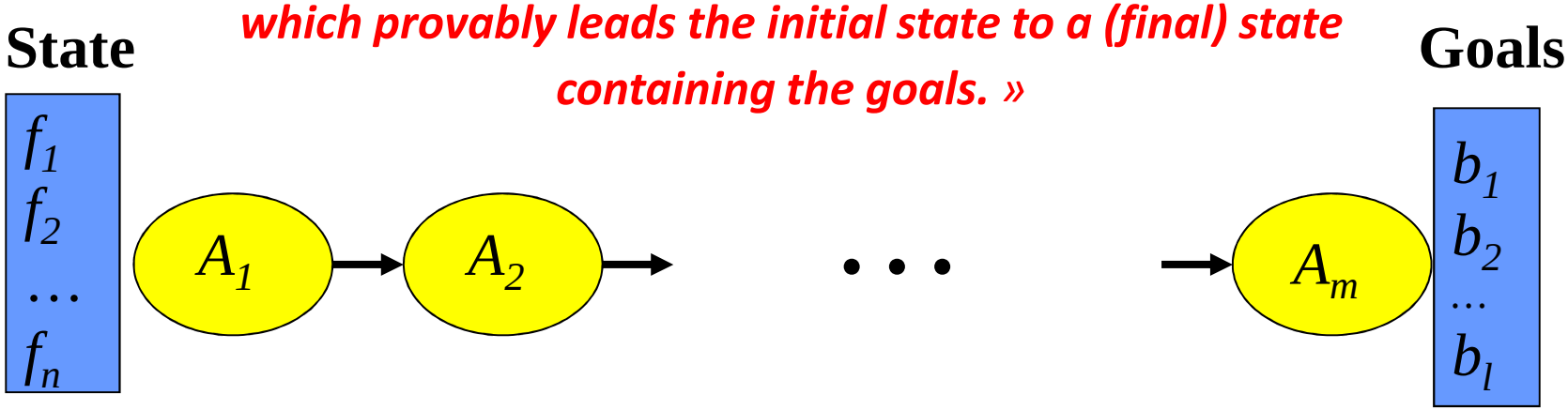
Action planning

Philippe Morignot
philippe.morignot@vedecom.fr

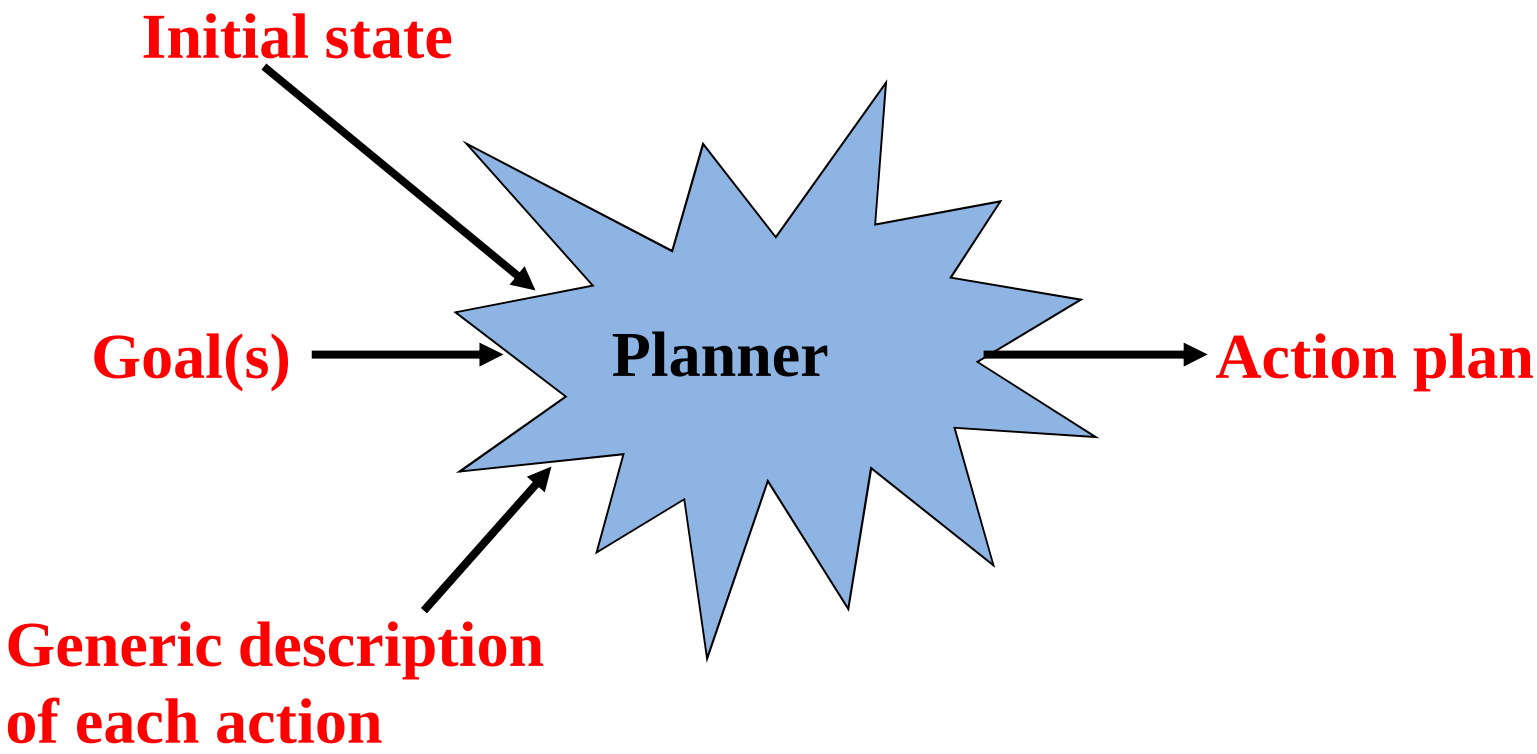
Statement of the problem

« Given possible generic actions,
an initial state and goals,
find a sequence of instantiated actions,
which provably leads the initial state to a (final) state
containing the goals. »



- « ***Action planning*** » / « ***plan synthesis*** » / « ***generation of action plan*** » : activity of constructing a plan.
- « ***Planner*** » / « ***task planner*** » / « ***action planner*** » : computer program which solves this problem.
  - Different from « path planner » in Robotics.

An action planner



Difficulty

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

- If $a = 5$, $b = 3$, $c = 3$, $d = 100$, then ~ 10^{277} states.
- Classical planning is NP.
- **All the states cannot be enumerated.**

Assumptions

- A1 : ***the agent is the sole cause of change in the environment***.
  - No other agent, artificial or human.
- A2 : ***the environment is totaly observable, the agent perfectly knows it***.
  - The agent does not reason (e.g., plan) on things it does not know.
- A3 : ***the environment is static***.
  - Even if the environment can have behavior laws, it does not spontaneously move.

Planning Domain Definition Language (PDDL)

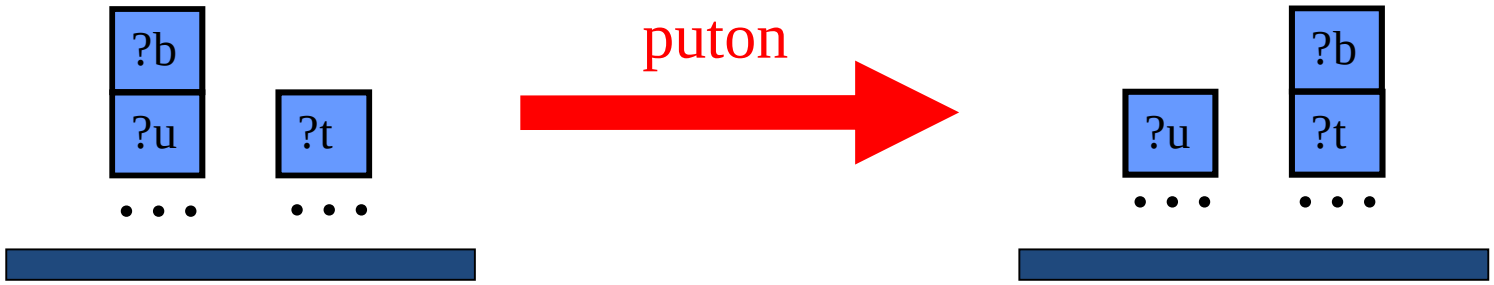
- Representation language which defines:
  - a domain: operators
  - a problem: state and goals.
- An operator is composed of:
  - **Pre-conditions:** terms which must hold for the action to execute.
  - **Effects / post-conditions:** terms which the execution of the action changes when compared to the incoming state (ADD-LIST / DELETE-LIST).
    - A post-condition can be positive or negative.
- A term can be sometimes true and sometimes false, depending on the time at which it is considered in the plan.
  - Connector *« not ».* Ex. : (**not** (ON MOUSE PAD))
  - *« Fluent » (litteral).* Ex. : (NOT MOUSE PAD)

