Nondeterminstic Discrete Stochastic Differential Dynamic Logic

Samuel Kim

December 10, 2019

$$\frac{\sigma(\rho(\alpha, P), p) \overset{\mathrm{d}\mathcal{L}}{\rightsquigarrow} Q}{\langle\!|\alpha|\!\rangle P \leq p \overset{\mathrm{d}\mathcal{L}}{\rightsquigarrow} Q} \ (\mathrm{d}\mathcal{L}\text{-}\langle\!|\,|\!\rangle)$$

Intro

- Stochastic hybrid systems: hybrid systems + stochastics

Intro

- Stochastic hybrid systems: hybrid systems + stochastics
- Powerful dynamic logics:

Intro

- Stochastic hybrid systems: hybrid systems + stochastics
- Powerful dynamic logics:
  - Stochastic Differential Dynamic Logic (Sd$\mathcal{L}$)

Intro

- Stochastic hybrid systems: hybrid systems + stochastics
- Powerful dynamic logics:
  - Stochastic Differential Dynamic Logic (Sd$\mathcal{L}$)
  - Nondeterminstic Stochastic Differential Dynamic Logic (NSd$\mathcal{L}$)

Intro

- Stochastic hybrid systems: hybrid systems + stochastics
- Powerful dynamic logics:
  - Stochastic Differential Dynamic Logic (Sd$\mathcal{L}$)
  - Nondeterminstic Stochastic Differential Dynamic Logic (NSd$\mathcal{L}$)
- Problem: no implementations!

NDSd$\mathcal{L}$

- Nondeterminstic Discrete Stochastic Differential Dynamic Logic

NDSd$\mathcal{L}$

- Nondeterminstic Discrete Stochastic Differential Dynamic Logic
- Based on NSd$\mathcal{L}$, but discrete

NDSd$\mathcal{L}$

- Nondeterminstic Discrete Stochastic Differential Dynamic Logic
- Based on NSd$\mathcal{L}$, but discrete
- Fully compiles down to d$\mathcal{L}$

NDSd$\mathcal{L}$

- Nondeterminstic Discrete Stochastic Differential Dynamic Logic
- Based on NSd$\mathcal{L}$, but discrete
- Fully compiles down to d$\mathcal{L}$
  - Can use KeYmaera X!

NDSd$\mathcal{L}$

- Nondeterminstic Discrete Stochastic Differential Dynamic Logic
- Based on NSd$\mathcal{L}$, but discrete
- Fully compiles down to d$\mathcal{L}$
  - Can use KeYmaera X!
- Translation soundness theorems

NDSd$\mathcal{L}$

- Nondeterminstic Discrete Stochastic Differential Dynamic Logic
- Based on NSd$\mathcal{L}$, but discrete
- Fully compiles down to d$\mathcal{L}$
  - Can use KeYmaera X!
- Translation soundness theorems
- Compiler implementation

NDSd$\mathcal{L}$ Syntax

$$\text{Term } e_1, e_2 ::= x \mid c \mid e_1 + e_2 \mid e_1 \cdot e_2$$

$$\text{Formula } P, Q ::= e_1 \geq e_2 \mid e_1 = e_2 \mid \neg P \mid P \odot Q \mid \forall x P \mid \exists x P \mid [\alpha]P \mid \langle\alpha\rangle P$$
$$\mid \langle\!\langle \alpha \rangle\!\rangle P \leq p \qquad (\odot \in \{\wedge, \vee, \rightarrow, \leftrightarrow\})$$

$$\text{Program } \alpha, \beta ::= x := e \mid x := * \mid x := \{p_1 : e_1, \ldots, p_n : e_n\} \mid ?P \mid \alpha; \beta$$
$$\mid \alpha^* \mid \alpha^{*:p} \mid \alpha \cup \beta \mid \bigoplus_{i=1}^{n} p_i \alpha_i \mid x' = f(x) \,\&\, P$$

NDSd$\mathcal{L}$ Syntax

$$\text{Term } e_1, e_2 ::= x \mid c \mid e_1 + e_2 \mid e_1 \cdot e_2$$

$$\text{Formula } P, Q ::= e_1 \geq e_2 \mid e_1 = e_2 \mid \neg P \mid P \odot Q \mid \forall x P \mid \exists x P \mid [\alpha]P \mid \langle\alpha\rangle P \mid \langle\!\langle \alpha \rangle\!\rangle P \leq p \qquad (\odot \in \{\wedge, \vee, \rightarrow, \leftrightarrow\})$$

$$\text{Program } \alpha, \beta ::= x := e \mid x := * \mid x := \{p_1 : e_1, \ldots, p_n : e_n\} \mid ?P \mid \alpha; \beta \mid \alpha^* \mid \alpha^{*:p} \mid \alpha \cup \beta \mid \bigoplus_{i=1}^{n} p_i \alpha_i \mid x' = f(x) \,\&\, P$$

- $\langle\!\langle \alpha \rangle\!\rangle P \leq p$: Probability bound

NDSd$\mathcal{L}$ Syntax

$$\text{Term } e_1, e_2 ::= x \mid c \mid e_1 + e_2 \mid e_1 \cdot e_2$$

$$\text{Formula } P, Q ::= e_1 \geq e_2 \mid e_1 = e_2 \mid \neg P \mid P \odot Q \mid \forall x P \mid \exists x P \mid [\alpha]P \mid \langle\alpha\rangle P \mid \langle\!\langle \alpha \rangle\!\rangle P \leq p \qquad (\odot \in \{\wedge, \vee, \rightarrow, \leftrightarrow\})$$

$$\text{Program } \alpha, \beta ::= x := e \mid x := * \mid x := \{p_1 : e_1, \ldots, p_n : e_n\} \mid ?P \mid \alpha; \beta \mid \alpha^* \mid \alpha^{*:p} \mid \alpha \cup \beta \mid \bigoplus_{i=1}^{n} p_i \alpha_i \mid x' = f(x) \,\&\, P$$

- $\langle\!\langle \alpha \rangle\!\rangle P \leq p$: Probability bound
- $x := \{p_1 : e_1, \ldots, p_n : e_n\}$: Probabilistic assignment

NDSd$\mathcal{L}$ Syntax

$$\text{Term } e_1, e_2 ::= x \mid c \mid e_1 + e_2 \mid e_1 \cdot e_2$$

$$\text{Formula } P, Q ::= e_1 \geq e_2 \mid e_1 = e_2 \mid \neg P \mid P \odot Q \mid \forall x P \mid \exists x P \mid [\alpha]P \mid \langle\alpha\rangle P \mid \langle\!\langle \alpha \rangle\!\rangle P \leq p \qquad (\odot \in \{\wedge, \vee, \rightarrow, \leftrightarrow\})$$

$$\text{Program } \alpha, \beta ::= x := e \mid x := * \mid x := \{p_1 : e_1, \ldots, p_n : e_n\} \mid ?P \mid \alpha; \beta \mid \alpha^* \mid \alpha^{*:p} \mid \alpha \cup \beta \mid \bigoplus_{i=1}^{n} p_i \alpha_i \mid x' = f(x) \,\&\, P$$

- $\langle\!\langle \alpha \rangle\!\rangle P \leq p$: Probability bound
- $x := \{p_1 : e_1, \ldots, p_n : e_n\}$: Probabilistic assignment
- $\alpha^{*:p}$: Probabilistic loop

NDSd$\mathcal{L}$ Syntax

$$\text{Term } e_1, e_2 ::= x \mid c \mid e_1 + e_2 \mid e_1 \cdot e_2$$

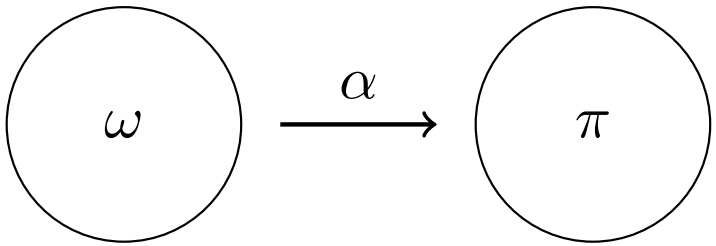
$$\text{Formula } P, Q ::= e_1 \geq e_2 \mid e_1 = e_2 \mid \neg P \mid P \odot Q \mid \forall x P \mid \exists x P \mid [\alpha]P \mid \langle\alpha\rangle P \mid \langle\!\langle \alpha \rangle\!\rangle P \leq p \qquad (\odot \in \{\wedge, \vee, \rightarrow, \leftrightarrow\})$$

$$\text{Program } \alpha, \beta ::= x := e \mid x := * \mid x := \{p_1 : e_1, \ldots, p_n : e_n\} \mid ?P \mid \alpha; \beta \mid \alpha^* \mid \alpha^{*:p} \mid \alpha \cup \beta \mid \bigoplus_{i=1}^{n} p_i \alpha_i \mid x' = f(x) \,\&\, P$$

- $\langle\!\langle \alpha \rangle\!\rangle P \leq p$: Probability bound
- $x := \{p_1 : e_1, \ldots, p_n : e_n\}$: Probabilistic assignment
- $\alpha^{*:p}$: Probabilistic loop
- $\bigoplus_{i=1}^{n} p_i \alpha_i$: Probabilistic choice

