Graphplan
Dinner Date Example

Jiří Vokřínek & Michal Čáp

A4M36PAH

Tutorial 22.4.2012

Materials

- These slides and the example used is based on Dana Nau's lecture slides

  http://www.cs.umd.edu/~nau/planning/slides/chapter06.pdf

Example

- Suppose you want to prepare dinner as a surprise for your sweetheart (who is asleep)

  s0 = {garbage, cleanHands, quiet}

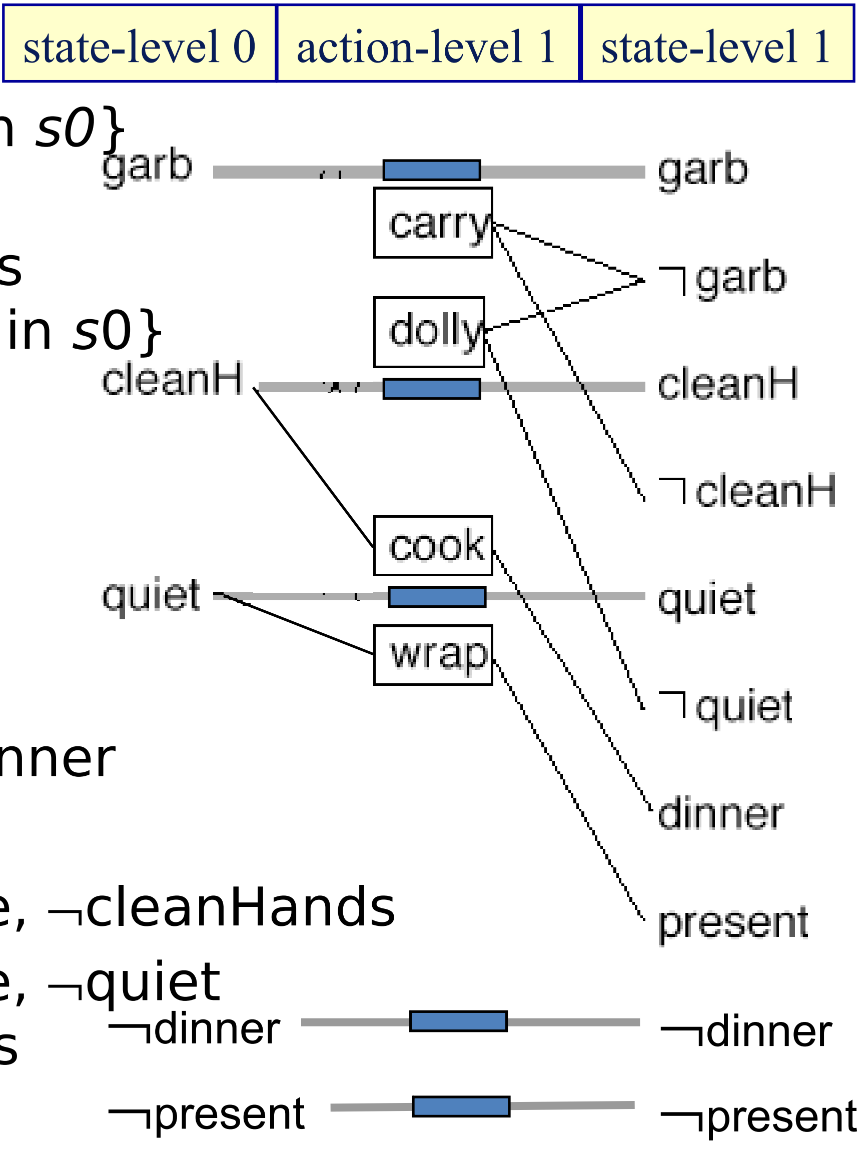
  g = {dinner, present, ¬garbage}

Action	**Preconditions**	**Effects**
cook()	cleanHands	dinner
wrap()	quiet	present
carry()	*none*	¬garbage, ¬cleanHands
dolly()	*none*	¬garbage, ¬quiet

Also have the maintenance actions: one for each literal

Example (continued)

- state-level 0:
  {all atoms in *s*0} U
  {negations of all atoms not in *s0*}
- action-level 1:
  {all actions whose preconditions
  are satisfied and non-mutex in *s*0}
- state-level 1:
  {all effects of all of the
  actions in action-level 1}

