Chunking

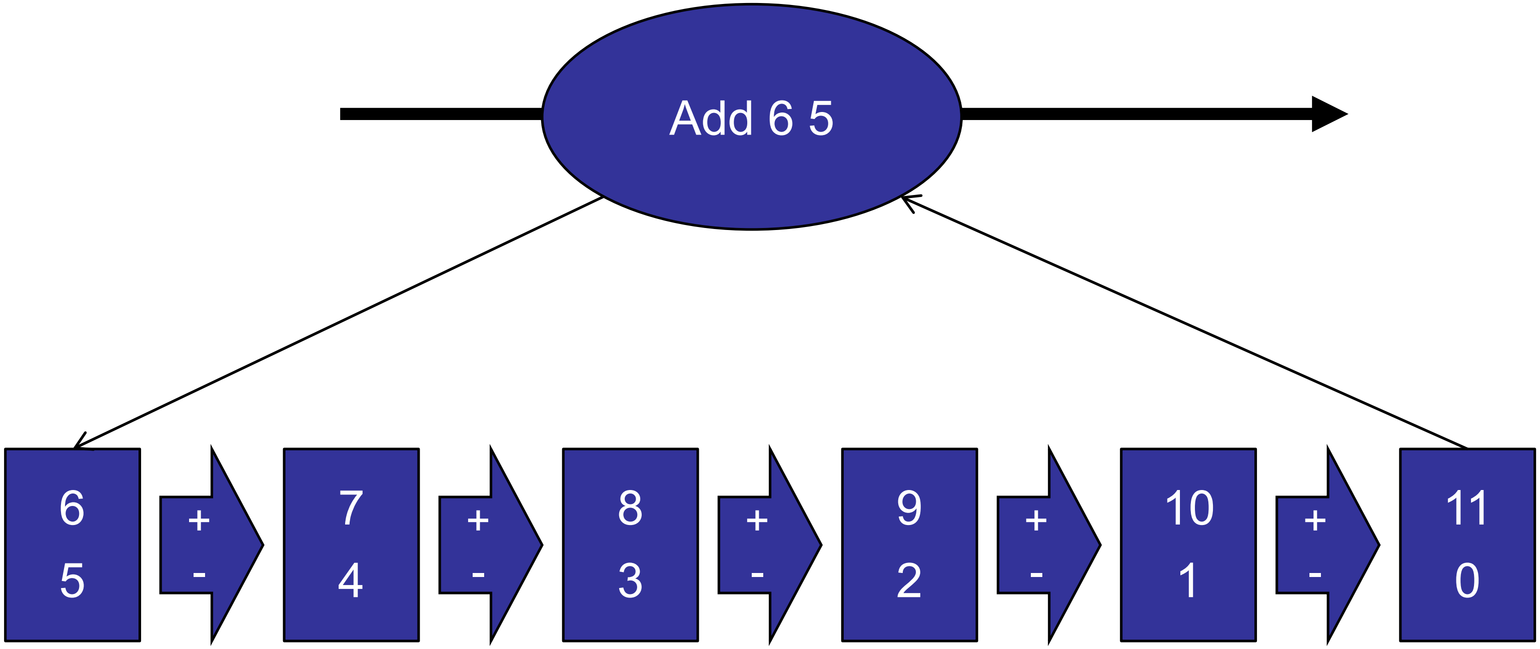
Soar Tutorial
May 7, 2019

Learning/Chunking

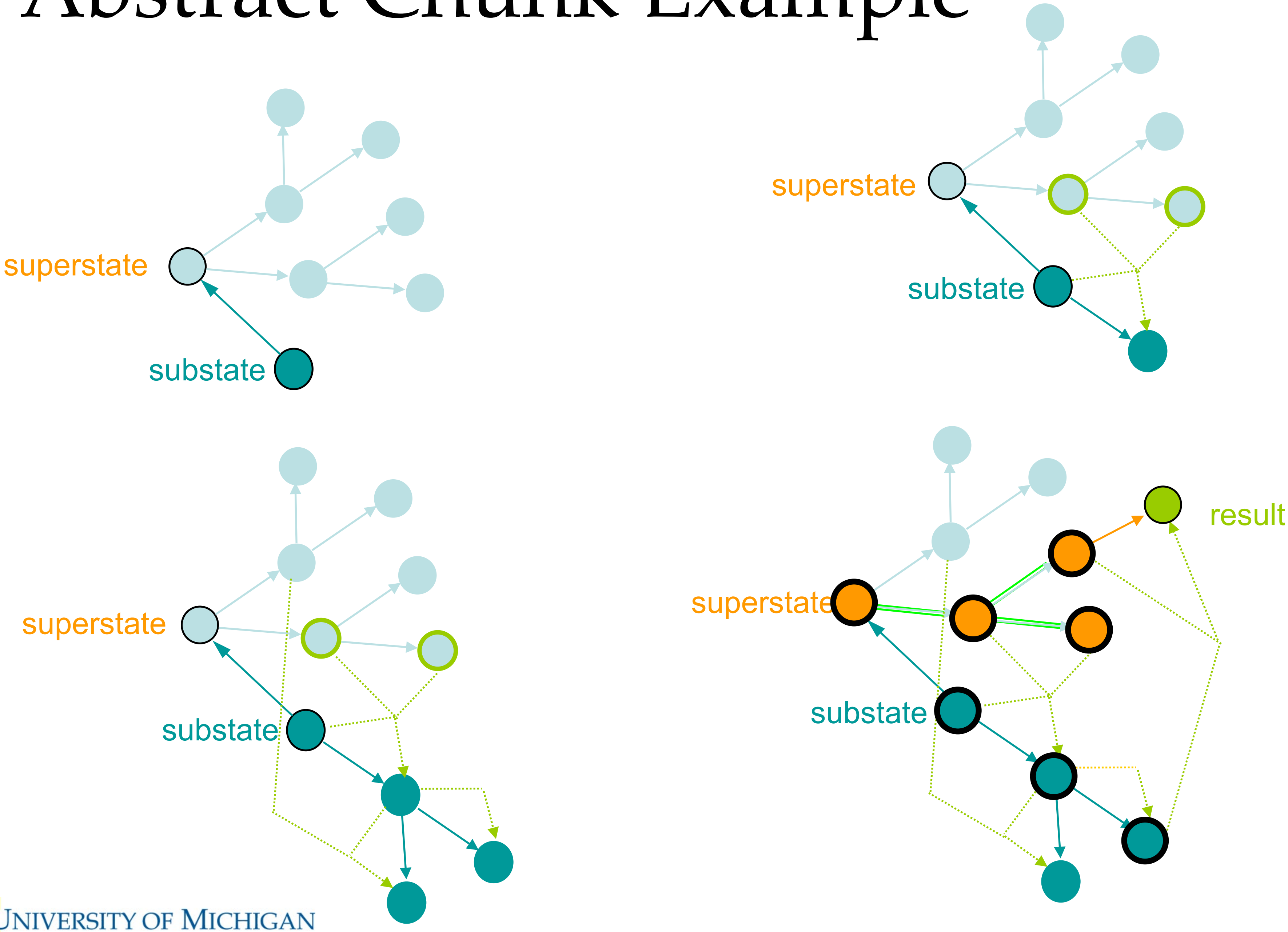
- Problem:
  - Subgoals “discover” knowledge to resolve impasses but it is lost after each problem solving episode
- Approach
  - Automatically build rules that summarize processing
    - Variablize justifications = chunks
    - Variablizes based on tests in rules that fired in substates
    - Conditions include only those test required to produce result
  - Chunks are built as soon as a result is produced
    - Immediate transfer is possible
  - One chunk for each result, where a result consists of connected WMEs that become results at the same time
    - Different results can lead to very different conditions
    - Improves generality of chunks

