Multi-Criteria Optimization in ASP and its Application to Linux Package Configuration

Martin Gebser Roland Kaminski Benjamin Kaufmann
Torsten Schaub

Institut für Informatik, Universität Potsdam

Outline

Outline

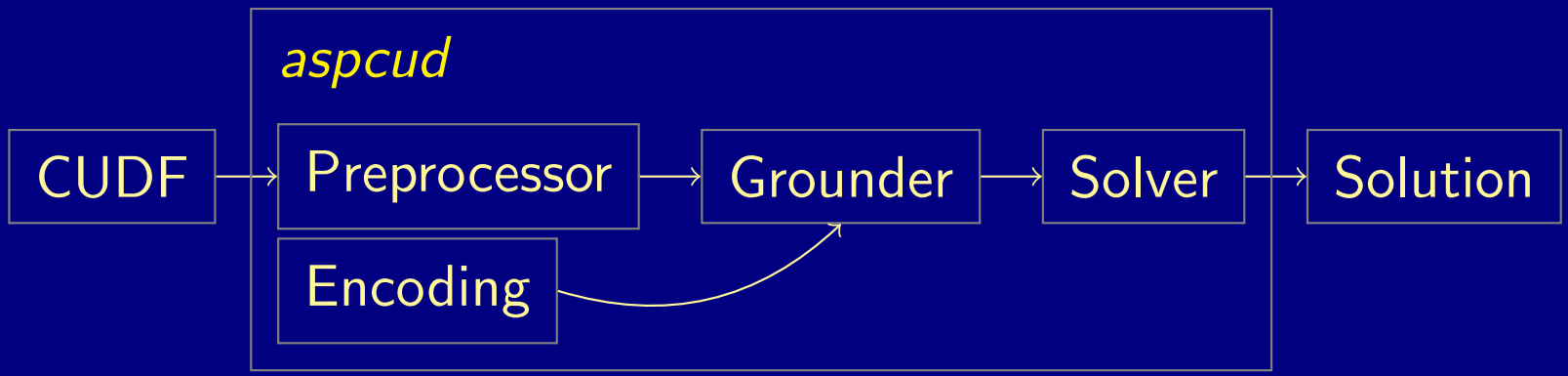
Motivation

- Maintaining packages in modern Linux distributions is difficult
  - Complex dependencies
  - Large package repositories
  - Ever changing in view of software development
- Challenges for package configuration tools
  - Large problem size
  - Soft (and hard) constraints
  - Multiple optimization criteria
- Contributions of this work
  - Package configuration via Answer Set Programming (ASP)
  - Uniform modeling by encoding plus instances
  - Solving techniques for multi-criteria optimization

Outline

Overview

aspcud Tool for solving package configuration problems



Preprocessor Converts CUDF input to ASP instance

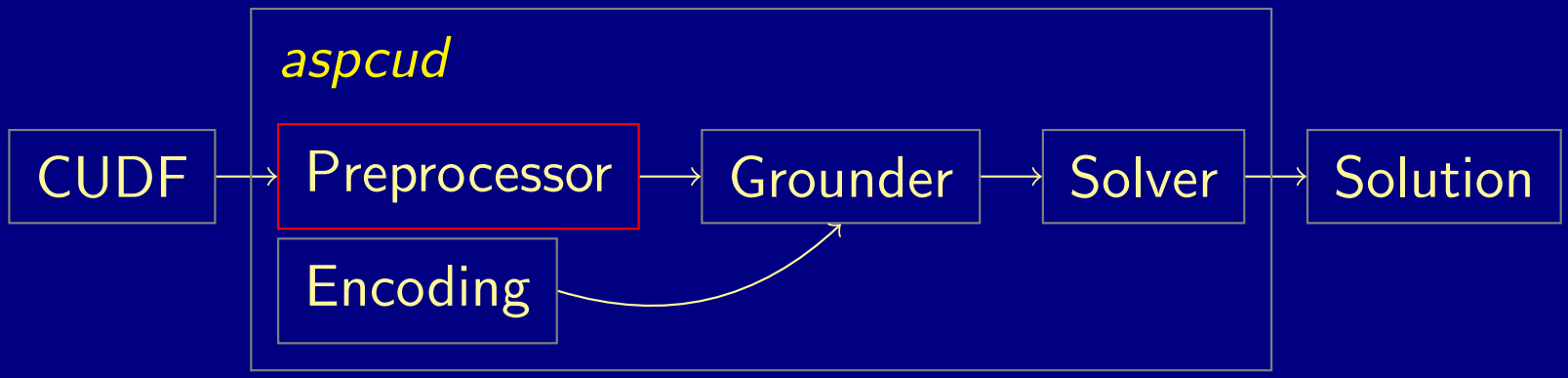
Encoding First-order problem specification

Grounder Instantiates first-order variables

Solver Searches for (optimal) answer sets

Overview

aspcud Tool for solving package configuration problems



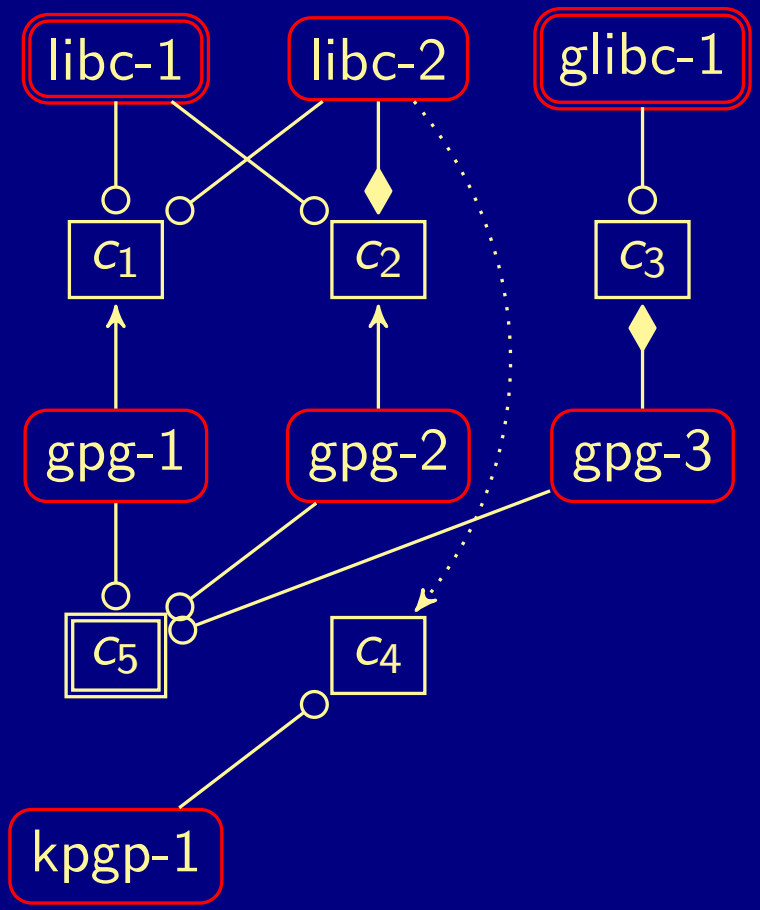
Preprocessor Converts CUDF input to ASP instance

