

Trabajo Práctico Nro 7

Ejercicio 1. Responder breve y claramente:

a. ¿Por qué en la complejidad espacial se utilizan MT con una cinta de entrada de sólo lectura?

- **No contar el tamaño de la entrada como espacio ocupado:**
Si se permitiera escribir en la cinta de entrada, esta podría ser utilizada como memoria auxiliar, lo que distorsionaría la verdadera medida del espacio **adicional** necesario para resolver el problema.
- **Permitir análisis más fino (espacio sub-lineal):**
Si se cuenta solo el espacio utilizado **más allá de la entrada**, es posible definir y estudiar clases como **LOGSPACE** ($O(\log n)$) o **PSPACE** ($\text{poly}(n)$). Estas serían imposibles si la entrada ya contara como espacio usado.
- **Modelo robusto y estandarizado:**
Este enfoque permite comparaciones justas entre algoritmos y garantiza que el análisis se enfoque solo en la memoria "de trabajo", sin contaminar la métrica con el tamaño del input.

b. ¿Por qué si una MT tarda tiempo $\text{poly}(n)$ entonces ocupa espacio $\text{poly}(n)$, y si ocupa espacio $\text{poly}(n)$ puede llegar a tardar tiempo $\exp(n)$?

- Una MT no puede usar más espacio del que le da tiempo a visitar.
- En cada paso del cómputo, como mucho puede mover su cabezal una celda (por cinta).
Por lo tanto, si el número total de pasos es acotado por un polinomio $T(n)$, entonces la MT no puede visitar más de $T(n)$ celdas.
- Aunque el espacio sea $S(n) = \text{poly}(n)$, la MT puede:
Explorar distintas combinaciones de contenido en esas celdas.
Cambiar estado muchas veces.
Visitar combinaciones distintas en un **espacio de configuraciones** enorme.

c. ¿Por qué los lenguajes de la clase LOGSPACE son tratables?

De antes: espacio $S(n)$ implica tiempo $c^{S(n)}$, con c constante

En particular: espacio $\log_2 n$ implica tiempo $c^{\log_2 n}$, con c constante

Pero: $c^{\log_2 n} = n^{\log_2 c}$, con c constante, es decir $\text{poly}(n)$

Por lo tanto: espacio $\log_2 n$ implica tiempo $\text{poly}(n)$

Extracto de la teoría, es decir el mayor tiempo posible para un espacio logarítmico es una constante elevado al logaritmo, hacemos unos trucos y nos queda n elevado a una constante lo cual nos da polinomial.

Ejercicio 2. Describir la idea general de una MT M que decida el lenguaje $L = \{a^n b^n \mid n \geq 1\}$ en espacio logarítmico. Ayuda: basarse en el ejemplo mostrado en clase.

Es una maquina con 4 cintas + cinta de entrada:

1. Cinta 1 hacemos $i = 1$
2. Cinta 2 hacemos $j = n$, si es impar rechaza.
3. Copia el símbolo i de w en la cinta 3.
4. Copia símbolo de j en la cinta 4.
5. Si $i = j - 1$: Si los símbolos son distintos acepta, sino rechaza.
 - Si $i \neq j$ si los símbolos son iguales rechaza.
6. Hace $i++$.
7. $j--$
8. Vuelve paso 3.

El espacio es logarítmico, puesto que los contadores i y j en el peor caso llegan a ser n , y en binario miden $O(\log_2(n))$.

Ejercicio 3. Describir la idea general de una MT M que decida el lenguaje SAT en espacio polinomial. Ayuda: la generación y la evaluación de una asignación de valores de verdad se pueden efectuar en tiempo polinomial.

Una MT que decide SAT:

- 2^N posibles asignaciones de valores de las variables.
- Simulamos cada posible combinación.
- La longitud de esta combinación es n (cantidad de variables de SAT)
- Si simulamos cada una de estas combinaciones, en su propia cinta o reutilizamos una cinta y simulamos uno por 1, el máximo espacio que va a ocupar es n .

MT:

1. $i = 1$
2. Generamos la i -ésima asignación para las n variables de w en una cinta, en tiempo polinomial por tanto ocupa espacio polinomial.
3. Verificamos la formula es verdadera, se hace en tiempo polinomial, por tanto ocupa espacio polinomial.
 1. Si es verdadera MT para y acepta.
 2. Si no es verdadera, aumenta contador y se reutiliza el espacio, y volvemos al punto 2.

3. Si no es satisfactoria e $i = 2^n$ la mt para y rechaza.

El contador binario, que estaría en su propia cinta ocuparía como máximo $O(\log_2(2^n)) = O(n)$ y la asignación de variables tiene $O(n)$. Por lo tanto $SAT \in PSPACE$.

Ejercicio 4. Justificar por qué el lenguaje QSAT no pertenecería a P ni a NP. Ayuda: ¿qué forma tienen los certificados asociados a QSAT?

