Linköpings universitet
Institutionen för Datavetenskap
Erik Sandewall

Exam
TDDA23 – Artificial Intelligence and Lisp
14-18, April 21, 2006-04-21

Points: The exam consists of exercises worth 22 points. To pass the exam you need 10 points.

Auxiliary items: English dictionary.

Directions: You can answer in English or Swedish. Use notions and methods that have been discussed in the course. If an exercise has not been specified completely as you see it, state which (reasonable) assumptions you have made. Begin each exercise on a new sheet of paper. Write only on one side of the paper, write clearly and make sure to give adequate explanations (motivera!) for all your answers.

Jour: Erik Sandewall, 0708-281408

1.

a) Define a Lisp function firstlast(x) that takes a list x as its argument and that returns t if the first element of the list is equal to the last element, and nil otherwise. If the argument is not a list or if it is the empty list then nil shall also be returned. (1 p)

b) Define a Lisp function firstlast2(x) that returns t if the argument x is a list of at least two elements, the first element equals the last one, and the second element equals the one before the last one. In all other cases it shall return nil. (2 p)

The solutions must be written as recursively defined functions without side-effects.

2.

The following is an example of a definition of a recursive function with tail recursion:

```
(defun last (x)
        (if (null (cdr x)) (car x) (last (cdr x))) )
```

The following definition is not tail recursive:

```
(defun length (x)
        (if (null x) 0 (+ 1 (length (cdr x)))) )
```

In general, a function is tail recursive if all calls to itself occur on the top level of a 'then' or 'else' part of a conditional expression. The definition of length is not tail recursive since the recursive call to itself occurs inside the (+ 1 ...) expression, which means that it is not on the top level of the 'else' subexpression.

(For the present purpose we make the restriction that there is only one function being defined. If there are several recursive functions that call each other back and forth then the definition is a bit more complicated).

Problem: define a lisp function tailrec(def) where the argument def is a function definition, and where the value is t if the given definition is tail recursive, and nil otherwise. The argument def shall always be a list beginning with the symbol defun. It is assumed that conditional definitions are always written using the 'if' operator with two or three arguments, and that they can be nested inside each other. Other conditional operators such as 'cond' are not used in the definition being analyzed. (3 p)

3.

The formal definition of a classical planning problem is to find a sequence of actions A1, A2, ... An such that

G <A1, A2,... An>[S0]

a) What is the structure of S0, of G, and of the actions Ai in this definition? (The answers can be for example "it is a complete state" or "it is a pair of a complete state and a partial state"). (2 p)

b) Explain the notation <A1, A2,... An>[S0] using the answer to the a) part of this problem. (2 p)

4.
The preference relation called 'chronological minimization of change' specifies a condition for when one robotic history is to be preferred over another one, for the purpose of identifying a set of most preferred histories. What is the precise definition of this condition? (3 p)

5.
Formal logic uses two operators that are usually written as |- and as |=.

a) Explain the meaning of each of these. (2 p)

b) How are these two operators related? (1 p)

6.
In a particular application using the situation calculus there is one action emptybottle(s,b) with the effect of emptying a particular bottle b of its contents, and another action fillbottle(s,b) with the effect of filling the bottle b with wine. What is the relation between the situation emptybottle(fillbottle(s,b),b) and the situation s ? (2 p)

7.
a) Explain what is "genetic" about genetic algorithms. (2 p)

b) Is it possible and meaningful to run a genetic algorithm where each individual is mated with itself, instead of with another individual? (1 p)

c) Is it possible and meaningful to run a genetic algorithm where each couple is formed by taking a parent and one of its 'children'? (1 p)

For questions b) and c) the reasons for the answer must be explained; just yes or no is not sufficient. A good explanation of the consequences of the arrangement in question gives one extra point.