Encoding First-order problem specification

Grounder Instantiates first-order variables

Solver Searches for (optimal) answer sets

Instance Format



Installable Packages:

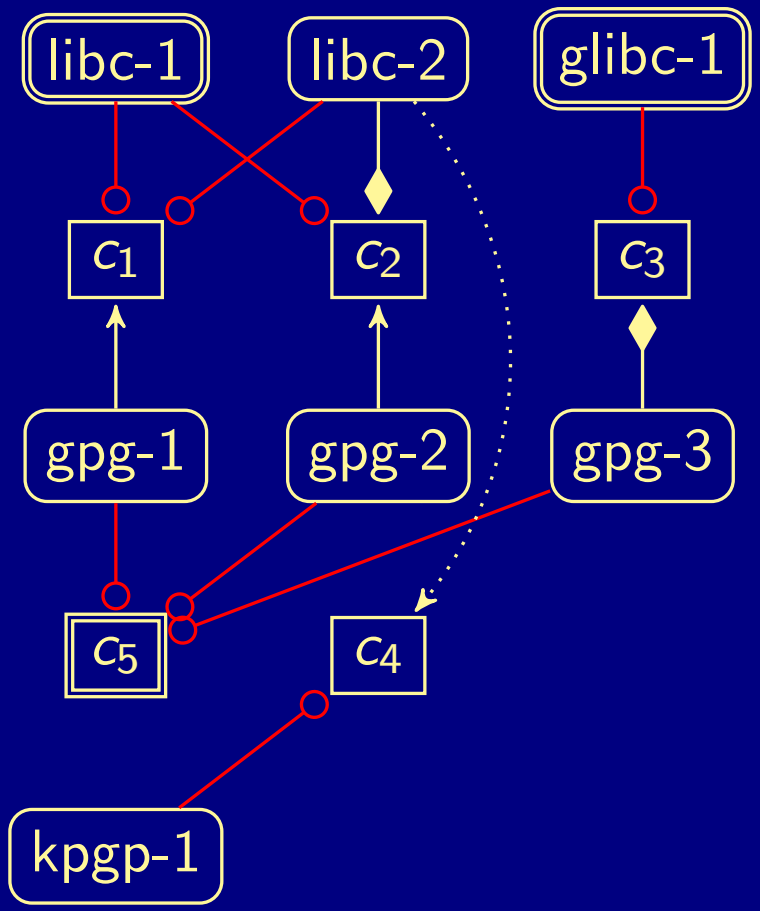
```
package(libc,1).
package(libc,2).

package(glibc,1).

package(gpg,1).
package(gpg,2).
package(gpg,3).

package(kpgp,1).
```

Instance Format



Clauses:

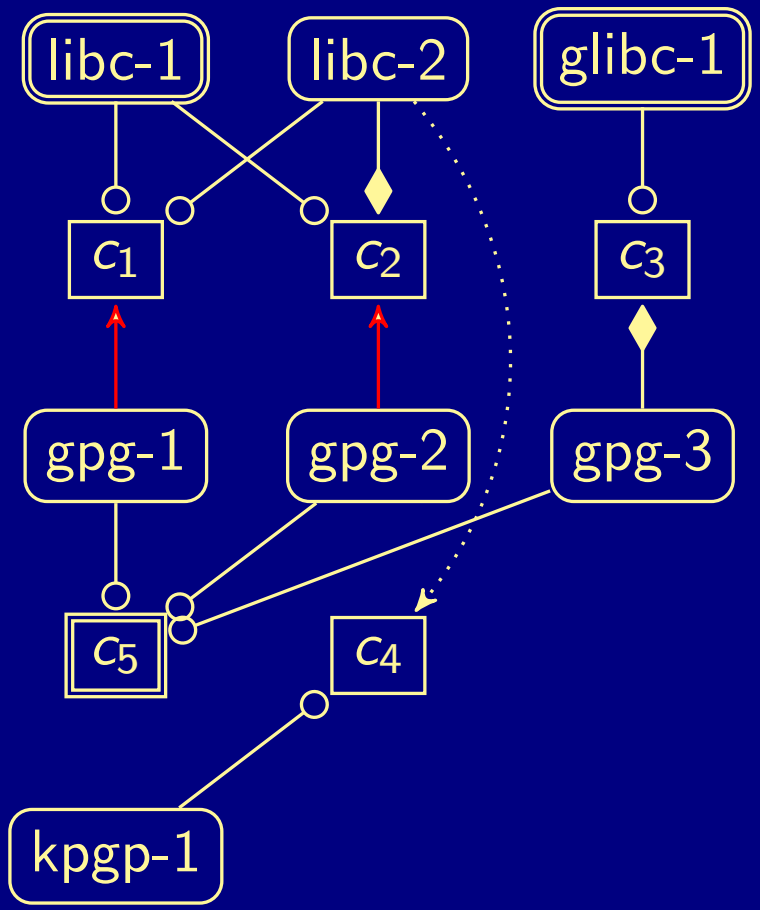
```
satisfies(libc,1,c1).
satisfies(libc,1,c2).
satisfies(libc,2,c1).

satisfies(glibc,1,c3).

satisfies(gpg,1,c5).
satisfies(gpg,2,c5).
satisfies(gpg,3,c5).

satisfies(kpgp,1,c4).
```

