pattern database

simplify the problem by removing some variables

---

variable removal

given: planning problem and state
(not necessarily the initial state!)

aim: estimate distance to the goal from the state
use: guide the search (A*, greedy best-first search,…)

how:

- remove some variables from the problem and the state
- compute optimal plan
- its length estimates the length of the optimal plan in the original problem

example

```
a: x ⇒ y,z
b: y ⇒ -z
c: x,y,-z ⇒ w
goal: w,z
```

calculate $h(x-y-z-w)$

optimal plan from state: a,b,c,a (lenght 4)
perfect heuristics is $h(x-y-z-w) = 4$

---

keep only z and w

```
a: x ⇒ y,z
b: y ⇒ -z
c: x,y,-z ⇒ w
goal: w,z
```

```
a: ⇒ z
b: ⇒ -z
c: -z ⇒ w
goal: w,z
```

remove x and y

state x-y-z-w becomes -z-w

optimal plan from state: c,a (lenght 2)
heuristics $h_{zw}(x-y-z-w)=2$

[note] Actions a and b have no precondition after removal. They can be executed in every state.

keep only x and w

```
a: x ⇒ y,z
b: y ⇒ -z
c: x,y,-z ⇒ w
goal: w,z
```

```
a: x ⇒
b: ⇒
c: x ⇒ w
goal: w
```

remove y and z

state x-y-z-w becomes x-w

optimal plan from state: c (length 1)
heuristics $h_{xw}(x\text{-}y\text{-}z\text{-}w)=1$

[note] Actions a and b end up with no effects after removal of variables. They are irrelevant to planning, so they could be deleted altogether. They still are of interest for the theoretical analysis.

observations

1. variables removed ⇒ shorter plans
   (4 becomes 2 or 1)
2. optimal plan length depends on which variables are deleted
   (2 or 1 depending on which variables are deleted)

is observation 1 true in general?

---

plan length

a plan for the original instance is also a plan after deletion:

- deleting preconditions makes actions still executable in the same state
- deleting effects in general makes some other actions no longer executable, but not in this case: effects also deleted from preconditions of all other actions
- same for deleting variables for the goal

deletion may introduce new plans (examples above)

what is this used for?

A*, greedy best-first search, etc. need an heuristics `h(s)`

heuristics:

`h(s)` = length of shortest plan from `s` to the goal after deleting variables

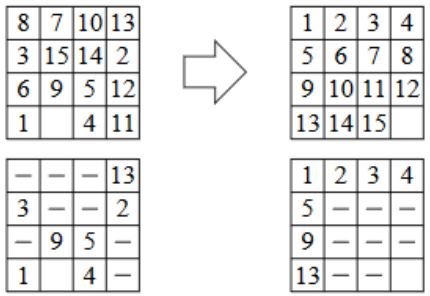
after deletion, compute shortest plan for all states
in the example: a plan for `-z-w`, a plan for `-zw`, a plan for `z-w` and a plan for `zw`
only four plans, not sixteen

---

why "pattern database"?

why this name?

originates from the 15-puzzle



fix some tiles, for example `1,2,3,4,5,9,13`
each disposition of these and the empty square is a pattern
for each pattern, store the minimal number of moves to get the tiles in position

database of minimal plan lengths for each *pattern*

erasing some tiles ⇒ delete some variables

search space

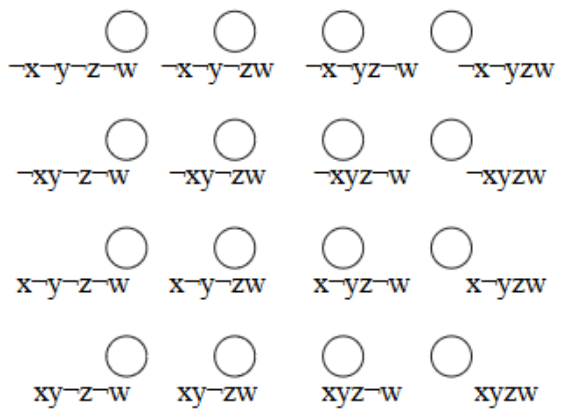
```
a: x ⇒ y,z
b: y ⇒ -z
c: x,y,-z ⇒ w
goal: w,z
```

