When can an FPT decision algorithm be used to count?

January 2016

Kitty Meeks

DECISION
Is there a witness?

DECISION
Is there a witness?

APPROX COUNTING
Approximately how many witnesses?

DECISION
Is there a witness?

APPROX COUNTING
Approximately how many witnesses?

EXACT COUNTING
Exactly how many witnesses?

Deciding, counting and enumerating

DECISION
Is there a witness?

EXTRACTION
Identify a single witness

APPROX COUNTING
Approximately how many witnesses?

EXACT COUNTING
Exactly how many witnesses?

Deciding, counting and enumerating

DECISION
Is there a witness?

EXTRACTION
Identify a single witness

APPROX COUNTING
Approximately how many witnesses?

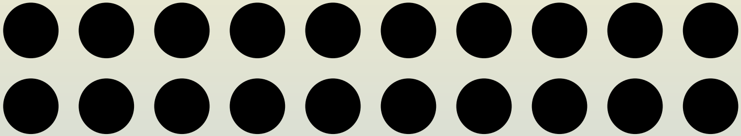
UNIFORM SAMPLING
Pick a single witness uniformly at random

EXACT COUNTING
Exactly how many witnesses?

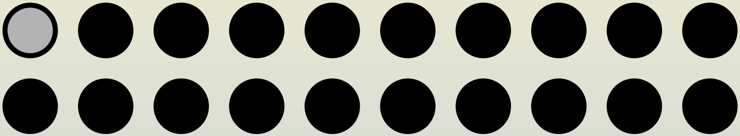
Deciding, counting and enumerating

DECISION Is there a witness?	**EXTRACTION** Identify a single witness
APPROX COUNTING Approximately how many witnesses?	**UNIFORM SAMPLING** Pick a single witness uniformly at random
EXACT COUNTING Exactly how many witnesses?	**ENUMERATION** List all witnesses

If we can decide, we can find a witness

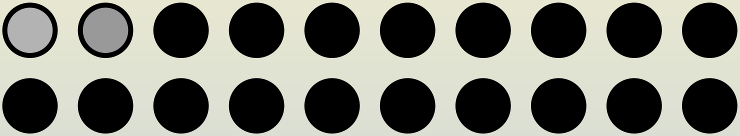


If we can decide, we can find a witness



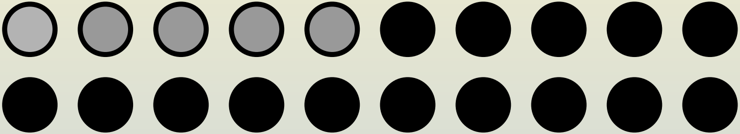
University of Glasgow

If we can decide, we can find a witness



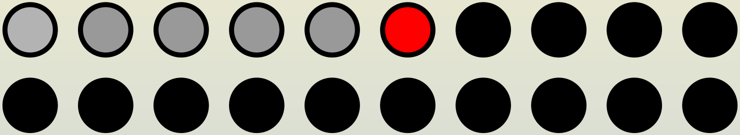
University of Glasgow

If we can decide, we can find a witness

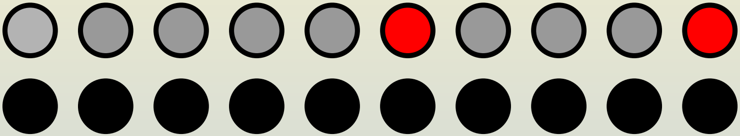


University of Glasgow

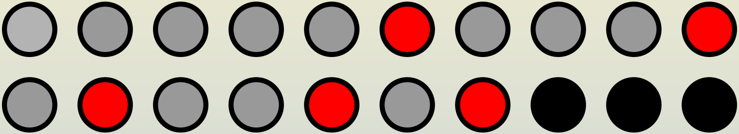
If we can decide, we can find a witness



If we can decide, we can find a witness



If we can decide, we can find a witness



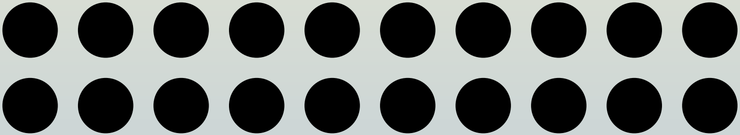
If we can decide, we can find a witness

Theorem (Björklund, Kaski and Kowalik, 2014)

There exists an algorithm that extracts a witness using at most

$$2k\left(\log_2 \frac{n}{k} + 2\right)$$

queries to a deterministic inclusion oracle.



