Problem Wk.13.1.1: Farmer et al.: Machine

Read Section 8.2 of the class notes. See also Section 8.3 of the class notes for an example.

One version of a standard puzzle called the *Farmer, Goat, Wolf, Cabbage* goes as follows:

- The farmer has a goat, a wolf and a head of cabbage (don't ask why a wolf...).
- They come to a river (they're on left bank) and need to get to the other side (the right bank).
- There's a boat there that fits at most two of them (it's a big head of cabbage); the farmer must always be one of the two in the boat.
- If the farmer leaves the goat and cabbage on the same side of the river, when he is not present the goat will eat the cabbage (so that's not a legal state).
- Similarly, if the farmer leaves the goat and the wolf on the same side of the river, when he is not present... well, that's not legal either.

So, the problem is to find a sequence of actions that go from the initial state where they are all on the left bank to the final state when they are all on the right bank.

We will implement this domain by defining a class which is a subclass of the `SM` class. We will then use `search.smSearch` to find sequences of actions.

The state of the state machine needs to keep track of which bank of the river everyone is on. So, a state will be:

```
(farmerLoc, goatLoc, wolfLoc, cabbageLoc)
```

where each of these locations is `'L'` or `'R'` (for left or right). The boat is always with the farmer so no need to keep track of that.

Assume that the actions are:

- `'takeNone'`,
- `'takeGoat'`,
- `'takeWolf'`, and
- `'takeCabbage'`.

Each action indicates a trip from whatever side of the river the farmer is on, to the opposite side of the river.

In this problem. we will consider how the state machine is supposed to operate. Assume we create the state machine:

```
sm = FarmerGoatWolfCabbage()
```

The output of the state machine should be the same as the next state, so `getNextValues` returns `(nextState, nextState)`. Recall that `sm.transduce` returns the list of outputs of a state machine for a given list of inputs.

Write the response to the following commands. For each command below, assume that the machine starts out in its initial state `('L', 'L', 'L', 'L')`. Recall that if an action is impossible or would lead to an illegal state, then the machine state remains unchanged. Subsequent actions are executed normally.

You should enter a list of states (4-tuples), that is, they should look like:

```
[('L', 'R', 'L', 'L'), ('L', 'L', 'R', 'L')]
```

1. From the initial state (`('L', 'L', 'L', 'L')`), what should be the output of `sm.transduce(['takeGoat'])` (enter a list of 4-tuples)?
2. From the initial state (`('L', 'L', 'L', 'L')`), what should be the output of `sm.transduce(['takeNone', 'takeGoat'])` (enter a list of 4-tuples)?
3. From the initial state (`('L', 'L', 'L', 'L')`), what should be the output of `sm.transduce(['takeGoat', 'takeNone', 'takeNone'])` (enter a list of 4-tuples)?

6.01SC Introduction to Electrical Engineering and Computer Science

Spring 2011