

Class 7: POP; Uncertainty; Creativity

Selmer Bringsjord

June 8, 1999

- Logistics:
 - Midterms back soon
 - Project 2 due for studio audience next class
 - Questions? Comments? Issues?
 - * grading re. packages
 - Student web page *browser*

- Today:
 - POP
 - Uncertainty and OSCAR
 - Creativity and BRUTUS

```

? (substitute '(a b c) (match '(?x ?y ?z) '(a b c)))
(A B C)
? (match '(?x ?y ?z) '(a b c))
((?Z C) (?Y B) (?X A) (T T))
? (substitute '(?x b c) '((?x a)))
(A B C)
? (substitute '(?x b ?y) '((?x a) (?y f)))
(A B F)
? (substitute '(?x (a b) ?y) '((?x a) (?y f)))
(A (A B) F)
? (substitute '(?x (a ?z) ?y) '((?x a) (?y f) (?z e)))
(A (A E) F)
? (rhs '((a b c) (d e f)))
(D E F)
? (lhs '((a b c) (d e f)))
(A B C)
? (apply-rule '(I am depressed)
'((I am ?x) (Why are you ?x ?)))
(WHY ARE YOU DEPRESSED ?)
? (apply-rule '(Man Socrates)
'((Man ?x) (Mortal ?x)))
(MORTAL SOCRATES)
? *grammar-rules*
(((JOHN ?V ?N) ((N JOHN)...
? (apply-rules '(John loves Mary) *grammar-rules*)
(S (N JOHN) (VP (V LOVES) (N MARY)))
? (match '(a b c) '(a b c))
((T T))

```

POP

A **plan** \mathcal{P} is a quadruple $(\mathcal{S}, \mathcal{O}, \mathcal{B}, \mathcal{L})$ where

- $\mathcal{S} = \{S_1, \dots, S_n\}$
- $\mathcal{O} = \{S_{i_1} \prec S_{j_1}, S_{i_2} \prec S_{j_2}, \dots, S_{i_m} \prec S_{j_m}\}$
- $\mathcal{B} = \{v_{p_1} = x_{q_1}, v_{p_2} = x_{q_2}, \dots\}$
- $\mathcal{L} = \{S_i \xrightarrow{c} S_j, \dots\}$

What are we looking for?

solution(\mathcal{P}) iff *complete*(\mathcal{P}) \wedge *consistent*(\mathcal{P}),
where

complete(\mathcal{P}) iff
 $\forall S_j \forall c (c \in \text{PRECOND}(S_j) \Rightarrow$
 $\exists S_i (c \in \text{EFFECTS}(S_i) \wedge S_i \prec S_j \wedge$
 $\neg \exists S_k (\neg c \in \text{EFFECTS}(S_k) \wedge$
 $S_i \prec S_k \prec S_j \text{ is a linearization of } \mathcal{P})))$.

consistent(\mathcal{P}) iff
 there are no contradictions in \mathcal{O} and \mathcal{B} .

- Checking this out on the shopping example . . .
- How challenging is it to check for inconsistency?

Uncertainty and OSCAR

- **Defeasible logic** (or revisable, or nonmonotonic, . . . logic) arises from **monotonicity** in FOL:
 - if α is provable from KB, then adding new information to KB, no matter what that information is, never sees to the retraction of $\text{KB} \vdash \phi$

- Our everyday reasoning **isn't** monotonic:

Nearly all of you will at one time or another have affirmed the proposition that birds can fly. Expressed in FOL, this fact could be captured by

$$(1) \forall x(Bx \rightarrow Fx).$$

You will also have confronted the fact that ostriches **can't** fly. Clearly, there is a need to revise what had earlier been believed. But just as clearly, it's very implausible that humans, in everyday reasoning, employ modifications of (1) like

$$\forall x((Bx \wedge \neg Ox) \rightarrow Fx),$$

because, for starters, there are an infinite number of exceptions to (1). The solution, in general, must (might?) be that we use a system that allows us to make *defeasible* or *revisable* inferences. Defeasible logics mark attempts to capture such systems.

