La clase de complejidad de funciones #P

IIC3810

Marcelo Arenas y Luis Alberto Croquevielle

Calculando una función

Dado un alfabeto Σ, consideramos funciones $f : \Sigma^* \to \mathbb{N}$

Representamos cada número natural como un string sobre el alfabeto $\{0, \ldots, 9\}$

- Si una MTC calcula una función $f : \Sigma^* \to \mathbb{N}$, entonces para cada $w \in \Sigma^*$ suponemos que $f(w)$ es retornado en la cinta de salida como un string sobre el alfabeto $\{0, \ldots, 9\}$

Las funciones computables en tiempo polinomial

Definición

Una función $f : \Sigma^* \to \mathbb{N}$ está en la clase FP si y sólo si existe una MTC que funciona en tiempo polinomial y calcula f

Las funciones computables en tiempo polinomial

Definición

Una función $f : \Sigma^* \to \mathbb{N}$ está en la clase FP si y sólo si existe una MTC que funciona en tiempo polinomial y calcula f

¿Qué funciones son difíciles de computar?

Las funciones computables en tiempo polinomial

Definición

Una función $f : \Sigma^* \rightarrow \mathbb{N}$ está en la clase FP si y sólo si existe una MTC que funciona en tiempo polinomial y calcula f

¿Qué funciones son difíciles de computar?

- Consideramos funciones tales que si son computables en tiempo polinomial, entonces nos dan algoritmos polinomiales para resolver problemas NP-completos

La clase de funciones #P

Sea M una MT no determinista con alfabeto de entrada Σ

La clase de funciones $\#$P

Sea M una MT no determinista con alfabeto de entrada Σ

Supuesto

Si decimos que M funciona en tiempo polinomial, entonces suponemos que existe un polinomio $p(n)$ tal que para cada $w \in \Sigma^*$ y cada ejecución de M con entrada w, el número de pasos en la ejecución es menor o igual a $p(|w|)$

La clase de funciones #P

Sea M una MT no determinista con alfabeto de entrada Σ

Supuesto

Si decimos que M funciona en tiempo polinomial, entonces suponemos que existe un polinomio $p(n)$ tal que para cada $w \in \Sigma^*$ y cada ejecución de M con entrada w, el número de pasos en la ejecución es menor o igual a $p(|w|)$

- ¿Por qué podemos realizar este supuesto sin perdida de generalidad?

La clase de funciones #P

Sea M una MT no determinista con alfabeto de entrada Σ

Supuesto

Si decimos que M funciona en tiempo polinomial, entonces suponemos que existe un polinomio $p(n)$ tal que para cada $w \in \Sigma^*$ y cada ejecución de M con entrada w, el número de pasos en la ejecución es menor o igual a $p(|w|)$

- ¿Por qué podemos realizar este supuesto sin perdida de generalidad?

Dado $w \in \Sigma^*$, definimos $\text{accept}_M(w)$ como el número de ejecuciones de M con entrada w que se detienen en un estado final.

- En particular, $w \in L(M)$ si y sólo si $\text{accept}_M(w) > 0$

La clase de funciones #P

Definición

Una función $f : \Sigma^* \to \mathbb{N}$ *está en* #P *si y sólo si existe una MT no determinista* M *con alfabeto de entrada* Σ*, que funciona en tiempo polinomial y tal que para cada* $w \in \Sigma^*$: $f(w) = accept_M(w)$

La clase de funciones $\#$P

Definición

Una función $f : \Sigma^* \to \mathbb{N}$ está en $\#P$ si y sólo si existe una MT no determinista M con alfabeto de entrada Σ, que funciona en tiempo polinomial y tal que para cada $w \in \Sigma^*$: $f(w) = accept_M(w)$

Ejercicio

Sea $\#$SAT una función tal que, dada una fórmula proposicional φ, retorna el número de valuaciones que satisfacen φ. Demuestre que $\#\text{SAT} \in \#\text{P}$

Para pensar . . .

Ejercicios

1. Demuestre que si $f, g \in \#\mathrm{P}$, entonces $f + 1 \in \#\mathrm{P}$ y $f + g \in \#\mathrm{P}$
2. Demuestre que si $f, g \in \#\mathrm{P}$, entonces $f \cdot g \in \#\mathrm{P}$
3. Demuestre que $\mathrm{FP} \subseteq \#\mathrm{P}$

Una primera noción de reducción para funciones

Definición

Dadas funciones $f, g : \Sigma^* \to \mathbb{N}$, *decimos que* f *se puede reducir a* g *de forma parsimoniosa, denotado como* $f \leq_{par}^{p} g$, *si existe una función* $h : \Sigma^* \to \Sigma^*$ *tal que* h *puede ser calculada en tiempo polinomial y para todo* $w \in \Sigma^*$*:*

$$f(w) \quad = \quad g(h(w))$$

Una primera noción de reducción para funciones

Definición

Dadas funciones $f, g : \Sigma^* \to \mathbb{N}$, *decimos que* f *se puede reducir a* g *de forma parsimoniosa, denotado como* $f \leq^p_{par} g$, *si existe una función* $h : \Sigma^* \to \Sigma^*$ *tal que* h *puede ser calculada en tiempo polinomial y para todo* $w \in \Sigma^*$*:*

$$f(w) \quad = \quad g(h(w))$$

Ejercicio

Demuestre que si $f_1 \leq^p_{par} f_2$ y $f_2 \leq^p_{par} f_3$, entonces $f_1 \leq^p_{par} f_3$

Una noción de reducción más general para funciones

Al igual que para las máquinas de Turing con oráculos para problemas de decisión, podemos definir las máquinas de Turing con oráculos para funciones

- ¿Cómo debería funcionar la cinta de consulta en este caso?

Una noción de reducción más general para funciones

Al igual que para las máquinas de Turing con oráculos para problemas de decisión, podemos definir las máquinas de Turing con oráculos para funciones

- ¿Cómo debería funcionar la cinta de consulta en este caso?

Dada una función $f : \Sigma^* \to \mathbb{N}$, denotamos como M^f a una MT que tiene a f como oráculo

Una noción de reducción más general para funciones

Al igual que para las máquinas de Turing con oráculos para problemas de decisión, podemos definir las máquinas de Turing con oráculos para funciones

- ¿Cómo debería funcionar la cinta de consulta en este caso?

Dada una función $f : \Sigma^* \to \mathbb{N}$, denotamos como M^f a una MT que tiene a f como oráculo

- M^f también puede ser una MTC

Una noción de reducción más general para funciones

Dadas funciones $f, g : \Sigma^* \to \mathbb{N}$, decimos que $f \in \mathrm{FP}^g$ si existe una MTC M^g que funciona en tiempo polinomial y calcula f

Una noción de reducción más general para funciones

Dadas funciones $f, g : \Sigma^* \to \mathbb{N}$, decimos que $f \in \mathrm{FP}^g$ si existe una MTC M^g que funciona en tiempo polinomial y calcula f

Definición

Dadas funciones $f, g : \Sigma^* \to \mathbb{N}$, decimos que f se puede reducir a g en tiempo polinomial, denotado como $f \leq_T^p g$, si $f \in FP^g$

Una noción de reducción más general para funciones

Dadas funciones $f, g : \Sigma^* \to \mathbb{N}$, decimos que $f \in \mathrm{FP}^g$ si existe una MTC M^g que funciona en tiempo polinomial y calcula f

Definición

Dadas funciones $f, g : \Sigma^* \to \mathbb{N}$, decimos que f se puede reducir a g en tiempo polinomial, denotado como $f \leq_T^p g$, si $f \in FP^g$

Ejercicio

Demuestre que si $f_1 \leq_T^p f_2$ y $f_2 \leq_T^p f_3$, entonces $f_1 \leq_T^p f_3$

Una noción de completitud para #P

Definición

Una función $f : \Sigma^* \to \mathbb{N}$ *es #P-hard si para cada* $g \in \#P$ *se tiene que* $g \leq_T^p f$. *Sí adicionalmente* $f \in \#P$, *entonces* f *se dice #P-completo.*

Una noción de completitud para #P

Definición

Una función $f : \Sigma^* \to \mathbb{N}$ es #P-hard si para cada $g \in$ #P se tiene que $g \leq_T^p f$. Sí adicionalmente $f \in$ #P, entonces f se dice #P-completo.

Si $\leq_T^p$ es reemplazado por $\leq_{par}^p$ en la definición anterior, entonces decimos que f es #P-hard o #P-completo bajo reducciones parsimoniosas

Una noción de completitud para $\#$P

Definición

Una función $f : \Sigma^* \to \mathbb{N}$ es $\#$P-hard si para cada $g \in \#P$ se tiene que $g \leq_T^p f$. Sí adicionalmente $f \in \#P$, entonces f se dice $\#$P-completo.

Si $\leq_T^p$ es reemplazado por $\leq_{par}^p$ en la definición anterior, entonces decimos que f es $\#$P-hard o $\#$P-completo bajo reducciones parsimoniosas

- Si f es $\#$P-hard bajo reducciones parsimoniosas entonces también es $\#$P-hard, puesto que $f \leq_{par}^p g$ implica $f \leq_T^p g$

Un primer problema #P-completo

Teorema

#SAT es #P-completo bajo reducciones parsimoniosas.

Un primer problema #P-completo

Teorema

#SAT es #P-completo bajo reducciones parsimoniosas.

Ejercicio

Demuestre el teorema.

Un segundo problema #P-completo

Sea #CNF-SAT una función que, dada una formula proposicional φ en CNF, retorna el número de valuaciones que satisfacen φ

Un segundo problema #P-completo

Sea #CNF-SAT una función que, dada una formula proposicional φ en CNF, retorna el número de valuaciones que satisfacen φ

Teorema

#CNF-SAT es #P-completo bajo reducciones parsimoniosas.

Un segundo problema #P-completo

Sea #CNF-SAT una función que, dada una formula proposicional φ en CNF, retorna el número de valuaciones que satisfacen φ

Teorema

#CNF-SAT es #P-completo bajo reducciones parsimoniosas.

Ejercicio

Demuestre el teorema.

Un tercer problema #P-completo

Sea #DNF-SAT una función que, dada una formula proposicional φ en DNF, retorna el número de valuaciones que satisface φ

Un tercer problema #P-completo

Sea #DNF-SAT una función que, dada una formula proposicional φ en DNF, retorna el número de valuaciones que satisface φ

Teorema

#DNF-SAT es #P-completo

Un tercer problema #P-completo

Sea #DNF-SAT una función que, dada una formula proposicional φ en DNF, retorna el número de valuaciones que satisface φ

Teorema

#DNF-SAT es #P-completo

Demostración: vamos a demostrar que #CNF-SAT $\leq_T^p$ #DNF-SAT

Sea φ una fórmula proposicional en CNF, y sea ψ una fórmula proposicional en DNF que es equivalente a $\neg\varphi$

- ψ puede ser construida en tiempo polinomial a partir de φ

La demostración del teorema

Definimos $\mathrm{nv}(\cdot)$ como una función que retorna el número de variables de una fórmula proposicional φ

- Por ejemplo, $\mathrm{nv}((p \vee q) \wedge \neg r) = 3$ y $\mathrm{nv}((r \vee q) \wedge \neg r) = 2$

La función $\mathrm{nv}(\cdot)$ es computable en tiempo polinomial

La demostración del teorema

Definimos $\text{nv}(\cdot)$ como una función que retorna el número de variables de una fórmula proposicional φ

- Por ejemplo, $\text{nv}((p \vee q) \wedge \neg r) = 3$ y $\text{nv}((r \vee q) \wedge \neg r) = 2$

La función $\text{nv}(\cdot)$ es computable en tiempo polinomial

Tenemos que:

$$\#\text{CNF-SAT}(\varphi) \quad = \quad 2^{\text{nv}(\psi)} - \#\text{DNF-SAT}(\psi)$$

La demostración del teorema

Definimos $\text{nv}(\cdot)$ como una función que retorna el número de variables de una fórmula proposicional φ

- Por ejemplo, $\text{nv}((p \vee q) \wedge \neg r) = 3$ y $\text{nv}((r \vee q) \wedge \neg r) = 2$

La función $\text{nv}(\cdot)$ es computable en tiempo polinomial

Tenemos que:

$$\#\text{CNF-SAT}(\varphi) \quad = \quad 2^{\text{nv}(\psi)} - \#\text{DNF-SAT}(\psi)$$

Concluimos entonces que $\#\text{CNF-SAT} \in \text{FP}^{\#\text{DNF-SAT}}$

- Por lo tanto, $\#\text{CNF-SAT} \leq_T^p \#\text{DNF-SAT}$ $\square$

Sobre la completitud de #DNF-SAT

¿Se puede demostrar que #DNF-SAT es #P-completo bajo reducciones parsimoniosas?

Sobre la completitud de #DNF-SAT

¿Se puede demostrar que #DNF-SAT es #P-completo bajo reducciones parsimoniosas?

Para responder esta pregunta, defina para cada función $f : \Sigma^* \to \mathbb{N}$ el siguiente problema de decisión:

$$L_f \quad = \quad \{w \in \Sigma^* \mid f(w) > 0\}$$

Sobre la completitud de #DNF-SAT

¿Se puede demostrar que #DNF-SAT es #P-completo bajo reducciones parsimoniosas?

Para responder esta pregunta, defina para cada función $f : \Sigma^* \to \mathbb{N}$ el siguiente problema de decisión:

$$L_f \quad = \quad \{w \in \Sigma^* \mid f(w) > 0\}$$

Ejemplo

Tenemos que $L_{\#\text{SAT}} = \text{SAT}$

Sobre la completitud de #DNF-SAT

Proposition

Si $f \leq_{par}^{p} g$ *y* $L_g \in P$*, entonces* $L_f \in P$

Sobre la completitud de #DNF-SAT

Proposition

Si $f \leq^p_{par} g$ *y* $L_g \in P$*, entonces* $L_f \in P$

Ejercicio

Demuestre la proposición.

