Task planning
&
Agent architectures

Philippe Morignot

Outline

- **Task planning**
  - Statement
  - Difficulty
  - An example
  - Some task planners
  - History & conclusion & references
- **Robotics-oriented agent architectures**
  - Statement
  - Difficulty
  - Some architectures
  - Conclusion & References
- **Conclusion**

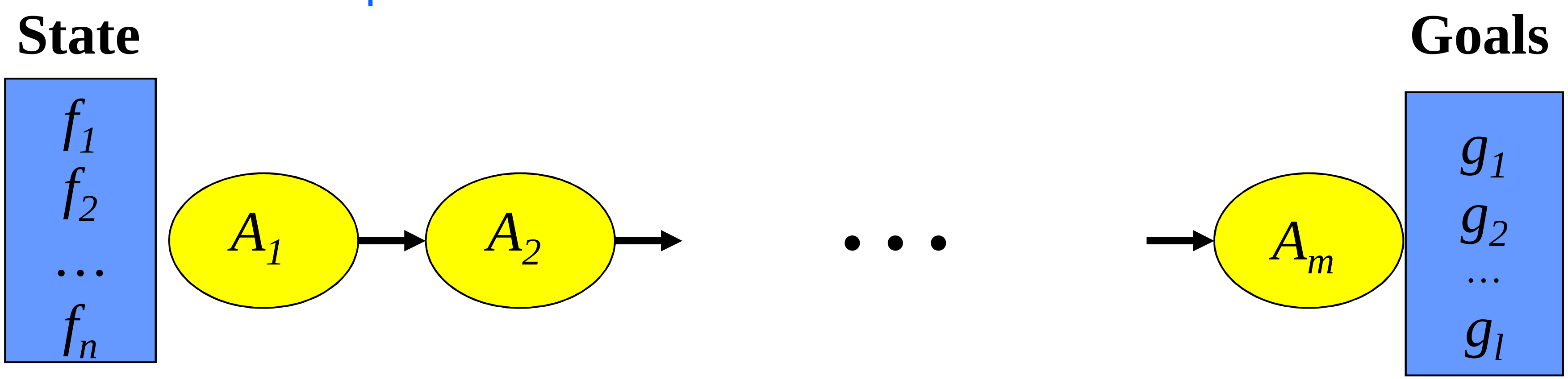
Part I ---

Task planning

Statement

« Given generic operators,
a state and goals,
find a sequence of instantiated operators,
which lead the initial state to a state
which includes the goals. »

- Solution-plan :



- « Task/action planning » / « plan synthesis » / « action plan generation » : activity of building a plan.
- « Planner » : computer software which solve this problem.

The problem

- The crane domain:
  - ✓ 1 crane, *a* locations, *b* trucks, *c* stacks, *d* containers.

- If $a = 5$, $b = 3$, $c = 3$, $d = 100$, then ~ 10^{277} states.
- Classical planning is in NP.

You cannot enumerate all states!

Assumptions

- A 1 : *The agent is the sole cause of change*.
  - No other agent, artificial or human.
- A 2 : The environment is totally observable, the agent has perfect knowledge of the environment.
  - The agent cannot reason (e.g., plan) on things he does not know.
- A 3 : *The environment is static.*
  - The environment does not change spontaneously.

Planning Domain Definition Language (PDDL)

- Representation language to define:
  - a domain : operators
  - a problem : initial state and goals.
- An operator is composed of :
  - Pre-condition : term which must hold for the action to be executable.
  - Effect / post-condition : term the truth value of which is changed by the action, when compared to the incoming situation.

- A term might be sometimes true, sometimes false, depending on the time at which it is considered.
  - Logical operator « *not* »
    - Example : (not (ON MOUSE PAD))
  - *« Fluent » (term)*
    - Example : (ON MOUSE PAD)

Definitions

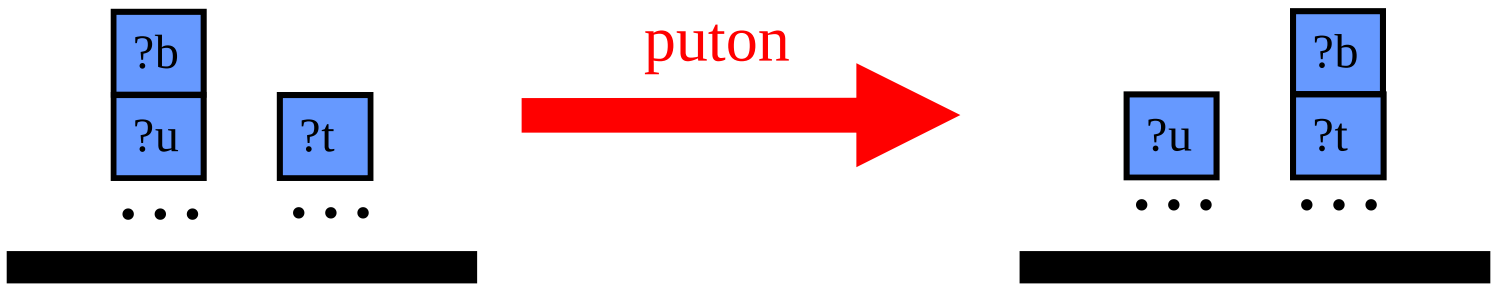
- **State**: a symbolic description of the agent / environment.
  - ✓ Changes over time, because of the agent actions.
- **Goal** : term which has to be satisfied. (= pre-condition).
  - ✓ Might be conjunctive, e.g. : Rich && Handsome && Famous.
- **Sub-goal** : goal obtained by regressing another goal. (= pre-condition).
- **Support** : post-conditions which unify with a given posterior pre-condition.
- **Causal link** : relation between a post-condition and a posterior pre-condition, which is satisfied by this post-condition.
  - ✓ E.g. : operator WIN-LOTO et goal (OWN $1,000,000,000)
- **Mutual exclusion** (*mutex*) : 2 pre- or post-conditions which conflict together.
  - ✓ E. g. : pre-condition (ON MOUSE PAD) and the post-condition (not (ON MOUSE PAD)) in parallel.

Example of PDDL operator: the blocks world

- Operator :

```
(:action puton
 :parameters (?b ?u ?t - block)
 :precondition (and (clear ?b) (on ?b ?u))
                    (clear ?t))
 :effect (and (not (on ?b ?u)) (clear ?u)
              (on ?b ?t) (not (clear ?t))))
```

puton ?b ?u ?t	
(clear ?b) (on ?b ?u) (clear ?t)	(not (on ?b ?u)) (clear ?u) (on ?b ?t) (not (clear ?t))



- What about the table? And the arm? What if several arms? What about colored blocks? Or with a notch? Or of different sizes?
- Conditionals ? Universal quantification?

Example : a planning problem in PDDL

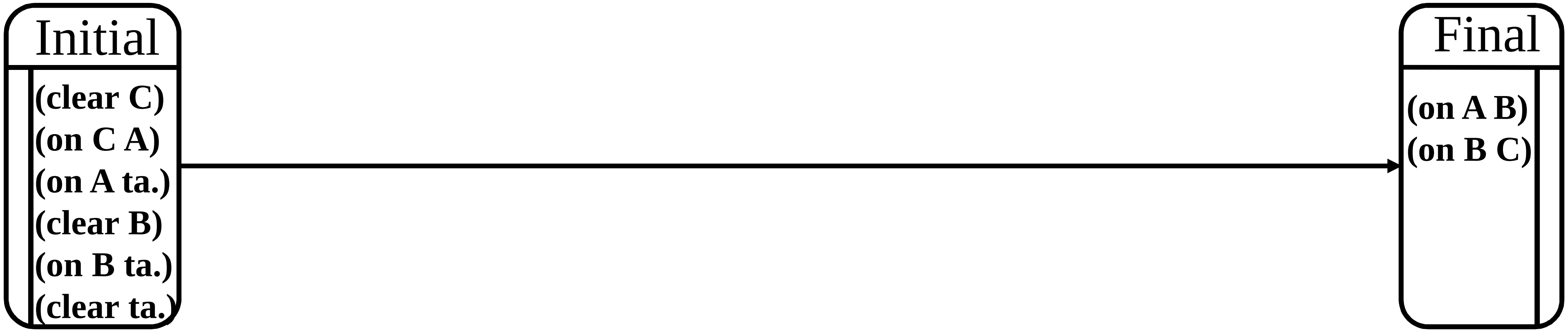
```
(define (problem blocks-24-1)
    (:domain blocks)
    (:objects X W V U T S R Q P O N M L K J I H G F E D C A B)
    (:init
        (CLEAR K) (CLEAR I) (ONTABLE C) (ONTABLE O)
        (ON K F) (ON F T) (ON T B) (ON B G) (ON G R)
        (ON R M) (ON M E) (ON E J) (ON J V) (ON V N)
        (ON N U) (ON U H) (ON H C) (ON I A) (ON A P)
        (ON P Q) (ON Q D) (ON D W) (ON W X) (ON X S)
        (ON S L) (ON L O) (HANDEMPTY))
    (:goal (and
        (ON L C) (ON C P) (ON P Q) (ON Q M) (ON M B)
        (ON B G) (ON G F) (ON F K) (ON K E) (ON E R)
        (ON R A) (ON A W) (ON W T) (ON T N) (ON N J)
        (ON J U) (ON U S) (ON S D) (ON D H) (ON H V)
        (ON V O) (ON O I) (ON I X))))
```

The anomaly of Gerald Jay Sussman (1/16)

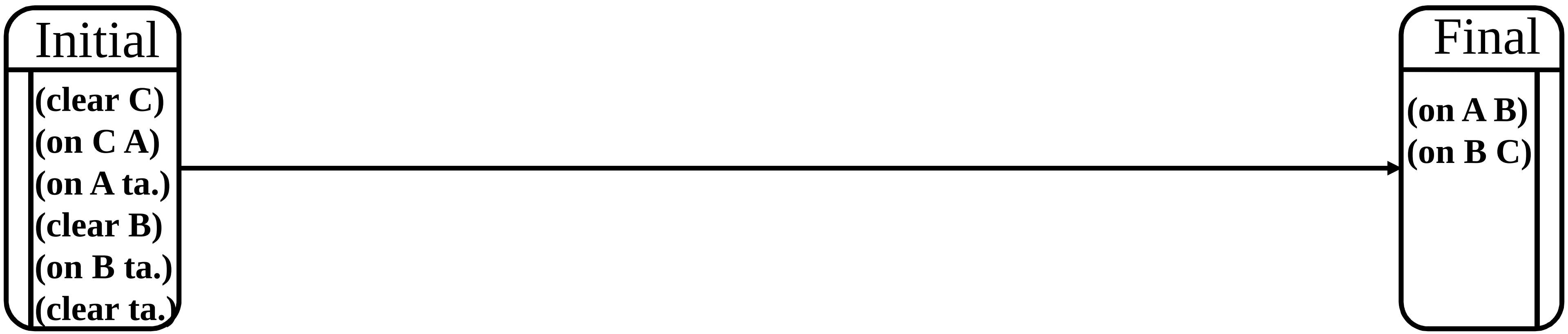
with :

puton ?b ?u ?t	
(clear ?b) (on ?b ?u) (clear ?t)	(not (on ?b ?u)) (=> (<> ?u table) (clear ?u)) (on ?b ?t) (=> (<> ?t table) (not (clear ?t)))

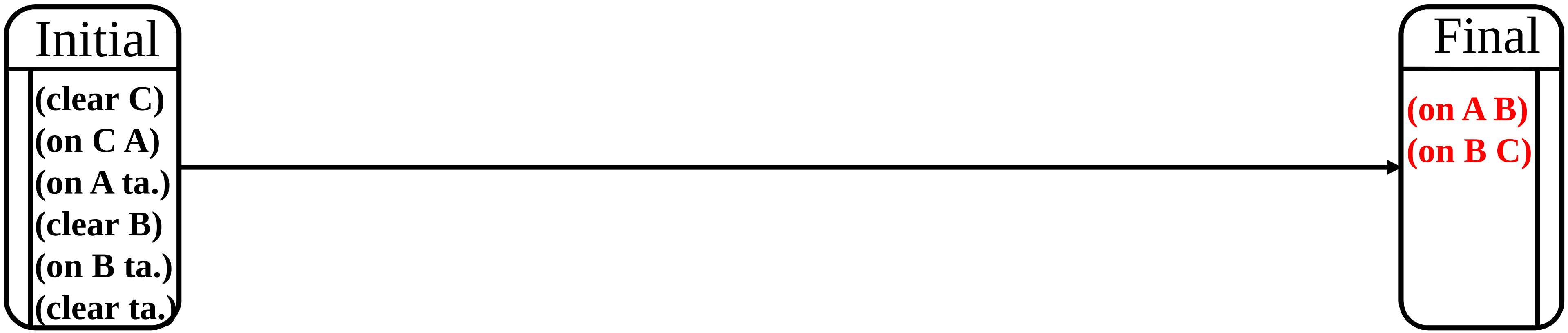
The anomaly of Gerald Jay Sussman (2/16)



