Planning with Specialized SAT Solvers

(NECTAR)

Jussi Rintanen

August 11, 2011

Contribution of the Work

- Heuristics for SAT-based planning (classical, non-optimizing), replacing VSIDS et al. in CDCL (Rintanen CP'10, AI'10)
  - A form of backward chaining with CDCL
  - Substantial speed-up for finding plans for most problem types
  - Applicable to almost all notions of plans used with SAT
- Experimental results: outperforms other planners

Background: Planning

- Since (Bonet & Geffner 1998), classical planning has become a monoculture of heuristic / explicit state-space search.
- Impact to the research topic:
  - Progress slow after what was initially perceived as successes.
  - Vast majority of published research incremental, implementation-oriented.
  - Other, promising approaches are being ignored.

Background: Planning with SAT

- Kautz & Selman (1996, 1999) still the starting point of recent works (-2010).
- Main research topics:
  - Encodings
  - Constraints to prune SAT solver search space.
- Progress small:
  - Early encodings (Kautz & Selman 1996) no less efficient than recent ones (see Sideris & Dimopoulos (2009).)
  - Runtime improvements small multiplicative factor from Kautz & Selman 1996 encodings, at most.
- Explanation for small progress: efficiency actually little affected by encoding!

Development of Planning as SAT

(As relevant to the planning techniques in this work)

1992-99	the approach is first developed	Kautz & Selman etc.
2004-06	practical (linear-size) encodings no more memory overflows	Rintanen et al.
2004-06	interleaved search strategies efficiency close to best planners	Rintanen
2010	planning-specific heuristics for SAT efficiency $\geq$ best planners	Rintanen

Development of Planning as SAT

(As relevant to the planning techniques in this work)

1992-99	the approach is first developed	Kautz & Selman etc.
2004-06	practical (linear-size) encodings no more memory overflows	Rintanen et al.
2004-06	interleaved search strategies efficiency close to best planners	Rintanen
2010	planning-specific heuristics for SAT efficiency $\geq$ best planners	Rintanen

Development of Planning as SAT

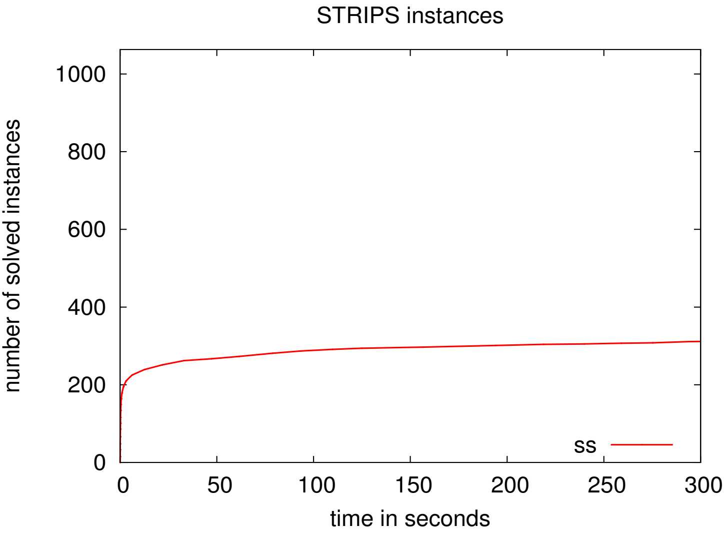
(As relevant to the planning techniques in this work)

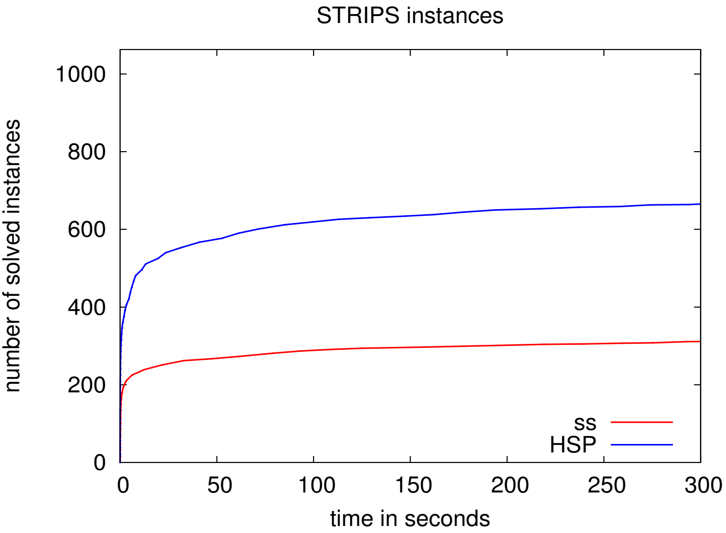
1992-99	the approach is first developed	Kautz & Selman etc.
2004-06	practical (linear-size) encodings no more memory overflows	Rintanen et al.
2004-06	interleaved search strategies efficiency close to best planners	Rintanen
2010	planning-specific heuristics for SAT efficiency $\geq$ best planners	Rintanen