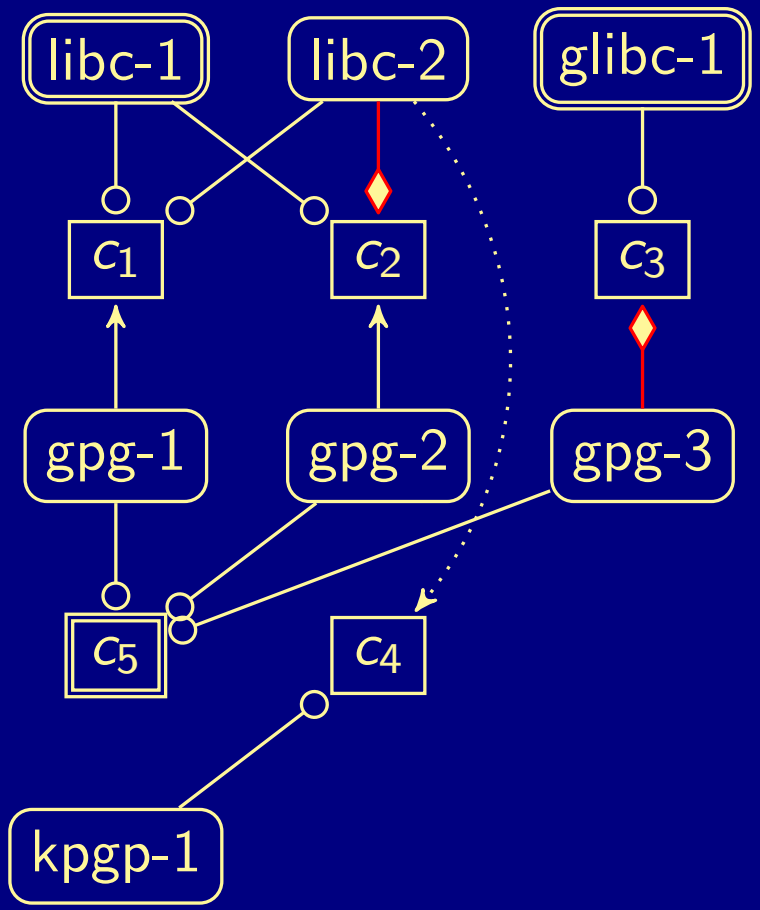
Instance Format



Package Dependencies:

```
depends(gpg,1,c1).
depends(gpg,2,c2).
```

Instance Format

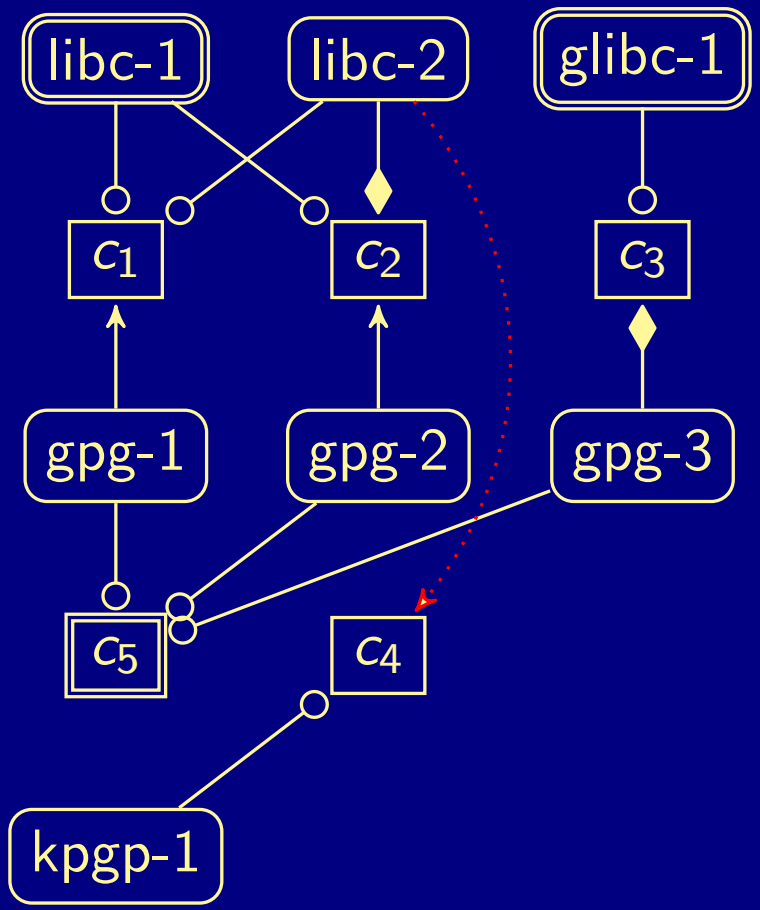


Package Conflicts:

```
conflicts(libc,2,c2).

conflicts(gpg,3,c3).
```

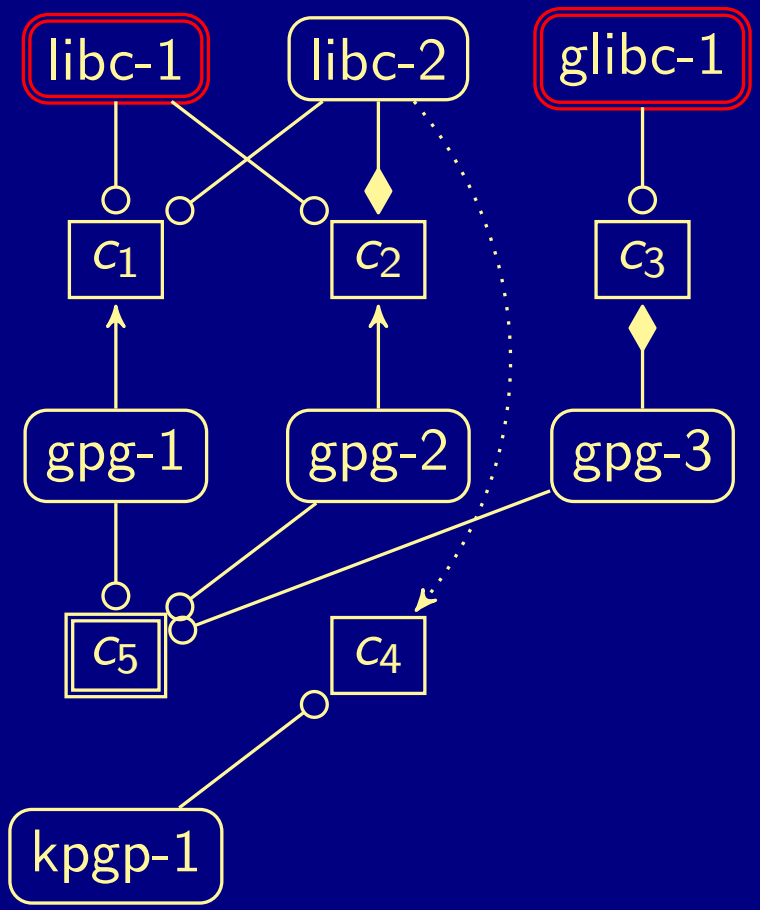
Instance Format



Package Recommendations:

```
recommends(libc,2,c4).
```

Instance Format

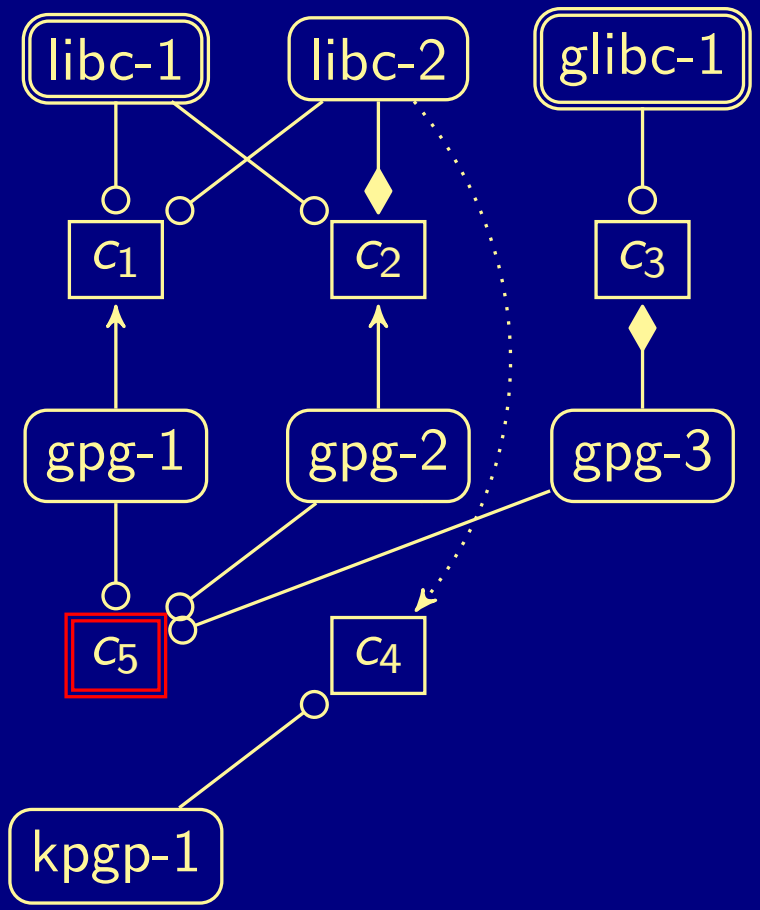


Installed Packages:

```
installed(libc,1).

installed(glibc,1).
```

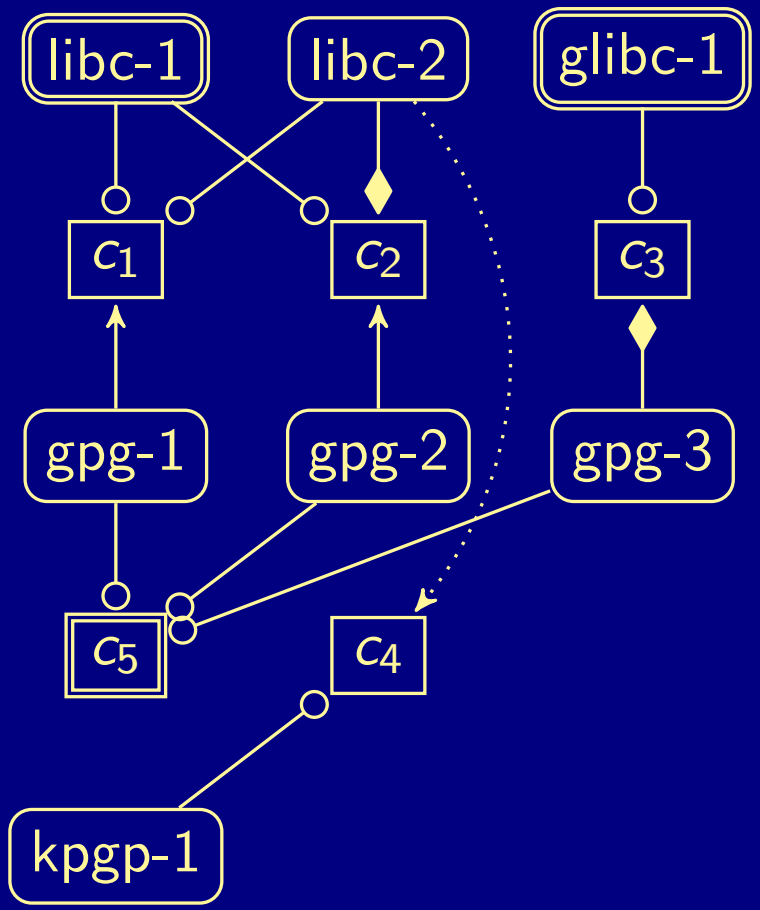
Instance Format



Requests:

```
requested(c5).
```

Instance Format

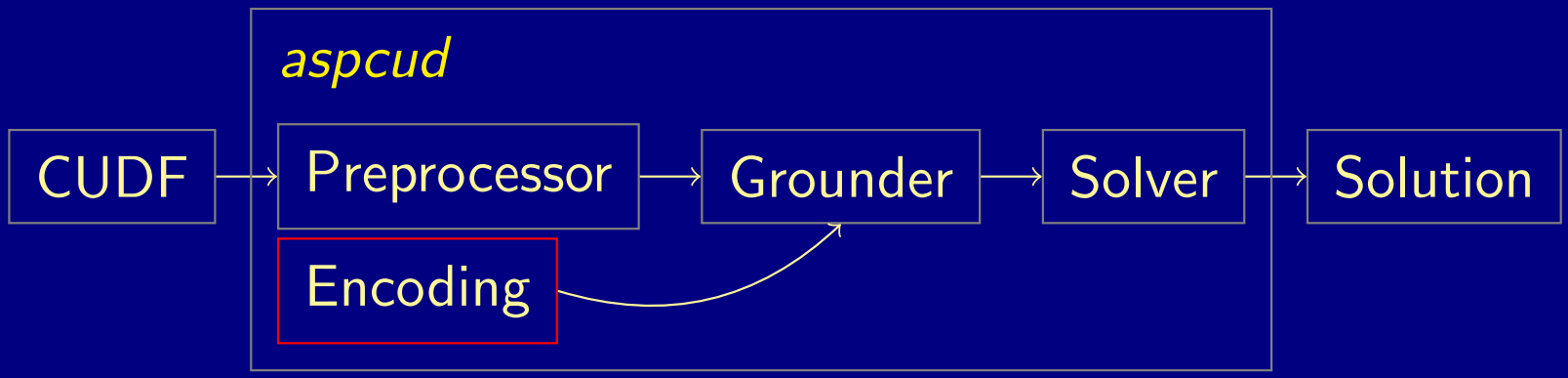


Optimization Criteria:

```
utility(delete,-1).
utility(change,-2).
```

Overview

aspcud Tool for solving package configuration problems



Preprocessor Converts CUDF input to ASP instance

Encoding First-order problem specification

Grounder Instantiates first-order variables

Solver Searches for (optimal) answer sets

Hard Constraints

```
% choose packages to install
{ install(N,V) } :- package(N,V).

% derive required clauses
exclude(C) :- install(N,V), conflicts(N,V,C).
include(C) :- install(N,V), depends(N,V,C).
% derive satisfied clauses
satisfy(C) :- install(N,V), satisfies(N,V,C).

% assert required clauses to be (un)satisfied
 :- exclude(C),     satisfy(C).
 :- include(C), not satisfy(C).
 :- request(C), not satisfy(C).
```