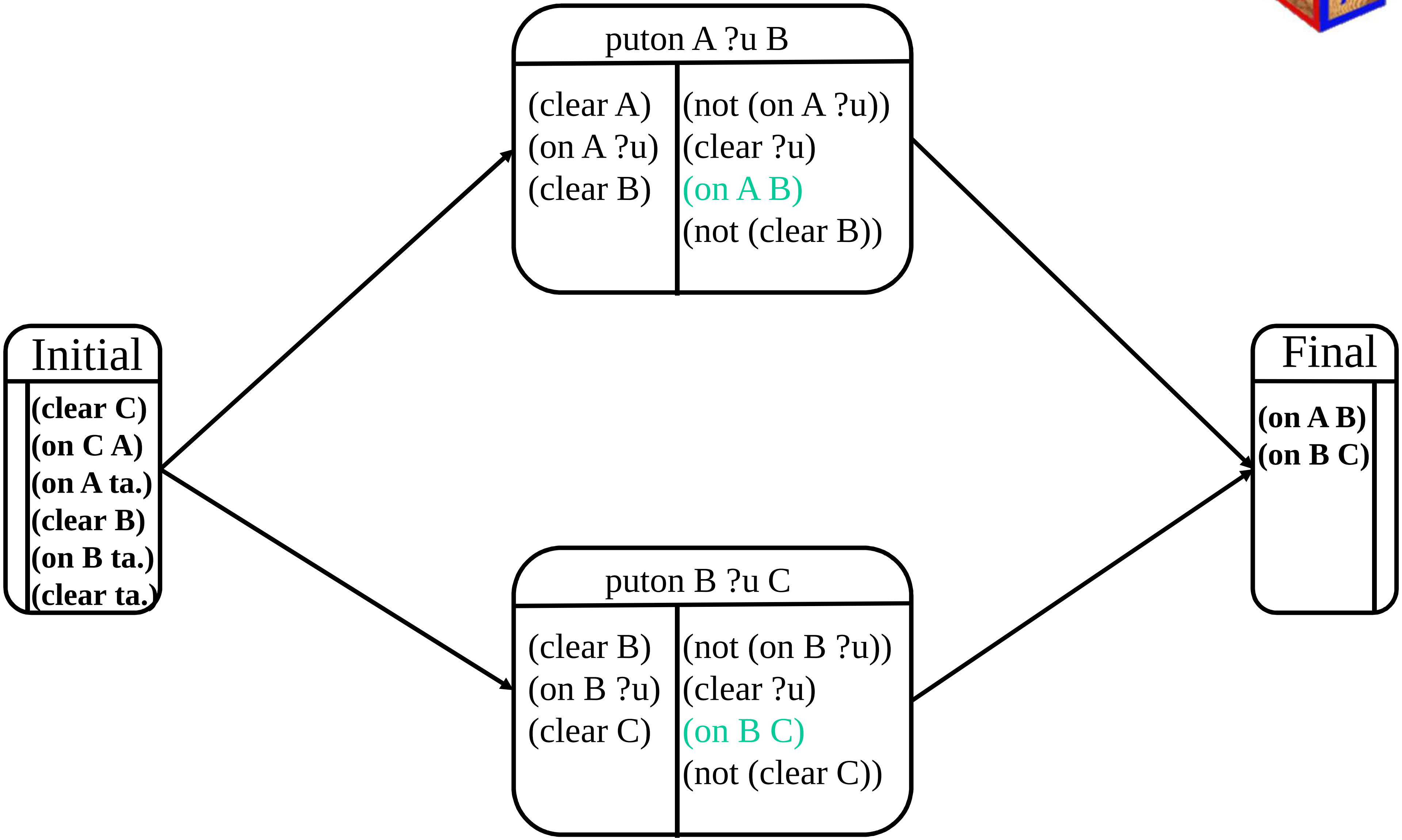
The anomaly of Gerald Jay Sussman (2/16)



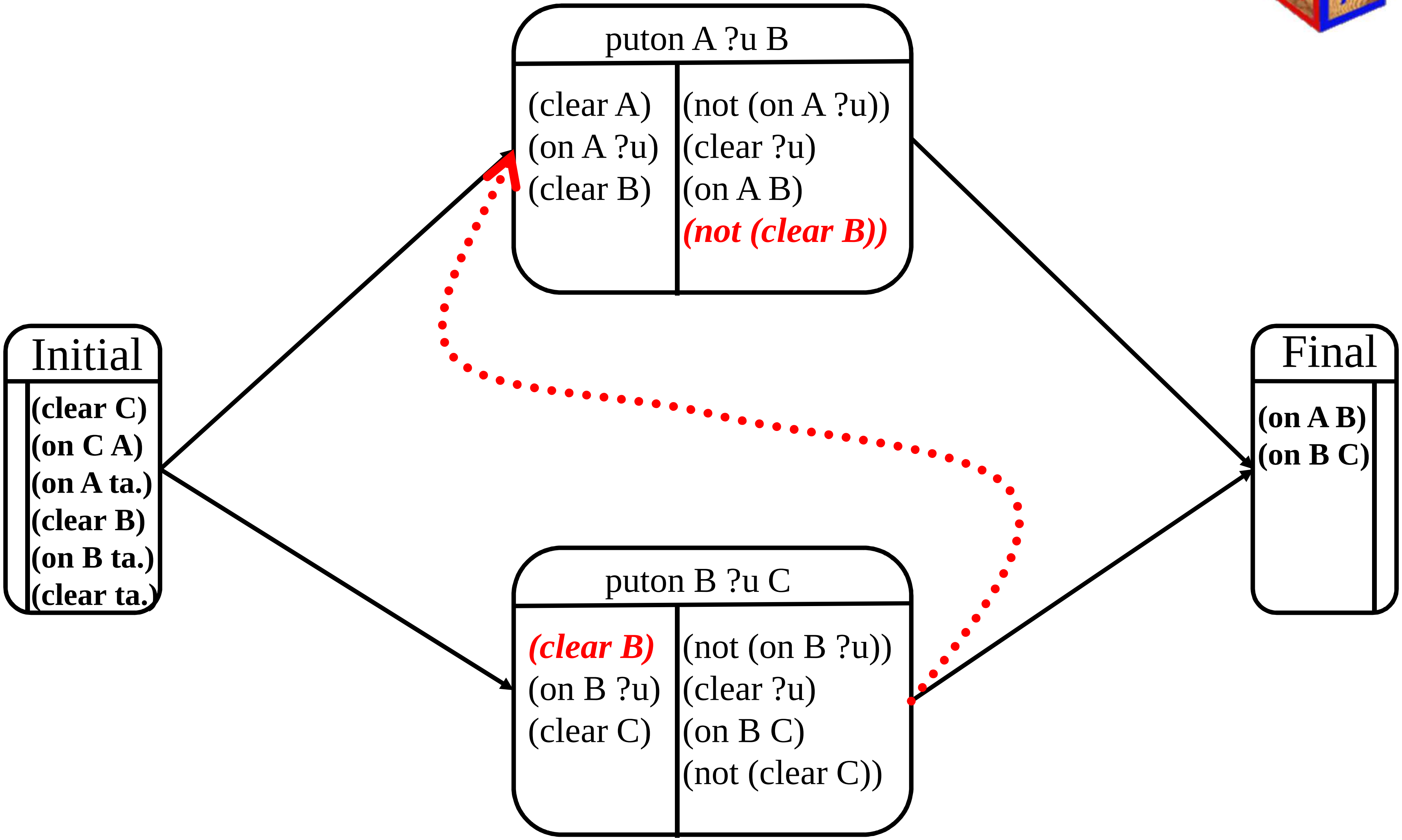
The anomaly of Gerald Jay Sussman (3/16)



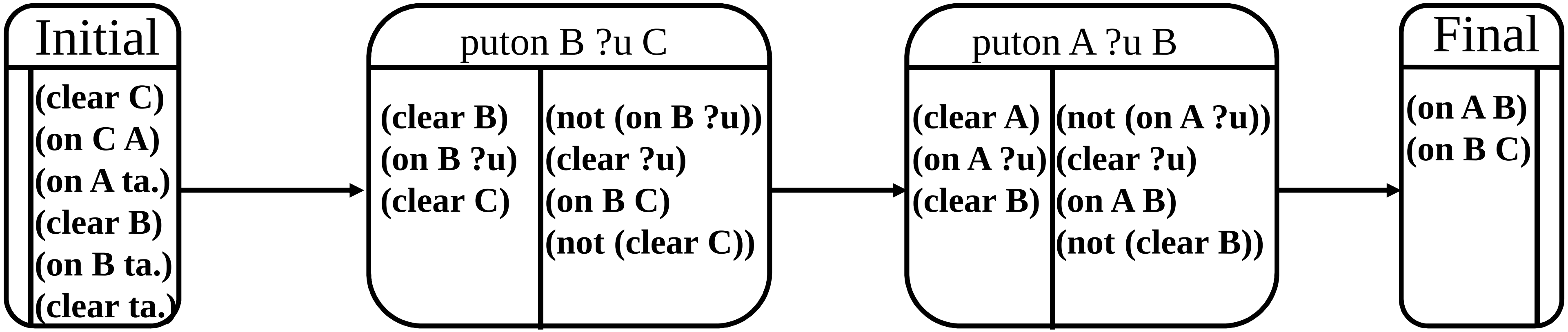
The anomaly of Gerald Jay Sussman (4/16)



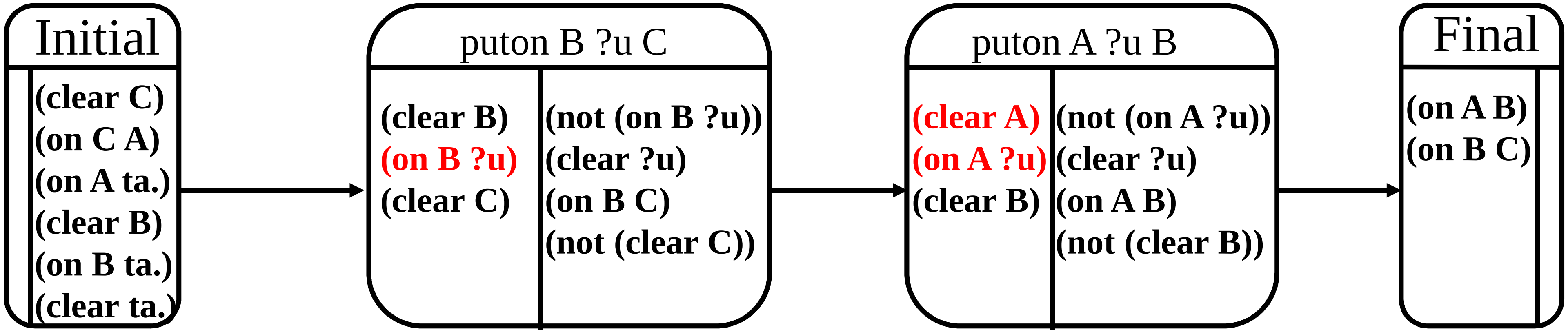
The anomaly of Gerald Jay Sussman (5/16)



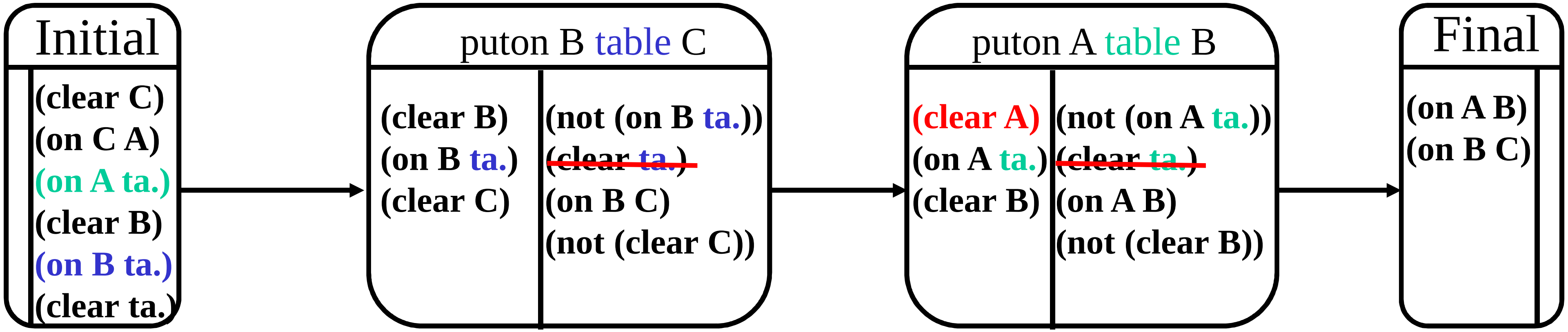
The anomaly of Gerald Jay Sussman (6/16)



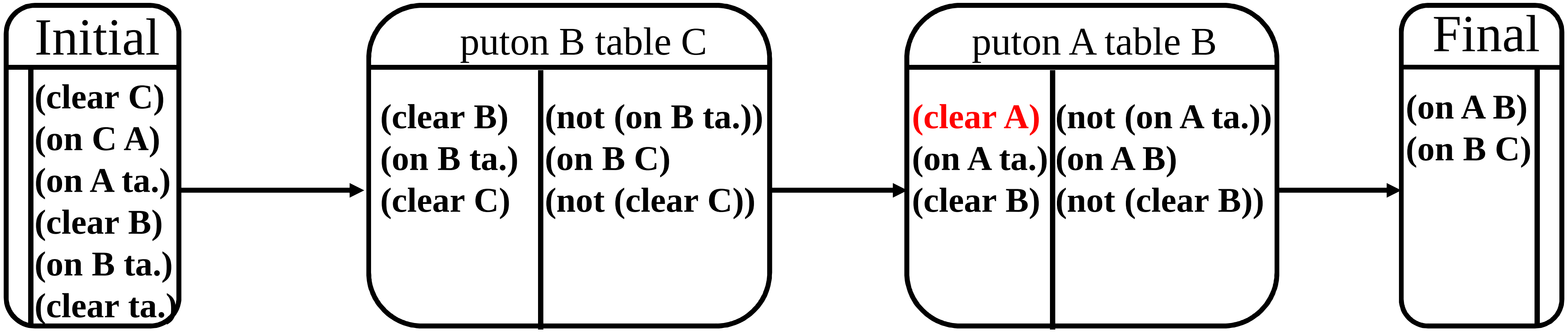
The anomaly of Gerald Jay Sussman (7/16)



