Universidad Nacional Autónoma de México

Facultad de Ciencias

Cadenas, anticadenas y conjuntos independientes en el orden de Turing

T E S I S

que para obtener el título de:

matemático

P R E S E N T A :

Andres Emmanuel Aim Cruz Tello

tutor

Dr. David Meza Alcántara

Ciudad Universitaria, Cd. Mx., 2023

Adán y Martha

Agradecimientos

Resumen

En esta tesis se discutirán conceptos básicos de Máquinas de Turing y Máquinas de Turing con Oráculo, lo cual permitirá introducir el concepto de Turing-reducibilidad y grados de Turing para así poder abordar problemas de insolubilidad y decibilidad.

Usaremos estas herramientas para probar la existencia de al menos un par de conjuntos incomparables según la Turing-reducibilidad, para luego extender esta construcción a una familia numerable de conjuntos independientes y finalmente demostrar la existencia de una cantidad no numerable de conjuntos incomparables.

Abstract

In this thesis, basic concepts of Turing Machines and Oracle Turing Machines will be discussed, which will allow to introduce the concept of Turing-reducibility and Turing degrees in order to address insolubility and decidability problems.

We will use these tools to prove the existence of at least one pair of Turing-reducible incomparable sets, to then extend this construction to a countable family of independent sets, and finally to prove the existence of an uncountable number of incomparable sets.

Índice general

Prefacio

La noción de algoritmo

La noción general de algoritmo no es algo ajeno a la vida diaria, muchas de las tareas más comunes en el día a día de la mayoría de la gente podrían ser descritas mediante algoritmos.

Una primera aproximación a la noción de algoritmo puede ser: "una sucesión de pasos que hay que realizar para obtener un resultado". Esta definición, a pesar de ser vaga, contiene esencialmente lo que se busca en una definición de algoritmo.

Consideremos, como habíamos mencionado en el párrafo inicial, algunas tareas comunes. Pensemos por ejemplo en el lavado de trastes; salvo algunas excepciones, las instrucciones incluyen los siguientes pasos: remojar, tallar y enjuagar. Siempre que tengamos un plato sucio podemos realizar esa sucesión de acciones y obtendremos como resultado ese mismo plato limpio. Notemos que si los pasos se realizan en un orden distinto tendremos igualmente un algoritmo aunque el algoritmo resultante no sea el adecuado para el objetivo que perseguimos.

Otro ejemplo un poco más complejo es una receta; en el ejemplo anterior considerábamos sólo un elemento (el plato) sobre el cual ejecutar las actvidades indicadas por el algoritmo (lo que más adelante llamaremos "entradas") y obteníamos un sólo resultado (más adelante llamaremos a este resultado "salida"). En contraste, al ejecutar una receta tenemos una multitud de elementos (los ingredientes) a los cuales habrá que realizarles distintas tareas para lograr el resultado (el platillo a preparar) y muchas veces, habrá tareas que se realizarán si y sólo si una tarea anterior se ha realizado con éxito. Sin embargo, si las tareas se realizan como señala la receta, tendremos

un resultado.

Ahora veamos un ejemplo un poco más cerca de casa. Consideremos la tarea de obtener el máximo común divisor de n números. Descuidando la eficiencia, un algoritmo para este proceso podría ser como sigue:

1. Hacer $n-1$ comparaciones y obtener m, el menor número de la colección.
2. Para cada número d entre m y 1 (recorriéndolos de manera decreciente), dividir los n números originales entre d. Si d divide a todos, entonces el resultado es d.

En este caso tenemos un algoritmo que acepta cualquier cantidad (finita) de entradas, ejecuta una cantidad de pasos que depende de la cantidad de entradas provistas y regresa una sola salida, un número.

Una ventaja de hablar de este tema en una época en la cual las computadoras se han establecido en la cotidianidad, es que podemos asociar al concepto abstracto de algoritmo algo un poco más "tangible", un programa computacional. Esta es justamente la razón de que el primer tópico que se toque en este proyecto de tesis sea la noción de algoritmo y el segundo, Máquinas de Turing.

Sobre Máquinas de Turing

Las Máquinas de Turing son el concepto abstracto que subyace al funcionamiento de las computadoras modernas, es decir, máquinas capaces de procesar información de una manera determinada siguiendo un conjunto de instrucciones predefinidas. Así como podemos escribir un programa para recibir dos números y que este calcule la suma de ambos números, existe una Máquina de Turing que calcula la suma de dos números cualesquiera.

En la teoría de la computabilidad existe la "convención" ampliamente aceptada y respaldada por la evidencia que ha surgido desde entonces, de que las Máquinas de Turing representan de una manera formal la noción de algoritmo, es decir, que si algo puede ser escrito en forma de algoritmo entonces existe una Máquina de Turing que puede representarlo. A esto se le llama la tesis de Church-Turing.

Es claro para cualquiera que haya tenido contacto con una computadora mo-

derna que sus capacidades no están limitadas a un cierto conjunto de instrucciones o programas sino que pueden realizar una gran cantidad de operaciones diferentes. En realidad, la computadora moderna es una implementación avanzada de algo que Turing llamó, y demostró que se puede construir, una **Máquina Computacional Universal** y que nosotros llamamos **Máquina de Turing Universal**.

Una Máquina de Turing Universal tiene la capacidad de, dentro de su mismo proceso, operar como cualquier otra Máquina de Turing.

Un ejemplo muy claro de esto, en una computadora moderna, es la capacidad de ejecutar diferentes aplicaciones como procesadores de textos o editores de video. Cada una de estas aplicaciones puede verse como un programa, es decir, una Máquina de Turing, distintos y no necesitamos cambiar de computadora para poder usar uno u otro.

Teniendo una herramienta de este calibre, uno se podría preguntar: ¿puede esta herramienta resolver todos mis problemas? y, a pesar de que ya se deben encontrar en desarrollo programas expertos en ayuda psicológica, desafortunadamente la respuesta es no.

Esta respuesta fue provista por el mismo Turing en el mismo artículo en el que presenta sus máquinas (Turing (1937)).

Problemas insolubles

Un problema es insoluble, como se podría esperar, si no tiene solución algorítimica, es decir, si no existe algún algoritmo o equivalentemente una Máquina de Turing que lo pueda resolver. Y sí, existen dichos problemas.

Uno de los ejemplos más famosos y justo el que menciona Turing en su artículo [1936] se denomina el **Problema de la detención** y su enunciado es relativamente sencillo: ¿existe algún algoritmo que nos diga, dado un programa, si este terminará su proceso en algún momento o no?

No es difícil imaginarse un programa que nunca se detenga, por ejemplo el siguiente conjunto de instrucciones:

1. Siempre que recibas un número par, súmale una unidad y vuelve a ejecutar este programa.
2. Siempre que recibas un número impar, réstale una unidad y vuelve a ejecutar este programa.

Es claro que si se ejecutara este programa y recibiera como entrada cualquier número entero aquel nunca se detendría. Sin embargo, pueden existir programas mucho más complejos que a simple vista no sería posible decidir si se detendrían o no con una cierta entrada. ¡Lo que nos dice el Problema de la detención es que no existe ningún programa o computadora que nos pueda ayudar a decidirlo!

Ahora, si se tiene una naturaleza curiosa, algunas preguntas que surgen de manera natural después de este resultado podrían ser: ¿existe algún otro problema insoluble? o ¿si pudiéramos resolver ese problema, ahora sí, podríamos resolver cualquier otro?

Las respuestas a esas preguntas son sí y no respectivamente. Y en realidad podemos resolver una con la otra de la siguiente manera:

Supongamos que logramos construir una máquina que resuelve el Problema de la detención, a esta máquina llamémosla la *Máquina D*. Entonces podríamos construir, para cualquier Máquina de Turing, una máquina similar pero con la capacidad de explotar los resultados que obtenemos de la *Máquina D* deteniendo su procesamiento y haciéndole una consulta a la *Máquina D* y usando dicha información para tomar decisiones dentro de su proceso.

Llamamos a estas nuevas máquinas, **Máquinas de Turing con Oráculo** y en este caso a la *Máquina D* el Oráculo de la máquina. Notemos que el Oráculo no tiene que ser necesariamiente la *Máquina D* sino cualquier máquina en realidad. El uso de la *Máquina D* en este ejemplo es para responder a la pregunta que hicimos inicialmente.

Usando el mismo razonamiento que se utilizó para demostrar la insolubilidad del Problema de la detención pero ahora para este conjunto de Máquinas con Oráculo, podemos demostrar que no existe ningún algoritmo que nos permita decidir si una de estas se detendrá o no. A este le llamamos el **Problema de la detención relativizado** y provee la respuesta para ambas preguntas planteadas, por un lado es un

problema insoluble similar pero diferente del Problema de la detención y por el otro es un problema que no podríamos resolver incluso si pudiésemos resolver el Problema de la detención.

Este procedimiento suena repetible y lo es. Por ejemplo, ¿qué pasaría si ahora usáramos el Problema de la detención relativizado como Oráculo para las Máquinas de Turing y nos preguntáramos nuevamente por el Problema de la detención?

Esto es posible y obtendríamos como resultado otro problema insoluble distinto. Aún más, esto lo podemos repetir cuantas veces queramos y siempre obtendremos un problema insoluble distinto, es decir, ¡Existen al menos tantos problemas insolubles diferentes como números naturales!

Comentarios personales

El estudio de la insolubilidad es uno de los ejemplos más palpables de la tenacidad y curiosidad de la que son capaces los matemáticos considerando que, a pesar de haber demostrado la imposibilidad de la solución de un problema, esto detonó todo un nuevo campo de investigación que, como sugiere la mera existencia de esta tesis, sigue dando frutos.

Espero que esta sucinta explicación logre inspirar suficiente interés en el lector para continuar su camino a través de los siguientes capítulos,

Capítulo 1

Introducción

Si algo parece imposible, no te rindas.
Primero prueba que es imposible.
Entonces ríndete.

Anónimo

Debemos saber, sabremos.

David Hilbert

El objetivo de este proyecto de tesis es probar el Teorema 3.5.1 que dice lo siguiente:

Teorema (Kleene-Post)**.** Existen conjuntos $A, B \leq_T \emptyset'$ tales que $A|_T B$ y, por lo tanto, $\emptyset <_T A, B <_T \emptyset'$.

Aquí, el símbolo $<_T$ representa reducibilidad de Turing y $\emptyset'$ representa el salto de Turing del conjunto vacío. Estos se construirán y definirán comenzando desde el concepto de Máquinas de Turing.

La Sección 2.1 servirá para definir las Máquinas de Turing, a las cuales nos referiremos de aquí en adelante como **MT**. Esta sección comienza con un acercamiento

informal cuyo objetivo es presentar las ideas que inspiran la definición formal. Continúa con una definición formal de Máquinas de Turing. Preferimos para esto la definición mediante cuádruplas que aparece en Davis (1982), simplemente por facilidad de lectura. Otra definición equivalente con quíntuplas puede encontrarse, por ejemplo, en Soare (2016).

Después de haber establecido la definición de una MT, en la Subsección 2.1.3 definimos formalmente el proceso que lleva a cabo una MT. Los conceptos más importantes que se discuten son los de *descripción instantánea* y *cómputo*. En esta subsección se aprovecha para asociar las MT con funciones, para definir a las *funciones computables*.

En la Subsección 2.1.4 se extienden las capacidades de cómputo de las MT definiendo las Máquinas de Turing con oráculo, lo que permite que estas puedan obtener información útil para su cómputo de un conjunto determinado A.

También asociamos las MT con oráculo, con funciones que se denominan $A-$ *computables*.

Terminamos la sección con algunos resultados que sirven para concluir que la *computabilidad* es un caso particular de la $A-$ *computabilidad* y en realidad es equivalente a la $\emptyset-$ *computablidad*.

En la sección Sección 2.2 mostramos algunas funciones notables que son computables tales como la función sucesor, la función constante cero o la i-ésima proyección.

Luego, en la Subsección 2.2.2 demostramos la existencia de MT regulares, es decir, máquinas cuya salida es adecuada para representar la entrada de otra máquina. Esto abre la posibilidad de realizar operaciones de composición sobre MT de manera análoga como lo haríamos con funciones. Aprovechamos esta nueva habilidad con las MT para demostrar la existencia de algunas MT que serán importantes para las siguientes secciones, por ejemplo, las MT que mantienen el valor de sus entradas (Corolario 2.2.3.1), la MT que cuenta el número de unos en una cinta (Teorema 2.2.8) o la MT que permite hacer iteraciones (Teorema 2.2.7).

Dos resultados cruciales de esta sección se presentan en Teorema 2.2.10 y Teorema 2.2.11, donde se prueba que las máquinas obtenidas mediante la operación de

composición y minimalización, respectivamente, son computables.

La Subsección 2.2.3 amplía la baraja de operaciones que podemos realizar sobre las MT al incluir la recursión primitiva, además de utilizar esta nueva herramienta para demostrar que funciones como la suma, la multiplicación, la exponenciación y el factorial son $A - computables$.

Continuamos con la Sección 2.3 en la cual definimos los predicados y las funciones recursivas, con el objetivo de establecer una relación entre la computabilidad y la recursividad.

Se demuestra que la computabilidad y recursividad son equivalentes, es decir, que toda MT tiene asociada una función recursiva y viceversa.

Un resultado importante se presenta en la Subsección 2.3.3 y es que se puede asociar a cada MT, un número natural. Esto se hace mediante una técnica de gödelización que además permite realizar esta asociación de una manera recursiva, o de otra manera, computable.

En la Sección 2.4 demostramos algunas propiedades del conjunto de las funciones computables, además de presentar resultados vitales para la indecidibilidad de problemas como el de la detención, en la Sección 3.1.

En la Sección 3.2 se presenta el concepto de Grados de Turing, junto con la reducibilidad de Turing y el símbolo $\leq_T$.

Finalmente en el Capítulo 3 se utilizan los resultados de los capítulos anteriores además de algunos conceptos de árboles que se obtienen de Kechris (1995) para probar el Teorema 3.5.1.

Capítulo 2

Máquinas de Turing

2.1. Máquinas de Turing.

Este capítulo servirá para establecer una definición formal de las Máquinas de Turing. En la primera sección describiremos informalmente el concepto que inspira la construcción formal, de tal manera que en la segunda sección, las ideas expresadas sean más familiares al lector.

2.1.1. Acercamiento informal

Consideremos un dispositivo mecánico finito que contiene una cinta de papel que es infinita en ambas direcciones. Dicha cinta está separada en espacios bien definidos. A estos sitios se les da la denominación de *celdas*. Así bien, el dispositivo está estructurado de tal manera, que la cinta corre a través de él dejando únicamente margen para que una celda quede dentro del mismo, y que a su vez lleva por nombre *celda examinada* o *celda que está siendo examinada por el dispositivo*. A las direcciones a lo largo de la cinta se les asignan los nombres de *izquierda* y *derecha*. El dispositivo es capaz de hacer las siguientes operaciones:

1. Puede escribir un símbolo en la celda que está examinando si es que ningún símbolo está escrito ya en la celda o reemplazar un símbolo por otro si es que la celda no está vacía (figura 2.1a).

2. Puede borrar el símbolo contenido en la celda que está examinando, dejándola *en blanco* o *vacía*, si es que la celda contiene o no un símbolo escrito en ella (figura 2.1b).
3. Puede examinar una celda que esté a la izquierda de la que está examinando actualmente (moviendo la cinta a la derecha, figura 2.1c).
4. Puede examinar una celda que esté a la derecha de la que está examinando actualmente (moviendo la cinta a la izquierda, figura 2.1d).

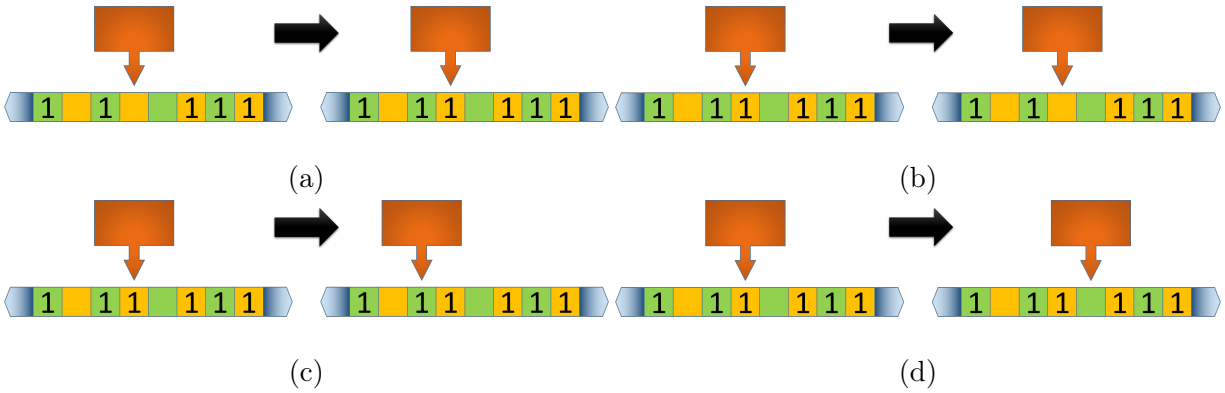

Figura 2.1: Distintas capacidades que tiene una Máquina de Turing

El dispositivo, cuando está activo, es capaz de realizar alguna de las acciones anteriores una a la vez. Nos referiremos a la realización de una de estas acciones como un *paso*. Después de haber realizado un paso, el dispositivo se posiciona en una configuración particular que llamaremos *estado interno*, o simplemente *estado* cuando no haya posibilidad de confusión. Usaremos los símbolos q_i con $i = 0, 1, 2, \ldots$ para referirnos a los distintos estados internos que puede tomar la máquina. Una Máquina de Turing solo puede colocarse en una cantidad finita de estados internos.

Finalmente, el dispositivo está construido de tal manera que se comporta de acuerdo a una cierta cantidad finita de reglas. Estas reglas determinan, de acuerdo al estado interno del dispositivo y la condición de la celda que está siendo examinada, el siguiente paso que hay que tomar y el estado interno que tomará al final de dicho paso.

Supongamos que $S_0 = B, S_1 = 1, S_2, \ldots$ denotan el posible contenido de cualquier celda (B significa que la celda está vacía). Supongamos también que $S_i (i \neq 0)$, B,

R, L denotan las operaciones básicas (1), (2), (3) y (4), respectivamente. Entonces el conjunto de reglas que definen el comportamiento del dispositivo pueden expresarse por medio de cuádruplas (sucesiones de cuatro símbolos) de la siguiente manera:

Cada cuádrupla consiste en símbolos acomodados en un orden específico, el primer símbolo es el símbolo de un estado interno, el segundo es un símbolo que representa el posible contenido de la celda que está siendo examinada, el tercero es un símbolo que representa una operación y el cuarto es un símbolo que representa un estado interno.

Una cuádrupla representa una regla de comportamiento que indica al dispositivo que cuando se encuentre en el estado representado por el primer símbolo, y el contenido de la celda que está siendo examinada es el que representa el segundo símbolo, entonces debe realizarse la operación representada por el tercer símbolo y terminar con el estado interno representado por el cuarto símbolo.

Por ejemplo, $q_i S_n R q_j$ corresponde a las siguiente regla: si el dispositivo se encuentra en el estado q_i, y el símbolo escrito en la celda que está siendo examinada en este momento es S_n, entonces se debe examinar la celda de la derecha y el dispositivo debe pasar al estado q_j; mientras que la cuádrupla $q_1 S_2 L q_4$ corresponde a: si el dispositivo se encuentra en el estado q_1 y el símbolo escrito en la celda que está siendo examinada en este momento es S_2, entonces se debe examinar la celda de la izquierda y el dispositivo debe pasar al estado q_4. Supongamos que una de las reglas de la máquina que estamos estudiando es $q_1 B L q_3$ entonces podemos visualizar este paso como sigue:

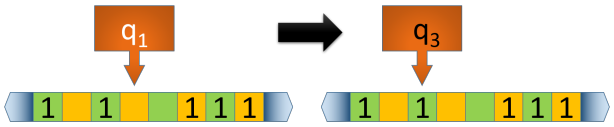

Figura 2.2: Comportamiento de una Máquina de Turing que contiene en su conjunto de reglas a la cuádrupla $q_1 B L q_3$

El proceso de la máquina continúa mientras exista una regla en el conjunto de cuádruplas que le diga a la máquina el siguiente paso a tomar; en otras palabras, el proceso de la máquina continúa mientras una de las cuádruplas inicie con un par de símbolos que corresponda el estado interno en el que se encuentre la máquina y el

símbolo escrito en la celda que está siendo examinada.

Si en algún momento esta situación no se presenta, decimos que la máquina se *detiene* o *converge*; puesto que no tiene cómo continuar su proceso. Llamamos a esta situación, *condición de detención* y una vez que se ha alcanzado, a la serie de pasos llevados desde el inicio de operación de la máquina hasta que se alcanzó la condición de detención le llamamos un *cómputo* de la máquina y a lo que queda impreso en la cinta una vez que se ha realizado un cómputo le llamamos *resultado del cómputo*.

Para ejemplificar este comportamiento, consideremos la Máquina de Turing $I = \{q_1 1 B q_1, q_1 B B q_2\}$. Supongamos que en la cinta tenemos escritos cuatro símbolos 1, entonces podemos visualizar el proceso que tendría esta máquina de la siguiente manera:

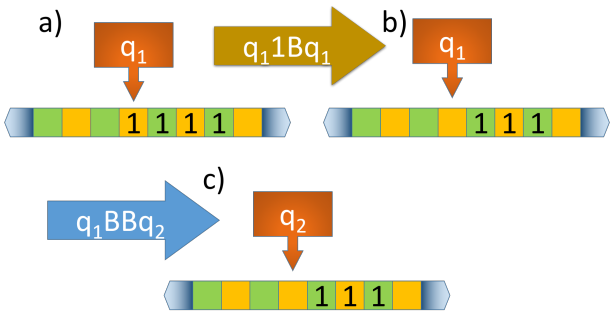

Figura 2.3: Comportamiento de la máquina I

Observamos que para el momento en el que el proceso llega a estar como en la figura *c*), la máquina debería realizar alguna acción al estar en estado interno q_2 leyendo en la cinta una celda vacía, sin embargo, no existe ninguna cuádrupla en la máquina I que inicie con esos dos símbolos, por lo que la condición de detención ha sido alcanzada y podemos decir que el resultado del cómputo de la máquina I cuando esta está siendo alimentada con cuatro símbolos 1 es tres.

Habiendo expuesto los conceptos básicos para describir el comportamiento de las Máquinas de Turing, podemos proceder a dar una definición formal de este concepto.

2.1.2. Definición Formal

En el acercamiento informal de la sección anterior imaginamos a las Máquinas de Turing como dispositivos físicos que efectúan tareas sobre una cinta, sin embargo, en la definición formal abstraeremos las funciones de dicho dispositivo y sus efectos sobre la cinta.

Comenzaremos con la definición de una expresión:

Definición 2.1.1. Una expresión es una sucesión finita de símbolos elegidos del conjunto: $\{S_n|n \in \omega\} \bigcup \{q_n|n > 0\} \bigcup \{R, L\}$. ♡

Definición 2.1.2. Sean S_i, S_j, q_k, q_l, q_m $i, j, k, l, m \in \omega$. Una cuádrupla es una expresión de alguna de las siguientes formas:

1. $q_m S_i S_j q_l$
2. $q_m S_i R q_l$
3. $q_m S_i L q_l$
4. $q_m S_i q_k q_l$ ♡

Por el momento nos referiremos sólo a las cuádruplas de tipo 1,2 y 3, las cuádruplas de tipo 4 aparecerán más adelante.

Definición 2.1.3 (Máquina de Turing)**.** Una *Máquina de Turing* es un conjunto no vacío de cuádruplas, tal que cualesquiera dos no comienzan con el mismo par de símbolos. Esta condición es llamada *restricción de consistencia.* ♡

La *restricción de consistencia* evita ambigüedad alguna en la ejecución de las reglas.

Definición 2.1.4. Sea Z una Máquina de Turing. Llamamos al conjunto $\{S_i|S_i$ aparece en algúna de las cuádruplas de $Z\}$ el *alfabeto* de Z. ♡

Podemos notar que la enumeración de los símbolos del alfabeto de Z es la misma para todas las MT.

2.1.3. Máquinas de Turing y funciones parciales

Dada una MT, una cinta, una celda en dicha cinta y un estado interno inicial, la MT realiza una sucesión de operaciones que está unívocamente definida como

consecuencia de la restricción de consistencia, esta sucesión de pasos puede o no terminar en una sucesión finita de estados. Definiremos este proceso:

Definición 2.1.5. Una *descripción instantánea* es una expresión que contiene exactamente una q_i, no contiene a R ni a L y además q_i no es el símbolo más a la derecha. ♡

Si Z es una MT y α es una descripción instantánea, decimos que α *es una descripción instantánea de* Z si la q_i que ocurre en α es un estado interno de Z y el resto de los símbolos pertenece a $\{S_0, S_1, \dots\}$.

La definición de descripción instantánea nos servirá para describir ciertos momentos dentro del proceso que realiza una MT. Esto con la ayuda de la siguiente definición:

Definición 2.1.6. Una expresión que consista enteramente de símbolos que pertenezcan a $\{S_0, S_1, \dots\}$ es llamada una *expresión de cinta.* ♡

Ahora definiremos los siguientes elementos que nos permitirán describir el comportamiento de una MT:

Definición 2.1.7. Sea Z una Máquina de Turing, y sea α una descripción instantánea de Z, además q_i es el símbolo de estado interno que aparece en α y S_j es el símbolo inmediatamente a la derecha de q_i. Entonces llamamos a q_i *el estado interno de* Z *en* α y a S_j le llamamos el *símbolo leído por* Z *en* α. La expresión de cinta obtenida al quitar a q_i de α le llamamos la *expresión de cinta de* Z *en* α. ♡

Por ejemplo, consideremos la máquina que se presenta en la Figura 2.3. Una descripción instantánea para la Máquina I podría ser $\alpha = q_1 S_2 1111 BBBB$, en este caso, q_1 sería el estado interno de Z en α, el símbolo leído por Z en α sería S_2 y la expresión de cinta de Z en α consiste en $S_2 1111 BBBB$.

Anteriormente describimos a las Máquinas de Turing como dispositivos que cambian de estados internos con relación a la forma en que están "programadas", es decir, con respecto a las cuádruplas que contienen. Ahora explicaremos este proceso mediante descripciones instantáneas. Esto nos permitirá estudiar formalmente el proceso que sigue una MT.

Definición 2.1.8. Sea Z una MT, sean α y β descripciones instantáneas y $S_j, S_k \in \{S_0, S_1, \dots\}$. Escribimos $\alpha \to \beta$ (Z), o si no existe ambigüedad $\alpha \to \beta$, para representar alguno de los siguientes casos:

1) Existen expresiones P y Q que pueden ser vacías tales que:

$$\begin{aligned} \alpha \quad es \quad Pq_iS_jQ, \\ \beta \quad es \quad Pq_lS_kQ, \end{aligned}$$

donde Z contiene a $q_iS_jS_kq_l$

2) Existen expresiones P y Q que pueden ser vacías tales que:

$$\begin{aligned} \alpha \quad es \quad Pq_iS_jS_kQ, \\ \beta \quad es \quad PS_jq_lS_kQ \end{aligned}$$

donde Z contiene a $q_iS_jRq_l$.

3) Existe una expresión P que puede ser vacía tal que:

$$\begin{aligned} \alpha \quad es \quad Pq_iS_j, \\ \beta \quad es \quad PS_jq_lB, \end{aligned}$$

donde Z contiene a $q_iS_jRq_l$.

4) Existen expresiones P y Q que pueden ser vacías tales que:

$$\begin{aligned} \alpha \quad es \quad PS_kq_iS_jQ, \\ \beta \quad es \quad Pq_lS_kS_jQ, \end{aligned}$$

donde Z contiene a $q_iS_jLq_l$.

5) Existe una expresión Q que puede ser vacía tal que:

$$\begin{aligned} \alpha \quad es \quad q_iS_jQ, \\ \beta \quad es \quad q_lBS_jQ, \end{aligned}$$

donde Z contiene a $q_iS_jLq_l$. ♡

Podemos considerar a la relación "$\to$" como la máquina Z pasando de la descripción instantánea α a la β por instrucciones de la cuádrupla correspondiente.

Por ejemplo, supongamos que $a = q_1B1q_2 \in Z$, $\alpha = 1111q_1BBBB$ y $\beta = 1111q_21BBB$, entonces por el inciso 1 de la definición, $\alpha \to \beta$. Esto lo podemos

interpretar como a indicándole a Z que cuando se encuentre en el estado interno q_1 y lea en la cinta un B (la celda esté vacía), escriba en la cinta un 1 y pase a estar en un estado interno q_2.

Para el inciso 2 de la de la definición, supongamos que $a = q_1BRq_2 \in Z$, $\alpha = 1111q_1BB1B$ y $\beta = 1111Bq_2B1B$ entonces, $\alpha \to \beta$. Esto lo podemos interpretar como a indicándole a Z que cuando se encuentre en el estado interno q_1 y lea en la cinta un B (la celda esté vacía), se mueva a leer el símbolo a la derecha del símbolo actual y pase a estar en un estado interno q_2.

Para el inciso 3 de la definición, supongamos que $a = q_1BRq_2 \in Z$, $\alpha = 1111q_1B$ y $\beta = 1111Bq_2B$, entonces $\alpha \to \beta$. Esto lo podemos interpretar como a indicándole a Z que cuando se encuentre en el estado interno q_1 y lea en la cinta un B (la celda esté vacía), se mueva a leer el símbolo a la derecha del símbolo actual y pase a estar en un estado interno q_2, pero, además, si es que la máquina llega al símbolo más a la derecha escrito en la cinta, podemos suponer que el resto de las celdas en la cinta a la derecha están vacías.

