TAC - Unidad Didáctica 04

Modelos de Computación (I)

Máquinas de Turing

1

Bibliografía

Sipser – Introduction to the Theory of Computation – Chapter 3.

Ampliado en :

Alfonseca – Teoría de Lenguajes, Gramáticas y Autómatas, 2007 - Capítulo2.

y

Hopcroft, Motwani, Ullman – Introducción a la Teoría de Autómatas, Lenguajes, y Computación – Capítulo 8.

2

01 ANÁLISIS DE ALGORITMOS / THE RUNNING TIME OF PROGRAMS

02 TEORÍA DE LA COMPUTABILIDAD

03 TEORÍA DE LA COMPLEJIDAD COMPUTACIONAL.

04 MODELOS DE COMPUTACIÓN.

- Maquinas de Turing

3

MÁQUINAS DE TURING I

MODELO DE COMPUTACIÓN ELEMENTAL
MUNDO DE LOS BLOQUES

4

Origen

Formulada por Turing como un medio para formalizar el proceso sistemático de resolver problemas de un matemático (p. ej.)

En sentido más amplio:

Para representar cualquier Algoritmo

Por precaución hablamos de Procedimientos Efectivos

Sirven para resolver problemas

Actualmente es una máquina evolucionada en base a contribuciones de Post, Turing y otros.

5

Utilidad

Turing se sirvió de ella para desarrollar diversas cuestiones:

- *Máquina de Turing Universal*
- *Problema de la Parada*

En equivalencia al Entscheidungsproblem de Gödel

- Modelo Universal de Computación

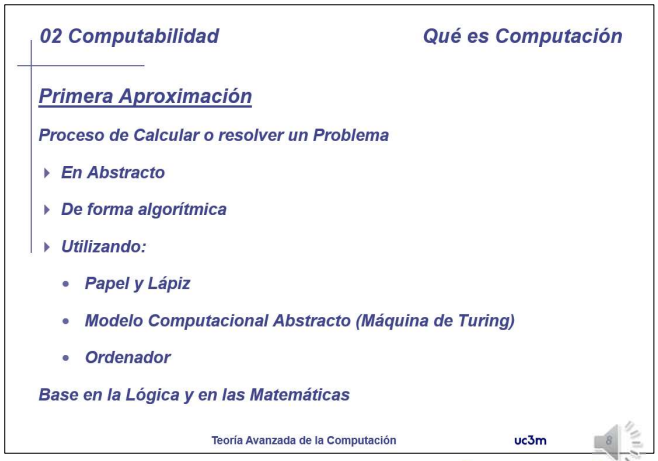
6

Modelo de Computación Universal

Ya planteamos las cuestiones de:

¿Qué es Computación?

¿Qué es Computable?



7

Modelo de Computación Universal

Podemos plantear:

¿Qué se puede computar con una Máquina de Turing?

?

??

???

¿Cualquier cosa computable?

(ciclo)

8

Modelo de Computación Universal

¿Qué se puede computar con una Máquina de Turing?

Muchos Modelos de Computación se basan en operaciones muy elementales

A partir de ellas se pueden construir procesamientos complejos.

➔ *Empecemos por lo básico*

9

Modelo de Computación Universal

Aritmética de Peano

Conjunto axiomático que sirve para definir los Números Naturales

R0 – Objeto Inicial	*0 es un objeto especial y es un Número Natural*
R1 – Sucesor	*para cada n Número Natural existe otro succ(n)*
R2 – Predecesor	*si a es succ(b) entonces b es pred(a)*
R3 – Única	*no hay dos NN diferentes con el mismo sucesor*
R4 – Igualdad	*comparar a y b NN en cuanto a igualdad*
	➔ Reflexiva aEa , simétrica aEb bEa, transitiva aEb bEc ➔ aEc
R5 – Inducción	*pred. P(n) es cierto ∀n si cumple condiciones*

10

Modelo de Computación Universal

Aritmética de Peano

Conjunto axiomático que sirve para definir los Números Naturales

R0 – Objeto Inicial 0 es un objeto especial y es un Número Natural

11

Modelo de Computación Universal

Aritmética de Peano

Conjunto axiomático que sirve para definir los Números Naturales

R1 – Sucesor para cada n Número Natural existe otro NN succ(n)

