The Semi-random Process

(The DEs method in action)

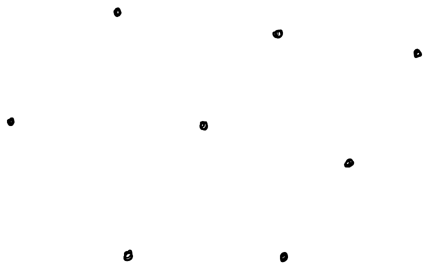
Paweł Prałat

Updated: 2023/11/29

Department of Mathematics, Toronto Metropolitan University

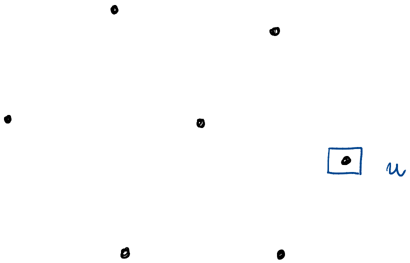
Definition of the Process

Definition



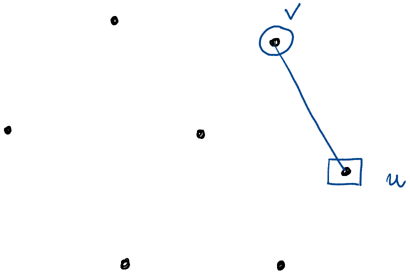
The semi-random graph process is a single player game in which the player is initially presented the empty graph G_0 on the vertex set $[n] := \{1, \ldots, n\}$.

Definition



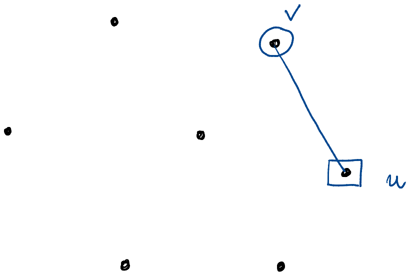
In each round $t \geq 1$, a vertex u_t (square) is drawn independently and u.a.r. (uniformly at random) from $[n]$ and then presented to the player.

Definition



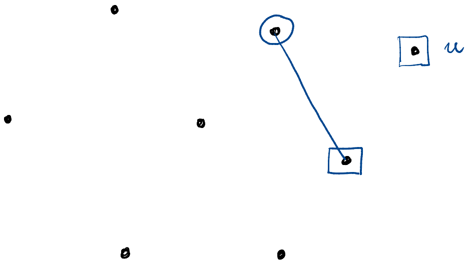
The player then adaptively selects a vertex v_t (circle), and adds the edge $u_t v_t$ to G_{t-1} to form the graph G_t.

Definition



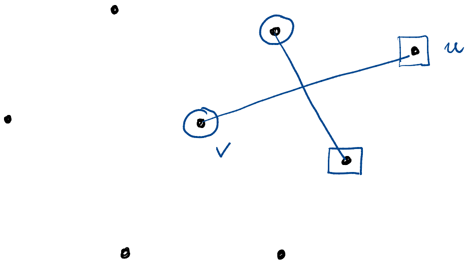
For a fixed (edge) monotonic graph property $\mathscr{P}$ (say, the existence of a perfect matching), the objective is to satisfy this property with high probability in as few rounds as possible.

Definition



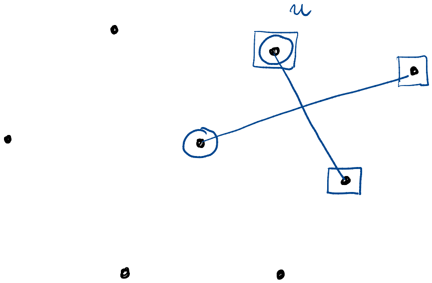
For a fixed (edge) monotonic graph property $\mathscr{P}$ (say, the existence of a perfect matching), the objective is to satisfy this property with high probability in as few rounds as possible.

Definition



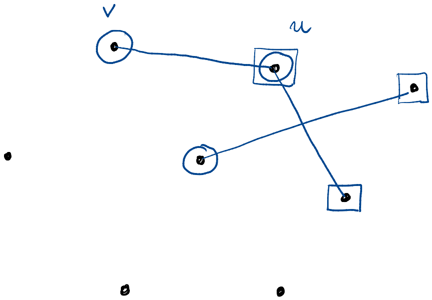
For a fixed (edge) monotonic graph property $\mathscr{P}$ (say, the existence of a perfect matching), the objective is to satisfy this property with high probability in as few rounds as possible.

Definition



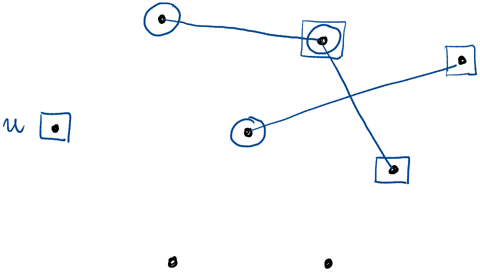
For a fixed (edge) monotonic graph property $\mathscr{P}$ (say, the existence of a perfect matching), the objective is to satisfy this property with high probability in as few rounds as possible.

Definition



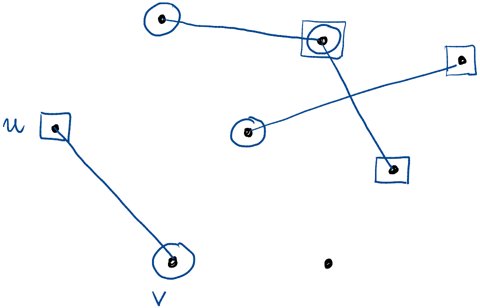
For a fixed (edge) monotonic graph property $\mathscr{P}$ (say, the existence of a perfect matching), the objective is to satisfy this property with high probability in as few rounds as possible.

Definition



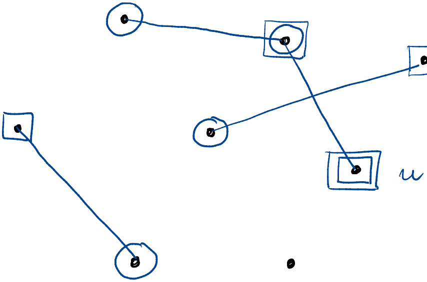
For a fixed (edge) monotonic graph property $\mathscr{P}$ (say, the existence of a perfect matching), the objective is to satisfy this property with high probability in as few rounds as possible.