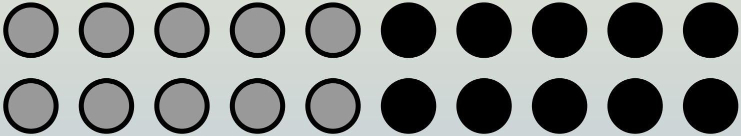
If we can decide, we can find a witness

Theorem (Björklund, Kaski and Kowalik, 2014)

There exists an algorithm that extracts a witness using at most

$$2k\left(\log_2 \frac{n}{k} + 2\right)$$

queries to a deterministic inclusion oracle.



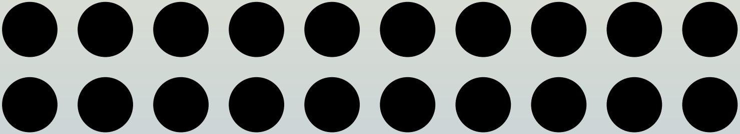
If we can decide, we can find a witness

Theorem (Björklund, Kaski and Kowalik, 2014)

There exists an algorithm that extracts a witness using at most

$$2k\left(\log_2 \frac{n}{k} + 2\right)$$

queries to a deterministic inclusion oracle.



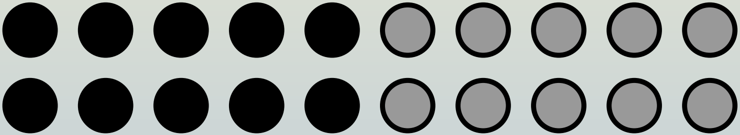
If we can decide, we can find a witness

Theorem (Björklund, Kaski and Kowalik, 2014)

There exists an algorithm that extracts a witness using at most

$$2k\left(\log_2 \frac{n}{k} + 2\right)$$

queries to a deterministic inclusion oracle.

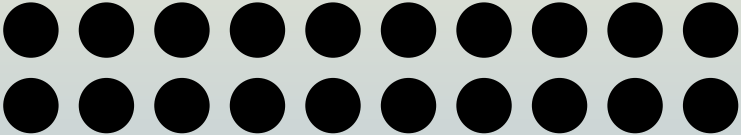


Theorem (Björklund, Kaski and Kowalik, 2014)

There exists an algorithm that extracts a witness using at most

$$2k\left(\log_2 \frac{n}{k} + 2\right)$$

queries to a deterministic inclusion oracle.



University of Glasgow

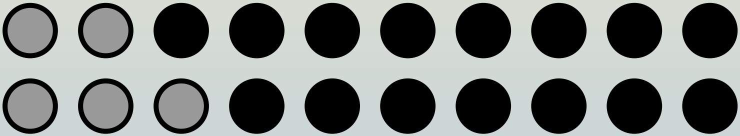
If we can decide, we can find a witness

Theorem (Björklund, Kaski and Kowalik, 2014)

There exists an algorithm that extracts a witness using at most

$$2k\left(\log_2 \frac{n}{k} + 2\right)$$

queries to a deterministic inclusion oracle.



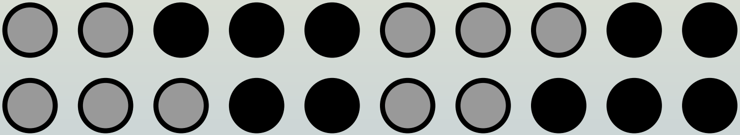
If we can decide, we can find a witness

Theorem (Björklund, Kaski and Kowalik, 2014)

There exists an algorithm that extracts a witness using at most

$$2k\left(\log_2 \frac{n}{k} + 2\right)$$

queries to a deterministic inclusion oracle.



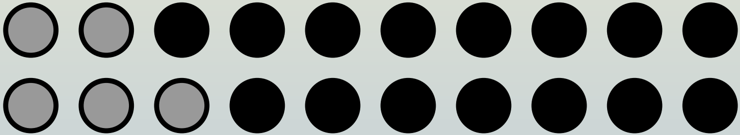
If we can decide, we can find a witness

Theorem (Björklund, Kaski and Kowalik, 2014)

There exists an algorithm that extracts a witness using at most

$$2k\left(\log_2 \frac{n}{k} + 2\right)$$

queries to a deterministic inclusion oracle.