PDDL: Example of domain
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))



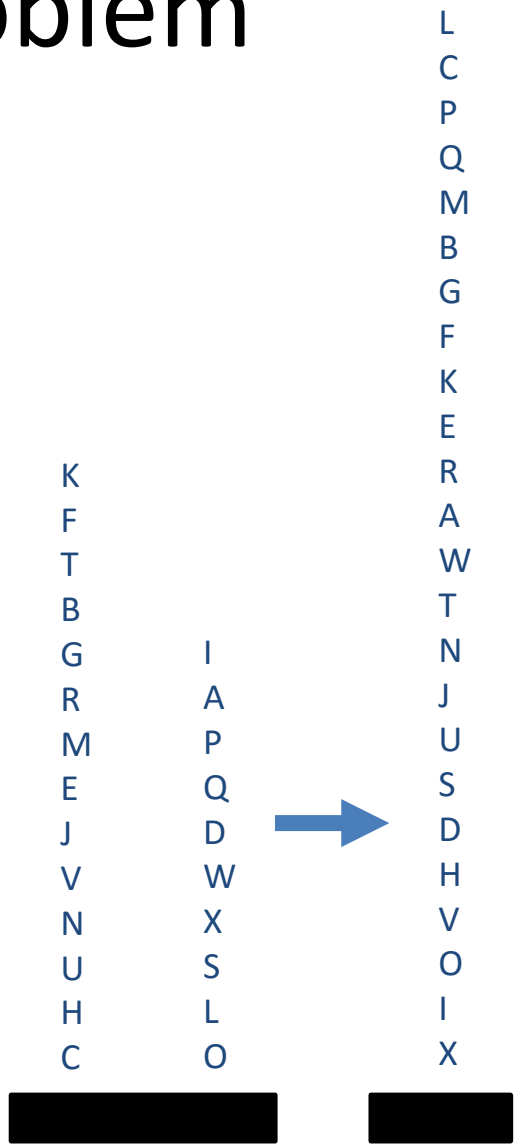
- What about the table ? And the arm ? What if several arms ? What if blocks have colors ? Or nicks ? Or multiple dimensions ? Conditionnals ? Universal quantification ?
- Qualification problem ; ramification problem.

Planning Domain Definition Language (PDDL)

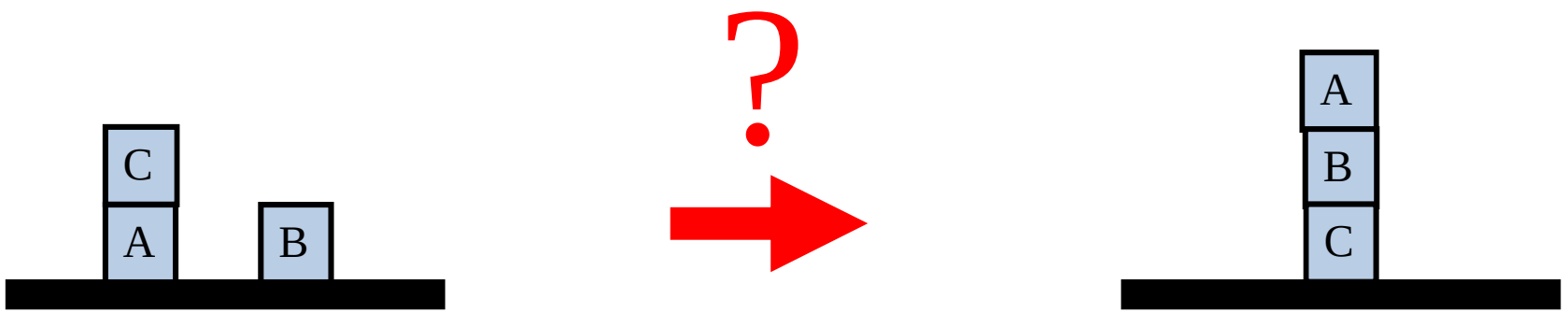
- **The frame problem:** when executing an operator, what is not explicitly changed is considered unchanged.
- **Closed world assumption:** in a state, a term not explicitly mentionned is considered to be *false*.
  - As opposed to the *open world assumption* (ontologies) : *unknown*.

PDDL: Example of problem

```
(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))))
```



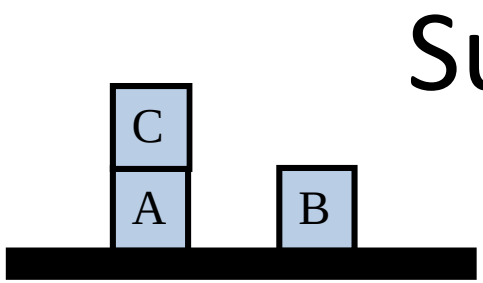
The anomaly of Gerald Jay Sussman (1/16)

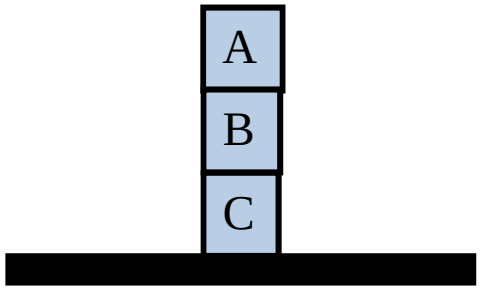


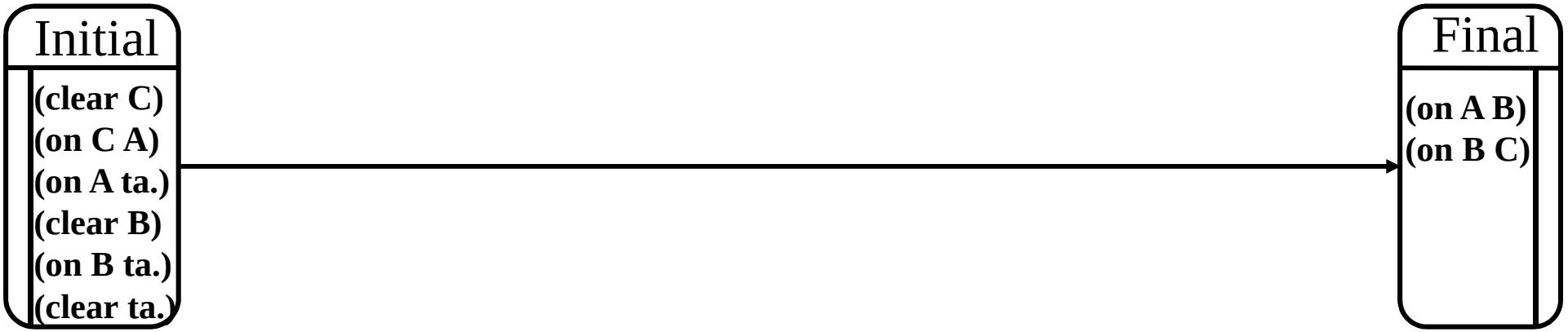
with:

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)))

