# 1. Otros Problemas Indecidibles

Hasta ahora hemos visto varios problemas indecidibles y no semidecidibles, pero todos eran problemas sobre Máquinas de Turing. La pregunta que nos hacemos es: ¿hay otros problemas no decidibles además de este tipo? la respuesta es afirmativa. Hay numerosos problemas indecidibles fuera del ámbito de los problemas sobre MTs. El primero que vamos a ver es el problema de las correspondencias de Post. Este se enuncia de la siguiente forma:

## 1.1. El Problema de las Correspondencias de Post

**El Problema de las Correspondencias de Post (PCP)**

Tenemos un alfabeto de referencia $A$, y dos listas con la misma longitud $B_1 = w_1, \ldots, w_k$, $B_2 = u_1, \ldots, u_k$ de palabras sobre $A$. El problema es determinar si existe una secuencia no vacía de enteros $i_1, \ldots, i_m$ tales que $w_{i_1} \ldots w_{i_m} = u_{i_1} \ldots u_{i_m}$.

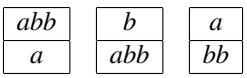
Esta es la definición formal del problema, pero se puede entender mejor si pensamos que cada pareja $(w_i, u_i)$ un bloque de construcción:


Entonces los datos del problema son un conjunto de bloques de construcción:


Lo que se pregunta es si podemos elegir una secuencia (finita) de bloques en la que se pueden repetir tantos bloques de cada tipo como queramos, de tal forma que poniéndolos todos juntos se lea lo mismo en la parte superior de los bloques que en la parte inferior. Está claro que es un problema de decisión con respuesta SI/NO y ya no es un problema de MTs.

**Ejemplo**

Si partimos de un caso con los bloques




La respuesta es afirmativa: si elegimos la secuencia de bloques: $1,3,1,1,3,2,2$, obtenemos

| | | | | | | |
|---|---|---|---|---|---|---|
| *abb* | *a* | *abb* | *abb* | *a* | *b* | *b* |
| *a* | *bb* | *a* | *a* | *bb* | *abb* | *abb* |

$\rightarrow$ *abbaabbabbabb*
*abbaabbabbabb*

Se puede leer lo mismo,*abbaabbabbabb*, en la parte superior que en la inferior.

Si visitáis la página web: `https://people.ksp.sk/~kuko/pcp/` podéis practicar con numerosos ejemplos de este puzle.

## 1.2. El Problema de las Correspondencias de Post Modificado

**PCP Modificado**

Tenemos un alfabeto de referencia $A$, y dos listas con la misma longitud $B_1 = w_1, \ldots, w_k$, $B_2 = u_1, \ldots, u_k$ de palabras sobre $A$ tales que $u_i, w_j \neq \varepsilon$ y un entero $i$. El problema es determinar si existe una secuencia no vacía de enteros $i_1, \ldots, i_m$ tales que $i_1 = i$ y $w_{i_1} \ldots w_{i_m} = u_{i_1} \ldots u_{i_m}$.

La única diferencia es que ahora nos dicen el bloque por el que necesariamente hay que comenzar y que las palabras son no vacías.

Vamos a suponer siempre que el bloque por el que hay que empezar es el primero de la lista de los bloques.

**Ejemplo**

Supongamos el ejemplo:

| | | |
|---|---|---|
| *a* | *ab* | *bba* |
| *baa* | *aa* | *bb* |

Como problema de las correspondencias de Post, tiene solución: 3,2,3,1 (respuesta afirmativa)
Efectivamente sale:

| | | | |
|---|---|---|---|
| *bba* | *ab* | *bba* | *a* |
| *bb* | *aa* | *bb* | *baa* |

Con lo que se lee *bbaabbbaa* en la parte superior e inferior.
Sin embargo, como PCP modificado no tiene solución (respuesta negativa) ya que en ese caso, una solución tiene que empezar por el primer bloque, y así la palabra superior empieza por *baa* y la inferior por *bb* y ya, pongamos lo que pongamos después, las palabras resultantes no pueden ser iguales.

Pensemos ahora en la semidecibilidad de estos problemas. ¿Podemos realizar un algoritmo que resuelva los casos positivos, aunque en algunos o todos los negativos cicle y no termine. La respuesta es claramente positiva y es más sencillo realizar un algoritmo no determinista. El algoritmo se da para el PCP pero sirve igual para el problema de las secuencias de Post modificado (con la condición de que la secuencia elegida empieza por 1).