Sobre la completitud de #DNF-SAT

Proposition

Si $f \leq^p_{par} g$ y $L_g \in P$, entonces $L_f \in P$

Ejercicio

Demuestre la proposición.

Concluimos que #DNF-SAT no puede ser #P-completo bajo reducciones parsimoniosas, a menos que P = NP

- Puesto que si #SAT $\leq^p_{par}$ #DNF-SAT, entonces SAT $\in$ P

¿Cómo podemos calcular una función #P-completa?

Sea $f : \Sigma^* \to \mathbb{N}$ una función en #P

Si la función f es #P-completa entonces no esperamos tener un algoritmo polinomial para calcularla.

¿Cómo podemos calcular una función #P-completa?

Sea $f : \Sigma^* \to \mathbb{N}$ una función en #P

Si la función f es #P-completa entonces no esperamos tener un algoritmo polinomial para calcularla.

Pero el hecho de que f sea #P-completa no implica que el valor de f no pueda ser aproximado de manera eficiente.

¿Cómo podemos calcular una función #P-completa?

Sea $f : \Sigma^* \to \mathbb{N}$ una función en #P

Si la función f es #P-completa entonces no esperamos tener un algoritmo polinomial para calcularla.

Pero el hecho de que f sea #P-completa no implica que el valor de f no pueda ser aproximado de manera eficiente.

- Vamos a estudiar una noción de aproximación aleatorizada para funciones en #P

Aproximación aleatorizada de funciones

Sea $f : \Sigma^* \to \mathbb{N}$

Aproximación aleatorizada de funciones

Sea $f : \Sigma^* \to \mathbb{N}$

Definición

Un algoritmo aleatorizado $\mathcal{A} : \Sigma^* \times (0,1) \to \mathbb{N}$ *es un* **fully polynomial randomized approximation scheme (FPRAS)** *para* f *si existe un polinomio* $p(x,y)$ *tal que para cada* $w \in \Sigma^*$ *y* $\varepsilon \in (0,1)$*:*

1. *El número de pasos ejecutados por* $\mathcal{A}(w,\varepsilon)$ *es menor o igual a* $p(|w|, \frac{1}{\varepsilon})$
2. $\mathbf{Pr}(|\mathcal{A}(w,\varepsilon) - f(w)| \leq \varepsilon \cdot f(w)) \geq \frac{3}{4}$

Un enfoque general para construir un FPRAS

Dada una función $f : \Sigma^* \to \mathbb{N}$, $w \in \Sigma^*$ y $\varepsilon \in (0, 1)$, definimos una variable aleatoria X tal que $\mathbf{E}[X] = f(w)$

- Para tener una estimación de $f(w)$ nos basta muestrear la variable aleatoria X

Un enfoque general para construir un FPRAS

Dada una función $f : \Sigma^* \to \mathbb{N}$, $w \in \Sigma^*$ y $\varepsilon \in (0, 1)$, definimos una variable aleatoria X tal que $\mathbf{E}[X] = f(w)$

- Para tener una estimación de $f(w)$ nos basta muestrear la variable aleatoria X

Debe ser posible muestrear X en tiempo polinomial en $|w|$ y $\frac{1}{\varepsilon}$, y además debemos tener que $\mathbf{Pr}(|X - f(w)| \leq \varepsilon \cdot f(w)) \geq \frac{3}{4}$

- Debemos entonces acotar inferiormente $\mathbf{Pr}(|X - \mathbf{E}[X]| \leq \varepsilon \cdot \mathbf{E}[X])$

Un enfoque general para construir un FPRAS

Dada una función $f : \Sigma^* \to \mathbb{N}$, $w \in \Sigma^*$ y $\varepsilon \in (0, 1)$, definimos una variable aleatoria X tal que $\mathbf{E}[X] = f(w)$

- Para tener una estimación de $f(w)$ nos basta muestrear la variable aleatoria X

Debe ser posible muestrear X en tiempo polinomial en $|w|$ y $\frac{1}{\varepsilon}$, y además debemos tener que $\mathbf{Pr}(|X - f(w)| \leq \varepsilon \cdot f(w)) \geq \frac{3}{4}$

- Debemos entonces acotar inferiormente $\mathbf{Pr}(|X - \mathbf{E}[X]| \leq \varepsilon \cdot \mathbf{E}[X])$

Vamos a estudiar entonces algunas desigualdades útiles para obtener una cota inferior para $\mathbf{Pr}(|X - \mathbf{E}[X]| \leq \varepsilon \cdot \mathbf{E}[X])$

La desigualdad de Markov

Teorema

Sea X una variable aleatoria no negativa. Para cada $a \in \mathbb{R}^+$ se tiene que:

$$\mathbf{Pr}(X \geq a) \quad \leq \quad \frac{\mathbf{E}[X]}{a}$$

Una demostración de la desigualdad de Markov

Suponemos que el recorrido de X es un conjunto finito $\Omega \subseteq \mathbb{R}_0^+$:

$$
\begin{aligned}
\mathbf{E}[X] &= \sum_{r \in \Omega} r \cdot \mathbf{Pr}(X = r) \\
&= \left(\sum_{r \in \Omega \,:\, r < a} r \cdot \mathbf{Pr}(X = r) \right) + \left(\sum_{s \in \Omega \,:\, s \geq a} s \cdot \mathbf{Pr}(X = s) \right) \\
&\geq \sum_{s \in \Omega \,:\, s \geq a} s \cdot \mathbf{Pr}(X = s) \\
&\geq \sum_{s \in \Omega \,:\, s \geq a} a \cdot \mathbf{Pr}(X = s) \\
&= a \cdot \left(\sum_{s \in \Omega \,:\, s \geq a} \mathbf{Pr}(X = s) \right) \\
&= a \cdot \mathbf{Pr}(X \geq a)
\end{aligned}
$$

Una demostración de la desigualdad de Markov

Suponemos que el recorrido de X es un conjunto finito $\Omega \subseteq \mathbb{R}_0^+$:

$$
\begin{aligned}
\mathbf{E}[X] &= \sum_{r \in \Omega} r \cdot \mathbf{Pr}(X = r) \\
&= \left(\sum_{r \in \Omega \,:\, r < a} r \cdot \mathbf{Pr}(X = r) \right) + \left(\sum_{s \in \Omega \,:\, s \geq a} s \cdot \mathbf{Pr}(X = s) \right) \\
&\geq \sum_{s \in \Omega \,:\, s \geq a} s \cdot \mathbf{Pr}(X = s) \\
&\geq \sum_{s \in \Omega \,:\, s \geq a} a \cdot \mathbf{Pr}(X = s) \\
&= a \cdot \left(\sum_{s \in \Omega \,:\, s \geq a} \mathbf{Pr}(X = s) \right) \\
&= a \cdot \mathbf{Pr}(X \geq a)
\end{aligned}
$$

Concluimos que $\mathbf{Pr}(X \geq a) \leq \frac{\mathbf{E}[X]}{a}$ $\square$

La desigualdad de Chebyshev

Teorema

$$\mathbf{Pr}(|X - \mathbf{E}[X]| \geq a) \quad \leq \quad \frac{\mathbf{Var}[X]}{a^2}$$

La desigualdad de Chebyshev

Teorema

$$\mathbf{Pr}(|X - \mathbf{E}[X]| \geq a) \quad \leq \quad \frac{\mathbf{Var}[X]}{a^2}$$

Demostración. Utilizando la desigualdad de Markov obtenemos:

$$\begin{aligned}
\mathbf{Pr}(|X - \mathbf{E}[X]| \geq a) &= \mathbf{Pr}((X - \mathbf{E}[X])^2 \geq a^2) \\
&\leq \frac{\mathbf{E}[(X - \mathbf{E}[X])^2]}{a^2} \\
&= \frac{\mathbf{Var}[X]}{a^2}
\end{aligned}$$

$\square$

Utilizando la desigualdad de Chebyshev

A partir de la desigualdad de Chebyshev obtenemos:

$$\mathbf{Pr}(|X - \mathbf{E}[X]| \leq \varepsilon \cdot \mathbf{E}[X]) \quad \geq \quad 1 - \frac{\mathbf{Var}[X]}{\varepsilon^2 \cdot \mathbf{E}[X]^2}$$

Utilizando la desigualdad de Chebyshev

A partir de la desigualdad de Chebyshev obtenemos:

$$\mathbf{Pr}(|X - \mathbf{E}[X]| \leq \varepsilon \cdot \mathbf{E}[X]) \quad \geq \quad 1 - \frac{\mathbf{Var}[X]}{\varepsilon^2 \cdot \mathbf{E}[X]^2}$$

Es posible mejorar esta cota inferior considerando el promedio de $k \geq 2$ muestras de X obtenidas de manera independiente.

- Así reducimos el impacto de las muestras alejadas de $\mathbf{E}[X]$

Utilizando la desigualdad de Chebyshev

Por la desigualdad de Chebyshev:

$$\mathbf{Pr}\left(\left|\frac{1}{k}\cdot\sum_{i=1}^{k}X_i - \mathbf{E}\left[\frac{1}{k}\cdot\sum_{i=1}^{k}X_i\right]\right| \geq \varepsilon\cdot\mathbf{E}\left[\frac{1}{k}\cdot\sum_{i=1}^{k}X_i\right]\right) \quad \leq \quad \frac{\mathbf{Var}\left[\frac{1}{k}\cdot\sum_{i=1}^{k}X_i\right]}{\varepsilon^2\cdot\mathbf{E}\left[\frac{1}{k}\cdot\sum_{i=1}^{k}X_i\right]^2}$$

Además, tenemos que:

$$\begin{aligned}
\mathbf{E}\left[\frac{1}{k}\cdot\sum_{i=1}^{k}X_i\right] &= \frac{1}{k}\cdot\mathbf{E}\left[\sum_{i=1}^{k}X_i\right] \\
&= \frac{1}{k}\cdot\sum_{i=1}^{k}\mathbf{E}[X_i] \\
&= \frac{1}{k}\cdot\sum_{i=1}^{k}\mathbf{E}[X] \\
&= \mathbf{E}[X]
\end{aligned}$$

Utilizando la desigualdad de Chebyshev

$$
\begin{aligned}
\mathbf{Var}\left[\frac{1}{k} \cdot \sum_{i=1}^{k} X_i\right] &= \frac{1}{k^2} \cdot \mathbf{Var}\left[\sum_{i=1}^{k} X_i\right] \\
&= \frac{1}{k^2} \cdot \sum_{i=1}^{k} \mathbf{Var}[X_i] \\
&= \frac{1}{k^2} \cdot \sum_{i=1}^{k} \mathbf{Var}[X] \\
&= \frac{1}{k} \cdot \mathbf{Var}[X]
\end{aligned}
$$

De lo cual concluimos que:

$$
\mathbf{Pr}\left(\left|\frac{1}{k} \cdot \sum_{i=1}^{k} X_i - \mathbf{E}[X]\right| \geq \varepsilon \cdot \mathbf{E}[X]\right) \leq \frac{\mathbf{Var}[X]}{k \cdot \varepsilon^2 \cdot \mathbf{E}[X]^2}
$$

Utilizando la desigualdad de Chebyshev

Realizando k muestras independientes de la variable aleatoria X obtenemos entonces:

$$\mathbf{Pr}\left(\left|\frac{1}{k}\cdot\sum_{i=1}^{k} X_i - \mathbf{E}[X]\right| \leq \varepsilon \cdot \mathbf{E}[X]\right) \quad \geq \quad 1 - \frac{\mathbf{Var}[X]}{k\cdot\varepsilon^2\cdot\mathbf{E}[X]^2}$$

Utilizando la desigualdad de Chebyshev

Realizando k muestras independientes de la variable aleatoria X obtenemos entonces:

$$\mathbf{Pr}\left(\left|\frac{1}{k}\cdot\sum_{i=1}^{k} X_i - \mathbf{E}[X]\right| \leq \varepsilon \cdot \mathbf{E}[X]\right) \quad \geq \quad 1 - \frac{\mathbf{Var}[X]}{k \cdot \varepsilon^2 \cdot \mathbf{E}[X]^2}$$

Podemos entonces ajustar el valor de k para obtener un FPRAS.

- El valor de $\mathbf{Var}[X]$ debe ser polinomial en el tamaño de la entrada para obtener un FPRAS.

Construyendo un FPRAS para #DNF-SAT

Sea $\varphi = D_1 \vee \cdots \vee D_m$ una formula proposicional en DNF

- Cada D_i es una conjunción de literales
- φ menciona n variables proposicionales, vale decir, $\text{nv}(\varphi) = n$

Suponemos que cada D_i no tiene literales repetidos ni complementarios

- ¿Por qué podemos suponer esto?
- Sea ℓ_i el número de literales mencionados en D_i

Un FPRAS para #DNF-SAT: primer intento

Suponga que las asignaciones de valores de verdad σ para φ son escogidas con distribución uniforme:

$$\mathbf{Pr}(\sigma) \quad = \quad \frac{1}{2^n}$$

Y considere la siguiente variable aleatoria:

$$X(\sigma) \quad = \quad \begin{cases} 2^n & \sigma(\varphi) = 1 \\ 0 & \sigma(\varphi) = 0 \end{cases}$$

Un FPRAS para #DNF-SAT: primer intento

Tenemos que:

$$\begin{aligned}
\mathbf{E}[X] &= \sum_{\sigma} X(\sigma) \cdot \mathbf{Pr}(\sigma) \\
&= \sum_{\sigma \,:\, \sigma(\varphi)=1} 2^n \cdot \frac{1}{2^n} \\
&= \sum_{\sigma \,:\, \sigma(\varphi)=1} 1 \\
&= \#\text{DNF-SAT}(\varphi)
\end{aligned}$$

Un FPRAS para #DNF-SAT: primer intento

Tenemos que:

$$\begin{aligned}
\mathbf{E}[X] &= \sum_{\sigma} X(\sigma) \cdot \mathbf{Pr}(\sigma) \\
&= \sum_{\sigma \,:\, \sigma(\varphi)=1} 2^n \cdot \frac{1}{2^n} \\
&= \sum_{\sigma \,:\, \sigma(\varphi)=1} 1 \\
&= \#\text{DNF-SAT}(\varphi)
\end{aligned}$$

Tenemos entonces que $\mathbf{E}[X] = \#\text{DNF-SAT}(\varphi)$, el primer requisito para la variable aleatoria que queremos construir

- Además se puede muestrear X en tiempo polinomial en n
  - ¿Cómo se hace esto?

