7 Prolog (ACR)

This question explores how we might use Prolog to match Regular Expressions.

We represent the sequence to be matched in Prolog using a list of atoms. For example, `aaba` would be represented as the list `[a,a,b,a,end]` using the atom `end` to encode the end of the string explicitly.

A simple scheme for writing Regular Expressions uses a single character as itself and uses the plus symbol (+) to indicate that there should be one or more instances of the previous character. In this question we consider the Regular Expression $\texttt{a}^+\texttt{b}^+\texttt{a}$ which means one or more occurrences of `a`, followed by one or more occurrences of `b`, followed by a single occurrence of `a`.

(*a*) Draw a state machine which is capable of matching the Regular Expression $\texttt{a}^+\texttt{b}^+\texttt{a}$. Clearly indicate the start and finish states. [2 marks]

(*b*) Define a predicate `t(A,B,C)` which encodes the transitions of your state machine. `t(A,B,C)` should be true if there is a transition from state `A` to state `B` when we see a character `C`. Indicate which of your definitions are facts and which are rules. [2 marks]

(*c*) Predicates for *testing* a solution do not always work when *generating* solutions. Demonstrate this by writing a Prolog predicate `matches(L)` which *tests* if `L` represents a string which matches the Regular Expression $\texttt{a}^+\texttt{b}^+\texttt{a}$. [5 marks]

(*d*) Why is your predicate `matches(L)` not a good solution for *generating* strings matching the Regular Expression $\texttt{a}^+\texttt{b}^+\texttt{a}$? Describe a specific execution path in which a problem can occur. [3 marks]

(*e*) Describe a better strategy for *generating* strings matching the Regular Expression $\texttt{a}^+\texttt{b}^+\texttt{a}$ and provide an implementation. Clearly explain the approach you are using and why it is a sensible choice. [8 marks]