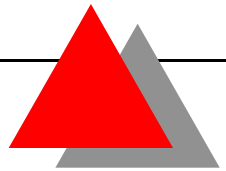
Square Placement Variants

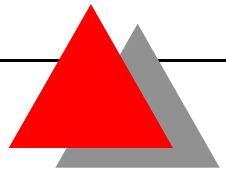
An example of search tree analysis

H. Simonis
COSYTEC SA



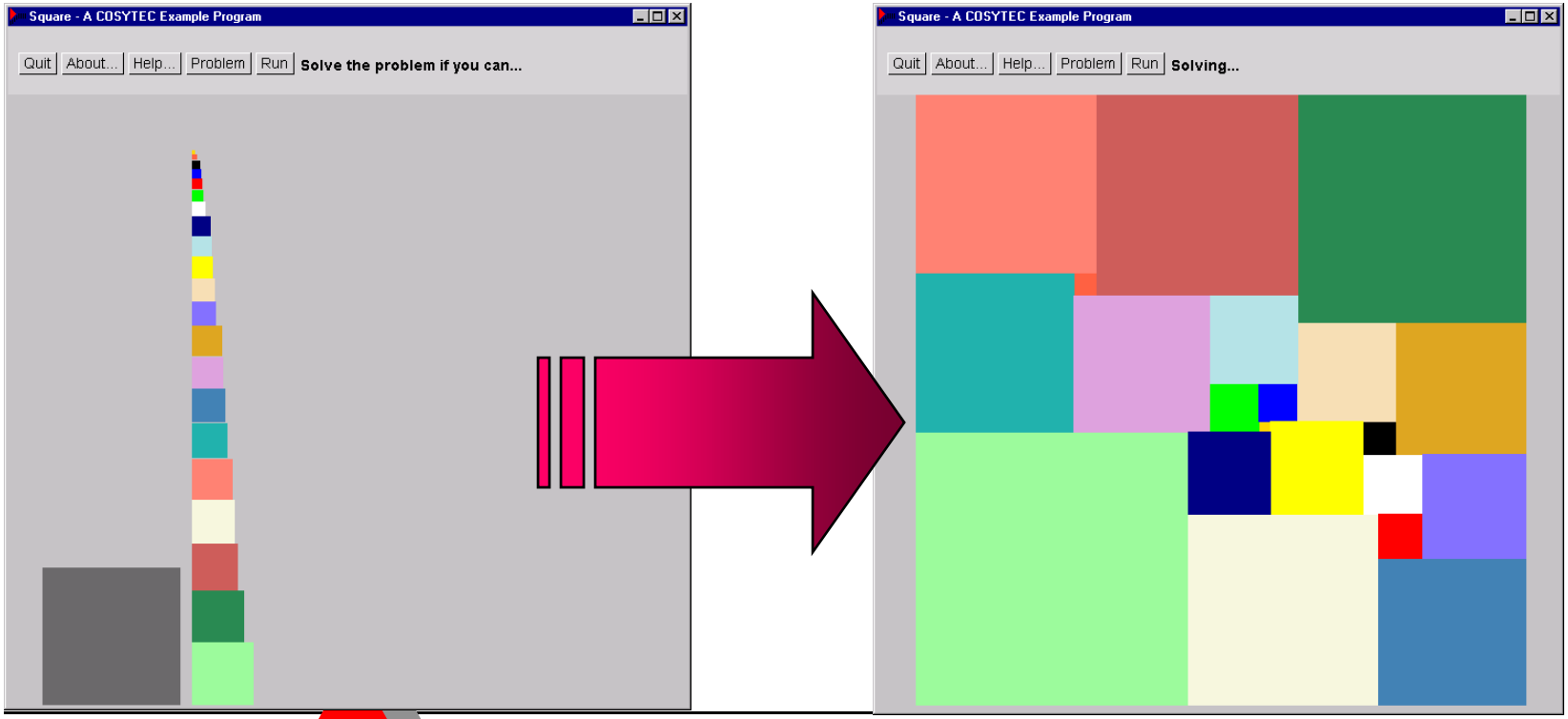
Overview

- **Problem**
- **Different Strategies**
- **Search tree views**
- **Search tree comparison**
- **Failure causes**



Problem

- Place 21 squares in square area
- Perfect placement problem



Program

```
run(X,Y):-
        data(Nr, Data, W, H),
        create_vars(Data,X,Y,Rectangles,W,H, Vars),
        cumulative(X,Data,Data,unused,unused,H,W,[H,0,1]),
        cumulative(Y,Data,Data,unused,unused,W,H,[W,0,1]),
        diffn(Rectangles,unused,unused,[W,H]),
        labeling(Rectangles).

data(21, [50, 42, 37, 35, 33, 29, 27, 25, 24, 19, 18, 17, 16, 15, 11, 9, 8, 7, 6, 4, 2], 112, 112).

create_vars([],[],[],[],_,_, []).
create_vars([L|L1],[X|X1],[Y|Y1],[[X,Y,L,L]|S1],W,H, [X, Y | XsYs]):-
        X :: 0..W,
        Y :: 0..H,
        create_vars(L1,X1,Y1,S1,W,H, XsYs).

labeling([]).
labeling([[X,Y,W,_]]):-
        indomain(X),
        indomain(Y).
labeling([[X,Y,W,_],[X1,Y1,W1,_]|R]):-
        indomain(X),
        indomain(Y),
        indomain(X1,max),
        indomain(Y1,max),
        labeling(R).
```

Different Strategies

- **Constraint options**
  - **diffn only**
  - **+cumulative limit**
    cumulative(X,Data,Data,unused,unused,H,unused,unused)
  - **+cumulative overall end**
    cumulative(X,Data,Data,unused,unused,H,W,unused)
  - **+ soft intermediate level**
    cumulative(X,Data,Data,unused,unused,H,W,[H,0])
  - **+ forced intermediate level**
    cumulative(X,Data,Data,unused,unused,H,W,[H,0,1])
  - **+ saturation for cumulative**
    - **change of cumulative**
  - **+re-execution of all methods at all times**
    - **change of cumulative**

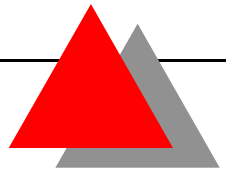
Different Strategies (2)

