Exploring the hierarchical structure of human plans via program generation

Carlos G. Correa, Sophia Sanborn, Mark K. Ho, Frederick Callaway, Nathaniel D. Daw, Thomas L. Griffiths

Presenter: Hanqi Zhou, 2024.03.19

Goal

- Exploring the hierarchical structure of human plans via program generation.

Goal

- Exploring the hierarchical structure of human plans via program generation.
- Task: make tomato sauce for dinner
- Plan:

  prepare the tomatoes

  place the tomatoes into a large oven or saucepan

The image is generated by ChatGPT

Goal

- Exploring the hierarchical structure of human plans via program generation.
- Task: make tomato sauce for dinner
- Plan:

prepare the tomatoes

prepare one tomato

prepare one tomato

⋮

prepare one tomato

peel

slice

place the tomatoes into a large oven or saucepan

Goal

- Exploring the hierarchical structure of human plans via program generation.
- Task: make tomato sauce for dinner
- Plan:

```
def make_tomato_sauce (N):
        i = 0
        ingredient = None
        while i < N:
            ingredient += rep ( prep_tomato, N )
            i ++
        place_oven (ingredient)
```

```
def prep_tomato ():
        …
```

```
def place_oven (x):
        …
```

```
def rep (x, y):
        …
```

Background - Bayesian Program Induction

- Infer programs (*actions*) π consistent with some observed data (*task*) d

$$p(\pi \mid d) \propto p(d \mid \pi)p(\pi)$$

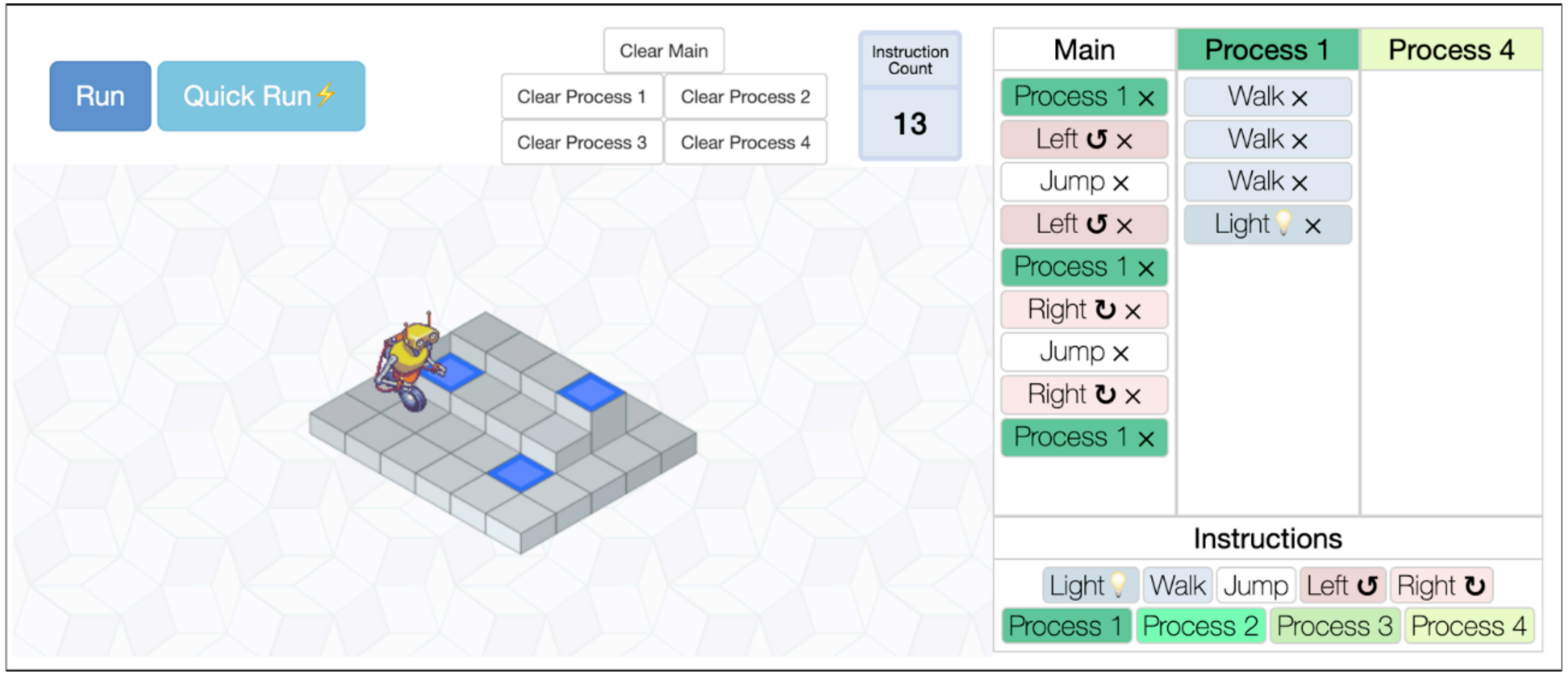
- Why **Bayesian induction** and why **program**?
  - Compositionally, causality and learning to learn (Lake et al., 2015)

Lake, B. M., Salakhutdinov, R., & Tenenbaum, J. B. (2015). Human-level concept learning through probabilistic program induction. Science, 350(6266), 1332–1338. https://doi.org/10.1126/science.aab3050

Assumptions

- Assumption:
  - Human behavior is inherently **hierarchical**, resulting from the decomposition of a task into subtasks or an abstract action into concrete actions.
- Method
  - Utilized a process-tracing experiment with the **Lightbot** game, applying **Bayesian program induction** to model and analyze hierarchical plan formation.

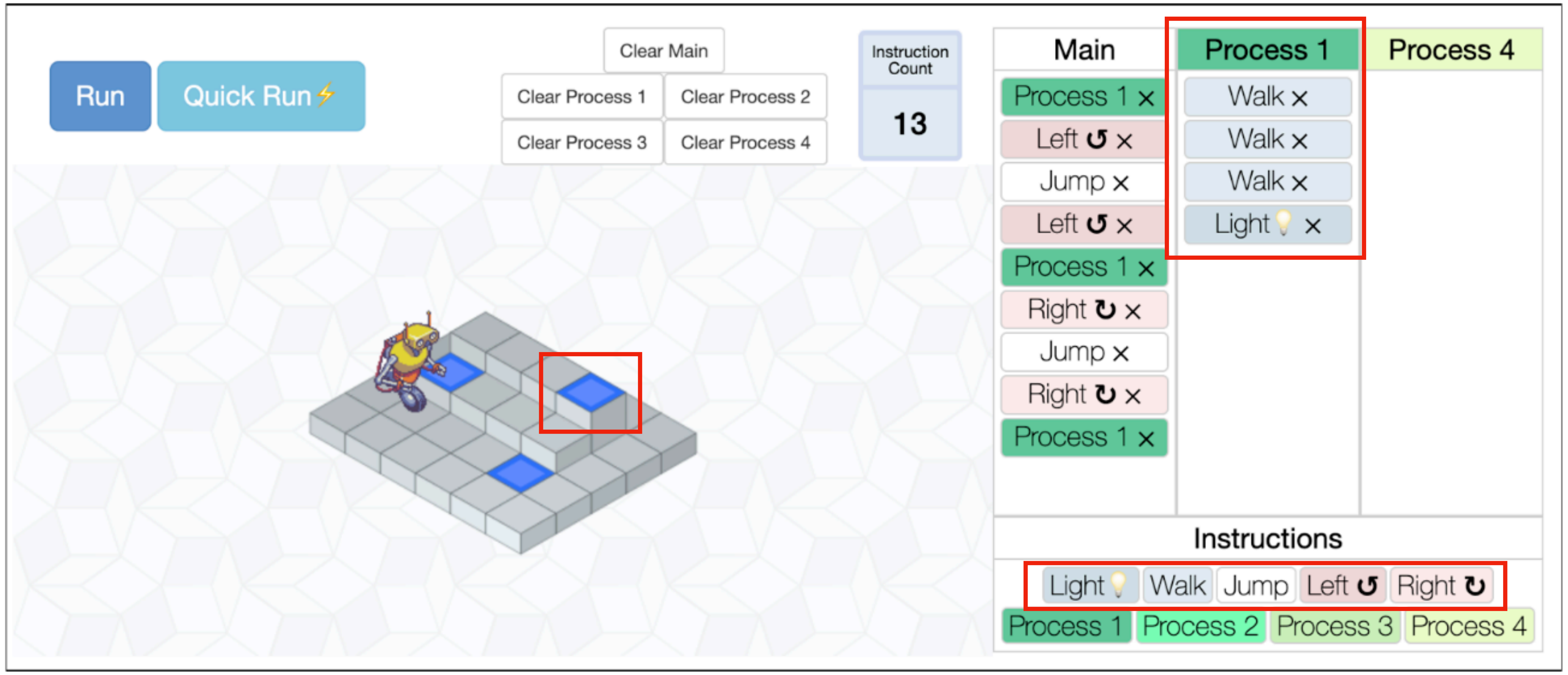
Problem setting



Lightbot - process tracing

- Goal: activate all blue squares
- Action set
    - Primitives
    - Subroutines/processes
- Execution trace
    - A deterministic sequence of actions

