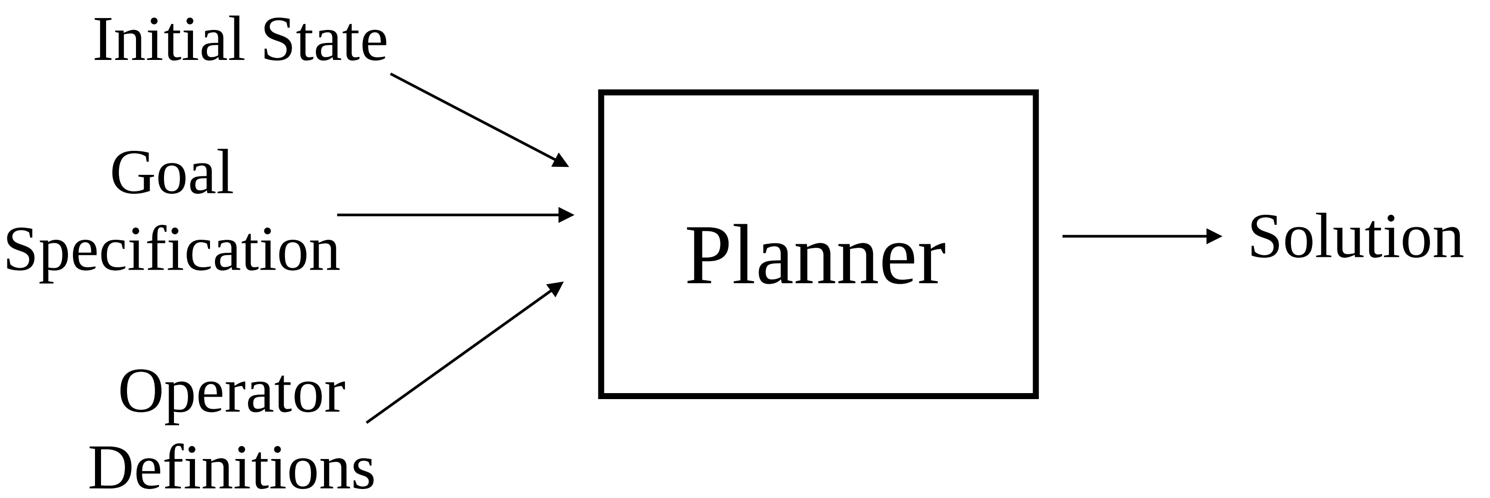
- Homework 2A due Thursday
- Planning: Today & part? of Thursday
- Next Reinforcement Learning
- Begins statistical AI
- Start reading Ch 17 & 21

Classical 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, Logistics, Semantic web support, Computer gaming, …

Planning vs. Search

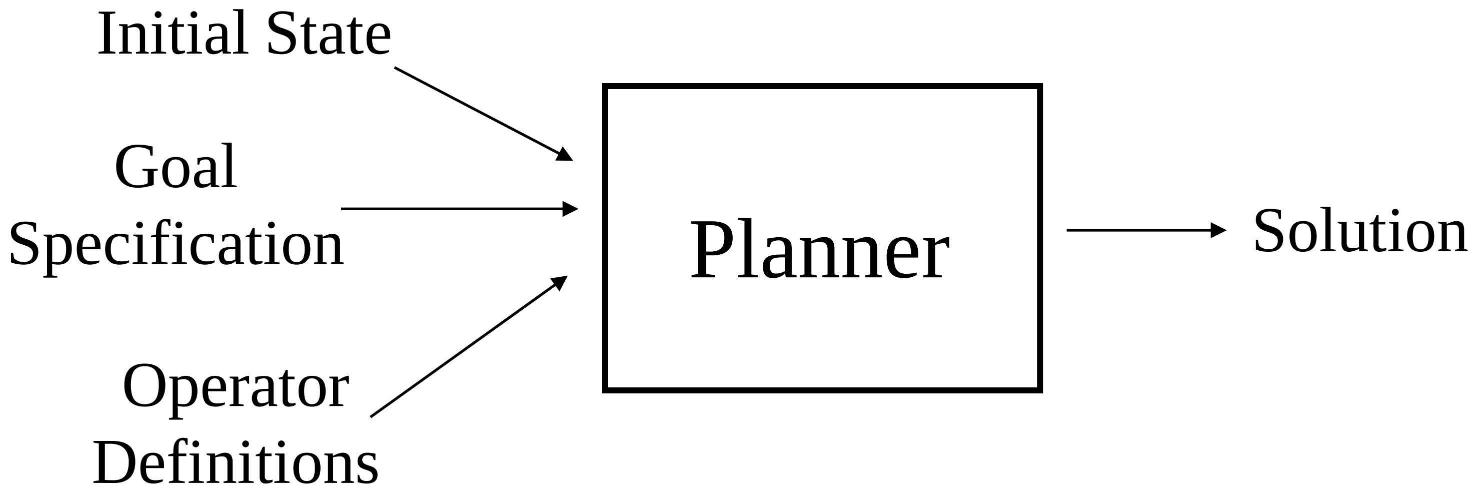
$$\left|\text{Interesting action sequences}\right| \ll \left|\text{All action sequences}\right|$$

