Resolubilidad

Teoría de la Computación

Grado en Ingeniería Informática

Contenidos

Contenidos

1 El problema de la parada

Hemos visto que una función f es **Turing computable** si existe una MT que computa $f(w)$ para toda cadena w perteneciente al dominio de f.

1 El problema de la parada

Hemos visto que una función f es **Turing computable** si existe una MT que computa $f(w)$ para toda cadena w perteneciente al dominio de f.

Un ejemplo de función computable es $\mathcal{X}_L(w)$, la **función característica de un lenguaje recursivo** L, que vale 1 (sí) cuando $w \in L$ y 0 (no) en caso contrario.

1 El problema de la parada

Hemos visto que una función f es **Turing computable** si existe una MT que computa $f(w)$ para toda cadena w perteneciente al dominio de f.

Un ejemplo de función computable es $\mathcal{X}_L(w)$, la **función característica de un lenguaje recursivo** L, que vale 1 (sí) cuando $w \in L$ y 0 (no) en caso contrario.

Y hemos visto también que la **función característica de un lenguaje recursivamente enumerable** que no es recursivo **no es computable**, ya que existen cadenas para las cuales la MT que acepta dicho lenguaje no para.

1 El problema de la parada

Hemos visto que una función f es **Turing computable** si existe una MT que computa $f(w)$ para toda cadena w perteneciente al dominio de f.

Un ejemplo de función computable es $\mathcal{X}_L(w)$, la **función característica de un lenguaje recursivo** L, que vale 1 (sí) cuando $w \in L$ y 0 (no) en caso contrario.

Y hemos visto también que la **función característica de un lenguaje recursivamente enumerable** que no es recursivo **no es computable**, ya que existen cadenas para las cuales la MT que acepta dicho lenguaje no para.

Por tanto, vemos que existen ejemplos de funciones computables y no computables.

1 El problema de la parada

Hemos visto que una función f es **Turing computable** si existe una MT que computa $f(w)$ para toda cadena w perteneciente al dominio de f.

Un ejemplo de función computable es $\mathcal{X}_L(w)$, la **función característica de un lenguaje recursivo** L, que vale 1 (sí) cuando $w \in L$ y 0 (no) en caso contrario.

Y hemos visto también que la **función característica de un lenguaje recursivamente enumerable** que no es recursivo **no es computable**, ya que existen cadenas para las cuales la MT que acepta dicho lenguaje no para.

Por tanto, vemos que existen ejemplos de funciones computables y no computables.

Estudiaremos ahora ciertas cuestiones relacionadas con la computabilidad, en situaciones donde el resultado de la computación es 1 (sí) o 0 (no). Los problemas de este tipo se conocen como **problemas de decisión** y dicha clase de computabilidad se denomina **resolubilidad** o **decidibilidad**.

1 El problema de la parada

Problemas de decisión son, por ejemplo, los siguientes:

- Dada G una gramática independiente del contexto, ¿$L(G)$ es vacío?

  Obsérvese que hay un número infinito de casos para este problema: uno por cada posible gramática independiente del contexto.

1 El problema de la parada

Problemas de decisión son, por ejemplo, los siguientes:

- Dada G una gramática independiente del contexto, ¿$L(G)$ es vacío?

  Obsérvese que hay un número infinito de casos para este problema: uno por cada posible gramática independiente del contexto.

- Dada G una gramática sensible al contexto y una cadena w, ¿$w \in L(G)$?

  Nuevamente hay infinitos casos, cada uno de ellos determinado por una gramática y una cadena concretas.

1 El problema de la parada

Problemas de decisión son, por ejemplo, los siguientes:

- Dada G una gramática independiente del contexto, ¿$L(G)$ es vacío?

  Obsérvese que hay un número infinito de casos para este problema: uno por cada posible gramática independiente del contexto.

- Dada G una gramática sensible al contexto y una cadena w, ¿$w \in L(G)$?

  Nuevamente hay infinitos casos, cada uno de ellos determinado por una gramática y una cadena concretas.

Definición

Se dice que un problema de decisión es **resoluble** o **decidible** si existe un algoritmo capaz de responder sí o no a cada uno de los casos de dicho problema.

Si dicho algoritmo no existe, el problema es **irresoluble** o **no decidible**.

1 El problema de la parada

Problemas de decisión son, por ejemplo, los siguientes:

- Dada G una gramática independiente del contexto, ¿$L(G)$ es vacío?

  Obsérvese que hay un número infinito de casos para este problema: uno por cada posible gramática independiente del contexto.

- Dada G una gramática sensible al contexto y una cadena w, ¿$w \in L(G)$?

  Nuevamente hay infinitos casos, cada uno de ellos determinado por una gramática y una cadena concretas.

Definición

Se dice que un problema de decisión es **resoluble** o **decidible** si existe un algoritmo capaz de responder sí o no a cada uno de los casos de dicho problema.

Si dicho algoritmo no existe, el problema es **irresoluble** o **no decidible**.

Los problemas de los dos ejemplos anteriores son ambos resolubles, ya que existen algoritmos que siempre responden sí o no a todas y cada una de las posibles instancias de dichos problemas.

1 El problema de la parada

Uno de los problemas irresolubles más conocidos es el **problema de la parada** para máquinas de Turing, que se enuncia como sigue.

El Problema de la Parada

Sea M una máquina de Turing arbitraria con alfabeto de entrada Σ y sea w una cadena de Σ^*. ¿Parará M con w como cadena de entrada?

1 El problema de la parada

Uno de los problemas irresolubles más conocidos es el **problema de la parada** para máquinas de Turing, que se enuncia como sigue.

El Problema de la Parada

Sea M una máquina de Turing arbitraria con alfabeto de entrada Σ y sea w una cadena de Σ^*. ¿Parará M con w como cadena de entrada?

El siguiente resultado demuestra la irresolubilidad de este problema.

Teorema

El problema de la parada para máquinas de Turing es irresoluble.

1 El problema de la parada

Uno de los problemas irresolubles más conocidos es el **problema de la parada** para máquinas de Turing, que se enuncia como sigue.

El Problema de la Parada

Sea M una máquina de Turing arbitraria con alfabeto de entrada Σ y sea w una cadena de Σ^*. ¿Parará M con w como cadena de entrada?

El siguiente resultado demuestra la irresolubilidad de este problema.

Teorema

El problema de la parada para máquinas de Turing es irresoluble.

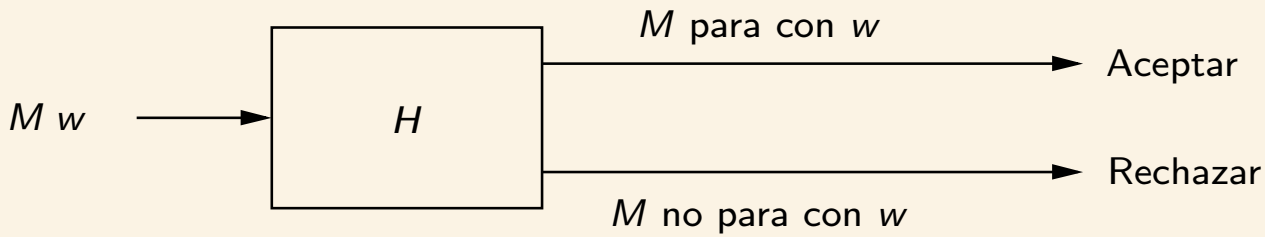
Demostración:

Supongamos que el problema es resoluble.

Entonces existe una MT H (*Halting*), similar a la Máquina de Turing Universal, que recibe como entrada la codificación de una MT M y de una cadena w, y que es capaz de determinar si M para al procesar w. .../...

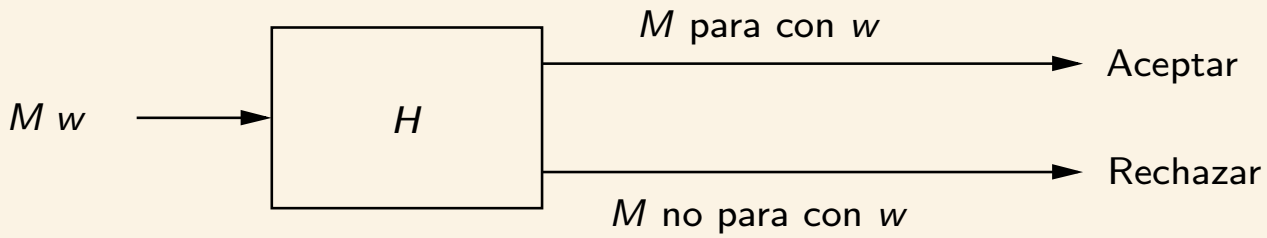
Teorema (continuación)

Es decir, la MT H tendría el siguiente comportamiento:

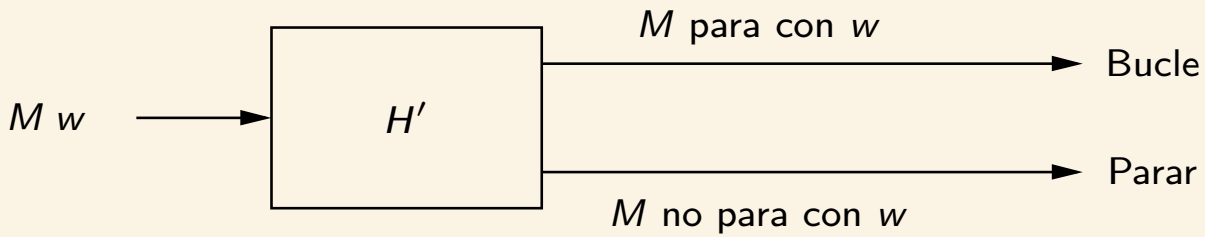


Teorema (continuación)

