Programming in Prolog: Integers

- Represent Non-Negative Integers as

  0, s(0), s(s(0)), s(s(s(0))), ...

- Constant: 0
  Unary function symbol: s

(Just for Pedagogical Reasons!)

Addition

- *Prolog* uses *predicates*; not *functions*.
  Use

  sum(X, Y, Z)

  which holds *iff* X + Y = Z
  [not "sum(X,Y)" returns "Z"]

- Axioms for Addition:

  sum(0, Y, Y). *(1)*
  sum(s(X), Y, s(Z)) :- sum(X,Y,Z). *(2)*

- Notice: If first arg is constant,
  $\Rightarrow$ only 1 head will unify.

sum($\bar{2}$, $\bar{3}$, $\bar{5}$)
$\equiv$ sum(s($\bar{1}$), $\bar{3}$, s($\bar{4}$)) $\overset{(2)}{\rightsquigarrow}$ sum($\bar{1}$, $\bar{3}$, $\bar{4}$)
$\equiv$ sum(s($\bar{0}$), $\bar{3}$, s($\bar{3}$)) $\overset{(2)}{\rightsquigarrow}$ sum(0, $\bar{3}$, $\bar{3}$)
$\overset{(1)}{\rightsquigarrow}$ SUCCESS

Computing Integer Sums

Find Tot s.t. 2+3= Tot.

sum(s(s(0)), s(s(s(0))), Tot)
(2)
X_1 = s(0)
Y_1 = s(s(s(0)))
Tot = s(Z_1)
sum(s(0), s(s(s(0))), Z_1)
(2)
X_2 = 0
Y_2 = s(s(s(0)))
Z_1 = s(Z_2)
sum(0, s(s(s(0))), Z_2)
(1)
Z_2 = s(s(s(0)))
success

so Z_1 = s(Z_2) = s(s(s(0))))
Tot = s(Z_1) = s(s(s(s(0)))))

Verifying Integer Sums

Does "2" + "3" = "6"?

sum(s(s(0)), s(s(s(0))), s(s(s(s(s(0))))))
(2)
X_1 = s(0)
Y_1 = s(s(s(0)))
Z_1 = s(s(s(s(0))))
sum(s(0), s(s(s(0))), s(s(s(s(0)))))
(2)
X_2 = 0
Y_2 = s(s(0))
Z_2 = s(s(s(0)))
sum(0, s(s(s(0))), s(s(s(0)))))
No possible unification!
failure

What User Really Types (III)

```
obed% sicstus
C-Prolog version 1.4
| ?- [user].          % User now in ''Assert-fact'' mode.
| sum(0, Y, Y).
| sum( s(X), Y, s(Z)) :- sum(X,Y,Z).
| ^D user con...      % Typing ''^D'' exits ''assert'' mode.
yes
| ?- sum( s(s(0)), s(s(s(0))), Tot).
Tot = s(s(s(s(s(0))))) _
                      %  Prolog returns this binding list.
                      %  User types CR ⇒ answer is accepted
yes                   % ≡ ''Prolog found accepted answer''
| ?- sum( s(s(0)), s(s(s(0))), s(s(s(s(s(0)))))) ).
no                    % ≡ Prolog found no answers
| ?-
```


Comments wrt sum

- Notation: $\bar{n}$ means $\underbrace{s(s(\ldots(s(0))\ldots)}_{n\ times}$

Eg: $\begin{cases} \bar{2} = s(s(0)) \\ \bar{0} = 0 \end{cases}$

- Derived:
  sum($\bar{2}$, $\bar{3}$, Tot)
  succeeds with Tot = $\bar{5}$.

- Verified:
  sum($\bar{2}$, $\bar{3}$, $\bar{5}$) succeeds
  sum($\bar{2}$, $\bar{3}$, $\bar{6}$) fails

Subtracting using sum

Find W s.t. $\bar{1}$ + W = $\bar{3}$

sum(s(0), W, s(s(0))))

(2)
X_1 = 0
Y_1 = W
Z_1 = s(s(0))

sum(0, Y_1, s(s(0)))

(1)
Y_1 = s(s(0))

success

$\Rightarrow$ W = Y_1 = s(s(0)); ie, W = $\bar{2}$

---

sum(D, $\bar{3}$, $\bar{8}$) succeeds with D = $\bar{5}$.

