Action Theories with Time Constraints

This is a short description of a first attempt on combining ASP reasoning techniques with constraint logic programming.

Michael, Jan 31, 05

The Problem

Given: a recorded history H **of the domain together with time constraints, e.g.**

$time(i, t)$ **-** t **is the actual time of execution of the** i**'th action of** H**;**

$duration(a, t)$ **-** t **is the time needed to execute** a**;**

$time\_boundary(a, d0, d1)$ **- execution time** D **of** a **belongs to interval** $[d1, d2]$**.**

Find: all possible assignments of time to time-steps of H**.**

ASP solution

The assignments can be found by a selection rule

```
{time(I,T) : time_init(T)} :- step(I).
```

and the appropriate constraints, e.g.

```
:- o(A,I),
   duration(A,D),
   time(I,D1),
   time(I+1,D2),
   D2 - D1 < D - D1.
```

In some applications the domain of I is too big for answer set solvers and computation becomes too slow.

Changing the inference engine

Main idea: Find the assignments using the resolution and constraint satisfaction algorithms.

A legal assignment will have a form $[T_1, \ldots, T_n]$ where T_i is the time assigned to step i of history H.

Domain of the variables is the set of integers from 1 to k where k is the maximum number of time units.

Stating the CLP problem

Constraint programming rule defining legal assignment of our problem is of the form:

```
legal_assignment(L) :-
                length(L,N),
                max_time(K),
                domain(L, 1, K),
                c0(L, N),
                c1(L, N),
                c2(L,N),
                c3(L,N),
                labeling([], L).
```

Stating the constraints

if $S1 < S2$ **then** $time(S1) \leq time(S2)$

```
c0(L, N) :-
    N #< 2.
c0([X | [Y|R]], N) :-
    X #=< Y,
    N1 is N - 1,
    c0([Y | R], N1).
```

Stating the constraints

A legal assignment must agree with time constraint of the form $time(i, t)$

```
c1(L, 0).
c1(L, I) :-
    (time(I, X) -> nth(I, L, X) ; true),
    I1 is I - 1,
    I1 >= 0,
    c1(L, I1).
```

To be done

- **Combine with APL-query to be able to reason about multiple models;**
- **Expand input tempral language;**
- **Study efficiency of different constraint solvers;**
- **Design an implement an answer set solver to more efficiently combine various reasoning algorithms.**