- OSCAR is designed to handle tough problems involving uncertainty, e.g.,

The Lottery Paradox

Suppose you hold one ticket (t_k , for some $k \geq 1$) in a fair lottery consisting of 1 million tickets, and suppose it is known that one and only one ticket will win. Since the probability is only .000001 of t_k 's being drawn, it seems reasonable to believe that t_k will not win. By the same reasoning it seems reasonable to believe that t_1 will not win, that t_2 will not win, ..., that $t_{1000000}$ will not win. Therefore it is reasonable to believe

$$\neg \exists t_i (t_i \text{ will win}).$$

But we know that

$$\exists t_i (t_i \text{ will win}).$$

So we have an outright contradiction.

- And ... the Paradox of the Preface
...
- How would you solve these problems using machinery gradually coming under your control in this course?
- Tackling this↑ a good way to learn
- The “guts” of OSCAR: only if you’re interested
- But let’s look at some pictures ...
- Comparing belief networks etc. on the hard cases

- Creativity, Chess & S^3G
 - defining creativity . . .
 - n-queens problem *Ralph*
 - Selmer's Challenge to Ralph: S^3G
 - JoyceBot2
 - Brutus v. Selmer in S^3G
 - BRUTUS.1 *Browser*
 - BRUTUS.1 & Betrayal . . .

Def_B 1 Agent s_r betrays agent s_d iff there exists some state of affairs p such that

1 s_d wants p to occur;

2 s_r believes that s_d wants p to occur;

4 s_r intends that p *not* occur;

5 s_r believes that s_d believes that s_r intends that p occur.

Problem: no need to *do* anything

Def_B 2 Agent s_r betrays agent s_d iff there exists some state of affairs p such that

- 1** s_d wants p to occur;
- 2** s_r believes that s_d wants p to occur;
- 3** s_r agrees with s_d that p ought to occur;
- 4** s_r intends that p *not* occur;
- 5** s_r believes that s_d believes that s_r intends that p occur.

Problem: Suppose that Horace wants President Clinton to make a trip to Moscow; and suppose as well that Joe believes that Horace wants Clinton to make this trip, and that Joe agrees with Horace that Clinton ought to go. However, assume in addition that Joe intends that Clinton not go — *but takes no action toward that end*. In this case it seems that since Joe does nothing (relevant), even if Clinton fails to go, there is no betrayal in the picture.

Def_B 5 Agent s_r betrays agent s_d iff there exists some state of affairs p such that

- 1** s_d wants p to occur;
- 2** s_r believes that s_d wants p to occur;
- 3** s_r agrees with s_d that p ought to occur;
- 4'** there is some action a which s_r performs in the belief that thereby p will *not* occur;
- 5'** s_r believes that s_d believes that there is some action a which s_r performs in the belief that thereby p *will* occur;
- 6'** s_d wants that there is some action a which s_r performs in the belief that thereby p *will* occur.

Def_C 1 Agent s is P-creative with respect to ϕ at t if and only if there is a time t' prior to t and knowledge-bases Φ_s and Φ'_s such that

1 $\Phi_s \not\vdash \phi$ at t' ;

3 $\Phi'_s \vdash \phi$ at t , where s changes Φ_s to Φ'_s at some time t'' later than t' but not later than t .

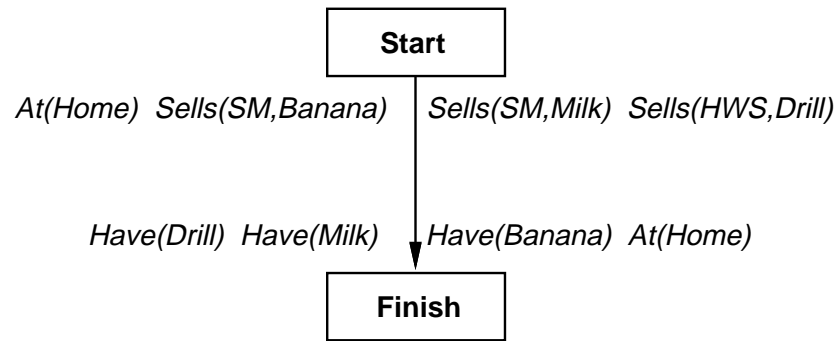


Figure 1: Initial Plan, Shopping.