12

Modelo de Computación Universal

Aritmética de Peano

Conjunto axiomático que sirve para definir los Números Naturales

R2 – Predecesor si a es succ(b) entonces b es pred(a)

13

Modelo de Computación Universal

Aritmética de Peano

Conjunto axiomático que sirve para definir los Números Naturales

R3 – Única no hay dos NN diferentes con el mismo sucesor

14

Modelo de Computación Universal

Aritmética de Peano

Conjunto axiomático que sirve para definir los Números Naturales

R4 – Igualdad comparar a y b NN en cuanto a igualdad

- ➔ *reflexiva Eq(a,a)*
- ➔ *simétrica Eq(a,b) ↔ Eq(b,a)*
- ➔ *transitiva Eq(a,b), Eq(b,c) ➔ Eq(a,c)*

15

Modelo de Computación Universal

Aritmética de Peano

Conjunto axiomático que sirve para definir los Números Naturales

R5 – Inducción predicado P(n) es cierto ∀ n si cumple:

- *P(0) cierta*
- *P(n) cierta para algún n y se puede demostrar que P(succ(n)) es cierta*

16

Modelo de Computación Universal

Operaciones Elementales con una Máquina de Turing

Por ejemplo

succ(k) *(Aritmética de Peano)*

succ(k) → k+1 *k, k+1 números Naturales*

17

Modelo de Computación Universal

Operaciones Elementales con una Máquina de Turing

succ(k) *(Aritmética de Peano)*

Suposiciones previas:

- *MT de cinta infinita,*
- *con Alfabeto = { b, 1 }* *b o # representan celda en blanco*
- *Alfabeto auxiliar con símbolos de marcado { $, X, etc. }*
- *Se representan Números Naturales en base 1*

  1, 11, 111, 1111, 11111, …. ,

18

Modelo de Computación Universal

Operaciones Elementales con una Máquina de Turing

succ(k) (Aritmética de Peano)

succ(k) → k+1

1 1 1 # # #

1 1 1 1 # #

19

Modelo de Computación Universal

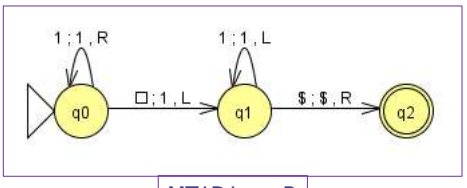
Operaciones Elementales con una Máquina de Turing

succ(k) (Aritmética de Peano)