Operator Implementation

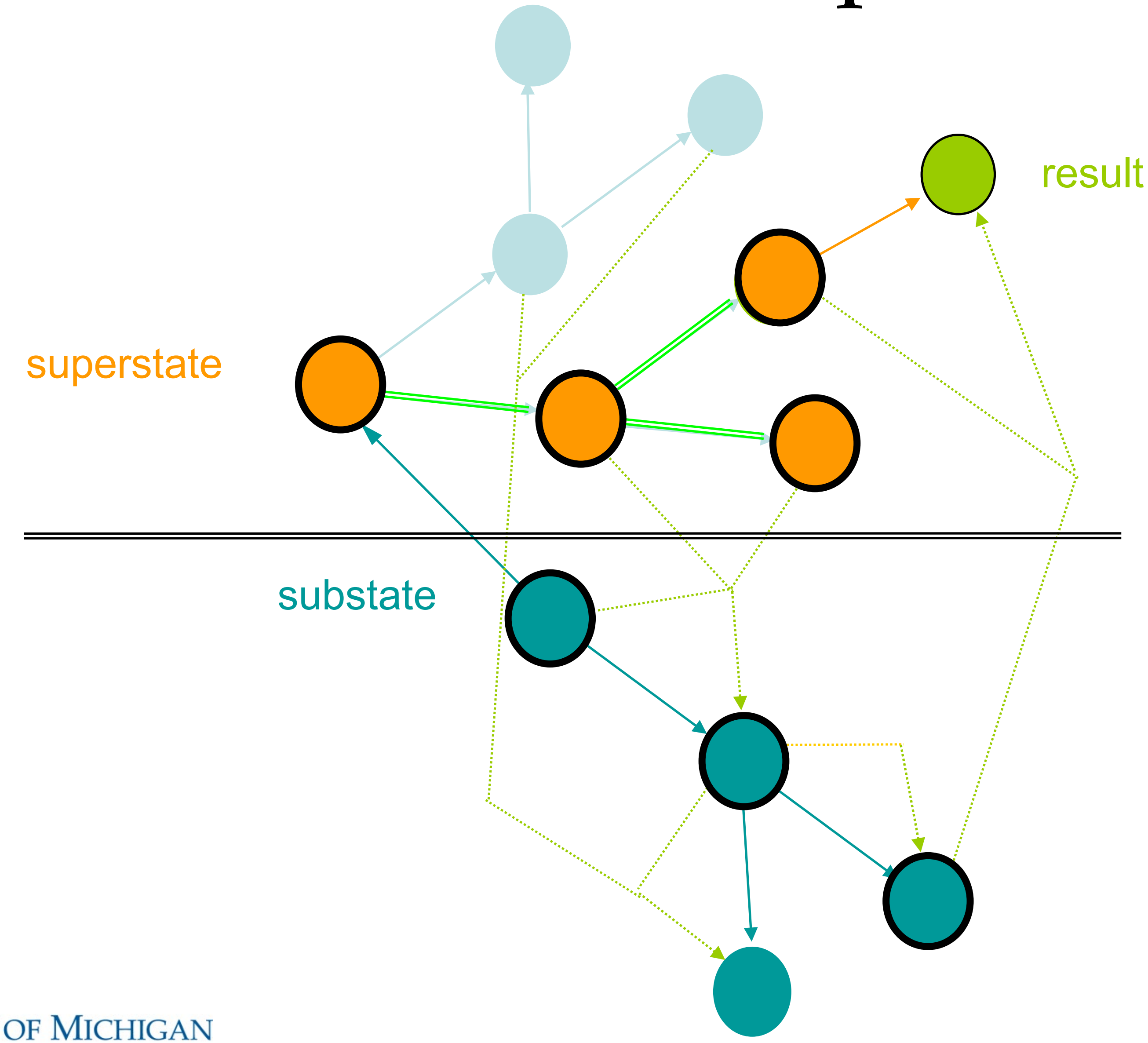
- Add two numbers by counting up and down.



