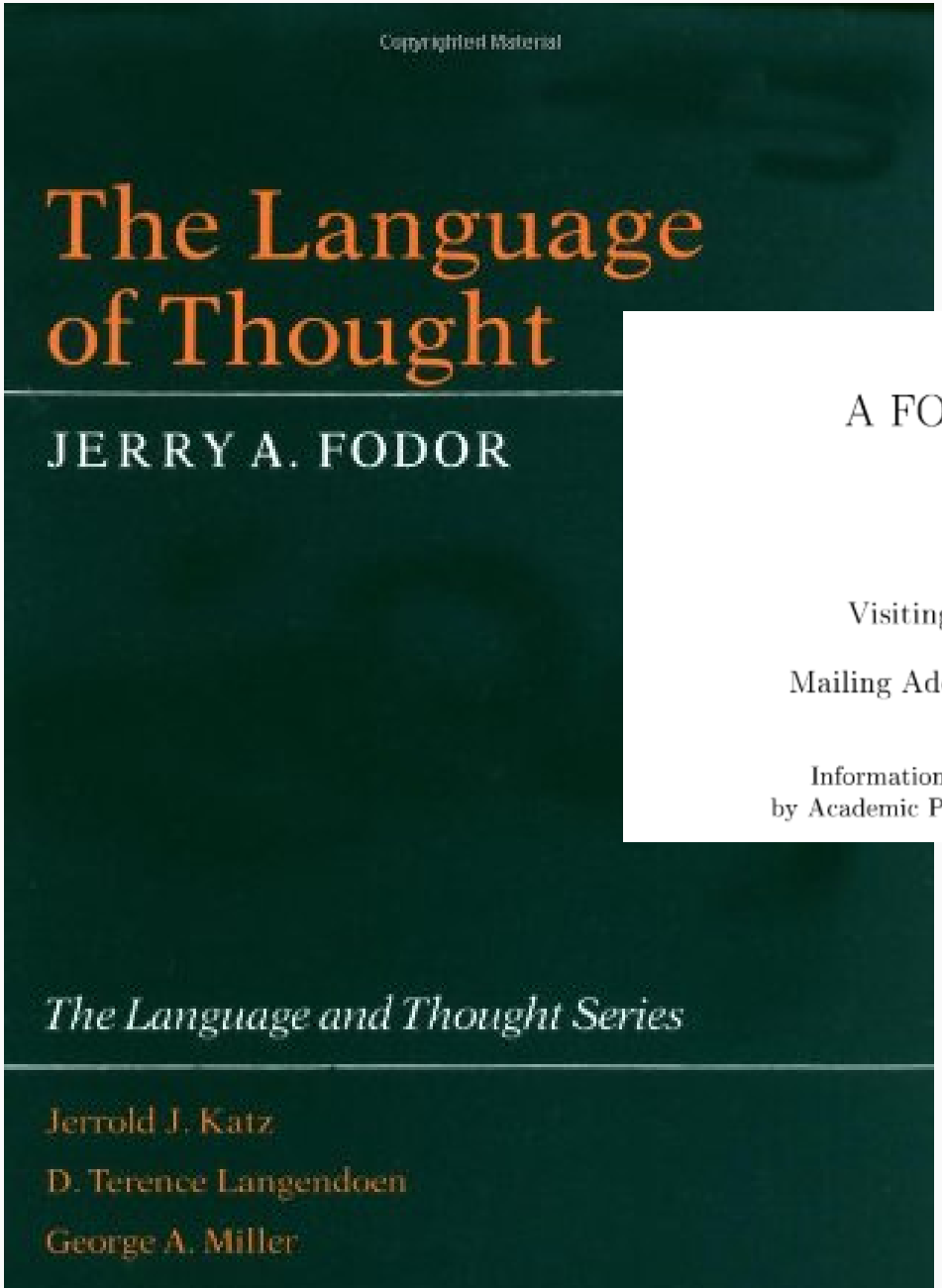
Growing libraries of concepts with wake-sleep program induction

Kevin Ellis & Mathias Sablé Meyer
Joint with: Lucas Morales, Armando Solar-Lezama, Joshua B. Tenenbaum
Heavy inspiration from: Eyal Dechter

July 22, 2018

MIT

The Language of Thought



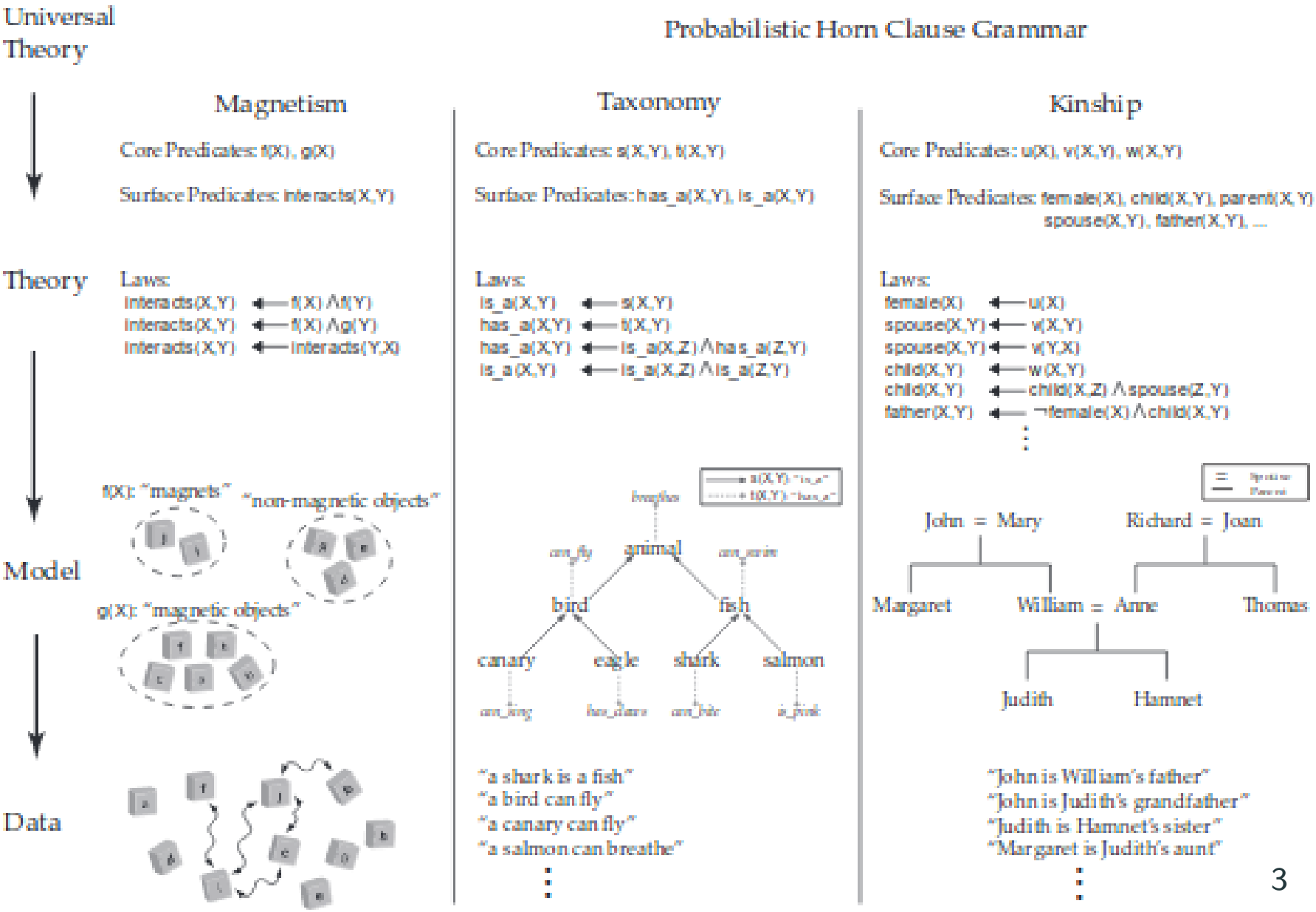
A FORMAL THEORY OF INDUCTIVE INFERENCE, Part I*†

Ray J. Solomonoff

Visiting Professor, Computer Learning Research Center
Royal Holloway, University of London
Mailing Address: P.O.B. 400404, Cambridge, Ma. 02140, U.S.A.