Search operators are "inferentially opaque"

Planning allows reasoning about state features

Domain Independent Planning

- Study the planning process
  - Abstract
  - Not domain dependent
- Ontology, operators, etc. define the domain



- Operators model world dynamics
  - Situation Calculus
  - Strips Operators
  - PDDL Operators*
- Search
  - State Space: Forward / Backward
  - Plan Space
- Heuristics
- Propositionalization

Pure FOPC
↓
Specialized syntax

* Ch10 R & N say PDDL but actually discuss Strips

All Reachable Situations are Defined

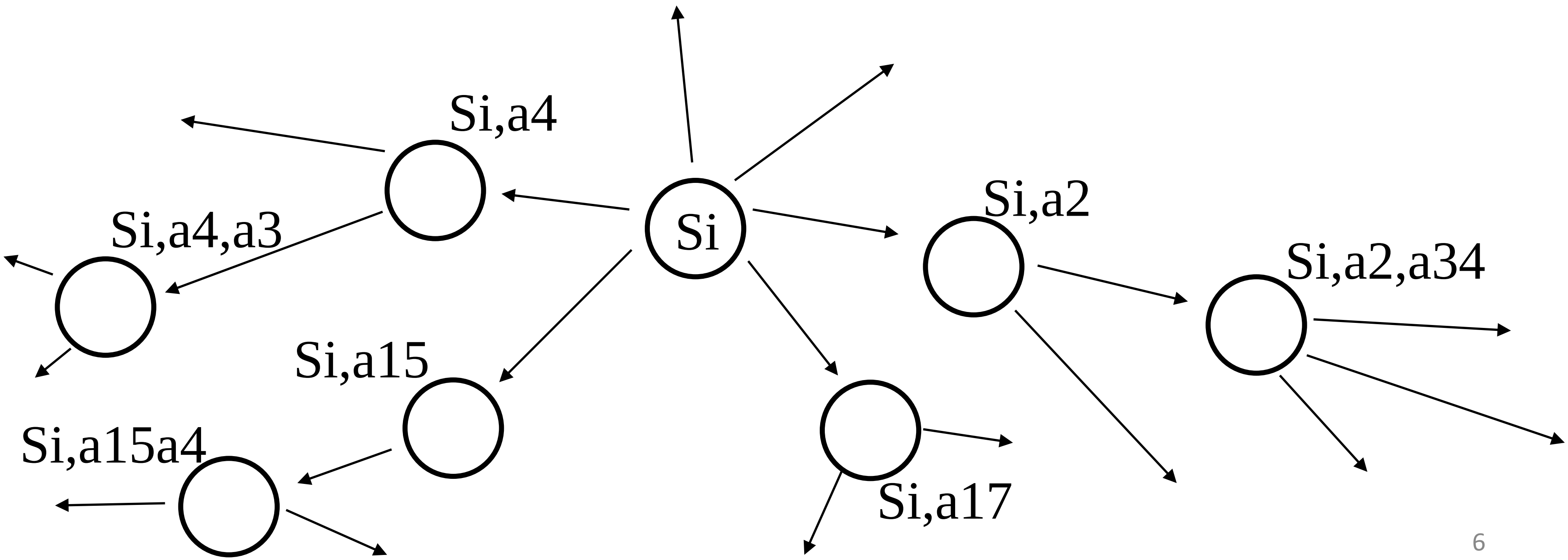
Given: 1) the Initial State

2) Axioms of World Change (operator definitions)

