Una mirada alternativa: protocolos de demostración interactivos

Considere el problema:

$$\text{SAT} \quad = \quad \{\varphi \mid \varphi \text{ es una fórmula proposicional satisfacible}\}$$

Tenemos un protocolo interactivo para demostrar que $\varphi \in \text{SAT}$

- El protocolo tiene dos participantes: **Verificador** y **Demostrador**
- **Verificador** trata de demostrar que $\varphi \in \text{SAT}$ haciendo preguntas a **Demostrador**
- **Demostrador** tiene poder de computación ilimitado
  - Puede tratar de engañar a **Verificador** dando información que indica que $\varphi \in \text{SAT}$ cuando φ es inconsistente

Un protocolo de demostración interactivo para SAT

Con entrada φ, el protocolo funciona de la siguiente forma:

1. **Verificador** pregunta a **Demostrador** por una valuación σ que satisfaga a φ
2. **Demostrador** responde con una valuación σ que satisfaga la condición anterior
3. **Verificador** chequea si $\sigma(\varphi) = 1$, y si es así acepta

Un protocolo de demostración interactivo para SAT

Con entrada φ, el protocolo funciona de la siguiente forma:

1. **Verificador** pregunta a **Demostrador** por una valuación σ que satisfaga a φ
2. **Demostrador** responde con una valuación σ que satisfaga la condición anterior
3. **Verificador** chequea si $\sigma(\varphi) = 1$, y si es así acepta

¿Puede engañar **Demostrador** a **Verificador** en este protocolo?

Un protocolo de demostración interactivo para SAT

Con entrada φ, el protocolo funciona de la siguiente forma:

1. **Verificador** pregunta a **Demostrador** por una valuación σ que satisfaga a φ
2. **Demostrador** responde con una valuación σ que satisfaga la condición anterior
3. **Verificador** chequea si $\sigma(\varphi) = 1$, y si es así acepta

¿Puede engañar **Demostrador** a **Verificador** en este protocolo?

- No por la verificación realizada en el paso 3

Una noción de protocolo más general

El protocolo mostrado en las transparencias anteriores puede ser extendido a cualquier lenguaje $L \in \text{NP}$

- ¿Cómo?

Una noción de protocolo más general

El protocolo mostrado en las transparencias anteriores puede ser extendido a cualquier lenguaje $L \in \text{NP}$

- ¿Cómo?

Es posible extender esta noción de protocolo en dos direcciones:

Una noción de protocolo más general

El protocolo mostrado en las transparencias anteriores puede ser extendido a cualquier lenguaje $L \in \text{NP}$

- ¿Cómo?

Es posible extender esta noción de protocolo en dos direcciones:

- Permitir varias rondas de pregunta y respuesta

Una noción de protocolo más general

El protocolo mostrado en las transparencias anteriores puede ser extendido a cualquier lenguaje $L \in \text{NP}$

- ¿Cómo?

Es posible extender esta noción de protocolo en dos direcciones:

- Permitir varias rondas de pregunta y respuesta
- Permitir que haya una probabilidad de error asociada a la respuesta final de **Verificador**

Una noción de protocolo más general

El protocolo mostrado en las transparencias anteriores puede ser extendido a cualquier lenguaje $L \in \text{NP}$

- ¿Cómo?

Es posible extender esta noción de protocolo en dos direcciones:

- Permitir varias rondas de pregunta y respuesta
- Permitir que haya una probabilidad de error asociada a la respuesta final de **Verificador**

Esto da origen a la clase de complejidad $\text{IP}[k]$

Una noción de protocolo más general

El protocolo mostrado en las transparencias anteriores puede ser extendido a cualquier lenguaje $L \in \text{NP}$

- ¿Cómo?