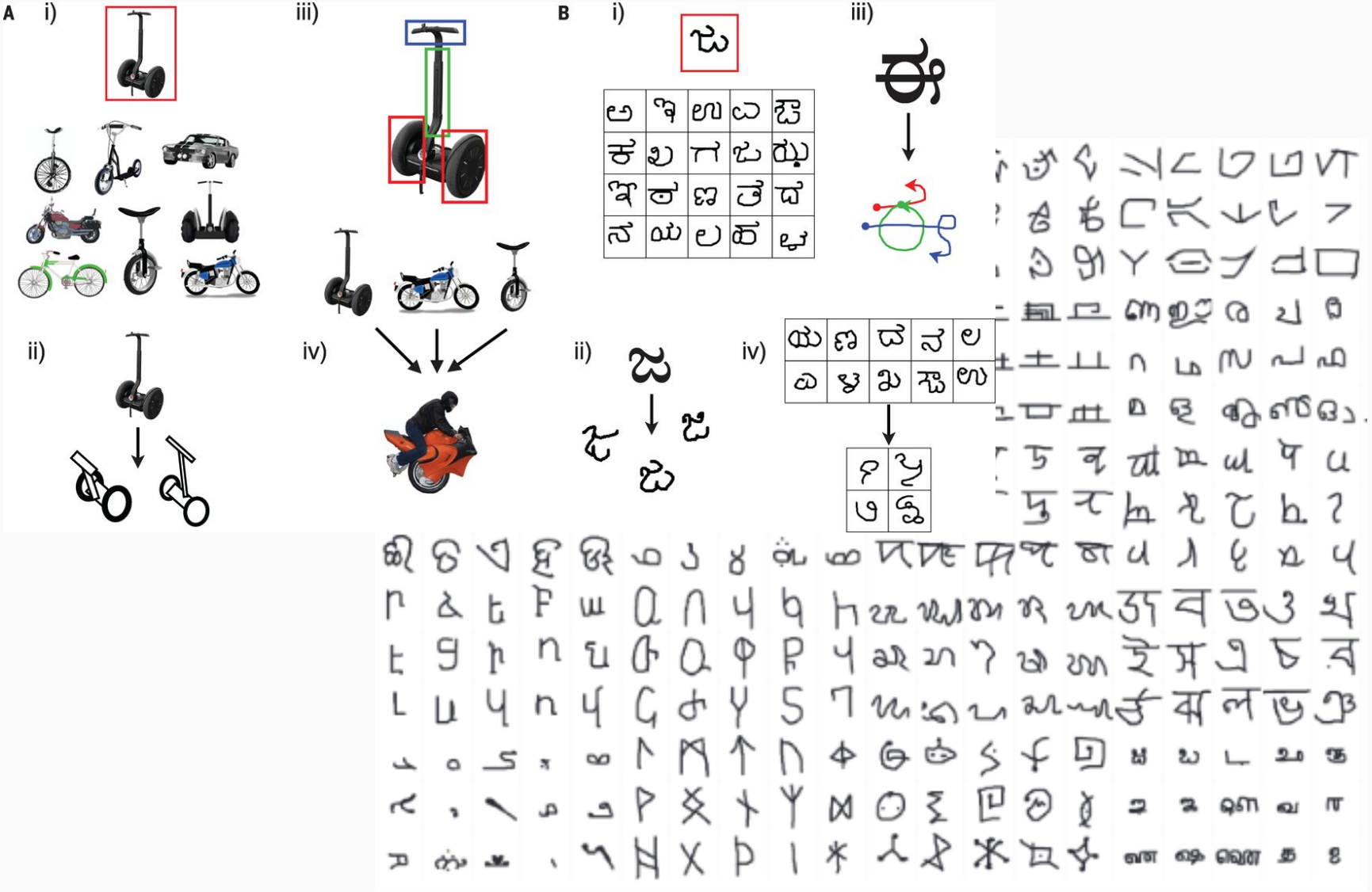
Information and Control, Volume 7, No. 1, Pp. 1-22, March 1964 Copyright by Academic Press Inc.

Engineering the language of thought



Ullman et al 2012

Engineering the language of thought



Lake et al 2015

Growing a domain-specific language of thought

Goal: acquire domain-specific knowledge needed to induce a class of programs

Growing a domain-specific language of thought

Goal: acquire domain-specific knowledge needed to induce a class of programs

- Library of concepts (declarative knowledge)
- Search strategy (procedural knowledge)

DSL: Library of concepts

Tasks and Programs

```
[7 2 3]→[7 3]
[1 2 3 4]→[3 4]
[4 3 2 1]→[4 3]
f(ℓ)=(f1 ℓ (λ (x) (> x 2)))
```

```
[7 3]→False
[3]→False
[9 0 0]→True
[0]→True
[0 7 3]→True
f(ℓ)=(f3 ℓ 0)
```

```
[2 7 8 1]→8
[3 19 14]→19
f(ℓ)=(f2 ℓ)
```

DSL

$f_0(\ell,\mathtt{r}) = \mathtt{(foldr\ r}\ \ell\ \mathtt{cons)}$
(f_0: *Append lists* r *and* ℓ)

$f_1(\ell,\mathtt{p}) = \mathtt{(foldr}\ \ell\ \mathtt{nil\ (}\lambda\ \mathtt{(x\ a)\ (if\ (p\ x)\ (cons\ x\ a)\ a)))}$
(f_1: *Higher-order filter function*)

$f_2(\ell) = \mathtt{(foldr}\ \ell\ \mathtt{0\ (}\lambda\ \mathtt{(x\ a)\ (if\ (>\ a\ x)\ a\ x)))}$
(f_2: *Maximum element in list* ℓ)

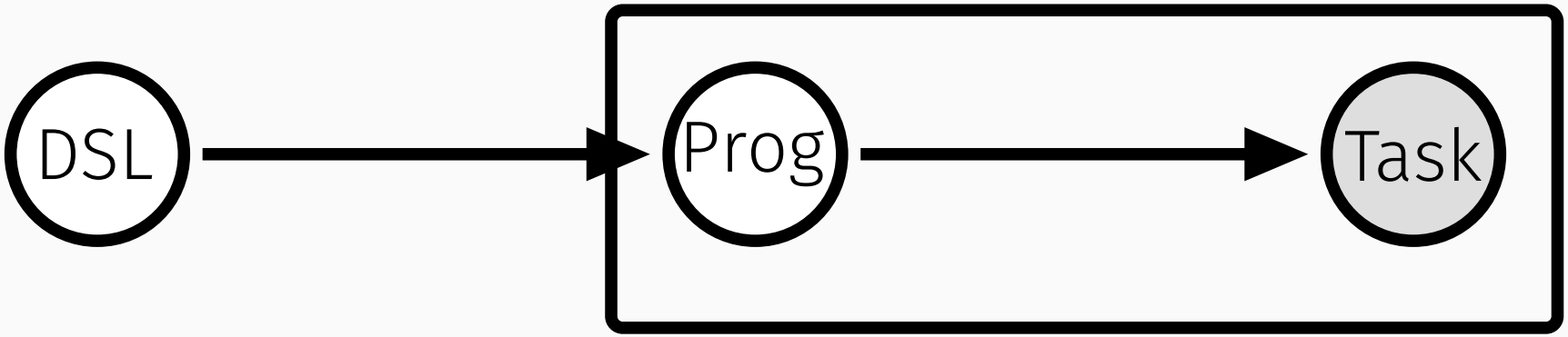
$f_3(\ell,\mathtt{k}) = \mathtt{(foldr}\ \ell\ \mathtt{(is\text{-}nil}\ \ell\mathtt{)\ (}\lambda\ \mathtt{(x\ a)\ (if\ a\ a\ (=\ k\ x))))}$
(f_2: *Whether* ℓ *contains* k)

DreamCoder

- **Wake:** Solve problems by writing programs
- **Sleep:** Improve DSL and neural recognition model:
  - **Sleep-G:** Improve DSL (**G**enerative model)
  - **Sleep-R:** Improve **R**ecognition model

Combines ideas from Wake-Sleep & Exploration-Compression algorithm by Eyal Dechter

