How long, O Bayesian network, will I sample thee?

A program analysis perspective on expected sampling times

Kevin Batz  Benjamin Lucien Kaminski
Joost-Pieter Katoen  Christoph Matheja

27th European Symposium on Programming
21st edition of the European Joint Conferences on Theory & Practice of Software

April 16, 2018, Thessaloniki, Greece
A program analysis perspective
Probabilistic Programs

\[
\{x := 7\}[1/3]\{x := 2\};
\]

\textbf{What does a probabilistic program do?}

Run program \(C\) on initial state \(\sigma\). Obtain a (sub-)
distribution over final states.

What is the runtime of \(C\) on input \(\sigma\)?

\textbf{Behavior of} \(C\) \textbf{not entirely determined by} \(\sigma\). Probabilistic nature of \(C\) influences its runtime.

Better Question: What is the expected runtime of \(C\) on input \(\sigma\)?
Probabilistic Programs

What does a probabilistic program $C$ do?

$$\{ x := 7 \} \cdot \frac{1}{3} \cdot \{ x := 2 \};$$

```
if (x > 5) {
    skip; skip; skip; skip
} else {
    skip
}
```
Probabilistic Programs

What does a probabilistic program $C$ do?

- Run program $C$ on initial state $\sigma$

$$\{ x := 7 \} \frac{1}{3} \{ x := 2 \} ;$$

if $(x > 5)$ {
  skip; skip; skip; skip
} else {
  skip
}
Probabilistic Programs

What does a probabilistic program $C$ do?

- Run program $C$ on initial state $\sigma$
- Obtain a distribution over final states

\[
\{ x := 7 \} \frac{1}{3} \{ x := 2 \};
\]

\[
\text{if } (x > 5) \{
\text{skip}; \text{skip}; \text{skip}; \text{skip}
\}\text{else}\{
\text{skip}
\}\]

Better Question: What is the expected runtime of $C$ on input $\sigma$?

Behavior of $C$ not entirely determined by $\sigma$
Probabilistic nature of $C$ influences its runtime

Benjamin Lucien Kaminski
How long, O Bayesian network, will I sample thee? 16.4.2018
Probabilistic Programs

What does a probabilistic program $C$ do?

- Run program $C$ on initial state $\sigma$
- Obtain a (sub-)distribution over final states

\[
\begin{align*}
\{ x := 7 \} & \frac{1}{3} \{ x := 2 \}; \\
\text{if} \ (x > 5) \{ & \\
& \text{skip; skip; skip; skip} \\
\} \ \text{else} \{ \\
& \text{skip} \ }
\end{align*}
\]
Probabilistic Programs

What does a probabilistic program $C$ do?

- Run program $C$ on initial state $\sigma$
- Obtain a (sub-)distribution over final states

What is the runtime of $C$ on input $\sigma$?

```plaintext
{ x := 7 } \frac{1}{3} \{ x := 2 \} ;
if (x > 5) {
    skip ; skip ; skip ; skip
} else {
    skip
}
```
Probabilistic Programs

What does a probabilistic program $C$ do?

- Run program $C$ on initial state $\sigma$
- Obtain a (sub-)distribution over final states

What is the runtime of $C$ on input $\sigma$?

- Behavior of $C$ not entirely determined by $\sigma$
Probabilistic Programs

What does a probabilistic program $C$ do?

- Run program $C$ on initial state $\sigma$
- Obtain a (sub-)distribution over final states

What is the runtime of $C$ on input $\sigma$?

- Behavior of $C$ not entirely determined by $\sigma$
- Probabilistic nature of $C$ influences its runtime
Probabilistic Programs

What does a probabilistic program $C$ do?

- Run program $C$ on initial state $\sigma$
- Obtain a (sub-)distribution over final states

What is the runtime of $C$ on input $\sigma$?

- Behavior of $C$ not entirely determined by $\sigma$
- Probabilistic nature of $C$ influences its runtime

Better Question:

What is the expected runtime of $C$ on input $\sigma$?
The ert Transformer [ESOP’16, LICS’16, JACM’18]

- Runtimes (random variables): \( T = \{ t \mid t : \text{States} \rightarrow \mathbb{R}_{\geq 0}^{\infty} \} \).
The ert Transformer [ESOP’16, LICS’16, JACM’18]

- **Runtimes (random variables):** \( T = \{ t \mid t: \text{States} \rightarrow \mathbb{R}_{\geq 0}^\infty \} \).

- **Use a continuation–passing style transformer** \( \text{ert}[C]: T \rightarrow T \).
The ert Transformer [ESOP’16, LICS’16, JACM’18]

- **Runtimes (random variables):** \( T = \{ t \mid t: \text{States} \rightarrow \mathbb{R}_{\geq 0}^\infty \} \).

- **Use a continuation–passing style transformer** \( \text{ert}[C]: T \rightarrow T \).

\[ C \]
The ert Transformer [ESOP’16, LICS’16, JACM’18]

- **Runtimes (random variables):** \( T = \{ t \mid t: \text{States} \rightarrow \mathbb{R}^\infty_{\geq 0} \} \).
- **Use a continuation–passing style transformer** \( \text{ert}[C]: T \rightarrow T \).

\[
\begin{align*}
C \quad & \quad t \\
\downarrow & \quad \uparrow \\
\text{time needed} & \quad \text{after executing } C
\end{align*}
\]
The ert Transformer [ESOP’16, LICS’16, JACM’18]

- Runtimes (random variables): $\mathbb{T} = \{ t \mid t: \text{States} \rightarrow \mathbb{R}_{\geq 0}^{\infty} \}$.
- Use a continuation-passing style transformer ert$_{\{C\}}$: $\mathbb{T} \rightarrow \mathbb{T}$.

![Diagram showing time $t$ and function $C$](image-url)
The ert Transformer [ESOP’16, LICS’16, JACM’18]

- **Runtimes (random variables):** \( T = \{ t \mid t: \text{States} \to \mathbb{R}_{\geq 0}^\infty \} \).

- Use a continuation–passing style transformer \( \text{ert}[C]: T \to T \).

\[
\text{expected time needed to execute } C \text{ and then let time } t \text{ pass}
\]

\[
\text{time needed after executing } C
\]
The ert Transformer [ESOP’16, LICS’16, JACM’18]

- Runtimes (random variables): \( \mathbb{T} = \{ t \mid t : \text{States} \to \mathbb{R}_{\geq 0}^\infty \} \).
- Use a continuation–passing style transformer \( \text{ert}[\mathcal{C}] : \mathbb{T} \to \mathbb{T} \).

\[
\begin{align*}
\text{C}' & \quad \text{ert} \ [\mathcal{C}] (t) \\
\mathcal{C} & \quad \text{expected time needed to execute } \mathcal{C} \\
& \text{and then let time } t \text{ pass} \\
& \quad \text{time needed after executing } \mathcal{C}'
\end{align*}
\]
The ert Transformer [ESOP’16, LICS’16, JACM’18]

- **Runtimes (random variables):** \( \mathbb{T} = \{ t \mid t: \text{States} \rightarrow \mathbb{R}^{\geq 0} \} \).
- **Use a continuation–passing style transformer** \( \text{ert}[C] : \mathbb{T} \rightarrow \mathbb{T} \).

\[ C' \xrightarrow{\text{ert } [C]} (t) \xrightarrow{C} t \]

- **expected time needed to execute** \( C \) and then let time \( t \) pass
- **time needed after executing** \( C' \)
The ert Transformer [ESOP’16, LICS’16, JACM’18]

- **Runtimes (random variables):** \( T = \{ t \mid t: \text{States} \rightarrow \mathbb{R}_{\geq 0}^\infty \} \).

- **Use a continuation–passing style transformer** ert\([C]\): \( T \rightarrow T \).

\[
\begin{align*}
\text{expected time needed to execute } C' \text{ and then let time } \text{ert}[C](t) \text{ pass} \\
\text{expected time needed to execute } C \text{ and then let time } t \text{ pass} \\
time needed after executing \ C
\end{align*}
\]
The ert Transformer [ESOP’16, LICS’16, JACM’18]

- **Runtimes (random variables):** \( T = \{ t \mid t: \text{States} \rightarrow \mathbb{R}_{\geq 0}^{\infty} \} \).
- **Use a continuation–passing style transformer** ert\([C]\) : \( T \rightarrow T \).

\[
\begin{align*}
\text{expected time needed to execute } C'' \\
\text{and then } \text{let time } ert\([C]\) (t) \text{ pass}
\end{align*}
\]

**in other words:**

\[
\begin{align*}
\text{expected time needed to execute } C'', \\
\text{then execute } C' \\
\text{and then let time } t \text{ pass}
\end{align*}
\]
\begin{align*}
\{ x := 7 \} & \cdot \frac{1}{3} \{ x := 2 \} \\
\text{if} \ (x > 5) \{ \\
& \quad \text{skip; skip; skip; skip} \\
\} \text{else} \{ \\
& \quad \text{skip} \\
\} \\
& \text{skip}
\end{align*}
\[
\{ x := 7 \} \left[ \frac{1}{3} \right] \{ x := 2 \} ;
\]

\[
\text{if } (x > 5) \{
\begin{align*}
\text{skip} ; \text{skip} ; \text{skip} ; \text{skip} \\
\end{align*}
\}
\]

\[
\text{else} \{
\begin{align*}
\text{skip} \\
\end{align*}
\}
\]

\[
\text{skip} \\
\text{// 0}
\]
\{ x := 7 \} \left[ \frac{1}{3} \right] \{ x := 2 \} ;

if (x > 5) {

    skip ; skip ; skip ; skip

} else {

    skip

} ;

