Inteligência Artificial — Aula 3

1. Defina um predicado `nOcorre(X,L,N)`, em que `N` é o número de ocorrências de `X` na lista `L`.

2. Defina um predicado `substitui(X,Y,L1,L2)`, tal que `L2` é a lista que resulta de substituir as ocorrências de `X` na lista `L1` por `Y`.

3. Considere um qualquer autómato finito determinístico do alfabeto $\Sigma = \{0,1\}$. Defina, por factos em Prolog, o estado inicial do autómato, os seus estados finais e o diagrama de transições (`estado_inicial/1`, `estado_final/1`, `transicao/3`).

4. Defina um predicado `sequencia/3` que dada uma lista de símbolos do alfabeto (palavra) e dois estados (`E1`, `E2`), determina se é possível transitar de E1 para E2 (no autómato) seguindo a lista de símbolos dada. Com base no predicado `sequencia/3` defina o predicado `aceita/1` que dada uma lista de símbolos do alfabeto (palavra), determina se o autómato reconhece a palavra.

5. Escreva um predicado `testa/1` que dado um átomo no argumento (`?- testa('0111011').`) diga se a palavra é reconhecida pelo autómato.

Sugestões: Use o *built-in* `name/2` que dado um átomo retorna uma lista com os códigos ASCII dos caracteres que compõem o átomo (e.g. `name('0111011',[48,49,49,49,48,49,49])`). Crie um predicado `sub/2` para obter uma lista da forma `[0,1,1,1,0,1,1]` dada uma lista da forma `[48,49,49,49,48,49,49]`.

6. Acrescente agora ao autómato uma transição a um estado pelo símbolo `0` onde já exista uma transição por `0` para outro estado. O autómato fica agora *não-determinístico*. Faça `?- spy sequencia/3` e siga passo a passo a execução do reconhecimento de palavras, observando como o *backtracking* permite implementar o não-determinísmo.

7. Acrescente agora transições ϵ ao autómato e modifique o predicado `sequencia/3` de forma a efectuar também estas transições.