DSL learning as Bayesian inference



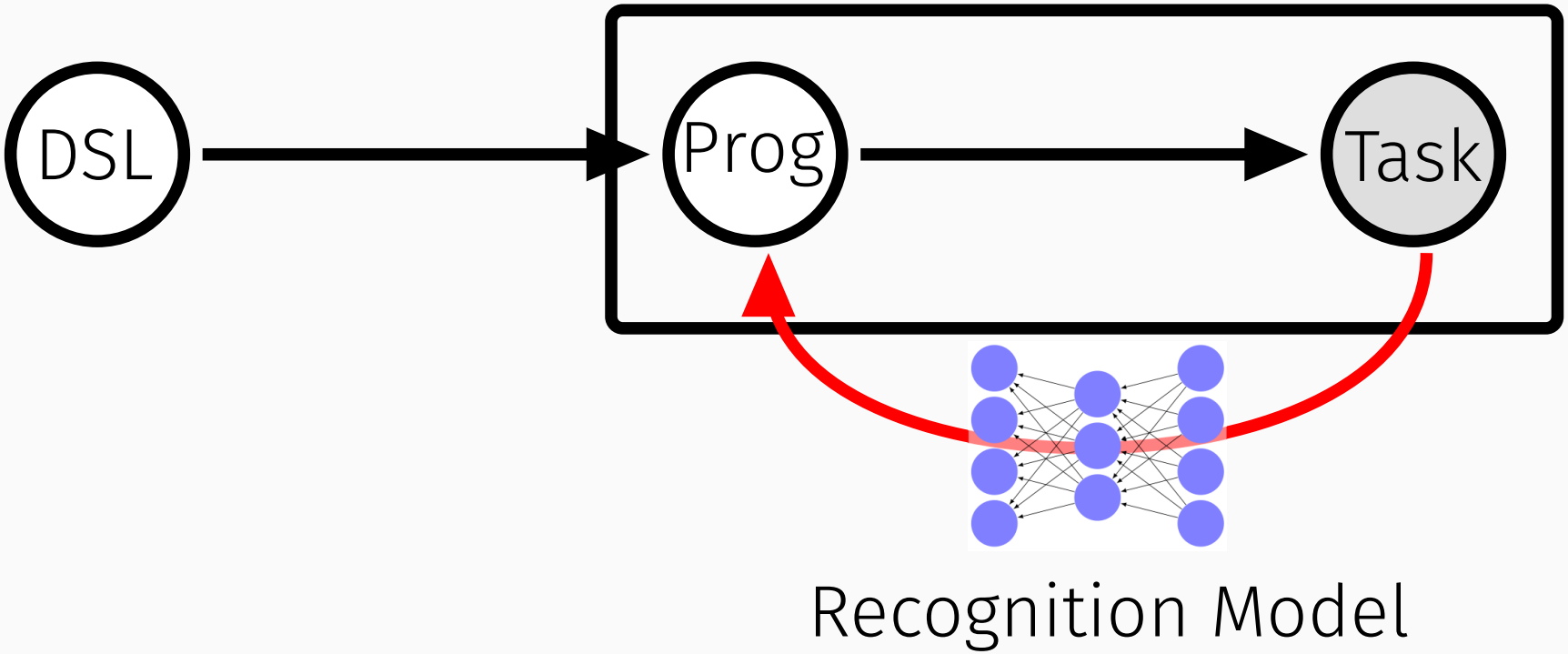
[Dechter et al., 2013] [Liang et al, 2010]; [Lake et al, 2015]

Gray: Observed.

White: Latent.

Boxed (plate): Repeated.

DSL learning as amortized Bayesian inference



Recognition model: Neural net.

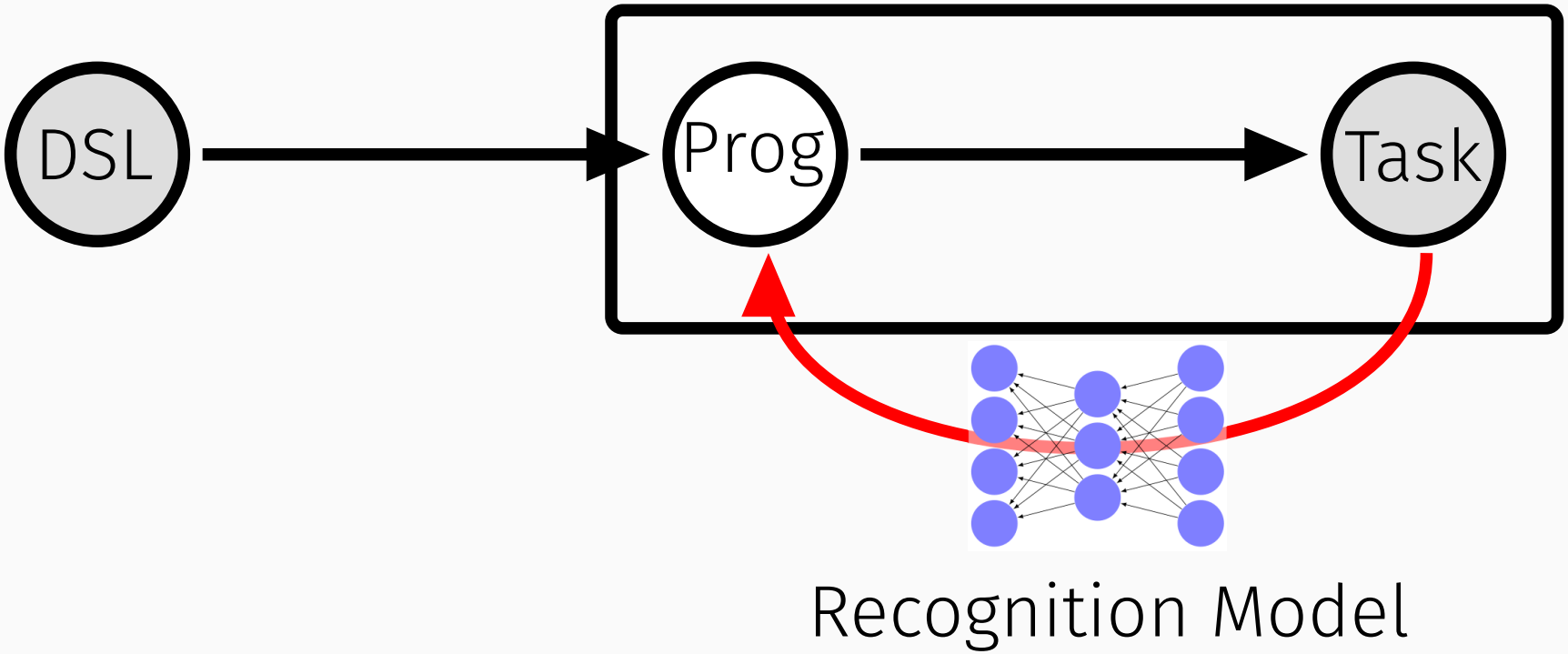
Red: Bottom-up inference.

Gray: Observed.

White: Latent.

Boxed (plate): Repeated.

Wake: Problem solving



Recognition model: Neural net.

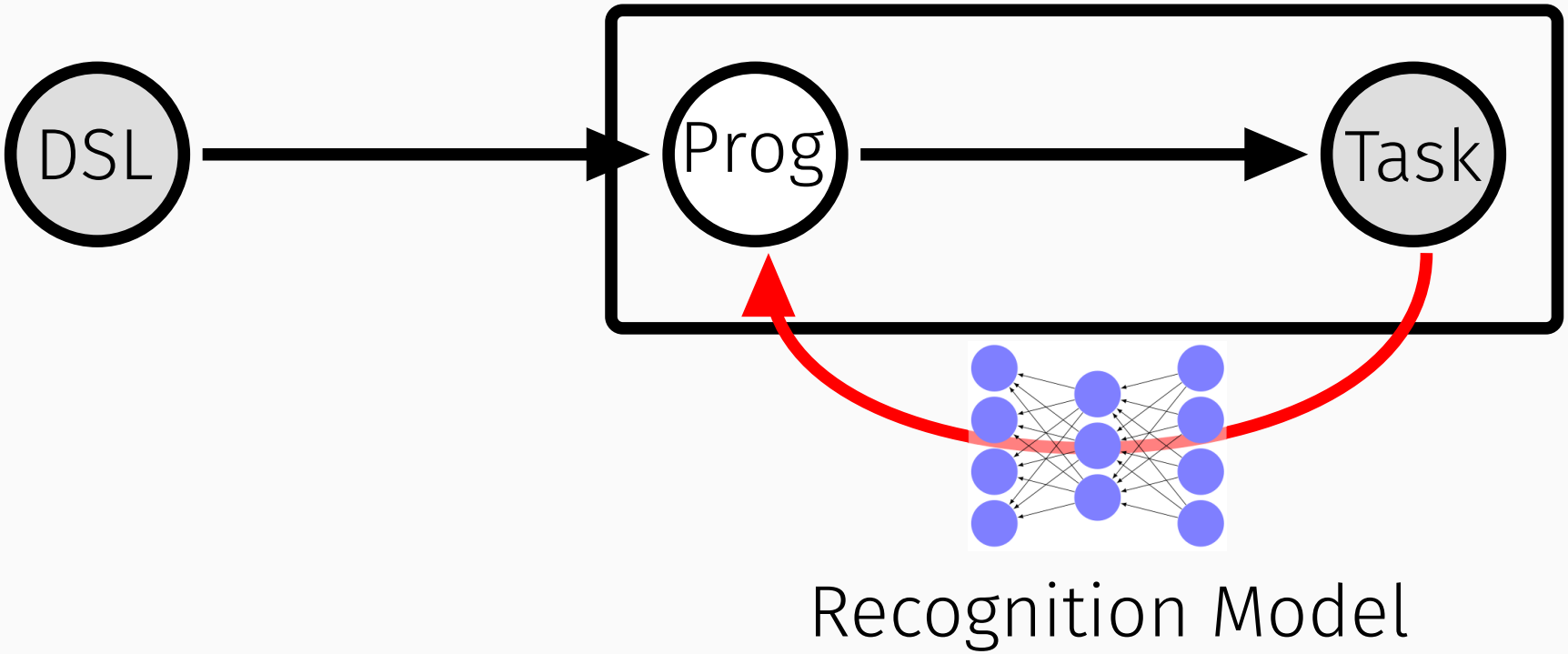
Red: Bottom-up inference.

