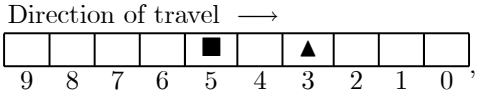
Assignment 2 — 2020

Theodore S. Norvell

5892 Due 14:00 Feb 28, 2020

Q0 [12] In the ancient board game of Yegimeli Zeri, players win by moving their token to square 0. On each turn, they can move forward by 1, 2, or 5 spaces. For example, if the state is this



Alice, represented by ▲, can finish by making any of the following sequence of moves

$$s(3) = \{[1,1,1]\,, [1,2], [2,1]\}$$

The White Knight, represented by the ■, can finish by making any of the following sequences of moves

$$\begin{aligned} s(5) \quad = \quad & \{[1,1,1,1,1]\,, [1,1,1,2]\,, [1,1,2,1]\,, [1,2,1,1]\,, [2,1,1,1]\,, \\ & [1,2,2]\,, [2,1,2], [2,2,1], [5]\} \end{aligned}$$

For each natural number i, let $s(i)$ be the set of all sequences of moves that a player on square i can make to finish the game: $s(3)$ is Alice's set of moves, $s(5)$ is the White Knight's sequence of moves, and $s(0) = \{[\,]\}$.

Function s has the following properties[0]

$$\begin{aligned} s(0) \quad &= \quad \{[\,]\} \\ s(i) \quad &= \quad \{[1]\hat{}w \mid w \in s(i-1)\}\,, \text{ for all } i = 1 \\ s(i) \quad &= \quad \{[1]\hat{}w \mid w \in s(i-1)\} \cup \{[2]\hat{}w \mid w \in s(i-2)\}\,, \text{ for all } i \in \{2,3,4\} \\ s(i) \quad &= \quad \{[1]\hat{}w \mid w \in s(i-1)\} \cup \{[2]\hat{}w \mid w \in s(i-2)\} \cup \{[5]\hat{}w \mid w \in s(i-5)\}\,, \text{ for all i} \in \{5,..\infty\} \end{aligned}$$

(a) [2] Write a contract for a procedure that takes a natural number i and returns the size of set $s(i)$.

[0] These properties fully define s, i.e., there is only one function from $\mathbb{N}$ that satsifies all these properties. If we are willing to extend the definition of s to negative numbers, we can also define s with just 3 equations

$$\begin{aligned} s(i) \quad &= \quad \emptyset, \text{ for all } i < 0 \\ s(0) \quad &= \quad \{[\,]\} \\ s(i) \quad &= \quad \{[1]\hat{}w \mid w \in s(i-1)\} \cup \{[2]\hat{}w \mid w \in s(i-2)\} \cup \{[5]\hat{}w \mid w \in s(i-5)\}\,, \text{ for all } i > 0 \end{aligned}$$

(b) [5] Write the pseudocode for the procedure. Give contracts and pseudocode for any additional procedures you need. Don't worry about efficiency. A recursive solution that takes exponential time is fine — in fact, expected.

(c) [5] Implement your pseudocode in the programming language of your choice. Test your code on some examples. Submit code and tests via D2L.

Q1 [12] Building on question 0:

(a) [2] Write a contract for a procedure that takes a natural number i and returns the set $s(i)$.

(b) [5] Write the pseudocode for the procedure. Give contracts and pseudocode for any additional procedures you need. Don't worry about efficiency. A recursive solution that takes exponential time is fine — in fact, expected.

(c) [5] Implement your pseudocode in the programming language of your choice. Test your code on some examples. Submit code and tests via D2L.

Here are some suggestions for languages

- Python 3: Python has a set type called 'frozen set' and a sequence type called 'tuple'. I'd suggest you use these types if you use Python.
- Java: Java has an interface 'java.util.Set' implemented by several types (for example HashSet); and it has an interface 'java.util.List' representing sequences, implemented by several classes. Java's List type lacks a concatenation operator, but you can concatenate two lists with the following code

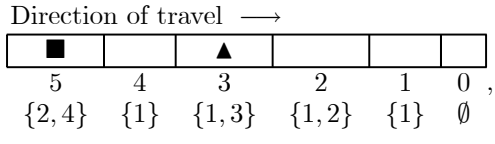
  ```
  ArrayList<Integer> z = new ArrayList<>() ;
  z.addAll(x) ;
  z.addAll(y) ;
  ```

  Similarly there is no union operator for sets, but you can create the union of two sets in a similar way.
- Dafny: In Dafny both sets and sequences are built-in to the language; for example

  ```
  s : set<seq<int>>
  ```

  declares s to represent a set of sequences of integers. Dafny overloads the + operator on sequences to represent concatenation and on sets to mean union. Dafny also supports set comprehension expressions.

Q2 [12]. Generalize the problem like this. Instead of the set $\{1, 2, 5\}$ suppose the set of possible moves on square i is a set $x(i)$; for each i, $x(i)$ is a finite set of positive integers. Define function $s(i, x)$ to be the set of all sequences that of moves that can be used by a player on square i to reach square 0. For example if the game state is



and x is

$$x = [\emptyset, \{1\}, \{1, 2\}, \{1, 3\}, \{1\}, \{2, 4\}]$$

Alice represented by ▲ can finish by making any of the following sequence of moves

$$s(3, x) = \{[1, 1, 1], [1, 2], [3]\}$$

and the White Knight ■ can finish by making any of the following sequences

$$s(5, x) = \{[2, 1, 1, 1], [2, 1, 2], [2, 3], [4, 1]\}$$

(a) [2] Write a contract for a procedure that takes a natural number i and a sequence of sets x and returns $s(i, x)$.

(b) [5] Write the pseudocode for the procedure. Give contracts and pseudocode for any additional procedures you need. Don't worry about efficiency. A recursive solution that takes exponential time is fine — in fact, expected.

(c) [5] Implement your pseudocode in the programming language of your choice. Test your code on some examples. Submit code and tests via D2L.

Bonus [5]: Can you use a bottom-up nonrecursive approach to make equivalent procedures that are significantly more efficient? See my notes 8.5.