Es decir, la MT H tendría el siguiente comportamiento:



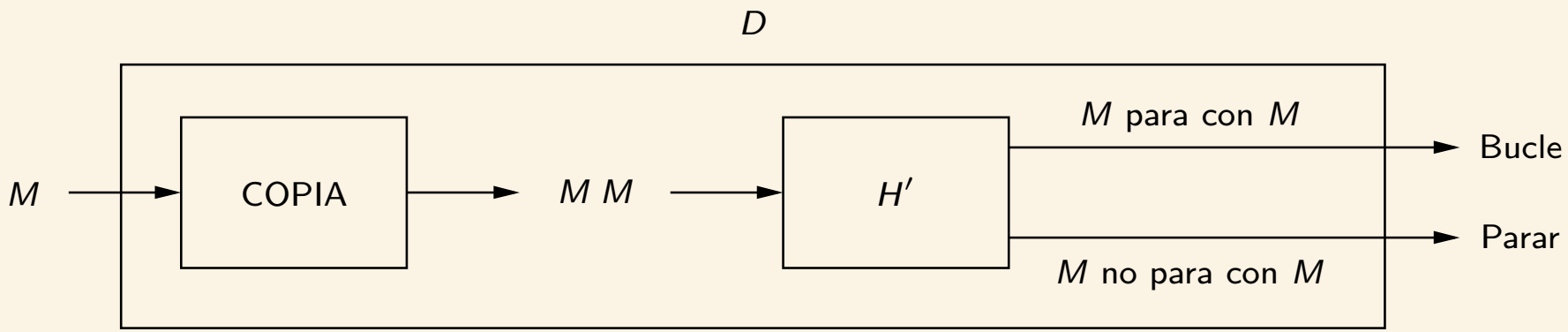
Entonces podemos modificar H para crear otra MT H' de la siguiente forma:



.../...

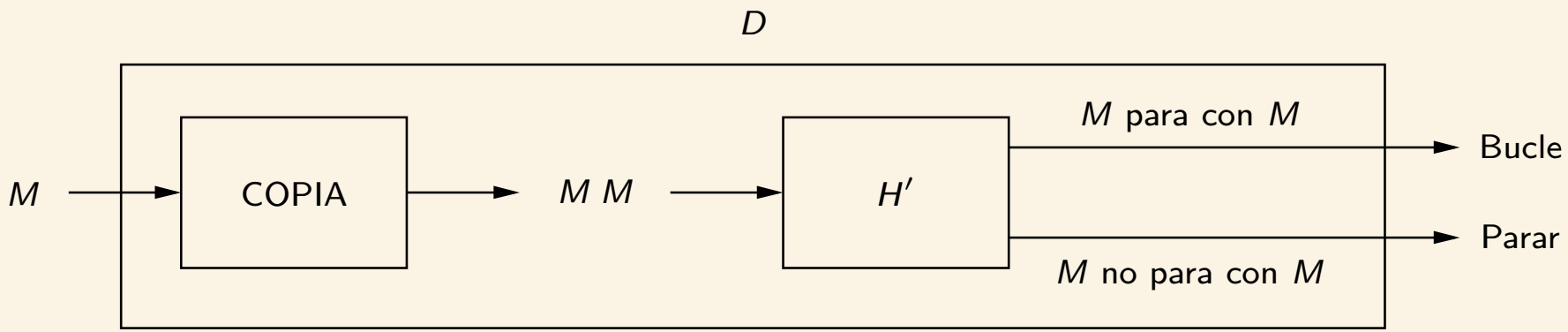
Teorema (continuación)

Hacemos entonces la siguiente composición:

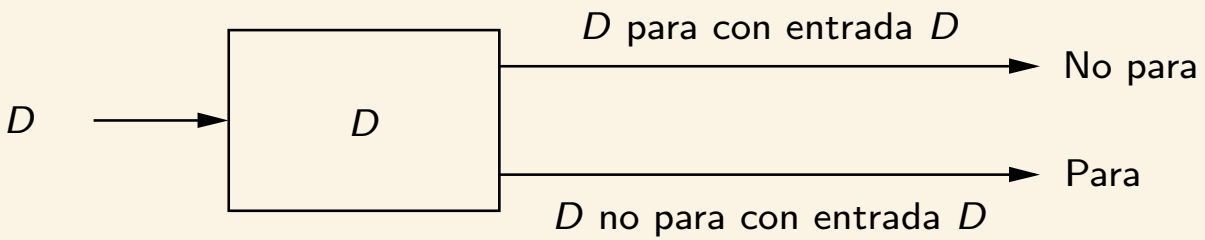


Teorema (continuación)

Hacemos entonces la siguiente composición:



Y llegamos a que:



Lo cual es una contradicción, que viene del hecho de haber supuesto que el problema de la parada era resoluble. □

1 El problema de la parada

La irresolubilidad del problema de la parada puede utilizarse para demostrar que otros problemas también son irresolubles.

1 El problema de la parada

La irresolubilidad del problema de la parada puede utilizarse para demostrar que otros problemas también son irresolubles.

Por ejemplo, el **problema de la cinta en blanco** (que consiste en decidir si una máquina de Turing parará o no cuando se le dé como entrada una cinta en blanco) es irresoluble.

1 El problema de la parada

La irresolubilidad del problema de la parada puede utilizarse para demostrar que otros problemas también son irresolubles.

Por ejemplo, el **problema de la cinta en blanco** (que consiste en decidir si una máquina de Turing parará o no cuando se le dé como entrada una cinta en blanco) es irresoluble.

Para demostrar que el problema de la cinta en blanco es irresoluble, demostraremos que, si fuera resoluble, entonces el problema de la parada también lo sería.

1 El problema de la parada

La irresolubilidad del problema de la parada puede utilizarse para demostrar que otros problemas también son irresolubles.

Por ejemplo, el **problema de la cinta en blanco** (que consiste en decidir si una máquina de Turing parará o no cuando se le dé como entrada una cinta en blanco) es irresoluble.

Para demostrar que el problema de la cinta en blanco es irresoluble, demostraremos que, si fuera resoluble, entonces el problema de la parada también lo sería.

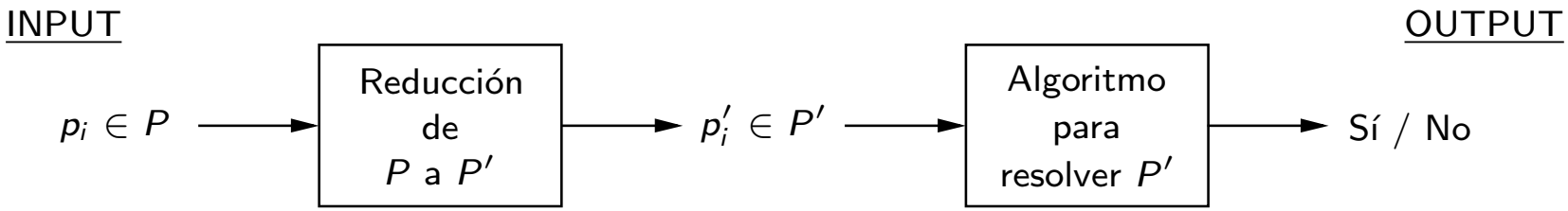
Esta técnica de resolución de problemas de decisión se conoce como **reducibilidad**.

Reducibilidad

Un problema de decisión P es **Turing reducible** a otro problema de decisión P'
si existe una MT que, para cualquier instancia de entrada $p_i \in P$,
produce como salida una nueva instancia asociada $p'_i \in P'$,
de tal forma que la respuesta a la instancia original p_i del problema P
se puede obtener a partir de la respuesta a la instancia p'_i del problema P'.

1 El problema de la parada

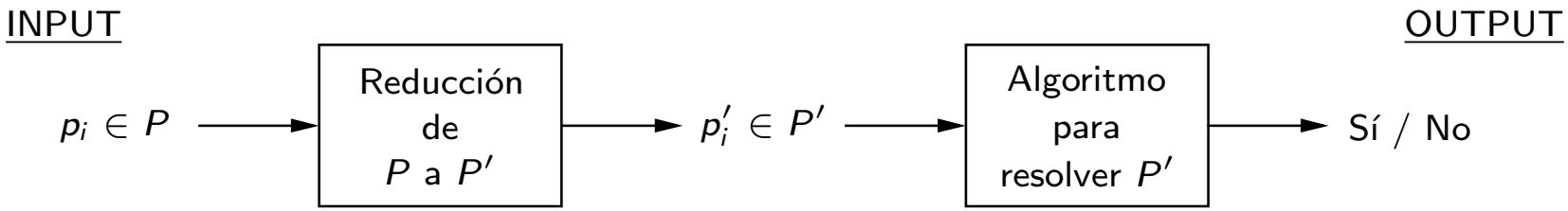
Por lo tanto, **si un problema de decisión P′ es decidible y P es reducible a P′, entonces P es también decidible**. La solución para P se puede obtener combinando el algoritmo de reducción con el algoritmo que resuelve P'.



La reducción es una técnica bastante común en la resolución de problemas de decisión. Cuando se afronta un nuevo problema, normalmente intentamos reducirlo a otro que ya ha sido previamente resuelto.

1 El problema de la parada

Por lo tanto, **si un problema de decisión P$'$ es decidible y P es reducible a P$'$, entonces P es también decidible**. La solución para P se puede obtener combinando el algoritmo de reducción con el algoritmo que resuelve P'.



La reducción es una técnica bastante común en la resolución de problemas de decisión. Cuando se afronta un nuevo problema, normalmente intentamos reducirlo a otro que ya ha sido previamente resuelto.