**Algoritmo no determinista para el problema de las correspondencias de Post**

Se supone que la entrada al problema contiene $k$ bloques

$$\boxed{\begin{array}{|c|}\hline w_1 \\ \hline u_1 \\ \hline \end{array}} \quad \cdots \quad \boxed{\begin{array}{|c|}\hline w_k \\ \hline u_k \\ \hline \end{array}}$$

El algoritmo es:

Secuencia = []
Fin = Falso
Mientras No Fin
    Elige $i$ entre 1 y $k$
    Añade $i$ a Secuencia
    Elige Fin = Verdadero o Fin = False
Comprueba que eligiendo los bloques en el orden Secuencia es una solución del PCP

El algoritmo resuelve el problema, ya que si la respuesta es afirmativa puede responder de forma afirmativa y si es negativa NUNCA responderá de forma afirmativa.

Un algoritmo determinista procedería buscando si existe una solución con las secuencias de longitud 1,2,3,4,... a ver si alguna es una solucion del problema. Si existe una solución, seguro que la encontrará, pero si no existe nunca terminará. Es posible que podamos adivinar algunas condiciones que nos permitan parar la búsqueda en el caso negativo pudiendo dar una respuesta correcta: por ejemplo si para una longitud, en cada una de las secuencias de esa longitud, resulta que ni la parte inferior es una subcadena de la superior, ni la superior una subcadena de la inferior, podemos terminar dando una respuesta negativa. Pero esa condición no sirve en todos los casos. La pregunta es, ¿podemos diseñar un conjunto de condiciones que permita dar la respuesta a todos los casos negativos en tiempo finito? o en otras palabras ¿es el problema decidible? La respuesta es que no (excepto si el alfabeto $A$ tiene un sólo símbolo), ni para el problema de las correspondencias de Post, ni para el modificado. Eso viene demostrado por el siguiente teorema. Su demostración se basa en mostrar como el cálculo de una MT se puede simular mediante un PCP modificado.

**Teorema**

El problema de las correspondencias de Post modificado es indecidible si $A$ tiene al menos 2 símbolos.

Si $A$ tiene dos símbolos, podemos codificar palabras de cualquier alfabeto, de manera que sea un homomorfismo.

Por ejemplo, si el alfabeto es $A = \{a,b,c\}$, podemos codificar $a = 00$, $b = 01$, $c = 11$. Así una ficha se codificaría símbolo a símbolo:

$$\boxed{\begin{array}{|c|}\hline abb \\ \hline ca \\ \hline \end{array}} \rightarrow \boxed{\begin{array}{|c|}\hline 000101 \\ \hline 1100 \\ \hline \end{array}}$$

Codificando así todas las fichas, el problema original tiene solución si y solo si el problema codificado con dos símbolos tiene solución.

Vamos a reducir el problema universal a este problema con un alfabeto arbitrario (después lo podemos codificar con un alfabeto de al menos dos símbolos).

Para llevar a cabo esta reducción tenemos que dar una transformación de un ejemplo del problema universidad $M$,$w$ en un PCP modificado tal que si $M$ acepta $w$ el problema tiene solución y si no la acepta no tiene solución. Dicha transformación tiene que poder ser implementada mediante un algoritmo:

Vamos a suponer una MT $M$ y una palabra $w$. La pregunta es si $M$ acepta $w$. Vamos a construir un problema de correspondencias de Post modificado con la misma solución.

El alfabeto que vamos a considerar para el alfabeto del problema de las correspondencias es el alfabeto de la MT, más el conjunto de estados, más un separador que no esté en los conjuntos anteriores $*$. Dados $M$ y $w$ vamos a construir los siguiente bloques:

- Si $w = a_1 \dots a_n$

  Introducimos el bloque $\boxed{\begin{array}{c} * \\ \hline *q_0 a_1 \dots a_n * \end{array}}$, que será el bloque de comienzo.
- Por cada transición $\delta(q,a) = (q',b,D)$, introducimos el bloque $\boxed{\begin{array}{c} qa \\ \hline bq' \end{array}}$
- Por cada transición $\delta(q,a) = (q',b,I)$, introducimos el bloque $\boxed{\begin{array}{c} cqa \\ \hline q'cb \end{array}}$ para cada $c$ del alfabeto de la MT
- Para cada símbolo $a$ de la MT introducir $\boxed{\begin{array}{c} a \\ \hline a \end{array}}$
- Los bloques $\boxed{\begin{array}{c} * \\ \hline * \end{array}}$, $\boxed{\begin{array}{c} * \\ \hline \#* \end{array}}$, $\boxed{\begin{array}{c} * \\ \hline *\# \end{array}}$
- Para cada $q \in F$ (estado final), añade los bloques $\boxed{\begin{array}{c} aq \\ \hline q \end{array}}$, $\boxed{\begin{array}{c} qa \\ \hline q \end{array}}$,

  y el bloque $\boxed{\begin{array}{c} q** \\ \hline * \end{array}}$.

La idea de la reducción es representar en los bloques de una posible solución el cálculo que realiza la MT para la palabra de entrada $w$ escribiendo el historial de sus configuraciones.

Cada configuración $(q,u,v)$ se representa por la cadena $uqv$ y las distintas configuraciones se separan por $*$.

En la parte de abajo se lleva una configuración de ventaja respecto a la parte de arriba. Eso ocurre para cada cálculo, pero cuando se llega a un estado de aceptación, entonces la parte de abajo empieza a disminuir símbolo a símbolo paso a paso y copiándose arriba, hasta que quede abajo $\dots * q*$ y arriba $\dots * qa*$ ó $\dots * aq*$ donde $q$ es el estado final. Entonces podemos completar con el último bloque $\boxed{\begin{array}{c} q** \\ \hline * \end{array}}$ y se completa la solución.

Así se obtiene una solución si y solo si el cálculo de la MT llega a un estado de aceptación.

Como el Problema Universal se reduce al PCP modificado y el problema universal no es decidible, entonces el PCP no es decidible.

**Ejemplo**

Vamos a ilustrar el razonamiento de la demostración con un ejemplo.
Supongamos la MT $M = (\{q_0, q_1, q_2, q_3, q_4\}, \{0,1\}, \{0,1,X,Y,\#\}, \delta, q_0, \#, \{q_4\})$ donde las transiciones son las siguientes:

$$\begin{array}{ll} \delta(q_0,0) = (q_1,X,D) & \delta(q_0,Y) = (q_3,Y,D) \\ \delta(q_1,0) = (q_1,0,D) & \delta(q_1,1) = (q_2,Y,I) \\ \delta(q_1,Y) = (q_1,Y,D) & \delta(q_2,0) = (q_2,0,I) \\ \delta(q_2,X) = (q_0,X,D) & \delta(q_2,Y) = (q_2,Y,I) \\ \delta(q_3,Y) = (q_3,Y,D) & \delta(q_3,\#) = (q_4,\#,D) \end{array}$$

(primer ejemplo de MT de este tema).
y la palabra de entrada 000111
Una solución del PCP modificado, tiene que empezar con la ficha:

| $*$ |
|---|
| $*q_0000111*$ |

Cómo $\delta(q_0,0) = (q_1,X,D)$, tenemos la ficha $\begin{array}{|c|}\hline q_00 \\ \hline Xq_1 \\ \hline \end{array}$ la única forma de proceder en una posible solución es:

| $*$ | $q_00$ | 0 | 0 | 1 | 1 | 1 | $*$ |
|---|---|---|---|---|---|---|---|
| $*q_0000111*$ | $Xq_1$ | 0 | 0 | 1 | 1 | 1 | $*$ |

Si nos fijamos, hemos copiado en la parte superior la configuración que teníamos en la parte inferior y en la inferior se ha añadido la configuración del siguiente paso en el cálculo de la MT. Se puede comprobar que esto es siempre así. De tal forma que al final del cálculo de la MT llegaremos a una situación en la que tendremos lo siguiente en las partes superior e inferior de una solución (representamos como una única ficha la solución parcial):

| $u*$ |
|---|
| $u*XXXYYY\#q_4*$ |

donde $u$ es la misma palabra en las dos partes. Podemos ver que aún no es una solucion, pero como $q_4$ es estado final, podemos continuar con los bloques

| $u*$ | $X$ | $X$ | $X$ | $Y$ | $Y$ | $Y$ | $\#q_4$ | $*$ |
|---|---|---|---|---|---|---|---|---|
| $u*XXXYYY\#q_4*$ | $X$ | $X$ | $X$ | $Y$ | $Y$ | $Y$ | $q_4$ | $*$ |