Los incisos 4 y 5 funcionan de manera similar a los incisos 2 y 3 respectivamente, pero cambiando el movimiento de derecha por izquierda.

Derivado de esta definición podemos demostrar los siguientes teoremas:

Teorema 2.1.1. *Si $\alpha \to \beta$ (Z) y $\alpha \to \gamma$ (Z), entonces $\beta = \gamma$*

Demostración. Sean P y Q expresiones, posiblemente vacías y además $S_j, S_k, S_l, S_m \in \{B, 1\}$.

Notemos que en cada caso, para que β suceda a α en el proceso de Z, la cuádrupla testigo de esto empieza con el símbolo de estado interno que aparece en α y el segundo símbolo en la cuádrupla es el símbolo leído por Z en α. Luego, debido a la restricción de consistencia, esta cuádrupla es única, sea a esa cuádrupla.

Luego, supongamos que $\alpha = Pq_iS_jQ$ y $a = q_iS_jS_kq_l$. Entonces el símbolo de estado interno que aparece en β y en γ es q_l y en el símbolo que lee Z en β y en γ es S_k, por lo tanto, $\beta = Pq_lS_kQ = \gamma$.

Luego, supongamos que $\alpha = Pq_iS_jS_kQ$ y $a = q_iS_jRq_l$. Entonces el símbolo de

estado interno que aparece en β y en γ es q_l y $\beta = PS_jq_lS_kQ = \gamma$.

Supongamos ahora que $\alpha = Pq_iS_j$ y $a = q_iS_jRq_l$. Entonces el símbolo de estado interno que aparece en β y en γ es q_l y $\beta = PS_jq_lB = \gamma$.

Luego, supongamos que $\alpha = PS_jq_iS_kQ$ y $a = q_iS_jLq_l$. Entonces el símbolo de estado interno que aparece en β y en γ es q_l y $\beta = Pq_lS_jS_kQ = \gamma$.

Supongamos ahora que $\alpha = q_iS_jQ$ y $a = q_iS_jLq_l$. Entonces el símbolo de estado interno que aparece en β y en γ es q_l y $\beta = q_lBS_jQ = \gamma$.

Por lo tanto, $\beta = \gamma$ □

Teorema 2.1.2. *Si $\alpha \to \beta$ (Z) y $Z \subset Z'$, entonces $\alpha \to \beta$ (Z')*

Demostración. Como $\alpha \to \beta$ (Z), tenemos que la cuádrupla testigo de esto, a, está en Z, como $Z \subset Z'$, tenemos que $a \in Z'$. Por lo tanto, $\alpha \to \beta$ (Z'). □

Ahora que podemos hablar acerca del proceso que conlleva tomar un paso para una Máquina de Turing, consideremos el proceso que lleva a cabo una Máquina desde el inicio.

Comencemos por saber cómo definir cuando una Máquina de Turing se ha detenido.

Definición 2.1.9. Llamamos a una descripción instantánea α *terminal* con respecto a Z si no existe alguna descripción instantánea β tal que $\alpha \to \beta$ (Z). ♡

Definición 2.1.10. Definimos un *cómputo* de una Máquina de Turing Z como una sucesión finita $\alpha_1, \alpha_2, \ldots, \alpha_p$ de descripciones instantáneas tales que para toda $1 \leq i < p$, se cumple que $\alpha_i \to \alpha_{i+1}$ (Z) y que α_p es terminal con respecto a Z. En tal caso, escribimos $\alpha_p = \text{Res}_Z(\alpha_1)$ y decimos que α_p es el *resultado* de α_1 con respecto a Z. ♡

Por convención, supondremos que el símbolo de estado interno que aparece en α_1 es q_1.

Por el momento hemos hablado de Máquinas de Turing que trabajan con entradas sin algún tipo de representación u orden. Sin embargo, nos interesan Máquinas de

Turing que puedan manejar números, por lo que tenemos que representar dichos números en el lenguaje que usan las Máquinas de Turing.

Para ese efecto, empecemos por definir lo siguiente: Si n es un entero positivo y $S_i \in \{B, 1\}$ escribimos S_i^n para la expresión $\underbrace{S_i S_i \dots S_i}_{\text{n-veces}}$ que consiste en n ocurrencias del símbolo S_i. Así, S_i^0 es la expresión vacía.

Con esto podemos establecer la siguiente definición:

Definición 2.1.11. Asociamos a cada número n la expresión de cinta $\overline{n}$, llamada *numeral* de n, donde $\overline{n} = 1^{n+1}$. ♡

Definición 2.1.12. Asociamos a cada n-ada $(k_1, k_2, \dots, k_n)$ de enteros la expresión de cinta $\overline{(k_1, k_2, \dots, k_n)}$ llamada *numeral* de $(k_1, k_2, \dots, k_n)$ donde:

$$\overline{(k_1, k_2, \dots, k_n)} = \overline{k_1} B \overline{k_2} B \dots B \overline{k_n}.$$ ♡

Derivado de esta definición tenemos que, por ejemplo: $\overline{4} = 11111$ y $\overline{(1, 2, 3, 4)} = 11B111B1111B11111$.

Ahora veremos cómo recuperar la información contenida en una expresión de cinta para convertirla en un número de nuevo:

Definición 2.1.13. Sea M una expresión, Entonces $\langle M \rangle$ es el número de ocurrencias del símbolo 1 en M. ♡

Por ejemplo, $\langle q_1 q_2 q_3 \rangle = 0$ y $\langle \overline{m-1} \rangle = m$ y además, si P y Q son expresiones, entonces $\langle PQ \rangle = \langle P \rangle + \langle Q \rangle$. Ahora podemos asociar una función parcial a cada MT de la siguiente manera:

Definición 2.1.14. Sea Z una Máquina de Turing. Entonces, para cada n, asociamos a Z la función parcial n-aria:

$$\Psi_Z^{(n)}(x_1, x_2, \dots, x_n)$$

como sigue. Para cada n-ada $(x_1, x_2, \dots, x_n)$, definimos $\alpha_1 = q_1 \overline{(x_1, x_2, \dots, x_n)}$ y sucede uno de los dos siguientes casos:

1) Existe un cómputo $\alpha_1, \alpha_2, \ldots, \alpha_p$ de Z. En ese caso definimos:

$$\Psi_Z^{(n)}(x_1, x_2, \ldots, x_n) = \langle \alpha_p \rangle = \langle \mathrm{Res}_Z(\alpha_1) \rangle$$

y decimos que la máquina *converge* con entrada $(x_1, x_2, \ldots, x_n)$, lo cual representamos con $\Psi_Z^{(n)} \downarrow$.

2) No existe ningún cómputo en Z. Es decir, $\mathrm{Res}_Z(\alpha_1)$ no está definido. En ese caso

$$\Psi_Z^{(n)}(x_1, x_2, \ldots, x_n)$$

no está definido.

Escribimos Ψ_Z para $\Psi_Z^{(1)}$. ♡

Definición 2.1.15. Sean $n \in \omega$. Una función parcial n-aria $f : \omega^n \to \omega$ es *computable* si existe una Máquina de Turing Z tal que para todo $(x_1, \ldots, x_n) \in \mathrm{Dom}(f) = \mathrm{Dom}(\Psi_Z^{(n)})$

$$f(x_1, \ldots, x_n) = \Psi_Z^{(n)}(x_1, \ldots, x_n).$$

En este caso, decimos que Z computa a f. ♡

2.1.4. Máquinas de Turing y Cómputos Relativos

En esta sección analizaremos un cambio a las Máquinas de Turing que permitirá ampliar su habilidad de computar funciones. En este momento entran en juego las cuádruplas de la forma $q_i S_j q_k q_l$. Estas servirán para que la Máquina de Turing consulte la información del "mundo exterior", es decir, la máquina podrá tomar decisiones tomando como base información provista de antemano por un conjunto de la siguiente manera:

Definición 2.1.16. Una *Máquina de Turing con oráculo* es una Máquina de Turing con una cinta añadida que es de solo lectura, esto quiere decir que los símbolos que existen sobre ella no pueden ser modificados. Sobre esta cinta se encuentra escrita la función característica de un conjunto A. Esta Máquina de Turing tiene la capacidad de mover ambas cintas (y por ende, de leer símbolos sobre ellas) de manera independiente.

♡

Para ello tenemos la siguiente:

Definición 2.1.17. Sean α y β descripciones instantáneas. Entonces definimos

$$\alpha \underset{A}{\rightarrow} \beta(Z)$$

si existen expresiones P y Q (posiblemente vacías) tales que:

$$\alpha \quad \text{es} \quad Pq_iS_jQ,$$
$$Z \quad \text{contiene a } q_iS_jq_kq_l,$$

y, además, ocurre una de las siguientes:

- La celda que ocupa la posición $\langle\alpha\rangle$-ésima en la cinta oráculo tiene escrito un 1 i.e., $\langle\alpha\rangle \in A$ y β es Pq_kS_jQ, o
- La celda que ocupa la posición $\langle\alpha\rangle$-ésima en la cinta oráculo tiene escrito un 0 i.e., $\langle\alpha\rangle \notin A$ y β es Pq_lS_jQ. ♡

Esto se podría interpretar como que la máquina tiene la capacidad de preguntar si la cantidad de 1's que aparece en su cinta en determinado momento, pertenece al conjunto oráculo, y tomar una decisión de acuerdo a la respuesta cambiando el estado de la máquina a uno u otro estado. Por ejemplo, si Z contiene a $q_iS_jq_kq_l$ podríamos interpretar que la máquina cambia al estado q_k si la respuesta es positiva, y al estado q_l si la respuesta es negativa.

Ahora veamos cómo una Máquina de Turing puede usar esta habilidad para llevar a cabo operaciones.

Definición 2.1.18. Sea α una descripción instantánea de la forma Pq_iS_jQ. Entonces, α es *final* con respecto a Z si Z no contiene ninguna cuádrupla, la cual tenga como primeros dos símbolos a q_iS_j. ♡

Podemos notar que el concepto de descripción instantánea terminal (Definición 2.1.9) y descripción instantánea final están relacionados con el final de un cómputo, sin embargo, se distinguen en el tipo de MT con respecto a las que ocurren, mientras que las descripciones instantáneas terminales solo pueden ocurrir para MT *simples* (es

decir, no contiene cuádruplas de la forma $q_iS_jq_kq_l$), las descripciones instantáneas finales pueden ocurrir con respecto a cualquier tipo de MT. Existen casos en los cuales ambos conceptos son equivalentes, como lo ilustran los siguientes teoremas.

Teorema 2.1.3. *Si Z es* simple*, entonces α es terminal con respecto a Z si y sólo si es final con respecto a Z.*

Demostración. Supongamos que Z es una Máquina de Turing simple, y que α es terminal con respecto a Z, lo cual significa que no existe ninguna descripción instantánea β tal que $\alpha \to \beta\ (Z)$. Supongamos que $\alpha = PS_mq_iS_jQ$ donde P y Q son expresiones posiblemente vacías y, además, que Z contiene a la cuádrupla $q_iS_jS_kq_l$, entonces $\alpha = PS_mq_iS_jQ \to PS_mq_lS_kQ\ (Z)$, lo cual es una contradicción. Lo mismo sucedería si Z contuviera a la cuádrupla $q_iS_jRq_l$, puesto que en ese caso $\alpha = PS_mq_iS_jQ \to PS_mS_jq_lQ\ (Z)$, o si Z contuviera a la cuádrupla $q_iS_jLq_l$, dado que en ese caso $\alpha = PS_mq_iS_jQ \to Pq_lS_mS_jQ\ (Z)$. Como Z es simple, estas son todas las posibles opciones para las cuádruplas que puede contener. Por lo tanto, Z no contiene ninguna cuádrupla que inicie con q_iS_j.

Por otro lado, supongamos que $\alpha = PS_mq_iS_jQ$ es final con respecto a Z, eso quiere decir que no hay ninguna cuádrupla en Z que tenga como pareja de símbolos iniciales a q_iS_j. Supongamos entonces que existe β tal que $\alpha \to \beta\ (Z)$.
En el caso de que $\beta = PS_mq_lS_kQ$, como $\alpha \to \beta\ (Z)$, tenemos que Z contiene la cuádrupla $q_iS_jS_kq_l$. De otra manera, si $\beta = PS_mS_jq_lQ$, como $\alpha \to \beta\ (Z)$, tenemos que Z contiene la cuádrupla $q_iS_jRq_l$. Por último, si $\beta = Pq_lS_mS_jQ$, como $\alpha \to \beta\ (Z)$, tenemos que Z contiene la cuádrupla $q_iS_jLq_l$.
En cualquiera de los tres casos, tenemos que Z contiene una cuádrupla tal que los primeros dos símbolos de la misma son q_iS_j, lo cual es una contradicción. Por lo tanto, se tiene el resultado. □

Teorema 2.1.4. *α es final con respecto a Z si y sólo si*

- *α es terminal con respecto a Z y,*
- *no existe un conjunto A ni descripción instantánea β tales que $\alpha \underset{A}{\to} \beta(Z)$.*

Demostración. Supongamos que $\alpha = QS_m q_i S_j T$ es final con respecto a Z, por los argumentos del teorema anterior tenemos que también es terminal con respecto a Z. Bastaría ver que no existe un conjunto A ni descripción instantánea β tales que $\alpha \underset{A}{\rightarrow} \beta$ (Z).
Supongamos entonces que existe un conjunto A y una descripción instantánea β tales que $\alpha \underset{A}{\rightarrow} \beta$ (Z). Pero esto significa que Z contiene una cuádrupla como la siguiente: $q_i S_j q_j q_l$, lo cual es una contradicción.
Por otro lado, supongamos que $\alpha = QS_m q_i S_j T$ es terminal con respecto a Z y no existe un conjunto A ni descripción instantánea β tales que $\alpha \underset{A}{\rightarrow} \beta(Z)$.
Por los argumentos del teorema anterior, sabemos que Z no puede contener ninguna cuádrupla de tipo 1, 2 o 3. Supongamos entonces que Z contiene una cuádrupla de tipo 4 de la siguiente forma: $q_i S_j q_k q_l$ Entonces tenemos que $\alpha \underset{\emptyset}{\rightarrow} QS_m q_l S_j T$, lo cual es una contradicción. Por lo tanto, se tiene el resultado. □

Ahora veamos cómo se realiza un cómputo donde interviene un oráculo.

Definición 2.1.19. Definimos un *A-cómputo* de una Máquina de Turing Z como una sucesión finita de descripciones instantáneas $\alpha_1, \alpha_2, \ldots, \alpha_p$ tal que, para cada i, $1 \leq i < p$ tenemos que:

$$\alpha_i \underset{A}{\rightarrow} \alpha_{i+1}(Z) \quad o \quad \alpha_i \rightarrow \alpha_{i+1}(Z)$$

y, además, α_p es final con respecto a Z. En este caso definimos $\alpha_p = \mathrm{Res}_Z^A(\alpha_1)$ y llamamos a α_p el *A-resultado* de α_1 con respecto a Z. ♡

De manera muy parecida a cómo asociamos una función parcial a cada Máquina de Turing simple, podemos asociar una función parcial a cada Máquina de Turing con oráculo de la siguiente manera:

Definición 2.1.20. Sea Z una Máquina de Turing. Entonces, para cada n, asociamos a Z con la función parcial n-aria (que generalmente depende del conjunto A):

$$\Psi_{Z;A}^{(n)}(x_1, x_2, \ldots, x_n)$$

como sigue. Para cada n-ada $(x_1, x_2, \ldots, x_n)$, definimos $\alpha_1 = q_1\overline{(x_1, x_2, \ldots, x_n)}$ y sucede uno de los dos siguientes casos:

1) Existe un A-cómputo $\alpha_1, \alpha_2, \ldots, \alpha_p$ de Z. En ese caso definimos:

$$\Psi^{(n)}_{Z;A}(x_1, x_2, \ldots, x_n) = \langle \alpha_p \rangle = \langle \mathrm{Res}^A_Z(\alpha_1) \rangle$$

y decimos que la máquina *converge* con entrada $(x_1, x_2, \ldots, x_n)$, lo cual representamos con $\Psi^{(n)}_{Z;A} \downarrow$.

2) No existe ningún A-cómputo en Z con la entrada dada. Es decir, $\mathrm{Res}^A_Z(\alpha_1)$ no está definido. En ese caso

$$\Psi^{(n)}_{Z;A}(x_1, x_2, \ldots, x_n)$$

no está definido.

Escribimos Ψ^A_Z para $\Psi^{(1)}_{Z;A}$ ♡

Notemos que derivado de la definición de $A-$cómputo se pueden concluir los siguientes lemas:

Lema 2.1.5. *Sea Z una MT y $A \subseteq \omega$. Si $\Psi^{(n)}_{Z;A} \downarrow$, entonces se cumple lo siguiente:*

1. *existe $s \subset A$ finito tal que $\Psi^{(n)}_{Z;s} \downarrow$, y*
2. *para todo conjunto S tal que $s \subseteq S$ se tiene que $\Psi^{(n)}_{Z;S} \downarrow$.*

Demostración. Para el punto 1, consideremos que por la definición de $A-$cómputo, al ser una sucesión finita de descripciones instantáneas, solo una cantidad finita de pasos pudieron haber sido consultas al oráculo A y, por lo tanto, solo se consulta un subconjunto finito de A. Para el punto 2, por un argumento similar al anterior, en un determinado cómputo solo se necesita consultar información del conjunto finito s, por lo tanto, todo conjunto que lo contenga contará con la misma información, y el cómputo procederá como lo haría con el conjunto s. □

Ahora podemos definir la clase de funciones A-computables, es decir, las funciones que son computables por Máquinas de Turing con oráculo A.

Definición 2.1.21. Sea $n \in \omega$. Una función parcial n-aria $f : \omega^n \to \omega$ es *A-computable* si existe una Máquina de Turing Z tal que para todo $(m_1, \ldots, m_n) \in \mathrm{Dom}(f) = \mathrm{Dom}(\Psi^{(n)}_{Z:A})$

$$f(m_1, \ldots, m_n) = \Psi^{(n)}_{Z:A}(m_1, \ldots, m_n).$$

En este caso, decimos que Z *A-computa* a f. ♡

Ahora veamos algunas relaciones entre la A-computabilidad y la computabilidad.

Corolario 2.1.5.1. *Si f es computable, entonces es A-computable para cualquier conjunto A.*

Demostración. Se sigue inmediatamente de la definición de A-computo puesto que todo cómputo es un A-cómputo para cualquier A. □

Teorema 2.1.6. *Para cada Máquina de Turing Z existe una Máquina de Turing simple Z' tal que*

$$\Psi^{(n)}_{Z'}(x_1, x_2, \ldots, x_n) = \Psi^{(n)}_{Z:\emptyset}(x_1, x_2, \ldots, x_n).$$

Demostración. Sea Z una Máquina de Turing. Como la función característica del conjunto vacío es la constante 0 podemos suponer que la respuesta a la pregunta que realizaría Z al oráculo es optar por q_l, por lo que podemos definir a Z' como la Máquina de Turing obtenida al reemplazar cada cuádrupla que tenga la forma $q_i S_j q_k q_l$ por la cuádrupla $q_i S_j S_j q_l$, y tendremos el resultado. □

Corolario 2.1.6.1. *Si una función f es $\emptyset$-computable, entonces es computable.*

Demostración. Usando el teorema anterior, podemos obtener una Máquina de Turing simple que computa a f. □

Teorema 2.1.7. *Una función f es computable si y sólo si es $\emptyset$-computable.*

Demostración. El resultado se obtiene de los corolarios anteriores. □

Esto quiere decir que para cualquier teorema que se pruebe para la A-computabilidad, podemos obtener un teorema para la computabilidad tomando a A como el conjunto vacío. Es decir, la computabilidad es un caso particular de computabilidad relativa.

Definición 2.1.22. Decimos que un conjunto S es *computable* (respectivamente *A-computable*) si su función característica $C_S(x)$ es computable (resp A-computable). ♡

Teorema 2.1.8. *A es A-computable.*

Antes de demostrar el teorema realicemos, un ejercicio mental muy común en el ámbito de la programación que consiste en imaginar cómo realizaríamos el proceso que queremos que la máquina realice si tuviéramos que hacerlo a mano.
En este caso, el proceso que se desea realizar es devolver un 1, si es que el número que tenemos como entrada de la máquina pertenece al conjunto A, o un 0, si el número no pertenece, esto quiere decir que, suponiendo que la entrada es el número n, se nos presentará una cinta con $n+1$ símbolos 1 escritos y al final del procedimiento debe haber escrito en la cinta un solo símbolo 1, si es que el número n pertenece a A, o ningún símbolo 1, en caso contrario, todo esto dentro del contexto de las operaciones permitidas para las Máquinas de Turing.

Una manera de llevar a cabo esto podría ser la siguiente:

Paso 1 Borrar un símbolo 1. Considerando que las MT solo tiene permitido preguntar al oráculo acerca de la pertenencia del número representado por la cantidad de símbolos 1 escritos en la cinta, es necesario borrar uno de ellos antes de realizar la pregunta.

Paso 2 Realizar la consulta de pertenencia de n en A. En caso de que n sí pertenezca a A, seguir al paso 3; en caso contrario, seguir al paso 4.

Paso 3 Tenemos dos casos:

- **Paso 3a** Si el símbolo leído es un 1, moverse un espacio a la derecha. De esta manera ese símbolo 1 no será parte del proceso del siguiente paso.
- **Paso 3b** Si el símbolo leído es B, escribir un 1 y moverse un espacio a la derecha.

Paso 4 Borrar los símbolos 1 a la derecha de la casilla actual hasta encontrar el primer símbolo B. Así, al final de este paso, en caso de que n pertenezca a A, en la cinta estará escrito el 1 que saltamos en el paso 3 y, en caso contrario, no habrá ningún símbolo 1 en la cinta.

Paso 5 Una vez encontrado el primer símbolo B, terminar.

Al terminar esta serie de pasos, tendremos como resultado la salida esperada. Ahora revisemos como realizar esto formalmente:

Demostración. Sea M la Máquina de Turing que contiene las siguientes cuádruplas:

a) $q_1 1 B q_2$
b) $q_2 B q_3 q_5$
c) $q_3 1 R q_4$
d) $q_3 B 1 q_3$
e) $q_4 1 B q_5$
f) $q_5 B R q_4$

Ahora veamos cómo se comporta esta máquina. Sea $n \in \omega$. Alimentando con n a la máquina, tenemos que la descripción instantánea al momento de iniciar es $q_1 1^{n+1}$ entonces:

$$q_1 1^{n+1} \to q_2 B 1^n \qquad \text{por a)}$$

Ahora, tenemos dos casos, supongamos que $n \in A$.
Si $n = 0$, la siguiente descripción instantánea es $q_2 B$, y entonces:

$$q_2 B \underset{A}{\to} q_3 B \qquad \text{por b)}$$
$$q_3 B \to q_3 1 \qquad \text{por d)}$$
$$q_3 1 \to 1 q_4 B \qquad \text{por c)}$$

Luego, como no existe ninguna cuádrupla que inicie con $q_4 B$, la descripción instantánea $1 q_4 B$ es final. Además, por el Teorema 2.1.4 , tenemos que $1 q_4 B$ es terminal con respecto a Z y, por lo tanto, $\text{Res}_Z^A(q_1 1^{n+1}) = 1 q_4 B$, lo que significa que $\langle \text{Res}_Z^A(q_1 \overline{n}) \rangle = 1$.
Por otro lado, si $n > 0$, entonces la siguiente descripción instantánea es $q_2 B 1^n$, y entonces:

$$q_2 B 1^n \underset{A}{\to} q_3 B 1^n \qquad \text{por b)}$$
$$q_3 B 1^n \to q_3 1 1^n \qquad \text{por d)}$$
$$q_3 1 1^n \to 1 q_4 1^n \qquad \text{por c)}$$
$$1 q_4 1^n \to 1 q_5 B 1^{n-1} \qquad \text{por e)}$$

$$
\begin{aligned}
1q_5B1^{n-1} &\to 1Bq_41^{n-1} && \text{por f)}\\
1Bq_41^{n-1} &\to 1Bq_5B1^{n-2} && \text{por e)}\\
1Bq_5B1^{n-2} &\to 1BBq_41^{n-2} && \text{por f)}\\
&\dots\\
1B^{n-1}q_41 &\to 1B^{n-1}q_5B && \text{por e)}\\
1B^{n-1}q_5B &\to 1B^nq_4B && \text{por f)}
\end{aligned}
$$

Análogamente al caso anterior, resulta que $\mathrm{Res}_Z^A(q_1 1^{n+1}) = 1B^nq_4B$, lo que significa que $\langle \mathrm{Res}_Z^A(q_1\overline{n})\rangle = 1$.

Supongamos ahora que $n \notin A$

$$
\begin{aligned}
q_2B1^n &\underset{A}{\to} q_5B1^n && \text{por b)}\\
q_5B1^n &\to Bq_41^n && \text{por f)}\\
Bq_41^{n-1} &\to Bq_5B1^{n-1} && \text{por e)}\\
Bq_5B1^{n-1} &\to BBq_41^{n-1} && \text{por f)}\\
BBq_41^{n-1} &\to BBq_5B1^{n-2} && \text{por e)}\\
BBq_5B1^{n-2} &\to BBBq_41^{n-2} && \text{por f)}\\
&\dots\\
BB^{n-1}q_41 &\to BB^{n-1}q_5B && \text{por e)}\\
BB^{n-1}q_5B &\to BB^nq_4B && \text{por f)}
\end{aligned}
$$

Análogamente a los casos anteriores, resulta que $\mathrm{Res}_Z^A(q_1 1^{n+1}) = B^{n+1}q_4B$, lo que significa que $\langle \mathrm{Res}_Z^A(q_1\overline{n})\rangle = 0$. Por lo tanto, $\Psi_M^A(n) = 0$, si $n \notin A$, y $\Psi_M^A(n) = 1$, si $n \in A$. □

2.2. Funciones computables

Esta sección servirá para mostrar ejemplos de funciones computables así como para desarrollar algunas propiedades de las mismas.

2.2.1. Algunas funciones computables notables

En esta sección presentaremos algunas funciones notables que son computables, las cuales nos serán útiles en las siguientes secciones.

Ejemplo 1. La función *sucesor*, i.e. $f(n) = n + 1$

Demostración. Consideremos la MT consistente de la cuádrupla q_1BBq_2. Alimentando a la máquina con el número de n, tenemos que la descripción instantánea al momento de iniciar es $q_1 1^{n+1}$, que es terminal. Entonces $\text{Res}_Z(q_1 1^{n+1}) = q_1 1^{n+1}$, lo que significa que $\langle \text{Res}_Z(q_1 1^{n+1}) \rangle = n + 1$. Por lo tanto, $\Psi_Z(n) = n + 1$ □

Ejemplo 2. La función *constante* 0, i.e. $f(n) = 0$.

