Pregunta 1

Definir el concepto de autómata determinista.

Un autómata determinista es una estructura $M = (K, \Sigma, \delta, q_0, F)$ donde:

(1) K es un conjunto finito y no vacío de estados.
(2) Σ es el alfabeto de entrada.
(3) δ es una función de $K \times \Sigma$ en K, a la que se denomina función de transición.
(4) $q_0 \in K$ es el estado inicial.
(5) $F \subseteq K$ es el conjunto de estados aceptadores.

Pregunta 2

Definir los conceptos de configuración y cómputo en un autómata determinista.

Si $M = (K, \Sigma, \delta, q_0, F)$ es un autómata determinista, definimos una configuración de M como una palabra $px \in K\Sigma^*$.

Si px, qy son configuraciones de M, decimos que hay un cómputo de px a qy, si en M podemos pasar de px a qy aplicando un número finito de veces la función de transición δ.

Pregunta 3

Definir el concepto de lenguaje asociado a un autómata determinista.

Sea $M = (K, \Sigma, \delta, q_0, F)$ un autómata determinista.

Una palabra $x \in \Sigma^*$ es reconocida por M, si el cómputo de M para la palabra x termina en un estado aceptador.

Definimos el lenguaje reconocido por M por

$$L(M) = \{x \in \Sigma^* : x \text{ es reconocida por } M\}.$$

Pregunta 4

Explicar el algoritmo visto en clase para programar un autómata determinista.

Para obtener un programa en Java o en C correspondiente a un autómata determinista, podemos proceder de la siguiente manera:

(1) Representamos a los estados del autómata por números naturales, representando por el 0 al estado inicial.
(2) Representamos a los símbolos del alfabeto de entrada por caracteres.
(3) Representamos el cálculo del autómata mediante un bucle "while", en el que describimos mediante una instrucción "switch-case" las transiciones del autómata en las que cambia el estado.
(4) Al salir del bucle "while", comprobamos si el estado en el que estamos es un estado aceptador.

Pregunta 5

Explicar el interés de los autómatas deterministas.

Los autómatas deterministas tienen interés, porque se utilizan para diseñar el analizador léxico de un compilador. También, se pueden utilizar para diseñar procesadores de textos. Y, asimismo, se utilizan para diseñar programas eficientes de búsqueda de patrones en textos.

Pregunta 6

Explicar las diferencias existentes entre los autómatas deterministas y los indeterministas.

En los autómatas indeterministas, se permite que desde cada estado se realizen cero, una o más transiciones para cada símbolo de la entrada, mientras que en el caso determinista se realiza desde cada estado exactamente una transición para cada símbolo de la entrada.

Además, en los autómatas indeterministas se permiten transiciones con la palabra vacía, es decir se permite pasar de un estado a otro sin leer ningún símbolo de la entrada.

Los autómatas indeterministas son modelos computacionales equivalentes a los autómatas deterministas, pero más fáciles de diseñar, porque normalmente tienen menos estados y menos transiciones.

Pregunta 7

Explicar el algoritmo visto en clase para transformar un autómata indeterminista en un autómata determinista equivalente.

Sea $M = (K, \Sigma, \Delta, q_0, F)$ un autómata indeterminista. En primer lugar, para todo estado $p \in K$, definimos $\Lambda(p)$ como el conjunto de aquellos estados de K a los que podemos llegar desde el estado p sin leer ningún símbolo de la entrada.
Definimos entonces el autómata determinista M' equivalente a M como $M' = (K', \Sigma, \delta', q_0', F')$ donde:

Pregunta 7

Explicar el algoritmo visto en clase para transformar un autómata indeterminista en un autómata determinista equivalente.

(1) $K' = P(K) =$ conjunto de subconjuntos de K,
(2) $q_0' = \Lambda(q_0)$,
(3) $F' = \{X \in K' : X \cap F \neq \emptyset\}$,
(4) Si $X \in K'$ y $a \in \Sigma$, definimos

$$\delta'(X, a) = \bigcup \{\Lambda(q) : q \in K \text{ y existe un estado } p \in X \text{ tal que } (p, a, q) \in \Delta\}.$$

Pregunta 8

Explicar las diferentes maneras en que se puede definir el concepto de lenguaje regular

Se puede definir mediante el concepto de expresión regular, ya que un lenguaje L es regular, si existe una expresión regular α tal que $L(\alpha) = L$.

También, se puede definir mediante el concepto de autómata determinista, ya que por un teorema visto en clase, sabemos que el concepto de expresión regular es equivalente al concepto de autómata determinista.

Y asimismo, se puede definir mediante el concepto de autómata indeterminista, ya que por otro teorema visto en clase, sabemos que el concepto de autómata determinista es equivalente al concepto de autómata indeterminista.

Pregunta 9

Describir las categorías sintácticas de un lenguaje de programación y explicar su utilidad en el diseño de compiladores

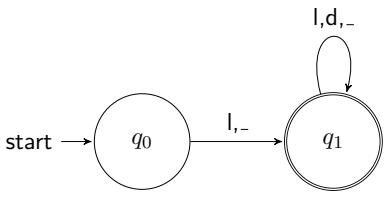
Las categorías sintácticas (o tokens) de un lenguaje de programación son la clase de los identificadores, cada tipo de datos del lenguaje de programación (tipo integer, tipo float, tipo char, etc), cada símbolo relacional, cada operador aritmético, cada símbolo de puntuación y cada palabra reservada.

Las categorías sintácticas se utilizan para simplificar la fase del compilador del análisis sintáctico, en la cual se determina si un programa escrito en un lenguaje de programación de alto nivel está correctamente escrito con respecto a las reglas que tenga dicho lenguaje de programación.

Pregunta 10

Definir autómatas indeterministas que reconozcan el conjunto de identificadores de C y el tipo integer

Para simplificar la construcción, denotamos por l a cualquier letra y por d a cualquier dígito. Tenemos el siguiente autómata para reconocer los identificadores de C:



Pregunta 10

Definir autómatas indeterministas que reconozcan el conjunto de identificadores de C y el tipo integer

Y tenemos el siguiente autómata para reconocer el tipo integer: