CSE505-Project

Yueqi Hu

Marple K, Gupta G. **Dynamic consistency checking in goal-directed answer set programming**[J]. Theory and Practice of Logic Programming, 2014, 14(4-5): 415-427.

Main Idea

- dependencies of (X) := {A : X depends on A}
- Odd loop over negation (OLON) rules
- ———> p:- B, not p. 1.p succeed 2. one of B fail
- Break OLON into NMR sub-checks to check dependency
- Splitting sets to determine relevant NMR sub-checks

Process

- Write the ASP problems into lparse style
- Use the lparse parsing into variable free
- Evaluate on the Galliwasp W/ W/O DCC
  - lparse: the front-end of smodel http://www.tcs.hut.fi/Software/smodels/

Process

```
% One day, three inhabitants (A, B, and C) of the island met a foreign
% tourist and gave the following information about themselves:
%
% 1. A said that B and C are both knights.
% 2. B said that A is a knave and C is a knight.
%
% What types are A, B, and C?
%

% There are three persons in this puzzle
person(a; b; c).

% Each person is either knight or knave, but not both.
1 { knight(P), knave(P) } 1 :- person(P).

% Rest of the rules model the hints.

%% Hint 1:

% If A tells the truth, both B and C are knights
2 { knight(b), knight(c) } 2 :- knight(a).

% If A lies, it is not possible that they are both knights
  :- knave(a), knight(b), knight(c).

%% Hint 2:

% If B tells the truth, A is a knave and C is a knight
2 { knave(a), knight(c) } 2 :- knight(b).

% If B lies, it is not possible that A is knave and C is knight
  :- knave(b), knave(a), knight(c).
```

=>

puzzle1.txt

```
1 1 3 0 2  3  4
1 1 3 0 5  2  4
1 1 1 0 6
2 6 2 2 2 7 4
1 1 1 0 8
2 8 2 2 2 5 3
1 1 1 0 9
2 9 2 2 2 2 10
1 1 1 0 11
2 11 2 0 2 4 7
1 1 1 0 12
2 12 2 0 2 3 5
1 1 1 0 13
2 13 2 0 2 10 2
1 1 2 0 10  14
2 14 2 2 1 4 3
1 1 2 0 10  15
1 1 2 0 3  16
2 16 2 2 1 4 2
1 1 2 0 3  17
3 2 3 4 1 0 10
3 2 2 4 1 0 3
3 2 4 7 0 0
3 2 3 5 0 0
3 2 10 2 0 0
1 18 0 0
1 19 0 0
1 20 0 0
0
2 knave(a)
3 knight(b)
4 knight(c)
5 knave(b)
7 knave(c)
10 knight(a)
18 person(c)
19 person(b)
20 person(a)
0
B+
0
B-
1
0
1
```

puzzle1-.txt

```
1 1 3 0 2  3  4
1 1 3 0 5  2  4
1 1 1 0 6
2 6 2 2 2 7 4
1 1 1 0 8
2 8 2 2 2 5 3
1 1 1 0 9
2 9 2 2 2 2 10
1 1 1 0 11
2 11 2 0 2 4 7
1 1 1 0 12
2 12 2 0 2 3 5
1 1 1 0 13
2 13 2 0 2 10 2
1 1 2 0 10  14
2 14 2 2 1 4 3
1 1 2 0 10  15
1 1 2 0 3  16
2 16 2 2 1 4 2
1 1 2 0 3  17
1 18 1 1 18
3 2 3 4 1 0 10
3 2 2 4 1 0 3
3 2 4 7 0 0
3 2 3 5 0 0
3 2 10 2 0 0
1 19 0 0
1 20 0 0
1 21 0 0
0
2 knave(a)
3 knight(b)
4 knight(c)
5 knave(b)
7 knave(c)
10 knight(a)
18 p
19 person(c)
20 person(b)
21 person(a)
0
B+
0
B-
1
0
1
```

Running Examples

- knight knave puzzle
  - 1. A said that B and C are both knights.
  - 2. B said that A is a knave and C is a knight.

```
{ knave(a), knave(b), knave(c), person(a), person(b), person(c) }

real    0m0.069s
user    0m0.042s
sys     0m0.016s
```

- knight knave puzzle
- + p:- not p W/DCC

```
{ knave(a), knave(b), knave(c), person(a), person(b), person(c) }

real    0m0.048s
user    0m0.039s
sys     0m0.011s
```

Running Examples

- 15puzzle 13steps + p:- not p.

```
false.

real    0m0.066s
user    0m0.041s
sys     0m0.018s
```

- 15puzzle 13steps + p:- not p. W/DCC

```
{ entry(0), entry(1), entry(10), entry(11), entry(12), entry(13), entry(14), ent
ry(15), entry(2), entry(3), entry(4), entry(5), entry(6), entry(7), entry(8), en
try(9), in0(1,1,1), in0(1,2,5), in0(1,3,2), in0(1,4,7), in0(2,1,0), in0(2,2,4),
in0(2,3,3), in0(2,4,11), in0(3,1,8), in0(3,2,9), in0(3,3,6), in0(3,4,15), in0(4,
1,12), in0(4,2,13), in0(4,3,10), in0(4,4,14), maxtime(13), pos(1), pos(2), pos(3
), pos(4), time(0), time(1), time(10), time(11), time(12), time(13), time(2), ti
me(3), time(4), time(5), time(6), time(7), time(8), time(9) }

real    0m0.048s
user    0m0.037s
sys     0m0.012s
```

