Synthesizing Probabilistic Invariants via Doob's decomposition

G. Barthe, *T. Espitau*, L.M.F Fioriti, J. Hsu

LIP6, 2016

Introduction

Introduction

1 | Probabilistic Computations

Ubiquitous in many fields
(ML, Crypto, Privacy,…)

But… Difficult to prove

Termination ?
(Certainly, almost sure, non terminating)

Introduction

1 | Probabilistic Computations

Ubiquitous in many fields
(ML, Crypto, Privacy,…)

But… Difficult to prove

Termination ?
(Certainly, almost sure, non terminating)

2 | Martingales?

Difficulty to transfer local to end of program

Reason on average values

Martingales have the required transfer property

Introduction

1 | Probabilistic Computations

Ubiquitous in many fields
(ML, Crypto, Privacy,…)

But… Difficult to prove

Termination ?
(Certainly, almost sure, non terminating)

2 | Martingales?

Difficulty to transfer local to end of program

Reason on average values

Martingales have the required transfer property

3 | Doob's Decomposition

But (*again*)… Difficult to find good ones

Automated generation?

Doob's decomposition
Formal method to generate martingales from a *seed*.

Martingale theory 101 (I)

Martingale theory 101 (I)

Step 1: Some probabilities

Probability space

- Ω set of outcomes.

- Sigma algebra:

  Set F of subsets of Ω

  Closed under complements,

  countable unions,

  countable intersections.

- Probability measure:

  Countably additive mapping $\mathbf{P} : F \longrightarrow [0, 1]$

  $\mathbf{P}(\Omega) = 1.$

Martingale theory 101 (II)

Step 2: Stochastic process

- Random variable:

  $X : \Omega \rightarrow \mathbf{R}$ measurable ($X^{-1}(\,(a,b]\,) \in F$)

- Filtration: $(F_i) \subset F$ s.t:

  $F_{i-1} \subset F_i$

- Process wrt filtration F_i:

  Sequence (X_i) s.t:

  X_i is F_i measurable

Martingale theory 101 (II)

Interlude: PL setting

Ω: *Element* = Possible outcome of *samples*

F_i: Events sampled at iteration i or before

Process (X_i) is adapted to the filtration iff:

```
i = 0

While b do

z[i] ←$ Samplings...

x[i] ← f(x[i-1], ... , f[0], z[i], ... ,z[0])

i++
end
```

X_i is defined in term of elements sampled at step i or before

Martingale theory 101 (III)

Step 3: Expectations & Moments

- Expectation:

$$\mathbf{E}[X] = \sum_{u \in \Omega} X(u)\, \mathbf{P}(u)$$

- Conditional expectation wrt $G \subset F$: $\mathbf{E}[X|G]$

Y G-mesurable st $\mathbf{E}[X.\mathbf{1}_A] = \mathbf{E}[Y.\mathbf{1}_A]$ for $A \in G$

Martingale theory 101 (IV)

Step 4 (Final!): Martingales

- Martingale:

$$\mathbf{E}\,[\,X_i \mid F_{i-1}\,] = X_{i-1}$$

Average value of the current step is equal to the value of the previous step

Playing with martingales

Doob's decomposition

(X_i) stochastic process $\longrightarrow$ (M_i) martingale

$$M_0 = X_0 \qquad\qquad M_i = X_0 + \sum_{j=1}^{i} X_j - \mathbf{E}\,[X_j \mid F_{j-1}]$$

Black Magic of martingales

Optional Stopping theorem

(M_i) martingale $\longrightarrow$ Expectations are invariants

$$\boxed{\mathbf{E}[M_j] = \mathbf{E}[M_0]}$$

Black Magic of martingales

Optional Stopping theorem

$$\boxed{\mathbf{E}[M_j] = \mathbf{E}[M_0]}$$

Black Magic of martingales

Optional Stopping theorem

$$\mathbf{E}[M_T] = \mathbf{E}[M_0]$$

For T a stopping time : $T : \Omega \to \mathbf{R}$ $\{ w \in \Omega \mid T(w) \leq i \} \subset F_i$

Black Magic of martingales

Optional Stopping theorem

$$\mathbf{E}[M_T] = \mathbf{E}[M_0]$$

For T a stopping time : $T : \Omega \to \mathbf{R}$ $\{ w \in \Omega \mid T(w) \leq i \} \subset F_i$

and...

$$|M_i - M_{i-1}| \leq C$$

$$\mathbf{E}[T] < \infty$$

Let’s play with a program…

Geometric distribution

$x[0] \leftarrow 0$;

while $(z \neq 0)$ **do**

$z \leftarrow\$ \text{Bern}(p, \{1, 0\})$;

$x \leftarrow x[-1] + z$;

end

Geometric distribution

```
x[0] ← 0;
while (z ≠ 0) do
z ←$ Bern(p, {1, 0});
x ← x[-1] + z;
end
```

Stopping time? (on average)

Geometric distribution

```
x[0] ← 0;
while (z ≠ 0) do
z ←$ Bern(p, {1, 0});
x ← x[-1] + z;
end
```

Stopping time? (on average)

$1/(1-p)$

Geometric distribution

Equation for x?

$$X_i = X_{i-1} + Z_i$$

```
x[0] ← 0;
while (z ≠ 0) do
z ←$ Bern(p, {1, 0});
x ← x[-1] + z;
end
```

Geometric distribution

Equation for x?

$X_i = X_{i-1} + Z_i$

```
x[0] ← 0;
while (z ≠ 0) do
z ←$ Bern(p, {1, 0});
x ← x[-1] + z;
end
```

Polynomial extraction

Geometric distribution

$$X_i = X_{i-1} + Z_i$$

```
x[0] ← 0;
while (z ≠ 0) do
z ←$ Bern(p, {1, 0});
x ← x[-1] + z;
end
```

$$M_0 = X_0 \qquad M_i = X_0 + \sum_{j=1}^{i} X_j - \mathbf{E}[X_j \mid F_{j-1}]$$

Doob

Geometric distribution

$$X_i = X_{i-1} + Z_i$$

```
x[0] ← 0;
while (z ≠ 0) do
z ←$ Bern(p, {1, 0});
x ← x[-1] + z;
end
```

$$M_0 = 0 \qquad M_i = X_0 + \sum_{j=1}^{i} X_j - \mathbf{E}[X_j \mid F_{j-1}]$$

Geometric distribution

$$X_i = X_{i-1} + Z_i$$

```
x[0] ← 0;
while (z ≠ 0) do
z ←$ Bern(p, {1, 0});
x ← x[-1] + z;
end
```

$$M_0 = 0 \qquad M_i = \sum_{j=1}^{i} X_j - \mathbf{E}[X_j \mid F_{j-1}]$$

Geometric distribution

$$X_i = X_{i-1} + Z_i$$

```
x[0] ← 0;
while (z ≠ 0) do
z ←$ Bern(p, {1, 0});
x ← x[-1] + z;
end
```

$$M_0 = 0 \qquad M_i = \sum_{j=1}^{i} X_j - \mathbf{E}[X_{j-1} + Z_i \mid F_{j-1}]$$

Geometric distribution

$$X_i = X_{i-1} + Z_i$$

```
x[0] ← 0;
while (z ≠ 0) do
z ←$ Bern(p, {1, 0});
x ← x[-1] + z;
end
```

$$M_0 = 0 \qquad M_i = \sum_{j=1}^{i} X_j - \mathbf{E}[X_{j-1} \mid F_{j-1}] + \mathbf{E}[Z_i \mid F_{j-1}]$$

