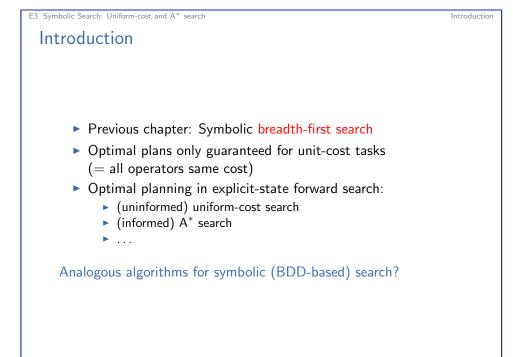


Planning and Optimization

E3.1 Introduction

Planning and Optimization December 19, 2016 — E3. Symbolic Search: Uniform-cost and A* search

E3.1 Introduction			
E3.2 Symbolic Uniform-Cost Search			
E3.3 Symbolic A*			
E3.4 Discussion			
E3.5 Summary			
M. Helmert, G. Röger (Universität Basel)	Planning and Optimization	December 19, 2016	2 / 32



3 / 32

M. Helmert, G. Röger (Universität Basel)

Ν

E3.2 Symbolic Uniform-Cost Search

M. Helmert, G. Röger (Universität Basel)

Planning and Optimization

E3. Symbolic Search: Uniform-cost and A^* search

Symbolic Uniform-Cost Search

December 19, 2016

5 / 32



- The apply function (previous chapter) computes the set of states S' that can be reached from a set of states S by applying one operator.
- This is called the image of S wrt. transition relation $T_V(O)$.
- ▶ Now: image computation for arbitrary transition relations.

```
def image(B, T):

B := bdd-intersection(B, T)

for each v \in V:

B := bdd-forget(B, v)

for each v \in V:

B := bdd-rename(B, v', v)

return B
```

Exactly like apply but gets transition relation as argument.



- Previously: one transition relation $T_V(O)$ for all operators
- Now: several transition relations for operators of same cost
- Set T of pairs (T, c), where T is a transition relation for one/some/all operators of cost c
 - All operators must be covered (and nothing else):
 U_{(T,c)∈T} r(T) = r(T_V(O))
 - The cost must be correct: For $(T, c') \in \mathcal{T}$: if $a \in r(T)$ then $a \models \bigvee_{o \in O: cost(o) = c'} \tau_V(o)$

Planning and Optimization

Many possibilities to split up $T_V(O)$ (discussed later)