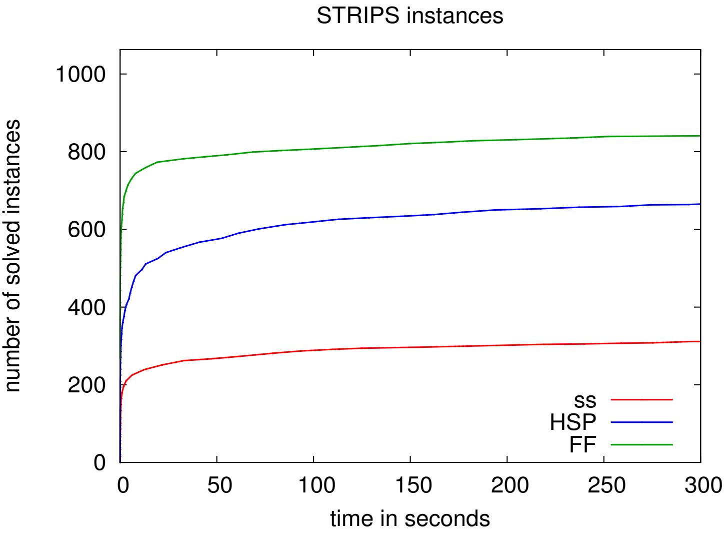
Development of Planning as SAT

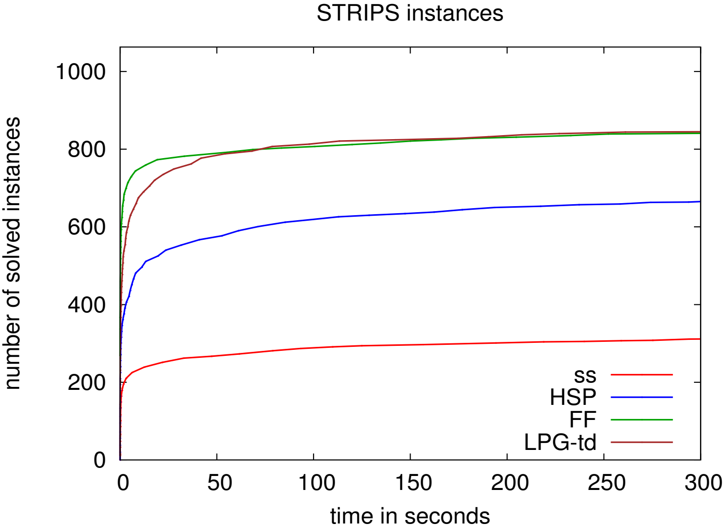
(As relevant to the planning techniques in this work)

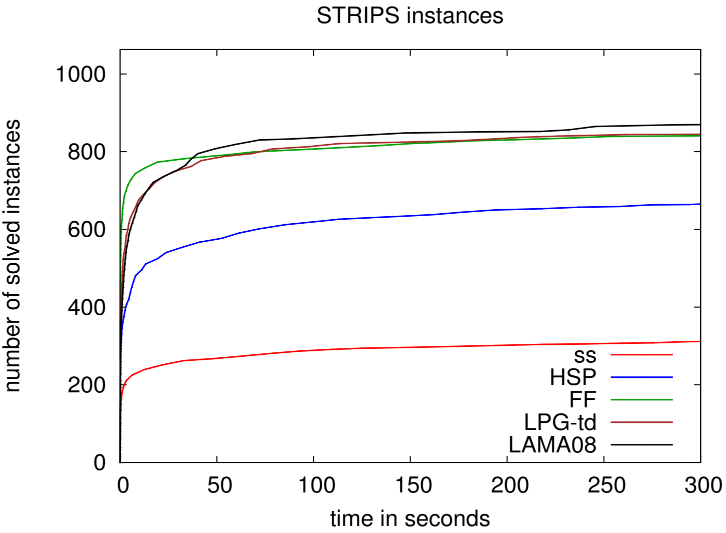
1992-99	the approach is first developed	Kautz & Selman etc.
2004-06	practical (linear-size) encodings no more memory overflows	Rintanen et al.
2004-06	interleaved search strategies efficiency close to best planners	Rintanen
2010	planning-specific heuristics for SAT efficiency $\geq$ best planners	Rintanen

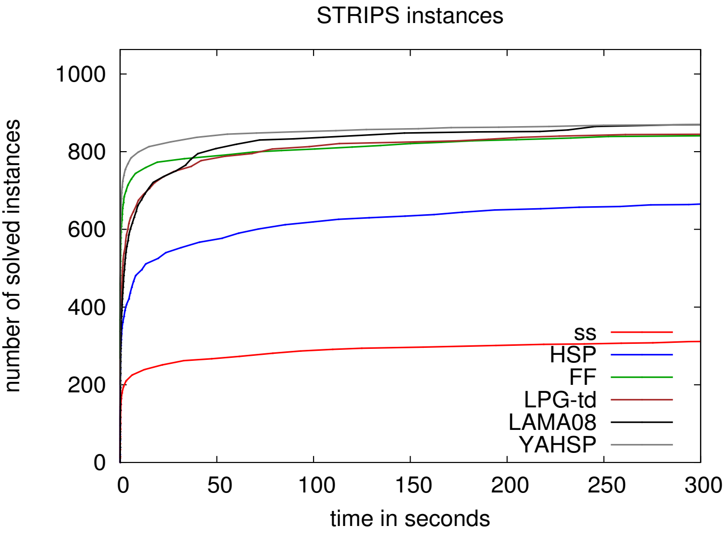

STRIPS instances
number of solved instances
0
200
400
600
800
1000
0
50
100
150
200
250
300
time in seconds
ss


STRIPS instances
number of solved instances
0
200
400
600
800
1000
0
50
100
150
200
250
300
time in seconds
ss
HSP

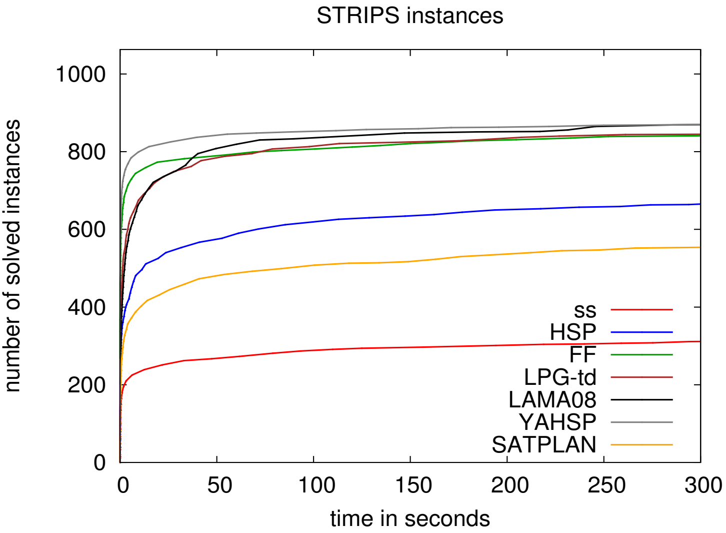

STRIPS instances
number of solved instances
0
200
400
600
800
1000
0
50
100
150
200
250
300
time in seconds
ss
HSP
FF

