Large Language Models as Commonsense Knowledge for Large-Scale Task Planning

NeurIPS 2023

Zirui Zhao

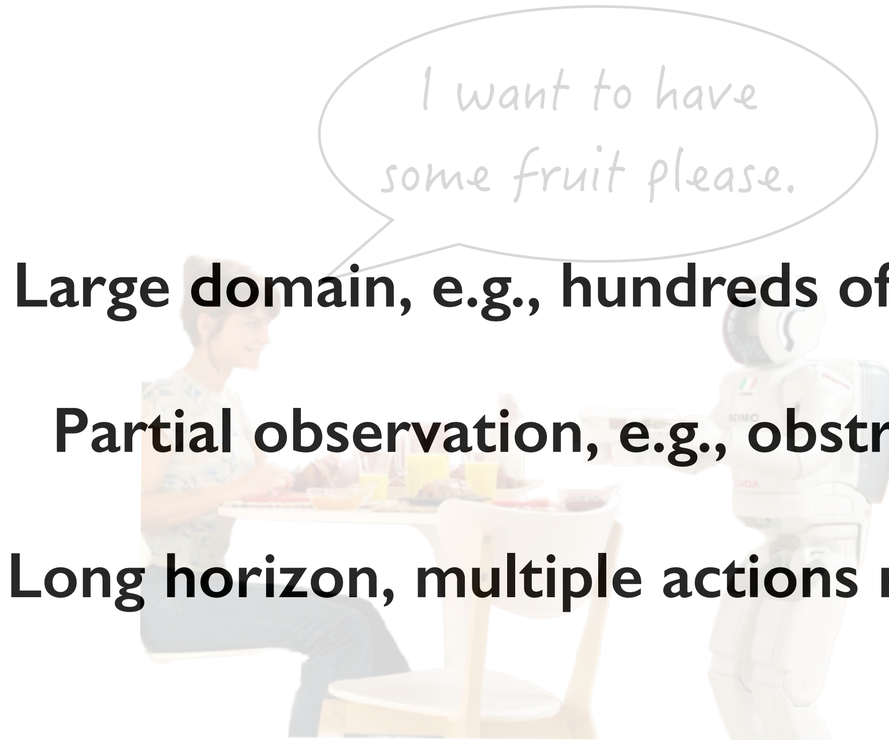
Wee Sun Lee

David Hsu

NEURAL INFORMATION PROCESSING SYSTEMS

NUS National University of Singapore | Computing

Planning in large-scale environments



Large domain, e.g., hundreds of objects

Partial observation, e.g., obstruction

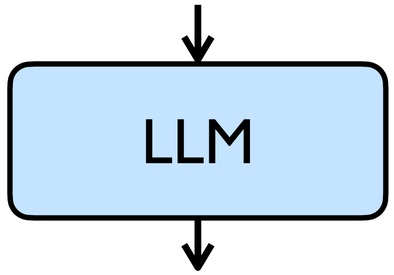
Long horizon, multiple actions required

Planning in large-scale environments

How to solve the challenging large-scale planning problems?

Planning with large language models

I want to have some fruit please

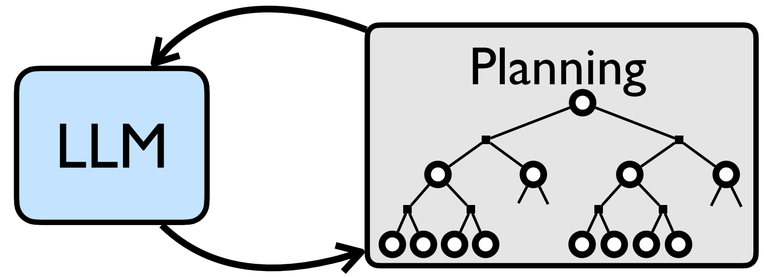


1. Go to kitchen; 2. Open fridge;
3 …

LLM as a policy

?

Action: Go to kitchen



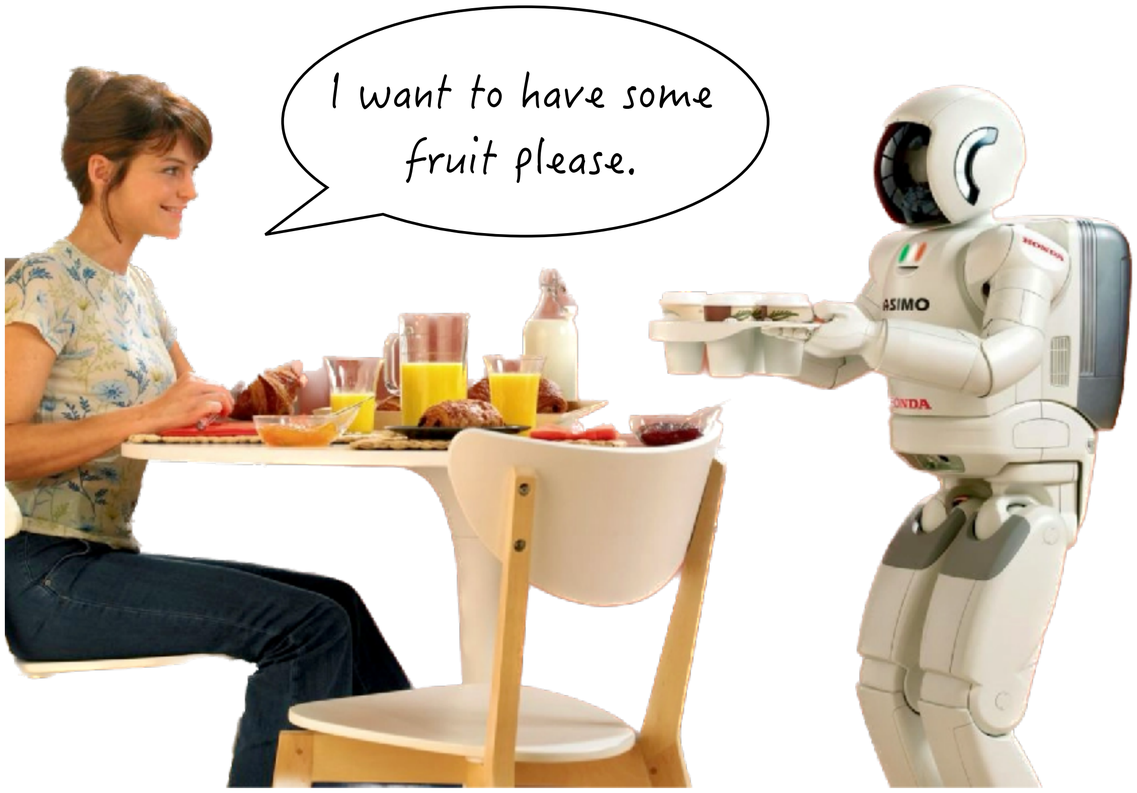
Observation: fridge in the kitchen;
Next state: you at kitchen, …;
Reward: …

