University of Basel

Extending SymPA with Unsolvability Certificates

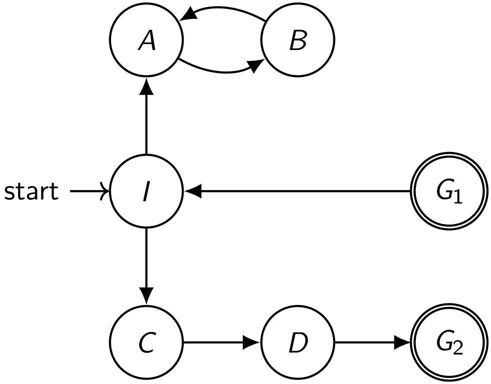
Bachelor Thesis

Claudia Grundke

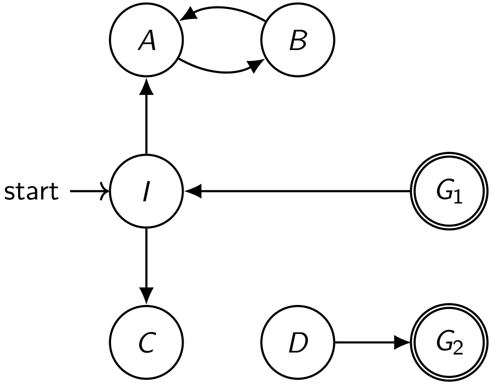
University of Basel
Departement of Mathematics and Computer Science

18 September, 2020

Classical Planning

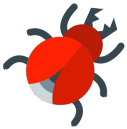

$$\begin{tikzcd}
 & A \arrow[r, bend right] & B \arrow[l, bend right] & \\
\text{start} \arrow[r] & I \arrow[u] \arrow[d] & & G_1 \arrow[ll] \\
 & C \arrow[r] & D \arrow[r] & G_2
\end{tikzcd}$$

Classical Planning



Correctness

How can we be sure that the planner gave the correct result?



https://icon-icons.com/icon/Bug/115148 (09.09.2020)

Certifying Algorithms

- emit certificate alongside result
- certificate justifies result and can be verified independently

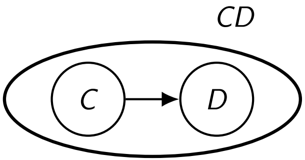
Certifying Algorithms

- emit certificate alongside result
- certificate justifies result and can be verified independently

- partially and fully certifying algorithms
- plan can serve as certificate

SymPA (Álvaro Torralba)

- **Sym**bolic **P**erimeter **A**bstractions
- forward and backward breadth first searches
- returns "solvable"
- or "unsolvable"

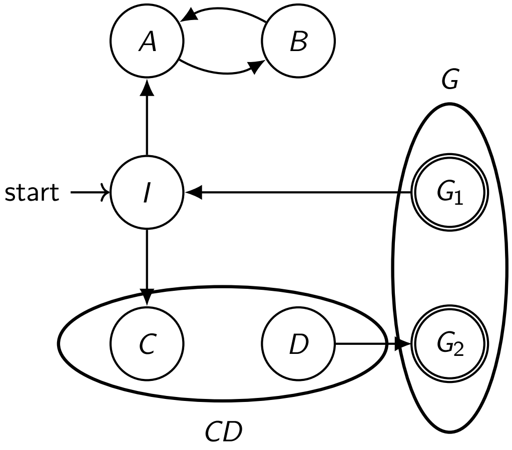


Certifying SymPA

- **Sym**bolic **P**erimeter **A**bstractions
- forward and backward breadth first searches

- returns plan
- or unsolvability certificate

Abstract Backward Search



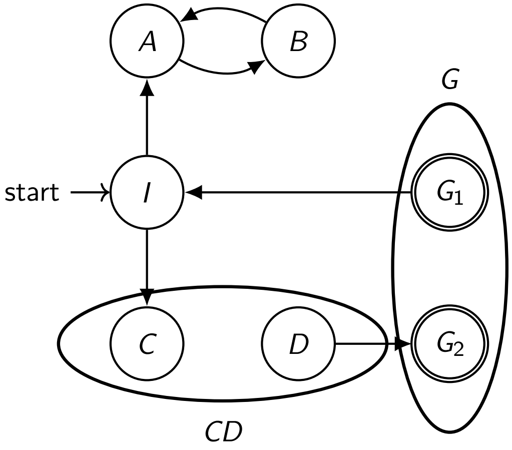
open list:

G

closed list:

-

Abstract Backward Search



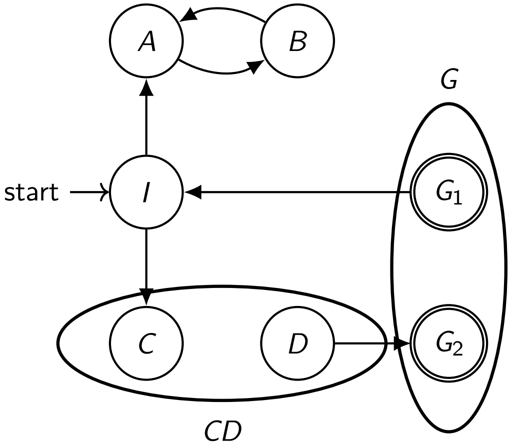
open list:

CD

closed list:

G

Abstract Backward Search



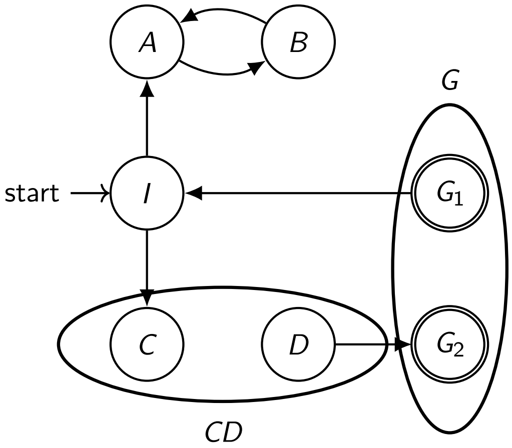
open list:

I

closed list:

G, CD

Abstract Backward Search



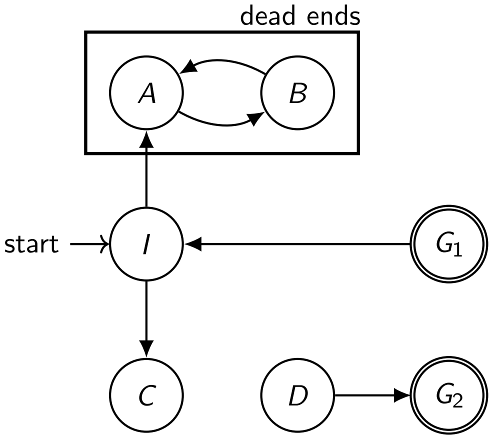
open list:

-

closed list:

G, CD, I

Forward Search



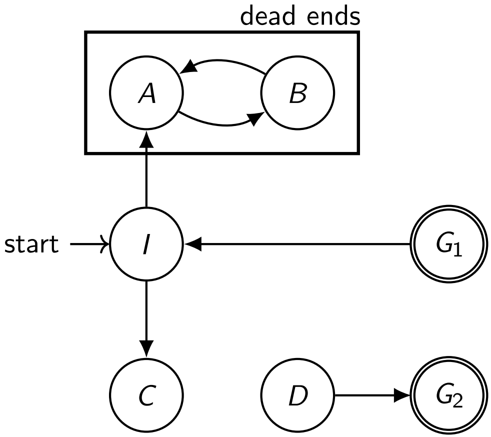
open list:

I

closed list:

-

Forward Search



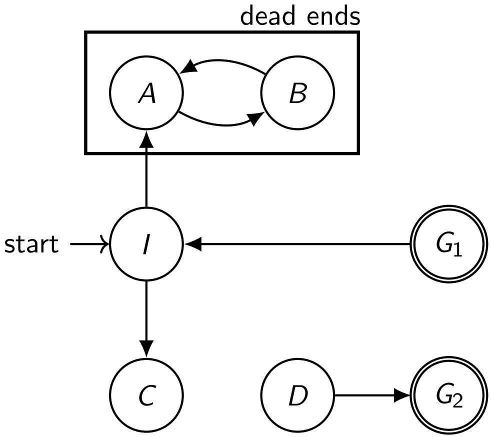
open list:

C

closed list:

I

Forward Search



open list:

-

closed list:

I, C

Unsolvability Proof System (Salomé Eriksson)

- proofs serve as unsolvability certificates
- core concept: dead states

Unsolvability Proof System (Salomé Eriksson)

- proofs serve as unsolvability certificates
- core concept: dead states
- initial state or all goal states dead $\Rightarrow$ task unsolvable

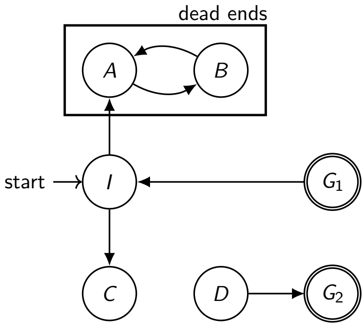
Forward Unsolvability Proof

- **D** is dead
    -
    -
- **CL** is dead
    -
    -
- initial state is in **CL**

D = set of dead ends, **CL** = closed list

Forward Unsolvability Proof

- **D** is dead
  - **D** contains no goal state
  - **D** cannot be left
- **CL** is dead
- initial state is in **CL**



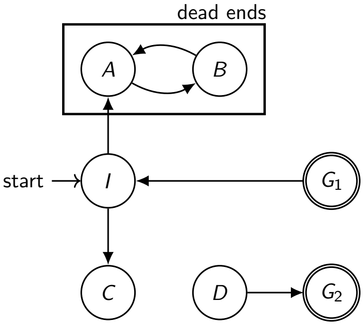
open list: -

closed list: *I*, *C*

D = set of dead ends, **CL** = closed list

Forward Unsolvability Proof

- **D** is dead
  - **D** contains no goal state
  - **D** cannot be left
- **CL** is dead
  - **CL** contains no goal state
  - all successors of **CL** either in **CL** itself or in **D**
- initial state is in **CL**

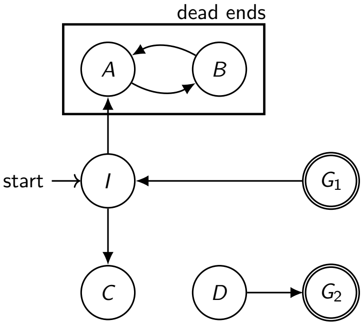


open list:	closed list:
-	I, C

D = set of dead ends, **CL** = closed list

Forward Unsolvability Proof

- **D** is dead
  - **D** contains no goal state
  - **D** cannot be left
- **CL** is dead
  - **CL** contains no goal state
  - all successors of **CL** either in **CL** itself or in **D**
- initial state is in **CL**



open list:	closed list:
-	I, C

D = set of dead ends, **CL** = closed list

Forward Unsolvability Proof

- **D** is dead
  - **D** contains no goal state
  - **D** cannot be left
- **CL** is dead
  - **CL** contains no goal state
  - all successors of **CL** either in **CL** itself or in **D**
- initial state is in **CL**

$\Rightarrow$ initial state dead

D = set of dead ends, **CL** = closed list

Forward Unsolvability Proof

- **D** is dead
  - **D** contains no goal state
  - **D** cannot be left
- **CL** is dead
  - **CL** contains no goal state
  - all successors of **CL** either in **CL** itself or in **D**
- initial state is in **CL**

$\Rightarrow$ initial state dead
$\Rightarrow$ **task unsolvable**

D = set of dead ends, **CL** = closed list

Evaluation

total number of tasks	352
returned "unsolvable"	104
valid certificates	83

average time	
search	1.11s
certificate generation	2.74s
verification	33.80s

Remaining Errors

- **D** is dead
  - **D** contains no goal state
  - **D** cannot be left
- **CL** is dead
  - **CL** contains no goal state
  - all successors of **CL** either in **CL** itself or in **D**
- initial state is in **CL**

