UNIVERSIDAD DE GRANADA

Departamento de Ciencias de la Computación
e Inteligencia Artificial

Modelos Avanzados de Computación

Práctica 2

Máquinas de Turing

Curso 2014-2015

Doble Grado en Ingeniería Informática y Matemáticas

Práctica 2

Máquinas de Turing

Un lenguaje L es recursivamente enumerable (r.e.) si y sólo si existe una máquina de Turing M tal que $L(M) = L$; esto es, existe una máquina de Turing que reconoce el lenguaje. Un lenguaje es recursivo (o, simplemente, decidible) si es aceptado por una MT que siempre para.

1. El origen de los términos "recursivamente enumerable" y "recursivo" proviene de una variante de la máquina de Turing denominada enumerador. Informalmente, un enumerador es una máquina de Turing con una impresora asociada, que puede utilizar como dispositivo de salida para imprimir cadenas de símbolos. Un enumerador E comienza con una entrada en blanco en su cinta de trabajo. Si no se detiene, podría imprimir una lista infinita de cadenas. El lenguaje enumerado por E es el conjunto de cadenas que eventualmente imprime. Un lenguaje es recursivamente enumerable si existe un enumerador que lo enumera (en cualquier orden y con posibles repeticiones). Como sucede con otros muchos modelos de cómputo, se puede demostrar la equivalencia de los enumeradores con las máquinas de Turing tradicionales. Hágalo.

   [Michael Sipser: "Introduction to the Theory of Computation", 3rd edition, pp. 180-182]

2. Discuta la posibilidad de asignar un número natural a cada MT con independencia del alfabeto de entrada.

3. Sean $L_1, \ldots, L_k$ un conjunto de lenguajes sobre el alfabeto A tales que:

   a) Para cada $i \neq j$, tenenos que $L_i \cap L_j = \emptyset$.

   b) $\bigcup_{i=1}^{k} L_i = A^*$.

   c) $\forall i \in \{1, \ldots, k\}$, el lenguaje L_i es r.e.

   Demostrar que $\forall i \in \{1, \ldots, k\}$, el lenguaje L_i es recursivo.

   [John Hopcroft's "Introduction to Automata Theory", 2nd edition, exercise 9.2.4]

4. Sea L r.e., pero no recursivo. Considérese el lenguaje

$$L' = \{0w \mid w \in L\} \cup \{1w \mid w \notin L\}$$

¿Puede asegurarse que L' o su complementario sean recursivos, r.e. o no r.e.?

[John Hopcroft's "Introduction to Automata Theory", 2nd edition, exercise 9.2.5]

5. Estudie si las clases de lenguajes recursivos y recursivamente enumerables son cerradas para las siguientes operaciones:

   a) Unión.
   b) Intersección.
   c) Concatenación.
   d) Clausura.
   e) Homomorfismo.
   f) Homomorfismo inverso.

[John Hopcroft's "Introduction to Automata Theory", 2nd edition, exercise 9.2.6]

6. El problema de la parada: Demuestre que el lenguaje asociado al conjunto de parejas (M, w) tales que la MT M para cuando tiene a w como entrada es recursivamente enumerable pero no recursivo.

[John Hopcroft's "Introduction to Automata Theory", 2nd edition, exercise 9.2.1]

7. Determine si los siguientes lenguajes son recursivos, r.e. o no r.e.:

   a) Determinar si el lenguaje de una MT contiene, al menos, dos palabras distintas.
   b) Determinar si el lenguaje de una MT es finito o infinito.
   c) Determinar si el lenguaje de una MT es independiente del contexto.
   d) Determinar si $L(M) = (L(M))^R$.
   e) Determinar si una MT con dos o más estados entrará alguna vez en un estado q.
   f) Determinar si una MT termina escribiendo un 1 cuando comienza con una cinta completamente en blanco.
   g) El conjunto de las MT que aceptan al menos un palíndromo.
   h) El conjunto de las MT que se paran para cualquier entrada.
   i) El conjunto de las MT que no se paran para ninguna entrada.

j) El conjunto de las MT que se paran, al menos, para una entrada.

k) El conjunto de las MT que no se paran, al menos, para una entrada.

l) El lenguaje L formado por pares de códigos de MT más un entero (M_1, M_2, k) tales que $L(M_1) \cap L(M_2)$ contiene, al menos, k palabras.