To do subtraction directly:

```
diff(X,Y,Z) :- sum(Y,Z,X)
```


Other uses of sum

- sum($\bar{3}$, $\bar{4}$, s(s(W))) succeeds with W = $\bar{5}$
  W s.t. $3+4=(W+1)+1$

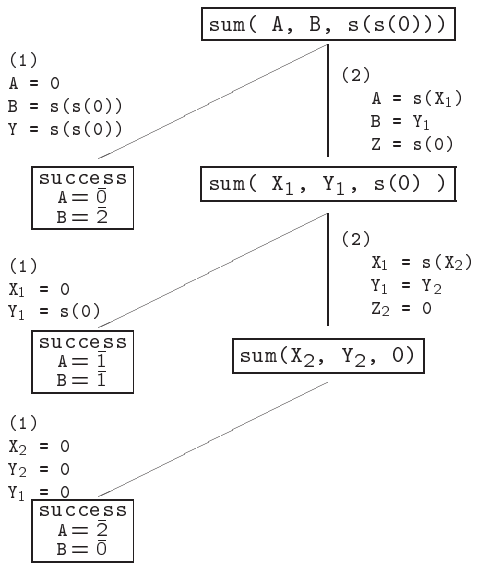
- sum($\bar{3}$, s(W), $\bar{6}$) succeeds with W = $\bar{2}$
  W s.t. $3+(W+1)=6$

- What about sum(A, B, $\bar{2}$)?

What User Really Types (IV)

```
| ?- [sumf].                    % File holds 2 sum facts.
yes
| ?- sum( s(0), W, s(s(0)) ).
                                % Find W s.t.  1̄ + W = 3̄.
W = s(s(0)) ;                   % Found W = 2̄
                                % User asks for another answer:
no                              % ≡ ¬∃ other answers
| ?- sum( D, s(s(s(0))), s(s(s(s(s(s(s(0))))))) ).
                                % Find D: D + 3̄ = 8̄.
D = s(s(s(s(s(0))))) _          % Found D = 5̄
yes
| ?- sum( s(s(s(0))), s(s(s(s(0)))), s(W) ).
                                % Find W: 3̄ + 4̄ = (W + 1) + 1
W = s(s(s(s(s(0))))) _          % Found W = 5̄
yes
| ?- sum( s(s(s(0))), s(W), s(s(s(s(s(s(0)))))) ).
                                % Find W: 3̄ + (W + 1) = 6̄
W = s(s(0)) _                   % Found W = 2̄
yes
| ?- sum( A, B, s(s(0)) ).
     ...
```


sum(A, B, s(s(0))) ?



sum and Variables

- sum(A, B, 5̄) succeeds with

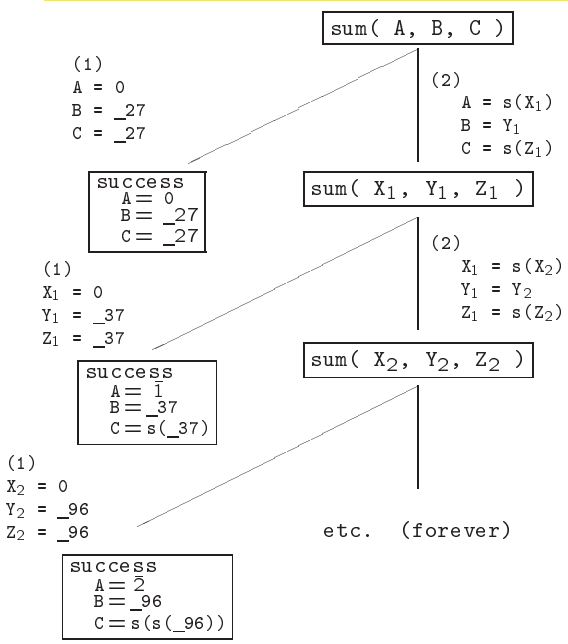
  A = 0̄ B = 5̄
  A = 1̄ B = 4̄
  A = 2̄ B = 3̄
  A = 3̄ B = 2̄
  A = 4̄ B = 1̄
  A = 5̄ B = 0̄

- What about sum(A, B, C) ?

  succeeds with A=0̄ B=_27 C=_27
  then A=1̄ B=_32 C=s(_32)
  then A=2̄ B=_97 C=s(s(_97))

  . . . for ever . . .

sum(A, B, C) ?



What User Really Types (V)

```
| ?- [sumf].          % File holds 2 sum facts.
yes
| ?- sum( A, B, s(s(0)) ).
                      % Find A, B s.t.  A + B = 2̄.
A = 0
B = s(s(0)) ;         % Found A = 0̄, B = 2̄
                      % User asks for another answer:
A = s(0)
B = s(0) ;            % Found A = 1̄, B = 1̄
A = s(s(0))
B = 0 ;               % Found A = 2̄, B = 0̄
no                    % ≡ There are no other answers
| ?- sum( s(s(A)), s(B), s(s(s(s(s(0))))) ).
                      % Find A, B s.t.
                      %     ((A + 1) + 1) + (B + 1)  =  4̄.
A = 0
B = s(0) ;            % Found A = 0̄, B = 1̄
A = s(0)
B = 0 _               % Found A = 1̄, B = 0̄
                      % User accepts this answer.
yes
| ?-
```