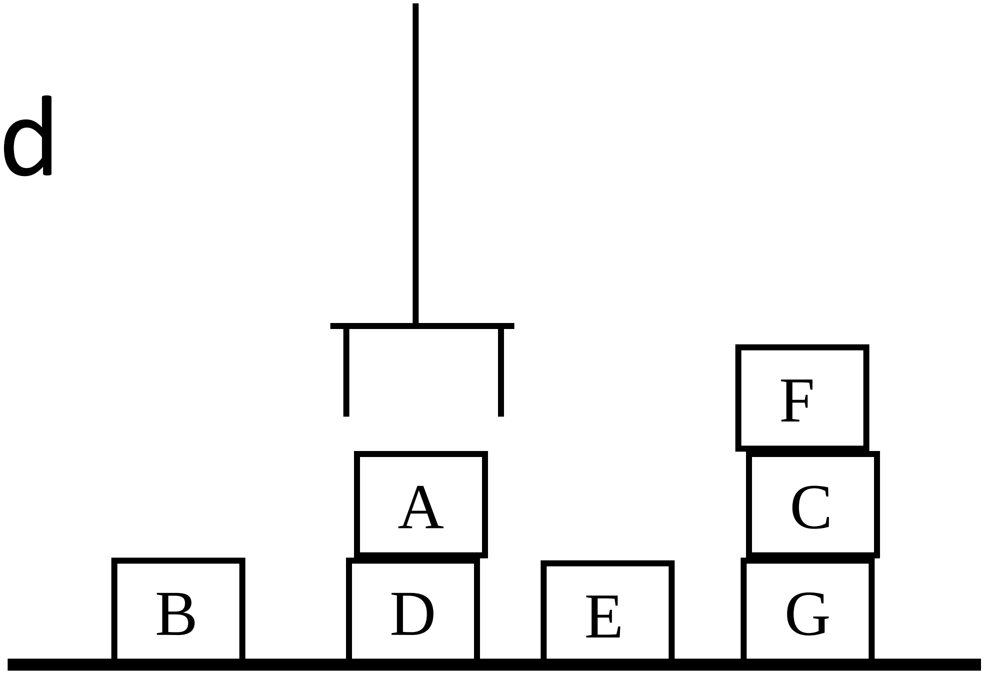
$\Delta \equiv$ Initial State $\cup$ Operator Definitions

