On Aharoni-Berger's conjecture of rainbow matchings

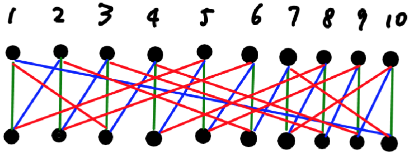
Jane Gao
Monash University

Discrete Mathematics Seminar 2018

Joint work with Reshma Ramadurai, Ian Wanless and Nick Wormald

Ryser-Brualdi-Stein Conjecture

1	2	3	4	5	6	7	8	9	10
2	3	4	5	6	7	8	9	10	1
5	6	1	8	9	2	10	3	4	7



Latin retangle $\Longleftrightarrow$ simple bipartite, union of PMs.

Ryser-Brualdi-Stein Conjecture

Conjecture (Ryser-Brualdi-Stein)

An $n \times n$ Latin square contains a partial transversal of size $n-1$. If n is odd, there exists a full transversal.

Aharoni-Berger Conjecture

Conjecture (Ryser-Brualdi-Stein)

An $n \times n$ Latin square contains a partial transversal of size $n-1$. If n is odd, there exists a full transversal.

A stronger version:

Conjecture (Aharoni-Berger 09)

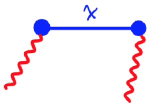
If G is a bipartite multigraph as the union of $n-1$ matchings in G, each of size n. Then G contains a full rainbow matching.

The general graph case

Conjecture (Aharoni, Berger, Chudnovsky, Howard and Seymour 16)

If G is a general graph as the union of $n-2$ matchings each of size n, then G contains a full rainbow matching.

A trivial lower bound



$|M| \to |M|-2$

If $|M| \le \frac{n}{2}$, then

M contains a full rainbow matching.

State of art

- Partial transversal in Latin square:
  - $(2n+1)/3$ – Koksma (1969);
  - $(3/4)n$ – Drake (1977);
  - $n-\sqrt{n}$ – Brouwer et al. (1978) and independently by Woolbright (1978.)
  - $n-O(\log^2 n)$ – Shor (1982).
- Full rainbow matching in bipartite (multi)graphs.
  - $n-o(n)$ (Latin rectangle) – Haggkvist and Johansson (2008).
  - $(4/7)n$ – Aharoni Charbit and Howard (2015).
  - $(3/5)n$ – Kotlar and Ziv (2014).
  - $(2/3)n+o(n)$ – Clemens and Ehrenmüller (2016).
  - $(2n-1)/3$ – Aharoni, Kotlar and Ziv (arXiv).
  - $n-o(n)$ – Pokrovskiy (arXiv).

Our results

Theorem (G., Ramadurai, Wanless, Wormald 2017+)

If G is a general graph and $|\mathcal{M}| \leq n - n^c$, where $c > 9/10$. Then $\mathcal{M}$ contains a full rainbow matching.

Theorem (G., Ramadurai, Wanless, Wormald 2017+)

- *Larger $|\mathcal{M}|$ if $\Delta(G)$ is smaller than n.*
- *Multigraph G with low multiplicity.*
- *Hypergraphs where no two vertices are contained in too many hyperedges.*

Keevash and Yepremyan (2017) — If G is an n-edge-coloured multigraph with low multiplicity, and each colour class contains $(1+\epsilon)n$ edges, then there is a partial rainbow matching of size $n - O(1)$.

Our results

Theorem (G., Ramadurai, Wanless, Wormald 2017+)

If G is a general graph and $|\mathcal{M}| \leq n - n^c$, where $c > 9/10$. Then $\mathcal{M}$ contains a full rainbow matching.

Theorem (G., Ramadurai, Wanless, Wormald 2017+)

- *Larger $|\mathcal{M}|$ if $\Delta(G)$ is smaller than n.*
- *Multigraph G with low multiplicity.*
- *Hypergraphs where no two vertices are contained in too many hyperedges.*

Keevash and Yepremyan (2017) — If G is an n-edge-coloured multigraph with low multiplicity, and each colour class contains $(1+\epsilon)n$ edges, then there is a partial rainbow matching of size $n - O(1)$.

Our results

Theorem (G., Ramadurai, Wanless, Wormald 2017+)

If G is a general graph and $|\mathcal{M}| \leq n - n^c$, where $c > 9/10$. Then $\mathcal{M}$ contains a full rainbow matching.

Theorem (G., Ramadurai, Wanless, Wormald 2017+)

- *Larger $|\mathcal{M}|$ if $\Delta(G)$ is smaller than n.*
- *Multigraph G with low multiplicity.*
- *Hypergraphs where no two vertices are contained in too many hyperedges.*

Keevash and Yepremyan (2017) — If G is an n-edge-coloured multigraph with low multiplicity, and each colour class contains $(1+\epsilon)n$ edges, then there is a partial rainbow matching of size $n - O(1)$.

Randomised algorithm and the DE method

Intuitively...