// 1
skip

// 0
\{ x := 7 \} \{ \frac{1}{3} \} \{ x := 2 \} ;

\textbf{if} (x > 5) \{

\quad \text{skip} ; \text{skip} ; \text{skip} ; \text{skip}
\quad \quad \text{// 1}
\}

\textbf{else} \{

\quad \text{skip}
\quad \quad \text{// 1}
\}

\text{// 1}

\text{skip}
\quad \text{// 0}
\{ x := 7 \} [1/3] \{ x := 2 \} ;

if (x > 5) {

    skip ; skip ; skip ; skip
    // 1
}
else {

    // 2
    skip
    // 1
}

// 1
skip
// 0
\[
\{ x := 7 \}[1/3] \{ x := 2 \} ;
\]

```plaintext
if (x > 5) {
    // 5
    skip ; skip ; skip ; skip
    // 1
} else {
    // 2
    skip
    // 1
} ;
// 1
skip
// 0
```
\[
\{ x := 7 \} \frac{1}{3} \{ x := 2 \} ;
\]

// 1 + [x > 5] \cdot 5 + [x \leq 5] \cdot 2

if (x > 5) {
    // 5
    skip ; skip ; skip ; skip
    // 1
} else {
    // 2
    skip
    // 1
}

// 1

skip
// 0
\[ 2 + \frac{1}{3} \cdot 5 + \frac{2}{3} \cdot 2 \]
\[
\{ x := 7 \} \left[ \frac{1}{3} \right] \{ x := 2 \};
\]
\[ 1 + [x > 5] \cdot 5 + [x \leq 5] \cdot 2 \]
\[ \text{if } (x > 5) \{ \]
\[ \quad 5 \]
\[ \quad \text{skip}; \quad \text{skip}; \quad \text{skip}; \quad \text{skip} \]
\[ \quad 1 \]
\[ \} \quad \text{else} \{ \]
\[ \quad 2 \]
\[ \quad \text{skip} \]
\[ \quad 1 \]
\[ \} \]
\[ 1 \]
\[ \text{skip} \]
\[ 0 \]
// 5
{ x := 7 \} [1/3] \{ x := 2 \};
// 1 + [x > 5] \cdot 5 + [x \leq 5] \cdot 2
if (x > 5) {
   // 5
   skip; skip; skip; skip
   // 1
} else {
   // 2
   skip
   // 1
}
};
// 1
// 1
skip
// 0
Expected Runtimes of Loops

Recall $T = \{ t \mid t : \text{States} \to \mathbb{R}^\infty \geq 0 \}$

$(T, \preceq)$ is a complete lattice with $s \preceq t$ iff $\forall \sigma \in \text{States}: s(\sigma) \leq t(\sigma)$

Expected runtime of $\text{while} (\phi) \{ C \}$ and then let time $t$ pass

Use a least fixed-point construct: Let $\Phi_t(X) = 1 + [\neg \phi] \cdot t + [\phi] \cdot \text{ert}_{J \{ C \} K}(X)$.

Then $\text{ert}_{J \{ C \} K}(t) = \text{lfp} \Phi_t$. 

How long, O Bayesian network, will I sample thee? 16.4.2018
Expected Runtimes of Loops

Recall $T = \{ t \mid t : \text{States} \rightarrow \mathbb{R}_{\geq 0}^\infty \}$
Expected Runtimes of Loops

- Recall $\mathbb{T} = \{ t \mid t: \text{States} \rightarrow \mathbb{R}^\infty_{\geq 0} \}$

- $(\mathbb{T}, \preceq)$ is a complete lattice with

  $$s \preceq t \quad \text{iff} \quad \forall \sigma \in \text{States}: \quad s(\sigma) \leq t(\sigma)$$
Expected Runtimes of Loops

- Recall \( T = \{ t \mid t: \text{States} \to \mathbb{R}^\infty_{\geq 0} \} \)
- \((T, \preceq)\) is a complete lattice with

\[
 s \preceq t \quad \text{iff} \quad \forall \sigma \in \text{States}: \quad s(\sigma) \leq t(\sigma)
\]

Expected runtime of \textbf{while}(\varphi)\{C\} and then let time \textbf{t} pass
Expected Runtimes of Loops

- Recall $\mathbb{T} = \{ t \mid t : \text{States} \to \mathbb{R}_{\geq 0} \}$
- $(\mathbb{T}, \preceq)$ is a complete lattice with
  \[
  s \preceq t \iff \forall \sigma \in \text{States}: s(\sigma) \leq t(\sigma)
  \]

Expected runtime of while ($\varphi$) { $C$ } and then let time $t$ pass

Use a least fixed–point construct: Let

\[
\Phi_t(X) = 1 + \lbrack \neg \varphi \rbrack \cdot t + \lbrack \varphi \rbrack \cdot \text{ert} \llbracket C \rrbracket (X).
\]
Expected Runtimes of Loops

- Recall $\mathcal{T} = \{ t \mid t : \text{States} \rightarrow \mathbb{R}^\infty_{\geq 0} \}$
- $(\mathcal{T}, \preceq)$ is a complete lattice with

$$s \preceq t \iff \forall \sigma \in \text{States}: \ s(\sigma) \leq t(\sigma)$$

Expected runtime of $\text{while}(\varphi)\{C\}$ and then let time $t$ pass

Use a least fixed-point construct: Let

$$\Phi_t(X) = 1 + [\neg \varphi] \cdot t + [\varphi] \cdot \text{ert} \mathcal{[C]}(X).$$

Then $\text{ert} \mathcal{[\text{while}(\varphi)\{C\}]}(t) = \text{lfp} \ \Phi_t.$
What is the expected runtime of \( \text{while}(\varphi)\{C\} \)?

Let \( \Phi_t(X) = 1 + [\neg \varphi] \cdot t + [\varphi] \cdot \text{ert} [C] (X) \).

Then \( \text{ert} \left[ \text{while}(\varphi)\{C\} \right] (t) = \text{lfp} \Phi_t \).
What is the expected runtime of $\text{while}(\varphi)\{C\}$?

Let $\Phi_t(X) = 1 + [\neg \varphi] \cdot t + [\varphi] \cdot \text{ert}[C](X)$.

Then $\text{ert}[\text{while}(\varphi)\{C\}](t) = \text{lfp} \Phi_t$.

Induction Rule for Upper Bounds
What is the expected runtime of $\text{while}(\varphi)\{C\}$?

Let $\Phi_t(X) = 1 + [\neg\varphi] \cdot t + [\varphi] \cdot \text{ert} [C](X)$.

Then $\text{ert} [\text{while}(\varphi)\{C\}](t) = \text{lfp} \Phi_t$.

Induction Rule for Upper Bounds

Choose $I \in T$. 
What is the expected runtime of while (\(\varphi\)) \(\{ C \}\)?

Let \(\Phi_t(X) = 1 + [\neg \varphi] \cdot t + [\varphi] \cdot \text{ert} [C](X)\).

Then \(\text{ert}[\text{while}(\varphi)\{C\}](t) = \text{lfp} \Phi_t\).

Induction Rule for Upper Bounds

Choose \(I \in T\). Then

\[\Phi_t(I) \leq I \text{ implies } \text{ert}[\text{while}(\varphi)\{C\}](t) \leq I\]
What is the expected runtime of $\textbf{while} (\varphi) \{ C \}$?

Let $\Phi_t(X) = 1 + [\neg \varphi] \cdot t + [\varphi] \cdot \text{ert} \ [C] (X)$.

Then $\text{ert} \ [\textbf{while} (\varphi) \{ C \}] (t) = \text{lfp} \ \Phi_t$.

**Induction Rule for Upper Bounds**

Choose $I \in T$. Then

$$\Phi_t(I) \preceq I \quad \text{implies} \quad \text{ert} \ [\textbf{while} (\varphi) \{ C \}] (t) \preceq I$$

**Rule for Lower Bounds**
What is the expected runtime of \( \text{while}(\varphi)\{C\} \)?

Let \( \Phi_t(X) = 1 + [\neg \varphi] \cdot t + [\varphi] \cdot \text{ert} [C] (X) \).

Then \( \text{ert} [\text{while}(\varphi)\{C\}] (t) = \text{lfp} \ \Phi_t \).

**Induction Rule for Upper Bounds**

Choose \( I \in T \). Then

\[
\Phi_t(I) \leq I \implies \text{ert} [\text{while}(\varphi)\{C\}] (t) \leq I
\]

**Rule for Lower Bounds**

Choose sequence \( 0 = I_0 \leq I_1 \leq I_2 \leq \cdots \subset T \).
What is the expected runtime of \( \text{while}(\varphi)\{C\} \)?

Let \( \Phi_t(X) = 1 + [\neg \varphi] \cdot t + [\varphi] \cdot \text{ert} \ [C] (X) \).

Then \( \text{ert} \ [\text{while}(\varphi)\{C\}] (t) = \text{lfp} \Phi_t \).

**Induction Rule for Upper Bounds**

Choose \( I \in T \). Then

\[
\Phi_t(I) \leq I \quad \text{implies} \quad \text{ert} \ [\text{while}(\varphi)\{C\}] (t) \leq I
\]

**Rule for Lower Bounds**

Choose sequence \( 0 = I_0 \leq I_1 \leq I_2 \leq \cdots \subset T \). Then

\[
I_{n+1} \leq \Phi_t(I_n) \quad \text{implies} \quad \sup_n I_n \leq \text{ert} \ [\text{while}(\varphi)\{C\}] (t)
\]
What is the expected runtime of while(\(\varphi\))\{C\}?