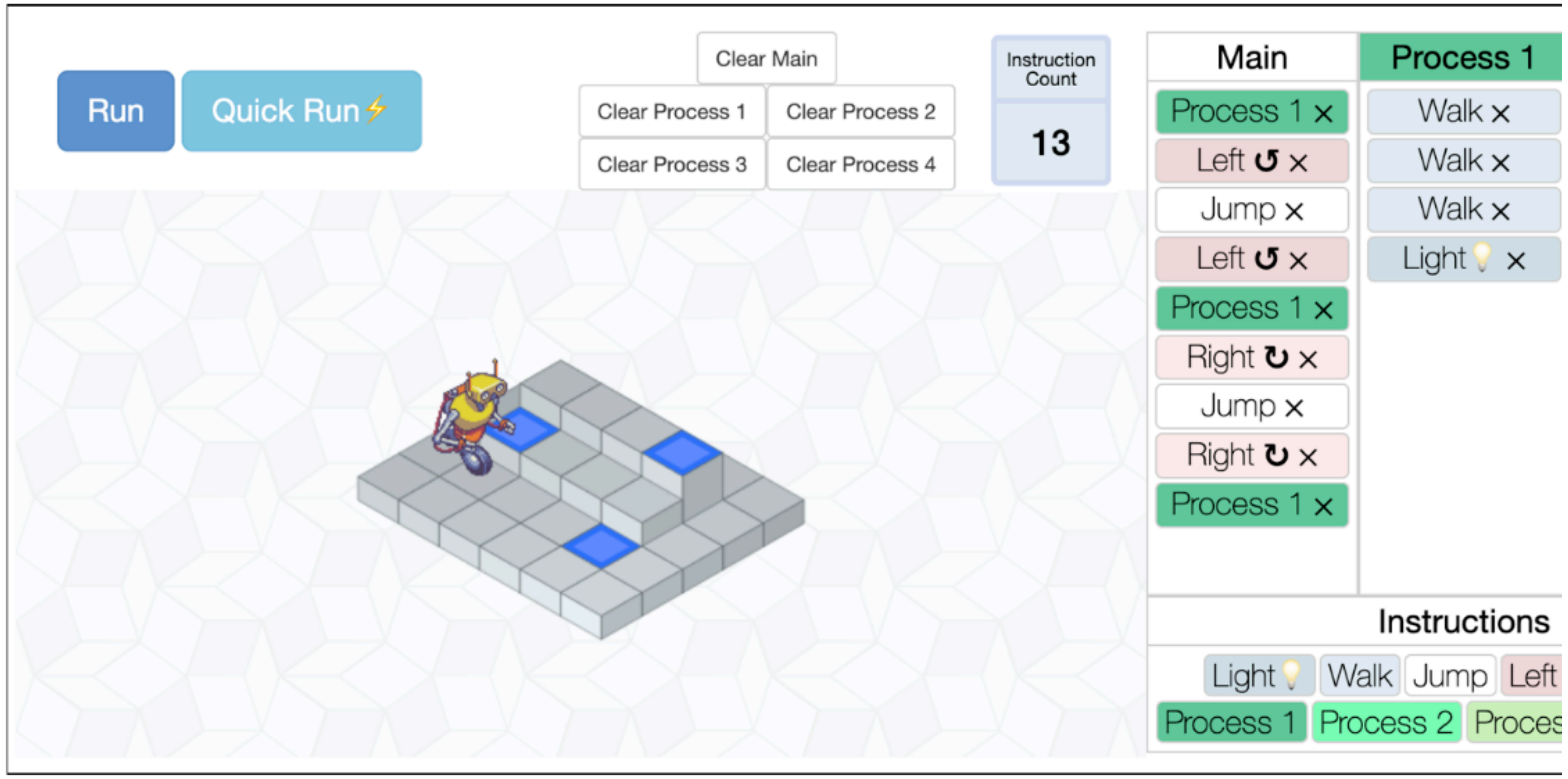
Problem setting



Lightbot - process tracing

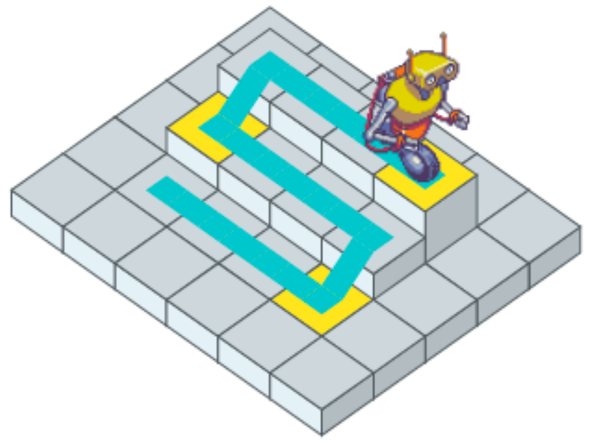
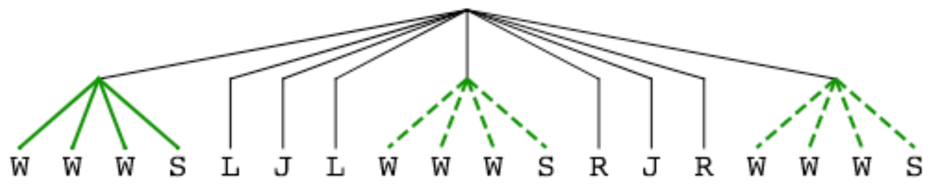
- Goal: activate all blue squares
- Action set
  - Primitives
  - Subroutines/processes
- Execution trace
  - A deterministic sequence of actions

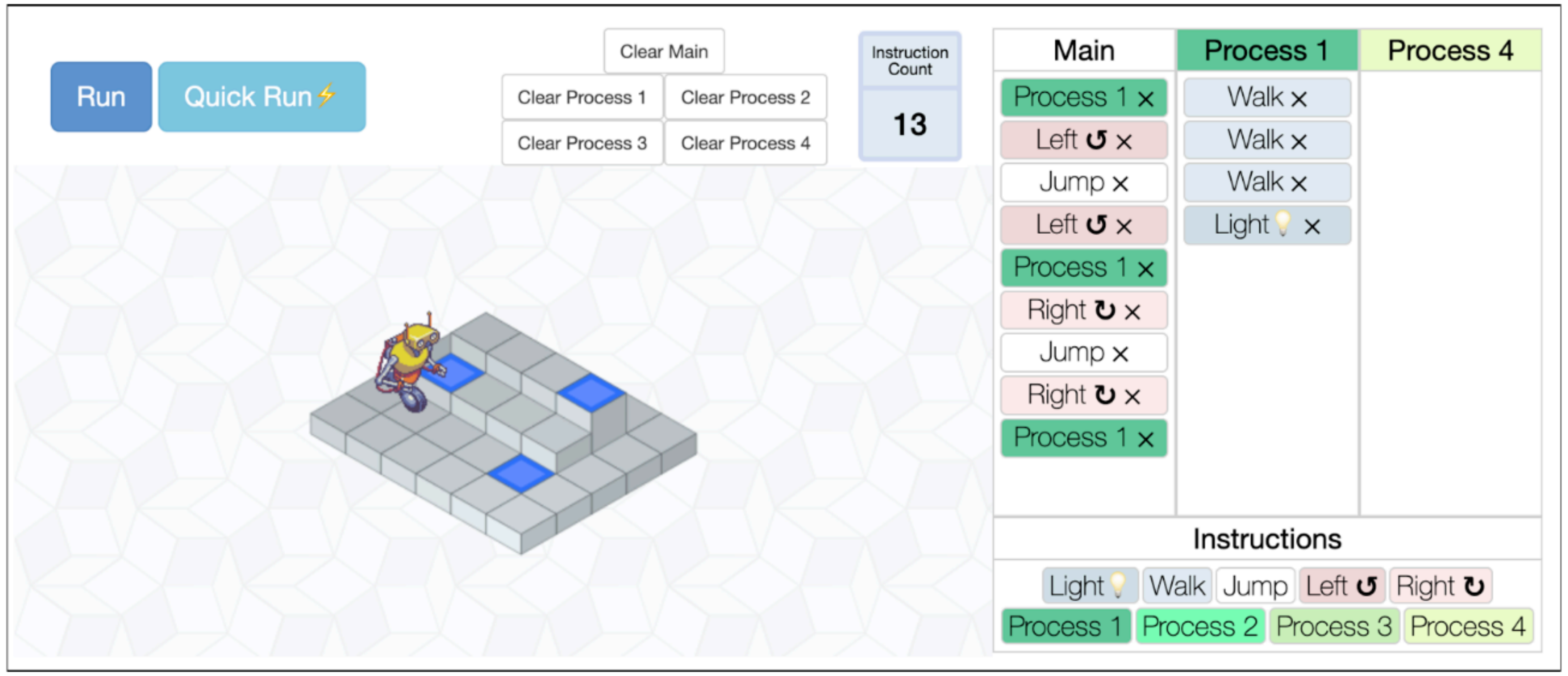
Model of the environment - MDP



- An undiscounted, deterministic Markov Decision Process (MDP)
  - State set $\mathcal{S}$: environment state (active or not), agent state (location & orientation)
    - Goal states $\mathcal{G} \subset \mathcal{S}$
  - Action set $\mathcal{A}$: five primitive actions
  - Transition function $T : \mathcal{S} \times \mathcal{A} \rightarrow \mathcal{S}$
  - Reward function $R : \mathcal{S} \times \mathcal{S} \rightarrow \mathbb{R}$: $R(s, g) = 1$ only if $g \in \mathcal{G}$