M. Helmert, G. Röger (Universität Basel)

December 19, 2016 6 / 32

```
E3. Symbolic Search: Uniform-cost and A* search Symbolic Uniform-Cost Search (Positive Operator Costs)
```

```
def symbolic-uniform-cost(V, I, O, \gamma):

goal := build-BDD(\gamma)

\mathcal{T} := make-transition-relations(V, O)

open_0 := bdd-state(I)

while \exists g : open_g \neq \mathbf{0}:

g := min\{g \mid open_g \neq \mathbf{0}\}

closed_g := open_g

if bdd-intersection(open_g, goal) \neq \mathbf{0}:

return construct-plan(I, O, goal, closed_*, g)

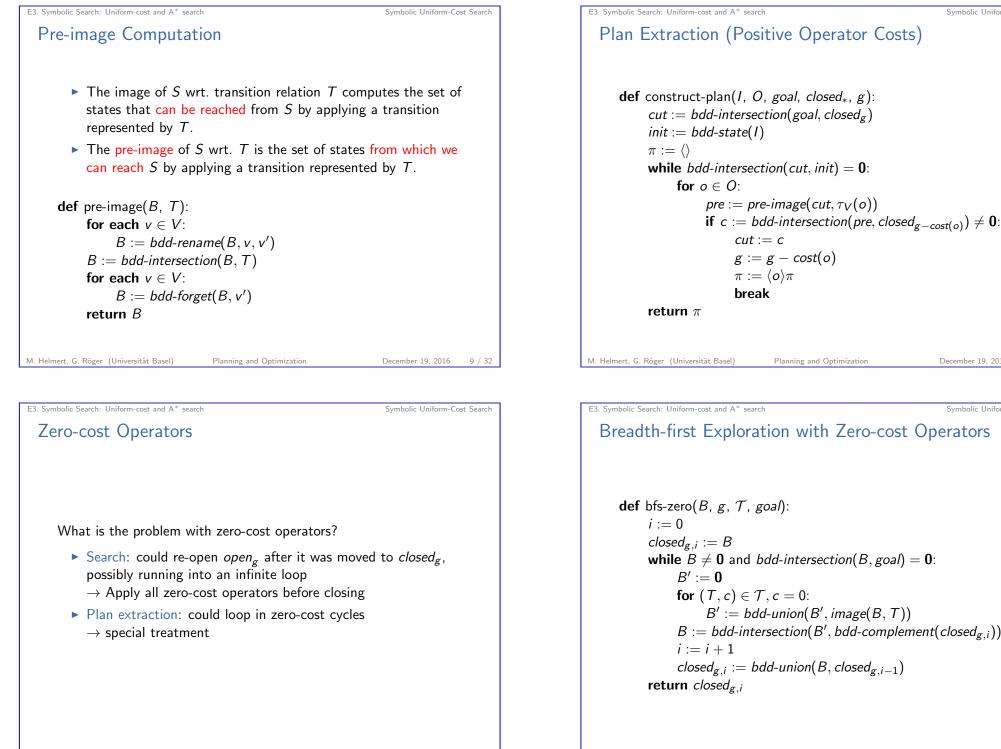
for all (T, c) \in \mathcal{T}:

open_{g+c} := bdd-union(open_{g+c}, image(open_g, T)))

open_g := \mathbf{0}

return unsolvable
```

Planning and Optimization



Symbolic Uniform-Cost Search

December 19, 2016 10 / 32

```
Symbolic Uniform-Cost Search
Breadth-first Exploration with Zero-cost Operators
        while B \neq \mathbf{0} and bdd-intersection(B, goal) = 0:
                  B' := bdd-union(B', image(B, T))
             B := bdd-intersection(B', bdd-complement(closed_{g,i}))
             closed_{g,i} := bdd-union(B, closed_{g,i-1})
```

11 / 32

Symbolic Uniform-Cost Search

def symbolic-uniform-cost(V , I , O , γ):	
$\mathit{goal} := \mathit{build}{ extsf{-BDD}}(\gamma)$	
$\mathcal{T}:=$ make-transition-relations (V,O)	
$open_0 := bdd$ -state(1)	
while $\exists g : open_g \neq 0$:	
$g := \min\{g \mid open_g \neq 0\}$	
$\textit{open}_{g} := \textit{bfs-zero}(\check{\textit{open}}_{g}, g, \mathcal{T}, \textit{goal})$	
$closed_g := open_g$	
if bdd-intersection(open _g , goal) \neq 0 :	
return construct-plan(1, O, goal, closed _* , g)	
for all $(T, c) \in \mathcal{T}$ with $c > 0$:	
$open_{g+c} := bdd$ -union($open_{g+c}$,	
$image(open_g, T))$	
$open_g := 0$	
return unsolvable	
M. Helmert, G. Röger (Universität Basel) Planning and Optimization December 19, 2016 13 /	/ 32

