# Randomised enumeration of small witnesses using a decision oracle

IPEC, Aarhus, 25th August 2016

Kitty Meeks

Many problems involve finding a *witness* $W$ (subset with some particular property) of size $k$ in a universe $U$ of size $n$.

Many problems involve finding a *witness* $W$ (subset with some particular property) of size $k$ in a universe $U$ of size $n$.

**ORA($X$)**
*Input:* $X \subseteq U$
*Output:* 1 if some witness is entirely contained in $X$; 0 otherwise.

## Witnesses and Oracles

Many problems involve finding a *witness* $W$ (subset with some particular property) of size $k$ in a universe $U$ of size $n$.

**ORA($X$)**
*Input:* $X \subseteq U$
*Output:* 1 if some witness is entirely contained in $X$; 0 otherwise.

In *self-contained k-witness problem*, we can obtain an oracle of this kind by calling a decision algorithm with universe $X$ rather than $U$ (so if $W \subseteq X \subseteq U$ then $W$ is a witness with respect to $X$ if and only if it is a witness with respect to $U$).

# Witnesses and Oracles

Many problems involve finding a *witness* $W$ (subset with some particular property) of size $k$ in a universe $U$ of size $n$.

**ORA($X$)**
*Input:* $X \subseteq U$
*Output:* 1 if some witness is entirely contained in $X$; 0 otherwise.

In *self-contained k-witness problem*, we can obtain an oracle of this kind by calling a decision algorithm with universe $X$ rather than $U$ (so if $W \subseteq X \subseteq U$ then $W$ is a witness with respect to $X$ if and only if it is a witness with respect to $U$).

**Examples**
$k$-CLIQUE
$k$-PATH

**Non-examples**
$k$-VERTEX COVER
$k$-DOMINATING SET

**DECISION**

Is there a witness?

**DECISION**
Is there a witness?

**APPROX COUNTING**
Approximately how many witnesses?

# Deciding, counting and enumerating

**DECISION**
Is there a witness?

**APPROX COUNTING**
Approximately how many witnesses?

**EXACT COUNTING**
Exactly how many witnesses?

University of Glasgow

# 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?

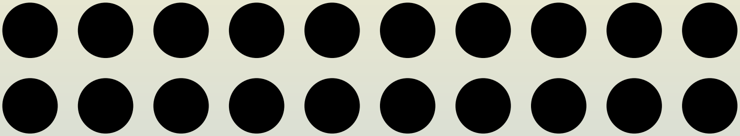
**UNIFORM SAMPLING**
Pick a single witness uniformly at random

**EXACT COUNTING**
Exactly how many witnesses?

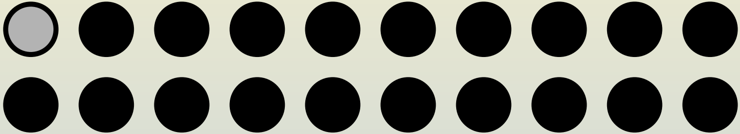
# Deciding, counting and enumerating

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

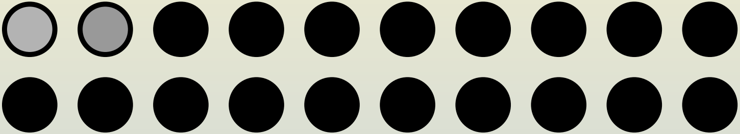
## If we can decide, we can find a witness



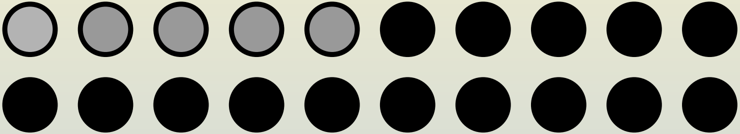
# If we can decide, we can find a witness



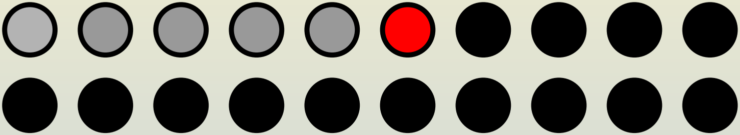
# If we can decide, we can find a witness



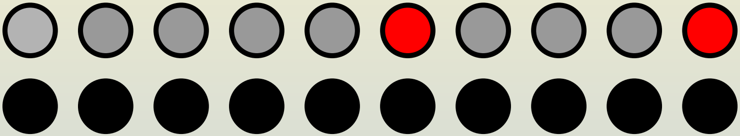
# If we can decide, we can find a witness