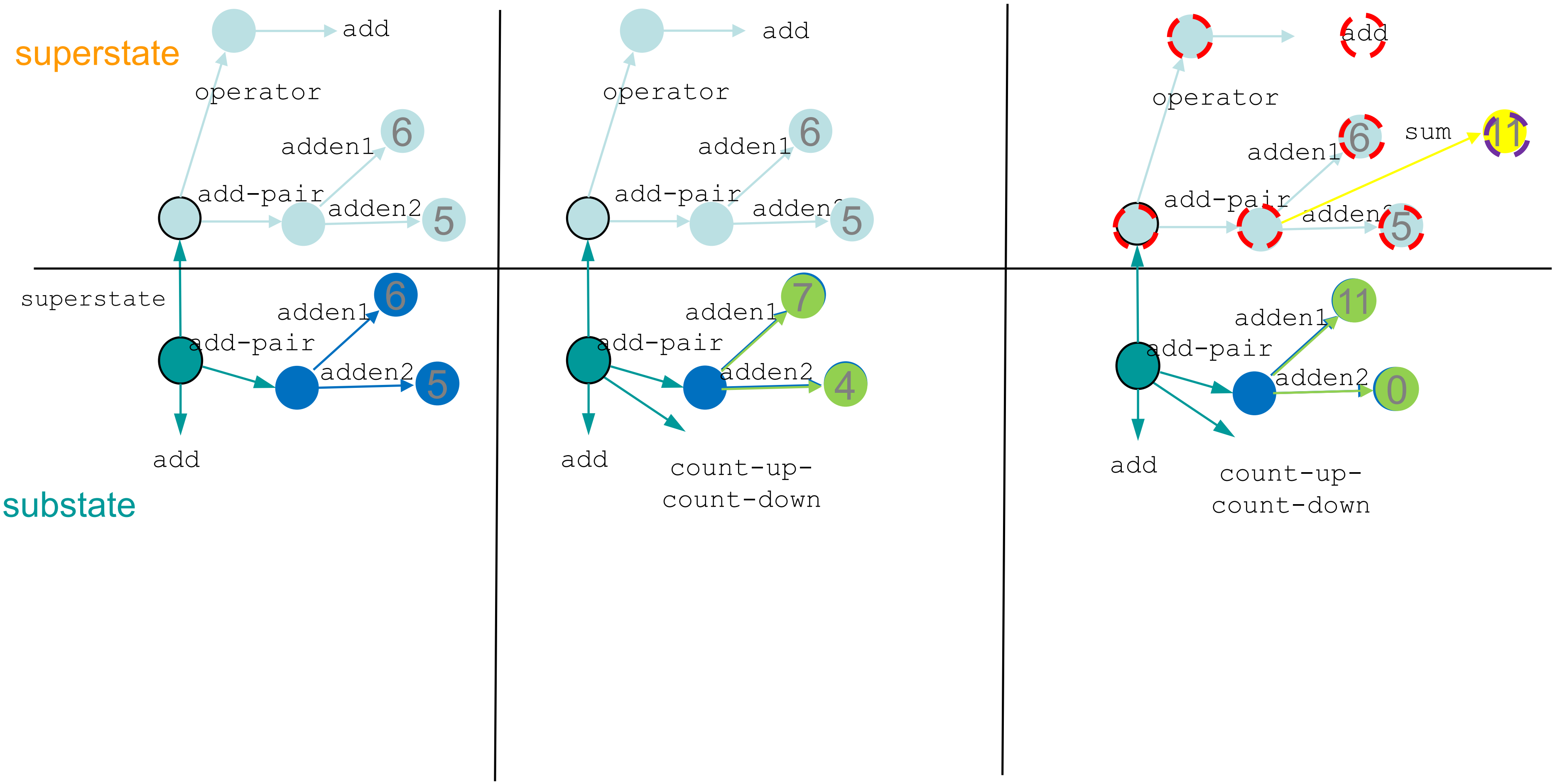
Abstract Chunk Example



Chunk Example

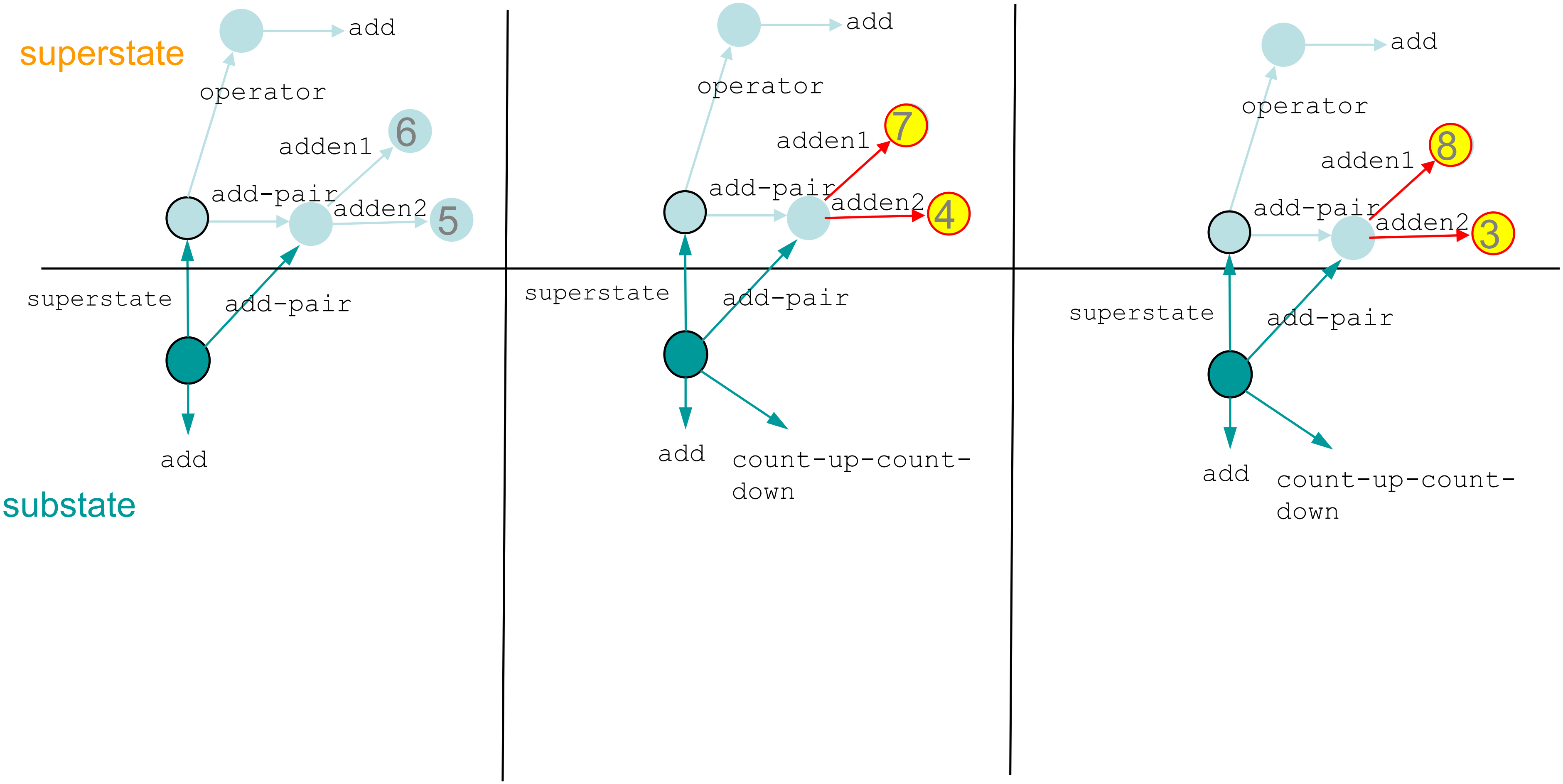


Problem Solving in Substate



UNIVERSITY OF MICHIGAN

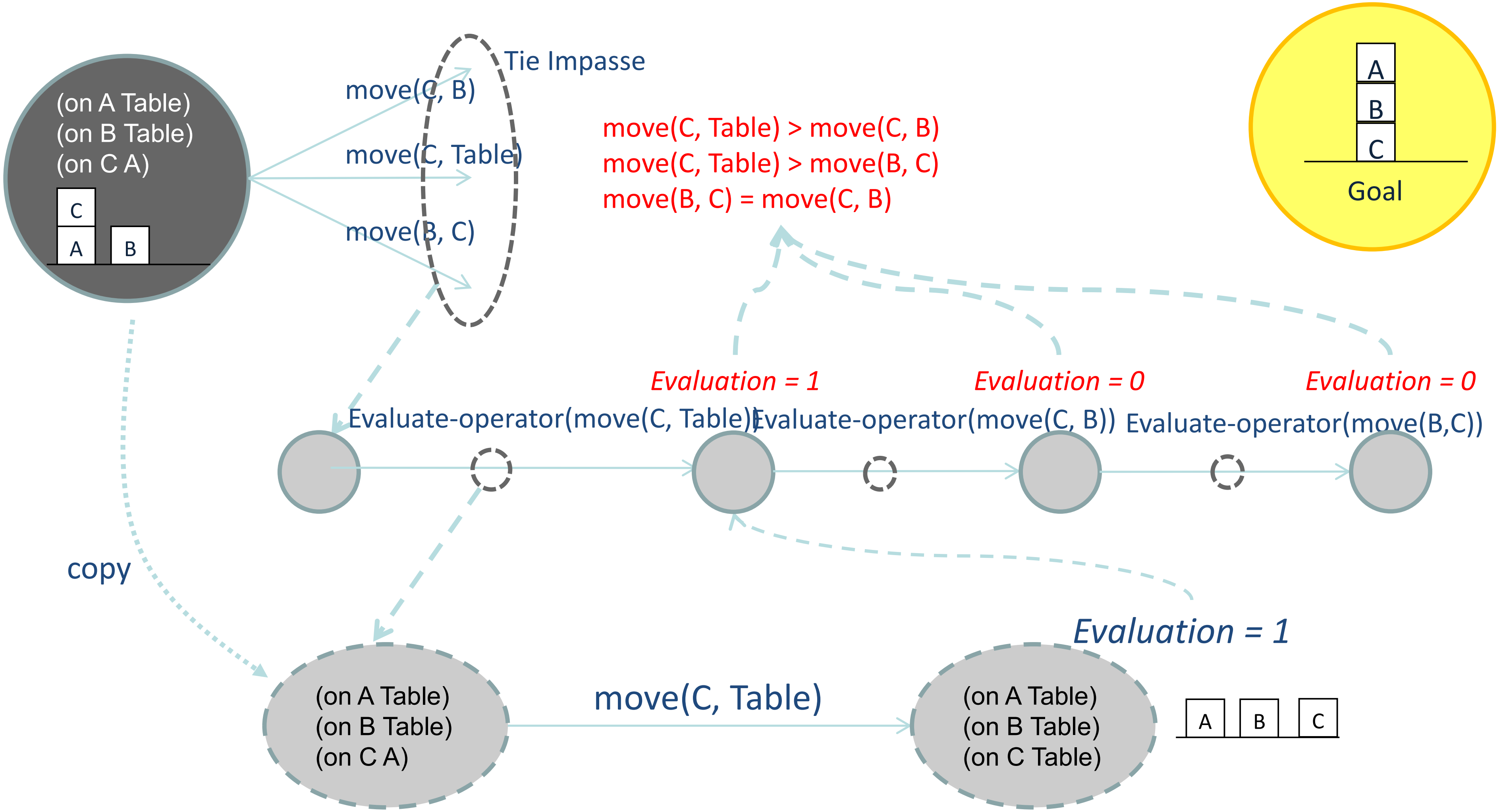
Problem Solving in Substate



Detail of Substates

- Does not fire all rules in parallel when there are multiple states.
- Fire rules in waves from top-state on down.
- Implication:
    - Chunks fire before processing in substates!

One-step Look-ahead: Learns selection rules



(on A Table)
(on B Table)
(on C A)

C
A B

move(C, B)
move(C, Table)
move(B, C)

Tie Impasse

move(C, Table) > move(C, B)
move(C, Table) > move(B, C)
move(B, C) = move(C, B)

A
B
C
Goal

Evaluation = 1
Evaluation = 0
Evaluation = 0

Evaluate-operator(move(C, Table)) Evaluate-operator(move(C, B)) Evaluate-operator(move(B,C))

copy

Evaluation = 1

(on A Table)
(on B Table)
(on C A)

move(C, Table)

(on A Table)
(on B Table)
(on C Table)