Demostración. Consideremos la MT que consiste del conjunto de cuádruplas:

a) $q_1 1 B q_2$

b) $q_2 B R q_1$

Alimentando a la máquina con el numeral de n, tenemos que la descripción instantánea al momento de iniciar es $q_1 1^{n+1}$, por lo tanto:

$$
\begin{aligned}
q_1 1^{n+1} &\to q_2 B 1^n && \text{por a)} \\
q_2 B 1^n &\to B q_1 1^n && \text{por b)} \\
B q_1 1^n &\to B q_2 B 1^{n-1} && \text{por a)} \\
B q_2 B 1^{n-1} &\to B B q_1 1^{n-1} && \text{por b)} \\
&\dots \\
B^n q_1 1 &\to B^{n-1} q_2 B && \text{por a)} \\
B^{n-1} q_2 B &\to B^n q_1 B && \text{por b)}
\end{aligned}
$$

Luego, no hay ninguna cuádrupla que inicie con el par de símbolos $q_1 B$ por lo que la expresión instantánea $B^n q_1 B$ es final y por lo tanto, terminal. Además, $\langle B^n q_1 B \rangle = 0$, por lo tanto, para todo n tenemos que $\Psi_Z(n) = 0$. □

Ejemplo 3. $U_i^n(x_1, \dots x_n) = x_i$, $1 \leq i \leq n$ o la *i-ésima proyección.*

Demostración. Sea $1 \leq i \leq n$. Probaremos que la i-ésima proyección es computable. Consideremos la MT que consiste del conjunto de cuádruplas:

a) $q_i 1 B q_i$ c) $q_{n+i+1} 1 R q_{n+i+1}$

b) $q_i B R q_{n+i+1}$ d) $q_{n+i+1} B R q_{i+1}$

para toda $j \neq i$, tal que $1 \leq j \leq n$, .

e) $q_j 1 B q_{n+j+1}$ f) $q_j B R q_{j+1}$ g) $q_{n+j+1} B R q_j$

Alimentando a la máquina con el numeral de $(k_1, k_2, \ldots, k_n)$, es decir, la descripción instantánea al momento de iniciar es $q_1\overline{(k_1, k_2, \ldots, k_n)}$, entonces, si $i = 1$:

$$q_1\overline{(k_1, k_2, \ldots, k_n)} \to q_1 B\overline{(k_1 - 1, k_2, \ldots, k_n)} \quad \text{por a)}$$

$$q_1 B\overline{(k_1 - 1, k_2, \ldots, k_n)} \to B q_{n+2}\overline{(k_1 - 1, k_2, \ldots, k_n)} \quad \text{por b)}$$

$$B q_{n+2}\overline{(k_1 - 1, k_2, \ldots, k_n)} \to B 1 q_{n+2}\overline{(k_1 - 2, k_2, \ldots, k_n)} \quad \text{por c)}$$

$$B 1 q_{n+2}\overline{(k_1 - 1, k_2, \ldots, k_n)} \to B 1^2 q_{n+2}\overline{(k_1 - 3, k_2, \ldots, k_n)} \quad \text{por c)}$$

$$\ldots$$

$$B 1^{k_1} q_{n+2} B\overline{(k_2, \ldots, k_n)} \to B 1^{k_1} B q_2\overline{(k_2, \ldots, k_n)} \quad \text{por d)}$$

$$B 1^{k_1} B q_2\overline{(k_2, \ldots, k_n)} \to B 1^{k_1} B q_{n+3} B\overline{(k_2 - 1, \ldots, k_n)} \quad \text{por e con } j = 2)$$

$$B 1^{k_1} B q_{n+3} B\overline{(k_2 - 1, \ldots, k_n)} \to B 1^{k_1} B B q_2\overline{(k_2 - 1, \ldots, k_n)} \quad \text{por g con } j = 2)$$

$$B 1^{k_1} B B q_2\overline{(k_2 - 1, \ldots, k_n)} \to B 1^{k_1} B B q_{n+3} B\overline{(k_2 - 2, \ldots, k_n)} \quad \text{por e con } j = 2)$$

$$B 1^{k_1} B B q_{n+3} B\overline{(k_2 - 2, \ldots, k_n)} \to B 1^{k_1} B B^2 q_{n+3}\overline{(k_2 - 3, \ldots, k_n)} \quad \text{por g con } j = 2)$$

$$\ldots$$

$$B 1^{k_1} B B^{k_2+1} q_2 B\overline{(k_3, \ldots, k_n)} \to B 1^{k_1} B B^{k_2+1} B q_3\overline{(k_3, \ldots, k_n)} \quad \text{por f con } j = 2)$$

$$\ldots$$

$$B 1^{k_1} B B^{k_2+1} \ldots B^{k_n+1} q_n B \to B 1^{k_1} B B^{k_2+1} \ldots B^{k_n+1} B q_{n+1} B \quad \text{por f con } j = n)$$

Luego, como ninguna cuádrupla comienza con q_{n+1}, tenemos que la expresión $B 1^{k_1} B B^{k_2+1} \ldots B^{k_n+1} B q_{n+1} B$ es final y, por lo tanto, terminal. Así, tenemos que $\langle B 1^{k_1} B B^{k_2+1} \ldots B^{k_n+1} B q_{n+1} B \rangle = \langle 1^{k_1} \rangle = k_1$. La prueba formal para $i > 1$ se puede consultar en Anexo I. □

2.2.2. Operaciones con Máquinas de Turing

De la misma manera en que es posible componer funciones usando la salida de una función para alimentar otra, es posible componer MT usando la expresión de cinta del resultado de una MT como entrada para otra. Necesitamos que estas máquinas se comporten de cierta manera, en particular que presenten sus resultados de forma que otra máquina pueda comenzar a trabajar con ellos.

Definición 2.2.1. Sea Z una MT. Entonces $\theta(Z)$ es el mayor número i tal que q_i es un símbolo de estado que aparece en Z. ♡

Definición 2.2.2. Sea Z una MT. Decimos que Z es *n-regular* con $n > 0$ si se cumple lo siguiente:

1. Existe una $s > 0$ tal que si $\text{Res}_Z^A[q_1(\overline{m_1, \ldots, m_n})]$ está definido, entonces este resultado tiene la forma $q_{\theta(Z)}\overline{(r_1, \ldots, r_s)}$ para algunas $r_1, \ldots, r_s$, y
2. No existe ninguna cuádrupla en Z que empiece con $q_{\theta(Z)}$. ♡

Se puede notar que las Máquinas de Turing n-regulares, presentan sus resultados de manera parecida a cómo las Máquinas de Turing reciben sus entradas, esto es importante debido a que nos permitirá relacionar las máquinas de nuevas formas.

Ahora presentamos algunos resultados que nos permitirán generar Máquinas de Turing a partir de otras conocidas.

Teorema 2.2.1. *Sea $W = \{S_0 = B, S_1 = 1, S_2, \ldots S_m\}$ un conjunto de símbolos. Entonces existe una Máquina de Turing Z_C tal que si $w = S_i S_{u_1} \ldots S_{u_k} S_j$ es una expresión consistente de símbolos de W, donde $S_{u_l} \neq S_i$, $S_{u_l} \neq S_j$, $S_i \neq S_j$, con $1 \leq l \leq k$ y, además, $S_i \neq 1 \neq S_j$, $S_i \neq B \neq S_j$, se tiene que:*

$$Res_{Z_C} q_1(w) = q_{\theta(Z_C)} w \overline{\langle w \rangle}.$$

Demostración. La idea básica de esta máquina será ir buscando uno por uno los símbolos 1 que aparezcan, sustituirlo con un símbolo especial y escribir un uno al final de la cinta. Repitiendo este proceso hasta llegar al símbolo S_j y luego regresando los símbolos especiales a símbolos 1, tendremos la salida esperada. Sea $p = m + 1$.

Consideremos la siguiente máquina, Z_C:

Para $1 \leq l \leq k$:

a) $q_1 S_i R q_2$
b) $q_2 S_{u_l} R q_2$
c) $q_2 1 S_p q_3$
d) $q_2 S_j R q_4$
e) $q_3 S_l R q_3$
f) $q_3 S_p R q_3$
g) $q_3 S_1 R q_3$
h) $q_3 S_j R q_5$
i) $q_5 1 R q_5$
j) $q_5 B 1 q_6$
k) $q_6 1 L q_6$
l) $q_6 S_j L q_6$
m) $q_6 S_l L q_6$
n) $q_6 B L q_6$
ñ) $q_6 S_p R q_2$
o) $q_4 1 R q_4$
p) $q_4 B 1 q_7$
q) $q_7 1 L q_7$
r) $q_7 S_j L q_7$
s) $q_7 S_l L q_7$
t) $q_7 S_p 1 q_7$
u) $q_7 S_i S_i q_8$

Notemos que las cuádruplas b, e, m y s representan cada una, una cantidad k de cuádruplas, una por cada valor de l. Ahora procederemos formalmente, supongamos entonces que $w = S_i S_{u_1} \dots S_{u_k} S_j$ es la entrada de la máquina Z_C. Entonces la descripción instantánea al momento de iniciar es $q_1 S_i S_{u_1} \dots S_{u_k} S_j$, por lo que:

$$q_1 S_i S_{u_1} \dots S_{u_k} S_j \to S_i q_2 S_{u_1} \dots S_{u_k} S_j \qquad \text{por a)}$$

Luego, si $u_1, u_2, \dots u_l$ son distintos de 1:

$$S_i S_{u_1} \dots q_2 S_{u_l} \dots S_{u_k} S_j \to S_i S_{u_1} \dots S_{u_l} q_2 S_{u_{l+1}} \dots S_{u_k} S_j \qquad \text{por b)}$$

Y para $S_{u_l} = 1$:

$$S_i S_{u_1} \dots q_2 S_1 \dots S_{u_k} S_j \to S_i S_{u_1} \dots q_3 S_p \dots S_{u_k} S_j \qquad \text{por c)}$$

$$S_i S_{u_1} \dots q_3 S_p \dots S_{u_k} S_j \to S_i S_{u_1} \dots S_p q_3 S_{u_{l+1}} \dots S_{u_k} S_j \qquad \text{por f)}$$

$$S_i S_{u_1} \dots S_p q_3 S_{u_{l+1}} \dots S_{u_k} S_j \to S_i S_{u_1} \dots S_p S_{u_{l+1}} q_3 S_{u_{l+2}} \dots S_{u_k} S_j \qquad \text{por e o g)}$$

$$\dots$$

$$S_i S_{u_1} \dots S_p \dots S_{u_k} q_3 S_j \to S_i S_{u_1} \dots S_p \dots S_{u_k} S_j q_5 B \qquad \text{por h)}$$

$$S_i S_{u_1} \dots S_p \dots S_{u_k} S_j q_5 B \to S_i S_{u_1} \dots S_p \dots S_{u_k} S_j q_6 1 \qquad \text{por j)}$$

$$S_i S_{u_1} \dots S_p \dots S_{u_k} S_j q_6 1 \to S_i S_{u_1} \dots S_p \dots S_{u_k} q_6 S_j 1 \qquad \text{por k)}$$

$$S_i S_{u_1} \dots S_p \dots S_{u_k} q_6 S_j 1 \to S_i S_{u_1} \dots S_p \dots q_6 S_{u_k} S_j 1 \qquad \text{por l, m o n)}$$

$$\vdots$$

$$S_iS_{u_1}\dots q_6S_p\dots S_{u_k}S_j1 \to S_iS_{u_1}\dots S_pq_2\dots S_{u_k}S_j1 \qquad \text{por ñ)}$$

Se reinicia la búsqueda por un símbolo 1 y se realiza todo el proceso de nuevo por cada símbolo 1 que aparezca:

$$S_iS_{u_1}\dots q_6S_p\dots S_{u_k}S_j1^{\langle w\rangle} \to S_iS_{u_1}\dots S_pq_2\dots S_{u_k}S_j1^{\langle w\rangle} \qquad \text{por ñ)}$$

$$\vdots$$

$$S_iS_{u_1}\dots q_6S_{u_k}S_j1^{\langle w\rangle} \to S_iS_{u_1}\dots S_{u_k}q_2S_j1^{\langle w\rangle} \qquad \text{por e o g)}$$

$$S_iS_{u_1}\dots S_{u_k}q_2S_j1^{\langle w\rangle} \to S_iS_{u_1}\dots S_{u_k}S_jq_41^{\langle w\rangle} \qquad \text{por d)}$$

$$S_iS_{u_1}\dots S_{u_k}S_jq_41^{\langle w\rangle} \to S_iS_{u_1}\dots S_{u_k}S_jq_411^{\langle w\rangle-1} \qquad \text{por o)}$$

$$\vdots$$

$$S_iS_{u_1}\dots S_{u_k}S_jq_411^{\langle w\rangle-1} \to S_iS_{u_1}\dots S_{u_k}S_j1^{\langle w\rangle}q_4B \qquad \text{por o)}$$

$$S_iS_{u_1}\dots S_{u_k}S_j1^{\langle w\rangle}q_4B \to S_iS_{u_1}\dots S_{u_k}S_j1^{\langle w\rangle}q_71 \qquad \text{por p)}$$

Ahora se inicia el proceso de regresar los símbolos S_p a símbolos 1 para dejar la cadena como estaba inicialmente:

$$S_iS_{u_1}\dots S_{u_k}S_j1^{\langle w\rangle}q_71 \to S_iS_{u_1}\dots S_{u_k}S_j1^{\langle w\rangle-1}q_711^2 \qquad \text{por q)}$$

$$\vdots$$

$$S_iS_{u_1}\dots S_{u_k}S_jq_71^{\langle w\rangle+1} \to S_iS_{u_1}\dots S_{u_k}q_7S_j1^{\langle w\rangle+1} \qquad \text{por q)}$$

$$S_iS_{u_1}\dots S_{u_k}q_7S_j1^{\langle w\rangle+1} \to S_iS_{u_1}\dots q_7S_{u_k}S_j1^{\langle w\rangle+1} \qquad \text{por r)}$$

$$\vdots$$

$$S_iS_{u_1}\dots S_pq_7S_{u_l}\dots S_{u_k}1^{\langle w\rangle+1} \to S_iS_{u_1}\dots q_7S_pS_{u_l}\dots S_{u_k}1^{\langle w\rangle+1} \qquad \text{por s)}$$

$$S_iS_{u_1}\dots q_7S_pS_{u_l}\dots S_{u_k}1^{\langle w\rangle+1} \to S_iS_{u_1}\dots q_71S_{u_l}\dots S_{u_k}1^{\langle w\rangle+1} \qquad \text{por t)}$$

$$S_iS_{u_1}\dots q_7S_{u_l-2}1S_{u_l}\dots S_{u_k}1^{\langle w\rangle+1} \to S_iS_{u_1}\dots q_71S_{u_l}\dots S_{u_k}1^{\langle w\rangle+1} \qquad \text{por t)}$$

$$\vdots$$

$$S_iq_7S_{u_1}\dots 1S_{u_l}\dots S_{u_k}1^{\langle w\rangle+1} \to q_7S_iS_{u_1}\dots 1S_{u_l}\dots S_{u_k}1^{\langle w\rangle+1} \qquad \text{por t)}$$

$$q_7S_iS_{u_1}\dots 1S_{u_l}\dots S_{u_k}1^{\langle w\rangle+1} \to q_8S_iS_{u_1}\dots 1S_{u_l}\dots S_{u_k}1^{\langle w\rangle+1} \qquad \text{por t)}$$

Luego, como ninguna cuádrupla de Z_C comienza con el estado q_8 esta descripción instantánea es final y se tiene el resultado. □

Definición 2.2.3. Sea Z una MT. Definimos $Z^{(n)}$ como la MT que se obtiene de Z al reemplazar cada estado interno q_i y todas sus ocurrencias en las cuádruplas por q_{n+i}. ♡

Teorema 2.2.2. *Para cada MT Z, existe una MT Z' tal que, para cada n, Z' es n-regular y además se tiene que:*

$$Res^{A}_{Z'}(q_1\overline{(m_1,\ldots,m_n)}) = q_{\theta(Z')}\overline{\Psi^{(n)}_{Z;A}(m_1,\ldots,m_n)}$$

Demostración. La prueba seguirá la siguiente idea: si Z termina en algún momento, entonces un proceso adicional realizará un procedimiento similar al de la máquina del Teorema 2.2.1 excepto por el hecho de que cuando se termine de contar la cantidad de unos en la máquina, se borrarán los símbolos que representan el proceso de Z. Un riesgo que se corre con este proceso es el de que nunca se puede estar seguro de cuándo se ha llegado al final de los símbolos escritos por Z. Esto se resolverá "acotando" la sección de la cinta usada por Z para llevar a cabo su cómputo, esto usando dos símbolos especiales como marcadores de inicio y fin respectivamente.

Formalmente:

Antes de iniciar el proceso de Z necesitamos posicionar los símbolos que serviran como cotas del segmento de cinta que se discutió antes. Por lo tanto, sean S_x y S_y los primeros dos símbolos que no aparecen en el alfabeto de Z y consideremos la siguiente MT auxiliar Z_A:

a) $q_1 1 L q_2$
b) $q_2 B S_x q_2$
c) $q_2 S_x R q_2$
d) $q_2 1 R q_2$
e) $q_2 B R q_3$
f) $q_3 1 1 q_2$
g) $q_3 B L q_4$
h) $q_4 B S_y q_5$
i) $q_5 S_y L q_5$
j) $q_5 1 L q_5$
k) $q_5 B L q_5$
l) $q_5 S_x R q_6$

Definiremos $Z_B = Z_A \bigcup Z^{(5)}$ Ahora definiremos una MT que realizará el mismo proceso que Z salvo por el hecho de que, cuando se encuentre con los símbolos S_x o

S_y, los desplazará a la izquierda o a la derecha respectivamente de tal manera que el procesamiento se mantenga dentro de los marcadores que hemos definido. Esto de la siguiente manera:

Sea $I = \{j \mid q_j \text{ es un estado interno de } Z^{(5)}\}$, sea $p = \theta(Z^{(5)})$ definimos para cada $i \in I$ la siguiente Máquina de Turing, Z_i:

a) $q_i S_x L q_{p+i}$
b) $q_{p+i} B S_x q_{p+i}$
c) $q_{p+i} S_x R q_{2p+i}$
d) $q_{2p+i} S_x B q_i$
e) $q_i S_y R q_{3p+i}$
f) $q_{3p+i} B S_y q_{p+i}$
g) $q_{p+i} S_y L q_{3p+i}$
h) $q_{3p+i} S_y B q_i$

Definimos $Z_C = Z_B \bigcup_{i \in I} Z_i$.

Una vez que se haya terminado el proceso de Z con la particularidad de que se lleva a cabo entre los símbolos S_x y S_y, se debe proceder a ordenar los símbolos 1 que aparezcan en la cinta.

Consideremos entonces el siguiente conjunto Fin $= \{(q_i, S_j) \mid q_i$ es un estado interno de Z_C, S_j pertenece al alfabeto de Z_C y ademas, ninguna cuádrupla de Z_C comienza con $q_i S_j\}$. Luego, consideremos la siguiente MT, Z_t, para cada $(q_i, S_j) \in$ Fin, tenemos que $q_i S_j S_j q_{4p+1} \in Z_t$.

Por lo que, si $Z_D = Z_C \bigcup Z_t$ y si $S_x P q_i Q S_y$ es final con respecto a Z_C, tenemos que $S_x P q_i Q S_y \twoheadrightarrow S_x P q_{4p+1} Q S_y\,(Z_D)$. [1]

Sea $S = \{j \mid S_j$ está en el alfabeto de $Z_C \,\&\, S_x \neq S_j \neq S_y\}$. Entonces definimos para cada S_k con $k \in S$:

1. $q_{4p+1} S_k L q_{4p+1}$
2. $q_{4p+1} S_x L q_{4p+2}$

Y definimos $Z_S = \bigcup_{k \in S} \{q_{4p+1} S_k L q_{4p+1}\} \bigcup \{q_{4p+1} S_x L q_{4p+2}\}$. Luego, por el Teorema 2.2.1, existe una Máquina de Turing Z_{con} tal que

$$\text{Res}_{Z_{\text{con}}} q_1(S_x P Q S_y) = q_{\theta(Z_{\text{con}})} S_x P Q S_y \overline{\langle S_x P Q S_y \rangle}.$$

Definimos entonces $Z_E = Z_D \bigcup Z_S \bigcup Z_{\text{con}}^{(4p+1)}$.

[1] El símbolo $\twoheadrightarrow$ representa el proceso completo de la máquina Z_D sobre la expresión instantánea de la izquierda que da por resultado la de la derecha.

Y por último, definimos Z_{bor} conteniendo las siguientes cuádruplas, sea $r = \theta(Z_E)$:

a) $q_r S_k B q_r$
b) $q_r S_x B q_r$
c) $q_r B R q_r$
d) $q_r S_y B q_{r+1}$
e) $q_{r+1} B R q_{r+2}$

y entonces definimos $Z' = Z_E \cup Z_{\text{bor}}$.

Ahora procedemos a analizar el comportamiento de la máquina Z':

Sea $(m_1, m_2, \ldots m_n)$ la entrada de la máquina Z'. Entonces la expresión instantánea inicial será $q_1\overline{(m_1, \ldots, m_n)}$ y tenemos:

$$q_1\overline{(m_1,\ldots,m_n)} \to q_2 B\overline{(m_1,\ldots,m_n)} \quad \text{por a)}$$
$$q_2 B\overline{(m_1,\ldots,m_n)} \to q_2 S_x\overline{(m_1,\ldots,m_n)} \quad \text{por b)}$$
$$q_2 S_x\overline{(m_1,\ldots,m_n)} \to S_x q_2 1\overline{(m_1-1,\ldots,m_n)} \quad \text{por c)}$$
$$S_x q_2 1\overline{(m_1-1,\ldots,m_n)} \to S_x 1 q_2 1\overline{(m_1-2,\ldots,m_n)} \quad \text{por d)}$$
$$\ldots$$
$$S_x 1^{m_1+1} q_2 B\overline{(m_2,\ldots,m_n)} \to S_x 1^{m_1+1} B q_3\overline{(m_2,\ldots,m_n)} \quad \text{por e)}$$
$$S_x 1^{m_1+1} B q_3\overline{(m_2,\ldots,m_n)} \to S_x 1^{m_1+1} B q_2 1\overline{(m_2-1,\ldots,m_n)} \quad \text{por f)}$$
$$\ldots$$
$$S_x\overline{(m_2,\ldots,m_n)} q_2 B \to S_x\overline{(m_2,\ldots,m_n)} B q_3 B \quad \text{por e)}$$
$$S_x\overline{(m_2,\ldots,m_n)} B q_3 B \to S_x\overline{(m_2,\ldots,m_n)} q_4 B B \quad \text{por g)}$$
$$S_x\overline{(m_2,\ldots,m_n)} q_4 B B \to S_x\overline{(m_2,\ldots,m_n)} q_5 S_y B \quad \text{por h)}$$
$$S_x\overline{(m_2,\ldots,m_n)} q_5 S_y B \to S_x\overline{(m_2,\ldots,m_n-1)} q_5 1 S_y B \quad \text{por i)}$$
$$S_x\overline{(m_2,\ldots,m_n-1)} q_5 1 S_y B \to S_x\overline{(m_2,\ldots,m_n-2)} q_5 11 S_y B \quad \text{por j)}$$
$$\ldots$$
$$S_x\overline{(m_2,\ldots,m_{n-1})} B q_5 1 1^{m_n} S_y \to S_x\overline{(m_2,\ldots,m_{n-1})} q_5 B 1^{m_n+1} S_y \quad \text{por j)}$$
$$S_x\overline{(m_2,\ldots,m_{n-1})} q_5 B 1^{m_n+1} S_y \to S_x\overline{(m_2,\ldots,m_{n-1}-1)} q_5 1 B 1^{m_n+1} S_y \quad \text{por k)}$$
$$\ldots$$
$$S_x q_5 1\overline{(m_1-1,\ldots,m_n)} S_y \to q_5 S_x\overline{(m_1,\ldots,m_n)} S_y \quad \text{por j)}$$

$$q_5 S_x \overline{(m_1, \ldots, m_n)} S_y \to S_x q_6 \overline{(m_1, \ldots, m_n)} S_y \qquad \text{por l)}$$

Lo cual termina el proceso de la máquina auxiliar Z_A. Luego, como Z' tiene como uniendo a $Z^{(5)}$, la cual tiene como estado inicial a q_6, se sigue que en este momento se inicia el proceso de $Z^{(5)}$.

Supongamos entonces que en algún momento del procesamiento de Z el cabezal se mueve a la extrema izquierda de la cinta, es decir, llega al momento de leer S_x, entonces tendríamos una descripción instantánea del tipo $q_j S_x P S_y$, donde q_j es una configuración interna de $Z^{(5)}$ y P es una cadena de símbolos del alfabeto de $Z^{(5)}$.

Por lo tanto por la máquina Z_j:

$$q_j S_x P S_y \to q_{p+j} B S_x P S_y \qquad \text{por a)}$$

$$q_{p+j} B S_x P S_y \to q_{p+j} S_x S_x P S_y \qquad \text{por b)}$$

$$q_{p+j} S_x S_x P S_y \to S_x q_{2p+j} S_x P S_y \qquad \text{por c)}$$

$$S_x q_{2p+j} S_x P S_y \to S_x q_j B P S_y \qquad \text{por d)}$$

Con lo cual el proceso de $Z^{(5)}$ puede continuar de manera normal.

Análogamente, supongamos que el cabezal se mueve a la extrema derecha, llegando a leer S_y tenemos entonces:

$$S_x P q_j S_y \to S_x P S_y q_{3p+j} B \qquad \text{por e)}$$

$$S_x P S_y q_{3p+j} B \to S_x P S_y q_{p+j} S_y \qquad \text{por f)}$$

$$S_x P S_y q_{p+j} S_y \to S_x P q_{3p+j} S_y S_y \qquad \text{por g)}$$

$$S_x P q_{3p+j} S_y S_y \to S_x P q_j B S_y \qquad \text{por h)}$$

Con lo cual el proceso de $Z^{(5)}$ puede continuar de manera normal.

Ahora supongamos que el proceso de $Z^{(5)}$ se detiene en algún momento con estado terminal q_l, es decir, la descripción instantánea al terminar el proceso de $Z^{(5)}$ sería de la forma $S_x P q_l Q S_y$ donde P y Q son cadenas de símbolos que pertenecen al alfabeto de $Z^{(5)}$.

Luego, por Z_t (y por lo tanto por Z') tenemos que $S_x P q_l Q S_y \to S_x P q_{4p+1} Q S_y$. Entonces ahora por el proceso de Z_S (y por lo tanto, de Z') se sigue que $S_x P q_{4p+1} Q S_y \twoheadrightarrow q_{4p+2} S_x P Q S_y (Z_S)$.

Después, como $Z_{con}^{(4p+1)}$ es un uniendo de Z', tenemos que se inicia el proceso de Z_{con}, por lo que se sigue que: $q_{4p+2}S_xPQS_y \twoheadrightarrow q_rS_xPQS_y\overline{\langle S_xPQS_y\rangle}(Z')$.

Por último, se ejecuta el proceso de borrado, por lo que se sigue que:

$$\begin{aligned}
q_rS_xPQS_y\overline{\langle S_xPQS_y\rangle} &\to q_rBPQS_y\overline{\langle S_xPQS_y\rangle} && \text{por b)}\\
q_rBPQS_y\overline{\langle S_xPQS_y\rangle} &\to Bq_rS_kRQS_y\overline{\langle S_xPQS_y\rangle} && \text{por c)}\\
Bq_rS_kRQS_y\overline{\langle S_xPQS_y\rangle} &\to Bq_rBRQS_y\overline{\langle S_xPQS_y\rangle} && \text{por a)}\\
&\dots\\
Bq_rBS_y\overline{\langle S_xPQS_y\rangle} &\to BBq_rS_y\overline{\langle S_xPQS_y\rangle} && \text{por c)}\\
BBq_rS_y\overline{\langle S_xPQS_y\rangle} &\to BBq_{r+1}B\overline{\langle S_xPQS_y\rangle} && \text{por d)}\\
BBq_{r+1}B\overline{\langle S_xPQS_y\rangle} &\to BBBq_{r+2}\overline{\langle S_xPQS_y\rangle} && \text{por e)}
\end{aligned}$$

Ahora, por la forma en la que se seleccionó r, la configuración interna q_{r+2} es final y, por lo tanto, el resultado del cómputo es: $q_{r+2}\overline{\langle S_xPQS_y\rangle}$, lo cual concluye la prueba. □

Teorema 2.2.3. *Para cada MT n-regular Z existe una MT n-regular Z_t tal que si w es una expresión y S_x y S_y son símbolos que no pertenecen a w ni al alfabeto de Z y:*

$$Res_Z^A(q_1\overline{(m_1,\dots,m_n)}) = q_{\theta(Z)}\overline{(r_1,\dots,r_s)},$$

entonces se tiene que también:

$$Res_{Z_t}^A(q_1S_xwS_y\overline{(m_1,\dots,m_n)}) = q_{\theta(Z_t)}S_xwS_y\overline{(r_1,\dots,r_s)}.$$

Además, si se tiene que $Res_Z^A(q_1\overline{(m_1,\dots,m_n)})$ no está definido, entonces $Res_{Z_p}^A(q_1S_xwS_y\overline{(m_1,\dots,m_n)})$ tampoco lo está.