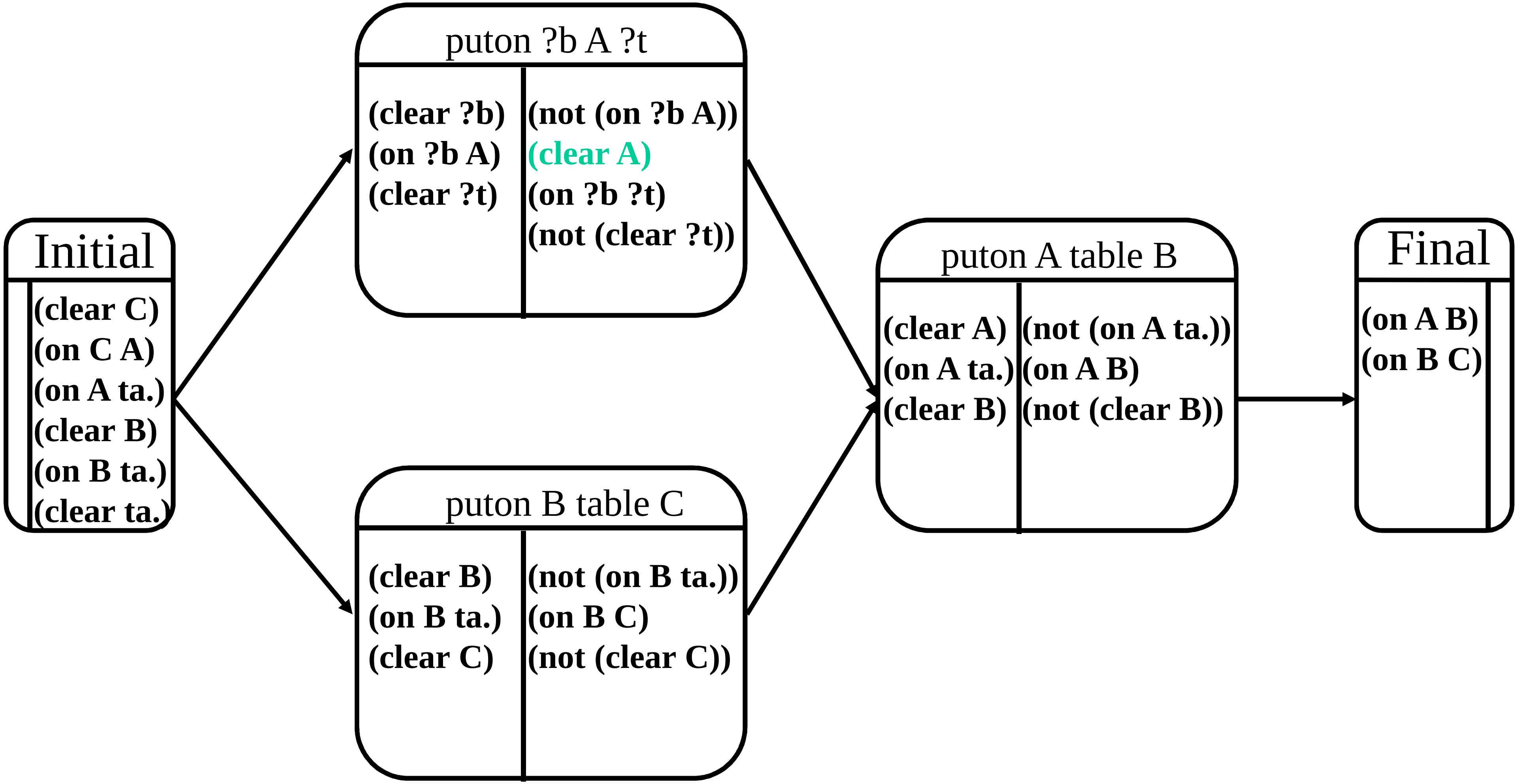
The anomaly of Gerald Jay Sussman (8/16)



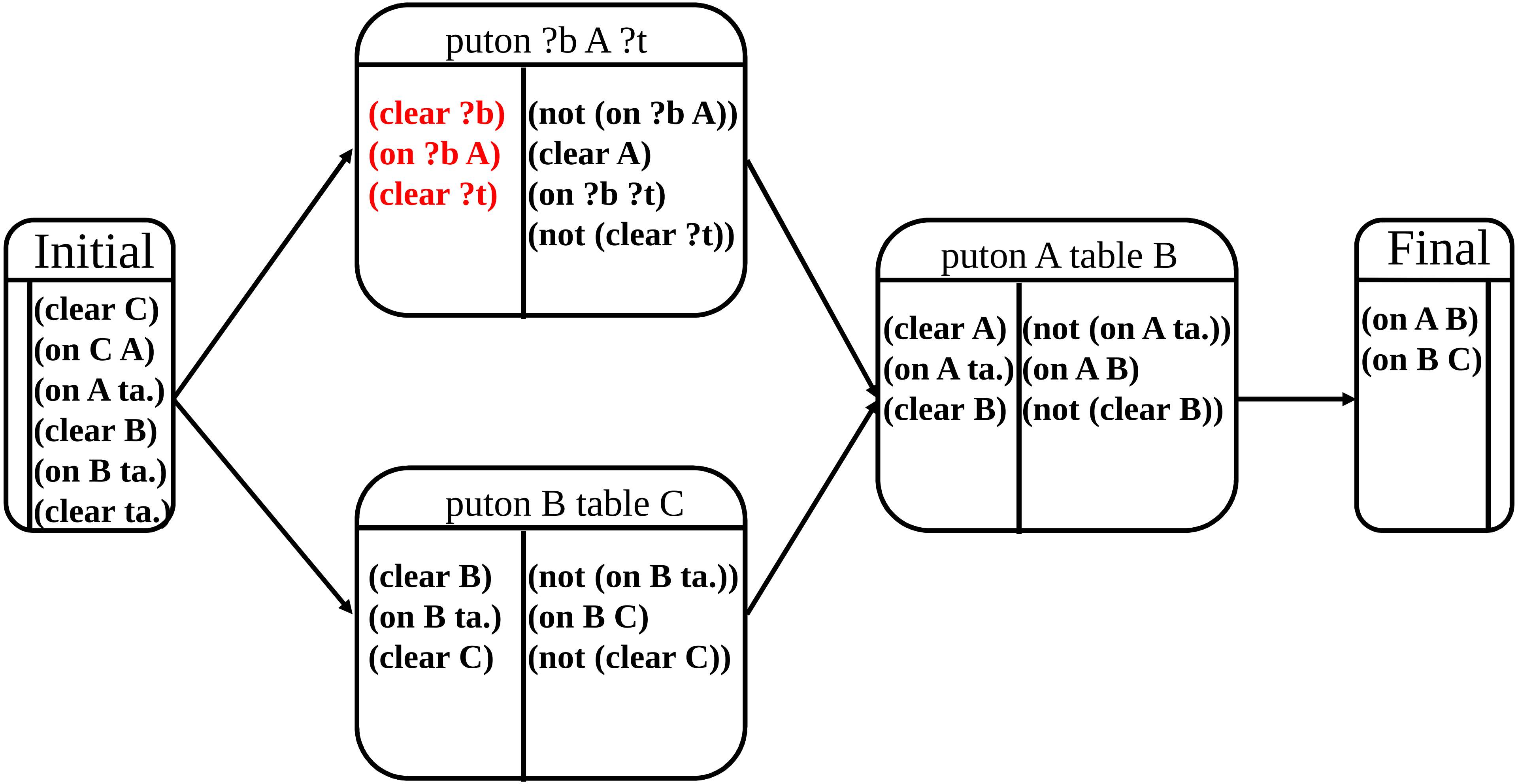
The anomaly of Gerald Jay Sussman (9/16)



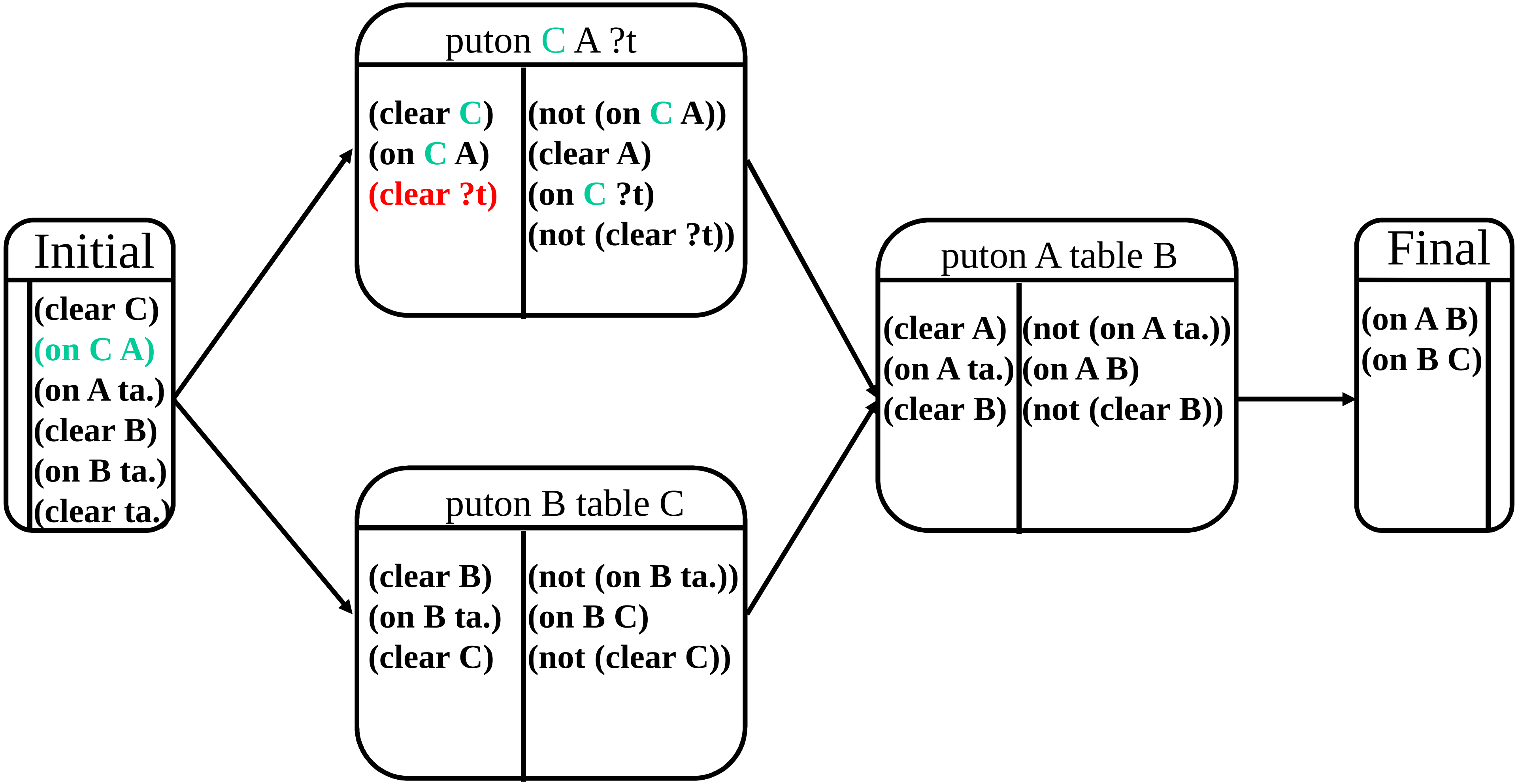
The anomaly of Gerald Jay Sussman (10/16)



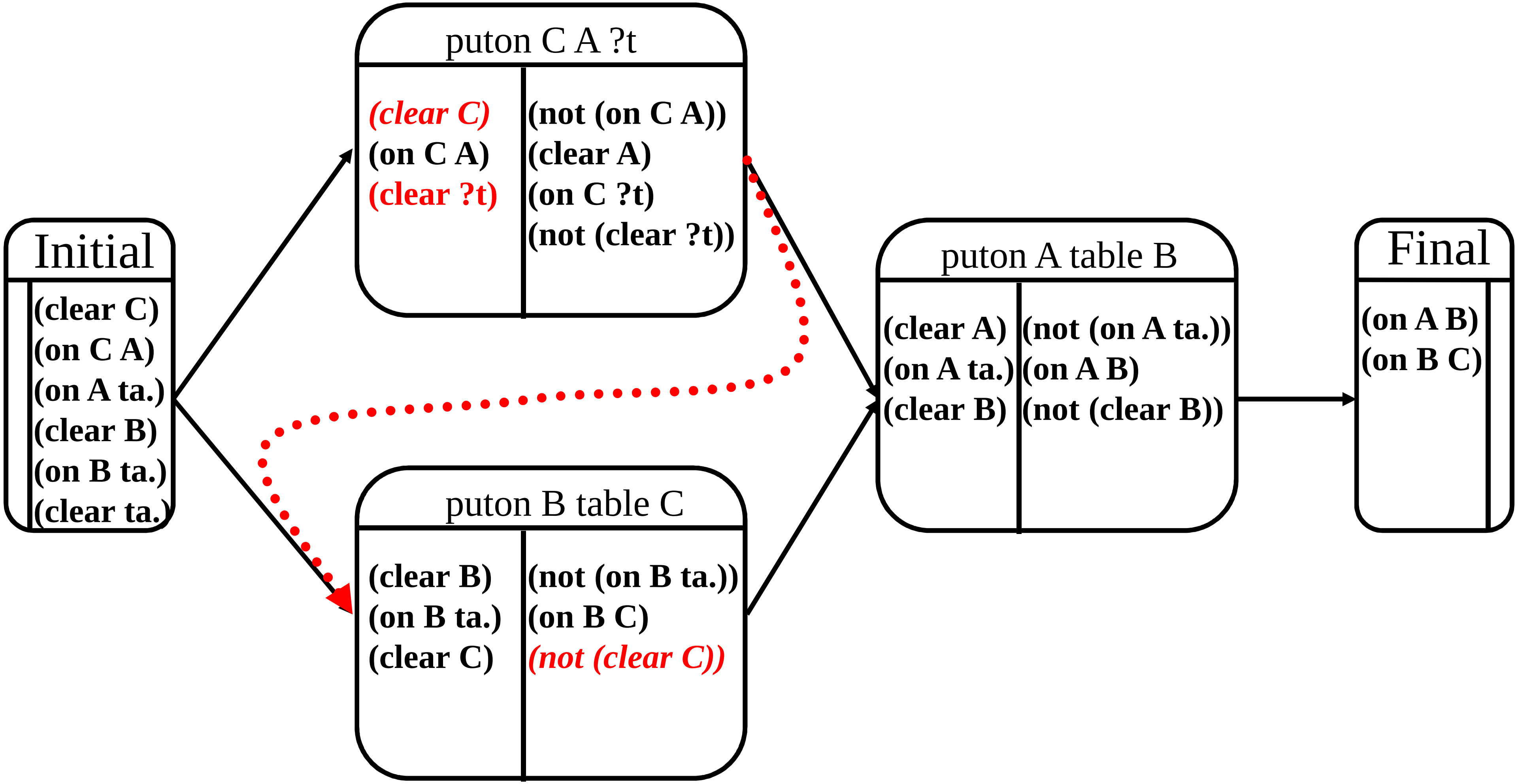
The anomaly of Gerald Jay Sussman (11/16)