A B C

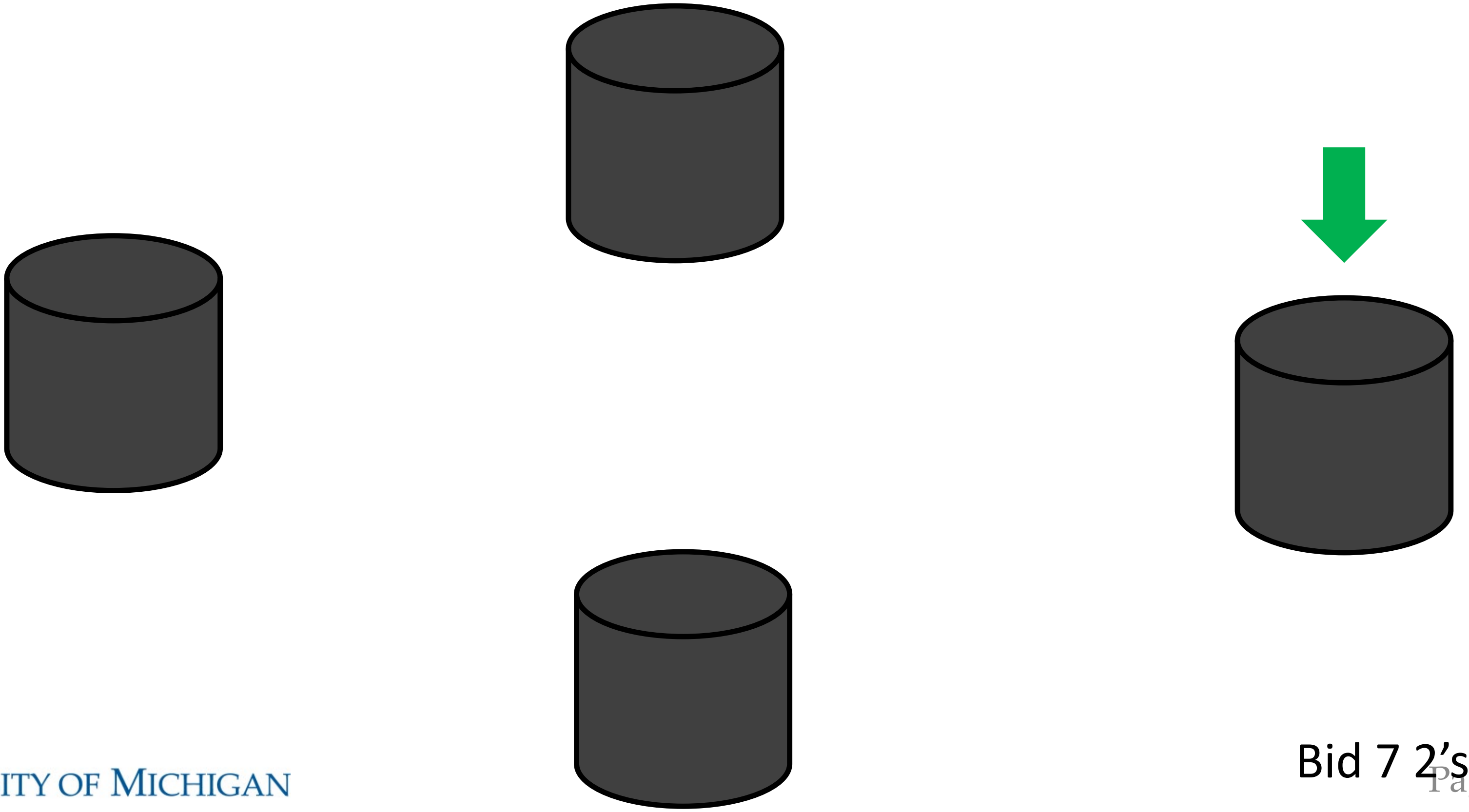
More details on Chunking

- Backtracing includes operator selection (preference) rules that were necessary to make the decision.
  - Can be disabled so only acceptable preferences included.
- Cannot backtrace through reason negation in substate doesn't match anything – source of overgeneralization.
- All identifiers are converted to variables.
  - Depends on how the variables were tested in the original rules.

[illegible] number of dice under cups.

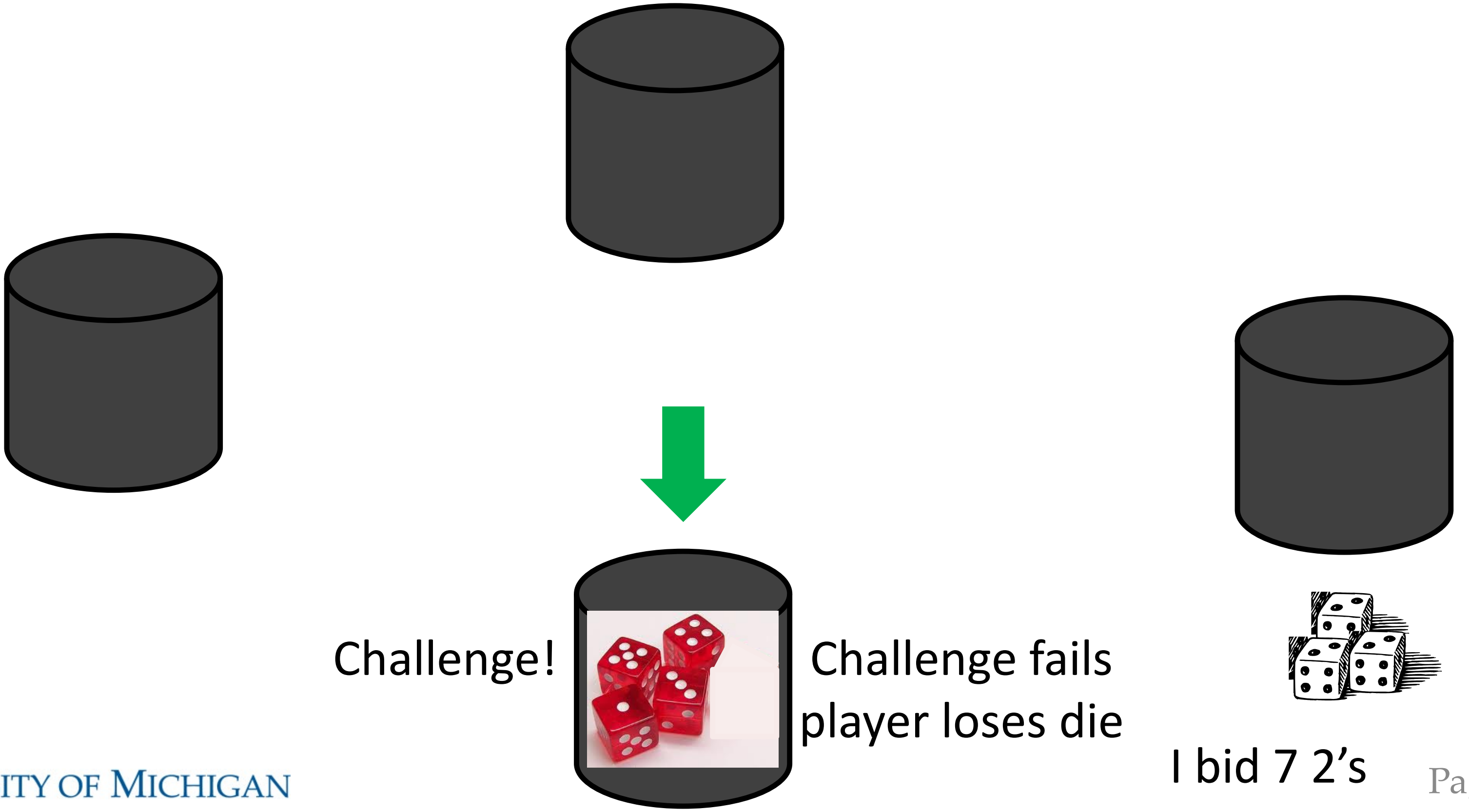
Bid 6 6’s
Bid 4 2’s

Players can "push" out a subset of their dice and reroll when bidding.