[John Hopcroft's "Introduction to Automata Theory", 2nd edition, exercise 9.3.x]

8. Demuestre que las siguientes cuestiones son decidibles:

   a) El conjunto de las MT M tales que, al comenzar con la cinta en blanco, en algún momento escribirán un símbolo no blanco en la cinta.

   b) El conjunto de las MT que nunca se mueven a la izquierda.

   c) El conjunto de los pares (M, w) tales que M, al actuar sobre la entrada w, nunca lee una casilla de la cinta más de una vez.

   [John Hopcroft's "Introduction to Automata Theory", 2nd edition, exercise 9.3.6]

9. **El castor atareado [busy beaver]**: Imagine una máquina de Turing que comienza con una cinta completamente en blanco y, eventualmente, para. Si la máquina deja n unos en la cinta cuando para, diremos que la productividad de la máquina es n. También diremos que la productividad de una máquina de Turing que no para es 0. La productividad, por tanto, es una función de números naturales (códigos de máquinas de Turing) a números naturales. Escribiremos p(T)=n para indicar que la productividad de la máquina T es n.

   De entre las máquinas de Turing con un número concreto de estados, existe una productividad máquina que una máquina de Turing con ese número de estados puede tener. Esto nos permite definir otra función de números naturales (el número de estados) a números naturales (la productividad máxima de una máquina con ese número de estados). Indicaremos mediante BB(k)=n el hecho de que la productividad máxima de una máquina de Turing con k estados es n. Pueden existir múltiples máquinas de Turing con k estados que alcancen dicha productividad, a las que llamaremos "castores atareados" [busy beaver].

   Demuestre que no existe ninguna máquina de Turing que sea capaz de calcular la función BB(k); esto es, que comenzando con una cinta con k 1's se detenga cuando en la cinta queden BB(k) 1's.

   NOTA: La función BB(k) crece más rápido que cualquier función computable.

   Tibor Rado: "On non-computable functions". Bell System Technical Journal 41(3):877-884, May 1962.

   DOI 10.1002/j.1538-7305.1962.tb00480.x.

Ejercicios complementarios

A continuación se incluyen algunos ejercicios adicionales que puede realizar si desea mejorar su destreza a la hora de trabajar con máquinas de Turing:

1. Utilice algún simulador de máquinas de Turing disponible en Internet para diseñar máquinas de Turing que reconozcan los siguientes lenguajes:

   - Palabras sobre el alfabeto $\{0, 1\}$ con el mismo número de ceros que de unos.
   - $L = \{a^n b^n c^n | n \geq 1\}$
   - $L = \{ww^{-1} \,|\, w \in \{0, 1\}^*\}$
   - $L = \{wcw \,|\, w \in \{0, 1\}^*\}$

   Puede utilizar un simulador escrito en JavaScript como el disponible en `http://morphett.info/turing/turing.html`, que puede utilizar directamente en su navegador web, o descargar un simulador como JFLAP, implementado en Java y disponible en `http://jflap.org/`.

   [John Hopcroft's "Introduction to Automata Theory", 2nd edition, exercise 8.2.2]

2. Construya una MT que, dada una entrada w, la convierta en una salida $w111w$.

3. Diseñe una máquina de Turing que haga algo 'útil' distinto a los ejemplos vistos en clase (y a los programas de ejemplo del simulador).

4. Diseñe una subrutina que desplace a la derecha todos los símbolos de la cinta desde la posición actual, dejando un espacio en dicha posición (para que se pueda insertar un nuevo símbolo en ella). La subrutina debe terminar con el cabezal de lectura en la misma posición en la que empezó.

   Pista: Utilice un símbolo especial para marcar la posición a la que debe volver el cabezal.

   [John Hopcroft's "Introduction to Automata Theory", 2nd edition, exercise 8.3.2]

5. Diseñe una MT con dos cintas que, dada una secuencia de unos de longitud n en la primera cinta, calcule en la segunda cinta el valor de n en binario.

6. Considere la MTND $M = (\{q_0, q_1, q_2, q_f\}, \{0, 1\}, \{0, 1, \#\}, \delta, q_0, \#, \{q_f\})$ con

   $$\begin{array}{ll} \delta(q_0, 0) = \{(q_0, 1, D), (q_1, 1, D)\} & \delta(q_1, 1) = \{(q_2, 0, I)\} \\ \delta(q_2, 1) = \{(q_0, 1, D)\} & \delta(q_1, \#) = \{(q_f, \#, D)\} \end{array}$$

   Describa el lenguaje generado por esta máquina de Turing no determinista.

   [John Hopcroft's "Introduction to Automata Theory", 2nd edition, exercise 8.4.4]