Assignment2: Search

– Given Nov. 3., Due Nov. 13. –

Problem 2.1 20 pt

Explain the difference between agent function and agent program. How many agent programs can there be for a given agent function?

Solution: The function specifies the input-output relation (outside view). The program implements the function (inside view).

The function takes the full sequence of percepts as arguments. The program uses the internal state to avoid that.

There are either none or infinitely many programs for a function.

Problem 2.2 20 pt

Explain the commonalities of and the differences between the performance measure and the utility function.

Solution: Both measure how well an agent is doing.

The performance measure is a meta-level object that defines the quality of any agent used to solve the task. It may be defined informally (but still precisely) because it only needs to be used by an outside observer, such as a human comparing multiple agents.

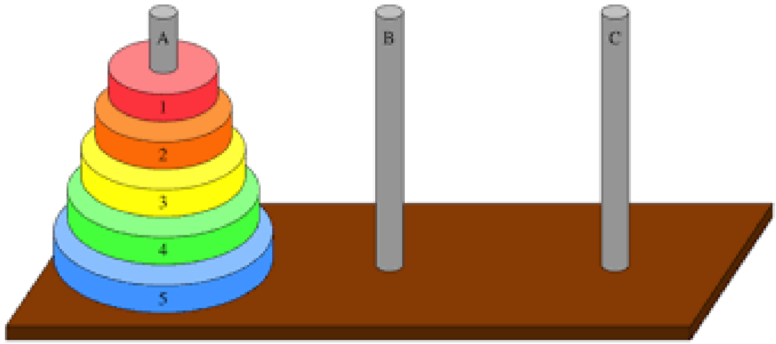
A utility function is a component of a particular utility-based agent. It must be defined formally (e.g., in a specification or programming language) because it must be computed as a part of applying the agent.

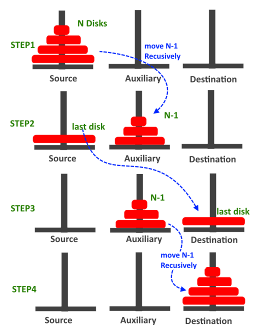
Problem 2.3 (Towers of Hanoi) 50 pt

The Towers of Hanoi is a mathematical puzzle. It consists of three pegs (A, B, and C) and a number of disks of different sizes, which can slide onto any peg. The puzzle starts with the disks in a stack in ascending order of size on one peg, the smallest at the top, thus making a conical shape. The objective of the puzzle is to move all disks from peg A to peg B, while obeying the following rules:

1. only one disk can be moved at a time,
2. each move consists of taking the upper disk from one of the stacks and placing it on top of another stack or on an empty peg,
3. no larger disk may be placed on top of a smaller disk.

The idea of the algorithm (for $N > 1$) is to move the top $N-1$ disks onto the auxiliary peg, then move the bottom disk to the destination peg, and finally moving the remaining $N-1$ disks from the auxiliary peg to the destination peg.





1. Write a `ProLog` predicate that prints out a solution for the Towers of Hanoi puzzle. Use the `write(X)` predicate that prints the value of X (X can be simple text or any type of argument) to the screen and `nl` that prints a new line to write a rule `move(N, A, B, C)` that prints out the solution for moving N disks from peg A to peg B, using C as the auxiliary peg.

   Each step of the solution should be of the form "Move top disk from X to Y".

   Examples:

   ```
   ?- write(hello), write('␣world!'), nl.
   hello world!
   true.
   ```

   ```
   ?- move(3, left, center, right).
   Move top disk from left to center
   Move top disk from left to right
   Move top disk from center to right
   Move top disk from left to center
   Move top disk from right to left
   Move top disk from right to center
   Move top disk from left to center
   true ;
   false.
   ```

2. Determine the complexity class of your algorithm in terms of the number of disks N and explain how you computed it.

Solution:

1.
```
move(1, A, B, _) :-
    write('Move␣top␣disk␣from␣'),
    write(A), write('␣to␣'), write(B), nl.
move(N, A, B, C) :-
    N>1, M is N-1,
    move(M, A, C, B), move(1, A, B, _), move(M, C, B, A).
```

2. Let $T(N)$ be the number of moves needed to move N disks from one peg to another. Clearly, $T(1) = 1$. For $T(N)$, we have the following recursive relation:
$$T(N) = 2T(N-1) + 1$$
The values for $N = 1, 2, 3, 4, 5$ are 1, $2+1$, 2^2+2+1, 2^3+2^2+2+1, and $2^4+2^3+2^2+2+1$. Thus, $O(T(n)) = 2^{n-1}$, which is exponential.

(You could also solve the non-homogenous linear recurrence to obtain a precise closed formula for $T(N)$.)

Problem 2.4 (Mathematical Notation) 10 pt

Let $\mathbb{N}$ be the set of natural numbers. A monoid is a mathematical structure $\langle U, \circ, e\rangle$ where U is a set, $\circ$ is an associative binary function on U, and e is the neutral element of $\circ$.

Express the following concepts in mathematical notation:

1. the set containing all natural numbers
2. the set containing the set of natural numbers
3. the set containing all square numbers
4. the set containing all even natural numbers
5. the set containing all even square numbers
6. the 3-tuple of 0, 1, and 2
7. the n-tuple of all numbers from 0 to $n-1$
8. the set of pairs of natural numbers and their squares
9. the pair of sets of natural numbers and square numbers
10. the monoid of natural numbers under addition
11. the pair of monoids of the natural numbers under addition and under multiplication

12. the set of the monoids of the natural numbers under addition and under multiplication
13. given a monoid $\langle U, \circ, e\rangle$, the set of elements that are not the neutral element
14. given a monoid $\langle U, \circ, e\rangle$, the monoid in which the operation is the same but with left and right argument switched.

Solution: Note: The notation $\langle a_1, \ldots, a_n\rangle$ for a tuple is specific to the AI lecture. You can also write the tuple as $(a_1, \ldots, a_n)$.

1. $\mathbb{N}$
2. $\{\mathbb{N}\}$
3. $Squares := \{n \in \mathbb{N} | \exists m \in \mathbb{N}. n = m^2\}$ (selecting a subset by a property) or $\{n^2 : n \in \mathbb{N}\}$ (generating a set by applying a function to all elements of a set)
4. $Evens := \{n \in \mathbb{N} | \exists m \in \mathbb{N}. n = 2m\}$ or $\{2n : n \in \mathbb{N}\}$
5. $Squares \cap Evens$
6. $(0, 1, 2)$
7. $(0, \ldots, n-1)$
8. $\{(n, n^2) : n \in \mathbb{N}\}$
9. $(\mathbb{N}, Squares)$
10. $NatAdd := (\mathbb{N}, +, 0)$
11. Let $NatMult := (\mathbb{N}, \cdot, 1)$. Then $(NatAdd, NatMult)$.
12. $\{NatAdd, NatMult\}$
13. $\{u \in U | u \neq e\}$
14. (U, o, e) where o is the function $(x, y) \mapsto y \circ x$