Demostración. La idea de la prueba será considerar a la expresión conformada por S_xwS_y como un solo bloque que no será corrompido por el cómputo de Z sino que cada vez que se necesite más espacio a la "izquierda", se moverá todo el bloque completo para poder seguir con el cómputo. Formalmente:

Sean

- $S_i \in \{S_j \in w | S_j \neq B\}$,
- $p = \theta(Z^{(2)}) + 1$,
- $w = S_{w_1} \dots S_{w_l}$,
- $k = max(\{x, y\} \bigcup \{w_j | S_{w_j} \in w\} \bigcup \{n | S_n \text{ está en el alfabeto de } Z\}) + 1$, y

para todo q_j que pertenece al conjunto de configuraciones internas de $Z^{(2)}$, consideremos la siguiente MT que llamaremos Z_{aux}:

a) $q_1 S_x R q_2$
b) $q_2 S_i R q_2$ (para cada i)
c) $q_2 B R q_2$
d) $q_2 S_y R q_3$
e) $q_j S_y S_{k+jy} q_p$
f) $q_p S_{k+jy} L q_p$
g) $q_p S_i L q_p$
h) $q_p B L q_p$
i) $q_p S_x L q_{p+1}$
j) $q_{p+1} B S_x q_{p+1}$
k) $q_{p+1} S_x R q_{p+2}$
l) $q_{p+2} S_x B q_{p+2}$
m) $q_{p+2} B R q_{p+2}$
n) $q_{p+2} S_i L q_{p+3+i}$
ñ) $q_{p+3+i} B S_i q_{p+3+i+k}$
o) $q_{p+3+i+k} S_i R q_{p+3+i}$
p) $q_{p+3+i} S_i B q_{p+2}$
q) $q_{p+2} S_{k+jy} L q_{p+2k+j+3}$
r) $q_{p+2k+j+3} B S_y q_{p+2k+j+3}$
s) $q_{p+2k+j+3} S_y R q_{p+2k+j+3}$
t) $q_{p+2k+j+3} S_{k+jy} B q_{p+3k+3+j}$
u) $q_{p+3k+j+3} B B q_j$

Entonces la máquina propuesta sería $Z_t = Z_{\text{aux}} \bigcup Z^{(2)}$ Por lo tanto, si la entrada para la máquina Z_t es $q_1 S_x w S_y \overline{(m_1, \dots, m_n)}$ tenemos que:

$$q_1 S_x w S_y \overline{(m_1, \dots, m_n)} \rightarrow S_x q_2 S_{w_1} \dots S_{w_l} S_y \overline{(m_1, \dots, m_n)} \quad \text{por a)}$$

$$\dots$$

$$S_x q_2 S_{w_1} \dots S_{w_l} S_y \overline{(m_1, \dots, m_n)} \rightarrow S_x S_{w_1} \dots S_{w_l} q_2 S_y \overline{(m_1, \dots, m_n)} \quad \text{por b o c)}$$

$$S_x S_{w_1} \dots S_{w_l} q_2 S_y \overline{(m_1, \dots, m_n)} \rightarrow S_x S_{w_1} \dots S_y q_3 \overline{(m_1, \dots, m_n)} \quad \text{por d)}$$

Como la máquina $Z^{(2)}$ inicia con el estado q_3, en este momento se inicia su proceso, supongamos entonces que en algún momento del proceso de $Z^{(2)}$ el cabezal se encuentra en el extremo izquierdo, esto es, se encuentra leyendo S_y.

$$S_x S_{w_1} \dots S_{w_l} S_y q_3 \overline{(m_1, \dots, m_n)} \underset{A}{\twoheadrightarrow} S_x S_{w_1} \dots S_{w_l} q_j S_y P \quad \text{por } Z^{(2)})$$

$$S_xS_{w_1}\dots S_{w_l}q_jS_yP \to S_xS_{w_1}\dots S_{w_l}q_pS_{k+jy}P \quad \text{por e)}$$

$$S_xS_{w_1}\dots S_{w_l}q_pS_{k+jy}P \to S_xS_{w_1}\dots q_pS_{w_l}S_{k+jy}P \quad \text{por f)}$$

$$\dots$$

$$S_xq_pS_{w_1}\dots S_{w_l}S_{k+jy}P \to q_pS_xS_{w_1}\dots S_{w_l}S_{k+jy}P \quad \text{por g o h)}$$

$$q_pS_xS_{w_1}\dots S_{w_l}S_{k+jy}P \to q_{p+1}BS_xS_{w_1}\dots S_{w_l}S_{k+jy}P \quad \text{por i)}$$

$$q_{p+1}BS_xS_{w_1}\dots S_{w_l}S_{k+jy}P \to q_{p+1}S_xS_xS_{w_1}\dots S_{w_l}S_{k+jy}P \quad \text{por j)}$$

$$q_{p+1}S_xS_xS_{w_1}\dots S_{w_l}S_{k+jy}P \to S_xq_{p+2}S_{S_x}S_{w_1}\dots S_{w_l}S_{k+jy}P \quad \text{por k)}$$

$$S_xq_{p+2}S_{S_x}S_{w_1}\dots S_{w_l}S_{k+jy}P \to S_xq_{p+2}BS_{w_1}\dots S_{w_l}S_{k+jy}P \quad \text{por l)}$$

$$S_xq_{p+2}BS_{w_1}\dots S_{w_l}S_{k+jy}P \to S_xBq_{p+2}S_{w_1}\dots S_{w_l}S_{k+jy}P \quad \text{por m)}$$

$$S_xBq_{p+2}S_{w_1}\dots S_{w_l}S_{k+jy}P \to S_xq_{p+3+w_1}BS_{w_1}\dots S_{w_l}S_{k+jy}P \quad \text{por n con i} = w_1)$$

$$S_xq_{p+3+w_1}BS_{w_1}\dots S_{w_l}S_{k+jy}P \to S_xq_{p+3+w_1+k}S_{w_1}S_{w_1}\dots S_{w_l}S_{k+jy}P \quad \text{por ñ con i} = w_1)$$

$$S_xq_{p+3+w_1+k}S_{w_1}S_{w_1}\dots S_{w_l}S_{k+jy}P \to S_xS_{w_1}q_{p+3+w_1}S_{w_1}\dots S_{w_l}S_{k+jy}P \quad \text{por o con i} = w_1)$$

$$S_xS_{w_1}q_{p+3+w_1}S_{w_1}\dots S_{w_l}S_{k+jy}P \to S_xS_{w_1}q_{p+2}B\dots S_{w_l}S_{k+jy}P \quad \text{por p con i} = w_1)$$

Este ciclo se repetirá por cada elemento de w hasta llegar a S_{k+jy}

$$S_xS_{w_1}\dots S_{w_l}q_{p+2}BS_{k+jy}P \to S_xS_{w_1}\dots S_{w_l}Bq_{p+2}S_{k+jy}P \quad \text{por m)}$$

$$S_xS_{w_1}\dots S_{w_l}Bq_{p+2}S_{k+jy}P \to S_xS_{w_1}\dots S_{w_l}q_{p+3+j+2k}BS_{k+jy}P \quad \text{por q)}$$

$$S_xS_{w_1}\dots S_{w_l}q_{p+3+j+2k}BS_{k+jy}P \to S_xS_{w_1}\dots S_{w_l}q_{p+3+j+2k}S_yS_{k+jy}P \quad \text{por r)}$$

$$S_xS_{w_1}\dots S_{w_l}q_{p+3+j+2k}S_yS_{k+jy}P \to S_xS_{w_1}\dots S_{w_l}S_yq_{p+3+j+2k}S_{k+jy}P \quad \text{por s)}$$

$$S_xS_{w_1}\dots S_{w_l}S_yq_{p+3+j+2k}S_{k+jy}P \to S_xS_{w_1}\dots S_{w_l}S_yq_{p+3+j+2k}BP \quad \text{por t)}$$

$$S_xS_{w_1}\dots S_{w_l}S_yq_{p+3+j+2k}BP \to S_xS_{w_1}\dots S_{w_l}S_yq_jBP \quad \text{por u)}$$

Por lo que el proceso de $Z^{(2)}$ puede continuar normalmente. Finalmente, como $Z^{(2)}$ es regular, tenemos que:

$$\text{Res}^A_{Z_t}(q_1S_xwS_y\overline{(m_1,\dots,m_n)}) = q_{\theta(Z_p)}S_xwS_y\overline{(r_1,\dots,r_s)} \qquad \square$$

Corolario 2.2.3.1. *Para cada MT n-regular Z y cada $p > 0$, existe una MT $(p+n)$-regular Z_p tal que si*

$$Res^A_Z(q_1\overline{(m_1,\dots,m_n)}) = q_{\theta(Z)}\overline{(r_1,\dots,r_s)},$$

entonces se tiene que también

$$Res^A_{Z_p}(q_1\overline{(k_1,\ldots,k_p,m_1,\ldots,m_n)}) = q_{\theta(Z_p)}\overline{(k_1,\ldots,k_p,r_1,\ldots,r_s)}.$$

Además si se tiene que $Res^A_Z(q_1\overline{(m_1,\ldots,m_n)})$ *no está definido, entonces* $Res^A_{Z_p}(q_1\overline{(k_1,\ldots,k_p,m_1,\ldots,m_n)})$ *tampoco lo está.*

Demostración. Este teorema es un caso particular del Teorema 2.2.3 cuando $w = \overline{(k_1,\ldots,k_p)}$. □

El teorema anterior nos muestra que para cada Máquina de Turing, podemos encontrar otra que se comporte de la misma manera que la primera, pero dejando intacta una sección de la entrada, e.g. $k_1,\ldots,k_p$.

Lema 2.2.4 (La máquina de traslación R_p)**.** *Para cada* $n > 0$ *y* $p > 0$*, definimos una MT* $(p+n)$*-regular* R_p *tal que*

$$Res_{R_p}(q_1\overline{(k_1,\ldots,k_p,m_1,\ldots,m_n)}) = q_{\theta(R_p)}\overline{(m_1,\ldots,m_n,k_1,\ldots,k_p)}$$

Es decir, los argumentos $m_1,\ldots,m_n$ *se intercambian de lugar con* $k_1,\ldots,k_p$*.*

Demostración. La prueba de este teorema se encuentra en el Anexo II

□

Teorema 2.2.5. *Para cada MT n-regular* Z*, existe una MT n-regular* Z' *tal que si*

$$Res^A_Z(q_1\overline{(m_1,\ldots,m_n)}) = q_{\theta(Z)}\overline{(r_1,\ldots,r_s)},$$

entonces se tiene que

$$Res^A_{Z'}(q_1\overline{(m_1,\ldots,m_n)}) = q_{\theta(Z')}\overline{(r_1,\ldots,r_s,m_1,\ldots,m_n)}.$$

Además, si $Res^A_Z(q_1\overline{(m_1,\ldots,m_n)})$ *no está definido, entonces* $Res^A_{Z'}(q_1\overline{(m_1,\ldots,m_n)})$ *tampoco lo está. En otras palabras, los argumentos son preservados en la cinta.*

Demostración. Este teorema se obtiene como combinación del Corolario 2.2.3.1 y Lema 2.2.4. □

Teorema 2.2.6. *Sean $Z_1, \ldots, Z_p$ Máquinas de Turing y sea $n > 0$. Entonces existe una MT n-regular Z', tal que:*

$$Res^A_{Z'}(q_1\overline{(m_1, \ldots, m_n)}) = q_{\theta(Z')}\overline{(\Psi^{(n)}_{Z_1;A}(m_1, \ldots, m_n), \ldots, \Psi^{(n)}_{Z_p;A}(m_1, \ldots, m_n))}.$$

Demostración. Procederemos por inducción sobre p. Para $p = 1$, el resultado es el Teorema 2.2.2. Supongamos que el resultado es cierto para $p = k$. Entonces sean $Z_1, \ldots, Z_{k+1}$, Máquinas de Turing. Por la hipótesis de inducción, existe una MT Z_K tal que

$$\text{Res}^A_{Z_K}(q_1\overline{(m_1, \ldots, m_n)}) = q_{\theta(Z_K)}\overline{(\Psi^{(n)}_{Z_1;A}(m_1, \ldots, m_n), \ldots, \Psi^{(n)}_{Z_k;A}(m_1, \ldots, m_n))}.$$

Luego, por el Teorema 2.2.5 existe una MT, Z'_K, tal que:

$$\text{Res}^A_{Z'_K}(q_1\overline{(m_1, \ldots, m_n)}) =$$
$$q_{\theta(Z'_K)}\overline{(\Psi^{(n)}_{Z_1;A}(m_1, \ldots, m_n), \ldots, \Psi^{(n)}_{Z_k;A}(m_1, \ldots, m_n), m_1, \ldots, m_n)}.$$

Después, por el Corolario 2.2.3.1, existe una MT $(k+n)$-regular que llamaremos Z_{K+1} tal que:

$$\text{Res}^A_{Z_K}(q_1\overline{(m_1, \ldots, m_n)}) =$$
$$q_{\theta(Z_K)}\overline{(\Psi^{(n)}_{Z_1;A}(m_1, \ldots, m_n), \ldots, \Psi^{(n)}_{Z_k;A}(m_1, \ldots, m_n), \Psi^{(n)}_{Z_{k+1};A}(m_1, \ldots, m_n))}$$

que es la MT que queríamos. Por lo tanto, se tiene el resultado. □

Teorema 2.2.7. *El ciclo for. Es decir, si Z es una MT n-regular tal que para toda $(x_1, \ldots, x_n) \in \omega^n$ se tiene que*

$$Res^A_Z(q_1\overline{(x_1, \ldots, x_n)}) = q_{\theta(Z)}\overline{(m_1, \ldots, m_n)},$$

entonces existe una máquina Z_F $(n+1)$-regular tal que para todo $z_1 > 0$

$$Res^A_{Z_F}(q_1\overline{(z_1, x_1, \ldots, x_n)}) = q_{\theta(Z_F)}\overline{Res^{z_1}_{Z;A}(m_1, \ldots, m_n)}.$$

Donde $Res^y_{Z;A}(m_1, \ldots, m_n)$ significa $\underbrace{Res^{(n)}_{Z;A}(Res^{(n)}_{Z;A}(\ldots(}_{y \text{ veces}} m_1, \ldots, m_n)\ldots)$

Demostración. Sean Z una MT como en el enunciado, $k_1 > 0$ y $\beta = m_1, \ldots, m_n$. Por el Teorema 2.2.3, existe una Máquina de Turing Z_1, tal que si

$$\mathrm{Res}_Z^A(q_1\overline{(\beta)}) = q_{\theta(Z)}\overline{(r_1, \ldots, r_n)},$$

entonces se tiene que también:

$$\mathrm{Res}_{Z_1}^A(q_1\overline{(k_1, \beta)}) = q_{\theta(Z_1)}\overline{(k_1, r_1, \ldots, r_n)}$$

Recordemos que $Z_1^{(n)}$ es la MT que se obtiene de Z_1 al reemplazar cada estado interno q_i y todas sus ocurrencias por q_{i+n}.
Entonces, consideremos la Máquina de Turing M que consiste de las siguientes cuádruplas:

a) $q_1 1 B q_1$
b) $q_1 B R q_2$
c) $q_2 1 1 q_3$
d) $q_2 B R q_{\theta(Z_1^{(2)})+1}$
e) $q_{\theta(Z_1^{(2)})} 1 1 q_1$

Definamos entonces $Z_F = M \bigcup Z_1^{(2)} \bigcup Z^{\theta(Z_1^{(2)})}$. El proceso comenzará con las cuádruplas c y d borrando un símbolo 1 de la expresión en la cinta que en este caso será el numeral de k_1. Las cuádruplas d y e se encargarán de elegir si se debe procesar la cinta con la máquina $Z_1^{(2)}$ cuando la primera entrada de la n-tupla representada en la cinta es mayor a 1, o con la máquina $Z^{\theta(Z_1^{(2)})}$, en cualquier otro caso.
Ahora procedamos formalmente, la descripción instantánea inicial será $q_1\overline{(k_1, \beta)}$ entonces:

$$q_1\overline{(k_1, \beta)} \to q_1 B\overline{(k_1 - 1, \beta)} \quad \text{por a)}$$

$$q_1 B\overline{(k_1 - 1, \beta)} \to B q_2\overline{(k_1 - 1, \beta)} \quad \text{por b)}$$

$$B q_2\overline{(k_1 - 1, \beta)} \to B q_3\overline{(k_1 - 1, \beta)} \quad \text{por c)}$$

$$B q_3\overline{(k_1 - 1, \beta)} \underset{A}{\twoheadrightarrow} B q_{\theta(Z_1^{(2)})}\overline{k_1 - 1} B\overline{\mathrm{Res}_{Z;A}(\beta)} \quad \text{por } Z_1^{(2)}$$

$$B q_{\theta(Z_1^{(2)})}\overline{k_1 - 1} B\overline{\mathrm{Res}_{Z;A}(\beta)} \to B q_1\overline{k_1 - 1} B\overline{\mathrm{Res}_{Z;A}(\beta)} \quad \text{por e)}$$

$$B q_1\overline{k_1 - 1} B\overline{\mathrm{Res}_{Z;A}(\beta)} \to B q_1 B\overline{k_1 - 2} B\overline{\mathrm{Res}_{Z;A}(\beta)} \quad \text{por a)}$$

$$B q_1 B\overline{k_1 - 2} B\overline{\mathrm{Res}_{Z;A}(\beta)} \to B B q_2\overline{k_1 - 2} B\overline{\mathrm{Res}_{Z;A}(\beta)} \quad \text{por b)}$$

$$BBq_2\overline{k_1-2}B\overline{\mathrm{Res}_{Z;A}(\beta)} \to BBq_3\overline{k_1-2}B\overline{\mathrm{Res}_{Z;A}(\beta)} \quad \text{por c)}$$

$$BBq_3\overline{k_1-2}B\overline{\mathrm{Res}_{Z;A}(\beta)} \underset{A}{\twoheadrightarrow} BBq_{\theta(Z_1^{(2)})}\overline{k_1-2}B\overline{\mathrm{Res}_{Z;A}(\mathrm{Res}_{Z;A}(\beta))} \quad \text{por } Z_1^{(2)}$$

$$BBq_{\theta(Z_1^{(2)})}\overline{k_1-2}B\overline{\mathrm{Res}_{Z;A}(\mathrm{Res}_{Z;A}(\beta))} \to BBq_1\overline{k_1-2}B\overline{\mathrm{Res}_{Z;A}(\mathrm{Res}_{Z;A}(\beta))} \quad \text{por e)}$$

$$\dots$$

$$B^{k_1}q_1\overline{(0,\gamma)} \to B^{k_1}q_1BB\overline{(\gamma)} \quad \text{por a)}$$

$$B^{k_1}q_1BB\overline{(0,\gamma)} \to B^{k_1}Bq_2B\overline{(\gamma)} \quad \text{por b)}$$

$$B^{k_1}Bq_2B\overline{(\gamma)} \underset{A}{\twoheadrightarrow} B^{k_1}BBq_{(\theta Z_1^{(2)})+1}\overline{(\gamma)} \quad \text{por d)}$$

Donde $\gamma = \underbrace{\mathrm{Res}_{Z;A}(\mathrm{Res}_{Z;A}(\dots(}_{\mathrm{k}_1 veces}\beta)\dots)$.

Luego, como ninguna cuádrupla comienza con el estado $q_{\theta Z_1^{(2)}+1}$, tenemos que la descripción instantánea $B^{k_1}BBq_{\theta Z_1^{(2)}+1}\overline{(\gamma)}$ es final y, por lo tanto, por el Teorema 2.1.4, terminal. Y finalmente se tiene el resultado. □

Teorema 2.2.8. *Por cada Máquina de Turing Z existe una Máquina de Turing Z_{co} tal que si w es una expresión cualquiera y S_x y S_y son dos símbolos que no pertenecen a w ni al alfabeto de Z, se tiene que:*

$$Res_{Z_{co}}q_1(S_xwS_y) = q_{\theta(Z_{co})}S_xwS_yRes_Z(\overline{\langle w\rangle}).$$

Demostración. Este teorema se obtiene como resultado del Teorema 2.2.1 y Teorema 2.2.3. □

Teorema 2.2.9 (Transitividad de la computabilidad de Turing)**.** *Sea A un conjunto B-computable, y B un conjunto C-computable. Entonces A es C-computable.*

Demostración. Sea Z_{AB} una MT que computa al conjunto A usando como oráculo al conjunto B y sea Z_{BC} una MT que computa al conjunto B usando como oráculo al conjunto C. La idea de esta prueba será utilizar la Máquina Z_{AB} hasta que se encuentre una cuádrupla que haga una consulta al oráculo, por ejemplo, $q_iS_jq_sq_n$. Cuando esto sucede se pausa el cómputo y procedemos a hacer el cómputo de la función característica de B con la máquina Z_{BC} y, dependiendo del resultado del

cómputo, se continúa con el cómputo de Z_{AB} con el estado q_s si el resultado fue 1, o con el estado q_n si el resultado fue 0.

Ahora procederemos formalmente, sean :

- $k = max\{j \mid S_j \in Z^{(2)}_{\text{AB}} \bigcup Z_{\text{BC}}\}$,
- $x = k + 1$ y $y = k + 2$,
- $p = \theta(Z^{(2)}_{\text{AB}})$, e
- $I = \{i \mid q_i \in Z^{(2)}_{\text{AB}}\}$.

Luego, por el Teorema 2.2.8 existe una MT, Z_{bc} tal que

$$\text{Res}_{Z_{\text{bc}}} q_1[S_x w S_y] = S_x w S_y q_{\theta(Z_{\text{bc}})} \text{Res}_{Z_{\text{BC}}}[\overline{\langle w \rangle}].$$

Supongamos que n es la entrada de la máquina, entonces la descripción instantánea al inicio del proceso será $q_1\overline{n}$. Ahora construiremos una máquina que se encargue de escribir los marcadores entre los que se llevará a cabo el cómputo además de moverlos a la izquierda y derecha cuando se necesite. Consideremos la siguiente máquina auxiliar Z_a, y para todo $i \in I$:

a) $q_1 1 R q_1$
b) $q_1 B S_y q_1$
c) $q_1 S_y L q_2$
d) $q_2 1 L q_2$
e) $q_2 B S_x q_2$
f) $q_2 S_x R q_3$
g) $q_i S_x L q_{p+i}$
h) $q_{p+i} B S_x q_{p+i}$
i) $q_{p+i} S_x R q_{2p+i}$
j) $q_{2p+i} S_x B q_i$
k) $q_i S_y R q_{3p+i}$
l) $q_{3p+i} B S_y q_{3p+i}$
m) $q_{3p+i} S_y L q_{4p+i}$
n) $q_{4p+i} S_y B q_i$

Ahora, sea (u_j, v_j, w_j, z_j), $j \in \{1, 2 \ldots l\}$ una enumeración de las cuádruplas de la forma $q_{u_j} S_{v_j} q_{w_j} q_{z_j}$ y $g_j = 2^{u_j} * 3^{v_j} * 5^{w_j} * 7^{z_j}$, además $r = \theta(Z_{\text{bc}})$. Ahora construiremos una máquina que se encargará de lo siguiente: cada vez que la máquina se encuentre en un estado y leyendo un símbolo de tal manera que la siguiente instrucción represente una consulta al oráculo, esta máquina escribirá un símbolo que servirá de marcador de la posición y el símbolo que está escrito en ese lugar, luego llevará el cabezal a la izquierda de la cinta hasta que encuentre el símbolo S_x y comenzará el proceso de la

máquina Z_{bc}, una vez que haya concluido el proceso de la máquina Z_{bc}, determinará si es que el resultado fue 1 o 0 y, en cada caso, regresará el cabezal a la posición inicial de este proceso, escribirá el símbolo que se encontraba en esa posición y pasará al estado que corresponda con respecto al resultado.

Para eso, consideremos para cada j la siguiente máquina auxiliar Z_j y sea S_f en el alfabeto de $Z^{(2)}_{\text{AB}}$, además $h_j = p + g_j * (r + 4)$:

ñ) $q_{u_j} S_{v_j} S_{y+g_j} q_{h_j+1}$

o) $q_{h_j+1} S_{y+g_j} L q_{h_j+1}$

p) $q_{h_j+1} S_f L q_{h_j+1}$

q) $q_{h_j+1} S_x S_x q_{h_j+2}$

r) $q_{h_j+r} 1 B q_{h_j+r+3}$

s) $q_{h_j+r+3} B L q_{h_j+r+3}$

t) $q_{h_j+r+3} S_y L q_{h_j+r+3}$

u) $q_{h_j+r+3} S_f L q_{h_j+r+3}$

v) $q_{h_j+r+3} S_{y+g_j} S_{v_j} q_{w_j}$

w) $q_{h_j+r} B L q_{h_j+r+4}$

x) $q_{h_j+r+4} S_y L q_{h_j+r+4}$

y) $q_{h_j+r+4} S_f L q_{h_j+r+4}$

z) $q_{h_j+r+4} S_{y+g_j} S_{v_j} q_{z_j}$

Sea entonces la máquina que buscamos, $Z_{\text{AC}} = Z^{(2)}_{\text{AB}} \bigcup_{j=1}^{l} Z^{(h_j+1)}_{\text{bc}} \bigcup Z_a \bigcup_{j=1}^{l} Z_j$.

Ahora procedemos al cómputo, supongamos entonces que la entrada de la máquina Z_{AC} es n. Entonces:

$$q_1 \overline{n} \to 1 q_1 \overline{n-1} \qquad \text{por a)}$$

$$\dots$$

$$\overline{n-1} q_1 \to \overline{n} q_1 B \qquad \text{por a)}$$

$$\overline{n} q_1 B \to \overline{n} q_1 S_y \qquad \text{por b)}$$

$$\overline{n} q_1 S_y \to \overline{n-1} q_2 1 S_y \qquad \text{por c)}$$

$$\overline{n-1} q_2 1 S_y \to \overline{n-2} q_2 1 1 S_y \qquad \text{por d)}$$

$$\dots$$

$$q_2 1 \overline{n-1} S_y \to q_2 B \overline{n} S_y \qquad \text{por d)}$$

$$q_2 B \overline{n} S_y \to q_2 S_x \overline{n} S_y \qquad \text{por e)}$$

$$q_2 S_x \overline{n} S_y \to S_x q_3 \overline{n} S_y \qquad \text{por f)}$$

luego, q_3 es el estado inicial de $Z^{(2)}_{\text{AB}}$ por lo que a partir de este momento, se inicia el

proceso de $Z^{(2)}_{\mathrm{AB}}$.

Supongamos que en algún momento del cómputo, el cabezal se mueve a la izquierda del uno que se encuentra en la extrema izquierda dentro de la zona delimitada por S_x y S_y, entonces el cómputo sería como sigue:

$$\begin{aligned}
S_x q_3 \overline{n} S_y &\twoheadrightarrow q_i S_x \alpha S_y && \text{por} Z^{(2)}_{\mathrm{AB}}) \\
q_i S_x \alpha S_y &\to q_{p+i} B S_x \alpha S_y && \text{por g)} \\
q_{p+i} B S_x \alpha S_y &\to q_{p+i} S_x S_x \alpha S_y && \text{por h)} \\
q_{p+i} S_x S_x \alpha S_y &\to S_x q_{2p+i} S_x \alpha S_y && \text{por i)} \\
S_x q_{2p+i} S_x \alpha S_y &\to S_x q_i B \alpha S_y && \text{por j)}
\end{aligned}$$

Supongamos que en algún momento del cómputo, el cabezal se mueve a la derecha del uno que se encuentra en la extrema derecha dentro de la zona delimitada por S_x y S_y, entonces el cómputo sería como sigue:

$$\begin{aligned}
S_x \alpha q_i S_y &\twoheadrightarrow S_x \alpha S_y q_{3p+i} B && \text{por k)} \\
S_x \alpha S_y q_{3p+i} B &\to S_x \alpha S_y q_{3p+i} S_y && \text{por l)} \\
S_x \alpha S_y q_{3p+i} S_y &\to S_x \alpha q_{4p+i} S_y S_y && \text{por m)} \\
S_x \alpha q_{4p+i} S_y S_y &\to S_x \alpha q_i B S_y && \text{por n)}
\end{aligned}$$

Por lo que en ambos casos el cómputo de $Z^{(2)}_{\mathrm{AB}}$ seguiría como lo haría normalmente. Ahora supongamos que durante el cómputo la máquina se encuentra en una parte del proceso en la que se encuentra una cuádrupla de la forma $q_{u_j} S_{v_j} q_{w_j} q_{z_j}$, es decir, se realiza una consulta al oráculo (en este caso, el conjunto B), entonces el cómputo continúa como sigue:

$$\begin{aligned}
S_x \alpha S_f q_{u_j} S_{v_j} \beta S_y &\to S_x \alpha S_f q_{h_j+1} S_{y+g_j} \beta S_y && \text{por ñ)} \\
S_x \alpha S_f q_{h_j+1} S_{y+g_j} \beta S_y &\to S_x \alpha q_{h_j+1} S_f S_{y+g_j} \beta S_y && \text{por o)} \\
&\dots \\
S_x q_{h_j+1} \alpha S_f S_{y+g_j} \beta S_y &\to q_{h_j+1} S_x \alpha S_f S_{y+g_j} \beta S_y && \text{por o)} \\
q_{h_j+1} S_x \alpha S_f S_{y+g_j} \beta S_y &\to q_{h_j+2} S_x \alpha S_f S_{y+g_j} \beta S_y && \text{por o)}
\end{aligned}$$

$$q_{h_j+2}S_x\alpha S_f S_{y+g_j}\beta S_y \underset{C}{\twoheadrightarrow} S_x\alpha S_f S_{y+g_j}\beta S_y q_{h_j+r}\mathrm{Res}_{Z_{\mathrm{BC}}}[\overline{\langle w\rangle}] \qquad \text{por } Z_{\mathrm{bc}})$$

Ahora, si $\overline{\langle w\rangle}$ está en el conjunto B, entonces $\mathrm{Res}_{Z_{\mathrm{BC}}}[\overline{\langle w\rangle}]$ será 1 y:

$$S_x\alpha S_{y+g_j}\beta S_f S_y q_{h_j+r}1 \to S_x\alpha S_{y+g_j}\beta S_f S_y q_{h_j+r+3}B \qquad \text{por r)}$$

$$S_x\alpha S_{y+g_j}\beta S_f q_{h_j+r+3}B \to S_x\alpha S_{y+g_j}\beta S_f q_{h_j+r+3}S_y B \qquad \text{por s)}$$

$$S_x\alpha S_{y+g_j}\beta S_f q_{h_j+r+3}S_y B \to S_x\alpha S_{y+g_j}\beta q_{h_j+r+3}S_f S_y B \qquad \text{por t)}$$

$$\dots$$

$$S_x\alpha q_{h_j+r+3}S_{y+g_j}\beta S_f S_y \to S_x\alpha q_{w_j}S_{v_j}\beta S_f S_y \qquad \text{por v)}$$

Por otro lado, si $\overline{\langle w\rangle}$ no está en el conjunto B, entonces $\mathrm{Res}_{Z_{\mathrm{BC}}}[\overline{\langle w\rangle}]$ será B y:

$$S_x\alpha S_{y+g_j}\beta q_{h_j+r}S_y B \to S_x\alpha S_{y+g_j}\beta q_{h_j+r+4}S_y B \qquad \text{por w)}$$

$$S_x\alpha S_{y+g_j}\beta q_{h_j+r+4}S_y B \to S_x\alpha S_{y+g_j}\beta q_{h_j+r+4}S_f S_y B \qquad \text{por x)}$$

$$\dots$$

$$S_x\alpha q_{h_j+r+4}S_{y+g_j}\beta S_y \to S_x\alpha q_{z_j}S_{v_j}\beta S_y \qquad \text{por z)}$$

En ambos casos, continuamos con el proceso de $Z^{(2)}_{\mathrm{AB}}$ continúa como lo haría al hacer la consulta al oráculo por lo que sigue su proceso normalmente.

De esta manera, la máquina se comporta como esperábamos y el resultado al final será que computa la función característica de A usando como oráculo al conjunto C. □

Composición y Minimalización La operación más natural entre máquinas es alimentar a una con la salida de otra, en esta sección se formaliza este proceso en la composición de las máquinas.