Geometric distribution

$$X_i = X_{i-1} + Z_i$$

```
x[0] ← 0;
while (z ≠ 0) do
z ←$ Bern(p, {1, 0});
x ← x[-1] + z;
end
```

$$M_0 = 0 \qquad M_i = \sum_{j=1}^{i} X_j - \mathbf{E}[X_{j-1} \mid F_{j-1}] + \mathbf{E}[Z_i]$$

Geometric distribution

$$X_i = X_{i-1} + Z_i$$

```
x[0] ← 0;
while (z ≠ 0) do
z ←$ Bern(p, {1, 0});
x ← x[-1] + z;
end
```

$$M_0 = 0 \qquad M_i = \sum_{j=1}^{i} X_j - \mathbf{E}[X_{j-1} \mid F_{j-1}] + p$$

Geometric distribution

$$X_i = X_{i-1} + Z_i$$

```
x[0] ← 0;
while (z ≠ 0) do
z ←$ Bern(p, {1, 0});
x ← x[-1] + z;
end
```

$$M_0 = 0 \qquad M_i = \sum_{j=1}^{i} X_j - X_{j-1} + p$$

Geometric distribution

$$X_i = X_{i-1} + Z_i$$

```
x[0] ← 0;
while (z ≠ 0) do
z ←$ Bern(p, {1, 0});
x ← x[-1] + z;
end
```

$$M_0 = 0 \qquad M_i = X_i - X_0 + i\,p$$

Geometric distribution

$X_i = X_{i-1} + Z_i$

```
x[0] ← 0;
while (z ≠ 0) do
z ←$ Bern(p, {1, 0});
x ← x[-1] + z;
end
```

$M_0 = 0$ $\quad M_i = X_i + i\,p$

Simplify...

Geometric distribution

$$X_i = X_{i-1} + Z_i$$

$$M_0 = 0 \qquad M_i = X_i + i\,p$$

```
x[0] ← 0;
while (z ≠ 0) do
z ←$ Bern(p, {1, 0});
x ← x[-1] + z;
end
```

Geometric distribution

$X_i = X_{i-1} + Z_i$

$M_0 = 0$

$M_i = X_i + i\,p$

```
x[0] ← 0;
while (z ≠ 0) do
z ←$ Bern(p, {1, 0});
x ← x[-1] + z;
end
```

$\mathbf{E}[\,M_0\,] = \mathbf{E}[\,M_T]$

Optional Stopping

Geometric distribution

$$X_i = X_{i-1} + Z_i$$

$$M_0 = 0 \qquad M_i = X_i + i\,p$$

```
x[0] ← 0;
while (z ≠ 0) do
z ←$ Bern(p, {1, 0});
x ← x[-1] + z;
end
```

$$0 = \mathbf{E}[\, M_T]$$

Geometric distribution

$$X_i = X_{i-1} + Z_i$$

$$M_0 = 0 \qquad M_i = X_i + i\,p$$

```
x[0] ← 0;
while (z ≠ 0) do
z ←$ Bern(p, {1, 0});
x ← x[-1] + z;
end
```

$$\mathbf{0} = \mathbf{E}[\,X_T - Tp\,]$$

Geometric distribution

$X_i = X_{i-1} + Z_i$

$M_0 = 0$ $\quad M_i = X_i + i\,p$

```
x[0] ← 0;
while (z ≠ 0) do
z ←$ Bern(p, {1, 0});
x ← x[-1] + z;
end
```

$0 = \mathbf{E}[X_T] - \mathbf{E}[Tp]$

Geometric distribution

$X_i = X_{i-1} + Z_i$

$M_0 = 0 \qquad M_i = X_i + i\,p$

```
x[0] ← 0;
while (z ≠ 0) do
z ←$ Bern(p, {1, 0});
x ← x[-1] + z;
end
```

Simplify...

$\mathbf{0} = \mathbf{E}[X_T] - p\,\mathbf{E}[T]$

Geometric distribution

$$X_i = X_{i-1} + Z_i$$

$$M_0 = 0 \qquad M_i = X_i + i\,p$$

```
x[0] ← 0;
while (z ≠ 0) do
z ←$ Bern(p, {1, 0});
x ← x[-1] + z;
end
```

Hint

$$0 = \mathbf{E}[T-1] - p\,\mathbf{E}[T]$$

$$X_T = T-1$$

Geometric distribution

$X_i = X_{i-1} + Z_i$

$M_0 = 0 \qquad M_i = X_i + i\,p$

```
x[0] ← 0;
while (z ≠ 0) do
z ←$ Bern(p, {1, 0});
x ← x[-1] + z;
end
```

$0 = \mathbf{E}[T] - 1 - p\,\mathbf{E}[T]$

Geometric distribution

$X_i = X_{i-1} + Z_i$

$M_0 = 0 \qquad M_i = X_i + i\,p$

```
x[0] ← 0;
while (z ≠ 0) do
z ←$ Bern(p, {1, 0});
x ← x[-1] + z;
end
```

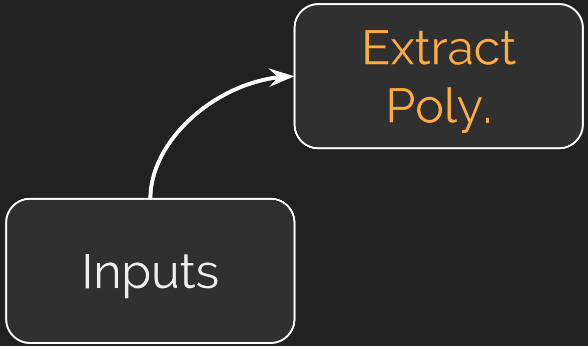
Simplify...

