A Note on Backtracking in Prolog

Tyng-Ruey Chuang

2009-06-10

Let's explain backtracking in Prolog using the following prolog clauses, in which we describe what natural numbers pass as even numbers, and what as odd numbers. All natural numbers are built from the constant `0` and the unary function symbol `s`.

```
even(0).
even(s(N)) :- odd(N).
odd(s(N)) :- even(N).
```

Note that we put the clause for the number `0` ahead of the clause for any other number `s(N)`.

Suppose we want to know what number `X` will make number `s(X)` even. That is, can `even(s(X))` success? The reasoning (execution order) in Prolog will look like the following.

Can `even(s(X))` success? There are two choices for `even(s(X))` to success.

1. Can `even(s(X))` unify with `even(0)`? **no.**
2. Can `even(s(X))` unify with `even(s(Y))`, and can `odd(Y)` success?
   Yes, if `X = Y`, and `odd(Y)` success. There is only one choice for `odd(Y)` to success.
   That is, can `odd(Y)` unify with `odd(s(Z))`, and can `even(Z)` success?
   Yes, if `Y = s(Z)`, and `even(Z)` success. There are two choices for `even(Z)` to success.
   (a) Can `even(Z)` unify with `even(0)`? **yes**, with `Z = 0` (*)
   Hence, the answer `X = Y = s(Z) = s(0)` *will satisfy* `even(s(X))`.
   If this is not the answer we want, Prolog voids the **yes** *at (*), as well as the unification* `Z = 0`. *It proceeds — backtracks! — to the other choice (below).*
   (b) Can `even(Z)` unify with `even(s(U))`, and can `odd(U)` success?
   Yes, if `Z = s(U)`, and `odd(U)` success. There is only one choice for `odd(U)` to success.
   That is, can `odd(U)` unify with `odd(s(V))`, and can `even(V)` success?
   Yes, if `U = s(V)`, and `even(V)` success. There are two choices for `even(V)` to success.
      i. Can `even(V)` unify with `even(0)`? **yes**, with `V = 0` (**)
      Hence, the answer `X = Y = s(Z) = s(s(U)) = s(s(s(V))) = s(s(s(0)))` *will satisfy* `even(s(X))`.
      If this is not the answer we want, Prolog voids the **yes** *at (**), as well as the unification* `V = 0`.
      It proceeds — backtracks! — to the other choice (below).
      ii. Can `even(V))` unify with `even(s(W))`, and can `odd(W)` success?
      Yes, if `V = s(W)`, and `odd(W)` success. There is only one choice for `odd(W)` to success.
      . . . and here we go again and again . . .

Here are two questions for you to ponder.

- What if we change the ordering of the two rules for `even`? That is, for the Prolog program

  ```
  even(s(N)) :- odd(N).
  even(0).
  odd(s(N)) :- even(N).
  ```

  What will become of the execution for the query `even(s(X))`?

- What if we introduce a cut at `even(0)` in the following program?

  ```
  even(0) :- ! .
  even(s(N)) :- odd(N).
  odd(s(N)) :- even(N).
  ```

  What change will this bring to the execution for the query `even(s(X))`?