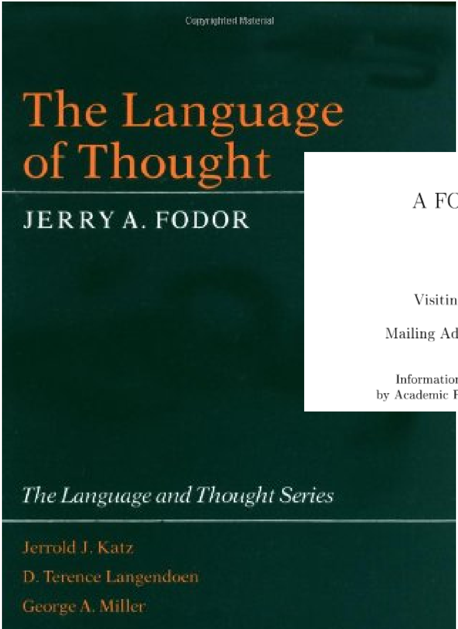
Growing libraries of concepts with wake-sleep program induction

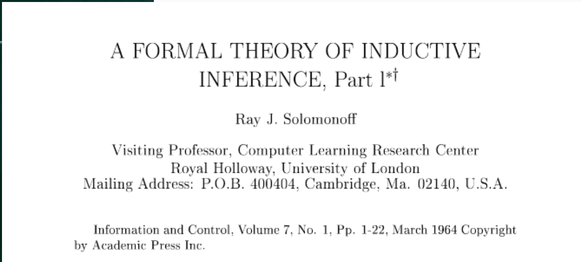
Kevin Ellis & Mathias Sablé Meyer
Joint with: Lucas Morales, Armando Solar-Lezama, Joshua B. Tenenbaum
Heavy inspiration from: Eyal Dechter

July 21, 2018

MIT

The Language of Thought





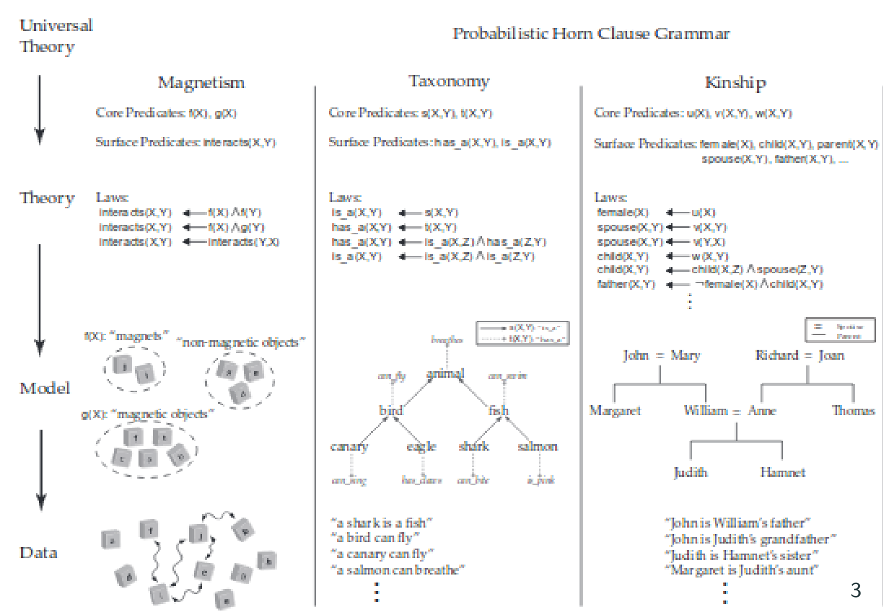
A FORMAL THEORY OF INDUCTIVE INFERENCE, Part 1*†

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.