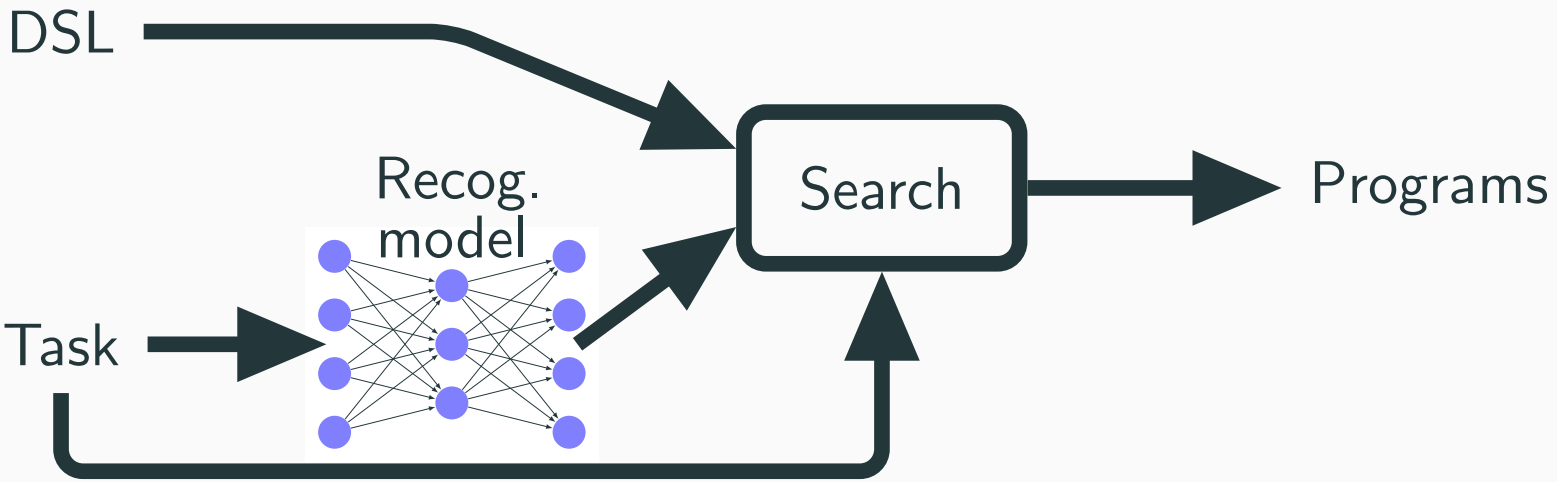
Gray: Observed.

White: Latent.

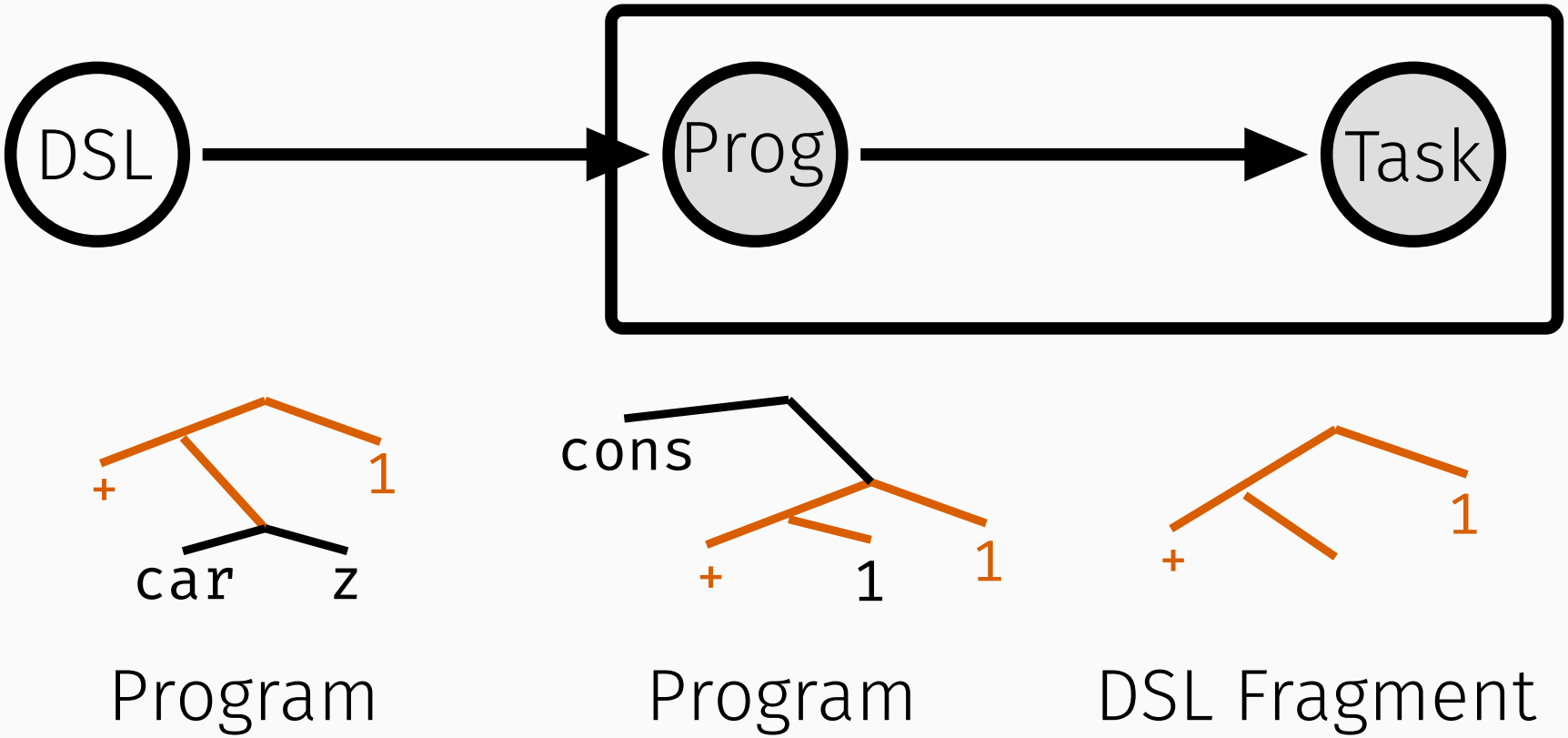
Boxed (plate): Repeated.