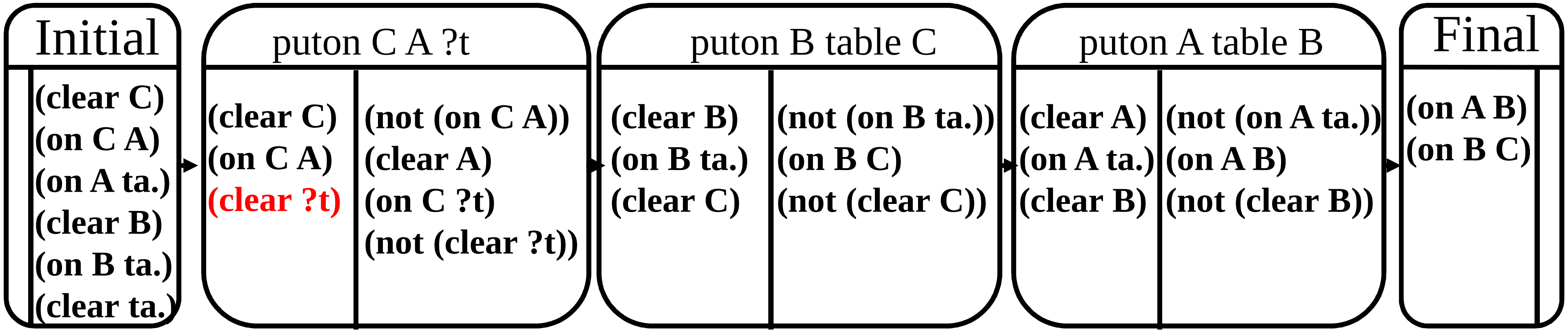
The anomaly of Gerald Jay Sussman (12/16)



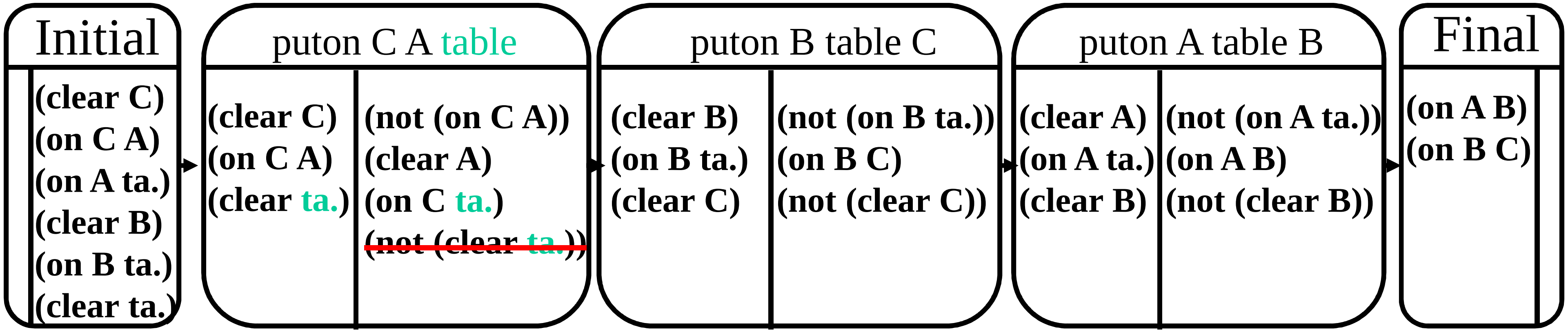
The anomaly of Gerald Jay Sussman (13/16)



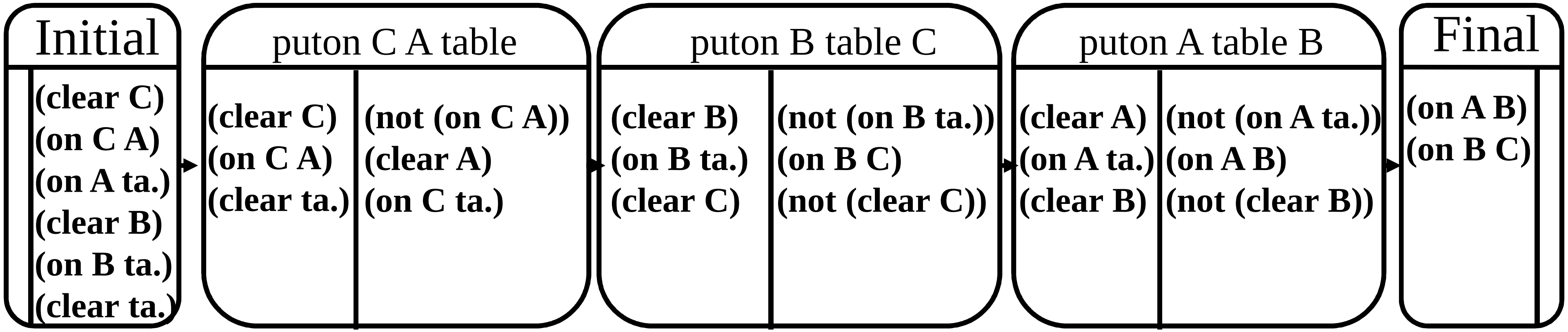
The anomaly of Gerald Jay Sussman (14/16)



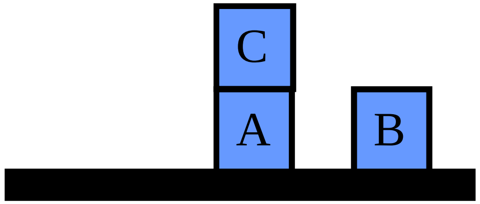
The anomaly of Gerald Jay Sussman (15/16)



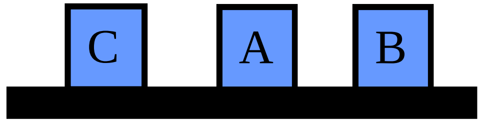
The anomaly of Gerald Jay Sussman (16/16)



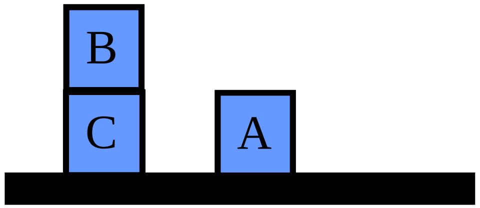
The anomaly of Gerald Jay Sussman: solution



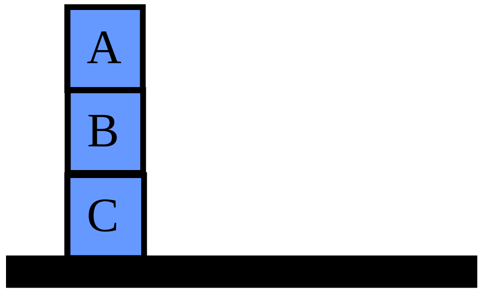
(1)



(2)



(3)



(4)

Algorithms (1/3) : intuitively …

- Plan = (Templates *T*, Operators *Op*, PartialOrder *O*, Unification *U*).
- Definitions :
  - ✓ Conflict : a post-condition might destroy a causal link (threat).
  - ✓ Satisfying a pre-condition *p* : adding an operator before *p*, which a post-condition *q* unifies with *p (i.e., q = p).*

- *PLANNER1(T, Op., O, U) :*

WHILE(conflict || at least 1 unsatisfied pre-condition)

1. *Solve conflicts :*
   - Add a unification / non unification constraint;
   - Add a precedence constraint.
2. *Choose an unsatisfied pre-condition p ;*
3. *Satisfy p :*
   - Add a unification / non-unification constraint;
   - Add a precedence constraint ;
   - Add an operator.

END WHILE

Algorithms (2/3) : in the state space

- A state S_i = a state of the environment.
  - ✓ S_0 = initial state, composed of the fluents $f_1, \ldots, f_n$.
  - ✓ Example : a given configuration of blocks.
- *Successors*(S_i) = n states $S_{i+1}, \ldots, S_{i+n}$ which can be reached by an instantiated operator applicable in S_i.
- *Solution*(S_i) iff S_i includes the goals $g_1, \ldots, g_l$.

- *PLANNER2(Alg., S_0, Successors, Solution) :*
  - ✓ Alg. = any state-space search algorithm.
  - ✓ Heuristics ?

- Example :
  - ✓ Algorithm A* of *Heuristic Search Planner* (HSP) [Bonet 98]
  - ✓ Heuristics: a graph plan without negative pre-conditions.

Algorithms (3/3) : in the plan space

- A state = a partially-ordered partially instantiated plan.

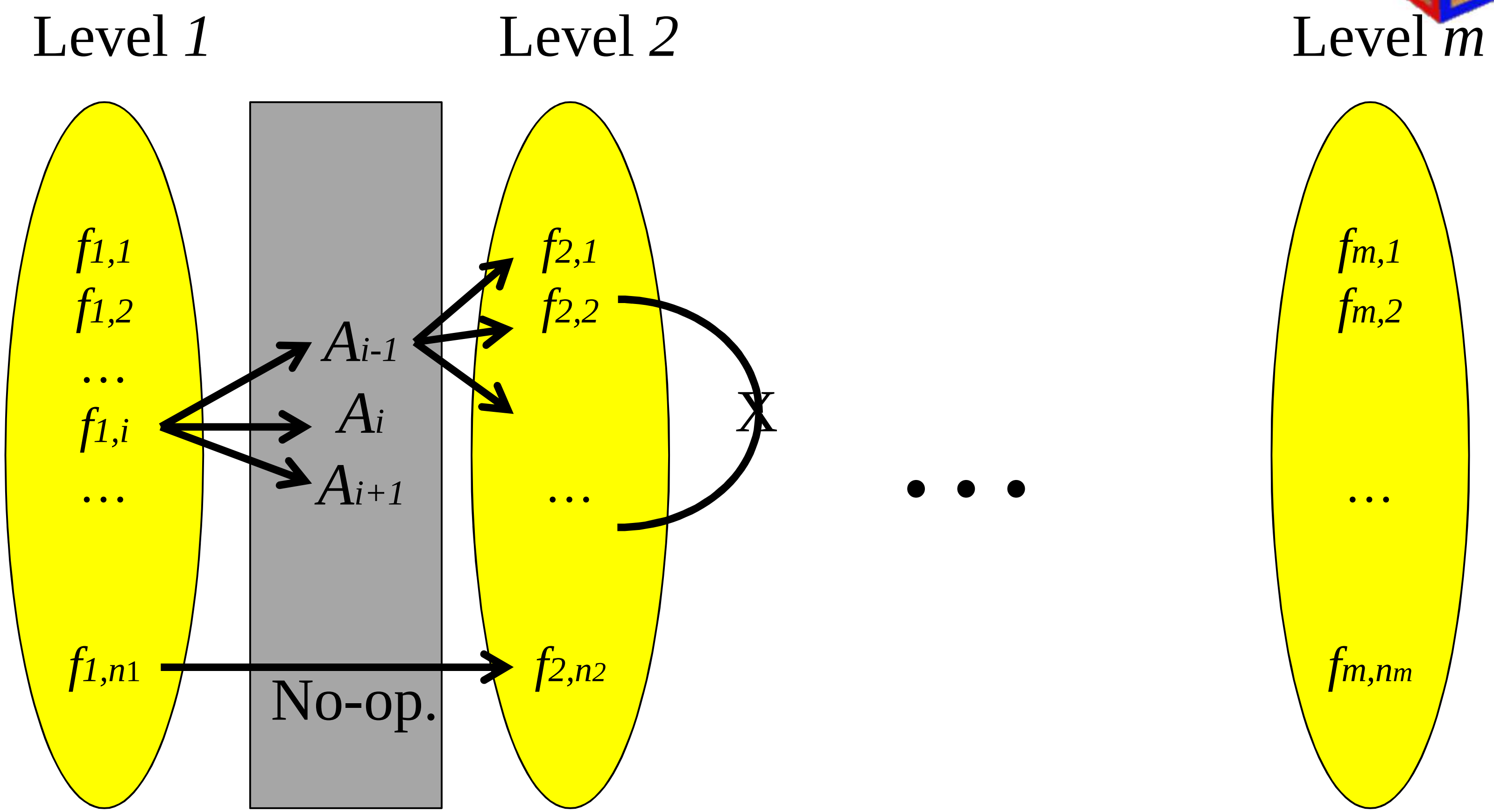
- *PLANNER3(T, Op., O, U) :*

  Search algorithm in a space of states (e.g., A*) :

  - *A state = a partial plan (T, Op., O', U').*
  - *A successor = obtained by solving a conflict, or satisfying a pre-condition.*
    - Adding a unification / non-unification constraint to *U*
    - Adding a precedence constraint to *O*
    - Adding an operator from T to *Op*
  - *A solution function = no conflict && all pre-conditions are satisfied.*

- Heuristics?
- Example:
  - ✓ Universally quantified Conditional Partial-Order Planner (UCPOP) [Penberthy 92]
  - ✓ http://www.cs.washington.edu/homes/weld/ucpop.html

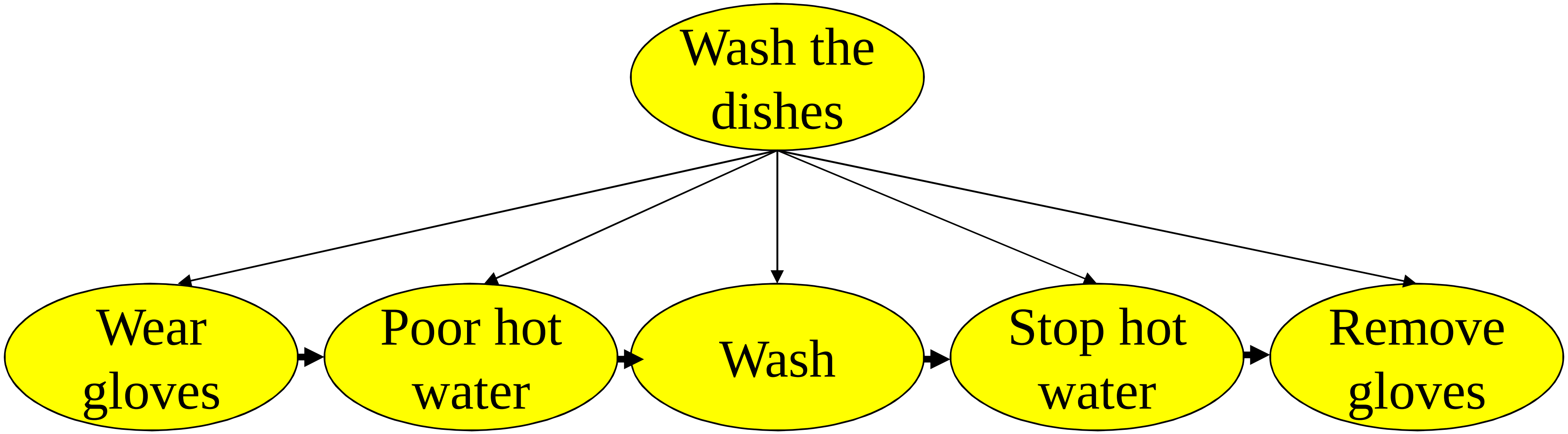
Plan of graph [Blum 97]



- ➢Levels include mutual exclusions (*mutex*) : a level is not a state!
- ➢Forward development of a plan of graph, backward search.