Model of the agent - Program

Trace	Program	Step Count	Program Length
		18	13



- Program $\pi = \{\rho^0, \rho^1, \dots\}$
  - Subroutine $\rho^i = \{\rho^i_0, \rho^i_1, \dots\}$
    - Instructions $\rho^i_j \in \mathscr{A} \cup \{\rho^1, \rho^2, \dots\}$

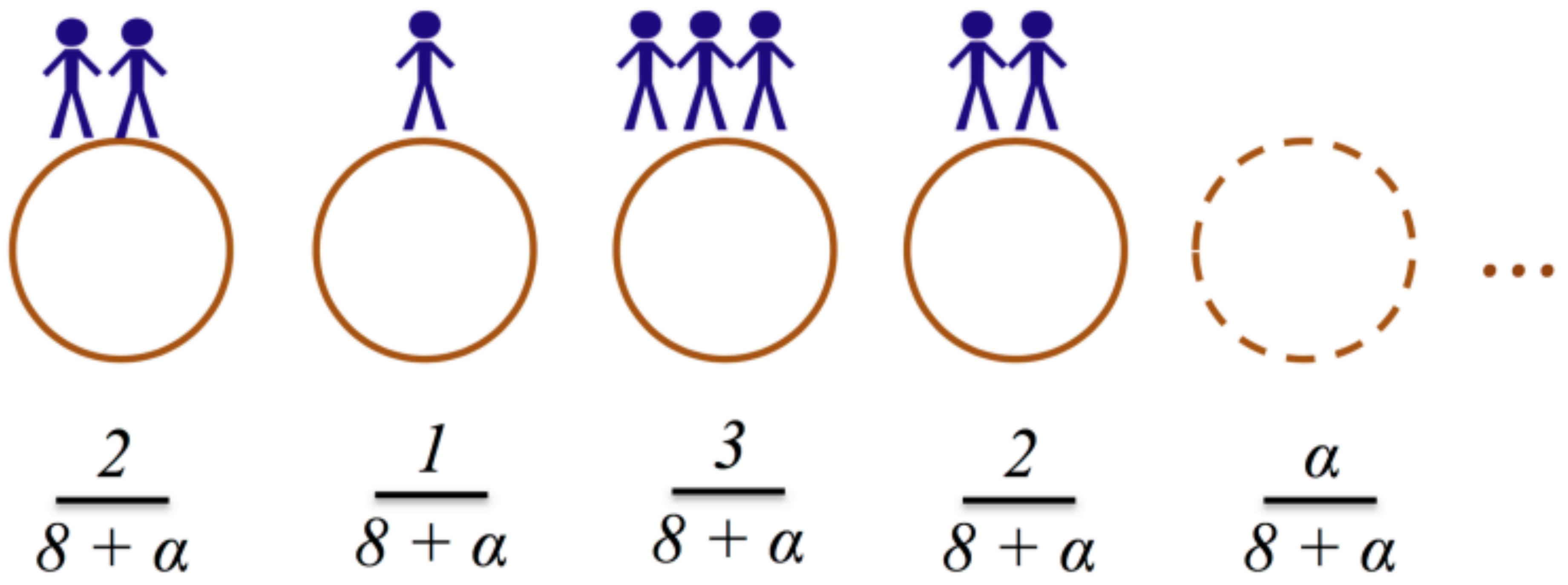
Model - Bayesian Program Inference

$$p(\pi \mid \text{success}) \propto p(\text{success} \mid \pi)\, p(\pi)$$

- Baseline models
  - Prior:
    - Random $\log p(\pi) = \frac{1}{|\Pi|}$
    - Step count $\log p(\pi) \propto -|\tau(\pi)|$
    - Program/Description length $\log p(\pi) \propto -\mathrm{DL}(\pi)$
    - Reuse-priority
  - Likelihood: binary

Model - Program Prior towards Reuse

- Key idea: past use of subroutines should inform the probability assigned to future use of subroutines.
- Method: Chinese Restaurant Process (CRP)