Wake: Problem solving





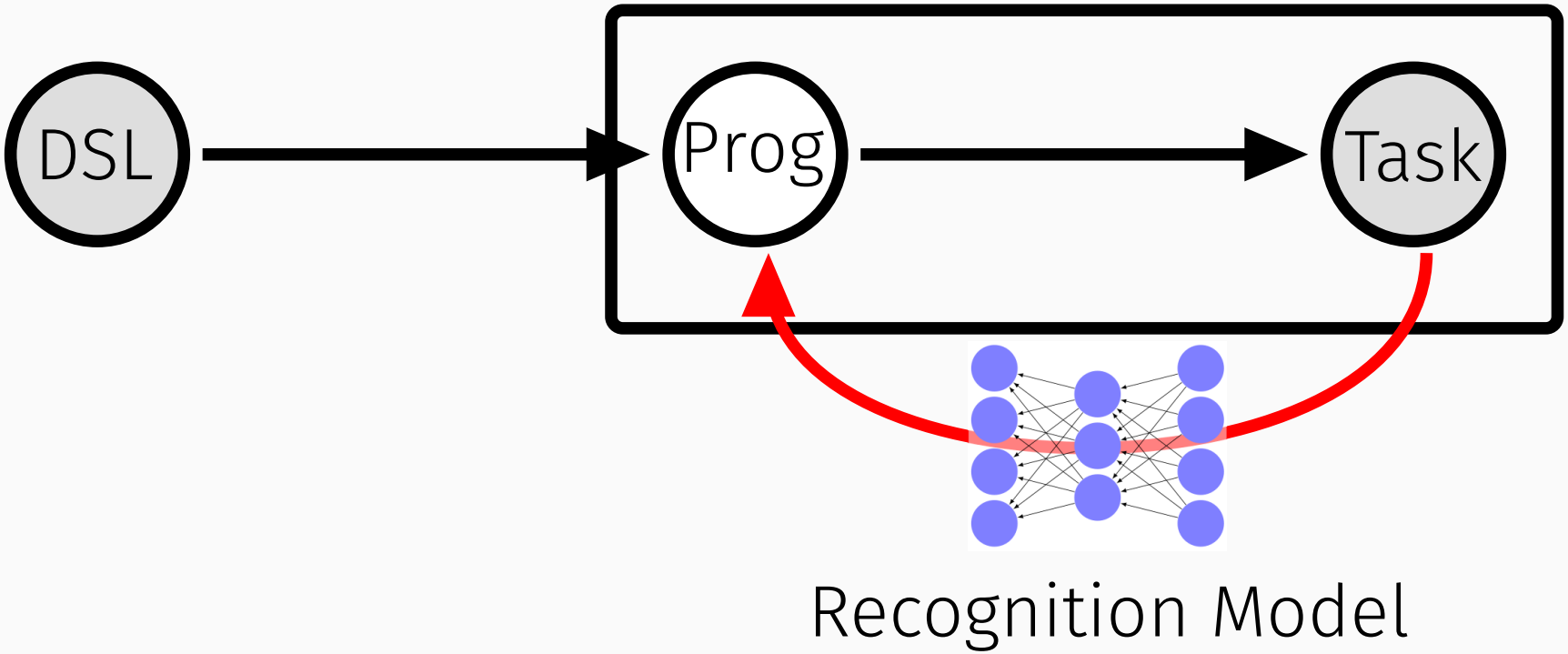
Sleep-G: Memory consolidation



Fragment Grammars: O'Donnell 2015.

Orange: Code fragments.

Sleep-R: Objective

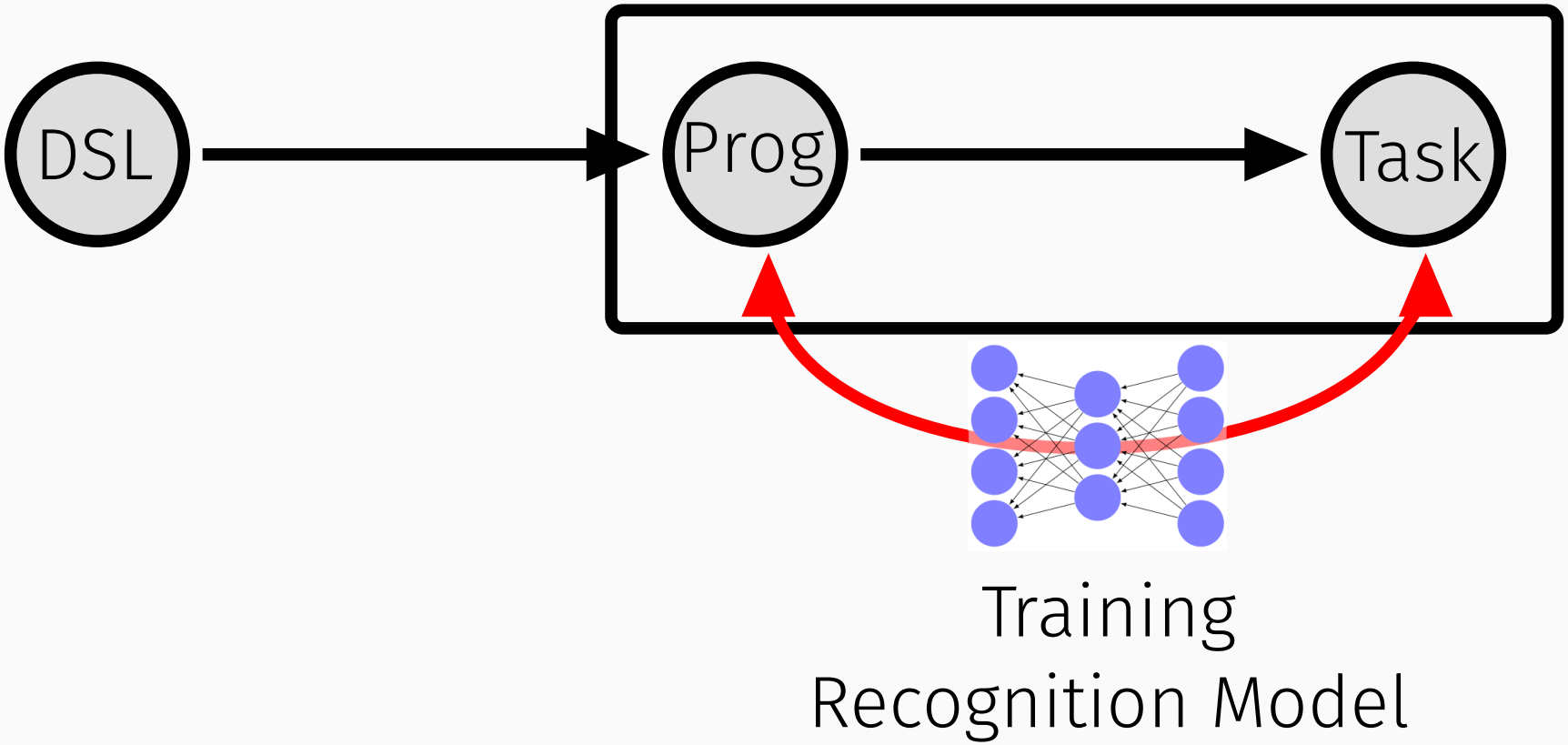


Recognition model predicts distribution over program, conditional on task.