Hard Constraints

```
% choose packages to install
{ install(N,V) } :- package(N,V).

% derive required clauses
exclude(C) :- install(N,V), conflicts(N,V,C).
include(C) :- install(N,V), depends(N,V,C).
% derive satisfied clauses
satisfy(C) :- install(N,V), satisfies(N,V,C).

% assert required clauses to be (un)satisfied
 :- exclude(C),     satisfy(C).
 :- include(C), not satisfy(C).
 :- request(C), not satisfy(C).
```

Hard Constraints

```
% choose packages to install
{ install(N,V) } :- package(N,V).

% derive required clauses
exclude(C) :- install(N,V), conflicts(N,V,C).
include(C) :- install(N,V), depends(N,V,C).
% derive satisfied clauses
satisfy(C) :- install(N,V), satisfies(N,V,C).

% assert required clauses to be (un)satisfied
 :- exclude(C),     satisfy(C).
 :- include(C), not satisfy(C).
 :- request(C), not satisfy(C).
```

Soft Constraints

```
% auxiliary definitions
install(N)   :- install(N,V).
installed(N) :- installed(N,V).

% derive optimization criteria violations
violate(newpkg,N) :-
    utility(newpkg,L), install(N), not installed(N).
violate(delete,N) :-
    utility(delete,L), installed(N), not install(N).
% similar for other criteria
...

% impose soft constraints
#minimize[ violate(U,T) = 1 @ -L : utility(U,L) : L < 0 ].
#maximize[ violate(U,T) = 1 @  L : utility(U,L) : L > 0 ].
```

Soft Constraints

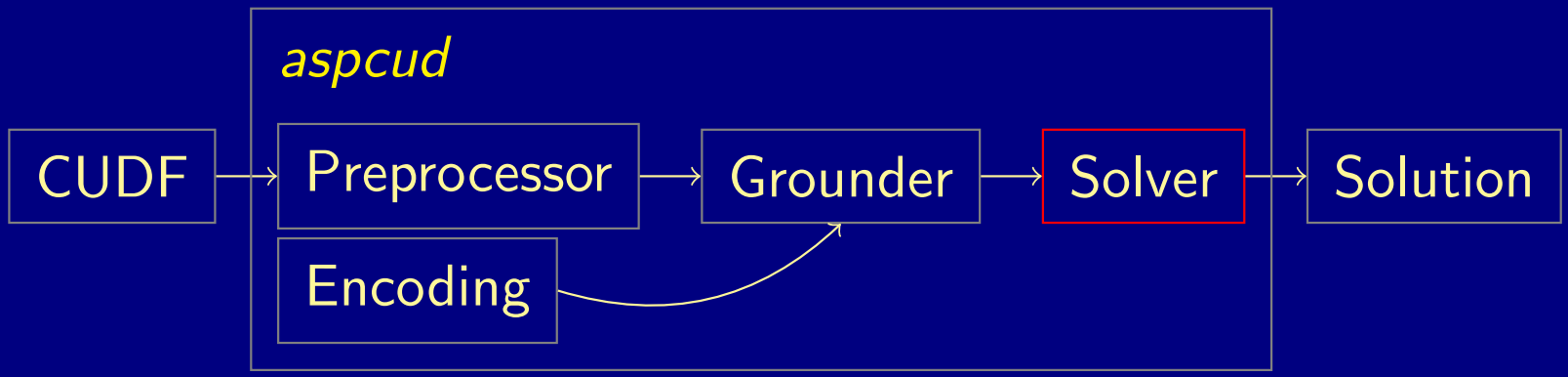
```
% auxiliary definitions
install(N)   :- install(N,V).
installed(N) :- installed(N,V).

% derive optimization criteria violations
violate(newpkg,N) :-
    utility(newpkg,L), install(N), not installed(N).
violate(delete,N) :-
    utility(delete,L), installed(N), not install(N).
% similar for other criteria
...

% impose soft constraints
#minimize[ violate(U,T) = 1 @ -L : utility(U,L) : L < 0 ].
#maximize[ violate(U,T) = 1 @  L : utility(U,L) : L > 0 ].
```

Outline

Optimization Algorithm



- Package configuration problems are often under-constrained
- Lexicographical optimization algorithm enumerates too much

Alternative Approach

- Optimize criteria in the order of significance
- Decrease upper bounds (costs) w.r.t. witnesses
- Proceed to next criterion upon unsatisfiability

Design Goals

- Incorporate into conflict-driven solving
- Keep as much learned information as possible
- Build upon standard features like assumptions

```
1  Model ← ⊥
2  foreach Criterion do
3      Lower ← 0
4      Upper ← eval(Criterion, Model)
5      while Lower < Upper do
6          add((Criterion ∪ ⟨∼Aux = −∞⟩) < Upper)
7          M ← solve({Aux})
8          if M ≠ ⊥ then
9              Model ← M
10             Upper ← eval(Criterion, Model)
11             simplify({Aux})
12         else
13             if Model = ⊥ then return ⊥
14             Lower ← Upper
15             simplify({∼Aux})
16 return M
```

Outline

Setup

- Benchmarks
  - 117 instances from the 3rd MISC-live run
  - Optimization criteria
    - paranoid, trendy
    - user1 (-notuptodate, -removed, -changed)
    - user2 (-changed, -removed, -unsat_recommends, -new)
    - user3 (-changed, -notuptodate, -removed, -new)
- Optimization algorithms
  - $clasp_0$: lexicographical optimization
  - $clasp_1$: hierarchical optimization
  - $clasp_2$: hierarchical optimization with exponential steps
- Optimization heuristics
  - $clasp_i^0$: no optimization-specific heuristic
  - $clasp_i^1$: falsify literals to minimize upon branching
  - $clasp_i^2$: falsify literals to minimize until conflict
  - $clasp_i^3$: combines $clasp_i^1$ and $clasp_i^2$
- Search restarts
  - $clasp_i^j$-r: perform restart after each model (mandatory with $clasp_i^2$ and $clasp_i^3$)
- Scoring like in MISC-live run