Un FPRAS para #DNF-SAT: primer intento

Nos falta calcular **Var**$[X]$ para saber si podemos obtener un FPRAS.

Un FPRAS para #DNF-SAT: primer intento

Nos falta calcular $\mathbf{Var}[X]$ para saber si podemos obtener un FPRAS.

Tenemos que:

$$\begin{aligned}
\mathbf{E}[X^2] &= \sum_{\sigma} X^2(\sigma) \cdot \mathbf{Pr}(\sigma) \\
&= \sum_{\sigma \,:\, \sigma(\varphi)=1} 2^{2 \cdot n} \cdot \frac{1}{2^n} \\
&= \sum_{\sigma \,:\, \sigma(\varphi)=1} 2^n \\
&= 2^n \cdot \#\text{DNF-SAT}(\varphi)
\end{aligned}$$

Un FPRAS para #DNF-SAT: primer intento

Nos falta calcular $\mathbf{Var}[X]$ para saber si podemos obtener un FPRAS.

Tenemos que:

$$\begin{aligned}
\mathbf{E}[X^2] &= \sum_{\sigma} X^2(\sigma) \cdot \mathbf{Pr}(\sigma) \\
&= \sum_{\sigma\,:\,\sigma(\varphi)=1} 2^{2\cdot n} \cdot \frac{1}{2^n} \\
&= \sum_{\sigma\,:\,\sigma(\varphi)=1} 2^n \\
&= 2^n \cdot \#\text{DNF-SAT}(\varphi)
\end{aligned}$$

Por lo tanto:

$$\begin{aligned}
\mathbf{Var}[X] &= \mathbf{E}[X^2] - \mathbf{E}[X]^2 \\
&= 2^n \cdot \#\text{DNF-SAT}(\varphi) - \#\text{DNF-SAT}(\varphi)^2 \\
&= (2^n - \#\text{DNF-SAT}(\varphi)) \cdot \#\text{DNF-SAT}(\varphi)
\end{aligned}$$

Un FPRAS para #DNF-SAT: primer intento

Realizando k muestras independientes de X obtenemos entonces:

$$\begin{aligned}
\mathbf{Pr}\left(\left|\frac{1}{k} \cdot \sum_{i=1}^{k} X_i - \mathbf{E}[X]\right| \geq \varepsilon \cdot \mathbf{E}[X]\right) &\leq \frac{\mathbf{Var}[X]}{k \cdot \varepsilon^2 \cdot \mathbf{E}[X]^2} \\
&= \frac{(2^n - \#\text{DNF-SAT}(\varphi)) \cdot \#\text{DNF-SAT}(\varphi)}{k \cdot \varepsilon^2 \cdot \#\text{DNF-SAT}(\varphi)^2} \\
&= \frac{2^n - \#\text{DNF-SAT}(\varphi)}{k \cdot \varepsilon^2 \cdot \#\text{DNF-SAT}(\varphi)}
\end{aligned}$$

Un FPRAS para #DNF-SAT: primer intento

Realizando k muestras independientes de X obtenemos entonces:

$$\mathbf{Pr}\left(\left|\frac{1}{k} \cdot \sum_{i=1}^{k} X_i - \mathbf{E}[X]\right| \geq \varepsilon \cdot \mathbf{E}[X]\right) \leq \frac{\mathbf{Var}[X]}{k \cdot \varepsilon^2 \cdot \mathbf{E}[X]^2}$$

$$= \frac{(2^n - \#\text{DNF-SAT}(\varphi)) \cdot \#\text{DNF-SAT}(\varphi)}{k \cdot \varepsilon^2 \cdot \#\text{DNF-SAT}(\varphi)^2}$$

$$= \frac{2^n - \#\text{DNF-SAT}(\varphi)}{k \cdot \varepsilon^2 \cdot \#\text{DNF-SAT}(\varphi)}$$

Por lo tanto, para obtener $\mathbf{Pr}(|\frac{1}{k} \cdot \sum_{i=1}^{k} X_i - \mathbf{E}[X]| \geq \varepsilon \cdot \mathbf{E}[X]) \leq \frac{1}{4}$ necesitamos imponer la siguiente restricción sobre k:

$$\frac{4 \cdot (2^n - \#\text{DNF-SAT}(\varphi))}{\varepsilon^2 \cdot \#\text{DNF-SAT}(\varphi)} \quad \leq \quad k$$

Un FPRAS para #DNF-SAT: primer intento

Tomando entonces:

$$k \;=\; \left\lceil \frac{4 \cdot (2^n - \#\text{DNF-SAT}(\varphi))}{\varepsilon^2 \cdot \#\text{DNF-SAT}(\varphi)} \right\rceil$$

obtenemos la cota superior deseada para la probabilidad de error.

Un FPRAS para #DNF-SAT: primer intento

Tomando entonces:

$$k \quad = \quad \left\lceil \frac{4 \cdot (2^n - \#\text{DNF-SAT}(\varphi))}{\varepsilon^2 \cdot \#\text{DNF-SAT}(\varphi)} \right\rceil$$

obtenemos la cota superior deseada para la probabilidad de error.

¡Pero el valor de k puede ser exponencial en n, lo cual significa que el algoritmo aleatorizado resultante es de tiempo exponencial!

- Por ejemplo, si #DNF-SAT(φ) es polinomial en n

Un FPRAS para #DNF-SAT: primer intento

Tomando entonces:

$$k \quad = \quad \left\lceil \frac{4 \cdot (2^n - \#\text{DNF-SAT}(\varphi))}{\varepsilon^2 \cdot \#\text{DNF-SAT}(\varphi)} \right\rceil$$

obtenemos la cota superior deseada para la probabilidad de error.

¡Pero el valor de k puede ser exponencial en n, lo cual significa que el algoritmo aleatorizado resultante es de tiempo exponencial!

- Por ejemplo, si $\#\text{DNF-SAT}(\varphi)$ es polinomial en n

Vamos a estudiar un segundo enfoque que sí da el resultado deseado.

- El primer enfoque no utilizaba el hecho de que φ está en DNF

Un FPRAS para #DNF-SAT: segundo intento

Recuerde que $\varphi = D_1 \vee \cdots \vee D_m$

- Donde cada D_i menciona ℓ_i literales, y no contiene literales repetidos ni complementarios

Para cada $i \in \{1, \ldots, m\}$ sea $S_i = |\{\sigma \mid \sigma(D_i) = 1\}|$, y sea $M = \displaystyle\sum_{i=1}^{m} S_i$

- Tenemos que $S_i = 2^{n-\ell_i}$

Además, para cada valuación σ sea $d(\sigma) = |\{i \in \{1, \ldots, m\} \mid \sigma(D_i) = 1\}|$

- Tenemos que $\sigma(\varphi) = 1$ si y sólo si $d(\sigma) \geq 1$

Un FPRAS para #DNF-SAT: segundo intento

Suponga que las asignaciones de valores de verdad σ para φ son escogidas de manera que:

$$\mathbf{Pr}(\sigma) \quad = \quad \frac{d(\sigma)}{M}$$

Nótese que esta probabilidad está bien definida puesto que $1 \leq M$ y $d(\sigma) \leq M$

Y considere la siguiente variable aleatoria:

$$Y(\sigma) \quad = \quad \begin{cases} \frac{M}{d(\sigma)} & \sigma(\varphi) = 1 \\ 0 & \sigma(\varphi) = 0 \end{cases}$$

Un FPRAS para #DNF-SAT: segundo intento

Tenemos que:

$$
\begin{aligned}
\mathbf{E}[Y] &= \sum_{\sigma} Y(\sigma) \cdot \mathbf{Pr}(\sigma) \\
&= \sum_{\sigma \,:\, \sigma(\varphi)=1} \frac{M}{d(\sigma)} \cdot \frac{d(\sigma)}{M} \\
&= \sum_{\sigma \,:\, \sigma(\varphi)=1} 1 \\
&= \#\text{DNF-SAT}(\varphi)
\end{aligned}
$$

Un FPRAS para #DNF-SAT: segundo intento

Tenemos que:

$$
\begin{aligned}
\mathbf{E}[Y] &= \sum_{\sigma} Y(\sigma) \cdot \mathbf{Pr}(\sigma) \\
&= \sum_{\sigma \,:\, \sigma(\varphi)=1} \frac{M}{d(\sigma)} \cdot \frac{d(\sigma)}{M} \\
&= \sum_{\sigma \,:\, \sigma(\varphi)=1} 1 \\
&= \#\text{DNF-SAT}(\varphi)
\end{aligned}
$$

Tenemos entonces que $\mathbf{E}[Y] = \#\text{DNF-SAT}(\varphi)$, el primer requisito que debemos cumplir

- ¿Cómo podemos muestrear Y en tiempo polinomial en el tamaño de φ?

Muestreando Y en tiempo polinomial

Para obtener un valor de Y realizamos los siguientes pasos:

1. Escoja $i \in \{1, \ldots, m\}$ con probabilidad $\dfrac{S_i}{M}$
2. Escoja σ tal que $\sigma(D_i) = 1$ con distribución uniforme
3. Retorne $Y(\sigma)$

Muestreando Y en tiempo polinomial

Para obtener un valor de Y realizamos los siguientes pasos:

1. Escoja $i \in \{1, \ldots, m\}$ con probabilidad $\dfrac{S_i}{M}$
2. Escoja σ tal que $\sigma(D_i) = 1$ con distribución uniforme
3. Retorne $Y(\sigma)$

¿Cómo se realiza el paso 1 en tiempo polinomial en el tamaño de φ?

Muestreando Y en tiempo polinomial

Para obtener un valor de Y realizamos los siguientes pasos:

1. Escoja $i \in \{1, \ldots, m\}$ con probabilidad $\dfrac{S_i}{M}$
2. Escoja σ tal que $\sigma(D_i) = 1$ con distribución uniforme
3. Retorne $Y(\sigma)$

¿Cómo se realiza el paso 1 en tiempo polinomial en el tamaño de φ?

- ¿Por qué no se puede hacer algo similar y escoger directamente σ con probabilidad $\frac{d(\sigma)}{M}$?

Muestreando Y en tiempo polinomial

Nos falta calcular la probabilidad de escoger σ en los pasos 1 y 2

Muestreando Y en tiempo polinomial

Nos falta calcular la probabilidad de escoger σ en los pasos 1 y 2

Tenemos que:

$$
\begin{aligned}
\mathbf{Pr}(\sigma) &= \sum_{i=1}^{m} \mathbf{Pr}(\sigma \mid D_i) \cdot \mathbf{Pr}(D_i) \\
&= \sum_{i \,:\, \sigma(D_i)=1} \mathbf{Pr}(\sigma \mid D_i) \cdot \mathbf{Pr}(D_i) \\
&= \sum_{i \,:\, \sigma(D_i)=1} \frac{1}{S_i} \cdot \frac{S_i}{M} \\
&= \sum_{i \,:\, \sigma(D_i)=1} \frac{1}{M} \\
&= \frac{1}{M} \cdot \sum_{i \,:\, \sigma(D_i)=1} 1 \\
&= \frac{d(\sigma)}{M}
\end{aligned}
$$

Un FPRAS para #DNF-SAT: segundo intento

Nos falta calcular **Var**$[Y]$ para saber si podemos obtener un FPRAS.

Un FPRAS para #DNF-SAT: segundo intento

Nos falta calcular $\mathbf{Var}[Y]$ para saber si podemos obtener un FPRAS.

- En lugar de hacer esto, vamos a acotar superiormente $\dfrac{\mathbf{Var}[Y]}{\mathbf{E}[Y]^2}$

Un FPRAS para #DNF-SAT: segundo intento

Nos falta calcular $\mathbf{Var}[Y]$ para saber si podemos obtener un FPRAS.