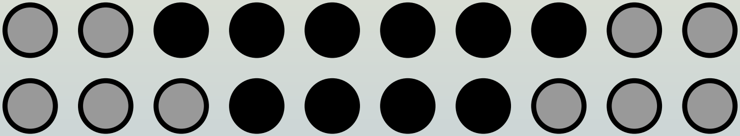
If we can decide, we can find a witness

Theorem (Björklund, Kaski and Kowalik, 2014)

There exists an algorithm that extracts a witness using at most

$$2k\left(\log_2 \frac{n}{k} + 2\right)$$

queries to a deterministic inclusion oracle.



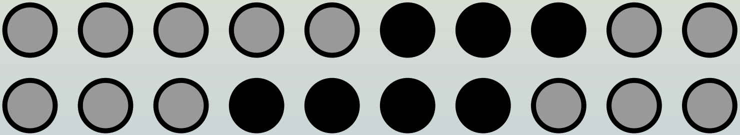
If we can decide, we can find a witness

Theorem (Björklund, Kaski and Kowalik, 2014)

There exists an algorithm that extracts a witness using at most

$$2k\left(\log_2 \frac{n}{k} + 2\right)$$

queries to a deterministic inclusion oracle.



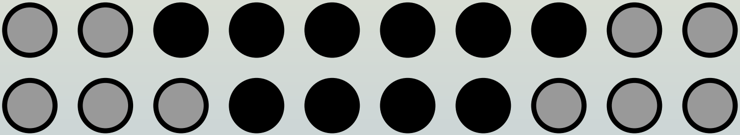
If we can decide, we can find a witness

Theorem (Björklund, Kaski and Kowalik, 2014)

There exists an algorithm that extracts a witness using at most

$$2k\left(\log_2 \frac{n}{k} + 2\right)$$

queries to a deterministic inclusion oracle.



University of Glasgow

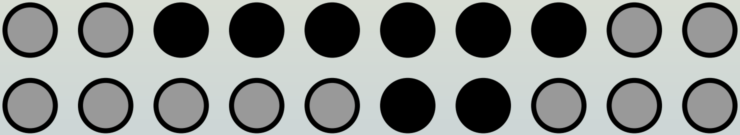
If we can decide, we can find a witness

Theorem (Björklund, Kaski and Kowalik, 2014)

There exists an algorithm that extracts a witness using at most

$$2k\left(\log_2 \frac{n}{k} + 2\right)$$

queries to a deterministic inclusion oracle.



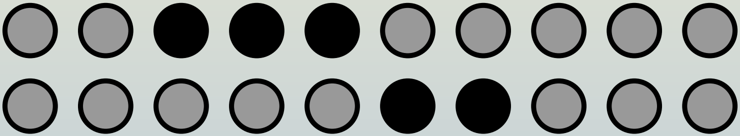
If we can decide, we can find a witness

Theorem (Björklund, Kaski and Kowalik, 2014)

There exists an algorithm that extracts a witness using at most

$$2k\left(\log_2 \frac{n}{k} + 2\right)$$

queries to a deterministic inclusion oracle.



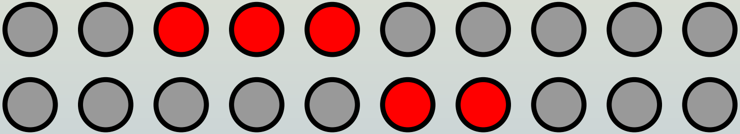
If we can decide, we can find a witness

Theorem (Björklund, Kaski and Kowalik, 2014)

There exists an algorithm that extracts a witness using at most

$$2k\left(\log_2 \frac{n}{k} + 2\right)$$

queries to a deterministic inclusion oracle.



If we can count approximately, we can decide

If we can count approximately, we can decide

... at least with high probability.

An FPRAS for a counting problem Π is a randomised approximation scheme that takes an instance I of Π (with $|I| = n$), and numbers $\epsilon > 0$ and $0 < \delta < 1$, and in time $poly(n, 1/\epsilon, \log(1/\delta))$ outputs a rational number z such that

$$\mathbb{P}[(1-\epsilon)\Pi(I) \leq z \leq (1+\epsilon)\Pi(I)] \geq 1 - \delta.$$

If we can count approximately, we can decide

... at least with high probability.