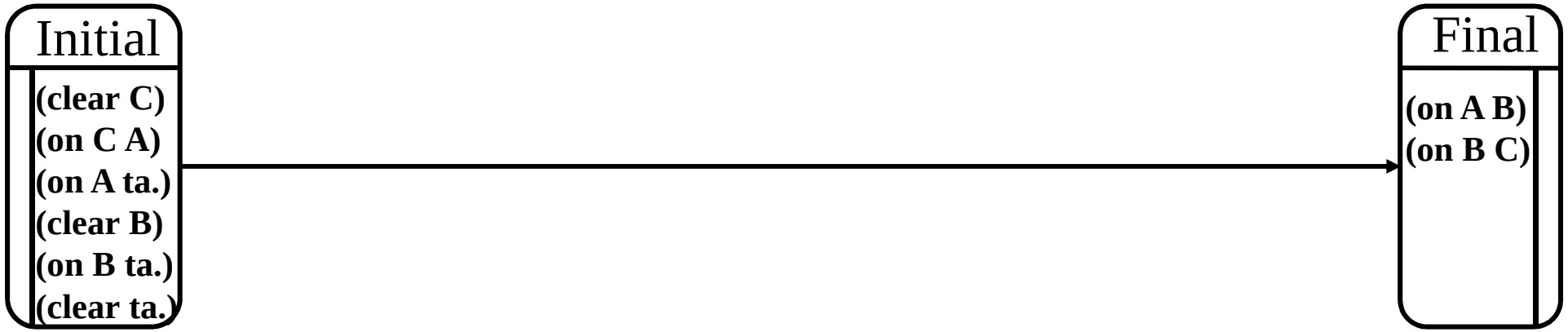
The anomaly of Gerald Jay Sussman (2/16)



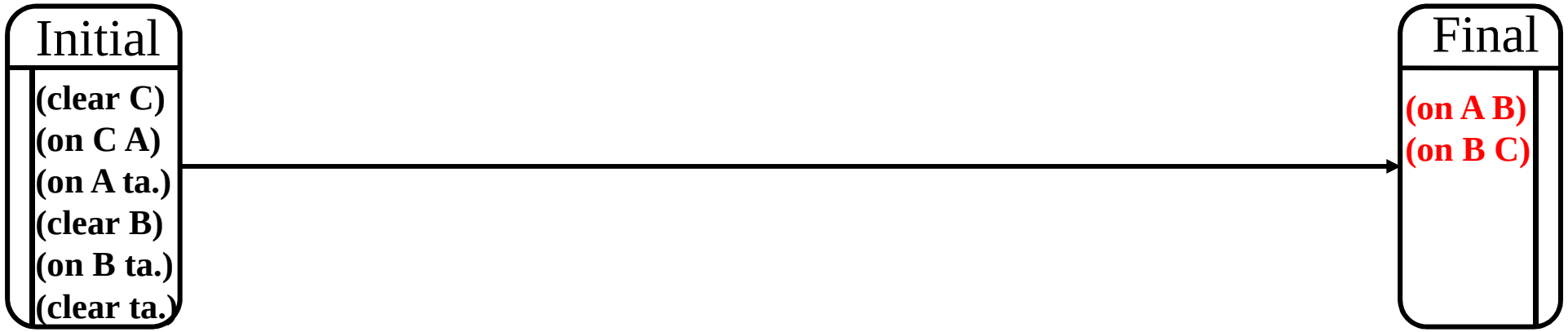




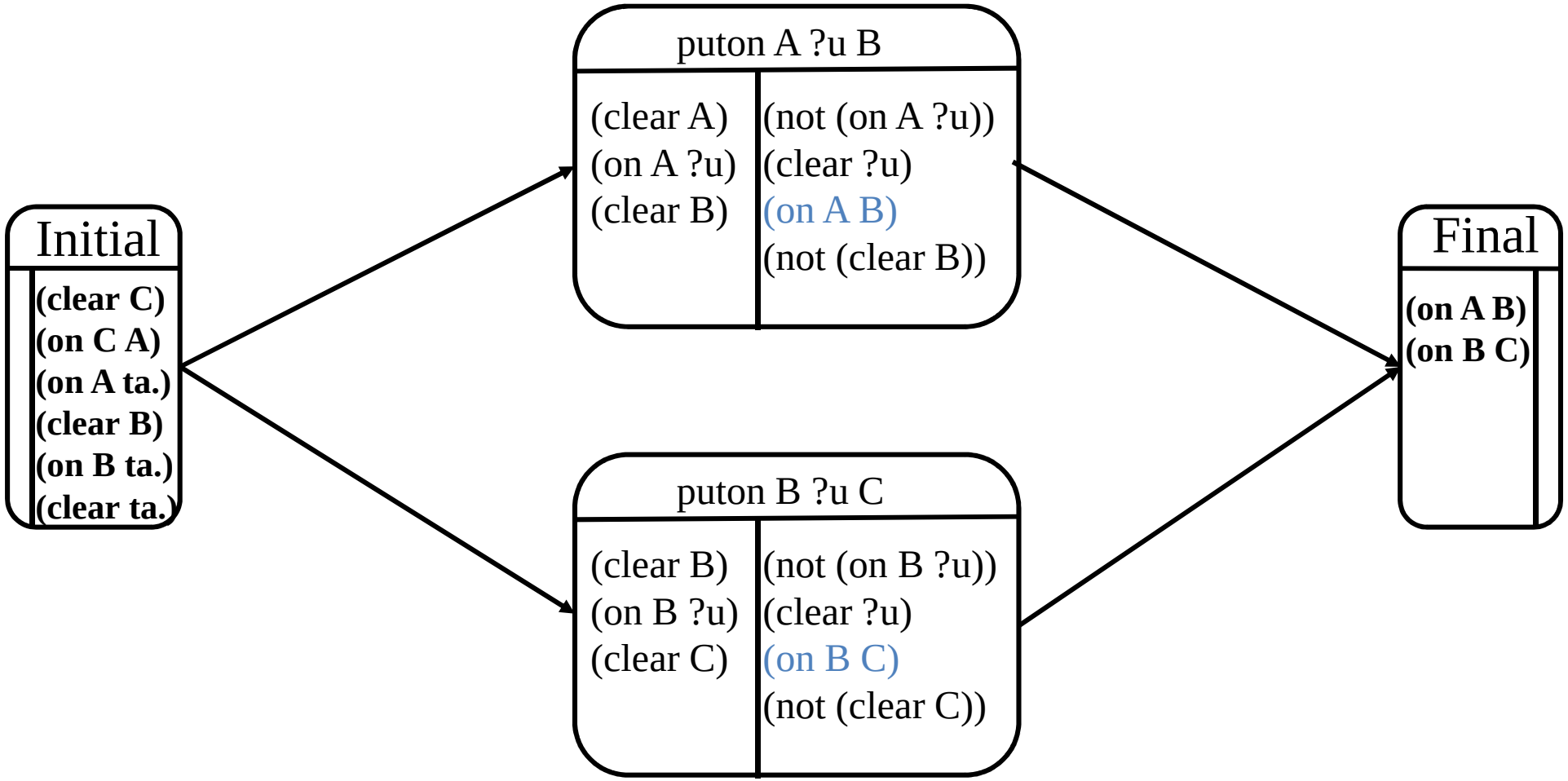
The anomaly of Gerald Jay Sussman (2/16)