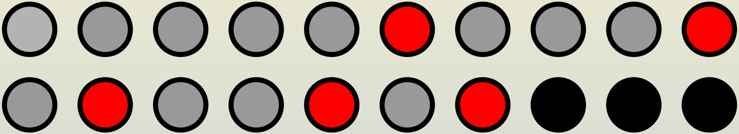
# If we can decide, we can find a witness



# 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, ESA 2014)**

*There exists an algorithm that extracts a witness using at most*

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

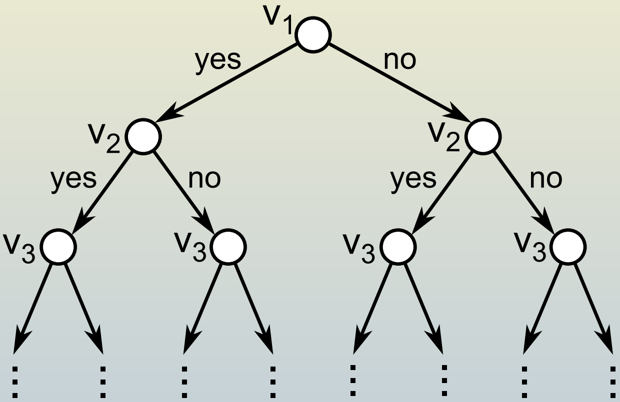
*queries to a deterministic decision algorithm.*

# With an **extension** oracle, we can find all witnesses

**EXT-ORA($X, Y$)**
*Input:* $X \subseteq U$ and $Y \subseteq X$
*Output:* 1 if there exists a witness $W$ with $Y \subseteq W \subseteq X$; 0 otherwise.

# With an **extension** oracle, we can find all witnesses

**EXT-ORA($X, Y$)**
*Input:* $X \subseteq U$ and $Y \subseteq X$
*Output:* 1 if there exists a witness $W$ with $Y \subseteq W \subseteq X$; 0 otherwise.



# Extension can be harder than inclusion

Suppose that a $k$-vertex subset is a witness if it either induces a clique or an independent set.

## Extension can be harder than inclusion

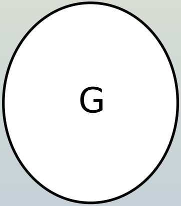
Suppose that a $k$-vertex subset is a witness if it either induces a clique or an independent set.

- The decision problem can be solved in time $f(k)$:
  - By Ramsey, for sufficiently large graphs the answer is always "yes".

# Extension can be harder than inclusion

Suppose that a $k$-vertex subset is a witness if it either induces a clique or an independent set.

- The decision problem can be solved in time $f(k)$:
  - By Ramsey, for sufficiently large graphs the answer is always "yes".
- The extension version is W[1]-hard:
  - Reduction from **p-**CLIQUE.

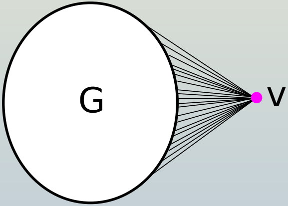


# Extension can be harder than inclusion

Suppose that a $k$-vertex subset is a witness if it either induces a clique or an independent set.

- The decision problem can be solved in time $f(k)$:
  - By Ramsey, for sufficiently large graphs the answer is always "yes".

- The extension version is W[1]-hard:
  - Reduction from **p-**CLIQUE.



# Enumerating without using extension

## Theorem

*There is a randomised algorithm to enumerate all witnesses of size k in a self-contained k-witness problem exactly once, whose expected number of calls to a deterministic decision oracle is at most $2^{O(k)} \log^2 n \cdot N$, where $N$ is the total number of witnesses.*

*Moreover, if an oracle call can be executed in time $g(k) \cdot n^{O(1)}$, then the expected total running time of the algorithm is*

$$2^{O(k)} \cdot g(k) \cdot n^{O(1)} \cdot N.$$

# Enumerating without using extension

## Definition

*A family $\mathcal{F}$ of hash functions from $[n]$ to $[k]$ is said to be k-perfect if, for every subset $A \subset [n]$ of size k, there exists $f \in \mathcal{F}$ such that the restriction of $f$ to $A$ is injective.*

## 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 $2^{O(k)} \cdot n \log n$.*

# Enumerating without using extension

**Definition**

*A family $\mathcal{F}$ of hash functions from $[n]$ to $[k]$ is said to be k-perfect if, for every subset $A \subset [n]$ of size k, there exists $f \in \mathcal{F}$ such that the restriction of f to A is injective.*

**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)

# Enumerating without using extension

### Definition

*A family $\mathcal{F}$ of hash functions from $[n]$ to $[k]$ is said to be k-perfect if, for every subset $A \subset [n]$ of size k, there exists $f \in \mathcal{F}$ such that the restriction of $f$ to $A$ is injective.*