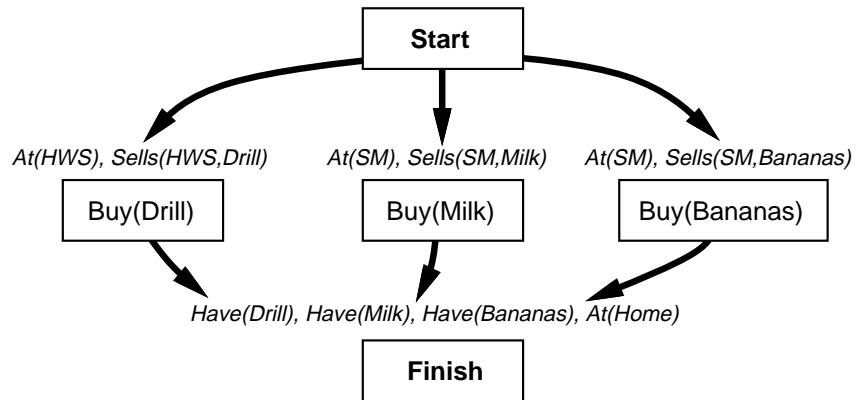
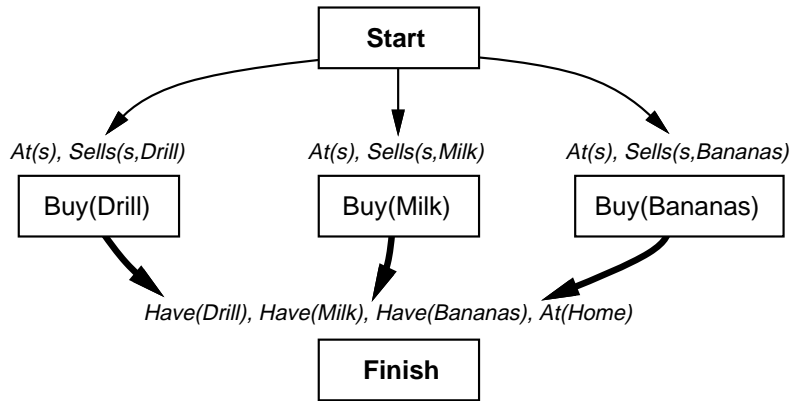


Figure 2: Partial Plan, Shopping.

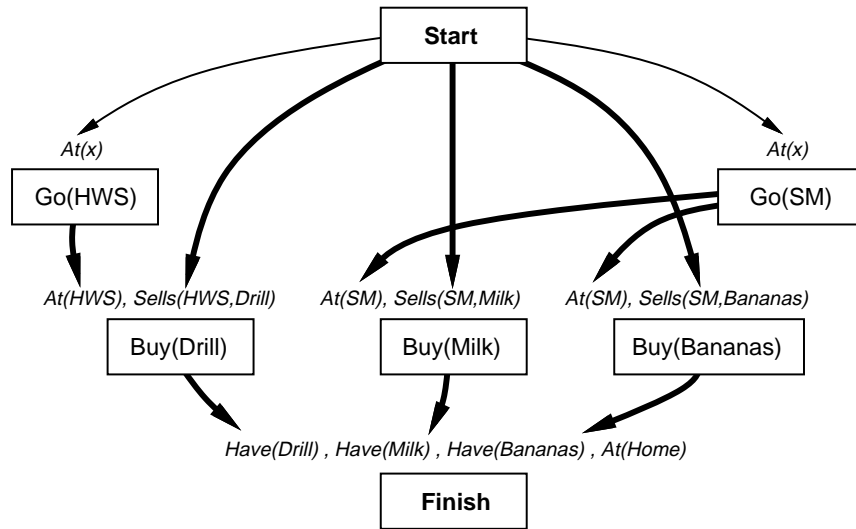


Figure 3: Partial Plan Achieves At Preconditions.