LLM as a world model

LLM as a *world model* + LLM as a *policy*

- LLM as *world model* and *policy* in *planning algorithm*

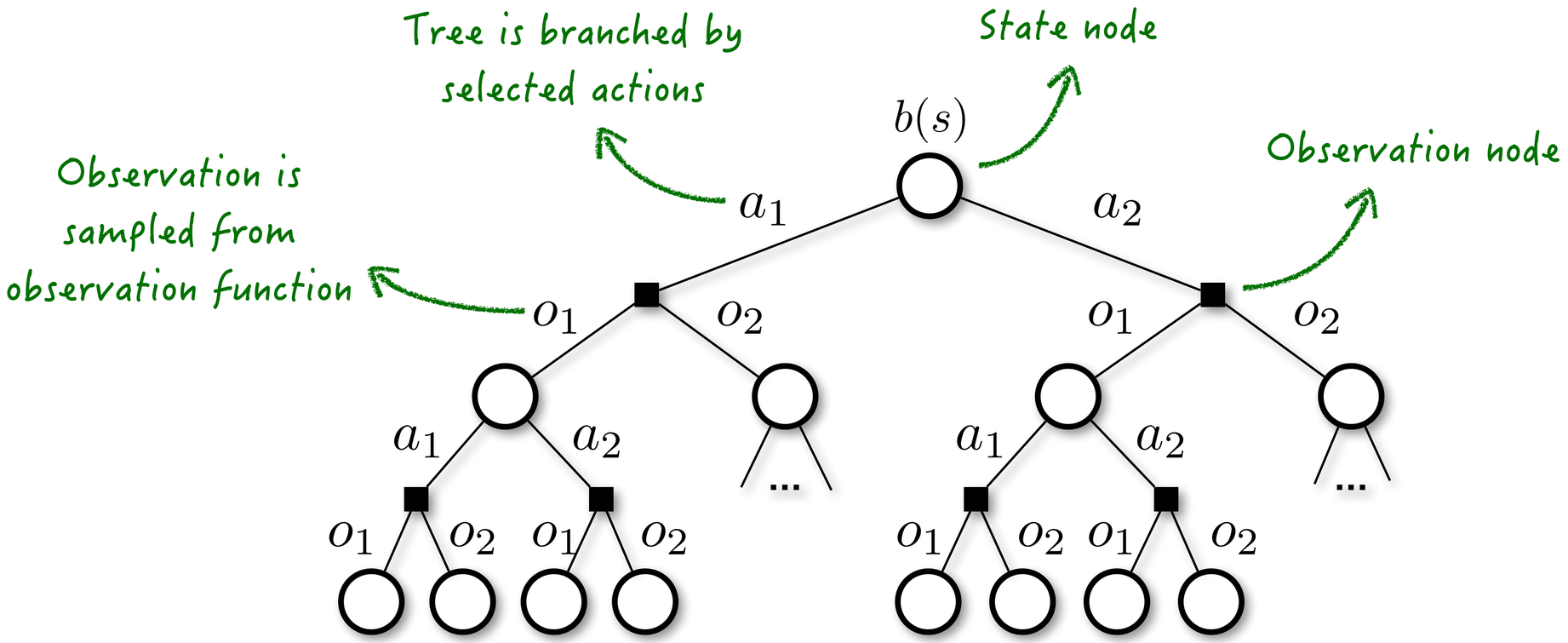
Formulation: Partially Observable Markov Decision Process (POMDP)



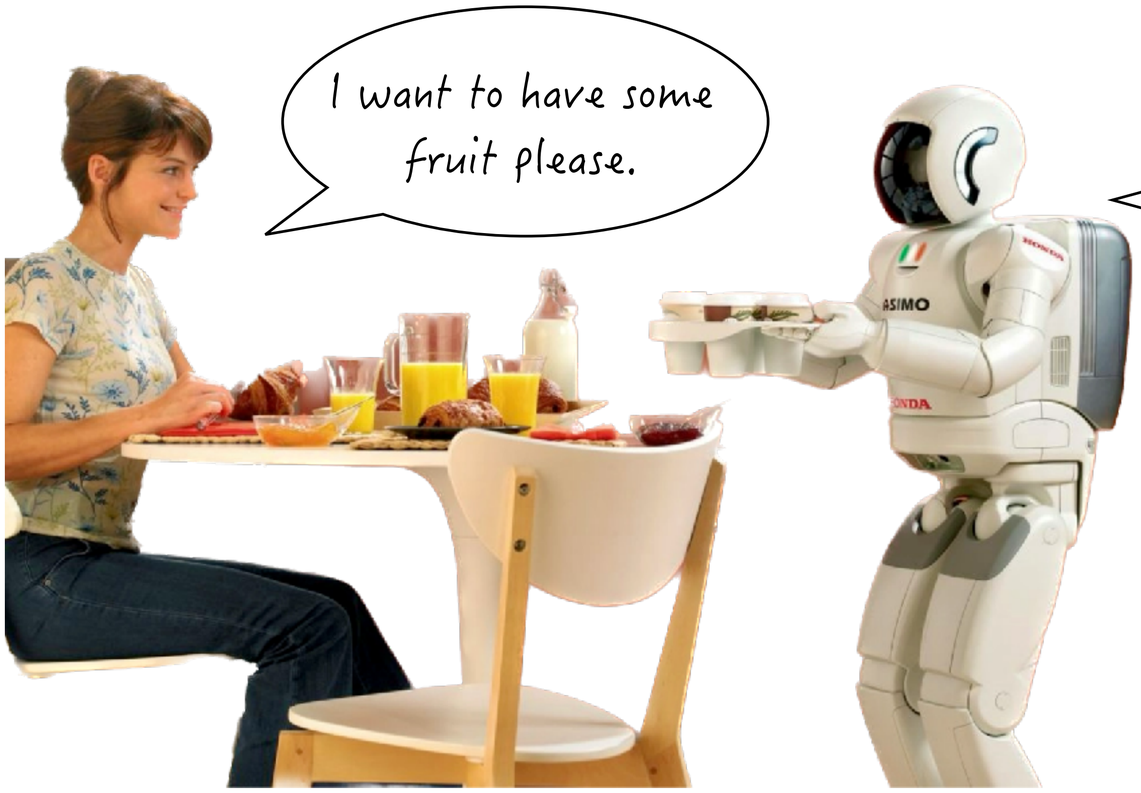
- POMDP Modeling
  - State s, Action a, Observations o
  - State-transition function $p(s'|s,a)$
    - Action uncertainty
    - Environment Changes
  - Observation function $p(o|s',a)$
    - Sensor noise
    - Observation uncertainty
  - Reward function $R(s,a)$
    - Planning objective