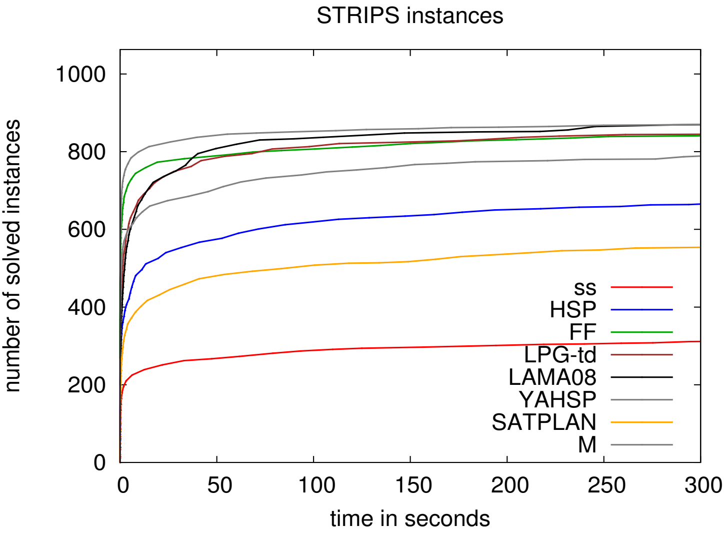

STRIPS instances
number of solved instances
1000
800
600
400
200
0
0
50
100
150
200
250
300
time in seconds
ss
HSP
FF
LPG-td

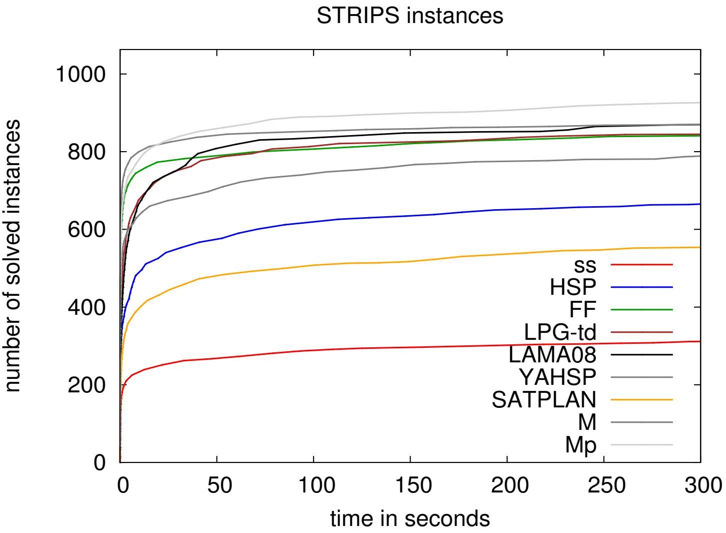

STRIPS instances
number of solved instances
0
200
400
600
800
1000
time in seconds
0
50
100
150
200
250
300
ss
HSP
FF
LPG-td
LAMA08



STRIPS instances
number of solved instances
0
200
400
600
800
1000
0
50
100
150
200
250
300
time in seconds
ss
HSP
FF
LPG-td
LAMA08
YAHSP


STRIPS instances
number of solved instances
time in seconds
0
200
400
600
800
1000
0
50
100
150
200
250
300
ss
HSP
FF
LPG-td
LAMA08
YAHSP
SATPLAN


STRIPS instances
number of solved instances
time in seconds
0
200
400
600
800
1000
0
50
100
150
200
250
300
ss
HSP
FF
LPG-td
LAMA08
YAHSP
SATPLAN
M


STRIPS instances
number of solved instances
0
200
400
600
800
1000
0
50
100
150
200
250
300
time in seconds
ss
HSP
FF
LPG-td
LAMA08
YAHSP
SATPLAN
M
Mp

Our Heuristics

- Replace VSIDS style heuristic by a planning-specific one in the Conflict-Driven Clause Learning (CDCL) algorithm for SAT.
- Force SAT solver to do backward chaining:
  1. Start from a top-level goal literal.
  2. Go backward to the time where the literal can turn from false to true.
  3. Choose an action to justify that change.
  4. Use the action variable as the CDCL decision variable.
  5. If such action there already, do the same with its preconditions.

The new planning heuristic for CDCL

Case 1: goal/subgoal x has no support yet

Value of a state variable x at different time points:

	$t-8$	$t-7$	$t-6$	$t-5$	$t-4$	$t-3$	$t-2$	$t-1$	t
x	0	0	0		1		1	1	$\mathbf{1}$
action 1	0	0	0			0	0	0	
action 2	0	0		0			0		
action 3	0	0	0	0		0	0		
action 4	0	0			0	0			

The new planning heuristic for CDCL

Case 1: goal/subgoal x has no support yet

Actions that make x true:

	$t-8$	$t-7$	$t-6$	$t-5$	$t-4$	$t-3$	$t-2$	$t-1$	t
x	0	0	0		1		1	1	$\mathbf{1}$
action 1	0	0	0			0	0	0	
action 2	0	0		0			0		
action 3	0	0	0	0		0	0		
action 4	0	0			0	0			

The new planning heuristic for CDCL

Case 1: goal/subgoal x has no support yet

Actions that make x true as early as possible (at $t-5$):

	$t-8$	$t-7$	$t-6$	$t-5$	$t-4$	$t-3$	$t-2$	$t-1$	t
x	0	0	0		1		1	1	**1**
action 1	0	0	0			0	0	0	
action 2	0	0		0			0		
action 3	0	0	0	0		0	0		
action 4	0	0			0	0			

The new planning heuristic for CDCL

Case 1: goal/subgoal x has no support yet

Choose action 2 or 4 at $t-6$ as the next CDCL decision variable.

	$t-8$	$t-7$	$t-6$	$t-5$	$t-4$	$t-3$	$t-2$	$t-1$	t
x	0	0	0		1		1	1	**1**
action 1	0	0	0			0	0	0	
action 2	0	0		0			0		
action 3	0	0	0	0		0	0		
action 4	0	0			0	0			

The new planning heuristic for CDCL

Case 2: goal/subgoal x already has support

Goal/subgoal is already made true at $t-4$ by action 4.