Pero la reducción tiene también implicaciones muy importantes relacionadas con la irresolubilidad. **Si P es no decidible y P es reducible a P$'$, entonces P$'$ es también no decidible**. Si P' fuera decidible, a través del algoritmo de reducción y del algoritmo que resuelve P', podríamos construir un nuevo algoritmo para resolver P.

1 El problema de la parada

Teorema

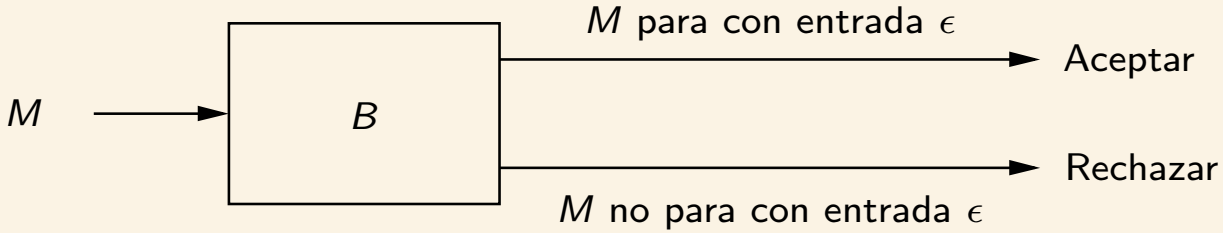
El problema de la cinta en blanco es irresoluble.

Teorema

El problema de la cinta en blanco es irresoluble.

Demostración:

Basta con ver que el problema de la parada es reducible al problema de la cinta en blanco. Para ello, supongamos que existe una MT B que resuelve el problema de la cinta en blanco.

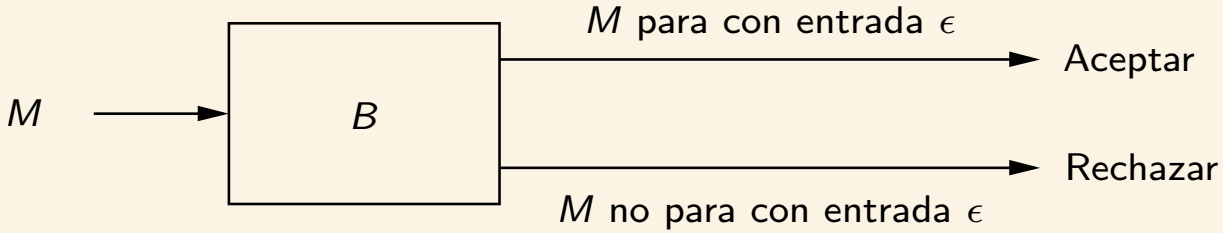


Teorema

El problema de la cinta en blanco es irresoluble.

Demostración:

Basta con ver que el problema de la parada es reducible al problema de la cinta en blanco. Para ello, supongamos que existe una MT B que resuelve el problema de la cinta en blanco.



Entonces construimos una nueva MT añadiendo un preprocesador N a B. La entrada de N será la codificación de una MT M y de una cadena w. La salida de N será la codificación de una MT M' que hará lo siguiente:

1. Escribir w sobre una cinta en blanco.
2. Posicionar la cabeza de L/E al principio de w y pasar al estado que antes era estado inicial de M.
3. Ejecutar M sobre w. .../...

Teorema (continuación)

M' se obtiene a partir de M añadiendo algunos estados (entre ellos, un nuevo estado inicial) y las transiciones (codificadas) que sean necesarias para escribir w, volver al principio y simular M.

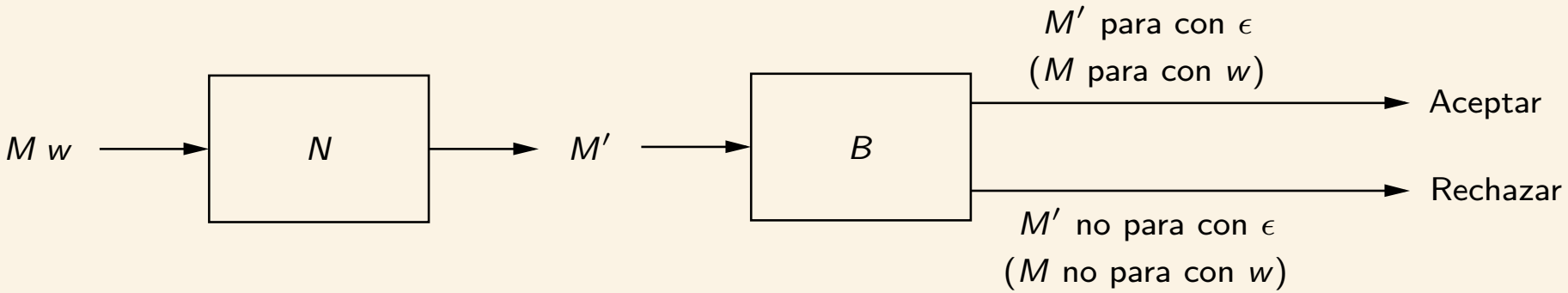
Lo importante es que M' ha sido construida de tal manera que M' para con $\epsilon \Leftrightarrow M$ para con w.

Teorema (continuación)

M' se obtiene a partir de M añadiendo algunos estados (entre ellos, un nuevo estado inicial) y las transiciones (codificadas) que sean necesarias para escribir w, volver al principio y simular M.

Lo importante es que M' ha sido construida de tal manera que M' para con $\epsilon \Leftrightarrow M$ para con w.

La ejecución secuencial de N y B produce entonces la siguiente composición:

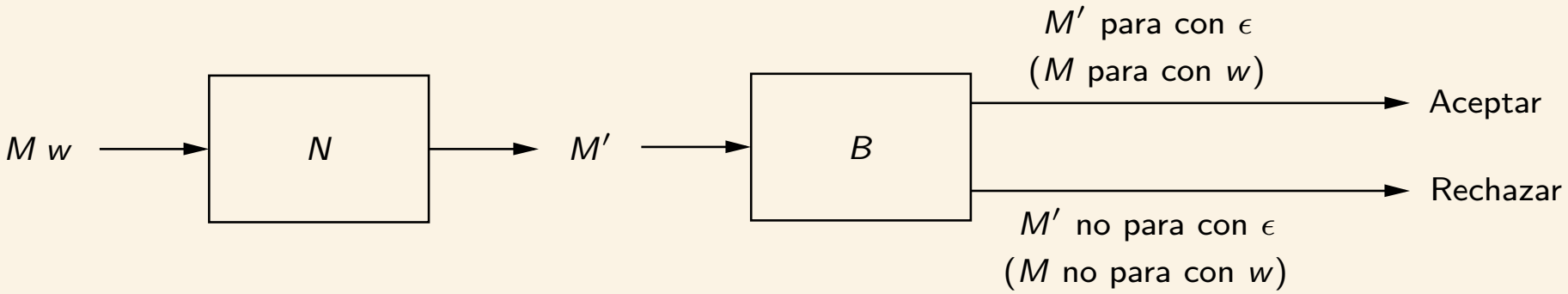


Teorema (continuación)

M' se obtiene a partir de M añadiendo algunos estados (entre ellos, un nuevo estado inicial) y las transiciones (codificadas) que sean necesarias para escribir w, volver al principio y simular M.

Lo importante es que M' ha sido construida de tal manera que M' para con $\epsilon \Leftrightarrow M$ para con w.

La ejecución secuencial de N y B produce entonces la siguiente composición:



El preprocesador N reduce el problema de la parada al problema de la cinta en blanco.
Por tanto, esta composición resolvería el problema de la parada, el cual es irresoluble.

La contradicción viene del hecho de haber supuesto que B existe.
Por tanto, el problema de la cinta en blanco es también irresoluble. □

1 El problema de la parada

Ejercicio

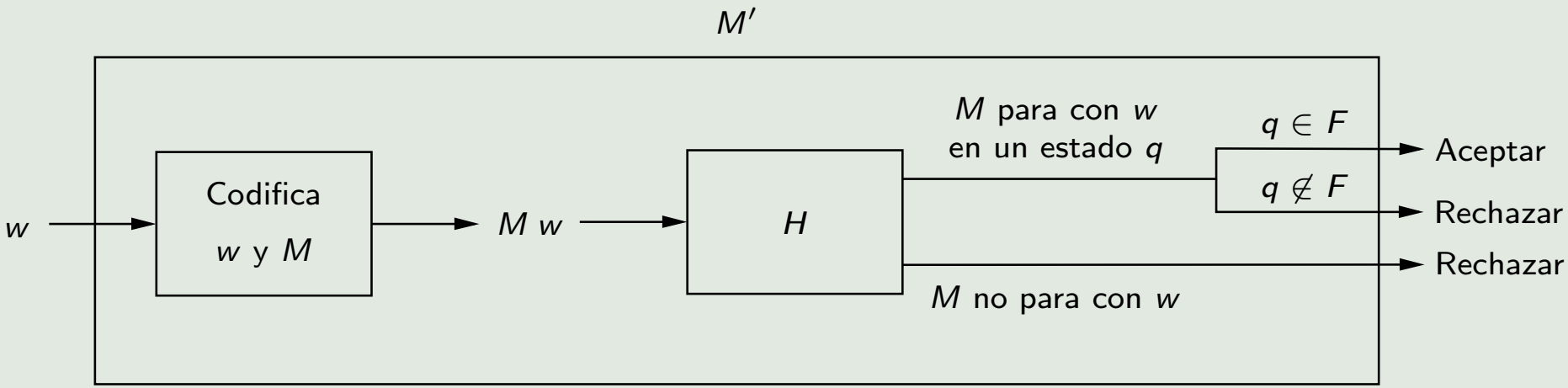
Demuestre que si el problema de la parada se pudiera resolver, entonces todo lenguaje recursivamente enumerable sería recursivo.