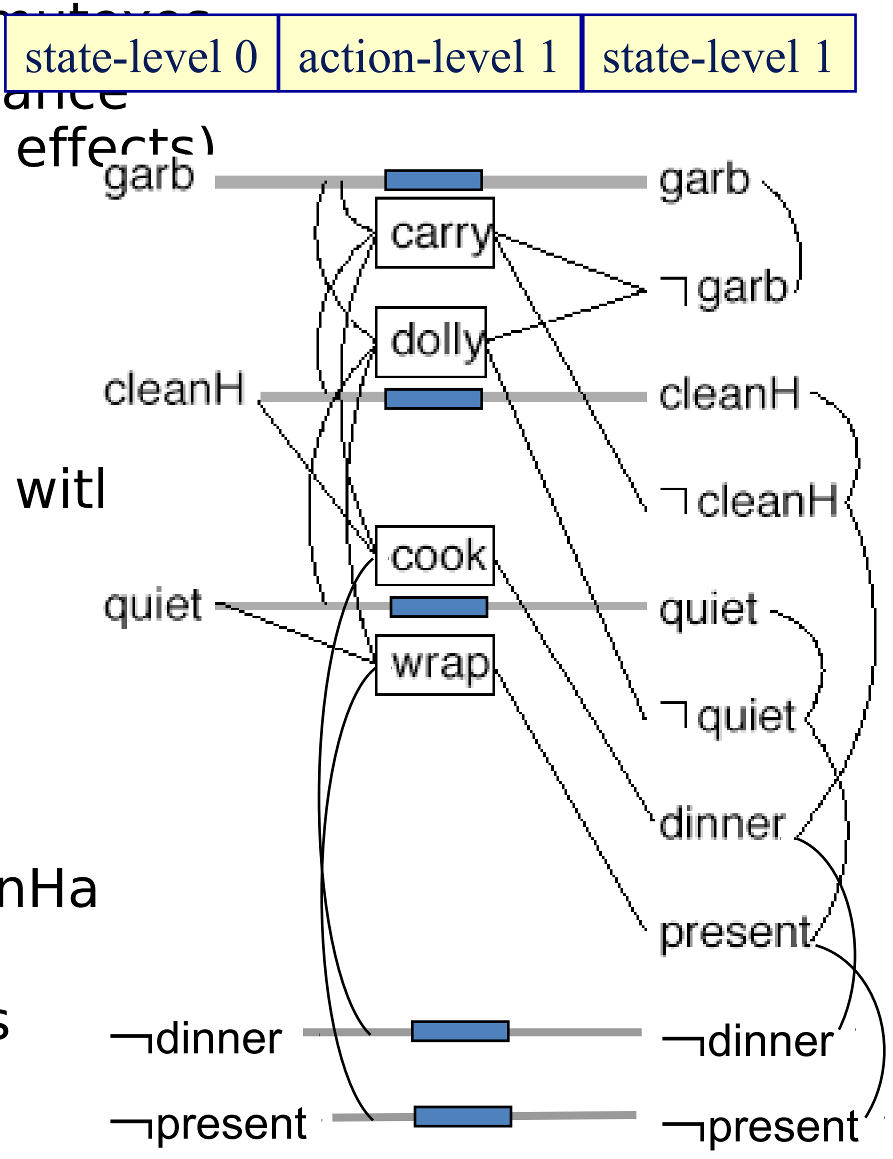


Action	**Preconditions**	**Effects**
cook()	cleanHands	dinner
wrap()	quiet	present
carry()	*none*	¬garbage, ¬cleanHands
dolly()	*none*	¬garbage, ¬quiet

Also have the maintenance actions

Example (continued)

- Augment the graph to indicate mutexes
- *carry* is mutex with the maintenance action for *garbage* (inconsistent effects)
- *dolly* is mutex with *wrap*
  - interference
- ~*quiet* is mutex with *present*
  - inconsistent support
- each of *cook* and *wrap* is mutex with a maintenance operation

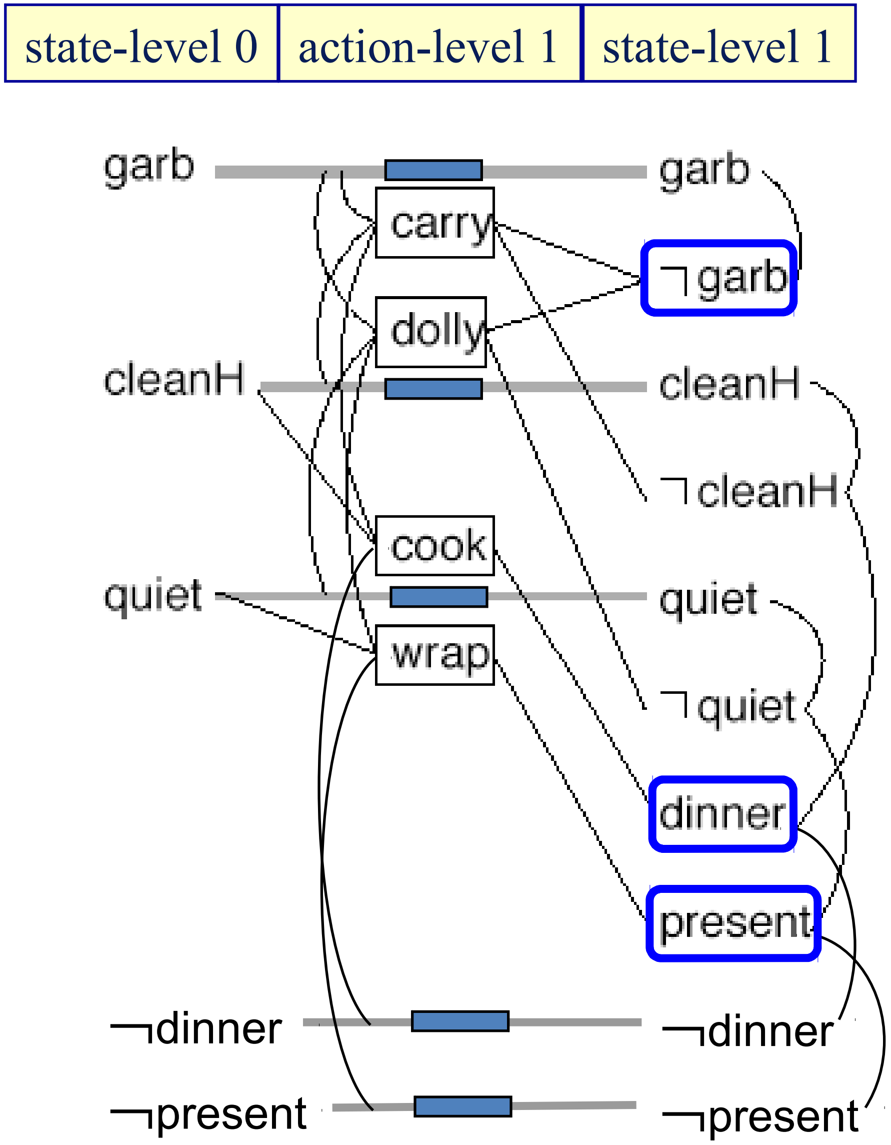


Action	**Preconditions**	**Effects**
cook()	cleanHands	dinner
wrap()	quiet	present
carry()	*none*	¬garbage, ¬cleanHa
dolly()	*none*	¬garbage, ¬quiet

Also have the maintenance actions

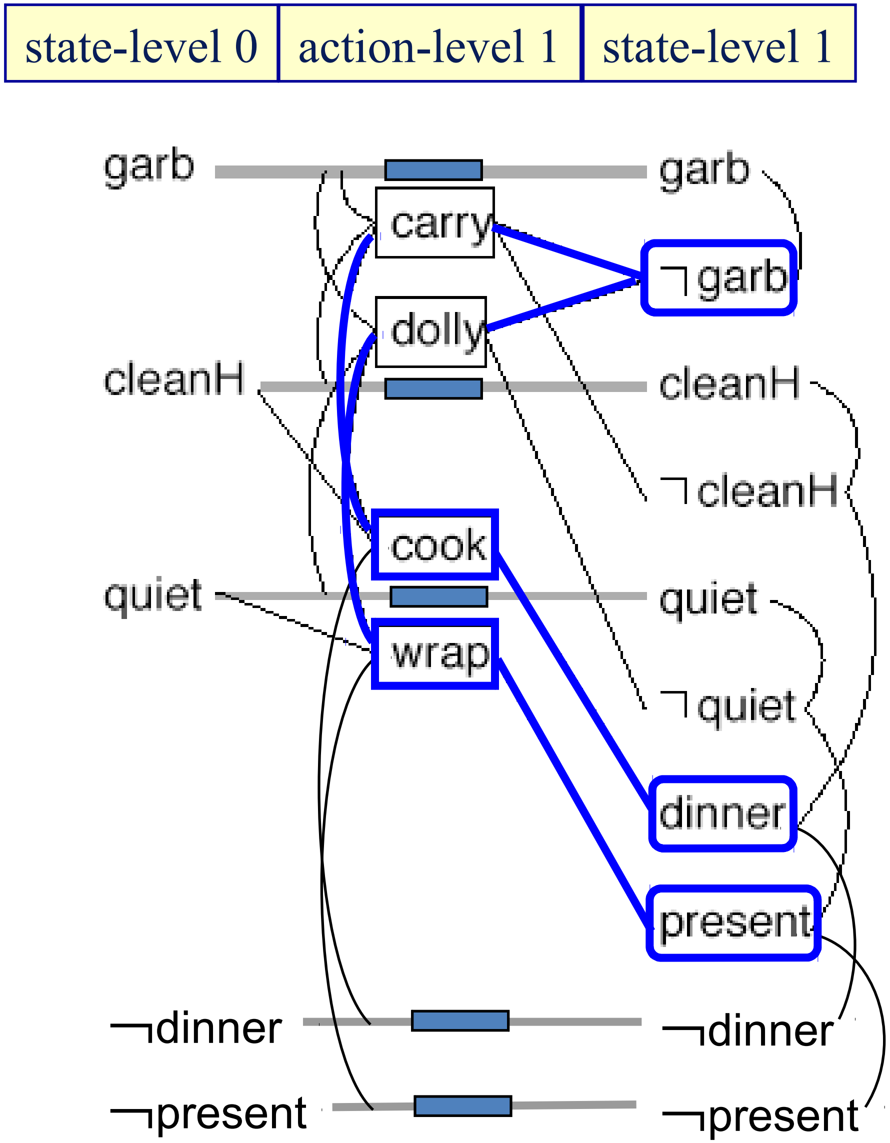
Example (continued)

- Check to see whether there’s a possible solution
- Recall that the goal is
  - {¬*garbage, dinner, present*}
- Note that in state-level 1,
  - All of them are there
  - None are mutex with each other
- Thus, there’s a chance that a plan exists
- Try to find it
  - Solution extraction



Example (continued)

- Two sets of actions for the goals at state-level 1
- Neither of them works
  - Both sets contain actions that are mutex



Recall what the algorithm does

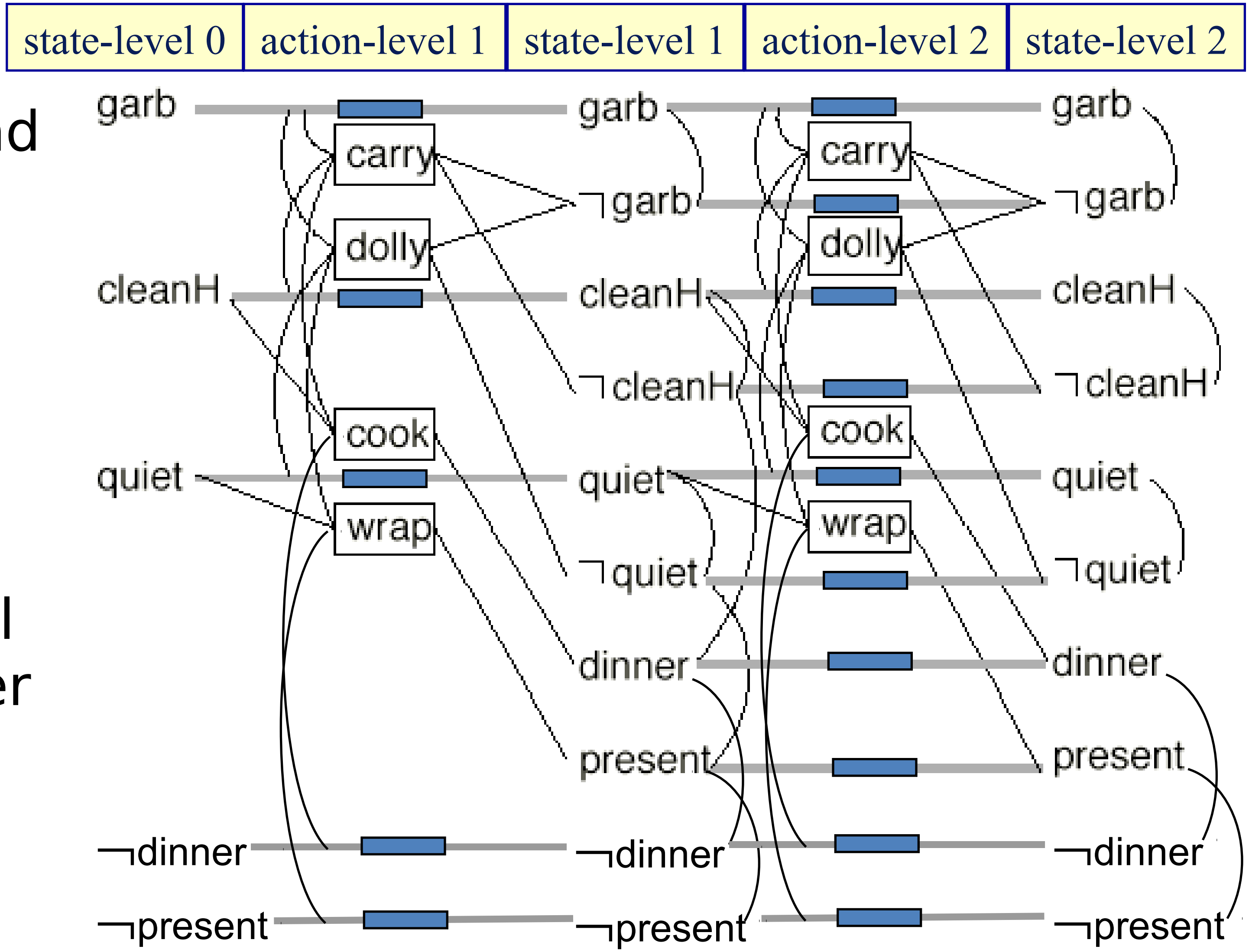
procedure Graphplan:

- for *k* = 0, 1, 2, …
  - *Graph expansion:*
  - create a “planning graph” that contains *k* “levels”
  - Check whether the planning graph satisfies a necessary
    (but insufficient) condition for plan existence
  - If it does, then
  - do *solution extraction:*
    - backward search, modified to consider only the actions in the planning graph
    - if we find a solution, then return it
- – If the graph is stabilized, solution is unreachable

Example (continued)

- Go back and do more graph expansion

- Generate another action-level and another state-level



state-level 0
action-level 1
state-level 1
action-level 2
state-level 2

garb
cleanH
quiet
¬dinner
¬present

carry
dolly
cook
wrap

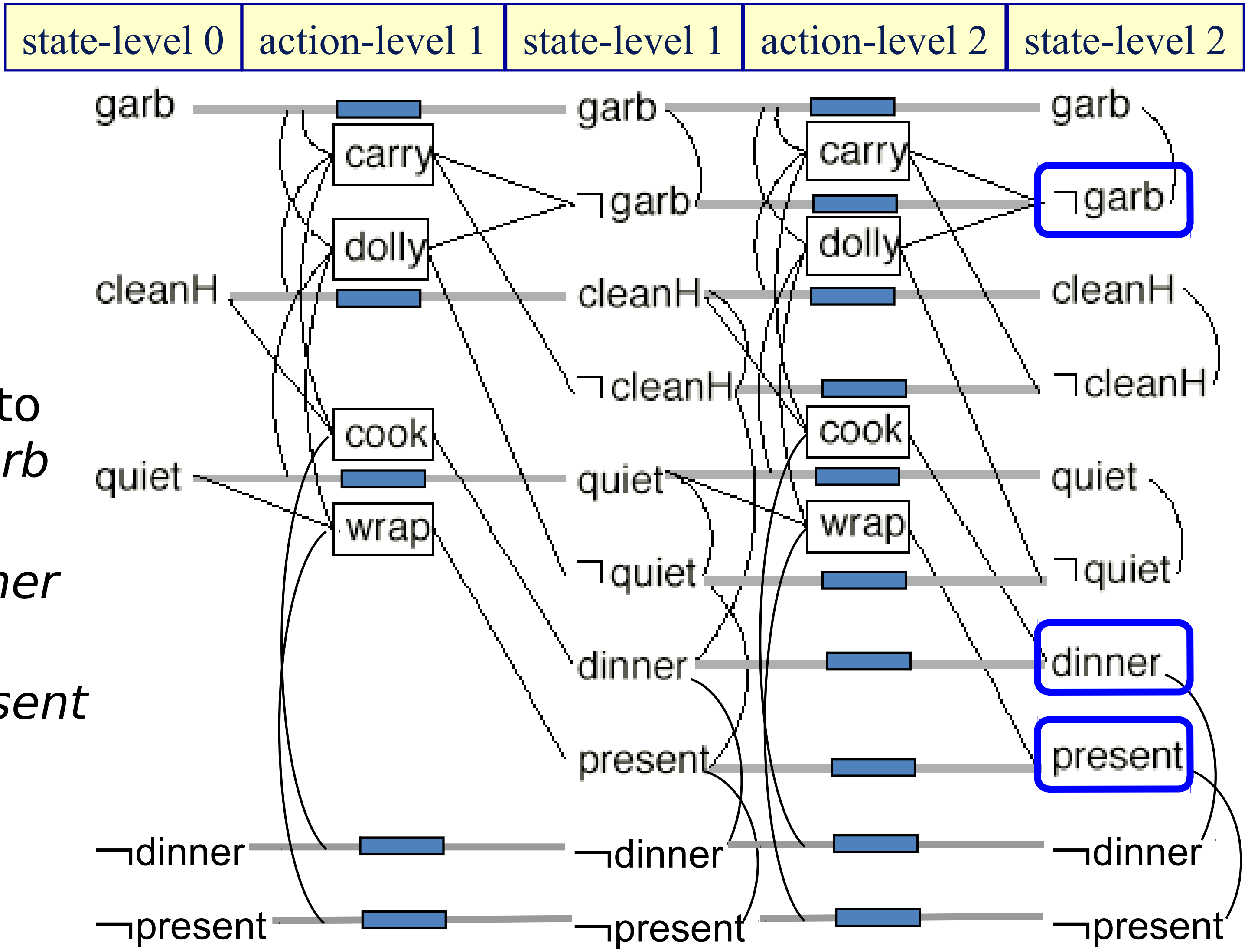
garb
¬garb
cleanH
¬cleanH
quiet
¬quiet
dinner
present
¬dinner
¬present