	$t-8$	$t-7$	$t-6$	$t-5$	$t-4$	$t-3$	$t-2$	$t-1$	t
x	0	0	0		1		1	1	**1**
action 1	0	0	0			0	0	0	
action 2	0	0		0			0		
action 3	0	0	0	0		0	0		
action 4	0	0		1	0	0			

Use precondition literals of action 4 as new subgoals at $t-5$.

The new planning heuristic for CDCL

Case 2: goal/subgoal x already has support

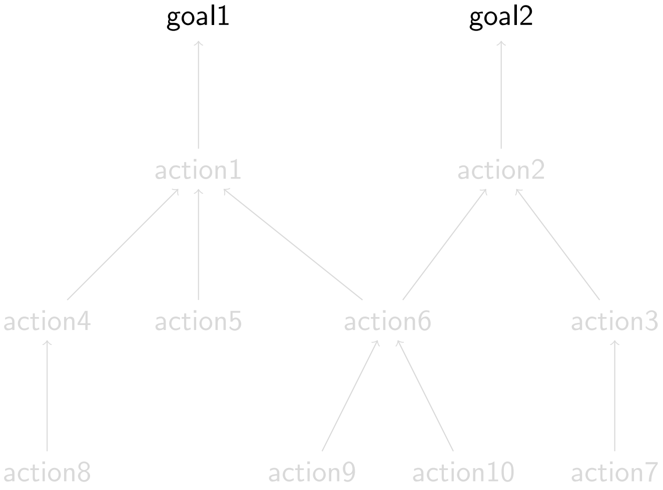
Goal/subgoal is already made true at $t-4$ by action 4.

	$t-8$	$t-7$	$t-6$	$t-5$	$t-4$	$t-3$	$t-2$	$t-1$	t
x	0	0	0		1		1	1	$\mathbf{1}$
action 1	0	0	0			0	0	0	
action 2	0	0		0			0		
action 3	0	0	0	0		0	0		
action 4	0	0		1	0	0			

Use precondition literals of action 4 as new subgoals at $t-5$.