$\mathbf{E}[T] = 1/(1-p)$

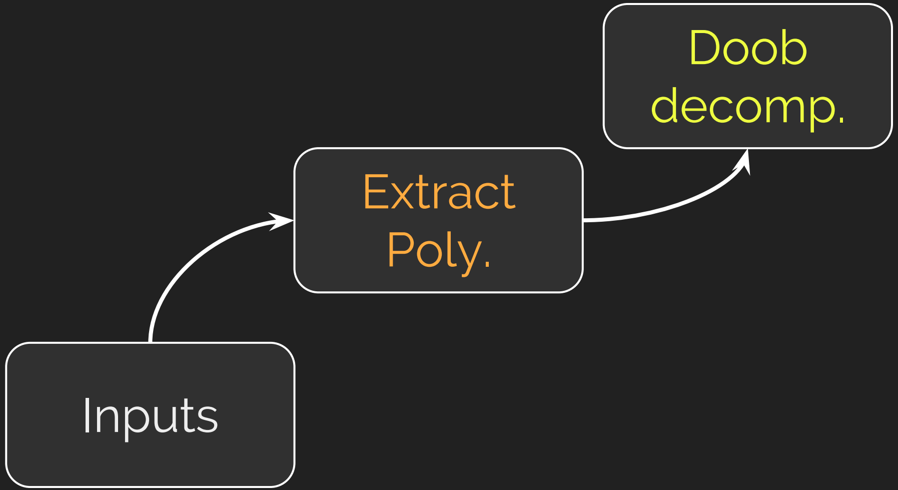
Automatization

Inputs

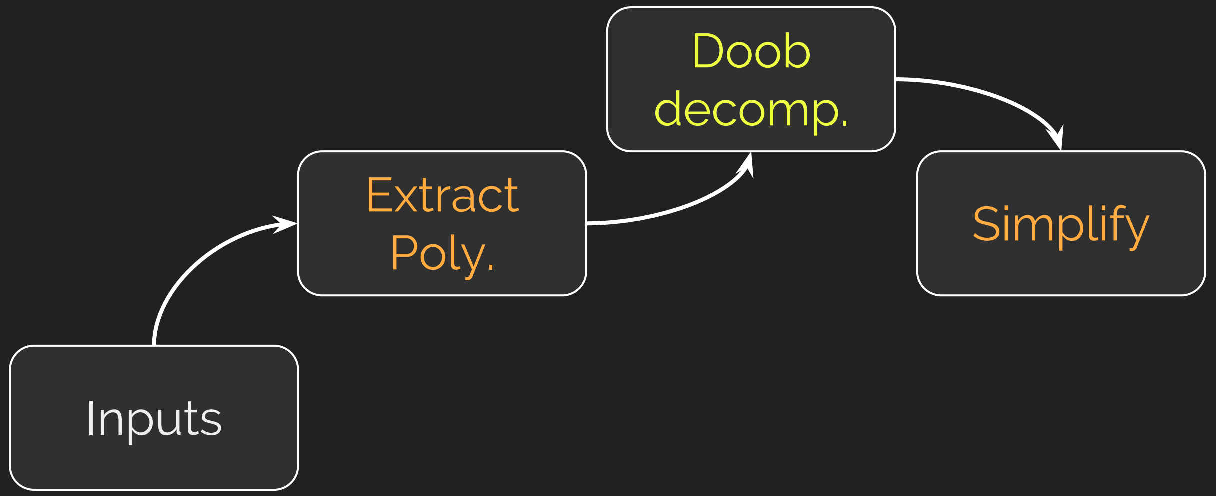
Automatization



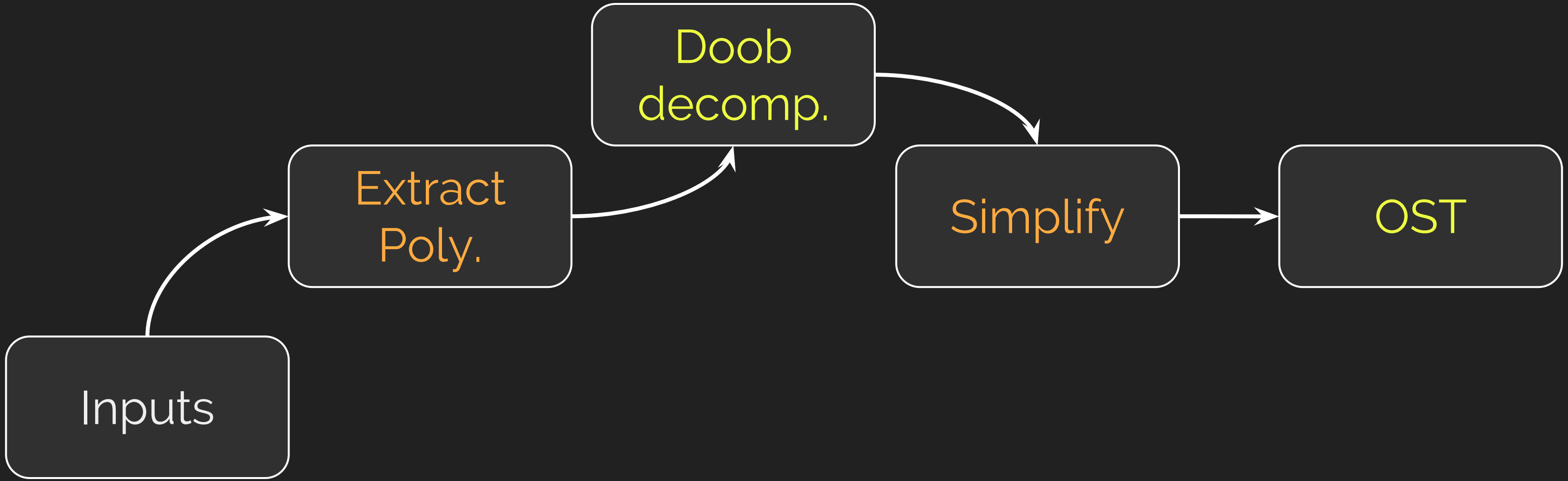
Automatization



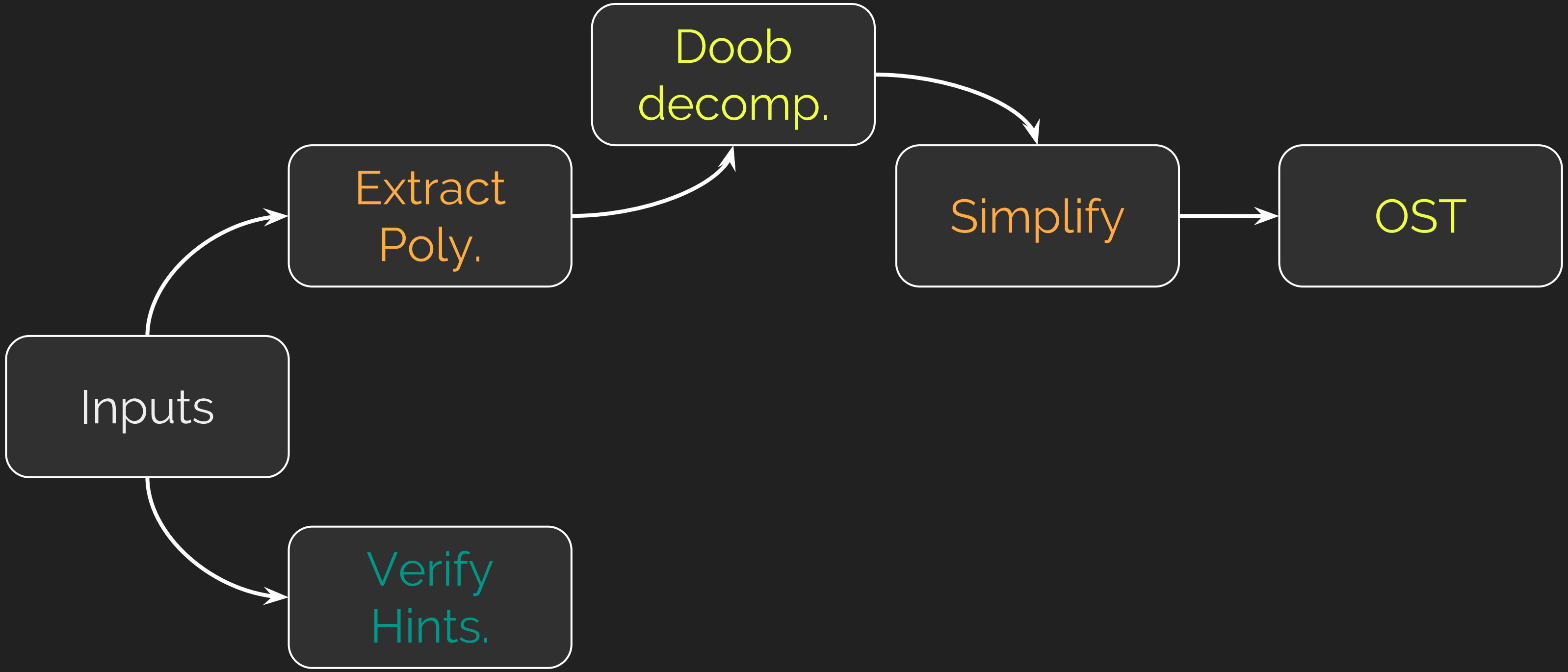
Automatization



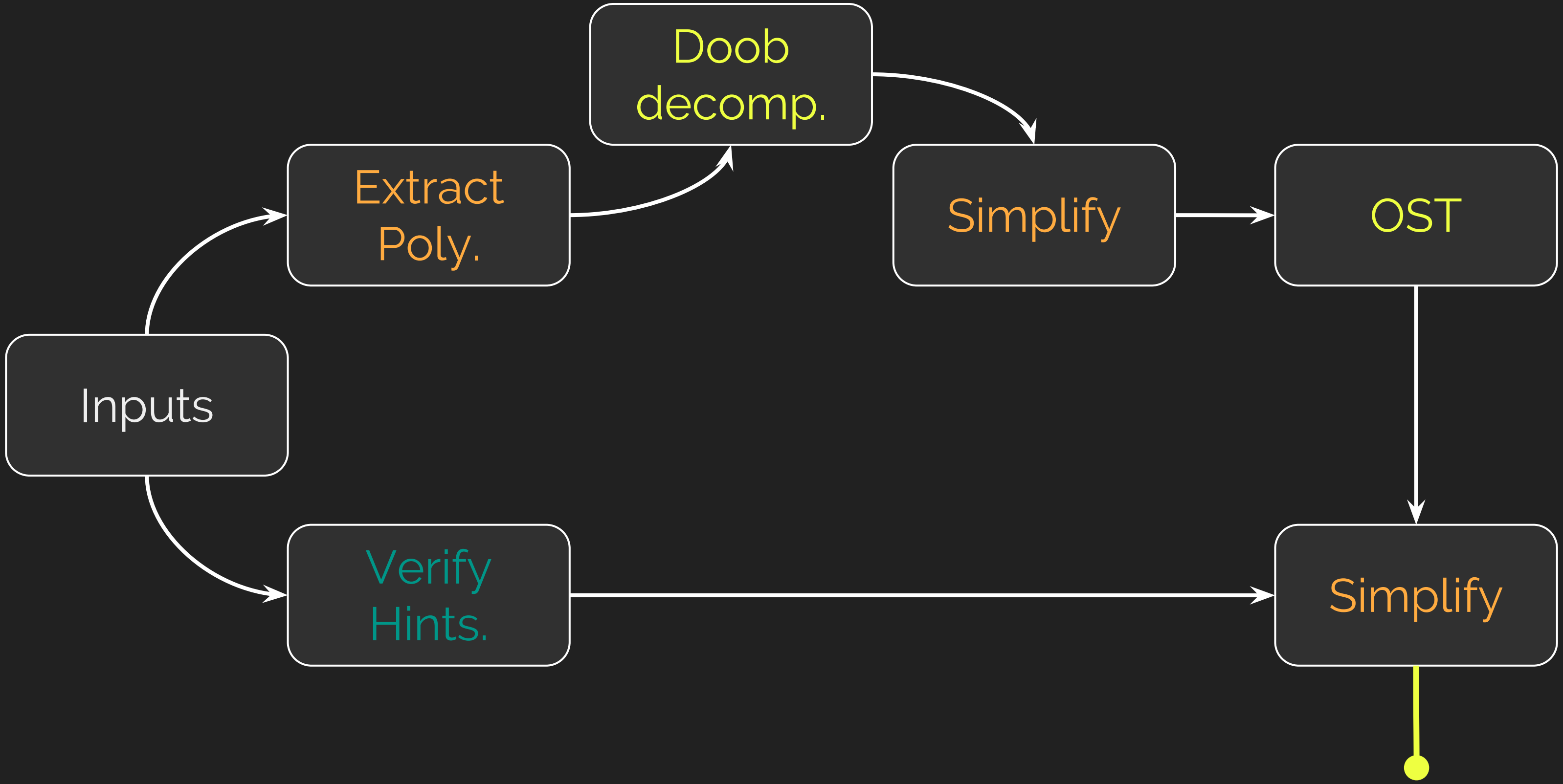
Automatization



Automatization