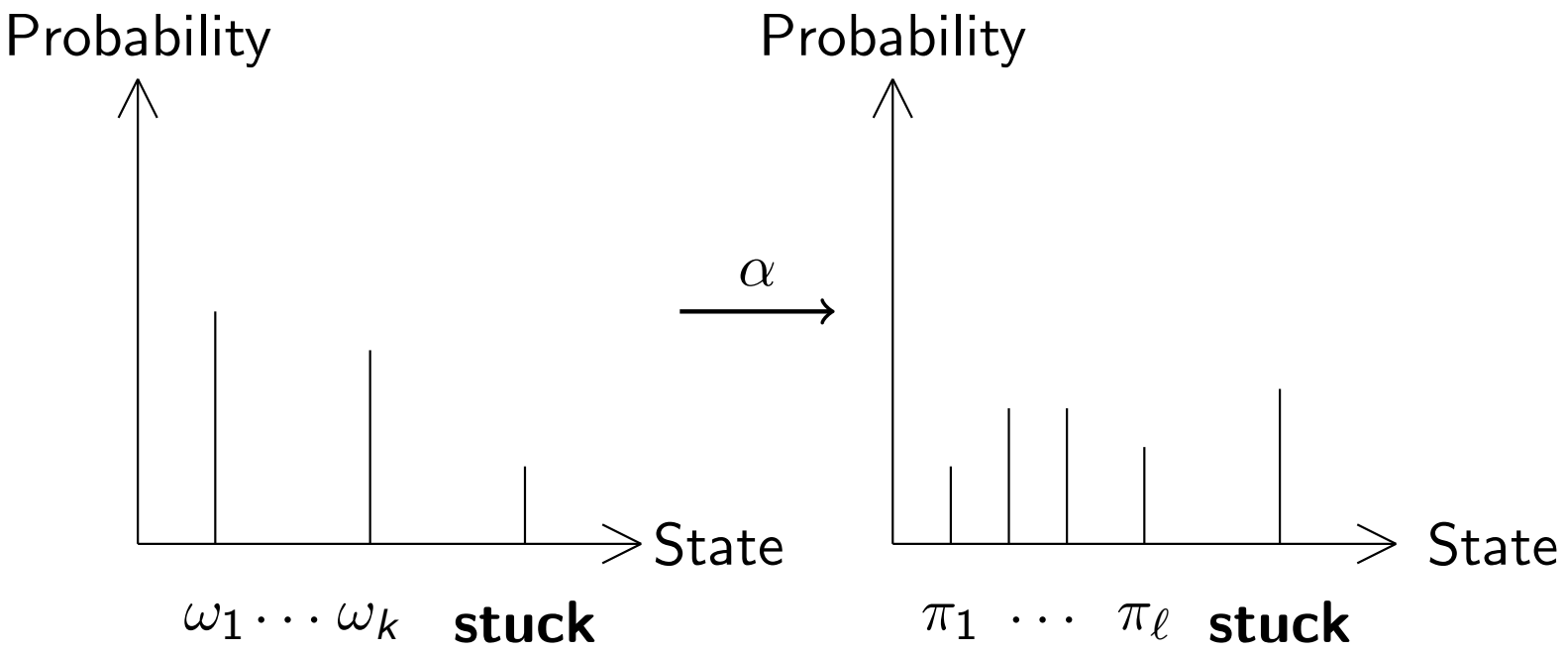
Program Semantics

- $\mathrm{d}\mathcal{L}$: transition between states



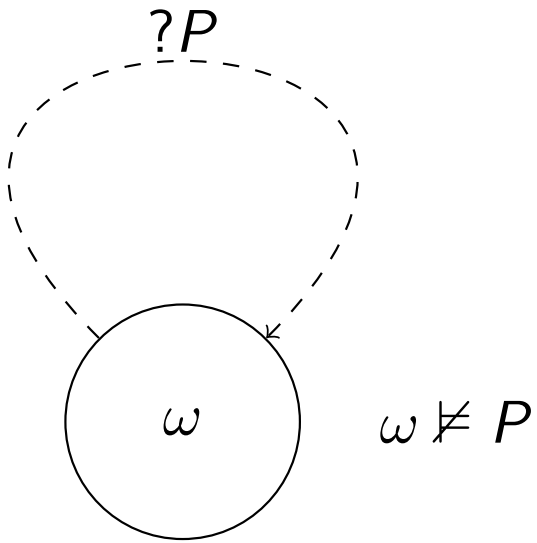
Program Semantics

- d$\mathcal{L}$: transition between states
- NDSd$\mathcal{L}$: transition between probability distributions of states



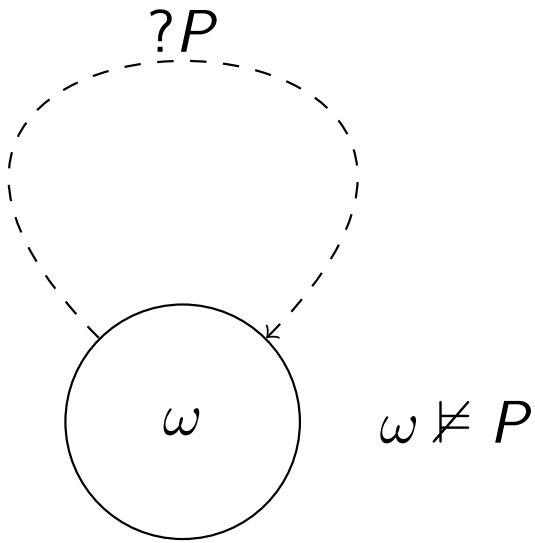
stuck?

- d$\mathcal{L}$: Failed test: no transition



stuck?

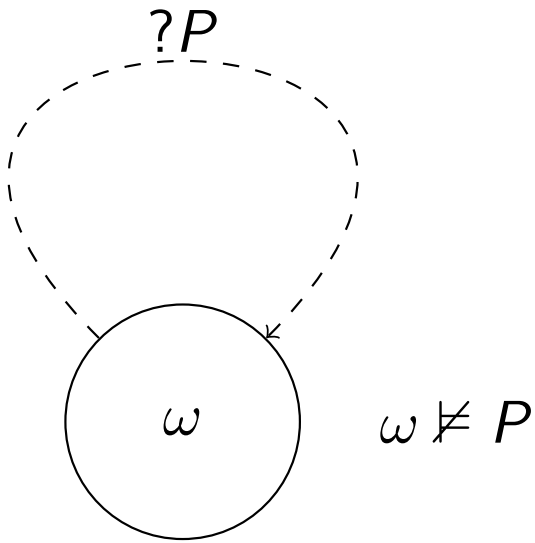
- d$\mathcal{L}$: Failed test: no transition



- NDSd$\mathcal{L}$: Failed test: mapped to **stuck**

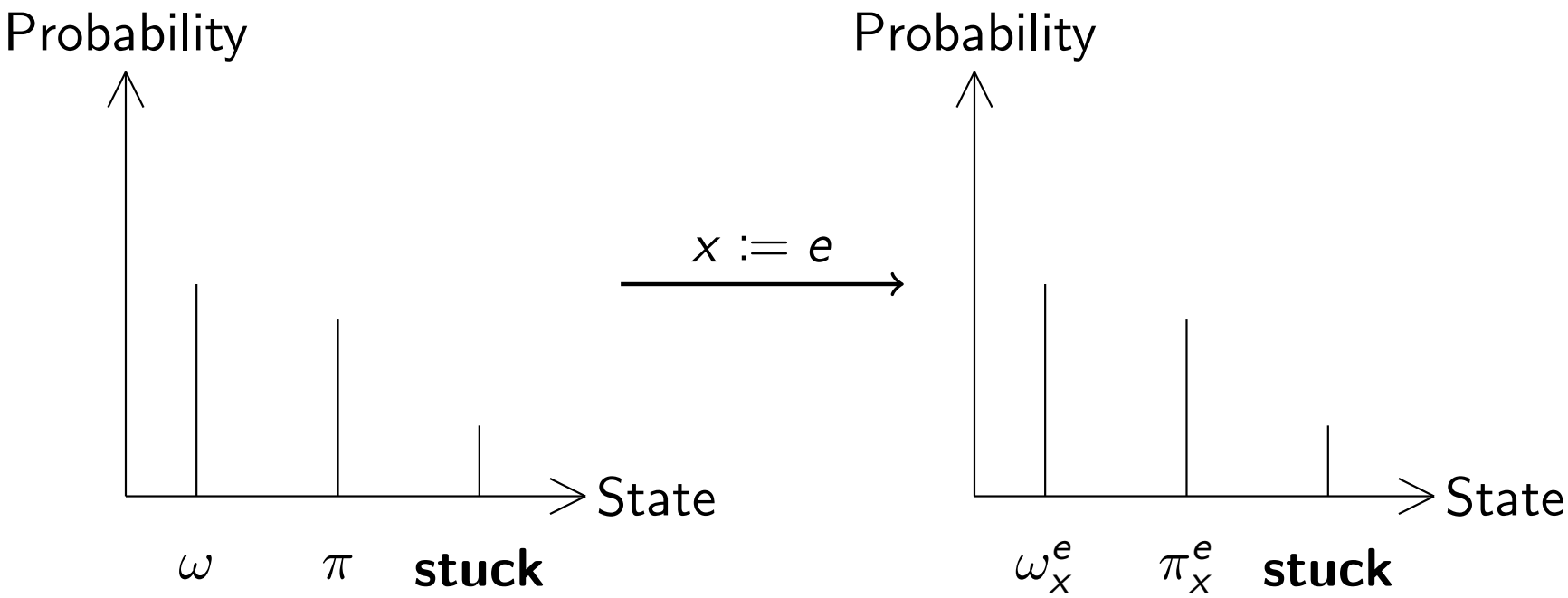
stuck?