Hierachical Tasks Network (HTN)

- Additional knowledge:
  - A task can be decomposed into sub-tasks.



- Search algorithm by refining plans.
  - Simple Task Network (STN).
- Example :
  - The HTN planner used by Jason Wolfe at Willow Garage (CA).
  - SHOP and SHOP2, by Dana Nau from Univ. Maryland.

SAT-based planning [Kautz 92]

The SAT problem:

- Find a truth value for each proposition, which together satisfy a given logical formula, using AND, OR and NOT.
- Proposition logics.
- 1st problem which has been proved NP-complete (1971).
- International Conference SAT'12. http://www.satisfiability.org/

Principle of SAT-based planning:

1. *n = 1* *// Length of the solution-plan.*
2. Turn a plan of length n into a formula in proposition logic.
3. Attempt to prove this formula using a SAT solver.
4. IF it fails, THEN (i) increment *n*, (ii) GOTO 2.

Example of formulas, in the blocks world :

- For all *x, y, z, i* :
    - *on(x, y, i) && clear(x, i) && clear(z, i) && puton(x, y, z, i) => clear(y, i+1) && on(x, z, i+1)*
- For all *x, x', y, y', z, z', i* :
    - *x <> x' && y <> y' && z <> z' => not puton(x, y, z ,i) || not puton(x' ,y' ,z' ,i)*

CSP-based planning

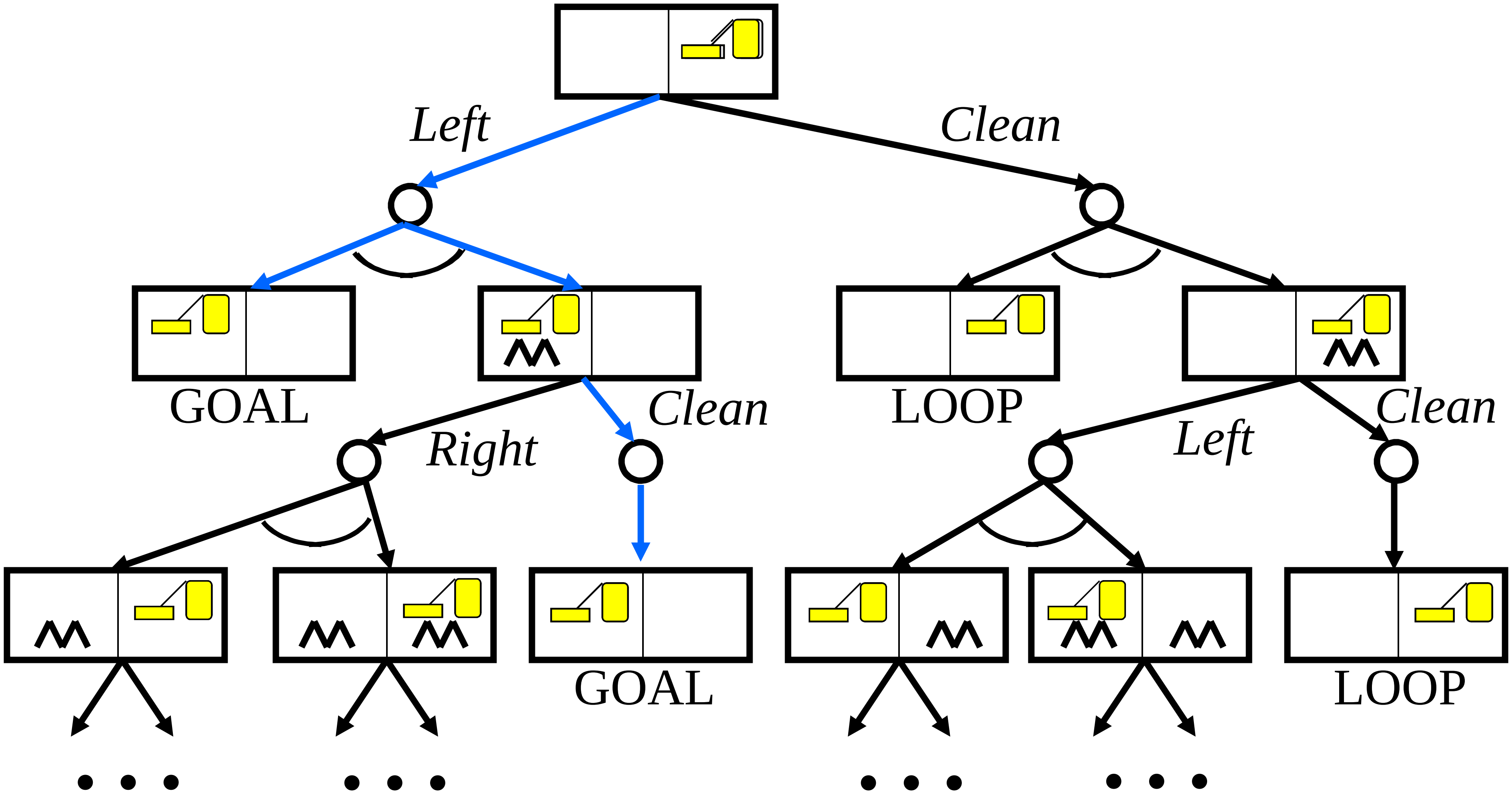
- **Constraint programming:**
  - ✓ **Statement:** for each variable, search for a value from the variable's domain, so that all values satisfy the constraints.
  - ✓ **Representation:** Variables / Domains / Constraints.
  - ✓ **Algorithm:**
    - ***Choose a variable***
    - ***Choose a value from the variable's domain***
    - ***Propagate this assignment through constraints***
    - ***If a domain becomes empty, backtrack on previous choices.***
- **Principle of CSP-based planning:**
  1. **Heuristically estimate the length *n* of a solution-plan;**
  2. **Turn the planning problem into a dynamic CSP;**
  3. **Attempt at finding a solution plan of length *n* using a CSP solver;**
  4. **IF failure, THEN increment n ; GOTO 2.**
- **Examples :**
  - ✓ **IxTeT planner from LAAS-CNRS in Toulouse [Laborie 95].**
    - ***http://spiderman-2.laas.fr/RIA/IxTeT/ixtet-planner.html***
  - ✓ **Constraint Programming Temporal planner (CPT) from ONERA Toulouse [Vidal 06].**
    - ***http://v.vidal.free.fr/onera/#cpt***

Conditional planning (1 / 2)

- The agent may not know the output of its own actions.
- Plans have branches:
  - ✓ IF < *test* > THEN Plan_A ELSE Plan_B
  - ✓ Obtain a plan in every case
- Full observability: the agent knows its state.
  - ✓ No need for an operator « observe ».
- Actions might fail: disjunctive effects.
- Conditional effects.
  - ✓ EFFECTS : IF To_Left, CleanL ; IF To_Right, CleanD.
- Conditional planning is harder than NP.
- Example (double Murphy) : a vacuum-cleaner agent must cleam all rooms.
  - ✓ Rule 1: the vacuum cleaner sometimes drops dust when it moves to a clean room.
  - ✓ Rule 2 : the vacuum cleaner sometimes drops dust if CLEAN is executed in a clean room.

Conditional planning (2 / 2)

➢Search in an AND-OR graph:



History

- 1971: STRIPS from Richard Fikes.
- 1977: NOAH from Earl Sacerdoti
- 1981: MOLGEN from Mark Stefik
- 1986: IxTeT from Malik Ghallab.
- 1986: SIPE from David Wilkins.
- 1987: TWEAK from David Chapman.
- 1991: SNLP from Mac Allister & Rosenblitt.
- 1992: UCPOP from Anthony Barrett & Daniel Weld.
- 1992: BLACKBOX/SATPLAN from Henry Kautz & Bart Selman.
- 1997: GRAPHPLAN from Avrim Blum & Merrick Furst.
- 2000: HSP from Hector Geffner.
- 2000: YAHSP from Vincent Vidal.
- 2001: FF from Jörg Hoffmann,
- 2005: CPT from Vincent Vidal.
- 2007: DAE from Marc Schoenauer.

References (1 / 2) – Part I

- [Weld 94] Daniel Weld, *An Introduction to Least Commitment Planning*, A. I. Magazine, 15(4), pages 27-61, Winter 1994.
- [Russel 2010] Stuart Russell, Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 2010, 3rd edition. Chapitre 10.
- [Ghallab et al. 04] Malik Ghallab, Dana Nau, Paolo Traverso. *Automated Planning: Theory and Practice*. Morgan Kaufmann, San Mateo, CA, May 04, 635 pages.
- PDDL 3.1. http://en.wikipedia.org/wiki/Planning_Domain_Definition_Language
- Conférences :
  - International Conference on Automated Planning and Scheduling (ICAPS). http://www.icaps.org
  - International Joint Conference on A.I. (IJCAI). http://www.ijcai.org
  - European Conference on A.I. (ECAI). http://www.ecai.org
  - National Conference on A.I. (AAAI). http://www.aaai.org
- Journals :
  - A. I. Journal (AIJ). http://www.elsevier.com/wps/find/journaldescription.cws_home/505601/description#description
  - Journal of A.I. Research (JAIR). http://www.jair.org/

References (2 / 2) – Part II

- **[Blum 97]** A. Blum, M. Furst. *Fast Planning through Planning Graph Analysis*. Artificial Intelligence, 90:281-300, 1997.
- **[Bonet 98]** B. Bonet, H. Geffner. *HSP: Heuristic Search Planner*. In Proceedings of Artificial Intelligence Planning Systems (AIPS), 1998.
- **[Kautz 92]** H. Kautz, B. Selman. *Planning as Satisfiability*. In Proceedings of ECAI'92.
- **[Laborie 95]** P. Laborie, M. Ghallab. *Planning with Sharable Resource Constraints*. In Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), 1995, pages 1643 – 1651.
- **[Penberthy 92]** J. S. Penberthy, D. Weld. *UCPOP: A Sound, Complete, Partial-Order Planner for ADL*. In Proceedings of 3rd International Conference on Knowledge Representation and Reasoning (KR'92), Cambridge, MA, 1992.
- **[Vidal 06]** V. Vidal, H. Geffner. *Branching and Pruning: An Optimal Temporal POCL Planner based on Constraint Programming*. Artificial Intelligence, 170(3): 298-335, 2006.

Conclusion of part I (1/2)

- Domain explored for 40 years
- Some planners now are available:
  - CPT, FF, SATPLAN, ...
  - International Planning Competition (IPC).
    - *http://ipc.icaps-conference.org/*
- Conditional planning: more difficult…
- Properties of a planner:
  - Correctness
  - Completeness
  - Optimality
  - Canonicity
  - Efficiency

- Hint:
  - Merge probabilistic planning (MDP) and symbolic planning (STRIPS).

Conclusion of part I (2/2)

- Demo of the CPT task planner…

Planning & Execution [Russel 2010]

➢ When to plan?

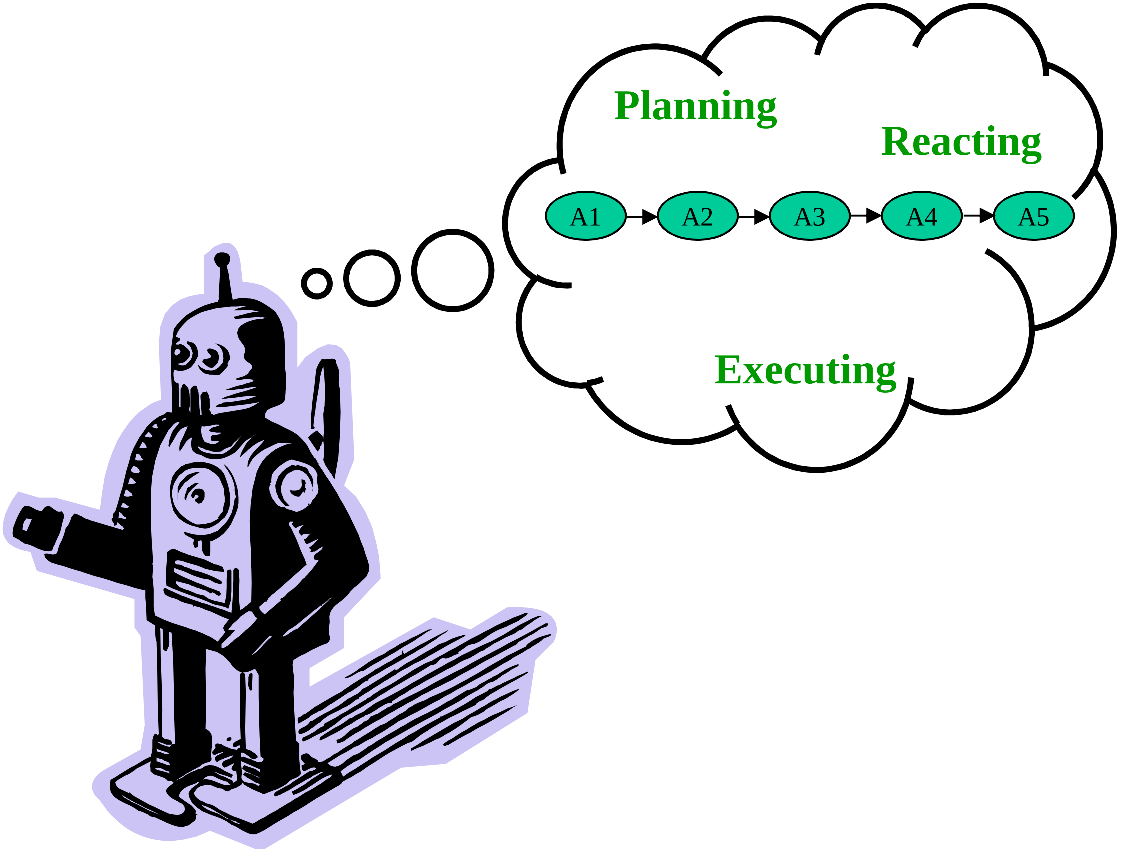
- ✓ Before executing (off-line planning).
- ✓ While executing (on-line planning).