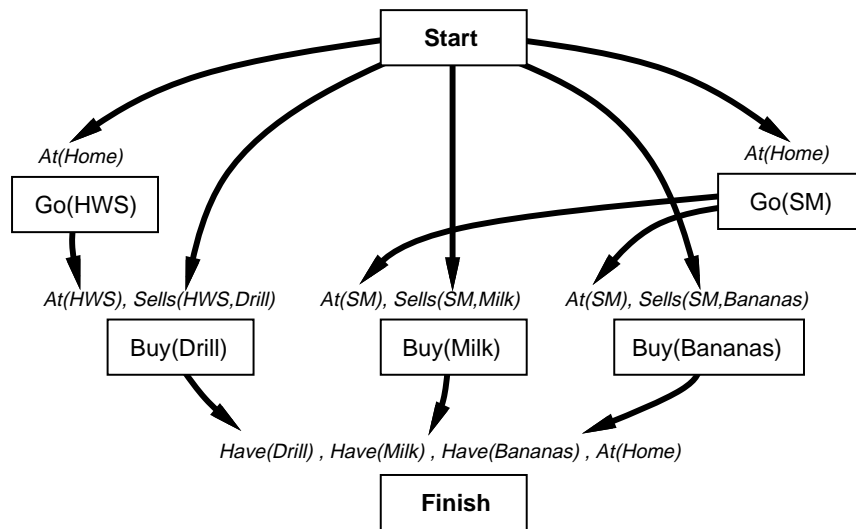


Figure 4: Flawed Plan.

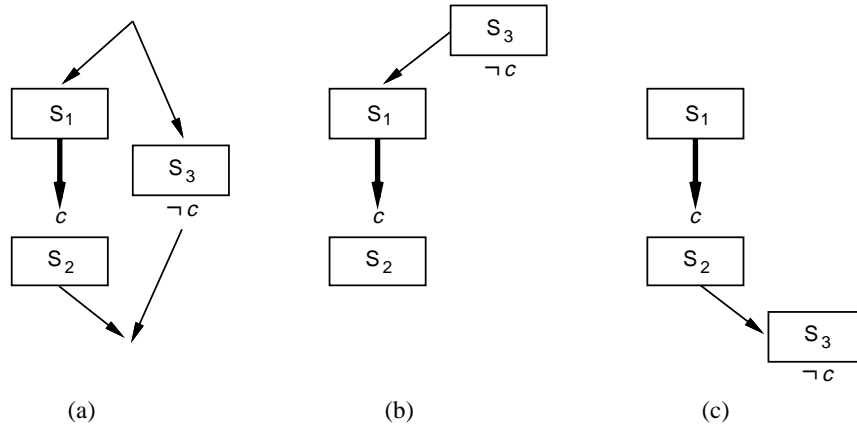


Figure 5: Protecting Causal Links.

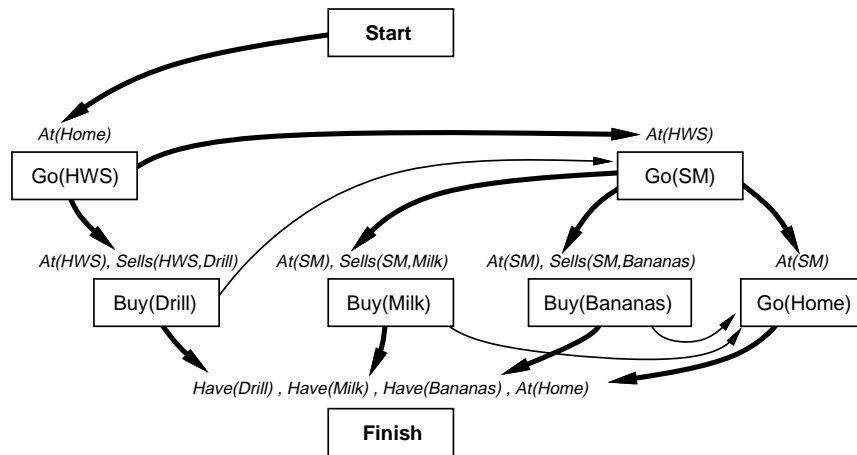


Figure 6: Causal Link Protection, Shopping.