- Take a surviving matching x, take a random edge in x and put it to the rainbow matching;
- Modify the remaining graph;
- Repeat.

A randomised algorithm induces a sequence of random variables $Z_0, Z_1, Z_2, \ldots$.

Randomised algorithm and the DE method

Intuitively...

- Take a surviving matching x, take a random edge in x and put it to the rainbow matching;
- Modify the remaining graph;
- Repeat.

A randomised algorithm induces a sequence of random variables $Z_0, Z_1, Z_2, \ldots$.

Randomised algorithm and the DE method

Intuitively...

- Take a surviving matching x, take a random edge in x and put it to the rainbow matching;
- Modify the remaining graph;
- Repeat.

A randomised algorithm induces a sequence of random variables $Z_0, Z_1, Z_2, \ldots$.

Randomised algorithm and the DE method

Intuitively...

- Take a surviving matching x, take a random edge in x and put it to the rainbow matching;
- Modify the remaining graph;
- Repeat.

A randomised algorithm induces a sequence of random variables $Z_0, Z_1, Z_2, \ldots$.

Randomised algorithm and the DE method

Intuitively...

- Take a surviving matching x, take a random edge in x and put it to the rainbow matching;
- Modify the remaining graph;
- Repeat.

A randomised algorithm induces a sequence of random variables $Z_0, Z_1, Z_2, \ldots$.

Randomised algorithm and the DE method

Suppose

$$\mathbf{E}(Z_{t+1} - Z_t \mid \text{history}) = f(Z_t/n) + \text{small error}.$$

Then if we know a priori that $Z_t/n \approx z(x)$ where $x = t/n$ then

$$\frac{dz}{dx} = f(x).$$

The DE method guarantees that $Z_t = z(t/n)n + \text{small error}$, provided

- Z_0 lies inside a "nice" open set;
- f is "nice" in that open set;
- $|Z_{t+1} - Z_t|$ is not too big.

Randomised algorithm and the DE method

Suppose

$$\mathbf{E}(Z_{t+1} - Z_t \mid \text{history}) = f(Z_t/n) + \text{small error}.$$

Then if we know a priori that $Z_t/n \approx z(x)$ where $x = t/n$ then

$$\frac{dz}{dx} = f(x).$$

The DE method guarantees that $Z_t = z(t/n)n + \text{small error}$, provided

- Z_0 lies inside a "nice" open set;
- f is "nice" in that open set;
- $|Z_{t+1} - Z_t|$ is not too big.

DE method hard to apply for the rainbow matching problem

Suppose

$$\mathbf{E}(Z_{t+1} - Z_t \mid \text{history}) = f(Z_t/n) + \text{small error},$$

- Overlap of M_i and M_j ($|V(M_i) \cap V(M_j)|$) may be non-uniformly initially;
- The overlaps change in the process.

DE method hard to apply for the rainbow matching problem

Suppose

$$\mathbf{E}(Z_{t+1} - Z_t \mid \text{history}) = f(Z_t/n) + \text{small error},$$

- Overlap of M_i and M_j ($|V(M_i) \cap V(M_j)|$) may be non-uniformly initially;
- The overlaps change in the process.

DE method hard to apply for the rainbow matching problem

Suppose

$$\mathbf{E}(Z_{t+1} - Z_t \mid \text{history}) = f(Z_t/n) + \text{small error},$$

- Overlap of M_i and M_j ($|V(M_i) \cap V(M_j)|$) may be non-uniformly initially;
- The overlaps change in the process.

Rödl nibble

Randomly partition matchings in $\mathcal{M}$ into chunks, each chunk containing ϵn matchings. In iteration i, matchings in chunk i are processed.

In iteration i,

- For every matching in chunk i, randomly pick an edge x;
- "Artificially zap" each remaining vertex with a proper probability;
- Deal with vertex collisions.

Rödl nibble

Randomly partition matchings in $\mathcal{M}$ into chunks, each chunk containing ϵn matchings. In iteration i, matchings in chunk i are processed.
In iteration i,

- For every matching in chunk i, randomly pick an edge x;
- "Artificially zap" each remaining vertex with a proper probability;
- Deal with vertex collisions.

Rödl nibble

Randomly partition matchings in $\mathcal{M}$ into chunks, each chunk containing ϵn matchings. In iteration i, matchings in chunk i are processed.
In iteration i,

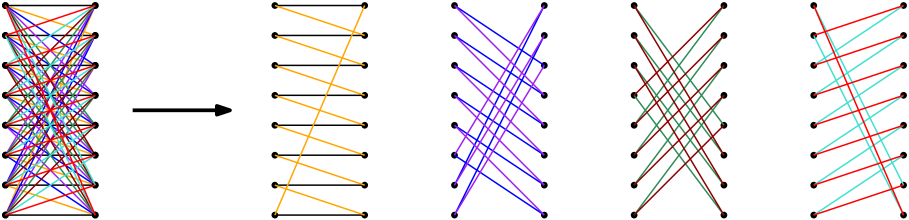
- For every matching in chunk i, randomly pick an edge x;
- "Artificially zap" each remaining vertex with a proper probability;
- Deal with vertex collisions.