- **Search**
  - **indomain alternating min/max**
    - **n-ary choice**
  - **indomain1 alternating min/max**
    - **binary choice, select/remove value**

```
indomain(X):-
        dom(X,L),
        member(X,L).
```

```
indomain1(X):-
        domain_info(X,Min,..),
        indomain1(X,Min).

indomain1(X,X).
indomain1(X,Min):-
        X #\= Min,
        indomain1(X).
```

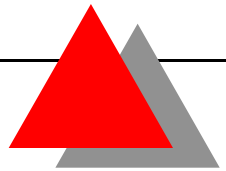


Different Strategies (3)

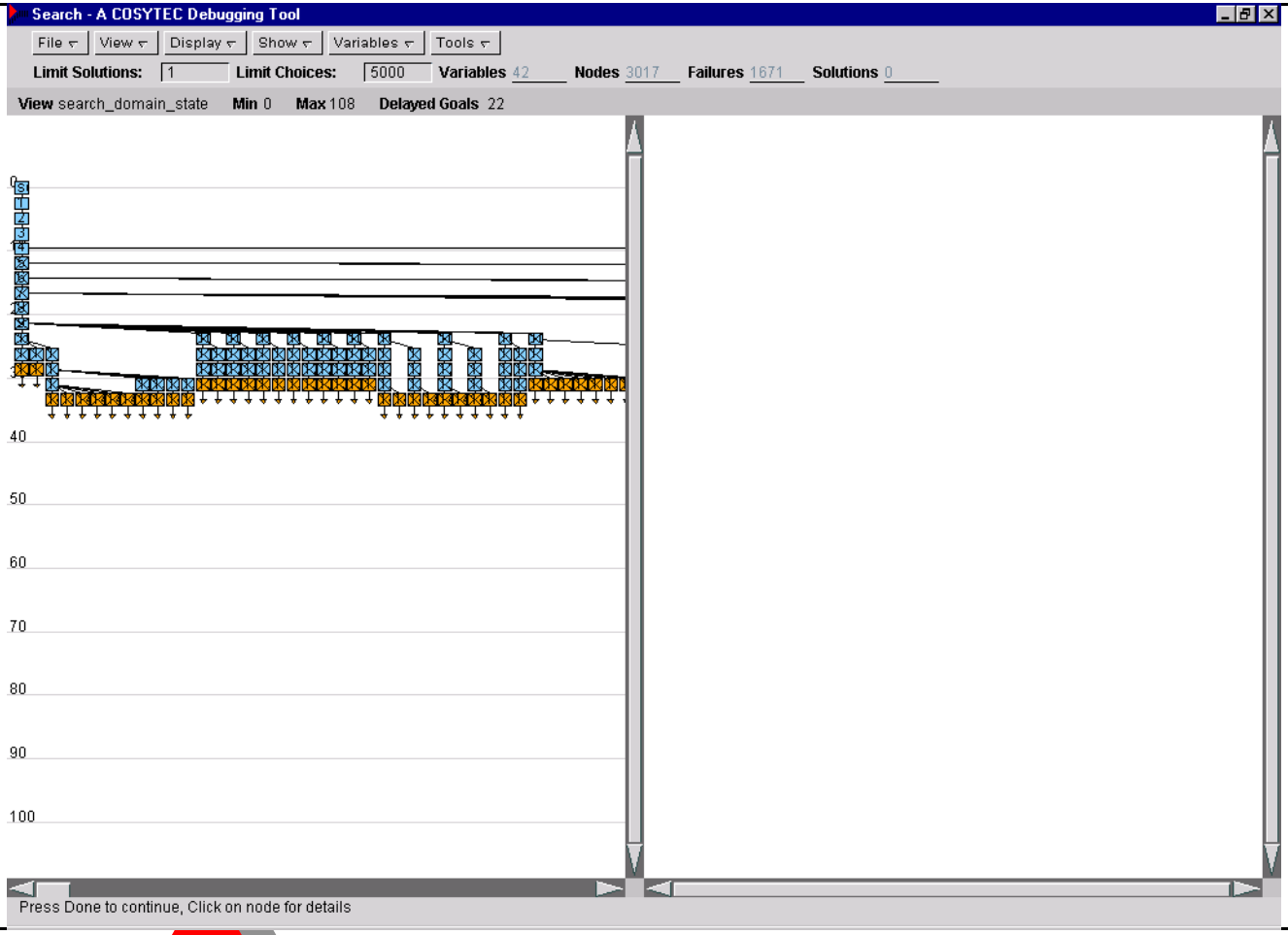
- **Shaving**
  - **test each variable to find which values are consistent with current state**
  - **enough: test next variable in static variable ordering**
  - **enough: shaving on min and max**
  - **pointless?**
    - **does not control variable selection**
    - **does not count as backtracking step**

Shaving for all values in domain

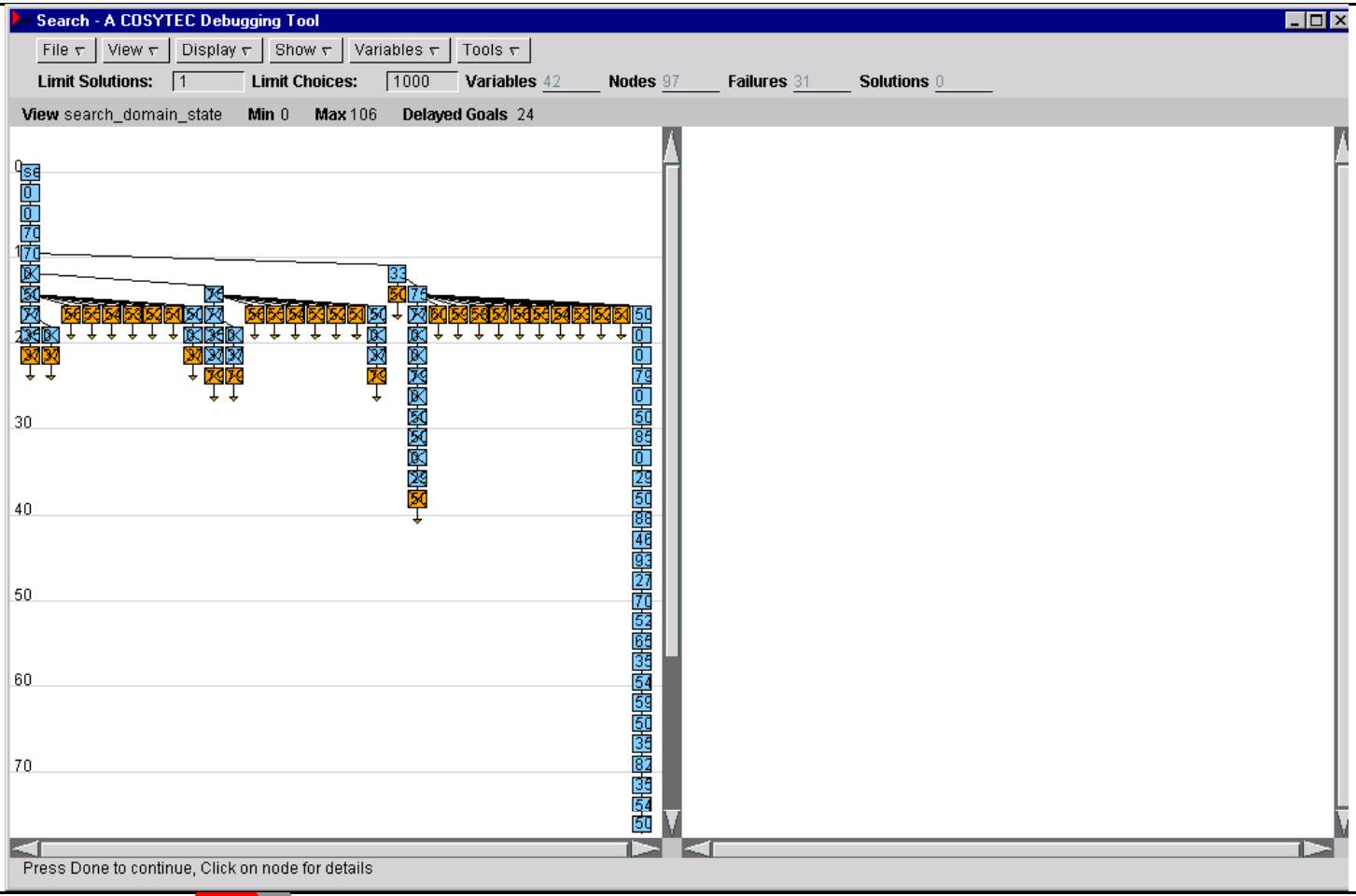
```
shaving(Variable,all):-
    dom(Variable,Values),
    findall(Variable,member(Variable,Values),[ValidValue|ValidValues]),
    NewVariable :: [ValidValue|ValidValues],
    Variable = NewVariable.
```



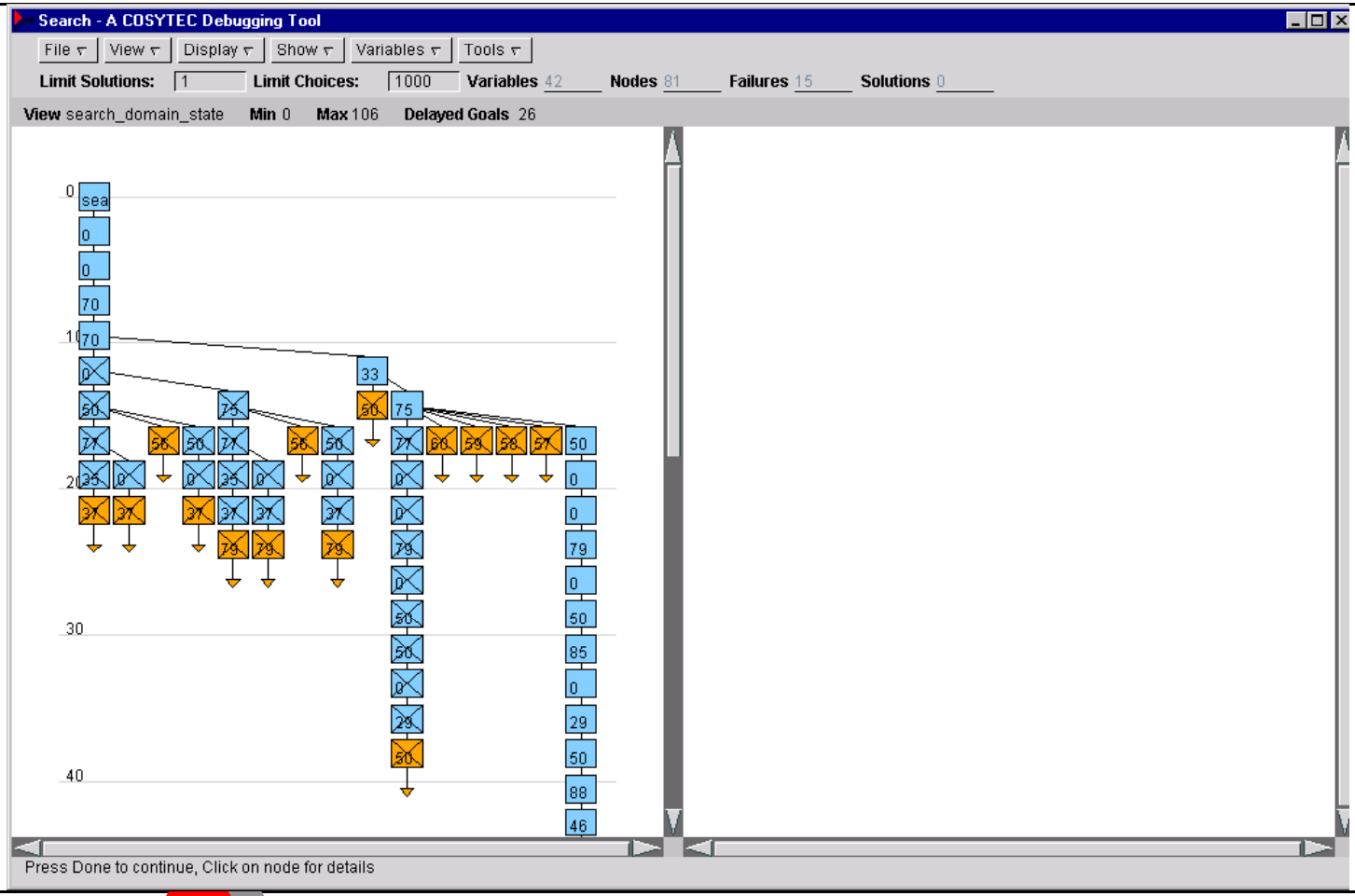
diffn only (initial part only)