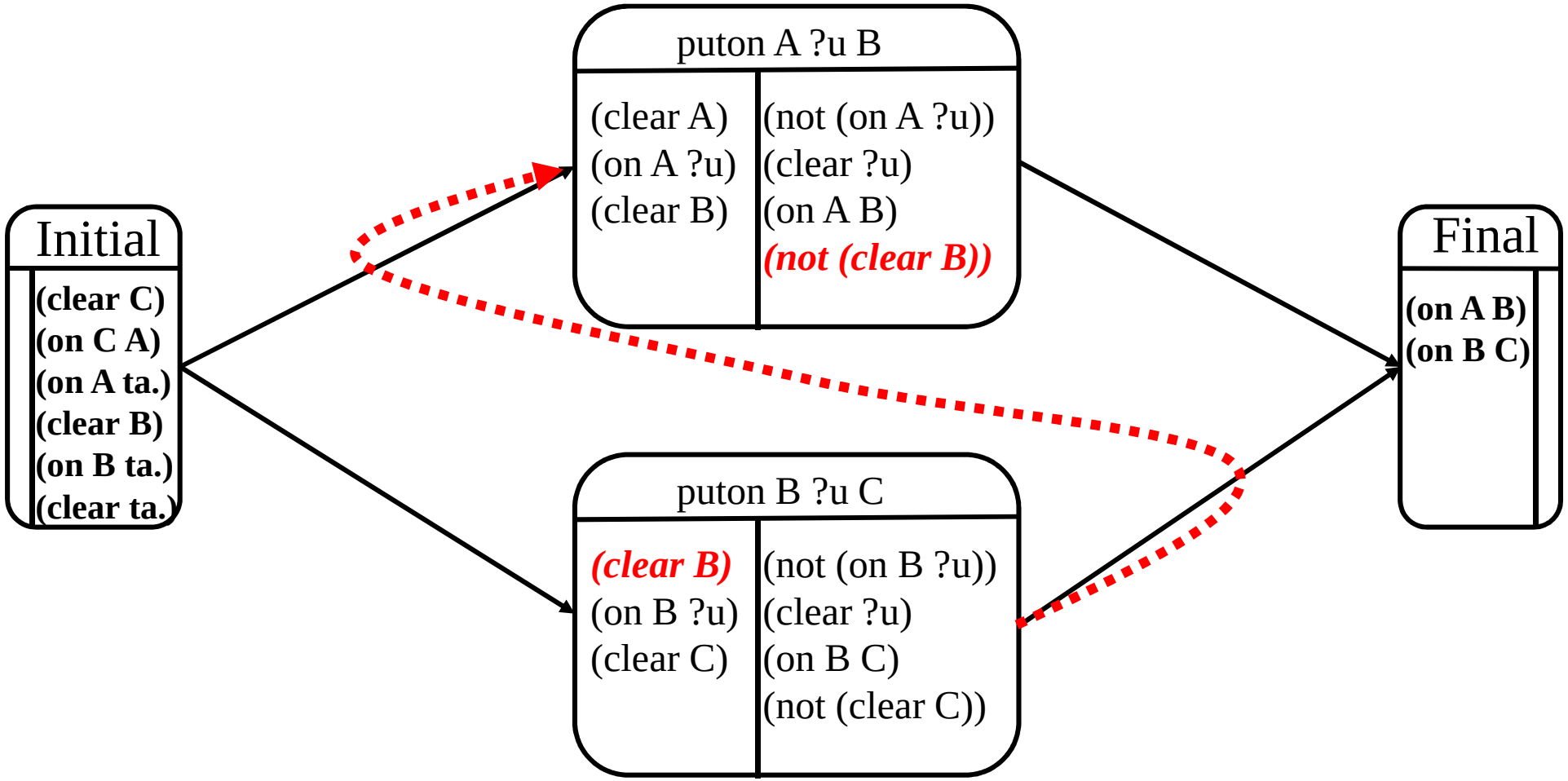
The anomaly of Gerald Jay Sussman (3/16)



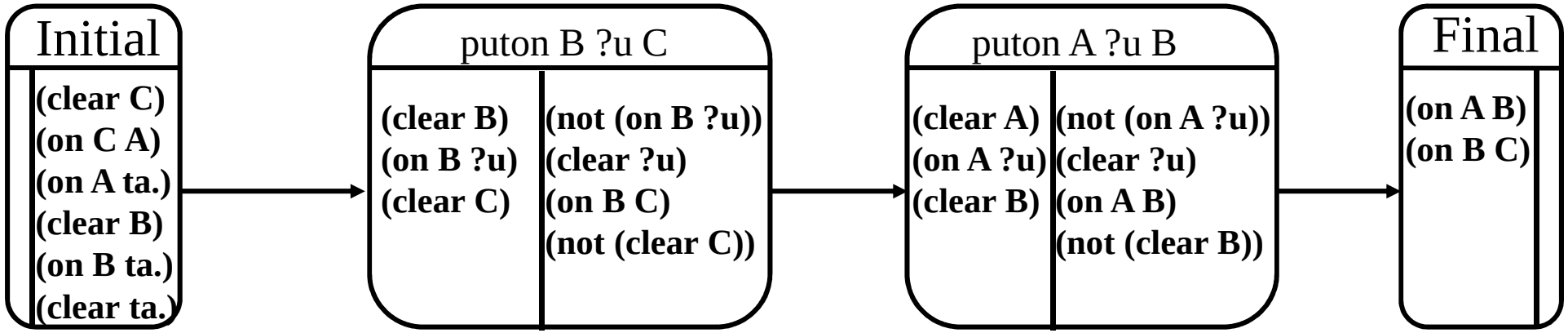
The anomaly of Gerald Jay Sussman (4/16)



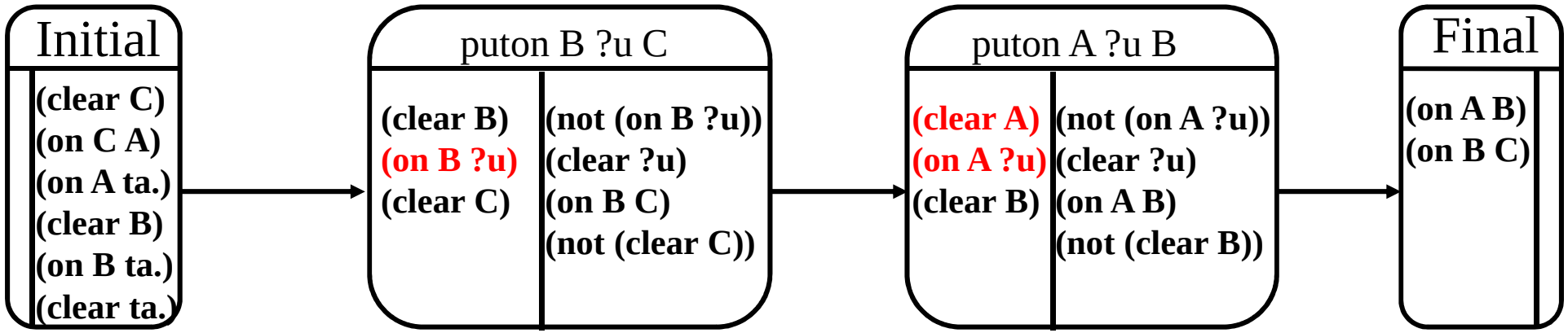
The anomaly of Gerald Jay Sussman (5/16)



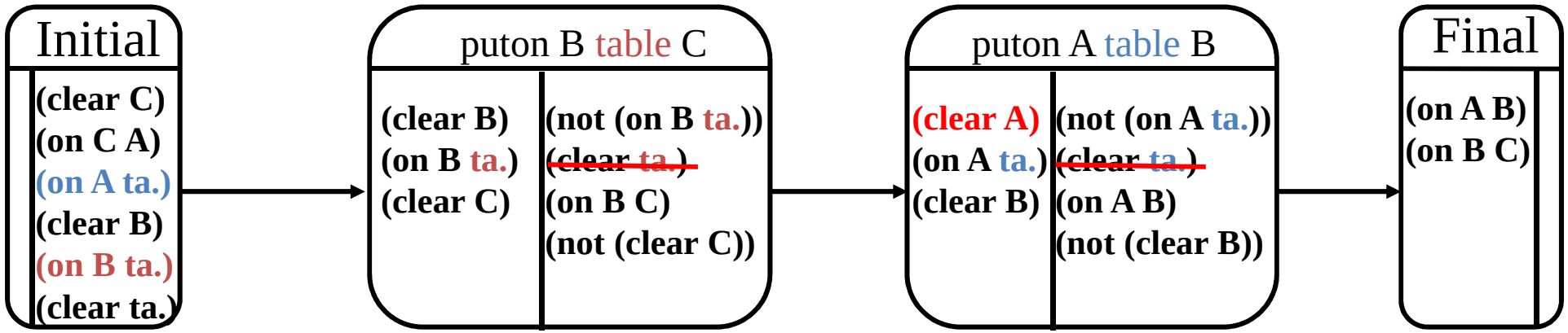
The anomaly of Gerald Jay Sussman (6/16)