carry
dolly
cook
wrap

garb
¬garb
cleanH
¬cleanH
quiet
¬quiet
dinner
present
¬dinner
¬present

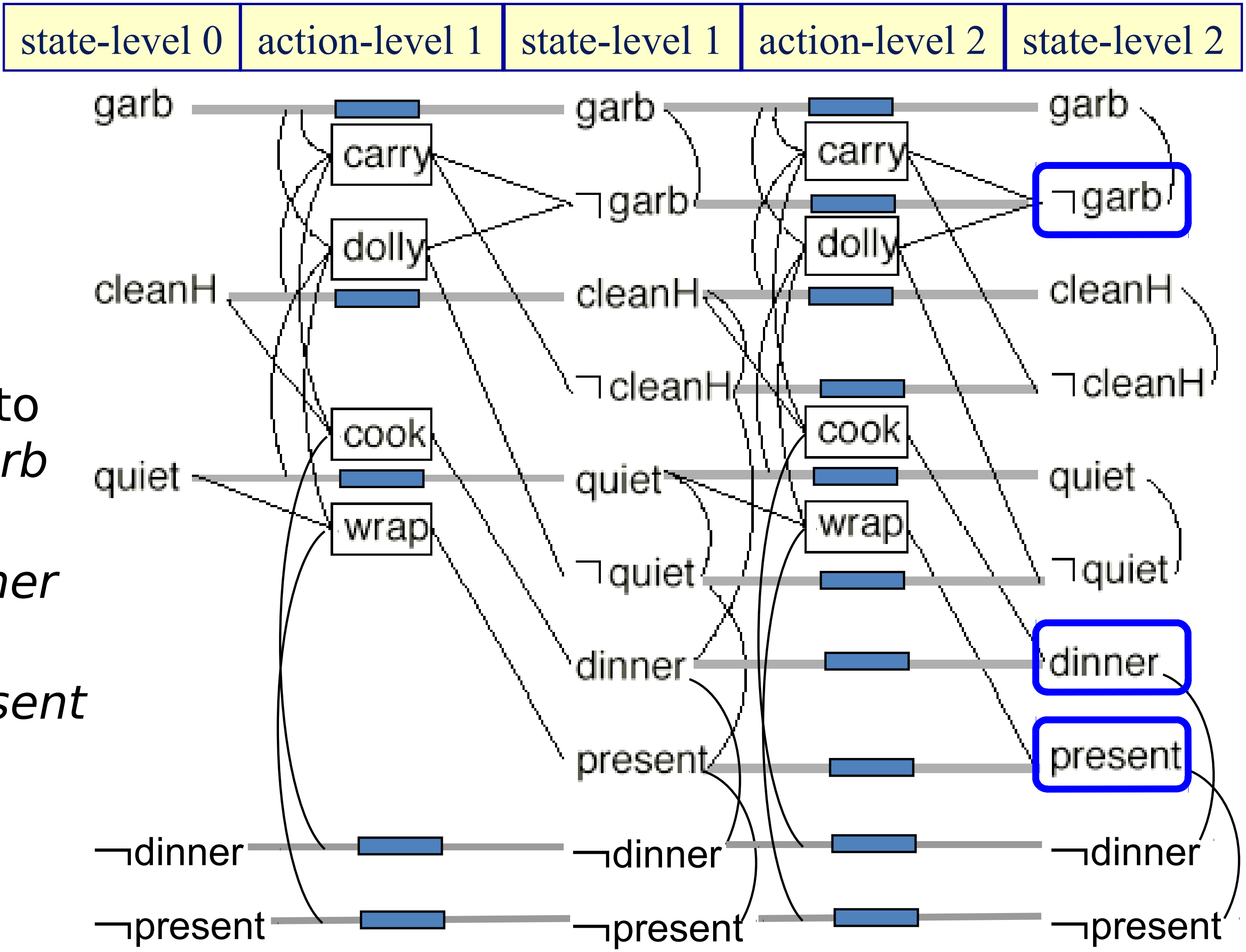
Example (continued)