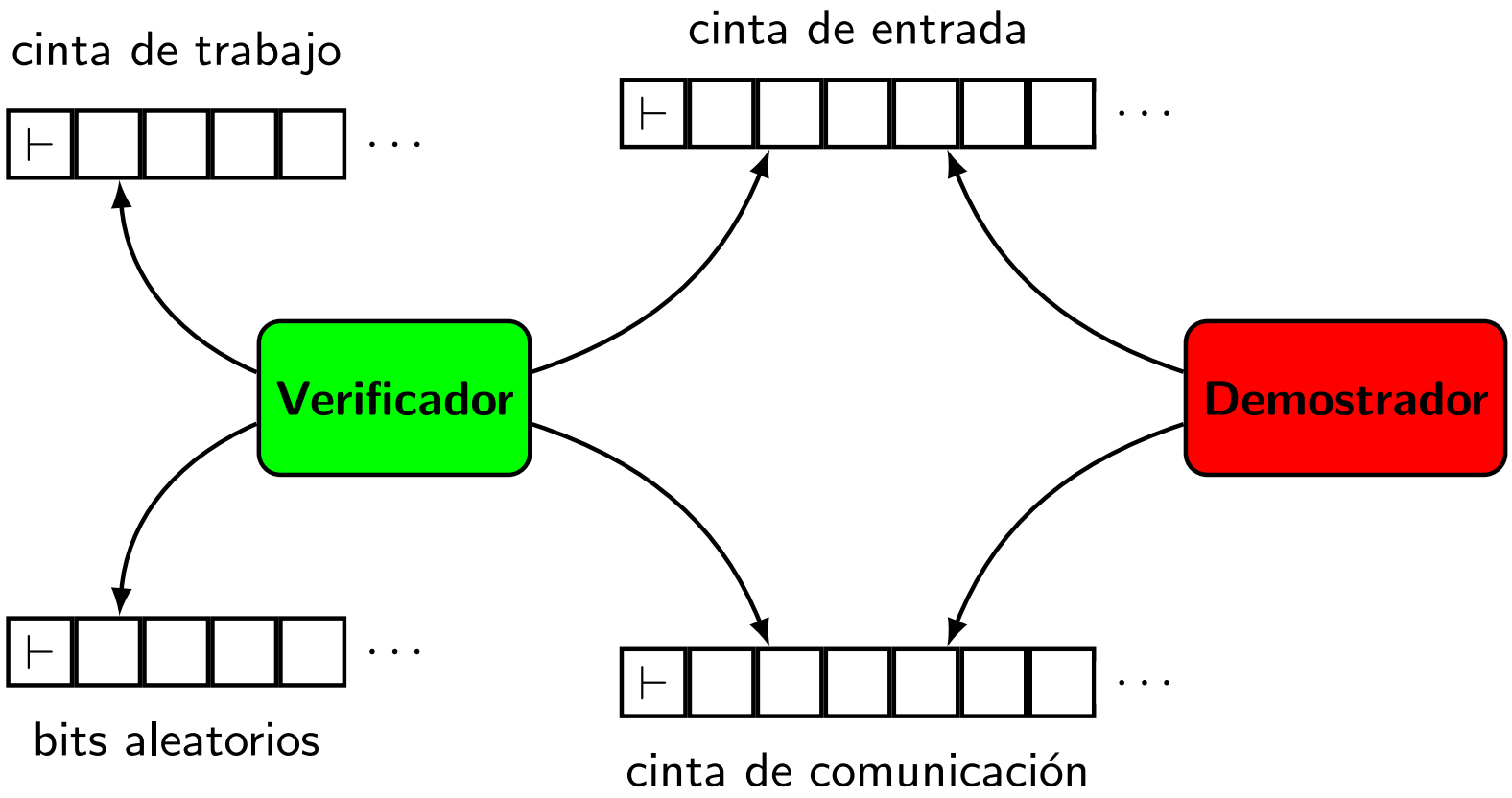
Es posible extender esta noción de protocolo en dos direcciones:

- Permitir varias rondas de pregunta y respuesta
- Permitir que haya una probabilidad de error asociada a la respuesta final de **Verificador**

Esto da origen a la clase de complejidad $\text{IP}[k]$

- k es usado para indicar el número de rondas de pregunta y respuesta

La clase IP[k]: definición del protocolo



La clase IP[k]: definición del protocolo

Verificador es una MT probabilística determinista que funciona en tiempo polinomial.

La clase IP[k]: definición del protocolo

Verificador es una MT probabilística determinista que funciona en tiempo polinomial.

Demostrador tiene poder de computación ilimitado

- Puede incluso decidir un problema indecidible

La clase IP[k]: definición del protocolo

Verificador es una MT probabilística determinista que funciona en tiempo polinomial.