- d$\mathcal{L}$: Failed test: no transition

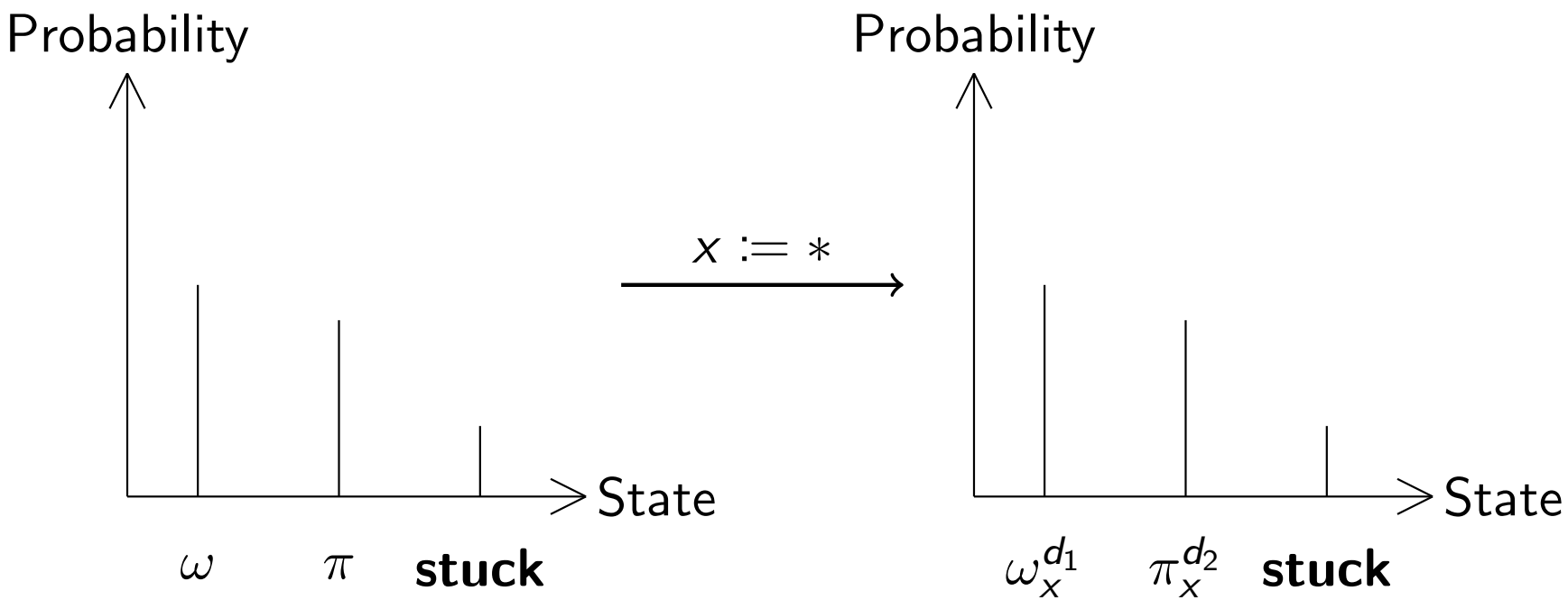


- NDSd$\mathcal{L}$: Failed test: mapped to **stuck**
- **stuck** is not a state!

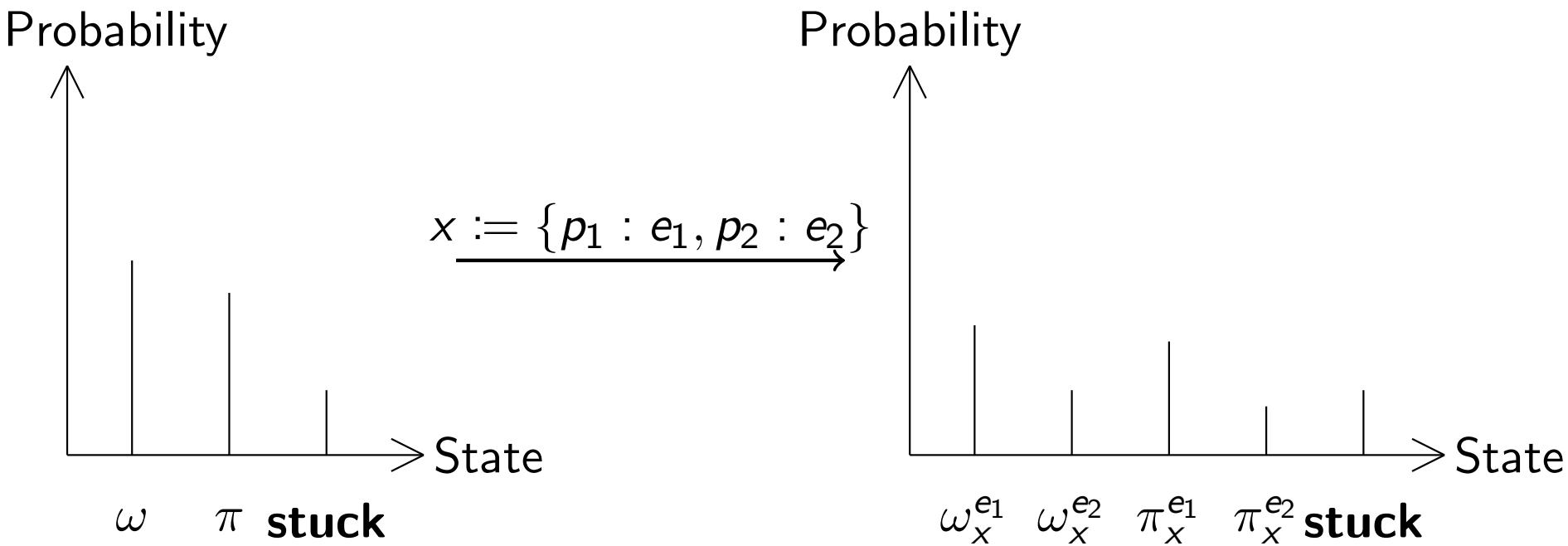
Program Semantics: Assignment



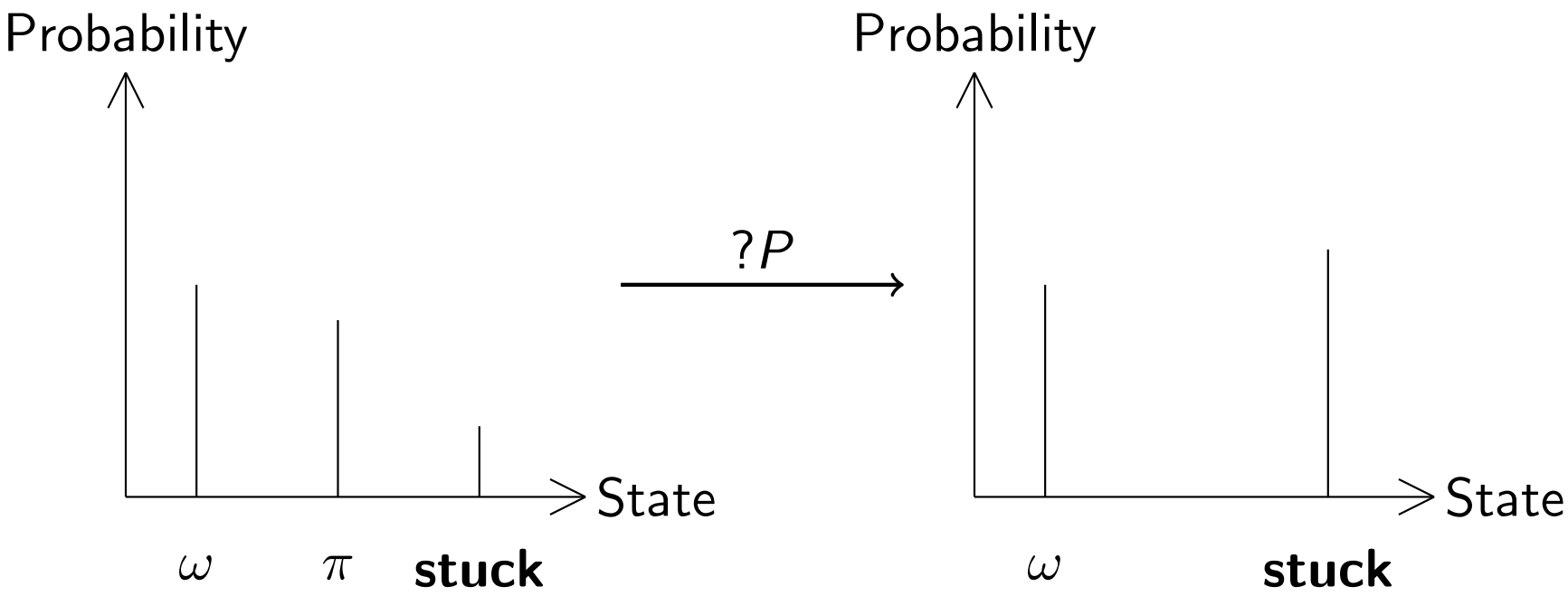
Program Semantics: Nondeterminstic Assignment