Para revisar que alguna relación no es vacía, un proceso "orgánico" puede ser probar cada uno de los números naturales para ver si dicha relación se cumple en algún momento, esto se formaliza en el concepto de minimalización.

Definición 2.2.4. Dadas $f^{(m)}, g_1^{(n)}, g_2^{(n)}, \dots, g_m^{(n)}$, la operación de *composición* asocia a las funciones $f, g_1, g_2, \dots, g_m$ con la función h definida de modo que para toda

$\beta \in \omega^n$:

$$h(\beta) = f(g_1(\beta), \ldots, g_m(\beta)).$$

La función h está definida para aquellas n-tuplas $(a_1, \ldots, a_n)$ que pertenecen al dominio de cada una de las funciones g_i con $1 \leq i \leq m$ y para las cuales la m-tupla

$$(g_1(a_1, \ldots, a_n), \ldots, g_m(a_1, \ldots, a_n))$$

está en el dominio de f. Además,

$$h((a_1, \ldots, a_n)) = f(g_1(a_1, \ldots, a_n), \ldots, g_m(a_1, \ldots, a_n)). \qquad \heartsuit$$

Con esta definición probaremos el siguiente teorema.

Teorema 2.2.10. *Sean $f^{(m)}, g_1^{(n)}, g_2^{(n)}, \ldots, g_m^{(n)}$ funciones parciales A-computables y sea $h^{(n)}$ como en la definición de composición. Entonces $h^{(n)}$ es parcial A-computable.*

Demostración. Sean f, g_i y h con $1 \leq i \leq m$ como en el enunciado. Por el Teorema 2.2.6, existe una MT n-regular Z tal que para toda $\beta \in \omega^{(n)}$ adecuada:

$$\mathrm{Res}_Z^A(q_1\overline{(\beta)}) = q_{\theta(Z)}\overline{(g_1(\beta), \ldots, g_m(\beta))}.$$

Luego, tomemos Z_1 tal que

$$\Psi_{Z_1;A}^{(m)}(\alpha) = f(\alpha)$$

y sea $Z' = Z \bigcup Z_1^{(\theta(Z)-1)}$. Entonces, con respecto a Z':

$$q_1\overline{(\beta)} \twoheadrightarrow q_{\theta(Z)}\overline{(g_1(\beta), \ldots, g_m(\beta))} \qquad \text{por } Z)$$
$$q_{\theta(Z)}\overline{(g_1(\beta), \ldots, g_m(\beta))} \twoheadrightarrow q_{\theta(Z')}\overline{\Psi_{Z_1;A}^{(m)}(g_1(\beta), \ldots, g_m(\beta))} \qquad \text{por } Z_1^{(\theta(Z)-1)})$$

en caso de que cada $g_i(\beta)$ y $f(g_1(\beta), \ldots, g_m(\beta))$ estén definidos. En otro caso ,$\mathrm{Res}_{Z'}^A[q_1(\beta)]$ no está definido. Por lo tanto,

$$\Psi_{Z';A}^{(n)} = f(g_1(\beta), \ldots, g_m(\beta)) = h(\beta). \qquad \square$$

Del teorema anterior se sigue el siguiente corolario importante:

Corolario 2.2.10.1. *La clase de las funciones parciales A-computables es cerrada bajo la operación de composición.*

Definición 2.2.5. La operación de *minimalización* asocia a cada función total $f^{(n+1)}(y, m_1, \ldots, m_n)$ la función parcial $h(m_1, \ldots, m_n)$, cuyo valor para una n-tupla dada $(m_1, \ldots, m_n)$ es el menor valor y, si este existe, tal que $f^{(n+1)}(y, m_1, \ldots, m_n) = 0$, y, si este y no existe, entonces h no está definida. Lo escribiremos como sigue:

$$h(m_1, \ldots, m_n) = \min_y[f^{(n+1)}(y, m_1, \ldots, m_n) = 0]. \qquad \heartsuit$$

Definición 2.2.6. Una función total $f^{(n+1)}(y, m_1, \ldots, m_n)$ es llamada regular si

$$h(m_1, \ldots, m_n) = \min_y[f^{(n+1)}(y, m_1, \ldots, m_n) = 0]$$

es total. $\heartsuit$

Teorema 2.2.11. *Si $f^{(n+1)}(y, m_1, \ldots, m_n)$ es A-computable, entonces*

$$h(m_1, \ldots, m_n) = \min_y[f^{(n+1)}(y, m_1, \ldots, m_n) = 0]$$

es parcial A-computable. Aún más, si $f^{(n+1)}(y, m_1, \ldots, m_n)$ es regular, entonces $h(m_1, \ldots, m_n)$ es total A-computable.

Demostración. La idea detrás de esta prueba será "probar" con cada valor de y, si la función $f^{(n+1)}(y, m_1, \ldots, m_n)$ es igual a 0 o no, si esto se cumple, entonces regresar como resultado del cómputo y símbolos 1, de otra manera, probar con $y + 1$. Así, la máquina o el proceso, deberá tener memoria acerca del valor que se está probando en un determinado momento.

Ahora procedemos formalmente:

Por el Teorema 2.2.6, existe una MT, Z_{pm} tal que

$$\text{Res}^A_{Z_{\text{pm}}}(q_1\overline{(y, m_1, \ldots, m_n)}) = \overline{q_{\theta(Z_{\text{pm}})}(f^{(n+1)}(y, m_1, \ldots, m_n), U_1^{(n+1)}(y, m_1, \ldots, m_n), \ldots, U_{n+1}^{(n+1)}(y, m_1, \ldots, m_n))}$$

y existe una MT, Z_{sp} tal que

$$\text{Res}^A_{Z_{sp}}(q_1\overline{(m_1, \ldots, m_{n+2})}) = q_{\theta(Z_{sp})}\overline{(U_2^{n+2}(m_1, \ldots, m_{n+2}) + 1, \ldots, U_{n+1}^{n+2}(m_1, \ldots, m_{n+2}))}.$$

Sea $Z_{p_1} = (U_1^{(n+2)})^{(\theta(Z_{\text{pm}})+1)}$, es decir, la primera proyección de $n + 2$ entradas, desplazada por $\theta(Z_{\text{pm}}) + 1$ en todos sus estados.

Consideremos la máquina auxiliar Z_a que consiste en las siguientes cuádruplas:

a) $q_1 1 B q_1$

b) $q_1 B R q_2$

c) $q_2 B R q_3$

d) $q_2 1 R q_{\theta(U_1^{n+2})}$

Consideremos entonces la máquina

$$Z_{\text{mín}} = Z_{\text{pm}} \bigcup Z_a^{(\theta(Z_{\text{pm}}))-1} \bigcup Z_{sp}^{(\theta(Z_{p_1})-1)} \bigcup Z_{p_1} \bigcup \{q_{\theta(Z_{sp}^{(\theta(Z_{p_2})-1)})} 11 q_1\}$$

Entonces si la entrada de la Máquina $Z_{\text{mín}}$ es $(m_1, \ldots, m_n)$, tenemos que la máquina iniciará su proceso con la descripción instantánea $q_1\overline{(0, m_1, \ldots, m_n)}$, por lo tanto, con respecto a Z_{min}:

$$\begin{aligned} q_1\overline{(0, m_1, \ldots, m_n)} \twoheadrightarrow\ & q_{\theta(Z_{\text{pm}})}\overline{(f^{(n+1)}(0, m_1, \ldots, m_n), U_1^{n+1}(0, m_1, \ldots, m_n),} \\ & \overline{\ldots, U_{n+1}^{n+1}(0, m_1, \ldots, m_n))} \qquad \text{por } Z_{\text{pm}}) \\ =\ & q_{\theta(Z_{\text{pm}})}\overline{(f^{(n+1)}(0, m_1, \ldots, m_n), 0, m_1, \ldots, m_n)}. \end{aligned}$$

Ahora, si el resultado de $f^{(n+1)}(0, m_1, \ldots, m_n)$ es cero, tenemos que:

$$\begin{aligned} & q_{\theta(Z_{\text{pm}})}\overline{(f^{(n+1)}(0, m_1, \ldots, m_n), 0, m_1, \ldots, m_n)} = \\ & \qquad q_{\theta(Z_{\text{pm}})} 1 B \overline{(0, m_1, \ldots, m_n)} \to q_{\theta(Z_{\text{pm}})} B B \overline{(0, m_1, \ldots, m_n)} \qquad \text{por a)} \\ & \qquad q_{\theta(Z_{\text{pm}})} B B \overline{(0, m_1, \ldots, m_n)} \to B q_{\theta(Z_{\text{pm}})+1} B \overline{(0, m_1, \ldots, m_n)} \qquad \text{por b)} \\ & \qquad B q_{\theta(Z_{\text{pm}})+1} B \overline{(0, m_1, \ldots, m_n)} \to B B q_{\theta(Z_{\text{pm}})+2} \overline{(0, m_1, \ldots, m_n)} \qquad \text{por c)} \\ & \qquad B B q_{\theta(Z_{\text{pm}})+2} \overline{(0, m_1, \ldots, m_n)} \twoheadrightarrow B B q_{\theta(Z_{p_1})} B \qquad \text{por } Z_{p_1}). \end{aligned}$$

Notemos que $\theta(Z_{p_1})$ es un estado terminal por lo que la máquina se detiene teniendo como resultado cero y, por lo tanto, sucede lo que se esperaba.

En el caso de que el resultado de $f^{(n+1)}(0, m_1, \ldots, m_n)$ no fuera cero, tenemos que:

$$\begin{aligned} & q_{\theta(Z_{\text{pm}})}\overline{(f^{(n+1)}(0, m_1, \ldots, m_n), 0, m_1, \ldots, m_n)} = \\ & q_{\theta(Z_{\text{pm}})} 11 \overline{(f^{(n+1)}(0, m_1, \ldots, m_n) - 2, 0, m_1, \ldots, m_n)} \to \\ & q_{\theta(Z_{\text{pm}})} B 1 \overline{(f^{(n+1)}(0, m_1, \ldots, m_n) - 2, 0, m_1, \ldots, m_n)} \text{ por a)} \\ & q_{\theta(Z_{\text{pm}})} B 1 \overline{(f^{(n+1)}(0, m_1, \ldots, m_n) - 2, 0, m_1, \ldots, m_n)} \to \\ & B q_{\theta(Z_{\text{pm}})+1} 1 \overline{(f^{(n+1)}(0, m_1, \ldots, m_n) - 2, 0, m_1, \ldots, m_n)} \text{ por b)} \end{aligned}$$

$$B q_{\theta(Z_{\mathrm{pm}})+1} 1 \overline{(f^{(n+1)}(0, m_1, \ldots, m_n) - 2, 0, m_1, \ldots, m_n)} \rightarrow$$

$$B q_{\theta(Z_{\mathrm{pm}})+1+\theta(U_1^{n+2})} 1 \overline{(f^{(n+1)}(0, m_1, \ldots, m_n) - 2, 0, m_1, \ldots, m_n)} \text{ por d)}$$

$$= B q_{\theta(Z_{p_1})} 1 \overline{(f^{(n+1)}(0, m_1, \ldots, m_n) - 2, 0, m_1, \ldots, m_n)} \twoheadrightarrow$$

$$B q_{Z_{\mathrm{sp}}^{(\theta(Z_{p_1})-1)}} \overline{(0+1, m_1, \ldots, m_n)} \text{ por } Z_{\mathrm{sp}}^{(\theta(Z_{p_1})-1)})$$

$$B q_{Z_{\mathrm{sp}}^{(\theta(Z_{p_1})-1)}} \overline{(0+1, m_1, \ldots, m_n)} \rightarrow$$

$$B q_1 \overline{(0+1, m_1, \ldots, m_n)} \text{ por } q_{\theta(Z_{\mathrm{sp}}^{(\theta(Z_{p_1})-1)})} 11 q_1.$$

En este caso, comienza el procedimiento de Z_{pm} de nuevo pero con la primera coordenada aumentada en uno. Un proceso análogo comprueba que esto es verdadero para $y > 0$, por lo que se tiene el resultado esperado. □

2.2.3. Recursión primitiva

Hasta ahora hemos discutido cómo obtener funciones computables a partir de algunas funciones, que de antemano sabemos que son computables, mediante minimalización o composición. Sin embargo, ninguno de estos mecanismos tiene la capacidad de construir una máquina que nos permita computar funciones (como x^y) que implican la aplicación sucesiva de una función, (en este caso $f(n) = x * \cdots * x$) una cantidad variable (y) de veces. Para eso necesitamos un instrumento nuevo que describiremos en esta sección y llamaremos *recursión primitiva*.

Teorema 2.2.12. *Sean $f^{(n)}$ y $g^{(n+2)}$ funciones totales. Entonces a lo más existe una función total, $h^{(n+1)}$, que satisface las ecuaciones de recursión:*

$$h(0, \gamma) = f(\gamma)$$

$$h(z+1, \gamma) = g(z, h(z, \gamma), \gamma).$$

para todo $\gamma \in \omega^n$ y $z \in \omega$.

Demostración. Supongamos que $h_1(z, \gamma)$ y $h_2(z, \gamma)$ son funciones tales que satisfacen las ecuaciones de recursión. Por inducción sobre z, demostraremos que $h_1(z, \gamma) = h_2(z, \gamma)$ para todo $z \in \omega$. Sea $\gamma \in \omega^n$. Para $z = 0$, tenemos que

$$h_1(z, \gamma) = h_1(0, \gamma) = f(\gamma) = h_2(0, \gamma) = h_2(z, \gamma)$$

Ahora, supongamos que $h_1(z,\gamma) = h_2(z,\gamma)$, entonces:

$$\begin{aligned} h_1(z+1,\gamma) &= g(z, h_1(z,\gamma), \gamma) \\ &= g(z, h_2(z,\gamma), \gamma) \\ &= h_2(z+1,\gamma). \end{aligned}$$

□

Teorema 2.2.13. *Sean $f^{(n)}$ y $g^{(n+2)}$ funciones totales. Entonces existe al menos una función total, $h^{(n+1)}$, que satisface las ecuaciones de recursión.*

Demostración. Consideremos conjuntos de $(n+2)$-túplas (y,u,γ) con $\gamma \in \omega^n$. Llamaremos a uno de esos conjuntos, dígase S, *satisfactorio* si:

- para cada $\gamma \in \omega^n$, $(0, f(\gamma), \gamma) \in S$, y
- para cada $\gamma \in \omega^n$, si $(z,u,\gamma) \in S$, entonces $(z+1, g(z,u,\gamma), \gamma) \in S$.

Sea Ω el conjunto de todos los conjuntos *satisfactorios*. Entonces Ω es no vacío ya que el conjunto de todas las $(n+2)$-túplas es *satisfactorio*, dado que las funciones f y g son totales. Sea S_0 la intersección de todos los conjuntos S que pertenecen a Ω. Este conjunto es satisfactorio y además está contenido en todo conjunto satisfactorio. Luego, supongamos que $(0,u,\gamma) \in S_0$. Tenemos entonces que $u = f(\gamma)$ puesto que, de otra manera, el conjunto obtenido de quitar $(0,u,\gamma)$ de S_0 también sería *satisfactorio* y no contendría a S_0.

Afirmación Para todo y y para todo γ existe una y solo una u tal que $(y,u,\gamma) \in S_0$. Por la justificación anterior, sabemos que esto es cierto para $y = 0$. Supongamos que es cierto para $y = z$. Entonces existe una y solo una u tal que $(z,u,\gamma) \in S_0$. Luego, como S_0 es *satisfactorio*, tenemos que $(z+1, g(z,u,\gamma), \gamma) \in S_0$. Supongamos que $(z+1, v, \gamma) \in S_0$, pero $v \neq g(z,u,\gamma)$. Entonces el conjunto obtenido al quitar $(z+1, v, \gamma)$ de S_0 sería *satisfactorio*, pero no contendría a S_0. Por lo tanto, la afirmación queda probada. Luego, definamos la función $h(y,\gamma)$ de la siguiente manera:

$$u = h(y,\gamma) \iff (y,u,\gamma) \in S_0.$$

Por la construcción de los conjuntos satisfactorios, tenemos que la función cumple las ecuaciones de recursión. □

Definición 2.2.7. La operación de *recursión primitiva* asocia a dos funciones totales $f^{(n)}$ y $g^{(n+2)}$ con la función $h^{(n+1)}$ donde:

$$h(0,\gamma) = f(\gamma)$$
$$h(z+1,\gamma) = g(z,h(z,\gamma),\gamma). \qquad \heartsuit$$

Teorema 2.2.14. *Sea $h^{(n+1)}$ obtenida por recursión primitiva de $f^{(n)}$ y $g^{(n+2)}$. Si $f^{(n)}$ y $g^{(n+2)}$ son A-computables, entonces $h^{(n+1)}$ también es A-computable.*

Demostración. Sean $f^{(n)}$, $g^{(n+2)}$ y $h^{(n+1)}$ como en el enunciado. Como $f^{(n)}$ y $g^{(n+2)}$ son computables, por el Teorema 2.2.2, existen Z_f y Z_g Máquinas de Turing que A-computan a f y g, respectivamente, y, además, son n-regular y $n+2$-regular, respectivamente. Construyamos ahora algunas máquinas que nos ayudarán en la prueba. De acuerdo al Teorema 2.2.6, existe una MT $(n+1)$-regular que llamaremos Z_a tal que:

$$\begin{aligned}\mathrm{Res}^A_{Z_a}(q_1\overline{(m_1,\ldots,m_{n+1})}) =& q_{\theta(Z_a)}\overline{(U_1(m_1,\ldots,m_{n+1})+1, 0, \Psi_{Z_f;A}(m_2,\ldots,m_{n+1})}\\ & \overline{\ldots, U_2(m_1,\ldots,m_{n+1}),\ldots,U_{n+1}(m_1,\ldots,m_{n+1}))}\\ =& q_{\theta(Z_a)}\overline{(m_1+1, 0, \Psi_{Z_f;A}(m_2,\ldots,m_{n+1}), m_2,\ldots,m_{n+1})},\end{aligned}$$

donde U_i es la $i-$ésima proyeción, como en el Ejemplo 3. Análogamente, existe una MT $(n+2)$-regular que llamaremos Z_b tal que:

$$\mathrm{Res}^A_{Z_b}(q_1\overline{(m_1,\ldots,m_{n+2})}) = q_{\theta(Z_b)}\overline{(m_1+1, \Psi_{Z_g;A}(m_1,\ldots,m_{n+2}), m_3,\ldots,m_{n+2})}.$$

Luego, por el Teorema 2.2.7, existe una MT $(n+3)$-regular que llamaremos Z_r tal que si $k_1 > 0$, entonces

$$\mathrm{Res}^A_{Z_r}[q_1\overline{(k_1, m_1,\ldots,m_{n+2})}] = q_{\theta(Z_r)}\overline{\Psi^{k_1}_{Z_b;A}(m_1,\ldots,m_{n+2})}.$$

Por otro lado, por el Ejemplo 3, existe una MT que llamaremos Z_{p_2} que computa la segunda proyección.

Sea $u = \theta(Z^{(2)}_f)$, $v = \theta(Z^{(u)}_a) - 1$ y $w = \theta(Z^{(v)}_r) - 1$. Entonces consideremos la Máquina de Turing Z_{aux} que consiste de las siguientes cuádruplas:

a) $q_1 1 B q_1$

b) $q_1 B R q_2$

c) $q_2 1 1 q_{u+1}$

d) $q_2 B R q_3$

y definimos:

$$Z_{\text{pr}} = Z_{\text{aux}} \bigcup Z_f^{(2)} \bigcup Z_a^{(u)} \bigcup Z_r^{(v)} \bigcup Z_{p_2}^{(w)}.$$

Ahora procedamos formalmente:

Sea $n \in \omega$ y $\beta \in \omega^n$, entonces tenemos 2 casos:

Supongamos que $n = 0$, por lo que la expresión instantánea inicial es $q_1\overline{(0,\beta)}$, entonces:

$$\begin{aligned}
q_1\overline{(0,\beta)} &\to q_1 B B\overline{\beta} && \text{por a)}\\
q_1 B B\overline{\beta} &\to B q_2 B\overline{\beta} && \text{por b)}\\
B q_2 B\overline{\beta} &\to B B q_3\overline{\beta} && \text{por d)}\\
B B q_3\overline{\beta} &\underset{A}{\twoheadrightarrow} q_u\overline{\Psi_{Z_f^{(2)};A}(\beta)} && \text{por } Z_f^{(2)}
\end{aligned}$$

Como $Z_f^{(2)}$ es regular y además $Z_a^{(u)}$, $Z_r^{(v)}$ y $Z_{p_2}^{(w)}$ contienen estados internos mayores a u, la descripción instantánea $q_u\overline{\Psi_{Z_f^{(2)};A}(\beta)}$ es final y, por lo tanto, $\text{Res}_{Z_{\text{pr}};A}(q_1\overline{(0,\beta)}) = q_u\overline{\Psi_{Z_f^{(2)};A}(\beta)}$ y $\Psi_{Z_{\text{pr}};A}(\beta) = \langle q_u\overline{\Psi_{Z_f^{(2)};A}(\beta)}\rangle = f(\beta)$

Supongamos que $n > 0$, por lo que la expresión instantánea inicial es $q_1\overline{(n,\beta)}$, entonces:

$$\begin{aligned}
q_1\overline{(n,\beta)} &\to q_1 B\overline{(n-1,\beta)} && \text{por a)}\\
q_1 B\overline{(n-1,\beta)} &\to B q_2\overline{(n-1,\beta)} && \text{por b)}\\
B q_2\overline{(n-1,\beta)} &\to B q_{u+1}\overline{(n-1,\beta)} && \text{por c)}\\
B q_{u+1}\overline{(n-1,\beta)} &\underset{A}{\twoheadrightarrow} B q_{\theta(Z_a^{(u)})}\overline{(n,0,\Psi_{Z_f}(\beta),\beta)} && \text{por } Z_a^{(u)}\\
B q_{\theta(Z_a^{(u)})}\overline{(n,0,\Psi_{Z_f}(\beta),\beta)} &\underset{A}{\twoheadrightarrow} B q_{\theta(Z_r^{(v)})}\overline{\Psi^n_{Z_b}(0,\Psi_{Z_f;A}(\beta),\beta)} && \text{por } Z_r^{(v)}\\
B q_{\theta(Z_r^{(v)})}\overline{\Psi^n_{Z_b}(0,\Psi_{Z_f}(\beta),\beta)} &\underset{A}{\twoheadrightarrow} B q_{\theta(Z_{p_2}^{(w)})}\overline{(\Psi^n_{Z_b}(0,\Psi_{Z_f}(\beta),\beta))-1} && \text{por } Z_{p_2}^{(w)}.
\end{aligned}$$

Afirmamos que para toda $n > 0$:

$$\text{Res}_{Z_r}(q_1\overline{(n,0,\Psi_{Z_f}(\beta),\beta)}) = q_{\theta(Z_r)}\overline{(n,h(n,\beta),\beta)}.$$

Observemos lo siguiente:

$$\begin{aligned}\mathrm{Res}_{Z_r}(q_1\overline{(1,0,\Psi_{Z_f}(\beta),\beta)}) &= q_{\theta(Z_r)}\mathrm{Res}_{Z_b}[q_1\overline{(0,\Psi_{Z_f}(\beta),\beta)}]\\ &= q_{\theta(Z_r)}\overline{(1,\Psi_{Z_g}(0,\Psi_{Z_f}(\beta),\beta),\beta)}.\end{aligned}$$

Y, por la definición de Z_f, Z_g y la función h, tenemos que :

$$\begin{aligned}\Psi_{Z_g}(0,\Psi_{Z_f}(\beta),\beta) &= g(0,\Psi_{Z_f}(\beta),\beta)\\ &= g(0,f(\beta),\beta)) = g(0,h(0,\beta),\beta)) = h(1,\beta).\end{aligned}$$

Por lo tanto:

$$\mathrm{Res}_{Z_r}(q_1\overline{(1,0,\Psi_{Z_f}(\beta),\beta)}) = q_{\theta(Z_r)}\overline{(1,h(1,\beta),\beta)}.$$

Supongamos ahora que se cumple para $n=k$, por la definición de Z_r, tenemos que:

$$\begin{aligned}\mathrm{Res}_{Z_r}(q_1\overline{(k+1,0,\Psi_{Z_f}(\beta),\beta)}) &= q_{\theta(Z_r)}\overline{\mathrm{Res}^{k+1}_{Z_b}(0,\Psi_{Z_f}(\beta),\beta)}\\ &= q_{\theta(Z_r)}\mathrm{Res}_{Z_b}(\mathrm{Res}^{k}_{Z_b}(\overline{(0,\Psi_{Z_f}(\beta),\beta)}))\\ &= q_{\theta(Z_r)}\mathrm{Res}_{Z_b}(\overline{(k,h(k,\beta),\beta)})\\ &= q_{\theta(Z_r)}\overline{(k+1,\Psi_{Z_g}(k,h(k,\beta),\beta),\beta)}\\ &= q_{\theta(Z_r)}\overline{(k+1,h(k+1,\beta),\beta)}.\end{aligned}$$

Por lo tanto, $\mathrm{Res}_{Z_{\mathrm{pr}}}(q_1\overline{(n,h(n,\beta),\beta)}) = q_{\theta(Z_{\mathrm{pr}})}\overline{h(n,\beta)-1}$ y $\Psi_{Z_{\mathrm{pr}}}(\beta) = \langle q_{\theta(Z_{\mathrm{pr}})}\overline{h(n,\beta)-1}\rangle = h(n,\beta)$. Y finalmente, tenemos el resultado. □

2.3. Recursividad y computabilidad

2.3.1. Funciones recursivas

Definición 2.3.1. Una función es *A-recursiva primitiva* o *recursiva primitiva en A* si puede ser obtenida mediante una cantidad finita de aplicaciones de composición o recursión primitiva empezando con funciones que pertenezcan a la siguiente lista:

1. $\chi_A(x)$, la función característica de A,
2. $S(x) = x+1$, la función sucesor,

3. $N(x) = 0$, la función constante 0, y
4. $U_i^n(x_1, \ldots x_n) = x_i$, $1 \leq i \leq n$, la i-ésima proyección.

Las funciones parciales $\emptyset$-recursivas serán llamadas simplemente *recursivas primitivas*. ♡

Definición 2.3.2. Una función es *A-recursiva* o *recursiva* en A si puede ser obtenida mediante una cantidad finita de aplicaciones de composición, recursión primitiva o minimalización de funciones comenzando con funciones de la lista de la Definición 2.3.1. Llamamos a las funciones $\emptyset$-recursivas simplemente *recursivas*. ♡

Observemos que las funciones *A-recursivas primitivas* son *A-recursivas*.

Teorema 2.3.1. *Si una función es A-recursiva, entonces es A-computable.*

Demostración. Que la función característica de un conjunto A es A-computable, se tiene como resultado del Teorema 2.1.8. Mientras que el hecho de que la función sucesor, la constante 0 y las proyecciones son computables se tiene como resultado del Ejemplo 1, el Ejemplo 2, y el Ejemplo 3, respectivamente. El resultado se obtiene entonces de los Teorema 2.2.10, Teorema 2.2.11 y Teorema 2.2.14. □

Ahora presentamos algunas funciones que son recursivas y, por el teorema anterior, también A-recursivas para todo A. Además, por el Teorema 2.3.1, también son computables y A-computables para cualquier A.

1. $x + y$

   Puesto que:

$$x + 0 = U_1^1(x),$$
$$x + S(y) = S(x + y).$$

2. $x * y$

   Puesto que:

$$x * 0 = N(x),$$
$$x * S(y) = (x * y) + U_1^2(x, y).$$

3. $P(x)$, donde $P(0) = 0$, $P(x) = x - 1$ si $x > 0$.

   Puesto que:

$$P(0) = N(x),$$
$$P(S(x)) = U_1^1(x).$$

4. $x \dot{-} y$

   Puesto que:

$$x \dot{-} 0 = U_1^1(x),$$
$$x \dot{-} S(y) = P(x \dot{-} y).$$

5. $n!$, donde $n! = 1 * 2 * 3 \cdots * n$.

   Puesto que:

$$0! = S(N(x)),$$
$$S(n)! = n! * S(n).$$

6. x^y

   Puesto que:

$$x^0 = S(N(x)),$$
$$x^{S(y)} = x^y * U_1^2(x, y).$$

7. $\alpha(x) = 1 \dot{-} x$

   Puesto que:

$$\alpha(x) = S(N(0)) \dot{-} U_1^1(x). \tag{2.1}$$

8. $|x - y|$

   Puesto que:

$$|x - y| = (x \dot{-} y) + (y \dot{-} x).$$

9. $\left[\sqrt{x}\right]$, el mayor entero menor o igual a $\sqrt{x}$.

$$\begin{aligned}\left[\sqrt{x}\right] &= \min_y[(y+1)^2 \dot{-} x \neq 0] \\ &= \min_y[\alpha((S(U_2^2(x,y)))^2 \dot{-} U_1^2(x,y)) = 0].\end{aligned}$$

10. $\left[\frac{x}{y}\right]$. Si $y \neq 0$, $[\frac{x}{y}]$ es el mayor entero menor que $\frac{x}{y}$. Si $y = 0$, entonces $\frac{x}{y} = 0$.

$$\begin{aligned}\left[\frac{x}{y}\right] &= \min_y[(y = 0 \vee y(z+1) > x] \\ &= \min_y[(y = 0 \vee y(z+1) \dot{-} x \neq 0] \\ &= \min_y[(y = 0 \vee \alpha(y(z+1) \dot{-} x) = 0] \\ &= \min_y[(y \cdot \alpha(y(z+1) \dot{-} x) = 0].\end{aligned}$$

11. $R(x, y)$. Si $y \neq 0$, $R(x, y)$ es el residuo de dividir x entre y, es decir:

$$R(x, y) = x \dot{-} y \cdot \left[\frac{x}{y}\right].$$

12. $K_1(x) = S(N(x)) = 1$.
13. $K_n(x) = \underbrace{S(S(\ldots S(}_{n-veces} N(x)) \ldots)) = n$.