Si nos damos cuenta, el estado final permite copiar lo que hay de más en la parte inferior en la parte superior, y en la parte inferior se copia todo menos un símbolo que esté al lado del estado final. Si vamos repitiendo este proceso, cada vez podemos disminuir lo que hay en la parte inferior en un símbolo de la cinta, hasta que lleguemos a una situación en la que tenemos:

| $w*$ |
|---|
| $w*q_4*$ |

Ahora completamos la solución con el bloque $\begin{array}{|c|}\hline q_4** \\ \hline * \\ \hline \end{array}$, que existe por ser $q_4$ final, obteniendo una solución:

| $w*$ | $q_4**$ |
|---|---|
| $w*q_4*$ | $*$ |

Veamos ahora que el PCP genérico es también indecidible.

**Teorema**

El Problema de las correspondencias de Post es indecidible

Esto se demuestra reduciendo el problema de las correspondencias de Post modificado al problema de las correspondencias de Post.

Supongamos que nos dan un caso del PCP modificado, es decir un conjunto de bloques $\boxed{\frac{u_1}{v_1}}, \boxed{\frac{u_2}{v_2}}, \cdots, \boxed{\frac{u_n}{v_n}}$ y que el primero de una solución es necesariamente es el bloque 1 y donde en todas las fichas $u_i, v_i \neq \varepsilon$.

Se construye el siguiente problema de Post con dos símbolos nuevos: ♦, ■ y con los bloques:

$$\boxed{\frac{♦u_1}{♦v_1♦}}, \boxed{\frac{♦u_1}{v_1♦}}, \boxed{\frac{♦u_2}{v_2♦}}, \cdots, \boxed{\frac{♦u_n}{v_n♦}}, \boxed{\frac{♦■}{■}}$$

Donde, si $u = a_1a_2\ldots a_n$, se entiende que

$$♦u = ♦a_1♦a_2\ldots♦a_n, \quad u♦ = a_1♦a_2\ldots♦a_n♦, \quad ♦u♦ = ♦a_1♦a_2\ldots♦a_n♦$$

Como se puede ver, las soluciones a ambos problemas son equivalentes. Si el modificado tiene solución, entonces los bloques asociados con los dos símbolos nuevos son una solución del PCP finalizados por el bloque $\boxed{\frac{♦■}{■}}$.

Por otra parte en cualquier solución del PCP, hay que empezar por la ficha $\boxed{\frac{♦u_1}{♦v_1♦}}$ ya que es la única que comienza en ambas partes por el mismo símbolo ♦ y además terminar por $\boxed{\frac{♦■}{■}}$. Si a todos los bloques distintos del último les quitamos los dos símbolos especiales obtenemos una solución del PCP modificado que empieza por la ficha 1,

## 2. Problemas sobre Gramáticas

El problema de las correspondencias de Post es base para demostrar que otros problemas no son decidibles, en particular los siguientes problemas sobre gramáticas. Una lista de estos problemas es la siguiente,

**Problemas sobre Gramáticas**

Supongamos que $G, G_1$ y $G_2$ son gramáticas independientes del contexto dadas y $R$ es un lenguaje regular (un autómata finito determinista), entonces los siguientes problemas son indecidibles

- Saber si $L(G_1) \cap L(G_2) = \emptyset$.
- Determinar si $L(G) = T^*$, donde $T$ es el conjunto de símbolos terminales.
- Comprobar si $L(G_1) = L(G_2)$.
- Determinar si $L(G_1) \subseteq L(G_2)$.
- Determinar si $L(G_1) = R$.
- Comprobar si $L(G)$ es regular.
- Determinar si $G$ es ambigua.
- Conocer si $L(G)$ es inherentemente ambiguo.
- Comprobar si $L(G)$ es determinista.

No vamos a demostrar todos los casos, sólo uno de ellos.

**Teorema**

Saber si una gramática independiente del contexto es ambigua es indecidible.

Para comprobar que el saber si una gramática independiente del contexto es indecidible, vamos a reducir el el problema de las correspondencias de Post al problema de saber si una gramática es ambigua.

Supongamos un caso del PCP sobre el alfabeto $A$ y los bloques $\boxed{\frac{u_1}{v_1}}, \cdots, \boxed{\frac{u_k}{v_k}}$.

