What is On / Off Policy?

- Q learns how to perform optimally even when we are following a non-optimal policy
- In ε–greedy, ε leaves no trace in Q
- SARSA is on-policy
- Learns the best policy given our systematic departures from true optimal
- In ε–greedy, ε is reflected within SARSA’s Q values

On Policy vs Off Policy

- Q, an off-policy learner:

$$Q(a,s) \leftarrow Q(a,s) + \alpha \cdot \left(r_s + \gamma \max_{a'} Q(a',s') - Q(a,s)\right)$$

- SARSA, an on-policy learner:

$$Q(a,s) \leftarrow Q(a,s) + \alpha \cdot \left(r_s + \gamma \cdot Q(a',s') - Q(a,s)\right)$$

- How do they differ?
  - Remember the persistent need for exploration
  - Consider cliff walking

Can Q use more Information?

- Efficient = use all information
- Indirect learners can us the approx. model
- No information until rewards
- Information is propagated one step back
- Can we do better?

Eligibility traces

Another parameter: λ

Between 0 and 1 (close to 1)

TD(λ), Q(λ), SARSA(λ)

Deficiencies with RL
(and statistical learning generally)

- Constrained expressiveness
- Generally limited to propositional representations
- Stems from formalizing with Random Variables
- What is Propositional?
- What are the alternatives?
- Complexity of change
- Structure among world states

A Relevant Example

Robot R delivers gifts from location J to locations G1 and G2

Thief T may steal the gifts

Actions: Left, Right, Up, Down

	X	G1		X
	R			
X			T	
			X	G2
		J		

MP2 World

Propositional state representation

Effects of actions “Right: $x \leftarrow x+1$

Structure among world states

Goals / subgoals

The Curse of Statistical Learning

- Information required for approx. π*
- Looser approximations require less information
  - How close to optimal?
  - 5x5 grid vs. 50x50
  - (how many places can *you* be)
- Empirical data makes up the difference
- Impoverished expressiveness
- Everything we know but do not tell, inflates the empirical data requirement
- Consider learning brain surgery by example

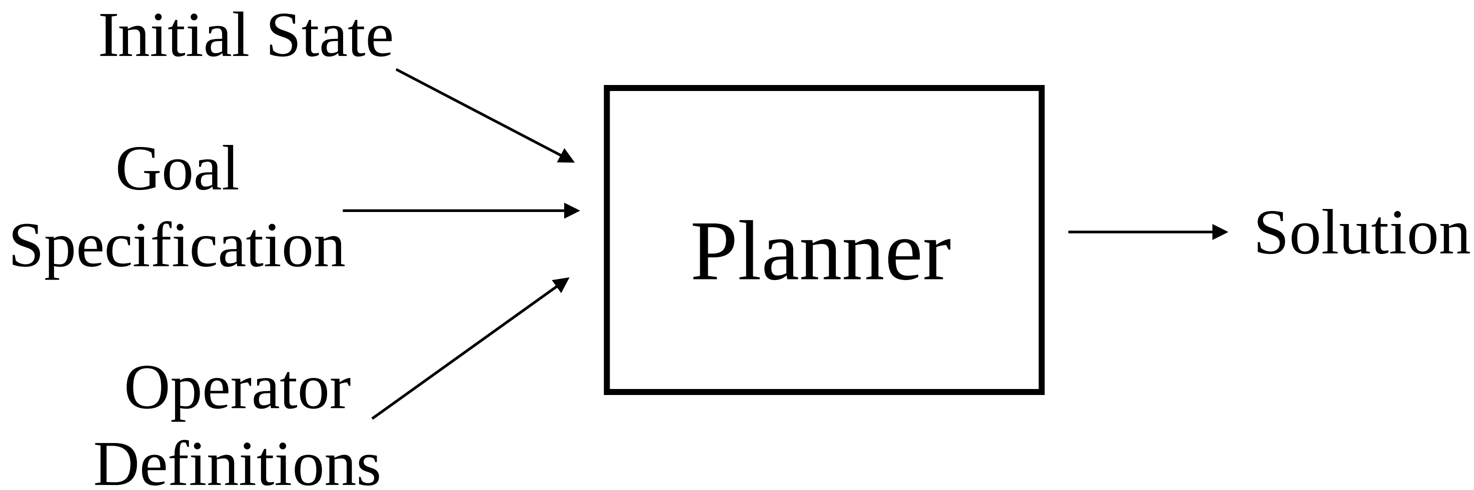
Reinforcement Learning (vs. Classical Planning)