Teorema 2.3.2. *Si una función parcial es recursiva, entonces es una función parcial A-recursiva para cualquier A.*

Demostración. Se deriva del hecho que:

$$\chi_\emptyset(x) = N(x).$$

Es decir, la función característica del vacío se puede sustituir por la función constante 0 en cualquier sucesión de aplicaciones de composición, recursión primitiva o minimalización. □

Teorema 2.3.3. *Sea $n \in \omega$ y $\beta \in \omega^n$. Si $f(n, \beta)$ es $A - (primitiva)\, recursiva$, entonces*

$$g(n, \beta) = \sum_{k=0}^{n} f(k, \beta)$$

y

$$h(n, \beta) = \prod_{k=0}^{n} f(n, \beta)$$

también lo son.

Demostración. Podemos definir a g y h como:

$$\begin{aligned} g(0,\beta) &= f(0,\beta), \\ g(S(n),\beta) &= g(n,\beta) + f(S(n),\beta) \text{ y} \\ h(0,\beta) &= f(0,\beta), \\ h(S(n),\beta) &= h(n,\beta) * f(S(n),\beta). \end{aligned}$$

□

Teorema 2.3.4. *Sea $S \subset \omega$, tal que $|S| = n$. Entonces la función característica de S, denotada por C_S, es recursiva.*

Demostración. Por inducción sobre n. Para $n = 1$, sea $S = \{l\}$, entonces $C_S(x) = \alpha(|x - K_l|)$. Supongamos que es cierto para $n = m$, entonces sea $S = \{k_1, k_2, \ldots, k_{m+1}\}$ $= \{k_1, k_2, \ldots, k_m\} \cup \{k_{m+1}\}$, por hipótesis de inducción. La función característica de $S' = \{k_1, k_2, \ldots, k_m\}$ es recursiva, digamos $C_{S'}$. Entonces $C_S = C_{S'} + C_{\{k_{m+1}\}}$ y, por lo tanto, recursiva.

De esta manera, se tiene el resultado. □

2.3.2. Predicados y conjuntos recursivos

Ahora discutiremos el concepto de conjunto recursivo:

Definición 2.3.3. Sea $S \subseteq \omega^n$. Decimos que S es *(A-)primitivo recursivo* si su función característica $C_S(x)$ lo es. ♡

Definición 2.3.4. Llamamos a un predicado P *(A-)primitivo recursivo* si su extensión, es decir, $E = \{\beta \in \omega^n | P(\beta)\}$ lo es. ♡

Teorema 2.3.5. *Sean P y Q predicados (A-)primitivos recursivos. Entonces $P \wedge Q$, $P \vee Q$ y $\sim Q$ también lo son.*

Demostración. Esto se sigue de las identidades:

$$\begin{aligned} C_{P \wedge Q} &= C_P * C_Q \\ C_{P \vee Q} &= (C_P + C_Q) \dot{-} (C_P * C_Q) \\ C_{\sim Q} &= 1 \dot{-} C_Q. \end{aligned}$$

□

Teorema 2.3.6. *Si $P(y,\beta)$ es un predicado $A-$(primitivo) recursivo, entonces para toda $z \in \omega$.*

$$\bigwedge_{y=0}^{z} P(y,\beta) \quad y \quad \bigvee_{y=0}^{z} P(y,\beta)$$

también lo son.

Demostración. Denotemos por $Q(z,\beta)$ a $\bigwedge_{y=0}^{z} P(y,\beta)$ Entonces

$$C_{Q(z,\beta)} = \prod_{y=0}^{z} C_P(z,\beta),$$

Y, por el Teorema 2.3.3, esta es A-(primitiva) recursiva. Luego observemos que: $\bigwedge_{y=0}^{z} P(y,\beta)$ equivale a $\sim \bigvee_{y=0}^{z} \sim P(y,\beta)$, la cual también es A-(primitiva) recursiva como consecuencia de la primera parte de esta prueba y el Teorema 2.3.5. □

Definición 2.3.5. Sea $P(y,\beta)$ un predicado. Entonces definimos:

$$f(z,\beta) = \mathfrak{M}_{y=0}^{z} P(y,\beta)$$

como la función total que satisface la ecuación

$$f(z,\beta) = \min_{y}[y \leq z \wedge P(y,\beta)]$$

si $\min_y[y \leq z \wedge P(y,\beta)]$ está definido y 0 en otro caso. ♡

Teorema 2.3.7. *Si $P(y,\beta)$ un predicado A-(primitivo) recursivo, entonces $f(z,\beta) = \mathfrak{M}_{y=0}^{z} P(y,\beta)$ también lo es.*

Demostración. Supongamos que existe $y \leq z$ tal que se cumple $P(y,\beta)$. Entonces sea:

$$\phi(t,\beta) = \prod_{y=0}^{t} \sim C_{P(y,\beta)}.$$

Entonces $\phi(t,\beta)$ es la función característica de $\bigvee_{y=0}^{t} \sim P(y,\beta)$. Entonces, el valor de t se incrementa desde 0 hasta z y tenemos que $\phi(t,\beta)$ vale 1 hasta que el valor de t llega a un t_0 tal que se cumple $P(t_0,\beta)$. Para t_0 y todos los valores siguientes tenemos que $\phi(t,\beta) = 0$. Por lo tanto,

$$\sum_{t=0}^{z} \phi(t,\beta) = \sum_{t=0}^{t_0-1} \phi(t,\beta)$$

$$= \sum_{t=0}^{t_0-1} 1$$
$$= t_0 = \mathfrak{M}_{y=0}^{z} P(y, \beta).$$

Luego, para cualquier caso tenemos que:

$$\mathfrak{M}_{y=0}^{z} P(y, \beta) = \alpha(\phi(z, \beta)) \cdot \sum_{t=0}^{z} \phi(z, \beta)$$
$$= \alpha(\prod_{y=0}^{z} \sim C_{P(y,\beta)}) \cdot \sum_{t=0}^{z} \prod_{y=0}^{t} \sim C_{P(y,\beta)}$$

Por lo que el teorema se sigue como consecuencia del Teorema 2.3.6 □

Teorema 2.3.8. *Los predicados $x = y$ y $x < y$ son recursivos primitivos.*

Demostración. Sus funciones características son: $\alpha(\alpha(|x - y|))$ y $\alpha(y \dot{-} x)$ respectivamente. □

Con esto podemos probar que algunos predicados que nos serán muy útiles más adelante son recursivos primitivos tales como:

1. $y|x$ (y divide a x)

   Puesto que:

   $$y|x \iff \bigvee_{z=0}^{x} x = yz.$$

2. Primo(x) (x es un número primo)

   Puesto que:

   $$\text{Primo}(x) \iff (x > 1) \wedge \bigwedge_{z=0}^{x} [(z = 1) \vee (z = x) \vee \sim (z|x)]$$

3. Pr(n) (el enésimo número primo, donde el 0-ésimo número primo es 0)

   Puesto que:

   $$\text{Pr}(0) = 0$$
   $$\text{Pr}(n+1) = \mathfrak{M}_{y=0}^{\text{Pr}(n)!+1} [\text{Primo}(y) \wedge y > \text{Pr}(n)].$$

2.3.3. Enumeración de las Máquinas de Turing

Ahora comenzaremos el proceso de asociar una Máquina de Turing con un número natural, lo que nos permitirá referirnos a las MT dentro del mismo lenguaje que las entradas de las Máquinas de Turing.

Los símbolos básicos usados en las Máquinas de Turing son:

$$R, L$$

$$S_0, S_1, S_2, \ldots$$

$$q_1, q_2, q_3, \ldots$$

A cada uno de estos símbolos los asociaremos con un número impar mayor o igual a 3 como sigue:

3,	5,	7,	9,	11,	13,	15,	17,	19,	21,	...
R,	L,	S_0,	q_1,	S_1,	q_2,	S_2,	q_3,	S_3,	q_4,	...

Es decir, para cada i, se asocia S_i con $4i + 7$ y q_i con $4i + 5$.

Definición 2.3.6. Sea M una expresión que consiste en los símbolos $\gamma_1, \gamma_2, \ldots, \gamma_n$. Sean $a_1, a_2, \ldots, a_n$ los enteros asociados con los símbolos respectivos. Entonces el *número de Gödel* de M es el entero:

$$r = \prod_{k=1}^{n} \Pr(k)^{a_k}.$$

Escribimos $\text{gn}(M) = r$. Si M es la expresión vacía, entonces $\text{gn}(M) = 1$. ♡

Entonces, por ejemplo, $\text{gn}(q_1 1 R q_2) = 2^9 \cdot 3^{11} \cdot 5^3 \cdot 7^{13}$.

Corolario 2.3.8.1. *Si M y N son expresiones tales que $gn(N) = gn(M)$, entonces $M = N$.*

Demostración. El teorema es una consecuencia inmediata del teorema fundamental de la aritmética, ya que al ser única la descomposición en potencias de primos de un número, si dos expresiones tienen el mismo número de Gödel, están compuestas de los mismos símbolos. □

Definición 2.3.7. Si $n = \text{gn}(M)$, entonces definimos $M = \text{Exp}(n)$. ♡

Definición 2.3.8. Sean $M_1, \ldots, M_n$ una sucesión finita de expresiones. Entonces el *número de Gödel* de esta sucesión de expresiones es:

$$\prod_{k=1}^{n} \text{Pr}(k)^{\text{gn}(M_k)}$$ ♡

De esta manera, el número de Gödel de la sucesión $q_1 1 B q_1$, $q_1 B R q_2$ sería $2^{2^9 3^{11} 5^7 7^9} \cdot 3^{2^9 3^7 5^3 7^{13}}$

Corolario 2.3.8.2. *Ningún entero es al mismo tiempo el número de Gödel de una expresión y de una sucesión de expresiones.*

Demostración. Un número de Gödel siempre es de la forma $2^n \cdot m$ con $n > 0$ y m impar. Sin embargo, si $2^n \cdot m$ es el número de Gödel de una expresión, n es impar mientras que si $2^n \cdot m$ es el número de Gödel de una sucesión de expresiones , n en sí mismo es el número de Gödel de una expresión y, por lo tanto, es par. □

Corolario 2.3.8.3. *Dos sucesiones de expresiones que tienen el mismo número de Gödel son idénticas.*

Demostración. Este corolario es consecuencia del teorema fundamental de la aritmética y del Corolario 2.3.8.1. □

Definición 2.3.9. Sea Z una Máquina de Turing y sea $\mathfrak{Z}(Z)$ el conjunto de todas las posibles sucesiones de expresiones que se pueden formar con todas las cuádruplas de Z, es decir, que todas las cuádruplas de Z aparezcan en la sucesión una vez. Entonces, el *número de Gödel de la Máquina de Turing* Z es el mínimo número de Gödel de M_Z con $M_Z \in \mathfrak{Z}$. ♡

Definición 2.3.10. La función $\text{Gn}(Z)$ asocia a cada Máquina de Turing con su número de Gödel correspondiente. Esta función está bien definida debido a los corolarios probados anteriormente. Luego, si $e = \text{Gn}(Z)$, entonces definimos $\Psi_{e;A}(m_1, \ldots, m_n) = \Psi_{Z;A}(m_1, \ldots, m_n)$ y si no hay ambiguedad con respecto a A, $\Psi_e(m_1, \ldots, m_n) = \Psi_Z(m_1, \ldots, m_n)$. ♡

Definición 2.3.11. Para cada $n > 0$ y para cada conjunto de enteros A, definimos $T_n^A(z, x_1, \ldots, x_n, y)$ como el predicado que significa que, dados $z, x_1, \ldots, x_n, y$, se tiene que z es el número de Gödel de una Máquina de Turing Z y que y es el número de Gödel de un A-cómputo con respecto a Z, empezando con la descripción instantánea $q_1\overline{(x_1, \ldots, x_n)}$. ♡

Usaremos la enumeración descrita para probar, el siguiente teorema:

Teorema 2.3.9. *$T_n^A(z, x_1, \ldots, x_n, y)$ es A recursivo.*

La prueba será mediante la demostración de que algunos enunciados y funciones son A recursivos, culminando con los predicados de la forma T_n^A y se puede consultar en el Anexo III.

Ahora usaremos este resultado para probar la equivalencia entre computabilidad y recursividad.

Teorema 2.3.10. *Sea Z_0 una Máquina de Turing y sea z_0 el número de Gödel de Z_0, además sea $\gamma \in \omega^n$. Entonces el dominio de la función $\Psi_{Z_0;A}^{(n)}(\gamma)$ es igual al dominio de $\min_y T_n^A(z_0, \gamma, y)$. Más aún,*

$$\Psi_{Z_0;A}^{(n)}(\gamma) = U(\min_y T_n^A(z_0, \gamma, y)),$$

donde U es la función definida en la Ecuación 21. Además, si $T_n^A(z_0, \gamma, y_0)$ se cumple para alguna γ, entonces

$$y_0 = \min_y T_n^A(z, \gamma, y).$$

Demostración. Para probar la igualdad de los dominios de $\Psi_{Z_0;A}^{(n)}(\gamma)$ y $\min_y T_n^A(z_0, \gamma, y)$, tenemos que $\min_y T_n^A(z_0, \gamma, y_0)$ está definida para γ si y sólo si existe un $A-$cómputo de Z_0 que comienza con $q_1(\overline{\gamma})$, es decir, $\Psi_{Z_0;A}^{(n)}(\gamma)$ está definida, por lo que $\gamma \in \text{Dom}(\min_y T_n^A(z_0, \gamma, y_0)) \iff \gamma \in \text{Dom}(\Psi_{Z_0;A}^{(n)})$. Luego, si $y_0 = \min_y T_n^A(z_0, \gamma, y)$ está definida para γ, entonces y_0 es el número de Gödel de un $A-$cómputo de Z_0 que comienza con $q_1(\overline{\gamma})$. Por lo que $U(y_0) = \text{Corn}(\mathfrak{L}(y_0)\text{Gl}y_0) = \langle\alpha\rangle$, donde α es la descripción instantánea final del $A-$cómputo. Pero $\langle\alpha\rangle = \Psi_{Z_0;A}^{(n)}(\gamma)$. □

De este resultado obtenemos el siguiente:

Corolario 2.3.10.1. *f es una función parcial A-computable si y sólo si existe un número z_0 tal que*

$$f(\alpha) = U(\min_y T_n^A(z_0, \alpha, y)).$$

Y del corolario anterior se obtiene el siguiente resultado:

Corolario 2.3.10.2. *Cada función parcial A-computable es parcial A-recursiva.*

Combinando los resultados obtenidos anteriormente, tenemos que:

Corolario 2.3.10.3. *Una función parcial A-computable si y sólo si es A-recursiva.*

Una Máquina de Turing Universal Si consideramos la función recursiva parcial $\psi(z, x) = U(\text{mín}_y T(z, x, y))$, entonces tenemos que esta función es computable y, por lo tanto, existe una Máquina de Turing Univ tal que:

$$\Psi^{(2)}_{\text{Univ}}(z, x) = \psi(z, x).$$

Llamamos a esta MT, una Máquina de Turing Universal, dado que puede ser usada para computar cualquier función parcialmente computable como sigue:

Si Z_0 es una MT y z_0 es un número de Gödel de Z_0, entonces

$$\Psi^{(2)}_{\text{Univ}}(z_0, x) = \Psi_{Z_0}(x).$$

2.4. Sobre Máquinas de Turing

En esta sección se obtendrán conclusiones acerca de algunas propiedades asociadas a las Máquinas de Turing y su relación con las funciones computables.

2.4.1. Algunos teoremas importantes

Teorema 2.4.1 (Cardinalidad de las funciones computables)**.** *Existen exactamente una cantidad numerable de funciones computables.*

Demostración. Existe al menos una cantidad numerable de funciones computables dado que para todo $n \in \omega$, la función constante n es computable.

Ahora, como cada Máquina de Turing corresponde a lo más a una función computable y por la Definición 2.3.10 la cardinalidad del conjunto de todas las MT es a lo más numerable, se puede concluir que existen a lo más una cantidad numerable de funciones computables. □

Teorema 2.4.2 (Existencia de funciones no recursivas)**.** *Existen funciones que no son recursivas.*

Demostración. Por el teorema de Cantor, existen $2^{\aleph_0}$ funciones. Luego, existen funciones que no son recursivas. □

Teorema 2.4.3 (Del Relleno)**.** *Cada función parcial tiene una cantidad numerable de Máquinas de Turing asociadas.*

Demostración. Sea M una Máquina de Turing regular que computa a una función f y $e = \text{gn}(M)$. Consideremos $a = \theta(M)$. Entonces consideremos para $z > a$, $M_z = M \bigcup \{q_z 11 q_z\}$. Notemos que, si $a < x < y$, entonces $M_x \neq M_y$ y, por lo tanto, $\text{gn}(M_x) \neq \text{gn}(M_y)$.

Observación. Para toda $z \geq a$, $f = \Psi_{M_z}$, pues por construcción, para toda $z > a$, tenemos que q_z no es un estado de M, entonces no hay ninguna cuádrupla de M que tenga como último símbolo a q_z. Por lo tanto, en ninguna descripción instantánea de ningún cómputo aparecerá el estado q_z y se conducirá como lo indique M. Por lo tanto, $\Psi_M = \Psi_{M_z}$ □

Teorema 2.4.4 (Teorema de enumeración)**.** *Existe un z tal que para todos x y y, $\Psi_z(x, y) = \Psi_x(y)$ si $\Psi_x(y)$ está definida; y $\Psi_z(x, y)$ no está definida si $\Psi_x(y)$ no está definida.*

Demostración. Este teorema se deriva de lo comentado en la Sección 2.3.3, ya que este z es el número de Gödel de la Máquina de Turing Universal. □

Teorema 2.4.5 (Teorema SMN)**.** *Existe una función recursiva $s(x, y)$ tal que para todos x, y, z, se cumple que:*

$$\Psi_{s(x,y)}(z) = \Psi_x(y, z).$$

La idea del Teorema SMN es que existe una función *computable* que, teniendo el número de Gödel de una Máquina de Turing m y un parámetro entero y, nos permite obtener el número de una MT que tendrá el mismo efecto sobre la entrada que el que tendría la máquina m si se alimentara con la entrada y el parámetro. El hecho más resaltable de este resultado es que la función sea computable, es decir, existe una MT que es capaz de proveer información acerca de otras MT, en este caso en particular, el número de Gödel.

Demostración. Comenzaremos por bosquejar cómo sería la MT cuyo número de Gödel asociado tendría que regresar la función $s(x, y)$.

Lo primero que tendría que hacer la máquina en cuestión sería escribir al principio de la cinta el numeral de y. Consideremos la máquina E_y que consiste en las siguientes cuádruplas:

a) $q_1 1 L q_1$,
b) $q_1 B L q_2$,
c) $q_{i+1} B 1 q_{i+1}$,
d) $q_{i+1} 1 L q_{i+2}$,
e) $q_{y+2} B L q_{y+3}$,

donde $1 \leq i \leq y$. Podemos ver que con respecto a E_y:

$$\begin{aligned}
q_1 \overline{z} &\to q_1 B \overline{z} && \text{por a)} \\
q_1 B \overline{z} &\to q_2 B B \overline{z} && \text{por b)} \\
q_2 B B \overline{z} &\to q_2 1 B \overline{z} && \text{por c)} \\
q_2 1 B \overline{z} &\to q_2 B 1 B \overline{z} && \text{por d)} \\
&\dots \\
q_{y+2} 1 \overline{(y-1, z)} &\to q_{y+3} B \overline{(y, z)} && \text{por e)}
\end{aligned}$$

Ahora, si consideramos a Z_x, tal que $\mathrm{Gn}(Z_x) = x$, entonces tenemos que la máquina en cuestión podría ser $Z_{(x,y)} = Z_x^{(y+2)} \bigcup E_y$, dado que:

$$\Psi_{\mathrm{Gn}(Z_{(x,y)})}(z) = \Psi_x(y,z).$$

Por lo tanto, basta con obtener una función recursiva que permita el cálculo del número de Gödel de esta función, dependiendo de las entradas x, y.

Podemos obtener el número de Gödel de E_y con una función como sigue:

$$\begin{aligned}
a &= \mathrm{gn}(q_1 1 L q_1) = 2^9 * 3^{11} * 5^5 * 7^9,\\
b &= \mathrm{gn}(q_1 B L q_2) = 2^9 * 3^7 * 5^5 * 7^{13},\\
c(i) &= \mathrm{gn}(q_{i+1} B 1 q_{i+1}) = 2^{4i+9} * 3^7 * 5^{11} * 7^{4i+9},\\
d(i) &= \mathrm{gn}(q_{i+1} 1 L q_{i+2}) = 2^{4i+9} * 3^{11} * 5^5 * 7^{4i+13},\\
e(y) &= \mathrm{gn}(q_{y+2} B L q_{y+3}) = 2^{4y+13} * 3^7 * 5^{11} * 7^{4y+17}.
\end{aligned}$$

Entonces la función $\varphi(y)$ definida como:

$$\varphi(y) = 2^a * 3^b * 5^{e(y)} * \prod_{i=1}^{y} [\mathrm{Pr}(i+3)^{c(i)} * \mathrm{Pr}(i+y+3)^{d(i)}].$$

Ahora, por el Ecuación 26, tenemos que la función $\mathrm{Tras}(x, y+2)$, es recursiva.

Luego, si hacemos:

$$s(x,y) = \varphi(y)\hat{}\,\mathrm{Tras}(x, y+2),$$

tenemos que para x y y dados, $s(x,y) = \mathrm{Gn}(Z_{(x,y)})$ si es que x es el número de Gödel de una MT y 0 en otro caso. □

Capítulo 3

Grados de Turing

En este capítulo mostraremos que algunos problemas son irresolubles usando MT sin oráculos y definiremos la relación de orden entre los diferentes problemas en términos de la dificultad de su solución.

3.1. El problema de la detención

Consideremos el siguiente problema: ¿Existe algún procedimiento efectivo de tal manera que, dados cualesquiera x y y, es posible determinar mediante este procedimiento si $\Psi_x(y)$ está definida o no? Es decir, ¿si $x = \mathrm{Gn}(Z)$ para alguna Máquina de Turing Z, entonces alimentando a Z con la entrada y, se detiene o no? Podemos formular estas preguntas de la siguiente manera: ¿Existe alguna Máquina de Turing, Z, tal que $\Psi_Z(x, y) = 1$, si $\Psi_x(y)$ es convergente, y $\Psi_Z(x, y) = 0$, si $\Psi_x(y)$ no lo es?

El siguiente teorema responde esta pregunta:

Teorema 3.1.1 (Problema de la detención)**.** *No existe una función recursiva g tal que $g(x, y) = 1$, si $\Psi_x(y)$ es convergente, y $g(x, y) = 0$, si $\Psi_x(y)$ no lo es.*

Demostración. Supongamos que existe una función g como la descrita en el enunciado y definamos una nueva función Ψ como sigue:

$$\Psi(x) = \begin{cases} 1, & \text{si } g(x,x) = 0 \\ \text{diverge}, & \text{si } g(x,x) = 1 \end{cases}$$

Es claro que Ψ es computable en g, por lo tanto, es computable. Sea e el número de Gödel de una MT que computa a Ψ. Entonces, por la definición de Ψ, $\Psi_e(e)$ converge si y solo si $g(e,e) = 0$,por lo que $\Psi_e(e)$ converge si y solo si $\Psi_e(e)$ no converge, lo cual es una contradicción. □

El teorema anterior y su formulación se conocen como el problema de la detención y representa un ejemplo de un problema insoluble.

Este problema es significativo puesto que abre la posibilidad de caracterizar otros problemas como insolubles si es posible encontrar una forma de reducir estos al problema de la detención.

Definición 3.1.1. Sea $K = \{x \mid \Psi_x(x) \text{converge}\}$. ♡

Teorema 3.1.2. *K no es computable.*

Demostración. Si K fuera computable, es decir, su función característica C_K fuera computable, entonces la función:

$$f(x) = \begin{cases} \Psi_x(x) + 1 & \text{si } x \in K \\ 0 & \text{si } x \notin K \end{cases}$$

también sería computable. Afirmamos que f es diferente a todas las funciones computables, para probarlo podemos observar que si f fuera igual a alguna función computable Ψ_e para alguna $e \in \omega$, entonces $e \in K \implies \Psi_e(e) = f(e) = \Psi_e(e) + 1$ y $e \notin K \implies \Psi_e(e)$ no está definido $= f(e) = 0$, lo cual es una contradicción en ambos casos y, por lo tanto, f no puede ser computable.. □

Definición 3.1.2. Sea $K_0 = \{(x,y) \mid \Psi_x(y) \text{converge}\}$. ♡

Corolario 3.1.2.1. *K_0 no es computable.*

Demostración. Notemos que $x \in K$ si y solo si $(x,x) \in K_0$, así que si K_0 fuera computable, también lo sería K. □

3.2. Grados de Turing

Habiendo demostrado la existencia de problemas insolubles en la sección anterior, es natural preguntarse si existen otros y cómo se relacionarían entre sí.

En esta sección se desarrolla la teoría que permitirá responder ambas cuestiones.

Definición 3.2.1. Sean $A, B \subseteq \omega$. Decimos que A es *Turing reducible* a B o $A \leq_T B$ si la función característica de A es B-computable. ♡

Definición 3.2.2. Decimos que A y B son *Turing-equivalentes*, i.e. $A \equiv_T B$ si $A \leq_T B$ y $B \leq_T A$. ♡

Definición 3.2.3. Decimos que A y B son *Turing-incompatibles*, i.e. $A|_T B$ si $A \not\leq_T B$ y $B \not\leq_T A$. ♡

Teorema 3.2.1. *$\equiv_T$ define una relación de equivalencia.*

Demostración. Por el Teorema 2.1.8, sabemos que $A \leq_T A$ y, por lo tanto, $\equiv_T$ es reflexiva. Por la definición de $\equiv_T$, se puede notar que es simétrica. La transitividad está dada por el Teorema 2.2.9. □

Definición 3.2.4. El *grado de Turing* de un conjunto A es la clase de equivalencia $\deg(A) = \{B | B \equiv_T A\}$ ♡

Definición 3.2.5. Definimos la suma directa de A y B como el conjunto:

$$\{2x | x \in A\} \bigcup \{2x+1 | x \in B\}$$

y lo denotamos como: $A \oplus B$. ♡

Ahora definiremos dos conjuntos que nos servirán más adelante, para lo cual necesitamos probar el siguiente teorema:

Lema 3.2.2. *Se cumple lo siguiente:*

1. $A \leq_T A \oplus B$,
2. $B \leq_T A \oplus B$, *y*
3. *si* $A \leq_T C$ *y* $B \leq_T C$ *entonces* $A \oplus B \leq_T C$.

Demostración. Para el inciso 1, podemos considerar la siguiente función:

$$f_A(x) = \begin{cases} 1 & \text{si } 2x \in A \oplus B \\ 0 & \text{si } 2x \notin A \oplus B. \end{cases}$$

Mientras que para el inciso 2, podemos considerar la siguiente función:

$$f_B(x) = \begin{cases} 1 & \text{si } 2x+1 \in A \oplus B \\ 0 & \text{si } 2x+1 \notin A \oplus B. \end{cases}$$

En ambos casos, por la Definición 3.2.5, se tiene que $f_A = \chi_A$ y $f_B = \chi_B$.

Para el inciso 3, por hipótesis, existen funciones g_{AC} y g_{BC} tales que computan a A y a B usando como oráculo a C, respectivamente. Entonces consideremos la función:

$$f_{\oplus}(x) = g_{\text{AC}}\Big(\Big[\frac{x}{2}\Big]\Big) + g_{\text{BC}}\Big(\Big[\frac{P(x)}{2}\Big]\Big).$$

Entonces, tenemos que $y \in A \oplus B \iff y \in \{2x | x \in A\} \bigvee y \in \{2x+1 | x \in B\} \iff \exists z \in A | 2z = y \bigvee \exists z \in B | 2z+1 = y \iff \frac{y}{2} \in A \bigvee \frac{y-1}{2} \in B$.

Por lo tanto, tenemos que $f_{\oplus}(x) = \chi_{A\oplus B}(x)$ y entonces $A \oplus B \leq_T C$. □

El conjunto $A \oplus B$ tiene la particularidad de comportarse como un *supremo* de los conjuntos A y B, es decir, como probamos en el lema anterior, para cualquier conjunto C, tal que tanto A como B sean reducibles a C, tenemos que $A \oplus B$ será también reducible al conjunto C.

Notación 3.2.1. Algunas definiciones sobre notación:

I Las letras minúsculas en negritas **a, b, c** denotan grados de Turing y **D** denota la clase de todos los grados de Turing.

II La clase de los grados **D** forma un conjunto parcialmente ordenado $(\mathbf{D}, \leq)$ bajo la relación $\deg(A) \leq \deg(B) \iff A \leq_T B$. Decimos que $\deg(A) < \deg(B)$ si $A <_T B$, es decir, si $A \leq_T B$ y $B \not\leq_T A$.