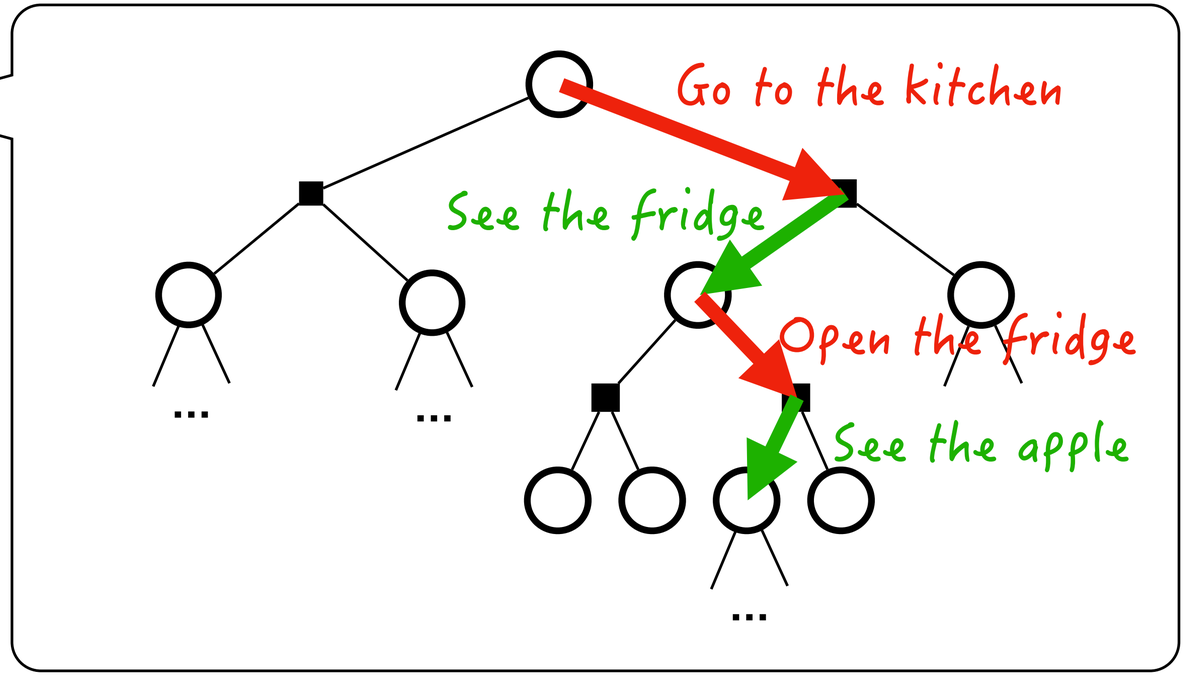
$$\arg\max_{a \in A} \mathbb{E}_{b(s_t)} \left[\sum_{i=0}^{\infty} \gamma^i R(s_{t+i}, a_{t+i}) | a_t = a \right]$$