Let \(\Phi_t(X) = 1 + [\neg \varphi] \cdot t + [\varphi] \cdot \text{ert}[C](X)\).

Then \(\text{ert}[\text{while}(\varphi)\{C\}](t) = \text{lfp } \Phi_t\).

**Induction Rule for Upper Bounds**

Choose \(I \in T\). Then

\[\Phi_t(I) \preceq I \implies \text{ert}[\text{while}(\varphi)\{C\}](t) \preceq I\]

**Rule for Lower Bounds**

Choose sequence \(0 = I_0 \preceq I_1 \preceq I_2 \preceq \cdots \subseteq T\). Then

\[I_{n+1} \preceq \Phi_t(I_n) \implies \sup_n I_n \preceq \text{ert}[\text{while}(\varphi)\{C\}](t)\]
What is the expected runtime of while ($\varphi$) $\{C\}$?

Let $\Phi_t(X) = 1 + [\neg \varphi] \cdot t + [\varphi] \cdot \text{ert} \mathcal{K}(C)(X)$.

Then $\text{ert} \mathcal{K} \text{while}(\varphi) \{C\}(t) = \text{lfp } \Phi_t$.

Induction Rule for Upper Bounds

Choose $I \in T$. Then

$\Phi_t(I) \preceq I$ implies $\text{ert} \mathcal{K} \text{while}(\varphi) \{C\}(t) \preceq I$

Rule for Lower Bounds

Choose sequence $0 = I_0 \preceq I_1 \preceq I_2 \preceq \cdots \subseteq T$. Then

$I_{n+1} \preceq \Phi_t(I_n)$ implies $\sup_n I_n \preceq \text{ert} \mathcal{K} \text{while}(\varphi) \{C\}(t)$
Proving upper bounds is "easy". (induction)

Proving lower bounds is hard. (the nasty sequence thing)

Proving exact runtimes is also hard.
Proving upper bounds is “easy”.

(induction)
Proving upper bounds is “easy”.  
(induction)

Proving lower bounds is hard.  
(the nasty sequence thing)
Proving upper bounds is “easy”.
(induction)

Proving lower bounds is hard.
(the nasty sequence thing)

Proving exact runtimes is also hard.
A new proof rule for exact expected runtimes
A New Proof Rule

The Coupon Collector

cp := [0, ..., 0]  i := 1  x := N
while (x > 0) {
    while (cp[i] ≠ 0) {
        i := Unif[1...N]
    }
    cp[i] := 1  x := x - 1
}

Expected runtime analysis quite involved
Part of the problem: Nested loops!
But the inner loop is simple!
The Coupon Collector

\[ cp := [0, \ldots, 0] ; i := 1 ; x := N \]
\[
\text{while}(x > 0) \{
\text{while}(cp[i] \neq 0) \{
\quad i \approx \text{Unif}[1\ldots N] \\
\}
\}
\]
\[ cp[i] := 1 ; x := x - 1 \]

- Expected runtime analysis quite involved
The Coupon Collector

\[ cp := [0, \ldots, 0] \quad i := 1 \quad x := N \]

\[ \text{while}(x > 0) \{ \]
\[ \quad \text{while}(cp[i] \neq 0) \{ \]
\[ \quad \quad i \approx \text{Unif}[1 \ldots N] \]
\[ \quad \} \]
\[ \quad cp[i] := 1 \quad x := x - 1 \]
\[ \} \]

- Expected runtime analysis quite involved
- Part of the problem: Nested loops!
A New Proof Rule

The Coupon Collector

\[ I := 0, \ldots, 0 \quad \#
\]
\[ i := 1 \quad \#
\]
\[ \alpha := N \]

Expected runtime analysis quite involved

Part of the problem: Nested loops!

But the inner loop is simple!

Applying ert to the Coupon Collector

This is an excerpt of the proof of the inner loop

Benjamin Lucien Kaminski
The Coupon Collector

\[ cp := [0, \ldots, 0] ; i := 1 ; x := N \]
\[ \text{while } (x > 0) \{ \]
\[ \quad \text{while } (cp[i] \neq 0) \{ \]
\[ \quad \\]
\[ \quad i \approx \text{Unif}[1\ldots N] \]
\[ \quad x := x - 1 \]
\[ \} \]
\[ cp[i] := 1 ; x := x - 1 \]
\[ \} \]

- Expected runtime analysis quite involved
- Part of the problem: \text{Nested loops!}
The Coupon Collector

\[ cp := [0, \ldots, 0] \; i := 1 \; x := N \]

\[
\text{while } (x > 0) \{ \\
\quad \text{while } (cp[i] \neq 0) \{ \\
\quad \quad i \sim \text{Unif}[1 \ldots N] \\
\quad \} \\
\quad cp[i] := 1 \; x := x - 1 \\
\} 
\]

- Expected runtime analysis quite involved
- Part of the problem: **Nested loops!**
- But the inner loop is simple!
The Coupon Collector’s Inner Loop

\[
\text{while}(cp[i] \neq 0)\
\quad i \sim \text{Unif}[1...N]
\]

Desideratum: Closed form for \( t \) of inner loop

Observations:
1. No information flow across loop iterations!
2. Loop terminates with "constant" probability after each iteration.

Simple loops should have simple runtime proofs!
The Coupon Collector’s Inner Loop

\[
\text{while}(cp[i] \neq 0) \{
    i \approx \text{Unif}[1 \ldots N]
\}
\]

- **Desideratum:** Closed form for ert of inner loop
The Coupon Collector’s Inner Loop

while \((cp[i] \neq 0)\) {
    \(i \sim \text{Unif}[1 \ldots N]\)
} 

- **Desideratum:** Closed form for ert of inner loop
- **Observations:**
The Coupon Collector’s Inner Loop

\[
\text{while}(cp[i] \neq 0) \{
\quad i \sim \text{Unif}[1...N]
\}
\]

- **Desideratum:** Closed form for ert of inner loop
- **Observations:**
  1. **No information flow** across loop iterations!
The Coupon Collector’s Inner Loop

\[
\text{while}(cp[i] \neq 0) \{
    i \sim \text{Unif}[1 \ldots N]
\}
\]

- **Desideratum:** Closed form for cdf of inner loop
- **Observations:**
  1. No information flow across loop iterations!
  2. Loop terminates with “constant” probability after each iteration.
The Coupon Collector’s Inner Loop

\[
\text{while}(cp[i] \neq 0)\
\begin{align*}
i & \approx \text{Unif}[1 \ldots N] \\
\end{align*}
\]

- **Desideratum:** Closed form for ert of inner loop
- **Observations:**
  1. No information flow across loop iterations!
  2. Loop terminates with “constant” probability after each iteration.
- **Simple loops should have simple runtime proofs!**
Weakest Preexpectations

Let $f \in \mathbb{T}$ (think: $f$ is a random variable).
Weakest Preexpectations

Let $f \in \mathbb{T}$ (think: $f$ is a random variable).

Weakest preexpectation of $C$ with respect to $f$: 

\[
\text{wp}_J(C)(f)(\sigma) = \text{EV of } f \text{ after executing } C \text{ on } \sigma
\]

Example:

\[
\text{wp}_J\{x := 5\}[\frac{1}{2}]\{x := 3\}K(x) = \frac{1}{2} \cdot 5 + \frac{1}{2} \cdot (x + 5)
\]
Weakest Preexpectations

Let \( f \in \mathbb{T} \) (think: \( f \) is a random variable).

Weakest preexpectation of \( C \) with respect to \( f \):

\[
\text{wp } \left[ C \right] (f) : \text{States} \rightarrow \mathbb{R}_{\geq 0},
\]
Weakest Preexpectations

Let $f \in T$ (think: $f$ is a random variable).

Weakest preexpectation of $C$ with respect to $f$:

$$\text{wp } [C](f) : \text{States } \rightarrow \mathbb{R}_{\geq 0},$$

$$\text{wp } [C](f)(\sigma) = \text{EV of } f \text{ after executing } C \text{ on } \sigma$$
Weakest Preexpectations

Let \( f \in \mathbb{T} \) (think: \( f \) is a random variable).

**Weakest preexpectation of** \( C \) **with respect to** \( f \):

\[
\text{wp } [C] (f) : \text{ States } \rightarrow \mathbb{R}_{\geq 0},
\]

\[
\text{wp } [C] (f) (\sigma) = \text{ EV of } f \text{ after executing } C \text{ on } \sigma
\]

**Example:**

\[
\text{wp } \left[ \{ x := 5 \} [1/2] \{ x := 3 \} \right] (x)
\]
Weakest Preexpectations

Let $f \in \mathbb{T}$ (think: $f$ is a random variable).

Weakest preexpectation of $C$ with respect to $f$:

$$wp \left[ C \right] (f) : \text{States} \rightarrow \mathbb{R}_{\geq 0},$$

$$wp \left[ C \right] (f) (\sigma) = \text{EV of } f \text{ after executing } C \text{ on } \sigma$$

Example:

$$wp \left[ \{ x := 5 \} \left[ \frac{1}{2} \right] \{ x := 3 \} \right] (x) = \frac{1}{2} \cdot 5 + \frac{1}{2} \cdot 3$$
Weakest Preexpectations

Let $f \in \mathbb{T}$ (think: $f$ is a random variable).