- En lugar de hacer esto, vamos a acotar superiormente $\dfrac{\mathbf{Var}[Y]}{\mathbf{E}[Y]^2}$

Si $\sigma(\varphi) = 1$ tenemos que $\dfrac{M}{m} \leq Y(\sigma) \leq M$

- Por lo tanto $\dfrac{M}{m} \leq \mathbf{E}[Y] \leq M$

Concluimos que:

$$\frac{M}{m} - M \leq Y - \mathbf{E}[Y] \leq M - \frac{M}{m}$$

Un FPRAS para #DNF-SAT: segundo intento

Tenemos entonces que:

$$|Y - \mathbf{E}[Y]| \quad \leq \quad \frac{M}{m} \cdot (m-1)$$

De lo cual se concluye que $(Y - \mathbf{E}[Y])^2 \leq (\frac{M}{m})^2 \cdot (m-1)^2$

- Por lo tanto $\mathbf{Var}[Y] \leq (\frac{M}{m})^2 \cdot (m-1)^2$

Un FPRAS para #DNF-SAT: segundo intento

Tenemos entonces que:

$$|Y - \mathbf{E}[Y]| \quad \leq \quad \frac{M}{m} \cdot (m-1)$$

De lo cual se concluye que $(Y - \mathbf{E}[Y])^2 \leq (\frac{M}{m})^2 \cdot (m-1)^2$

- Por lo tanto $\mathbf{Var}[Y] \leq (\frac{M}{m})^2 \cdot (m-1)^2$

Como $\frac{M}{m} \leq \mathbf{E}[Y]$, sabemos que $\frac{1}{\mathbf{E}[Y]} \leq \frac{m}{M}$

Concluimos que: $\dfrac{\mathbf{Var}[Y]}{\mathbf{E}[Y]^2} \;\leq\; \left(\dfrac{m}{M}\right)^2 \cdot \left(\dfrac{M}{m}\right)^2 \cdot (m-1)^2 \;=\; (m-1)^2$

Un FPRAS para #DNF-SAT: segundo intento

Realizando k muestras independientes de Y obtenemos entonces:

$$\begin{aligned}
\mathbf{Pr}\left(\left|\frac{1}{k}\cdot\sum_{i=1}^{k} Y_i - \mathbf{E}[Y]\right| \geq \varepsilon\cdot\mathbf{E}[Y]\right) &\leq \frac{\mathbf{Var}[Y]}{k\cdot\varepsilon^2\cdot\mathbf{E}[Y]^2} \\
&\leq \frac{(m-1)^2}{k\cdot\varepsilon^2}
\end{aligned}$$

Un FPRAS para #DNF-SAT: segundo intento

Realizando k muestras independientes de Y obtenemos entonces:

$$\begin{aligned}
\mathbf{Pr}\left(\left|\frac{1}{k} \cdot \sum_{i=1}^{k} Y_i - \mathbf{E}[Y]\right| \geq \varepsilon \cdot \mathbf{E}[Y]\right) &\leq \frac{\mathbf{Var}[Y]}{k \cdot \varepsilon^2 \cdot \mathbf{E}[Y]^2} \\
&\leq \frac{(m-1)^2}{k \cdot \varepsilon^2}
\end{aligned}$$

Por lo tanto, para obtener $\mathbf{Pr}(|\frac{1}{k} \cdot \sum_{i=1}^{k} Y_i - \mathbf{E}[Y]| \geq \varepsilon \cdot \mathbf{E}[Y]) \leq \frac{1}{4}$ imponemos la siguiente restricción sobre k:

$$\frac{4 \cdot (m-1)^2}{\varepsilon^2} \leq k$$

Un FPRAS para #DNF-SAT: segundo intento

Tomando entonces:

$$k = \left\lceil \frac{4 \cdot (m-1)^2}{\varepsilon^2} \right\rceil$$

obtenemos la cota superior deseada para la probabilidad de error.

Un FPRAS para #DNF-SAT: segundo intento

Tomando entonces:

$$k \quad = \quad \left\lceil \frac{4 \cdot (m-1)^2}{\varepsilon^2} \right\rceil$$

obtenemos la cota superior deseada para la probabilidad de error.

¡El algoritmo aleatorizado resultante funciona en tiempo polinomial en el tamaño de φ y $\frac{1}{\varepsilon}$!

Un FPRAS para #DNF-SAT: segundo intento

Tomando entonces:

$$k \quad = \quad \left\lceil \frac{4 \cdot (m-1)^2}{\varepsilon^2} \right\rceil$$

obtenemos la cota superior deseada para la probabilidad de error.

¡El algoritmo aleatorizado resultante funciona en tiempo polinomial en el tamaño de φ y $\frac{1}{\varepsilon}$!

- Debemos realizar k muestras independientes de Y, donde k depende polinomialmente de m y $\frac{1}{\varepsilon}$
- Cada una de las muestras puede ser obtenida en tiempo polinomial en el tamaño de φ

¿Toda función en #P admite un FPRAS?

¿Toda función en #P admite un FPRAS?

Proposition

Si una función f admite un FPRAS, entonces $L_f \in BPP$

¿Toda función en #P admite un FPRAS?

Proposition

Si una función f admite un FPRAS, entonces $L_f \in BPP$

Demostración: Suponga que $f : \Sigma^* \to \mathbb{N}$, y sea $\mathcal{A} : \Sigma^* \times (0,1) \to \mathbb{N}$ un FPRAS para f

Dado $w \in \Sigma^*$, considere el siguiente algoritmo aleatorizado $\mathcal{B}$ para verificar si $w \in L_f$:

1. Sea k el resultado de ejecutar $\mathcal{A}(w, \frac{1}{2})$
2. Si $k > 0$ entonces retorne **sí**, en caso contrario retorne **no**

Una demostración de la proposición

Necesitamos calcular la probabilidad de error del algoritmo $\mathcal{B}$

Suponga primero que $w \in L_f$, vale decir, $f(w) > 0$

Dado que $\mathcal{A}$ es un FPRAS para f, tenemos que:

$$\mathbf{Pr}(|\mathcal{A}(w, \frac{1}{2}) - f(w)| \leq \frac{1}{2} \cdot f(w)) \quad \geq \quad \frac{3}{4}$$

Lo cual es equivalente a:

$$\mathbf{Pr}(\frac{1}{2} \cdot f(w) \leq \mathcal{A}(w, \frac{1}{2}) \leq \frac{3}{2} \cdot f(w)) \quad \geq \quad \frac{3}{4}$$

Una demostración de la proposición

Concluimos entonces que:

$$\mathbf{Pr}(\frac{1}{2} \cdot f(w) \leq \mathcal{A}(w, \frac{1}{2})) \quad \geq \quad \frac{3}{4}$$

Dado que $f(w) > 0$, tenemos que $\frac{1}{2} \cdot f(w) > 0$, por lo que tenemos lo siguiente:

$$\mathbf{Pr}(0 < \mathcal{A}(w, \frac{1}{2})) \quad \geq \quad \frac{3}{4}$$

Vale decir, $\mathbf{Pr}(\mathcal{B}$ retorne sí$) \geq \frac{3}{4}$

Una demostración de la proposición

Suponga ahora que $w \notin L_f$, vale decir, $f(w) = 0$

Sabemos que:

$$\mathbf{Pr}(\frac{1}{2} \cdot f(w) \leq \mathcal{A}(w, \frac{1}{2}) \leq \frac{3}{2} \cdot f(w)) \quad \geq \quad \frac{3}{4}$$

Dado que $f(w) = 0$, concluimos entonces que $\mathbf{Pr}(\mathcal{A}(w, \frac{1}{2}) = 0) \geq \frac{3}{4}$

Vale decir, $\mathbf{Pr}(\mathcal{B}$ retorne no$) \geq \frac{3}{4}$

Una demostración de la proposición

Suponga ahora que $w \notin L_f$, vale decir, $f(w) = 0$

Sabemos que:

$$\mathbf{Pr}(\frac{1}{2} \cdot f(w) \leq \mathcal{A}(w, \frac{1}{2}) \leq \frac{3}{2} \cdot f(w)) \quad \geq \quad \frac{3}{4}$$

Dado que $f(w) = 0$, concluimos entonces que $\mathbf{Pr}(\mathcal{A}(w, \frac{1}{2}) = 0) \geq \frac{3}{4}$

Vale decir, $\mathbf{Pr}(\mathcal{B}$ retorne no$) \geq \frac{3}{4}$

De los dos casos concluimos que $\mathbf{Pr}(\mathcal{B}$ retorne un resultado incorrecto$) \leq \frac{1}{4}$ $\square$

¿Qué funciones en #P tienen FPRAS?

Por la proposición anterior no esperamos tener FPRAS para #SAT y #CNF-SAT

- A menos que NP $\subseteq$ BPP

¿Qué funciones en #P tienen FPRAS?

Por la proposición anterior no esperamos tener FPRAS para #SAT y #CNF-SAT

- A menos que NP $\subseteq$ BPP

¿Cada función $f \in$ #P tal que $L_f \in$ BPP admite un FPRAS?

¿Qué funciones en #P tienen FPRAS?

Por la proposición anterior no esperamos tener FPRAS para #SAT y #CNF-SAT

- A menos que NP $\subseteq$ BPP

¿Cada función $f \in$ #P tal que $L_f \in$ BPP admite un FPRAS?

- ¿Al menos esto se cumple para cada función $g \in$ #P tal que $L_g \in$ P?

¿Qué funciones en #P tienen FPRAS?

Por la proposición anterior no esperamos tener FPRAS para #SAT y #CNF-SAT

- A menos que NP $\subseteq$ BPP

¿Cada función $f \in$ #P tal que $L_f \in$ BPP admite un FPRAS?

- ¿Al menos esto se cumple para cada función $g \in$ #P tal que $L_g \in$ P?

Para responder esta pregunta necesitamos considerar otros problemas #P-completos

Conjuntos independientes de un grafo

Dado un grafo $G = (N, A)$, decimos que $S \subseteq N$ es un conjunto independiente si para cada $a, b \in S$ se tiene que $(a, b) \notin A$

- Vale decir, los nodos mencionados en S no están conectados entre sí en el grafo

Sea $\text{GIS} = \{(G, k) \mid G \text{ tiene un conjunto independiente } S \text{ con } |S| \geq k\}$

Ejercicio

Demuestre que GIS es NP-completo

Contando el número de conjuntos independientes

Sea #GIS una función que, dado un grafo G y un número natural k, retorna el número de conjuntos independientes S de G tales que $|S| \geq k$

Contando el número de conjuntos independientes

Sea #GIS una función que, dado un grafo G y un número natural k, retorna el número de conjuntos independientes S de G tales que $|S| \geq k$

Proposition

#GIS es #P-completo bajo reducciones parsimoniosas.

Una función de conteo más simple

Sea $\#$LIS una función que, dado un grafo G y un número natural k, retorna el número de conjuntos independientes S de G tales que $|S| \leq k$

Una función de conteo más simple

Sea #LIS una función que, dado un grafo G y un número natural k, retorna el número de conjuntos independientes S de G tales que $|S| \leq k$

Proposition

#LIS es #P-completo.

Una función de conteo más simple

Sea $\#$LIS una función que, dado un grafo G y un número natural k, retorna el número de conjuntos independientes S de G tales que $|S| \leq k$

Proposition

$\#LIS$ es $\#P$-completo.

Ejercicio

Muestre que $L_{\#\text{LIS}} \in \text{P}$

- Es posible entonces que $\#$LIS tenga un FPRAS

#LIS es #P-completo: demostración

Dado un grafo G con n nodos, y un número natural $k \geq 1$, se tiene que:

$$\#\mathrm{GIS}(G, k) \quad = \quad \#\mathrm{LIS}(G, n) - \#\mathrm{LIS}(G, k - 1)$$

#LIS es #P-completo: demostración

Dado un grafo G con n nodos, y un número natural $k \geq 1$, se tiene que:

$$\#\text{GIS}(G, k) \quad = \quad \#\text{LIS}(G, n) - \#\text{LIS}(G, k - 1)$$

Concluimos entonces que $\#\text{GIS} \in \text{FP}^{\#\text{LIS}}$, vale decir, $\#\text{GIS} \leq_T^p \#\text{LIS}$

- Tenemos que #LIS es #P-hard
- Dado que $\#\text{LIS} \in \#\text{P}$, tenemos entonces que #LIS es #P-completo

$\square$

¿Existe un FPRAS para #LIS?

¿Existe un FPRAS para #LIS?

Vamos a dar una respuesta negativa a esta pregunta basada en una suposición de complejidad.

- Este resultado nos va a ayudar a identificar otros problemas que no admiten FPRAS

¿Existe un FPRAS para #LIS?

Vamos a dar una respuesta negativa a esta pregunta basada en una suposición de complejidad.

- Este resultado nos va a ayudar a identificar otros problemas que no admiten FPRAS

Teorema

Si existe un FPRAS para #LIS, entonces $NP = RP$

#LIS no admite un FPRAS: demostración

Vamos a demostrar que si existe un FPRAS para #LIS, entonces GIS $\in$ BPP

- Dado que GIS es NP-completo, se concluye que NP $\subseteq$ BPP

#LIS no admite un FPRAS: demostración

Vamos a demostrar que si existe un FPRAS para #LIS, entonces GIS $\in$ BPP

- Dado que GIS es NP-completo, se concluye que NP $\subseteq$ BPP

Para terminar la demostración necesitamos el siguiente resultado:

Teorema

Si NP $\subseteq$ BPP, entonces NP $=$ RP

#LIS no admite un FPRAS: demostración

Vamos a demostrar que si existe un FPRAS para #LIS, entonces GIS $\in$ BPP

- Dado que GIS es NP-completo, se concluye que NP $\subseteq$ BPP

Para terminar la demostración necesitamos el siguiente resultado:

Teorema

Si NP $\subseteq$ BPP, entonces NP $=$ RP

Ejercicio

Demuestre el teorema.

Técnica de demostración: construyendo un grafo aumentado

Sea $G = (N, A)$ un grafo con $|N| = n$, y sea $k \geq 1$ un número natural.

- (G, k) es una entrada de GIS

Técnica de demostración: construyendo un grafo aumentado

Sea $G = (N, A)$ un grafo con $|N| = n$, y sea $k \geq 1$ un número natural.

- (G, k) es una entrada de GIS

Consideramos un número natural r, cuyo valor definiremos más tarde, y para cada $v \in N$ definimos un nuevo conjunto de r nodos:

$$C_v = \{v_1, v_2, \ldots, v_r\}$$

Técnica de demostración: construyendo un grafo aumentado

Usamos r para definir un nuevo grafo $G^r = (N^r, A^r)$:

$$\begin{aligned} N^r &= \bigcup_{v \in N} C_v \\ A^r &= \bigcup_{(u,v) \in A} \{(a, b) \mid a \in C_u \text{ y } b \in C_v\} \end{aligned}$$

G^r es un grafo con $r \cdot n$ nodos, donde dos nodos son adyacentes si y sólo si sus nodos de origen en G son adyacentes.