POMDP Planning as Belief Tree Search



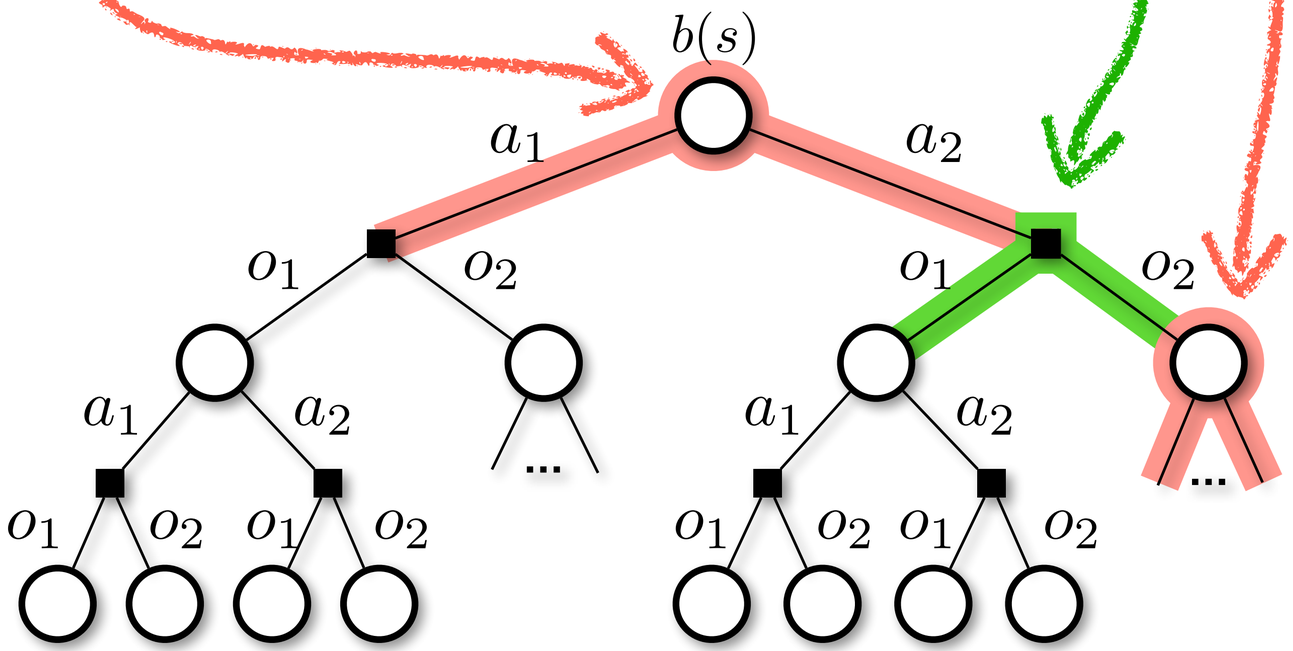
POMDP Planning as Belief Tree Search





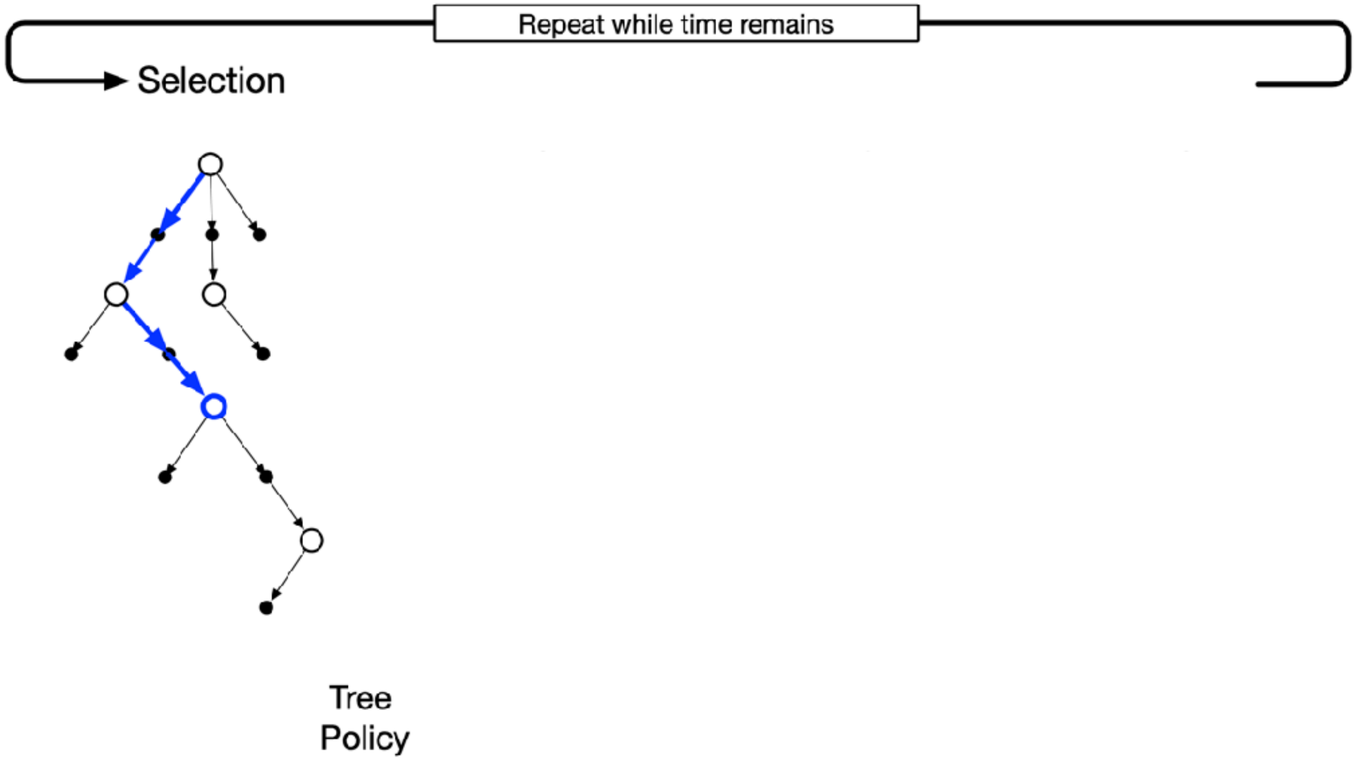
POMDP Planning as Belief Tree Search

$$V(b) = \max_{a \in A} \left\{ \int_{s \in S} R(s,a) b(s) ds + \gamma \int_{o \in O} p(o|b,a) V(b') do \right\}$$



Classical Planning: Monte-Carlo Tree Search

- Sampling from belief tree for approximate planning
  - Selection: select action branches until reaching leaf nodes
  - Expansion: sample observation and next state
  - Simulation: use rollout policy to plan
  - Backup: update the estimated Q function



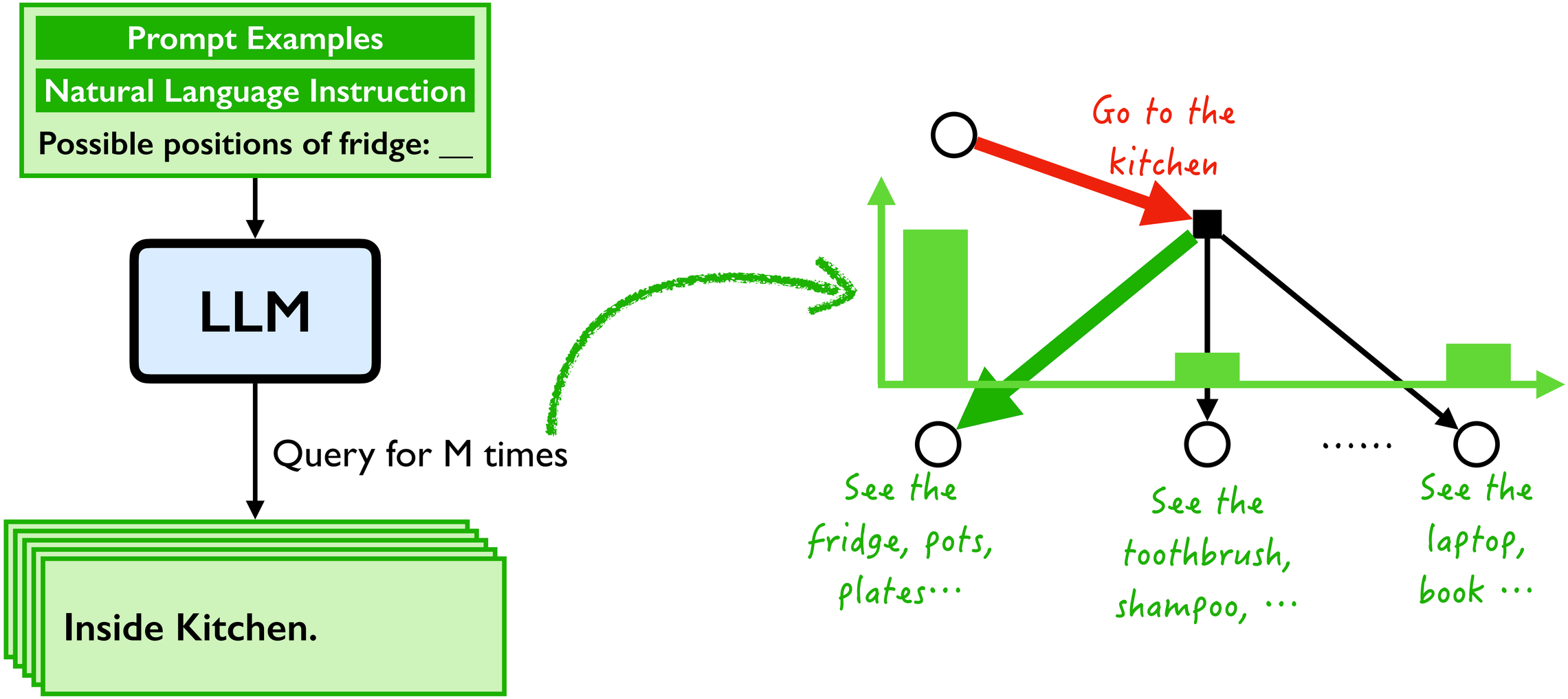
$$\arg\max_{a \in A} \mathbb{E}_{b(s_t)} \left[\sum_{i=0}^{\infty} \gamma^i R(s_{t+i}, a_{t+i}) | a_t = a \right]$$

Figure adapted from Sutton & Barto

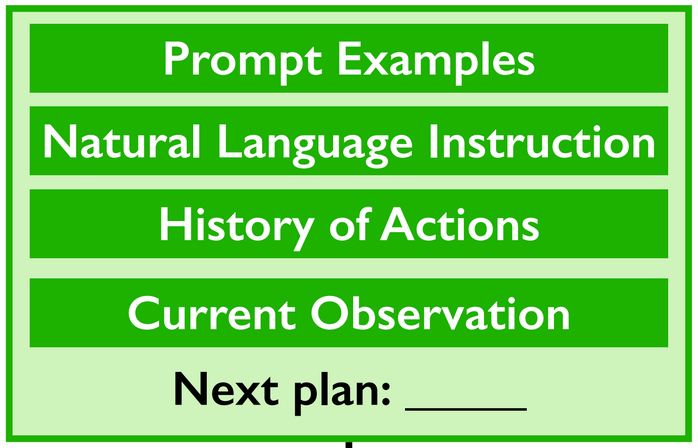
LLM as a *world model* + LLM as a *policy*

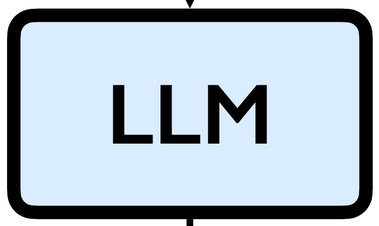
- LLM as *world model* and *policy* in *planning algorithm* (Monte Carlo Tree Search)