Definition

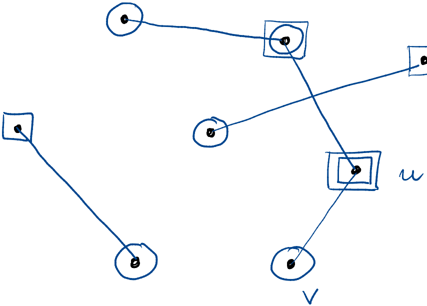


For a fixed (edge) monotonic graph property $\mathscr{P}$ (say, the existence of a perfect matching), the objective is to satisfy this property with high probability in as few rounds as possible.

Definition

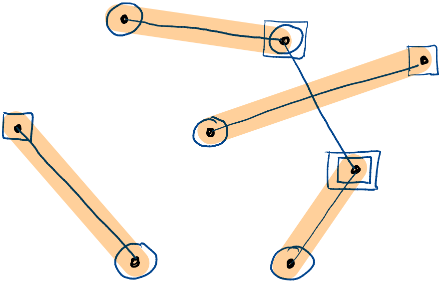


For a fixed (edge) monotonic graph property $\mathscr{P}$ (say, the existence of a perfect matching), the objective is to satisfy this property with high probability in as few rounds as possible.



For a fixed (edge) monotonic graph property $\mathscr{P}$ (say, the existence of a perfect matching), the objective is to satisfy this property with high probability in as few rounds as possible.

Definition



For a fixed (edge) monotonic graph property $\mathscr{P}$ (say, the existence of a perfect matching), the objective is to satisfy this property with high probability in as few rounds as possible.

Definition

The semi-random process $(G_t)_{t \geq 0}$ was suggested by Peleg Michaeli, and formally introduced by Ben-Eliezer, Hefetz, Kronenberg, Parczyk, Shikhelman, and Stojaković (RSA, 2020).

Definition

The semi-random process $(G_t)_{t\geq 0}$ was suggested by Peleg Michaeli, and formally introduced by Ben-Eliezer, Hefetz, Kronenberg, Parczyk, Shikhelman, and Stojaković (RSA, 2020).

It may be viewed as a generalization of the Erdős–Rényi random graph process. (The player chooses v_t u.a.r.)

Definition

The semi-random process $(G_t)_{t \geq 0}$ was suggested by Peleg Michaeli, and formally introduced by Ben-Eliezer, Hefetz, Kronenberg, Parczyk, Shikhelman, and Stojaković (RSA, 2020).

It may be viewed as a generalization of the Erdős–Rényi random graph process. (The player chooses v_t u.a.r.)

But, in fact, it generalizes many interesting and well-studied processes.

Definition

The semi-random process $(G_t)_{t\geq 0}$ was suggested by Peleg Michaeli, and formally introduced by Ben-Eliezer, Hefetz, Kronenberg, Parczyk, Shikhelman, and Stojaković (RSA, 2020).

It may be viewed as a generalization of the Erdős–Rényi random graph process. (The player chooses v_t u.a.r.)

But, in fact, it generalizes many interesting and well-studied processes.

Our results are asymptotic in nature, that is, we will always assume that $n \to \infty$.

Definition

The semi-random process $(G_t)_{t\geq 0}$ was suggested by Peleg Michaeli, and formally introduced by Ben-Eliezer, Hefetz, Kronenberg, Parczyk, Shikhelman, and Stojaković (RSA, 2020).

It may be viewed as a generalization of the Erdős–Rényi random graph process. (The player chooses v_t u.a.r.)

But, in fact, it generalizes many interesting and well-studied processes.

Our results are asymptotic in nature, that is, we will always assume that $n \to \infty$.

Event $\mathscr{E} = (\mathscr{E}_n)_{n\geq 1}$ holds asymptotically almost surely (a.a.s.) if the probability that $\mathscr{E}_n$ holds tends to 1 as $n \to \infty$.

Definition

Upper Bounds: Show that there exists a strategy $\mathcal{S}$ and a function $\tau = \tau(n)$, such that $G_\tau^{\mathcal{S}}$ satisfies $\mathscr{P}$ a.a.s.

($G_\tau^{\mathcal{S}}$ is the (random) graph on $[n]$ formed after executing $\mathcal{S}$ for τ rounds.)

Definition

Upper Bounds: Show that there exists a strategy $\mathcal{S}$ and a function $\tau = \tau(n)$, such that $G_\tau^{\mathcal{S}}$ satisfies $\mathcal{P}$ a.a.s.

($G_\tau^{\mathcal{S}}$ is the (random) graph on $[n]$ formed after executing $\mathcal{S}$ for τ rounds.)

Lower Bounds: Show that there exists a function $\tau = \tau(n)$, such that for any strategy $\mathcal{S}$, $G_\tau^{\mathcal{S}}$ does not satisfy $\mathcal{P}$ a.a.s.

Definition

Upper Bounds: Show that there exists a strategy $\mathcal{S}$ and a function $\tau = \tau(n)$, such that $G_\tau^{\mathcal{S}}$ satisfies $\mathscr{P}$ a.a.s.

($G_\tau^{\mathcal{S}}$ is the (random) graph on $[n]$ formed after executing $\mathcal{S}$ for τ rounds.)

Lower Bounds: Show that there exists a function $\tau = \tau(n)$, such that for any strategy $\mathcal{S}$, $G_\tau^{\mathcal{S}}$ does not satisfy $\mathscr{P}$ a.a.s.

Ben-Eliezer et al. (SODA, 2020) showed that if H_n is a bounded degree spanning graph, then there is a strategy which constructs a copy of H_n in a linear number of rounds a.a.s.

Definition

Upper Bounds: Show that there exists a strategy $\mathcal{S}$ and a function $\tau = \tau(n)$, such that $G_\tau^{\mathcal{S}}$ satisfies $\mathscr{P}$ a.a.s.

($G_\tau^{\mathcal{S}}$ is the (random) graph on $[n]$ formed after executing $\mathcal{S}$ for τ rounds.)

Lower Bounds: Show that there exists a function $\tau = \tau(n)$, such that for any strategy $\mathcal{S}$, $G_\tau^{\mathcal{S}}$ does not satisfy $\mathscr{P}$ a.a.s.

Ben-Eliezer et al. (SODA, 2020) showed that if H_n is a bounded degree spanning graph, then there is a strategy which constructs a copy of H_n in a linear number of rounds a.a.s.

Thus, for a specific graph H_n (such as a perfect matching, or Hamiltonian cycle), the goal is to find the optimal (linear-time) strategy.

Summary of Results

Summary – Perfect Matchings

Perfect Matchings:

– upper bound $1.73576n$ improved to $1.20524n$.