Demostrador tiene poder de computación ilimitado

- Puede incluso decidir un problema indecidible

Verificador y **Demostrador** comparten dos cintas

- Una cinta de entrada donde se coloca el string w para el cual queremos verificar si está en un cierto lenguaje
  - Esta cinta es sólo de lectura
- Una cinta de comunicación donde **Verificador** puede colocar una consulta que es respondida por **Demostrador**
  - La respuesta de **Demostrador** a cada consulta de **Verificador** debe ser de tamaño polinomial en $|w|$

La clase IP[k]: definición del protocolo

Verificador además tiene dos cintas a las cuales **Demostrador** no tiene acceso

- Una cinta de trabajo que es de lectura y escritura
- Una cinta con bits aleatorios que es sólo de lectura y cuya cabeza sólo se mueve hacia la derecha

La clase IP[k]: definición del protocolo

Verificador además tiene dos cintas a las cuales **Demostrador** no tiene acceso

- Una cinta de trabajo que es de lectura y escritura
- Una cinta con bits aleatorios que es sólo de lectura y cuya cabeza sólo se mueve hacia la derecha

Inicialmente el protocolo entrega el control a **Verificador**

- Este control permanece en el poder de **Verificador**, hasta que **Verificador** realiza una consulta a **Demostrador** y le pasa el control
- Una vez que la consulta ha sido respondida **Demostrador** le devuelve el control a **Verificador**
- **Verificador** es quien decide si aceptar el string de entrada w

La clase IP[k]: definición del protocolo

El número de rondas efectuadas por el protocolo se define como el número de veces que el control cambia de dueño.

- Por ejemplo, decimos que tenemos 2 rondas si el control pasa de **Verificador** a **Demostrador** por una consulta, y luego de **Demostrador** a **Verificador** por la respuesta a la consulta

Verificador debe tener el control al momento de decidir si acepta el string de entrada.

- Como esta operación termina la ejecución del protocolo, el número de rondas debe ser par

La clase IP[k]: definición de la clase

Sea L un lenguaje sobre un alfabeto Σ

La clase IP[k]: definición de la clase

Sea L un lenguaje sobre un alfabeto Σ

Un protocolo (**Verificador**, **Demostrador**) es un protocolo interactivo de demostración para L si para todo $w \in \Sigma^*$:

La clase IP[k]: definición de la clase

Sea L un lenguaje sobre un alfabeto Σ

Un protocolo (**Verificador**, **Demostrador**) es un protocolo interactivo de demostración para L si para todo $w \in \Sigma^*$:

- Si $w \in L$, entonces:

$$\Pr((\textbf{Verificador}, \textbf{Demostrador}) \text{ acepte } w) \geq \frac{3}{4}$$

La clase IP[k]: definición de la clase

Sea L un lenguaje sobre un alfabeto Σ

Un protocolo (**Verificador**, **Demostrador**) es un protocolo interactivo de demostración para L si para todo $w \in \Sigma^*$:

- Si $w \in L$, entonces:

$$\Pr((\textbf{Verificador}, \textbf{Demostrador}) \text{ acepte } w) \geq \frac{3}{4}$$

- Si $w \notin L$, entonces para todo **Demostrador'**:

$$\Pr((\textbf{Verificador}, \textbf{Demostrador'}) \text{ acepte } w) \leq \frac{1}{4}$$

La clase IP[k]: definición de la clase

Finalmente decimos que $L \in \text{IP}[k]$ si existe un protocolo interactivo de demostración (**Verificador**, **Demostrador**) para L tal que el número de rondas del protocolo es k

La clase IP[k]: definición de la clase

Finalmente decimos que $L \in \text{IP}[k]$ si existe un protocolo interactivo de demostración (**Verificador**, **Demostrador**) para L tal que el número de rondas del protocolo es k

Ejercicio

Demuestre que SAT $\in$ IP[2] y GRAPH-ISO $\in$ IP[2]

¿Por qué nos interesa IP$[k]$?

No sabemos si $\overline{\text{GRAPH-ISO}} \in \text{NP}$

- Pero demostramos que $\overline{\text{GRAPH-ISO}} \in \text{AM}$

¿Por qué nos interesa IP[k]?