succ(k) → k+1

1 1 1 # # #

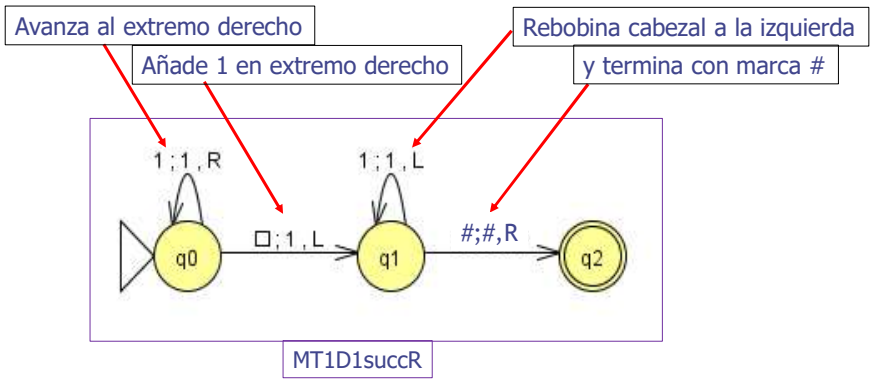
1 1 1 1 # #



MT1D1succR

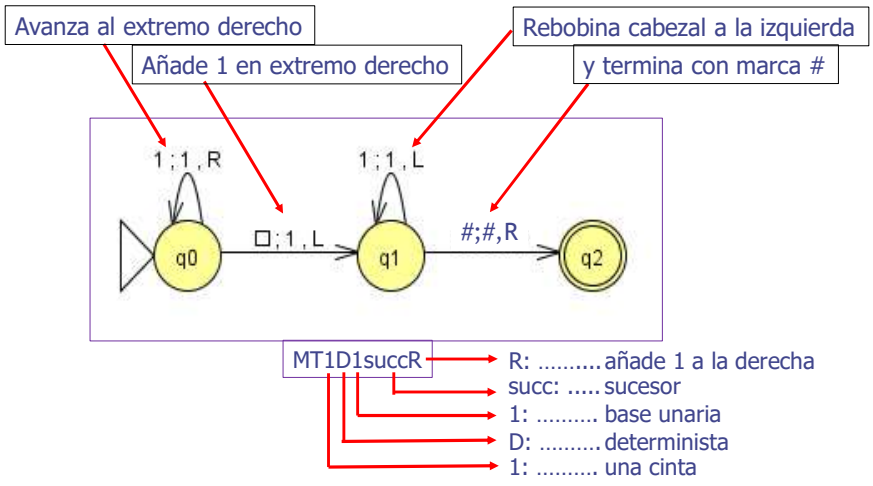
20

Modelo de Computación Universal



21

Modelo de Computación Universal



22

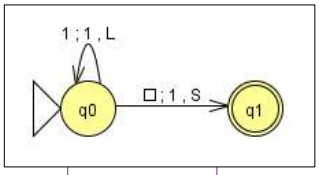
Modelo de Computación Universal

Operaciones Elementales con una Máquina de Turing

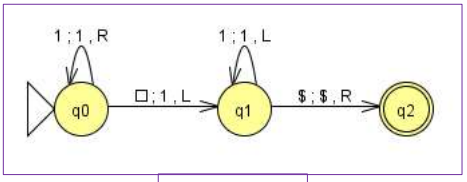
succ(k) (Aritmética de Peano)

succ(k) → k+1

Hay una versión L:



MT1D1succL



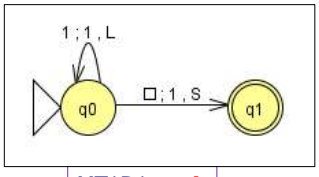
MT1D1succR

23

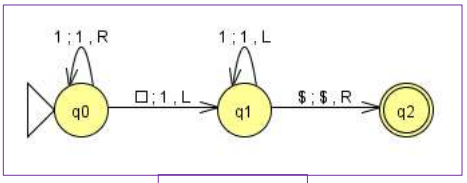
Modelo de Computación Universal

Usaremos R o L dependiendo de circunstancias.

L parece más eficiente.



MT1D1succL



MT1D1succR

24

Modelo de Computación Universal

¿Qué más podemos computar?

pred(k) → k-1

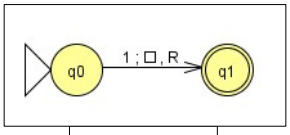
Ojo con pred(k=1), no está definido si no incluimos 0

25

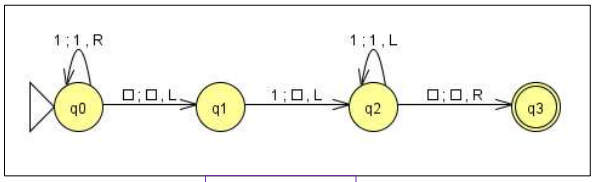
Modelo de Computación Universal

¿Qué más podemos computar?

pred(k) → k-1



MT1D1predL



MT1D1predR

Ojo con pred(k=1), no está definido si no incluimos 0

26

Modelo de Computación Universal

¿Qué más podemos computar?

Podemos combinar/concatenar estas Máquinas

Por ejemplo:

MTs (MTp (MTp (k))) (ojo, para k=2 falla)

es lo mismo que MTp (k)

¿Tiene utilidad usar la versión larga de calcular MTp (k)?

Faltan funcionalidades adicionales

27

Modelo de Computación Universal

¿Qué más podemos computar?

MTs (MTp (MTp (k))) es lo mismo que MTp (k)

ojo, para k=2 falla

Nuestra representación es insuficiente

⇒ tenemos que mejorarla

⇒ (mejor otro día)

28

Modelo de Computación Universal

¿Qué más podemos computar?

¿Qué tal si añadimos otra máquina?

➔ Máquinas Condicionales

29

Modelo de Computación Universal

¿Qué más podemos computar?