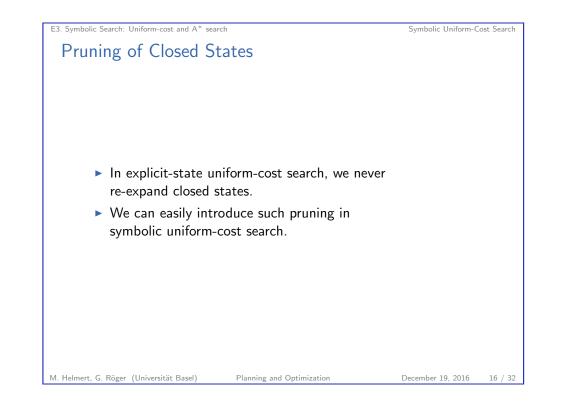
E3. Symbolic Search: Uniform-cost and A* search Symbolic Uniform-Cost Search Plan Extraction: Zero-Cost Plan Fragment **def** get-to-bfs-level- $0(cut, g, closed_{g,*}, \pi, O)$: level := 0while *bdd-intersection*(*cut*, *closed*_{g,level}) = $\mathbf{0}$: level := level + 1while *level* \neq 0: for $o \in O$ with cost(o) = 0: $pre := pre-image(cut, \tau_V(o))$ if c := bdd-intersection(pre, closed_{g,level-1}) $\neq \mathbf{0}$: cut := clevel := level - 1 $\pi := \langle o \rangle \pi$ break return *cut*, π

Planning and Optimization

Plan Extraction with Zero-cost Operators

Needs all closed sets form bfs-zero and symbolic-uniform-cost.

def construct-plan(*I*, *O*, *goal*, *closed*_{*,*}, *g*): cut := bdd-intersection(goal, closed_g) *init* := *bdd-state*(*I*); $\pi := \langle \rangle$ while *bdd-intersection*(*cut*, *init*) = $\mathbf{0}$: $cut, \pi := get-to-bfs-level-0(cut, g, closed_{g,*}, \pi, O)$ if g = 0: return π for $o \in O$ with cost(o) > 0: $pre := pre-image(cut, \tau_V(o))$ if c := bdd-intersection(pre, closed_{g-cost(o)}) $\neq 0$: *cut* := *c*; π := $\langle o \rangle \pi$ g := g - cost(o)break return π M. Helmert, G. Röger (Universität Basel) Planning and Optimization December 19, 2016 14 / 32



15 / 32

Symbolic Uniform-Cost Search

Symbolic Uniform-Cost Search

December 19, 2016

17 / 32

Uniform-Cost Search with Pruning of Closed States

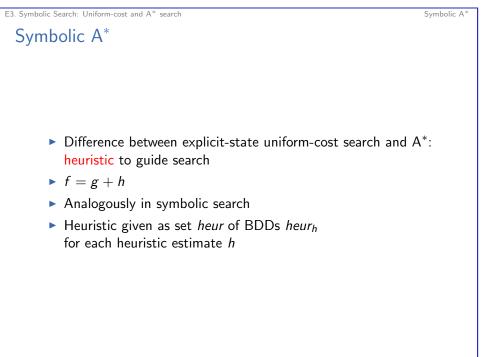
```
def symbolic-uniform-cost(V, I, O, \gamma):
      goal := build-BDD(\gamma)
      \mathcal{T} := make-transition-relations(V, O)
      open_0 := bdd-state(I)
      while \exists g : open_{\sigma} \neq \mathbf{0}:
            g := \min\{\overline{g} \mid open_g \neq \mathbf{0}\}
            open_g := bfs-zero(open_g, g, T, goal, closed_*)
            closed_g := open_g
            if bdd-intersection(open<sub>g</sub>, goal) \neq 0:
                  return construct-plan(1, O, goal, closed<sub>*</sub>, g)
            for all (T, c) \in \mathcal{T} with c > 0:
                  open_{g+c} := bdd-union(open_{g+c},
                                                image(open_{\sigma}, T))
            open_g := \mathbf{0}
      return unsolvable
```

