ecg.pl

Tim Menzies
Lane Deptartment of Computer Science and Electrical Engineering
West Virginia University
Morgantown, WV 26506
tim@menzies.us

Contents

1 Extended clause grammers

Speed up the manipulation of named fields within DCGs.

2 Header

2.1 Loads */

```
:- [defs]. /*
```

3 Body

3.1 Standard Optimizations */

```
goal_expansion(append(A,B,_),true) :- ground(A),ground(B).
goal_expansion(append(A,_,C),true) :- ground(A),ground(C).
goal_expansion(append(_,B,C),true) :- ground(B),ground(C). /*
```

3.2 Reparing DCG expansion

Bunch of tricks to repair over-zealous DCG expansion. */

```
goal_expansion(true(X,X),      true).
goal_expansion(fail(X,X),      fail).
goal_expansion(once(X,Y,Y),    once(X)).
goal_expansion(print(X,Y,Y),   print(X)).
goal_expansion(format(X,Y,Z,Z),format(X,Y)). /*
```

The above code drops in a lot of `true`s that we really we should cull- but not tonight.

3.3 Hooks into the defs systems */

```
goal_expansion( -(F,X,Y),  Z) :- clause(at(-F,    X,Y),Z).
goal_expansion( +(F,X,Y),  Z) :- clause(at(+F,    X,Y),Z).
goal_expansion( +(F,V,X,Y),Z) :- clause(at(F+V,   X,Y),Z).
goal_expansion( =(F,V,X,Y),Z) :- clause(at(F =  V,X,Y),Z).
goal_expansion(is(F,V,X,Y),Z) :- clause(at(F is V,X,Y),Z).
goal_expansion(>=(F,V,X,Y),Z) :- clause(at(F >= V,X,Y),Z).
goal_expansion( >(F,V,X,Y),Z) :- clause(at(F >  V,X,Y),Z).
goal_expansion( <(F,V,X,Y),Z) :- clause(at(F <  V,X,Y),Z).
goal_expansion(=<(F,V,X,Y),Z) :- clause(at(F =< V,X,Y),Z).
goal_expansion(\=(F,V,X,Y),Z) :- clause(at(F \= V,X,Y),Z).
goal_expansion(:=(F,V,X,Y),Z) :- clause(at(F:=V,  X,Y),Z). /*
```

3.4 Optimizing *Solo* Calls

For certain clauses, if there is only one clause that matches some sub-goal, then we can eval it at load time with safety. */

```
eval_if_solo(at(_,_,_,_,_)).
eval_if_solo(at(_,_,_)).

solo(X) :- Y='#solo', flag(Y,_,0), \+ solo1(Y,X), flag(Y,1,1).
solo1(Sym,X) :- clause(X,_),flag(Sym,N,N+1),N > 1,!.

goal_expansion(X,true) :- eval_if_solo(X), solo(X), X.  /*
```

3.5 Maintaining Context

If we know what kind of `def` we are currently expanding, we can check for missing fields. */

```
goal_expansion(in(T,X,Y),true)     :- in(T,X,Y). /*
```

This next one is tricky: SWI's DCG expansion does not unify the output variable after a {X} expansion so our meta-knowledge that we are carrying round a certain `def` is lost. So we have to force that particular unification: */

```
goal_expansion(A=(B=C),true) :- ground(B),def(B,_), A=(B=C). /*
```

Which means that we can't use 'X;Y' or 'X -> Y ; Z' in ECGs. So replace these with the usual expansions; i.e.

```
X :- Y,!,Z.
X.

X :- Y.
X :- Z. */
```