$\Rightarrow$ initial state dead
$\Rightarrow$ **task unsolvable**

D = set of dead ends, **CL** = closed list

Summary

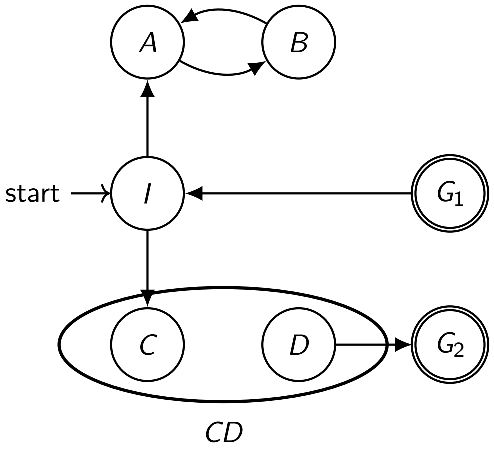
- fully certifying version of SymPA
- verifiable unsolvability proofs

Questions?

Comments?

Discussion!

Goals intersect Dead Ends



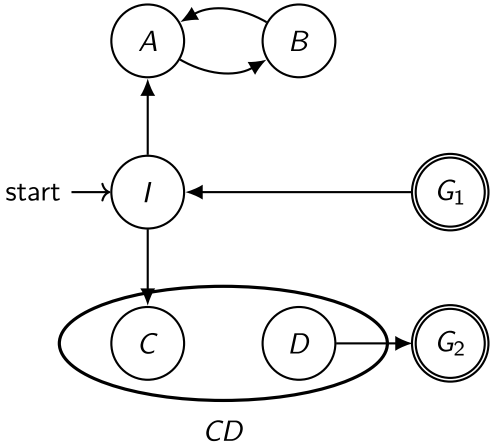
abstract forward search

open list:	I
closed list:	-
D_{bw}:	-

abstract backward search

open list:	G_1, G_2
closed list:	-
D_{fw}:	-

Goals intersect Dead Ends



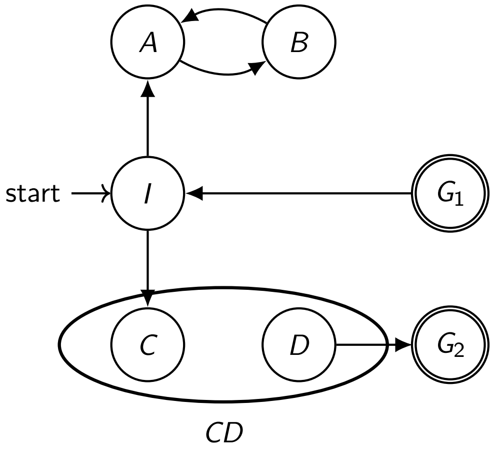
abstract forward search

open list:	A, CD
closed list:	I
D_{bw}:	-

abstract backward search

open list:	G_1, G_2
closed list:	-
D_{fw}:	-

Goals intersect Dead Ends



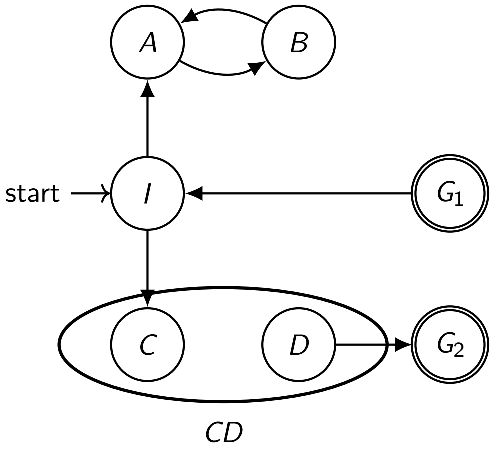
abstract forward search

open list:	B, CD
closed list:	I, A
D_{bw}:	-

abstract backward search

open list:	G_1, G_2
closed list:	-
D_{fw}:	-

Goals intersect Dead Ends



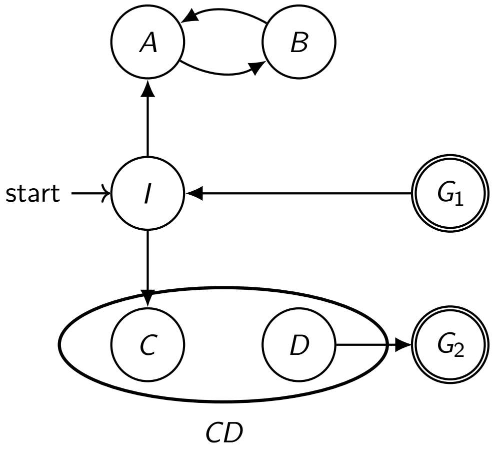
abstract forward search

open list:	CD
closed list:	I, A, B
D_{bw}:	-

abstract backward search

open list:	G_1, G_2
