Planning (05)

Planning graph

Alban Grastien
alban.grastien@rsise.anu.edu.au

Principle of the planning graph

- A planning graph consists in a sequence of *levels* that correspond to time steps
- Level 0 is the initial state
- Each level contains a set of literals that *could*[1] be true at this time step
- Each level contains a set of actions that *could*[2] be applied at this time step

[1] This is an optimistic estimation
[2] This is an optimistic estimation

Example – problem description

The “have cake and eat cake too” problem

$Init(Have(Cake))$

$Goal(Have(Cake) \wedge Eaten(Cake))$

$Action(Eat(Cake)$

PRECOND: $Have(Cake)$
EFFECT: $\neg Have(Cake) \wedge Eaten(Cake)$

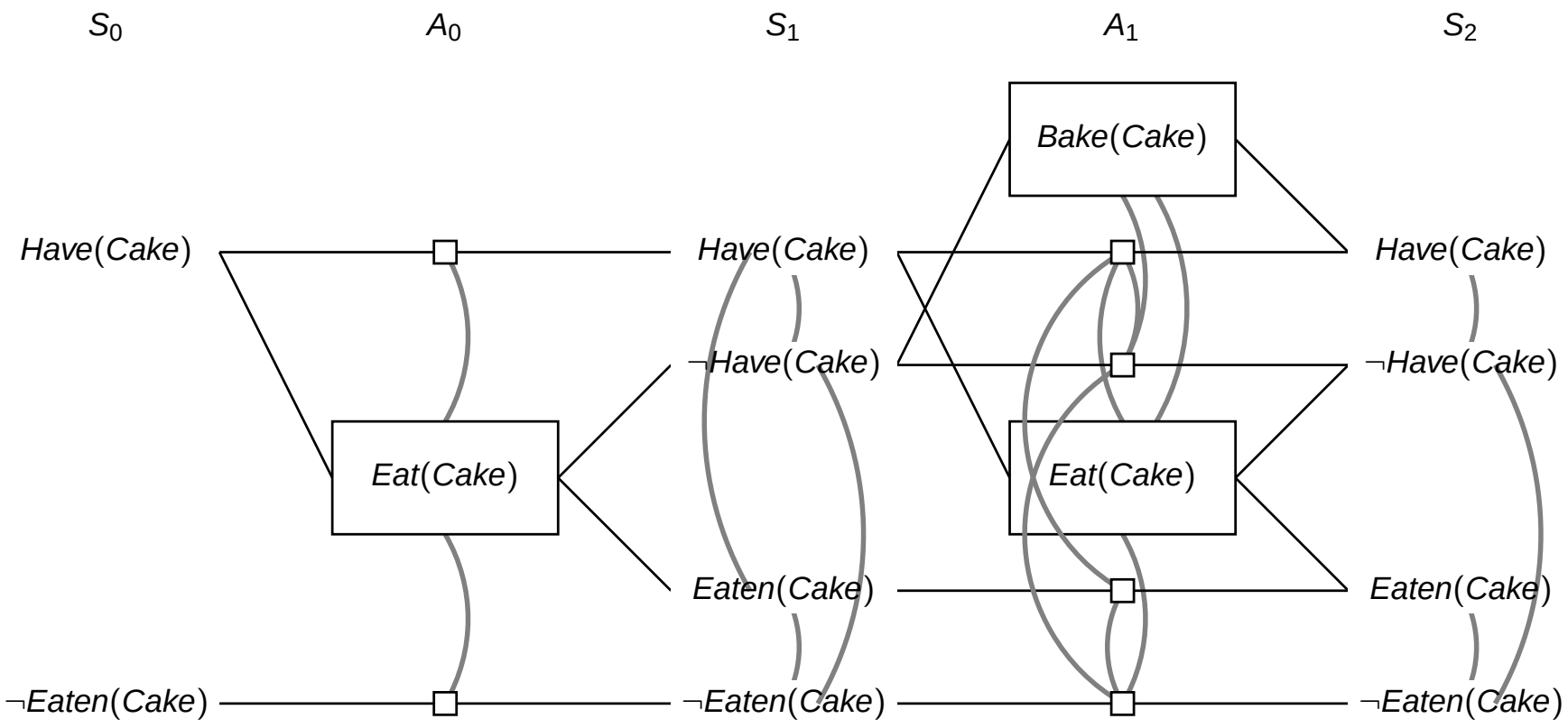
)

$Action(Bake(Cake)$

PRECOND: $\neg Have(Cake)$
EFFECT: $Have(Cake)$

)

Example – planning graph



Planning graph: explanation

- Level A_0 contains all the actions that *could* occur in state S_0.
- Persistence actions (small boxes) represent the fact that one literal is not modified.
- Mutual exclusions (*mutexes*, gray lines) represent *conflicts* between actions.
- To go from level 0 to the level 1, you pick a set of non exclusives actions (for instance, action $Eat(Cake)$)

- Level S_1 contains all the literals that could result from picking any subset of actions in A_0.
- Mutexes represent *conflicts* between literals.

How to build the planning graph

1. Start from S_0
2. $i = 0$
3. Find all the actions of A_{i+1} applicable in S_i given the mutexes
4. Compute the mutexes between the actions of A_{i+1}
5. Compute the literals reachable in S_{i+1}
6. Compute the mutexes in S_{i+1}
7. If $S_{i+1} \neq S_i$, then increment i by 1 and go to 3

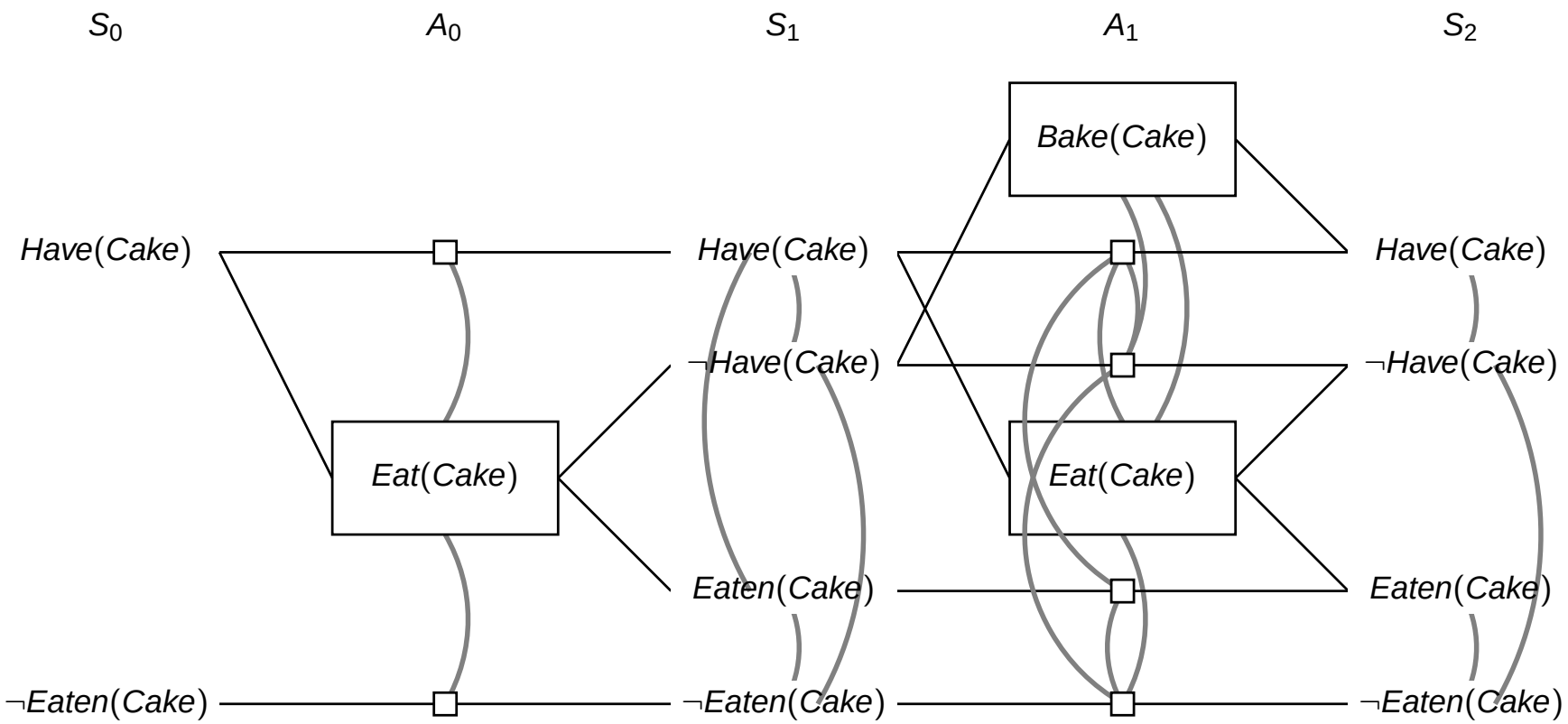
Mutexes

- A mutex between two actions indicates that it is impossible to perform these actions in parallel.
- A mutex between two literals indicates that it is impossible to have these both literals true at this stage.

How to compute mutexes

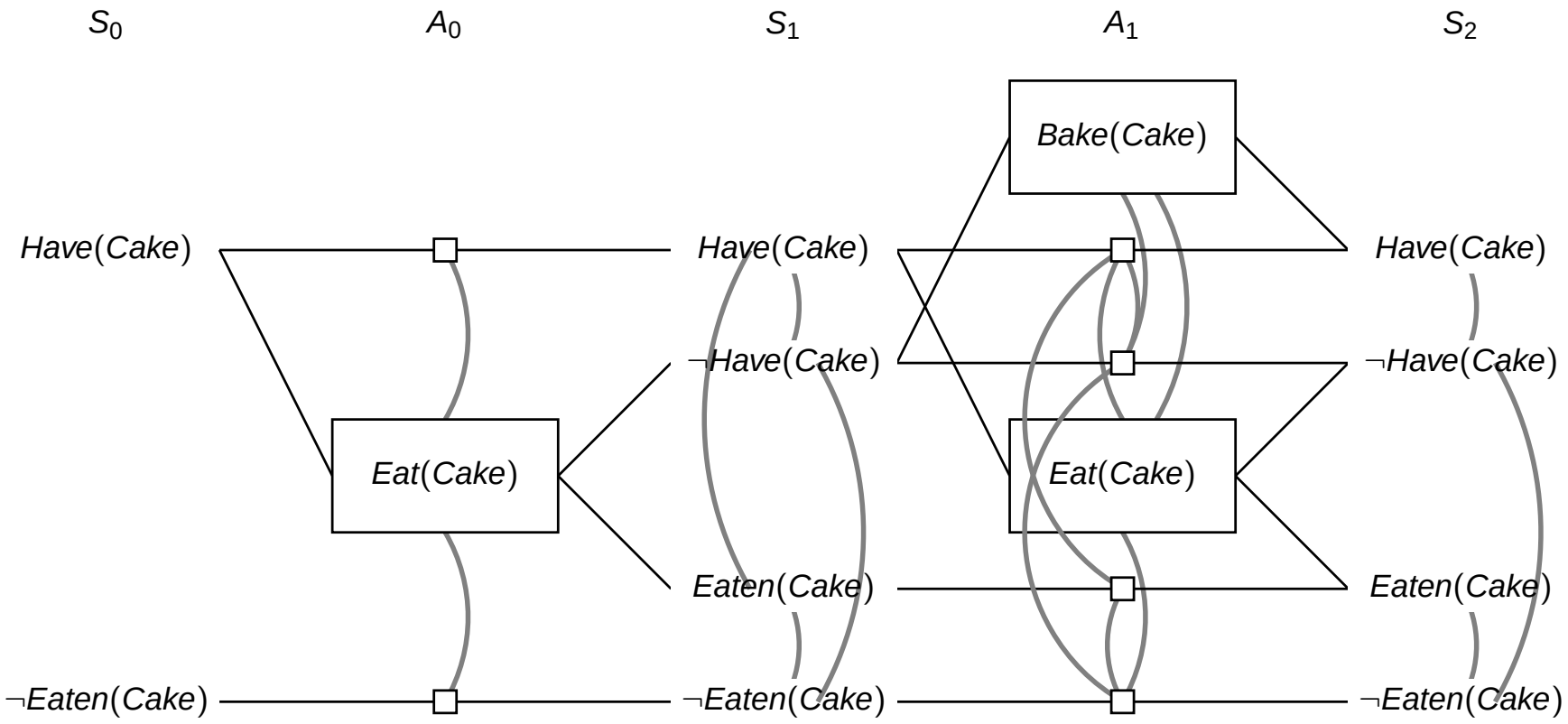
- Actions
  - Inconsistent effects: two actions that lead to inconsistent effects
  - Interference: an effect of the first action negates the precondition of the other action
  - Competing needs: a precondition of the first action is mutually exclusive with a precondition of the second action.
- Literals
  - one literal is the negation of the other one
  - Inconsistensy support: each pair of action archieving the two literals are mutually exclusive.