Ejercicio

Demuestre que si el problema de la parada se pudiera resolver, entonces todo lenguaje recursivamente enumerable sería recursivo.

Solución:

Sea L un lenguaje recursivamente enumerable, sea M una MT tal que $L(M) = L$ y sea H (*Halting*) la MT que supuestamente resuelve el problema de la parada. Entonces construimos una nueva MT M', a partir de M y H, como sigue:



La MT M' acepta el lenguaje L y se detiene ante cualquier cadena de entrada w. Por tanto, el lenguaje L sería recursivo.

Ejercicio

Demuestre que el problema de la entrada en un estado es irresoluble. Este problema se conoce también como SEP (*State Entry Problem*) y se enuncia como sigue:

> *Dada una MT arbitraria* $M = (Q, \Sigma, \Gamma, s, B, F, \delta)$*, una cadena* $w \in \Sigma^*$ *y un estado* $q \in Q$*, ¿entrará* M *en* q *al procesar* w*?*

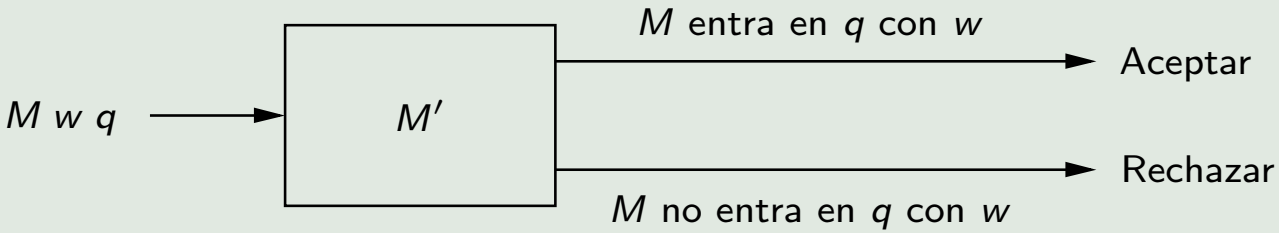
Ejercicio

Demuestre que el problema de la entrada en un estado es irresoluble. Este problema se conoce también como SEP (*State Entry Problem*) y se enuncia como sigue:

> *Dada una MT arbitraria* $M = (Q, \Sigma, \Gamma, s, B, F, \delta)$*, una cadena* $w \in \Sigma^*$ *y un estado* $q \in Q$*, ¿entrará* M *en* q *al procesar* w*?*

Solución:

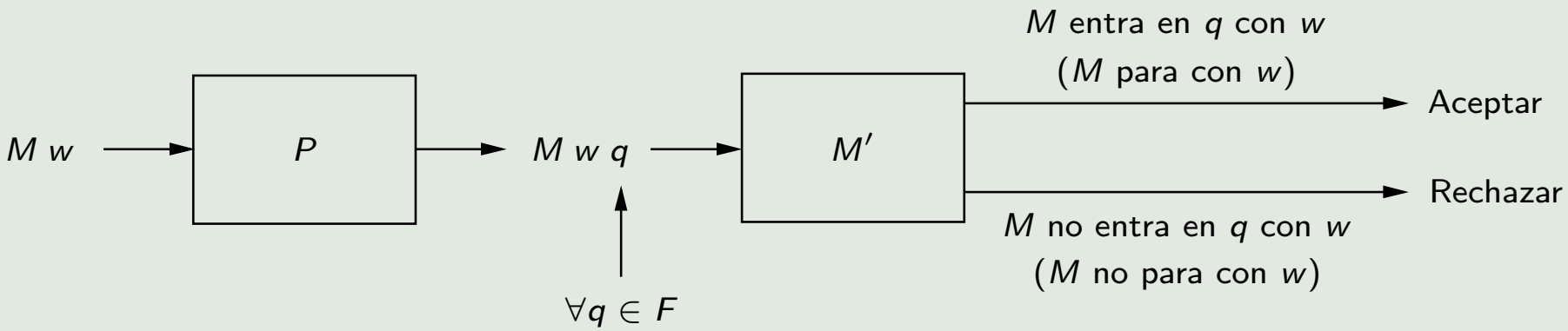
La demostración se hace aplicando una reducción adecuada del problema de la parada al SEP. Así pues, supongamos que el SEP es resoluble mediante una MT M':



A continuación, añadimos un preprocesador P que reduce el problema de la parada al SEP, alimentando M' con todos los estados finales de M. .../...

Ejercicio (continuación)

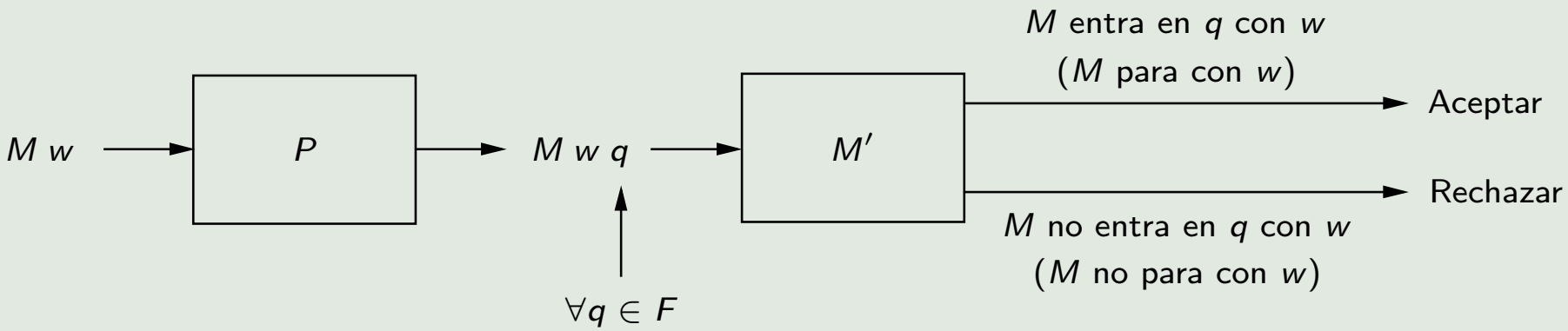
Obtenemos entonces la siguiente composición:



Esta composición estaría resolviendo el problema de la parada.
Por lo tanto, el SEP es irresoluble.

Ejercicio (continuación)

Obtenemos entonces la siguiente composición:



Esta composición estaría resolviendo el problema de la parada.
Por lo tanto, el SEP es irresoluble.

Ejercicio propuesto

Utilizando las reducciones apropiadas, demuestre que el siguiente problema es irresoluble:

> *Dada una MT arbitraria M, con alfabeto de cinta Γ, y un símbolo $\gamma \in \Gamma$, si M comienza con la cinta en blanco, ¿escribirá M el símbolo γ en la cinta alguna vez?*

Contenidos

2 El problema de correspondencia de Post

Los problemas irresolubles que hemos visto están relacionados con las propiedades de las MT,s aunque tienen consecuencias en otras áreas. Sin embargo, a veces es difícil obtener dichas consecuencias a partir del problema de la parada. Por tanto, en esta sección obtendremos la irresolubilidad del **problema de correspondencia de Post** o **PCP**.

2 El problema de correspondencia de Post

Los problemas irresolubles que hemos visto están relacionados con las propiedades de las MT,s aunque tienen consecuencias en otras áreas. Sin embargo, a veces es difícil obtener dichas consecuencias a partir del problema de la parada. Por tanto, en esta sección obtendremos la irresolubilidad del **problema de correspondencia de Post** o **PCP**.

Definición

Un caso del PCP se denomina **sistema de correspondencia de Post** o **SCP**, y está compuesto por tres elementos:

- Un alfabeto Σ.
- Y dos conjuntos A y B de cadenas de Σ^+, donde ambos conjuntos tienen el mismo cardinal, es decir, $A = \{u_1, u_2, \ldots, u_k\}$ y $B = \{v_1, v_2, \ldots, v_k\}$.

2 El problema de correspondencia de Post

Los problemas irresolubles que hemos visto están relacionados con las propiedades de las MT,s aunque tienen consecuencias en otras áreas. Sin embargo, a veces es difícil obtener dichas consecuencias a partir del problema de la parada. Por tanto, en esta sección obtendremos la irresolubilidad del **problema de correspondencia de Post** o **PCP**.

Definición

Un caso del PCP se denomina **sistema de correspondencia de Post** o **SCP**, y está compuesto por tres elementos:

- Un alfabeto Σ.
- Y dos conjuntos A y B de cadenas de Σ^+, donde ambos conjuntos tienen el mismo cardinal, es decir, $A = \{u_1, u_2, \ldots, u_k\}$ y $B = \{v_1, v_2, \ldots, v_k\}$.