Planning is theorem proving

Find a situation where the goal holds



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

Silly domain but concisely illustrates many GENERAL planning issues

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

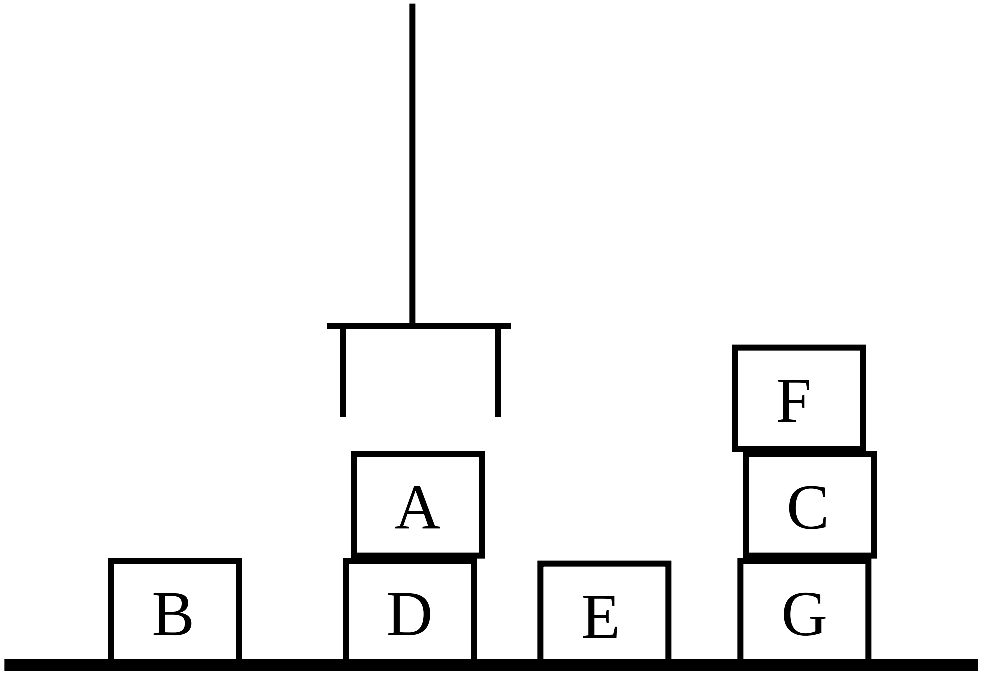
Assumed Hardware Support

Low-Level Ontology

Action(Motor3, Voltage7)

Artificially and unnecessarily difficult

Traditional Blocks World



Only support relationships change: On, Clr

A block can support at most one other block

The table can support any number of blocks

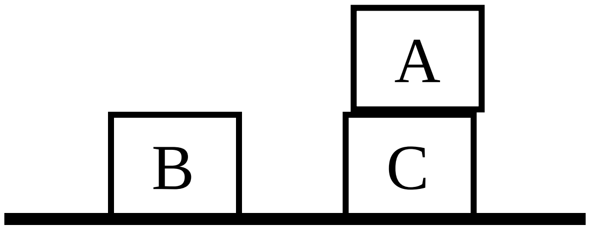
Generalized block movement – no gripper

Operators: Situation Calculus

FOPC with some conventions

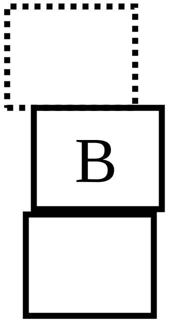
Assume a Move-Block ontology with
- at most one block directly on top of another
- a big table (always empty space available)

Move(x, y, z) operator to move x from y to z



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)

Strips and PDDL can use this.
Situation Calculus cannot. (why?)

FOPC Inference is Monotonic

On(A, B) and Clr(C)

vs.

Blk(A) and Table(Tbl)

Situation Calculus solution:

On(A, B, S)

Holds(On(A, B), S)

Situation Designator

BTW, what's a predicate?

Fluents

- Relationships that may be situation sensitive
- “On” & “Clr” relationships can change
- On(x, y) or On(x, y, s) is a fluent
- Blk(x) need not be

The "Result" Function:

Result: Action × Situation → Situation

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

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

It denotes; it is not truth-valuable

Straightforward generalization to variables:

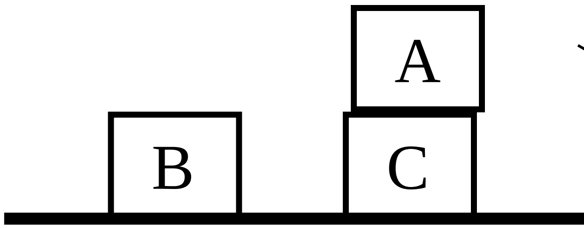
Result (Move (?x, ?y, C), Si)

denotes the set of situations where something was just moved to C from the initial state Si

Useful in "Goal Regression" planning

World Change

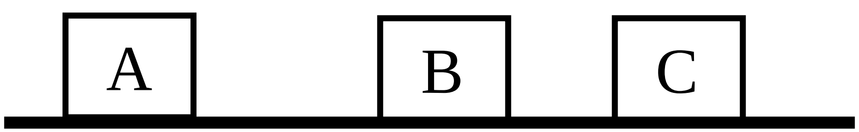
Initial State: Si



…
On(A, C, Si)
...