Rödl nibble

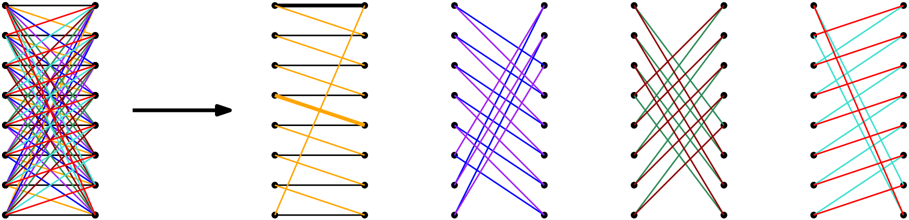
Randomly partition matchings in $\mathcal{M}$ into chunks, each chunk containing ϵn matchings. In iteration i, matchings in chunk i are processed.
In iteration i,

- For every matching in chunk i, randomly pick an edge x;
- "Artificially zap" each remaining vertex with a proper probability;
- Deal with vertex collisions.

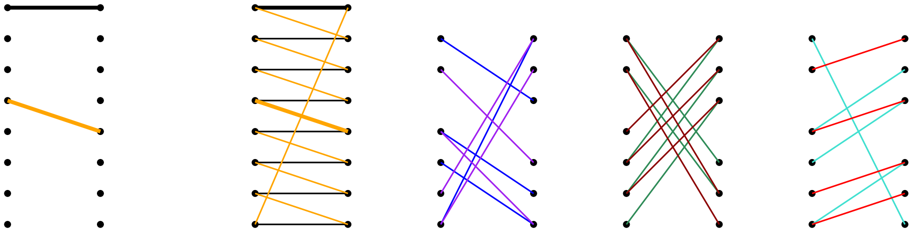
A randomised algorithm



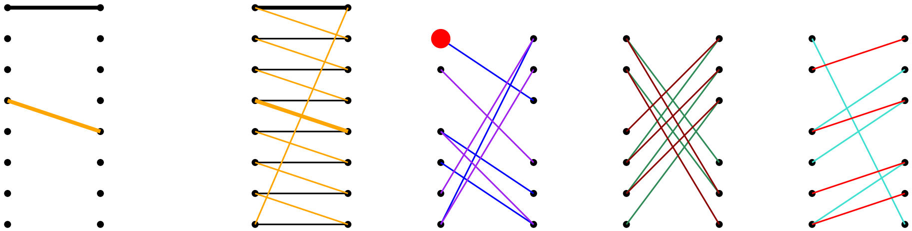
A randomised algorithm



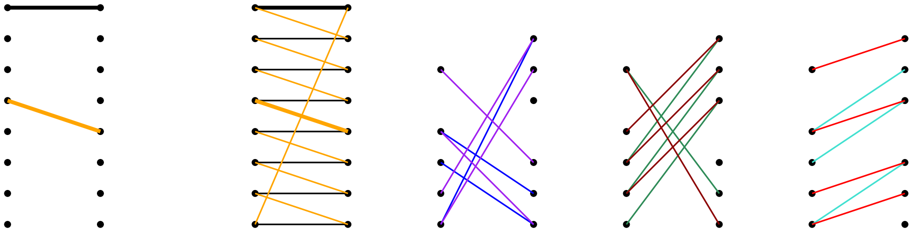
A randomised algorithm



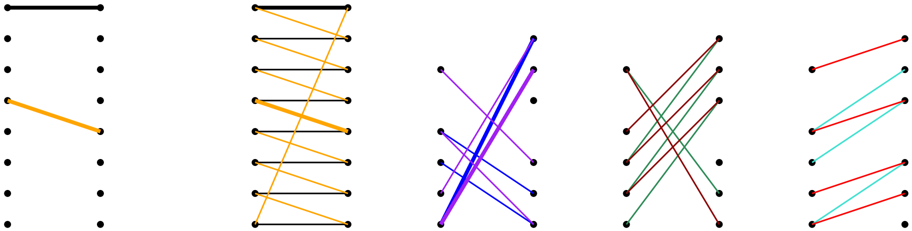
A randomised algorithm



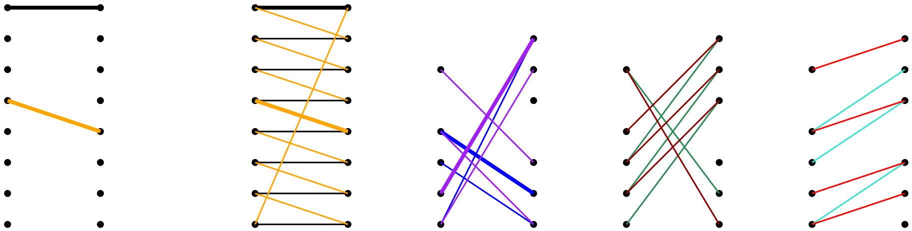
A randomised algorithm



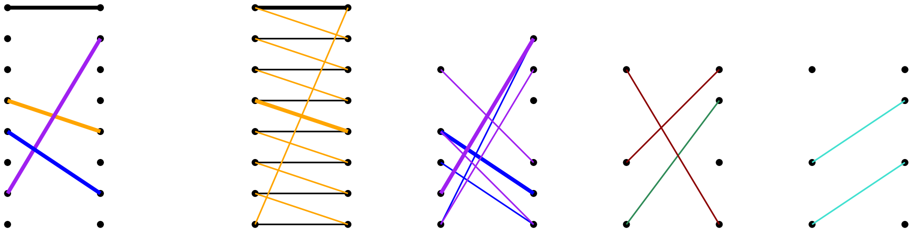
A randomised algorithm



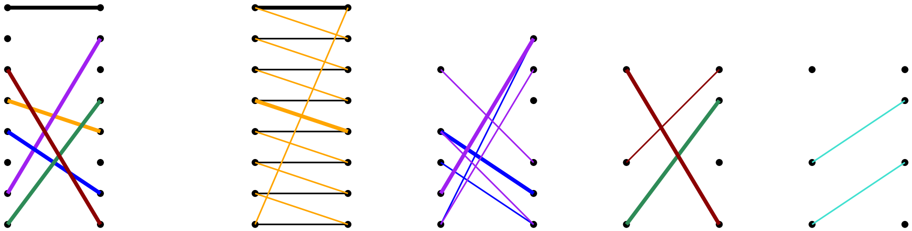
A randomised algorithm



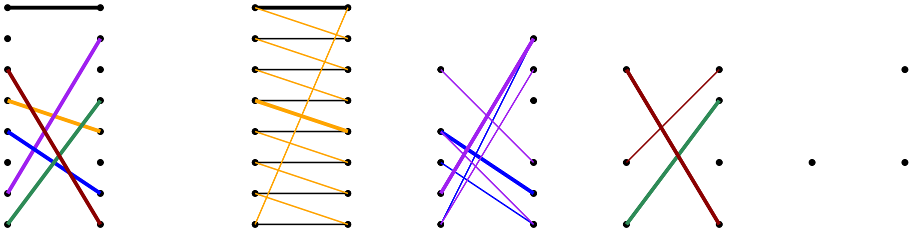
A randomised algorithm



A randomised algorithm



A randomised algorithm



How to zap vertices?

$$P(x \text{ is chosen}) = \frac{1}{|M|}$$

$$P(u \text{ is ``killed''}) \lessapprox \frac{d_u^{(i)}(i-1)}{|M|} \leq \frac{\max_u d_u^{(i)}(i-1)}{|M|}$$

Matching size and vertex degree