Weakest preexpectation of $C$ with respect to $f$:

$$\text{wp } [C] (f) : \text{States} \rightarrow \mathbb{R}_{\geq 0},$$

$$\text{wp } [C] (f) (\sigma) = \text{EV of } f \text{ after executing } C \text{ on } \sigma$$

Example:

$$\text{wp } [[[\{ x := 5 \} [1/2] \{ x := 3 \}]] (x) = 4$$
Weakest Preexpectations

Let $f \in \mathbb{T}$ (think: $f$ is a random variable).

Weakest preexpectation of $C$ with respect to $f$:

$$wp\ [C](f) : \text{States} \rightarrow \mathbb{R}_{\geq 0},$$

$$wp\ [C](f)(\sigma) = \text{EV of } f \text{ after executing } C \text{ on } \sigma$$

**Example:**

$$wp\ [\{x := 5\} [1/2] \{x := 3\}] (x) = 4$$

$$wp\ [\{x := 5\} [1/2] \{x := x + 5\}] (x)$$
Weakest Preexpectations

Let $f \in \mathbb{T}$ (think: $f$ is a random variable).

Weakest preexpectation of $C$ with respect to $f$:

$$
\text{wp} \left[ C \right] (f) : \text{States} \rightarrow \mathbb{R}_{\geq 0},
$$

$$
\text{wp} \left[ C \right] (f) (\sigma) = \text{EV of } f \text{ after executing } C \text{ on } \sigma
$$

Example:

$$
\text{wp} \left[ \{ x := 5 \} [1/2] \{ x := 3 \} \right] (x) = 4
$$

$$
\text{wp} \left[ \{ x := 5 \} [1/2] \{ x := x + 5 \} \right] (x) = \frac{1}{2} \cdot 5 + \frac{1}{2} \cdot (x + 5)
$$
Weakest Preexpectations

Let $f \in \mathbb{T}$ (think: $f$ is a random variable).

Weakest preexpectation of $C$ with respect to $f$:

$$\text{wp} [C] (f) : \text{States} \rightarrow \mathbb{R}_{\geq 0},$$

$$\text{wp} [C] (f) (\sigma) = \text{EV of } f \text{ after executing } C \text{ on } \sigma$$

Example:

$$\text{wp} \left[ \{ x := 5 \} \left[ \frac{1}{2} \right] \{ x := 3 \} \right] (x) = 4$$

$$\text{wp} \left[ \{ x := 5 \} \left[ \frac{1}{2} \right] \{ x := x + 5 \} \right] (x) = \frac{x}{2} + 5$$
Theorem: A New Proof Rule for Expected Runtimes

Let \( \text{while} (\phi) \{ C \} \) be \( t \)-i.i.d. Every loop iteration has equal termination probability \( \text{EV} \) determined by a single loop iteration \( C \) terminates almost–surely (with probability 1) Every loop iteration takes equal expected time Then 
\[
ert_J \text{while} (\phi) \{ C \} K(t) \text{is given by}
\]
\[
1 + \left[ \neg \phi \right] \cdot t + \left[ \phi \right] \cdot \left( 1 + ert_J K(\left[ \neg \phi \right] \cdot t) \right) - wp_J K(\left[ \phi \right])
\]
Closed form for exact expected runtime! No (sequence) guessing needed!
Theorem: A New Proof Rule for Expected Runtimes

- Let $\text{while}(\varphi)\{C\}$ be $t$–i.i.d.
Theorem: A New Proof Rule for Expected Runtimes

- Let $\textbf{while}(\varphi)\{C\}$ be $t$–i.i.d.
  - Every loop iteration has equal termination probability

Closed form for exact expected runtime!
No (sequence) guessing needed!
Theorem: A New Proof Rule for Expected Runtimes

- Let \( \textbf{while} (\varphi) \{ C \} \) be \( t \)-i.i.d.
  - Every loop iteration has equal \textit{termination probability}
  - \( \text{EV of } t \) determined by a single loop iteration

\[ 1 + [\neg \varphi] \cdot t + [\varphi] \cdot 1 + \text{ert}_{J} C_{K}(\neg \varphi \cdot t) \]

Closed form for exact expected runtime!
No (sequence) guessing needed!
Theorem: A New Proof Rule for Expected Runtimes

- Let $\text{while}(\varphi)\{C\}$ be $t$-i.i.d.
  - Every loop iteration has equal termination probability
  - EV of $t$ determined by a single loop iteration
- $C$ terminates almost–surely (with probability 1)
Theorem: A New Proof Rule for Expected Runtimes

- Let $\textbf{while}(\varphi)\{C\}$ be $t$-i.i.d.
  - Every loop iteration has equal \textit{termination probability}
  - EV of $t$ determined by a single loop iteration
- $C$ terminates almost-surely (with probability 1)
- Every loop iteration takes equal expected time
Theorem: A New Proof Rule for Expected Runtimes

- Let $\text{while}(\varphi)\{C\}$ be $t$–i.i.d.
  - Every loop iteration has equal termination probability
  - EV of $t$ determined by a single loop iteration
- $C$ terminates almost–surely (with probability 1)
- Every loop iteration takes equal expected time

Then $\text{ert}[\text{while}(\varphi)\{C\}](t)$ is given by

\[
1 + [\neg\varphi] \cdot t + [\varphi] \cdot \frac{1 + \text{ert}[C]([\neg\varphi] \cdot t)}{1 - \text{wp}[C]([\varphi])}
\]
Theorem: A New Proof Rule for Expected Runtimes

- Let $\text{while}(\varphi)\{C\}$ be $t$–i.i.d.
  - Every loop iteration has equal termination probability
  - EV of $t$ determined by a single loop iteration
- $C$ terminates almost–surely (with probability 1)
- Every loop iteration takes equal expected time

Then $\text{ert}[\text{while}(\varphi)\{C\}](t)$ is given by

$$1 + [\neg \varphi] \cdot t + [\varphi] \cdot \frac{1 + \text{ert}[C] ([\neg \varphi] \cdot t)}{1 - \text{wp}[C] ([\varphi])}$$

Closed form for exact expected runtime!
Theorem: A New Proof Rule for Expected Runtimes

- Let \( \text{while}(\varphi)\{C\} \) be \( t\text{-i.i.d.} \).
  - Every loop iteration has equal termination probability
  - EV of \( t \) determined by a single loop iteration
- \( C \) terminates almost–surely (with probability 1)
- Every loop iteration takes equal expected time

Then \( \text{ert}[\text{while}(\varphi)\{C\}](t) \) is given by

\[
1 + [\neg \varphi] \cdot t + [\varphi] \cdot \frac{1 + \text{ert}[C]([\neg \varphi] \cdot t)}{1 - \text{wp}[C]([\varphi])}
\]

Closed form for exact expected runtime!
No (sequence) guessing needed!
How long, O Bayesian network, will I sample thee?
Bayesian Network $\rightarrow$ Probabilistic Program

$D \rightarrow G \rightarrow M$

$P \rightarrow G$

$G$

repeat

$x_D \approx 0.95 \cdot \langle 0 \rangle + 0.05 \cdot \langle 1 \rangle$

$x_P \approx 0.98 \cdot \langle 0 \rangle + 0.02 \cdot \langle 1 \rangle$

if $(x_D = 0 \land x_P = 0)$

$x_G \approx 0.99 \cdot \langle 0 \rangle + 0.01 \cdot \langle 1 \rangle$

else

$x_G \approx 0.01 \cdot \langle 0 \rangle + 0.99 \cdot \langle 1 \rangle$

endif

if $(x_G = 0)$

$x_M \approx 0.9 \cdot \langle 0 \rangle + 0.1 \cdot \langle 1 \rangle$

else

$x_M \approx 0.3 \cdot \langle 0 \rangle + 0.7 \cdot \langle 1 \rangle$

endif

until $(x_G = 0 \land x_P = 0)$
Bayesian Network → Probabilistic Program

\[
x_D \approx 0.95 \cdot \langle 0 \rangle + 0.05 \cdot \langle 1 \rangle;
\]

<table>
<thead>
<tr>
<th></th>
<th>(D = 0)</th>
<th>(D = 1)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.95</td>
<td>0.05</td>
</tr>
</tbody>
</table>
Bayesian Network $\rightarrow$ Probabilistic Program

\[ x_D \approx 0.95 \cdot \langle 0 \rangle + 0.05 \cdot \langle 1 \rangle; \]
\[ x_P \approx 0.98 \cdot \langle 0 \rangle + 0.02 \cdot \langle 1 \rangle; \]
Bayesian Network $\rightarrow$ Probabilistic Program

$$x_D \approx 0.95 \cdot \langle 0 \rangle + 0.05 \cdot \langle 1 \rangle;$$
$$x_P \approx 0.98 \cdot \langle 0 \rangle + 0.02 \cdot \langle 1 \rangle;$$

if \((x_D = 0 \land x_P = 0)\)

$$x_G \approx 0.99 \cdot \langle 0 \rangle + 0.01 \cdot \langle 1 \rangle;$$

...
Bayesian Network $\rightarrow$ Probabilistic Program

$x_D \approx 0.95 \cdot \langle 0 \rangle + 0.05 \cdot \langle 1 \rangle$

$x_P \approx 0.98 \cdot \langle 0 \rangle + 0.02 \cdot \langle 1 \rangle$

if ($x_D = 0 \land x_P = 0$)