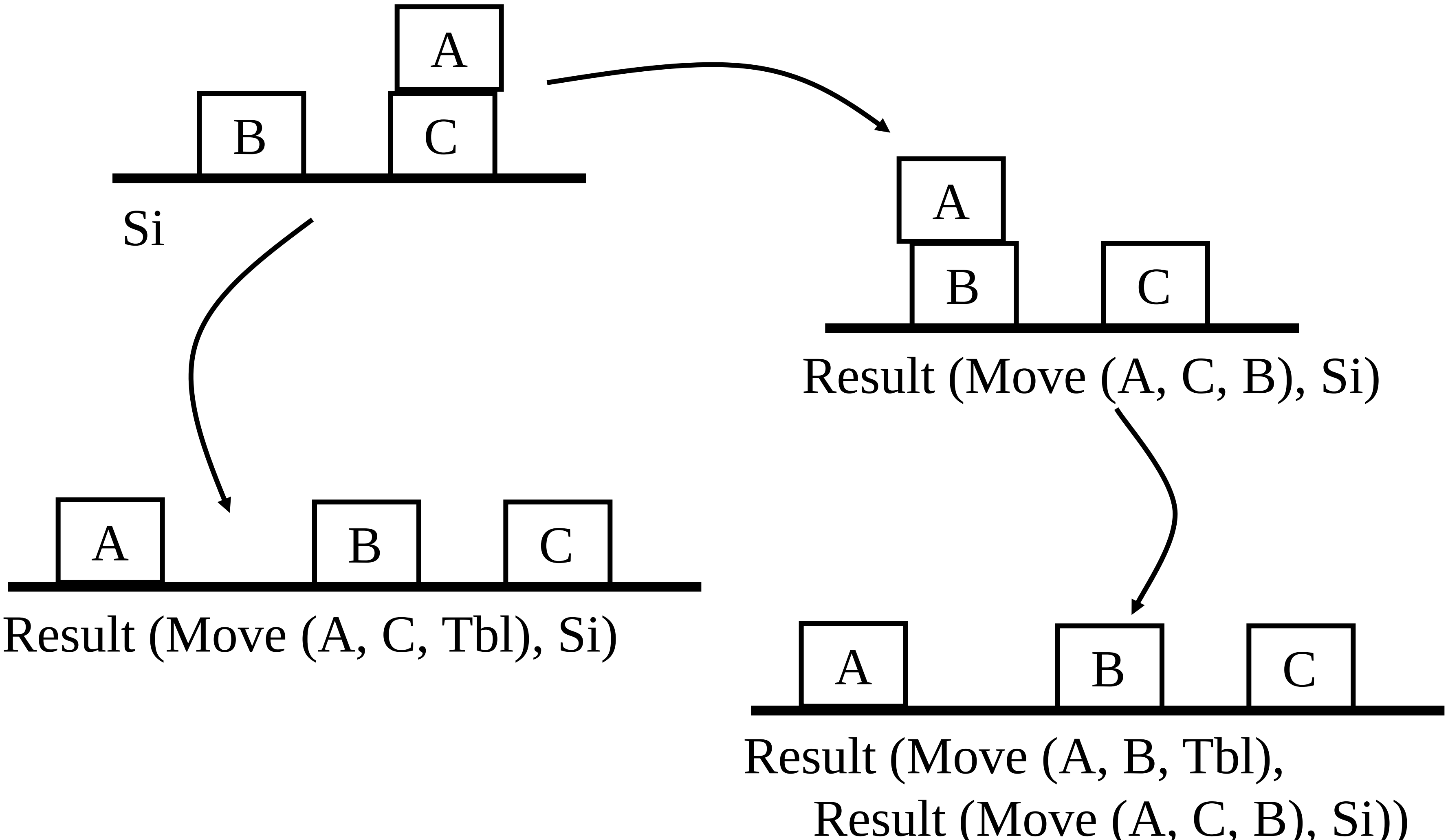
Move (A, C, Tbl)

Next State: Result (Move (A, C, Tbl), Si)



…
On(A, Tbl, Result (Move (A, C, Tbl), Si))
...

In Situation Calculus, States / Situations are Individuated by History and not Block Configuration



the Move operator

Move(x, y, z) definition has the form:

$\forall x \, \forall y \, \forall z \, \forall s \; \Theta \Rightarrow \Psi$

If Θ holds (things in s)

x is on y

z is clear

x is a block

x is clear

...

