Abstraction

Basics

- dropping some distionctions between states
- we can't distuingish some states (the states collapse into just one state)
- we want to have the problem so small to be able to solve it optimally in a short time
- we solve the problem in the abstracted space and use the result as a heuristic for the original problem - *is the heuristic admissible?*

Combining abstractions

- maximum
    - *admissible?*
    - *picture*
- sum
    - *admissible?*
    - *can be?*

Orthogonality

- formal definition
- a transition (not self loop) must remain in at most on of the abstractions
    - *picture*
- then the sum of such abstractions is admissible

Projection

- FDR language
- abstraction based on leaving out some of the state variables
- projection defines which state variables are taken into account
- **atomic projection**
    - projection of exactly one state variable

Example

Assume that there is a monkey in a room with some bananas hanging out of reach from the ceiling, but a box is available that will enable the monkey to reach the bananas if he climbs on it. Initially, the monkey is at A, the bananas at B, and the box at C. The monkey and box have height LOW, but if the monkey climbs onto the box, he will have height HIGH, the same as the bananas. The actions available to the monkey include GO from one place to another, PUSH an object from one place to another, CLIMB onto an object, and GRASP an object. Grasping results in holding the object if the monkey and object are in the same place at the same height. The monkey wants to get the bananas.

variables:

monkey = {A, B, C, O}

bananas = {A, B, C, M}
box = {A, B, C}

states:

(monkey;bananas,box)

init:

(A;B;C)

goal:

(?;M;?)

operators:

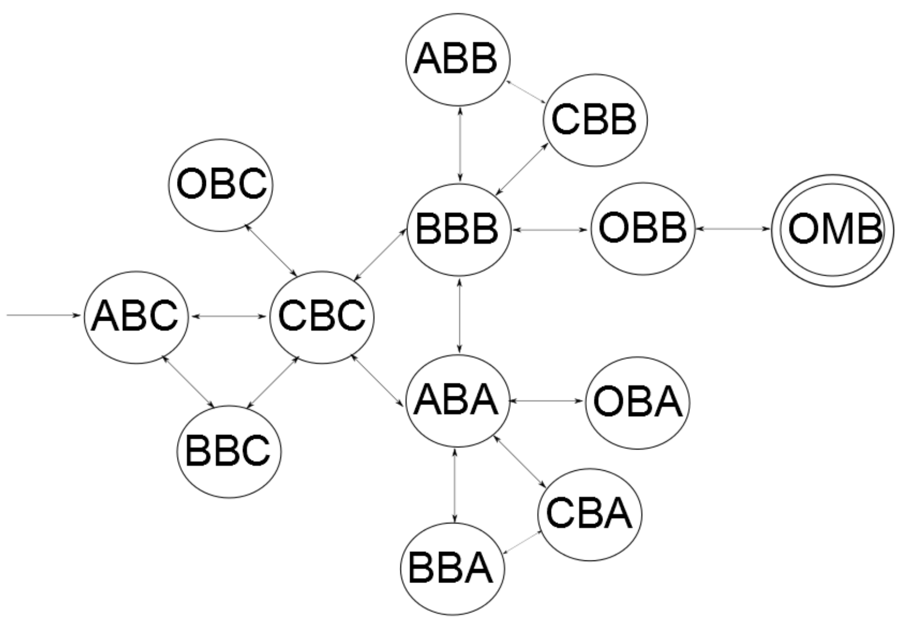
$go_{i,j}$ (monkey = i; monkey:=j),

$push_{i,j}$ (monkey = i, box = i; monkey:=j, box:=j),

up_i (...), $down_i$ (...),

$take_i$ (...)

1. Draw the state space



2. Draw projection p^{monkey} - ...
3. Projection $p^{bananas}$ - ...

4. Projection p^{box} - ...
5. Projection $p^{bananas,box}$

PDB heuristics

- maintains collection of patterns
- pattern is a definition of the projection
- computes different pattern heuristics
    - can use maximum
    - we want to use sum
- Additive Pattern Set
    - Pattern Set is additive if there exists no operator that has an effect on different variables in different patterns
    - reflects the orthogonality
    - *which of the drawn abstractions are additive?*
- compatibility graph
    - maintains additive set (an edge is present if the patterns are orthogonal)
    - find cliques
        - sum the heuristic values of those patterns in the clique
        - max the sums
        - $h(s) = max(D \text{ in } cliques(C)) \text{ } sum(P \text{ in } D) \text{ } h^P(s)$
- *example*
    - *compatibility graph for the monkey example*
    - *compute the heuristic for the initial state*

Merge and Shrink

- algorithm
    - construct atomic projections
    - select A_1, A_2
    - shrink ($A_1|A_2$) or both until size(A_1) x size(A_2) <= N
    - merge ($A_1 \times A_2$)
- synchronized product *definition*
- *example*
    - N = 7
    - $A_1 = p^{bananas}$, $A_2 = p^{box}$
    - abstraction on $p^{bananas,box}$
    - abstraction on p^{monkey}
    - merge
    - compute the heuristic for the initial stae