Automatization



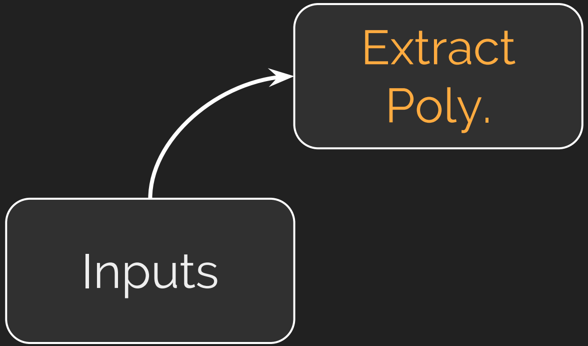
Gambler’s ruin

Gambler's ruin

X

```
x[0] ← a;
while (0 < x < b) do
z ←$ Bern(1/2, {-1, 1});
x ← x + z;
end
```

Automatization

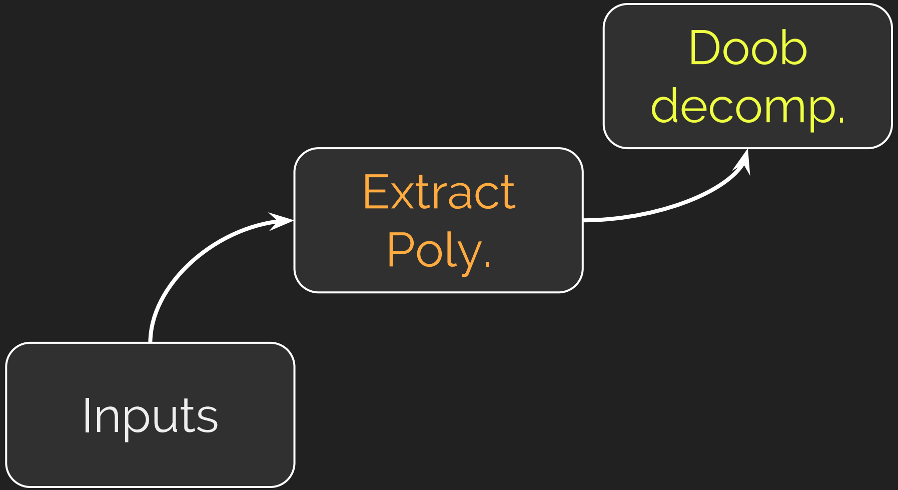


Gambler's ruin

```
x[0] ← a;
while (0 < x < b) do
z ←$ Bern(1/2, {-1, 1});
x ← x + z;
end
```