Preconditions

Then Ψ will hold (things in Result(Move(x,y,z),s))

x is on z

y is clear

...

Effects

the Move operator

(partial)

Move(x, y, z)

$\forall x\ \forall y\ \forall z\ \forall s$ [

(Clr (x, s) $\wedge$ Clr (z, s) $\wedge$ On (x, y, s)
$\wedge$ Blk (x) $\wedge$ Diff(x, z) $\wedge$ Diff(y, z))

$\Rightarrow$

(On (x, z, Result (Move (x, y, z), s)) $\wedge$

Clr (y, Result (Move (x, y, z), s)) $\wedge$

Clr (x, Result (Move (x, y, z), s)) $\wedge$

Table (z) $\Rightarrow$
Clr (z, Result (Move (x, y, z), s)))]

Conditional Effect

Only Partial. Why?

Do we need to retract fluents?

On (x, y, s) - situation-specific relations

Do we need to assert negative fluents?

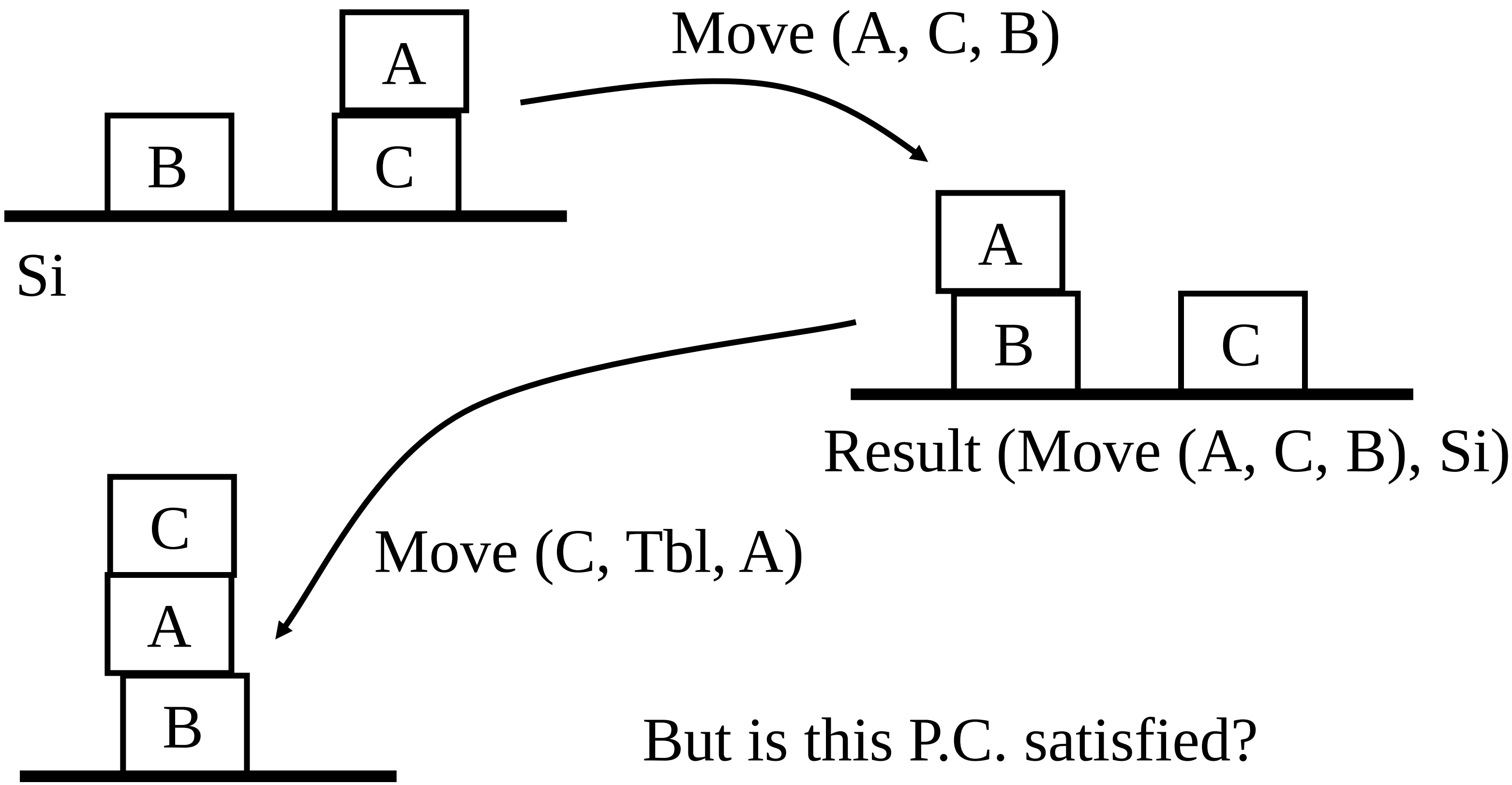
$\neg$ On (x, y, Result (Move (x, y, z), s)) ?