Mutexes between actions – example



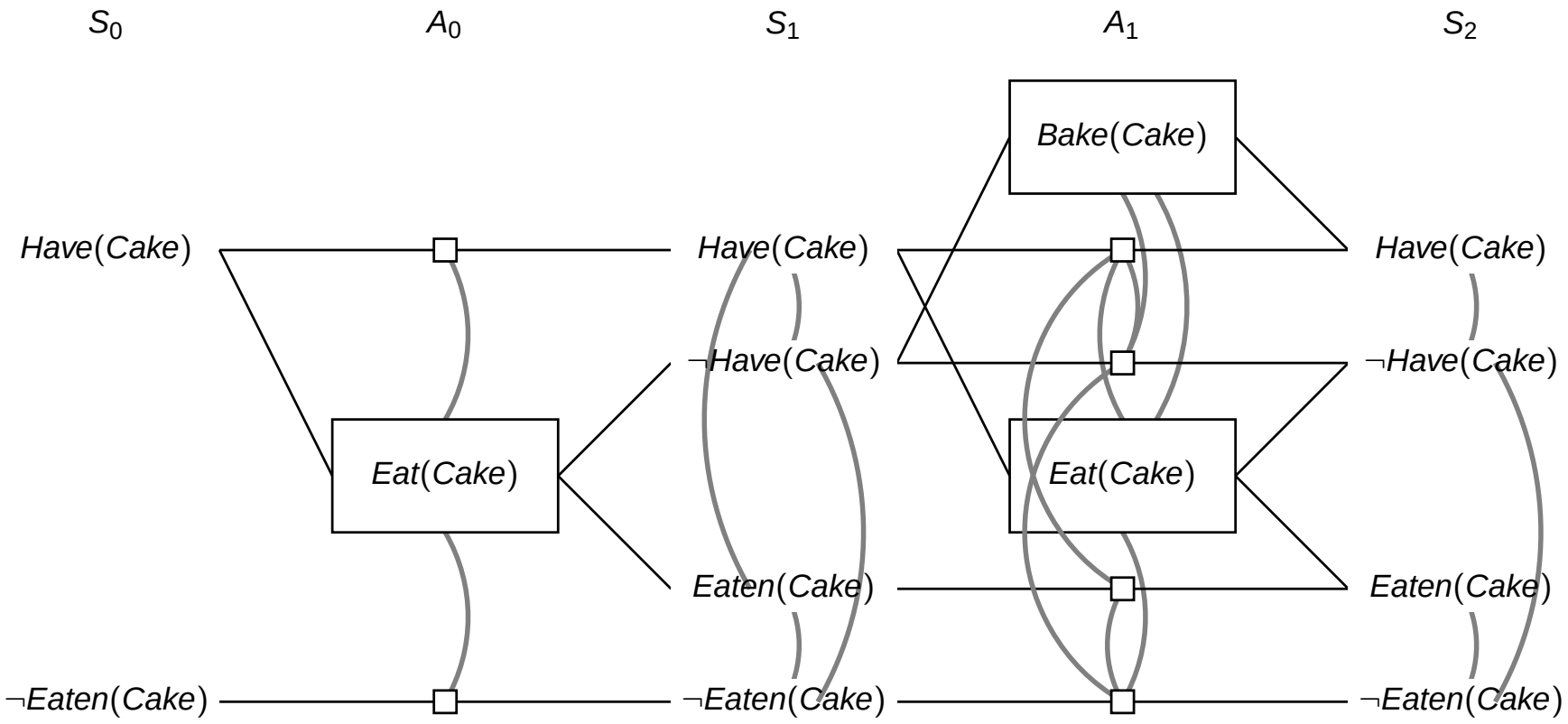
Inconsistent effects: two actions that lead to inconsistent effects