– lower bound $0.69314n$ improved to $0.93261n$.

(Gao, MacRury, Prałat, SIDMA, 2022)

Summary – Hamilton Cycles

Hamilton Cycles:

– upper bound $3n$ improved to $2.61135n$.

– lower bound $1.21973n$ improved by ϵn.

(Gao, Kamiński, MacRury, Prałat, Eur J Comb, 2022)

Summary – Hamilton Cycles

Hamilton Cycles:

– upper bound $3n$ improved to $2.61135n$.

– lower bound $1.21973n$ improved by ϵn.

(Gao, Kamiński, MacRury, Prałat, Eur J Comb, 2022)

– upper bound improved to $2.01678n$.

– lower bound improved to $1.26575n$.

(Gao, MacRury, Prałat, RANDOM 2022, 2022)

Summary – Hamilton Cycles

Hamilton Cycles:

– upper bound $3n$ improved to $2.61135n$.

– lower bound $1.21973n$ improved by ϵn.

(Gao, Kamiński, MacRury, Prałat, Eur J Comb, 2022)

– upper bound improved to $2.01678n$.

– lower bound improved to $1.26575n$.

(Gao, MacRury, Prałat, RANDOM 2022, 2022)

– upper bound further improved to $1.84887n$.

(Frieze, Sorkin, ArXiv, 2022)

Summary – Hamilton Cycles

Hamilton Cycles:

– upper bound $3n$ improved to $2.61135n$.

– lower bound $1.21973n$ improved by ϵn.

(Gao, Kamiński, MacRury, Prałat, Eur J Comb, 2022)

– upper bound improved to $2.01678n$.

– lower bound improved to $1.26575n$.

(Gao, MacRury, Prałat, RANDOM 2022, 2022)

– upper bound further improved to $1.84887n$.

(Frieze, Sorkin, ArXiv, 2022)

– upper bound further improved to $1.81701n$.

(Gao, Frieze, MacRury, Prałat, Sorkin, ArXiv, 2023+)

Summary – Small Subgraphs

Small Subgraphs:

– construct fixed graph G of degeneracy d.

– upper bound $n^{(d-1)/d}\omega$ (Ben-Eliezer et al., RSA, 2020).

– lower bound $n^{(d-1)/d}/\omega$ for K_{d+1} (Ben-Eliezer et al., RSA, 2020).

Summary – Small Subgraphs

Small Subgraphs:

– construct fixed graph G of degeneracy d.

– upper bound $n^{(d-1)/d}\omega$ (Ben-Eliezer et al., RSA, 2020).

– lower bound $n^{(d-1)/d}/\omega$ for K_{d+1} (Ben-Eliezer et al., RSA, 2020).

– lower bound $n^{(d-1)/d}/\omega$ for any graph G.

– generalization to hypergraphs.

– tight results for 1 square and any number of circles
(Behague, Marbach, Prałat, Ruciński, ArXiv, 2021+)

– many open questions are left for at least 2 squares!
(Behague, Prałat, Ruciński, 2023++)

Summary – Other Directions

Other Directions:

– sharp thresholds (more general class of processes)

(MacRury, Surya, ArXiv, 2022+)

Summary – Other Directions

Other Directions:

– sharp thresholds (more general class of processes)
(MacRury, Surya, ArXiv, 2022+)

– k-factors and k-connectivity
(Koerts, MSc. thesis, 2022)

Summary – Other Directions

Other Directions:

– sharp thresholds (more general class of processes)
(MacRury, Surya, ArXiv, 2022+)

– k-factors and k-connectivity
(Koerts, MSc. thesis, 2022)

– (large) complete graphs, independent sets, chromatic number
(Gamarnik, Kang, Prałat, ArXiv, 2023+)

Summary – Generalizations

– Hypergraphs

(Behague, Marbach, Prałat, Ruciński, ArXiv, 2021+)

(Behague, Prałat, Ruciński, 2023++)

(Molloy, Prałat, Sorkin, 2023++)

Summary – Generalizations

– Hypergraphs

(Behague, Marbach, Prałat, Ruciński, ArXiv, 2021+)

(Behague, Prałat, Ruciński, 2023++)

(Molloy, Prałat, Sorkin, 2023++)

– select an edge from a random spanning tree of K_n

(Burova, Lichev, ArXiv, 2022+)

Summary – Generalizations

– Hypergraphs
(Behague, Marbach, Prałat, Ruciński, ArXiv, 2021+)
(Behague, Prałat, Ruciński, 2023++)
(Molloy, Prałat, Sorkin, 2023++)

– select an edge from a random spanning tree of K_n
(Burova, Lichev, ArXiv, 2022+)

– vertices presented follow a random permutation
(Gilboa, Hefetz, EuroComb 2021, 2021)

Summary – Generalizations

– Hypergraphs
(Behague, Marbach, Prałat, Ruciński, ArXiv, 2021+)
(Behague, Prałat, Ruciński, 2023++)
(Molloy, Prałat, Sorkin, 2023++)

– select an edge from a random spanning tree of K_n
(Burova, Lichev, ArXiv, 2022+)

– vertices presented follow a random permutation
(Gilboa, Hefetz, EuroComb 2021, 2021)

– "power of k choices"
(Prałat, Singh, ArXiv, 2023+)

Perfect Matchings

Emulating k-out Process

The semi-random process $(G_t)_{t \geq 0}$ can emulate the well-known k-out process $H(k)$: each vertex independently connects to k randomly selected vertices.

Emulating k-out Process

The semi-random process $(G_t)_{t\geq 0}$ can emulate the well-known k-out process $H(k)$: each vertex independently connects to k randomly selected vertices.

Formally, we want $H(k)$ to be a subgraph of G_t for some value of $t \geq kn$, usually close to each other.

Emulating k-out Process

The semi-random process $(G_t)_{t \geq 0}$ can emulate the well-known k-out process $H(k)$: each vertex independently connects to k randomly selected vertices.

Formally, we want $H(k)$ to be a subgraph of G_t for some value of $t \geq kn$, usually close to each other.

(If $H(k)$ has some monotone property $\mathcal{P}$, then G_t has it too.)

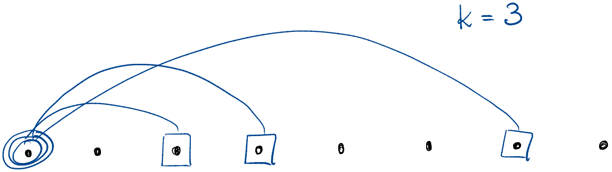
Emulating k-out Process

The semi-random process $(G_t)_{t\geq 0}$ can emulate the well-known k-out process $H(k)$: each vertex independently connects to k randomly selected vertices.