Every vertex is deleted with equal probability $\Rrightarrow$

- every surviving matchings are of approximately equal size

Matching size and vertex degree

Every vertex is deleted with equal probability $\Rightarrow$

- every surviving matchings are of approximately equal size $|M(i)| \approx r(x_i)n$

Matching size and vertex degree

Every vertex is deleted with equal probability $\Rrightarrow$

- every surviving matchings are of approximately equal size $|M(i)| \approx r(x_i)n$
- degrees of each vertex decrease with approximately equal rate

Matching size and vertex degree

Every vertex is deleted with equal probability $\Rrightarrow$

- every surviving matchings are of approximately equal size $|M(i)| \approx r(x_i)n$
- degrees of each vertex decrease with approximately equal rate $d_v^{(j)}(i) \approx \epsilon d_v g(x_i)$

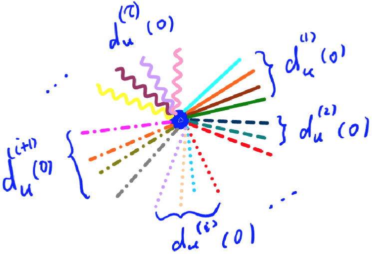
Matching size and vertex degree

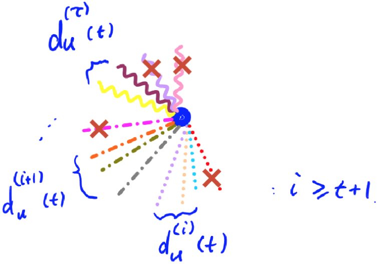
Every vertex is deleted with equal probability $\Rrightarrow$

- every surviving matchings are of approximately equal size $|M(i)| \approx r(x_i)n$
- degrees of each vertex decrease with approximately equal rate $d_v^{(j)}(i) \approx \epsilon d_v g(x_i)$

here $x_i = i\epsilon$.