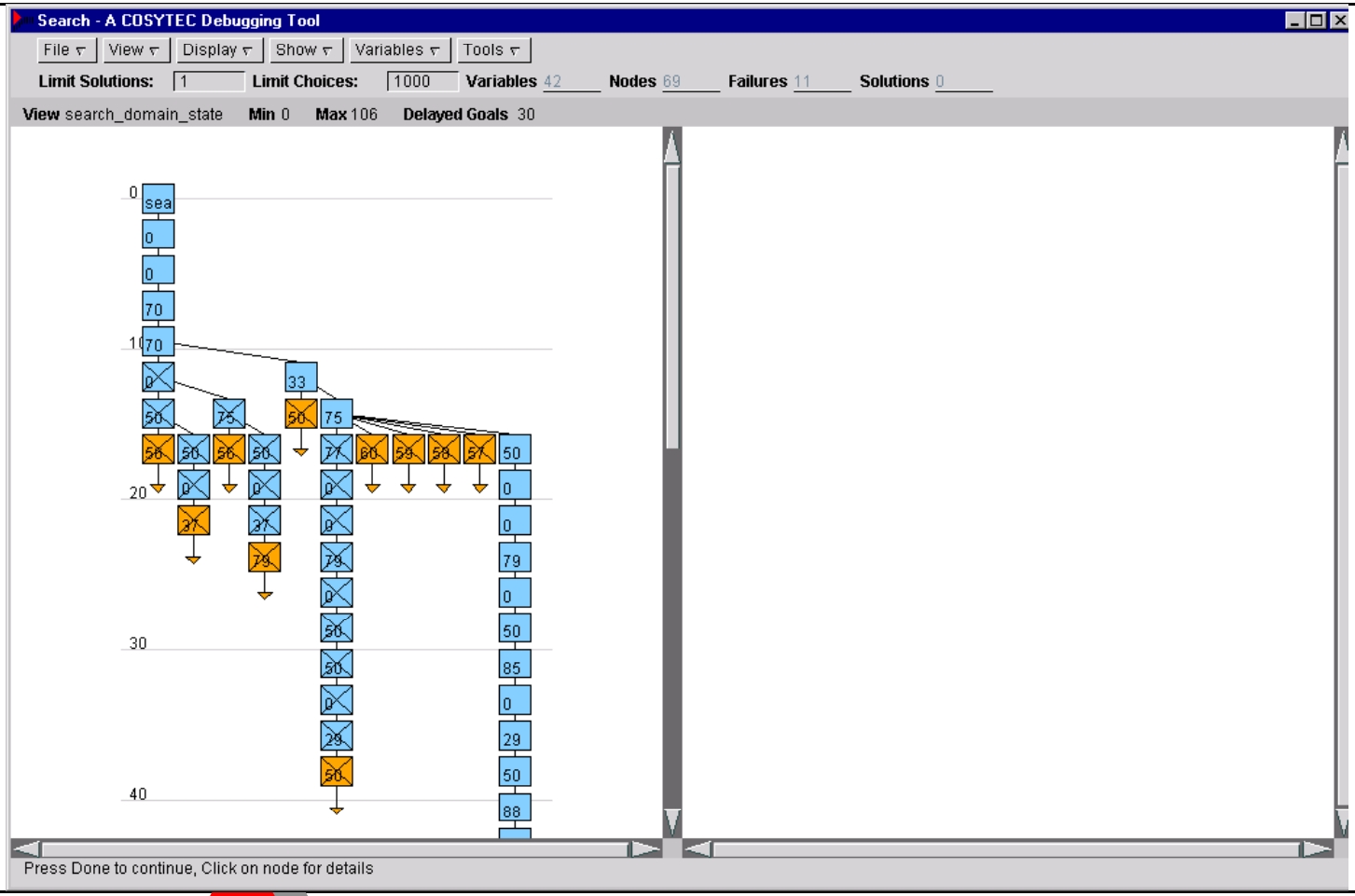
cumulative limit+end



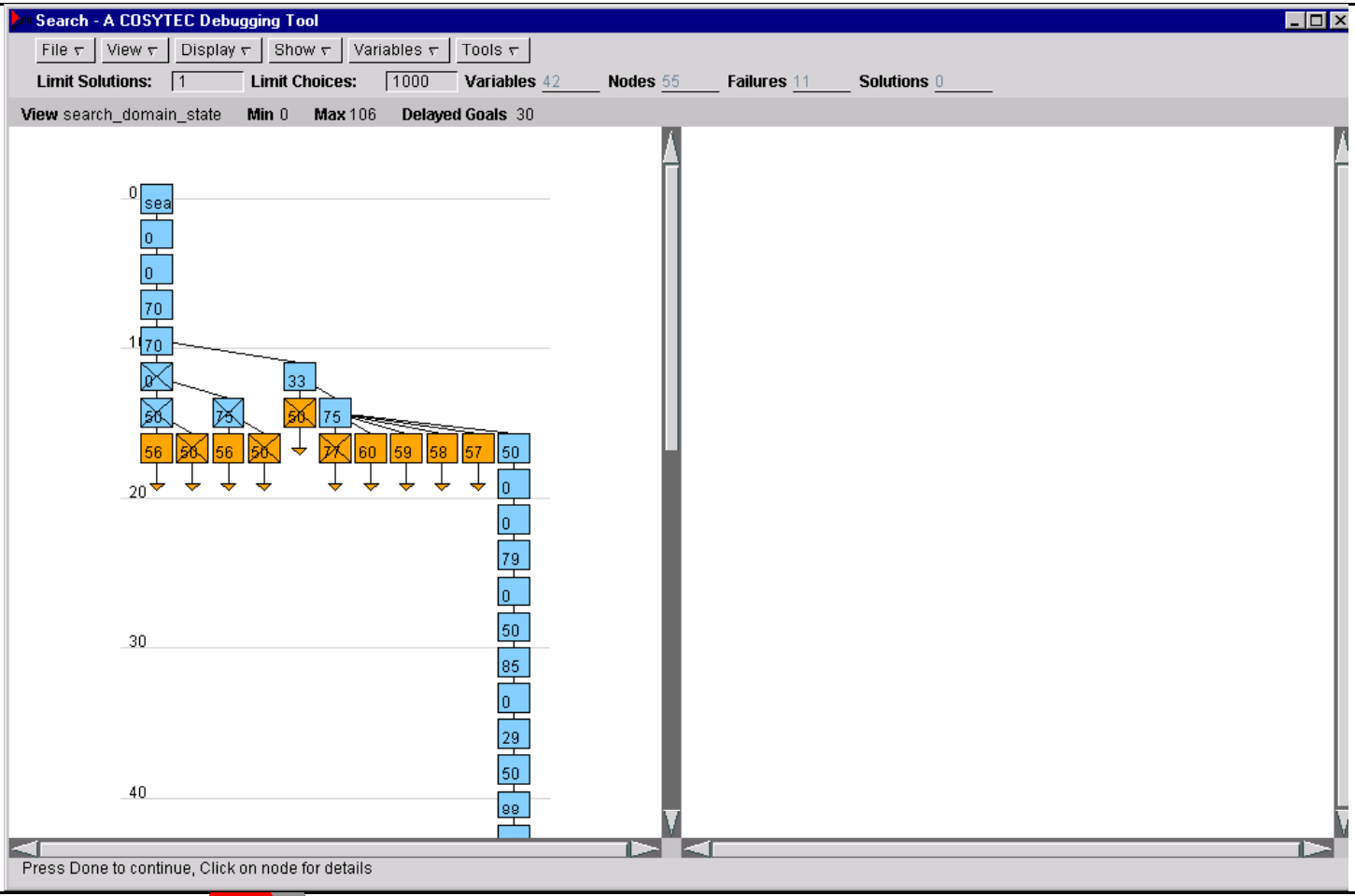
forced intermediate



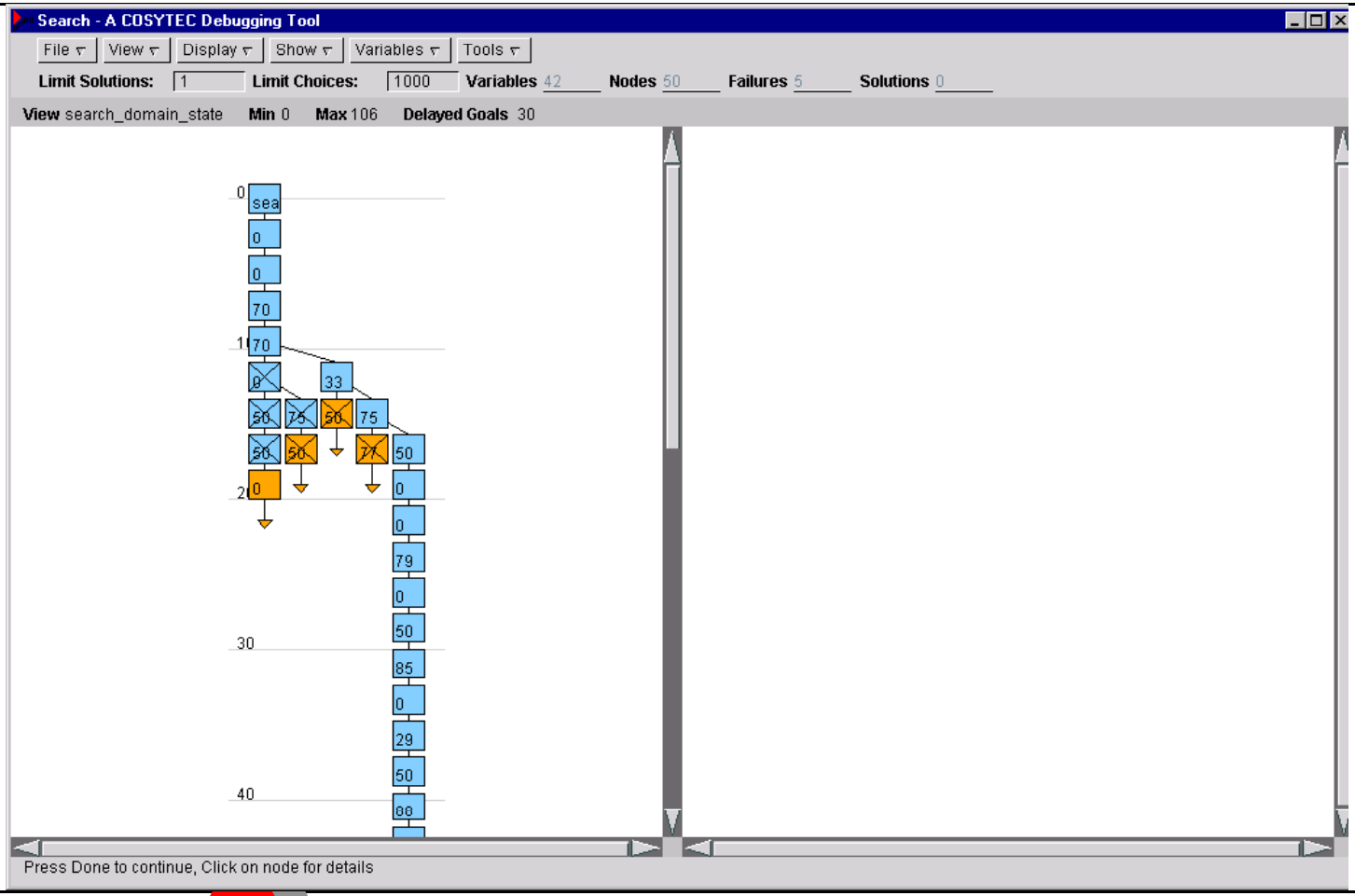
saturation cumulative



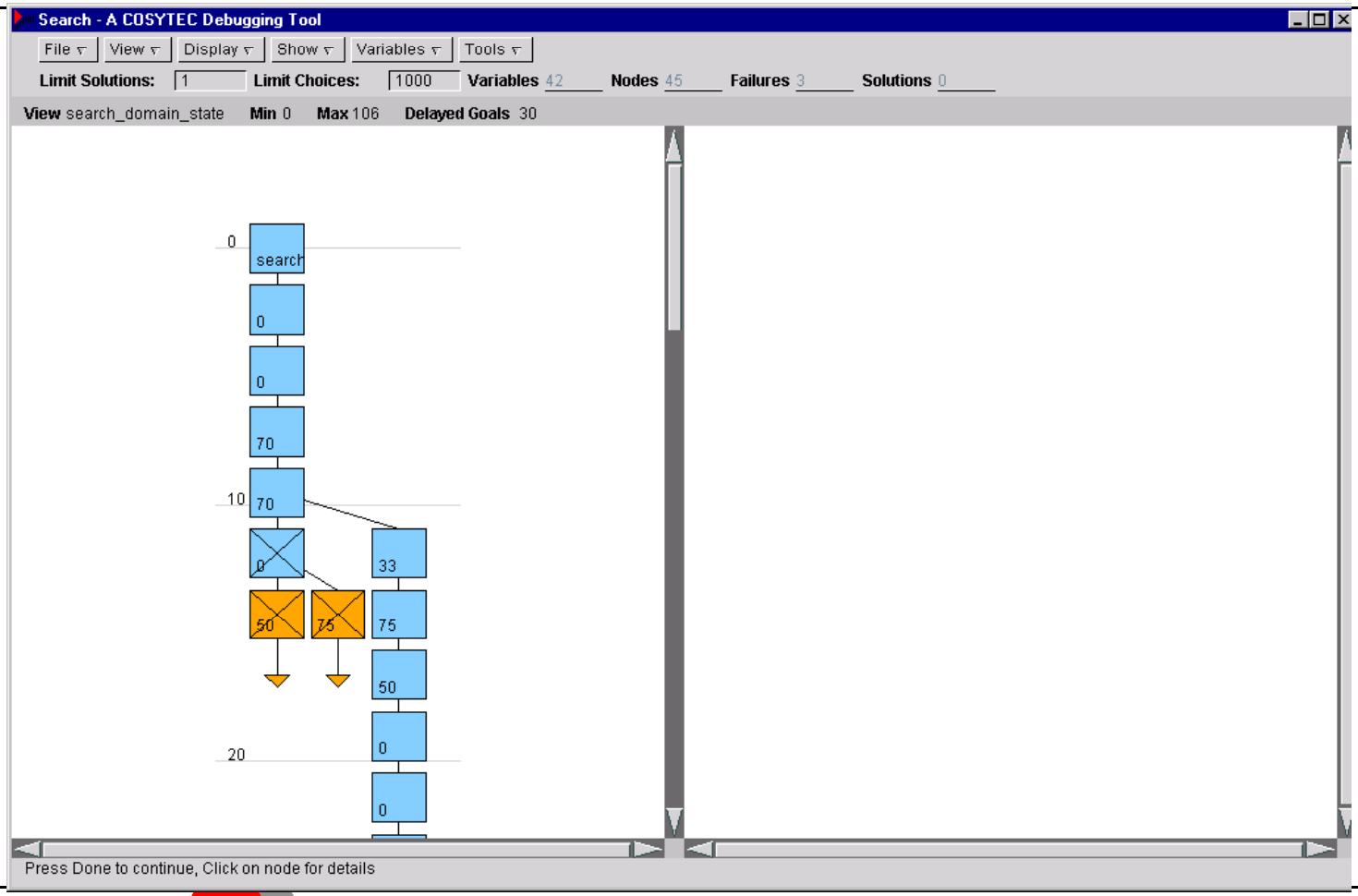
all methods, all times



indomain1


Search - A COSYTEC Debugging Tool
File
View
Display
Show
Variables
Tools
Limit Solutions: 1
Limit Choices: 1000
Variables 42
Nodes 50
Failures 5
Solutions 0
View search_domain_state