- Robust
  - Fewer *a priori* assumptions (esp. actions)
  - Empirical model
  - Fit (via parameter adjustment) to the *observed* world
- Scaling difficulties
  - Propositional expressiveness
  - Complexity space / time (?)
  - States / Features (e.g., block positions)
- Markov assumption
  - Our world?
  - Implications for sensors & convergence
  - Discretizing may not respect Markov

Classical Planning

Domain Independent Planning

Using inference to find a sequence of operator instances (actions) that transform an initial state into a state in which the goal is satisfied.



Real World Applications:

Scheduling, Semantic web support, Computer gaming, …

(limited by assumption of an analytic model)

Recall from Search

| Interesting action sequences | << | All action sequences |

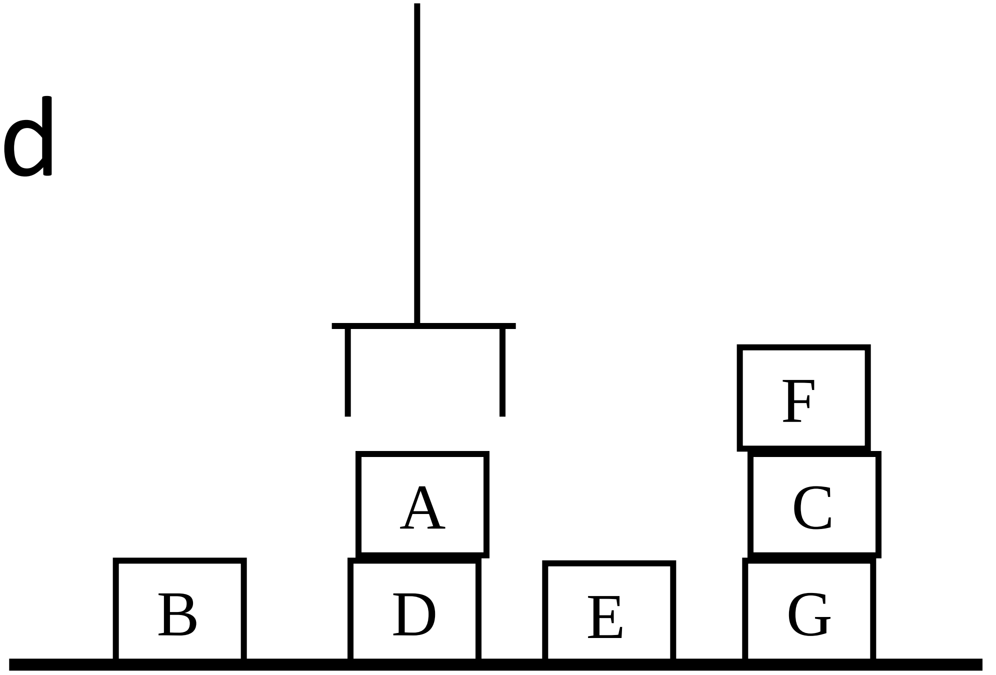
RL actions are "inferentially opaque"