Matching size and vertex degree





G_t: the graph obtained after t iterations

Probability of vertex deletion

$$\begin{aligned} d_v^{(j)}(i-1) &\approx \epsilon g(x_{i-1}) d_v \le \epsilon g(x_{i-1}) n \\ |M(i-1)| &\approx r(x_{i-1}) n \end{aligned}$$

$\Rightarrow$ Every vertex is deleted with probability roughly

$$\frac{\max_v \{d_v^{(j)}(i-1)\}}{|M(i-1)|} \le \frac{\epsilon g(x_{i-1})}{r(x_{i-1})} = f(x_{i-1}).$$

Probability of vertex deletion

$$\begin{aligned} d_v^{(j)}(i-1) &\approx \epsilon g(x_{i-1}) d_v \le \epsilon g(x_{i-1}) n \\ |M(i-1)| &\approx r(x_{i-1}) n \end{aligned}$$

$\Rrightarrow$ Every vertex is deleted with probability roughly

$$\frac{\max_v \{ d_v^{(j)}(i-1) \}}{|M(i-1)|} \le \frac{\epsilon g(x_{i-1})}{r(x_{i-1})} = f(x_{i-1}).$$

Deducing the ODEs

$$\begin{aligned}\mathbf{E}(|M(i)| - |M(i-1)|) &\approx -2f(x_{i-1})|M(i-1)|;\\ \mathbf{E}(d_v^j(i) - d_v^j(i-1)) &\approx -f(x_{i-1})d_v^j(i-1).\end{aligned}$$

Recall
$$\begin{aligned}f(x_{i-1}) &= \epsilon\frac{g(x_{i-1})}{r(x_{i-1})}\\ |M(i-1)| &\approx r(x_{i-1})n\\ d_v^j(i-1) &\approx \epsilon g(x_{i-1}).\end{aligned}$$

$\Longrightarrow$
$$\begin{aligned}r'(x) &= -2g(x);\\ g'(x) &= -\frac{g(x)^2}{r(x)}.\end{aligned}$$

Deducing the ODEs

$$\begin{aligned}\mathbf{E}(|M(i)| - |M(i-1)|) &\approx -2f(x_{i-1})|M(i-1)|;\\ \mathbf{E}(d_v^j(i) - d_v^j(i-1)) &\approx -f(x_{i-1})d_v^j(i-1).\end{aligned}$$

Recall

$$\begin{aligned}f(x_{i-1}) &= \epsilon\frac{g(x_{i-1})}{r(x_{i-1})}\\ |M(i-1)| &\approx r(x_{i-1})n\\ d_v^j(i-1) &\approx \epsilon g(x_{i-1}).\end{aligned}$$

$\Longrightarrow$

$$\begin{aligned}r'(x) &= -2g(x);\\ g'(x) &= -\frac{g(x)^2}{r(x)}.\end{aligned}$$

Deducing the ODEs

$$\begin{aligned} \mathbf{E}(|M(i)| - |M(i-1)|) &\approx -2f(x_{i-1})|M(i-1)|; \\ \mathbf{E}(d_v^j(i) - d_v^j(i-1)) &\approx -f(x_{i-1})d_v^j(i-1). \end{aligned}$$

Recall

$$\begin{aligned} f(x_{i-1}) &= \epsilon\frac{g(x_{i-1})}{r(x_{i-1})} \\ |M(i-1)| &\approx r(x_{i-1})n \\ d_v^j(i-1) &\approx \epsilon g(x_{i-1}). \end{aligned}$$

$\Rrightarrow$

$$\begin{aligned} r'(x) &= -2g(x); \\ g'(x) &= -\frac{g(x)^2}{r(x)}. \end{aligned}$$

ODE solution

The solution to the ODE with $r(0) = 1$ and $g(0) = 1$ is

$$r(x) = (1-x)^2, \quad g(x) = 1 - x.$$

Thus $r(x) > 0$ for all $x < 1$.

Let $\tau - 1 \approx (1-\epsilon_0)/\epsilon$ be the second last iteration of the algorithm. If $|M(i)| \approx r(x_i)n$ for every i, then $|M(\tau-1)| \approx r(1-\epsilon_0)n = \epsilon_0^2 n$.

If $\epsilon_0^2 n \geq (2+?)\epsilon n$ then we can process the last chunk of matchings greedily.

ODE solution

The solution to the ODE with $r(0) = 1$ and $g(0) = 1$ is

$$r(x) = (1-x)^2, \quad g(x) = 1-x.$$

Thus $r(x) > 0$ for all $x < 1$.

Let $\tau - 1 \approx (1-\epsilon_0)/\epsilon$ be the second last iteration of the algorithm. If $|M(i)| \approx r(x_i)n$ for every i, then $|M(\tau-1)| \approx r(1-\epsilon_0)n = \epsilon_0^2 n$.

If $\epsilon_0^2 n \geq (2+?)\epsilon n$ then we can process the last chunk of matchings greedily.