Formally, we want $H(k)$ to be a subgraph of G_t for some value of $t \geq kn$, usually close to each other.

(If $H(k)$ has some monotone property $\mathcal{P}$, then G_t has it too.)

There exists a strategy such that a.a.s. $H(k) \subseteq G_{kn+\omega}$.
(Ben-Eliezer, Hefetz, Kronenberg, Parczyk, Shikhelman, and Stojaković, RSA, 2020)



Perfect Matchings: Upper Bounds

A.a.s. k-out process has a perfect matching when $k \geq 2$.
(Frieze, JCTB, 1986)

Perfect Matchings: Upper Bounds

A.a.s. k-out process has a perfect matching when $k \geq 2$.
(Frieze, JCTB, 1986)
Implication: a.a.s. there exists a strategy to create a perfect matching in $(2 + o(1))n$ rounds.

Perfect Matchings: Upper Bounds

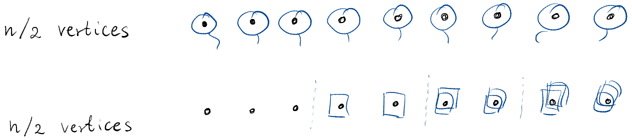
A.a.s. k-out process has a perfect matching when $k \geq 2$.
(Frieze, JCTB, 1986)
Implication: a.a.s. there exists a strategy to create a perfect matching in $(2 + o(1))n$ rounds.

The semi-random process $(G_t)_{t \geq 0}$ can emulate the $1 + 2/e$-out bipartite process:

Perfect Matchings: Upper Bounds

A.a.s. k-out process has a perfect matching when $k \geq 2$.
(Frieze, JCTB, 1986)
Implication: a.a.s. there exists a strategy to create a perfect matching in $(2 + o(1))n$ rounds.

The semi-random process $(G_t)_{t \geq 0}$ can emulate the $1 + 2/e$-out bipartite process:
– start with the bipartite version of the 1-out process,



Perfect Matchings: Upper Bounds

A.a.s. k-out process has a perfect matching when $k \geq 2$.
(Frieze, JCTB, 1986)
Implication: a.a.s. there exists a strategy to create a perfect matching in $(2 + o(1))n$ rounds.

The semi-random process $(G_t)_{t \geq 0}$ can emulate the $1 + 2/e$-out bipartite process:

– start with the bipartite version of the 1-out process,

– unpopular vertices (chosen by at most one vertex) chose another out-neighbour.

Perfect Matchings: Upper Bounds

A.a.s. k-out process has a perfect matching when $k \geq 2$.
(Frieze, JCTB, 1986)
Implication: a.a.s. there exists a strategy to create a perfect matching in $(2 + o(1))n$ rounds.

The semi-random process $(G_t)_{t \geq 0}$ can emulate the $1 + 2/e$-out bipartite process:
– start with the bipartite version of the 1-out process,
– unpopular vertices (chosen by at most one vertex) chose another out-neighbour.

A.a.s. $1 + 2/e$-out bipartite process has a perfect matching.
(Karoński, Overman, Pittel, JCTB, 2020).

Perfect Matchings: Upper Bounds

A.a.s. k-out process has a perfect matching when $k \geq 2$. (Frieze, JCTB, 1986)
Implication: a.a.s. there exists a strategy to create a perfect matching in $(2 + o(1))n$ rounds.

The semi-random process $(G_t)_{t \geq 0}$ can emulate the $1 + 2/e$-out bipartite process:
– start with the bipartite version of the 1-out process,
– unpopular vertices (chosen by at most one vertex) chose another out-neighbour.

A.a.s. $1 + 2/e$-out bipartite process has a perfect matching. (Karoński, Overman, Pittel, JCTB, 2020).
Implication: a.a.s. there exists a strategy to create a perfect matching in $(1 + 2/e + o(1))n < 1.73576n$ rounds.

Perfect Matchings: Upper Bounds

A.a.s. there exists a strategy to create a perfect matching in $(\beta + 10^{-5})n \leq 1.20524n$ rounds, where β is derived from a system of differential equations.
(Gao, MacRury, Prałat, SIDMA, 2022)

Perfect Matchings: Upper Bounds

A.a.s. there exists a strategy to create a perfect matching in $(\beta + 10^{-5})n \leq 1.20524n$ rounds, where β is derived from a system of differential equations.
(Gao, MacRury, Prałat, SIDMA, 2022)

– Fully adaptive algorithm

Perfect Matchings: Upper Bounds

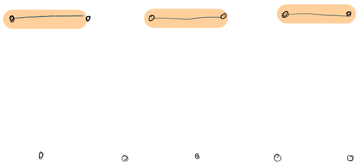
A.a.s. there exists a strategy to create a perfect matching in $(\beta + 10^{-5})n \leq 1.20524n$ rounds, where β is derived from a system of differential equations.
(Gao, MacRury, Prałat, SIDMA, 2022)

– Fully adaptive algorithm
– Our randomized algorithm (giving an upper bound of $1.28n$) keeps building the matching greedily whenever possible but also keeps one random extension for future augmentations

Perfect Matchings: Upper Bounds

A.a.s. there exists a strategy to create a perfect matching in $(\beta + 10^{-5})n \leq 1.20524n$ rounds, where β is derived from a system of differential equations.
(Gao, MacRury, Prałat, SIDMA, 2022)

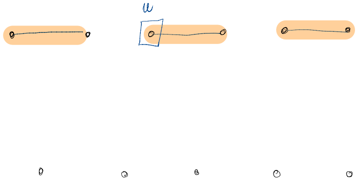
– Fully adaptive algorithm
– Our randomized algorithm (giving an upper bound of $1.28n$) keeps building the matching greedily whenever possible but also keeps one random extension for future augmentations



Perfect Matchings: Upper Bounds

A.a.s. there exists a strategy to create a perfect matching in $(\beta + 10^{-5})n \leq 1.20524n$ rounds, where β is derived from a system of differential equations.
(Gao, MacRury, Prałat, SIDMA, 2022)

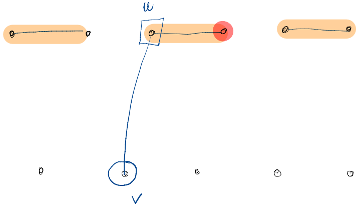
– Fully adaptive algorithm
– Our randomized algorithm (giving an upper bound of $1.28n$) keeps building the matching greedily whenever possible but also keeps one random extension for future augmentations