LLMs as Commonsense World Model

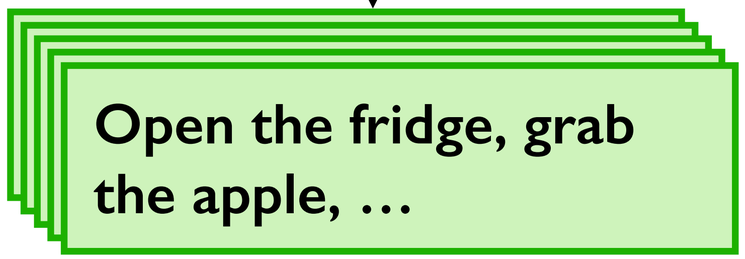


LLMs as Commonsense Heuristic Policy

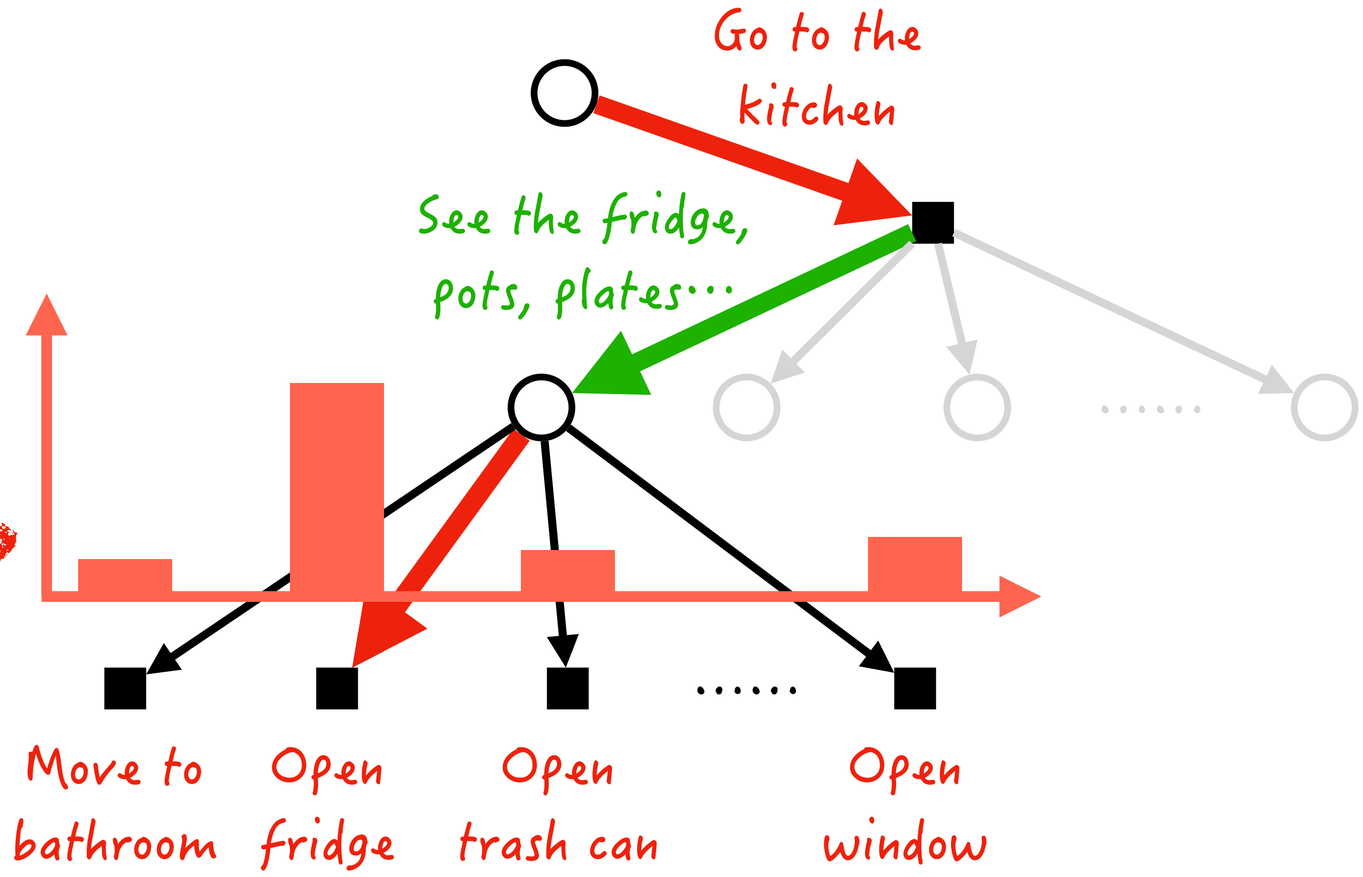




Query for M times



$\hat{\pi}(a|h)$

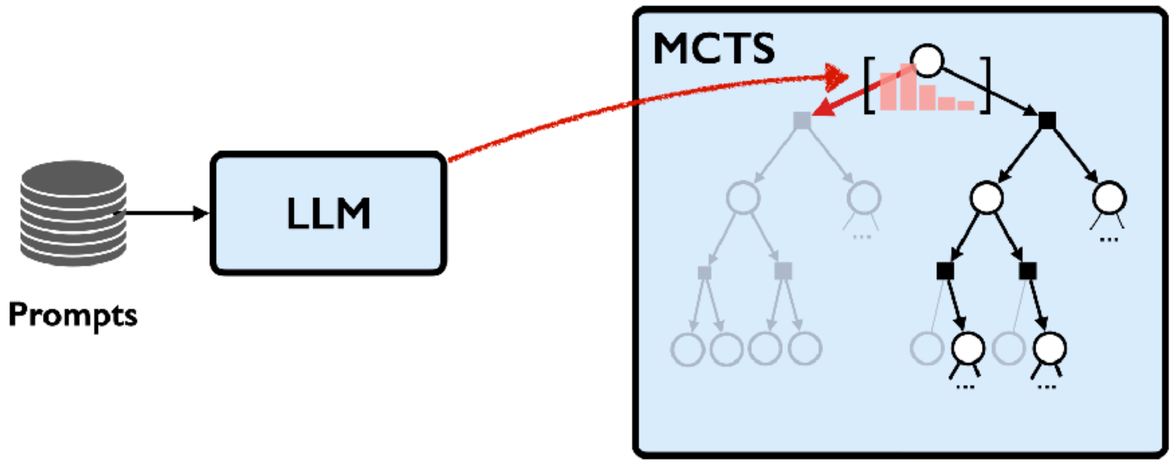


$$a^* = \arg\max_{a \in A} Q(h,a) + c\hat{\pi}(a|h)\frac{\sqrt{N(h)}}{N(h,a)+1}$$