Training: (program, task) pairs

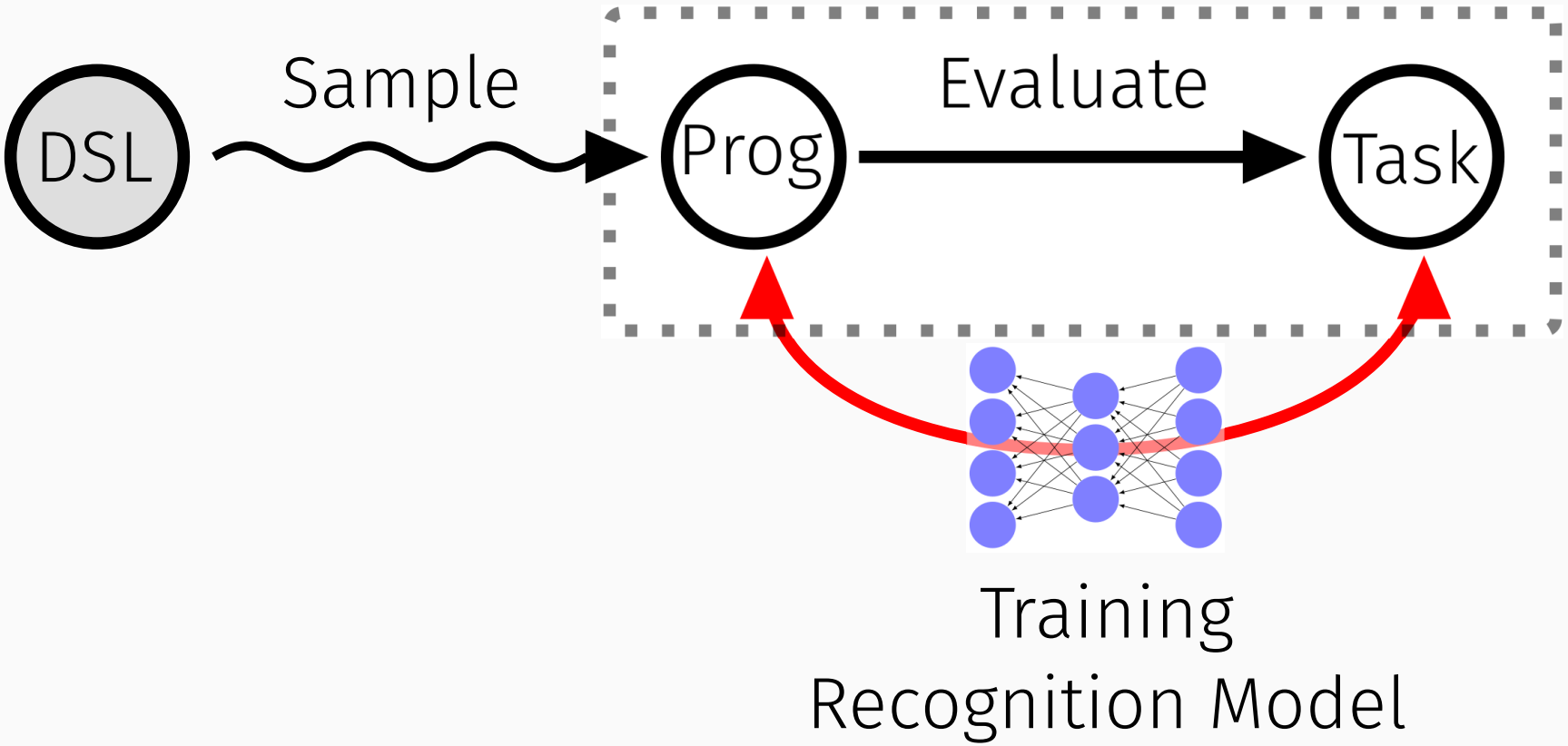
Objective: Predict program w/ (1) high prior under DSL & (2) high likelihood for task

Sleep-R: Experience replay



Train on (program, task) pairs found during waking

Sleep-R: Dreaming



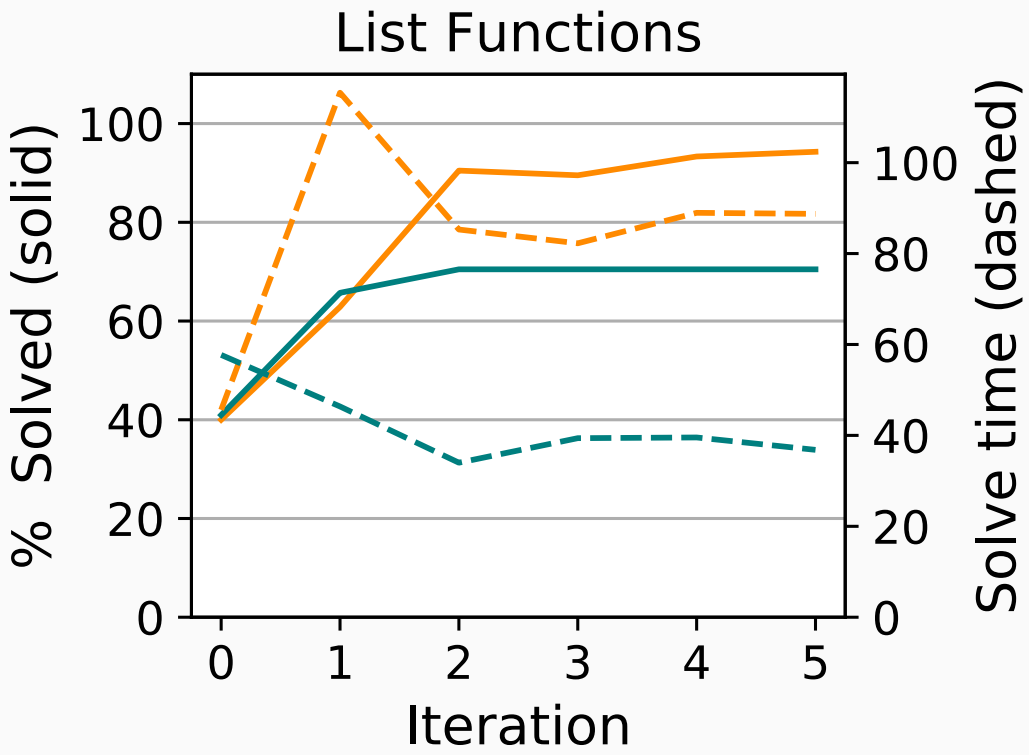
Train on (program, task) pairs sampled from DSL

List functions — Created & investigated by Lucas Morales

Name	Input	Output
repeat-3	[7 0]	[7 0 7 0 7 0]
drop-3	[0 3 8 6 4]	[6 4]
rotate-2	[8 14 1 9]	[1 9 8 14]
count-head-in-tail	[1 2 1 1 3]	2
keep-div-5	[5 9 14 6 3 0]	[5 0]
product	[7 1 6 2]	84

Discovers 38 concepts, including ‘filter’

List functions: Learning curves on hold out tasks



Learning curves for DreamCoder both with (in orange) and without (in teal) the recognition model. Solid lines: % holdout testing tasks solved w/ 10m timeout. Dashed lines: Average solve time, averaged only over tasks that are solved.

Turtle graphics — Created & investigated by Mathias Sablé–Meyer

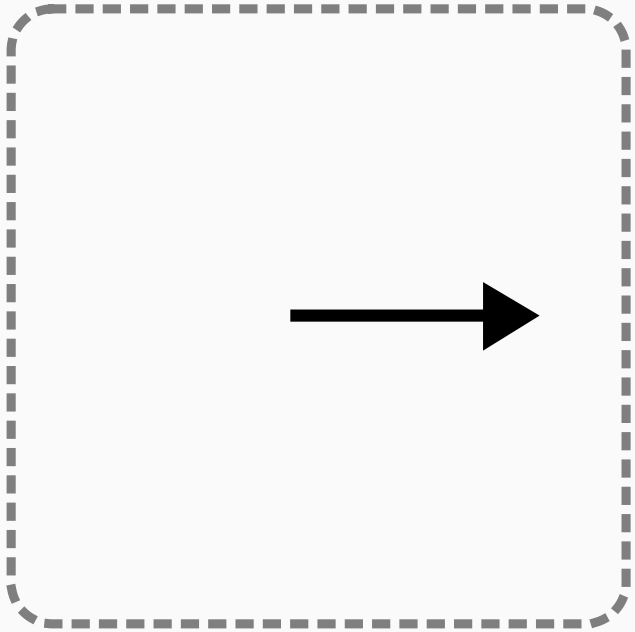
DSL

```
OP ::= FW x | RT x | UP | DOWN | SET state
```

Tasks

```
task :  unit -> image
```

```
FW 1
```



Turtle graphics — Created & investigated by Mathias Sablé–Meyer

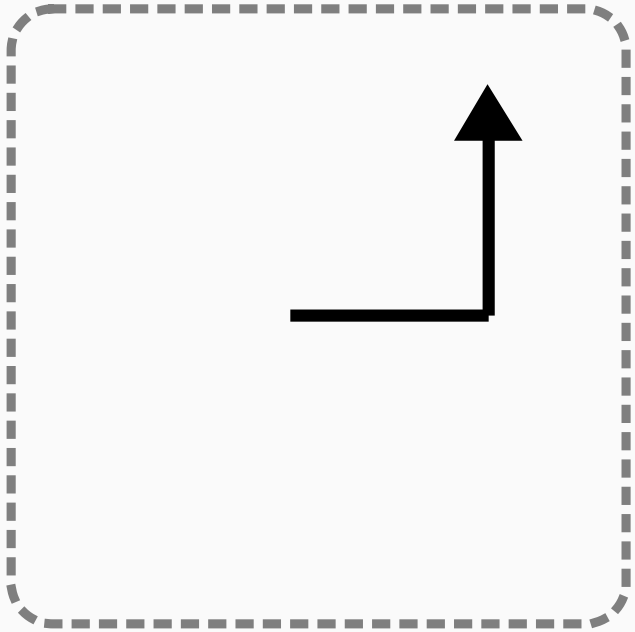
DSL

```
OP ::= FW x | RT x | UP | DOWN | SET state
```