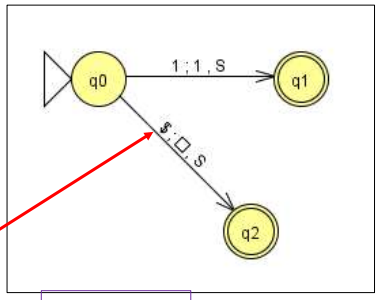
¿Qué tal si añadimos otra máquina?

➔ Máquinas Condicionales

Que detecte si k=0… ⇒ MTz (k)

bifurca si k=0

Debería ser #, pero podemos usar una marca $ para el 0



MT1D1zeroQ

30

Modelo de Computación Universal

¿Qué podemos computar combinando MTp, MTs y MTz?

?

??

???

31

Modelo de Computación Universal

¿Qué podemos computar combinando MTp, MTs y MTz?

¿Sabemos calcular algo?

¿por ejemplo una suma? x+y=?

```
while (! MTz (x)) {
   y ← MTs (y)
   x ← MTp (x)
}
```

y ← y + x

32

Modelo de Computación Universal

¿Podemos calcular x+y=? combinando MTp, MTs y MTz?

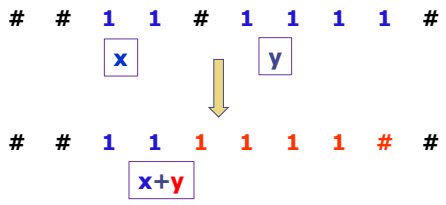
- *x e y en unario, separados por un espacio u otra marca*
- *Abstraemos al mundo de los bloques*

33

Modelo de Computación Universal

¿Podemos calcular x+y=? combinando MTp, MTs y MTz?

- *x e y en unario, separados por un espacio u otra marca*



34

Modelo de Computación Universal

¿Podemos calcular x+y=? combinando MTp, MTs y MTz?

- *x e y en unario, separados por un espacio u otra marca*
- *Abstraemos al mundo de los bloques:*

➔ *Nos hacen falta algunas MT auxiliares adicionales*

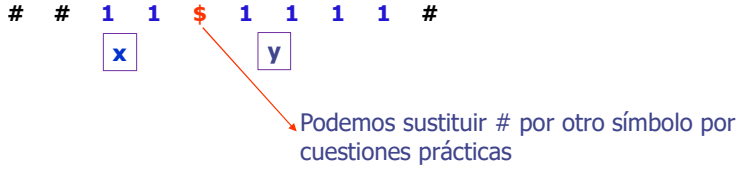
35

Modelo de Computación Universal

¿Podemos calcular x+y=? combinando MTp, MTs y MTz?

Hay que desplazar el cabezal de un campo al otro

Nos hacen falta algunas MT auxiliares adicionales



36

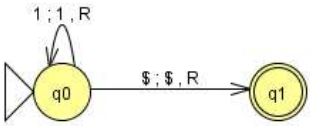
Modelo de Computación Universal

¿Podemos calcular x+y=? combinando MTp, MTs y MTz?

Hay que desplazar el cabezal de un campo al otro

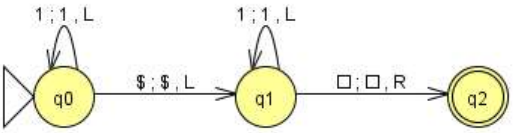
MT1D1skipR

Cambia de x a y



MT1D1skipL

Cambia de y a x



37

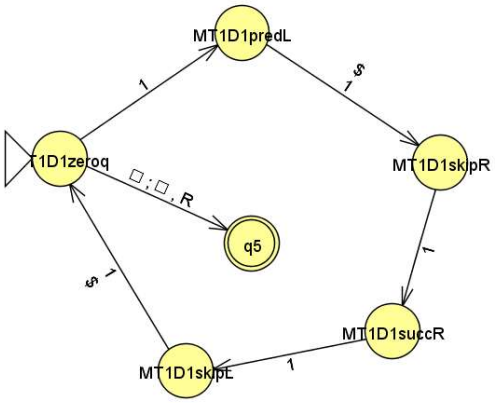
Modelo de Computación Universal

¿Podemos calcular x+y=? combinando MTp, MTs, MTz, MTkL, MTkR?