III $\deg(A) \vee \deg(B) = \deg(A \oplus B)$. ♡

3.3. El salto de Turing

El fenómeno de la detención se puede extrapolar para distintos conjuntos, usándolos como oráculos.

Formalmente esa operación de considerar el problema de la detención para un cierto conjunto se ve reflejada en la operación del Salto de Turing.

Definición 3.3.1. Sea $K^A = \{x|\exists y \in \mathbb{N} \quad T^A(x,x,y)\}$, donde $T^A(x,x,y)$ se define como en Definición 2.3.11, este conjunto es llamado el *salto de Turing* de A y se denota como A'. ♡

Definición 3.3.2. $A^{(n)}$ es el n-ésimo salto de A y se obtiene iterando el salto n veces, es decir, $A^{(0)} = A$, $A^{(n+1)} = (A^{(n)})'$. ♡

3.4. Enunciados Σ_1

En esta sección discutiremos una clase de enunciados y conjuntos especiales: los enunciados o conjuntos Σ_1.

Definición 3.4.1. Un conjunto (enunciado) B es Σ_0, Π_0, o Δ_0 si y sólo si B es computable.

Para $n \geq 1$, $B \in \Sigma_n$ si existe una relación computable $R(x, y_1, y_2, \ldots, y_n)$ tal que

$$x \in B \iff (\exists y_1)(\forall y_2)(\exists y_3)\cdots(Qy_n)R(x, y_1, y_2, \ldots, y_n),$$

donde Q es $\exists$, si n es impar, y $\forall$, si n es par. Para $n \geq 1$, $B \in \Pi_n$ si existe una relación computable $R(x, y_1, y_2, \ldots, y_n)$ tal que

$$x \in B \iff (\forall y_1)(\exists y_2)(\forall y_3)\cdots(Qy_n)R(x, y_1, y_2, \ldots, y_n),$$

donde Q es $\exists$, si n es par, y $\forall$, si n es impar. Además, $B \in \Delta_n$ si $B \in \Sigma_n$ y $B \in \Pi_n$. ♡

Definición 3.4.2. Sea A un conjunto. Si reemplazamos en la Definición 3.4.1 la palabra "computable" por "A-computable", entonces tenemos una definición para que B sea Σ_n , Π_n o Δ_n en A. ♡

En particular notemos que los conjuntos Σ_1 son aquellos de la forma $\exists y R(x,y)$ con R computable.

Teorema 3.4.1. *Para todo conjunto $A \in \Sigma_1$, se tiene que $A <_T K_0$ (con K_0 como en la Definición 3.1.2).*

Demostración. Sea $A \in \Sigma_1$, y $x \in A \iff \exists y R(x,y)$ con $R(x,y)$ computable. Definimos $f(x) = \text{mín}_y[R(x,y)]$ y $e = \text{Gn}(f)$ que existe, pues f es parcial computable. Entonces,

$$C(x) = \begin{cases} 1, & \text{si } (e,x) \in K_0 \\ 0, & \text{si } (e,x) \notin K_0, \end{cases}$$

lo cual muestra que C es computable en K_0. Ahora notemos que:

$\chi_A(x) = 1 \iff x \in A \iff \exists y R(x,y) \iff x \in \text{Dom}(f) \iff Z_e(x)$ se detiene $\iff (e,x) \in K_0 \iff C(x) = 1$, y

$\chi_A(x) = 0 \iff x \notin A \iff \neg\exists y R(x,y) \iff x \notin \text{Dom}(f) \iff$ $Z_e(x)$ no se detiene $\iff (e,x) \notin K_0 \iff C(x) = 0$.

Por lo tanto, $C = \chi_A$ y χ_A es computable en K_0.

Ahora, notemos que $K_0 \not\leq_T A$, dado que si $K_0 \leq_T A$, entonces K_0 sería computable, dado que A es computable, lo cual es una contradicción. □

3.5. Anticadenas en los grados de Turing

En este capítulo se presentarán teoremas que, con base en todo el esfuerzo realizado anteriormente, tendrán como consecuencia el siguiente resultado debido a Kleene y Post (1954):

Teorema 3.5.1 (Kleene-Post)**.** *Existen conjuntos $A, B \leq_T \emptyset'$ tales que $A|_T B$ y, por lo tanto, $\emptyset <_T A, B <_T \emptyset'$.*

Antes de comenzar con la prueba de este teorema necesitamos las siguientes definiciones:

Definición 3.5.1. Sean $A \neq \emptyset$ un conjunto y $n \in \omega$. Se define:

1. $A^n = \{s | \{0, 1, \ldots, n-1\} \to A : \text{s es función}\}$,
2. $A^{<\omega} = \bigcup_{n\in\omega} A^n$ y
3. Si $s \in A^n$, se denomina a n como la *longitud* de s y se denota por $l(s)$. ♡

Definición 3.5.2. Sean $u, v \in A^{<\omega}$. Decimos que u extiende a v, representado por $u \succ v$ si $l(u) > l(v)$ y, además, para toda $m < l(v)$ se tiene que $u(m) = v(m)$. ♡

Lema 3.5.2. *Existe una biyección recursiva entre $\omega^{<\omega}$ y ω.*

Demostración. Consideremos la siguiente función $\eta : \omega^{<\omega} \to \omega$

$$\eta(s) = \prod_{k=1}^{n} \Pr(k)^{s(k)}$$

la cual es recursiva por ser la composición de las funciones recursivas $\Pr(k)$ y x^y y por el Teorema 2.3.3, y consideremos la función $g : \omega \to \omega^{<\omega}$

$$g(n) = (1 \operatorname{Gl} n, \ldots, \mathfrak{L}(n) \operatorname{Gl} n),$$

la cual es recursiva por la Ecuación 5 y la Ecuación 6 y sabemos que es una biyección, por el Teorema fundamental de la aritmética. Notemos que la función η induce un orden en $\omega^{<\omega}$, es decir, para todo $u, v \in \omega^{<\omega}$ decimos que $u <_\eta v \iff \eta(u) < \eta(v)$. □

Demostración. Del teorema Teorema 3.5.1. Para esta prueba, construiremos las funciones características de estos conjuntos paso a paso, de tal manera que $\chi_A = \bigcup_{s\in\omega} \sigma_s$ y $\chi_B = \bigcup_{s\in\omega} \tau_s$, donde σ_s y τ_s son las funciones parciales construidas en el paso $s \in \omega$.

La construcción de σ_s y τ_s en el paso s se realizará de tal manera que sean computables en $\emptyset'$, por lo que las sucesiones $\{\sigma_s\}_{s\in\omega}$ y $\{\tau_s\}_{s\in\omega}$ son $\emptyset' - computables$, de esto se desprende que $A, B \leq_T \emptyset'$.

De esto, para obtener el resultado basta con cumplir las siguientes condiciones:

1. para todo $e \in \omega$, la función característica de A no es la $e-$ésima función $B-$computable, es decir: $\forall e \in \omega$, $\chi_A \neq \Psi_e^B$, y
2. para todo $e \in \omega$, la función característica de B no es la $e-$ésima función $A-$computable, es decir: $\forall e \in \omega$, $\chi_B \neq \Psi_e^A$

para poder asegurar que $A \not\leq_T B$ y $B \not\leq_T A$.

Para comenzar, definimos $\sigma_0 = \tau_0 = \emptyset$. Supongamos ahora que hemos construido σ_s y τ_s . Entonces para el paso impar $s+1 = 2e+1$, definimos $n = |\sigma_s| = \text{mín}_x[x \notin \text{Dom}(\sigma_s)]$. Usando a $\emptyset'$ como oráculo, nos preguntamos si el siguiente enunciado es cierto o falso:

$$(\exists \rho)[\rho \succ \tau_s \quad \& \quad \Psi_e^{\rho}(n) \downarrow]. \tag{3.1}$$

Notemos que el primer conyunto del enunciado es una relación recursiva entre sucesiones mientras que el segundo conyunto es computable por la fórmula (37), por lo que la expresión (3.1) es un enunciado Σ_1 y por el Teorema 3.4.1 es $\emptyset'$-computable.

Ahora supongamos que se satisface la fórmula (3.1). Elegimos entonces el ρ más pequeño (de acuerdo a $<_\eta$ definido en el Lema 3.5.2) que satisfaga la expresión y definimos definimos: $\tau_{s+1} = \rho$ y $\sigma_{s+1}(n) = 1 - \Psi_e^{\rho}(n)$. Así, se tiene que $\sigma_{s+1}(n) \neq \Psi_e^{\rho}(n)$.

Ahora bien, suponiendo que no se satisface la fórmula (3.1), definimos $\sigma_{s+1} = \sigma_s^\frown 0$ y $\tau_{s+1} = \tau_s^\frown 0$. En cualquier caso, $|\sigma_{s+1}|$, $|\tau_{s+1}| \geq s+1$ y si $f \succeq \sigma_{s+1}$ y $g \succeq \tau_{s+1}$, entonces $f(n) \neq \Psi_e^g(n)$, donde $n = |\sigma_s|$.

Para el paso $s+1 = 2e+2$ procedemos análogamente que en el paso $s+1 = 2e+1$, pero intercambiando los papeles de σ_s y τ_s. Esto es: definimos $n = |\tau_s| = \text{mín}_x[x \notin \text{Dom}(\tau_s)]$. Usando a $\emptyset'$ como oráculo, nos preguntamos si el siguiente enunciado es cierto o falso:

$$(\exists \rho)[\rho \succ \sigma_s \quad \& \quad \Psi_e^{\rho}(n) \downarrow]. \tag{3.2}$$

Notemos que el primer conyunto del enunciado es una relación recursiva entre sucesiones mientras que el segundo conyunto es computable por la fórmula (37). Por lo que la expresión (3.2) es un enunciado Σ_1 y, por el Teorema 3.4.1, es $\emptyset'$-computable.

Ahora supongamos que se satisface la fórmula (3.2). Elegimos entonces el ρ más pequeño (de acuerdo a $<_\eta$ definido en el Lema 3.5.2) que satisfaga la expresión y definimos: $\sigma_{s+1} = \rho$ y $\tau_{s+1}(n) = 1 - \Psi_e^{\rho}(n)$, así se tiene que $\tau_{s+1}(n) \neq \Psi_e^{\rho}(n)$.

Ahora bien, suponiendo que no se satisface la fórmula (3.2), definimos $\tau_{s+1} = \tau_s^\frown 0$ y $\sigma_{s+1} = \sigma_s^\frown 0$.

En cualquier caso, $|\sigma_{s+1}|$, $|\tau_{s+1}| \geq s+1$. Por lo tanto, $\chi_A = \bigcup_s \sigma_s$ y $\chi_B = \bigcup_s \tau_s$ están definidas para todos los argumentos. Finalmente veamos que $\chi_A \neq \Psi_e^B$ y $\chi_B \neq \Psi_e^A$ para todo $e \in \omega$. Sea $e \in \omega$, por la construcción de las sucesiones $\{\sigma_s\}_{s\in\omega}$ y $\{\tau_s\}_{s\in\omega}$ sabemos que existen σ_l y τ_k tales que si $n = |\tau_{k-1}|$ y $m = |\sigma_{l-1}|$ entonces $\tau_k(m) \neq \Psi_e^{\sigma_l}(m)$ y $\sigma_l(n) \neq \Psi_e^{\tau_k}(n)$. Finalmente, por el Lema 2.1.5 y por la definición de χ_A y χ_B se tiene que $\chi_A \neq \Psi_e^B$ y $\chi_B \neq \Psi_e^A$. De donde se concluye que $A|_T B$. □

Ahora extenderemos esta técnica para encontrar una anticadena de tamaño numerable de conjuntos no comparables entre $\emptyset$ y $\emptyset'$.

Definición 3.5.3. Para $n \in \omega$ definimos la suma directa finita de una sucesión de conjuntos $\{A_i\}_{i<n}$ como sigue:

$$\bigoplus\{A_j\}_{j<n} := \{\langle x, y\rangle | x \in A_y\}_{y<n}. \qquad \heartsuit$$

Definición 3.5.4. Definimos la suma directa infinita de una sucesión de conjuntos $\{A_i\}_{i\in\omega}$ como sigue:

$$\bigoplus\{A_j\}_{j\in\omega} := \{\langle x, y\rangle | x \in A_y\}_{y\in\omega}. \qquad \heartsuit$$

Definición 3.5.5. Llamamos a una sucesión numerable de conjuntos $\{A_i\}_{i\in\omega}$ *computablemente independiente* si para cada i, se tiene que $A_i \not\leq_T \oplus\{A_j : j \neq i\}$, donde $\oplus\{A_j : j \neq i\}$ se define como en la Definición 3.5.4. $\heartsuit$

Teorema 3.5.3. *Existe una sucesión de conjuntos $\{A_i\}_{i\in\omega}$ computablemente independiente tal que para todo $i \in \omega$ se tiene que $A_i \leq_T \emptyset'$.*

Demostración. Definiremos a la familia $\{A_i\}_{i\in\omega}$ recursivamente. Sea $b_r : \omega^2 \to \omega$ una biyección recursiva . Nos auxiliaremos en la construcción de cadenas finitas $\{\rho_j^s | j, s \in \omega\}$, de modo que $\forall j, A_j = \bigcup_s \rho_j^s$. Para el paso $s = 0$, sea $\rho_j^s = \emptyset$ para todo j. Supongamos que $s = b_r(e, i)$ y $\rho_j^s = \emptyset$ para todos, excepto una cantidad finita de $j \in \omega$. Sea $n = |\rho_i^s|$. Consideremos el enunciado siguiente:

Existe una sucesión finita $\sigma_1, \sigma_2, \ldots, \sigma_k$ de sucesiones finitas (o formalmente, existe un código para tal sucesión) y $m \in \omega$ tales que $\forall j \leq k$, $\rho_j^s \preceq \sigma_j$ y $\Psi_e^{\oplus\{\sigma_j : j\neq i\}}(n) = m$.

Este enunciado es Σ_1 por lo que, usando a $\emptyset'$ como oráculo, podemos examinar si se satisface o no. Supongamos que el enunciado se satisface, entonces hacemos:

$$\rho_i^{s+1}(n) = 1 \dot{-} m \text{ y } \rho_j^{s+1} = \sigma_j \text{ para } j \neq i \text{ con } j \leq k \ .$$

En cambio, si el enunciado no se satisface hacemos:

$$\rho_i^{s+1}(n) = 0 \text{ y } \rho_j^{s+1} = \rho_j^s \hat{} 0 \text{ para } j \neq i \text{ con } j \leq k \ .$$

De este modo, A_i no es computable en $\oplus\{\sigma_j : j \neq i \text{ con } j \leq k\}$. Sea $i, e \in \omega$. Supongamos que $\Psi_e^{\oplus\{A_j : j \neq i, j \leq k_s\}}(s) = m$. Entonces, por el Lema 2.1.5, $\Psi_e^{\oplus\{A_j : j \neq i\}(s)} = m$, pero en este caso $\chi_{A_i}(n) \neq m$. Por el contrario, supongamos que $\Psi_e^{\oplus\{A_j : j \neq i, j \leq k_s\}}(s)$ no está definida. Entonces $\Psi_e^{\oplus\{A_j : j \neq i\}}$ tampoco lo está, puesto que ninguna extensión de las $\rho_j^s (j \neq i)$ hacen que Ψ_e se detenga.

Por lo tanto, $\Psi_e^{\oplus\{A_j : j \neq i\}}(n)$ no coincide con $\chi_{A_i}(n)$. □

Ahora usaremos esta técnica para demostrar el siguiente:

Teorema 3.5.4. *Existen $2^{\aleph_0}$ grados mutuamente incomparables.*

Para la prueba de este teorema necesitaremos algunas definiciones relacionadas con árboles:

Definición 3.5.6. Un *árbol* en un conjunto A es un subconjunto T del conjunto parcialmente ordenado $(A^{<\omega}, \subseteq)$ tal que si $s \in T$ y $n < l(s)$, entonces $s \restriction n$ también está en T y, además, para todo $u \in T$, el conjunto $\{v | v < u\}$ es bien ordenado. ♡

Definición 3.5.7. Una *rama* del árbol T es una cadena infinita x de elementos de T. ♡

Demostración. Del Teorema 3.5.4. Para esta prueba construiremos una sucesión de árboles T_n $(n \in \omega)$ definidos recursivamente, de tal manera que las ramas del árbol $T = \bigcap_{n \in \omega} T_n$ sean funciones totales mutuamente incomparables, i.e., $f, g \in [T], f \neq g \implies f|_T g$.

Ahora procedemos recursivamente:

$T_0 = \{\emptyset\}$. Ahora supongamos que hemos construido T_e ($e \in \omega$), definamos L_e como el conjunto de las hojas del árbol T_e de la siguiente manera:

$$L_e = \{\sigma : \sigma \in T_e \quad \& \quad (\forall \tau \succ \sigma)[\tau \notin T_e]\}.$$

Ahora definimos los sucesores de las hojas:

$$S_e = \{\sigma\hat{}0 : \sigma \in L_e\} \bigcup \{\sigma\hat{}1 : \sigma \in L_e\}.$$

Supongamos que enumeramos a los elementos de S_e, es decir, $S_e = \{\rho_i | i < 2^{e+1}\}$. Sean $i, j < 2^{e+1}$ con $i \neq j$. Ahora, utilizaremos la técnica del Teorema 3.5.1 para reemplazar los sucesores ρ_i y ρ_j por otras cadenas $\lambda_i \succ \rho_i$ y $\lambda_j \succ \rho_j$ que satisfagan:

$$(\forall f \succ \lambda_i)(\forall g \succ \lambda_j)[\Psi_e^f \neq g \quad \& \quad \Psi_e^g \neq f].$$

Veamos cómo sería esto para el árbol T_1:

$$L_1 = \{\emptyset\} \text{ y } S_1 = \{\emptyset\hat{}0, \emptyset\hat{}1\} = \{\langle 0\rangle = \rho_0, \langle 1\rangle = \rho_1\}.$$

Entonces usando a $\emptyset'$ como oráculo, nos preguntamos si el siguiente enunciado es cierto o falso:

$$(\exists \rho)[\rho \succ \langle 0\rangle \quad \& \quad \Psi_0^\rho(0) \downarrow]. \tag{3.3}$$

Ahora supongamos que se satisface la fórmula (3.3). Elegimos entonces el ρ más pequeño que satisfaga la expresión y definimos: $\tau_1 = \rho$ y $\sigma_1(0) = 1 - \Psi_0^\rho(0)$, así se tiene que $\sigma_1(0) \neq \Psi_0^\rho(0)$. Ahora bien, suponiendo que no se satisface la fórmula (3.3), definimos $\sigma_1 = \langle 0\rangle$ y $\tau_1 = \langle 1\rangle$.

De esta manera podemos asegurar que para toda $g \succ \tau_1$ y $f \succ \sigma_1$ se cumplirá que $\Psi_0^g \neq f$, sin embargo, falta asegurar que se tendrá que $\Psi_0^f \neq g$ para lo cual hacemos lo siguiente:

Sea $n = |\tau_1|$. Usando a $\emptyset'$ como oráculo, nos preguntamos si el siguiente enunciado es cierto o falso:

$$(\exists \delta)[\rho \succ \sigma_1 \quad \& \quad \Psi_0^\delta(n) \downarrow]. \tag{3.4}$$

Ahora supongamos que se satisface la fórmula (3.4). Elegimos entonces el δ más pequeño que satisfaga la expresión y definimos: $\lambda_0 = \delta$ y $\lambda_1(n) = 1 - \Psi_0^\delta(n)$ así se tiene que $\lambda_1(n) \neq \Psi_0^\delta(n)$.

Ahora bien, suponiendo que no se satisface la fórmula (3.3), definimos $\lambda_0 = \sigma_1{}^\frown 0$ y $\lambda_1 = \tau_1{}^\frown 1$. En cualquier caso, $|\lambda_0|, |\lambda_1| > 0$.

El procedimiento empleado con las hojas $\langle 0 \rangle$, y $\langle 1 \rangle$ será el mismo a realizar para cualquier par de hojas ρ_i y ρ_j del árbol T_e, con $i < j < 2^{e+1}$.

Notemos que en la construcción del árbol T_{e+1}, la hoja ρ_i se verá extendida sucesivamente por hojas $\rho_i = \delta_0 \prec \delta_1 \prec \cdots \prec \delta_{2^{e+1}-1} = \lambda_i$, siendo esta última la hoja que se incorpora al árbol T_{e+1}.

Esto es,

$$T_{e+1} = \{\lambda_i \succ \rho_i | i < 2^{e+1}\},$$

donde λ_i es el resultado final de todas las extensiones de ρ_i.

Probemos que si $f \neq g \in [T]$, entonces son mutuamente no reducibles. Sea $e \in \omega$ y $m >$ mín$\{n \in \omega : f(n) \neq g(n)\}$ tales que Ψ_e es equivalente a la máquina Ψ_m, la cual existe debido al Teorema 2.4.3. Así, $\sigma = T_m \cap f \neq T_m \cap g = \tau$ y, por la construcción de T_m, existen k y l tales que $\Psi_m^\sigma(k) \neq \tau(k)$ y $\Psi_m^\tau(l) \neq \sigma(l)$. Por lo tanto, $\Psi_e^f(k)$ computa lo mismo que $\Psi_m^f(k)$, que es un resultado distinto de $\tau(k) = g(k)$ y, por otro lado, $\Psi_e^g(l)$ computa lo mismo que $\Psi_m^g(l)$, que es un resultado distinto $\sigma(l) = f(l)$.

Como $e \in \omega$ es arbitraria, tenemos que f y g son mutuamente no reducibles. □

Algo notable de las pruebas de los teoremas anteriores es que para la construcción de los conjuntos irreducibles según Turing no se eligieron características particulares intrínsecas a cada conjunto sino que se extendían conforme a, por ejemplo, la expresión 3.1 y la información obtenida al consultar a $\emptyset'$ como oráculo. Esto quiere decir que usando una expresión similar a la 3.1, un conjunto arbitrario C y a su salto (C') como oráculo es posible probar versiones relativizadas de los teoremas previos para generar las mismas anticadenas entre C y C', como se puede ver en Soare (2016) y originalmente en Kleene y Post (1954). De esta manera, podemos observar que cada vez que se aplica el salto de Turing dejamos atrás una cantidad infinita de grados incomparables entre sí. Estos resultados, más las pruebas que aparecen en la sección 4 de Kleene y Post (1954) permiten describir la estructura del orden de Turing como una semiretícula superior.

Actualmente, el problema más importante acerca del orden de Turing concierne con su grupo de automorfismos $Aut(\mathbf{D})$. Un automorfismo f de un conjunto parcialmente ordenado $(A, <)$ es una biyección de A en sí mismo que preserva el orden, es decir, $a < b$ si y solo si $f(a) < f(b)$. Un orden parcial es más homogéneo en tanto que su grupo de automorfismos es más grande. Por ejemplo, $\mathbb{Q}$ cumple que cada isomorfismo parcial con dominio finito se puede extender a un automorfismo de $\mathbb{Q}$. En cambio, para ω y cualquier otro conjunto bien ordenado, $Aut(\omega)$ es trivial, esto es, la identidad es el único automorfismo. Slaman y Woodin (1986) conjeturan que el orden de Turing también es así. Al respecto se sabe que $Aut(\mathbf{D})$ es a lo más numerable como se discute en Slaman (2005) o Kjos-Hanssen (2018).

Anexos

I. Continuación de prueba de Ejemplo 3

Supongamos que $i > 1$:

$$q_1\overline{(k_1, k_2, \ldots, k_n)} \to q_{n+2}B\overline{(k_1 - 1, k_2, \ldots, k_n)} \quad \text{por e con } j = 1)$$

$$q_{n+2}B\overline{(k_1 - 1, k_2, \ldots, k_n)} \to Bq_1\overline{(k_1 - 1, k_2, \ldots, k_n)} \quad \text{por g con } j = 1)$$

$$Bq_1\overline{(k_1 - 1, k_2, \ldots, k_n)} \to Bq_{n+2}B\overline{(k_1 - 2, k_2, \ldots, k_n)} \quad \text{por e con } j = 1)$$

$$Bq_{n+2}B\overline{(k_1 - 1, k_2, \ldots, k_n)} \to B1^2q_1\overline{(k_1 - 3, k_2, \ldots, k_n)} \quad \text{por g con } j = 1)$$

$$\ldots$$

$$B^{k_1+1}B \ldots BB^{k_{i-1}+1}q_{i-1}B\overline{(k_i, k_{i+1}, \ldots, k_n)} \to B^{k_1+1}B \ldots BB^{k_{i-1}+1}Bq_i\overline{(k_i, k_{i+1}, \ldots, k_n)} \quad \text{por f con } j = i - 1)$$

$$B^{k_1+1}B \ldots BB^{k_{i-1}+1}Bq_i\overline{(k_i, k_{i+1}, \ldots, k_n)} \to B^{k_1+1}B \ldots BB^{k_{i-1}+1}Bq_iB\overline{(k_i - 1, k_{i+1}, \ldots, k_n)} \quad \text{por a)}$$

$$B^{k_1+1}B \ldots BB^{k_{i-1}+1}Bq_iB\overline{(k_i - 1, k_{i+1}, \ldots, k_n)} \to B^{k_1+1}B \ldots BB^{k_{i-1}+1}BBq_{n+i+1}\overline{(k_i - 1, k_{i+1}, \ldots, k_n)} \quad \text{por b)}$$

$$B^{k_1+1}B \ldots BB^{k_{i-1}+1}BBq_{n+i+1}\overline{(k_i - 1, k_{i+1}, \ldots, k_n)} \to B^{k_1+1}B \ldots BB^{k_{i-1}+1}BB1q_{n+i+1}\overline{(k_i - 2, k_{i+1}, \ldots, k_n)} \quad \text{por c)}$$

$$\ldots$$

$$B^{k_1+1}B \ldots BB^{k_{i-1}+1}BB1^{k_i}q_{n+i+1}B\overline{(k_{i+1}, \ldots, k_n)} \to B^{k_1+1}B \ldots BB^{k_{i-1}+1}BB1^{k_i}Bq_{i+1}\overline{(k_{i+1}, \ldots, k_n)} \quad \text{por d)}$$

$$B^{k_1+1}B \ldots BB^{k_{i-1}+1}BB1^{k_i}Bq_{i+1}\overline{(k_{i+1}, \ldots, k_n)} \to B^{k_1+1}B \ldots BB^{k_{i-1}+1}BB1^{k_i}Bq_{n+(i+1)+1}B\overline{(k_{i+1} - 1, \ldots, k_n)} \quad \text{por a con } j = i + 1)$$

$$\ldots$$

$$B^{k_1+1}B \ldots BB^{k_{i-1}+1}BB1^{k_i}B \ldots B^{k_n+1}q_nB \to B^{k_1+1}B \ldots BB^{k_{i-1}+1}BB1^{k_i}B \ldots B^{k_n+1}Bq_{n+1}B \quad \text{por f con } j = n)$$

Podemos observar que la máquina descrita borra un símbolo 1 de bloque en la posición i-ésima y deja todos los demás símbolos de ese bloque intactos mientras que para el resto de los bloques se borran todos los símbolos 1, teniendo así el resultado.