Perfect Matchings: Upper Bounds

A.a.s. there exists a strategy to create a perfect matching in $(\beta + 10^{-5})n \leq 1.20524n$ rounds, where β is derived from a system of differential equations.
(Gao, MacRury, Prałat, SIDMA, 2022)

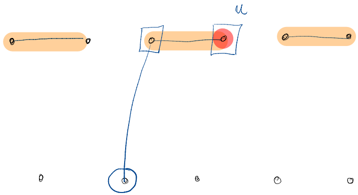
– Fully adaptive algorithm
– Our randomized algorithm (giving an upper bound of $1.28n$) keeps building the matching greedily whenever possible but also keeps one random extension for future augmentations



Perfect Matchings: Upper Bounds

A.a.s. there exists a strategy to create a perfect matching in $(\beta + 10^{-5})n \leq 1.20524n$ rounds, where β is derived from a system of differential equations.
(Gao, MacRury, Prałat, SIDMA, 2022)

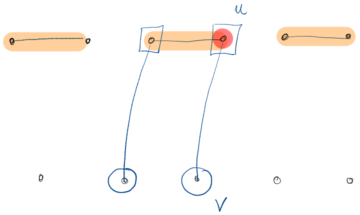
– Fully adaptive algorithm
– Our randomized algorithm (giving an upper bound of $1.28n$) keeps building the matching greedily whenever possible but also keeps one random extension for future augmentations



Perfect Matchings: Upper Bounds

A.a.s. there exists a strategy to create a perfect matching in $(\beta + 10^{-5})n \leq 1.20524n$ rounds, where β is derived from a system of differential equations.
(Gao, MacRury, Prałat, SIDMA, 2022)

– Fully adaptive algorithm
– Our randomized algorithm (giving an upper bound of $1.28n$) keeps building the matching greedily whenever possible but also keeps one random extension for future augmentations



Perfect Matchings: Upper Bounds

A.a.s. there exists a strategy to create a perfect matching in $(\beta + 10^{-5})n \leq 1.20524n$ rounds, where β is derived from a system of differential equations.
(Gao, MacRury, Prałat, SIDMA, 2022)

– Fully adaptive algorithm
– Our randomized algorithm (giving an upper bound of $1.28n$) keeps building the matching greedily whenever possible but also keeps one random extension for future augmentations

Perfect Matchings: Upper Bounds

A.a.s. there exists a strategy to create a perfect matching in $(\beta + 10^{-5})n \le 1.20524n$ rounds, where β is derived from a system of differential equations.
(Gao, MacRury, Prałat, SIDMA, 2022)

– Fully adaptive algorithm
– Our randomized algorithm (giving an upper bound of $1.28n$) keeps building the matching greedily whenever possible but also keeps one random extension for future augmentations
– Our deterministic algorithm keeps all extensions, has k deterministic greedy phases for $k \ge 1100$, and concludes by executing the randomized algorithm.

Analysing the Randomized Algorithm

We use the differential equation (DE) method of Nick Wormald for analysis.

Analysing the Randomized Algorithm

We use the differential equation (DE) method of Nick Wormald for analysis.

Given $t \geq 0$, let $X(t)$ denote the number of matched vertices, and let $R(t)$ denote the number of red vertices.

Analysing the Randomized Algorithm

We use the differential equation (DE) method of Nick Wormald for analysis.

Given $t \geq 0$, let $X(t)$ denote the number of matched vertices, and let $R(t)$ denote the number of red vertices.

Let $H_t := (X(i), R(i))_{0 \leq i \leq t}$. Note that H_t does *not* encompass the full history of random graph process at time t (i.e., $G_0, \ldots, G_t$)

Analysing the Randomized Algorithm

We use the differential equation (DE) method of Nick Wormald for analysis.

Given $t \geq 0$, let $X(t)$ denote the number of matched vertices, and let $R(t)$ denote the number of red vertices.

Let $H_t := (X(i), R(i))_{0 \leq i \leq t}$. Note that H_t does *not* encompass the full history of random graph process at time t (i.e., $G_0, \ldots, G_t$)

We condition on less information so that the circle placements amongst the unsaturated vertices remain u.a.r.

Deriving the Differential Equations

$$\mathbb{E}[X(t+1) - X(t) \mid H_t] \quad = \quad \frac{2(n - X(t) + R(t))}{n} + O(1/n),$$

Deriving the Differential Equations

$$
\begin{aligned}
\mathbb{E}[X(t+1) - X(t) \mid H_t] &= \frac{2(n - X(t) + R(t))}{n} + O(1/n), \\
\mathbb{E}[R(t+1) - R(t) \mid H_t] &= \frac{n - X(t)}{n} \cdot \frac{-2R(t)}{n - X(t)} \\
&\quad + \frac{R(t)}{n}\left(-1 - \frac{2(R(t)-1)}{n - X(t)}\right) \\
&\quad + \frac{X(t) - 2R(t)}{n} + O(1/n).
\end{aligned}
$$

Deriving the Differential Equations

$$
\begin{aligned}
\mathbb{E}[X(t+1) - X(t) \mid H_t] &= \frac{2(n - X(t) + R(t))}{n} + O(1/n), \\
\mathbb{E}[R(t+1) - R(t) \mid H_t] &= \frac{n - X(t)}{n} \cdot \frac{-2R(t)}{n - X(t)} \\
&\quad + \frac{R(t)}{n}\left(-1 - \frac{2(R(t) - 1)}{n - X(t)}\right) \\
&\quad + \frac{X(t) - 2R(t)}{n} + O(1/n).
\end{aligned}
$$

By writing $x(s) = X(sn)/n$ and $r(s) = R(sn)/n$ for $s \in [0, \infty)$, we have that

$$
\begin{aligned}
x' &= 2(1 - x + r), \\
r' &= \frac{-2r}{1 - x}(1 - x + r) - r + x - 2r,
\end{aligned}
$$

with the initial conditions $x(0) = r(0) = 0$.

Applying the DE Method

By DE method, a.a.s. $X(t) = x(t/n) \cdot n + o(n)$ for all $t \geq 0$.

Applying the DE Method

By DE method, a.a.s. $X(t) = x(t/n) \cdot n + o(n)$ for all $t \geq 0$.

Numerical DE solver shows that $x(s) = 1$ for $s \geq 1.28$, so $X(t) = n - o(n)$ for $t \geq 1.28n$.

Applying the DE Method

By DE method, a.a.s. $X(t) = x(t/n) \cdot n + o(n)$ for all $t \geq 0$.