restricting to some variables simplifies the problem

obvious when looking at the search spaces

---

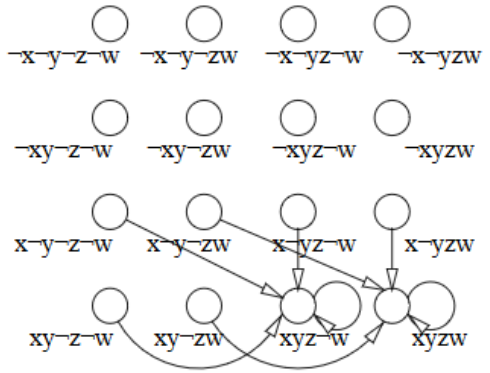
states



original problem

search space, no actions

action `a`

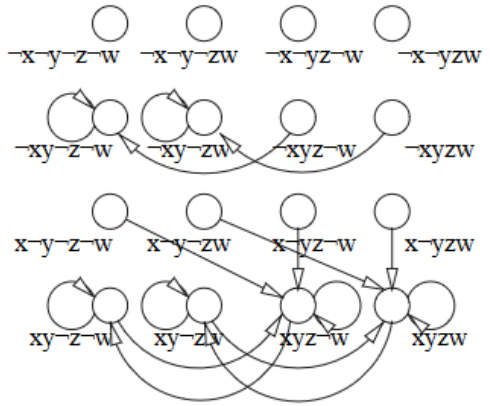


original problem

states and action `a: x ⇒ y,z`

---

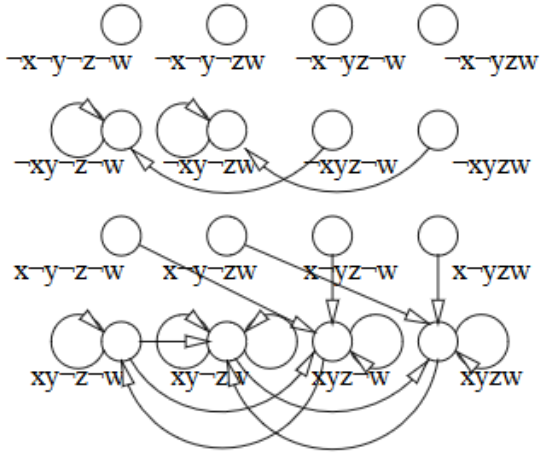
action `b`



original problem

states and actions `a` and `b: y ⇒ -z`

action `c`

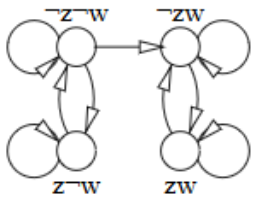


original problem

states and all three actions, including `c: x,y,-z ⇒ w`

---

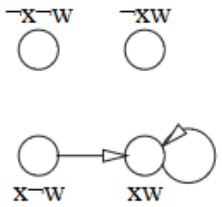
search space: only `z` and `w`



two variables ⇒ four states

example: executing `a` in `-zw` leads to itself

search space: only x and w



---

smaller is better?

more variables removed:

- smaller search space
  ⇒ easier to search for an optimal plan
  but…
- optimal plan may become too short
  ⇒ heuristics underestimate the cost too much

---

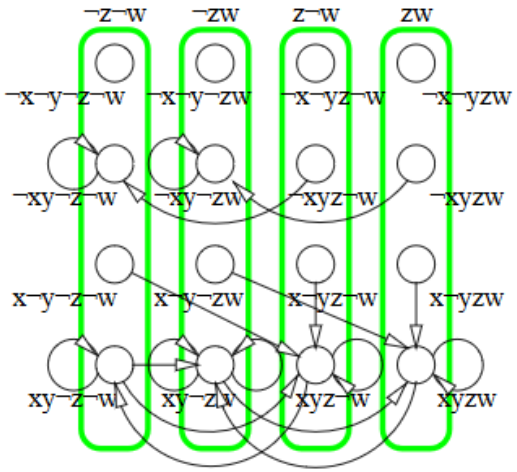
grouping states

deleting variables affects the search space

for example, the four states in the original instance:
`-x-y-z-w`,
`-xy-z-w`,
`x-y-z-w`,
`xy-z-w`

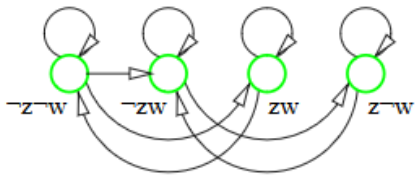
when deleting `xy`, all become:
`-z-w`

grouping states, in the search space



every column of four states form a group

every group is a state of the simplified problem



---

abstraction heuristics

group sets of states

grouping by ignoring some variables ⇒ pattern database

other methods exist