No sabemos si $\overline{\text{GRAPH-ISO}} \in \text{NP}$

- Pero demostramos que $\overline{\text{GRAPH-ISO}} \in \text{AM}$

También es posible demostrar que existe un protocolo para aceptar grafos no isomorfos:

Proposición

$\overline{\textit{GRAPH-ISO}} \in \textit{IP}[4]$

Una demostración de que $\overline{\text{GRAPH-ISO}} \in \text{IP}[4]$

Con entrada (G_1, G_2) el protocolo funciona de la siguiente forma:

Una demostración de que $\overline{\text{GRAPH-ISO}} \in \text{IP}[4]$

Con entrada (G_1, G_2) el protocolo funciona de la siguiente forma:

1. **Verificador** primero revisa si G_1 y G_2 tienen distinto número de nodos. Si es así acepta, si no va al paso 2

Una demostración de que $\overline{\text{GRAPH-ISO}} \in \text{IP}[4]$

Con entrada (G_1, G_2) el protocolo funciona de la siguiente forma:

1. **Verificador** primero revisa si G_1 y G_2 tienen distinto número de nodos. Si es así acepta, si no va al paso 2

2. Sea m el número de nodos de G_1 y G_2

Una demostración de que $\overline{\text{GRAPH-ISO}} \in \text{IP}[4]$

Con entrada (G_1, G_2) el protocolo funciona de la siguiente forma:

1. **Verificador** primero revisa si G_1 y G_2 tienen distinto número de nodos. Si es así acepta, si no va al paso 2

2. Sea m el número de nodos de G_1 y G_2

3. **Verificador** repite 2 veces los pasos 3.1 – 3.5

Una demostración de que $\overline{\text{GRAPH-ISO}} \in \text{IP}[4]$

Con entrada (G_1, G_2) el protocolo funciona de la siguiente forma:

1. **Verificador** primero revisa si G_1 y G_2 tienen distinto número de nodos. Si es así acepta, si no va al paso 2

2. Sea m el número de nodos de G_1 y G_2

3. **Verificador** repite 2 veces los pasos 3.1 – 3.5

    3.1 **Verificador** escoge con distribución uniforme un número $i \in \{1, 2\}$ y una permutación $f : \{1, \ldots, m\} \to \{1, \ldots, m\}$

Una demostración de que $\overline{\text{GRAPH-ISO}} \in \text{IP}[4]$

Con entrada (G_1, G_2) el protocolo funciona de la siguiente forma:

1. **Verificador** primero revisa si G_1 y G_2 tienen distinto número de nodos. Si es así acepta, si no va al paso 2

2. Sea m el número de nodos de G_1 y G_2

3. **Verificador** repite 2 veces los pasos 3.1 – 3.5

   3.1 **Verificador** escoge con distribución uniforme un número $i \in \{1, 2\}$ y una permutación $f : \{1, \ldots, m\} \to \{1, \ldots, m\}$

   3.2 Sea $H = f(G_i)$

Una demostración de que $\overline{\text{GRAPH-ISO}} \in \text{IP}[4]$

3.3 **Verificador** pone H en la cinta de comunicación y pregunta a **Demostrador** si es isomorfo a G_1

Una demostración de que $\overline{\text{GRAPH-ISO}} \in \text{IP}[4]$

3.3 **Verificador** pone H en la cinta de comunicación y pregunta a **Demostrador** si es isomorfo a G_1

3.4 **Demostrador** responde **sí** si H y G_1 son isomorfos, y **no** en caso contrario

Una demostración de que $\overline{\text{GRAPH-ISO}} \in \text{IP}[4]$

3.3 **Verificador** pone H en la cinta de comunicación y pregunta a **Demostrador** si es isomorfo a G_1

3.4 **Demostrador** responde **sí** si H y G_1 son isomorfos, y **no** en caso contrario

3.5 Si $i = 1$ y **Demostrador** respondió **no**, o si $i = 2$ y **Demostrador** respondió **sí**, entonces **Verificador** rechaza

Una demostración de que $\overline{\text{GRAPH-ISO}} \in \text{IP}[4]$

3.3 **Verificador** pone H en la cinta de comunicación y pregunta a **Demostrador** si es isomorfo a G_1

3.4 **Demostrador** responde **sí** si H y G_1 son isomorfos, y **no** en caso contrario

3.5 Si $i = 1$ y **Demostrador** respondió **no**, o si $i = 2$ y **Demostrador** respondió **sí**, entonces **Verificador** rechaza

4. **Verificador** acepta

Una demostración de que $\overline{\text{GRAPH-ISO}} \in \text{IP}[4]$

El protocolo tiene 4 rondas.

Además, tenemos que:

Una demostración de que $\overline{\text{GRAPH-ISO}} \in \text{IP}[4]$

El protocolo tiene 4 rondas.