An FPRAS for a counting problem Π is a randomised approximation scheme that takes an instance I of Π (with $|I| = n$), and numbers $\epsilon > 0$ and $0 < \delta < 1$, and in time $poly(n, 1/\epsilon, \log(1/\delta))$ outputs a rational number z such that

$$\mathbb{P}[(1-\epsilon)\Pi(I) \leq z \leq (1+\epsilon)\Pi(I)] \geq 1 - \delta.$$

Set $\epsilon < \frac{1}{2}$, and we will distinguish between 0 and at least 1 with probability at least $1 - \delta$.

University of Glasgow

Uniform sampling is harder than finding a witness

GENCYCLE
Input: A directed graph G.
Output: A cycle selected uniformly, at random, from the set of all directed cycles of G.

Theorem (Jerrum, Valiant, Vazirani, 1986)

Suppose there exists a polynomial time bounded Probabilistic Turing Machine which solves the problem GENCYCLE. *Then NP = RP.*

A relation $R \subseteq \Sigma^* \times \Sigma^*$ is *self-reducible* if and only if:

- there exists a polynomial time computable function $g \in \Sigma^* \to \mathbb{N}$ such that $xRy \implies |y| = g(x)$;
- there exist polynomial time computable functions $\psi \in \Sigma^* \times \Sigma^* \to \Sigma^*$ and $\sigma \in \Sigma^* \to \mathbb{N}$ satisfying:
  - $\sigma(x) = O(\log |x|)$
  - $g(x) > 0 \implies \sigma(x) > 0 \quad \forall x \in \Sigma^*$
  - $|\psi(x, w)| \leq |x| \quad \forall x, w \in \Sigma^*$,

  and such that, for all $x \in \Sigma^*$, $y = y_1 \ldots y_n \in \Sigma^*$,

$$\langle x, y_1 \ldots y_n \rangle \in R \iff \langle \psi(x, y_1 \ldots y_{\sigma(x)}), y_{\sigma(x)+1} \ldots y_n \rangle \in R.$$

A relation $R \subseteq \Sigma^* \times \Sigma^*$ is *self-reducible* if and only if:

- there exists a polynomial time computable function $g \in \Sigma^* \to \mathbb{N}$ such that $xRy \implies |y| = g(x)$;
- there exist polynomial time computable functions $\psi \in \Sigma^* \times \Sigma^* \to \Sigma^*$ and $\sigma \in \Sigma^* \to \mathbb{N}$ satisfying:
  - $\sigma(x) = O(\log |x|)$
  - $g(x) > 0 \implies \sigma(x) > 0 \quad \forall x \in \Sigma^*$
  - $|\psi(x, w)| \leq |x| \quad \forall x, w \in \Sigma^*$,

  and such that, for all $x \in \Sigma^*$, $y = y_1 \ldots y_n \in \Sigma^*$,

$$\langle x, y_1 \ldots y_n \rangle \in R \iff \langle \psi(x, y_1 \ldots y_{\sigma(x)}), y_{\sigma(x)+1} \ldots y_n \rangle \in R.$$

Theorem (Jerrum, Valiant, Vazirani, 1986)

For self-reducible problems, approximate counting and almost-uniform sampling are polynomial-time inter-reducible.

Parameterised subgraph problems

Let Φ be a family $(\phi_1, \phi_2, \ldots)$ of functions, such that ϕ_k is a mapping from labelled graphs on k-vertices to $\{0, 1\}$.

p-INDUCED SUBGRAPH WITH PROPERTY(Φ) (**p-**ISWP(Φ))
Input: A graph $G = (V, E)$ and an integer k.
Parameter: k.
Question: Is there a tuple $(v_1, \ldots, v_k) \in V^k$ such that $v_1, \ldots, v_k$ are all distinct and $\phi_k(G[v_1, \ldots, v_k]) = 1$?

Parameterised subgraph problems

Let Φ be a family $(\phi_1, \phi_2, \ldots)$ of functions, such that ϕ_k is a mapping from labelled graphs on k-vertices to $\{0, 1\}$.

p-INDUCED SUBGRAPH WITH PROPERTY(Φ) (**p-**ISWP(Φ))
Input: A graph $G = (V, E)$ and an integer k.
Parameter: k.
Question: Is there a tuple $(v_1, \ldots, v_k) \in V^k$ such that $v_1, \ldots, v_k$ are all distinct and $\phi_k(G[v_1, \ldots, v_k]) = 1$?

p-MISWP(Φ)
Input: A graph $G = (V, E)$, an integer k and a colouring $f : V \to \{1, \ldots, k\}$.
Parameter: k.
Question: Is there a tuple $(v_1, \ldots, v_k) \in V^k$ such that $\{f(v_1), \ldots, f(v_k)\} = \{1, \ldots, k\}$ and $\phi_k(G[v_1, \ldots, v_k]) = 1$?