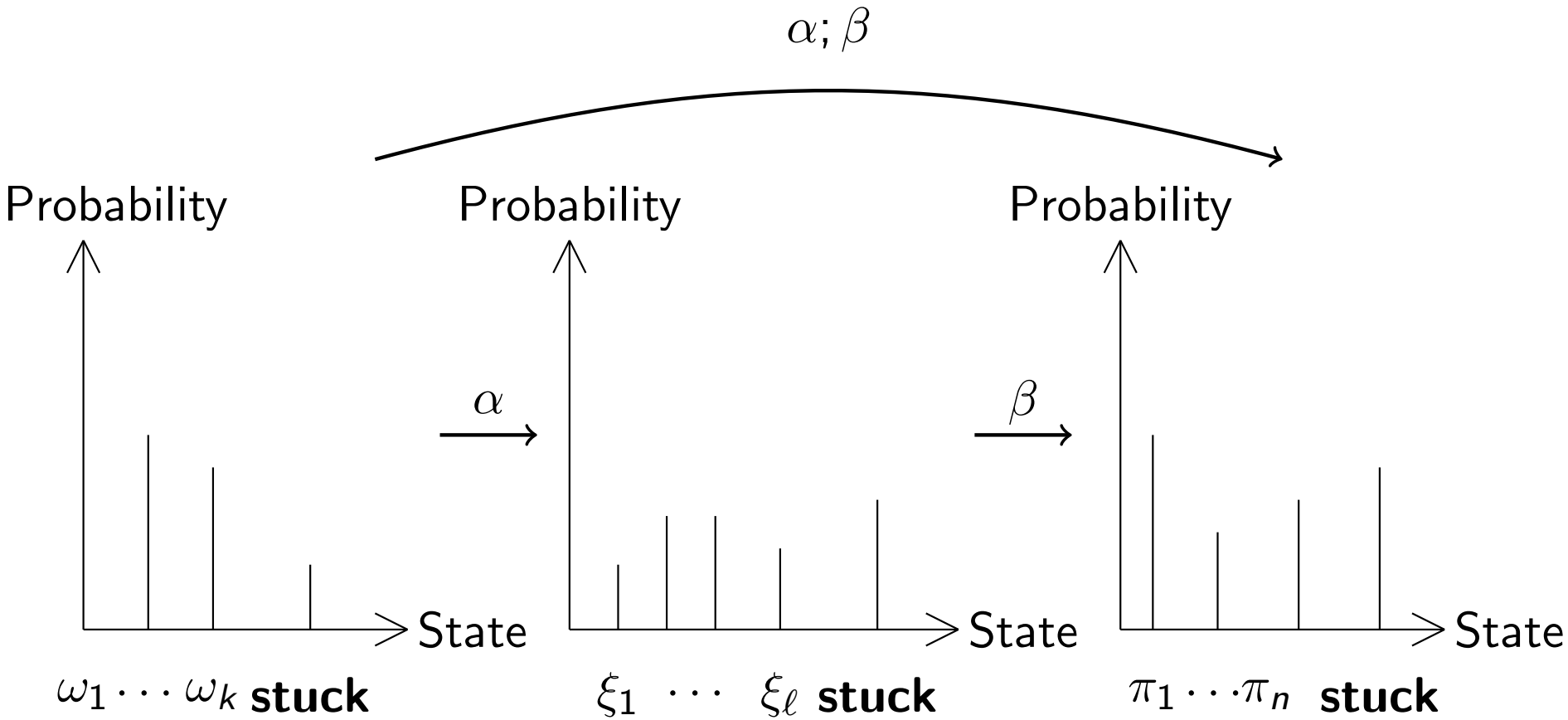
Program Semantics: Probabilistic Assignment