ODE solution

The solution to the ODE with $r(0) = 1$ and $g(0) = 1$ is

$$r(x) = (1-x)^2, \quad g(x) = 1-x.$$

Thus $r(x) > 0$ for all $x < 1$.

Let $\tau - 1 \approx (1-\epsilon_0)/\epsilon$ be the second last iteration of the algorithm. If $|M(i)| \approx r(x_i)n$ for every i, then $|M(\tau - 1)| \approx r(1-\epsilon_0)n = \epsilon_0^2 n$.

If $\epsilon_0^2 n \geq (2+?)\epsilon n$ then we can process the last chunk of matchings greedily.

Proof sketch

Next we sketch a proof for the following simpler version.

Theorem

For any $\epsilon_0 > 0$ there exists $N_0 > 0$ such that the following holds. If G is a simple graph and $|\mathcal{M}| \leq (1 - \epsilon_0)n$ where $n \geq N_0$, then $\mathcal{M}$ contains a full rainbow matching.

Proof sketch

Let $\epsilon > 0$ be sufficiently small so that $\epsilon_0^2 \geq 3\epsilon$. The matchings are then partitioned into $(1-\epsilon_0)/\epsilon$ chunks.

We will specify a_i, b_i such that

$$a_i = O(\epsilon n), \quad b_i = O(\epsilon^2 n)$$

and for iteration i $(0 \leq i \leq ((1-\epsilon_0)/\epsilon))$, with high probability,

$(A1)$ $|M(i)|$ is between $(1-i\epsilon)^2 n - a_i$ and $(1-i\epsilon)^2 n + a_i$;

$(A2)$ $d_v^{(j)}(i)$ is at most $\epsilon(1-i\epsilon)n + b_i$.

If (A1) and (A2) holds for every step, then by the beginning of the last iteration,

$$|M| = \epsilon_0^2 n + O(\epsilon n) \geq 2\epsilon n,$$

and there are at most ϵn matchings left. We can process the last chunk greedily.

Proof sketch

Let $\epsilon > 0$ be sufficiently small so that $\epsilon_0^2 \geq 3\epsilon$. The matchings are then partitioned into $(1-\epsilon_0)/\epsilon$ chunks.
We will specify a_i, b_i such that

$$a_i = O(\epsilon n), \quad b_i = O(\epsilon^2 n)$$

and for iteration i $(0 \leq i \leq ((1-\epsilon_0)/\epsilon))$, with high probability,

(A1) $|M(i)|$ is between $(1-i\epsilon)^2 n - a_i$ and $(1-i\epsilon)^2 n + a_i$;

(A2) $d_v^{(j)}(i)$ is at most $\epsilon(1-i\epsilon)n + b_i$.

If (A1) and (A2) holds for every step, then by the beginning of the last iteration,

$$|M| = \epsilon_0^2 n + O(\epsilon n) \geq 2\epsilon n,$$

and there are at most ϵn matchings left. We can process the last chunk greedily.

Proof sketch

Let $\epsilon > 0$ be sufficiently small so that $\epsilon_0^2 \geq 3\epsilon$. The matchings are then partitioned into $(1-\epsilon_0)/\epsilon$ chunks.
We will specify a_i, b_i such that

$$a_i = O(\epsilon n), \quad b_i = O(\epsilon^2 n)$$

and for iteration i $(0 \leq i \leq ((1-\epsilon_0)/\epsilon))$, with high probability,

$(A1)$ $|M(i)|$ is between $(1-i\epsilon)^2 n - a_i$ and $(1-i\epsilon)^2 n + a_i$;

$(A2)$ $d_v^{(j)}(i)$ is at most $\epsilon(1-i\epsilon)n + b_i$.

If (A1) and (A2) holds for every step, then by the beginning of the last iteration,

$$|M| = \epsilon_0^2 n + O(\epsilon n) \geq 2\epsilon n,$$

and there are at most ϵn matchings left. We can process the last chunk greedily.

Proof sketch

Base case $i = 0$:

$|M(0)| = n$ for all M. $\Rightarrow$ (A1)✔

$d_v(0) = \epsilon n + O(\sqrt{n}\log n)$ (standard concentration)

$\Rightarrow$ (A2) with $b_0 = O(\sqrt{n}\log n)$ ✔

Proof sketch

Base case $i = 0$:

$|M(0)| = n$ for all M. $\Rightarrow$ (A1)✔

$d_v(0) = \epsilon n + O(\sqrt{n}\log n)$ (standard concentration)

$\Rightarrow$ (A2) with $b_0 = O(\sqrt{n}\log n)$ ✔

Proof sketch

Base case $i = 0$:

$$|M(0)| = n \text{ for all } M. \Rightarrow \text{(A1)}✔$$

$$d_v(0) = \epsilon n + O(\sqrt{n}\log n) \text{ (standard concentration)}$$

$$\Rightarrow \text{(A2) with } b_0 = O(\sqrt{n}\log n) ✔$$

Proof sketch

Inductive step $i+1$:

Zap vertices so that every vertex is deleted with probability

$$\frac{\epsilon g(x_i)n+b_i}{r(x_i)n-a_i}=\frac{\epsilon}{1-i\epsilon}+O\left(\frac{\epsilon a_i}{r(x_i)n}+\frac{b_i}{r(x_i)n}\right).$$

Then, with high probability

$$\begin{aligned} d_v(i+1) &\leq d_v(i)-\frac{\epsilon g(x_i)n}{r(x_i)n}d_v(i)+O(\sqrt{n}\log n) \\ &\leq (\epsilon(1-i\epsilon)n+b_i)\left(1-\frac{\epsilon}{1-i\epsilon}\right)+O(\sqrt{n}\log n) \\ &\leq \epsilon(1-(i+1)\epsilon)n+b_i+O(\sqrt{n}\log n) \end{aligned}$$

$\Rrightarrow$ (A2) for iteration $i+1$ with $b_{i+1}=b_i+K(\sqrt{n}\log n)$. ✔
$\Rrightarrow$ $b_i=O((1/\epsilon)\sqrt{n}\log n)$ for all $0\leq i\leq\tau$. ✔

Proof sketch

Inductive step $i+1$:
Zap vertices so that every vertex is deleted with probability

$$\frac{\epsilon g(x_i)n + b_i}{r(x_i)n - a_i} = \frac{\epsilon}{1 - i\epsilon} + O\left(\frac{\epsilon a_i}{r(x_i)n} + \frac{b_i}{r(x_i)n}\right).$$

Then, with high probability

$$\begin{aligned} d_v(i+1) &\leq d_v(i) - \frac{\epsilon g(x_i)n}{r(x_i)n} d_v(i) + O(\sqrt{n}\log n) \\ &\leq (\epsilon(1-i\epsilon)n + b_i)\left(1 - \frac{\epsilon}{1-i\epsilon}\right) + O(\sqrt{n}\log n) \\ &\leq \epsilon(1-(i+1)\epsilon)n + b_i + O(\sqrt{n}\log n) \end{aligned}$$

$\Rightarrow$ (A2) for iteration $i+1$ with $b_{i+1} = b_i + K(\sqrt{n}\log n)$. ✔
$\Rightarrow$ $b_i = O((1/\epsilon)\sqrt{n}\log n)$ for all $0 \leq i \leq \tau$. ✔

Proof sketch

Inductive step $i+1$:
Zap vertices so that every vertex is deleted with probability

$$\frac{\epsilon g(x_i)n + b_i}{r(x_i)n - a_i} = \frac{\epsilon}{1-i\epsilon} + O\left(\frac{\epsilon a_i}{r(x_i)n} + \frac{b_i}{r(x_i)n}\right).$$

Then, with high probability

$$\begin{aligned} d_v(i+1) &\leq d_v(i) - \frac{\epsilon g(x_i)n}{r(x_i)n} d_v(i) + O(\sqrt{n}\log n) \\ &\leq (\epsilon(1-i\epsilon)n + b_i)\left(1 - \frac{\epsilon}{1-i\epsilon}\right) + O(\sqrt{n}\log n) \\ &\leq \epsilon(1-(i+1)\epsilon)n + b_i + O(\sqrt{n}\log n) \end{aligned}$$

$\Rightarrow$ (A2) for iteration $i+1$ with $b_{i+1} = b_i + K(\sqrt{n}\log n)$. ✔
$\Rightarrow$ $b_i = O((1/\epsilon)\sqrt{n}\log n)$ for all $0 \leq i \leq \tau$. ✔

Proof sketch

Inductive step $i+1$:
Zap vertices so that every vertex is deleted with probability

$$\frac{\epsilon g(x_i)n + b_i}{r(x_i)n - a_i} = \frac{\epsilon}{1-i\epsilon} + O\left(\frac{\epsilon a_i}{r(x_i)n} + \frac{b_i}{r(x_i)n}\right).$$

Then, with high probability

$$\begin{aligned} d_v(i+1) &\leq d_v(i) - \frac{\epsilon g(x_i)n}{r(x_i)n} d_v(i) + O(\sqrt{n}\log n) \\ &\leq (\epsilon(1-i\epsilon)n + b_i)\left(1 - \frac{\epsilon}{1-i\epsilon}\right) + O(\sqrt{n}\log n) \\ &\leq \epsilon(1-(i+1)\epsilon)n + b_i + O(\sqrt{n}\log n) \end{aligned}$$

$\Rightarrow$ (A2) for iteration $i+1$ with $b_{i+1} = b_i + K(\sqrt{n}\log n)$. ✔
$\Rightarrow$ $b_i = O((1/\epsilon)\sqrt{n}\log n)$ for all $0 \leq i \leq \tau$. ✔