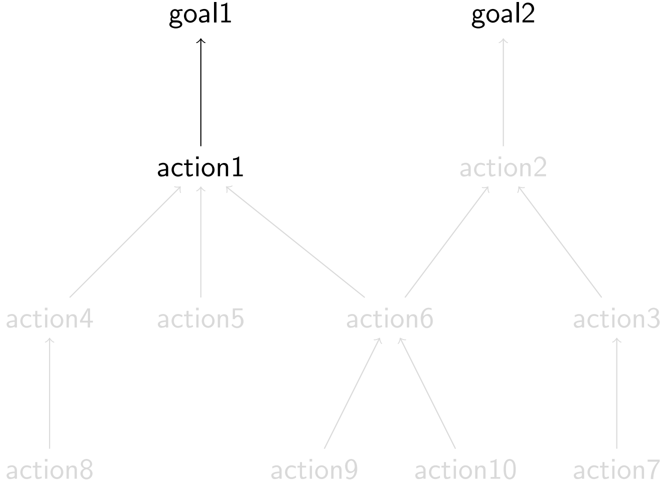
The variable selection scheme

Version 1: strict depth-first search



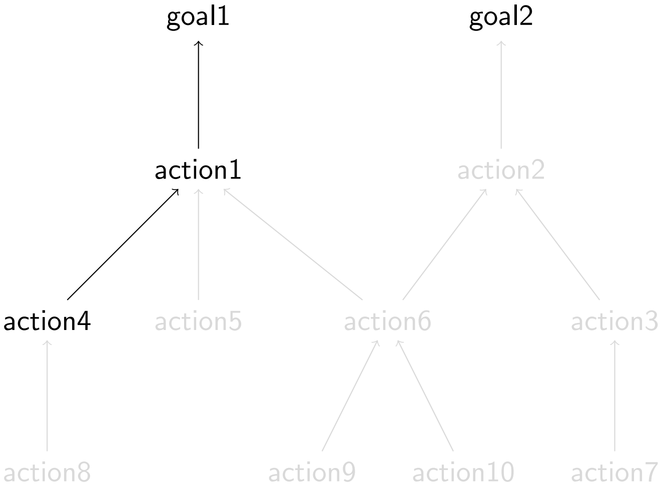
The variable selection scheme

Version 1: strict depth-first search



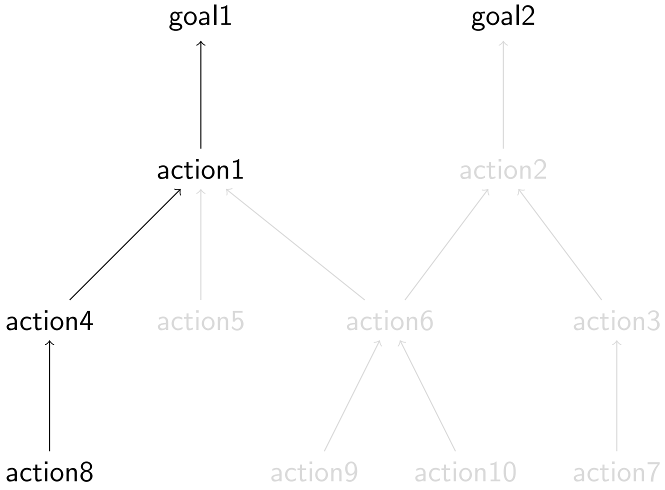
The variable selection scheme

Version 1: strict depth-first search



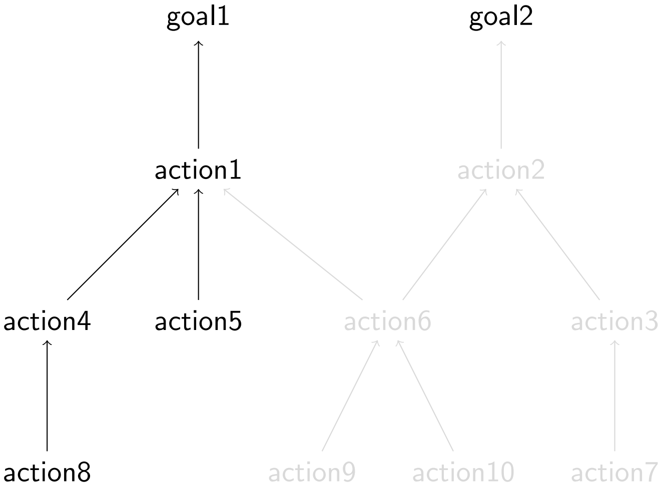
The variable selection scheme

Version 1: strict depth-first search



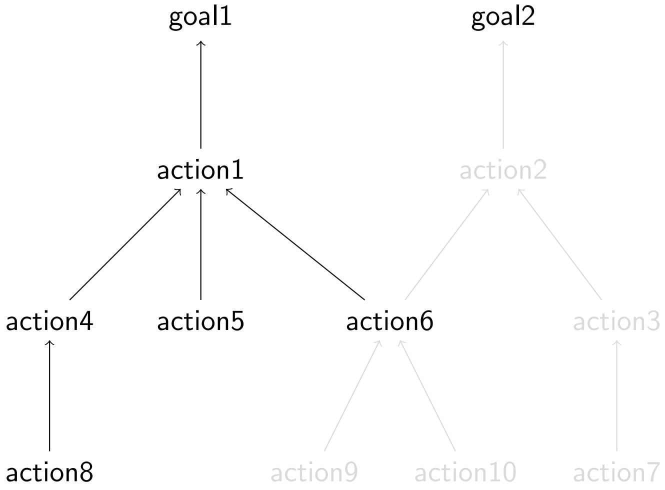
The variable selection scheme

Version 1: strict depth-first search



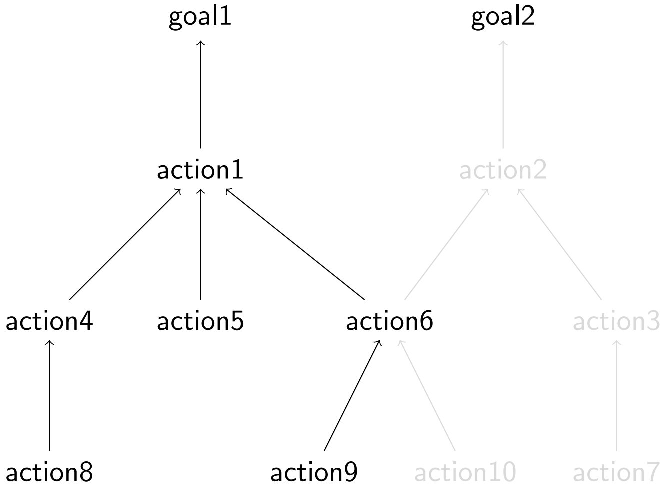
The variable selection scheme

Version 1: strict depth-first search



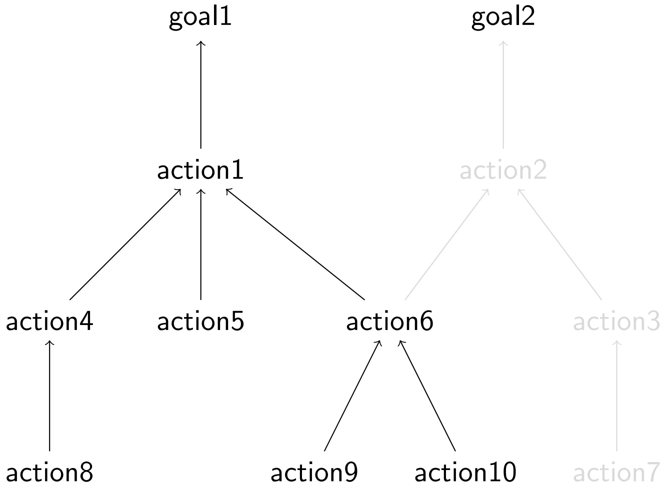
The variable selection scheme

Version 1: strict depth-first search