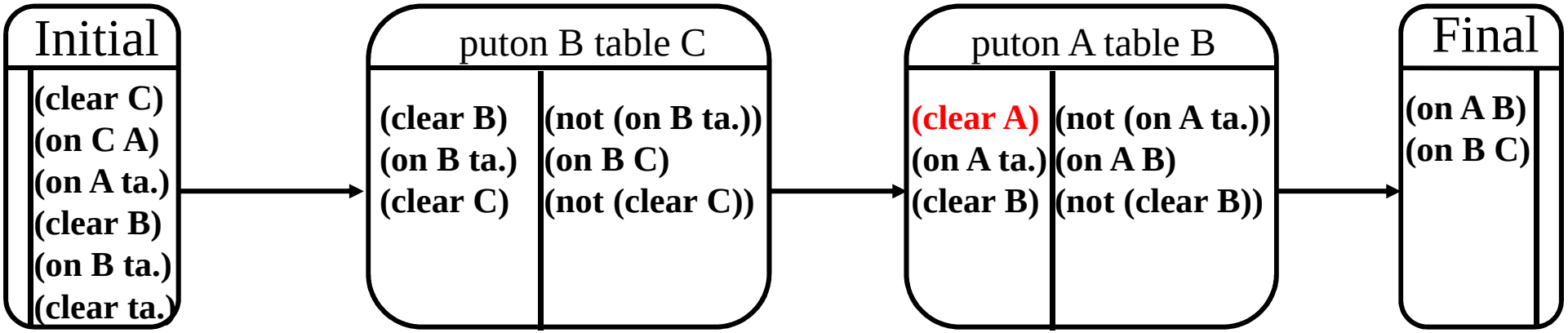
The anomaly of Gerald Jay Sussman (7/16)



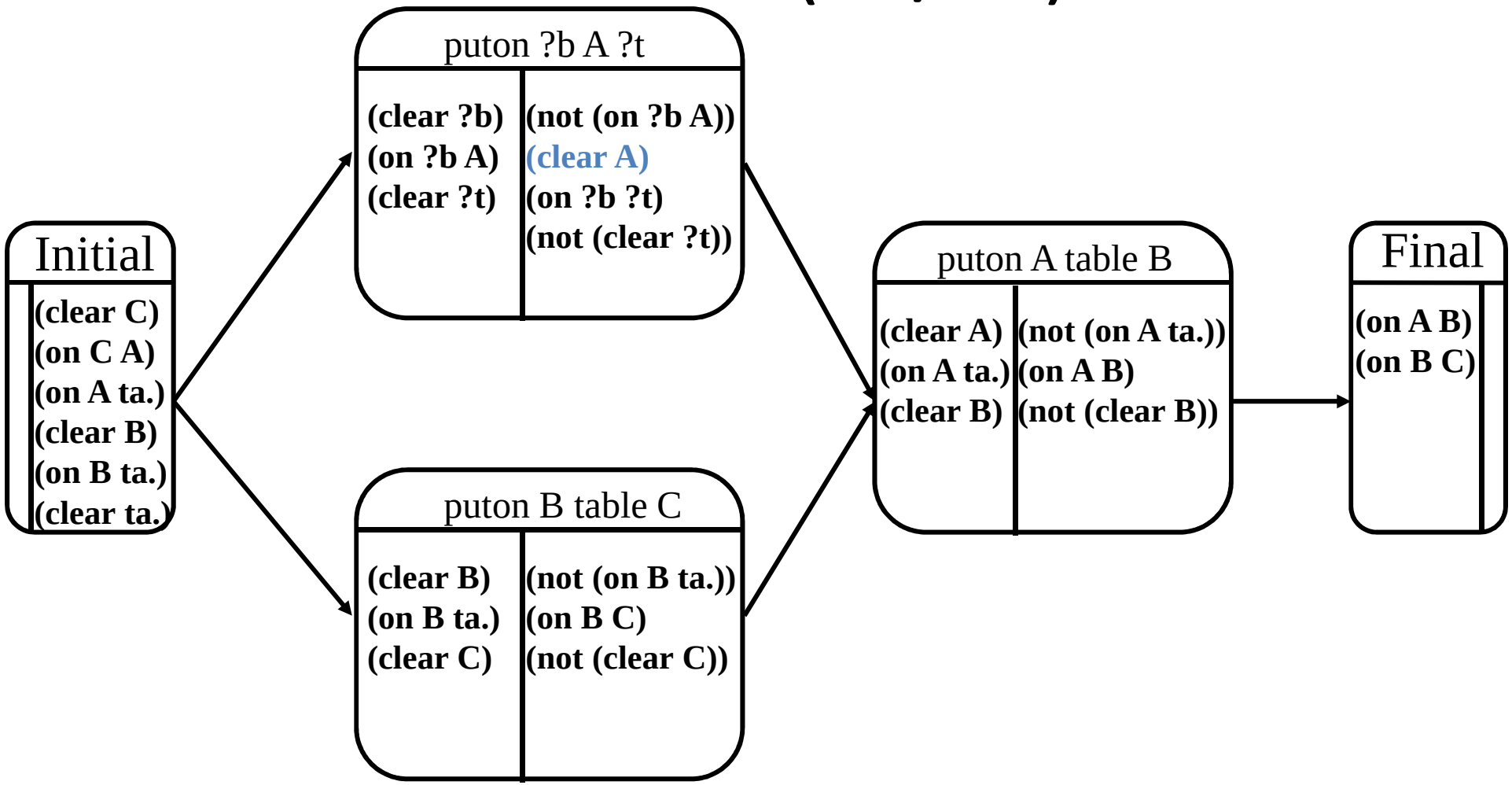
The anomaly of Gerald Jay Sussman (8/16)



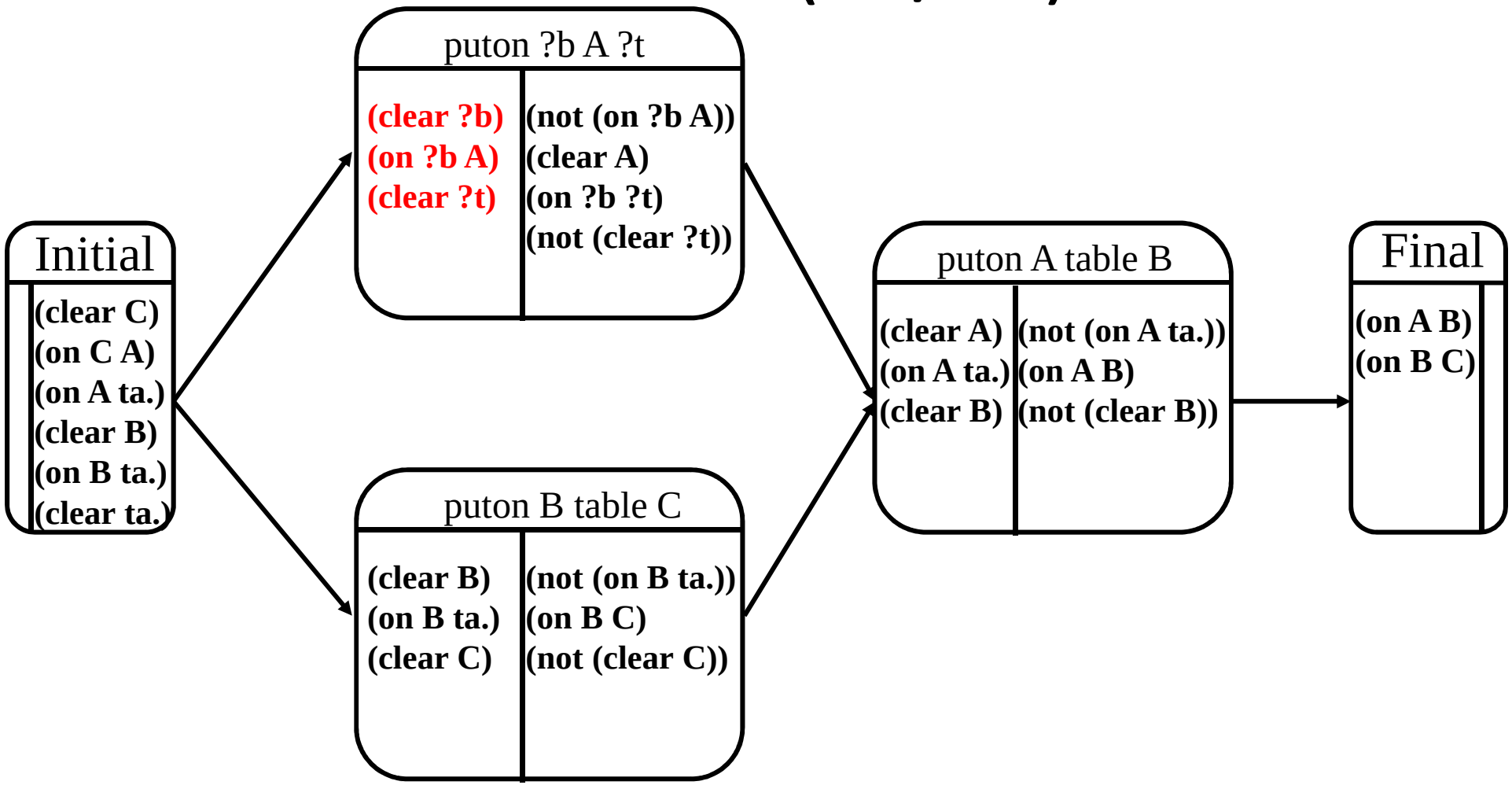
The anomaly of Gerald Jay Sussman (9/16)