Mutexes between actions – example



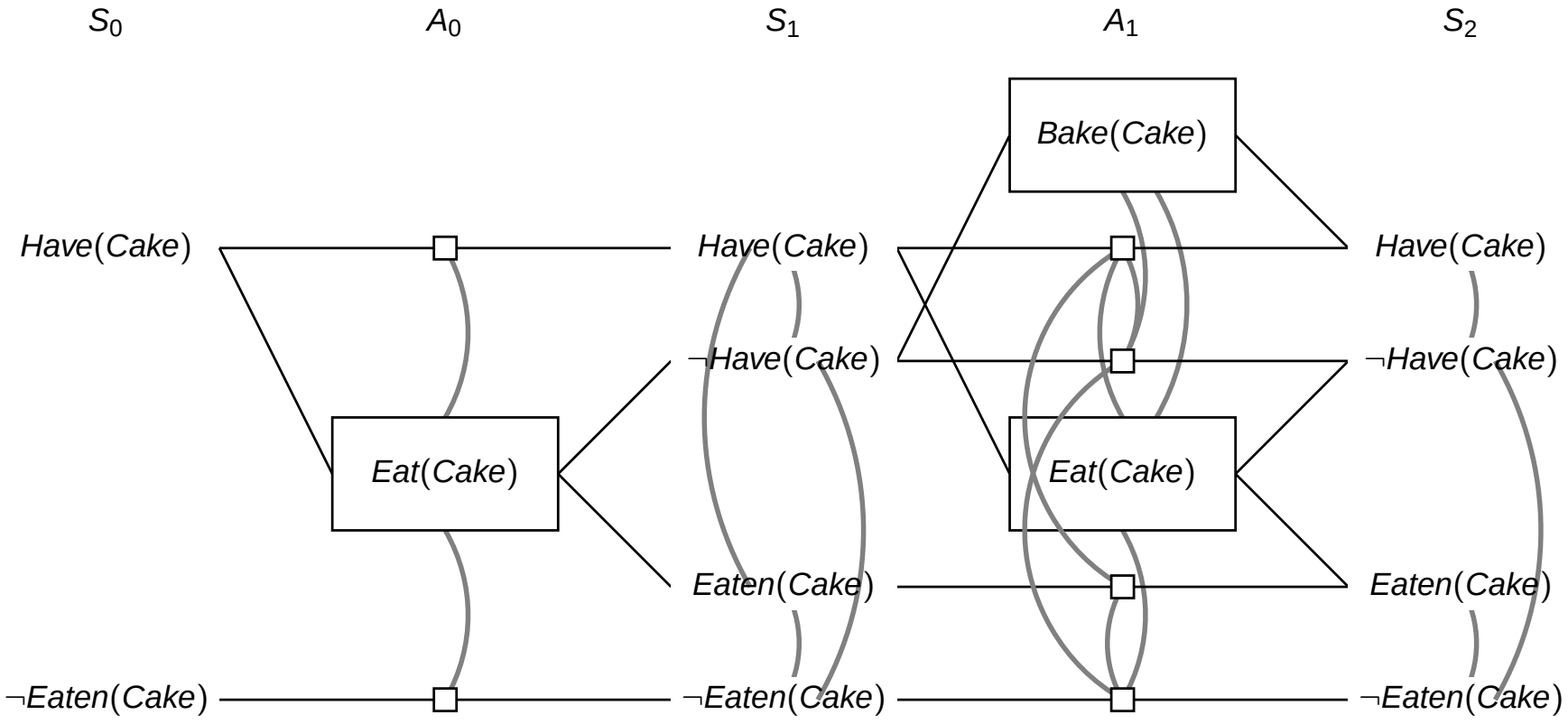
Interference: an effect of the first action negates the precondition of the other action