Una solución para el SCP es una secuencia de índices $i_1, i_2, \ldots, i_n$, tal que

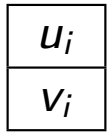
$$u_{i_1}\, u_{i_2}\, \ldots\, u_{i_n} = v_{i_1}\, v_{i_2}\, \ldots\, v_{i_n}$$

Por ejemplo, si $\Sigma = \{a, b\}$, $A = \{a, abaaa, ab\}$ y $B = \{aaa, ab, b\}$, la solución a este SCP viene dada mediante la secuencia de índices 2,1,1,3, ya que

$$u_2\, u_1\, u_1\, u_3 = v_2\, v_1\, v_1\, v_3 = abaaaaaab.$$

2 El problema de correspondencia de Post

A veces resulta muy útil interpretar el SCP como un conjunto de bloques o "fichas de dominó" de la forma:



2 El problema de correspondencia de Post

A veces resulta muy útil interpretar el SCP como un conjunto de bloques o "fichas de dominó" de la forma:

u_i
v_i

Por tanto, para el SCP del ejemplo anterior, tenemos:

a	$abaaa$	ab
aaa	ab	b
1	2	3

2 El problema de correspondencia de Post

A veces resulta muy útil interpretar el SCP como un conjunto de bloques o "fichas de dominó" de la forma:

u_i
v_i

Por tanto, para el SCP del ejemplo anterior, tenemos:

a	$abaaa$	ab
aaa	ab	b
1	2	3

La solución del SCP se corresponde entonces con la forma en que se colocan los bloques o fichas, de tal manera que la cadena formada por las celdas superiores sea igual a la cadena formada por las celdas inferiores. Por tanto, la solución anterior se representa como:

$abaaa$	a	a	ab
ab	aaa	aaa	b
2	1	1	3

2 El problema de correspondencia de Post

Sin embargo, consideremos ahora el SCP siguiente:

ab	baa	aba
aba	aa	baa
1	2	3

2 El problema de correspondencia de Post

Sin embargo, consideremos ahora el SCP siguiente:

ab	*baa*	*aba*
aba	*aa*	*baa*
1	2	3

Cualquier solución a este SCP debe empezar con la ficha 1, ya que es la única donde ambas cadenas empiezan con el mismo símbolo. La siguiente ficha debe empezar con *a* arriba. Es decir, en principio valdrían la 1 y 3. Pero la 1 no sirve ya que obtendríamos cadenas diferentes arriba y abajo. Por tanto, la segunda ficha debe ser la 3, para obtener:

ab	*aba*
aba	*baa*
1	3

2 El problema de correspondencia de Post

Sin embargo, consideremos ahora el SCP siguiente:

ab	*baa*	*aba*
aba	*aa*	*baa*
1	2	3

Cualquier solución a este SCP debe empezar con la ficha 1, ya que es la única donde ambas cadenas empiezan con el mismo símbolo. La siguiente ficha debe empezar con *a* arriba. Es decir, en principio valdrían la 1 y 3. Pero la 1 no sirve ya que obtendríamos cadenas diferentes arriba y abajo. Por tanto, la segunda ficha debe ser la 3, para obtener:

ab	*aba*
aba	*baa*
1	3

De forma similar, observamos que la tercera ficha tiene que ser también la 3, obteniéndose:

ab	*aba*	*aba*
aba	*baa*	*baa*
1	3	3

2 El problema de correspondencia de Post

Sin embargo, consideremos ahora el SCP siguiente:

ab	*baa*	*aba*
aba	*aa*	*baa*
1	2	3

Cualquier solución a este SCP debe empezar con la ficha 1, ya que es la única donde ambas cadenas empiezan con el mismo símbolo. La siguiente ficha debe empezar con *a* arriba. Es decir, en principio valdrían la 1 y 3. Pero la 1 no sirve ya que obtendríamos cadenas diferentes arriba y abajo. Por tanto, la segunda ficha debe ser la 3, para obtener:

ab	*aba*
aba	*baa*
1	3

De forma similar, observamos que la tercera ficha tiene que ser también la 3, obteniéndose:

ab	*aba*	*aba*
aba	*baa*	*baa*
1	3	3

Pero este razonamiento es infinito, ya que nunca podemos elegir una ficha final que consiga que la cadena superior llegue a tener la misma longitud que la cadena inferior, siendo ambas iguales. Por tanto, este SCP no tiene solución.

2 El problema de correspondencia de Post

Definición

El **Problema de Correspondencia de Post** o **PCP** consiste precisamente en determinar si un SCP arbitrario tiene solución o no.

2 El problema de correspondencia de Post

Definición

El **Problema de Correspondencia de Post** o **PCP** consiste precisamente en determinar si un SCP arbitrario tiene solución o no.

Hemos visto dos ejemplos de SCP: uno tenía solución y el otro no. Sin embargo, veremos que no existe ningún algoritmo capaz de decidir si un SCP arbitrario tiene solución o no. Es decir, **el PCP es irresoluble**.

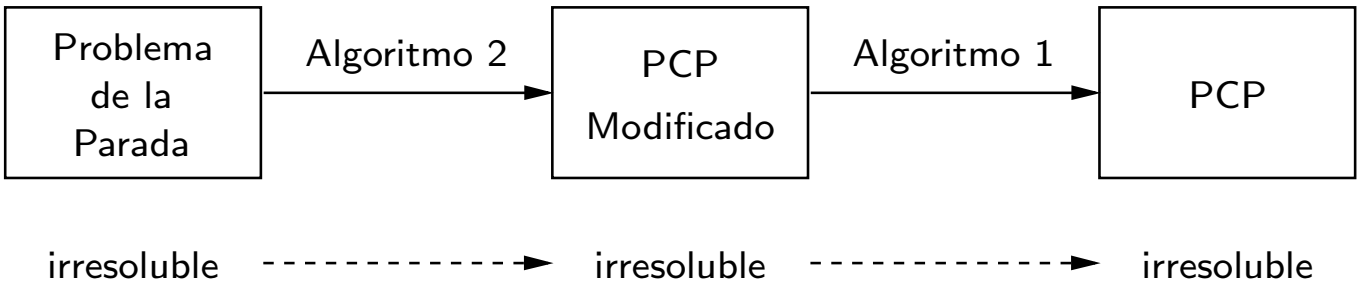
2 El problema de correspondencia de Post

Definición

El **Problema de Correspondencia de Post** o **PCP** consiste precisamente en determinar si un SCP arbitrario tiene solución o no.

Hemos visto dos ejemplos de SCP: uno tenía solución y el otro no. Sin embargo, veremos que no existe ningún algoritmo capaz de decidir si un SCP arbitrario tiene solución o no. Es decir, **el PCP es irresoluble**.

Lo probaremos demostrando que si fuera resoluble, entonces podríamos resolver también el problema de la parada. Es decir, utilizaremos también el principio de reducción, en este caso en dos pasos:



2 El problema de correspondencia de Post

ALGORITMO 1. Primero modificaremos el PCP y demostraremos que si el PCP fuera resoluble, el *PCP Modificado* o *PCPM* también lo sería. El PCPM consiste en buscar una solución a un SCP en el que la primera ficha a colocar es fija y se le asigna el índice 1.

2 El problema de correspondencia de Post

ALGORITMO 1. Primero modificaremos el PCP y demostraremos que si el PCP fuera resoluble, el *PCP Modificado* o *PCPM* también lo sería. El PCPM consiste en buscar una solución a un SCP en el que la primera ficha a colocar es fija y se le asigna el índice 1.

Sean $A = \{u_1, u_2, \ldots, u_k\}$ y $B = \{v_1, v_2, \ldots, v_k\}$ una muestra del PCPM sobre un alfabeto Σ. Supongamos que cada $u_i = a_{i_1} a_{i_2} \ldots a_{i_{m_i}}$ y cada $v_i = b_{i_1} b_{i_2} \ldots b_{i_{n_i}}$. Entonces, para cada i, hacemos lo siguiente:

$$y_i = a_{i_1} \$ a_{i_2} \$ \ldots a_{i_{m_i}} \$ \qquad z_i = \$ b_{i_1} \$ b_{i_2} \ldots \$ b_{i_{n_i}}$$

donde $\$ \notin \Sigma$. Obsérvese que y_i es u_i añadiendo \$ después de cada símbolo, y z_i es v_i añadiendo \$ antes de cada símbolo.

2 El problema de correspondencia de Post

ALGORITMO 1. Primero modificaremos el PCP y demostraremos que si el PCP fuera resoluble, el *PCP Modificado* o *PCPM* también lo sería. El PCPM consiste en buscar una solución a un SCP en el que la primera ficha a colocar es fija y se le asigna el índice 1.

Sean $A = \{u_1, u_2, \ldots, u_k\}$ y $B = \{v_1, v_2, \ldots, v_k\}$ una muestra del PCPM sobre un alfabeto Σ. Supongamos que cada $u_i = a_{i_1} a_{i_2} \ldots a_{i_{m_i}}$ y cada $v_i = b_{i_1} b_{i_2} \ldots b_{i_{n_i}}$. Entonces, para cada i, hacemos lo siguiente:

$$y_i = a_{i_1} \$ a_{i_2} \$ \ldots a_{i_{m_i}} \$ \qquad z_i = \$ b_{i_1} \$ b_{i_2} \ldots \$ b_{i_{n_i}}$$

donde $\$ \notin \Sigma$. Obsérvese que y_i es u_i añadiendo \$ después de cada símbolo, y z_i es v_i añadiendo \$ antes de cada símbolo.

Sea ahora otro símbolo $\% \notin \Sigma$. Hacemos:

$$y_0 = \$ y_1 \qquad z_0 = z_1 \qquad y_{k+1} = \% \qquad z_{k+1} = \$ \%$$

Lo que hemos hecho es construir un ejemplo de PCP formado por las fichas:

y_0	y_1	$\cdots$	y_k	y_{k+1}
z_0	z_1	$\cdots$	z_k	z_{k+1}

2 El problema de correspondencia de Post

Esta muestra del PCP tiene solución si y sólo si el PCPM original dado por A y B tiene solución. Para probar esto, supongamos que $i_1, i_2, \ldots, i_r$ constituye una solución del PCP:

- Dado que todos los z_i empiezan por \$ y sólo y_0 empieza por \$, entonces $i_1 = 0$.
- Es más, $i_r = k + 1$, puesto que sólo y_{k+1} y z_{k+1} coinciden en su último símbolo.

2 El problema de correspondencia de Post

Esta muestra del PCP tiene solución si y sólo si el PCPM original dado por A y B tiene solución. Para probar esto, supongamos que $i_1, i_2, \ldots, i_r$ constituye una solución del PCP:

- Dado que todos los z_i empiezan por \$ y sólo y_0 empieza por \$, entonces $i_1 = 0$.
- Es más, $i_r = k + 1$, puesto que sólo y_{k+1} y z_{k+1} coinciden en su último símbolo.