No, not in Situation Calculus
(why not?)

Frame Axioms

- Logic requires an inference path to determine that something holds
- Some relations are not involved
- May need to use these relations later
- If they don’t persist through Move…

The Need for Frame Axioms



But is this P.C. satisfied?
On (C, Tbl, Result (Move (A, C, B), Si)

And suppose there were other blocks: D, E, F…

Move Frame Axioms

Move(x, y, z)

$\forall x\ \forall y\ \forall z\ \forall s\ [$

$(Clr\ (x, s) \wedge Clr\ (z, s) \wedge On\ (x, y, s) \wedge Blk\ (x) \wedge Diff(x, z))$

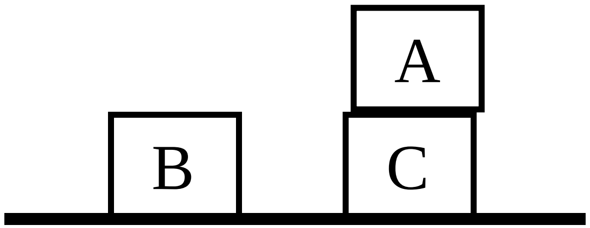
$\Rightarrow$

$([\forall v\ \forall w\ (On\ (v, w, s) \wedge Diff(v, x)) \Rightarrow$

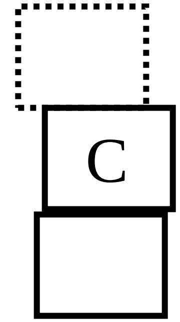
$On\ (v, w, Result\ (Move\ (x, y, z), s))\ )\ ] \wedge$

$[\forall v\ (Clr\ (v, s) \wedge Diff(v, z)) \Rightarrow$

$Clr\ (v, Result\ (Move\ (x, y, z), s))\ )\ ]\ )\ ]$



Initial State Si



Goal ?s

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)

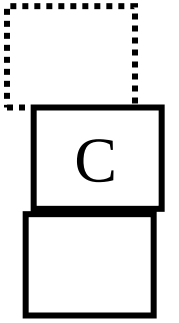
Find an ?x and ?s s.t.:
On(C, ?x, ?s)
Blk(?x)

Axioms ≡ Operator definitions ∪ Initial State

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)))



Goal ?s

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

Negate Goal, add to axioms w/ Answer literal

Goal: $\exists x\ \exists s\ [\mathrm{On}(C, x, s) \wedge \mathrm{Blk}(x)]$

Negated Goal $\forall x\ \forall s\ [\neg\ \mathrm{On}(C, x, s) \vee \neg\ \mathrm{Blk}(x)]$

Clause form w/ Answer literal, variables standardized apart and designated with '?' {¬On(C, ?x6, ?s8), ¬Blk(?x6), Answer(?s8)}