Sea $B = A \cup \{b_1, \ldots, b_k\}$ donde $b_i \notin A$ (es decir $k$ símbolos nuevos) y construimos la siguiente gramática:

$$
\begin{array}{l}
S \to C|D \\
C \to u_i C b_i | u_i b_i, \quad i = 1, \ldots, k \\
D \to v_i D b_i | v_i b_i, \quad i = 1, \ldots, k
\end{array}
$$

La equivalencia de ambos problemas (PCP original y ambigüedad de la gramática construida) se basa en lo siguiente:

- Las palabras generadas por la gramática son las generadas a partir de $C$ más las generadas a partir de $D$
- Las palabras generadas a partir de $C$ son de la forma $u_{i_1} \ldots u_{i_n} b_{i_l} \ldots b_{i_n}$, donde $i_j \in \{1, \ldots, k\}$. Sólo hay una forma de generar una de estas palabras a partir de $C$.
- Las palabras generadas a partir de $D$ son de la forma $v_{i_1} \ldots v_{i_n} b_{i_l} \ldots b_{i_n}$, donde $i_j \in \{1, \ldots, k\}$. Sólo hay una forma de generar una de estas palabras a partir de $D$.

- La gramática es ambigua cuando una misma palabra $u$ es generada a partir de $C$ y a partir de $D$, es decir cuando $u = u_{i_1} \dots u_{i_n} b_{i_l} \dots b_{i_n} = v_{i'_1} \dots v_{i'_{n'}} b_{i'_l} \dots b_{i'_{n'}}$. Esto solo ocurre si $n' = n$ e $i_j = i'_j, \forall j$. Es decir cuando existen $(i_1, \dots, i_n)$ tal que $u_{i_1} \dots u_{i_n} = v_{i_1} \dots v_{i_n}$, es decir cuando el PCP tiene solución.

**Ejemplo**

Si partimos del ejemplo de los bloques:

| $abb$ | $b$ | $a$ |
|---|---|---|
| $a$ | $abb$ | $bb$ |

La gramática asociada tiene como símbolos terminales $\{a, b, b_1, b_2, b_3\}$. Se añaden tres símbolos nuevos $b_1, b_2, b_3$, uno por cada bloque.
Según hemos visto, la gramática asociada es:
$S \to C|D$
$C \to abbCb_1|abbb_1|bCb_2|bb_2|aCb_3|ab_3$
$D \to aDb_1|ab_1|abbDb_2|abbb_2|bbDb_3|bbb_3$
Como vemos $C$ deriva palabras que en la primera parte contienen partes superiores de las fichas y después una lista de $b_i$ que representan las fichas usadas (en orden inverso). $D$ deriva palabras de forma análoga, pero ahora con las partes inferiores de las fichas.
En PCP tiene una respuesta afirmativa: eligiendo la secuencia de bloques: $1, 3, 1, 1, 3, 2, 2$

| $abb$ | $a$ | $abb$ | $abb$ | $a$ | $b$ | $b$ |
|---|---|---|---|---|---|---|
| $a$ | $bb$ | $a$ | $a$ | $bb$ | $abb$ | $abb$ |

$\to$ $abbaabbabbabb$
$abbaabbabbabb$

Esto corresponde a una derivación doble de la palabra: $abbaabbabbabbb_2b_2b_3b_1b_1b_3b_1$ (la parte común más una sucesión de $b_i$ indicando las fichas usadas en orden inverso. En efecto, esta palabra se puede obtener de dos formas:

$$S \Rightarrow C \Rightarrow abbCb_1 \Rightarrow abbaCb_3b_1 \Rightarrow abbaabbCb_1b_3b_1 \Rightarrow abbaabbabbCb_1b_1b_3b_1 \Rightarrow$$
$$abbaabbabbaCb_3b_1b_1b_3b_1 \Rightarrow abbaabbabbabCb_2b_3b_1b_1b_3b_1 \Rightarrow abbaabbabbabbb_2b_2b_3b_1b_1b_3b_1$$

$$S \Rightarrow D \Rightarrow aDb_1 \Rightarrow abbDb_3b_1 \Rightarrow abbaDb_1b_3b_1 \Rightarrow abbaaDb_1b_1b_3b_1 \Rightarrow$$
$$abbaabbDb_3b_1b_1b_3b_1 \Rightarrow abbaabbabbDb_2b_3b_1b_1b_3b_1 \Rightarrow abbaabbabbabbb_2b_2b_3b_1b_1b_3b_1$$