- Solution extraction
- Twelve combinations at level 4
  - Three ways to achieve ¬*garb*
  - Two ways to achieve *dinner*
  - Two ways to achieve *present*



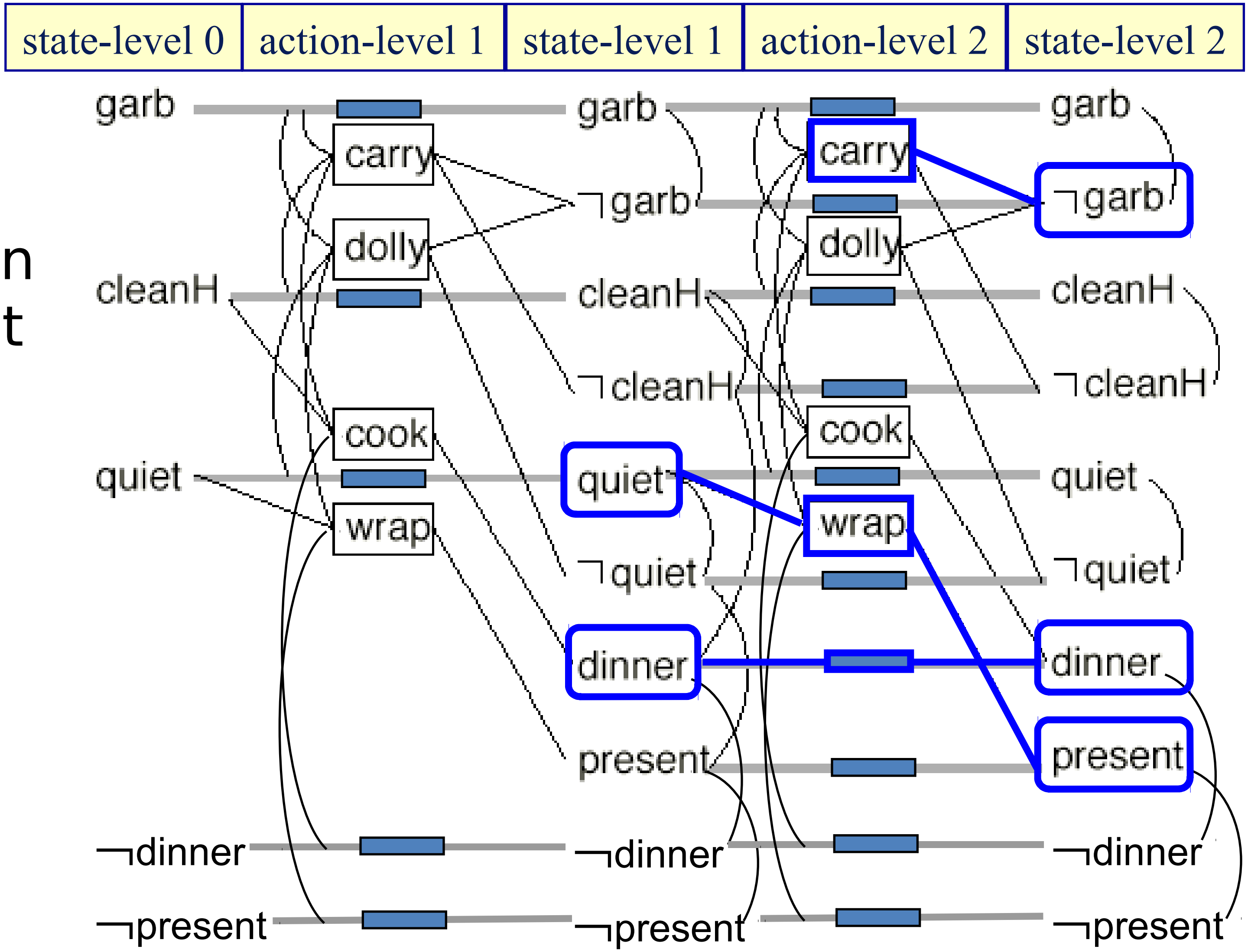
Example (continued)