Además, tenemos que:

- Si G_1 y G_2 no son isomorfos:

$$\Pr((\textbf{Verificador}, \textbf{Demostrador}) \text{ acepte } (G_1, G_2)) \quad = \quad 1$$

Una demostración de que $\overline{\text{GRAPH-ISO}} \in \text{IP}[4]$

El protocolo tiene 4 rondas.

Además, tenemos que:

- Si G_1 y G_2 no son isomorfos:

$$\Pr((\textbf{Verificador}, \textbf{Demostrador}) \text{ acepte } (G_1, G_2)) \quad = \quad 1$$

- Si G_1 y G_2 son isomorfos, entonces para todo **Demostrador'**:

$$\Pr((\textbf{Verificador}, \textbf{Demostrador'}) \text{ acepte } (G_1, G_2)) \quad = \quad \frac{1}{4}$$

Podemos disminuir el número de rondas para $\overline{\text{GRAPH-ISO}}$

Corolario

$\overline{\textit{GRAPH-ISO}} \in \textit{IP}[2]$

Podemos disminuir el número de rondas para $\overline{\text{GRAPH-ISO}}$

Corolario

$\overline{\textit{GRAPH-ISO}} \in \textit{IP}[2]$

Ejercicio

Demuestre el corolario.

¿Cuál es la relación de los protocolos interactivos de demostración con AM?

Definimos la clase AM[k] como IP[k] pero con una restricción adicional:

> Cada vez que **Verificador** envía una pregunta a **Demostrador** tiene que enviarle adicionalmente los bits aleatorios usados

¿Cuál es la relación de los protocolos interactivos de demostración con AM?

Definimos la clase AM[k] como IP[k] pero con una restricción adicional:

> Cada vez que **Verificador** envía una pregunta a **Demostrador** tiene que enviarle adicionalmente los bits aleatorios usados

Hablamos entonces de protocolos con bits aleatorios públicos

- Nótese que **Demostrador** conoce los bits aleatorios usados por **Verificador**, no conoce los que **Verificador** podría usar en el futuro

¿Cuál es la relación de los protocolos interactivos de demostración con AM?

Definimos la clase AM[k] como IP[k] pero con una restricción adicional:

> Cada vez que **Verificador** envía una pregunta a **Demostrador** tiene que enviarle adicionalmente los bits aleatorios usados

Hablamos entonces de protocolos con bits aleatorios públicos

- Nótese que **Demostrador** conoce los bits aleatorios usados por **Verificador**, no conoce los que **Verificador** podría usar en el futuro

Bajo esta definición no es claro que $\overline{\text{GRAPH-ISO}} \in \text{AM}[k]$

- El protocolo definido en las transparencias anteriores no funciones si los bit aleatorios usados son públicos

¿Cuál es la relación de los protocolos interactivos de demostración con AM?

Teorema

$\overline{\textit{GRAPH-ISO}} \in \textit{AM}[2]$

¿Cuál es la relación de los protocolos interactivos de demostración con AM?

Teorema

$\overline{\textit{GRAPH-ISO}} \in \textit{AM}[2]$

Ejercicio

Demuestre el teorema utilizando la MT probabilística no determinista definida en la demostración del Teorema de Schöning.

¿Cuál es la relación de los protocolos interactivos de demostración con AM?

Es posible demostrar que: $\mathsf{AM} = \bigcup_{\ell \geq 1} \mathsf{AM}[2\ell] = \mathsf{AM}[2]$

De hecho, la clase AM fue definida originalmente en términos de protocolos de demostración interactivos con bit aleatorios públicos.

¿Cuál es la relación de los protocolos interactivos de demostración con AM?

¿Tiene sentido que AM sea igual a $\bigcup_{\ell \geq 1} \text{AM}[2\ell]$?

- ¿Por qué es importante para este resultado que los bit aleatorios sean públicos?

¿Cuál es la relación de los protocolos interactivos de demostración con AM?

¿Tiene sentido que AM sea igual a $\bigcup_{\ell \geq 1} \text{AM}[2\ell]$?

- ¿Por qué es importante para este resultado que los bit aleatorios sean públicos?

La caracterización de AM en términos de protocolos interactivos de demostración nos da un punto de vista alternativo del resultado $\overline{\text{GRAPH-ISO}} \in \text{AM}$