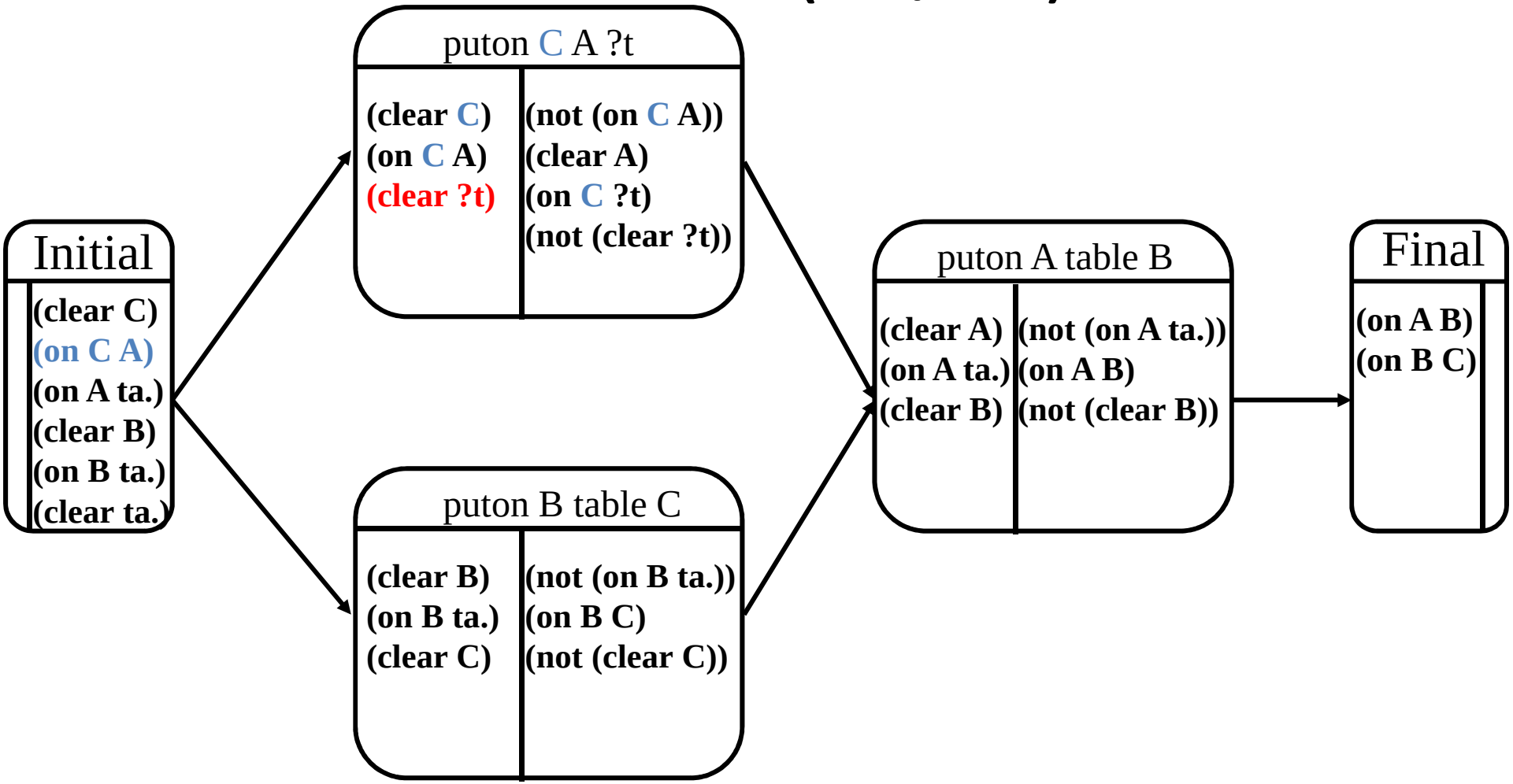
The anomaly of Gerald Jay Sussman (10/16)



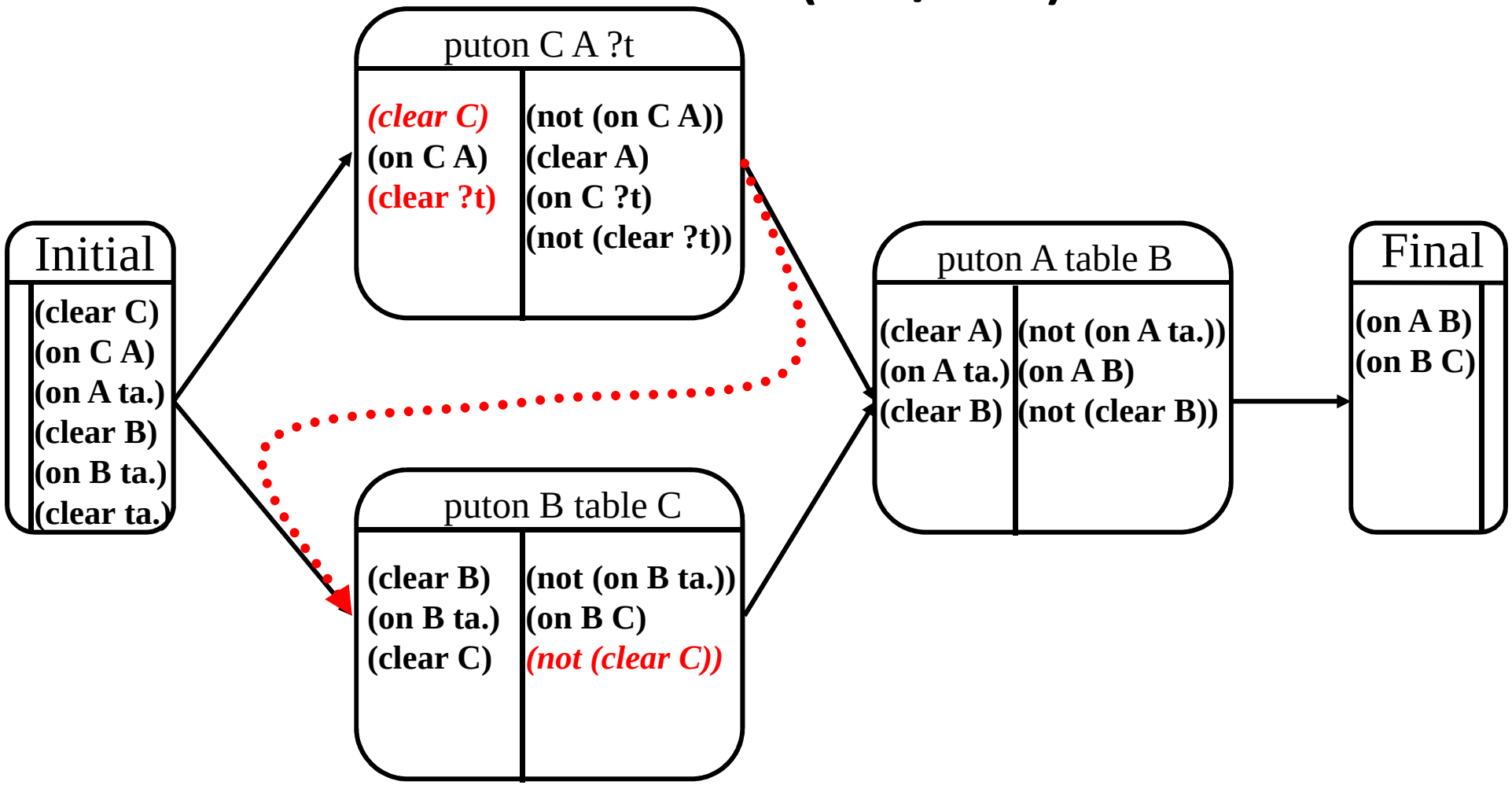
The anomaly of Gerald Jay Sussman (11/16)