II. Continuación prueba del Lema 2.2.4

Esta máquina tendrá la tarea de mover uno a uno los primeros p parámetros a la extrema derecha, la cuál podrá encontrar puesto que será la primera sucesión de dos símbolos B que encontrará. Formalmente, para todo $1 \leq i \leq p$, R_p contendrá una lista como la siguiente:

a) $q_i 1 R q_i$

b) $q_i B S_{1+i} q_{p+1}$

c) $q_{p+1} 1 R q_{p+1}$

d) $q_{p+1} S_{1+i} R q_{p+1}$

e) $q_{p+1} B R q_{p+2}$

f) $q_{p+2} 1 R q_{p+1}$

g) $q_{p+2} B 1 q_{p+3}$

h) $q_{p+3} 1 L q_{p+3}$

i) $q_{p+3} S_{i+1} L q_{p+3}$

j) $q_{p+3} B L q_{p+4}$

k) $q_{p+4} 1 L q_{p+3}$

l) $q_{p+4} B R q_{p+5}$

m) $q_{p+5} B R q_{p+5}$

n) $q_{p+5} 1 B q_{p+6}$

ñ) $q_{p+6} B R q_{p+6}$

o) $q_{p+6} 1 R q_{p+1}$

p) $q_{p+6} S_{i+1} B q_{p+7+i}$, excepto para i=p

q) $q_{p+7+i} B R q_{i+1}$

r) $q_{p+6} S_{p+1} B q_{2p+7}$

s) $q_{2p+7} B R q_{2p+8}$

Ahora, si $p > 0$ y $n > 0$ tenemos que con respecto a R_p:

$$q_1 \overline{k_1, \ldots, k_p, m_1, \ldots, m_n} \to 1 q_1 1 \overline{k_1 - 2, \ldots, k_p, m_1, \ldots, m_n} \quad \text{por a)}$$

$$\ldots$$

$$1^{k_1} q_1 1 B \overline{k_2, \ldots, k_p, m_1, \ldots, m_n} \to 1^{k_1+1} q_1 B \overline{k_2, \ldots, k_p, m_1, \ldots, m_n} \quad \text{por a)}$$

$$1^{k_1+1}q_1B\overline{k_2,\ldots,k_p,m_1,\ldots,m_n} \to 1^{k_1+1}q_{p+1}S_2\overline{k_2,\ldots,k_p,m_1,\ldots,m_n} \quad \text{por b)}$$

$$1^{k_1+1}q_{p+1}S_2\overline{k_2,\ldots,k_p,m_1,\ldots,m_n} \to 1^{k_1+1}S_2q_{p+1}1\overline{k_2-1,\ldots,k_p,m_1,\ldots,m_n} \quad \text{por d)}$$

$$1^{k_1+1}S_2q_{p+1}1\overline{k_2-1,\ldots,k_p,m_1,\ldots,m_n} \to 1^{k_1+1}S_21q_{p+1}1\overline{k_2-2,\ldots,k_p,m_1,\ldots,m_n} \quad \text{por c)}$$

$$\ldots$$

$$1^{k_1+1}S_21^{k_2}q_{p+1}1B\overline{k_3,\ldots,k_p,m_1,\ldots,m_n} \to 1^{k_1+1}S_21^{k_2+1}q_{p+1}B\overline{k_3,\ldots,k_p,m_1,\ldots,m_n} \quad \text{por c)}$$

$$1^{k_1+1}S_21^{k_2+1}q_{p+1}B\overline{k_3,\ldots,k_p,m_1,\ldots,m_n} \to 1^{k_1+1}S_21^{k_2+1}Bq_{p+2}1\overline{k_3-1,\ldots,k_p,m_1,\ldots,m_n} \quad \text{por e)}$$

$$1^{k_1+1}S_21^{k_2+1}Bq_{p+2}1\overline{k_3-1,\ldots,k_p,m_1,\ldots,m_n} \to 1^{k_1+1}S_21^{k_2+1}B1q_{p+1}1\overline{k_3-2,\ldots,k_p,m_1,\ldots,m_n} \quad \text{por f)}$$

$$\ldots$$

$$\overline{k_1}S_2\overline{k_2,\ldots,k_p,m_1,\ldots,m_n}q_{p+1}B \to \overline{k_1}S_2\overline{k_2,\ldots,k_p,m_1,\ldots,m_n}Bq_{p+2}B \quad \text{por e)}$$

$$\overline{k_1}S_2\overline{k_2,\ldots,k_p,m_1,\ldots,m_n}Bq_{p+2}B \to \overline{k_1}S_2\overline{k_2,\ldots,k_p,m_1,\ldots,m_n}Bq_{p+3}1 \quad \text{por g)}$$

$$\overline{k_1}S_2\overline{k_2,\ldots,k_p,m_1,\ldots,m_n}Bq_{p+3}1 \to \overline{k_1}S_2\overline{k_2,\ldots,k_p,m_1,\ldots,m_n}q_{p+3}B1 \quad \text{por h)}$$

$$\overline{k_1}S_2\overline{k_2,\ldots,k_p,m_1,\ldots,m_n}q_{p+3}B1 \to \overline{k_1}S_2\overline{k_2,\ldots,k_p,m_1,\ldots,m_n-1}q_{p+4}1B1 \quad \text{por j)}$$

$$\overline{k_1}S_2\overline{k_2,\ldots,k_p,m_1,\ldots,m_n-1}q_{p+4}1B1 \to \overline{k_1}S_2\overline{k_2,\ldots,k_p,m_1,\ldots,m_n-2}q_{p+3}11B1 \quad \text{por k)}$$

$$\ldots$$

$$\overline{k_1}q_{p+3}S_2\overline{k_2,\ldots,k_p,m_1,\ldots,m_n}B1 \to \overline{k_1-1}q_{p+3}1S_2\overline{k_2,\ldots,k_p,m_1,\ldots,m_n}B1 \quad \text{por i)}$$

$$\ldots$$

$$q_{p+3}B\overline{k_1}S_2\overline{k_2,\ldots,k_p,m_1,\ldots,m_n}B1 \to q_{p+4}BB\overline{k_1}S_2\overline{k_2,\ldots,k_p,m_1,\ldots,m_n}B1 \quad \text{por j)}$$

$$q_{p+4}BB\overline{k_1}S_2\overline{k_2,\ldots,k_p,m_1,\ldots,m_n}B1 \to Bq_{p+5}B\overline{k_1}S_2\overline{k_2,\ldots,k_p,m_1,\ldots,m_n}B1 \quad \text{por l)}$$

$$Bq_{p+5}B\overline{k_1}S_2\overline{k_2,\ldots,k_p,m_1,\ldots,m_n}B1 \to BBq_{p+5}S_1\overline{k_1-1}S_2\overline{k_2,\ldots,k_p,m_1,\ldots,m_n}B1 \quad \text{por m)}$$

$$BBq_{p+5}S_1\overline{k_1-1}S_2\overline{k_2,\ldots,k_p,m_1,\ldots,m_n}B1 \to BBq_{p+6}B\overline{k_1-1}S_2\overline{k_2,\ldots,k_p,m_1,\ldots,m_n}B1 \quad \text{por n)}$$

$$BBq_{p+6}B\overline{k_1-1}S_2\overline{k_2,\ldots,k_p,m_1,\ldots,m_n}B \to BBBq_{p+6}S_1\overline{k_1-2}S_2\overline{k_2,\ldots,k_p,m_1,\ldots,m_n}B1 \quad \text{por ñ)}$$

$$BBBq_{p+6}1\overline{k_1-2}S_2\overline{k_2,\ldots,k_p,m_1,\ldots,m_n}B1 \to BBB1q_{p+1}\overline{k_1-2}S_2\overline{k_2,\ldots,k_p,m_1,\ldots,m_n}B1 \quad \text{por o)}$$

$$\ldots$$

$$B^{k_1}q_{p+5}1S_2\overline{k_2,\ldots,k_p,m_1,\ldots,m_n}B\overline{k_1} \to B^{k_1}q_{p+6}BS_2\overline{k_2,\ldots,k_p,m_1,\ldots,m_n}B\overline{k_1} \quad \text{por n)}$$

$$B^{k_1}q_{p+6}BS_2\overline{k_2,\ldots,k_p,m_1,\ldots,m_n}B\overline{k_1} \to B^{k_1+1}q_{p+6}S_2\overline{k_2,\ldots,k_p,m_1,\ldots,m_n}B\overline{k_1} \quad \text{por ñ)}$$

$$B^{k_1+1}q_{p+6}S_2\overline{k_2,\ldots,k_p,m_1,\ldots,m_n}B\overline{k_1} \to B^{k_1+1}q_{p+8}B\overline{k_2,\ldots,k_p,m_1,\ldots,m_n}B\overline{k_1} \quad \text{por p)}$$

$$B^{k_1+1}q_{p+8}B\overline{k_2,\ldots,k_p,m_1,\ldots,m_n}B\overline{k_1} \to B^{k_1+1}q_21\overline{k_2-1,\ldots,k_p,m_1,\ldots,m_n}B\overline{k_1} \quad \text{por q)}$$

$$\ldots$$

$$B^{k_1+1}\overline{k_2}q_2B\overline{k_3,\ldots,k_p,m_1,\ldots,m_n}B\overline{k_1} \to B^{k_1+1}\overline{k_2}q_{p+1}S_3\overline{k_3,\ldots,k_p,m_1,\ldots,m_n}B\overline{k_1} \quad \text{por b)}$$

$$\ldots$$

$$q_{p+6}S_{p+1}\overline{m_1,\ldots,m_n,k_1,\ldots,k_p} \to q_{2p+7}B\overline{m_1,\ldots,m_n,k_1,\ldots,k_p} \quad \text{por r)}$$

$$q_{2p+7}B\overline{m_1,\ldots,m_n,k_1,\ldots,k_p} \to Bq_{2p+8}\overline{m_1,\ldots,m_n,k_1,\ldots,k_p} \quad \text{por s)}$$

El estado q_{2p+8} no aparece al inicio de ninguna cuádrupla por lo que la descripción instantánea $Bq_{2p+8}\overline{m_1,\ldots,m_n,k_1,\ldots,k_p}$ es final y se tiene el resultado.

III. Prueba del Teorema 2.3.9

Funciones y predicados que tienen que ver con los números de Gödel de expresiones y sucesiones de expresiones

$$n\,\mathrm{Gl}\,x = \mathfrak{M}^x_{y=0}[(\mathrm{Pr}(n)^y|x) \bigwedge \sim (\mathrm{Pr}(n)^{y+1}|x)]. \tag{5}$$

Si $x = \mathrm{gn}(M)$, donde M consiste en los símbolos $\gamma_1, \ldots, \gamma_p$ y si $0 < n \leq p$, entonces $n\,\mathrm{Gl}\,x$ es el número de Gödel de γ_n mientras que si $n = 0$ o $n > p$ entonces $n\,\mathrm{Gl}\,x = 0$. Si x es el número de Gödel de una sucesión de expresiones $M_1, \ldots, M_p$, y si $0 < n \leq p$ entonces $n\,\mathrm{Gl}\,x = \mathrm{gn}(M_n)$ mientras que si $n = 0$ o $n > p$, entonces $n\,\mathrm{Gl}\,x = 0$.

$$\mathfrak{L}(x) = \mathfrak{M}^x_{y=0}[(y\,\mathrm{Gl}\,x > 0) \wedge \bigwedge_{i=0}^{x}((y+i+1)\,\mathrm{Gl}\,x = 0)] \tag{6}$$

Si $x = \mathrm{gn}(M)$, entonces $\mathfrak{L}(x)$ es el número de símbolos que ocurren en M. Si x es el número de Gödel de la sucesión de expresiones $M_1, \ldots, M_n$, entonces $\mathfrak{L}(x) = n$.

$$\mathrm{GN}\,(x) \iff \sim \bigvee_{y=1}^{\mathfrak{L}(x)+1} [(y\,\mathrm{Gl}\,x = 0) \wedge ((y+1)\,\mathrm{Gl}\,x \neq 0)] \tag{7}$$

$\mathrm{GN}(x)$ es cierto sí y solo sí existen enteros positivos a_k, $1 \leq k \leq n$, tales que

$$x = \prod_{k=1}^{n} \mathrm{Pr}(k)^{a_k}.$$

$$\mathrm{Term}(x,z) \iff \mathrm{GN}(z) \wedge \bigvee_{n=0}^{\mathfrak{L}(z)} [(x = n\,\mathrm{Gl}\,z) \wedge (n \neq 0)]. \tag{8}$$

$\mathrm{Term}(x,z)$ es cierta si y sólo si $z = \prod_{k=1}^{n} \mathrm{Pr}(k)^{a_k}$ para $a_k > 0$ adecuados y además $x = a_k$ para alguna k, $1 \leq k \leq n$.

$$x \hat{} y = x \cdot \prod_{i=0}^{\mathfrak{L}(y) \dot{-} 1} \mathrm{Pr}(\mathfrak{L}(x) + i + 1)^{(i+1)\mathrm{Gl}y}. \tag{9}$$

Si M y N son expresiones, entonces $\mathrm{gn}(MN) = \mathrm{gn}(M) \hat{} \mathrm{gn}(N)$. Si x y y son números de Gödel de las sucesiones de expresiones $M_1, \ldots, M_n$ y $N_1, \ldots, N_p$ respectivamente, entonces $x \hat{} y$ es el número de Gödel de la sucesión $M_1, \ldots, M_n, N_1, \ldots, N_p$.

Funciones y predicados que tienen que ver con la estructura básica de Máquinas de Turing

$$\mathrm{EI}(x) \iff \bigvee_{y=0}^{x}(x = 4y + 9) \tag{10}$$

$\mathrm{EI}(x)$ es cierto si y sólo sí x es el número asignado a uno de los estados q_i.

$$\mathrm{ei}(x) = \chi_{\{y|\mathrm{EI}(y)\}} \tag{11}$$

$\mathrm{ei}(x)$ es 1 si x es el número asignado a uno de los estados q_i y 0 en otro caso.

$$\mathrm{Al}(x) \iff \bigvee_{y=0}^{x}(x = 4y + 7) \tag{12}$$

$\mathrm{Al}(x)$ es cierto si y sólo si x es el número asignado a uno de los S_i.

$$\mathrm{Imp}(x) \iff \bigvee_{y=0}^{x}(x = 2y + 3) \tag{13}$$

$\mathrm{Imp}(x)$ es cierto si y sólo si x es un número impar mayor o igual a tres.

$$\mathrm{Cuad}(x) \iff \mathrm{GN}(x) \wedge (\mathfrak{L}(x) = 4) \wedge \mathrm{EI}(1\,\mathrm{Gl}\,x) \wedge Al(2\,\mathrm{Gl}\,x) \wedge \mathrm{Imp}(3\,\mathrm{Gl}\,x) \wedge \mathrm{EI}(4\,\mathrm{Gl}\,x). \tag{14}$$

$\mathrm{Cuad}(x)$ es cierto si y sólo si $\mathrm{Exp}(x)$ es una cuádrupla.

$$\mathrm{Inc}(x, y) \iff \mathrm{Cuad}(x) \wedge \mathrm{Cuad}(y) \wedge (1\,\mathrm{Gl}\,x = 1\mathrm{Gl}y) \wedge (2\,\mathrm{Gl}\,x = 2\,\mathrm{Gl}\,y) \wedge (x \neq y) \tag{15}$$

$\mathrm{Inc}(x, y)$ es cierto si y sólo si x y y son números de Gödel de dos cuádruplas distintas que empiezan con el mismo primer par de símbolos.

$$\mathrm{TM}(x) \iff \mathrm{GN}(x) \wedge \bigwedge_{n=1}^{\mathfrak{L}(x)} \left[\mathrm{Cuad}(n\,\mathrm{Gl}\,x) \wedge \bigwedge_{m=1}^{\mathfrak{L}(x)} \sim (\mathrm{Inc}(n\,\mathrm{Gl}\,x, m\,\mathrm{Gl}\,x)) \right]. \tag{16}$$

$\mathrm{TM}(x)$ es cierto si y sólo si x es el número de Gödel de una Máquina de Turing.

$$\mathrm{tm}(x) = \chi_{\{y|\mathrm{TM}(y)\}} \tag{17}$$

$\mathrm{tm}(x)$ es 1 si x es el número asignado a una Máquina de Turing y 0 en otro caso.

$$\begin{aligned} \mathrm{MR}(0) &= 2^{11} \\ \mathrm{MR}(n+1) &= 2^{11}\hat{}\mathrm{MR}(n) \end{aligned} \tag{18}$$

Entonces, $\mathrm{MR}(n) = \mathrm{gn}(1^{n+1}) = \mathrm{gn}(\overline{n})$.

$$\begin{aligned} \mathrm{CU}(n,x) &= 0 \text{ si } n \,\mathrm{Gl}\, x \neq 11, \\ \mathrm{CU}(n,x) &= 1 \text{ si } n \,\mathrm{Gl}\, x = 11, \end{aligned} \tag{19}$$

$\mathrm{CU}(n,x)$ es la función característica del predicado $n \,\mathrm{Gl}\, x \neq 11$ y como tal, es recursiva primitiva.

$$\mathrm{Corn}(x) = \sum_{n=1}^{\mathfrak{L}(x)} CU(n,x). \tag{20}$$

Si $x = \mathrm{gn}(M)$, entonces $\mathrm{Corn}(x) = \langle M \rangle$.

$$U(y) = \mathrm{Corn}(\mathfrak{L}(y) \,\mathrm{Gl}\, y). \tag{21}$$

Si y es el número de Gödel de una sucesión de expresiones $M_1, \ldots, M_n$, entonces $U(y) = \langle M_n \rangle$.

$$\mathrm{DI}(x) \iff \mathrm{GN}(x) \wedge \bigvee_{n=1}^{\mathfrak{L}(x)\dot{-}1} \left\{ \mathrm{EI}(n \,\mathrm{Gl}\, x) \wedge \bigwedge_{m=1}^{\mathfrak{L}(x)} [(m=n) \vee \mathrm{Al}(m \,\mathrm{Gl}\, x)] \right\} \tag{22}$$

$\mathrm{DI}(x)$ es cierto si y sólo si x es el número de Gödel de una descripción instantánea.

$$\mathrm{Init}_n(x_1, \ldots, x_n) = 2^9\hat{}\mathrm{MR}(x_1)\hat{}2^7\hat{}\mathrm{MR}(x_2)\hat{}\cdots\hat{}2^7\hat{}\mathrm{MR}(x_n) \tag{23}$$

$\mathrm{Init}_n(x_1, \ldots, x_n) = \quad \mathrm{gn}(q_1\overline{(x_1, \ldots, x_n)})$.

$$\mathrm{Max}(x,y) \iff \mathrm{TM}(x) \wedge \bigvee_{n=1}^{\mathfrak{L}(x)} \left[y = (1 \,\mathrm{Gl}\, (n \,Gl\, x)) \wedge \bigvee_{m=1}^{\mathfrak{L}(x)} \sim (1 \,\mathrm{Gl}\, m > 1 \,\mathrm{Gl}\, n) \right] \tag{24}$$

$\mathrm{Max}(x, y)$ es cierto sí y sólo si x es el número de Gödel de una MT y además y es el máximo índice que aparece como estado en alguna de las cuádruplas de la MT cuyo número es x.

$$\Theta(x) = \mathfrak{M}_{y=0}^{\mathfrak{L}(x)}[\mathrm{Max}(x, y)]. \tag{25}$$

$\Theta(x)$ es la función que asocia a un número x con el máximo índice que aparece como estado en alguna de las cuádruplas de la MT cuyo número es x si ese fuera el caso, o 0 en caso de que x no corresponda al número de Gödel una MT.

$$\mathrm{Tras}(x, k) = \mathrm{tm}(x) * \prod_{t=1}^{\mathfrak{L}(x)} \mathrm{Pr}(t)^{\prod_{s=1}^{\mathfrak{L}(t \,\mathrm{Gl}\, x)} \mathrm{Pr}(s)^{\mathrm{ei}(s \,\mathrm{Gl}\, (t \,\mathrm{Gl}\, x)) * 2 * k + s \,\mathrm{Gl}\, (t \,\mathrm{Gl}\, x)}}. \tag{26}$$

$\mathrm{Tras}(x, k)$ es el número de Gödel de la MT con número x, desplazada por k si es que x es el número de Gödel de una MT o 0 en otro caso.

Funciones y predicados relacionados con la relación $\rightarrow$

$$\begin{aligned} \mathrm{Paso}_1(x, y, z) \iff & \mathrm{DI}(x) \wedge \mathrm{DI}(y) \wedge \mathrm{TM}(z) \wedge \\ & \bigvee_{F=0}^{x} \bigvee_{G=0}^{x} \bigvee_{r=0}^{x} \bigvee_{s=0}^{x} \bigvee_{t=0}^{x} \bigvee_{u=0}^{x} [(x = F \hat{} 2^r \hat{} 2^s \hat{} G) \\ & \wedge (y = F \hat{} 2^t \hat{} 2^u \hat{} G) \wedge \mathrm{EI}(r) \wedge \mathrm{EI}(t) \wedge \mathrm{Al}(s) \wedge \mathrm{Al}(u) \\ & \wedge \mathrm{Term}(2^r \cdot 3^s \cdot 5^u \cdot 7^t, z)]. \end{aligned} \tag{27}$$

$\mathrm{Paso}_1(x, y, z)$ es cierta si y sólo si x y y son números de Gödel de descripciones instantáneas, z es el número de Gödel de una Máquina de Turing Z y $\mathrm{Exp}(x) \rightarrow \mathrm{Exp}(y)$, (Z), en el caso 1 de la Definición 2.1.8.

$$\begin{aligned} \mathrm{Paso}_2(x, y, z) \iff & \mathrm{DI}(x) \wedge \mathrm{DI}(y) \wedge \mathrm{TM}(z) \wedge \\ & \bigvee_{F=0}^{x} \bigvee_{G=0}^{x} \bigvee_{r=0}^{x} \bigvee_{s=0}^{x} \bigvee_{t=0}^{x} [(x = F \hat{} 2^r \hat{} 2^s \hat{} 2^t \hat{} G) \\ & \wedge (y = F \hat{} 2^s \hat{} 2^u \hat{} 2^t \hat{} G) \wedge \mathrm{EI}(r) \wedge \mathrm{EI}(t) \wedge \mathrm{Al}(s) \wedge \mathrm{Al}(t) \\ & \wedge \mathrm{Term}(2^r \cdot 3^s \cdot 5^3 \cdot 7^u, z)]. \end{aligned} \tag{28}$$

$\text{Paso}_2(x,y,z)$ es como $\text{Paso}_1(x,y,z)$ pero con el caso 2 de la Definición 2.1.8.

$$\begin{aligned}\text{Paso}_3(x,y,z) \iff & \text{DI}(x) \wedge \text{DI}(y) \wedge \text{TM}(z) \wedge \\ & \bigvee_{F=0}^{x}\bigvee_{r=0}^{x}\bigvee_{s=0}^{x}\bigvee_{t=0}^{x}[(x = F\hat{}2^r\hat{}2^s) \\ & \wedge (y = F\hat{}2^s\hat{}2^t\hat{}2^7) \wedge \text{EI}(r) \wedge \text{EI}(t) \wedge \text{Al}(s) \\ & \wedge \text{Term}(2^r \cdot 3^s \cdot 5^3 \cdot 7^t, z)].\end{aligned} \tag{29}$$

$\text{Paso}_3(x,y,z)$ es como $\text{Paso}_1(x,y,z)$ pero con el caso 3 de la Definición 2.1.8.

$$\begin{aligned}\text{Paso}_4(x,y,z) \iff & \text{DI}(x) \wedge \text{DI}(y) \wedge \text{TM}(z) \wedge \\ & \bigvee_{F=0}^{x}\bigvee_{G=0}^{x}\bigvee_{r=0}^{x}\bigvee_{s=0}^{x}\bigvee_{t=0}^{x}\bigvee_{u=0}^{x}[(x = F\hat{}2^r\hat{}2^s\hat{}2^t\hat{}G) \\ & \wedge (y = F\hat{}2^u\hat{}2^r\hat{}2^t\hat{}G) \wedge \text{EI}(s) \wedge \text{EI}(u) \wedge \text{Al}(r) \wedge \text{Al}(t) \\ & \wedge \text{Term}(2^s \cdot 3^t \cdot 5^5 \cdot 7^u, z)].\end{aligned} \tag{30}$$

$\text{Paso}_4(x,y,z)$ es como $\text{Paso}_1(x,y,z)$ pero con el caso 4 de la Definición 2.1.8.

$$\begin{aligned}\text{Paso}_5(x,y,z) \iff & \text{DI}(x) \wedge \text{DI}(y) \wedge \text{TM}(z) \wedge \\ & \bigvee_{G=0}^{x}\bigvee_{r=0}^{x}\bigvee_{s=0}^{x}\bigvee_{t=0}^{x}[(x = 2^r\hat{}2^s\hat{}G) \\ & \wedge (y = 2^t\hat{}2^7\hat{}2^s\hat{}G) \wedge \text{EI}(r) \wedge \text{EI}(t) \wedge \text{Al}(s) \\ & \wedge \text{Term}(2^r \cdot 3^s \cdot 5^5 \cdot 7^t, z)].\end{aligned} \tag{31}$$

$\text{Paso}_5(x,y,z)$ es como $\text{Paso}_1(x,y,z)$ pero con el caso 5 de la Definición 2.1.8.

$$\begin{aligned}\text{Paso}(x,y,z) \iff & \text{Paso}_1(x,y,z) \vee \text{Paso}_2(x,y,z) \\ & \vee \text{Paso}_3(x,y,z) \vee \text{Paso}_4(x,y,z) \\ & \vee \text{Paso}_5(x,y,z).\end{aligned} \tag{32}$$

$\text{Paso}(x,y,z)$ es cierto si y sólo si x y y son números de Gödel de descripciones instantáneas, z es el número de Gödel de una Máquina de Turing Z y además

$\mathrm{Exp}(x) \to \mathrm{Exp}(y)\ (Z)$.

$$\begin{aligned}\mathrm{Fin}(x,z) \iff & \mathrm{DI}(x) \wedge \mathrm{TM}(z) \\ & \wedge \bigvee_{F=0}^{x} \bigvee_{G=0}^{x} \bigvee_{r=0}^{x} \bigvee_{s=0}^{x} \Big\{(x = F\hat{}2^r\hat{}2^s\hat{}G) \wedge \mathrm{EI}(r) \wedge \mathrm{Al}(s) \\ & \wedge \bigwedge_{n=1}^{\mathfrak{L}(z)} [(1\,\mathrm{Gl}\,(n\,GL\,z) \neq r) \vee (2\,\mathrm{Gl}\,(n\,\mathrm{Gl}\,z) \neq s)]\Big\}.\end{aligned} \tag{33}$$

$\mathrm{Fin}(x,z)$ es cierta si y solo si z es el número de Gödel de una Máquina de Turing Z y x es el número de Gödel de una descripción instantánea que es final con respecto a Z.

$$\begin{aligned}H_A(x,y,z) \iff & \mathrm{DI}(x) \wedge \mathrm{DI}(y) \wedge \mathrm{TM}(z) \\ & \wedge \bigvee_{F=0}^{x} \bigvee_{G=0}^{x} \bigvee_{r=0}^{x} \bigvee_{s=0}^{x} \bigvee_{t=0}^{x} \bigvee_{u=0}^{x} \{(x = F\hat{}2^r\hat{}2^s\hat{}G) \\ & \wedge \mathrm{EI}(r) \wedge \mathrm{EI}(t) \wedge \mathrm{EI}(u) \wedge \mathrm{Al}(s) \wedge \mathrm{Term}(2^r \cdot 3^s \cdot 5^t \cdot 7^u, z) \\ & \wedge [(C_A(\mathrm{Corn}(x)) = 0 \wedge y = F\hat{}2^t\hat{}2^s\hat{}G) \\ & \wedge (C_A(\mathrm{Corn}(x)) = 1 \wedge y = F\hat{}2^u\hat{}2^s\hat{}G)]\}.\end{aligned} \tag{34}$$

$H_A(x,y,z)$ es A-primitiva recursiva. $H_A(x,y,z)$ es cierta si y sólo si z es el número de Gödel de una Máquina de Turing Z, x y y son los números de Gödel de descripciones instantáneas y además $\mathrm{Exp}(x) \underset{A}{\to} \mathrm{Exp}(y)\ (Z)$.

$$\begin{aligned}\mathrm{Part}_A(y,z) \iff & \mathrm{TM}(z) \wedge \mathrm{GN}(y) \\ & \bigwedge_{n=1}^{\mathfrak{L}(y) \dot{-} 1} [\mathrm{Paso}(n\,\mathrm{Gl}\,y, (n+1)\,\mathrm{Gl}\,y, z) \\ & \vee H_A(n\,\mathrm{Gl}\,y, (n+1)\,\mathrm{Gl}\,y, z)].\end{aligned} \tag{35}$$

$\mathrm{Part}_A(y,z)$ es A recursiva. $\mathrm{Part}_A(y,z)$ es cierta si y sólo si z es el número de Gödel de una Máquina de Turing Z y y es el número de Gödel de una sucesión de expresiones instantáneas $\{e_1, e_2, \ldots e_n\}$ tales que $e_n \underset{A}{\to} e_{n+1}(Z)$ o $e_n \to e_{n+1}\ (Z)$.

$$\mathrm{Comp}_A(y,z) \iff \mathrm{Part}_A(y,z) \wedge \mathrm{Fin}(\mathfrak{L}(y)\,\mathrm{Gl}\,y, z). \tag{36}$$

$\mathrm{Comp}_A(y,z)$ es A recursiva. $\mathrm{Comp}_A(y,z)$ es cierta si y sólo si z es el número de Gödel de una Máquina de Turing Z y y es el número de Gödel de un A-cómputo de Z.

$$\mathrm{Comp}_A^k(y,z) \iff \mathrm{Comp}_A(y,z) \wedge (\mathfrak{L}(y) < k). \tag{37}$$

$\mathrm{Comp}_A^k(y,z)$ es A recursiva. $\mathrm{Comp}_A^k(y,z)$ es cierta si y sólo si z es el número de Gödel de una Máquina de Turing Z, y es el número de Gödel de un A-cómputo de Z y el cómputo se llevó a cabo en menos de k pasos.

Con esto podemos demostrar el resultado:

Teorema III.1. *$T_n^A(z, x_1, \ldots, x_n, y)$ es A recursivo.*

Demostración. Por la Definición 2.1.19 tenemos que:

$$T_n^A(z, x_1, \ldots, x_n, y) \iff \mathrm{Comp}_A(y,z) \wedge (1\,\mathrm{Gl}\,y = \mathrm{Init}_n(x_1, \ldots, x_n)). \qquad \square$$

Bibliografía

Davis, M. *Computability & unsolvability*. Dover, New York (1982)

Hernández, F. *Teoría de conjuntos*. Sociedad Matemática Mexicana, México (1998)

Kechris, A.S. *Classical Descriptive Set Theory*. Springer - Verlag, New York (1995)

Kjos-Hanssen, B. Permutations Of The Integers Induce Only The Trivial Automorphism Of The Turing Degrees. *The Bulletin of Symbolic Logic* **24**(2):165–174 (2018)

Kleene, S.C. y Post, E.L. The Upper Semi-Lattice of Degrees of Recursive Unsolvability. *Annals of Mathematics* **59**(3):379–407 (1954)

Slaman, T.A. *Global Properties Of The Turing Degrees And The Turing Jump*, págs. 83–101 (2005). https://www.worldscientific.com/doi/pdf/10.1142/9789812794055_0002

Slaman, T.A. y Woodin, W.H. Definability in the Turing degrees. *Illinois Journal of Mathematics* **30**:320–334 (1986)

Soare, R.I. *Turing computability : theory and applications*. Springer, Berlin (2016)

Turing, A.M. On Computable Numbers, with an Application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society* **s2-42**(1):230–265 (1937). https://londmathsoc.onlinelibrary.wiley.com/doi/pdf/10.1112/plms/s2-42.1.230