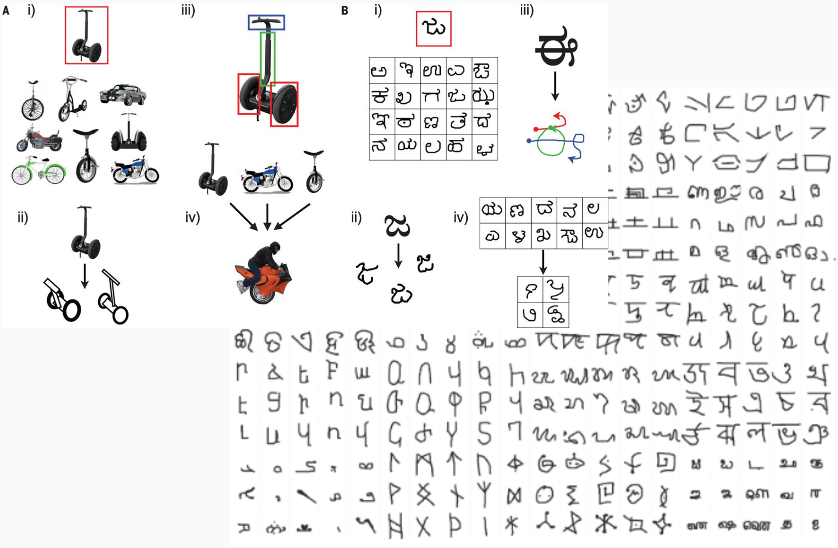
Information and Control, Volume 7, No. 1, Pp. 1-22, March 1964 Copyright by Academic Press Inc.

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

$[7\ 2\ 3] \to [7\ 3]$
$[1\ 2\ 3\ 4] \to [3\ 4]$
$[4\ 3\ 2\ 1] \to [4\ 3]$
$f(\ell) = (f_1\ \ell\ (\lambda\ (\mathtt{x})\ (>\ \mathtt{x}\ 2)))$

$[7\ 3] \to \mathtt{False}$
$[3] \to \mathtt{False}$
$[9\ 0\ 0] \to \mathtt{True}$
$[0] \to \mathtt{True}$
$[0\ 7\ 3] \to \mathtt{True}$
$f(\ell) = (f_3\ \ell\ 0)$

$[2\ 7\ 8\ 1] \to 8$
$[3\ 19\ 14] \to 19$
$f(\ell) = (f_2\ \ell)$

DSL

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

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

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

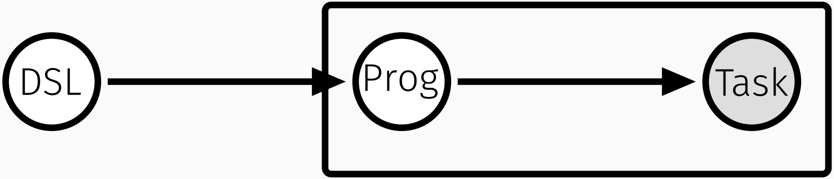
$f_3(\ell,\mathtt{k}) = (\mathtt{foldr}\ \ell\ (\mathtt{is\text{-}nil}\ \ell)\ (\lambda\ (\mathtt{x\ a})\ (\mathtt{if\ a\ a}\ (=\ \mathtt{k\ x}))))$
(f_2: *Whether* ℓ *contains* k)

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

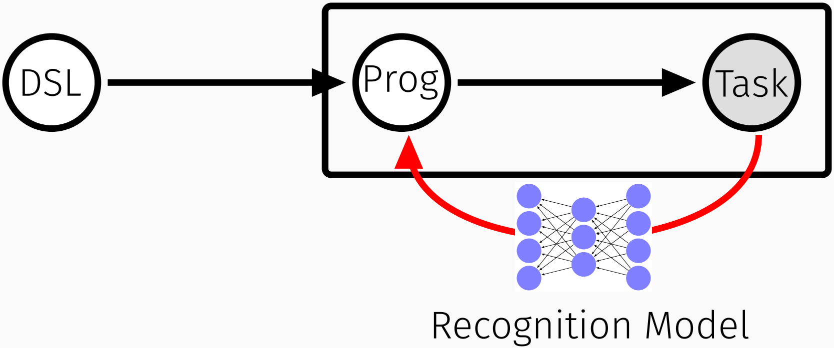
DSL learning as Bayesian inference



[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.