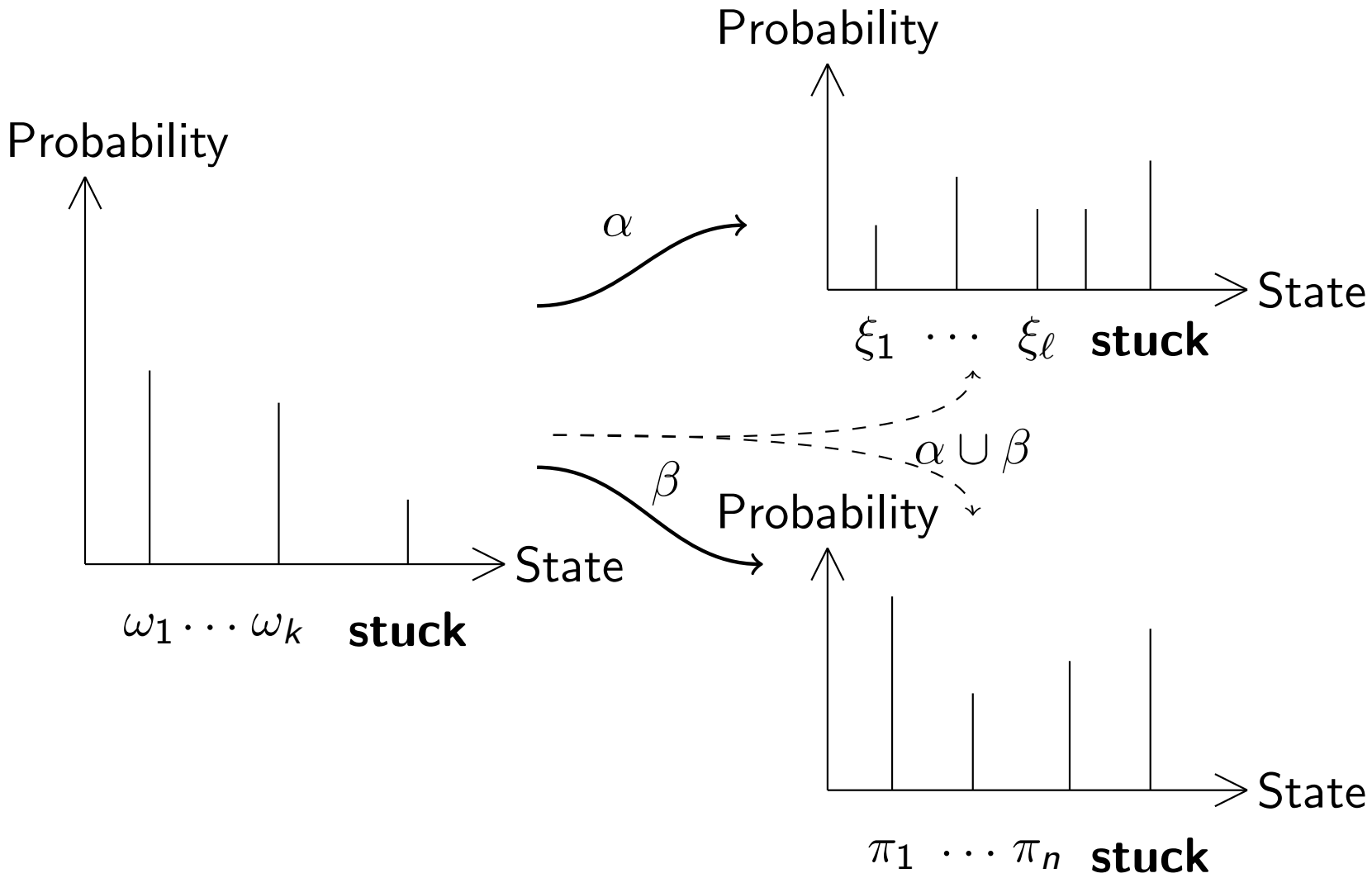
Program Semantics: Test



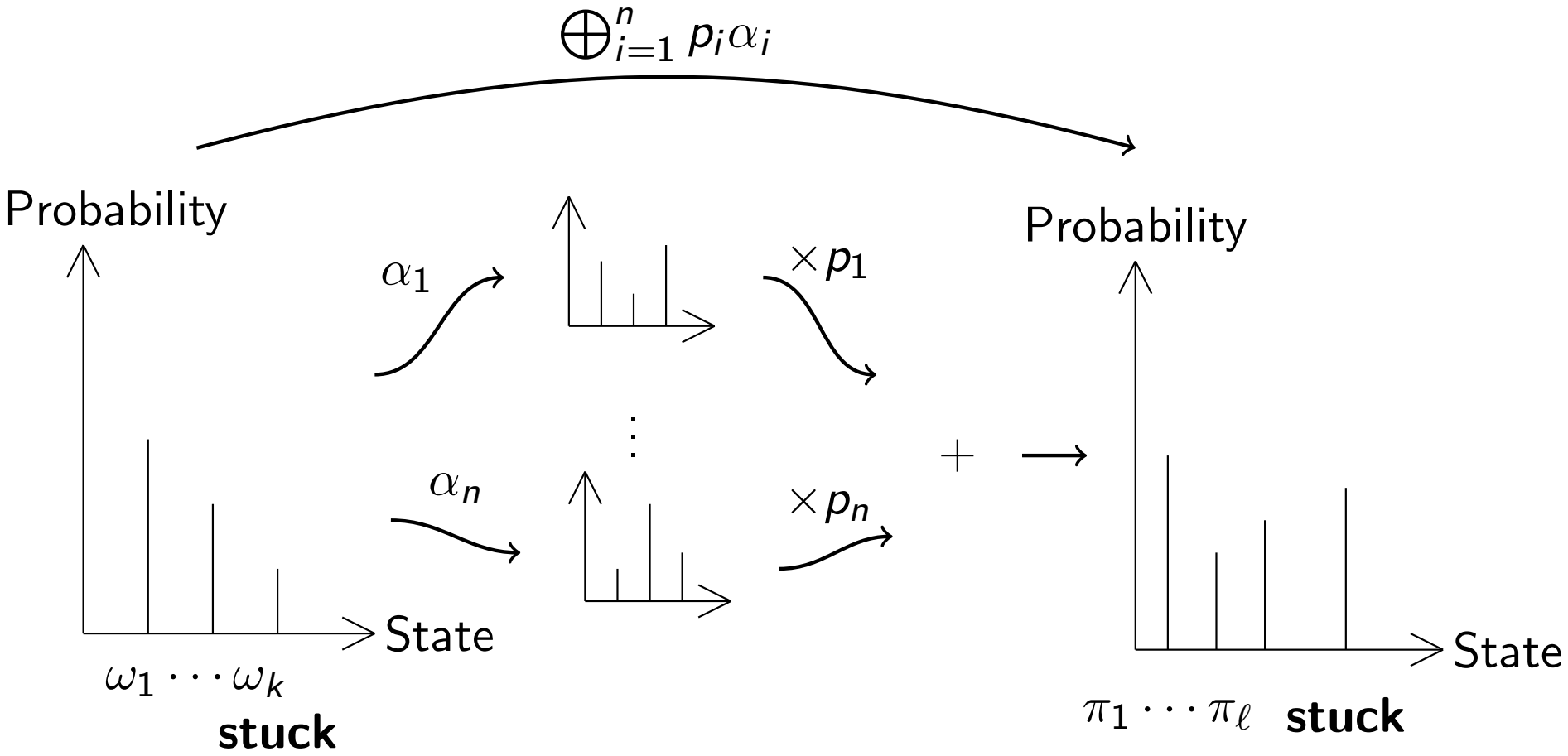
Program Semantics: Sequential Composition



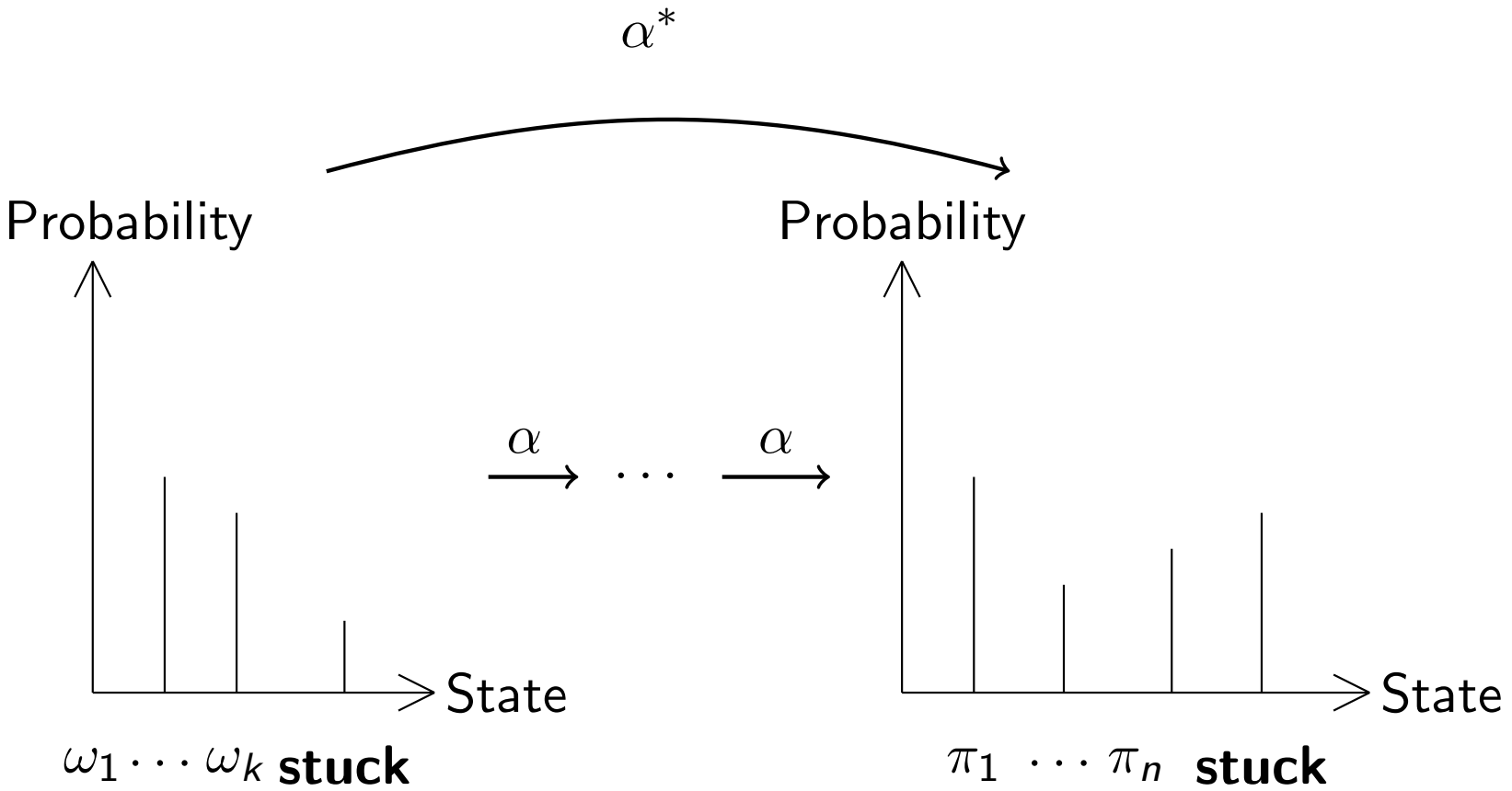
Program Semantics: Nondeterministic Choice