Parameterised subgraph problems

Let Φ be a family $(\phi_1, \phi_2, \ldots)$ of functions, such that ϕ_k is a mapping from labelled graphs on k-vertices to $\{0, 1\}$.

p-Induced Subgraph With Property(Φ) (**p-**ISWP(Φ))
Input: A graph $G = (V, E)$ and an integer k.
Parameter: k.
Question: Is there a tuple $(v_1, \ldots, v_k) \in V^k$ such that $v_1, \ldots, v_k$ are all distinct and $\phi_k(G[v_1, \ldots, v_k]) = 1$?

p-Ext-ISWP(Φ)
Input: A graph $G = (V, E)$, an integer k and subset $U \subset V$ of cardinality at most k.
Parameter: k.
Question: Is there a tuple $(v_1, \ldots, v_k) \in V^k$ such that $v_1, \ldots, v_k$ are all distinct, $U \subseteq \{v_1, \ldots, v_k\}$, and $\phi_k(G[v_1, \ldots, v_k]) = 1$?

Proposition

Suppose that ISWP(Φ) belongs to FPT. Then the following three statements are equivalent:

1. ISWP(Φ) *is self-reducible;*
2. MISWP(Φ) *belongs to FPT;*
3. EXT-ISWP(Φ) *belongs to FPT.*

Theorem (Arvind and Raman (2002); Jerrum and M. (2015); M. (2016))

*Suppose that Φ is a monotone property, and that **p-**$\mathrm{ISWP}(\Phi)$ is self-reducible. Then, if **p-**$\mathrm{ISWP}(\Phi)$ belongs to FPT, there is an FPTRAS for **p-#**$\mathrm{ISWP}(\Phi)$.*

Proposition

Suppose that, for each k and any graph G on n vertices, the number of k-vertex (labelled) subgraphs of G that satisfy ϕ_k is either

1. *zero, or*
2. *at least*

$$\frac{1}{g(k)p(n)}\binom{n}{k}.$$

Then there exists an FPTRAS for **p-#**$\mathrm{ISWP}(\Phi)$.

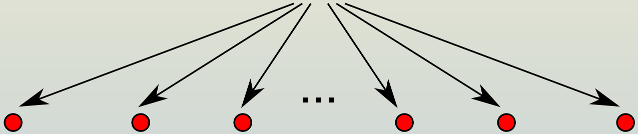
Few witnesses

Theorem

Suppose that $\boldsymbol{p}$-$\mathrm{MISWP}(\Phi)$ *belongs to FPT. Then we can enumerate (and hence count) all witnesses in time* $f(k) \cdot n^{O(1)} \cdot N$*, where* N *is the total number of witnesses.*

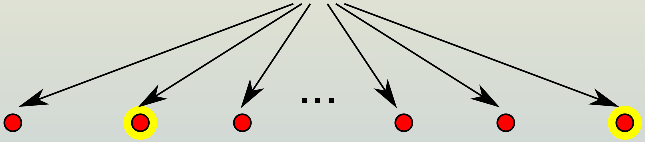
Theorem

Suppose that $\boldsymbol{p}$-$\mathrm{MISWP}(\Phi)$ *belongs to FPT. Then we can enumerate (and hence count) all witnesses in time* $f(k)\cdot n^{O(1)}\cdot N$*, where* N *is the total number of witnesses.*



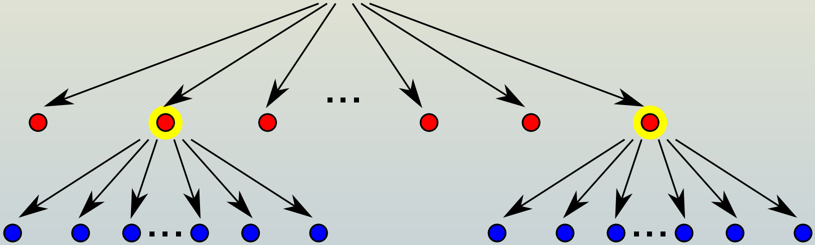
Theorem

Suppose that $\boldsymbol{p}$-$\mathrm{MISWP}(\Phi)$ *belongs to FPT. Then we can enumerate (and hence count) all witnesses in time* $f(k) \cdot n^{O(1)} \cdot N$*, where* N *is the total number of witnesses.*