Part II ---

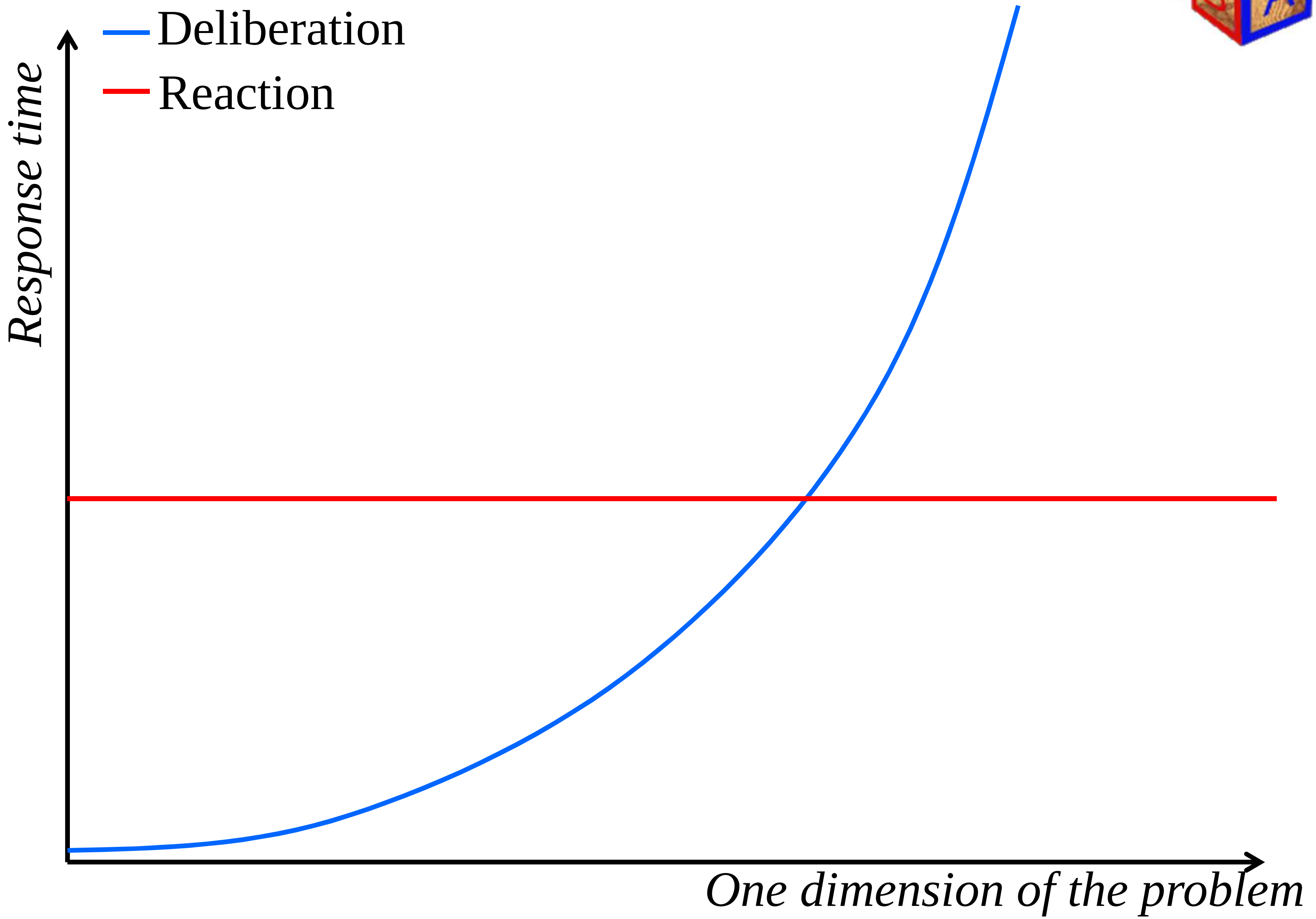
Robotics-oriented agent architectures

Statement

- « An agent is a system including reasoning (e.g., temporal), perceiving its environment, acting on it and interacting with other agents (artificial or human). »



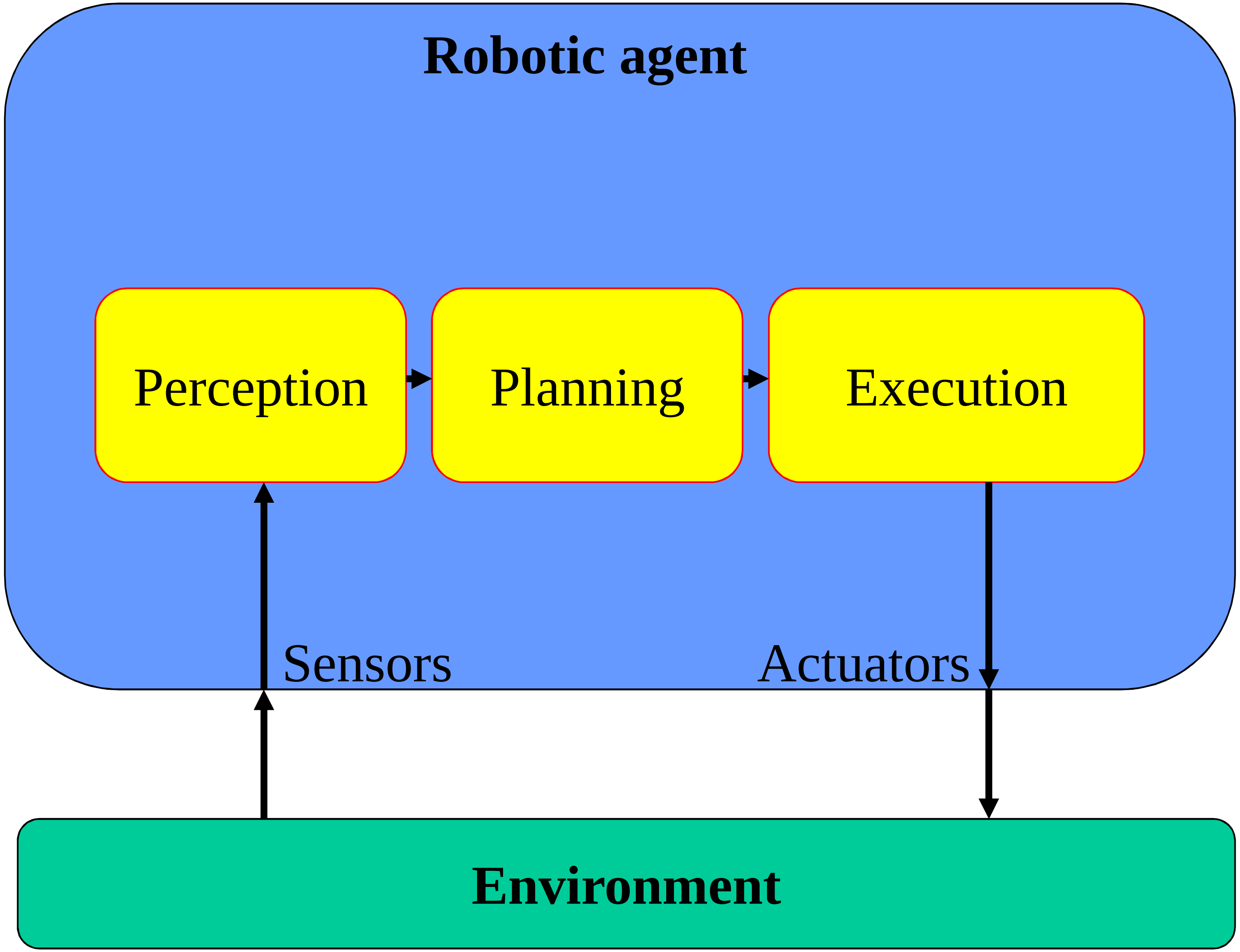
Difficulty



Deliberation
Reaction
Response time
One dimension of the problem

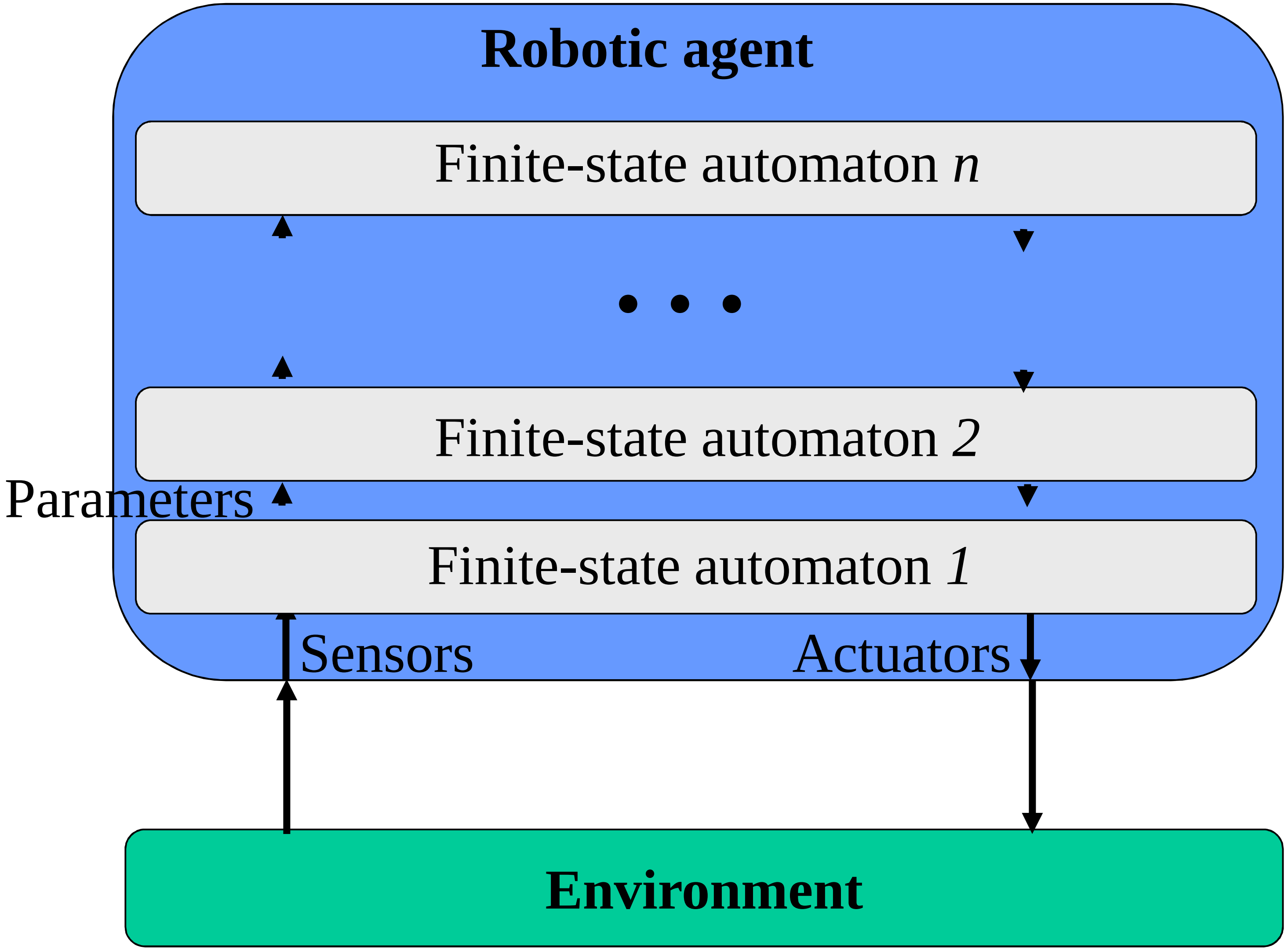
Examples of robotic agents

Architecture Sense-Plan-Act [Nilsson 80]

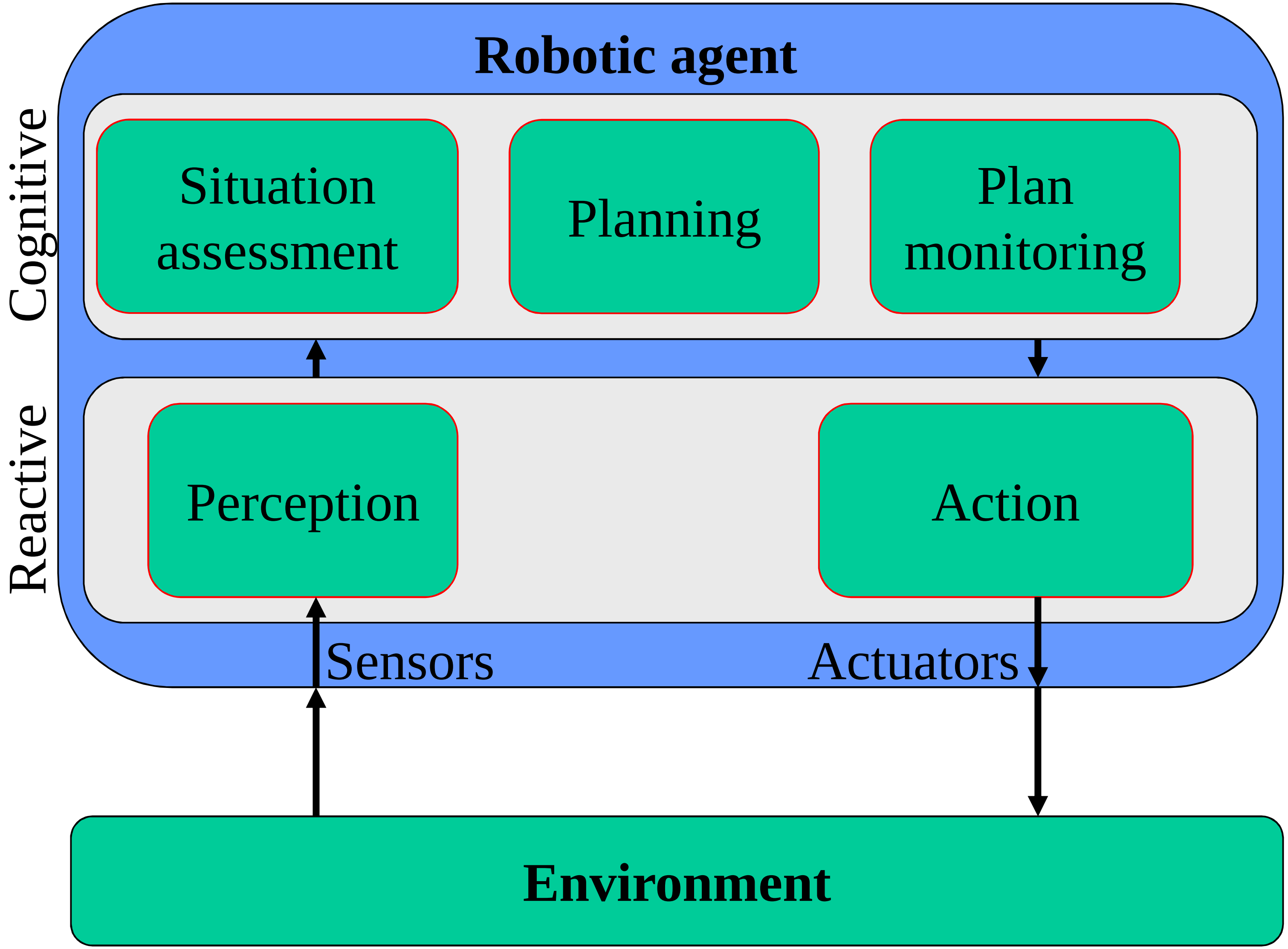


Subsumption architecture [Brooks 85]