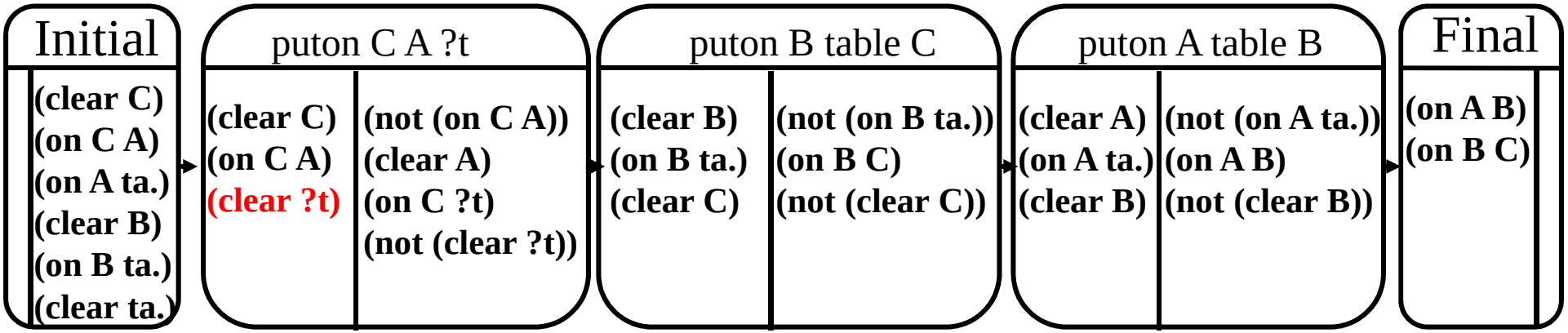
The anomaly of Gerald Jay Sussman (12/16)



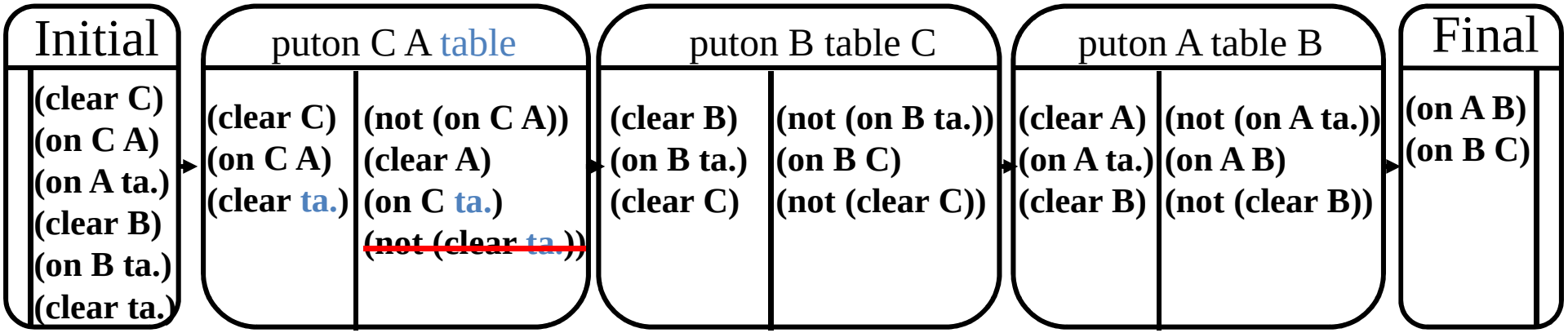
The anomaly de Gerald Jay Sussman (13/16)



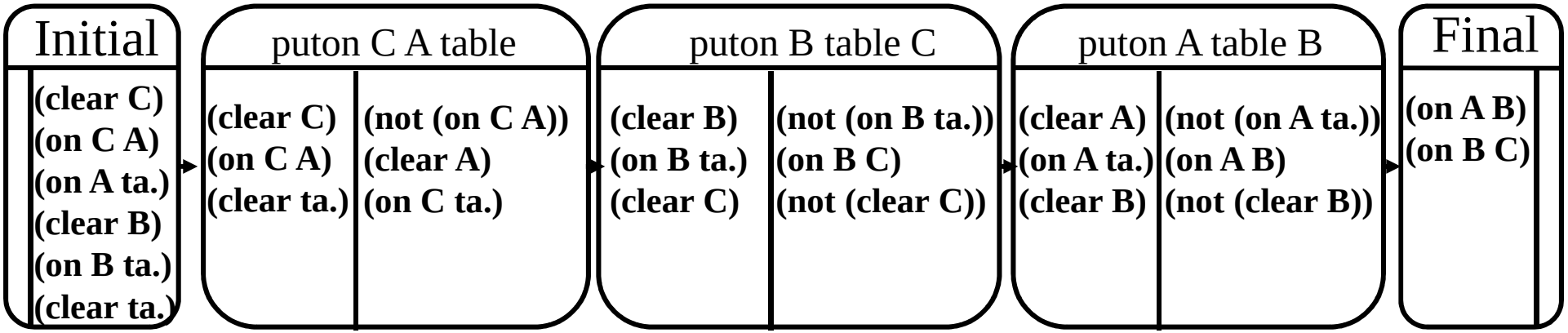
The anomaly of Gerald Jay Sussman (14/16)



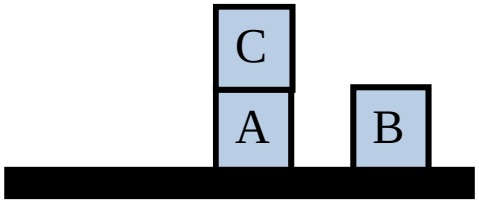
The anomaly of Gerald Jay Sussman (15/16)



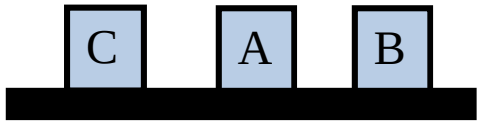
The anomaly of Gerald Jay Sussman (16/16)



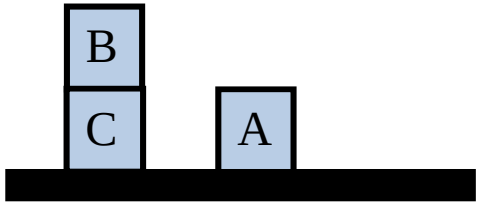
The anomaly of Gerald Jay Sussman : solution



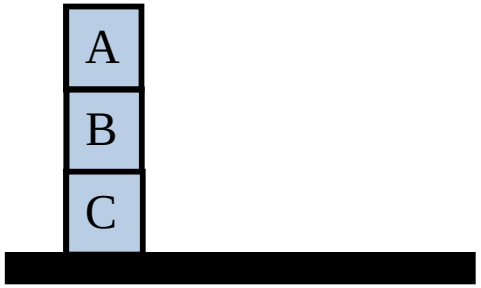
(1)



(2)



(3)