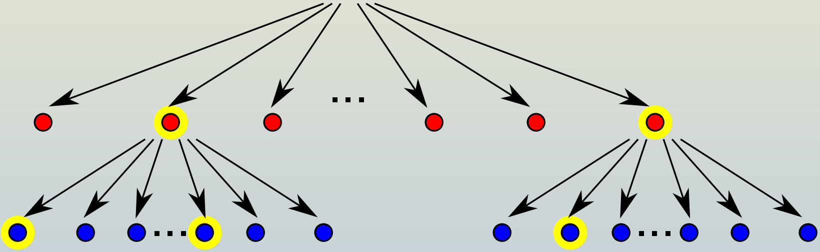
Theorem

Suppose that $\boldsymbol{p}$-$\mathrm{MISWP}(\Phi)$ *belongs to FPT. Then we can enumerate (and hence count) all witnesses in time* $f(k) \cdot n^{O(1)} \cdot N$*, where* N *is the total number of witnesses.*



Theorem

Suppose that $\textbf{\textit{p}}$-$\mathrm{MISWP}(\Phi)$ *belongs to FPT. Then we can enumerate (and hence count) all witnesses in time* $f(k) \cdot n^{O(1)} \cdot N$*, where* N *is the total number of witnesses.*



Not all subgraph problems are self-reducible

Let $\phi_k(H) = 1$ if and only if H is either a clique or an independent set.
Then:

Not all subgraph problems are self-reducible

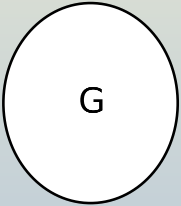
Let $\phi_k(H) = 1$ if and only if H is either a clique or an independent set. Then:

- **p-**$\mathrm{ISWP}(\Phi)$ is in FPT:
  - By Ramsey, for sufficiently large graphs the answer is always “yes”.

Not all subgraph problems are self-reducible

Let $\phi_k(H) = 1$ if and only if H is either a clique or an independent set.
Then:

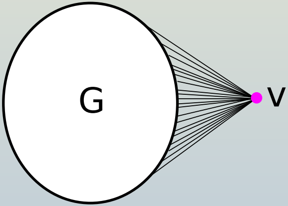
- **p-**ISWP(Φ) is in FPT:
  - By Ramsey, for sufficiently large graphs the answer is always “yes”.
- **p-**EXT-ISWP(Φ) is W[1]-complete:
  - Reduction from **p-**CLIQUE.



Not all subgraph problems are self-reducible

Let $\phi_k(H) = 1$ if and only if H is either a clique or an independent set. Then:

- **p-**ISWP(Φ) is in FPT:
  - By Ramsey, for sufficiently large graphs the answer is always "yes".
- **p-**EXT-ISWP(Φ) is W[1]-complete:
  - Reduction from **p-**CLIQUE.



Theorem (Alon, Yuster, Zwick, 1995)

For all $n, k \in \mathbb{N}$ there is a k-perfect family $\mathcal{F}_{n,k}$ of hash functions from $[n]$ to $[k]$ of cardinality $2^{O(k)} \cdot \log n$. Furthermore, given n and k, a representation of the family $\mathcal{F}_{n,k}$ can be computed in time $2^{O(k)} \cdot n \log n$.

Theorem (Alon, Yuster, Zwick, 1995)

For all $n, k \in \mathbb{N}$ there is a k-perfect family $\mathcal{F}_{n,k}$ of hash functions from $[n]$ to $[k]$ of cardinality $2^{O(k)} \cdot \log n$. Furthermore, given n and k, a representation of the family $\mathcal{F}_{n,k}$ can be computed in time $2^{O(k)} \cdot n \log n$.

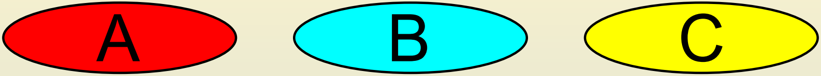
- IDEA: create many coloured instances, and enumerate the colourful copies in each (omitting duplicates)

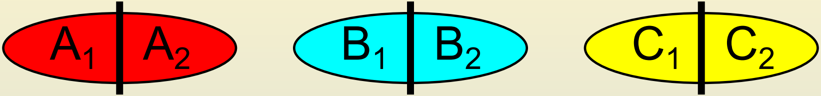
Theorem (Alon, Yuster, Zwick, 1995)

For all $n, k \in \mathbb{N}$ there is a k-perfect family $\mathcal{F}_{n,k}$ of hash functions from $[n]$ to $[k]$ of cardinality $2^{O(k)} \cdot \log n$. Furthermore, given n and k, a representation of the family $\mathcal{F}_{n,k}$ can be computed in time $2^{O(k)} \cdot n \log n$.