Mutexes between actions – example



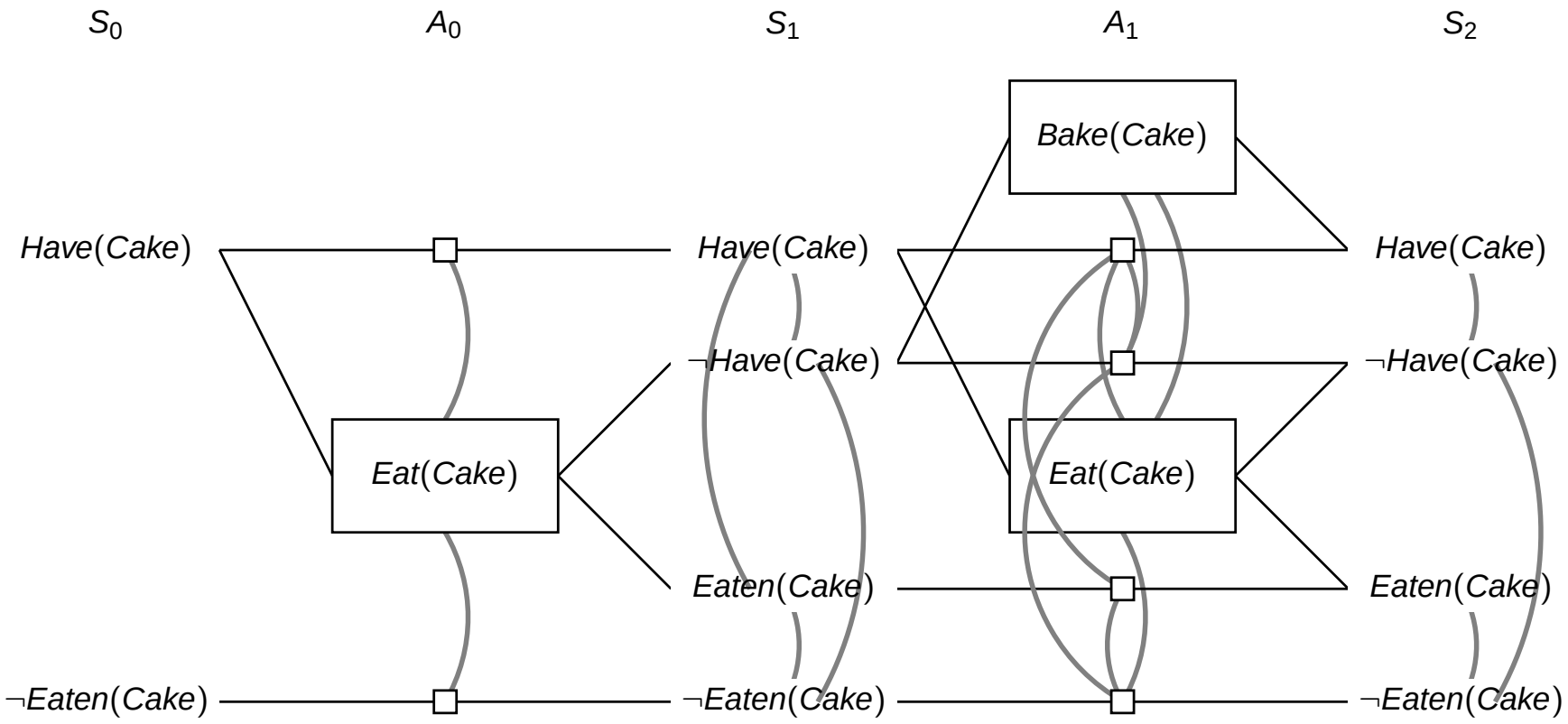
Competing needs: a precondition of the first action is mutually exclusive with a precondition of the second action.

Mutexes between literals – example



One literal is the negation of the other one

Mutexes between literals – example



Inconsistensy support: each pair of action archieving the two literals are mutually exclusive.