Bid 7 2's

Player can Challenge previous bid.
All dice are revealed

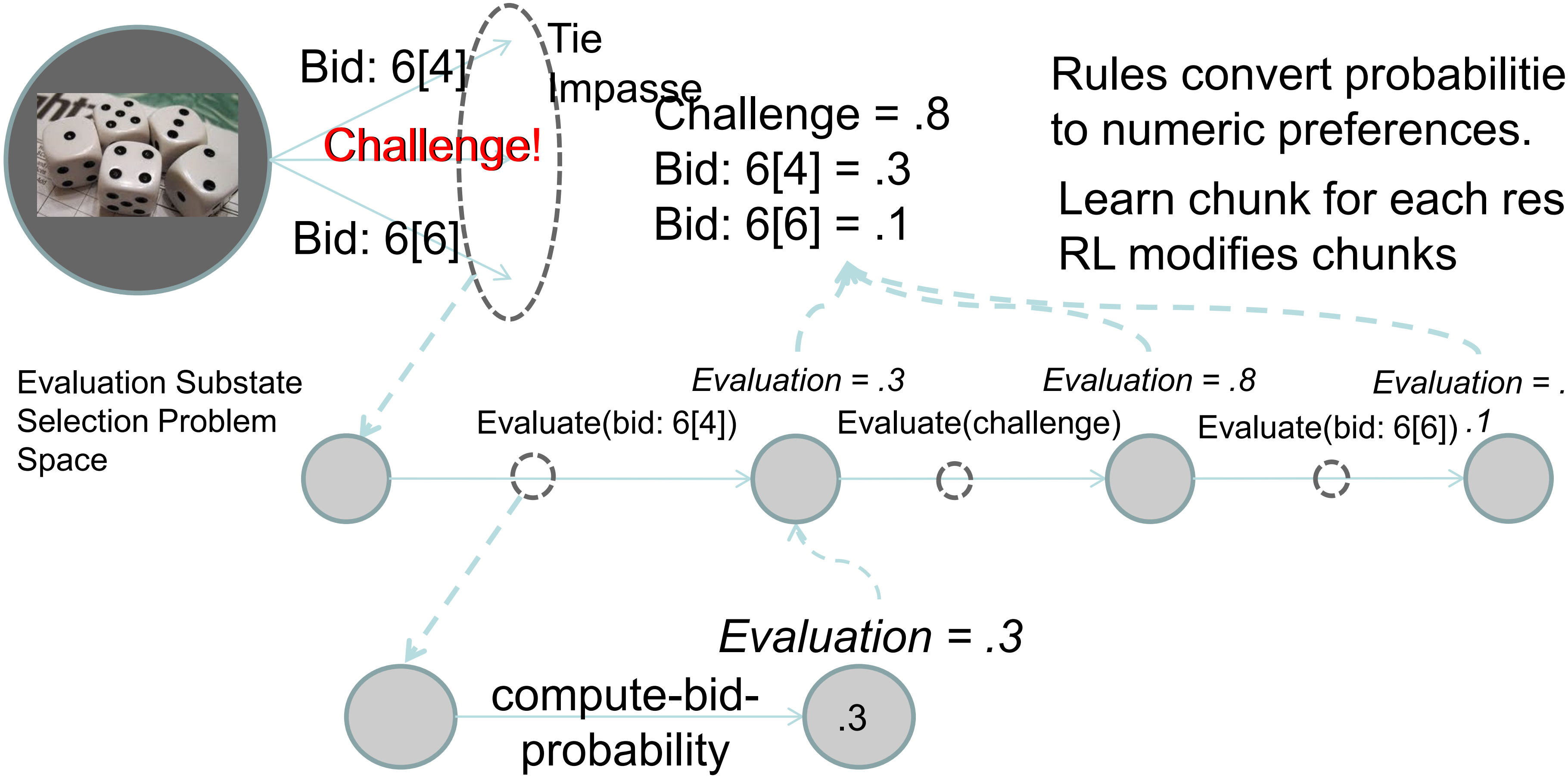


Challenge!

Challenge fails player loses die

I bid 7 2’s

Evaluation with Probability Calculation



Bid: 6[4]

Challenge!

Bid: 6[6]

Tie Impasse

Challenge = .8
Bid: 6[4] = .3
Bid: 6[6] = .1

Evaluation Substate
Selection Problem
Space

Evaluate(bid: 6[4])

Evaluation = .3

Evaluate(challenge)

Evaluation = .8

Evaluate(bid: 6[6]) *.1*

Evaluation = .1

Evaluation = .3

compute-bid-probability

.3

Rules convert probabilities to numeric preferences.

Learn chunk for each result.
RL modifies chunks

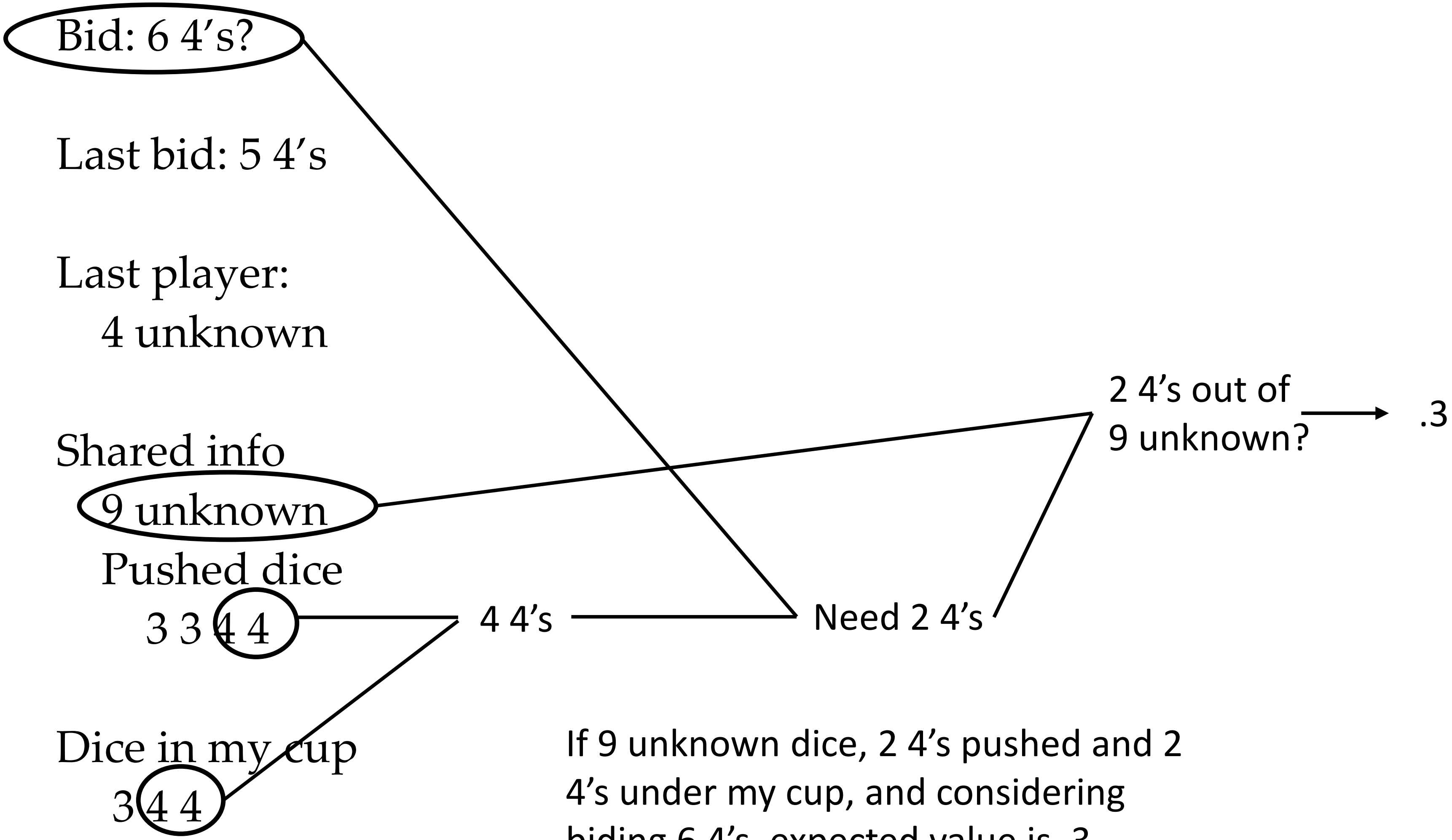
Chunking over Substate Processing

- For each preference created in the substate, chunking creates a new rule
  - Actions are numeric or symbolic preferences
  - Conditions based on working memory elements tested in substate

- Reinforcement learning then tunes RL rules based on experience

Learning Selection-rules

Using only probability calculation



Bid: 6 4's?