Min 0
Max 106
Delayed Goals 30
0
10
20
30
40
sea
Press Done to continue, Click on node for details

shaving


Search - A COSYTEC Debugging Tool
File
View
Display
Show
Variables
Tools
Limit Solutions: 1
Limit Choices: 1000
Variables 42
Nodes 45
Failures 3
Solutions 0
View search_domain_state
Min 0
Max 106
Delayed Goals 30
0
search
0
0
70
10
70
0
50
75
33
75
50
0
20
0
Press Done to continue, Click on node for details

Comparing two search methods

- **Often interested in comparison of two methods**
- **Difficult to achieve with two programs**
  - **two machines?**
- **Change search tree call**

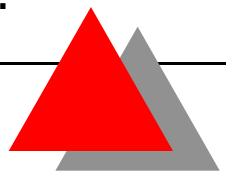
OLD:

```
search_start(Vars,labeling(Vars))
```

NEW:

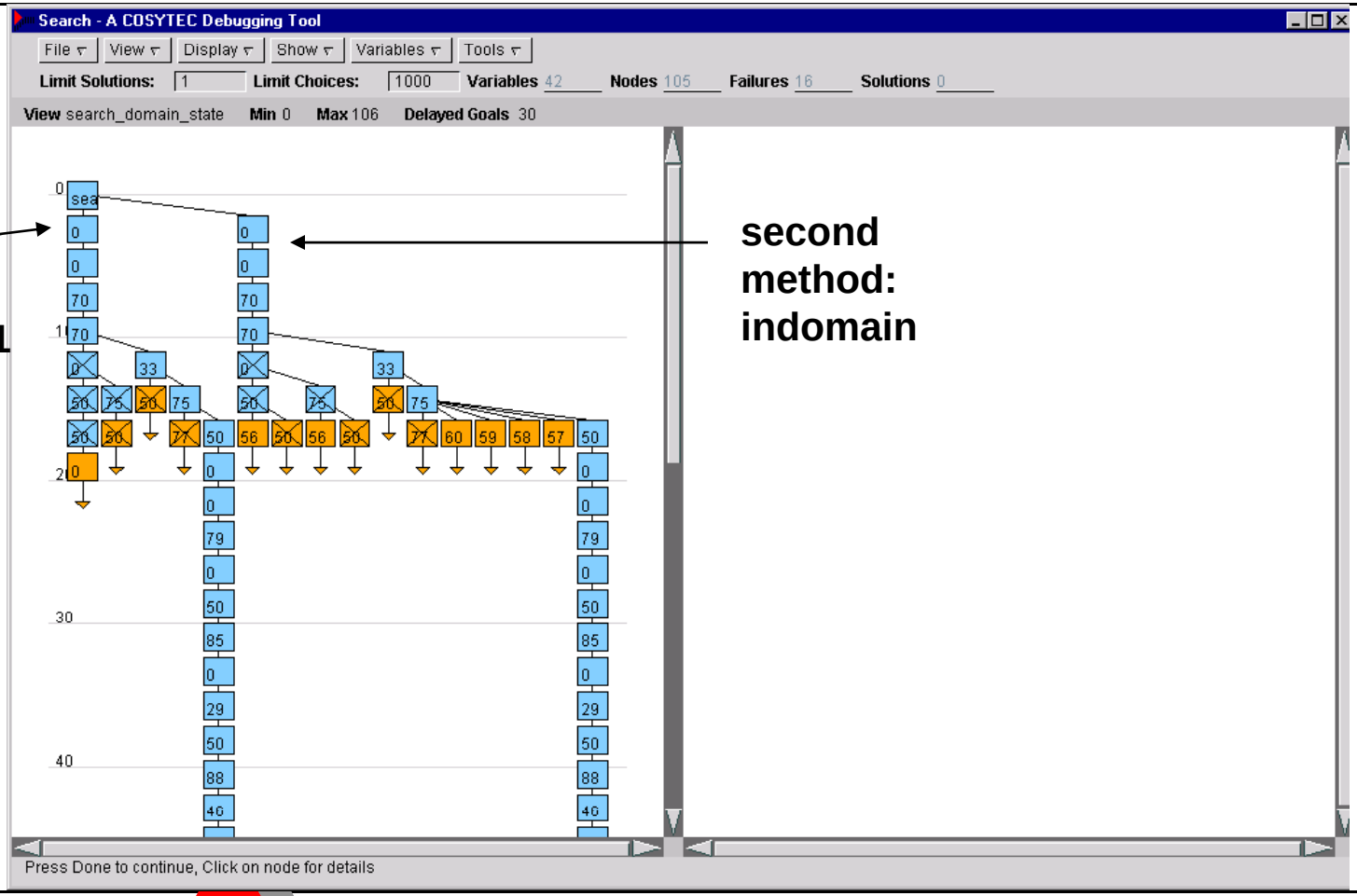
```
search_start(Vars,try(Vars))

try(Vars):-
        once(labeling1(Vars)),
        fail.
try(Vars):-
        once(labeling2(Vars)),
        fail.
```