La noción de testigo

Definición

Dados conjuntos independientes S *de* G *y* T *de* G^r*, decimos que* T *es un testigo para* S *si:*

$$S = \{v \in N \mid C_v \cap T \neq \emptyset\}$$

La noción de testigo

Definición

Dados conjuntos independientes S de G y T de G^r, decimos que T es un testigo para S si:

$$S = \{v \in N \mid C_v \cap T \neq \emptyset\}$$

Un conjunto independiente de G^r es testigo de un único conjunto independiente de G

- Un conjunto independiente de G puede tener muchos testigos en G^r

La noción de testigo

Definición

Dados conjuntos independientes S de G y T de G^r, decimos que T es un testigo para S si:

$$S = \{v \in N \mid C_v \cap T \neq \emptyset\}$$

Un conjunto independiente de G^r es testigo de un único conjunto independiente de G

- Un conjunto independiente de G puede tener muchos testigos en G^r

Dado un conjunto independiente T de G^r, denotamos como S_T al único independiente de G que tiene como testigo a T

El número de testigos

Definimos los conjuntos:

$$\begin{aligned} I^r(G,k) &= \{T \mid T \text{ es un conjunto independiente de } G^r \text{ tal que } |S_T| = k\} \\ M^r(G,k) &= \{T \mid T \text{ es un conjunto independiente de } G^r \text{ tal que } |S_T| < k\} \end{aligned}$$

El número de testigos

Definimos los conjuntos:

$$\begin{aligned} I^r(G,k) &= \{T \mid T \text{ es un conjunto independiente de } G^r \text{ tal que } |S_T| = k\} \\ M^r(G,k) &= \{T \mid T \text{ es un conjunto independiente de } G^r \text{ tal que } |S_T| < k\} \end{aligned}$$

La conexión fundamental:

$$(G,k) \in \text{GIS si y sólo si } I^r(G,k) \neq \emptyset$$

El número de testigos

Definimos los conjuntos:

$$\begin{aligned} I^r(G,k) &= \{T \mid T \text{ es un conjunto independiente de } G^r \text{ tal que } |S_T| = k\} \\ M^r(G,k) &= \{T \mid T \text{ es un conjunto independiente de } G^r \text{ tal que } |S_T| < k\} \end{aligned}$$

La conexión fundamental:

$$(G,k) \in \text{GIS si y sólo si } I^r(G,k) \neq \emptyset$$

Suponiendo que existe un FPRAS para #LIS, vamos a desarrollar un algoritmo aleatorizado de tiempo polinomial para verificar si $I^r(G,k) \neq \emptyset$

- Obtenemos un algoritmo aleatorizado de tiempo polinomial para GIS

El número de testigos

Para $\ell \in \{0, 1, \ldots, k\}$ definimos t_ℓ como la cantidad de testigos en G^r para un conjunto independiente de G con ℓ nodos.

El número de testigos

Para $\ell \in \{0, 1, \dots, k\}$ definimos t_ℓ como la cantidad de testigos en G^r para un conjunto independiente de G con ℓ nodos.

Por la definición de G^r tenemos que $t_\ell = (2^r - 1)^\ell$

- ¿Por qué?

El número de testigos

Para $\ell \in \{0, 1, \ldots, k\}$ definimos t_ℓ como la cantidad de testigos en G^r para un conjunto independiente de G con ℓ nodos.

Por la definición de G^r tenemos que $t_\ell = (2^r - 1)^\ell$

- ¿Por qué?

Además, G tiene a lo más $\binom{n}{\ell}$ conjuntos independientes con ℓ nodos

El número de testigos

A partir de lo anterior concluimos que:

$$
\begin{aligned}
|M^r(G,k)| &\leq \sum_{\ell=0}^{k-1} \binom{n}{\ell} t_\ell \\
&\leq \sum_{\ell=0}^{k-1} \binom{n}{\ell} t_{k-1} \\
&\leq t_{k-1} \sum_{\ell=0}^{n} \binom{n}{\ell} \\
&= 2^n \cdot t_{k-1} \\
&= 2^n \cdot (2^r - 1)^{k-1}
\end{aligned}
$$

Técnica de demostración: aumentando las diferencias

Por otra parte, si $(G, k) \in \text{GIS}$ tenemos que:

$$|I^r(G, k)| \;\geq\; t_k \;=\; (2^r - 1)^k$$

Técnica de demostración: aumentando las diferencias

Por otra parte, si $(G, k) \in \text{GIS}$ tenemos que:

$$|I^r(G, k)| \;\geq\; t_k \;=\; (2^r - 1)^k$$

Concluimos que:

- Si $(G, k) \notin \text{GIS}$: $\#\text{LIS}(G^r, k \cdot r) \;=\; |M^r(G, k)| \;\leq\; 2^n(2^r - 1)^{k-1}$
- Si $(G, k) \in \text{GIS}$: $\#\text{LIS}(G^r, k \cdot r) \;\geq\; |I^r(G, k)| \;\geq\; (2^r - 1)^k$

Técnica de demostración: aumentando las diferencias

Por otra parte, si $(G, k) \in \mathrm{GIS}$ tenemos que:

$$|I^r(G,k)| \;\geq\; t_k \;=\; (2^r - 1)^k$$

Concluimos que:

- Si $(G, k) \notin \mathrm{GIS}$: $\#\mathrm{LIS}(G^r, k \cdot r) \;=\; |M^r(G,k)| \;\leq\; 2^n(2^r - 1)^{k-1}$
- Si $(G, k) \in \mathrm{GIS}$: $\#\mathrm{LIS}(G^r, k \cdot r) \;\geq\; |I^r(G,k)| \;\geq\; (2^r - 1)^k$

En particular, eligiendo $r = n + 3$:

- Si $(G, k) \notin \mathrm{GIS}$: $\#\mathrm{LIS}(G^{n+3}, k \cdot (n+3)) \;\leq\; 2^n(2^{n+3} - 1)^{k-1}$
- Si $(G, k) \in \mathrm{GIS}$: $\#\mathrm{LIS}(G^{n+3}, k \cdot (n+3)) \;\geq\; (2^{n+3} - 1)^k$

Un algoritmo aleatorizado para GIS

Suponemos que existe un FPRAS $\mathcal{A}$ para $\#$LIS, y mostramos como esto puede ser utilizado para obtener que GIS $\in$ BPP

Un algoritmo aleatorizado para GIS

Suponemos que existe un FPRAS $\mathcal{A}$ para $\#$LIS, y mostramos como esto puede ser utilizado para obtener que GIS $\in$ BPP

Definimos el siguiente algoritmo aleatorizado $\mathcal{B}$ para GIS:

1. Dada la entrada (G, k), donde G es un grafo con n nodos, si $k = 0$ retorne **sí**, en otro caso vaya al paso 2
2. Genere el grafo G^{n+3}
3. Sea s el resultado de ejecutar $\mathcal{A}(G^{n+3}, k \cdot (n+3), \frac{1}{2})$
4. Si $s > (1 + \frac{1}{2}) \cdot 2^n (2^{n+3} - 1)^{k-1}$, entonces retorne **sí**, en caso contrario retorne **no**

Un algoritmo aleatorizado para GIS

Suponemos que existe un FPRAS $\mathcal{A}$ para #LIS, y mostramos como esto puede ser utilizado para obtener que GIS $\in$ BPP

Definimos el siguiente algoritmo aleatorizado $\mathcal{B}$ para GIS:

1. Dada la entrada (G, k), donde G es un grafo con n nodos, si $k = 0$ retorne **sí**, en otro caso vaya al paso 2
2. Genere el grafo G^{n+3}
3. Sea s el resultado de ejecutar $\mathcal{A}(G^{n+3}, k \cdot (n+3), \frac{1}{2})$
4. Si $s > (1 + \frac{1}{2}) \cdot 2^n(2^{n+3} - 1)^{k-1}$, entonces retorne **sí**, en caso contrario retorne **no**

Nótese que $\mathcal{B}$ funciona en tiempo polinomial en el tamaño de la entrada (G, k)

- Dado que el tamaño de G^{n+3} es polinomial en el tamaño de G (G^{n+3} tiene $n \cdot (n+3)$ nodos), $\mathcal{A}$ es un FPRAS y $\varepsilon = \frac{1}{2}$ cuando se ejecuta $\mathcal{A}$

La probabilidad de error del algoritmo

Si $(G, k) \notin$ GIS obtenemos que $\mathbf{Pr}(\mathcal{B}(G, k)$ retorne **sí**) es igual a:

La probabilidad de error del algoritmo

Si $(G, k) \notin \text{GIS}$ obtenemos que $\mathbf{Pr}(\mathcal{B}(G, k)$ retorne **sí**) es igual a:

$$
\begin{aligned}
&\mathbf{Pr}(\mathcal{A}(G^{n+3}, k \cdot (n+3), \tfrac{1}{2}) > (1 + \tfrac{1}{2}) \cdot 2^n (2^{n+3} - 1)^{k-1}) \\
&\leq \mathbf{Pr}(\mathcal{A}(G^{n+3}, k \cdot (n+3), \tfrac{1}{2}) > (1 + \tfrac{1}{2}) \cdot \#\text{LIS}(G^{n+3}, k \cdot (n+3))) \\
&\leq \mathbf{Pr}(\mathcal{A}(G^{n+3}, k \cdot (n+3), \tfrac{1}{2}) > (1 + \tfrac{1}{2}) \cdot \#\text{LIS}(G^{n+3}, k \cdot (n+3)) \vee \\
&\qquad\qquad \mathcal{A}(G^{n+3}, k \cdot (n+3), \tfrac{1}{2}) < (1 - \tfrac{1}{2}) \cdot \#\text{LIS}(G^{n+3}, k \cdot (n+3))) \\
&= 1 - \mathbf{Pr}(\mathcal{A}(G^{n+3}, k \cdot (n+3), \tfrac{1}{2}) \leq (1 + \tfrac{1}{2}) \cdot \#\text{LIS}(G^{n+3}, k \cdot (n+3)) \wedge \\
&\qquad\qquad \mathcal{A}(G^{n+3}, k \cdot (n+3), \tfrac{1}{2}) \geq (1 - \tfrac{1}{2}) \cdot \#\text{LIS}(G^{n+3}, k \cdot (n+3))) \\
&= 1 - \mathbf{Pr}(|\mathcal{A}(G^{n+3}, k \cdot (n+3), \tfrac{1}{2}) - \#\text{LIS}(G^{n+3}, k \cdot (n+3))| \ \leq \\
&\qquad\qquad \tfrac{1}{2} \cdot \#\text{LIS}(G^{n+3}, k \cdot (n+3))) \\
&\leq 1 - \frac{3}{4} \ = \ \frac{1}{4}
\end{aligned}
$$

La probabilidad de error del algoritmo

Consideramos ahora el caso en que $(G, k) \in \text{GIS}$

La probabilidad de error del algoritmo

Consideramos ahora el caso en que $(G, k) \in \text{GIS}$

Primero debemos demostrar que:

$$(1 - \frac{1}{2}) \cdot \#\text{LIS}(G^{n+3}, k \cdot (n+3)) \quad > \quad (1 + \frac{1}{2}) \cdot 2^n(2^{n+3} - 1)^{k-1}$$

Dado que $\#\text{LIS}(G^{n+3}, k \cdot (n+3)) \geq (2^{n+3} - 1)^k$, basta demostrar que:

$$(1 - \frac{1}{2}) \cdot (2^{n+3} - 1)^k \quad > \quad (1 + \frac{1}{2}) \cdot 2^n(2^{n+3} - 1)^{k-1}$$

La probabilidad de error del algoritmo

Tenemos que:

$$
\begin{aligned}
& (1-\frac{1}{2})\cdot(2^{n+3}-1)^k \;>\; (1+\frac{1}{2})\cdot 2^n(2^{n+3}-1)^{k-1}\\
\Leftrightarrow\quad & \frac{1}{2}\cdot(2^{n+3}-1)^k \;>\; \frac{3}{2}\cdot 2^n(2^{n+3}-1)^{k-1}\\
\Leftrightarrow\quad & \frac{1}{2}\cdot(2^{n+3}-1) \;>\; \frac{3}{2}\cdot 2^n\\
\Leftrightarrow\quad & 2^{n+3}-1 \;>\; 3\cdot 2^n\\
\Leftrightarrow\quad & 2^n(2^3-3)-1 \;>\; 0\\
\Leftrightarrow\quad & 5\cdot 2^n-1 \;>\; 0
\end{aligned}
$$

Esta última condición es cierta para todo $n \geq 0$, de lo cual concluimos que $(1-\frac{1}{2})\cdot \#\mathrm{LIS}(G^{n+3}, k\cdot(n+3)) > (1+\frac{1}{2})\cdot 2^n(2^{n+3}-1)^{k-1}$

La probabilidad de error del algoritmo

Si $(G, k) \in \text{GIS}$ obtenemos que $\mathbf{Pr}(\mathcal{B}(G, k)$ retorne **no**) es igual a:

La probabilidad de error del algoritmo

Si $(G, k) \in \text{GIS}$ obtenemos que $\mathbf{Pr}(\mathcal{B}(G, k)$ retorne **no**$)$ es igual a:

$$
\begin{aligned}
&\mathbf{Pr}(\mathcal{A}(G^{n+3}, k \cdot (n+3), \tfrac{1}{2}) \leq (1 + \tfrac{1}{2}) \cdot 2^n (2^{n+3} - 1)^{k-1}) \\
&\leq \mathbf{Pr}(\mathcal{A}(G^{n+3}, k \cdot (n+3), \tfrac{1}{2}) < (1 - \tfrac{1}{2}) \cdot \#\text{LIS}(G^{n+3}, k \cdot (n+3))) \\
&\leq \mathbf{Pr}(\mathcal{A}(G^{n+3}, k \cdot (n+3), \tfrac{1}{2}) < (1 - \tfrac{1}{2}) \cdot \#\text{LIS}(G^{n+3}, k \cdot (n+3)) \vee \\
&\qquad\qquad \mathcal{A}(G^{n+3}, k \cdot (n+3), \tfrac{1}{2}) > (1 + \tfrac{1}{2}) \cdot \#\text{LIS}(G^{n+3}, k \cdot (n+3))) \\
&= 1 - \mathbf{Pr}(\mathcal{A}(G^{n+3}, k \cdot (n+3), \tfrac{1}{2}) \geq (1 - \tfrac{1}{2}) \cdot \#\text{LIS}(G^{n+3}, k \cdot (n+3)) \wedge \\
&\qquad\qquad \mathcal{A}(G^{n+3}, k \cdot (n+3), \tfrac{1}{2}) \leq (1 + \tfrac{1}{2}) \cdot \#\text{LIS}(G^{n+3}, k \cdot (n+3))) \\
&= 1 - \mathbf{Pr}(|\mathcal{A}(G^{n+3}, k \cdot (n+3), \tfrac{1}{2}) - \#\text{LIS}(G^{n+3}, k \cdot (n+3))| \leq \\
&\qquad\qquad \tfrac{1}{2} \cdot \#\text{LIS}(G^{n+3}, k \cdot (n+3))) \\
&\leq 1 - \frac{3}{4} = \frac{1}{4}
\end{aligned}
$$

#LIS no admite un FPRAS: conclusión

Tenemos que en ambos casos el error del algoritmo $\mathcal{B}$ está acotado superiormente por $\frac{1}{4}$

Concluimos entonces que GIS $\in$ BPP

- Por lo tanto NP $\subseteq$ BPP $\square$

Utilizando la técnica anterior en otros problemas

Sea $\#$IS una función que, dado un grafo G, retorna el número de conjuntos independientes de G

Utilizando la técnica anterior en otros problemas

Sea #IS una función que, dado un grafo G, retorna el número de conjuntos independientes de G

Teorema

Si existe un FPRAS para #IS, entonces NP $=$ RP

Utilizando la técnica anterior en otros problemas

Sea $\#$IS una función que, dado un grafo G, retorna el número de conjuntos independientes de G

Teorema

Si existe un FPRAS para $\#$IS, entonces $NP = RP$

Ejercicio

Demuestre el teorema utilizando la técnica usada para el caso de $\#$LIS

Una solución del ejercicio

Al igual que para el caso de #LIS, demostramos que si existe un FPRAS para #IS, entonces GIS $\in$ BPP

- Dado que GIS es NP-completo, se concluye que NP $\subseteq$ BPP
- Concluimos entonces que NP $=$ RP, dado que NP $\subseteq$ BPP implica que NP $=$ RP

Una solución del ejercicio

Al igual que para el caso de $\#$LIS, demostramos que si existe un FPRAS para $\#$IS, entonces GIS $\in$ BPP

- Dado que GIS es NP-completo, se concluye que NP $\subseteq$ BPP
- Concluimos entonces que NP $=$ RP, dado que NP $\subseteq$ BPP implica que NP $=$ RP

Dada una entrada (G, k) de GIS, definimos G^r, $I^r(G, k)$ y $M^r(G, k)$ como para el caso de $\#$LIS

Una solución del ejercicio

Al igual que para el caso de #LIS, demostramos que si existe un FPRAS para #IS, entonces GIS $\in$ BPP

- Dado que GIS es NP-completo, se concluye que NP $\subseteq$ BPP
- Concluimos entonces que NP $=$ RP, dado que NP $\subseteq$ BPP implica que NP $=$ RP

Dada una entrada (G, k) de GIS, definimos G^r, $I^r(G, k)$ y $M^r(G, k)$ como para el caso de #LIS

Tenemos entonces que:

- Si $(G, k) \not\in$ GIS: $\#\mathrm{IS}(G^r) \ = \ |M^r(G, k)| \ \leq \ 2^n(2^r - 1)^{k-1}$
- Si $(G, k) \in$ GIS: $\#\mathrm{IS}(G^r) \ \geq \ |I^r(G, k)| \ \geq \ (2^r - 1)^k$

Una solución del ejercicio

Concluimos entonces que para $r = n + 3$:

- Si $(G, k) \notin$ GIS: $\#\text{IS}(G^{n+3}) \;\leq\; 2^n(2^{n+3} - 1)^{k-1}$
- Si $(G, k) \in$ GIS: $\#\text{IS}(G^{n+3}) \;\geq\; (2^{n+3} - 1)^k$

Una solución del ejercicio

Concluimos entonces que para $r = n + 3$:

- Si $(G, k) \notin$ GIS: $\#\text{IS}(G^{n+3}) \leq 2^n(2^{n+3} - 1)^{k-1}$
- Si $(G, k) \in$ GIS: $\#\text{IS}(G^{n+3}) \geq (2^{n+3} - 1)^k$

Al igual que para el caso de #LIS, suponemos que existe un FPRAS para #IS, y mostramos como esto puede ser utilizado para obtener que GIS $\in$ BPP

- El algoritmo aleatorizado de tiempo polinomial para GIS es definido de la misma forma que para el caso de #LIS
- Las cotas mencionadas arriba son utilizadas para demostrar que el error de este algoritmo está acotado superiormente por $\frac{1}{4}$ $\square$

¿Cuál es la idea central en la técnica mostrada?

Sea (G, k) una entrada de GIS con $k \geq 1$

- Sabemos que $(G, k) \in$ GIS si y sólo si $|I^1(G, k)| \geq 1$

¿Cuál es la idea central en la técnica mostrada?

Sea (G, k) una entrada de GIS con $k \geq 1$

- Sabemos que $(G, k) \in \text{GIS}$ si y sólo si $|I^1(G, k)| \geq 1$

Dado que $|I^1(G, k)| = \#\text{LIS}(G, k) - \#\text{LIS}(G, k-1)$, podemos intentar utilizar un FPRAS para #LIS para determinar si $(G, k) \in \text{GIS}$

¿Cuál es la idea central en la técnica mostrada?

Sea (G, k) una entrada de GIS con $k \geq 1$

- Sabemos que $(G, k) \in$ GIS si y sólo si $|I^1(G, k)| \geq 1$

Dado que $|I^1(G, k)| = \#\text{LIS}(G, k) - \#\text{LIS}(G, k-1)$, podemos intentar utilizar un FPRAS para #LIS para determinar si $(G, k) \in$ GIS

- Generamos así un algoritmo aleatorizado de tiempo polinomial para GIS

¿Cuál es la idea central en la técnica mostrada?

Sea (G, k) una entrada de GIS con $k \geq 1$

- Sabemos que $(G, k) \in \text{GIS}$ si y sólo si $|I^1(G, k)| \geq 1$

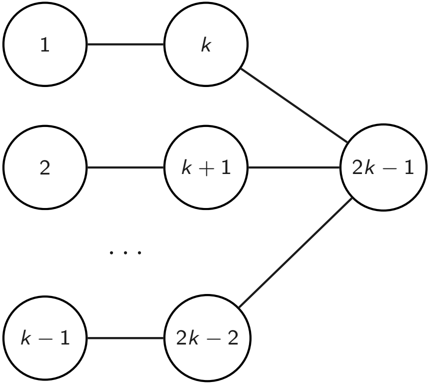
Dado que $|I^1(G, k)| = \#\text{LIS}(G, k) - \#\text{LIS}(G, k-1)$, podemos intentar utilizar un FPRAS para #LIS para determinar si $(G, k) \in \text{GIS}$

- Generamos así un algoritmo aleatorizado de tiempo polinomial para GIS

El éxito de este enfoque depende de qué tan grande sea el valor de $\#\text{LIS}(G, k) - \#\text{LIS}(G, k-1)$

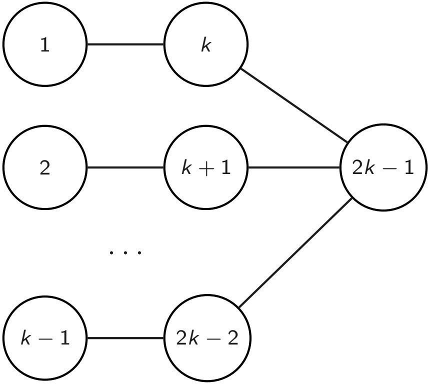
¿Qué tan grande es $\#\mathsf{LIS}(H, k) - \#\mathsf{LIS}(H, k-1)$?

Considere el siguiente grafo H:



¿Qué tan grande es $\#\text{LIS}(H, k) - \#\text{LIS}(H, k-1)$?

Considere el siguiente grafo H:



En este caso tenemos que $|I^1(H, k)| = \#\text{LIS}(H, k) - \#\text{LIS}(H, k-1) = 1$

- Además, $\#\text{LIS}(H, k-1) \geq 3^{k-1}$

Amplificando la diferencia

La diferencia entre $\#\text{LIS}(G, k)$ y $\#\text{LIS}(G, k-1)$ puede ser pequeña.

- La utilización de un FPRAS para $\#\text{LIS}$ no nos garantiza la existencia de un algoritmo aleatorizado de tiempo polinomial para GIS
  - No podemos acotar superiormente el error de algoritmo

Amplificando la diferencia

La diferencia entre $\#\mathrm{LIS}(G, k)$ y $\#\mathrm{LIS}(G, k-1)$ puede ser pequeña.

- La utilización de un FPRAS para #LIS no nos garantiza la existencia de un algoritmo aleatorizado de tiempo polinomial para GIS
  - No podemos acotar superiormente el error de algoritmo

Para solucionar este problema *amplificamos* $\#\mathrm{LIS}(G, k) - \#\mathrm{LIS}(G, k-1)$

Amplificando la diferencia

De manera más precisa, de la definición de G^r tenemos:

$$\#\text{LIS}(G, k) - \#\text{LIS}(G, k-1) > 0 \text{ si y sólo si}$$
$$\#\text{LIS}(G^r, k \cdot r) - \#\text{LIS}(G^r, (k-1) \cdot r) > 0$$

Amplificando la diferencia

De manera más precisa, de la definición de G^r tenemos:

$$\#\text{LIS}(G, k) - \#\text{LIS}(G, k-1) > 0 \text{ si y sólo si } \#\text{LIS}(G^r, k \cdot r) - \#\text{LIS}(G^r, (k-1) \cdot r) > 0$$

Así, usamos la diferencia más grande $\#\text{LIS}(G^r, k \cdot r) - \#\text{LIS}(G^r, (k-1) \cdot r)$ para verificar si $\#\text{LIS}(G, k) - \#\text{LIS}(G, k-1) > 0$

¿Cuánto amplificando la diferencia?

Suponiendo que G tiene n nodos y tomando $r = n + 3$ obtenemos:

$$
\begin{aligned}
\#\mathsf{LIS}(G^r, k \cdot r) - \#\mathsf{LIS}(G^r, (k-1) \cdot r) \quad & \geq \quad (2^r - 1)^k - 2^n \cdot (2^r - 1)^{k-1} \\
& = \quad (2^{n+3} - (2^n + 1)) \cdot (2^{n+3} - 1)^{k-1} \\
& \geq \quad (2^{n+3} - 2^{n+1}) \cdot (2^{n+3} - 1)^{k-1} \\
& = \quad 3 \cdot 2^{n+1} \cdot (2^{n+3} - 1)^{k-1}
\end{aligned}
$$

¿Cuánto amplificando la diferencia?

Suponiendo que G tiene n nodos y tomando $r = n + 3$ obtenemos:

$$
\begin{aligned}
\#\mathrm{LIS}(G^r, k \cdot r) - \#\mathrm{LIS}(G^r, (k-1) \cdot r) &\geq (2^r - 1)^k - 2^n \cdot (2^r - 1)^{k-1} \\
&= (2^{n+3} - (2^n + 1)) \cdot (2^{n+3} - 1)^{k-1} \\
&\geq (2^{n+3} - 2^{n+1}) \cdot (2^{n+3} - 1)^{k-1} \\
&= 3 \cdot 2^{n+1} \cdot (2^{n+3} - 1)^{k-1}
\end{aligned}
$$

¡Podemos entonces verificar si $\#\mathrm{LIS}(G, k) - \#\mathrm{LIS}(G, k-1) > 0$ considerando una brecha de tamaño exponencial en n!

Una tercera aplicación de la técnica

Sea $G = (N, A)$ un grafo (dirigido) sin loops.