$$X_i = X_{i-1} + Z_i$$

Automatization

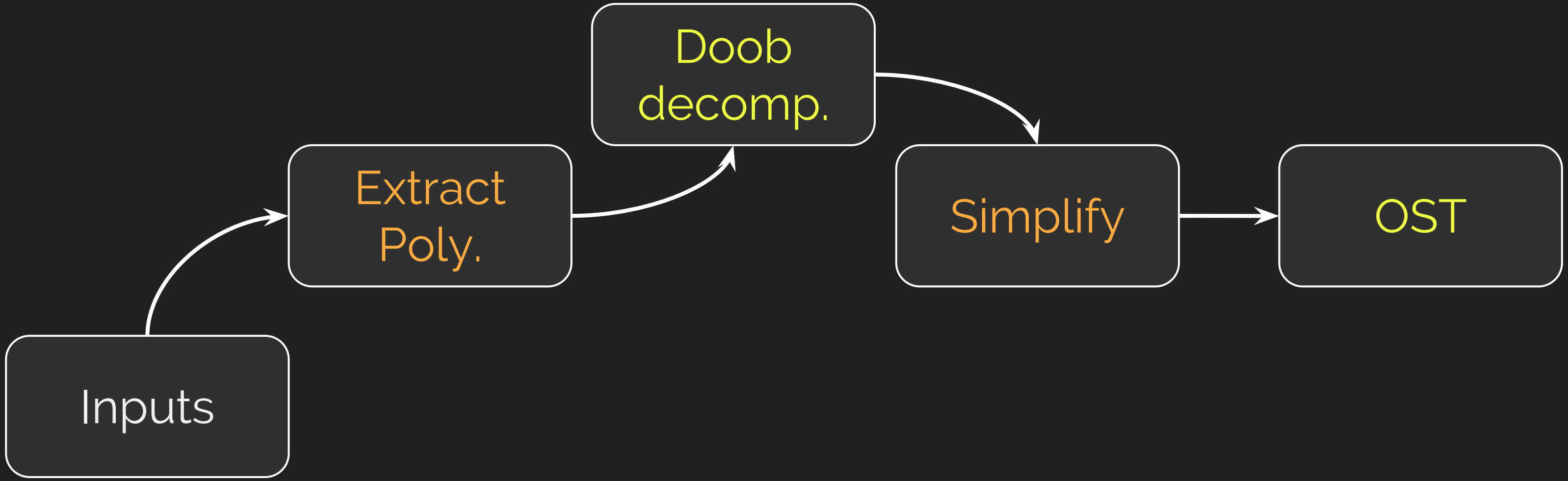


Gambler's ruin

```
x[0] ← a;
while (0 < x < b) do
z ←$ Bern(1/2, {-1, 1});
x ← x + z;
end
```

$$X_i = X_{i-1} + Z_i \longrightarrow M_0 = X_0 \qquad M_i = X_i$$

Automatization



Gambler's ruin

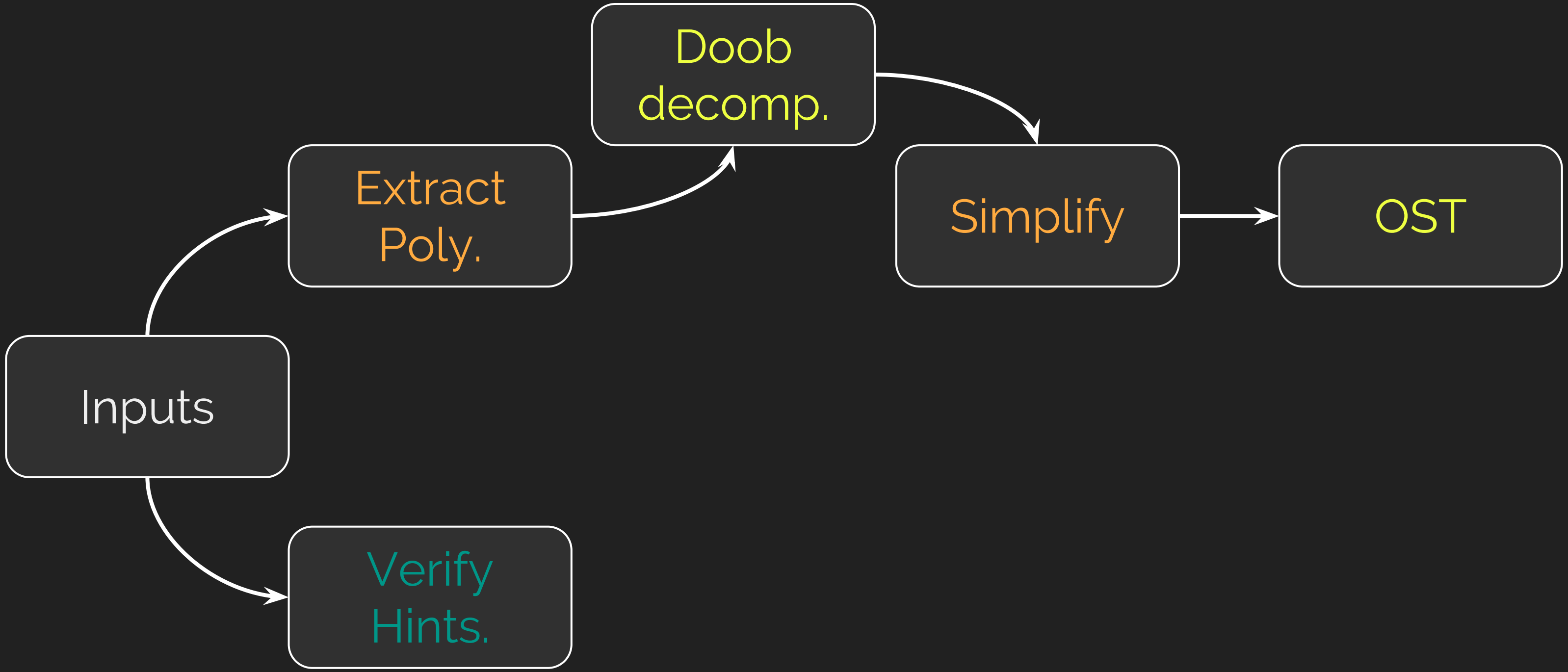
```
x[0] ← a;
while (0 < x < b) do
z ←$ Bern(1/2, {-1, 1});
x ← x + z;
end
```

$$X_i = X_{i-1} + Z_i$$

$$M_0 = X_0 \qquad M_i = X_i$$

$$a = \mathbf{E}[X_0] = \mathbf{E}[X_T]$$

Automatization



Gambler's ruin

```
x[0] ← a;
while (0 < x < b) do
z ←$ Bern(1/2, {-1, 1});
x ← x + z;
end
```