World state is a collection of relevant *objects* and their *relationships*

Features impose a structure among states

Planning allows reasoning about state features

Blocks World



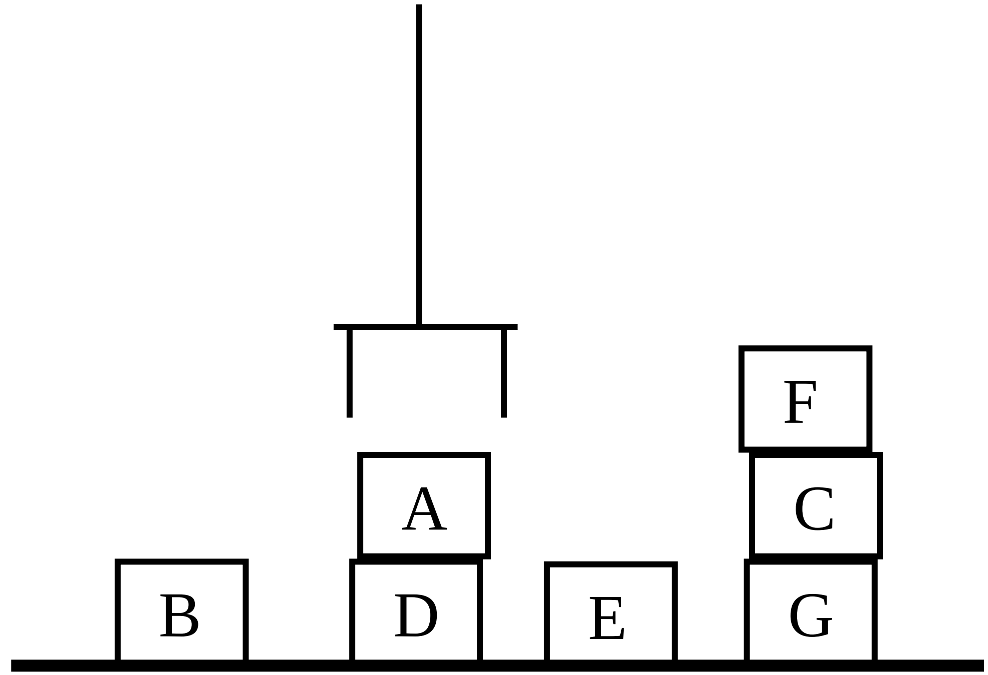
Several ontologies possible (ways to conceptualize the world and its changes)

Operator - General knowledge of one kind of change

Action - Ground instance of an operator (RL has only these)

Silly domain but concisely illustrates many GENERAL planning issues

Traditional Blocks World



Only keep track of support relationships: On, Clr

A block can support at most one other block

The table can support any number of blocks

Generalized block movement – no gripper

Alternative Ontologies

change a block's position differently

Move-Block

Move-Gripper
Grasp-Block
Move-Gripper
UnGrasp-Block

Move-Gripper
Open-Gripper
Move-Gripper
Close-Gripper
…

Motor1-Velocity
Motor2-Velocity
…

Motor1-Voltage (Current, Duty Cycle)
Motor2-Voltage
…

…

Levels of Ontological Commitment

Planning

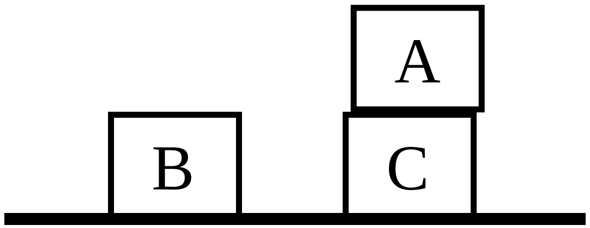
Abstract, High-Level Ontology
Action(Achieve-Block-Configuration3)
Problem is trivialized

Mid-Level Ontology
Action(…)

Assume we're here

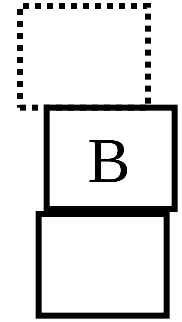
Assumed Hardware Support

Low-Level Ontology
Action(Motor3, Voltage7)
Artificially and unnecessarily difficult



Initial State

On(A, C)
On(C,Tbl)
On(B,Tbl)
Blk(A)
Blk(B)
Blk(C)
Table(Tbl)
Clr(A)
Clr(B)
Clr(Tbl)