MT1D1add

Nuestro primer

MTprograma



38

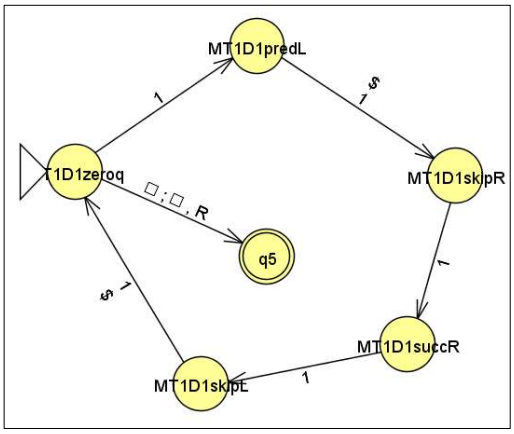
Modelo de Computación Universal

Podemos calcular x+y=? combinando MTp, MTs y MTz , MTkL, MTkR

Son equivalentes

```
while (! MTz (x)) {
      x ← MTp (x)
      y ← MTs (y)
}
```

≡



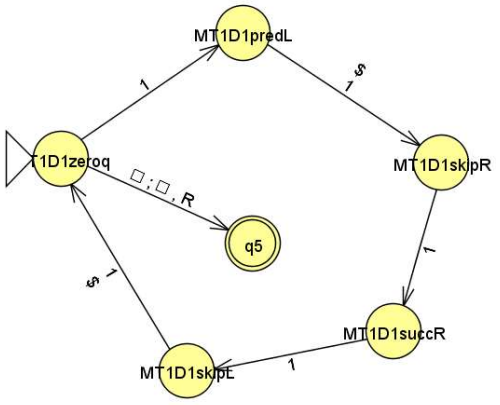
39

Modelo de Computación Universal

La MT permite construir procesadores más elaborados en base a bloques

A semejanza de un programa

procedimental



40

Modelo de Computación Universal

¿Qué más podemos computar?

*Multiplicación x*y = ?*

consiste sumar x

y veces

Nos harán falta (entre otras):

- *Mtadd*
- *MT para copiar variables (x)*

➔ Eso está hecho ➔ MTmult

41

Modelo de Computación Universal

¿Qué más podemos computar?

Factorial(x) = x! = ?

consiste sumar en multiplicar x con (x-1), (x-2), …, 1

Nos harán falta (entre otras):

- *MTmult*
- *Mtpred (para x)*

➔ Eso está hecho ➔ Mtfact(x)

42

Modelo de Computación Universal

¿Qué más podemos computar?

¿Cualquier cosa?

?

??

???

43

Modelo de Computación Universal

¿Qué más podemos computar?

Incluso el factorial. Aprovechando que nuestra cinta es infinita...

Ok, por ahora va bien.

Es probable que podamos calcular divisiones, potencias, logaritmos, etc.

- Abreviando: si podemos
- Hay que mejorar la representación 0, 1, 11, … ?

44

Modelo de Computación Universal

Las Máquinas de Turing definen un Modelo de Computación

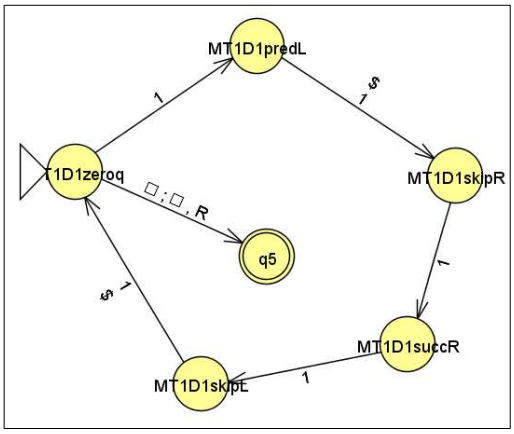
que es la base de los

Lenguajes Imperativos y

Procedimentales

```
while (! MTz (x)) {
      x ← MTp (x)
      y ← MTs (y)
}
```

≡



45

Modelo de Computación Universal

Podemos calcular divisiones, potencias, logaritmos, etc.

¿Pero es eficiente?

NO

La MT es un modelo de Computación Abstracta

No busca eficiencia.

Es más, parece un "overkill". Igual sabe hacer cosas que podríamos calcular de otra forma

No obstante, vamos a estudiar la eficiencia ⇒ Coste Computacional.

46