.

With the choice of S we can achieve that the coefficient of n in the lower bound on $|J|$ is positive.

Proof: The end

We are given ε, ℓ, s and we got to choose the value S.

With the choice of S we can achieve that the coefficient of n in the lower bound on $|J|$ is positive.

After this for large enough n we have many good vertices.

Proof: The end

We are given ε, ℓ, s and we got to choose the value S.

With the choice of S we can achieve that the coefficient of n in the lower bound on $|J|$ is positive.

After this for large enough n we have many good vertices.

This guarantees the induction step.

Break

What do we know and what don't?

What do we know and what don't?

First, if the forbidden subgraph T is bipartite and contains a cycle then we do not know too much.

What do we know and what don't?

First, if the forbidden subgraph T is bipartite and contains a cycle then we do not know too much.

Based on the quoted results, in all other cases we know the exact order of magnitude of $ext(n; T)$.

What do we know and what don't?

First, if the forbidden subgraph T is bipartite and contains a cycle then we do not know too much.

Based on the quoted results, in all other cases we know the exact order of magnitude of $ext(n; T)$.

If T is a bipartite graph with cycle, then considering $ext(n; T)$ is called *the degenerated case*.

What do we know and what don't?

First, if the forbidden subgraph T is bipartite and contains a cycle then we do not know too much.

Based on the quoted results, in all other cases we know the exact order of magnitude of $ext(n; T)$.

If T is a bipartite graph with cycle, then considering $ext(n; T)$ is called *the degenerated case*.

The degenerated case we have only a few exact results. We the know the exact the order of magnitude of $ext(n; T)$, when T is C_4, C_6, C_{10} or $K_{2,k}$, $K_{3,k}$.

What do we know and what don't?

First, if the forbidden subgraph T is bipartite and contains a cycle then we do not know too much.

Based on the quoted results, in all other cases we know the exact order of magnitude of $ext(n; T)$.

If T is a bipartite graph with cycle, then considering $ext(n; T)$ is called *the degenerated case*.

The degenerated case we have only a few exact results. We the know the exact the order of magnitude of $ext(n; T)$, when T is C_4, C_6, C_{10} or $K_{2,k}$, $K_{3,k}$.

For example the case of C_8, C_{12}, C_{14}, ..., $K_{4,4}$, $K_{4,5}$, ..., or the cube graph (the order of magnitude of $ext(n; T)$) is not known.

The forbidden C_4

The forbidden C_4

Determining the order of magnitude of $ext(n; C_4)$ consists of two parts.

The forbidden C_4

Determining the order of magnitude of $ext(n; C_4)$ consists of two parts.

We need to prove a mathematical statement: If $G \in \mathcal{G}_n$ is C_4-free graph them it cannot have too many edges.

The forbidden C_4

Determining the order of magnitude of $ext(n; C_4)$ consists of two parts.

We need to prove a mathematical statement: If $G \in \mathcal{G}_n$ is C_4-free graph them it cannot have too many edges.

On the other hand, we need to construct one $G \in \mathcal{G}_n$ that is C_4-free, and has many edges.

The mathematical theorem

The mathematical theorem

Theorem

Let $G \in \mathcal{G}_n$ be a C_4-free graph. Them

$$|E(G)| \leq \frac{1}{4} \cdot n\sqrt{4n-3} + \frac{1}{4} \cdot n.$$

The construction

The construction

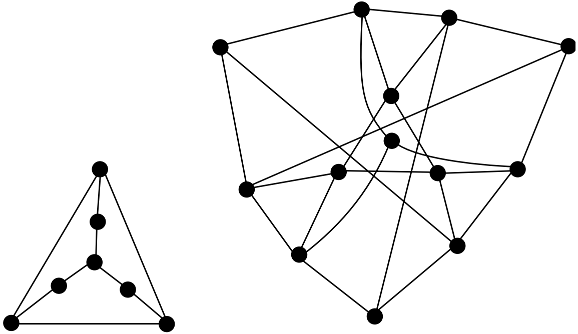
The construction is involved. It requires basic knowledge of finite fields and finite geometries.

The construction

The construction is involved. It requires basic knowledge of finite fields and finite geometries. We do not discuss.

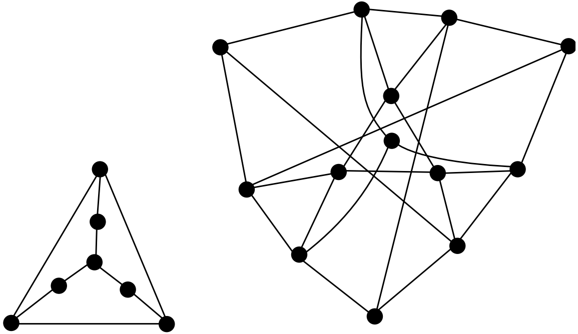
The construction

The construction is involved. It requires basic knowledge of finite fields and finite geometries. We do not discuss.



The construction

The construction is involved. It requires basic knowledge of finite fields and finite geometries. We do not discuss.



We just mention one important property for us: $|E| \sim \frac{1}{2}n^{3/2}$.

The summary of the results

The summary of the results

Theorem

$$ext(n, C_4) \sim \frac{1}{2} n^{\frac{3}{2}}.$$

This is the end!

Thank you for your attention!