PUCT action selection

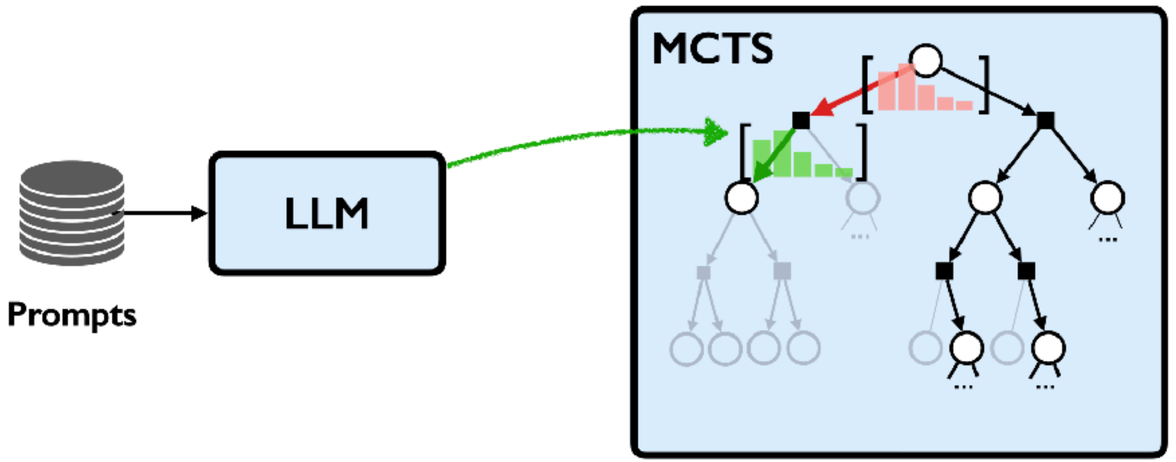
Monte Carlo Planning with LLM

- Sampling from belief tree for approximate planning
  - Selection: select action biasedly according to commonsense



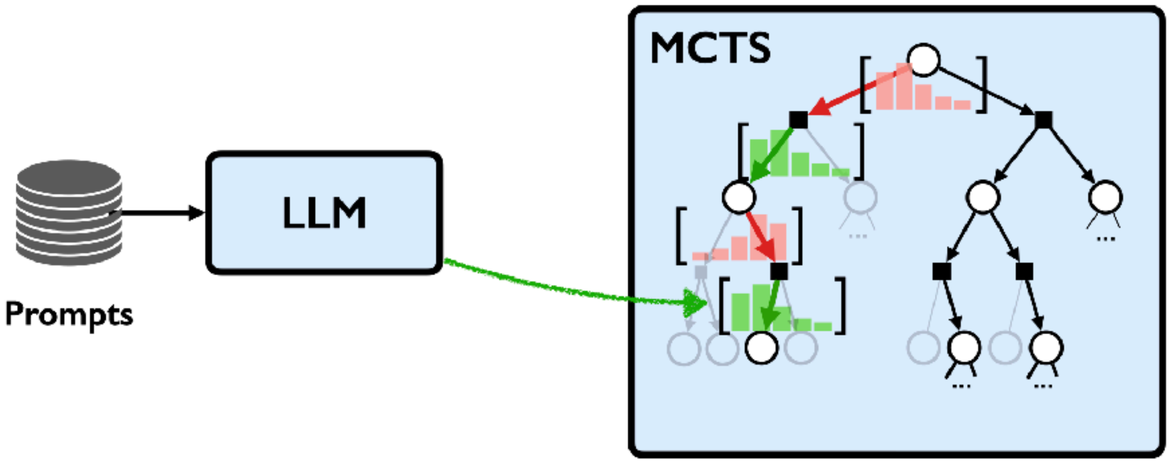
Monte Carlo Planning with LLM

- Sampling from belief tree for approximate planning
  - Selection: select action biasedly according to commonsense
  - Expansion: sample observation according to commonsense



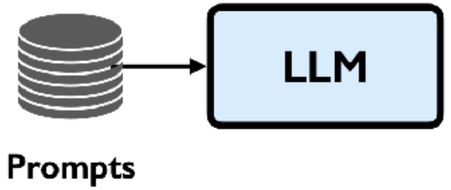
Monte Carlo Planning with LLM

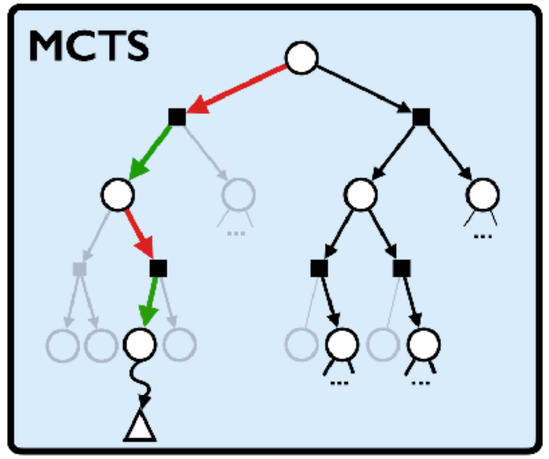
- Sampling from belief tree for approximate planning
  - Selection: select action biasedly according to commonsense
  - Expansion: sample observation according to commonsense



Monte Carlo Planning with LLM

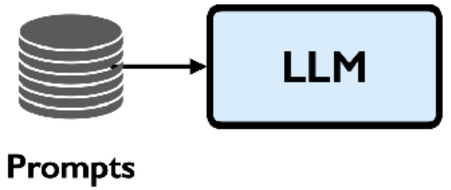
- Sampling from belief tree for approximate planning
  - Selection: select action biasedly according to commonsense
  - Expansion: sample observation according to commonsense
  - Simulation: random rollout to plan

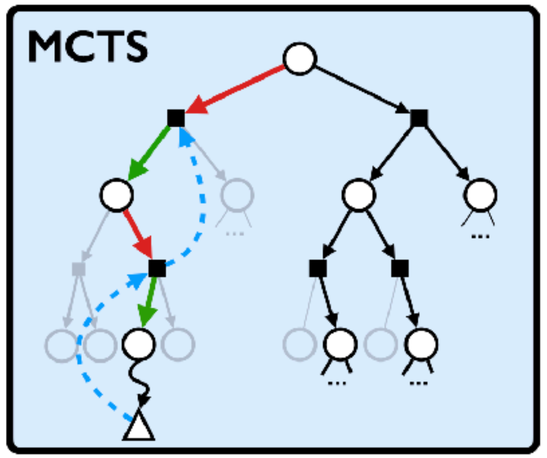




Monte Carlo Planning with LLM

- Sampling from belief tree for approximate planning
  - Selection: select action biasedly according to commonsense
  - Expansion: sample observation according to commonsense
  - Simulation: random rollout to plan
  - Backup: update the estimated Q function