Last bid: 5 4's

Last player:
4 unknown

Shared info
9 unknown
Pushed dice
3 3 4 4

Dice in my cup
3 4 4

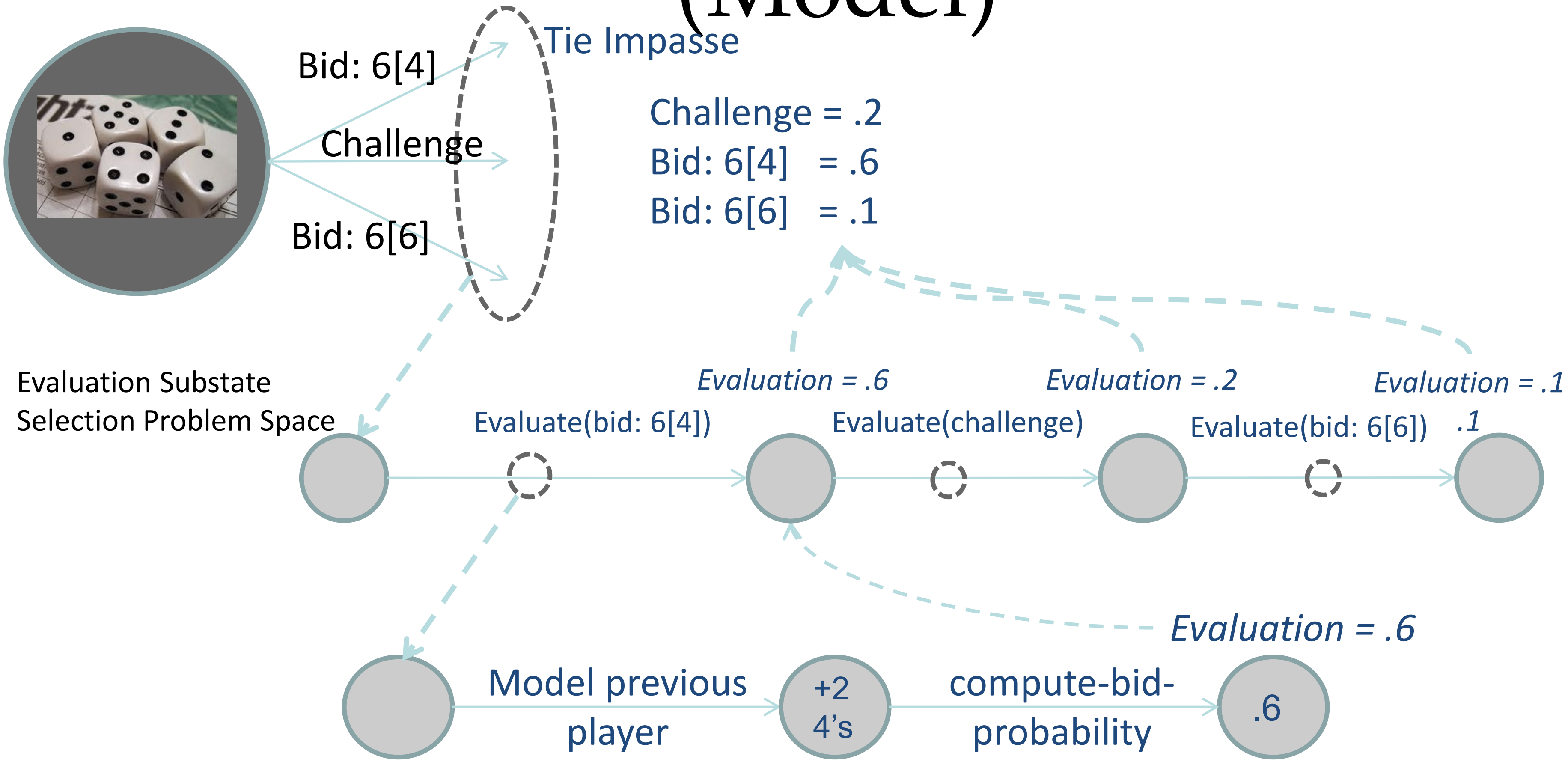
4 4's

Need 2 4's

2 4's out of
9 unknown? → .3

If 9 unknown dice, 2 4's pushed and 2 4's under my cup, and considering biding 6 4's, expected value is .3

Using Additional Background Knowledge (Model)



Bid: 6[4]

Challenge

Bid: 6[6]

Tie Impasse

Challenge = .2

Bid: 6[4] = .6

Bid: 6[6] = .1

Evaluation Substate

Selection Problem Space

Evaluation = .6

Evaluate(bid: 6[4])

Evaluation = .2

Evaluate(challenge)

Evaluation = .1

Evaluate(bid: 6[6])

.1

Evaluation = .6

Model previous player

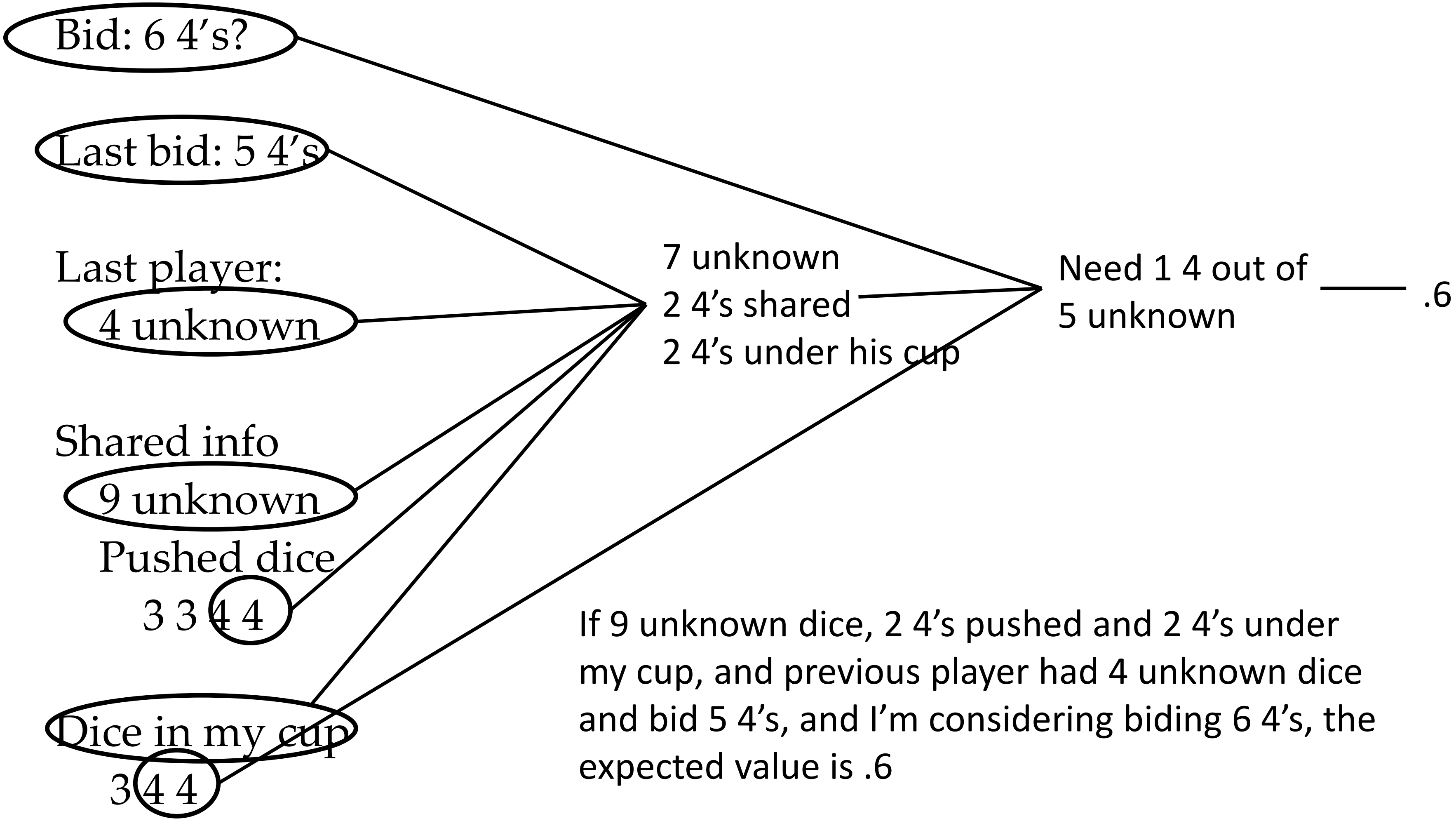
+2 4’s

compute-bid-probability

.6

Learning Selection Rules

Using probability and model



Bid: 6 4's?

Last bid: 5 4's

Last player:
4 unknown

Shared info
9 unknown

Pushed dice
3 3 4 4

Dice in my cup
3 4 4

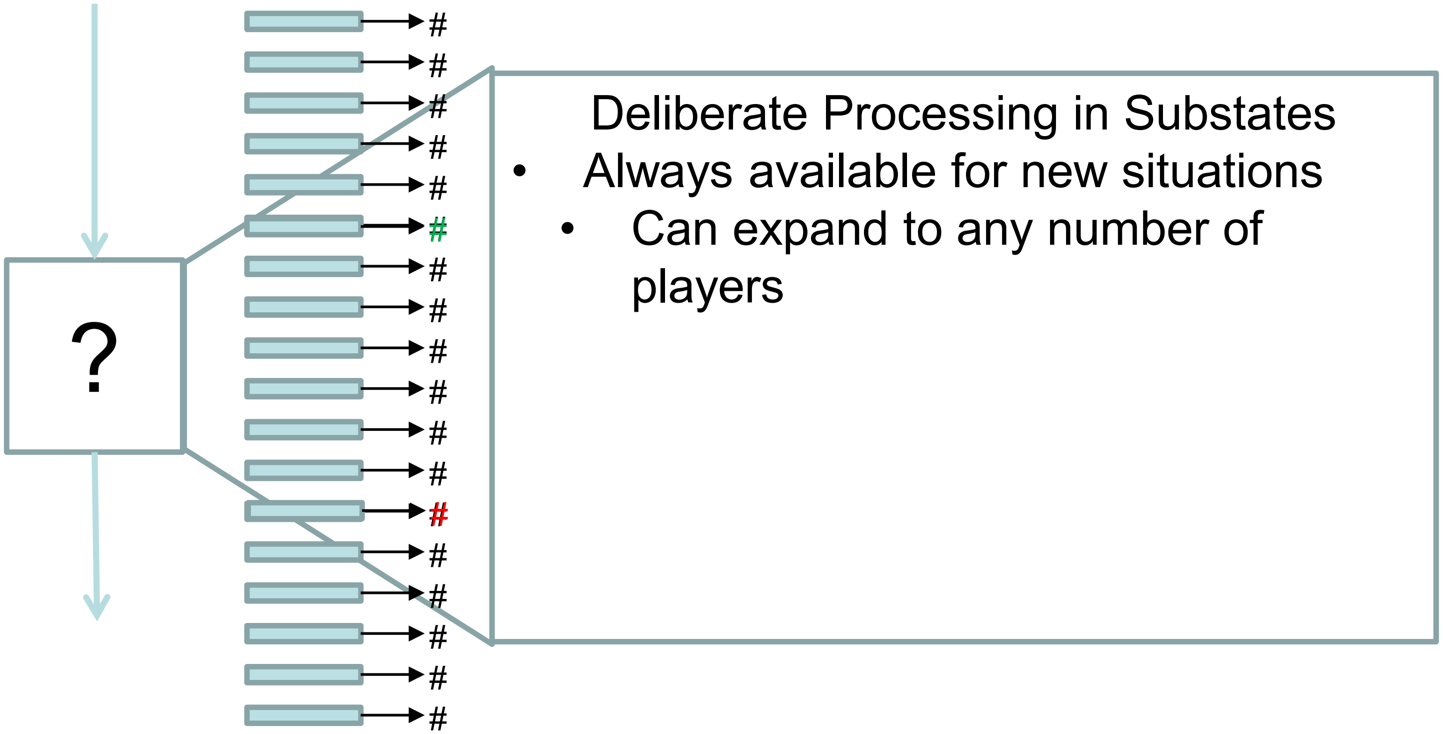
7 unknown
2 4's shared
2 4's under his cup