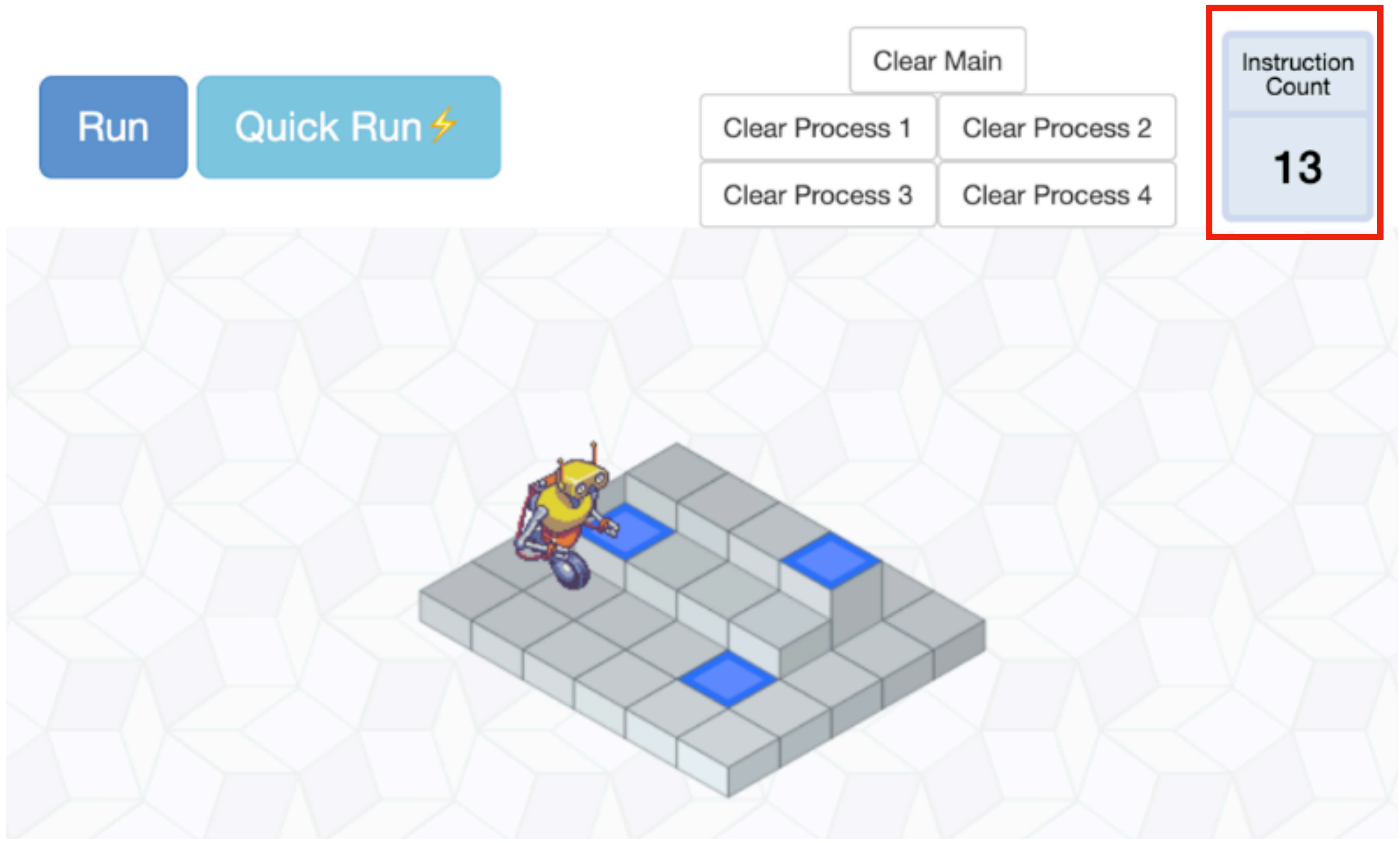
$$p\left(\rho^{i}\right)=p_{\text{end}}\, p\left(\rho_{0}^{i}=a\right) \prod_{j=1}\left(1-p_{\text{end}}\right) p\left(\rho_{j}^{i}=a\right)$$

$$p\left(\rho^{n+1} \mid \rho^{1:n}\right)=\begin{cases}\frac{\alpha}{n+\alpha} & \text{if } \rho^{n+1}=m+1 \\ \frac{n_k}{n+\alpha} & \text{if } 1 \leq \rho^{n+1}=k \leq m\end{cases}$$

Image source: https://www.cs.cmu.edu/~epxing/Class/10708-16/note/10708_scribe_lecture19.pdf

Experiments

- Participants
  - have to solve 10 tasks
  - are incentivized to write short programs (to encourage subroutine use)