Proof sketch

Inductive step $i+1$:
Zap vertices so that every vertex is deleted with probability

$$\frac{\epsilon g(x_i)n + b_i}{r(x_i)n - a_i} = \frac{\epsilon}{1 - i\epsilon} + O\left(\frac{\epsilon a_i}{r(x_i)n} + \frac{b_i}{r(x_i)n}\right).$$

Then, with high probability

$$\begin{aligned} d_v(i+1) &\leq d_v(i) - \frac{\epsilon g(x_i)n}{r(x_i)n} d_v(i) + O(\sqrt{n}\log n) \\ &\leq (\epsilon(1-i\epsilon)n + b_i)\left(1 - \frac{\epsilon}{1-i\epsilon}\right) + O(\sqrt{n}\log n) \\ &\leq \epsilon(1-(i+1)\epsilon)n + b_i + O(\sqrt{n}\log n) \end{aligned}$$

⇛ (A2) for iteration $i+1$ with $b_{i+1} = b_i + K(\sqrt{n}\log n)$. ✔
⇛ $b_i = O((1/\epsilon)\sqrt{n}\log n)$ for all $0 \leq i \leq \tau$. ✔

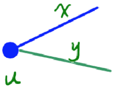
Proof sketch

With high probability

$$|M(i+1)| = |M(i)| - \left(\frac{2\epsilon}{1 - i\epsilon} + O\left(\frac{\epsilon a_i}{r(x_i)n} + \frac{b_i}{r(x_i)n} \right) \right) |M(i)| + O(\sqrt{n}\log n + \epsilon^2 n)$$

Proof sketch

$$P(u \text{ or } v \text{ is deleted}) = -2f(x_i) + O(\varepsilon^2)$$



$$\sum_{x\sim y} I_{xy} \le \sum_u \binom{d_u^{(i+1)}(i)}{2} \cdot \frac{1}{|M(i)|^2}$$

$$= O\left(\frac{\varepsilon g(x_i) n}{(r(x_i)n)^2} \sum_u d_u^{(i+1)}(i)\right)$$

$$= O\left(\frac{\varepsilon}{n} \cdot \varepsilon n \cdot n\right) = O(\varepsilon^2 n)$$

Proof sketch

With high probability

$$|M(i+1)| = |M(i)| - \left(\frac{2\epsilon}{1-i\epsilon} + O\left(\frac{\epsilon a_i}{r(x_i)n} + \frac{b_i}{r(x_i)n}\right)\right)|M(i)| + O(\sqrt{n}\log n + \epsilon^2 n)$$
$$= (1-(i+1)\epsilon)^2 n \pm a_i + O(\epsilon a_i + b_i + \epsilon^2 n).$$

$\Rightarrow$ (A1) for iteration $i+1$ with $a_{i+1} = (1+K\epsilon)a_i + K\epsilon^2 n$. ✔

$1/\epsilon$ iterations $\Rightarrow$ $a_i \le (1/\epsilon)(1+K\epsilon)^{1/\epsilon}K\epsilon^2 n = O(\epsilon n)$. ✔ □

Proof sketch

With high probability

$$|M(i+1)| = |M(i)| - \left(\frac{2\epsilon}{1-i\epsilon} + O\left(\frac{\epsilon a_i}{r(x_i)n} + \frac{b_i}{r(x_i)n}\right)\right)|M(i)|$$
$$+O(\sqrt{n}\log n + \epsilon^2 n)$$
$$= (1-(i+1)\epsilon)^2 n \pm a_i + O(\epsilon a_i + b_i + \epsilon^2 n).$$

$\Rrightarrow$ (A1) for iteration $i+1$ with $a_{i+1} = (1+K\epsilon)a_i + K\epsilon^2 n$. ✔

$1/\epsilon$ iterations $\Rrightarrow$ $a_i \le (1/\epsilon)(1+K\epsilon)^{1/\epsilon}K\epsilon^2 n = O(\epsilon n)$. ✔ □

Proof sketch

With high probability

$$
\begin{aligned}
|M(i+1)| &= |M(i)| - \left(\frac{2\epsilon}{1-i\epsilon} + O\left(\frac{\epsilon a_i}{r(x_i)n} + \frac{b_i}{r(x_i)n}\right)\right)|M(i)| \\
&\quad + O(\sqrt{n}\log n + \epsilon^2 n) \\
&= (1-(i+1)\epsilon)^2 n \pm a_i + O(\epsilon a_i + b_i + \epsilon^2 n).
\end{aligned}
$$

$\Rightarrow$ (A1) for iteration $i+1$ with $a_{i+1} = (1+K\epsilon)a_i + K\epsilon^2 n$. ✔

$1/\epsilon$ iterations $\Rightarrow$ $a_i \leq (1/\epsilon)(1+K\epsilon)^{1/\epsilon} K\epsilon^2 n = O(\epsilon n)$. ✔ □

Future directions

- How to cope with multigraphs?
- Transversal in high dimensional Latin cubes.