Por tanto, si hay una solución para el PCP, ésta debe ser $0, i_2, i_3, \ldots, i_{r-1}, k + 1$, es decir:

y_0	$\cdots$	y_{k+1}
z_0	$\cdots$	z_{k+1}

o de forma equivalente:

$$\$\, a_{1_1}\, \$\, a_{1_2}\, \$ \ldots \$\, a_{1_{m_1}}\, \$ \ldots \$\, \% = \$\, b_{1_1}\, \$\, b_{1_2}\, \$ \ldots \$\, b_{1_{n_1}}\, \$ \ldots \$\, \%$$

2 El problema de correspondencia de Post

Esta muestra del PCP tiene solución si y sólo si el PCPM original dado por A y B tiene solución. Para probar esto, supongamos que $i_1, i_2, \ldots, i_r$ constituye una solución del PCP:

- Dado que todos los z_i empiezan por \$ y sólo y_0 empieza por \$, entonces $i_1 = 0$.
- Es más, $i_r = k + 1$, puesto que sólo y_{k+1} y z_{k+1} coinciden en su último símbolo.

Por tanto, si hay una solución para el PCP, ésta debe ser $0, i_2, i_3, \ldots, i_{r-1}, k + 1$, es decir:

y_0	$\cdots$	y_{k+1}
z_0	$\cdots$	z_{k+1}

o de forma equivalente:

$$\$\, a_{1_1}\, \$\, a_{1_2}\, \$ \ldots \$\, a_{1_{m_1}}\, \$ \ldots \$\, \% = \$\, b_{1_1}\, \$\, b_{1_2}\, \$ \ldots \$\, b_{1_{n_1}}\, \$ \ldots \$\, \%$$

Y si se ignoran los símbolos \$, se obtiene:

$$u_1\, u_{i_2} \ldots u_{i_{r-1}} = v_1\, v_{i_2} \ldots v_{i_{r-1}}$$

lo cual es una solución del PCPM. Es decir, si el PCP es resoluble, entonces el PCPM también es resoluble.

2 El problema de correspondencia de Post

ALGORITMO 2. Ahora veremos que si el PCPM fuera resoluble, el problema de la parada también lo sería. Supongamos que el PCPM es resoluble, es decir, que existe un algoritmo para determinar si cualquier PCPM tiene o no solución. Nuestro objetivo será demostrar que existe un algoritmo para determinar si una MT arbitraria parará con cualquier cadena w en su cinta.

2 El problema de correspondencia de Post

ALGORITMO 2. Ahora veremos que si el PCPM fuera resoluble, el problema de la parada también lo sería. Supongamos que el PCPM es resoluble, es decir, que existe un algoritmo para determinar si cualquier PCPM tiene o no solución. Nuestro objetivo será demostrar que existe un algoritmo para determinar si una MT arbitraria parará con cualquier cadena w en su cinta.

Sea $M = (Q, \Sigma, \Gamma, s, B, F, \delta)$ una MT que sólo para en los estados de aceptación (esto puede hacerse sin modificar $L(M)$, añadiendo transiciones que provoquen bucles infinitos cada vez que M pare en un estado no final) y sea w una cadena de Σ^*. Mostraremos cómo construir una muestra del PCPM, para la cual la capacidad de determinar si existe o no solución implica que existe la capacidad de determinar si M para con entrada w (y, por tanto, si $w \in L(M)$).

2 El problema de correspondencia de Post

ALGORITMO 2. Ahora veremos que si el PCPM fuera resoluble, el problema de la parada también lo sería. Supongamos que el PCPM es resoluble, es decir, que existe un algoritmo para determinar si cualquier PCPM tiene o no solución. Nuestro objetivo será demostrar que existe un algoritmo para determinar si una MT arbitraria parará con cualquier cadena w en su cinta.

Sea $M = (Q, \Sigma, \Gamma, s, B, F, \delta)$ una MT que sólo para en los estados de aceptación (esto puede hacerse sin modificar $L(M)$, añadiendo transiciones que provoquen bucles infinitos cada vez que M pare en un estado no final) y sea w una cadena de Σ^*. Mostraremos cómo construir una muestra del PCPM, para la cual la capacidad de determinar si existe o no solución implica que existe la capacidad de determinar si M para con entrada w (y, por tanto, si $w \in L(M)$).

A partir de la MT M y de la cadena w, y siendo $\$ \notin \Gamma$, construimos $A = \{u_1, u_2, \ldots, u_k\}$ y $B = \{v_1, v_2, \ldots, v_k\}$, los conjuntos de la muestra del PCPM, mediante cuatro grupos de fichas:

1. La primera ficha será:

$
sw

2. Para cada símbolo $\sigma \in \Gamma$, excepto para el símbolo blanco, construimos:

σ
σ

Y construimos también la ficha:

$
$

3. Derivadas de las transiciones de M:

$$\text{Si } \delta(q,\sigma) = (p,\tau,R), \text{ añadimos } \boxed{\begin{array}{|c|}\hline q\sigma \\ \hline \tau p \\ \hline \end{array}}$$

$$\text{Si } \delta(q,\sigma) = (p,\tau,L), \text{ añadimos } \boxed{\begin{array}{|c|}\hline \gamma q\sigma \\ \hline p\gamma\tau \\ \hline \end{array}} \ \forall \gamma \in \Gamma - \{B\}$$

$$\text{Si } \delta(q,B) = (p,\tau,R), \text{ añadimos } \boxed{\begin{array}{|c|}\hline q\$ \\ \hline \tau p\$ \\ \hline \end{array}}$$

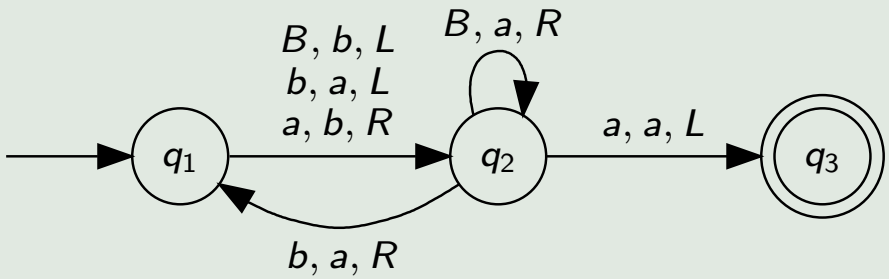
$$\text{Si } \delta(q,B) = (p,\tau,L), \text{ añadimos } \boxed{\begin{array}{|c|}\hline \gamma q\$ \\ \hline p\gamma\tau\$ \\ \hline \end{array}} \ \forall \gamma \in \Gamma - \{B\}$$

4. Derivadas del conjunto de estados finales F, $\forall q \in F$ y $\forall \sigma, \tau \in \Gamma - \{B\}$, añadimos:

$$\boxed{\begin{array}{|c|}\hline \sigma q\tau \\ \hline q \\ \hline \end{array}} \quad \boxed{\begin{array}{|c|}\hline \sigma q\$ \\ \hline q\$ \\ \hline \end{array}} \quad \boxed{\begin{array}{|c|}\hline \$q\tau \\ \hline \$q \\ \hline \end{array}} \quad \boxed{\begin{array}{|c|}\hline q\$\$ \\ \hline \$ \\ \hline \end{array}}$$

Ejercicio

Obtenga la muestra del PCPM derivada de la siguiente MT M y de la cadena $w = ab$.



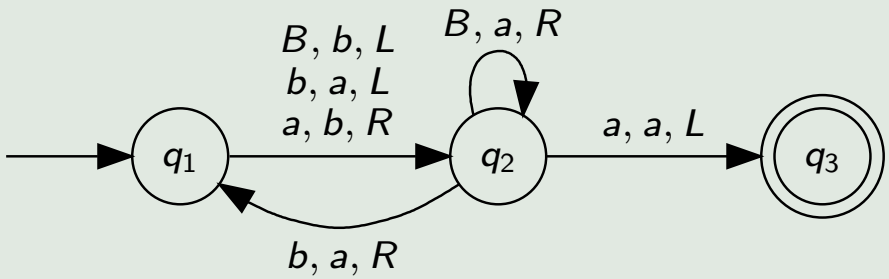
La siguiente computación muestra que $w \in L(M)$:

$$q_1\, a\, b \vdash b\, q_2\, b \vdash b\, a\, q_1 \vdash b\, q_2\, a\, b \vdash q_3\, b\, a\, b$$

Por tanto, obtenga también una solución para la muestra del PCPM derivada de M y w.

Ejercicio

Obtenga la muestra del PCPM derivada de la siguiente MT M y de la cadena $w = ab$.



La siguiente computación muestra que $w \in L(M)$:

$$q_1\, a\, b \vdash b\, q_2\, b \vdash b\, a\, q_1 \vdash b\, q_2\, a\, b \vdash q_3\, b\, a\, b$$

Por tanto, obtenga también una solución para la muestra del PCPM derivada de M y w.

Solución:

La muestra del PCPM derivada de M y w estará formada por las siguientes fichas:

1. Primera ficha:

$
$q_1 ab$

1

2. A partir del alfabeto de la cinta:

a	b	$
a	b	$

2 3 4

.../...