Tasks

```
task :  unit -> image
```

```
FW 1
RT π/2
FW 1
```



Turtle graphics — Created & investigated by Mathias Sablé–Meyer

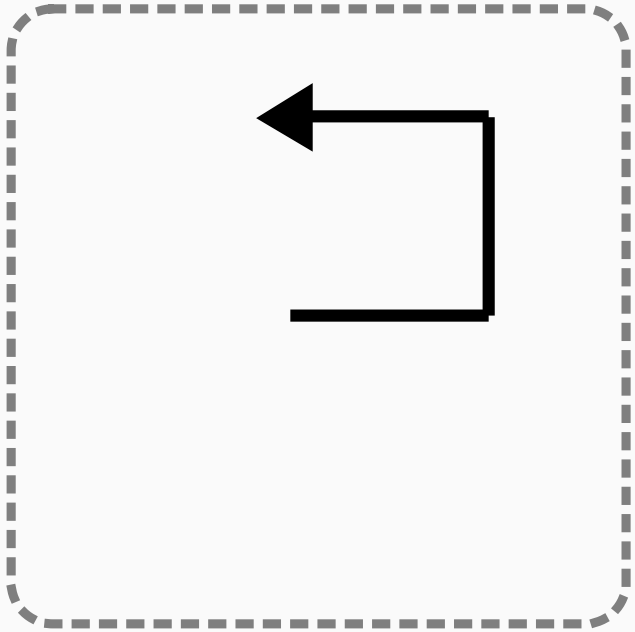
DSL

```
OP ::= FW x | RT x | UP | DOWN | SET state
```

Tasks

```
task :  unit -> image
```

```
FW 1
RT π/2
FW 1
RT π/2
FW 1
```



Turtle graphics — Created & investigated by Mathias Sablé–Meyer

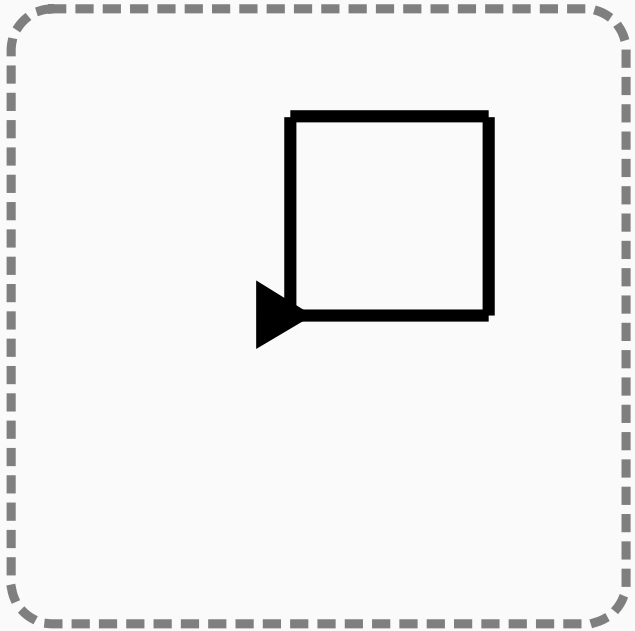
DSL

```
OP ::= FW x | RT x | UP | DOWN | SET state
```

Tasks

```
task :  unit -> image
```

```
for i in range(4)
> FW 1
> RT π/2
```



Turtle graphics — Created & investigated by Mathias Sablé–Meyer

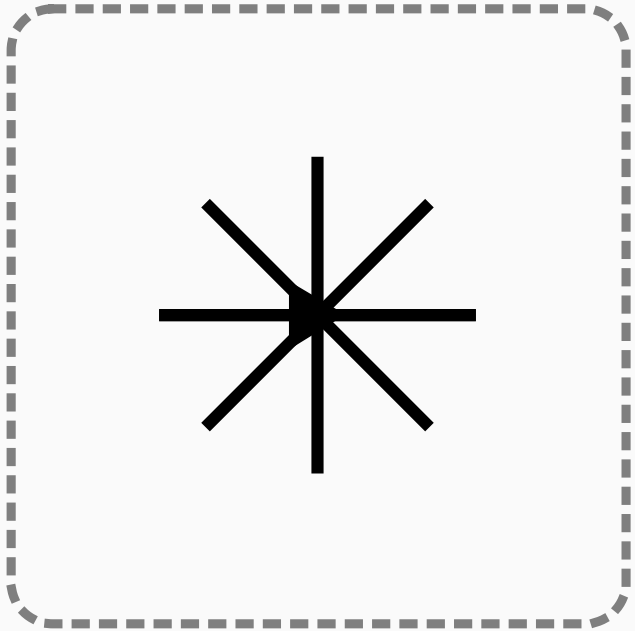
DSL

```
OP ::= FW x | RT x | UP | DOWN | SET state
```

Tasks

```
task :  unit -> image
```

```
for i in range(8)
> FW 1
> SET origin
> RT 2π/8
```



Turtle graphics — Created & investigated by Mathias Sablé–Meyer

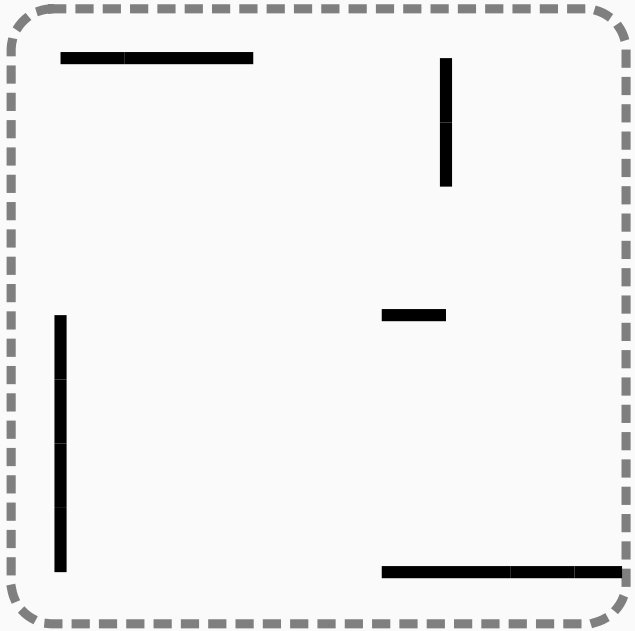
DSL

```
OP ::= FW x | RT x | UP | DOWN | SET state
```

Tasks

```
task :  unit -> image
```

```
for i in range(8)
> PU
> FW i/2
> PD
> FW i/2
> RT π/2
```



Turtle graphics — Created & investigated by Mathias Sablé–Meyer

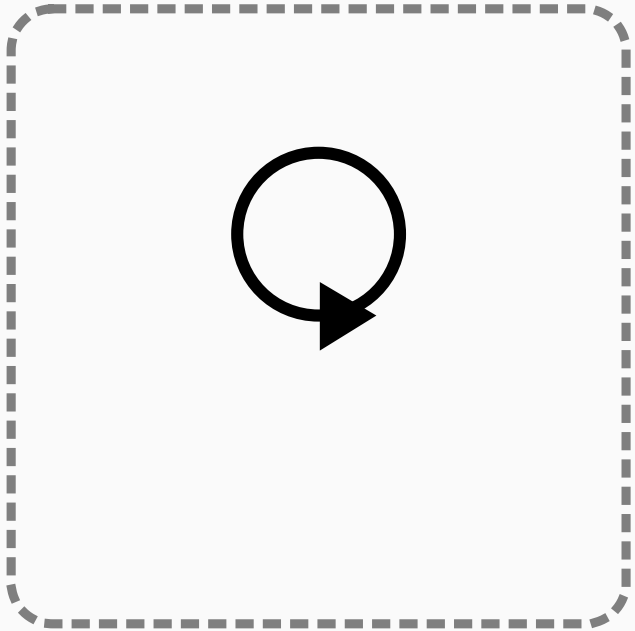
DSL

```
OP ::= FW x | RT x | UP | DOWN | SET state
```

Tasks

```
task :  unit -> image
```

```
for i in range(∞)
> FW ε
> RT ε
```



Turtle graphics — Created & investigated by Mathias Sablé–Meyer

DSL

```
OP ::= FW x | RT x | UP | DOWN | SET state
```

Tasks

```
task :  unit -> image
```

```
for i in range(5 × ∞)
> FW i × ε
> RT ε
```