\begin{align*}
  x_G &\approx 0.99 \cdot \langle 0 \rangle + 0.01 \cdot \langle 1 \rangle;
\end{align*}

else

\begin{align*}
  x_G &\approx 0.01 \cdot \langle 0 \rangle + 0.99 \cdot \langle 1 \rangle;
\end{align*}

until ($x_G = 0 \land x_P = 0$)
Bayesian Network $\rightarrow$ Probabilistic Program

\[
\begin{align*}
  x_D &\approx 0.95 \cdot \langle 0 \rangle + 0.05 \cdot \langle 1 \rangle; \\
  x_P &\approx 0.98 \cdot \langle 0 \rangle + 0.02 \cdot \langle 1 \rangle; \\
  \text{if}(x_D = 0 \land x_P = 0) &\{ \\
    x_G &\approx 0.99 \cdot \langle 0 \rangle + 0.01 \cdot \langle 1 \rangle; \\
    \vdots \\
  \} \text{ else } &\{ \\
    x_G &\approx 0.01 \cdot \langle 0 \rangle + 0.99 \cdot \langle 1 \rangle; \\
  \} \\
  \text{if}(x_G = 0) &\{ \\
    x_M &\approx 0.9 \cdot \langle 0 \rangle + 0.1 \cdot \langle 1 \rangle;
  \}
\end{align*}
\]
Bayesian Network → Probabilistic Program

\[
\begin{align*}
G &= 0 & M = 0 & 0.9 & M = 1 & 0.1 \\
G &= 1 & 0.3 & 0.7
\end{align*}
\]

\[
x_D \approx 0.95 \cdot \langle 0 \rangle + 0.05 \cdot \langle 1 \rangle; \\
x_P \approx 0.98 \cdot \langle 0 \rangle + 0.02 \cdot \langle 1 \rangle;
\]

\[
\text{if}(x_D = 0 \land x_P = 0)\{
  x_G \approx 0.99 \cdot \langle 0 \rangle + 0.01 \cdot \langle 1 \rangle;
  \\
\}
\]

\[
\text{else} \{
  x_G \approx 0.01 \cdot \langle 0 \rangle + 0.99 \cdot \langle 1 \rangle;
\}
\]

\[
\text{if}(x_G = 0)\{
  x_M \approx 0.9 \cdot \langle 0 \rangle + 0.1 \cdot \langle 1 \rangle;
\}
\]

\[
\text{else} \{
  x_M \approx 0.3 \cdot \langle 0 \rangle + 0.7 \cdot \langle 1 \rangle;
\}\]
Bayesian Network → Probabilistic Program

\[
x_D :\approx 0.95 \cdot \langle 0 \rangle + 0.05 \cdot \langle 1 \rangle;
\]
\[
x_P :\approx 0.98 \cdot \langle 0 \rangle + 0.02 \cdot \langle 1 \rangle;
\]
\[
\text{if}(x_D = 0 \land x_P = 0)\{
\quad x_G :\approx 0.99 \cdot \langle 0 \rangle + 0.01 \cdot \langle 1 \rangle;
\}\]
\[
\text{else}\{
\quad x_G :\approx 0.01 \cdot \langle 0 \rangle + 0.99 \cdot \langle 1 \rangle;
\}\]
\[
\text{if}(x_G = 0)\{
\quad x_M :\approx 0.9 \cdot \langle 0 \rangle + 0.1 \cdot \langle 1 \rangle;
\}\]
\[
\text{else}\{
\quad x_M :\approx 0.3 \cdot \langle 0 \rangle + 0.7 \cdot \langle 1 \rangle;
\}\]
Bayesian Network $\rightarrow$ Probabilistic Program

\[
x_D :\approx 0.95 \cdot \langle 0 \rangle + 0.05 \cdot \langle 1 \rangle; \\
x_P :\approx 0.98 \cdot \langle 0 \rangle + 0.02 \cdot \langle 1 \rangle; \\
\text{if}(x_D = 0 \land x_P = 0)\{ \\
\hspace{1em} x_G :\approx 0.99 \cdot \langle 0 \rangle + 0.01 \cdot \langle 1 \rangle; \\
\hspace{1em} \vdots \\
\} \text{ else } \{ \\
\hspace{1em} x_G :\approx 0.01 \cdot \langle 0 \rangle + 0.99 \cdot \langle 1 \rangle; \\
\text{if}(x_G = 0)\{ \\
\hspace{1em} x_M :\approx 0.9 \cdot \langle 0 \rangle + 0.1 \cdot \langle 1 \rangle; \\
\} \text{ else } \{ \\
\hspace{1em} x_M :\approx 0.3 \cdot \langle 0 \rangle + 0.7 \cdot \langle 1 \rangle; \\
\text{if}(x_G = 0)\{ \\
\hspace{1em} x_M :\approx 0.9 \cdot \langle 0 \rangle + 0.1 \cdot \langle 1 \rangle; \\
\} \text{ else } \{ \\
\hspace{1em} x_M :\approx 0.3 \cdot \langle 0 \rangle + 0.7 \cdot \langle 1 \rangle; \\
\text{else} \{ \\
\hspace{1em} x_M :\approx 0.3 \cdot \langle 0 \rangle + 0.7 \cdot \langle 1 \rangle; \\
\}
\]

Conditioning: $G \overset{\perp}{=} 0$ and $P \overset{\perp}{=} 0$
Bayesian Network $\rightarrow$ Probabilistic Program

repeat {
    $x_D :\approx 0.95 \cdot \langle 0 \rangle + 0.05 \cdot \langle 1 \rangle$;
    $x_P :\approx 0.98 \cdot \langle 0 \rangle + 0.02 \cdot \langle 1 \rangle$;
    if ($x_D = 0 \land x_P = 0$) {
        $x_G :\approx 0.99 \cdot \langle 0 \rangle + 0.01 \cdot \langle 1 \rangle$;
    } else {
        $x_G :\approx 0.01 \cdot \langle 0 \rangle + 0.99 \cdot \langle 1 \rangle$;
    }
    if ($x_G = 0$) {
        $x_M :\approx 0.9 \cdot \langle 0 \rangle + 0.1 \cdot \langle 1 \rangle$;
    } else {
        $x_M :\approx 0.3 \cdot \langle 0 \rangle + 0.7 \cdot \langle 1 \rangle$;
    }
} until ($x_G = 0 \land x_P = 0$)
Translation: Bayesian Network to Probabilistic Program

Our translation $\text{BN} \rightarrow \text{C}$ is correct: Distribution established by $\text{C}$ corresponds to the conditional distribution of $\text{BN}$ given observed evidence. Executing $\text{C}$ corresponds to sampling $\text{BN}$: Running $\text{C}$ once corresponds to obtaining a single sample from $\text{BN}$ that satisfies the observed evidence. Expected runtime of $\text{C}$ corresponds to expected time to obtain a single sample from $\text{BN}$ using rejection sampling. The repeat...until-loop in $\text{C}$ is always t-i.i.d for any $t$. Our closed form proof rule applies to $\text{C}$! Formal analysis of a probabilistic program yields expected sampling time of a Bayesian network.
Translation: Bayesian Network to Probabilistic Program

- Our translation $BN \rightarrow C$ is correct:
Our translation $BN \rightarrow C$ is correct:

- Distribution established by $C$ corresponds to the conditional distribution of $BN$ given observed evidence.

Executing $C$ corresponds to sampling $BN$:

- Running $C$ once corresponds to obtaining a single sample from $BN$ that satisfies the observed evidence.

Expected runtime of $C$ corresponds to expected time to obtain a single sample from $BN$ using rejection sampling.

The repeat... until loop in $C$ is always t.i.i.d. for any $t$.

Our closed form proof rule applies to $C$.

Formal analysis of a probabilistic program yields expected sampling time of a Bayesian network.
Translation: Bayesian Network to Probabilistic Program

- Our translation $\text{BN} \rightarrow \text{C}$ is correct:
  - Distribution established by $\text{C}$ corresponds to the conditional distribution of $\text{BN}$ given observed evidence
  - Executing $\text{C}$ corresponds to sampling $\text{BN}$:
Translation: Bayesian Network to Probabilistic Program

- Our translation $BN \rightarrow C$ is correct:
  - Distribution established by $C$ corresponds to the conditional distribution of $BN$ given observed evidence
  - Executing $C$ corresponds to sampling $BN$:
    - Running $C$ once corresponds to obtaining a single sample from $BN$ that satisfies the observed evidence
Translation: Bayesian Network to Probabilistic Program

- Our translation $BN \rightarrow C$ is correct:
  - Distribution established by $C$ corresponds to the conditional distribution of $BN$ given observed evidence
  - Executing $C$ corresponds to sampling $BN$:
    - Running $C$ once corresponds to obtaining a single sample from $BN$ that satisfies the observed evidence
    - **Expected runtime of $C$** corresponds to expected time to obtain a single sample from $BN$ using rejection sampling
Translation: Bayesian Network to Probabilistic Program

- Our translation $BN \rightarrow C$ is correct:
  - Distribution established by $C$ corresponds to the conditional distribution of $BN$ given observed evidence

- Executing $C$ corresponds to sampling $BN$:
  - Running $C$ once corresponds to obtaining a single sample from $BN$ that satisfies the observed evidence
  - **Expected runtime of $C$** corresponds to expected time to obtain a single sample from $BN$ using rejection sampling

- The repeat...until–loop in $C$ is always $t$–i.i.d. for any $t$
Translation: Bayesian Network to Probabilistic Program