Setup

- Benchmarks
  - 117 instances from the 3rd MISC-live run
  - Optimization criteria
    - paranoid, trendy
    - user1 (-notuptodate, -removed, -changed)
    - user2 (-changed, -removed, -unsat_recommends, -new)
    - user3 (-changed, -notuptodate, -removed, -new)
- Optimization algorithms
  - $clasp_0$: lexicographical optimization
  - $clasp_1$: hierarchical optimization
  - $clasp_2$: hierarchical optimization with exponential steps
- Optimization heuristics
  - $clasp_i^0$: no optimization-specific heuristic
  - $clasp_i^1$: falsify literals to minimize upon branching
  - $clasp_i^2$: falsify literals to minimize until conflict
  - $clasp_i^3$: combines $clasp_i^1$ and $clasp_i^2$
- Search restarts
  - $clasp_i^j$-r: perform restart after each model (mandatory with $clasp_i^2$ and $clasp_i^3$)
- Scoring like in MISC-live run

Setup

- Benchmarks
  - 117 instances from the 3rd MISC-live run
  - Optimization criteria
    - paranoid, trendy
    - user1 (-notuptodate, -removed, -changed)
    - user2 (-changed, -removed, -unsat_recommends, -new)
    - user3 (-changed, -notuptodate, -removed, -new)
- Optimization algorithms
  - $clasp_0$: lexicographical optimization
  - $clasp_1$: hierarchical optimization
  - $clasp_2$: hierarchical optimization with exponential steps
- Optimization heuristics
  - $clasp_i^0$: no optimization-specific heuristic
  - $clasp_i^1$: falsify literals to minimize upon branching
  - $clasp_i^2$: falsify literals to minimize until conflict
  - $clasp_i^3$: combines $clasp_i^1$ and $clasp_i^2$
- Search restarts
  - $clasp_i^j$-r: perform restart after each model (mandatory with $clasp_i^2$ and $clasp_i^3$)
- Scoring like in MISC-live run

Setup

- Benchmarks
  - 117 instances from the 3rd MISC-live run
  - Optimization criteria
    - paranoid, trendy
    - user1 (-notuptodate, -removed, -changed)
    - user2 (-changed, -removed, -unsat_recommends, -new)
    - user3 (-changed, -notuptodate, -removed, -new)
- Optimization algorithms
  - $clasp_0$: lexicographical optimization
  - $clasp_1$: hierarchical optimization
  - $clasp_2$: hierarchical optimization with exponential steps
- Optimization heuristics
  - $clasp_i^0$: no optimization-specific heuristic
  - $clasp_i^1$: falsify literals to minimize upon branching
  - $clasp_i^2$: falsify literals to minimize until conflict
  - $clasp_i^3$: combines $clasp_i^1$ and $clasp_i^2$
- Search restarts
  - $clasp_i^j$-r: perform restart after each model (mandatory with $clasp_i^2$ and $clasp_i^3$)
- Scoring like in MISC-live run

Setup

- Benchmarks
  - 117 instances from the 3rd MISC-live run
  - Optimization criteria
    - paranoid, trendy
    - user1 (-notuptodate, -removed, -changed)
    - user2 (-changed, -removed, -unsat_recommends, -new)
    - user3 (-changed, -notuptodate, -removed, -new)
- Optimization algorithms
  - $clasp_0$: lexicographical optimization
  - $clasp_1$: hierarchical optimization
  - $clasp_2$: hierarchical optimization with exponential steps
- Optimization heuristics
  - $clasp_i^0$: no optimization-specific heuristic
  - $clasp_i^1$: falsify literals to minimize upon branching
  - $clasp_i^2$: falsify literals to minimize until conflict
  - $clasp_i^3$: combines $clasp_i^1$ and $clasp_i^2$
- Search restarts
  - $clasp_i^j$-r: perform restart after each model (mandatory with $clasp_i^2$ and $clasp_i^3$)
- Scoring like in MISC-live run

	paranoid		trendy		user1		user2		user3	
Solver	S	T/O	S	T/O	S	T/O	S	T/O	S	T/O
$clasp_0^0$-r	431	2,287/6	1730	23,829/ 80	935	14,349/35	525	5,097/12	1031	14,184/37
$clasp_0^0$	416	2,294/6	2375	29,781/105	1727	21,897/73	1224	14,697/45	671	11,178/21
$clasp_0^1$-r	**410**	2,210/6	1560	22,660/ 73	898	13,466/30	502	4,654/ 9	980	13,682/35
$clasp_0^1$	**410**	2,326/6	2079	26,471/ 92	1723	21,525/72	922	10,767/31	658	10,675/23
$clasp_0^2$-r	427	2,135/6	712	16,867/ 51	527	5,891/11	426	2,981/ 5	587	7,628/20
$clasp_0^3$-r	429	**2,134**/6	740	17,079/ 52	507	5,863/12	425	3,044/ 6	576	7,769/21
$clasp_1^0$-r	425	2,428/6	579	16,713/ 50	550	5,819/14	434	3,000/ 6	710	8,958/25
$clasp_1^0$	417	2,418/6	549	16,544/ 50	**475**	5,318/12	**411**	2,538/ 5	502	6,279/16
$clasp_1^1$-r	429	2,405/6	622	17,304/ 50	518	5,908/13	438	2,976/ 6	676	8,938/23
$clasp_1^1$	427	2,372/6	613	16,946/ 49	490	5,478/12	416	2,562/ 5	496	6,144/16
$clasp_1^2$-r	427	2,352/6	571	16,646/ 50	518	5,358/13	418	2,582/ 5	471	6,356/16
$clasp_1^3$-r	429	2,346/6	**547**	**16,386**/ 50	499	**5,306**/12	413	2,498/ 5	497	6,255/16
$clasp_2^0$-r	425	2,392/6	806	16,598/ 50	523	5,583/13	421	2,677/ 6	479	5,548/12
$clasp_2^0$	417	2,364/7	748	17,132/ 50	487	5,823/14	422	2,583/ 5	482	5,592/15
$clasp_2^1$-r	416	2,378/6	752	17,269/ 52	492	5,663/12	414	**2,409**/ 5	451	**5,349**/11
$clasp_2^1$	425	2,365/6	864	17,128/ 51	517	6,151/15	412	2,681/ 5	463	5,972/14
$clasp_2^2$-r	445	2,402/6	706	16,551/ 50	528	5,788/13	419	2,700/ 5	**436**	5,519/13
$clasp_2^3$-r	434	2,345/6	748	16,982/ 51	518	5,850/14	415	2,559/ 5	457	5,360/13
cudf2msu	610	3,051/8	**669**	**5,318**/ 8	1270	8,709/18	548	**3,238**/ 7	**504**	4,750/ 9
cudf2pbo	465	**2,727**/7	1082	21,302/ 68	520	6,168/13	**462**	3,575/ 7	537	**3,487**/ 8
p2cudf	**463**	2,920/8	696	19,105/ 60	**516**	**3,947**/ 7	573	6,927/16	577	8,063/21