Compare indomain1 - indomain

first method: indomain1

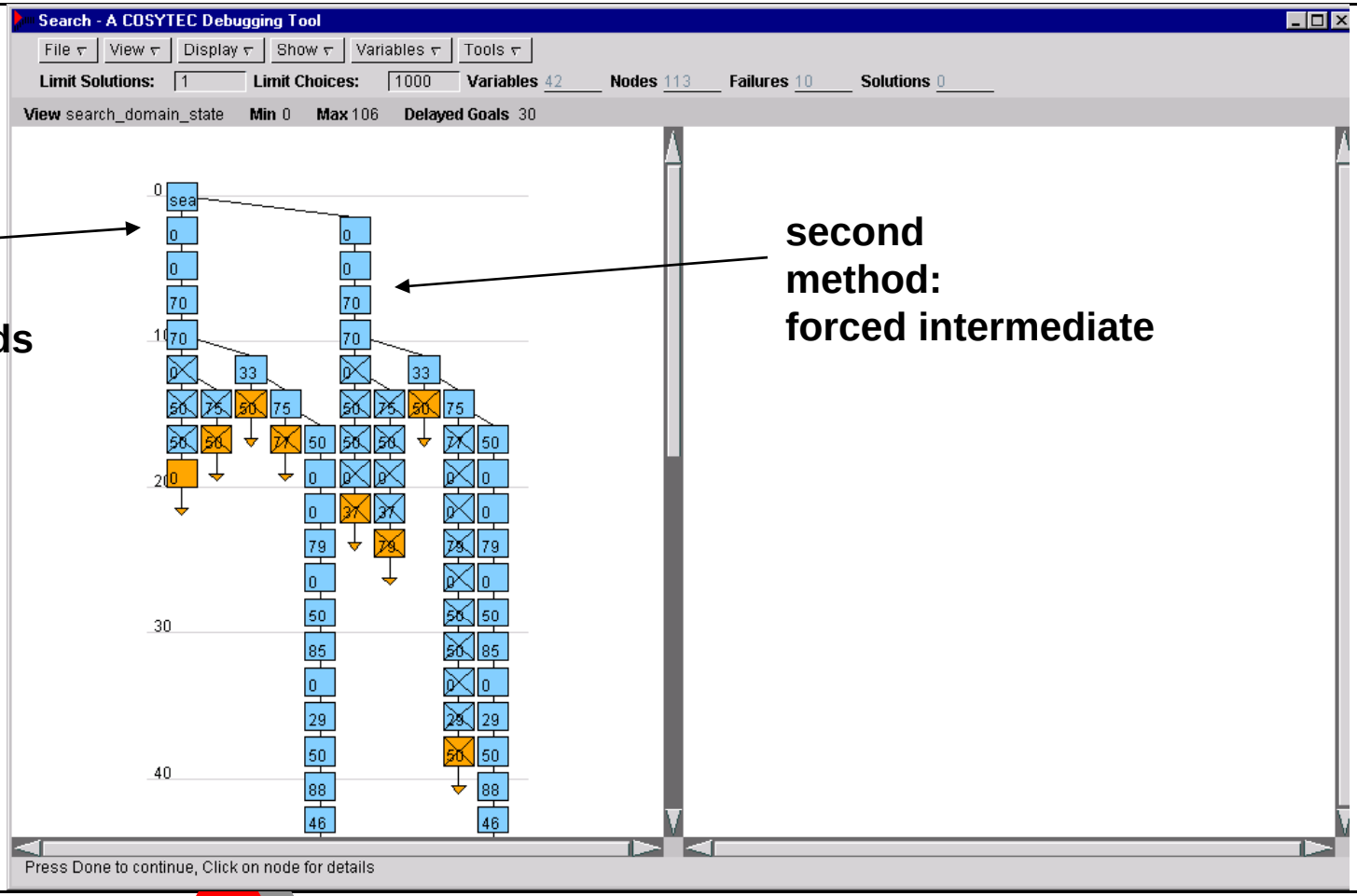


second method: indomain

Compare cumulative - all methods, all times; indomain1

first method: all methods

second method: forced intermediate



Backtracking count

Methods	Backtrack	Time	Backtrack	Time
Diffn only	2975	22653	2993 (?)	18757
Cumulative limit	2261	21521	2259	17836
Cumulative limit+end	55	1292	48	1392
Cumulative intermediate	55	1272	48	1342
Forced intermediate	39	1402	35	1392
Saturation cumul	27	1072	21	961
All methods, all times	7	5197	7	5799
Shaving	3	6709	3	6911