closed list:	-
D_{fw}:	-

Goals intersect Dead Ends



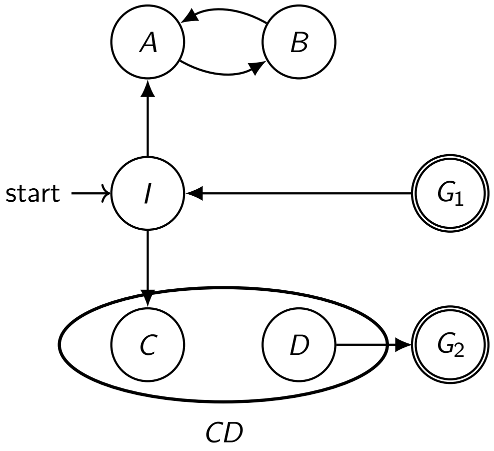
abstract forward search

open list:	G_2
closed list:	I, A, B, CD
D_{bw}:	-

abstract backward search

open list:	G_1, G_2
closed list:	-
D_{fw}:	-

Goals intersect Dead Ends



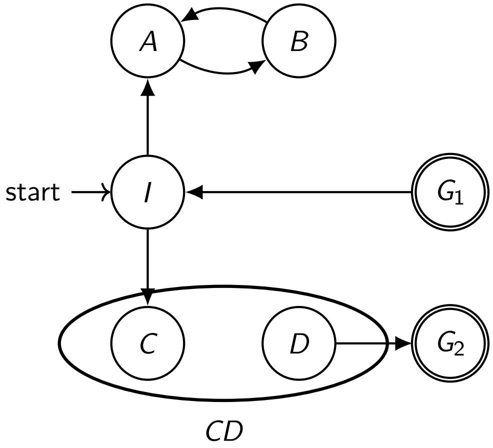
abstract forward search

open list:	-
closed list:	I, A, B, CD, G_2
D_{bw}:	-

abstract backward search

open list:	G_1, G_2
closed list:	-
D_{fw}:	-

Goals intersect Dead Ends



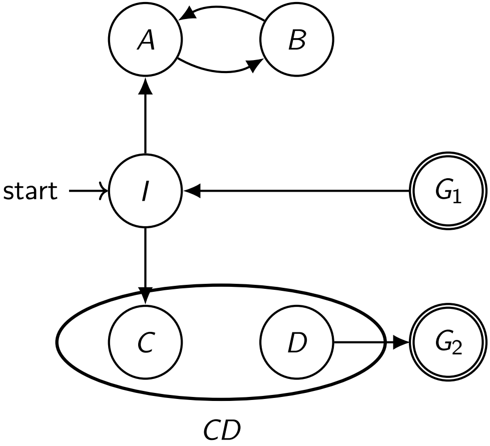
abstract forward search

open list:	-
closed list:	I, A, B, CD, G_2
D_{bw}:	G_1

abstract backward search

open list:	G_1, G_2
closed list:	-
D_{fw}:	-

Goals intersect Dead Ends



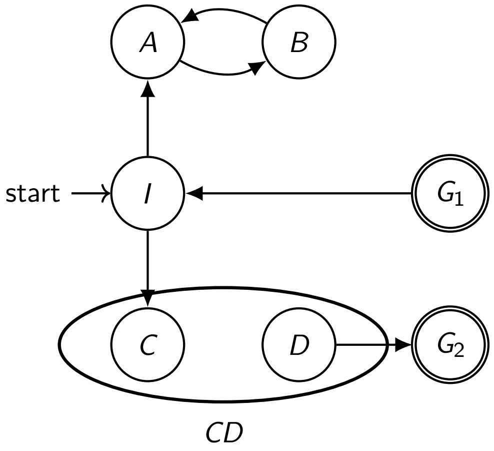
abstract forward search

open list:	-
closed list:	I, A, B, CD, G_2
D_{bw}:	G_1

abstract backward search

open list:	G_2
closed list:	-
D_{fw}:	-

Goals intersect Dead Ends



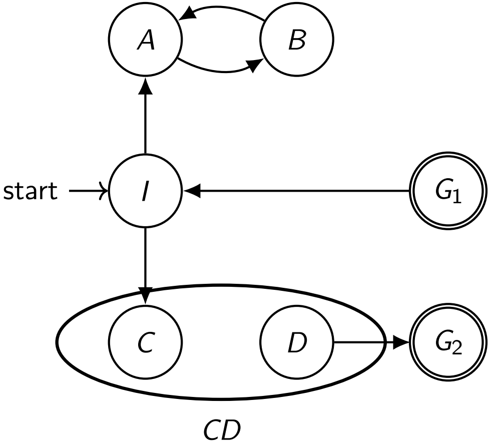
abstract forward search

open list:	-
closed list:	I, A, B, CD, G_2
D_{bw}:	G_1

abstract backward search