Situation Calculus

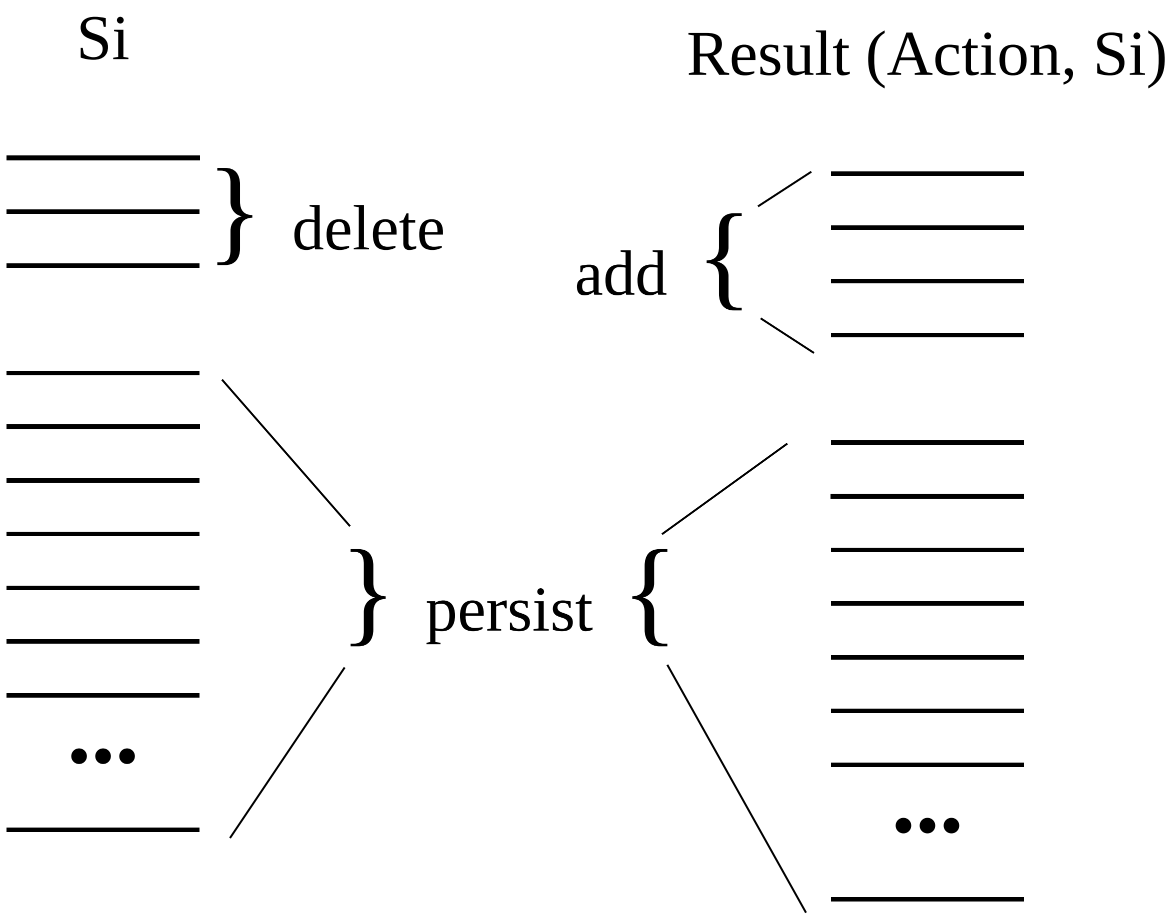
- No central operator “definition”
- Knowledge about an operator can be distributed across many WFFs
- Consider “Move” in clause form

STRIPS Operators

- Frames from animated cartoon “frames”
- Writing them can be tedious
- Luckily relatively few things change
- Strips operators are more concise
- Historically: Stanford Research Institute Problem Solver

World Changes

Action must fully define resulting world state



Operators

In Situation Calculus

Specify fluents
- Add set
- Persist set

No mention =
- no inference path

By default
- fluents are Deleted

In Strips

Specify fluents
- Delete set
- Add set

By default
- fluents Persist

More concise because usually
|Persist| >> |Delete|

Strips Operators

- Preconditions - list of positive literals
- Effects also positive literals (N.B. below)
  - Delete list - things to be retracted
  - Add list - things to be asserted
- Effects can be combined in one list (as R & N)
  - Delete elements designated with "$\neg$"
  - This is *not* logical negation (think about why)

Representations

In Situation Calculus

Δ contains all initial WFFs

No distinction between operators and initial state

Operator definitions distributed throughout Δ

In Strips

Operator information is centralized

Operator information is stored separately

State information is stored separately for each state

No longer need a situation designator

Closed world assumption