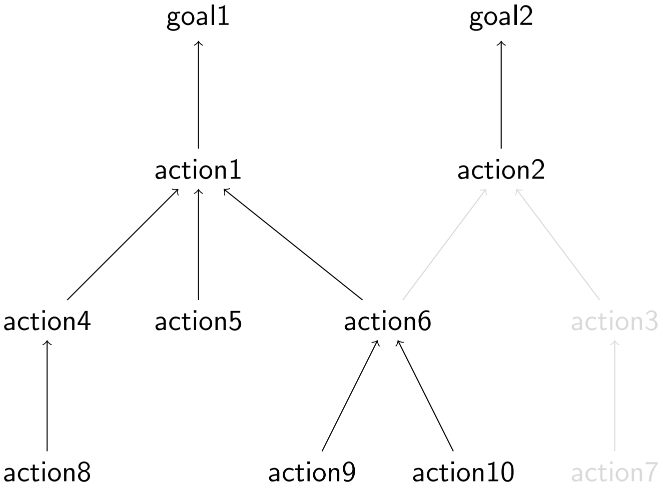
The variable selection scheme

Version 1: strict depth-first search



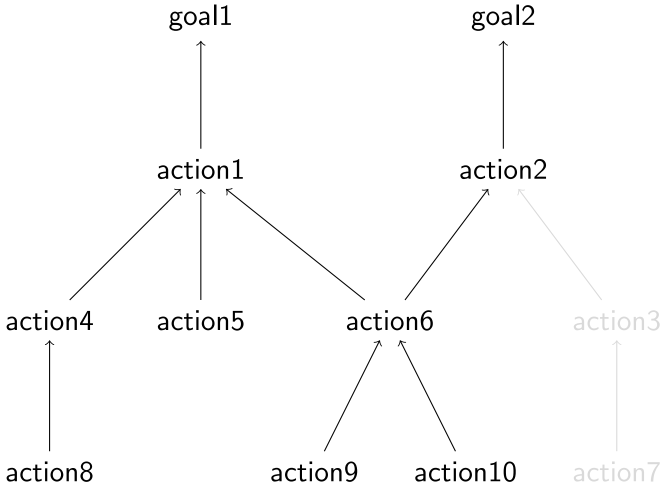
The variable selection scheme

Version 1: strict depth-first search



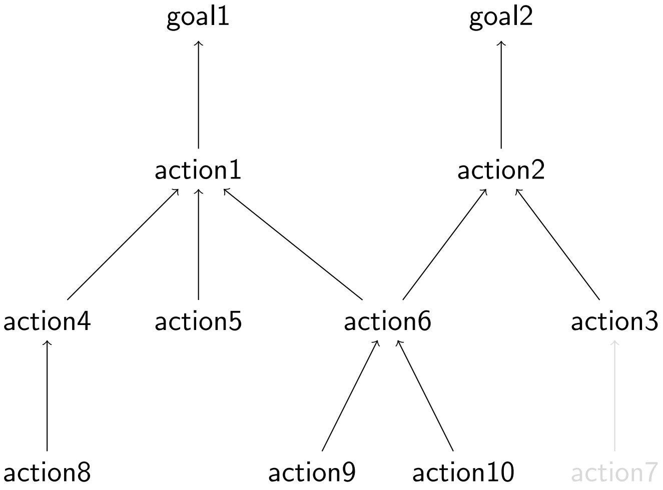
The variable selection scheme

Version 1: strict depth-first search



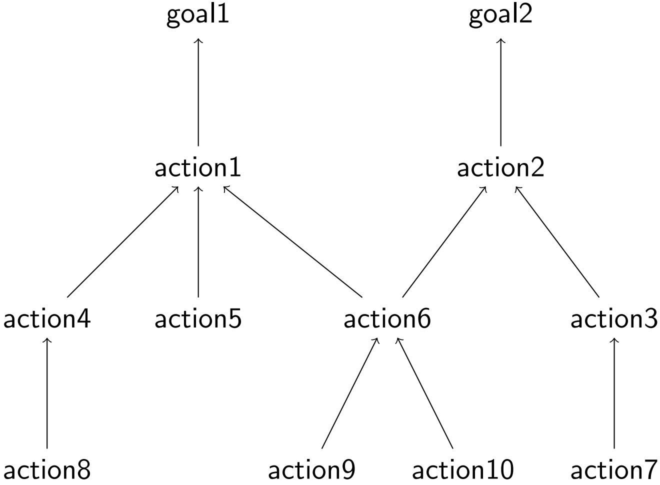
The variable selection scheme

Version 1: strict depth-first search



The variable selection scheme

Version 1: strict depth-first search

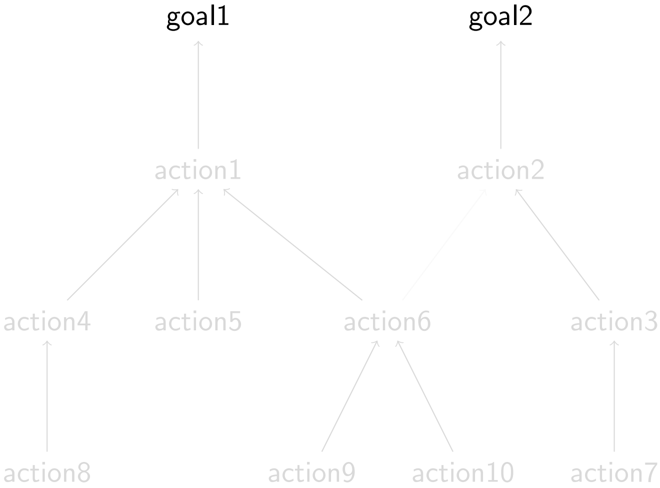


The variable selection scheme: two refinements

1. Compute several actions (directly or indirectly) supporting the goal, and randomly choose one of them.
2. Replace the stack for depth-first search by a priority queue. Use heuristics for ordering the subgoals.