Ejercicio (continuación)

3. A partir de $\delta(q_1, a) = (q_2, b, R)$:

q_1a
bq_2

5

A partir de $\delta(q_2, a) = (q_3, a, L)$:

aq_2a	bq_2a
q_3aa	q_3ba

10 11

A partir de $\delta(q_1, b) = (q_2, a, L)$:

aq_1b	bq_1b
q_2aa	q_2ba

6 7

A partir de $\delta(q_2, b) = (q_1, a, R)$:

q_2b
aq_1

12

A partir de $\delta(q_1, B) = (q_2, b, L)$:

$aq_1\$$	$bq_1\$$
$q_2ab\$$	$q_2bb\$$

8 9

A partir de $\delta(q_2, B) = (q_2, a, R)$:

$q_2\$$
$aq_2\$$

13

.../...

Ejercicio (continuación)

4. A partir de los estados finales:

aq_3a	aq_3b	bq_3a	bq_3b
q_3	q_3	q_3	q_3
14	15	16	17

$aq_3\$$	$bq_3\$$
$q_3\$$	$q_3\$$
18	19

$\$q_3a$	$\$q_3b$
$\$q_3$	$\$q_3$
20	21

$q_3\$\$$
$\$$
22

Ejercicio (continuación)

4. A partir de los estados finales:

aq_3a	aq_3b	bq_3a	bq_3b
q_3	q_3	q_3	q_3
14	15	16	17

$aq_3\$$	$bq_3\$$
$q_3\$$	$q_3\$$
18	19

$\$q_3a$	$\$q_3b$
$\$q_3$	$\$q_3$
20	21

$q_3\$\$$
$\$$
22

Una solución para esta muestra del PCPM viene dada por la siguiente secuencia de fichas:

$\$$	q_1a	b	$\$$	b	q_2b	$\$$	b	$aq_1\$$	bq_2a	b
$\$q_1ab\$$	bq_2	b	$\$$	b	aq_1	$\$$	b	$q_2ab\$$	q_3ba	b
1	5	3	4	3	12	4	3	8	11	3

$\$q_3b$	a	b	$\$q_3a$	b	$\$q_3b$	$\$$	$q_3\$\$$
$\$q_3$	a	b	$\$q_3$	b	$\$q_3$	$\$$	$\$$
21	2	3	20	3	21	4	22

Ejercicio (continuación)

4. A partir de los estados finales:

aq_3a	aq_3b	bq_3a	bq_3b
q_3	q_3	q_3	q_3
14	15	16	17

$aq_3\$$	$bq_3\$$
$q_3\$$	$q_3\$$
18	19

$\$q_3a$	$\$q_3b$
$\$q_3$	$\$q_3$
20	21

$q_3\$\$$
$\$$
22

Una solución para esta muestra del PCPM viene dada por la siguiente secuencia de fichas:

$\$$	q_1a	b	$\$$	b	q_2b	$\$$	b	$aq_1\$$	bq_2a	b
$\$q_1ab\$$	bq_2	b	$\$$	b	aq_1	$\$$	b	$q_2ab\$$	q_3ba	b
1	5	3	4	3	12	4	3	8	11	3

$\$q_3b$	a	b	$\$q_3a$	b	$\$q_3b$	$\$$	$q_3\$\$$
$\$q_3$	a	b	$\$q_3$	b	$\$q_3$	$\$$	$\$$
21	2	3	20	3	21	4	22

Como se puede obsevar, tanto los símbolos de la parte superior, como los de la parte inferior de las fichas, representan precisamente la secuencia de configuraciones instantáneas (separadas por símbolos \$) por las que va pasando la MT M al procesar la cadena w.

2 El problema de correspondencia de Post

Por tanto, si M no parara sobre una cadena w, entonces la muestra del PCPM derivada no tendría solución. Las cadenas u_i y v_i se agrupan para formar configuraciones consecutivas de M, donde cada configuración aparece entre símbolos \$. Si M nunca pasa a un estado de parada, no se puede añadir ninguna ficha del cuarto grupo. Por inducción, se podría demostrar que las cadenas de u_i y v_i siempre tendrían un número distinto de símbolos \$, y por tanto la muestra del PCPM no tendría solución.

Por tanto, si M no parara sobre una cadena w, entonces la muestra del PCPM derivada no tendría solución. Las cadenas u_i y v_i se agrupan para formar configuraciones consecutivas de M, donde cada configuración aparece entre símbolos \$. Si M nunca pasa a un estado de parada, no se puede añadir ninguna ficha del cuarto grupo. Por inducción, se podría demostrar que las cadenas de u_i y v_i siempre tendrían un número distinto de símbolos \$, y por tanto la muestra del PCPM no tendría solución.

En resumen:

Teorema

Una MT arbitraria M para sobre cualquier cadena w si y sólo si la muestra del PCPM derivada a partir de M y w tiene solución. □

Por tanto, si M no parara sobre una cadena w, entonces la muestra del PCPM derivada no tendría solución. Las cadenas u_i y v_i se agrupan para formar configuraciones consecutivas de M, donde cada configuración aparece entre símbolos \$. Si M nunca pasa a un estado de parada, no se puede añadir ninguna ficha del cuarto grupo. Por inducción, se podría demostrar que las cadenas de u_i y v_i siempre tendrían un número distinto de símbolos \$, y por tanto la muestra del PCPM no tendría solución.

En resumen:

Teorema

Una MT arbitraria M para sobre cualquier cadena w si y sólo si la muestra del PCPM derivada a partir de M y w tiene solución. □

Así pues, queda demostrado que el PCPM es irresoluble y en consecuencia:

Teorema

El Problema de Correspondencia de Post es irresoluble. □

Contenidos

3 Problemas no decidibles en LIC,s

Las GIC,s constituyen una herramienta muy útil para definir la sintaxis de los lenguajes de programación. Así pues, es necesario señalar que la irresolubilidad del PCP se puede utilizar para establecer la irresolubilidad de algunas cuestiones importantes que afectan a los LIC,s (es decir, a los lenguajes generados por las GIC,s).

3 Problemas no decidibles en LIC,s

Las GIC,s constituyen una herramienta muy útil para definir la sintaxis de los lenguajes de programación. Así pues, es necesario señalar que la irresolubilidad del PCP se puede utilizar para establecer la irresolubilidad de algunas cuestiones importantes que afectan a los LIC,s (es decir, a los lenguajes generados por las GIC,s).

Sea $C = \{(u_1, v_1), (u_2, v_2), \ldots, (u_n, v_n)\}$ un SCP, donde las cadenas u_i y v_i están formadas con símbolos del alfabeto Σ_C. A partir de las fichas ordenadas de C, podemos construir dos gramáticas independientes del contexto G_U y G_V, como sigue:

$$
\begin{aligned}
G_U: \quad & N_U = \{S_U\} \\
& \Sigma_U = \Sigma_C \cup \{1, 2, \ldots, n\} \\
& P_U = \{S_U \to u_i\, S_U\, i,\ S_U \to u_i\, i \mid i = 1, 2, \ldots, n\} \\
G_V: \quad & N_V = \{S_V\} \\
& \Sigma_V = \Sigma_C \cup \{1, 2, \ldots, n\} \\
& P_V = \{S_V \to v_i\, S_V\, i,\ S_V \to v_i\, i \mid i = 1, 2, \ldots, n\}
\end{aligned}
$$

3 Problemas no decidibles en LIC,s

Las GIC,s constituyen una herramienta muy útil para definir la sintaxis de los lenguajes de programación. Así pues, es necesario señalar que la irresolubilidad del PCP se puede utilizar para establecer la irresolubilidad de algunas cuestiones importantes que afectan a los LIC,s (es decir, a los lenguajes generados por las GIC,s).

Sea $C = \{(u_1, v_1), (u_2, v_2), \ldots, (u_n, v_n)\}$ un SCP, donde las cadenas u_i y v_i están formadas con símbolos del alfabeto Σ_C. A partir de las fichas ordenadas de C, podemos construir dos gramáticas independientes del contexto G_U y G_V, como sigue:

$$
\begin{aligned}
G_U: \quad & N_U = \{S_U\} \\
& \Sigma_U = \Sigma_C \cup \{1, 2, \ldots, n\} \\
& P_U = \{S_U \to u_i\, S_U\, i,\ S_U \to u_i\, i \mid i = 1, 2, \ldots, n\} \\
G_V: \quad & N_V = \{S_V\} \\
& \Sigma_V = \Sigma_C \cup \{1, 2, \ldots, n\} \\
& P_V = \{S_V \to v_i\, S_V\, i,\ S_V \to v_i\, i \mid i = 1, 2, \ldots, n\}
\end{aligned}
$$

Determinar si el SCP C tiene una solución se reduce a determinar si tienen respuesta ciertas cuestiones relativas a las derivaciones de las gramáticas G_U y G_V.

3 Problemas no decidibles en LIC,s

La gramática G_U genera las cadenas de símbolos que aparecen en la parte superior de las fichas del sistema C, pero acompañadas al final de los números de ficha utilizados, en orden inverso. De manera similar, G_V genera cadenas del mismo estilo que corresponden a la parte inferior de las fichas de C.

3 Problemas no decidibles en LIC,s

La gramática G_U genera las cadenas de símbolos que aparecen en la parte superior de las fichas del sistema C, pero acompañadas al final de los números de ficha utilizados, en orden inverso. De manera similar, G_V genera cadenas del mismo estilo que corresponden a la parte inferior de las fichas de C.

El SCP C tiene una solución si existe una secuencia $i_1 i_2 \dots i_k$ tal que:

$$u_{i_1} u_{i_2} \dots u_{i_k} = v_{i_1} v_{i_2} \dots v_{i_k}$$

Pero en este caso, G_U y G_V pueden realizar las siguientes derivaciones:

$$S_U \overset{*}{\Rightarrow} u_{i_1} u_{i_2} \dots u_{i_k} i_k \dots i_2 i_1$$
$$S_V \overset{*}{\Rightarrow} v_{i_1} v_{i_2} \dots v_{i_k} i_k \dots i_2 i_1$$

donde $u_{i_1} u_{i_2} \dots u_{i_k} i_k \dots i_2 i_1 = v_{i_1} v_{i_2} \dots v_{i_k} i_k \dots i_2 i_1$. Así pues, en este caso, la intersección de $L(G_U)$ y $L(G_V)$ es no vacía.