Running Examples

- hanoi tower1 3-7

```
{ disk(1), disk(2), disk(3), move(1,1,2), move(2,1,3), move(3,2,3), move(4,1,2),
 move(5,3,1), move(6,3,2), move(7,1,2), moven(1,0,1,1,2,3), moven(1,2,3,2,3,1),
moven(1,4,5,3,1,2), moven(1,6,7,1,2,3), moven(2,0,3,1,3,2), moven(2,4,7,3,2,1),
moven(3,0,7,1,2,3), peg(1), peg(2), peg(3), time(0), time(1), time(2), time(3),
time(4), time(5), time(6), time(7) }

real    0m0.239s
user    0m0.118s
sys     0m0.024s
```

- hanoi tower1 4-15

```
{ disk(1), disk(2), disk(3), disk(4), move(1,1,3), move(10,3,1), move(11,2,1), m
ove(12,3,2), move(13,1,3), move(14,1,2), move(15,3,2), move(2,1,2), move(3,3,2),
 move(4,1,3), move(5,2,1), move(6,2,3), move(7,1,3), move(8,1,2), move(9,3,2), m
oven(1,0,1,1,3,2), moven(1,10,11,2,1,3), moven(1,12,13,1,3,2), moven(1,14,15,3,2
,1), moven(1,2,3,3,2,1), moven(1,4,5,2,1,3), moven(1,6,7,1,3,2), moven(1,8,9,3,2
,1), moven(2,0,3,1,2,3), moven(2,12,15,1,2,3), moven(2,4,7,2,3,1), moven(2,8,11,
3,1,2), moven(3,0,7,1,3,2), moven(3,8,15,3,2,1), moven(4,0,15,1,2,3), peg(1), pe
g(2), peg(3), time(0), time(1), time(10), time(11), time(12), time(13), time(14)
, time(15), time(2), time(3), time(4), time(5), time(6), time(7), time(8), time(
9) }

real    0m1.000s
user    0m0.831s
sys     0m0.078s
```

Running Examples

- hanoi tower1 4-15
- + p:- not p

```
false.

real    0m5.325s
user    0m5.128s
sys     0m0.232s
```

- hanoi tower1 4-15
- + p:- not p W/DCC

```
{ disk(1), disk(2), disk(3), disk(4), move(1,1,3), move(10,3,1), move(11,2,1), m
ove(12,3,2), move(13,1,3), move(14,1,2), move(15,3,2), move(2,1,2), move(3,3,2),
 move(4,1,3), move(5,2,1), move(6,2,3), move(7,1,3), move(8,1,2), move(9,3,2), m
oven(1,0,1,1,3,2), moven(1,10,11,2,1,3), moven(1,12,13,1,3,2), moven(1,14,15,3,2
,1), moven(1,2,3,3,2,1), moven(1,4,5,2,1,3), moven(1,6,7,1,3,2), moven(1,8,9,3,2
,1), moven(2,0,3,1,2,3), moven(2,12,15,1,2,3), moven(2,4,7,2,3,1), moven(2,8,11,
3,1,2), moven(3,0,7,1,3,2), moven(3,8,15,3,2,1), moven(4,0,15,1,2,3), peg(1), pe
g(2), peg(3), time(0), time(1), time(10), time(11), time(12), time(13), time(14)
, time(15), time(2), time(3), time(4), time(5), time(6), time(7), time(8), time(
9) }

real    0m5.219s
user    0m5.058s
sys     0m0.201s
```

```
{ disk(1), disk(2), disk(3), disk(4), peg(1), peg(2), peg(3), time(0), time(1),
time(10), time(11), time(12), time(13), time(14), time(15), time(2), time(3), ti
me(4), time(5), time(6), time(7), time(8), time(9) }

real    0m1.021s
user    0m0.835s
sys     0m0.101s
```

Running Examples

- Hanoi-tower2 6-36

```
{ disk(1), disk(2), disk(3), disk(4), disk(5), disk(6), disk(7), disk(8), disk(9
), on(0,1,4), on(0,2,5), on(0,3,8), on(0,4,7), on(0,5,6), on(0,8,9), on0(1,4), o
n0(2,5), on0(3,8), on0(4,7), on0(5,6), on0(8,9), time(0), time(1), time(10), tim
e(11), time(12), time(13), time(14), time(15), time(16), time(17), time(18), tim
e(19), time(2), time(20), time(21), time(22), time(23), time(24), time(25), time
(26), time(27), time(28), time(29), time(3), time(30), time(31), time(32), time(
33), time(34), time(35), time(4), time(5), time(6), time(7), time(8), time(9) }

real    0m0.320s
user    0m0.199s
sys     0m0.029s
```

- Hanoi-tower2 6-36
- + p:- not p W/DCC

```
{ disk(1), disk(2), disk(3), disk(4), disk(5), disk(6), disk(7), disk(8), disk(9
), on(0,1,4), on(0,2,5), on(0,3,8), on(0,4,7), on(0,5,6), on(0,8,9), on0(1,4), o
n0(2,5), on0(3,8), on0(4,7), on0(5,6), on0(8,9), time(0), time(1), time(10), tim
e(11), time(12), time(13), time(14), time(15), time(16), time(17), time(18), tim
e(19), time(2), time(20), time(21), time(22), time(23), time(24), time(25), time
(26), time(27), time(28), time(29), time(3), time(30), time(31), time(32), time(
33), time(34), time(35), time(4), time(5), time(6), time(7), time(8), time(9) }

real    0m0.312s
user    0m0.189s
sys     0m0.030s
```

Remain Work

- To write more complicated ASP programs to evaluate
- To combine the problems together
- To increase the size of the inconsistency in experiment

Thanks!