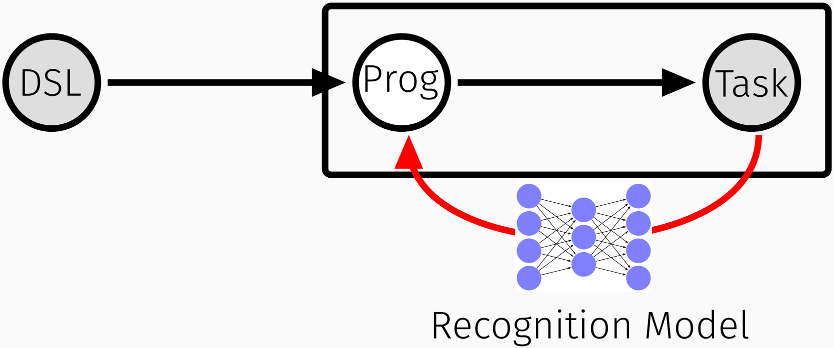
Red: Bottom-up inference.

Gray: Observed.

White: Latent.

Boxed (plate): Repeated.

Wake: Problem solving



Recognition model: Neural net.

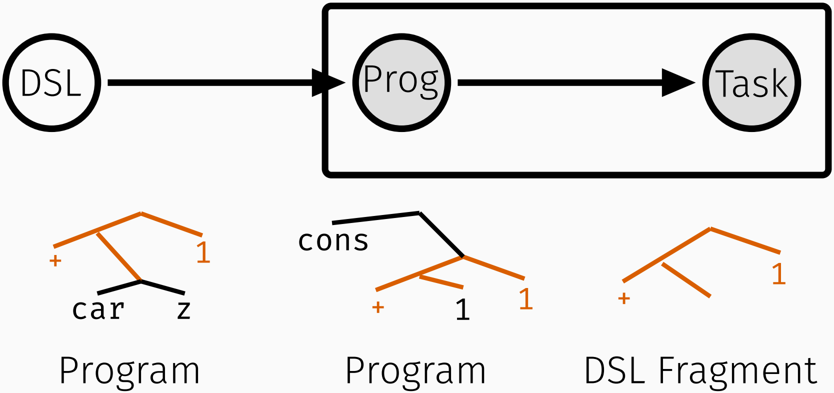
Red: Bottom-up inference.

Gray: Observed.

White: Latent.

Boxed (plate): Repeated.

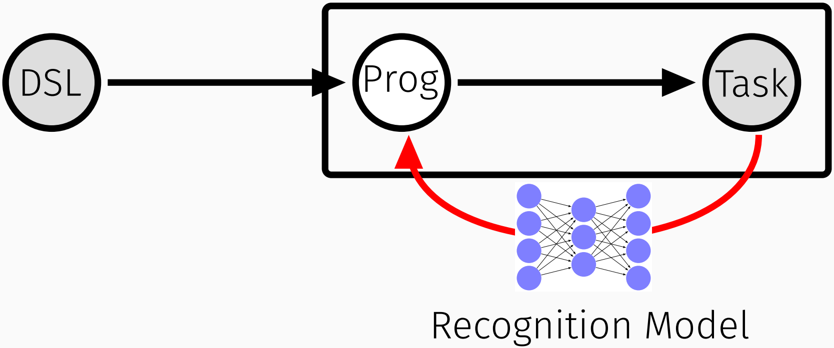
Sleep-G: Memory consolidation



Fragment Grammars: O'Donnell 2015.

Orange: Code fragments.

Sleep-R: Objective

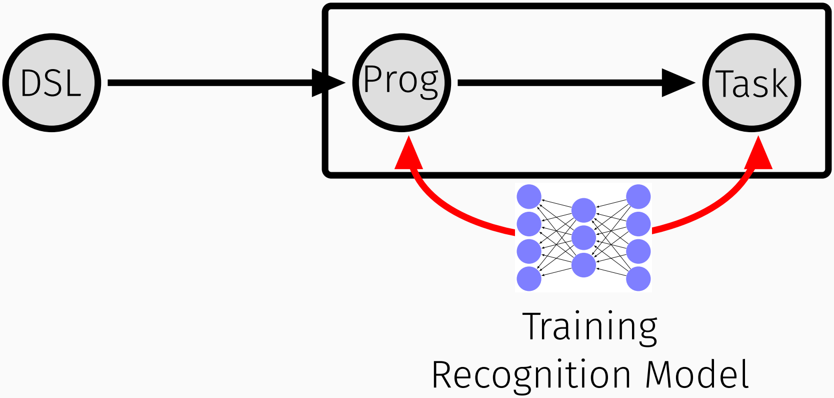


Recognition model predicts distribution over program, conditional on task.