Planning and Optimization

E3. Symbolic Search: Uniform-cost and A* search Symbolic A* E3.3 Symbolic A* Planning and Optimization December 19, 2016 19 / 32 E3. Symbolic Search: Uniform-cost and A* search

bfs-zero with Pruning of Closed States

def bfs-zero(B, g, T, goal, prune): for $P \in prune$: B := bdd-intersection(B, bdd-complement(P)) i := 0 $closed_{g,i} := B$ while $B \neq \mathbf{0}$ and *bdd-intersection*(B, *goal*) = **0**: B' := 0for $(T, c) \in \mathcal{T}, c = 0$: B' := bdd-union(B', image(B, T))B := bdd-intersection(B', bdd-complement($closed_{g,i}$)) for $P \in prune$: B := bdd-intersection(B, bdd-complement(P)) i := i + 1 $closed_{g,i} := bdd-union(B, closed_{g,i-1})$ return closed_{g,i} M. Helmert, G. Röger (Universität Basel) Planning and Optimization December 19, 2016 18 / 32



M. Helmert, G. Röger (Universität Basel)

M. Helmert, G. Röger (Universität Basel)

Symbolic A* (with Consistent Heuristic)

def symbolic-AStar(V, I, O, γ , heur): $goal := build-BDD(\gamma)$ $\mathcal{T} := make-transition-relations(V, O)$ $open_{0,h(I)} := bdd-state(I)$ while $\exists g, h : open_{g,h} \neq \mathbf{0}$: $f := \min\{f \mid \exists g, h : open_{g,h} \neq \mathbf{0}, f = g + h\}$ $g := \min\{g \mid \exists h : open_{g,h} \neq \mathbf{0}, f = g + h\}$ $open_{g,*} := expand_0(open_{*,*}, g, h, \mathcal{T}, goal, heur, closed_*)$ $closed_g := bdd-union(closed_g, open_{g,h})$ if bdd-intersection($open_{g,h}, goal$) $\neq \mathbf{0}$: $return \ construct-plan(I, O, goal, closed_*, g)$ $open_{*,*} := expand_{>0}(open_{*,*}, g, h, \mathcal{T}, heur)$ $open_{g,h} := \mathbf{0}$ return unsolvable For performance it is important to expand the minimum g value.

Planning and Optimization

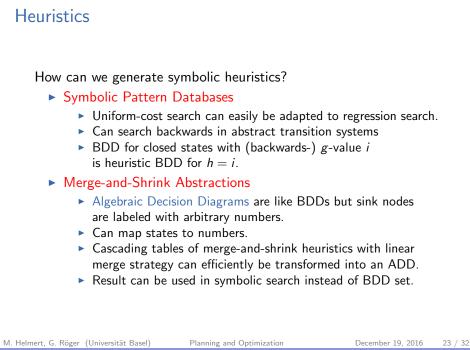
M. Helmert, G. Röger (Universität Basel)

December 19, 2016 21 / 32

Symbolic A*

Symbolic A

E3. Symbolic Search: Uniform-cost and A^* search



```
E3. Symbolic Search: Uniform-cost and A* search
                                                                                                  Symbolic A
  Expand States and Update Open Lists
       def expand<sub>0</sub>(open<sub>*,*</sub>, g, h, \mathcal{T}, goal, heur, prune):
              B := bfs-zero(open_{g,h}, (g, h), \mathcal{T}, goal, prune)
             for heur_{h'} \in heur, h \leq h' < \infty:
                    B' := bdd-intersection(heur<sub>h'</sub>, open-zero)
                    open_{g,h'} := bdd-union(open_{g,h'}, B')
             return open<sub>\sigma</sub> *
       def expand<sub>>0</sub>(open<sub>* *</sub>, g, h, \mathcal{T}, heur):
             for all (T, c) \in \mathcal{T}, c > 0:
                    B := image(open_{\sigma h}, T)
             for heur_{h'} \in heur, h - c \le h' < \infty:
                    B' := bdd-intersection(heur<sub>h'</sub>, open-zero)
                    open_{\sigma+c,h'} := bdd-union(open_{\sigma+c,h'}, B')
             return open<sub>**</sub>
M. Helmert, G. Röger (Universität Basel)
                                           Planning and Optimization
                                                                                  December 19, 2016
                                                                                                      22 / 32
E3. Symbolic Search: Uniform-cost and A* search
                                                                                                    Discussio
```

E3.4 Discussion

Planning and Optimization