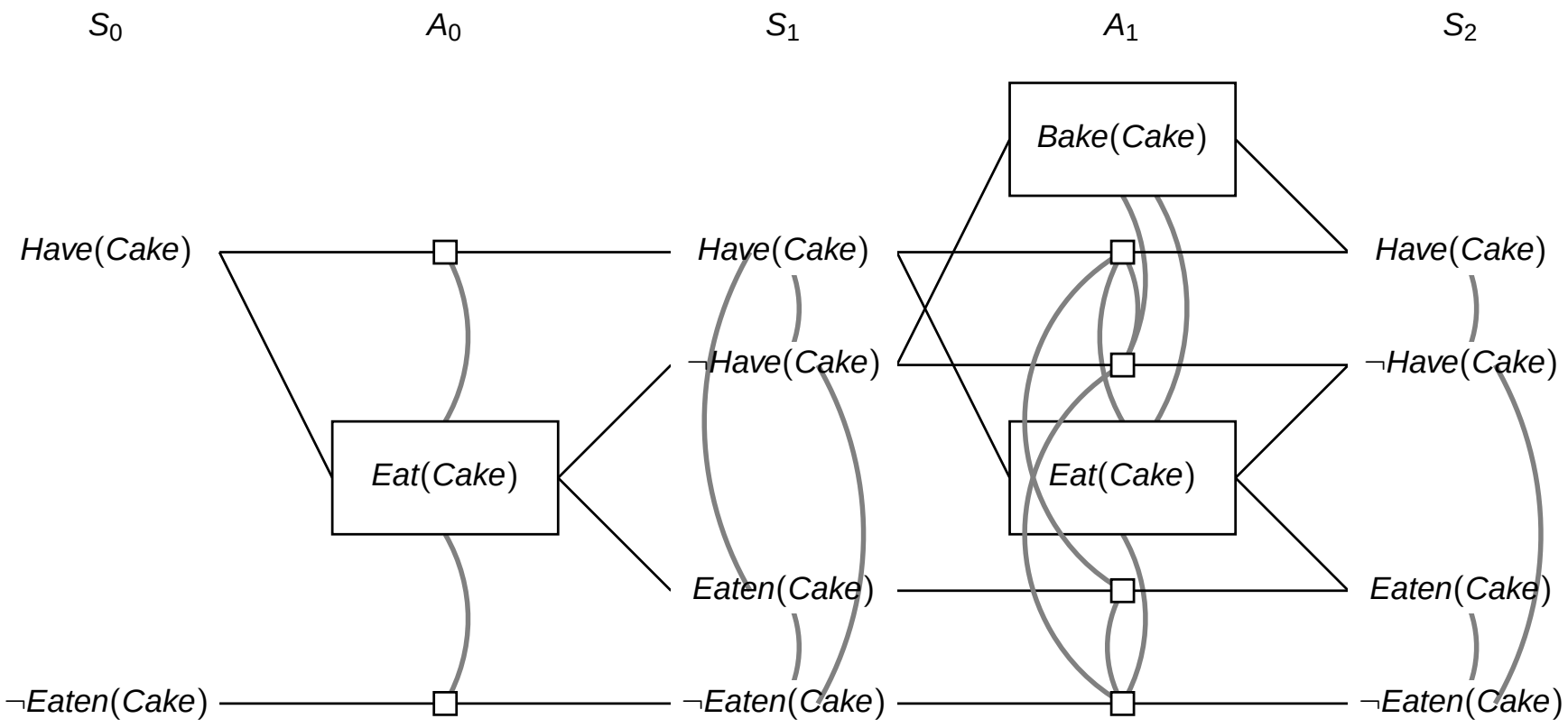
Planning graph for heuristic search

- Using the planning graph to estimate the number of actions to reach a goal
- If a literal does not appear in the planning graph, then there is no plan that achieve this literal
  - $h = \infty$

Possible heuristics

- max-level: take the maximum level where any literal of the goal first appears
  - admissible
- level-sum: take the sum of the levels where any literal of the goal first appears
  - not admissible, but generally efficient (specially for independant subplans)
- set-level: take the minimum level where all the literals of the goal appear and are free of mutex
  - admissible

Example – planning graph



GRAPHPLAN

```
function GRAPHPLAN(Problem)
graph ← INITIAL-PLANNING-GRAPH(Problem)
goals ← GOALS(Problem)
loop
   if goals all non-mutex in last level of graph then
      solution ←
      EXTRACT-SOLUTION(graph, goals, LENGTH(graph))
      if solution ≠ failure then
         return solution
      else if NO-SOLUTION-POSSIBLE(graph) then
         return failure
      end if
   end if
   graph ← EXPAND-GRAPH(graph, problem)
end loop
```