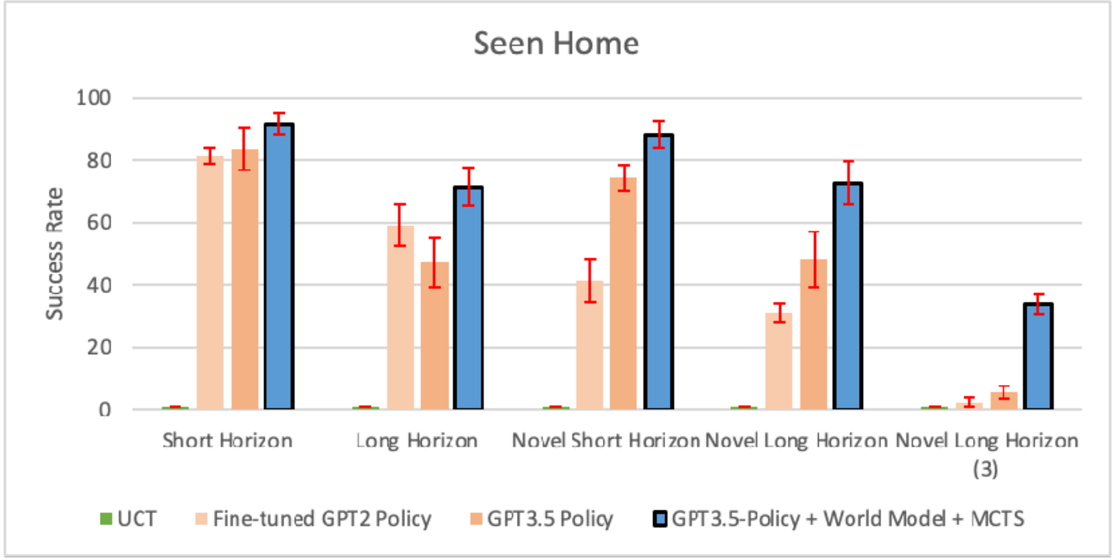


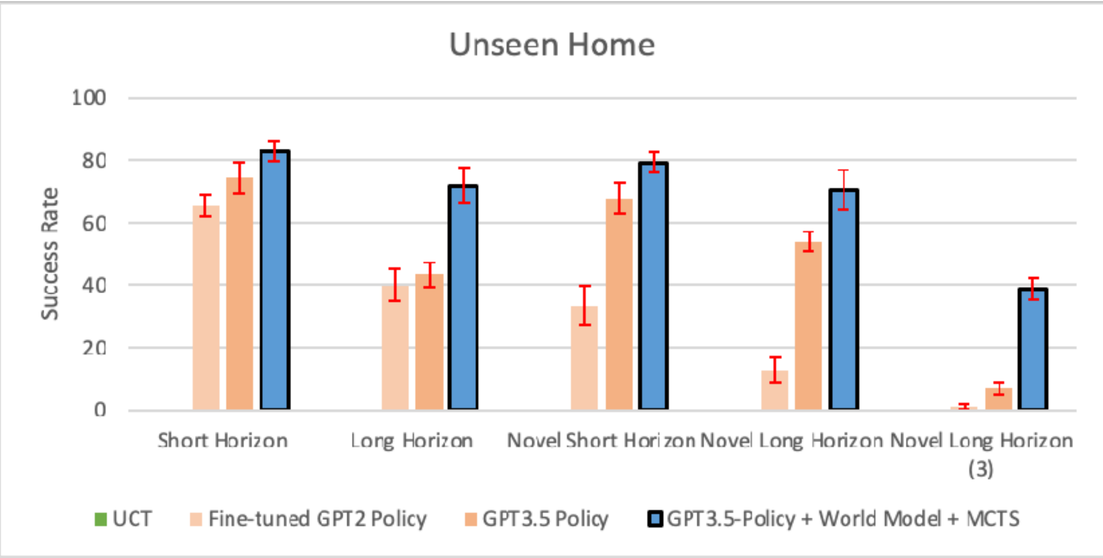
Experiments

- VirtualHome simulator
- Task: object rearrangements in household environments
  - Simple v.s. compositional tasks
  - In-distribution v.s. novel tasks
- Baselines:
  - LLM as world model: Upper confidence tree (UCT) without heuristic
  - LLM as Policy: GPT3.5 and GPT2 policy

Experimental results

- LLM as both the world model and policy outperforms either alone
  - A more accurate LLM world model improve the accuracy of LLM policy
  - LLM policy guides planning to make it more efficient

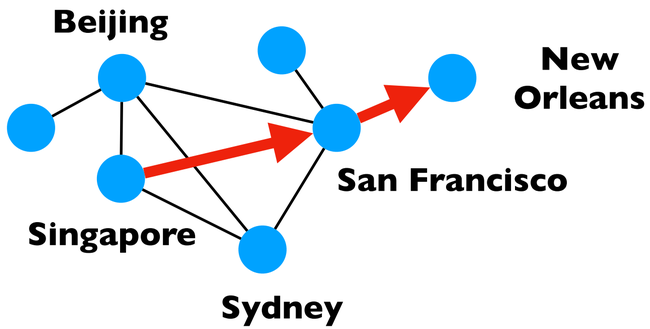


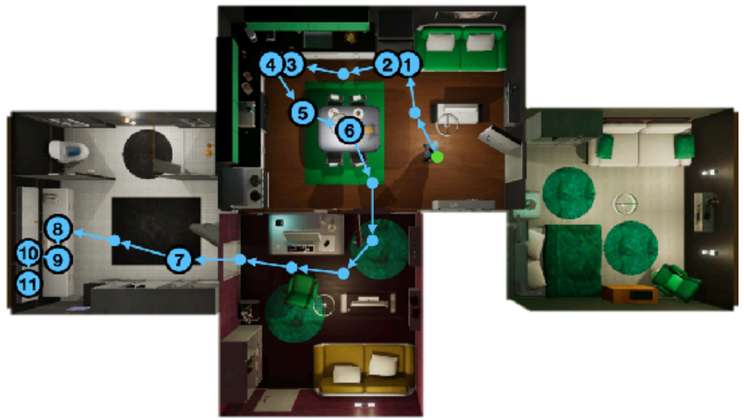


Data efficiency: Occam's Razor

- Using LLM as ***World Model*** or ***Policy***, which is better?
- *Minimum Description Length* (MDL): method with ***shorter description length*** has smaller generalization error[1]
- Analysis and experiments: multi-digit multiplication, travel planning, object rearrangement, …

14128 x 8634 = ?





Instruction: Put one apple on the kitchen table and one toothbrush inside the bathroom cabinet.
1: Walk to fridge
2: Open fridge
3: Walk to apple
4: Grab apple
5: Walk to kitchen table
6: Put apple on kitchen table
7: Walk to bathroom
8: Walk to toothbrush
9: Grab toothbrush
10: Open bathroom cabinet
11: Put toothbrush inside bathroom cabinet

[1] Shai Shalev-Shwartz and Shai Ben-David. Understanding machine learning: From theory to algorithms. Cambridge university press, 2014.

Example: multi-digit multiplication

- LLM Policy
  - Table with inputs and results
  - Description length: $O(n10^n)$
- LLM World Model + algorithm
  - Single-digit multiplication table by LLM
  - Algorithm
  - Description length: constant
- Empirical results

LLM Policy

	0	1	2	…	10^n -1
0	0	0	0	…	0
1	0	1	2	…	10^n -1
2	0	2	4		…
…	…	…		…	…
10^n -1	0	10^n -1	…	…	…

LLM World Model + Algorithm

	0	1	…	9
0	0	0	…	0
1	0	1	…	9
…	…	…	…	…
9	0	9		81

```
function multiply (x[1..p], y[1..q]):
  // multiply x for each y[i]
  for i = q to 1
    carry = 0
    for j = p to 1
      t = x[j] * y[i]
      t += carry
      carry = t // 10
      digits[j] = t mod 10
    summands[i] = digits

  // add partial results (computation not shown)
  product = ∑_{i=1}^{q} summands[q+1-i] · 10^{i-1}
  return product
```