Need 1 4 out of
5 unknown

.6

If 9 unknown dice, 2 4's pushed and 2 4's under my cup, and previous player had 4 unknown dice and bid 5 4's, and I'm considering biding 6 4's, the expected value is .6

Two-Stage Learning



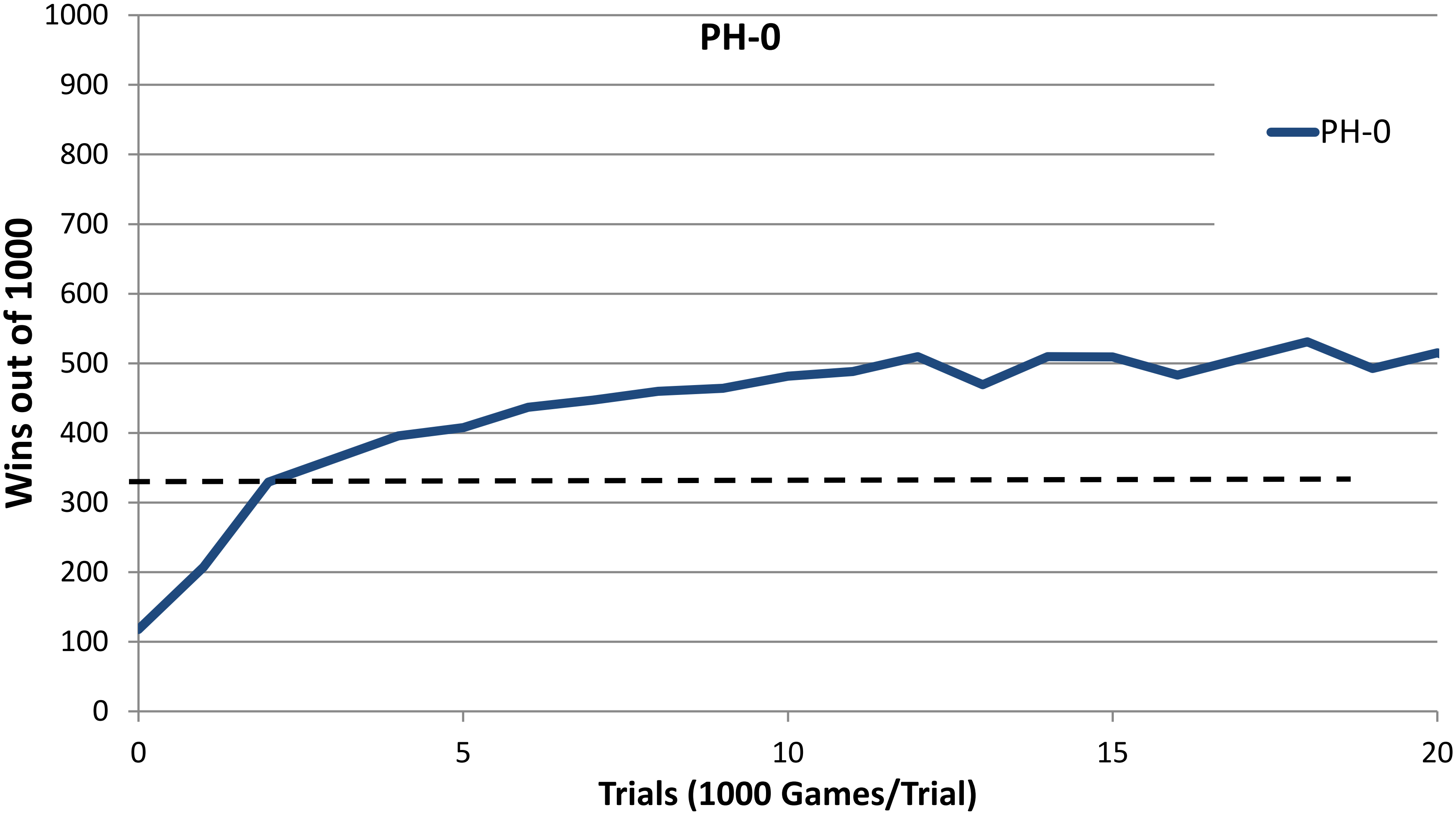
Select action

Deliberate Processing in Substates

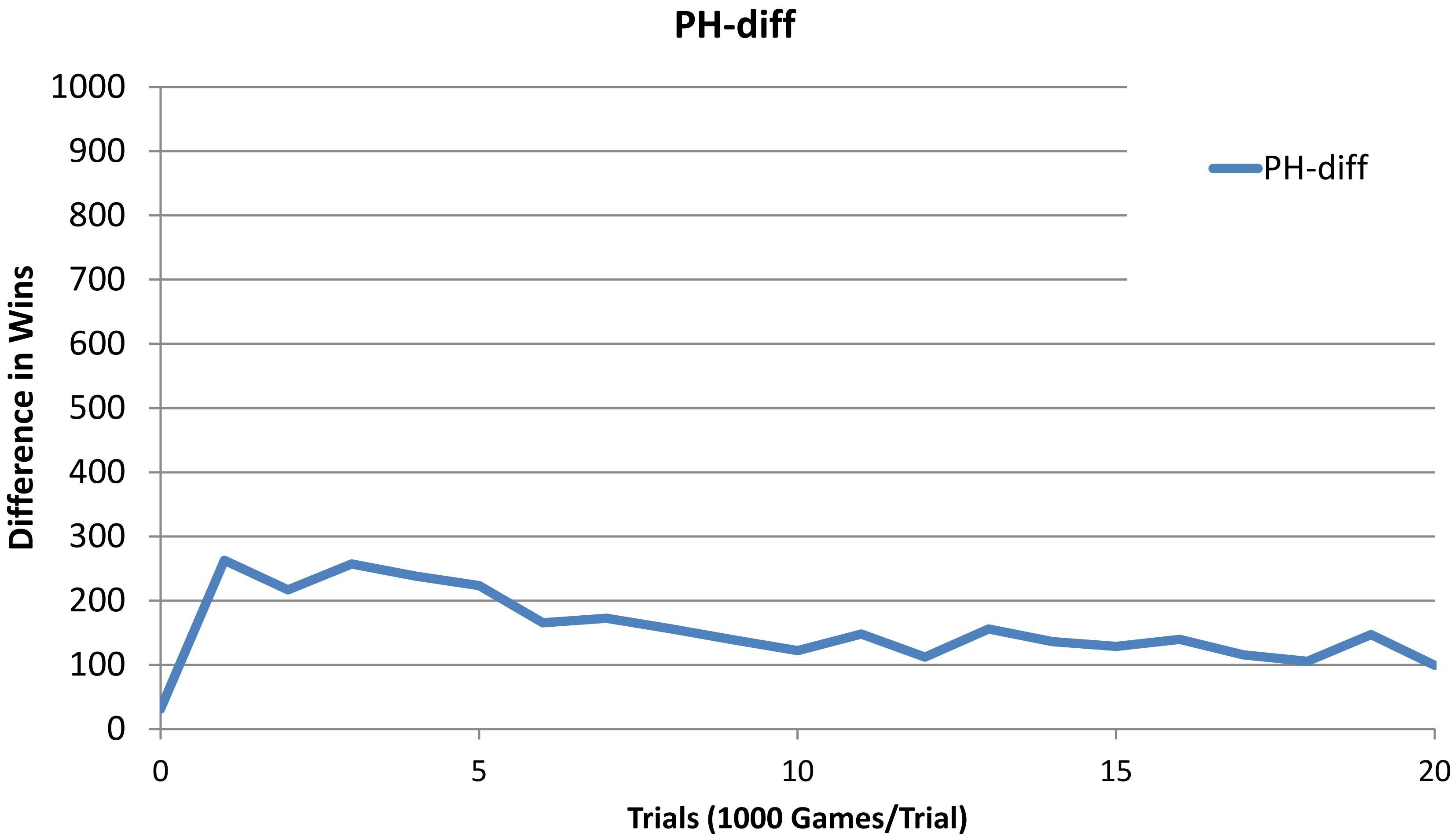
- Always available for new situations
  - Can expand to any number of players