- Our translation $BN \rightarrow C$ is correct:
  - Distribution established by $C$ corresponds to the conditional distribution of $BN$ given observed evidence
  - Executing $C$ corresponds to sampling $BN$:
    - Running $C$ once corresponds to obtaining a single sample from $BN$ that satisfies the observed evidence
    - Expected runtime of $C$ corresponds to expected time to obtain a single sample from $BN$ using rejection sampling
  - The repeat... until-loop in $C$ is always t-i.i.d. for any $t$
  - Our closed form proof rule applies to $C$!
Translation: Bayesian Network to Probabilistic Program

- Our translation $BN \rightarrow C$ is *correct*:
  - Distribution established by $C$ corresponds to the *conditional* distribution of $BN$ given observed evidence
  - Executing $C$ corresponds to *sampling* $BN$:
    - Running $C$ once corresponds to obtaining a *single sample* from $BN$ that satisfies the observed evidence
    - *Expected runtime* of $C$ corresponds to expected time to obtain a single sample from $BN$ using rejection sampling
  - The repeat...until-loop in $C$ is always *t-i.i.d.* for any $t$
  - Our closed form proof rule applies to $C$!
  - Formal analysis of a probabilistic program yields expected sampling time of a Bayesian network.
<table>
<thead>
<tr>
<th>BN</th>
<th>#obs</th>
<th>Time</th>
<th>EST</th>
<th>#obs</th>
<th>Time</th>
<th>EST</th>
</tr>
</thead>
<tbody>
<tr>
<td>hailfinder</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>#nodes: 56, #edges: 66, avg. Markov Blanket: 3.54</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>0.63</td>
<td>$5.016 \cdot 10^5$</td>
<td></td>
<td>9</td>
<td>0.46</td>
<td>$9.048 \cdot 10^6$</td>
</tr>
<tr>
<td>hepar2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>#nodes: 70, #edges: 123, avg. Markov Blanket: 4.51</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1.84</td>
<td>$1.579 \cdot 10^2$</td>
<td></td>
<td>2</td>
<td>MO</td>
<td>—</td>
</tr>
<tr>
<td>win95pts</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>#nodes: 76, #edges: 112, avg. Markov Blanket: 5.92</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0.36</td>
<td>$2.284 \cdot 10^3$</td>
<td></td>
<td>3</td>
<td>0.36</td>
<td>$4.296 \cdot 10^5$</td>
</tr>
<tr>
<td>12</td>
<td>0.42</td>
<td>$3.973 \cdot 10^7$</td>
<td></td>
<td>17</td>
<td>61.73</td>
<td>$1.110 \cdot 10^{15}$</td>
</tr>
<tr>
<td>pathfinder</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>#nodes: 135, #edges: 200, avg. Markov Blanket: 3.04</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0.53</td>
<td>$1.050 \cdot 10^4$</td>
<td></td>
<td>3</td>
<td>31.31</td>
<td>$2.872 \cdot 10^4$</td>
</tr>
<tr>
<td>7</td>
<td>5.44</td>
<td>$\infty$</td>
<td></td>
<td>7</td>
<td>480.83</td>
<td>$\infty$</td>
</tr>
<tr>
<td>andes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>#nodes: 223, #edges: 338, avg. Markov Blanket: 5.61</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>MO</td>
<td>—</td>
<td></td>
<td>3</td>
<td>1.66</td>
<td>$5.251 \cdot 10^3$</td>
</tr>
<tr>
<td>7</td>
<td>0.99</td>
<td>$8.904 \cdot 10^4$</td>
<td></td>
<td>9</td>
<td>0.90</td>
<td>$6.637 \cdot 10^5$</td>
</tr>
<tr>
<td>pigs</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>#nodes: 441, #edges: 592, avg. Markov Blanket: 3.66</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0.74</td>
<td>$2.952 \cdot 10^3$</td>
<td></td>
<td>3</td>
<td>0.88</td>
<td>$2.362 \cdot 10^3$</td>
</tr>
<tr>
<td>7</td>
<td>1.02</td>
<td>$1.511 \cdot 10^6$</td>
<td></td>
<td>8</td>
<td>MO</td>
<td>—</td>
</tr>
<tr>
<td>munin</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>#nodes: 1041, #edges: 1397, avg. Markov Blanket: 3.54</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1.47</td>
<td>$3.648 \cdot 10^4$</td>
<td></td>
<td>3</td>
<td>1.37</td>
<td>$1.824 \cdot 10^7$</td>
</tr>
<tr>
<td>9</td>
<td>1.79</td>
<td>$1.824 \cdot 10^{16}$</td>
<td></td>
<td>10</td>
<td>65.64</td>
<td>$1.153 \cdot 10^{18}$</td>
</tr>
</tbody>
</table>

bnlearn.com/bnrepository
Benjamin Lucien Kaminski
How long, O Bayesian network, will I sample thee?
16.4.2018
<table>
<thead>
<tr>
<th>BN</th>
<th>#obs</th>
<th>Time</th>
<th>EST</th>
<th>#obs</th>
<th>Time</th>
<th>EST</th>
</tr>
</thead>
<tbody>
<tr>
<td>hailfinder</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>hepar2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>win95pts</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pathfinder</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>andes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pigs</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>munin</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

bnlearn.com/bnrepository
<table>
<thead>
<tr>
<th>BN</th>
<th>#obs</th>
<th>Time</th>
<th>EST</th>
<th>#obs</th>
<th>Time</th>
<th>EST</th>
</tr>
</thead>
<tbody>
<tr>
<td>hailfinder</td>
<td>5</td>
<td>0.63</td>
<td>5.016 × 10^5</td>
<td>9</td>
<td>0.46</td>
<td>9.048 × 10^6</td>
</tr>
<tr>
<td>hep2</td>
<td>1</td>
<td>1.84</td>
<td>1.579 × 10^2</td>
<td>2</td>
<td>MO</td>
<td>—</td>
</tr>
<tr>
<td>win95pts</td>
<td>1</td>
<td>0.36</td>
<td>2.284 × 10^3</td>
<td>3</td>
<td>0.36</td>
<td>4.296 × 10^5</td>
</tr>
<tr>
<td></td>
<td>12</td>
<td>0.42</td>
<td>3.973 × 10^7</td>
<td>3</td>
<td>0.36</td>
<td>4.296 × 10^5</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>5.14</td>
<td>∞</td>
<td>7</td>
<td>480.83</td>
<td>∞</td>
</tr>
<tr>
<td>pathfinder</td>
<td>1</td>
<td>0.53</td>
<td>1.050 × 10^4</td>
<td>3</td>
<td>31.31</td>
<td>2.872 × 10^4</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>5.14</td>
<td>∞</td>
<td>7</td>
<td>480.83</td>
<td>∞</td>
</tr>
<tr>
<td>andes</td>
<td>1</td>
<td>6.30</td>
<td>—</td>
<td>3</td>
<td>1.66</td>
<td>5.251 × 10^3</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>0.99</td>
<td>2.904 × 10^1</td>
<td>9</td>
<td>0.90</td>
<td>6.637 × 10^5</td>
</tr>
<tr>
<td>pigs</td>
<td>1</td>
<td>0.74</td>
<td>2.952 × 10^3</td>
<td>3</td>
<td>6.88</td>
<td>2.362 × 10^3</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>1.09</td>
<td>1.511 × 10^6</td>
<td>8</td>
<td>MO</td>
<td>—</td>
</tr>
<tr>
<td>munin</td>
<td>1</td>
<td>1.47</td>
<td>3.648 × 10^4</td>
<td>3</td>
<td>1.37</td>
<td>1.824 × 10^7</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>1.79</td>
<td>1.824 × 10^16</td>
<td>10</td>
<td>65.64</td>
<td>1.153 × 10^18</td>
</tr>
</tbody>
</table>

bnlearn.com/bnrepository
## Application to Bayesian Networks

### Evaluation

<table>
<thead>
<tr>
<th>BN</th>
<th>#obs</th>
<th>Time</th>
<th>EST</th>
<th>#obs</th>
<th>Time</th>
<th>EST</th>
</tr>
</thead>
<tbody>
<tr>
<td>hailfinder</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>0.63</td>
<td>$5.016 \cdot 10^5$</td>
<td>9</td>
<td>0.46</td>
<td>$9.048 \cdot 10^6$</td>
</tr>
<tr>
<td>hepar2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1.84</td>
<td>$1.579 \cdot 10^2$</td>
<td>2</td>
<td>MO</td>
<td>—</td>
</tr>
<tr>
<td>win95pts</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0.36</td>
<td>$2.284 \cdot 10^3$</td>
<td>3</td>
<td>0.36</td>
<td>$4.296 \cdot 10^5$</td>
</tr>
<tr>
<td></td>
<td>12</td>
<td>0.42</td>
<td>$3.973 \cdot 10^7$</td>
<td>17</td>
<td>61.73</td>
<td>$1.110 \cdot 10^{15}$</td>
</tr>
<tr>
<td>pathfinder</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0.53</td>
<td>$1.050 \cdot 10^4$</td>
<td>3</td>
<td>31.31</td>
<td>$2.872 \cdot 10^4$</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>5.44</td>
<td>$\infty$</td>
<td>7</td>
<td>480.83</td>
<td>$\infty$</td>
</tr>
<tr>
<td>andes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>MO</td>
<td>—</td>
<td>3</td>
<td>1.66</td>
<td>$5.251 \cdot 10^3$</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>0.99</td>
<td>$8.904 \cdot 10^4$</td>
<td>9</td>
<td>0.90</td>
<td>$6.637 \cdot 10^5$</td>
</tr>
<tr>
<td>pigs</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0.74</td>
<td>$2.952 \cdot 10^3$</td>
<td>3</td>
<td>0.88</td>
<td>$2.362 \cdot 10^3$</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>1.02</td>
<td>$1.511 \cdot 10^6$</td>
<td>8</td>
<td>MO</td>
<td>—</td>
</tr>
<tr>
<td>munin</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1.47</td>
<td>$3.648 \cdot 10^4$</td>
<td>3</td>
<td>1.37</td>
<td>$1.824 \cdot 10^7$</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>1.79</td>
<td>$1.824 \cdot 10^{16}$</td>
<td>10</td>
<td>65.64</td>
<td>$1.153 \cdot 10^{18}$</td>
</tr>
</tbody>
</table>

[bnlearn.com/bnrepository](bnlearn.com/bnrepository)

Benjamin Lucien Kaminski

How long, O Bayesian network, will I sample thee?