Example: multi-digit multiplication

LLM World Model + Algorithm

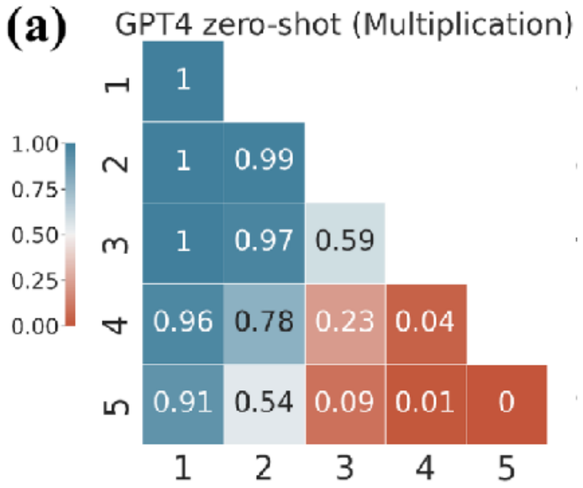
	0	1	...	9
0	0	0	...	0
1	0	1	...	9
...	...	...	...	...
9	0	9		81

```
function multiply (x[1..p], y[1..q]):
  // multiply x for each y[i]
  for i = q to 1
    carry = 0
    for j = p to 1
      t = x[j] * y[i]
      t += carry
      carry = t // 10
      digits[j] = t mod 10
    summands[i] = digits

  // add partial results (computation not shown)
  product = ∑_{i=1}^{q} summands[q+1-i] · 10^{i-1}
  return product
```

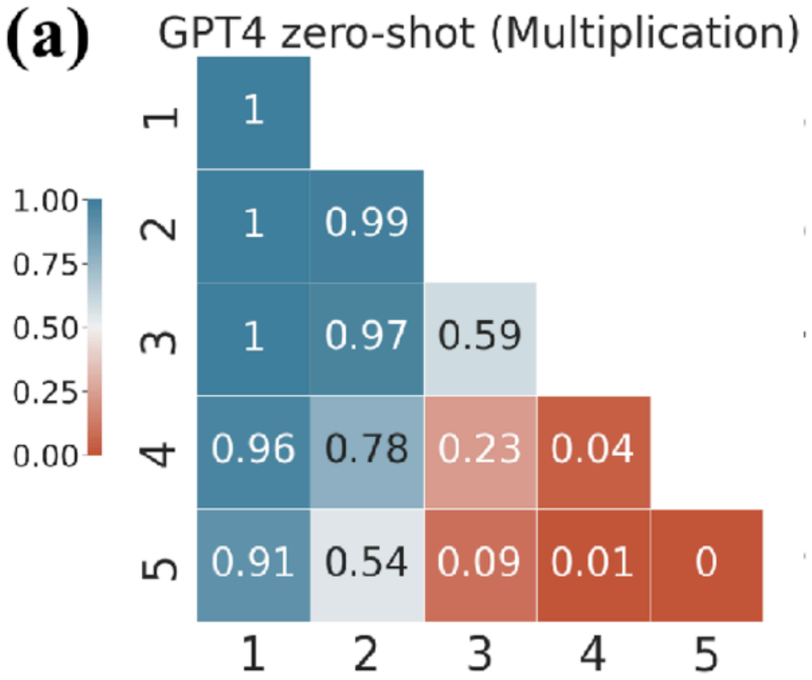
LLM Policy

	0	1	2	...	10^n -1
0	0	0	0	...	0
1	0	1	2	...	10^n -1
2	0	2	4		...
...	...	...		...	...
10^n -1	0	10^n -1	...	...	...



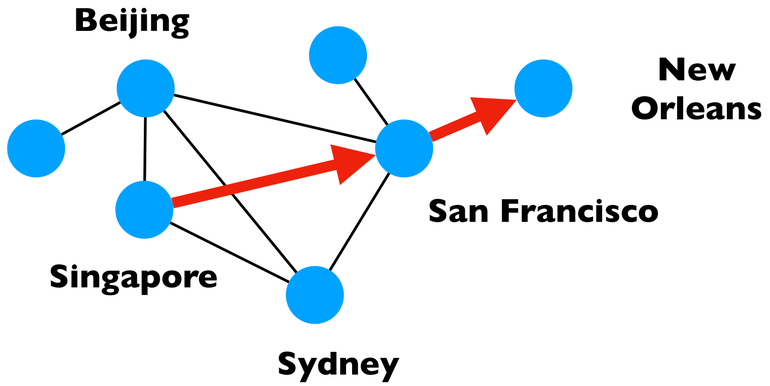
Example: multi-digit multiplication

- LLM Policy
  - Table with inputs and results
  - Description length: $O(n10^n)$
- LLM World Model + algorithm
  - Single-digit multiplication table by LLM
  - Algorithm
  - Description length: constant
- Empirical results



Example: travel planning

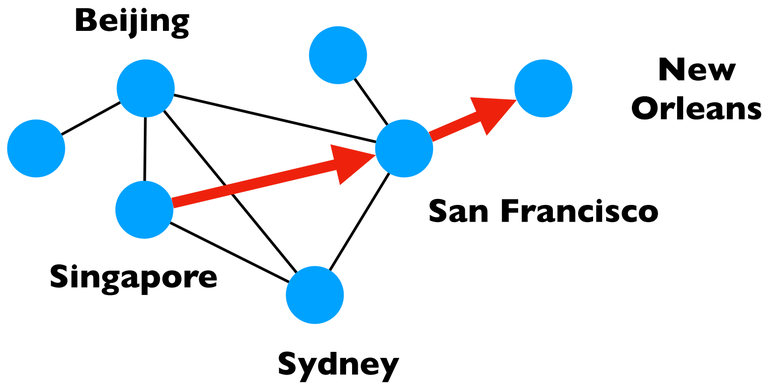
- Problem: predict flight routes between given cities
- LLM Policy: table of travel
  - Description length: $O(n^2 \log n)$
- LLM World Model solution: flight graph+search
  - Description length: $O(n \log n)$
- Results: LLM World Model solution works better



Current\goal	New Orleans	Sydney	…
Singapore	San Francisco	Sydney	
Sydney	San Francisco	—	
San Francisco	New Orleans	Sydney	
…			

Example: travel planning

- Problem: predict flight routes between given cities
- LLM Policy: table of travel
  - Description length: $O(n^2 \log n)$
- LLM World Model solution: flight graph+search
  - Description length: $O(n \log n)$
- Results: LLM World Model solution works better

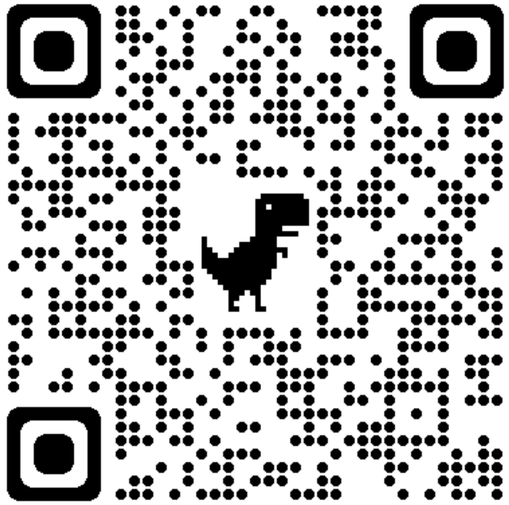


Summary

- LLM as world model and policy outperforms either one
- Choose between LLM world model and policy? Use MDL principle: shorter description length is better

Thank You!

Paper and Code



Contact

ziruiz@comp.nus.edu.sg

Website

https://llm-mcts.github.io