Training: (program, task) pairs

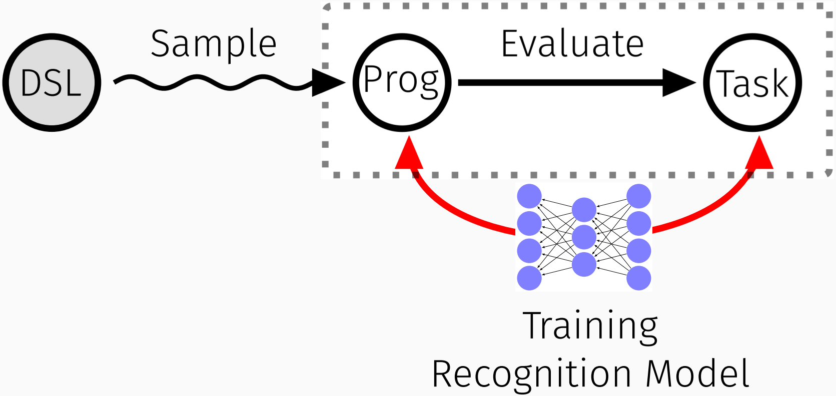
Objective: Predict program w/ (1) high prior under DSL & (2) high likelihood for task

Sleep-R: Experience replay



Train on (program, task) pairs found during waking

Sleep-R: Dreaming



Train on (program, task) pairs sampled from DSL

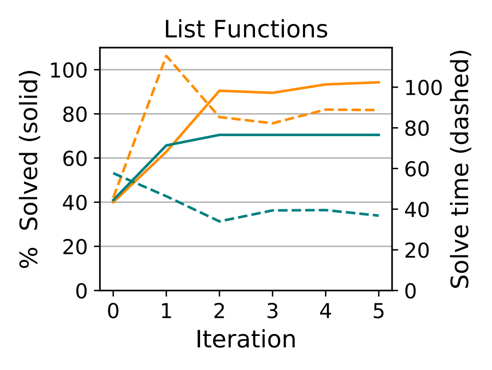
List functions

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

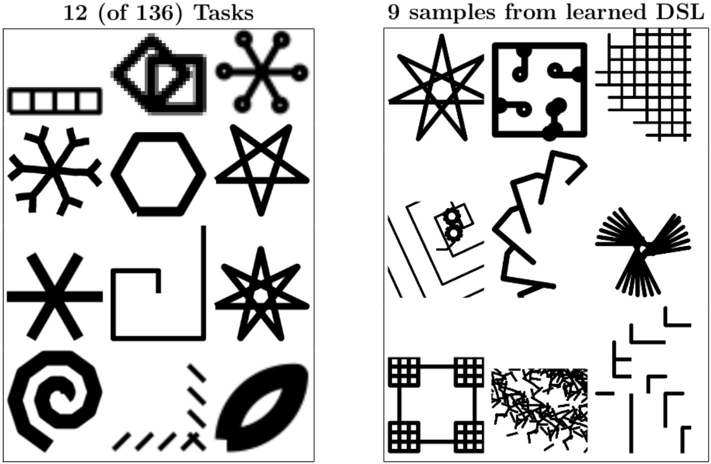
- pen up/pen down
- move pen (forward+rotate)
- get/set pen state
- ‘for’ loops

```
# Hexagon
for n in range(6):
    move(distance=1cm, angle= 2pi / 6)

# Spiral
for n in range(20):
    move(distance = n cm, angle= 2pi / 4)
```

Turtle graphics

Created+Investigated by Mathias Sablé Meyer



Left: Agent controls a ‘pen’ – tasked with drawing pictures. **Right:** During Sleep-R, ‘dream’ by sampling programs from learned DSL and rendering them.

Contributions

Takeaway:

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

Contributions

Takeaway:

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

Future work

- More human-like learning: intelligently composing new tasks,

Contributions

Takeaway:

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

Future work

- More human-like learning: intelligently composing new tasks,
- Theory learning
- Planning