16.4.2018
<table>
<thead>
<tr>
<th>BN</th>
<th>#obs</th>
<th>Time</th>
<th>EST</th>
<th>#obs</th>
<th>Time</th>
<th>EST</th>
</tr>
</thead>
<tbody>
<tr>
<td>hailfinder</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>hepar2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>win95pts</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pathfinder</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>andes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pigs</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>munin</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BN</td>
<td>#obs</td>
<td>Time</td>
<td>EST</td>
<td>#obs</td>
<td>Time</td>
<td>EST</td>
</tr>
<tr>
<td>----------</td>
<td>------</td>
<td>----------</td>
<td>----------</td>
<td>------</td>
<td>----------</td>
<td>----------</td>
</tr>
<tr>
<td>hailfinder</td>
<td>5</td>
<td>0.63</td>
<td>$5.016 \cdot 10^5$</td>
<td>9</td>
<td>0.46</td>
<td>$9.048 \cdot 10^6$</td>
</tr>
<tr>
<td>#nodes: 56, #edges: 66, avg. Markov Blanket: 3.54</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>hepar2</td>
<td>1</td>
<td>1.84</td>
<td>$1.579 \cdot 10^2$</td>
<td>2</td>
<td>MO</td>
<td>—</td>
</tr>
<tr>
<td>#nodes: 70, #edges: 123, avg. Markov Blanket: 4.51</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>win95pts</td>
<td>1</td>
<td>0.36</td>
<td>$2.284 \cdot 10^3$</td>
<td>3</td>
<td>0.36</td>
<td>$4.296 \cdot 10^5$</td>
</tr>
<tr>
<td>12</td>
<td>0.42</td>
<td>$3.973 \cdot 10^7$</td>
<td>17</td>
<td>61.73</td>
<td>$1.110 \cdot 10^{15}$</td>
<td></td>
</tr>
<tr>
<td>#nodes: 76, #edges: 112, avg. Markov Blanket: 5.92</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pathfinder</td>
<td>1</td>
<td>0.53</td>
<td>$1.050 \cdot 10^4$</td>
<td>3</td>
<td>31.31</td>
<td>$2.872 \cdot 10^4$</td>
</tr>
<tr>
<td>7</td>
<td>5.44</td>
<td>$\infty$</td>
<td>7</td>
<td>480.83</td>
<td>$\infty$</td>
<td></td>
</tr>
<tr>
<td>#nodes: 135, #edges: 200, avg. Markov Blanket: 3.04</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>andes</td>
<td>1</td>
<td>MO</td>
<td>—</td>
<td>3</td>
<td>1.66</td>
<td>$5.251 \cdot 10^3$</td>
</tr>
<tr>
<td>7</td>
<td>0.99</td>
<td>$8.904 \cdot 10^4$</td>
<td>9</td>
<td>0.90</td>
<td>$6.637 \cdot 10^5$</td>
<td></td>
</tr>
<tr>
<td>#nodes: 223, #edges: 338, avg. Markov Blanket: 5.61</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pigs</td>
<td>1</td>
<td>0.74</td>
<td>$2.952 \cdot 10^3$</td>
<td>3</td>
<td>0.88</td>
<td>$2.362 \cdot 10^3$</td>
</tr>
<tr>
<td>7</td>
<td>1.02</td>
<td>$1.511 \cdot 10^6$</td>
<td>8</td>
<td>MO</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>#nodes: 441, #edges: 592, avg. Markov Blanket: 3.66</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>munin</td>
<td>1</td>
<td>1.47</td>
<td>$3.648 \cdot 10^4$</td>
<td>3</td>
<td>1.37</td>
<td>$1.824 \cdot 10^7$</td>
</tr>
<tr>
<td>9</td>
<td>1.79</td>
<td>$1.824 \cdot 10^{16}$</td>
<td>10</td>
<td>65.64</td>
<td>$1.153 \cdot 10^{18}$</td>
<td></td>
</tr>
<tr>
<td>#nodes: 1041, #edges: 1397, avg. Markov Blanket: 3.54</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

bnlearn.com/bnrepository
<table>
<thead>
<tr>
<th>BN</th>
<th>#obs</th>
<th>Time</th>
<th>EST</th>
<th>#obs</th>
<th>Time</th>
<th>EST</th>
</tr>
</thead>
<tbody>
<tr>
<td>hailfinder</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>0.63</td>
<td>$5.016 \cdot 10^5$</td>
<td>9</td>
<td>0.46</td>
<td>$9.048 \cdot 10^6$</td>
</tr>
<tr>
<td>hepar2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1.84</td>
<td>$1.579 \cdot 10^2$</td>
<td>2</td>
<td>MO</td>
<td>—</td>
</tr>
<tr>
<td>win95pts</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0.36</td>
<td>$2.284 \cdot 10^3$</td>
<td>3</td>
<td>0.36</td>
<td>$4.296 \cdot 10^5$</td>
</tr>
<tr>
<td></td>
<td>12</td>
<td>0.42</td>
<td>$3.973 \cdot 10^7$</td>
<td>17</td>
<td>61.73</td>
<td>$1.110 \cdot 10^{15}$</td>
</tr>
<tr>
<td>pathfinder</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0.53</td>
<td>$1.050 \cdot 10^4$</td>
<td>3</td>
<td>31.31</td>
<td>$2.872 \cdot 10^4$</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>5.44</td>
<td>$\infty$</td>
<td>7</td>
<td>480.83</td>
<td>$\infty$</td>
</tr>
<tr>
<td>andes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>MO</td>
<td>—</td>
<td>3</td>
<td>1.66</td>
<td>$5.251 \cdot 10^3$</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>0.99</td>
<td>$8.904 \cdot 10^4$</td>
<td>9</td>
<td>0.90</td>
<td>$6.637 \cdot 10^5$</td>
</tr>
<tr>
<td>pigs</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0.74</td>
<td>$2.952 \cdot 10^3$</td>
<td>3</td>
<td>0.88</td>
<td>$2.362 \cdot 10^3$</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>1.02</td>
<td>$1.511 \cdot 10^6$</td>
<td>8</td>
<td>MO</td>
<td>—</td>
</tr>
<tr>
<td>munin</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1.47</td>
<td>$3.648 \cdot 10^4$</td>
<td>3</td>
<td>1.37</td>
<td>$1.824 \cdot 10^7$</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>1.79</td>
<td>$1.824 \cdot 10^{16}$</td>
<td>10</td>
<td>65.64</td>
<td>$1.153 \cdot 10^{18}$</td>
</tr>
</tbody>
</table>

bnlearn.com/bnrepository
## Application to Bayesian Networks

### Evaluation

<table>
<thead>
<tr>
<th>BN</th>
<th>#obs</th>
<th>Time</th>
<th>EST</th>
<th>#obs</th>
<th>Time</th>
<th>EST</th>
</tr>
</thead>
<tbody>
<tr>
<td>hailfinder</td>
<td>5</td>
<td>0.63</td>
<td>$5.016 \cdot 10^5$</td>
<td>9</td>
<td>0.46</td>
<td>$9.048 \cdot 10^6$</td>
</tr>
<tr>
<td>#nodes: 56, #edges: 66, avg. Markov Blanket: 3.54</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>hepar2</td>
<td>1</td>
<td>1.84</td>
<td>$1.579 \cdot 10^2$</td>
<td>2</td>
<td>MO</td>
<td>—</td>
</tr>
<tr>
<td>#nodes: 70, #edges: 123, avg. Markov Blanket: 4.51</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>win95pts</td>
<td>1</td>
<td>0.36</td>
<td>$2.284 \cdot 10^3$</td>
<td>3</td>
<td>0.36</td>
<td>$4.296 \cdot 10^5$</td>
</tr>
<tr>
<td>12</td>
<td>0.42</td>
<td>$3.973 \cdot 10^7$</td>
<td>17</td>
<td>61.73</td>
<td>$1.110 \cdot 10^{15}$</td>
<td></td>
</tr>
<tr>
<td>#nodes: 76, #edges: 112, avg. Markov Blanket: 5.92</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pathfinder</td>
<td>1</td>
<td>0.53</td>
<td>$1.050 \cdot 10^4$</td>
<td>3</td>
<td>31.31</td>
<td>$2.872 \cdot 10^4$</td>
</tr>
<tr>
<td>7</td>
<td>5.44</td>
<td>$\infty$</td>
<td>7</td>
<td>480.83</td>
<td>$\infty$</td>
<td></td>
</tr>
<tr>
<td>#nodes: 135, #edges: 200, avg. Markov Blanket: 3.04</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>andes</td>
<td>1</td>
<td>MO</td>
<td>—</td>
<td>3</td>
<td>1.66</td>
<td>$5.251 \cdot 10^3$</td>
</tr>
<tr>
<td>7</td>
<td>0.99</td>
<td>$8.904 \cdot 10^4$</td>
<td>9</td>
<td>0.90</td>
<td>$6.637 \cdot 10^5$</td>
<td></td>
</tr>
<tr>
<td>#nodes: 223, #edges: 338, avg. Markov Blanket: 5.61</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pigs</td>
<td>1</td>
<td>0.74</td>
<td>$2.952 \cdot 10^3$</td>
<td>3</td>
<td>0.88</td>
<td>$2.362 \cdot 10^3$</td>
</tr>
<tr>
<td>7</td>
<td>1.02</td>
<td>$1.511 \cdot 10^6$</td>
<td>8</td>
<td>MO</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>#nodes: 441, #edges: 592, avg. Markov Blanket: 3.66</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>munin</td>
<td>1</td>
<td>1.47</td>
<td>$3.648 \cdot 10^4$</td>
<td>3</td>
<td>1.37</td>
<td>$1.824 \cdot 10^7$</td>
</tr>
<tr>
<td>9</td>
<td>1.79</td>
<td>$1.824 \cdot 10^{16}$</td>
<td>10</td>
<td>65.64</td>
<td>$1.153 \cdot 10^{18}$</td>
<td></td>
</tr>
<tr>
<td>#nodes: 1041, #edges: 1397, avg. Markov Blanket: 3.54</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