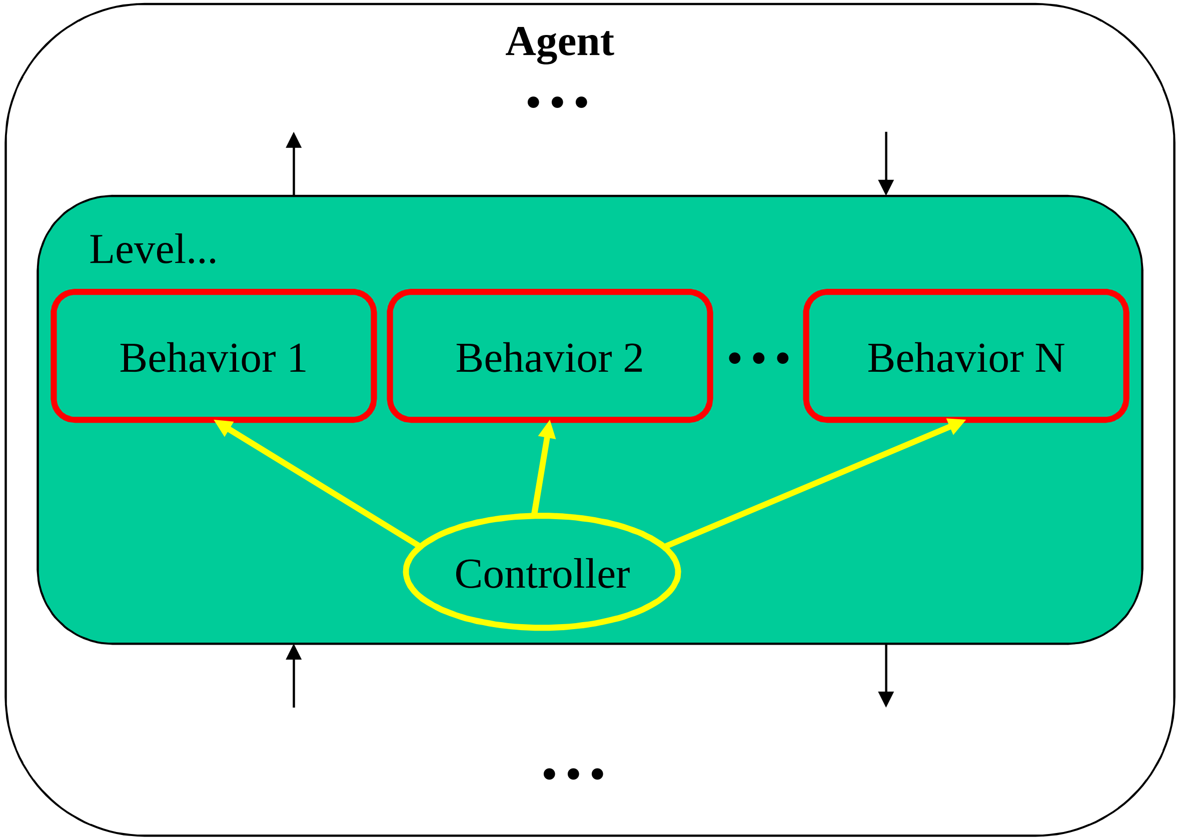
- No symbol [Brooks 91].



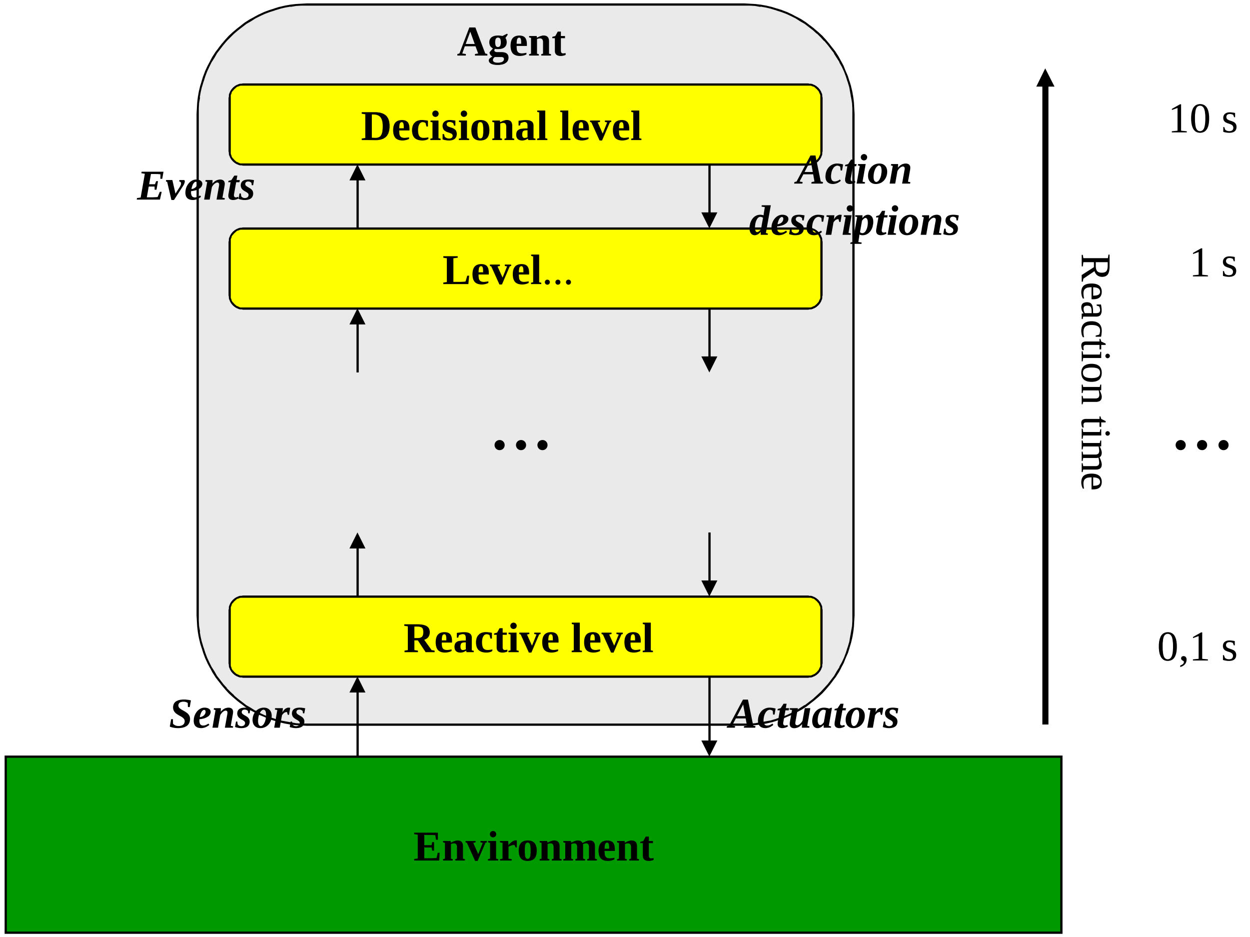
2-level architecture [Hayes-Roth et al. 95] (1/3)



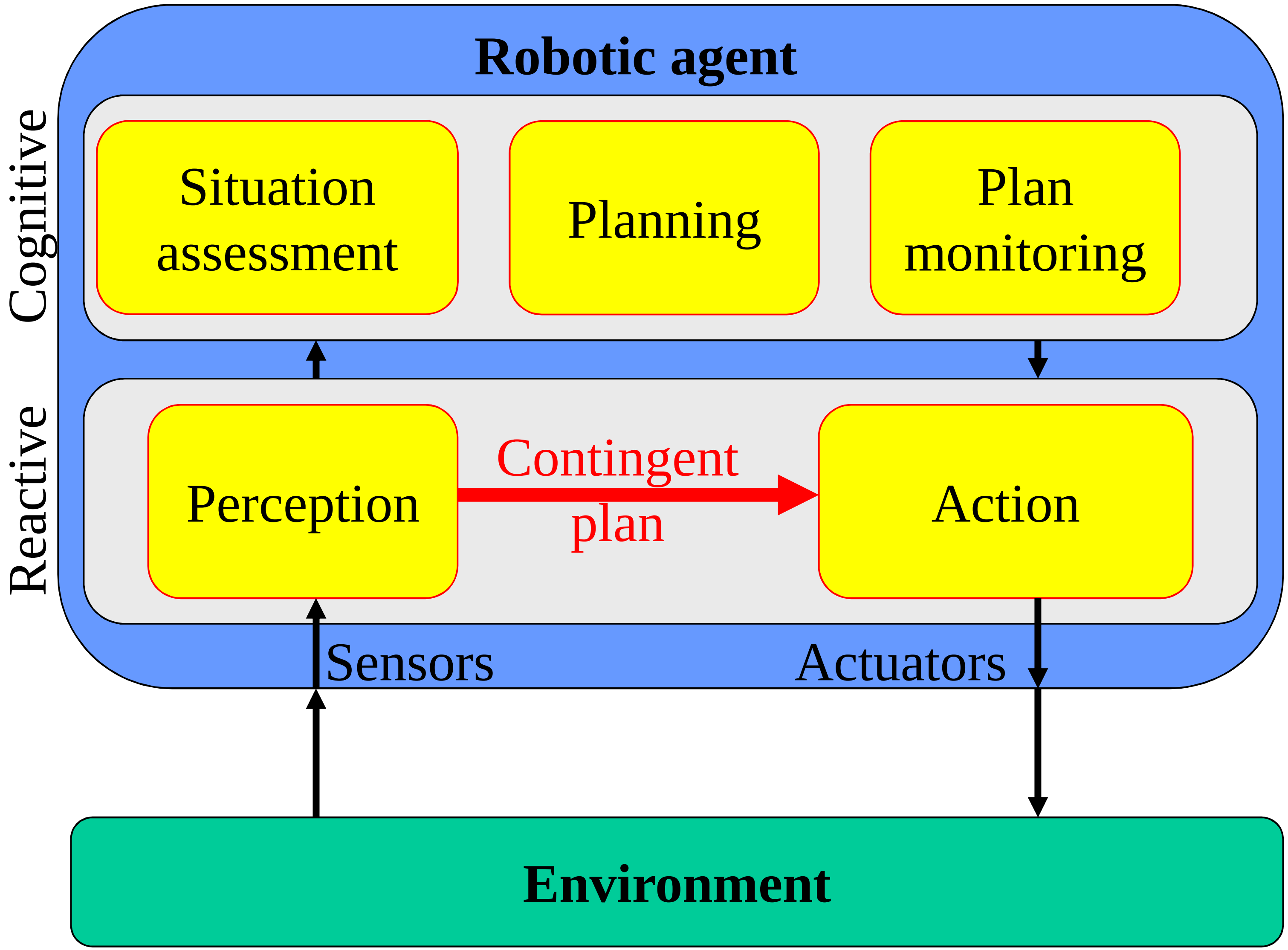
2-level architecture [Hayes-Roth et al. 95] (2/3)