open list:	CD
closed list:	G_2
D_{fw}:	-

Goals intersect Dead Ends



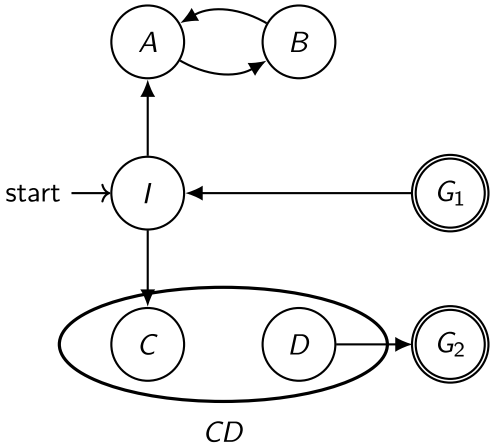
abstract forward search

open list:	-
closed list:	I, A, B, CD, G_2
D_{bw}:	G_1

abstract backward search

open list:	I
closed list:	G_2, CD
D_{fw}:	-

Goals intersect Dead Ends



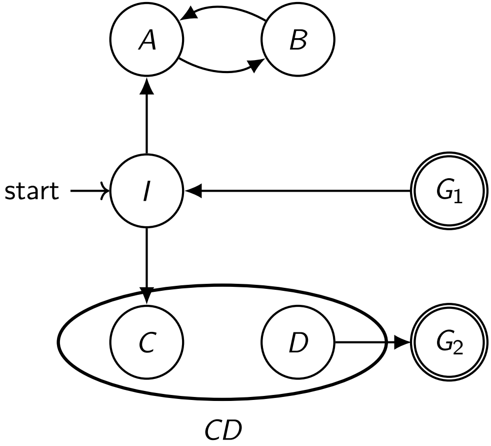
abstract forward search

open list:	-
closed list:	I, A, B, CD, G_2
D_{bw}:	G_1

abstract backward search

open list:	-
closed list:	G_2, CD, I
D_{fw}:	-

Goals intersect Dead Ends



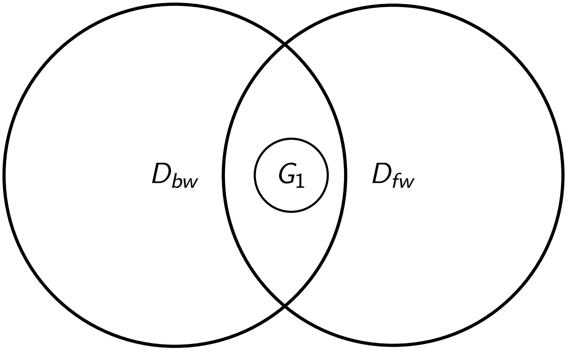
abstract forward search

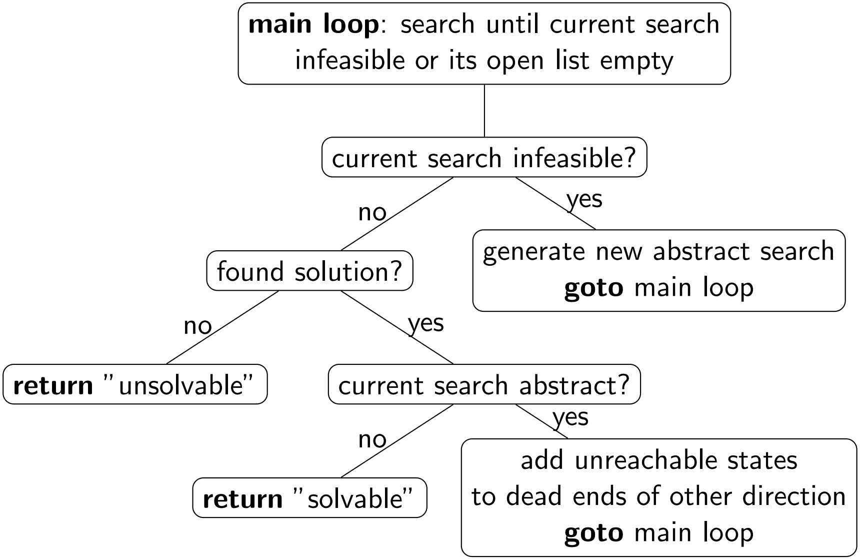
open list:	-
closed list:	I, A, B, CD, G_2
D_{bw}:	G_1

abstract backward search

open list:	-
closed list:	G_2, CD, I
D_{fw}:	A, B, G_1

Goals intersect Dead Ends




main loop: search until current search infeasible or its open list empty
current search infeasible?
no
yes
found solution?
generate new abstract search **goto** main loop
no
yes
return "unsolvable"
current search abstract?
no
yes
return "solvable"