[bnlearn.com/bnrepository](bnlearn.com/bnrepository)

Benjamin Lucien Kaminski

How long, O Bayesian network, will I sample thee?

16.4.2018
Summary

The program analysis perspective:

The Bayesian Network perspective:
Summary

The program analysis perspective:

- A fairly checkable condition (t–i.i.d.–ness)

The Bayesian Network perspective:
Summary

The program analysis perspective:

- A fairly checkable condition (t–i.i.d.–ness)
- A proof rule yielding exact expected runtimes.

The Bayesian Network perspective:
Summary

The program analysis perspective:

- A fairly checkable condition (t-i.i.d.-ness)
- A proof rule yielding exact expected runtimes.
  No need for guessing invariants, supermartingales, or alike

The Bayesian Network perspective:
Summary

The program analysis perspective:

- A fairly checkable condition (t-i.i.d.-ness)
- A proof rule yielding exact expected runtimes. No need for guessing invariants, supermartingales, or alike
- A syntactic language fragment of probabilistic programs to which our proof rule automatically applies

The Bayesian Network perspective:
Summary

The program analysis perspective:

- A fairly checkable condition (t–i.i.d.–ness)
- A proof rule yielding exact expected runtimes.
  No need for guessing invariants, supermartingales, or alike
- A syntactic language fragment of probabilistic programs
to which our proof rule automatically applies

The Bayesian Network perspective:

- A method to obtain expected sampling times
  using techniques from program analysis
Summary

The program analysis perspective:

- A fairly checkable condition (t–i.i.d.–ness)
- A proof rule yielding exact expected runtimes.
  No need for guessing invariants, supermartingales, or alike
- A syntactic language fragment of probabilistic programs
to which our proof rule automatically applies

The Bayesian Network perspective:

- A method to obtain expected sampling times
  using techniques from program analysis
- We are not solving an NP–hard problem more efficiently
Summary

The program analysis perspective:

- A fairly checkable condition ($t$–i.i.d.–ness)
- A proof rule yielding exact expected runtimes. No need for guessing invariants, supermartingales, or alike
- A syntactic language fragment of probabilistic programs to which our proof rule automatically applies

The Bayesian Network perspective:

- A method to obtain expected sampling times using techniques from program analysis
- We are not solving an NP–hard problem more efficiently
- We have shown that our method works on “very large” and even “massive” Bayesian networks
Backup: \( t \)-i.i.d.-ness
Backup: \textit{t--i.i.d.--ness}

\[ \text{Vars}(t) = \{ x \mid \exists \sigma, v, v': t(\sigma [x \mapsto v]) \neq t(\sigma [x \mapsto v']) \} \]
Backup: \( t\text{-i.i.d.-ness} \)

- \( \text{Vars}(t) = \{ x \mid \exists \sigma, v, v': t(\sigma \[ x \mapsto v \]) \neq t(\sigma \[ x \mapsto v' \]) \} \)
- \( \text{Mod}(C) \): set of variables that occur on left hand side of an assignment in program \( C \)
Backup: \( t\text{-i.i.d.-ness} \)

- \( \text{Vars}(t) = \{ x \mid \exists \sigma, v, v': t(\sigma[x \mapsto v]) \neq t(\sigma[x \mapsto v']) \} \)
- \( \text{Mod}(C) \): set of variables that occur on left hand side of an assignment in program \( C \)
- \( t \not\sqsubset C \iff \text{Vars}(f) \cap \text{Mod}(C) = \emptyset \)
Backup: $t$–i.i.d.–ness

- $\text{Vars}(t) = \{ x \mid \exists \sigma, v, v': t(\sigma [x \mapsto v]) \neq t(\sigma [x \mapsto v']) \}$

- $\text{Mod}(C)$: set of variables that occur on left hand side of an assignment in program $C$

- $t \not\models C$ iff $\text{Vars}(f) \cap \text{Mod}(C) = \emptyset$

---

$f$–i.i.d.–ness

The loop $\text{while}(\varphi)\{C\}$ is called $t$–i.i.d. iff
Backup:  $t$–i.i.d.–ness

- $\text{Vars}(t) = \{ x \mid \exists \sigma, v, v': t(\sigma[x \mapsto v]) \neq t(\sigma[x \mapsto v']) \}$

- $\text{Mod}(C)$: set of variables that occur on left hand side of an assignment in program $C$

- $t \not\sqcap C$ iff $\text{Vars}(f) \cap \text{Mod}(C) = \emptyset$

---

$f$–i.i.d.–ness

The loop $\text{while} (\varphi) \{ C \}$ is called $t$–i.i.d. iff

$$\wp [C] ([\varphi]) \not\sqcap C \quad \text{and} \quad \wp [C] ([\neg \varphi] \cdot t) \not\sqcap C.$$
Backup: The Coupon Collector’s Inner Loop is \( i \)-i.i.d.

\[
\text{while} \ (cp[i] \neq 0) \{
    i \sim \text{Unif}[1 \ldots N]
\}
\]
Backup: The Coupon Collector’s Inner Loop is \( i \)-i.i.d.

\[
\text{while}(cp[i] \neq 0)\
\quad \begin{align*}
&i \ dispos \ Unif[1 \ldots N] \\
&\end{align*}
\]

- The coupon collector’s inner loop is \( i \)-i.i.d.
Backup: The Coupon Collector’s Inner Loop is $i$–i.i.d.

\[ \text{while}(cp[i] \neq 0) \{
    i \approx \text{Unif}[1 \ldots N]
\} \]

- The coupon collector’s inner loop is $i$–i.i.d.
  - \[ \text{wp } [i \approx \ldots] ([cp[i] \neq 0]) = \frac{1}{N} \sum_{k=1}^{N} [cp[k] \neq 0] \not\subset C \]
Backup: The Coupon Collector’s Inner Loop is $i$–i.i.d.

\[
\text{while}(cp[i] \neq 0) \{
    i : \approx \text{Unif}[1 \ldots N]
\}
\]

- The coupon collector’s inner loop is $i$–i.i.d.
  - \( \text{wp} \ [i : \approx \ldots] \ (cp[i] \neq 0) = \frac{1}{N} \sum_{k=1}^{N} [cp[k] \neq 0] \not\supseteq C \)
  - \( \text{wp} \ [i : \approx \ldots] \ (cp[i] = 0) \cdot i = \frac{1}{N} \sum_{k=1}^{N} [cp[k] = 0] \cdot k \not\supseteq C \)
Backup: The Coupon Collector’s Inner Loop is \( i \)-i.i.d.

\[
\text{while}( cp[i] \neq 0 ) \{
    i \sim \text{Unif}[1 \ldots N]
\}
\]

- The coupon collector’s inner loop is \( i \)-i.i.d.
  - \( \text{wp} \left[ i \sim \ldots \right] \left( [cp[i] \neq 0] \right) = \frac{1}{N} \sum_{k=1}^{N} [cp[k] \neq 0] \not\in C \)
  - \( \text{wp} \left[ i \sim \ldots \right] \left( [cp[i] = 0] \cdot i \right) = \frac{1}{N} \sum_{k=1}^{N} [cp[k] = 0] \cdot k \not\in C \)
- **In fact:** The inner loop is \( f \)-i.i.d. for any \( f \)
Backup: The Bayesian Network Language

\[
C \rightarrow \text{Seq} \mid \text{repeat}\{\psi\}\text{until} (\text{Seq}) \mid C ; C
\]

\[
\text{Seq} \rightarrow \text{Seq} ; \text{Seq} \mid B_{x_1} \mid B_{x_2} \mid \ldots
\]

\[
B_{x_i} \rightarrow x_i \approx \sum_{j=1}^{n} p_j \cdot \langle a_j \rangle \mid \text{if } (\varphi) \{ x_i \approx \mu \} \text{ else } \{ B_{x_i} \}
\]

(rule exists for all \( x_i \in \text{Vars} \))