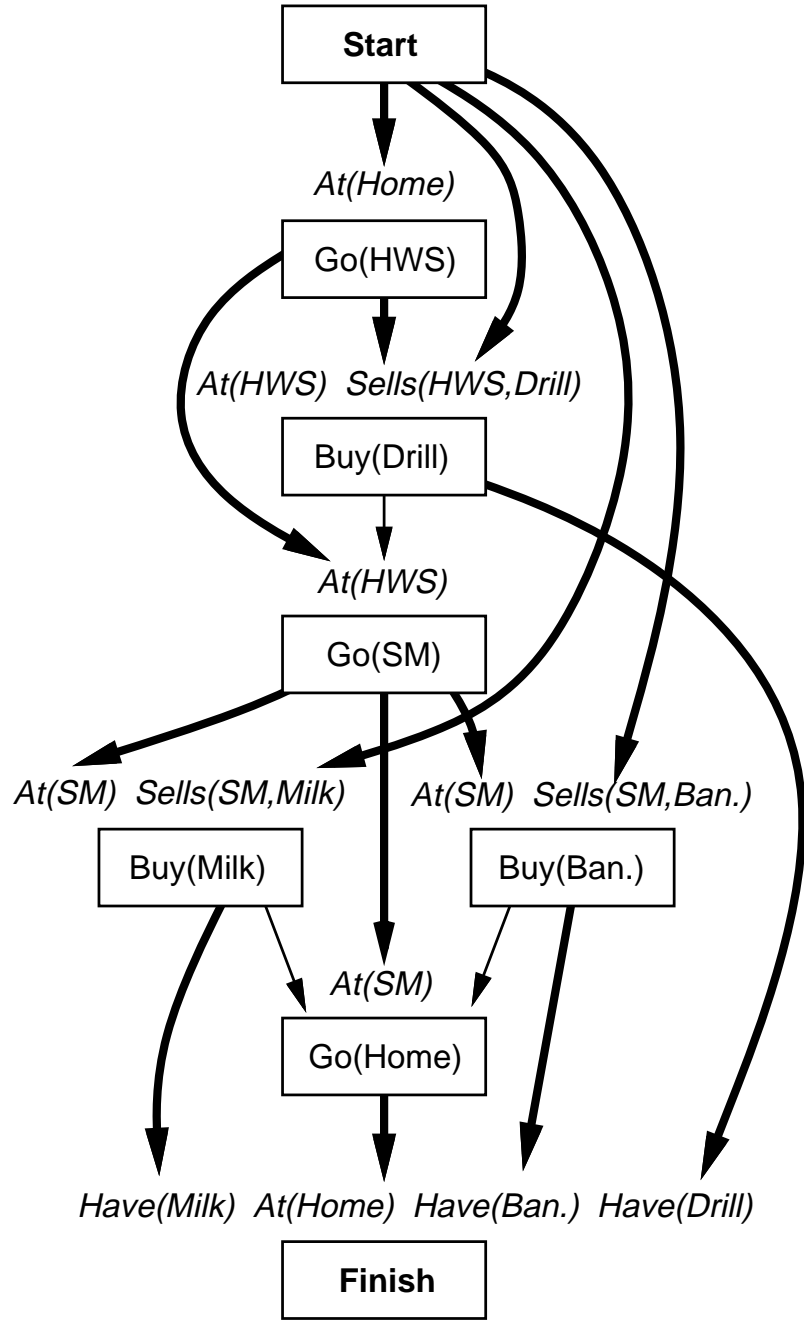


Figure 7: Solution, Shopping.

```

function POP(initial, goal, operators) returns plan

  plan ← MAKE-MINIMAL-PLAN(initial, goal)
  loop do
    if SOLUTION?(plan) then return plan
    Sneed, c ← SELECT-SUBGOAL(plan)
    CHOOSE-OPERATOR(plan, operators, Sneed, c)
    RESOLVE-THREATS(plan)
  end