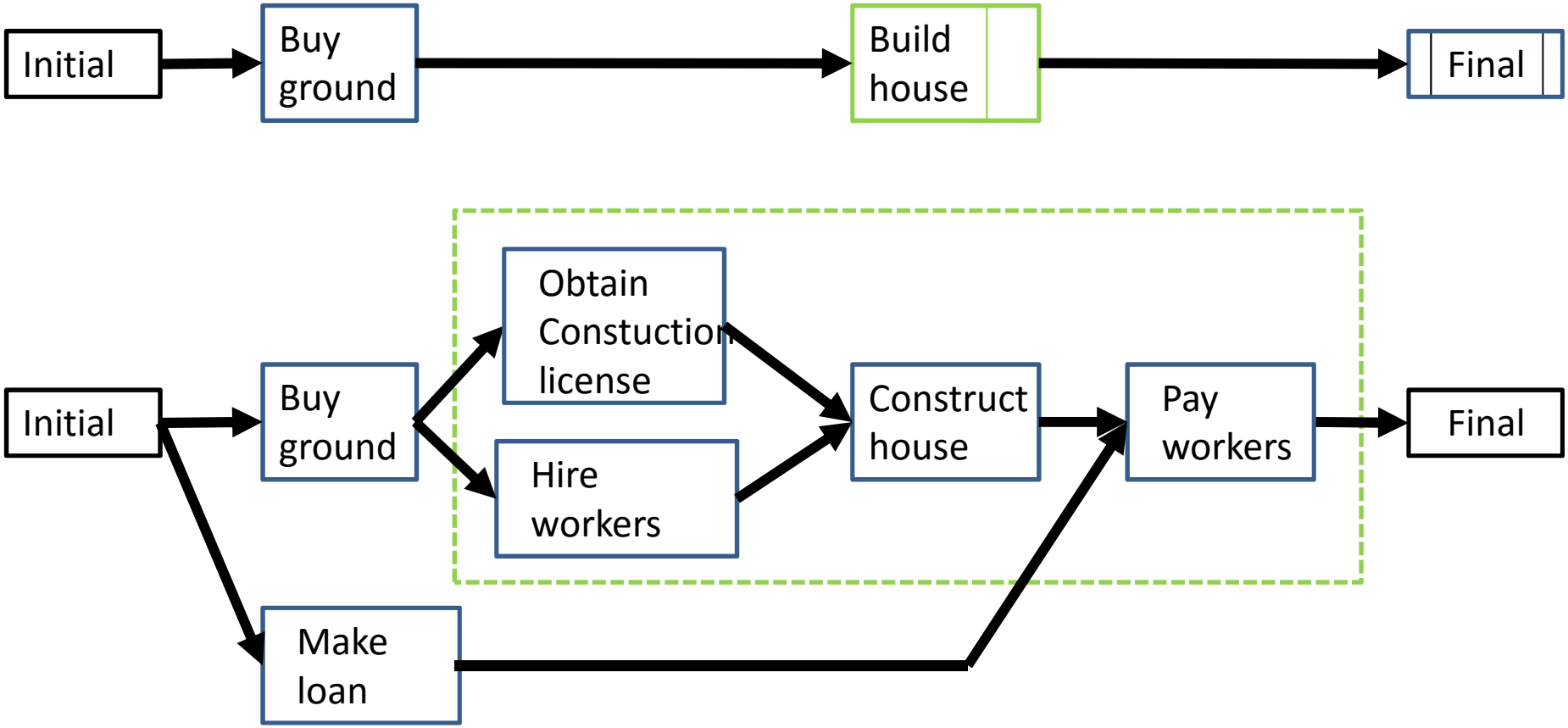


(4)

Planners …

- Planners using forward search in a state space (Jorg Hoffman, Hector Geffner).
- Planners using backward search in a state space (Malte Helmert).
- Planners using (forward) search in a plan space (Anthony Barrett).
- Planners using evolutionnary algorithms (Marc Schoenauer)
- Planners using temporal logic (Patrick Doherty).
- Planners using constraint programming (Vincent Vidal).
- Planners using SAT solvers (Henry Kautz & Bart Selman, Jussi Rintanen).
- Planner using mixed integer programming (Dana Nau).
- Planner using hierarchical task networks (Dana Nau).

Hierarchical task networks

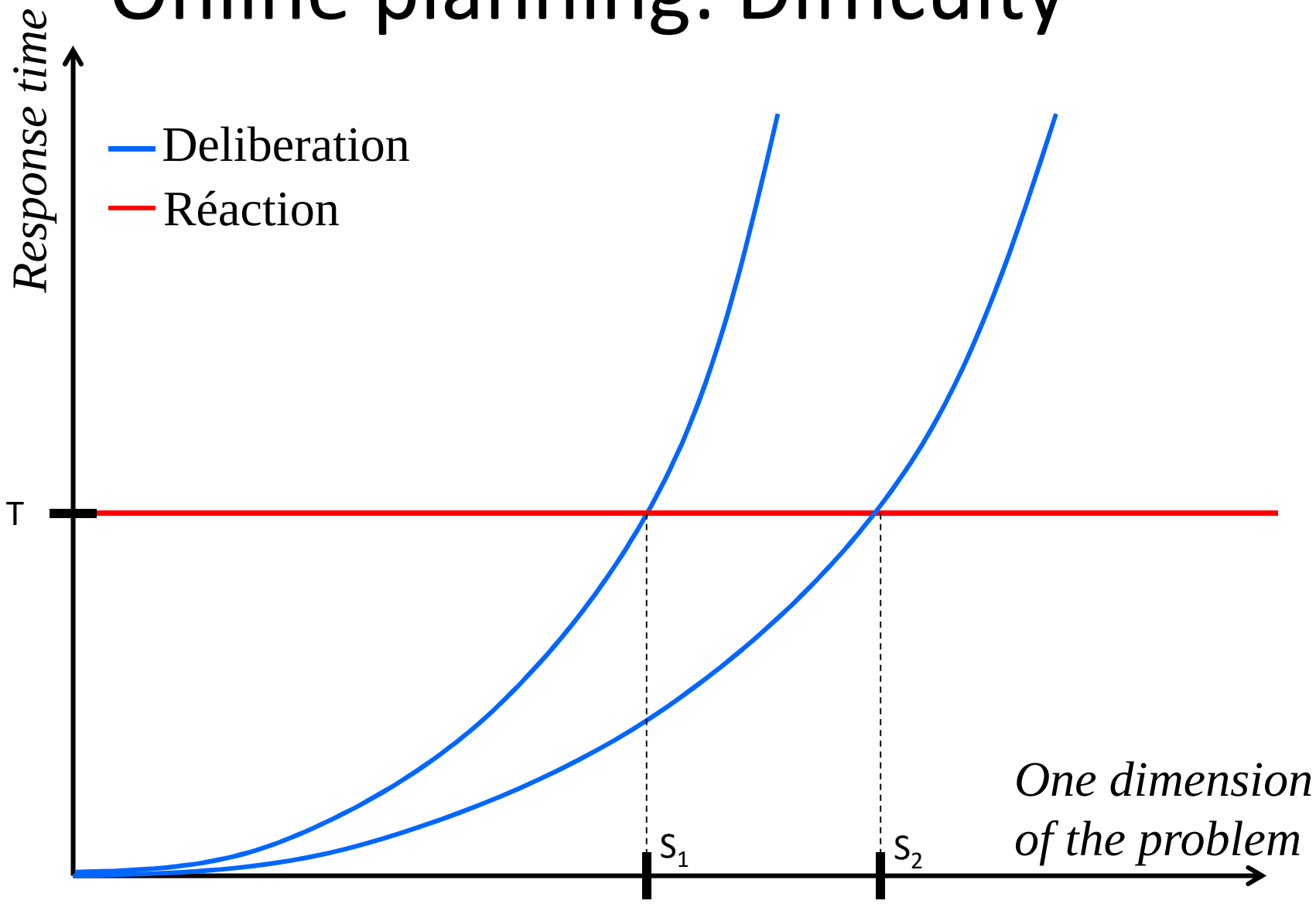


Applications (1 / 2)

- Advise a worker to disassemble a car engine (NOAH, Earl Sacerdoti 1974)
- Organize the logistics of the military invasion of Iraq during the first Gulf War (SIPE, David Wilkins, 1980).
- Reactivate the electronics components of a spatial probe cruising around Jupiter (2000).
- Debug a xerox machine.
- Determine the actions of characters in a video game (Eric Jacopin, 2008).
- Interactive story telling (Marc Cavazza, 2010).

Applications (2 / 2)

Online planning: Difficulty



References

- **[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 11.**
- **[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://ipc.informatik.uni-freiburg.de/PddlExtension**
- **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**
- **Journaux :**
  - **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/**

Conclusion

- Action planning consists of finding a sequence of instantiated actions (a plan of operators) which provably leads an initial state to a (final) state containing predefined goals.
  - Difficult because interaction among actions and combinatorial explosion.
- Operators are expressed in the Planning Domain Definition Language (PDDL) and are composed of pre-conditions and post-conditions.
- Several approaches to implement an action planner.
- Planning while executing is online planning: a fast reaction time is required whereas action planning is a combinatorial problem.