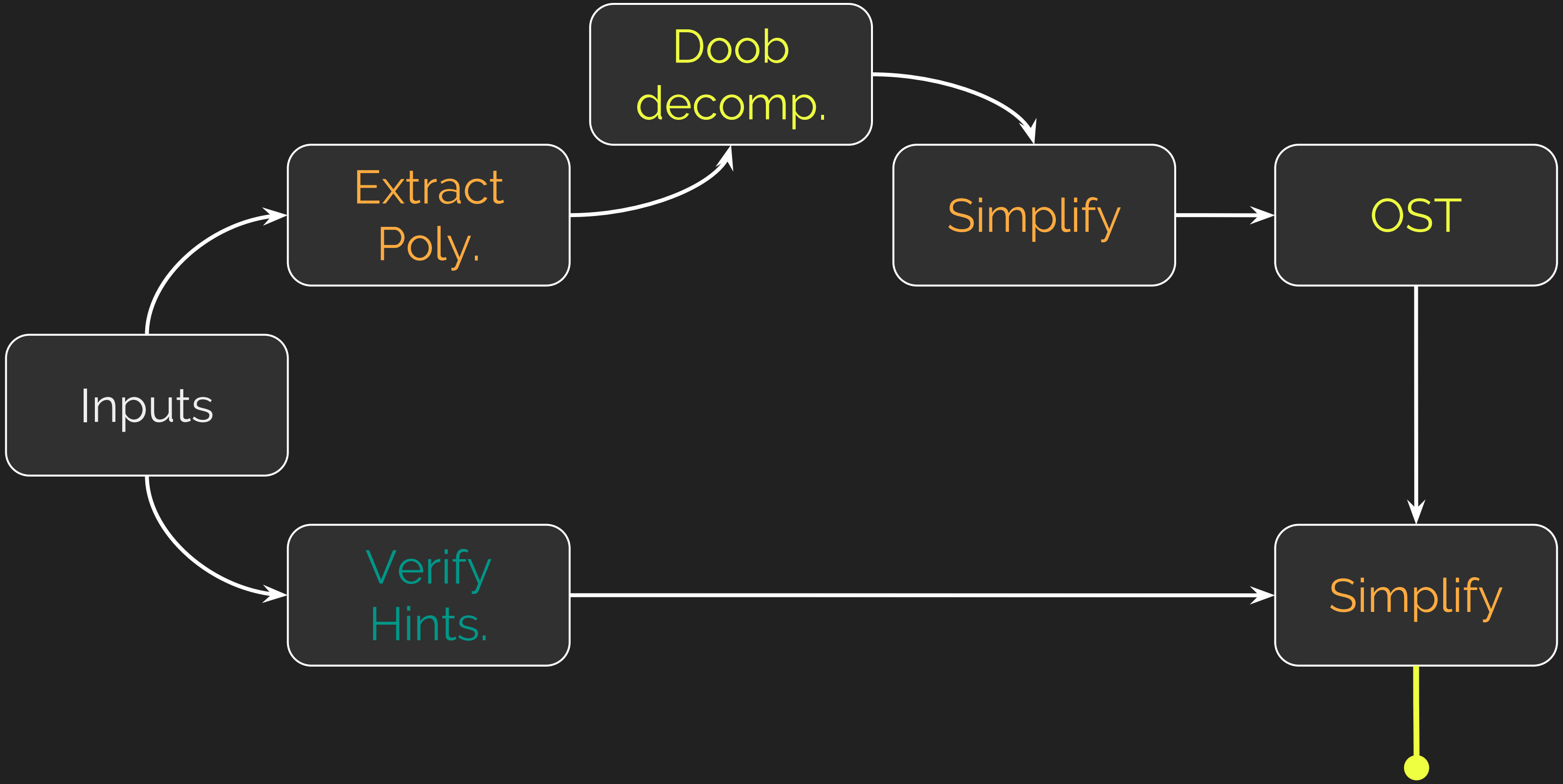
$X_i = X_{i-1} + Z_i$

$M_0 = X_0$ $M_i = X_i$

$x=0$ or $x=b$

$a = \mathbf{E}[X_0] = \mathbf{E}[X_T]$

Automatization



Gambler's ruin

```
x[0] ← a;
while (0 < x < b) do
z ←$ Bern(1/2, {-1, 1});
x ← x + z;
end
```

$X_i = X_{i-1} + Z_i$

$M_0 = X_0$ $\quad$ $M_i = X_i$

$a = \mathbf{E}[X_0] = \mathbf{E}[X_T]$

$x=0$ or $x=b$

$a = b\,\mathbf{P}[x=b]$

Gambler's ruin

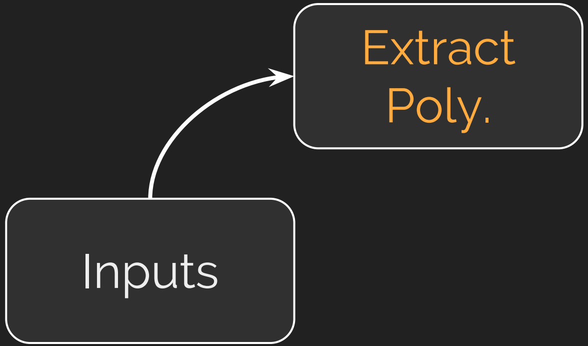
```
x[0] ← a;
while (0 < x < b) do
z ←$ Bern(1/2, {-1, 1});
x ← x + z;
end
```

Gambler's ruin

X^2

```
x[0] ← a;
while (0 < x < b) do
z ←$ Bern(1/2, {-1, 1});
x ← x + z;
end
```

Automatization



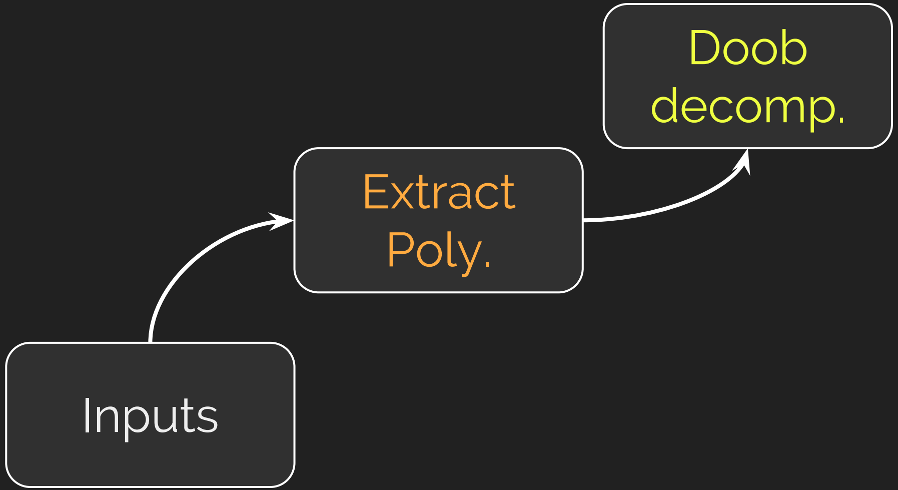
Gambler's ruin

X^2

```
x[0] ← a;
while (0 < x < b) do
z ←$ Bern(1/2, {-1, 1});
x ← x + z;
end
```

$$X_i^2 = (X_{i-1} + Z_i)^2$$

Automatization

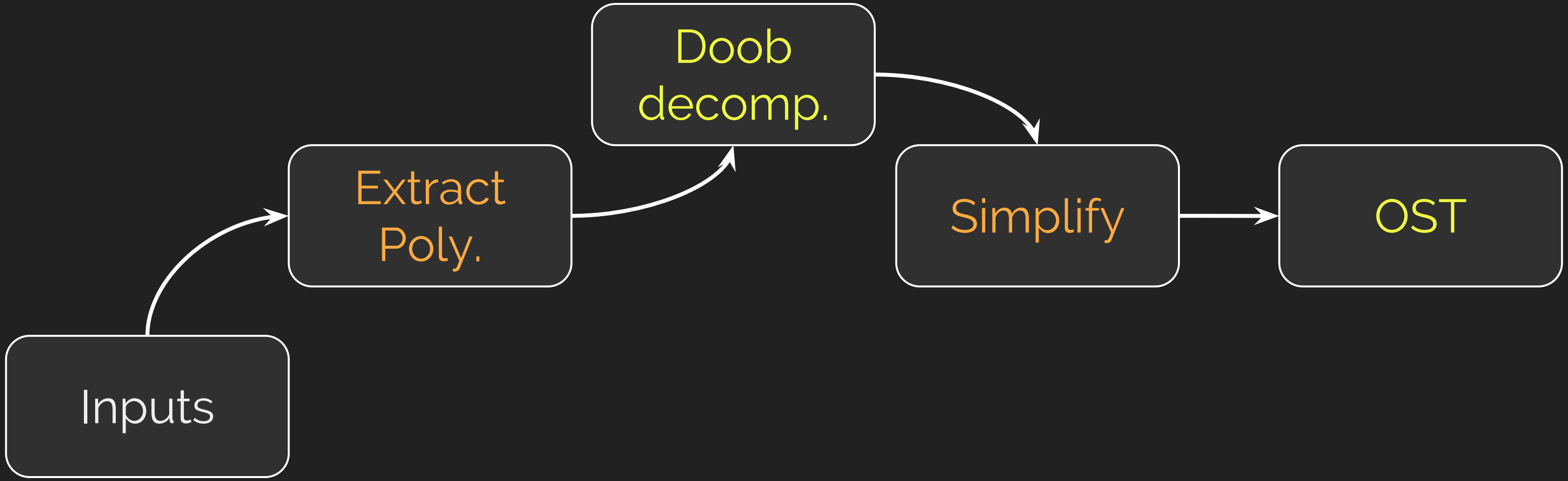


Gambler's ruin

```
x[0] ← a;
while (0 < x < b) do
z ←$ Bern(1/2, {-1, 1});
x ← x + z;
end
```