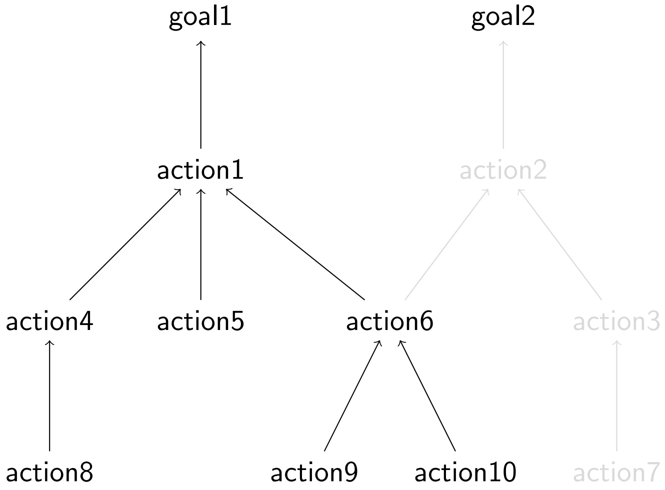
The variable selection scheme

Version 2: randomized action selection



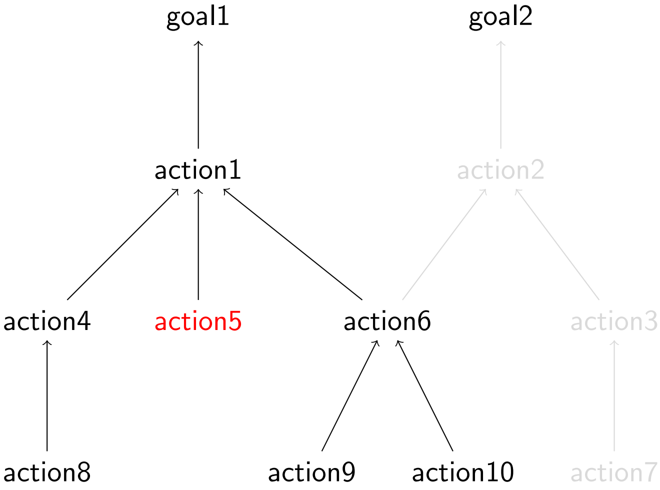
The variable selection scheme

Version 2: randomized action selection



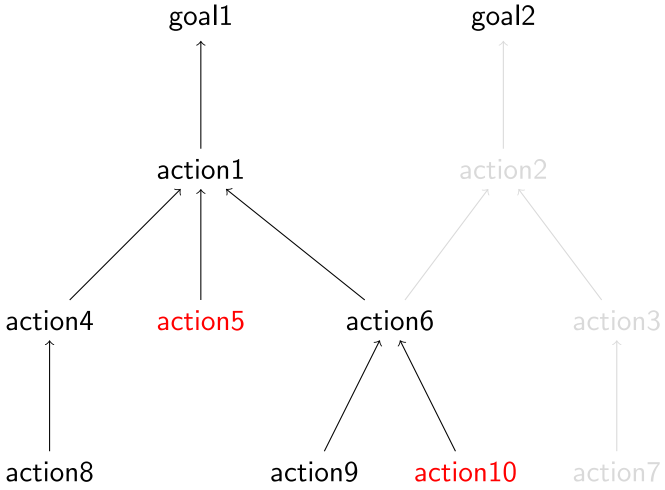
The variable selection scheme

Version 2: randomized action selection



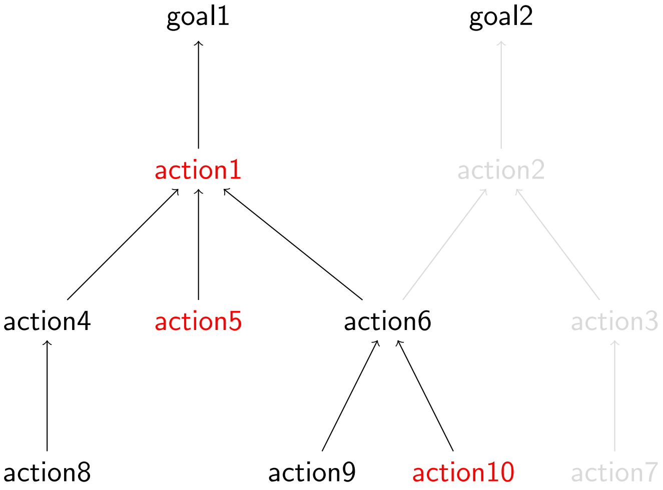
The variable selection scheme

Version 2: randomized action selection



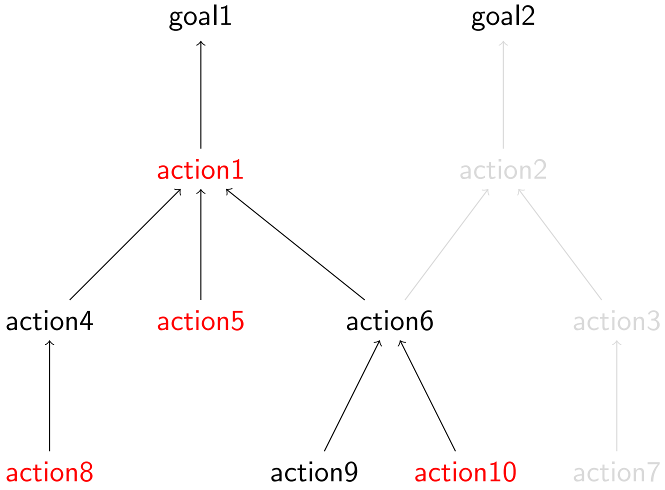
The variable selection scheme

Version 2: randomized action selection



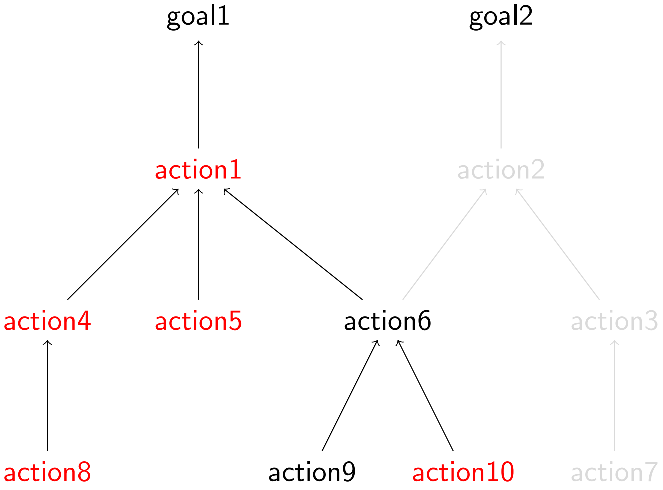
The variable selection scheme

Version 2: randomized action selection



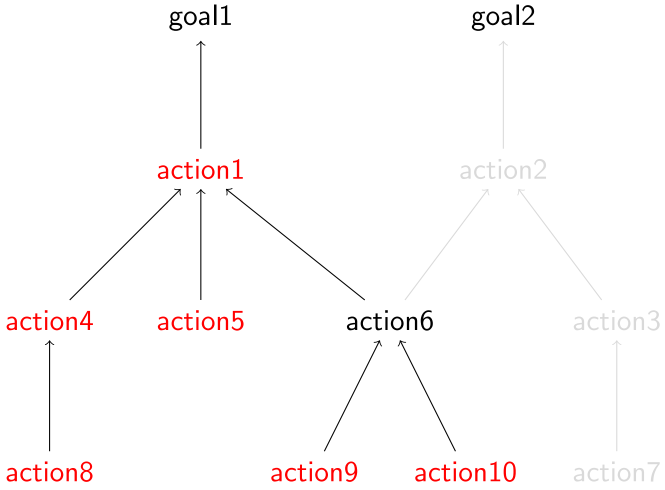
The variable selection scheme

Version 2: randomized action selection



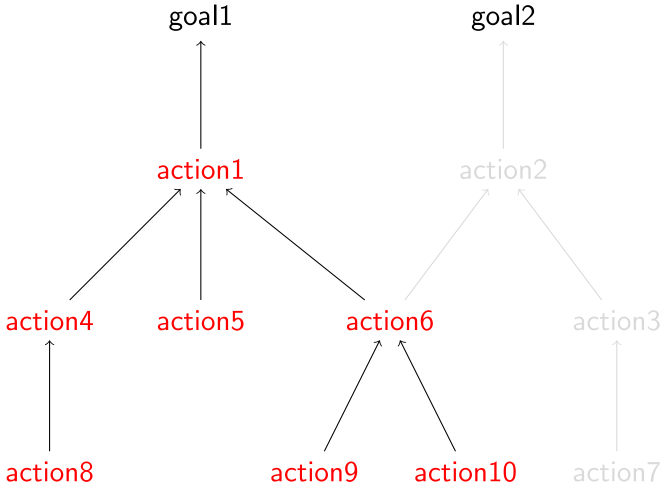
The variable selection scheme

Version 2: randomized action selection



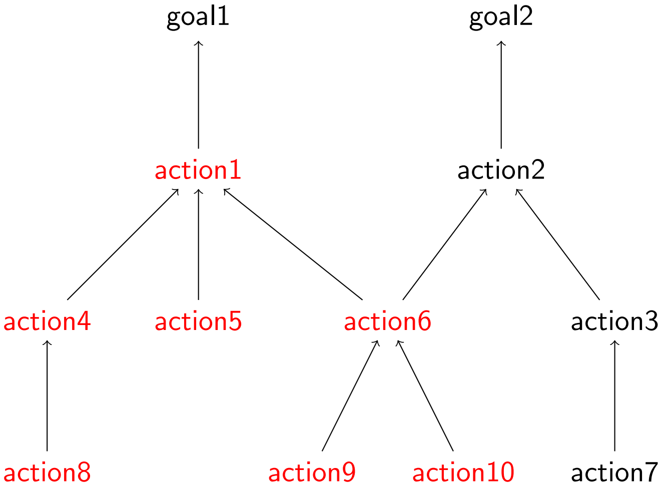
The variable selection scheme

Version 2: randomized action selection



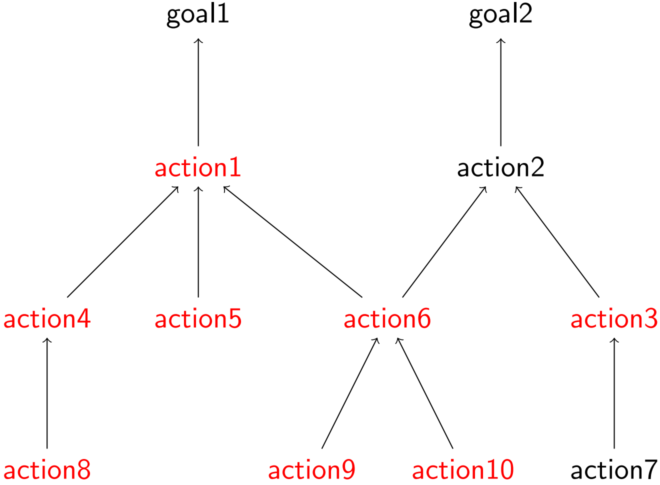
The variable selection scheme