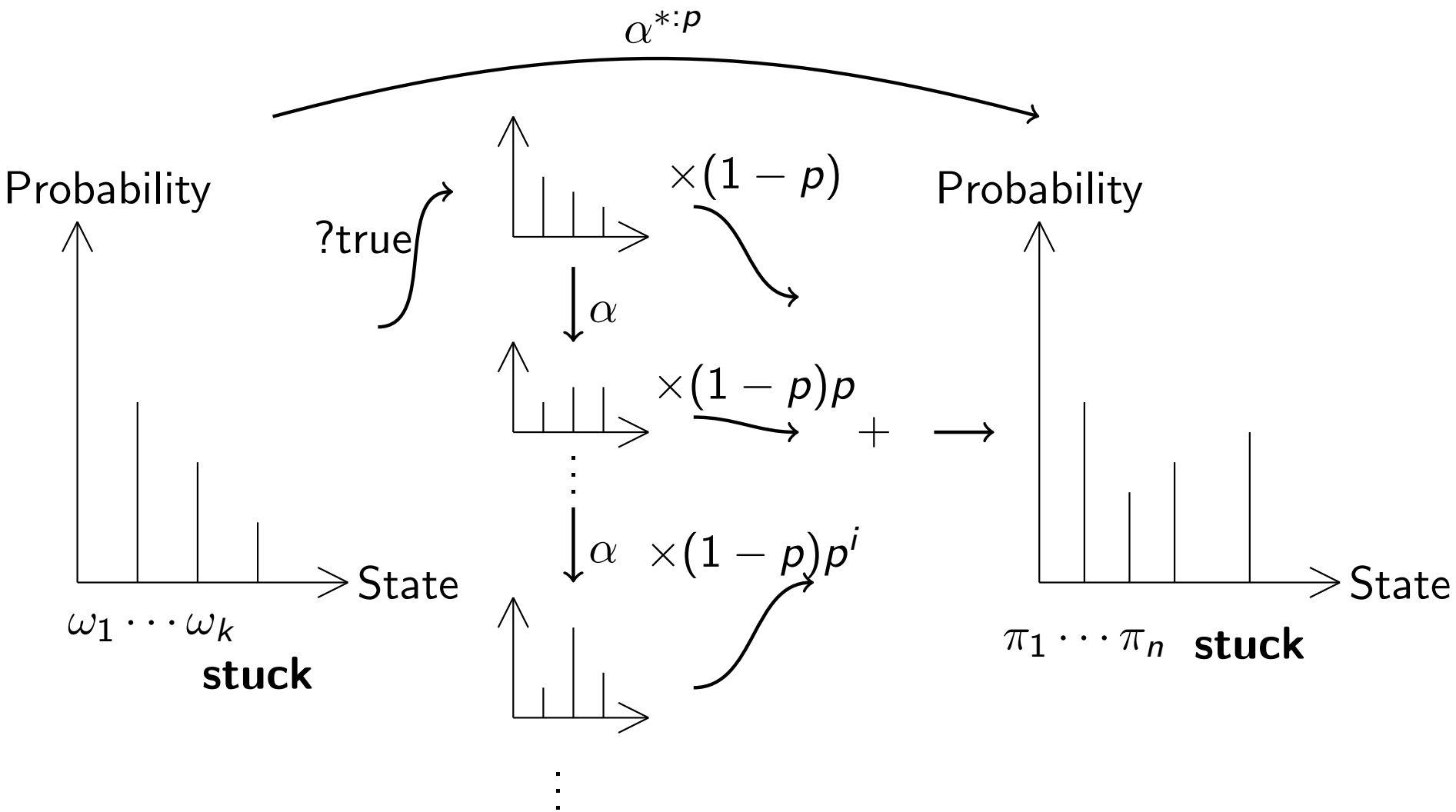
Program Semantics: Probabilistic Choice



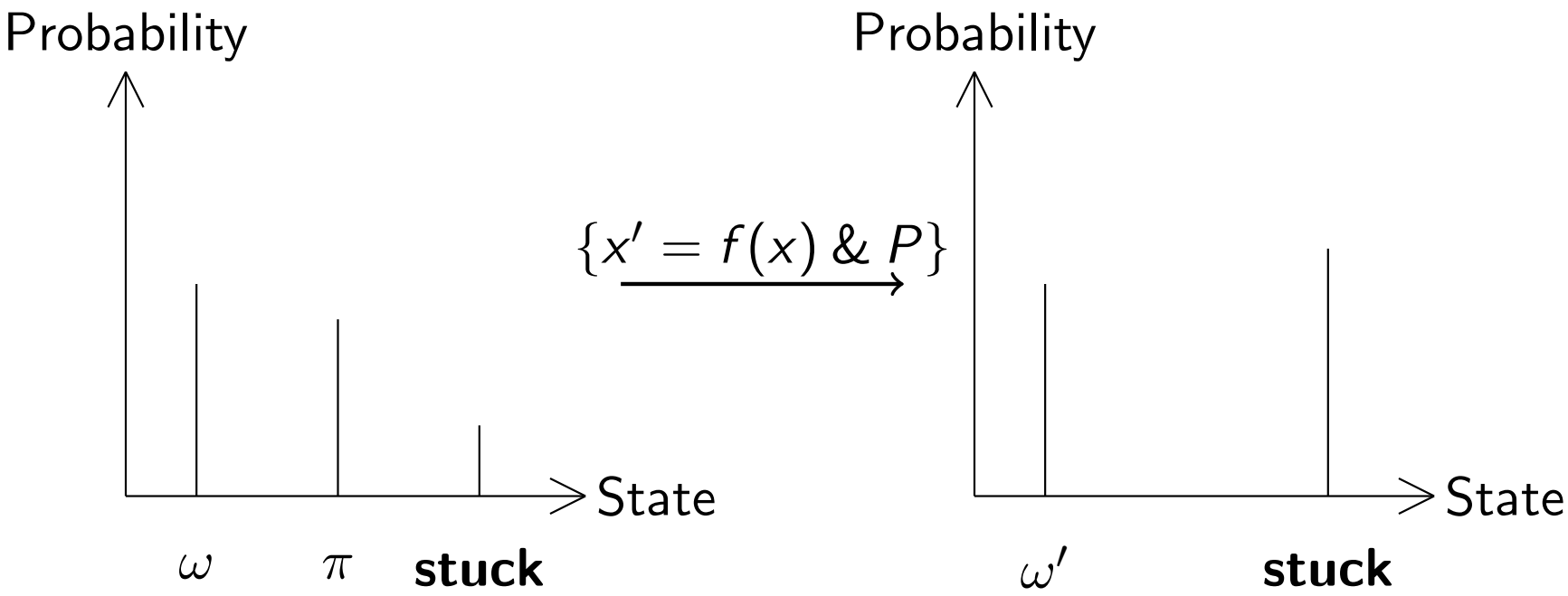
Program Semantics: Nondeterministic Loop



Program Semantics: Probabilistic Loop



Program Semantics: ODE



Formula Semantics

- Mostly the same as in d$\mathcal{L}$

Formula Semantics

- Mostly the same as in d$\mathcal{L}$
- Modalities need to be modified

Formula Semantics

- Mostly the same as in d$\mathcal{L}$
- Modalities need to be modified
- Probability bound:

$$\omega \in [\![\langle\!|\alpha|\!\rangle P \leq p]\!] \text{ iff } \omega \in [\![\sigma(\rho(\alpha, P), p)]\!]$$

Probability Bound

$$\sigma(\rho(\alpha, P), p)$$

- $\mathcal{I}$: indicator

$$\mathcal{I}(P, y) \equiv (P \wedge (y = 1)) \vee (\neg P \wedge (y = 0))$$

Probability Bound

$$\sigma(\rho(\alpha, P), p)$$

- $\mathcal{I}$: indicator

$$\mathcal{I}(P, y) \equiv (P \wedge (y = 1)) \vee (\neg P \wedge (y = 0))$$

- ρ: bound probability

$$\rho(\alpha, P) \equiv (\text{preconditions}, \text{bound})$$

Probability Bound

$$\sigma(\rho(\alpha, P), p)$$

- $\mathcal{I}$: indicator

$$\mathcal{I}(P, y) \equiv (P \wedge (y = 1)) \vee (\neg P \wedge (y = 0))$$

- ρ: bound probability

$$\rho(\alpha, P) \equiv (\text{preconditions}, \text{bound})$$

- σ: finalize ρ

$$\sigma((\{P_1, \ldots, P_n\}, e), p) \equiv \left(\bigwedge_{i=1}^{n} P_i \right) \rightarrow (e \leq p)$$

Probability Bound: ρ

$$\rho(\cdot, P) = (\{\mathcal{I}(P, y)\}, y) \qquad (y \text{ fresh})$$

Probability Bound: ρ

$$\rho(\cdot, P) = (\{\mathcal{I}(P, y)\}, y) \qquad (y \text{ fresh})$$

$$\rho(\beta; \alpha_1 \cup \alpha_2, P) = (X_1 \cup X_2, \max(e_1, e_2))$$

$$\text{where } (X_i, e_i) = \rho(\beta; \alpha_i, P)$$

Probability Bound: ρ

$$\rho(\cdot, P) = (\{\mathcal{I}(P, y)\}, y) \qquad (y \text{ fresh})$$

$$\rho(\beta; \alpha_1 \cup \alpha_2, P) = (X_1 \cup X_2, \max(e_1, e_2))$$
$$\text{where } (X_i, e_i) = \rho(\beta; \alpha_i, P)$$

$$\rho\left(\beta; \bigoplus_{i=1}^{n} p_i \alpha_i, P\right) = \left(\bigcup_{i=1}^{n} X_i, \sum_{i=1}^{n} p_i e_i\right)$$
$$\text{where } (X_i, e_i) = \rho(\beta; \alpha_i, P)$$

Probability Bound: ρ

$$\rho(\cdot, P) = (\{\mathcal{I}(P, y)\}, y) \qquad (y \text{ fresh})$$

$$\rho(\beta; \alpha_1 \cup \alpha_2, P) = (X_1 \cup X_2, \max(e_1, e_2))$$
$$\text{where } (X_i, e_i) = \rho(\beta; \alpha_i, P)$$

$$\rho\left(\beta; \bigoplus_{i=1}^{n} p_i \alpha_i, P\right) = \left(\bigcup_{i=1}^{n} X_i, \sum_{i=1}^{n} p_i e_i\right)$$
$$\text{where } (X_i, e_i) = \rho(\beta; \alpha_i, P)$$

$$\rho(\beta; x := \{p_1 : e_1, \ldots, p_n : e_n\}, P) = \rho\left(\beta; \bigoplus_{i=1}^{n} p_i \{x := e_i\}, P\right)$$

Probability Bound: ρ

$$\rho(\cdot, P) = (\{\mathcal{I}(P, y)\}, y) \qquad (y \text{ fresh})$$

$$\rho(\beta; \alpha_1 \cup \alpha_2, P) = (X_1 \cup X_2, \max(e_1, e_2))$$
$$\text{where } (X_i, e_i) = \rho(\beta; \alpha_i, P)$$

$$\rho\left(\beta; \bigoplus_{i=1}^{n} p_i \alpha_i, P\right) = \left(\bigcup_{i=1}^{n} X_i, \sum_{i=1}^{n} p_i e_i\right)$$
$$\text{where } (X_i, e_i) = \rho(\beta; \alpha_i, P)$$

$$\rho(\beta; x := \{p_1 : e_1, \ldots, p_n : e_n\}, P) = \rho\left(\beta; \bigoplus_{i=1}^{n} p_i \{x := e_i\}, P\right)$$

$$\rho(\beta; \alpha, P) = \rho(\beta, \langle\alpha\rangle P)$$

Loop Probability Bounds

$$\rho(\beta; \alpha^*, P) = \left(\bigcup_{i=0}^{\infty} X_i, \sup_i e_i \right) \text{ where } (X_i, e_i) = \rho(\beta; \alpha^i, P) \text{ for } i \in \mathbb{N}$$

Loop Probability Bounds

$$\rho(\beta; \alpha^*, P) = \left(\bigcup_{i=0}^{\infty} X_i, \sup_i e_i \right) \text{ where } (X_i, e_i) = \rho(\beta; \alpha^i, P) \text{ for } i \in \mathbb{N}$$

What is sup?

Loop Probability Bounds

$$\rho(\beta; \alpha^*, P) = \left(\bigcup_{i=0}^{\infty} X_i, \sup_i e_i \right) \text{ where } (X_i, e_i) = \rho(\beta; \alpha^i, P) \text{ for } i \in \mathbb{N}$$

What is sup?