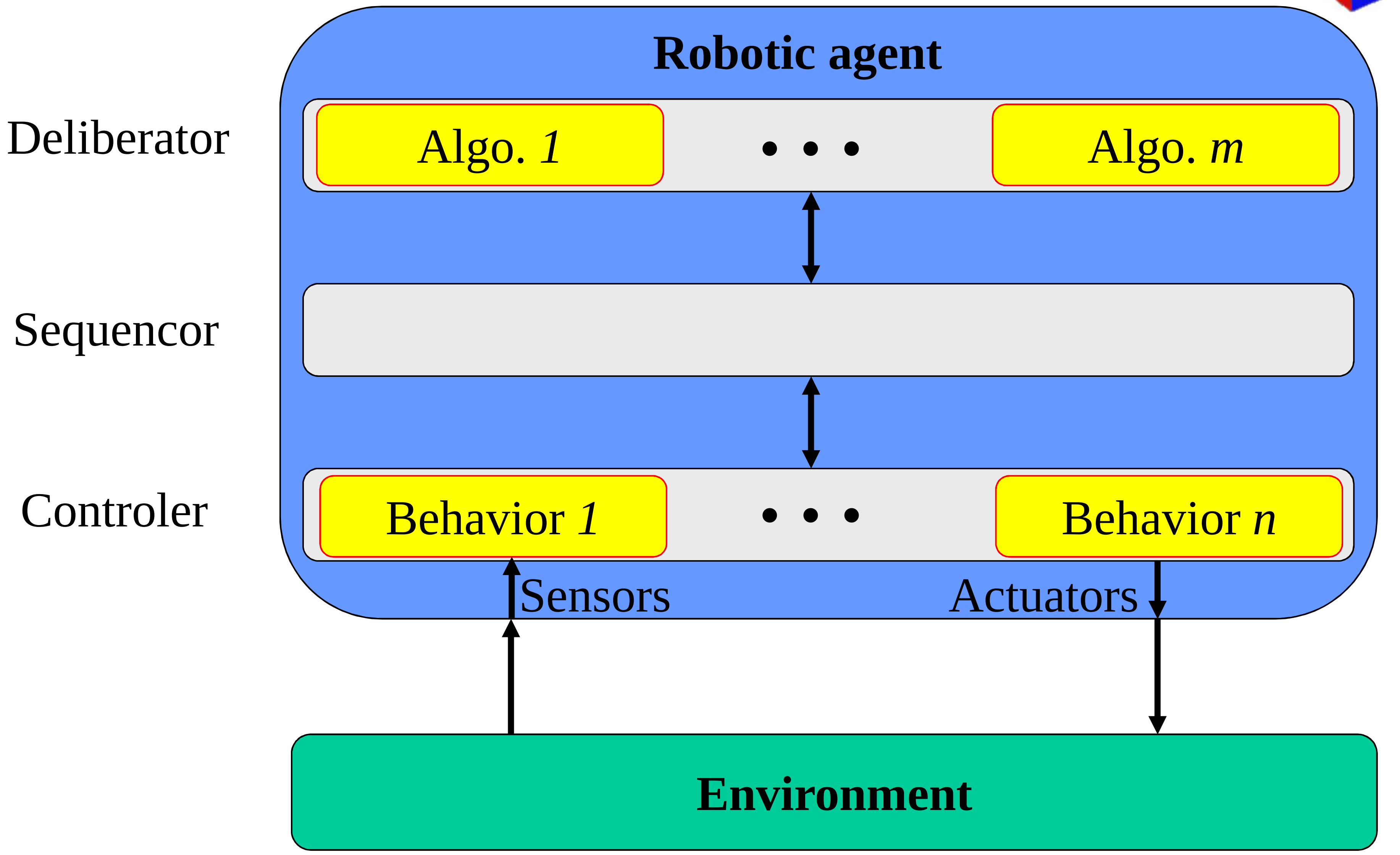
2-level architecture [Hayes-Roth et al. 95] (3/3)



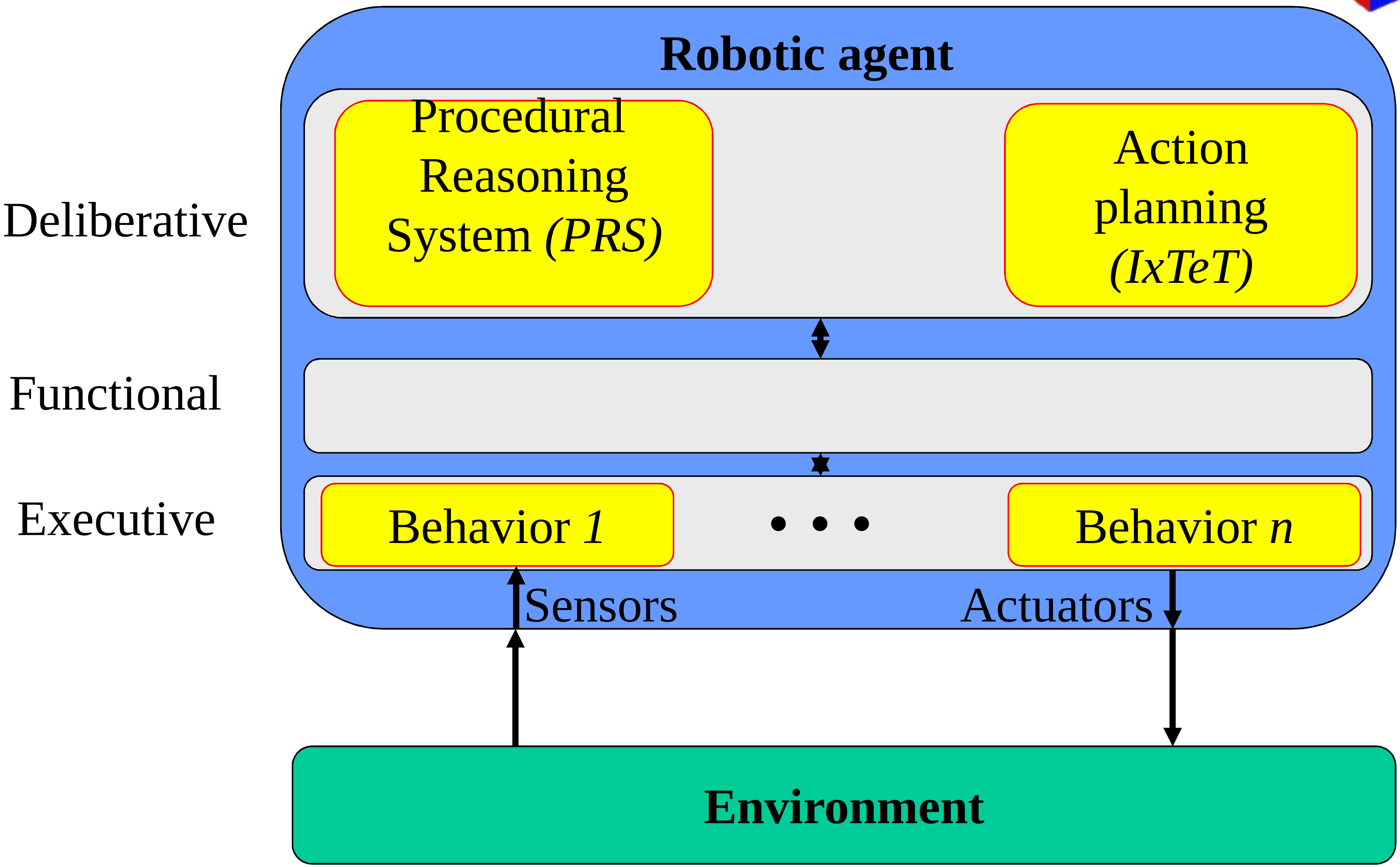
2-level++ architecture [Baltié et al. 07]



3-level architecture [Gat 98]



LAAS-CNRS architecture [Alami et al. 98]

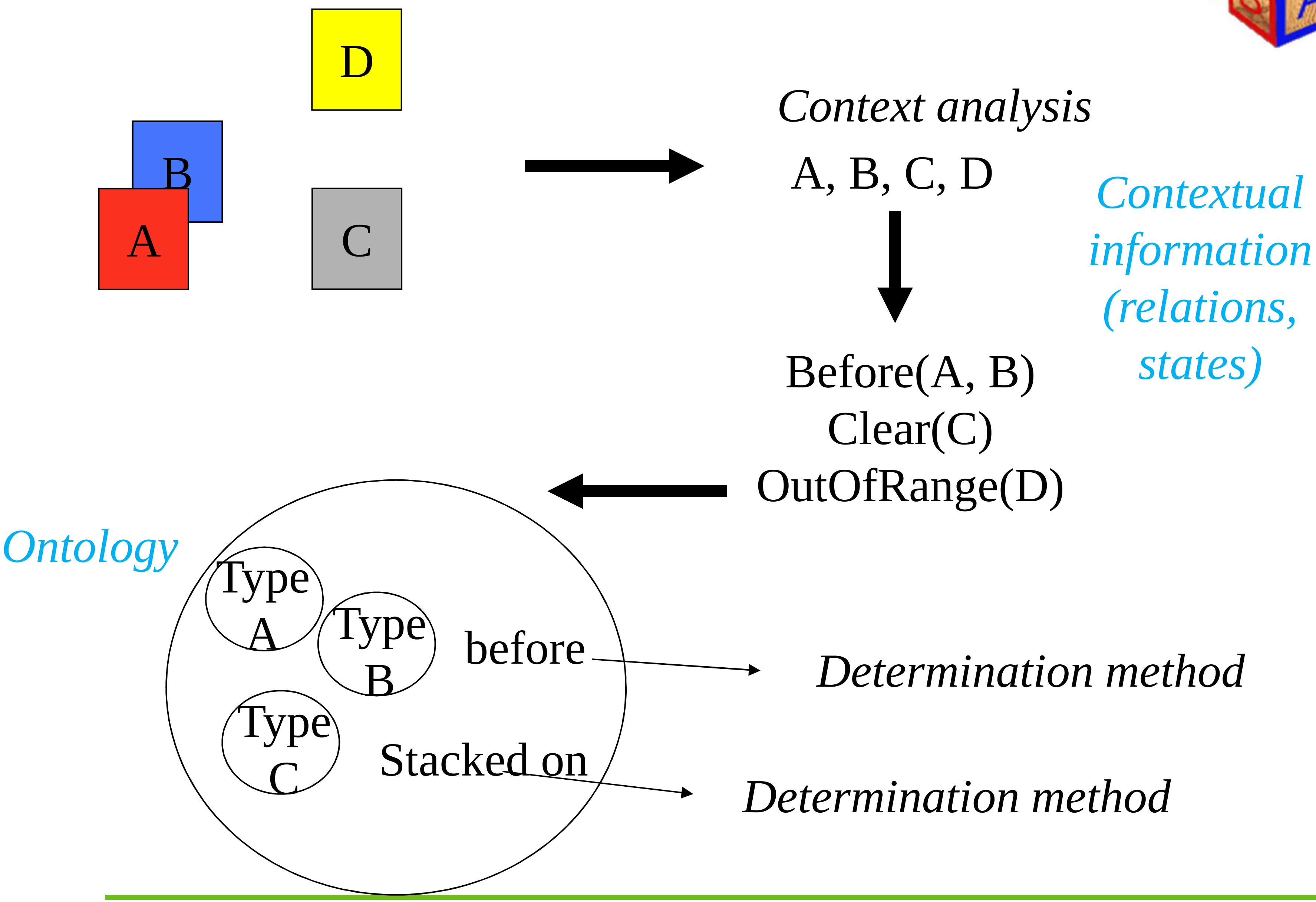


Architectures for robots

- Open RObotic COntrol Software, from H. Bruyninckx (Belgium).
  - Ontology, no task planning.

- Robotic Operating System (ROS), from Willow Garage (CA).
  - An HTN planner [Wolfe 10], no ontology.

An ontology for the robot SAM



Conclusion of Part II

- Domain explored since [Nilsson 80], i.e., ~30 years.
- No unique architecture makes consensus!
- Critical properties:
  - ***Real-time***
    - *How to get immediately a good reaction?*
  - ***Safety***
    - *How to get a good reaction in the worst case?*
- Hints:
  - Ontology to analyze the context.
  - Multi-Agent Systems : *intelligence emerges from interaction among agents*.

References (1 / 2) – Part II

- **[Alami et al. 98]** R. Alami, R. Chatila, S. Fleury, M. Ghallab, F. Ingrand. *An Architecture for Autonomy*. In *International Journal of Robotics Research* (Special Issue on ``Integrated Architectures for Robot Control and Programming"), Vol 17, N° 4, Apri1 1998. LAAS Report N°97352.
- **[Baille et al. 99]** Gérard Baille & al, *Le CyCab de l'INRIA Rhône-Alpes*. Rapport de recherche de l'INRIA Rhône-Alpes n°0229, April 1999 (in French).
- **[Baltie et al. 07]** J. Baltié, E. Bensana, P. Fabiani, J. – L. Farges, S. Millet, P. Morignot, B. Patin, G. Petitjean, G. Pitois, J. – C. Poncet. *Multi-Vehicle Missions: Architecture and Algorithms for Distributed On Line Planning*. In Dimitri Vrakas and Ioannis Vlahavas (eds.), Artificial Intelligence for Advanced Problem Solving Techniques, Information Science Reference. December 2007.
- **[Beetz et al. 10]** M. Beetz, D. Jain, L. Mösenlechner, M. Tenorth. Towards Performing Everyday Manipulation Activities. Robotics and Autonomous Systems, April 2010.
- **[Brooks 85]** Brooks, R. A. *A Robust Layered Control System for a Mobile Robot*. In *IEEE Journal of Robotics and Automation*, Vol. 2, No. 1, March 1986, pp. 14–23.
- **[Brooks 91]** R. Brooks. *Intelligence without reason*. Proceedings of 12th Int. Joint Conf. on Artificial Intelligence (IJCAI'91), Sydney, Australia, August 1991, pp. 569–595.
- **[Campa et al. 96]** Giampiero Campa, Mario Innocenti, Jacqueline Wilkie. *Model-Based Robust Control for a Towed Underwater Vehicle*. AIAA Guidance, *Navigation and Control Conference*, San Diego, California, July 29-31, 1996.
- **[Gat 98]** Gat, E. *Three-layer architectures*. In D. Kortenkamp et al. Eds. *A.I. and mobile robots*. AAAI Press, 1998.
- **[Hayes-Roth et al. 95]** Hayes-Roth, B.; Pfleger, K.; Morignot, P.; & Lalanda, P. *Plans and Behavior in Intelligent Agents*. Knowledge Systems Laboratory, KSL-95-35, Stanford Univ., CA, March, 1995.

References (2 / 2) – Part II

- **[Nilsson 80]** Nils J. Nilsson. *Principles of ArtificialIntelligence*. Palo Alto: Tioga. 1980.
- **[Muscettola et al. 98]** N. Muscettota, P. Pandurag Nayak, Barney Pell and B.C. Williams. *Remote Agent: to Boldly go where no AI System Has Gone Before*. *Artificial Intelligence*, Elsevier, 103, pp.5-47, 1998.
- **[Russel 2010]** Stuart Russell, Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 2010, 3rd edition. Chapitre 11.
- **[Schoppers 95]** Schoppers, M. *The use of dynamics in an intelligent controller for a space faring rescue robot*. In *Artificial Intelligence Journal*, 73 (1995):175-230.
- **[Teichteil et al. 11]** F. Teichteil-Königsburg, C. Lesire, G. Infantes. *A Generic Framework for Anytime Execution-Driven Planning in Robotics*. In Proceedings of the International Conference on Robotics and Automation, Shanghai, China, May 2011, pages 299-304.
- **[Wolfe et al. 10]** J. Wolfe, B. Marthi, S. Russell. *Combining Task and Motion Planning for Mobile Manipulation*. In Proceedings of the International Conference on Automated Planning and Scheduling, Toronto, Canada, 2010.

General conclusion

$$f(x) = e^x$$

If P is different than NP,

then we are fighting against the exponential function

in the worst case!

- Thank you for your attention!