3 Problemas no decidibles en LIC,s

La gramática G_U genera las cadenas de símbolos que aparecen en la parte superior de las fichas del sistema C, pero acompañadas al final de los números de ficha utilizados, en orden inverso. De manera similar, G_V genera cadenas del mismo estilo que corresponden a la parte inferior de las fichas de C.

El SCP C tiene una solución si existe una secuencia $i_1 i_2 \dots i_k$ tal que:

$$u_{i_1} u_{i_2} \dots u_{i_k} = v_{i_1} v_{i_2} \dots v_{i_k}$$

Pero en este caso, G_U y G_V pueden realizar las siguientes derivaciones:

$$S_U \overset{*}{\Rightarrow} u_{i_1} u_{i_2} \dots u_{i_k} i_k \dots i_2 i_1$$
$$S_V \overset{*}{\Rightarrow} v_{i_1} v_{i_2} \dots v_{i_k} i_k \dots i_2 i_1$$

donde $u_{i_1} u_{i_2} \dots u_{i_k} i_k \dots i_2 i_1 = v_{i_1} v_{i_2} \dots v_{i_k} i_k \dots i_2 i_1$. Así pues, en este caso, la intersección de $L(G_U)$ y $L(G_V)$ es no vacía.

Y de manera inversa, supongamos que $w \in L(G_U) \cap L(G_V)$.
Entonces w consta de una cadena $w' \in \Sigma_C^+$ seguida de una secuencia $i_k \dots i_2 i_1$.
Y la cadena $w' = u_{i_1} u_{i_2} \dots u_{i_k} = v_{i_1} v_{i_2} \dots v_{i_k}$ es una solución para C.

3 Problemas no decidibles en LIC,s

Así pues, completamos la sección demostrando los tres teoremas que se citan a continuación. Veremos que todas las demostraciones utilizan la misma técnica: reducen el PCP a cada uno de los problemas considerados, resultando por tanto que todos ellos son también irresolubles.

3 Problemas no decidibles en LIC,s

Así pues, completamos la sección demostrando los tres teoremas que se citan a continuación. Veremos que todas las demostraciones utilizan la misma técnica: reducen el PCP a cada uno de los problemas considerados, resultando por tanto que todos ellos son también irresolubles.

Teorema

No existe ningún algoritmo que determine si los lenguajes generados por dos GIC,s son disjuntos.

3 Problemas no decidibles en LIC,s

Así pues, completamos la sección demostrando los tres teoremas que se citan a continuación. Veremos que todas las demostraciones utilizan la misma técnica: reducen el PCP a cada uno de los problemas considerados, resultando por tanto que todos ellos son también irresolubles.

Teorema

No existe ningún algoritmo que determine si los lenguajes generados por dos GIC,s son disjuntos.

Demostración:

Supongamos que tal algoritmo existe. Entonces, el PCP podría ser resuelto como sigue:

1. Para un SCP arbitrario C, construimos las gramáticas G_U y G_V a partir de las fichas ordenadas de C.
2. Utilizamos el algoritmo para determinar si $L(G_U)$ y $L(G_V)$ son disjuntos.
3. Entonces C tiene solución si y sólo si $L(G_U) \cap L(G_V)$ es no vacía.

3 Problemas no decidibles en LIC,s

Así pues, completamos la sección demostrando los tres teoremas que se citan a continuación. Veremos que todas las demostraciones utilizan la misma técnica: reducen el PCP a cada uno de los problemas considerados, resultando por tanto que todos ellos son también irresolubles.

Teorema

No existe ningún algoritmo que determine si los lenguajes generados por dos GIC,s son disjuntos.

Demostración:

Supongamos que tal algoritmo existe. Entonces, el PCP podría ser resuelto como sigue:

1. Para un SCP arbitrario C, construimos las gramáticas G_U y G_V a partir de las fichas ordenadas de C.
2. Utilizamos el algoritmo para determinar si $L(G_U)$ y $L(G_V)$ son disjuntos.
3. Entonces C tiene solución si y sólo si $L(G_U) \cap L(G_V)$ es no vacía.

El paso 1 reduce el PCP al problema de determinar si dos LIC,s son disjuntos.
Dado que el PCP es irresoluble, la cuestión de la disjunción de LIC,s también lo es. □

3 Problemas no decidibles en LIC,s

Teorema

No existe ningún algoritmo que determine si el lenguaje generado por una GIC $G = (N, \Sigma, P, S)$ es Σ^*.

3 Problemas no decidibles en LIC,s

Teorema

No existe ningún algoritmo que determine si el lenguaje generado por una GIC $G = (N, \Sigma, P, S)$ es Σ^*.

Demostración:

En general, los LIC,s no son cerrados bajo la operación de complementario.
Sin embargo, para un SCP arbitrario C, los lenguajes $\overline{L(G_U)}$ y $\overline{L(G_V)}$ son siempre LIC,s.

Así pues, nos fijaremos también en que $L = \Sigma^*$ es equivalente a $\overline{L} = \emptyset$,
y demostraremos que no existe ningún algoritmo que determine si $\overline{L(G)}$ es vacío.

3 Problemas no decidibles en LIC,s

Teorema

No existe ningún algoritmo que determine si el lenguaje generado por una GIC $G = (N, \Sigma, P, S)$ es Σ^*.

Demostración:

En general, los LIC,s no son cerrados bajo la operación de complementario.
Sin embargo, para un SCP arbitrario C, los lenguajes $\overline{L(G_U)}$ y $\overline{L(G_V)}$ son siempre LIC,s.

Así pues, nos fijaremos también en que $L = \Sigma^*$ es equivalente a $\overline{L} = \emptyset$,
y demostraremos que no existe ningún algoritmo que determine si $\overline{L(G)}$ es vacío.

Dado que $\overline{L(G_U)}$ y $\overline{L(G_V)}$ son LIC,s, también lo es $L = \overline{L(G_U)} \cup \overline{L(G_V)}$.
Pero el complementario de L es $\overline{L} = L(G_U) \cap L(G_V)$.

Así que un algoritmo que determine si $\overline{L} = \emptyset$ podría ser utilizado para determinar si $L(G_U)$ y $L(G_V)$ son disjuntos, lo cual hemos visto que es un problema irresoluble. □

3 Problemas no decidibles en LIC,s

Teorema

No existe ningún algoritmo que determine si una GIC arbitraria es ambigua.

Teorema

No existe ningún algoritmo que determine si una GIC arbitraria es ambigua.

Demostración:

Una GIC es ambigua si contiene una cadena que puede ser generada por dos derivaciones a izquierdas diferentes. Como antes, partimos de un SCP arbitrario C y construimos G_U y G_V. Ahora las combinamos para obtener esta nueva gramática, cuyo símbolo inicial es S:

$$\begin{aligned} G: \quad & N = \{S, S_U, S_V\} \\ & \Sigma = \Sigma_C \cup \{1, 2, \ldots, n\} \\ & P = P_U \cup P_V \cup \{S \to S_U,\ S \to S_V\} \end{aligned}$$

Teorema

No existe ningún algoritmo que determine si una GIC arbitraria es ambigua.

Demostración:

Una GIC es ambigua si contiene una cadena que puede ser generada por dos derivaciones a izquierdas diferentes. Como antes, partimos de un SCP arbitrario C y construimos G_U y G_V. Ahora las combinamos para obtener esta nueva gramática, cuyo símbolo inicial es S:

$$\begin{aligned} G: \quad & N = \{S, S_U, S_V\} \\ & \Sigma = \Sigma_C \cup \{1, 2, \ldots, n\} \\ & P = P_U \cup P_V \cup \{S \to S_U,\ S \to S_V\} \end{aligned}$$

Claramente, todas las derivaciones de G son a izquierdas, ya que toda forma sentencial contiene como máximo un símbolo no terminal. Una derivación de G consiste en la aplicación de una de las reescrituras de S, seguida de una derivación de G_U o de G_V. Y las gramáticas G_U y G_V no son ambiguas, ya que las diferentes derivaciones generan diferentes sufijos de números enteros.

Teorema

No existe ningún algoritmo que determine si una GIC arbitraria es ambigua.

Demostración:

Una GIC es ambigua si contiene una cadena que puede ser generada por dos derivaciones a izquierdas diferentes. Como antes, partimos de un SCP arbitrario C y construimos G_U y G_V. Ahora las combinamos para obtener esta nueva gramática, cuyo símbolo inicial es S:

$$\begin{aligned} G: \quad & N = \{S, S_U, S_V\} \\ & \Sigma = \Sigma_C \cup \{1, 2, \ldots, n\} \\ & P = P_U \cup P_V \cup \{S \to S_U,\ S \to S_V\} \end{aligned}$$

Claramente, todas las derivaciones de G son a izquierdas, ya que toda forma sentencial contiene como máximo un símbolo no terminal. Una derivación de G consiste en la aplicación de una de las reescrituras de S, seguida de una derivación de G_U o de G_V. Y las gramáticas G_U y G_V no son ambiguas, ya que las diferentes derivaciones generan diferentes sufijos de números enteros.

Esto implica que G es ambigua si y sólo si $L(G_U) \cap L(G_V)$ es distinta del vacío. Pero, una vez más, esta condición es equivalente a la existencia de una solución para C. Así pues, el PCP es reducible al problema de la ambigüedad de una GIC, con lo que deducimos que este último problema es también irresoluble. □

Fin del capítulo

Fin del capítulo

"Resolubilidad"