RL rules updated based on agent's experience

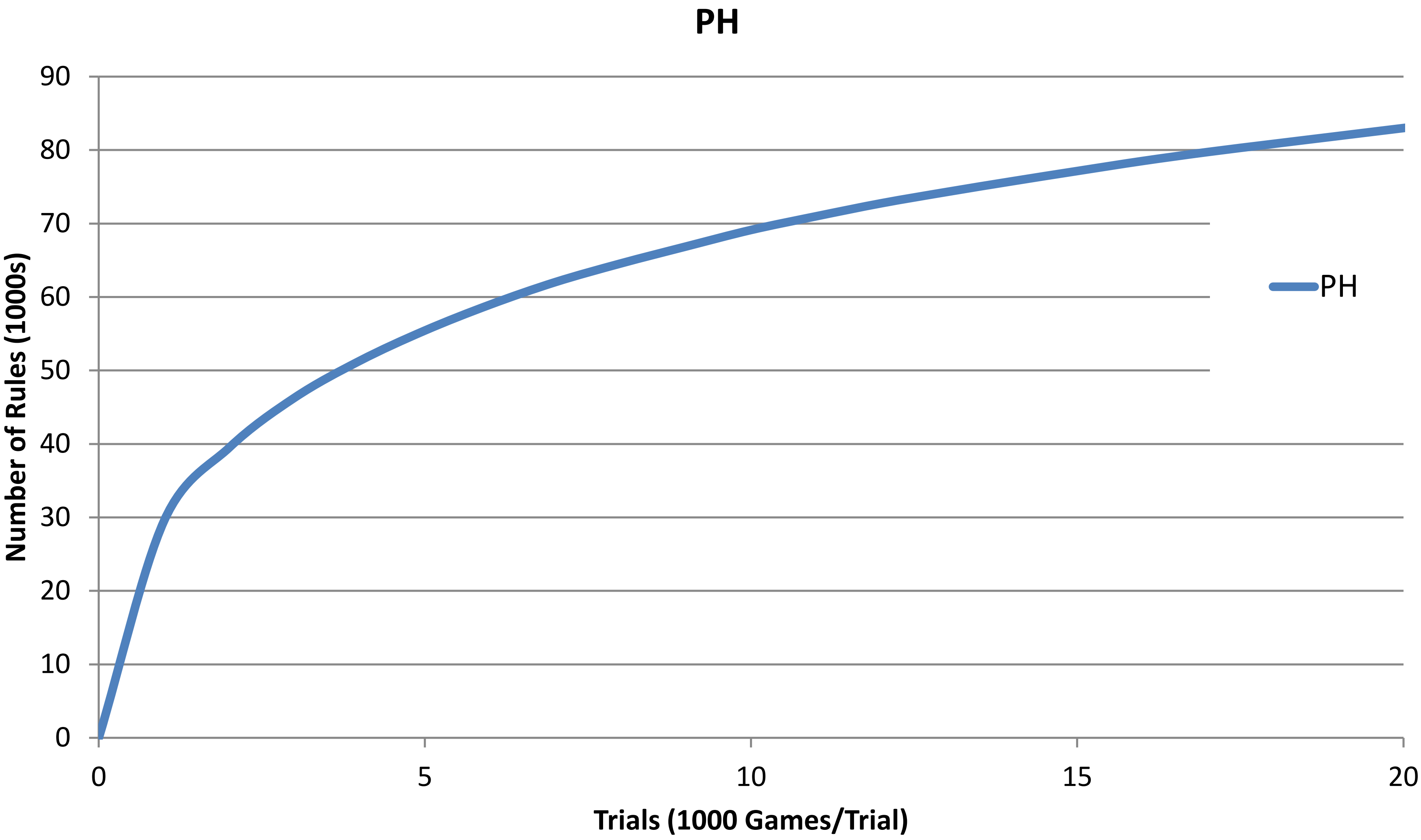
Learning Agent with Initial Values = 0



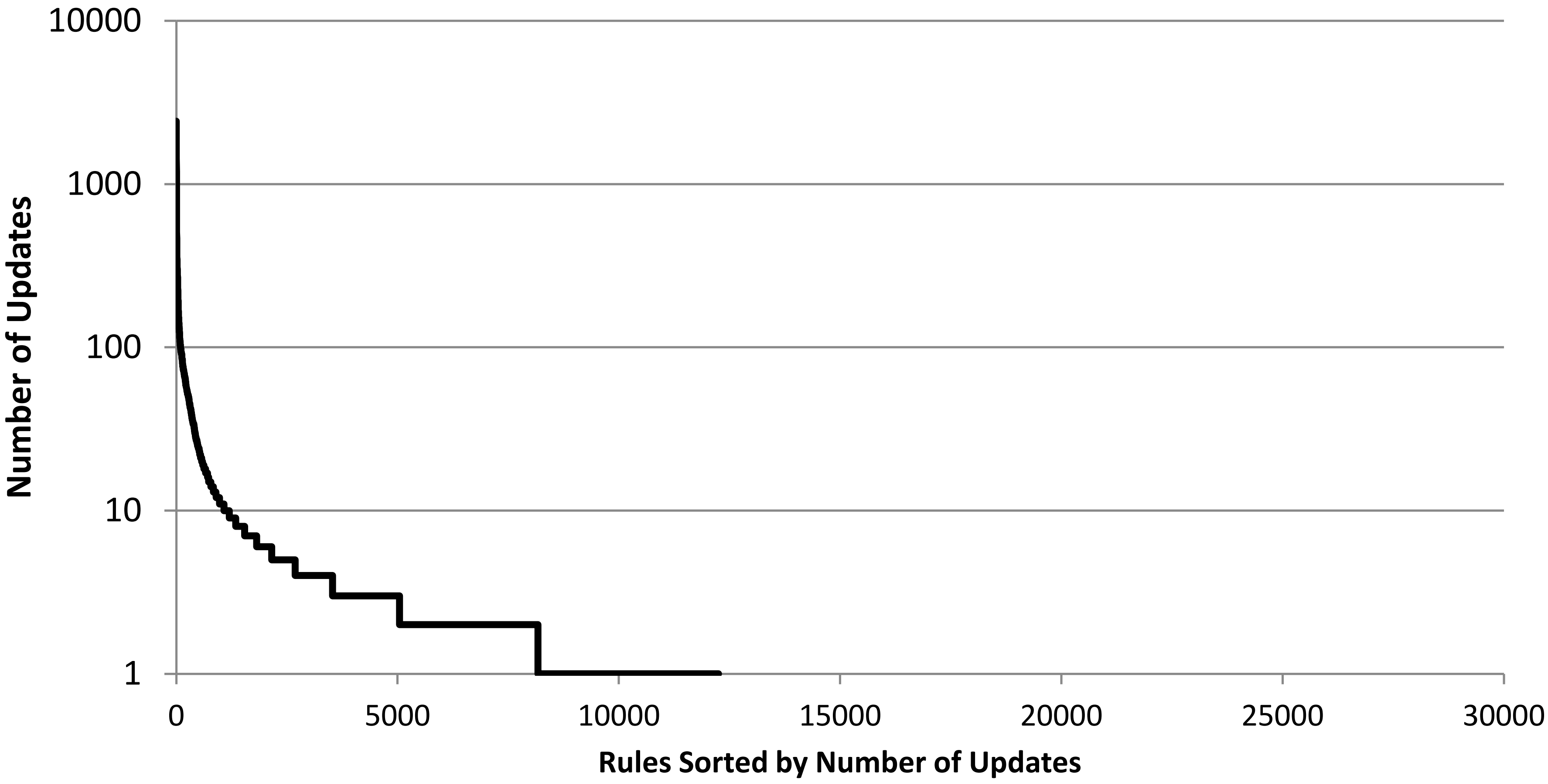
Difference Between Initialized and Uninitialized



Number of Rules Learned



Rules Sort by Updates



Chunking Analysis

- Converts deliberate reasoning/planning to reaction
- Generality of learning based on generality of reasoning
  - Leads to many different types learning
  - If reasoning is inductive, so is learning
- Soar only learns what it thinks about
- All learning is impasse driven
  - Learning arises from a lack of knowledge