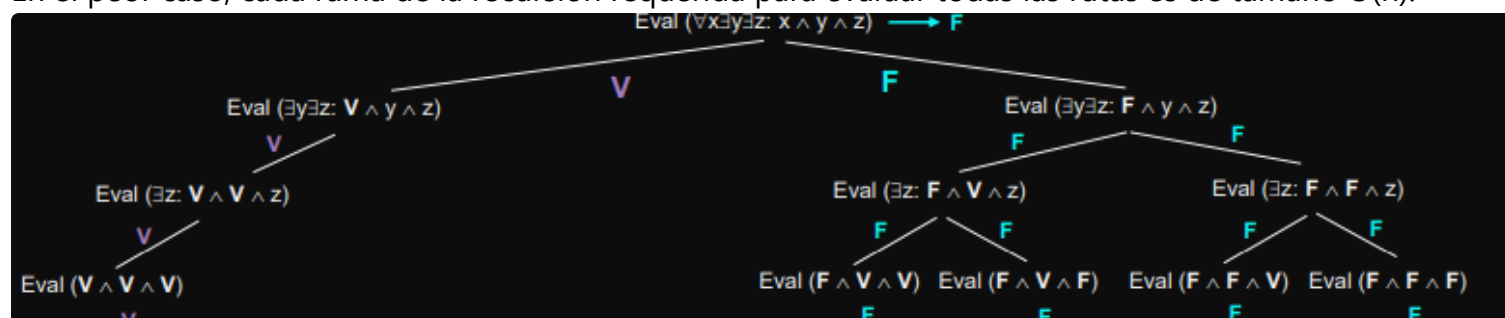
$QSAT \in PSPACE$, $QSAT = \{\varphi \mid \varphi \text{ es una fórmula booleana con cuantificadores, no tiene variables libres, y es verdadera}\}$.

Por ejemplo:

$(\forall x \exists y \exists z: x \wedge y \wedge z) \rightarrow$ Es una formula falsa.

La prueba de porque $QSAT \in PSPACE$ y por tanto no a P ni NP es la función eval:

En el peor caso, cada rama de la recursión requerida para evaluar todas las rutas es de tamaño $O(n)$:



Como vemos, el tamaño de máximo de las ramas coincide con la entrada:

- Podemos reutilizar ramas, por tanto el tamaño máximo no es $\exp(n)$.
- El tamaño de cada instancia, es $O(n) \rightarrow$ datos necesarios como valores asignados, y otras cosas.
- Por tanto el orden $O(n^2)$.

Los certificados por otro lado no son sucintos (son arboles), así que ni estaría en NP.

Ejercicio 5. Probar que $NP \subseteq PSPACE$. Ayuda: Si L pertenece a NP, entonces existe una MT M1 capaz de verificar en tiempo $\text{poly}(|w|)$ si una cadena w pertenece a L, con la ayuda de un certificado x de tamaño $\text{poly}(|w|)$. De esta manera, existe también una MT M2 que decide L en espacio $\text{poly}(|w|)$, sin la ayuda de ningún certificado.

NP es subconjunto de PSPACE:

- Cualquier problema de NP se puede resolver en tiempo $\text{poly}(|w|)$ con una entrada del problema y un certificado con tamaño $\text{poly}(|w|)$.
- Todos los posibles certificados del lenguaje L que pueden ser chequeados por M1 son $2^{p(|w|)}$
- MT 2 entonces lo que es:

1. Genero certificado en orden.
2. Lo verifico simulando M1 en el certificado
 1. Si falla aumento algún contador y vuelvo a generar el certificado.
 2. Si es correcto termine.
 3. Si falla y el contador = $2^{(p|w|)}$
- Si cada vez que generamos un certificado reutilizamos el espacio del anterior, el espacio máximo ocupado esta en $O(\text{poly}|w|)$ porque el tamaño máximo del certificado es $\text{poly}(|w|)$.
- En conclusión, $NP \subseteq PSPACE$ porque existe una MT2 que genera todos los certificados uno por uno y los verifica, la tarea solo requiere en el peor caso $O(\text{poly}|w|)$ para el espacio si reutilizamos el espacio ya usado.

Ejercicio 6. Supongamos que existe una MT M de tiempo polinomial que, dado un grafo G, devuelve un circuito de Hamilton de G si existe o responde no si no existe. Describir la idea general de una MT M' que, utilizando M, decida en tiempo polinomial si un grafo G tiene un circuito de Hamilton. Ayuda: basarse en el ejemplo mostrado en clase con FSAT y SAT.

- CH problema de decisión
- FCH problema de búsqueda
- Ambos CH y FCH le entra un grafo
- Tenemos una MT M que resuelve FCH
- M' entonces hace lo siguiente:
 - Recibe un grafo G
 - Ejecuta M sobre G
 - Si devuelve un circuito M' acepta
 - Si devuelve no M' rechaza
- M es polinomial, porque M' también lo es, la verificación es trivial.

Ejercicio 7. Sea la MTP M que definimos en clase para decidir probabilísticamente el lenguaje MAYSAT = $\{\varphi \mid \varphi \text{ es una fórmula booleana satisfactible por más de la mitad de las posibles asignaciones de verdad}\}$. Hemos indicado que para toda φ , si $\varphi \in \text{MAYSAT}$ entonces M la acepta con probabilidad $> 1/2$, y si $\varphi \notin \text{MAYSAT}$ entonces M la rechaza con probabilidad $\leq 1/2$. Precizando más la primera probabilidad: asumiendo que M tarda $p(n)$, si $\varphi \in$

MAYSAT entonces M la acepta con probabilidad $\geq 1/2 + 1/2p(n)$.
Explicar por qué. Ayuda: en tiempo $p(n)$, M puede producir $2^{p(n)}$ computaciones posibles, y entonces, ¿cuántas son de aceptación como mínimo si $\varphi \in \text{MAYSAT}$?