Decimos que $(v_1, \ldots, v_n)$ es un ciclo simple en G si:

1. $n \geq 2$
2. $(v_i, v_{i+1}) \in A$ para cada $1 \leq i \leq n-1$, y $(v_n, v_1) \in A$
3. $v_i \neq v_j$ para cada $1 \leq i < j \leq n$

Una tercera aplicación de la técnica

Un ciclo simple $(v_1, \ldots, v_n)$ con n vértices puede ser visto como un conjunto con n arcos:

$$\{(v_1, v_2), \ldots (v_{n-1}, v_n), (v_n, v_1)\}$$

En este capítulo utilizamos esta representación de los ciclos simples, y decimos que el largo de este ciclo es n

Una tercera aplicación de la técnica

Un ciclo simple $(v_1, \ldots, v_n)$ con n vértices puede ser visto como un conjunto con n arcos:

$$\{(v_1, v_2), \ldots (v_{n-1}, v_n), (v_n, v_1)\}$$

En este capítulo utilizamos esta representación de los ciclos simples, y decimos que el largo de este ciclo es n

Nótese que dada la representación de ciclos simples tenemos que $(a, b, c) = (b, c, a) = (c, a, b)$, puesto que:

$$\{(a, b), (b, c), (c, a)\} = \{(b, c), (c, a), (a, b)\} = \{(c, a), (a, b), (b, c)\}$$

Una tercera aplicación de la técnica

Sea $\#$CicloSimple una función que, dado un grafo G, retorna el número de ciclos simples en G

Una tercera aplicación de la técnica

Sea #CicloSimple una función que, dado un grafo G, retorna el número de ciclos simples en G

Ejercicio

Muestre que $L_{\#\text{CicloSimple}} \in \mathrm{P}$

Una tercera aplicación de la técnica

Sea #CicloSimple una función que, dado un grafo G, retorna el número de ciclos simples en G

Ejercicio

Muestre que $L_{\#\text{CicloSimple}} \in \text{P}$

Teorema

Si existe un FPRAS para #CicloSimple, entonces $NP = RP$

#CicloSimple no admite un FPRAS: demostración

Sea HAM $= \{G \mid G$ tiene un ciclo hamiltoniano$\}$

Vamos a demostrar que si existe un FPRAS para #CicloSimple, entonces HAM $\in$ BPP

- Dado que HAM es NP-completo, se concluye que NP $\subseteq$ BPP
- Al igual que en los casos anteriores, concluimos que NP $=$ RP a partir de que NP $\subseteq$ BPP

#CicloSimple no admite un FPRAS: demostración

Sea HAM $= \{G \mid G$ tiene un ciclo hamiltoniano$\}$

Vamos a demostrar que si existe un FPRAS para #CicloSimple, entonces HAM $\in$ BPP

- Dado que HAM es NP-completo, se concluye que NP $\subseteq$ BPP
- Al igual que en los casos anteriores, concluimos que NP $=$ RP a partir de que NP $\subseteq$ BPP

Sea $G = (N, A)$ un grafo con $|N| = n$ y $n \geq 2$

- G es una entrada de HAM

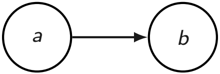
¿Cómo construimos el grafo aumentado?

A partir de un ciclo simple con ℓ nodos esperamos generar $2^{\ell \cdot r}$ ciclos simples en el grafo aumentado

¿Cómo construimos el grafo aumentado?

A partir de un ciclo simple con ℓ nodos esperamos generar $2^{\ell \cdot r}$ ciclos simples en el grafo aumentado

Reemplazamos cada arco

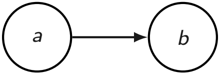


por

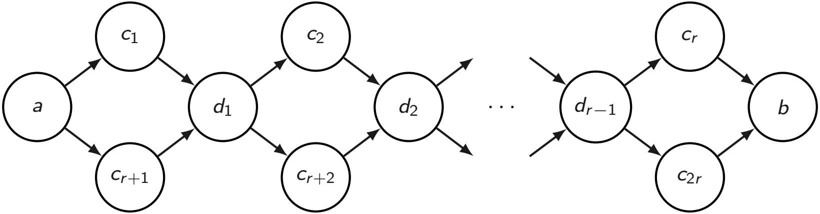
¿Cómo construimos el grafo aumentado?

A partir de un ciclo simple con ℓ nodos esperamos generar $2^{\ell \cdot r}$ ciclos simples en el grafo aumentado

Reemplazamos cada arco



por



donde $c_1, \ldots, c_r, c_{r+1}, \ldots, c_{2r}, d_1, \ldots, d_{r-1}$ son nuevos nodos

El grafo aumentado

Sea G^r el grafo definido reemplazando cada arco como fue mostrado en la transparencia anterior.

Ejercicio

Defina formalmente el grafo G^r

El grafo aumentado

Sea G^r el grafo definido reemplazando cada arco como fue mostrado en la transparencia anterior.

Ejercicio

Defina formalmente el grafo G^r

Dado que G no tiene loops puede tener a lo más $n \cdot (n-1)$ arcos

- Por lo tanto, G^r tiene a lo más $n + n \cdot (n-1) \cdot (3 \cdot r - 1)$ nodos, y es de tamaño polinomial en el tamaño de G

La noción de testigo

Cada ciclo simple de G de largo $\ell \geq 2$ corresponde con $2^{\ell \cdot r}$ ciclos de G^r de largo $2 \cdot \ell \cdot r$

- Cada uno de estos ciclos de largo $2 \cdot \ell \cdot r$ en G^r es considerado como un testigo del ciclo de largo ℓ en G

Cada ciclo simple de G^r es testigo de un único ciclo simple de G

La noción de testigo

Cada ciclo simple de G de largo $\ell \geq 2$ corresponde con $2^{\ell \cdot r}$ ciclos de G^r de largo $2 \cdot \ell \cdot r$

- Cada uno de estos ciclos de largo $2 \cdot \ell \cdot r$ en G^r es considerado como un testigo del ciclo de largo ℓ en G

Cada ciclo simple de G^r es testigo de un único ciclo simple de G

Ejercicio

Formalice la noción de testigo.

La noción de testigo

La conexión fundamental entre los grafos:

$G \in \mathrm{HAM}$ si y sólo si G^r tiene un cliclo simple de largo $2 \cdot r \cdot n$

La noción de testigo

La conexión fundamental entre los grafos:

$G \in \text{HAM}$ si y sólo si G^r tiene un ciclo simple de largo $2 \cdot r \cdot n$

Utilizando esta conexión y eligiendo un valor adecuado para r vamos a construir un algoritmo aleatorizado de tiempo polinomial para HAM

- Teniendo como hipótesis la existencia de un FPRAS para #CicloSimple

El número de testigos

Definimos los conjuntos:

$$
\begin{aligned}
I^r(G) &= \{C \mid C \text{ es un ciclo simple en } G^r \text{ de largo } 2 \cdot r \cdot n\} \\
M^r(G) &= \{C \mid C \text{ es un ciclo simple en } G^r \text{ de largo menor que } 2 \cdot r \cdot n\}
\end{aligned}
$$

Tenemos que $G \in \text{HAM}$ si y sólo si $I^r(G) \neq \emptyset$

- Suponiendo la existencia de un FPRAS para #CicloSimple, vamos a desarrollar un algoritmo aleatorizado de tiempo polinomial para verificar si $I^r(G) \neq \emptyset$

El número de testigos

Para $\ell \in \{2, 3, \ldots, n\}$ definimos t_ℓ como la cantidad de testigos en G^r para un ciclo simple en G de largo ℓ

El número de testigos

Para $\ell \in \{2, 3, \ldots, n\}$ definimos t_ℓ como la cantidad de testigos en G^r para un ciclo simple en G de largo ℓ

Por la definición de G^r tenemos que $t_\ell = 2^{\ell \cdot r}$

El número de testigos

Para $\ell \in \{2, 3, \ldots, n\}$ definimos t_ℓ como la cantidad de testigos en G^r para un ciclo simple en G de largo ℓ

Por la definición de G^r tenemos que $t_\ell = 2^{\ell \cdot r}$

Además, el número de ciclos simples en G de largo ℓ está acotado por:

$$\binom{n \cdot (n-1)}{\ell}$$

El número de testigos

A partir de lo anterior concluimos que:

$$
\begin{aligned}
|M^r(G)| &\leq \sum_{\ell=2}^{n-1} \binom{n\cdot(n-1)}{\ell} t_\ell \\
&\leq \sum_{\ell=2}^{n-1} \binom{n\cdot(n-1)}{\ell} t_{n-1} \\
&\leq t_{n-1} \sum_{\ell=0}^{n\cdot(n-1)} \binom{n\cdot(n-1)}{\ell} \\
&= 2^{n\cdot(n-1)} \cdot t_{n-1} \\
&= 2^{n\cdot(n-1)} \cdot 2^{(n-1)\cdot r} \\
&= 2^{(n-1)\cdot(n+r)}
\end{aligned}
$$

Idea fundamental: aumentando las diferencias

Por otra parte, si $G \in \text{HAM}$ tenemos que:

$$|I^r(G)| \;\geq\; t_n \;=\; 2^{n \cdot r}$$

Idea fundamental: aumentando las diferencias

Por otra parte, si $G \in \text{HAM}$ tenemos que:

$$|I^r(G)| \;\geq\; t_n \;=\; 2^{n \cdot r}$$

Concluimos que:

- Si $G \notin \text{HAM}$: $\#\text{CicloSimple}(G^r) \;=\; |M^r(G)| \;\leq\; 2^{(n-1)\cdot(n+r)}$
- Si $G \in \text{HAM}$: $\#\text{CicloSimple}(G^r) \;\geq\; |I^r(G)| \;\geq\; 2^{n \cdot r}$

Idea fundamental: aumentando las diferencias

Por otra parte, si $G \in \text{HAM}$ tenemos que:

$$|I^r(G)| \geq t_n = 2^{n \cdot r}$$

Concluimos que:

- Si $G \notin \text{HAM}$: $\#\text{CicloSimple}(G^r) = |M^r(G)| \leq 2^{(n-1)\cdot(n+r)}$
- Si $G \in \text{HAM}$: $\#\text{CicloSimple}(G^r) \geq |I^r(G)| \geq 2^{n \cdot r}$

En las siguientes transparencias vamos a obtener un valor de r que nos permita tener un algoritmo aleatorizado de tiempo polinomial para HAM

Un algoritmo aleatorizado para HAM

Suponemos que existe un FPRAS $\mathcal{A}$ para #CicloSimple, y mostramos como esto puede ser utilizado para obtener que HAM $\in$ BPP

Un algoritmo aleatorizado para HAM

Suponemos que existe un FPRAS $\mathcal{A}$ para #CicloSimple, y mostramos como esto puede ser utilizado para obtener que HAM $\in$ BPP

Definimos el siguiente algoritmo aleatorizado $\mathcal{B}$ para HAM:

1. Dada la entrada G, donde G es un grafo con n nodos, si $n \leq 1$ retorne **no**, en otro caso vaya al paso 2
2. Genere el grafo G^r
3. Sea s el resultado de ejecutar $\mathcal{A}(G^r, \frac{1}{2})$
4. Si $s > (1 + \frac{1}{2}) \cdot 2^{(n-1)\cdot(n+r)}$, entonces retorne **sí**, en caso contrario retorne **no**

Un algoritmo aleatorizado para HAM

Suponemos que existe un FPRAS $\mathcal{A}$ para #CicloSimple, y mostramos como esto puede ser utilizado para obtener que HAM $\in$ BPP

Definimos el siguiente algoritmo aleatorizado $\mathcal{B}$ para HAM:

1. Dada la entrada G, donde G es un grafo con n nodos, si $n \leq 1$ retorne **no**, en otro caso vaya al paso 2
2. Genere el grafo G^r
3. Sea s el resultado de ejecutar $\mathcal{A}(G^r, \frac{1}{2})$
4. Si $s > (1 + \frac{1}{2}) \cdot 2^{(n-1)\cdot(n+r)}$, entonces retorne **sí**, en caso contrario retorne **no**

Nótese que $\mathcal{B}$ funciona en tiempo polinomial en el tamaño de la entrada G si el valor de r es polinomial en n

- Recuerde que tenemos que determinar el valor de r

La probabilidad de error del algoritmo

Si $G \notin$ HAM obtenemos que $\mathbf{Pr}(\mathcal{B}(G)$ retorne **sí**$)$ es igual a:

La probabilidad de error del algoritmo

Si $G \notin$ HAM obtenemos que $\mathbf{Pr}(\mathcal{B}(G)$ retorne **sí**) es igual a:

$$
\begin{aligned}
&\mathbf{Pr}(\mathcal{A}(G^r, \tfrac{1}{2}) > (1 + \tfrac{1}{2}) \cdot 2^{(n-1)\cdot(n+r)}) \\
&\quad \leq \mathbf{Pr}(\mathcal{A}(G^r, \tfrac{1}{2}) > (1 + \tfrac{1}{2}) \cdot \#\text{CicloSimple}(G^r)) \\
&\quad \leq \mathbf{Pr}(\mathcal{A}(G^r, \tfrac{1}{2}) > (1 + \tfrac{1}{2}) \cdot \#\text{CicloSimple}(G^r)) \vee \\
&\qquad\qquad\qquad \mathcal{A}(G^r, \tfrac{1}{2}) < (1 - \tfrac{1}{2}) \cdot \#\text{CicloSimple}(G^r)) \\
&\quad = 1 - \mathbf{Pr}(\mathcal{A}(G^r, \tfrac{1}{2}) \leq (1 + \tfrac{1}{2}) \cdot \#\text{CicloSimple}(G^r) \wedge \\
&\qquad\qquad\qquad \mathcal{A}(G^r, \tfrac{1}{2}) \geq (1 - \tfrac{1}{2}) \cdot \#\text{CicloSimple}(G^r)) \\
&\quad = 1 - \mathbf{Pr}(|\mathcal{A}(G^r, \tfrac{1}{2}) - \#\text{CicloSimple}(G^r)| \leq \tfrac{1}{2} \cdot \#\text{CicloSimple}(G^r)) \\
&\quad \leq 1 - \frac{3}{4} \;=\; \frac{1}{4}
\end{aligned}
$$

La probabilidad de error del algoritmo

Consideramos ahora el caso en que $G \in \text{HAM}$

La probabilidad de error del algoritmo

Consideramos ahora el caso en que $G \in \text{HAM}$

Primero debemos encontrar un valor de r tal que:

$$(1 - \frac{1}{2}) \cdot \#\text{CicloSimple}(G^r) \quad > \quad (1 + \frac{1}{2}) \cdot 2^{(n-1)\cdot(n+r)}$$

Dado que $\#\text{CicloSimple}(G^r) \geq 2^{n \cdot r}$, basta entonces encontrar un valor de r tal que:

$$(1 - \frac{1}{2}) \cdot 2^{n \cdot r} \quad > \quad (1 + \frac{1}{2}) \cdot 2^{(n-1)\cdot(n+r)}$$

La probabilidad de error del algoritmo

Tenemos que:

$$
\begin{aligned}
& (1 - \frac{1}{2}) \cdot 2^{n \cdot r} \; > \; (1 + \frac{1}{2}) \cdot 2^{(n-1)\cdot(n+r)} \\
\Leftrightarrow \quad & \frac{1}{2} \cdot 2^{n \cdot r} \; > \; \frac{3}{2} \cdot 2^{(n-1)\cdot(n+r)} \\
\Leftrightarrow \quad & 2^{n \cdot r} \; > \; 3 \cdot 2^{(n-1)\cdot(n+r)} \\
\Leftrightarrow \quad & 2^{n \cdot r} \; > \; 2^{(n-1)\cdot(n+r) + \log_2(3)} \\
\Leftrightarrow \quad & n \cdot r \; > \; n \cdot (n-1) + (n-1) \cdot r + \log_2(3) \\
\Leftrightarrow \quad & r \; > \; n \cdot (n-1) + \log_2(3)
\end{aligned}
$$

La probabilidad de error del algoritmo

Tenemos que:

$$
\begin{aligned}
& (1-\frac{1}{2}) \cdot 2^{n \cdot r} \;>\; (1+\frac{1}{2}) \cdot 2^{(n-1)\cdot(n+r)} \\
\Leftrightarrow\quad & \frac{1}{2} \cdot 2^{n \cdot r} \;>\; \frac{3}{2} \cdot 2^{(n-1)\cdot(n+r)} \\
\Leftrightarrow\quad & 2^{n \cdot r} \;>\; 3 \cdot 2^{(n-1)\cdot(n+r)} \\
\Leftrightarrow\quad & 2^{n \cdot r} \;>\; 2^{(n-1)\cdot(n+r)+\log_2(3)} \\
\Leftrightarrow\quad & n \cdot r \;>\; n \cdot (n-1) + (n-1) \cdot r + \log_2(3) \\
\Leftrightarrow\quad & r \;>\; n \cdot (n-1) + \log_2(3)
\end{aligned}
$$

Considerando $r = n \cdot (n-1) + 2$ se cumple la condición.