Turtle graphics — Created & investigated by Mathias Sablé–Meyer

DSL

```
OP ::= FW x | RT x | UP | DOWN | SET state
```

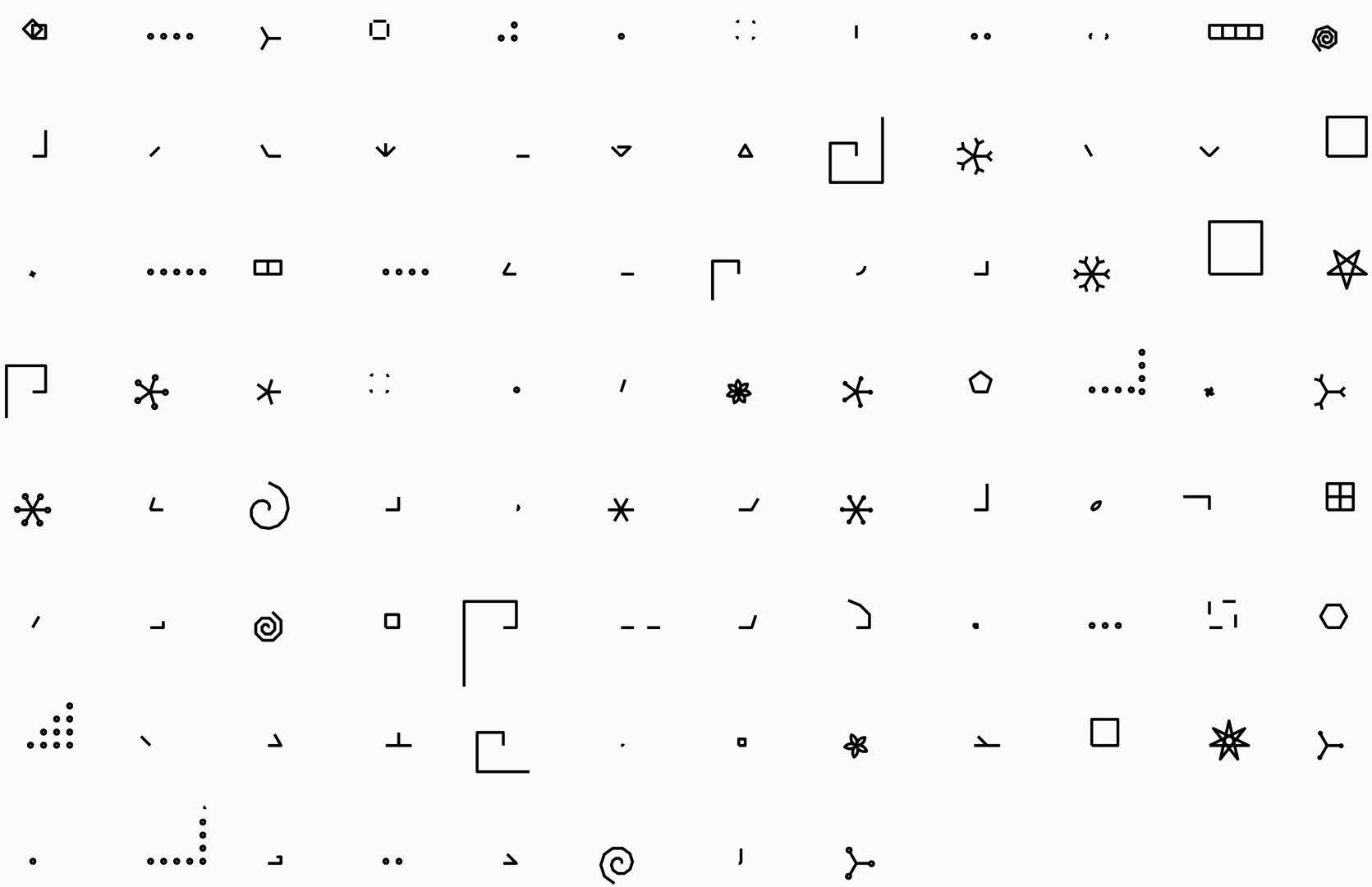
Tasks

```
task :  unit -> image
```

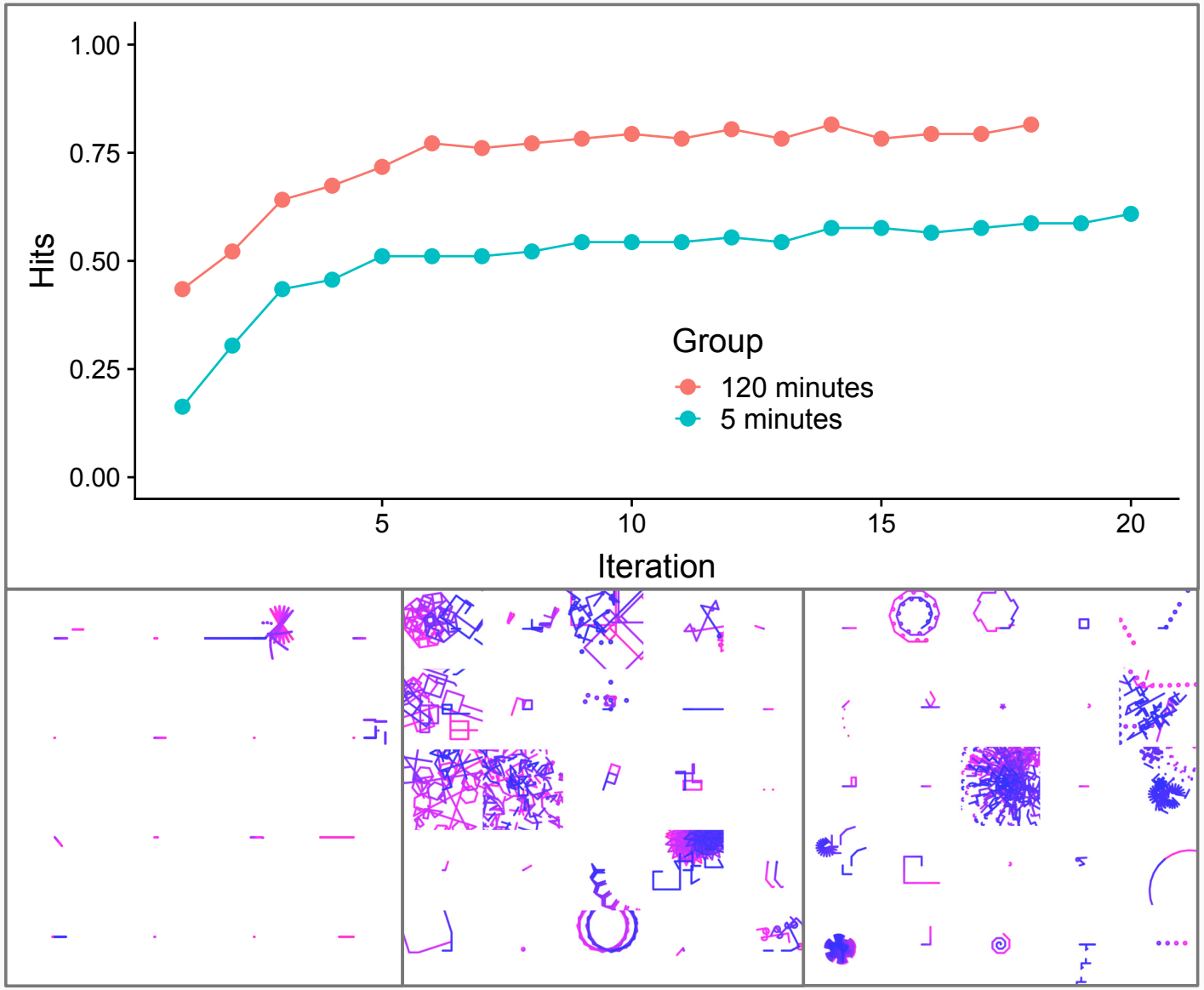
```
for i in range(5 × ∞)
> FW i × ε
> RT ε
```

```
NUM ::= 1 | π | ∞ | ε | + | - | * | /
```

Turtle graphics — Training tasks



Turtle graphics — Some results



Vision

Takeaway:

- Humans flexibly adapt to diverse sets of new problem domains
  — DreamCoder takes a step in this direction

Vision

Takeaway:

- Humans flexibly adapt to diverse sets of new problem domains
  — DreamCoder takes a step in this direction