```

```

function SELECT-SUBGOAL(plan) returns Sneed, c

  pick a plan step Sneed from STEPS(plan)
  with a precondition c that has not been achieved
  return Sneed, c

```

```

procedure CHOOSE-OPERATOR(plan, operators, Sneed, c)

  choose a step Sadd from operators or STEPS(plan) that has c as an effect
  if there is no such step then fail
  add the causal link  $S_{add} \xrightarrow{c} S_{need}$  to LINKS(plan)
  add the ordering constraint  $S_{add} \prec S_{need}$  to ORDERINGS(plan)
  if Sadd is a newly added step from operators then
    add Sadd to STEPS(plan)
    add  $Start \prec S_{add} \prec Finish$  to ORDERINGS(plan)

```

```

procedure RESOLVE-THREATS(plan)

  for each Sthreat that threatens a link  $S_i \xrightarrow{c} S_j$  in LINKS(plan) do
    choose either
      Promotion: Add  $S_{threat} \prec S_i$  to ORDERINGS(plan)
      Demotion: Add  $S_j \prec S_{threat}$  to ORDERINGS(plan)
    if not CONSISTENT(plan) then fail
  end

```

Figure 8: POP.

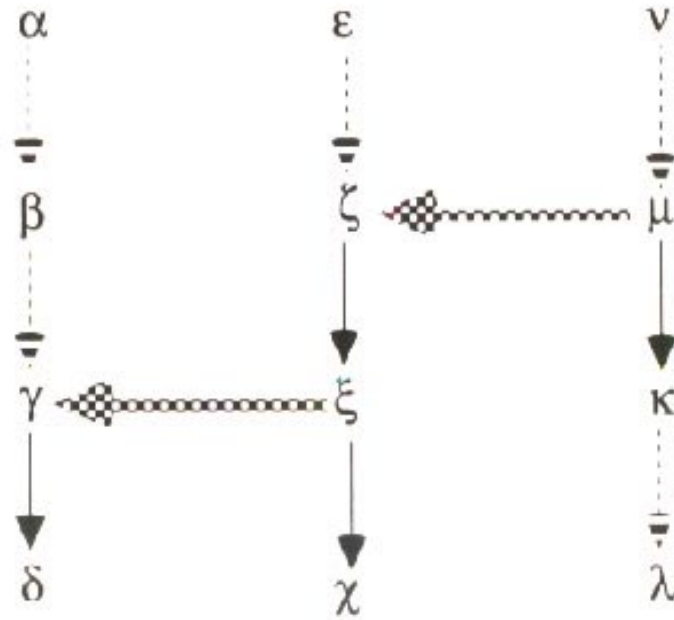
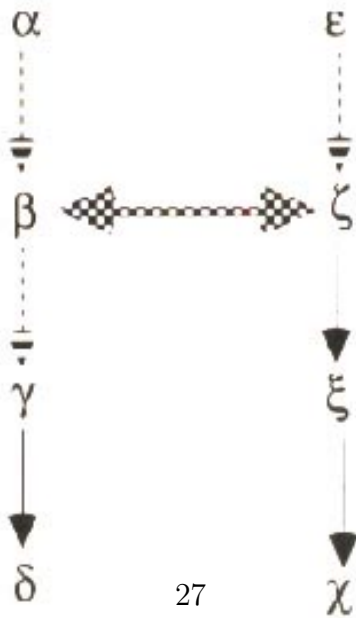