$X_i^2 = (X_{i-1} + Z_i)^2$ → $M_0 = X_0^2$ $M_i = X_i^2 - i$

Automatization



Gambler's ruin

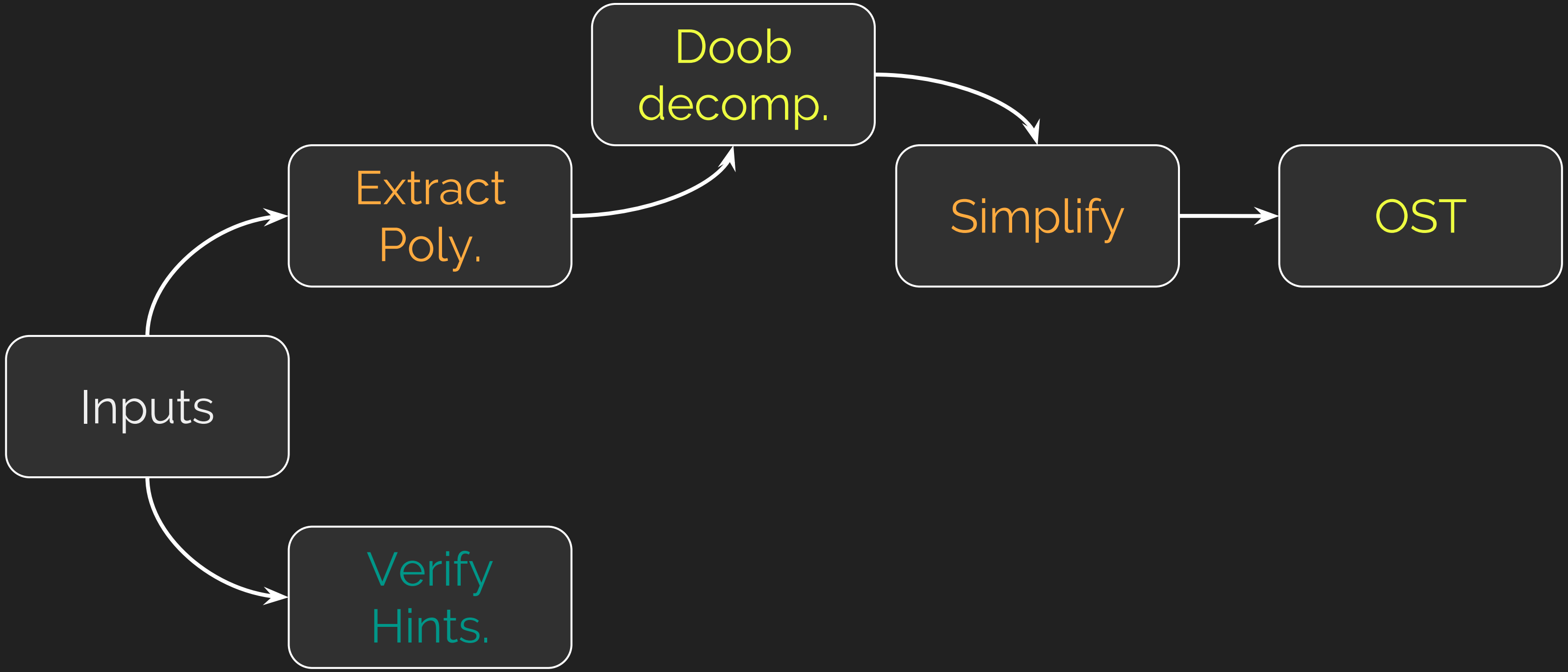
```
x[0] ← a;
while (0 < x < b) do
z ←$ Bern(1/2, {-1, 1});
x ← x + z;
end
```

$$X_i^2 = (X_{i-1} + Z_i)^2$$

$$M_0 = X_0^2 \qquad M_i = X_i^2 - i$$

$$a^2 = \mathbf{E}[X_0^2] = \mathbf{E}[X_T^2 - T]$$

Automatization



Gambler's ruin

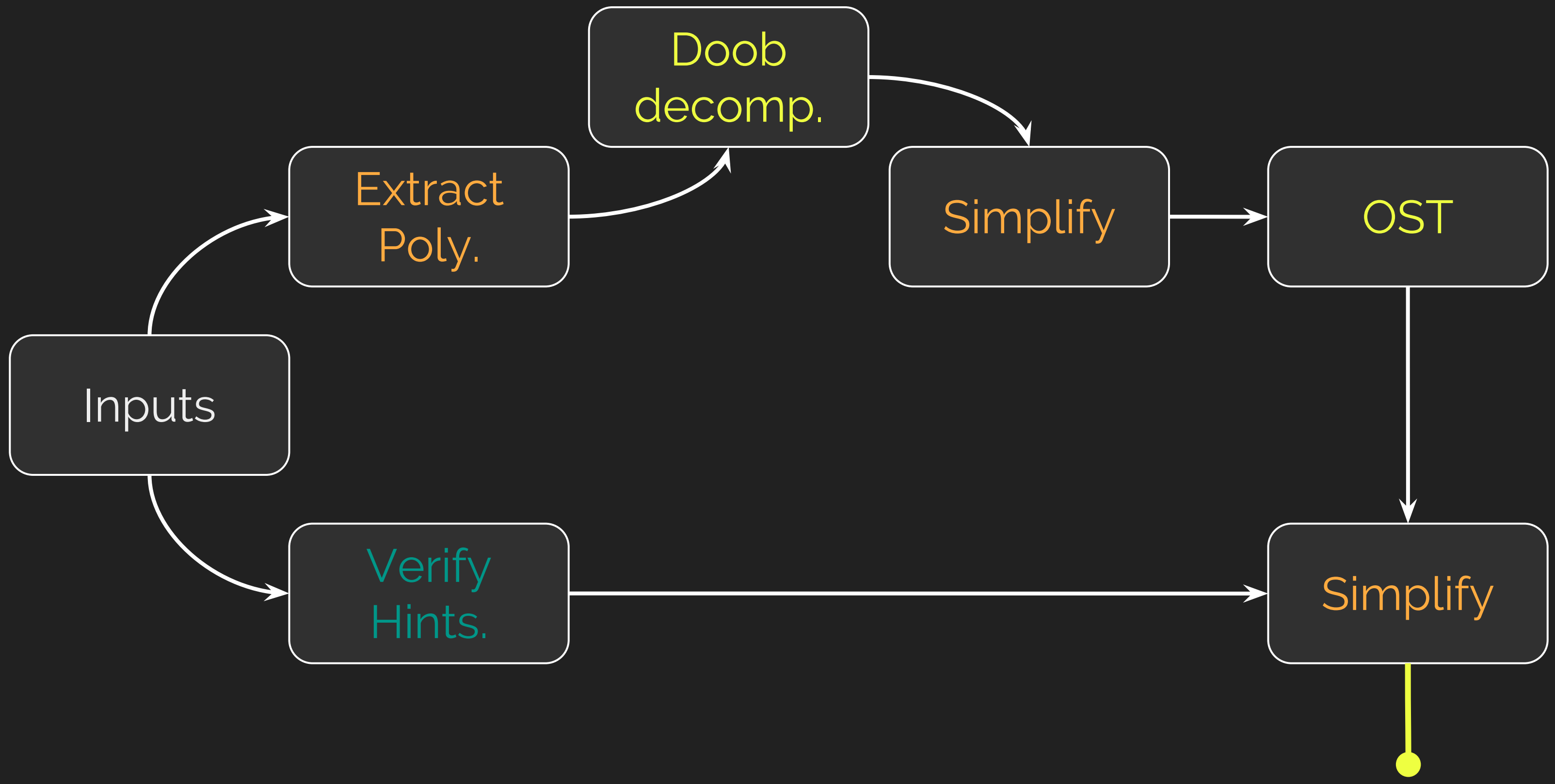
```
x[0] ← a;
while (0 < x < b) do
z ←$ Bern(1/2, {-1, 1});
x ← x + z;
end
```

