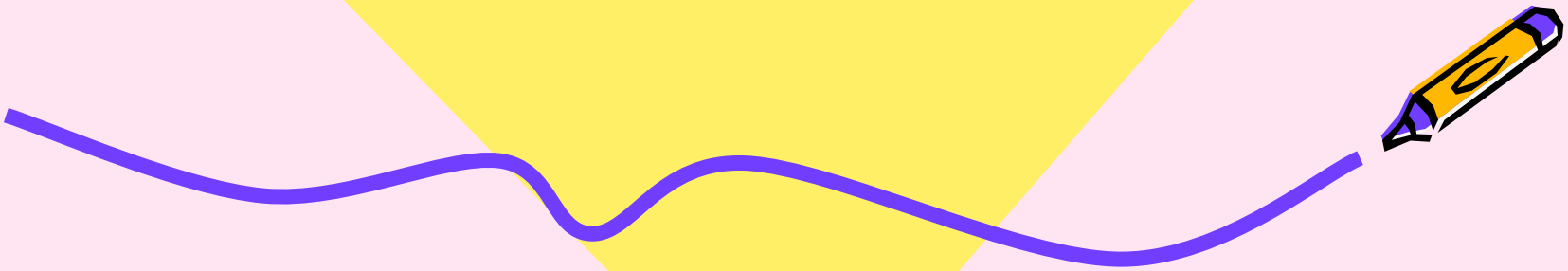
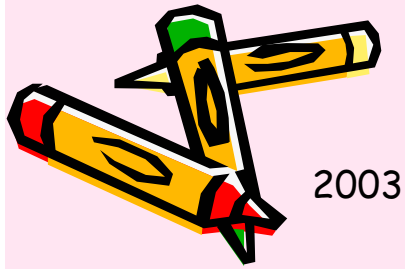
Learning Action Strategies For Multi-agent Planning Domains By Reinforcement Learning

Esin SAKA

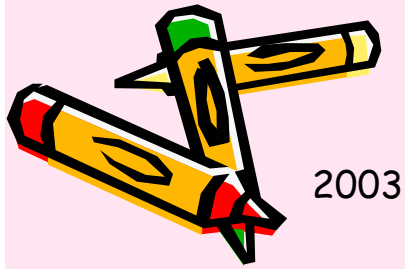


Outline



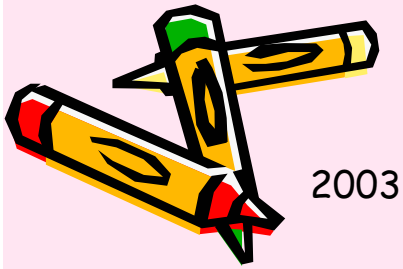
Planning Learning

- Planner
  - Algorithm
  - Performs a search on possible actions
  - Finds a plan of action
- Strategy
  - Algorithm
  - Solves planning problems in a particular domain
- Aim:
  - Given any planning domain
  - Produce a strategy to solve problems in that domain



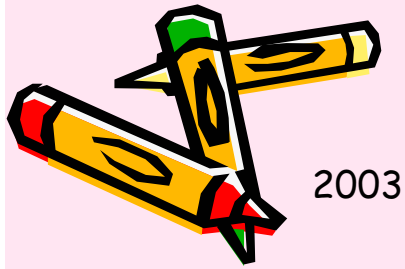
Input

- Description of domain
  - Names of predicates
  - Models of actions
  - Support predicates (optional)
- Problems in domain
- Evaluation function



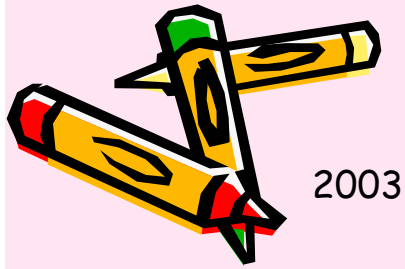
Output

- Decision list
  - Ordered list of existentially quantified rules



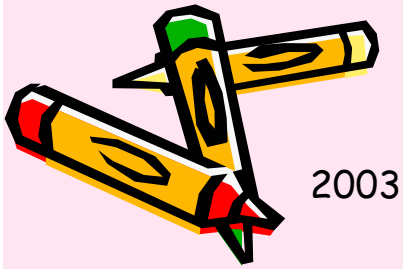
Example

- Problem:
  - There are
    - n blocks
    - 1 agent
  - Move all the blocks



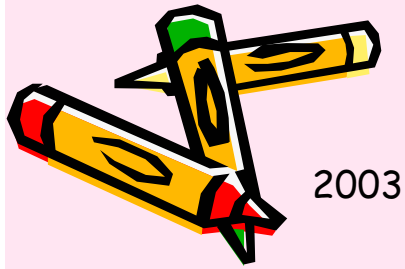
Example (cont.)