Failure Events

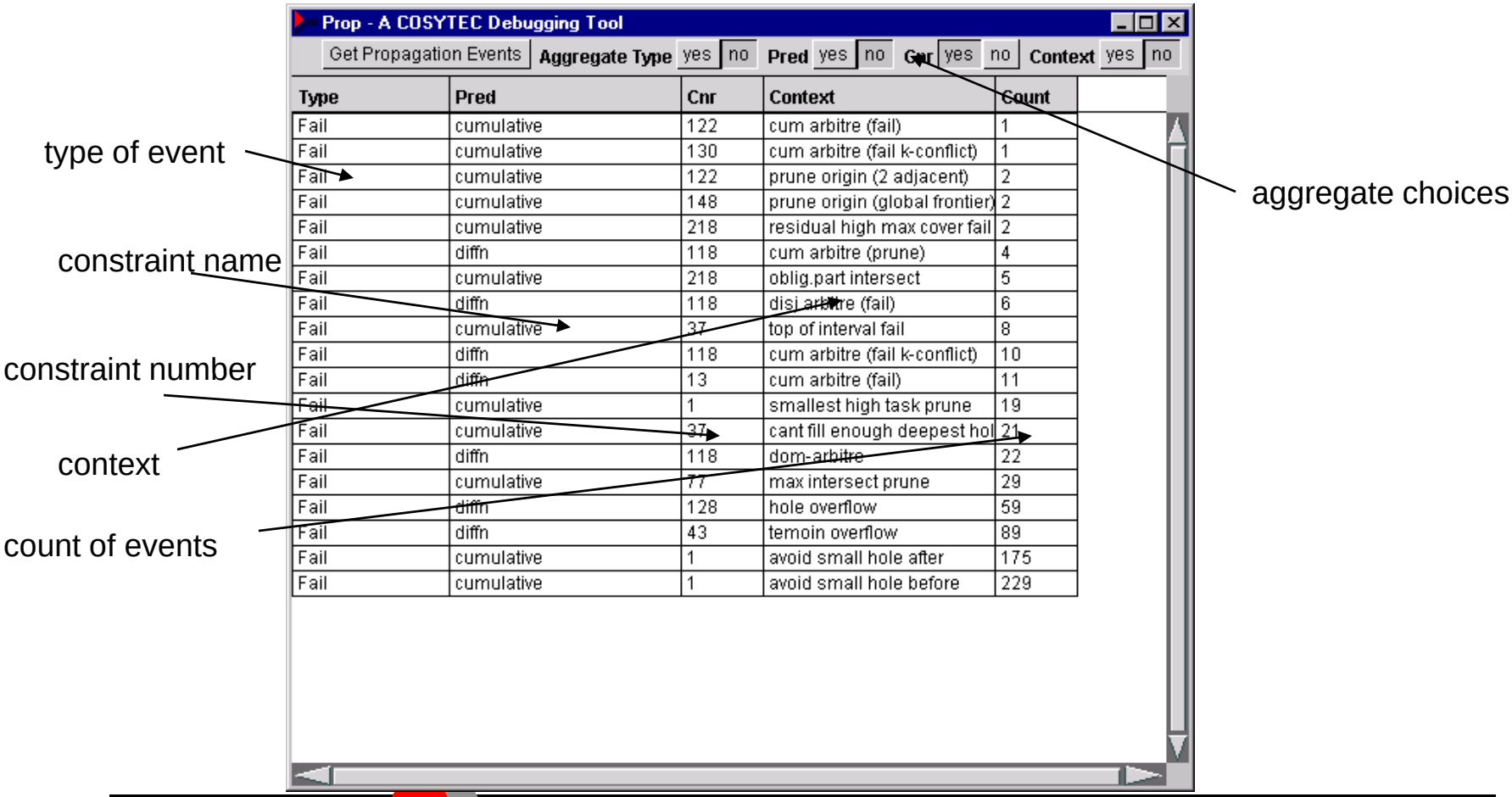
Constraint	Context	All methods	Cumul saturation	Force intermediate	Cumul limit+end	Diff only
cumulative	avoid small hole before	295	108	104	104	0
cumulative	avoid small hole after	176	104	102	102	0
diffn	temoin overflow	54	104	183	191	29489
diffn	hole overflow	31	95	117	280	12133
cumulative	max intersect prune	30	35	32	32	0
cumulative	smallest high task prune	22	4	2	0	0
diffn	dom-arbitre	17	42	51	59	4871
cumulative	cant fill enough deepest hole	17	22	21	0	0
diffn	cum arbitre (fail k-conflict)	9	7	17	17	180
cumulative	oblig.part intersect	6	1	0	0	0
diffn	disj arbitre (fail)	5	15	12	13	249
diffn	cum arbitre (prune)	4	4	0	0	0
diffn	cum arbitre (fail)	2	6	12	12	182
cumulative	top of interval fail	2	5	5	4	0
cumulative	prune origin (global frontier)	2	2	2	2	0
cumulative	residual high max cover fail	1	1	2	2	0

Failure Events (2)

cumulative	Cant reach intermediate level	0	7	7	1	0
cumulative	Cum arbitre (fail)	0	1	1	0	0
Diffn	overlapping	0	0	8	9	243
diffn	Disj arbitre (before-after)	0	0	1	1	0
Cumulative	First interval fail	0	0	1	0	0
diffn	Eval earliest end	0	0	0	0	359
diffn	Min ori prune	0	0	0	0	217
diffn	Hole include big size	0	0	0	0	7
diffn	Disj arbitre fail (marge)	0	0	0	0	6
diffn	Max ori prune	0	0	0	0	3
diffn	Hole overflow2	0	0	0	0	2
diffn	Disj partititon	0	0	0	0	2
total	-	673	563	680	829	47943

not shown for shaving, as shaving failures can not be separated

Newprop tool



Conclusion

- Example of problem analysis with search tree tool
- Comparison of strategies inside tool
- Backtracking count vs failure count
- Failure events
  - variety of methods
  - change of methods with small change of programs
- Newprop tool
  - listing of all propagation events
  - aggregation
  - internal use only
  - for tool developer
- Significant improvement of program by small modification