	paranoid		trendy		user1		user2		user3	
Solver	S	T/O	S	T/O	S	T/O	S	T/O	S	T/O
$clasp_0^0$-r	431	2,287/6	1730	23,829/ 80	935	14,349/35	525	5,097/12	1031	14,184/37
$clasp_0^0$	416	2,294/6	2375	29,781/105	1727	21,897/73	1224	14,697/45	671	11,178/21
$clasp_0^1$-r	**410**	2,210/6	1560	22,660/ 73	898	13,466/30	502	4,654/ 9	980	13,682/35
$clasp_0^1$	**410**	2,326/6	2079	26,471/ 92	1723	21,525/72	922	10,767/31	658	10,675/23
$clasp_0^2$-r	427	2,135/6	712	16,867/ 51	527	5,891/11	426	2,981/ 5	587	7,628/20
$clasp_0^3$-r	429	**2,134**/6	740	17,079/ 52	507	5,863/12	425	3,044/ 6	576	7,769/21
$clasp_1^0$-r	425	2,428/6	579	16,713/ 50	550	5,819/14	434	3,000/ 6	710	8,958/25
$clasp_1^0$	417	2,418/6	549	16,544/ 50	**475**	5,318/12	**411**	2,538/ 5	502	6,279/16
$clasp_1^1$-r	429	2,405/6	622	17,304/ 50	518	5,908/13	438	2,976/ 6	676	8,938/23
$clasp_1^1$	427	2,372/6	613	16,946/ 49	490	5,478/12	416	2,562/ 5	496	6,144/16
$clasp_1^2$-r	427	2,352/6	571	16,646/ 50	518	5,358/13	418	2,582/ 5	471	6,356/16
$clasp_1^3$-r	429	2,346/6	**547**	**16,386**/ 50	499	**5,306**/12	413	2,498/ 5	497	6,255/16
$clasp_2^0$-r	425	2,392/6	806	16,598/ 50	523	5,583/13	421	2,677/ 6	479	5,548/12
$clasp_2^0$	417	2,364/7	748	17,132/ 50	487	5,823/14	422	2,583/ 5	482	5,592/15
$clasp_2^1$-r	416	2,378/6	752	17,269/ 52	492	5,663/12	414	**2,409**/ 5	451	**5,349**/11
$clasp_2^1$	425	2,365/6	864	17,128/ 51	517	6,151/15	412	2,681/ 5	463	5,972/14
$clasp_2^2$-r	445	2,402/6	706	16,551/ 50	528	5,788/13	419	2,700/ 5	**436**	5,519/13
$clasp_2^3$-r	434	2,345/6	748	16,982/ 51	518	5,850/14	415	2,559/ 5	457	5,360/13
cudf2msu	610	3,051/8	**669**	**5,318**/ 8	1270	8,709/18	548	**3,238**/ 7	**504**	4,750/ 9
cudf2pbo	465	**2,727**/7	1082	21,302/ 68	520	6,168/13	**462**	3,575/ 7	537	**3,487**/ 8
p2cudf	**463**	2,920/8	696	19,105/ 60	**516**	**3,947**/ 7	573	6,927/16	577	8,063/21

	paranoid		trendy		user1		user2		user3	
Solver	S	T/O	S	T/O	S	T/O	S	T/O	S	T/O
$clasp_0^0$-r	431	2,287/6	1730	23,829/ 80	935	14,349/35	525	5,097/12	1031	14,184/37
$clasp_0^0$	416	2,294/6	2375	29,781/105	1727	21,897/73	1224	14,697/45	671	11,178/21
$clasp_0^1$-r	**410**	2,210/6	1560	22,660/ 73	898	13,466/30	502	4,654/ 9	980	13,682/35
$clasp_0^1$	**410**	2,326/6	2079	26,471/ 92	1723	21,525/72	922	10,767/31	658	10,675/23
$clasp_0^2$-r	427	2,135/6	712	16,867/ 51	527	5,891/11	426	2,981/ 5	587	7,628/20
$clasp_0^3$-r	429	**2,134**/6	740	17,079/ 52	507	5,863/12	425	3,044/ 6	576	7,769/21
$clasp_1^0$-r	425	2,428/6	579	16,713/ 50	550	5,819/14	434	3,000/ 6	710	8,958/25
$clasp_1^0$	417	2,418/6	549	16,544/ 50	**475**	5,318/12	**411**	2,538/ 5	502	6,279/16
$clasp_1^1$-r	429	2,405/6	622	17,304/ 50	518	5,908/13	438	2,976/ 6	676	8,938/23
$clasp_1^1$	427	2,372/6	613	16,946/ 49	490	5,478/12	416	2,562/ 5	496	6,144/16
$clasp_1^2$-r	427	2,352/6	571	16,646/ 50	518	5,358/13	418	2,582/ 5	471	6,356/16
$clasp_1^3$-r	429	2,346/6	**547**	**16,386**/ 50	499	**5,306**/12	413	2,498/ 5	497	6,255/16
$clasp_2^0$-r	425	2,392/6	806	16,598/ 50	523	5,583/13	421	2,677/ 6	479	5,548/12
$clasp_2^0$	417	2,364/7	748	17,132/ 50	487	5,823/14	422	2,583/ 5	482	5,592/15
$clasp_2^1$-r	416	2,378/6	752	17,269/ 52	492	5,663/12	414	**2,409**/ 5	451	**5,349**/11
$clasp_2^1$	425	2,365/6	864	17,128/ 51	517	6,151/15	412	2,681/ 5	463	5,972/14
$clasp_2^2$-r	445	2,402/6	706	16,551/ 50	528	5,788/13	419	2,700/ 5	**436**	5,519/13
$clasp_2^3$-r	434	2,345/6	748	16,982/ 51	518	5,850/14	415	2,559/ 5	457	5,360/13
cudf2msu	610	3,051/8	**669**	**5,318**/ 8	1270	8,709/18	548	**3,238**/ 7	**504**	4,750/ 9
cudf2pbo	465	**2,727**/7	1082	21,302/ 68	520	6,168/13	**462**	3,575/ 7	537	**3,487**/ 8
p2cudf	**463**	2,920/8	696	19,105/ 60	**516**	**3,947**/ 7	573	6,927/16	577	8,063/21