Theorem (Loop unrolling)

$$[\![\alpha^*]\!] = [\![\{?\mathit{true}\} \cup \{\alpha; \alpha^*\}]\!]$$

Loop Probability Bounds

$$\rho(\beta; \alpha^*, P) = \left(\bigcup_{i=0}^{\infty} X_i, \sup_i e_i \right) \text{ where } (X_i, e_i) = \rho(\beta; \alpha^i, P) \text{ for } i \in \mathbb{N}$$

What is sup?

Theorem (Loop unrolling)

$$[\![\alpha^*]\!] = [\![\{?\mathit{true}\} \cup \{\alpha; \alpha^*\}]\!]$$

Theorem (Probabilistic loop unrolling)

$$[\![\alpha^{*:p}]\!] = [\![(1-p)\{?\mathit{true}\} \oplus p\{\alpha; \alpha^{*:p}\}]\!]$$

Translation to d$\mathcal{L}$: Formulas

Mostly the same as d$\mathcal{L}$

Translation to d$\mathcal{L}$: Formulas

Mostly the same as d$\mathcal{L}$

$$\frac{\alpha \overset{\mathsf{d}\mathcal{L}}{\rightsquigarrow} \overline{\alpha} \quad P \overset{\mathsf{d}\mathcal{L}}{\rightsquigarrow} \overline{P}}{[\alpha]P \overset{\mathsf{d}\mathcal{L}}{\rightsquigarrow} [\overline{\alpha}]\overline{P}} \ (\mathsf{d}\mathcal{L}\text{-}[])$$

Translation to d$\mathcal{L}$: Formulas

Mostly the same as d$\mathcal{L}$

$$\frac{\alpha \overset{\mathrm{d}\mathcal{L}}{\rightsquigarrow} \overline{\alpha} \quad P \overset{\mathrm{d}\mathcal{L}}{\rightsquigarrow} \overline{P}}{[\alpha]P \overset{\mathrm{d}\mathcal{L}}{\rightsquigarrow} [\overline{\alpha}]\overline{P}} \ (\mathrm{d}\mathcal{L}\text{-}[])$$

$$\frac{\sigma(\rho(\alpha, P), p) \overset{\mathrm{d}\mathcal{L}}{\rightsquigarrow} Q}{\langle\!\langle \alpha \rangle\!\rangle P \leq p \overset{\mathrm{d}\mathcal{L}}{\rightsquigarrow} Q} \ (\mathrm{d}\mathcal{L}\text{-}\langle\!\langle\rangle\!\rangle)$$

Translation to d$\mathcal{L}$: Programs

Replace probabilistic operators with nondeterministic operators

Translation to d$\mathcal{L}$: Programs

Replace probabilistic operators with nondeterministic operators

$$\frac{\alpha \overset{\mathsf{d}\mathcal{L}}{\rightsquigarrow} \overline{\alpha} \quad 0 < p < 1}{\alpha^{*:p} \overset{\mathsf{d}\mathcal{L}}{\rightsquigarrow} \overline{\alpha}^{*}} \; (\mathsf{d}\mathcal{L}\text{-}*\text{:}p)$$

Translation to d$\mathcal{L}$: Programs

Replace probabilistic operators with nondeterministic operators

$$\frac{\alpha \overset{\mathrm{d}\mathcal{L}}{\rightsquigarrow} \overline{\alpha} \quad 0 < p < 1}{\alpha^{*:p} \overset{\mathrm{d}\mathcal{L}}{\rightsquigarrow} \overline{\alpha}^{*}} \ (\mathrm{d}\mathcal{L}\text{-}*\text{:}p)$$

$$\frac{\forall 1 \leq i \leq n,\ \alpha_i \overset{\mathrm{d}\mathcal{L}}{\rightsquigarrow} \overline{\alpha_i}}{\bigoplus_{i=1}^{n} p_i \alpha_i \overset{\mathrm{d}\mathcal{L}}{\rightsquigarrow} \bigcup_{\substack{i=1 \\ p_i \neq 0}}^{n} \overline{\alpha_i}} \ (\mathrm{d}\mathcal{L}\text{-}\oplus)$$

Theorems

Theorem (Soundness of Translation)

Let $P \overset{\mathrm{d}\mathcal{L}}{\rightsquigarrow} \overline{P}$. *$P$ is valid in NDS*$\mathrm{d}\mathcal{L}$ *iff* $\overline{P}$ *is valid in* $\mathrm{d}\mathcal{L}$.

Theorems

Theorem (Soundness of Translation)

Let $P \overset{\mathrm{d}\mathcal{L}}{\rightsquigarrow} \overline{P}$. *$P$ is valid in NDS*$\mathrm{d}\mathcal{L}$ *iff* $\overline{P}$ *is valid in* $\mathrm{d}\mathcal{L}$.

Theorem (Probability Bound)

If $\omega \vDash \langle\!\langle \alpha \rangle\!\rangle P \leq p$ *then* $\forall (Det(\omega), \nu) \in [\![\alpha]\!]$, $\mathbf{P}\left[\pi \vDash P\right] \leq p$ *for* $\pi \sim \nu$.

Theorems

Theorem (Soundness of Translation)

Let $P \overset{\mathrm{d}\mathcal{L}}{\rightsquigarrow} \overline{P}$. *$P$ is valid in NDS*$\mathrm{d}\mathcal{L}$ *iff* $\overline{P}$ *is valid in* $\mathrm{d}\mathcal{L}$.

Theorem (Probability Bound)

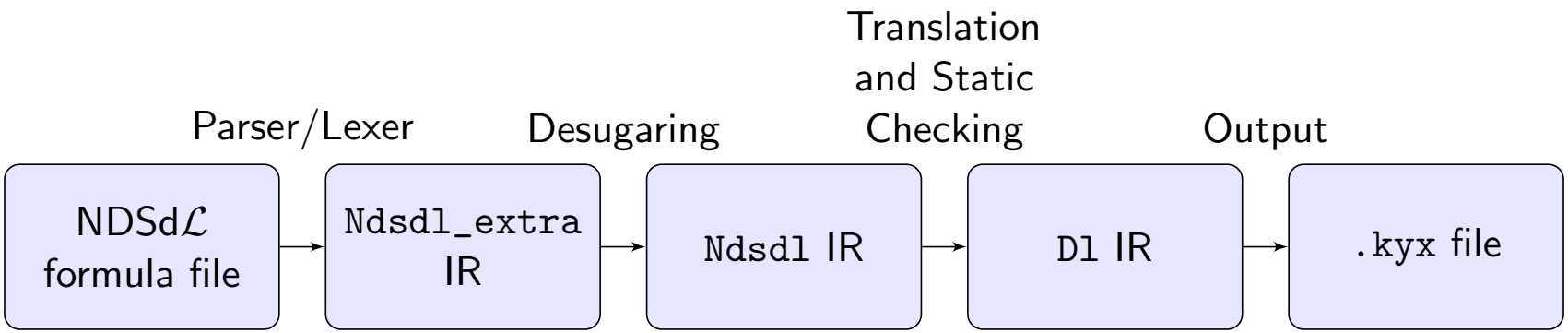
If $\omega \vDash \langle\!\langle \alpha \rangle\!\rangle P \leq p$ *then* $\forall (Det(\omega), \nu) \in [\![\alpha]\!]$, $\mathbf{P}\left[\pi \vDash P\right] \leq p$ *for* $\pi \sim \nu$.

Theorem (Conservative Extension)

Every valid $\mathrm{d}\mathcal{L}$ *formula is also a valid NDS*$\mathrm{d}\mathcal{L}$ *formula.*