Numerical DE solver shows that $x(s) = 1$ for $s \geq 1.28$, so $X(t) = n - o(n)$ for $t \geq 1.28n$.

The remaining $o(n)$ unsaturated vertices are matched via a clean-up algorithm which is analysed by a (lossy) elementary analysis.

Perfect Matchings: Lower Bounds

Trivial observation: no strategy can create a perfect matching in less than $n/2$ rounds.

Perfect Matchings: Lower Bounds

Trivial observation: no strategy can create a perfect matching in less than $n/2$ rounds.

There are two obvious necessary conditions, both giving exactly the same lower bound:
– the graph has the minimum degree at least 1,

Perfect Matchings: Lower Bounds

Trivial observation: no strategy can create a perfect matching in less than $n/2$ rounds.

There are two obvious necessary conditions, both giving exactly the same lower bound:

– the graph has the minimum degree at least 1,

– there are at least $n/2$ vertices with at least one square.

Perfect Matchings: Lower Bounds

Trivial observation: no strategy can create a perfect matching in less than $n/2$ rounds.

There are two obvious necessary conditions, both giving exactly the same lower bound:
– the graph has the minimum degree at least 1,
– there are at least $n/2$ vertices with at least one square.

A.a.s. no strategy can create a perfect matching in less than $(\ln(2) + o(1))n \geq 0.69314n$ rounds.
(Ben-Eliezer, Hefetz, Kronenberg, Parczyk, Shikhelman, and Stojaković, RSA, 2020)

Perfect Matchings: Lower Bounds

Let

$$\alpha = \inf\{b \geq 0 : g(b) \geq 1/2\},$$

where

$$g(b) := 1 + \frac{1-2b}{2}\exp(-b) - (b+1)\exp(-2b) - \frac{1}{2}\exp(-3b).$$

Then, a.a.s. no strategy can create a perfect matching in less than $(\alpha + o(1))n \geq 0.93261n$ rounds.
(Gao, MacRury, Prałat, SIDMA, 2022)

Perfect Matchings: Lower Bounds

Let

$$\alpha = \inf\{b \geq 0 : g(b) \geq 1/2\},$$

where

$$g(b) := 1 + \frac{1-2b}{2}\exp(-b) - (b+1)\exp(-2b) - \frac{1}{2}\exp(-3b).$$

Then, a.a.s. no strategy can create a perfect matching in less than $(\alpha + o(1))n \geq 0.93261n$ rounds.
(Gao, MacRury, Prałat, SIDMA, 2022)

We again use the DE method, though we must restrict to "well-behaved" strategies.

Annoying issue: the player may put more than $\omega = \sqrt{n}$ circles on one vertex or create multi-edges.

Annoying issue: the player may put more than $\omega = \sqrt{n}$ circles on one vertex or create multi-edges.
It is clearly a suboptimal strategy but we cannot prevent the player from doing it.

Perfect Matchings: Lower Bounds

Annoying issue: the player may put more than $\omega = \sqrt{n}$ circles on one vertex or create multi-edges.
It is clearly a suboptimal strategy but we cannot prevent the player from doing it.

Solution: We offer a deal the player will gladly accept:

Perfect Matchings: Lower Bounds

Annoying issue: the player may put more than $\omega = \sqrt{n}$ circles on one vertex or create multi-edges.
It is clearly a suboptimal strategy but we cannot prevent the player from doing it.

Solution: We offer a deal the player will gladly accept:
– Put at most 2ω circles on one vertex.

Annoying issue: the player may put more than $\omega = \sqrt{n}$ circles on one vertex or create multi-edges.
It is clearly a suboptimal strategy but we cannot prevent the player from doing it.

Solution: We offer a deal the player will gladly accept:
– Put at most 2ω circles on one vertex.
– Create a matching consisting of $n/2 - n/\omega$ edges in at most n rounds.

Perfect Matchings: Lower Bounds

Annoying issue: the player may put more than $\omega = \sqrt{n}$ circles on one vertex or create multi-edges.
It is clearly a suboptimal strategy but we cannot prevent the player from doing it.

Solution: We offer a deal the player will gladly accept:
– Put at most 2ω circles on one vertex.
– Create a matching consisting of $n/2 - n/\omega$ edges in at most n rounds.
– Never create multi-edges.

Perfect Matchings: Lower Bounds

$X(t)$: the number of vertices with at least one square at time t.

Perfect Matchings: Lower Bounds

$X(t)$: the number of vertices with at least one square at time t.

A.a.s. $X(t) = (1 + o(1))n(1 - e^{-t/n})$.

Perfect Matchings: Lower Bounds

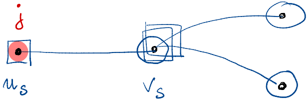
$X(t)$: the number of vertices with at least one square at time t.

A.a.s. $X(t) = (1 + o(1))n(1 - e^{-t/n})$.

Vertex j is redundant at time $t \geq 0$ if:

– j is covered by precisely one square, say u_s for $s \leq t$,

– circle v_s connected to u_s by the player is covered by at least one square, which arrives after round s.

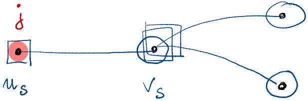


Perfect Matchings: Lower Bounds

$X(t)$: the number of vertices with at least one square at time t.

A.a.s. $X(t) = (1 + o(1))n(1 - e^{-t/n})$.

Vertex j is redundant at time $t \geq 0$ if:
– j is covered by precisely one square, say u_s for $s \leq t$,
– circle v_s connected to u_s by the player is covered by at least one square, which arrives after round s.

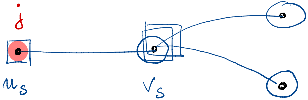


$U(t)$: the number of redundant vertices at time t.

$X(t)$: the number of vertices with at least one square at time t.

A.a.s. $X(t) = (1 + o(1))n(1 - e^{-t/n})$.

Vertex j is redundant at time $t \geq 0$ if:
– j is covered by precisely one square, say u_s for $s \leq t$,
– circle v_s connected to u_s by the player is covered by at least one square, which arrives after round s.



$U(t)$: the number of redundant vertices at time t.

$U(t)$ depends only on the placement of the squares, not the strategy.

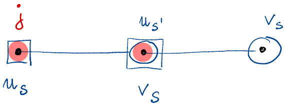
Perfect Matchings: Lower Bounds

Suppose that j is redundant at time t thanks to the arrival of the square u_s at time $s \leq t$.

Perfect Matchings: Lower Bounds

Suppose that j is redundant at time t thanks to the arrival of the square u_s at time $s \leq t$.