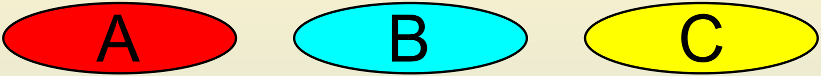
### Theorem (Alon, Yuster, Zwick, 1995)

*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 $2^{O(k)} \cdot n \log n$.*

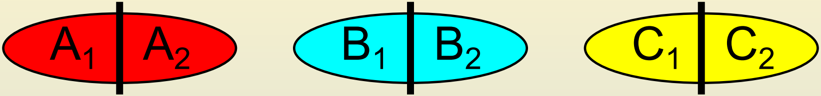
- IDEA: create many coloured instances, and enumerate the colourful copies in each (omitting duplicates)

- PROBLEM: although we're now looking for colourful witnesses, we still only have a decision algorithm for the uncoloured version...

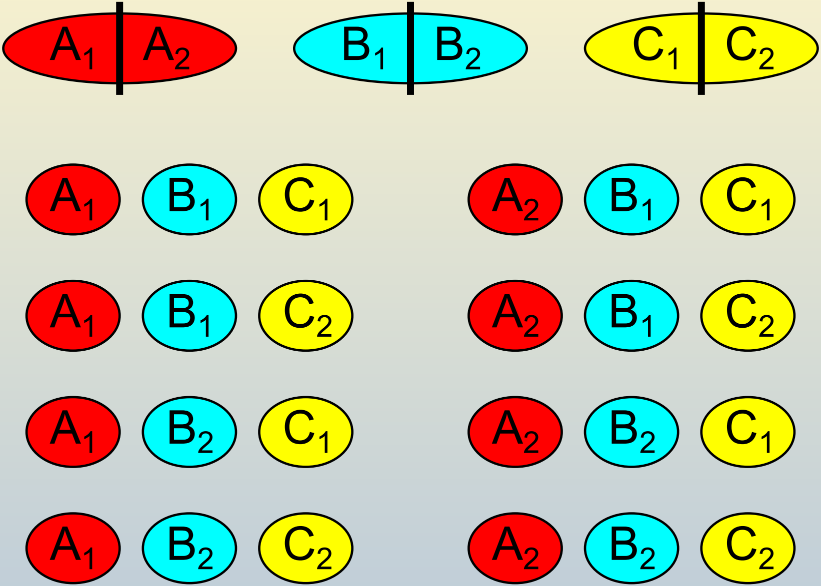
# Enumerating without using extension



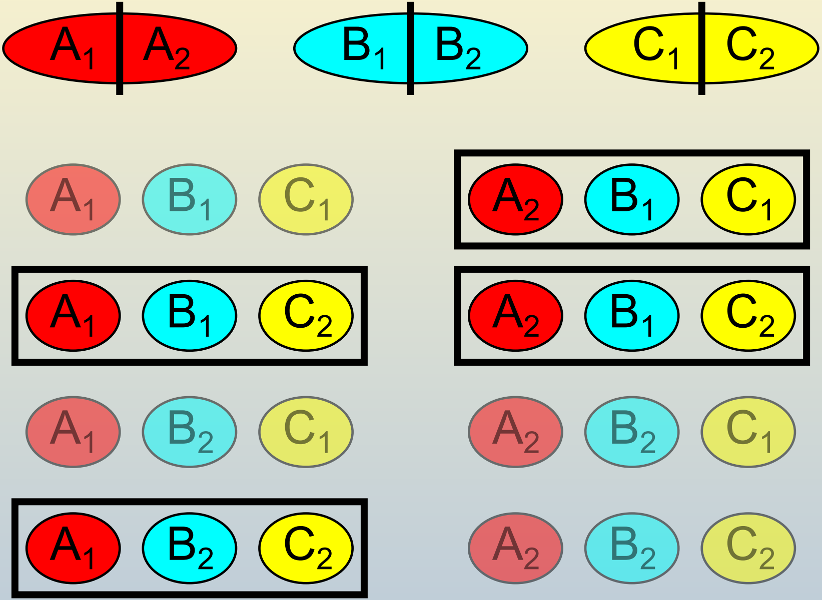
# Enumerating without using extension



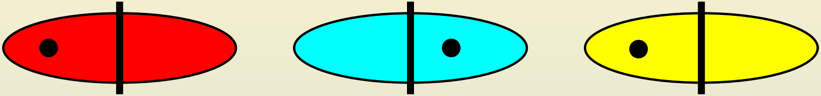
# Enumerating without using extension



# Enumerating without using extension



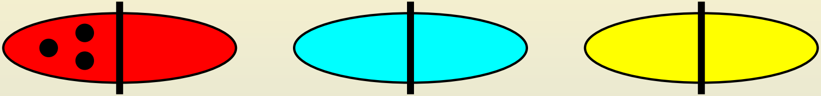
# Enumerating without using extension



If a witness is colourful:

- It will always survive in exactly one combination

# Enumerating without using extension



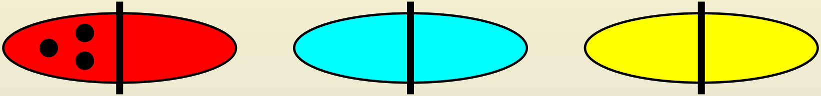
If a witness is colourful:

- 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

# Enumerating without using extension



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.

# Application to counting

## Theorem

*Let $\Pi$ be a self-contained k-witness problem, and suppose that $0 < \delta \leq \frac{1}{2}$ and $M \in \mathbb{N}$. Then there exists a randomised algorithm which makes at most $2^{O(k)} \log^2 n \, M \log(\delta^{-1})$ calls to a deterministic decision oracle for $\Pi$, and*

1. *if the number of witnesses in the instance of $\Pi$ is at most $M$, outputs with probability at least $1 - \delta$ the exact number of witnesses in the instance;*
2. *if the number of witnesses in the instance of $\Pi$ is strictly greater than $M$, always outputs "More than $M$."*

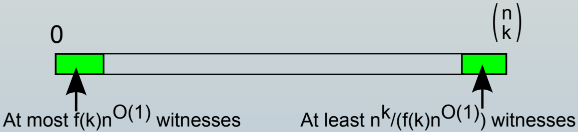
*Moreover, if there is an algorithm solving the decision version of $\Pi$ in time $g(k) \cdot n^{O(1)}$, then the expected running time of the randomised algorithm is bounded by $2^{O(k)} \cdot g(k) \cdot n^{O(1)} \cdot M \cdot \log(\delta^{-1})$.*

## Theorem

*Let $\Pi$ be a self-contained k-witness problem, and suppose that $0 < \delta \leq \frac{1}{2}$ and $M \in \mathbb{N}$. Then there exists a randomised algorithm which makes at most $2^{O(k)} \log^2 n\, M \log(\delta^{-1})$ calls to a deterministic decision oracle for $\Pi$, and*

1. *if the number of witnesses in the instance of $\Pi$ is at most $M$, outputs with probability at least $1 - \delta$ the exact number of witnesses in the instance;*
2. *if the number of witnesses in the instance of $\Pi$ is strictly greater than $M$, always outputs "More than M."*

*Moreover, if there is an algorithm solving the decision version of $\Pi$ in time $g(k) \cdot n^{O(1)}$, then the expected running time of the randomised algorithm is bounded by $2^{O(k)} \cdot g(k) \cdot n^{O(1)} \cdot M \cdot \log(\delta^{-1})$.*



# Application to counting

## Theorem

*Let $\Pi$ be a self-contained k-witness problem, and suppose that $0 < \delta \leq \frac{1}{2}$ and $M \in \mathbb{N}$. Then there exists a randomised algorithm which makes at most $2^{O(k)} \log^2 n\, M \log(\delta^{-1})$ calls to a deterministic decision oracle for $\Pi$, and*

1. *if the number of witnesses in the instance of $\Pi$ is at most $M$, outputs with probability at least $1 - \delta$ the exact number of witnesses in the instance;*
2. *if the number of witnesses in the instance of $\Pi$ is strictly greater than $M$, always outputs "More than $M$."*

*Moreover, if there is an algorithm solving the decision version of $\Pi$ in time $g(k) \cdot n^{O(1)}$, then the expected running time of the randomised algorithm is bounded by $2^{O(k)} \cdot g(k) \cdot n^{O(1)} \cdot M \cdot \log(\delta^{-1})$.*

# Open problems

- Can the randomised enumeration process be derandomised?

## Open problems

- Can the randomised enumeration process be derandomised?
- How common are self-contained $k$-witness problems whose decision version is FPT but for which the extension problem is W[1]-hard?

# Open problems

- Can the randomised enumeration process be derandomised?
- How common are self-contained $k$-witness problems whose decision version is FPT but for which the extension problem is W[1]-hard?
- Can we improve the algorithm to bound the expected time between finding one witness and the next?

# Open problems

- Can the randomised enumeration process be derandomised?
- How common are self-contained $k$-witness problems whose decision version is FPT but for which the extension problem is W[1]-hard?
- Can we improve the algorithm to bound the expected time between finding one witness and the next?

# **Thank you**