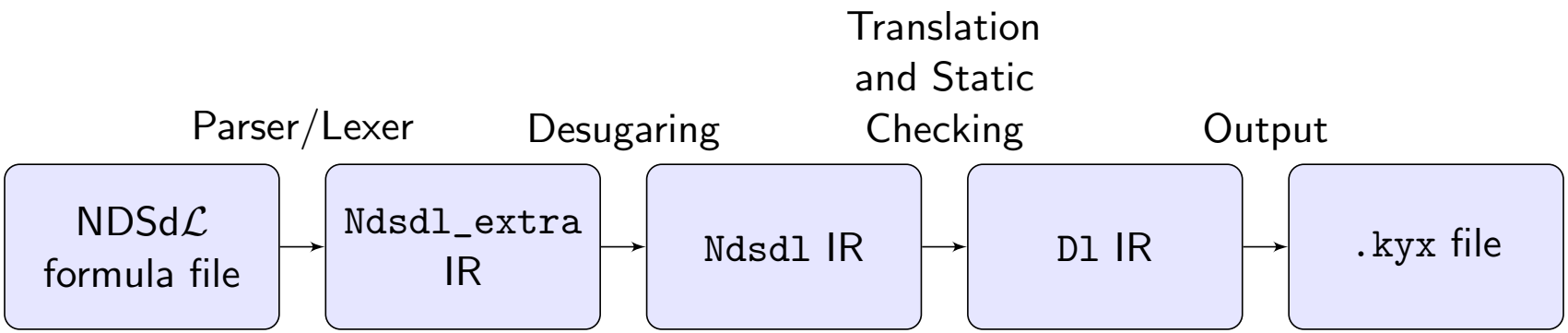
Compiler

- ∼850 lines of OCaml



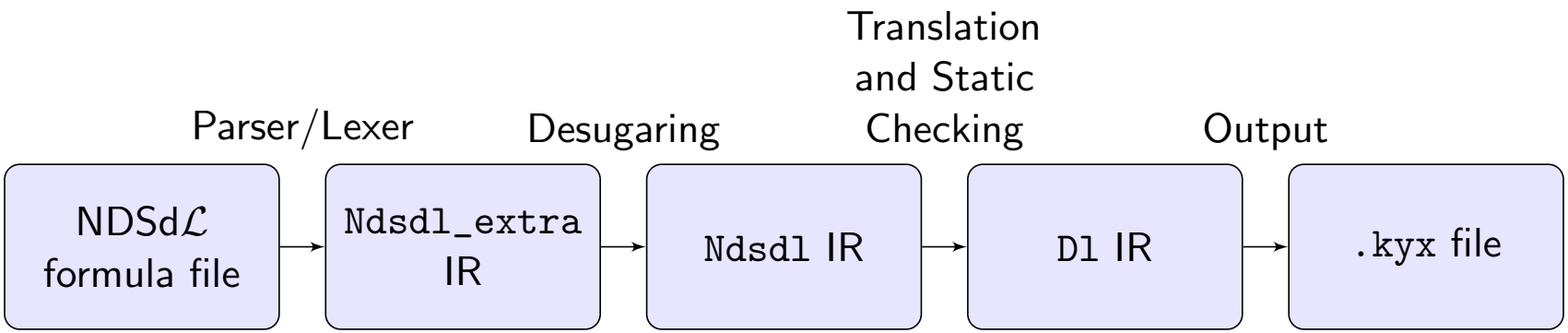
Compiler

- ∼850 lines of OCaml
- Syntactic sugar:



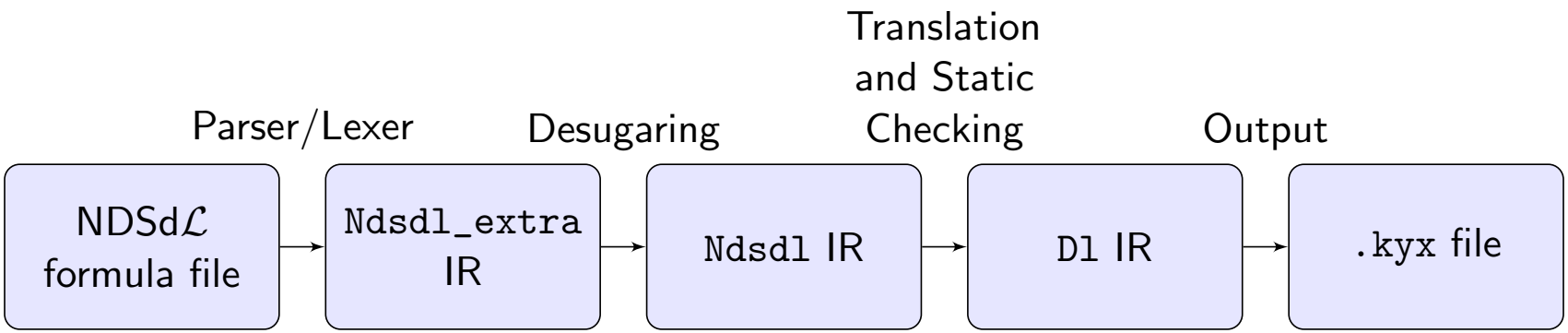
Compiler

- ∼850 lines of OCaml
- Syntactic sugar:
  - `x := Bernoulli(p)` ≡ `x := {p:1, (1-p):0}`



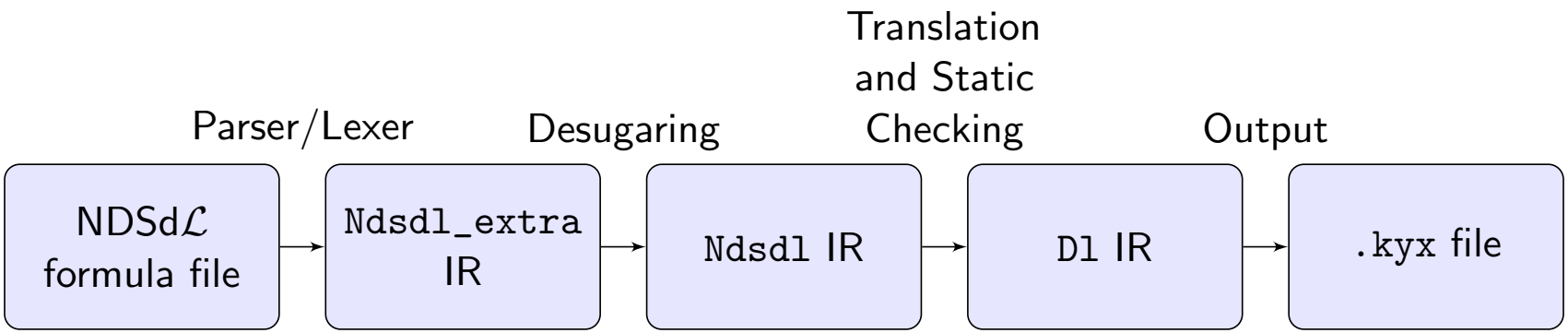
Compiler

- ~850 lines of OCaml
- Syntactic sugar:
  - `x := Bernoulli(p)` $\equiv$ `x := {p:1, (1-p):0}`
  - `x := Geometric(p)` $\equiv$ `x := 1; {x := x+1}*:(1-p)`



Compiler

- ~850 lines of OCaml
- Syntactic sugar:
  - `x := Bernoulli(p)` $\equiv$ `x := {p:1, (1-p):0}`
  - `x := Geometric(p)` $\equiv$ `x := 1; {x := x+1}*:(1-p)`
  - `{a}*@unroll(n)`, `{a}*:p@unroll(n)`, `x := Geometric(p)@unroll(n)`



Example

- Geometric distribution with success probability $3/4$

Example

- Geometric distribution with success probability $3/4$
- Want to prove: probability of ≥ 3 trials is bounded by $1/16$

Example

- Geometric distribution with success probability $3/4$
- Want to prove: probability of ≥ 3 trials is bounded by $1/16$

```
<| x := Geometric(3/4) |>(x>=3) <= 1/16
```

Example

- Geometric distribution with success probability $3/4$
- Want to prove: probability of ≥ 3 trials is bounded by $1/16$

```
<| x := Geometric(3/4) |>(x>=3) <= 1/16
```

Not valid!

Example

- Geometric distribution with success probability $3/4$
- Want to prove: probability of ≥ 3 trials is bounded by $1/16$

```
<| x := Geometric(3/4) |>(x>=3) <= 1/16
```

Not valid!

```
<| x := Geometric(3/4)@unroll(2) |>(x>=3) <= 1/16
```

Example (translated)

```
ProgramVariables
  Real tt_0;
  Real tt_1;
  Real tt_2;
  Real x;
End.

Problem
  ((((true) & (((<x := 1;>(<?true;>((x) >= (3)))) & ((tt_0) = (1))) | ((!(<x :=
       1;>(<?true;>((x) >= (3))))) & ((tt_0) = (0))))) & (((<x := 1;>(<x := (x)
       +(1);>(<?true;>((x) >= (3))))) & ((tt_1) = (1))) | ((!(<x := 1;>(<x := (x)
       +(1);>(<?true;>((x) >= (3)))))) & ((tt_1) = (0))))) & (((<x := 1;>(<x := (
       x)+(1);>(<x := (x)+(1);>(<{x := (x)+(1);}*>((x) >= (3)))))) & ((tt_2) =
       (1))) | ((!(<x := 1;>(<x := (x)+(1);>(<x := (x)+(1);>(<{x := (x)+(1);}*>((
       x) >= (3)))))))) & ((tt_2) = (0))))) -> ((((0)+(((1)-((1)-((3)/(4))))*(tt_0
       )))+(((1)-((3)/(4)))*(((0)+(((1)-((1)-((3)/(4))))*(tt_1)))+(((1)-((3)/(4))
       )*(tt_2))))) <= ((1)/(16)))
End.
```

Conclusion

- NDSd$\mathcal{L}$: dynamic logic for stochastic hybrid systems

Conclusion

- NDSd$\mathcal{L}$: dynamic logic for stochastic hybrid systems
- Compiles to d$\mathcal{L}$

Conclusion

- NDSd$\mathcal{L}$: dynamic logic for stochastic hybrid systems
- Compiles to d$\mathcal{L}$
- Soundness theorems

Conclusion

- NDSd$\mathcal{L}$: dynamic logic for stochastic hybrid systems
- Compiles to d$\mathcal{L}$
- Soundness theorems
- Compiler implementation

Conclusion

- NDSd$\mathcal{L}$: dynamic logic for stochastic hybrid systems
- Compiles to d$\mathcal{L}$
- Soundness theorems
- Compiler implementation
- Future work: continuous probability distributions

Questions?

$$\frac{\sigma(\rho(\alpha, P), p) \overset{\mathrm{d}\mathcal{L}}{\rightsquigarrow} Q}{\langle\!|\alpha|\!\rangle P \leq p \overset{\mathrm{d}\mathcal{L}}{\rightsquigarrow} Q} \ (\mathrm{d}\mathcal{L}\text{-}\langle\!|\,|\!\rangle)$$