

CS 405/505 ASSIGNMENT #7

Due Wednesday, March 28, 2007

- A Sudoku (<http://www.sudoku.com>) puzzle is solved on a 9×9 grid, which may be subdivided as a 3×3 grid of 3×3 grids. The object of the puzzle is to fill in the grid so that every row, every column, and every 3×3 box contains the digits 1 through 9. Usually some elements of the grid are already filled in when the puzzle begins. An example of such a puzzle is:

5	2	7	6	9	8	1	4	3
6	9	1	7	4	3	5	2	8
8	4	3	2	1	5	6	7	9
2	6	8	5	3	9	7	1	4
9	7	5	4	6	1	8	3	2
1	3	4	8	2	7	9	6	5
3	5	6	1	8	2	4	9	7
4	8	9	3	7	6	2	5	1
7	1	2	9	5	4	3	8	6

Note that the digits in bold face are the ones already given.

This problem is to write a set of Prolog predicates which solve a Sudoku puzzle.

- Write a Prolog predicate `puzzle` (`P`) to instantiate `P` to the puzzle above as a list of 9 rows of entries, each row being a list of 9 entries. Use `[]` to indicate the entries which are not set initially (i.e., those that are not bold).
- Write a Prolog predicate `row` (`P`, `N`, `R`) which takes a puzzle `P` and extracts the `N`-th row, with `N = 1` being the first row, into `R`. If `P` does not have an `N`-th row, then the predicate fails. Test your predicate on the following, assuming `P` is the example puzzle given above:


```
row(P, 2, R) => R = [ [], [], 1, 7, [], 3, 5, [], [] ]
row(P, 9, R) => R = [ [], 1, [], [], [], [], [], 8, [] ]
row(P, 0, R) => no
```
- Write a Prolog predicate `column` (`P`, `N`, `C`) which takes a puzzle `P` and extracts the `N`-th column, with `N = 1` being the first column, into `C`. If `P` does not have an `N`-th column, then the predicate fails. Test your predicate on the following, assuming `P` is the example puzzle given above:


```
column(P, 2, C) => C = [ 2, [], [], [], 7, [], [], [], 1 ]
column(P, 9, C) => C = [ [], [], 9, [], [], [], 7, [], [] ]
column(P, 0, C) => no
```
- Write a Prolog predicate `box` (`P`, `I`, `J`, `B`) which takes a puzzle `P` and extracts the box `B` at coordinates `I` and `J`, with `(I, J) = (1, 1)` being the first box. If `P` does not have the indicated box, then the predicate fails. Test your predicate on the following, assuming `P` is the example puzzle given above:


```
box(P, 1, 2, B) => B = [ [], [], [], 7, [], 3, [], 1, [] ]
box(P, 3, 3, B) => B = [ 4, [], 7, 2, [], [], [], 8, [] ]
box(P, 2, 4, B) => no
```

- (e) Write a Prolog predicate `elements (L, E)` which takes a row, column or box L and returns the values of the fixed (i.e., a specific integer rather than a list of possibilities, including []) row, column or box elements as a linear list. Test your predicate on the following:

```
elements([7, 1, 2, 9, 5, 4, 3, 8, 6], E) ⇒ E = [7, 1, 2, 9, 5, 4, 3, 8, 6]
elements([2, [], [], [], 7, [], [], [], 1], E) ⇒ E = [2, 7, 1]
elements([[2, 4, 6, 9], [2, 4, 6, 9], 1, 7, [2, 4, 6, 9], 3, 5, [2, 4, 6, 9], [2, 4, 6, 9]], E) ⇒ E = [1, 7, 3, 5]
```

- (f) Write a Prolog predicate `initialize (L1, L2)` which takes a row, column or box L1 and returns L1, with all null values set to the initial set of candidates (i.e., those digits not yet fixed), as L2. Test your predicate on the following:

```
initialize([7, 1, 2, 9, 5, 4, 3, 8, 6], L) ⇒
L = [7, 1, 2, 9, 5, 4, 3, 8, 6]
initialize([2, [], [], [], 7, [], [], [], 1], L) ⇒ L = [2, [3, 4, 5, 6, 8, 9], [3, 4, 5, 6, 8, 9], [3, 4, 5, 6, 8, 9], 7, [3, 4, 5, 6, 8, 9], [3, 4, 5, 6, 8, 9], [3, 4, 5, 6, 8, 9], 1]
initialize([], [], 1, 7, [], 3, 5, [], []) ⇒ L = [[2, 4, 6, 8, 9], [2, 4, 6, 8, 9], 1, 7, [2, 4, 6, 8, 9], 3, 5, [2, 4, 6, 8, 9], [2, 4, 6, 8, 9]]
```

- (g) Write a Prolog predicate `update (L1, L2)` which takes a row, column or box L1 and returns L1 with all fixed digits removed from the candidate sets, as L2. Test your predicate on the following:

```
update([7, 1, 2, 9, 5, 4, 3, 8, 6], L) ⇒ L = [7, 1, 2, 9, 5, 4, 3, 8, 6]
update([2, [3, 4, 5, 6, 8, 9], 4, [3, 4, 5, 6, 8, 9], 7, [3, 4, 5, 6, 8, 9], 5, [3, 4, 5, 6, 8, 9], 1], L) ⇒ L = [2, [3, 6, 8, 9], 4, [3, 6, 8, 9], 7, [3, 6, 8, 9], 5, [3, 6, 8, 9], 1]
update([[2, 4, 6, 8, 9], 9, 1, 7, 4, 3, 5, 2, [2, 4, 6, 8, 9]], L) ⇒
L = [[6, 8], 9, 1, 7, 4, 3, 5, 2, [6, 8]]
```

- (h) You will be provided with a predicate `solve (P, S)` which takes a puzzle P and returns the solution S. This predicate will call the predicates you have written above. Run `solve` to test the system.

2. Consider the Core program segment S below:

```
a := x; b := y;
while (a != b) loop
  if (a < b) then
    b := b - a;
  else
    a := a - b;
  end if;
end loop;
z := a;
```

Using the axiomatic semantics of Core, show that the assertion

$$\{x > 0 \wedge y > 0\} S \{z = \text{gcd}(x, y)\}$$

is correct, assuming the following mathematical properties of gcd.

$$\text{gcd}(x, x) = x$$

$$\text{gcd}(x, y) = \text{gcd}(y, x)$$

$$\text{gcd}(x, y) = \text{gcd}(x + y, y)$$

Hint: Use the loop invariant $\{a > 0 \wedge b > 0 \wedge \text{gcd}(a, b) = \text{gcd}(x, y)\}$. The consequence rules may be used to force the other rules into a form consistent with this loop invariant. (You should analyze the program to understand why this loop invariant was chosen.)