Figure 2. Inference graph



27

Figure 3. Collective defeat

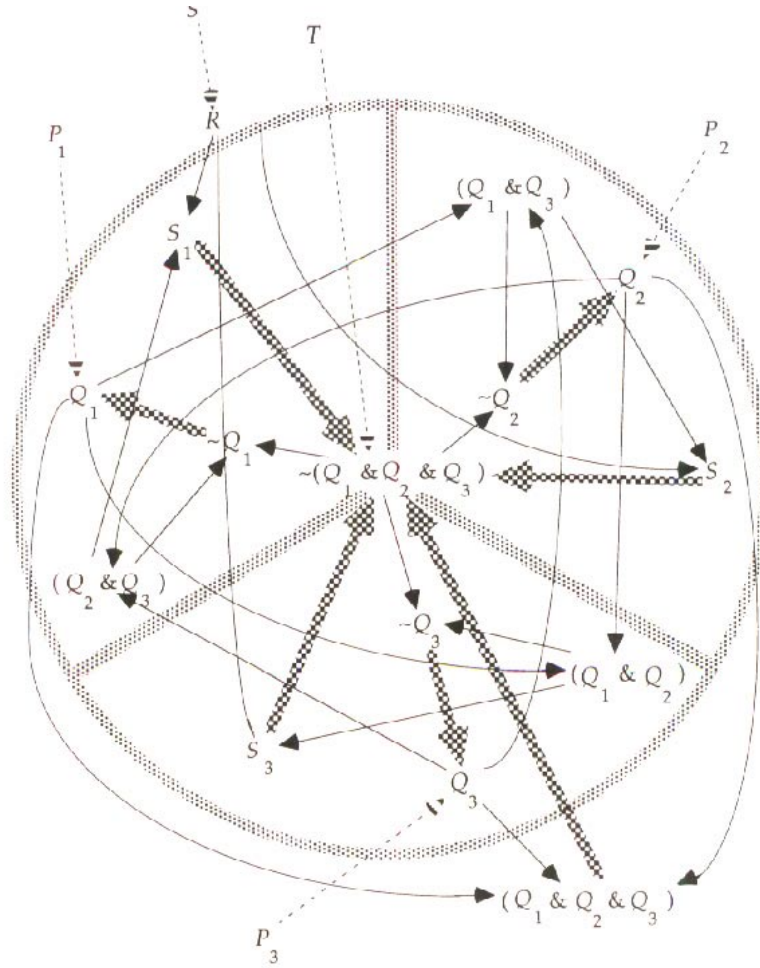


Figure 14. Structure of the paradox of the preface

Figure 10: Pollock 2.

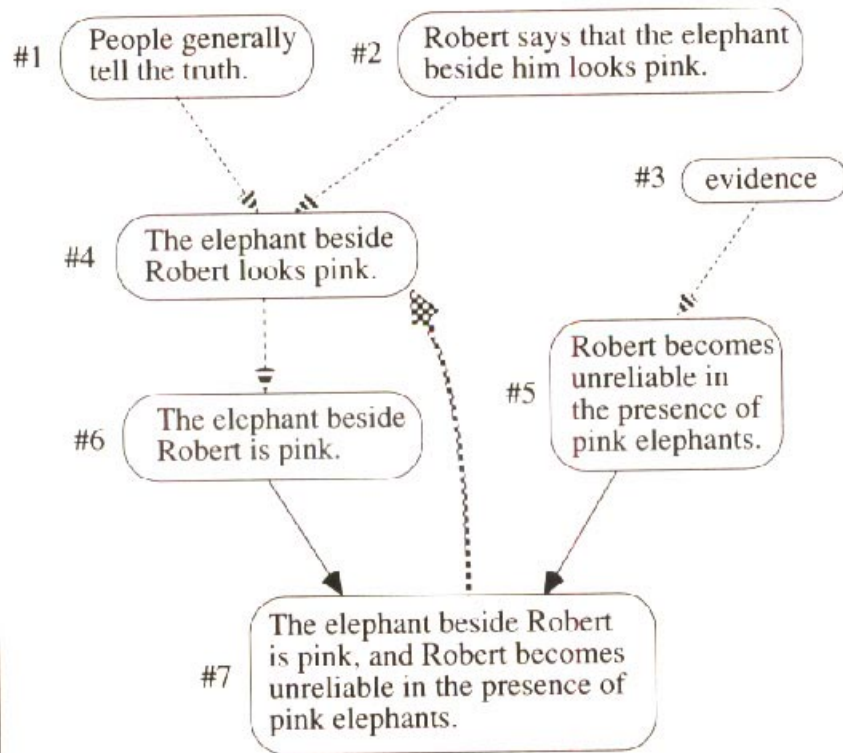


Figure 10. Argument with a self-defeating conclusion

Figure 11: Pollock 3.