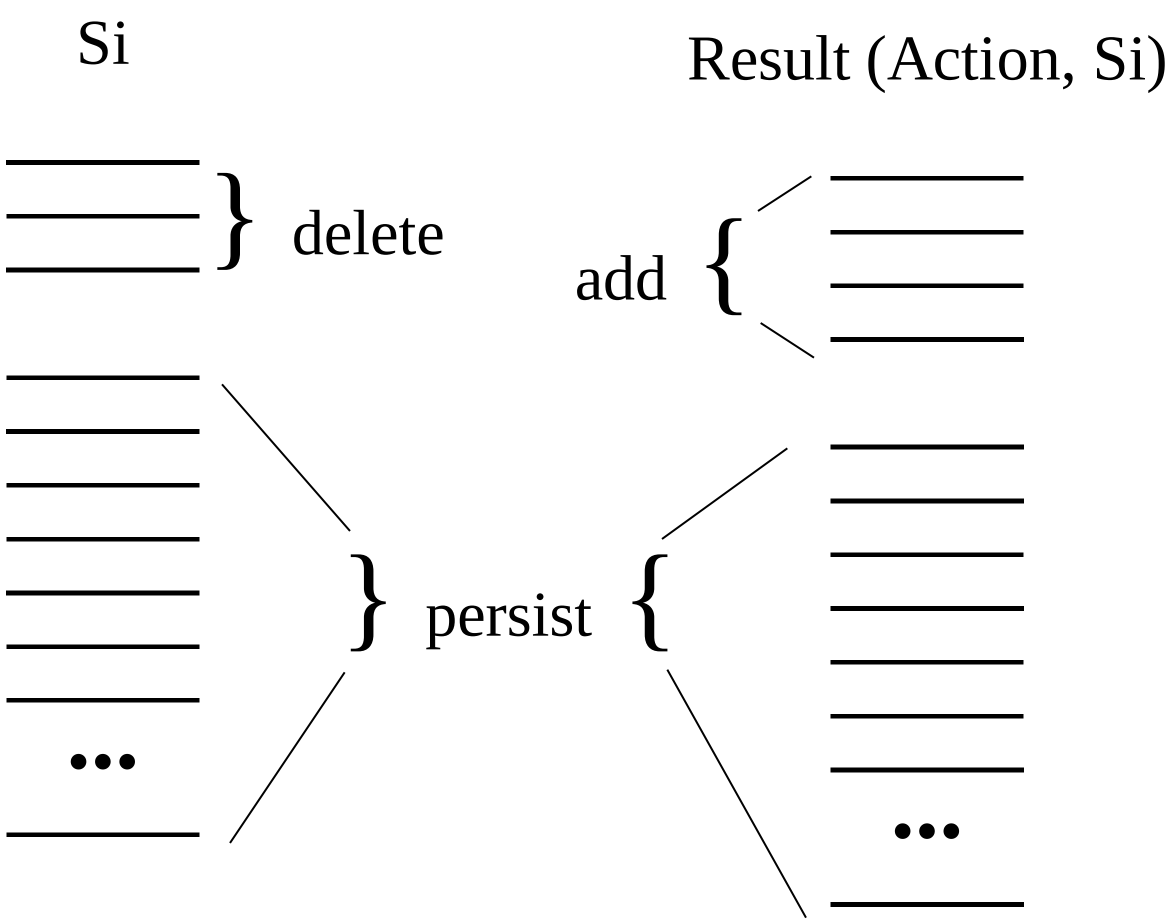


Goal

On(B, ?x)
Blk(?x)

World Changes

Operator must fully define resulting world state from any legal previous state



Strips Operators

- Preconditions: list of features that must hold in the world to apply the operator
- Effects
  - Delete list: features ceasing to hold
  - Add list: features asserted by the action

the Move operator

obj from to

Move(?x, ?y, ?z)

What holds in the resulting situation?

?x is on ?z

?y is clear

...

What must hold now?

?x is on ?y

?z is clear

?x is a block

?x is clear

...