	paranoid		trendy		user1		user2		user3	
Solver	S	T/O	S	T/O	S	T/O	S	T/O	S	T/O
$clasp_0^0$-r	431	2,287/6	1730	23,829/ 80	935	14,349/35	525	5,097/12	1031	14,184/37
$clasp_0^0$	416	2,294/6	2375	29,781/105	1727	21,897/73	1224	14,697/45	671	11,178/21
$clasp_0^1$-r	**410**	2,210/6	1560	22,660/ 73	898	13,466/30	502	4,654/ 9	980	13,682/35
$clasp_0^1$	**410**	2,326/6	2079	26,471/ 92	1723	21,525/72	922	10,767/31	658	10,675/23
$clasp_0^2$-r	427	2,135/6	712	16,867/ 51	527	5,891/11	426	2,981/ 5	587	7,628/20
$clasp_0^3$-r	429	**2,134**/6	740	17,079/ 52	507	5,863/12	425	3,044/ 6	576	7,769/21
$clasp_1^0$-r	425	2,428/6	579	16,713/ 50	550	5,819/14	434	3,000/ 6	710	8,958/25
$clasp_1^0$	417	2,418/6	549	16,544/ 50	**475**	5,318/12	**411**	2,538/ 5	502	6,279/16
$clasp_1^1$-r	429	2,405/6	622	17,304/ 50	518	5,908/13	438	2,976/ 6	676	8,938/23
$clasp_1^1$	427	2,372/6	613	16,946/ 49	490	5,478/12	416	2,562/ 5	496	6,144/16
$clasp_1^2$-r	427	2,352/6	571	16,646/ 50	518	5,358/13	418	2,582/ 5	471	6,356/16
$clasp_1^3$-r	429	2,346/6	**547**	**16,386**/ 50	499	**5,306**/12	413	2,498/ 5	497	6,255/16
$clasp_2^0$-r	425	2,392/6	806	16,598/ 50	523	5,583/13	421	2,677/ 6	479	5,548/12
$clasp_2^0$	417	2,364/7	748	17,132/ 50	487	5,823/14	422	2,583/ 5	482	5,592/15
$clasp_2^1$-r	416	2,378/6	752	17,269/ 52	492	5,663/12	414	**2,409**/ 5	451	**5,349**/11
$clasp_2^1$	425	2,365/6	864	17,128/ 51	517	6,151/15	412	2,681/ 5	463	5,972/14
$clasp_2^2$-r	445	2,402/6	706	16,551/ 50	528	5,788/13	419	2,700/ 5	**436**	5,519/13
$clasp_2^3$-r	434	2,345/6	748	16,982/ 51	518	5,850/14	415	2,559/ 5	457	5,360/13
cudf2msu	610	3,051/8	**669**	**5,318**/ 8	1270	8,709/18	548	**3,238**/ 7	**504**	4,750/ 9
cudf2pbo	465	**2,727**/7	1082	21,302/ 68	520	6,168/13	**462**	3,575/ 7	537	**3,487**/ 8
p2cudf	**463**	2,920/8	696	19,105/ 60	**516**	**3,947**/ 7	573	6,927/16	577	8,063/21

	paranoid		trendy		user1		user2		user3	
Solver	S	T/O	S	T/O	S	T/O	S	T/O	S	T/O
$clasp_0^0$-r	431	2,287/6	1730	23,829/ 80	935	14,349/35	525	5,097/12	1031	14,184/37
$clasp_0^0$	416	2,294/6	2375	29,781/105	1727	21,897/73	1224	14,697/45	671	11,178/21
$clasp_0^1$-r	**410**	2,210/6	1560	22,660/ 73	898	13,466/30	502	4,654/ 9	980	13,682/35
$clasp_0^1$	**410**	2,326/6	2079	26,471/ 92	1723	21,525/72	922	10,767/31	658	10,675/23
$clasp_0^2$-r	427	2,135/6	712	16,867/ 51	527	5,891/11	426	2,981/ 5	587	7,628/20
$clasp_0^3$-r	429	**2,134**/6	740	17,079/ 52	507	5,863/12	425	3,044/ 6	576	7,769/21
$clasp_1^0$-r	425	2,428/6	579	16,713/ 50	550	5,819/14	434	3,000/ 6	710	8,958/25
$clasp_1^0$	417	2,418/6	549	16,544/ 50	**475**	5,318/12	**411**	2,538/ 5	502	6,279/16
$clasp_1^1$-r	429	2,405/6	622	17,304/ 50	518	5,908/13	438	2,976/ 6	676	8,938/23
$clasp_1^1$	427	2,372/6	613	16,946/ 49	490	5,478/12	416	2,562/ 5	496	6,144/16
$clasp_1^2$-r	427	2,352/6	571	16,646/ 50	518	5,358/13	418	2,582/ 5	471	6,356/16
$clasp_1^3$-r	429	2,346/6	**547**	**16,386**/ 50	499	**5,306**/12	413	2,498/ 5	497	6,255/16
$clasp_2^0$-r	425	2,392/6	806	16,598/ 50	523	5,583/13	421	2,677/ 6	479	5,548/12
$clasp_2^0$	417	2,364/7	748	17,132/ 50	487	5,823/14	422	2,583/ 5	482	5,592/15
$clasp_2^1$-r	416	2,378/6	752	17,269/ 52	492	5,663/12	414	**2,409**/ 5	451	**5,349**/11
$clasp_2^1$	425	2,365/6	864	17,128/ 51	517	6,151/15	412	2,681/ 5	463	5,972/14
$clasp_2^2$-r	445	2,402/6	706	16,551/ 50	528	5,788/13	419	2,700/ 5	**436**	5,519/13
$clasp_2^3$-r	434	2,345/6	748	16,982/ 51	518	5,850/14	415	2,559/ 5	457	5,360/13
cudf2msu	610	3,051/8	**669**	**5,318**/ 8	1270	8,709/18	548	**3,238**/ 7	**504**	4,750/ 9
cudf2pbo	465	**2,727**/7	1082	21,302/ 68	520	6,168/13	**462**	3,575/ 7	537	**3,487**/ 8
p2cudf	**463**	2,920/8	696	19,105/ 60	**516**	**3,947**/ 7	573	6,927/16	577	8,063/21

Outline

Discussion

- Multi-criteria optimization algorithm
  - optimizing criteria in the order of significance
  - keeping learned information whenever possible
  - retracting invalid constraints using assumptions
  - avoiding solver relaunches after unsatisfiability proofs
- Optimization-oriented heuristics to guide search for optima
- Techniques used in package configuration tool *aspcud*
- Future work: combination with lower bound refinement