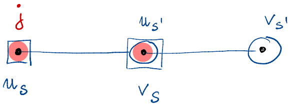
j is well-positioned, if v_s is also redundant at time t.



Perfect Matchings: Lower Bounds

Suppose that j is redundant at time t thanks to the arrival of the square u_s at time $s \le t$.

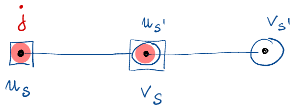
j is well-positioned, if v_s is also redundant at time t.



$W(t)$: the number of well-positioned redundant vertices at time t. (Clearly, $W(t) \le U(t)$.)

Perfect Matchings: Lower Bounds

Suppose that j is redundant at time t thanks to the arrival of the square u_s at time $s \le t$.
j is well-positioned, if v_s is also redundant at time t.



$W(t)$: the number of well-positioned redundant vertices at time t. (Clearly, $W(t) \le U(t)$.)

To get the lower bound, we use the following inequality:

$$X(t) - U(t) + W(t) \ge \frac{n}{2} - \frac{3t}{\omega},$$

and the DE method of Nick Wormald.

Hamilton Cycles

Hamilton Cycles: Upper Bounds

A.a.s. k-out process has a Hamilton cycle when $k \geq 3$.
(Bohman, Frieze, RSA, 2009)

Hamilton Cycles: Upper Bounds

A.a.s. *k-out process* has a Hamilton cycle when $k \geq 3$.
(Bohman, Frieze, RSA, 2009)

Implication: a.a.s. there exists a strategy to create a perfect matching in $(3 + o(1))n$ rounds.

Hamilton Cycles: Upper Bounds

A.a.s. k-out process has a Hamilton cycle when $k \geq 3$.
(Bohman, Frieze, RSA, 2009)
Implication: a.a.s. there exists a strategy to create a perfect matching in $(3 + o(1))n$ rounds.

There exists a strategy to create a Hamilton cycle in βn rounds a.a.s., where β is the result of a high dimensional optimization problem. Numerical computations indicate that $\beta < 2.61135$.
(Gao, Kamiński, MacRury, Prałat, Euro. J. of Comb., 2022)

Hamilton Cycles: Upper Bounds

A.a.s. k-out process has a Hamilton cycle when $k \geq 3$.
(Bohman, Frieze, RSA, 2009)
Implication: a.a.s. there exists a strategy to create a perfect matching in $(3 + o(1))n$ rounds.

There exists a strategy to create a Hamilton cycle in βn rounds a.a.s., where β is the result of a high dimensional optimization problem. Numerical computations indicate that $\beta < 2.61135$.
(Gao, Kamiński, MacRury, Prałat, Euro. J. of Comb., 2022)

We recently analysed a fully adaptive greedy augmentation algorithm to attain an upper bound of $2.01678n$.
(Gao, MacRury, Prałat, RANDOM 2022, 2022)

Hamilton Cycles: Upper Bounds

A.a.s. k-out process has a Hamilton cycle when $k \geq 3$.
(Bohman, Frieze, RSA, 2009)
Implication: a.a.s. there exists a strategy to create a perfect matching in $(3 + o(1))n$ rounds.

There exists a strategy to create a Hamilton cycle in βn rounds a.a.s., where β is the result of a high dimensional optimization problem. Numerical computations indicate that $\beta < 2.61135$.
(Gao, Kamiński, MacRury, Prałat, Euro. J. of Comb., 2022)

We recently analysed a fully adaptive greedy augmentation algorithm to attain an upper bound of $2.01678n$.
(Gao, MacRury, Prałat, RANDOM 2022, 2022)

One more trick brings it down to to $1.81696n$.
(Gao, Frieze, MacRury, Prałat, Sorkin, 2023+)

Hamilton Cycles: Lower Bounds

Obvious necessary condition: the graph has the minimum degree at least 2.

Hamilton Cycles: Lower Bounds

Obvious necessary condition: the graph has the minimum degree at least 2.

A.a.s. no strategy can create a Hamilton cycle in less than $(\ln 2 + \ln(1 + \ln 2) + o(1))n \geq 1.21973n$ rounds.
(Ben-Eliezer, Hefetz, Kronenberg, Parczyk, Shikhelman, and Stojaković, RSA, 2020)

Hamilton Cycles: Lower Bounds

Obvious necessary condition: the graph has the minimum degree at least 2.

A.a.s. no strategy can create a Hamilton cycle in less than $(\ln 2 + \ln(1 + \ln 2) + o(1))n \geq 1.21973n$ rounds.
(Ben-Eliezer, Hefetz, Kronenberg, Parczyk, Shikhelman, and Stojaković, RSA, 2020)

A.a.s. no strategy can create a Hamilton cycle in less than $(\ln 2 + \ln(1 + \ln 2) + \varepsilon + o(1))n$ rounds for some universal constant $\varepsilon > 10^{-8}$.
(Gao, Kamiński, MacRury, Prałat, Euro. J. of Comb., 2022)

Hamilton Cycles: Lower Bounds

Obvious necessary condition: the graph has the minimum degree at least 2.

A.a.s. no strategy can create a Hamilton cycle in less than $(\ln 2 + \ln(1 + \ln 2) + o(1))n \geq 1.21973n$ rounds.
(Ben-Eliezer, Hefetz, Kronenberg, Parczyk, Shikhelman, and Stojaković, RSA, 2020)

A.a.s. no strategy can create a Hamilton cycle in less than $(\ln 2 + \ln(1 + \ln 2) + \varepsilon + o(1))n$ rounds for some universal constant $\varepsilon > 10^{-8}$.
(Gao, Kamiński, MacRury, Prałat, Euro. J. of Comb., 2022)

We recently improved this bound to $1.26575n$ using similar techniques as in the perfect matching problem.

Small Subgraphs

Small Subgraphs: Upper Bound

Let G be a fixed graph with degeneracy equal to $d \geq 2$.

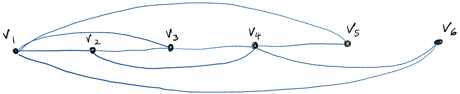
Small Subgraphs: Upper Bound

Let G be a fixed graph with degeneracy equal to $d \geq 2$.

A.a.s. there exists a strategy to create G in $n^{(d-1)/d}\omega$ rounds, where $\omega = \omega(n) \to \infty$ as $n \to \infty$.
(Ben-Eliezer, Hefetz, Kronenberg, Parczyk, Shikhelman, and Stojaković, RSA, 2020)

Let G be a fixed graph with degeneracy equal to $d \geq 2$.