- Input
  - Predicates
    - arm_empty()
    - on(x1, x2)
    - ontable(x1)
    - clear(x1)
    - hold(x1)
  - Actions
    - arm_empty() & clear(x1) -> pop(x1) -> hold(x1)
      - (i.e: del ontable(x1) & del clear(x1) & del arm_empty() & add hold(x1)
    - hold(x1) -> drop(x1) -> arm_empty()



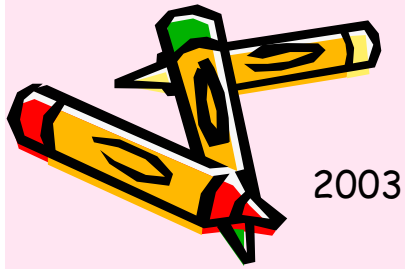
Example (cont.)

- Output
  - arm_empty() & clear(x1) -> pop(x1)
  - hold(x1) -> drop(x1)



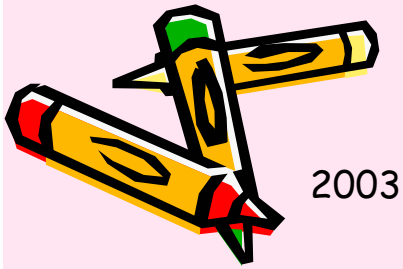
Example-2

- Problem:
  - There are
    - n blocks
    - 2 agents
  - Move all the blocks



Example (cont.)

- Input
  - Predicates
    - arm_empty(a1)
    - on(x1, x2)
    - ontable(x1)
    - clear(x1)
    - hold(a1, x1)
  - Actions
    - arm_empty(a1) & clear(x1) -> pop(a1, x1) -> hold(a1, x1)
      - (i.e: del ontable(x1) & del clear(x1) & del arm_empty(a1) & add hold(a1, x1)
    - hold(a1, x1) -> drop(a1, x1) -> arm_empty(a1)

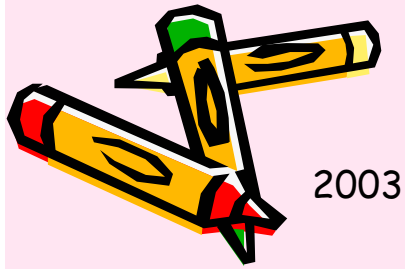


Example (cont.)

- Evaluation function
  - #blocks moved
  - #blocks - #blocks not moved -> (i.e # of blocks moved or holding)

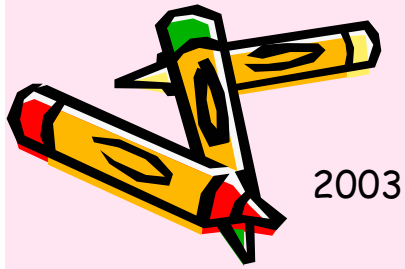
- Output
  - arm_empty(a1) & clear(x1) -> pop(a1, x1)
  - hold(a1, x1) -> drop(a1, x1)

  or
  - arm_empty(a1) & clear(x1) -> pop(a1, x1)
  - arm_empty(a1) & clear(x1) -> pop(a1, x1)
  - hold(a1, x1) -> drop(a1, x1)
  - hold(a1, x1) -> drop(a1, x1)



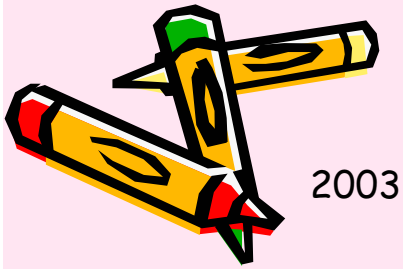
Algorithm

- Enumerate all rules under consideration
  - Enumerate all examples in data set
    - Enumerate all possible bindings
  - Initialize the decision list to empty list
  - While the data set is not empty
    - Choose the rule with best evaluation result
    - Add it to the end of decision list
    - Remove all examples that are covered by this rule from data set



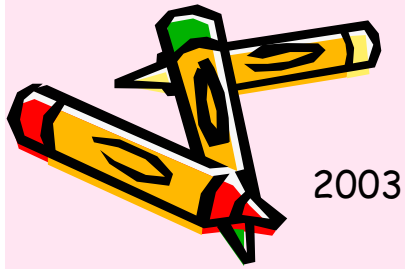
Conclusion & Future Study

- Works on simple problem
- Test on constrained blocks world domain
  - Multi-agent version of classical blocks world domain
- Evaluation function is important
  - Try different RL's like Q-learning or QACE.



Bibliography

- R. Khardon, Learning Action Strategies for planning domains, Artificial Intelligence 113: 125-148, 1999
- R. Khardon, L2Act User Manual, 1997
- F. Polat, O. Abul, Learning Sequences of Compatible Actions Among Agents, Artificial Intelligence17:21-37, 2002
- R. Rivest, Learning Decision Lists, Machine Learning 2(3): 229-246, 1987.



Thank you for your attention

☺