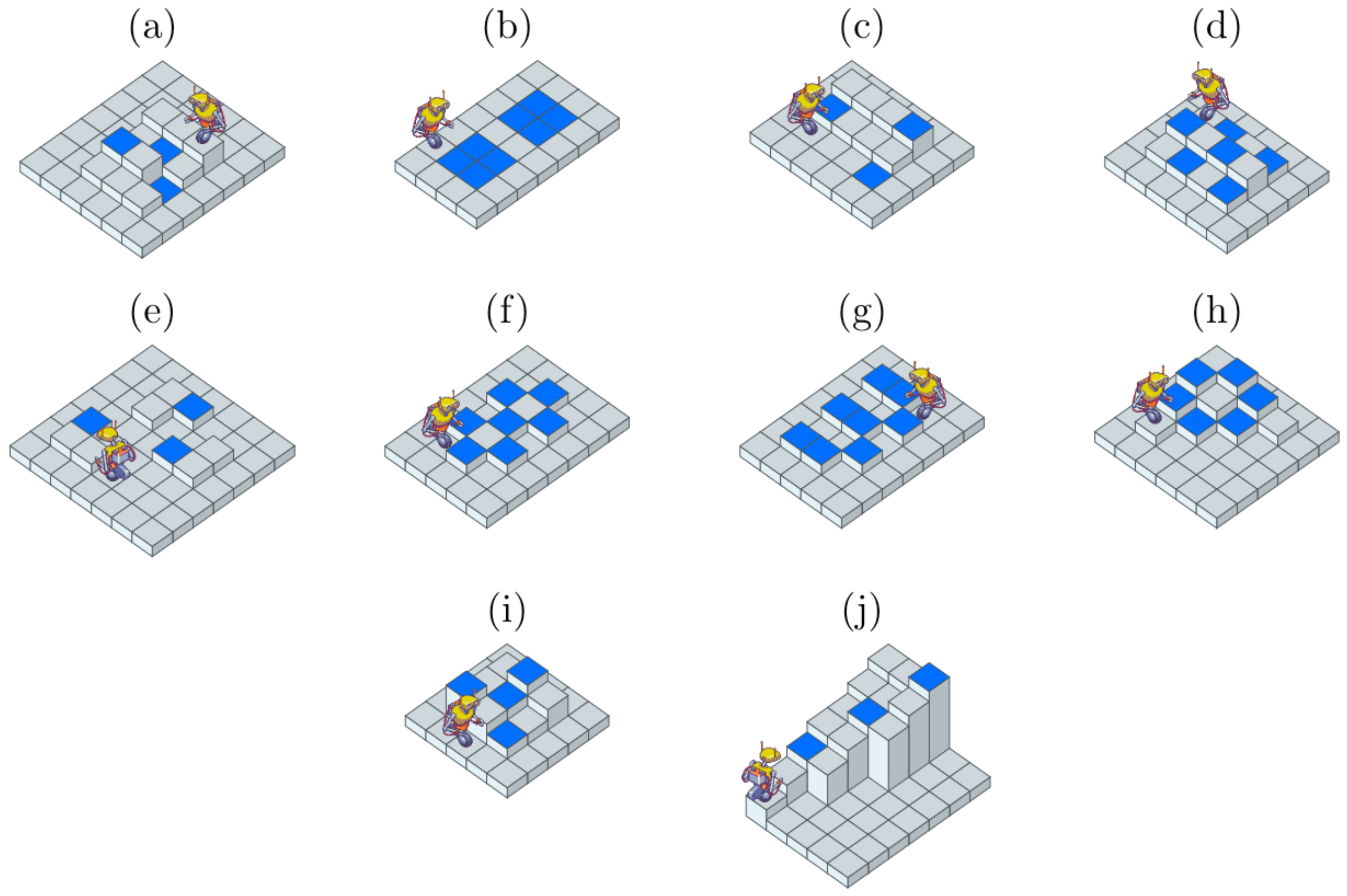
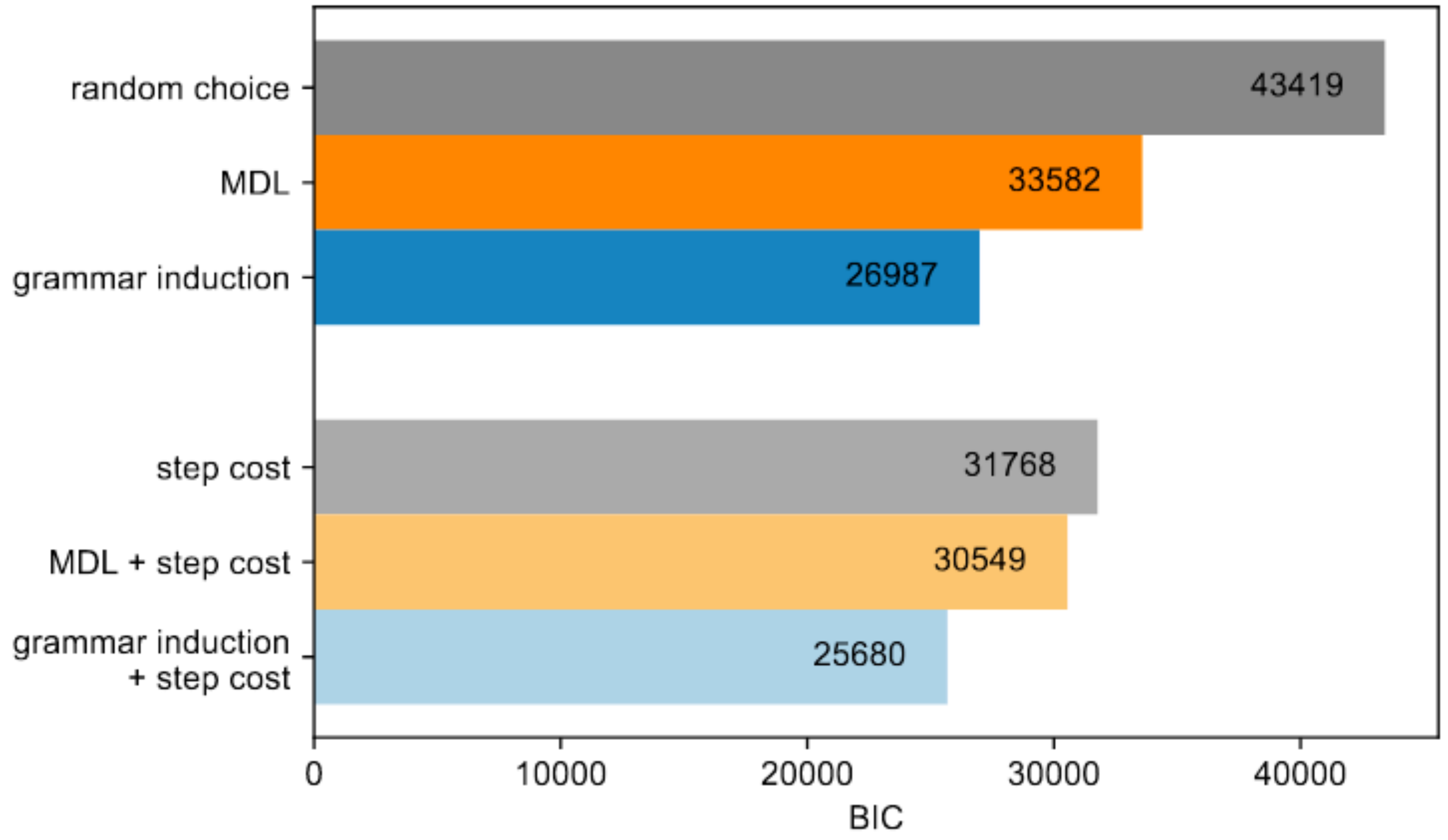