Two Strips Operators for Move

MoveToBlock (x, y, z):

PC: Clr (x), Clr (z), On (x, y), Blk (x),
Blk (z), Diff (x, z), Diff (y, z)

Delete: On (x, y), Clr (z),
Add On (x, z), Clr (y)

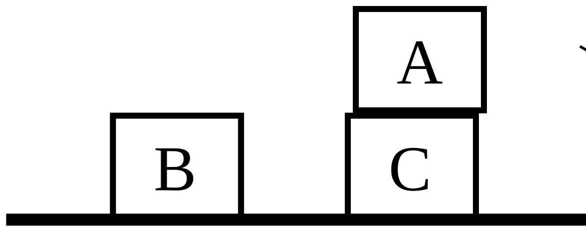
MoveToTable (x, y):

PC: Clr (x), On (x, y), Blk (x), Diff (y, Tbl)

Delete: On (x, y)
Add: On (x, Tbl), Clr (y)

World Change

Initial State: Si



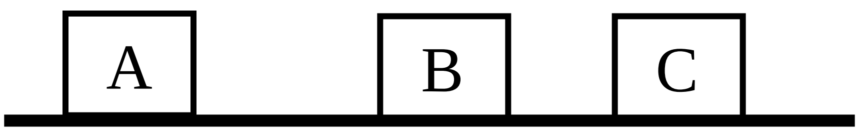
…

On(A, C)

...

MoveToTable (A, C)

State after MoveToTable (A, C) in Si



…

On(A, Tbl)

...

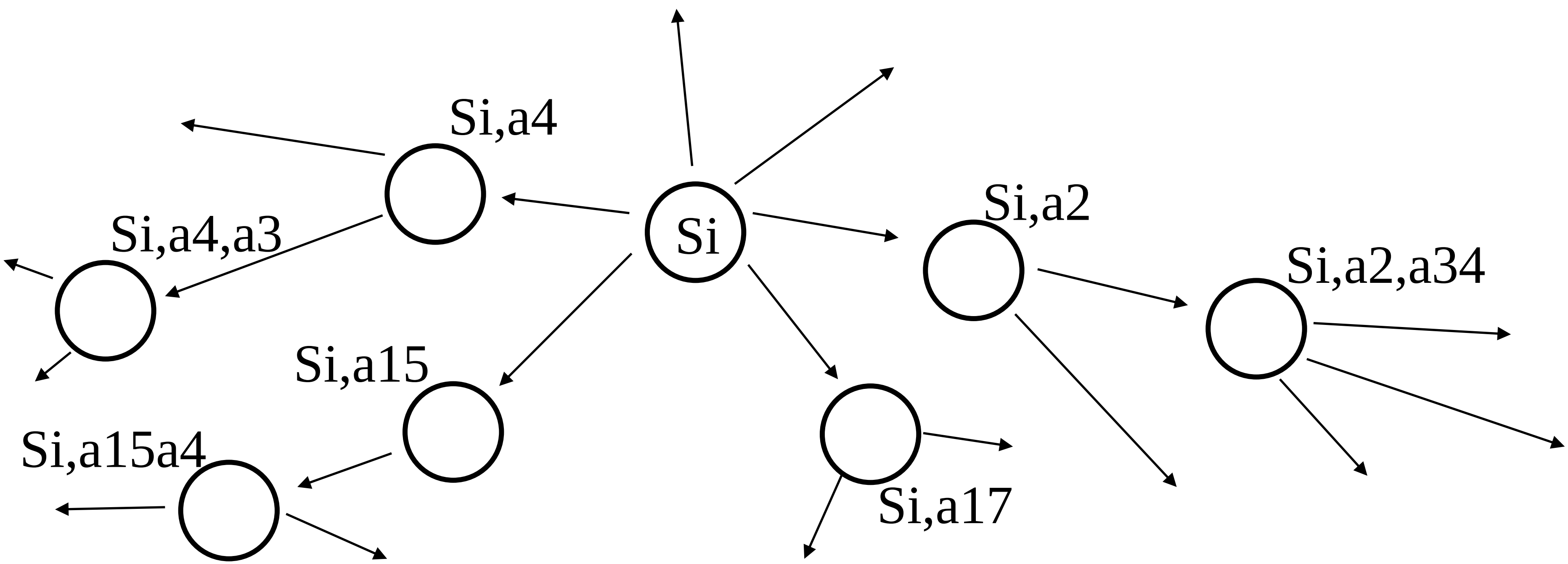
All Reachable Situations are Defined

Given: 1) the Initial State

2) Axioms of World Change (operator definitions)