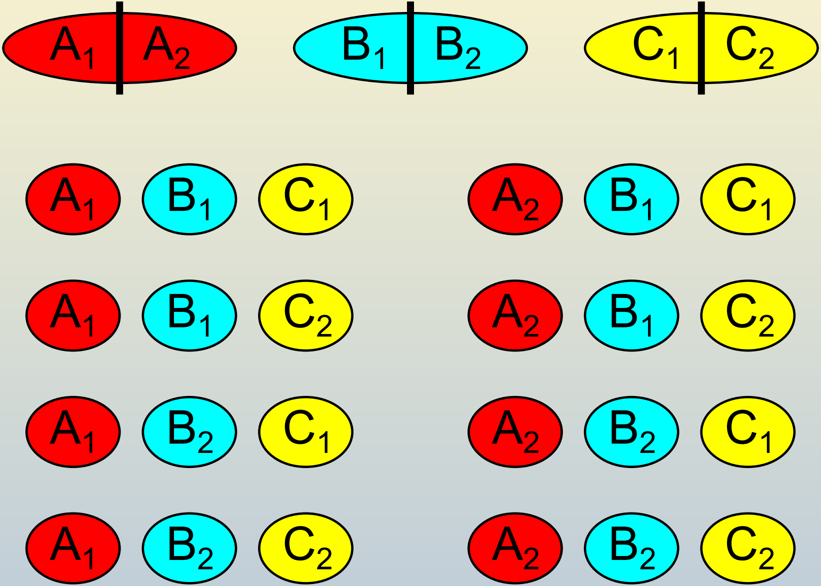
- IDEA: create many coloured instances, and enumerate the colourful copies in each (omitting duplicates)
- PROBLEM: although we're now looking for colourful witnesses, we still only have a decision oracle for the uncoloured version...

A randomised approach

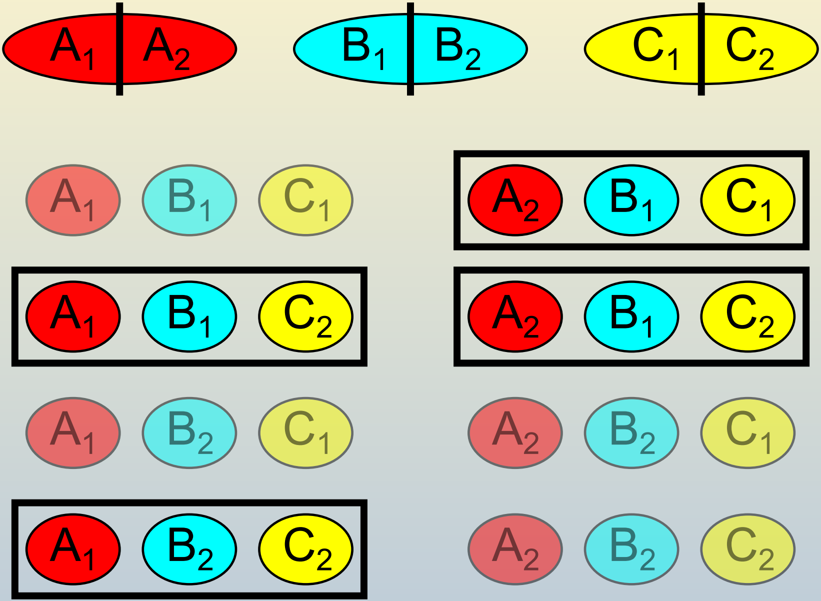



A_1
A_2
B_1
B_2
C_1
C_2

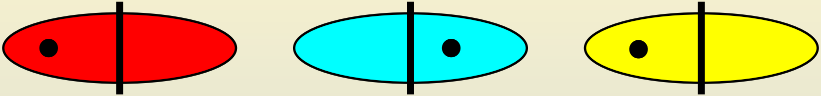
A randomised approach



A randomised approach



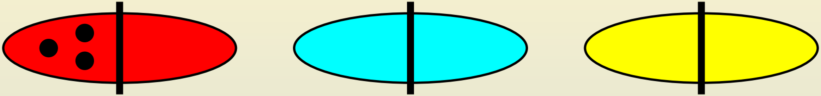
A randomised approach



If a witness is colourful:

- It will always survive in exactly one combination

A randomised approach

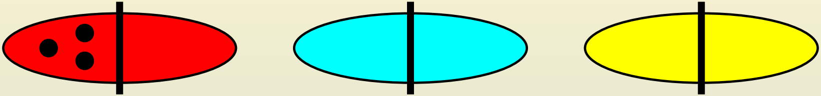


If a witness is colourful:

- It will always survive in exactly one combination

If a witness contains vertices of only $\ell < k$ colours:

- the probability it survives in at least one combination is at most $2^{-(k-\ell)}$
- if it survives in any combination, it will survive in exactly $2^{k-\ell}$ combinations



If a witness is colourful:

- It will always survive in exactly one combination

If a witness contains vertices of only $\ell < k$ colours:

- the probability it survives in at least one combination is at most $2^{-(k-\ell)}$
- if it survives in any combination, it will survive in exactly $2^{k-\ell}$ combinations

It can then be shown that, for **any** witness, the **expected** number of combinations in which it survives at each level is at most one.

Few witnesses, revisited

Theorem

Suppose that $\mathrm{ISWP}(\Phi)$ is in FPT. Then there is a randomised algorithm which enumerates all witnesses for $\mathrm{ISWP}(\Phi)$ in expected time $f(k) \cdot n^{O(1)} \cdot N$, where N is the total number of witnesses in the instance.

Theorem

Suppose that $\mathrm{ISWP}(\Phi)$ *is in FPT. Then there is a randomised algorithm which enumerates all witnesses for* $\mathrm{ISWP}(\Phi)$ *in expected time* $f(k) \cdot n^{O(1)} \cdot N$*, where* N *is the total number of witnesses in the instance.*

Corollary

Suppose that $\mathrm{ISWP}(\Phi)$ *is in FPT and that, for each* k *and any graph* G *on* n *vertices, the number of* k*-vertex (labelled) subgraphs of* G *that satisfy* ϕ_k *is at most* $f(k)n^{O(1)}$*. Then there exists an FPTRAS for* $\boldsymbol{p}\text{-}\mathrm{ISWP}(\Phi)$.

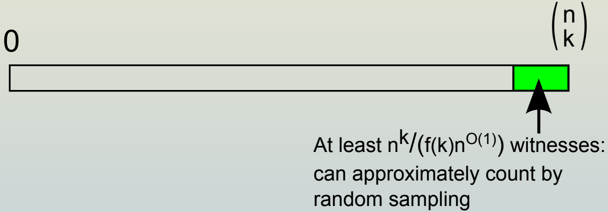
Open problems

- Can the randomised enumeration process be derandomised?

Open problems

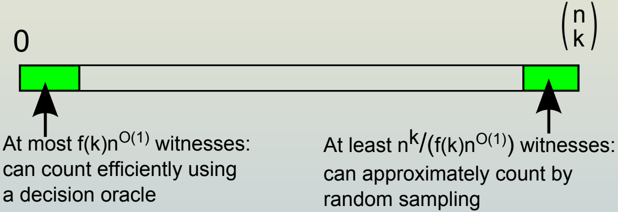
- Can the randomised enumeration process be derandomised?
- How common are non-self-reducible subgraph problems?

- Can the randomised enumeration process be derandomised?
- How common are non-self-reducible subgraph problems?
- Can we close the gap?



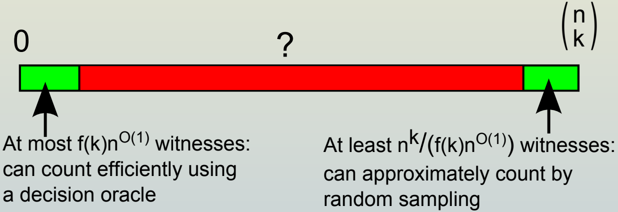
Open problems

- Can the randomised enumeration process be derandomised?
- How common are non-self-reducible subgraph problems?
- Can we close the gap?



Open problems

- Can the randomised enumeration process be derandomised?
- How common are non-self-reducible subgraph problems?
- Can we close the gap?



Thank you