Version 2: randomized action selection



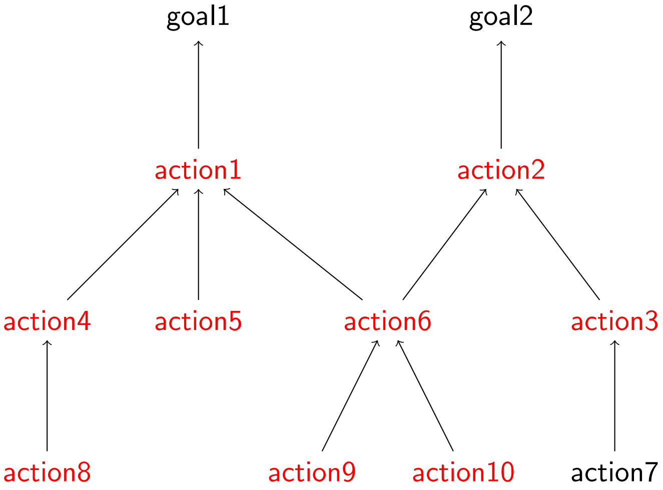
The variable selection scheme

Version 2: randomized action selection



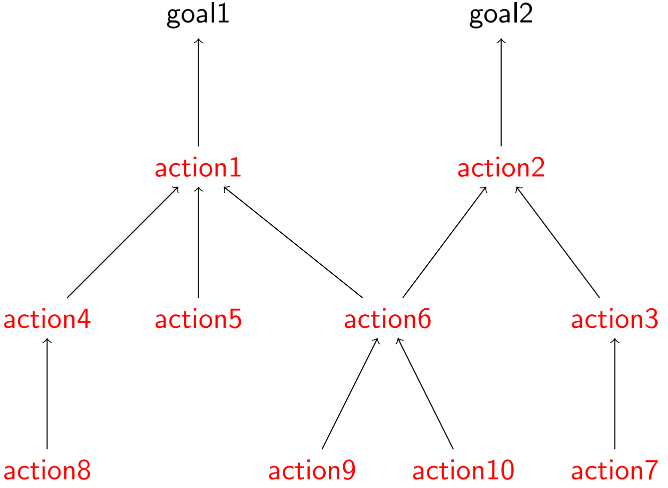
The variable selection scheme

Version 2: randomized action selection



The variable selection scheme

Version 2: randomized action selection



goal1

goal2

action1

action2

action4

action5

action6

action3

action8

action9

action10

action7

Significance

- The heuristic is more understandable than VSIDS etc.
- It is simpler than ones used with explicit state-space search.
- Lots of potential: simplicity, has not been researched much.
- Potentially wide applicability, as SAT very strong in other areas such as model-checking (CAV), DES diagnosis.

Conclusions

- We presented variable selection heuristics for planning within the CDCL framework.
- On average comparable with best planners that use state-space search; for many benchmark domains outperforms them.

Future work:

- Extend this with features from VSIDS to do still better.
- Try with Bounded LTL Model-Checking, Discrete Event Systems diagnosis, ...