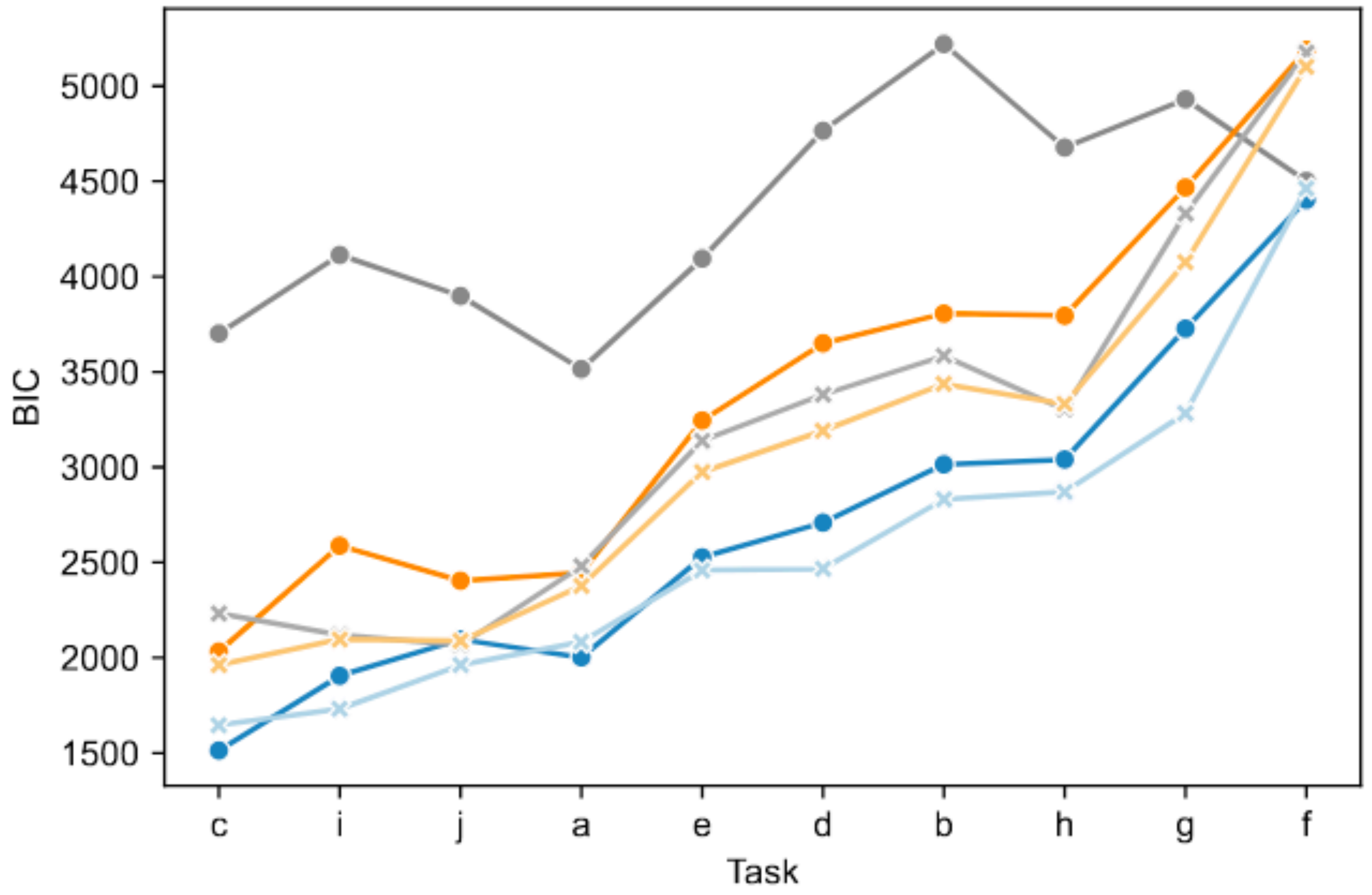
Figure 3: The tasks participants completed in the experiment.

Results

The step cost model is a better fit to behavior than the MDL



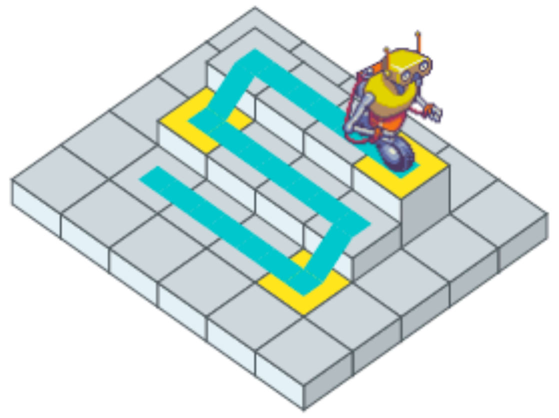
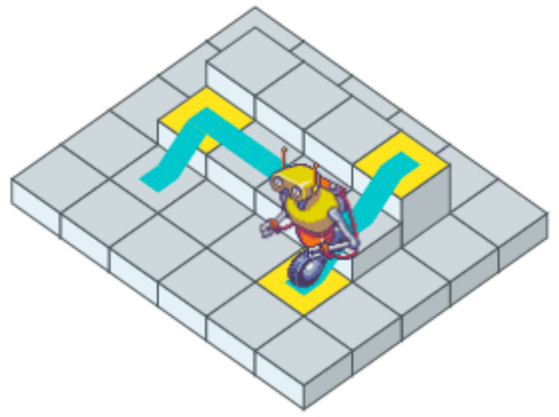
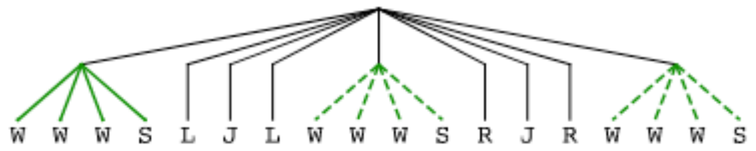
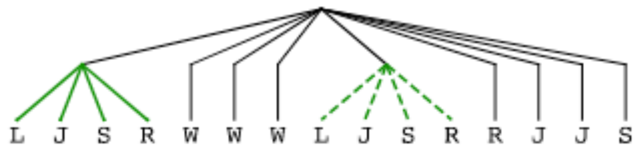
(a)



(b)