User's Typing — (con't)

```
| ?- sum( A, B, C).
                      % Find A, B, C   s.t.  A + B = C.
A = 0
B = _1
C = _1 ;              % Found   A = 0̄, B = C = anything
A = s(0)
B = _1
C = s(_1) ;           % Found  A = 1̄, B = anything
                      %   C = 1 + B
A = s(s(0))
B = _1
C = s(s(_1)) ;        % Found  A = 2̄, B = anything
                      %   C = 2 + B
A = s(s(s(0)))
B = _1
C = s(s(s(_1))) _     % Found  A = 3̄, B = anything
                      %   C = 3 + B
                      % User accepts this answer.
yes
| ?-
```


sum and half?

- What about Halving?

  `sum(H,H,1̄4̄)` succeeds with `H=7̄` .

- Consider

```
half(X,H) :- sum(H,H,X).
```

  . . . but

  half(5̄,H) ↦ sum(H, H, 5̄)
  ↦ sum(H, s(H), 4̄)
  ↦ sum(H, s(s(H)), 3̄)
  ↦ . . .
  ↦ sum(H, s(s(s(s(s(H))))), 0̄)
  ↦ ×

  fails, after 6 steps.

Need to worry about { *Success* / *Failure* } behavior!

Better Definition of half

- Consider

```
half(      0,  0).
half(s(s(X)),  s(H)) :- half(X,H).
```

Eg 1

`half(6̄,F)` ↦ half(4̄, H_1) [F = s(H_1)]
↦ half(2̄, H_2) [H_1 = s(H_2)]
↦ half(0̄, H_3) [H_2 = s(H_3)]
↦ $H_3 = 0 \Rightarrow F = s(s(H_3))) = \bar{3}$

which succeeds after *only 3 steps*!

Eg 2

`half(5̄,F)` ↦ half(3̄, H_1) [F = s(H_1)]
↦ half(1̄, H_2) [H_1 = s(H_2)]
↦ ×

which fails after *only 2 steps*!

- This def'n is
  Faster on BOTH success and failure.

Another Integer Example: Multiplication

```
| ?- [user].
| prod( 0, Y, 0 ).
| prod( s(X), Y, Z ) :- prod(X,Y,W), sum(Y,W,Z).
| ^D user con...
yes
| ?- prod( 0, s(s(s(0))), P1)
                    % Find P1:  0 × 4 = P1
P1 = 0 _            % succeeds with P1 = 0
yes
| ?- prod( s(0), s(s(s(0))), P2)
                    % Find P2:  1 × 4 = P2
P2 = s(s(s(0))) _
                    % succeeds with P2 = 4
yes                 % (requires 1 addition)
| ?- prod( s(s(s(0))), s(s(s(0))), P3)
                    % Find P3:  3 × 4 = P3
P3 = s(s(s(s(s(s(s(s(s(s(s(s(0)))))))))))) _
                    % succeeds with P3 = 12
yes                 % (requires 3 additions)
| ?-
```


Division using `prod`?

- Consider:

  `prod(2̄, Q, 6̄)` ⇝ `prod(1̄,Q,W),sum(Q,W,6̄)`

  `prod(1̄, Q, W)` ⇝ `prod(0,Q,W'),sum(Q,W',W)`

  `prod(0, Q, W')` ⇝ `{W'/0}`

  `sum(Q,W',W){W'/0}` ⇝ `sum(Q,0,W)`

  Need to solve (correct) subgoal:

  `sum(Q,0,W), sum(Q,W,6̄)`

tries		
	$Q = \bar{0},\ W = \bar{0}$	fails
	$Q = \bar{1},\ W = \bar{1}$	fails
	$Q = \bar{2},\ W = \bar{2}$	fails
	$Q = \bar{3},\ W = \bar{3}$	succeeds

  A terrible way to do division!

- Worse yet:

  `prod(2̄,Q,5̄)` runs forever!

  (can fix by re-ordering 2^{nd} axiom.)

How not to Program Using `prod` for Square Root

- `prod(S,S,4̄9)`

  ⇒ { S=s(X), S=Y, Z=4̄9 }

  ⇒ { S/s(X), s(X)/Y, Z/4̄9 }

  `prod(X,s(X),W), sum(s(X),W,4̄9)`

- Eventually succeeds with X=6̄, W=4̄2 and so S=7̄.

- Atrocious way of finding square roots!

Lessons

- No substitute for algorithm development
- Need to know how axioms can be used
- Should consider failure behavior