- Concluimos que $(1 - \frac{1}{2}) \cdot \#\text{CicloSimple}(G^r) > (1 + \frac{1}{2}) \cdot 2^{(n-1)\cdot(n+r)}$

La probabilidad de error del algoritmo

Si $G \in$ HAM y $r = n \cdot (n-1) + 2$ tenemos que $\mathbf{Pr}(\mathcal{B}(G)$ retorne **no**) es igual a:

La probabilidad de error del algoritmo

Si $G \in \text{HAM}$ y $r = n \cdot (n-1) + 2$ tenemos que $\mathbf{Pr}(\mathcal{B}(G)$ retorne **no**$)$ es igual a:

$$
\begin{aligned}
&\mathbf{Pr}(\mathcal{A}(G^r, \tfrac{1}{2}) \leq (1 + \tfrac{1}{2}) \cdot 2^{(n-1)\cdot(n+r)}) \\
&\qquad \leq \mathbf{Pr}(\mathcal{A}(G^r, \tfrac{1}{2}) < (1 - \tfrac{1}{2}) \cdot \#\text{CicloSimple}(G^r)) \\
&\qquad \leq \mathbf{Pr}(\mathcal{A}(G^r, \tfrac{1}{2}) < (1 - \tfrac{1}{2}) \cdot \#\text{CicloSimple}(G^r) \vee \\
&\qquad\qquad\qquad \mathcal{A}(G^r, \tfrac{1}{2}) > (1 + \tfrac{1}{2}) \cdot \#\text{CicloSimple}(G^r)) \\
&\qquad = 1 - \mathbf{Pr}(\mathcal{A}(G^r, \tfrac{1}{2}) \geq (1 - \tfrac{1}{2}) \cdot \#\text{CicloSimple}(G^r) \wedge \\
&\qquad\qquad\qquad \mathcal{A}(G^r, \tfrac{1}{2}) \leq (1 + \tfrac{1}{2}) \cdot \#\text{CicloSimple}(G^r)) \\
&\qquad = 1 - \mathbf{Pr}(|\mathcal{A}(G^r, \tfrac{1}{2}) - \#\text{CicloSimple}(G^r)| \leq \tfrac{1}{2} \cdot \#\text{CicloSimple}(G^r)) \\
&\qquad \leq 1 - \frac{3}{4} \;=\; \frac{1}{4}
\end{aligned}
$$

#CicloSimple no admite un FPRAS: conclusión

Tenemos que en ambos casos el error del algoritmo $\mathcal{B}$ está acotado superiormente por $\frac{1}{4}$

- Considerando $r = n \cdot (n-1) + 2$ en ambos casos

Concluimos entonces que HAM $\in$ BPP

- Por lo tanto NP $\subseteq$ BPP $\square$

La existencia de FPRAS y las reducciones parsimoniosas

Sean $f, g : \Sigma^* \to \mathbb{N}$ dos funciones en $\#$P

Teorema

Si $f \leq^p_{par} g$ y existe un FPRAS para g, entonces existe un FPRAS para f

La existencia de FPRAS y las reducciones parsimoniosas

Sean $f, g : \Sigma^* \to \mathbb{N}$ dos funciones en #P

Teorema

Si $f \leq^p_{par} g$ y existe un FPRAS para g, entonces existe un FPRAS para f

Demostración: Suponga que $h : \Sigma^* \to \Sigma^*$ es una función computable en tiempo polinomial tal que $f(w) = g(h(w))$ para todo $w \in \Sigma^*$

La existencia de FPRAS y las reducciones parsimoniosas

Además, suponga que $\mathcal{A}$ es un FPRAS para g, y defina un algoritmo aleatorizado $\mathcal{B} : \Sigma^* \times (0,1) \to \mathbb{N}$ de la siguiente forma:

$$\text{Para cada } w \in \Sigma^* \text{ y } \varepsilon \in (0,1)\text{: } \mathcal{B}(w,\varepsilon) = \mathcal{A}(h(w),\varepsilon)$$

La existencia de FPRAS y las reducciones parsimoniosas

Además, suponga que $\mathcal{A}$ es un FPRAS para g, y defina un algoritmo aleatorizado $\mathcal{B} : \Sigma^* \times (0,1) \to \mathbb{N}$ de la siguiente forma:

$$\text{Para cada } w \in \Sigma^* \text{ y } \varepsilon \in (0,1)\text{: } \mathcal{B}(w, \varepsilon) = \mathcal{A}(h(w), \varepsilon)$$

Para cada $w \in \Sigma^*$ y $\varepsilon \in (0,1)$:

$$\begin{aligned}
\mathbf{Pr}\Big(|\mathcal{B}(w,\varepsilon) - f(w)| \leq \varepsilon \cdot f(w)\Big) \\
&= \mathbf{Pr}\Big(|\mathcal{A}(h(w),\varepsilon) - g(h(w))| \leq \varepsilon \cdot g(h(w))\Big) \\
&\geq \frac{3}{4}
\end{aligned}$$

La existencia de FPRAS y las reducciones parsimoniosas

Además, suponga que $\mathcal{A}$ es un FPRAS para g, y defina un algoritmo aleatorizado $\mathcal{B} : \Sigma^* \times (0,1) \to \mathbb{N}$ de la siguiente forma:

Para cada $w \in \Sigma^*$ y $\varepsilon \in (0,1)$: $\mathcal{B}(w,\varepsilon) = \mathcal{A}(h(w),\varepsilon)$

Para cada $w \in \Sigma^*$ y $\varepsilon \in (0,1)$:

$$\begin{aligned}\mathbf{Pr}\Big(|\mathcal{B}(w,\varepsilon) - f(w)| \leq \varepsilon \cdot f(w)\Big) \\ &= \mathbf{Pr}\Big(|\mathcal{A}(h(w),\varepsilon) - g(h(w))| \leq \varepsilon \cdot g(h(w))\Big) \\ &\geq \frac{3}{4}\end{aligned}$$

Por lo tanto $\mathcal{B}$ es un FPRAS para f $\square$

Utilizando las reducciones parsimoniosas

Suponiendo que $f \leq^{p}_{par} g$, el resultado anterior puede utilizarse de dos formas:

- Si tenemos un FPRAS para g, entonces podemos construir un FPRAS para f
- Si tenemos una demostración que no existe un FPRAS para f, entonces concluimos que no existe un FPRAS para g

Utilizando las reducciones parsimoniosas

Suponiendo que $f \leq^p_{par} g$, el resultado anterior puede utilizarse de dos formas:

- Si tenemos un FPRAS para g, entonces podemos construir un FPRAS para f
- Si tenemos una demostración que no existe un FPRAS para f, entonces concluimos que no existe un FPRAS para g

Vamos a encontrar otras funciones que no admiten FPRAS utilizando el resultado anterior.

Contando el número de cliques

Dado un grafo $G = (N, A)$, decimos que $S \subseteq N$ es un clique si para cada $a, b \in S$ se tiene que $(a, b) \in A$

Sea $\#\text{CLIQUE}$ una función que, dado un grafo G, retorna el número de cliques de G

Contando el número de cliques

Dado un grafo $G = (N, A)$, decimos que $S \subseteq N$ es un clique si para cada $a, b \in S$ se tiene que $(a, b) \in A$

Sea #CLIQUE una función que, dado un grafo G, retorna el número de cliques de G

Teorema

Si existe un FPRAS para #CLIQUE, entonces $NP = RP$

No existe un FPRAS para #CLIQUE: demostración

Dado un grafo $G = (N, A)$, defina $\overline{G} = (N, \overline{A})$ con $\overline{A} = (N \times N) \smallsetminus A$

No existe un FPRAS para #CLIQUE: demostración

Dado un grafo $G = (N, A)$, defina $\overline{G} = (N, \overline{A})$ con $\overline{A} = (N \times N) \smallsetminus A$

Tenemos que $\#\text{IS}(G) = \#\text{CLIQUE}(\overline{G})$

No existe un FPRAS para #CLIQUE: demostración

Dado un grafo $G = (N, A)$, defina $\overline{G} = (N, \overline{A})$ con $\overline{A} = (N \times N) \smallsetminus A$

Tenemos que $\#\text{IS}(G) = \#\text{CLIQUE}(\overline{G})$

Concluimos que $\#\text{IS} \leq^p_{par} \#\text{CLIQUE}$

- Por lo tanto, si existe un FPRAS para #CLIQUE, entonces existe un FPRAS para #IS y NP = RP

Otro ejemplo: las cláusulas de Horn

Recuerde que una cláusula se dice de Horn si contiene a lo más un literal positivo.

Ejemplo

$(p \vee \neg q \vee \neg r)$, p, $(\neg q \vee \neg r)$ y $\neg q$ son cláusulas de Horn, mientras que $(p \vee q)$ y $(p \vee \neg q \vee r \vee \neg s)$ no lo son.

Además, decimos que una fórmula proposicional φ es de Horn si φ es una conjunción de cláusulas de Horn.

La función #HORN-SAT

Sea #HORN-SAT una función que, dada una fórmula proposicional φ de Horn, retorna el número de valuaciones que satisfacen φ

La función #HORN-SAT

Sea #HORN-SAT una función que, dada una fórmula proposicional φ de Horn, retorna el número de valuaciones que satisfacen φ

Teorema

#HORN-SAT es #P-completo.

La función #HORN-SAT

Sea #HORN-SAT una función que, dada una fórmula proposicional φ de Horn, retorna el número de valuaciones que satisfacen φ

Teorema

#HORN-SAT es #P-completo.

Ejercicio

Demuestre que $L_{\#\text{HORN-SAT}} \in \text{P}$

No existe un FPRAS para #HORN-SAT

Teorema

Si existe un FPRAS para #HORN-SAT, entonces $NP = RP$

No existe un FPRAS para #HORN-SAT

Teorema

Si existe un FPRAS para #HORN-SAT, entonces NP $=$ RP

Demostración: Sea $G = (N, A)$ un grafo tal que $N \neq \emptyset$

No existe un FPRAS para #HORN-SAT

Teorema

Si existe un FPRAS para #HORN-SAT, entonces NP = RP

Demostración: Sea $G = (N, A)$ un grafo tal que $N \neq \emptyset$

Por cada $v \in N$, considere una variable proposicional x_v, y defina φ_G como la siguiente fórmula proposicional:

$$\left(\bigwedge_{a \in N} (x_a \vee \neg x_a) \right) \wedge \left(\bigwedge_{(b,c) \in A} (\neg x_b \vee \neg x_c) \right)$$

No existe un FPRAS para #HORN-SAT: demostración

Tenemos que φ_G es una fórmula proposicional de Horn
y $\#\text{IS}(G) = \#\text{HORN-SAT}(\varphi_G)$

- ¿Cómo se puede extraer un conjunto independiente de G desde una valuación que satisface φ_G?

No existe un FPRAS para #HORN-SAT: demostración

Tenemos que φ_G es una fórmula proposicional de Horn
y $\#\text{IS}(G) = \#\text{HORN-SAT}(\varphi_G)$

- ¿Cómo se puede extraer un conjunto independiente de G desde una valuación que satisface φ_G?

Concluimos que $\#\text{IS} \leq^p_{par} \#\text{HORN-SAT}$

- ¿Cómo se maneja el caso en que $N = \emptyset$ en la reducción?

No existe un FPRAS para #HORN-SAT: demostración

Por lo tanto, si existe un FPRAS para #HORN-SAT, entonces existe un FPRAS para #IS y NP = RP $\square$

No existe un FPRAS para #HORN-SAT: demostración

Por lo tanto, si existe un FPRAS para #HORN-SAT, entonces existe un FPRAS para #IS y NP $=$ RP $\square$

Comentario final

Dado que #IS $\leq^p_{par}$ #HORN-SAT, se tiene que #HORN-SAT es #P-hard

- Como #HORN-SAT $\in$ #P, concluimos que #HORN-SAT es #P-completo