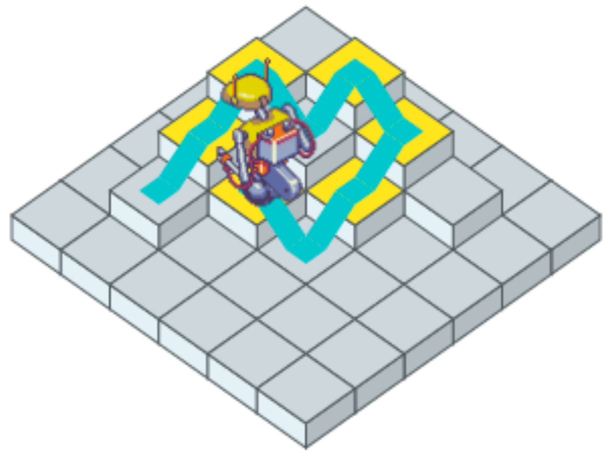
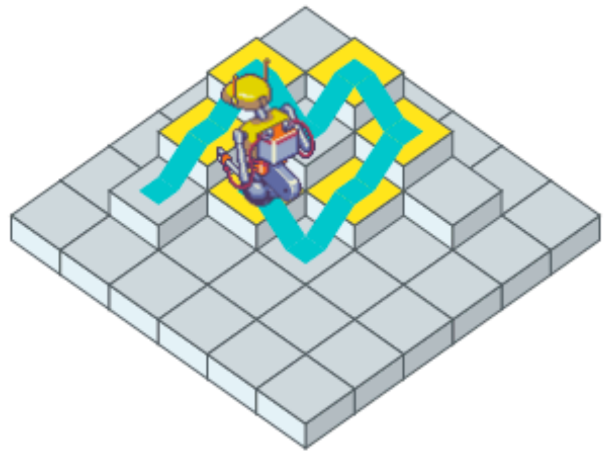
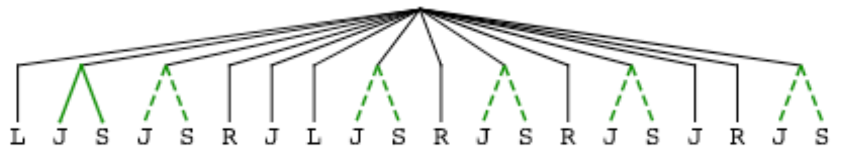
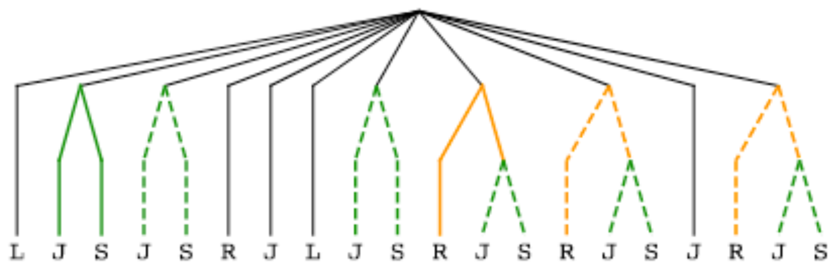
Results

Participant programs are inconsistent with alternative accounts

trace		
program		
participant count	59	1
step count	18	15
program length	13	13
grammar induction prior (log)	-31.97	-33.17

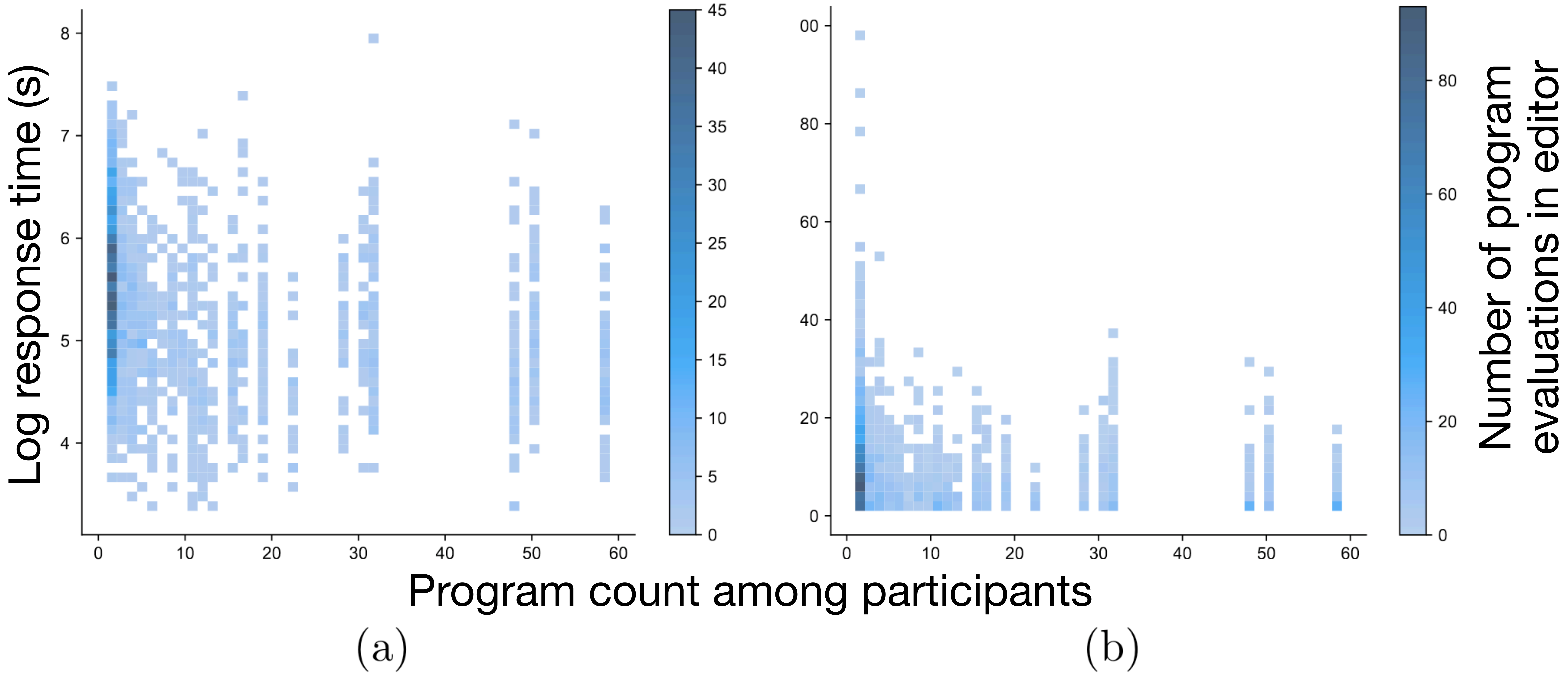
Results

Participant programs are inconsistent with alternative accounts

trace		
program		
participant count	19	3
step count	20	20
program length	16	15
grammar induction prior (log)	-35.06	-37.49

Results

Common programs are easier to write



Findings

- In this work, we find evidence that people favor both shorter programs and those with fewer actions.
- However, we additionally identify examples in which people prefer reusing subroutines above and beyond the predictions of either account.
- We formalize this preference as a prior belief that subroutine use should be biased towards subroutines that were used often in the past.