- Solution extraction
- Twelve combinations at level 4
  - Three ways to achieve ¬*garb*
  - Two ways to achieve *dinner*
  - Two ways to achieve *present*



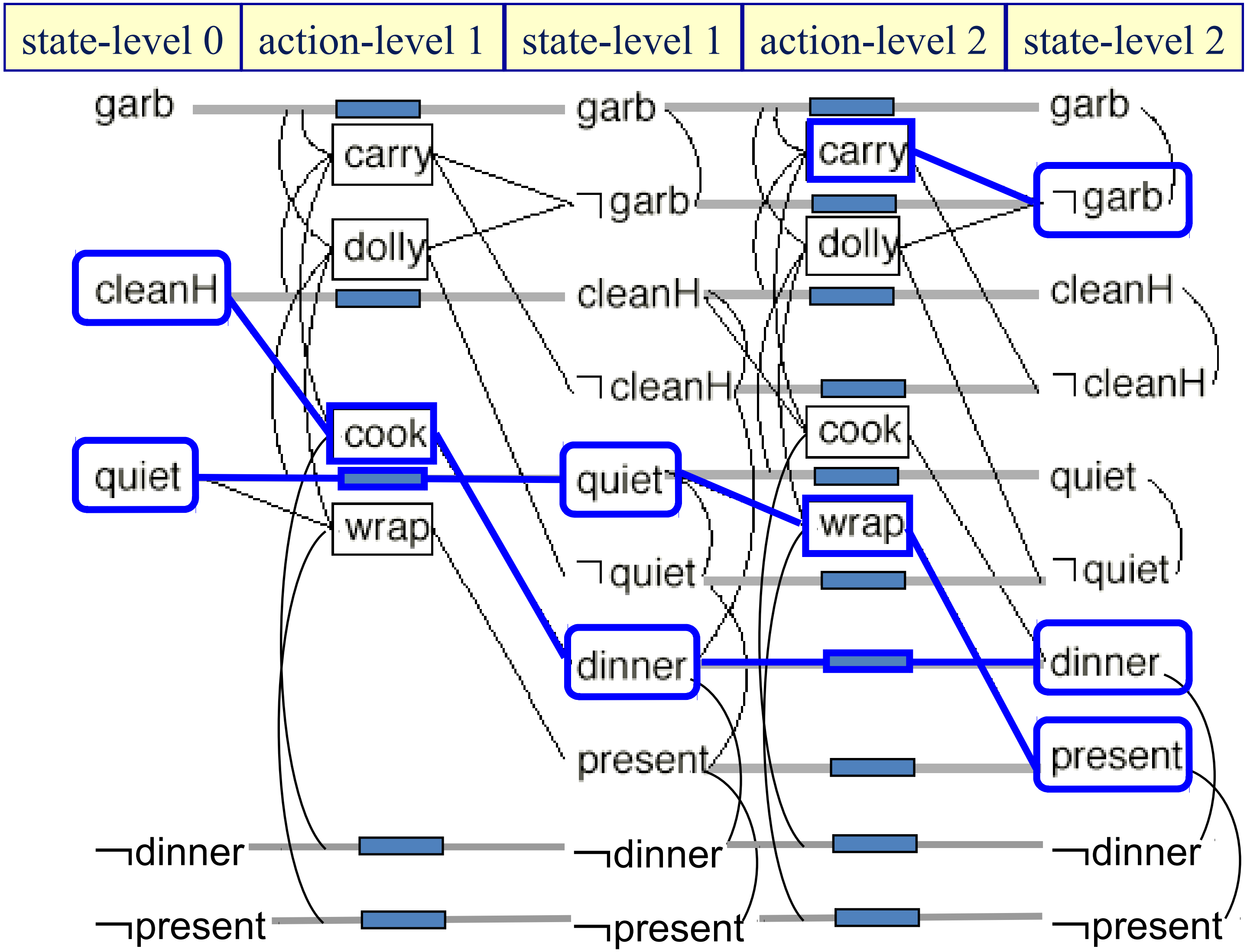
Example (continued)

- Several of the combinations look OK at level 2
- Here’s one of them



Example (continued)

- Call Solution-Extraction recursively at level 2
- It succeeds
- Solution whose *parallel length* is 2


state-level 0
action-level 1
state-level 1
action-level 2
state-level 2
garb
cleanH
quiet
¬dinner
¬present
carry
dolly
cook
wrap
garb
¬garb
cleanH
¬cleanH
quiet
¬quiet
dinner
present
¬dinner
¬present
carry
dolly
cook
wrap
garb
¬garb
cleanH
¬cleanH
quiet
¬quiet
dinner
present
¬dinner
¬present