$X_i^2 = (X_{i-1} + Z_i)^2$ → $M_0 = X_0^2$ $\quad M_i = X_i^2 - i$

$x=0$ or $x=b$

$a^2 = \mathbf{E}[X_0^2] = \mathbf{E}[X_T^2 - T]$

Automatization



Gambler's ruin

```
x[0] ← a;
while (0 < x < b) do
z ←$ Bern(1/2, {-1, 1});
x ← x + z;
end
```

$X_i^2 = (X_{i-1} + Z_i)^2$

$M_0 = X_0^{\ 2}$ $\quad$ $M_i = X_i^2 - i$

$a^2 = \mathbf{E}[X_0^{\ 2}] = \mathbf{E}[X_T^{\ 2} - T]$

x=0 or x=b

$a^2 = b^2\ \mathbf{P}[x=b] - \mathbf{E}[T]$

Gambler's ruin

```
x[0] ← a;
while (0 < x < b) do
z ←$ Bern(1/2, {-1, 1});
x ← x + z;
end
```

$a = b\ \mathbf{P}[x=b]$

$a^2 = b^2\ \mathbf{P}[x=b] - \mathbf{E}[T]$

$\mathbf{E}[T] = a(b-a)$

Gambler's ruin

$x[0] \leftarrow a;$

while $(0 < x < b)$ **do**

$z \leftarrow\$ \text{Bern}(1/2, \{-1, 1\});$

$x \leftarrow x + z;$

end

" Everything on black is not a risk management strategy, Hoskins."

Abracadabra

$match_0[0] \leftarrow 1;$
$match_1[0] \leftarrow 0;$
...
$match_{11}[0] \leftarrow 0;$

while $(match_{11} == 0)$ **do**

$s \leftarrow\$ \text{UnifMatches};$

$match_{11} \leftarrow match_{10}[-1] * \pi_{11}(s);$
$match_{10} \leftarrow match_9[-1] * \pi_{10}(s);$
...
$match1 \leftarrow match_0[-1] * \pi_1(s);$

end

Abracadabra

$match_0[0] \leftarrow 1;$
$match_1[0] \leftarrow 0;$
...
$match_{11}[0] \leftarrow 0;$

while $(match_{11} == 0)$ **do**

$s \leftarrow\$ \text{UnifMatches};$

$match_{11} \leftarrow match_{10}[-1] * \pi_{11}(s);$
$match_{10} \leftarrow match_9[-1] * \pi_{10}(s);$
...
$match1 \leftarrow match_0[-1] * \pi_1(s);$

end

Abracadabra

$match_0[0] \leftarrow 1;$
$match_1[0] \leftarrow 0;$
...
$match_{11}[0] \leftarrow 0;$

while (*$match_{11}$ == 0*) **do**

$s \leftarrow \$ \text{ UnifMatches};$

$match_{11} \leftarrow match_{10}[-1] * \pi_{11}(s);$
$match_{10} \leftarrow match_9[-1] * \pi_{10}(s);$
...
$match1 \leftarrow match_0[-1] * \pi_1(s);$

end

$$1 + L \cdot match_1 + \dots + L^{11} \cdot match_{11}$$

Abracadabra

```
match_0[0] ←1;
match_1[0] ←0;
...
match_11[0] ← 0;

while (match_11 == 0) do

s ←$ UnifMatches;

match_11 ← match_10[-1] * π_11(s);
match_10 ← match_9[-1] * π_10(s);
...
match1 ← match_0[-1] * π_1(s);

end
```

$$1+ L\cdot \mathit{match}_1 + \cdots + L^{11} \cdot \mathit{match}_{11}$$

$1+L+\ldots+L^{11}$ - [$1+ L\cdot \mathit{match}_1 + \cdots + L^{11} \cdot \mathit{match}_{11}$] decreases with Pr $1/L$

Abracadabra

$match_0[0] \leftarrow 1;$
$match_1[0] \leftarrow 0;$
...
$match_{11}[0] \leftarrow 0;$

while ($match_{11}$ == 0) **do**

$s \leftarrow\$ \text{UnifMatches};$

$match_{11} \leftarrow match_{10}[-1] * \pi_{11}(s);$
$match_{10} \leftarrow match_9[-1] * \pi_{10}(s);$
...
$match1 \leftarrow match_0[-1] * \pi_1(s);$

end

$$1 + L \cdot match_1 + \dots + L^{11} \cdot match_{11}$$

$$\mathbf{E}[T] = L + L^4 + L^{11}$$

Abracadabra

$match_0[0] \leftarrow 1;$
$match_1[0] \leftarrow 0;$
...
$match_{11}[0] \leftarrow 0;$

while (*$match_{11}$ == 0*) **do**

s ←$ UnifMatches;

$match_{11} \leftarrow match_{10}[-1] * \pi_{11}(s);$
$match_{10} \leftarrow match_9[-1] * \pi_{10}(s);$
...
$match1 \leftarrow match_0[-1] * \pi_1(s);$

end

$$1 + L \cdot match_1 + \dots + L^{11} \cdot match_{11}$$

$$\downarrow$$

$$\mathbf{E}[T] = 26 + 26^4 + 26^{11}$$

Abracadabra

$match_0[0] \leftarrow 1;$
$match_1[0] \leftarrow 0;$
...
$match_{11}[0] \leftarrow 0;$

while ($match_{11} == 0$) **do**

$s \leftarrow\$ \text{UnifMatches};$

$match_{11} \leftarrow match_{10}[-1] * \pi_{11}(s);$
$match_{10} \leftarrow match_{9}[-1] * \pi_{10}(s);$
...
$match1 \leftarrow match_{0}[-1] * \pi_{1}(s);$

end

$$1 + L \cdot match_1 + \dots + L^{11} \cdot match_{11}$$

$$\downarrow$$

E[T] = 3670344487444778

Abracadabra

$match_0[0] \leftarrow 1;$
$match_1[0] \leftarrow 0;$
...
$match_{11}[0] \leftarrow 0;$

while ($match_{11}$ == 0) **do**

$s \leftarrow\$ \text{UnifMatches};$

$match_{11} \leftarrow match_{10}[-1] * \pi_{11}(s);$
$match_{10} \leftarrow match_{9}[-1] * \pi_{10}(s);$
...
$match1 \leftarrow match_{0}[-1] * \pi_{1}(s);$

end

$1 + L \cdot match_1 + \dots + L^{11} \cdot match_{11}$

$\downarrow$

E[T] = 3670344487444778

116 385 860 years...

In a nutshell

- Program + Seed → *Martingale*
- *Martingale* + OST → Expectation at the stopping time of loop
- Works symbolically , only requires AS-termination
- POC written in Python+SymPy, runs in less than a second for simplest example to 6s for Abracadabra.

Questions?