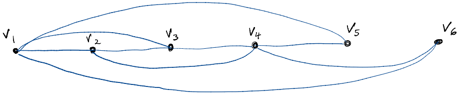
A.a.s. there exists a strategy to create G in $n^{(d-1)/d}\omega$ rounds, where $\omega = \omega(n) \to \infty$ as $n \to \infty$.
(Ben-Eliezer, Hefetz, Kronenberg, Parczyk, Shikhelman, and Stojaković, RSA, 2020)

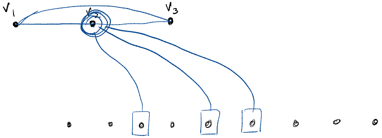


Small Subgraphs: Upper Bound

Let G be a fixed graph with degeneracy equal to $d \geq 2$.

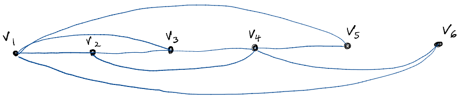
A.a.s. there exists a strategy to create G in $n^{(d-1)/d}\omega$ rounds, where $\omega = \omega(n) \to \infty$ as $n \to \infty$.
(Ben-Eliezer, Hefetz, Kronenberg, Parczyk, Shikhelman, and Stojaković, RSA, 2020)

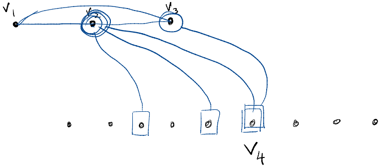




Let G be a fixed graph with degeneracy equal to $d \geq 2$.

A.a.s. there exists a strategy to create G in $n^{(d-1)/d}\omega$ rounds, where $\omega = \omega(n) \to \infty$ as $n \to \infty$.
(Ben-Eliezer, Hefetz, Kronenberg, Parczyk, Shikhelman, and Stojaković, RSA, 2020)





Small Subgraphs: Lower Bounds

Let G be a fixed graph with degeneracy equal to $d \geq 2$.

Conjecture: a.a.s. no strategy can create G in $n^{(d-1)/d}/\omega$ rounds, where $\omega = \omega(n) \to \infty$ as $n \to \infty$.

The conjecture is true for $G = K_{d+1}$.

(Ben-Eliezer, Hefetz, Kronenberg, Parczyk, Shikhelman, and Stojaković, RSA, 2020)

Let G be a fixed graph with degeneracy equal to $d \geq 2$.

Conjecture: a.a.s. no strategy can create G in $n^{(d-1)/d}/\omega$ rounds, where $\omega = \omega(n) \to \infty$ as $n \to \infty$.
The conjecture is true for $G = K_{d+1}$.
(Ben-Eliezer, Hefetz, Kronenberg, Parczyk, Shikhelman, and Stojaković, RSA, 2020)

The conjecture is true.
(Behague, Marbach, Prałat, Ruciński, ArXiv, 2021+)

Small Subgraphs: Lower Bounds

Let G be a fixed graph with degeneracy equal to $d \geq 2$.

Conjecture: a.a.s. no strategy can create G in $n^{(d-1)/d}/\omega$ rounds, where $\omega = \omega(n) \to \infty$ as $n \to \infty$.
The conjecture is true for $G = K_{d+1}$.
(Ben-Eliezer, Hefetz, Kronenberg, Parczyk, Shikhelman, and Stojaković, RSA, 2020)

The conjecture is true.
(Behague, Marbach, Prałat, Ruciński, ArXiv, 2021+)

The semi-random process can be generalized to hypergraphs and some results can be transferred.

Small Subgraphs: Lower Bounds

Let G be a fixed graph with degeneracy equal to $d \geq 2$.

Conjecture: a.a.s. no strategy can create G in $n^{(d-1)/d}/\omega$ rounds, where $\omega = \omega(n) \to \infty$ as $n \to \infty$.

The conjecture is true for $G = K_{d+1}$.

(Ben-Eliezer, Hefetz, Kronenberg, Parczyk, Shikhelman, and Stojaković, RSA, 2020)

The conjecture is true.

(Behague, Marbach, Prałat, Ruciński, ArXiv, 2021+)

The semi-random process can be generalized to hypergraphs and some results can be transferred. But some questions are still open, for example, $G = K_6^{(3)}$ and 2 squares (1 circle).

Open Problems

Open Problems

Recall that Ben-Eliezer, Gishboliner, Hefetz and Krivelevich (SODA, 2020) considered the general problem of constructing a copy of a spanning graph H of max-degree Δ.

Open Problems

Recall that Ben-Eliezer, Gishboliner, Hefetz and Krivelevich (SODA, 2020) considered the general problem of constructing a copy of a spanning graph H of max-degree Δ.

Specifically, they showed that a copy of H can be constructed in $\frac{3}{2}(\Delta + o(\Delta))n$ rounds a.a.s.

Open Problems

Recall that Ben-Eliezer, Gishboliner, Hefetz and Krivelevich (SODA, 2020) considered the general problem of constructing a copy of a spanning graph H of max-degree Δ.

Specifically, they showed that a copy of H can be constructed in $\frac{3}{2}(\Delta + o(\Delta))n$ rounds a.a.s.

The $o(\Delta)$ term prevents this result from yielding good bounds when Δ is constant.

Open Problems

Recall that Ben-Eliezer, Gishboliner, Hefetz and Krivelevich (SODA, 2020) considered the general problem of constructing a copy of a spanning graph H of max-degree Δ.

Specifically, they showed that a copy of H can be constructed in $\frac{3}{2}(\Delta + o(\Delta))n$ rounds a.a.s.

The $o(\Delta)$ term prevents this result from yielding good bounds when Δ is constant.

Compute a (small/explicit) universal constant $C > 0$ such that for any bounded degree spanning graph H, H can be constructed in $C \cdot \Delta$ rounds a.a.s.

Open Problems

Recall that Ben-Eliezer, Gishboliner, Hefetz and Krivelevich (SODA, 2020) considered the general problem of constructing a copy of a spanning graph H of max-degree Δ.

Specifically, they showed that a copy of H can be constructed in $\frac{3}{2}(\Delta + o(\Delta))n$ rounds a.a.s.

The $o(\Delta)$ term prevents this result from yielding good bounds when Δ is constant.

Compute a (small/explicit) universal constant $C > 0$ such that for any bounded degree spanning graph H, H can be constructed in $C \cdot \Delta$ rounds a.a.s.

A starting point may be to consider when some additional structure is assumed to hold on H – i.e., it is vertex transitive, or at least Δ-regular.

THE
END