add unreachable states to dead ends of other direction **goto** main loop

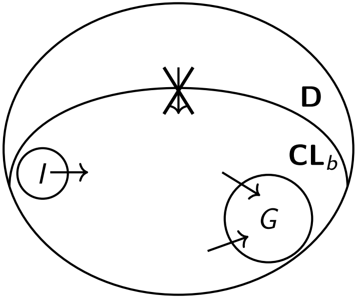
SymPA Algorithm

```
1.  search until current search infeasible or its open list empty
2.  **if** current search infeasible:
3.      start new abstract search
4.      **goto** 1.
5.  **if** found no solution:
6.      **return** "unsolvable"
7.  **if** search is not abstract:
8.      **return** "solvable"
9.  add unreachable states to dead ends of other direction
10. **goto** 1.
```

Certifying SymPA

```
1.  search until current search infeasible or its open list empty
2.  **if** current search infeasible:
3.      start new abstract search
4.      **goto** 1.
5.  **if** found no solution:
6.      **return** ~~"unsolvable"~~ generate unsolvability certificate
7.  **if** search is not abstract:
8.      **return** ~~"solvable"~~ generate plan
9.  add unreachable states to dead ends of other direction
10. **goto** 1.
```

Forward Dead Ends



Implementation

- only unsolvability certificates
- certificate consists of three files
- conversion multivalued variables to binary variables