Can be realized as predicate calculus theorem proving

$\Delta \equiv$ Initial State $\cup$ Operator Definitions



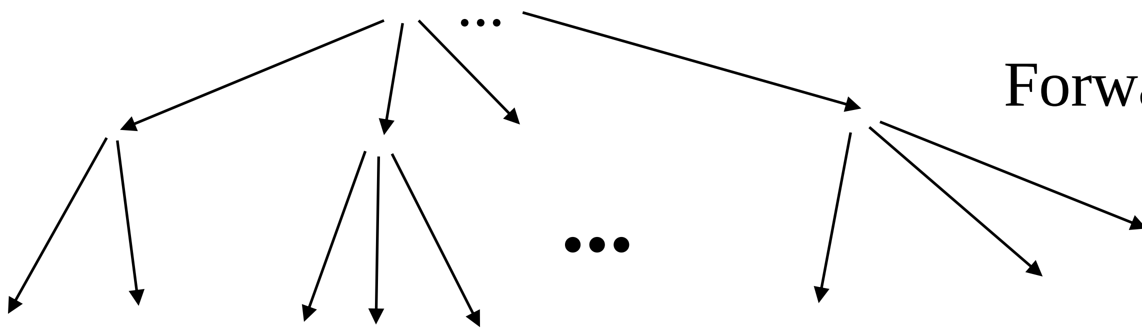
Planners

- State space vs Plan space
- Linear vs. Nonlinear
- Goal-stack planning
- POP
- Difficulties with classical planning

Learning search heuristics
ADL – action description language
HTN – hierarchical task network planning
GraphPlan
SatPlan

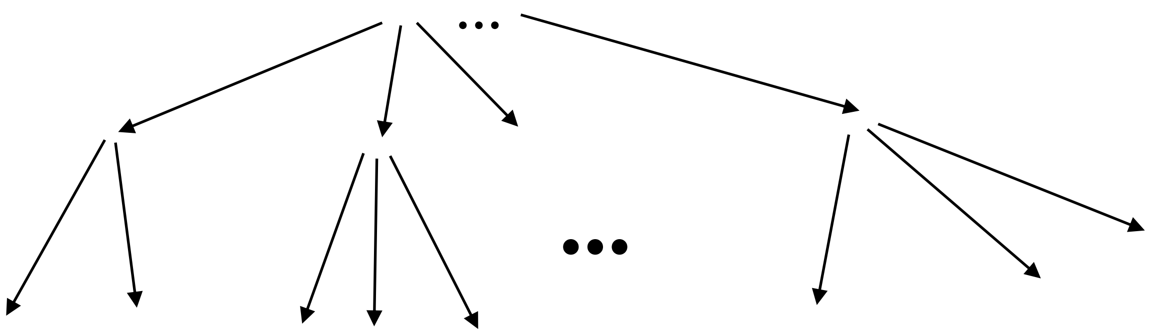
State Space Planner

Initial State



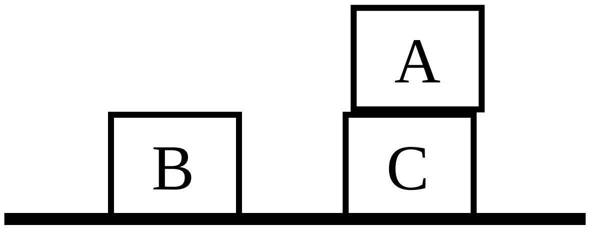
Forward Search for Goal

Goal



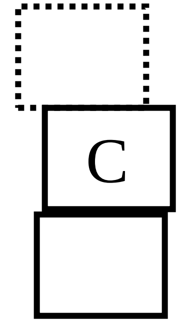
Backward Search for subset of Initial State fluents

One reason why planning beats searching



Initial State Si

On(A, C, Si)
On(C,Tbl, Si)
On(B,Tbl, Si)
Blk(A)
Blk(B)
Blk(C)
Table(Tbl)
Clr(A, Si)
Clr(B, Si)
Clr(Tbl, Si)



Goal ?s

Find an ?x and ?s s.t.:
On(C, ?x, ?s)
Blk(?x)

Negate Goal, add to axioms w/ Answer literal

Answer (?s) should yield something like

Answer (Result (Move (C, Tbl, B),
Result (Move (A, C, Tbl), Si)))