Pauta de Corrección

Certamen Recuperativo
Informática Teórica

28 de noviembre de 2024

1. Debemos verificar que M es un DFA (si usamos nuestra codificación de máquinas de Turing, solo tiene movidas hacia la derecha, solo ingresa al estado final con movidas leyendo B —desde los estados finales al final de la palabra— y hay exactamente una movida desde cada estado leyendo cada posible símbolo). Enseguida podemos ver si el DFA (en nuestro formato tradicional, sin omitir estados muertos) todos los estados son finales. Alternativamente, al minimizar obtenemos un autómata con un único estado final.

Puntajes

Total	25
Verificar que es un DFA	5
Explicar y justificar	20

2. Una solución es escribir programas que hagan las tareas correctamente, justificando que siempre se ejecutan en tiempo acotado por un polinomio en el tamaño de la palabra de entrada. Usaremos C++ para escribir nuestros programas.

   Dados lenguajes L_1 y L_2, suponemos funciones `bool f1(std::string sigma);` y `bool f2(std::string sigma);` que deciden si sigma pertenece a L_1 o L_2, respectivamente, en tiempos acotados por polinomios $p_1(n)$ y $p_2(n)$, respectivamente. Entonces:

   a) Para reconocer si sigma está en $L_1 \cup L_2$ podemos escribir la función del listado 1.

```
bool in_union(std::string sigma)
{
    return f1(sigma) || f2(sigma);
}
```

   Listado 1: Decidir unión $L_1 \cup L_2$

   Es claro que el resultado es correcto, y el tiempo de ejecución está acotado por $p_1(|\sigma|) + p_2(|\sigma|)$, que es un polinomio en $|\sigma|$.

   b) Para reconocer si sigma está en $L_1 \cdot L_2$ podemos escribir la función del listado 2.

```
bool in_concatenation(std::string sigma)
{
    for(int k = 0; k <= sigma.length(); ++k) {
        std::string sigma1 = sigma.substr(0, k);
        std::string sigma2 = sigma.substr(k);
        if(f1(sigma1) && f2(sigma2))
            return true;
    }
    return false;
}
```

   Listado 2: Decidir concatenación $L_1 \cdot L_2$

   Sin pérdida de generalidad, podemos suponer que p_1 y p_2 son monótonamente crecientes. Para simplificar lo que sigue, definamos $n = |\sigma|$. El dividir σ en dos tiene costo acotado por cn para alguna constante c, con lo que el tiempo de ejecución de nuestra función está acotado por:

$$\begin{aligned}\sum_{0\le k\le n} (p_1(k) + p_2(n-k) + n) &\le \sum_{0\le k\le n} (p_1(n) + p_2(n) + cn) \\ &= (n+1)(p_1(n) + p_2(n) + cn)\end{aligned}$$

   Esto claramente es un polinomio en n.

Puntajes

Total		30
a) Unión		15
Construcción clara	8	
Justificar tiempo polinomial	7	
a) Concatenación		15
Construcción clara	8	
Justificar tiempo polinomial	7	

3. Cada uno por turno, con una breve explicación. Consideramos problemas D_i decidibles, I_i no decidibles, E_i computacionalmente enumerables no decidibles.

   a) $\overline{E_1} \leq I_1$: $\overline{E}_1$ no es computacionalmente enumerable, con lo que si esto se cumple I_1 también debe serlo. No hay contradicción, esto es posible.

   b) $E_2 \leq D_2$: Imposible. Esta reducción permitiría decidir E_2.

   c) $E_3 \leq I_3$: Esto se cumple sin problemas.

   d) $D_2 \leq \overline{D_3}$: Si D_3 es decidible, lo es su complemento $\overline{D}_3$. Esto es posible.

Puntajes

Total		20
a) Explicación y conclusión	5	
b) Explicación y conclusión	5	
c) Explicación y conclusión	5	
d) Explicación y conclusión	5	

4. Cada uno por turno, con una breve explicación. Consideramos problemas $P_i \in \mathsf{P}$, $N_i \in \mathsf{NP}$, C_i es NP-completo, y X_i es desconocido; queremos saber qué se puede concluir sobre X_i en cada caso.

   a) **$X_1 \leq_p N_1$:** De $X_1 \leq_p N_1$ concluimos que X_1 está en NP.

   b) **$X_2 \leq C_2$:** Concluimos que X_2 está en NP, igual que el caso anterior. El que C_2 sea NP-completo (no solo en NP) no aporta información adicional.

   c) **$X_3 \leq_p P_3$ y $X_3 \leq_p C_3$:** De $X_3 \leq_p P_3$ podemos concluir que X_3 está en P. Como $\mathsf{P} \subseteq \mathsf{NP}$, lo segundo no aporta información adicional.

   d) **$C_4 \leq_p X_4$ y $X_4 \leq_p C_4$:** De $C_4 \leq_p X_4$ concluimos que X_4 es NP-duro, de $X_4 \leq_p C_4$ sabemos que X_4 está en NP; con lo que X_4 es NP-completo.

Puntajes

Total	20
a) Explicación y conclusión	5
b) Explicación y conclusión	5
c) Explicación y conclusión	5
d) Explicación y conclusión	5

5. De partida, el autor de la consulta tiene una confusión, no se explica el $2m/3$ que plantea. Parece ser que originalmente consideró $S(\mathscr{A}) = m$, para luego cambiar la notación a lo que usa y olvidó ajustar este valor.

   a) Es incorrecto decir que «hallar una instancia de SUBSET SUM es NP-completo». Construir una instancia del problema es fácil. Hallar el subconjunto que suma a S es difícil, ya que SUBSET SUM es NP-completo, pero al no ser un problema de decisión no es aplicable el clasificarlo como NP-completo. Pero esto puede tomarse simplemente como uso coloquial.

   El problema de fondo es que si $\mathsf{P} \neq \mathsf{NP}$, la reducción esbozada *no* es posible de efectuar en tiempo polinomial (para el problema NP-completo SUBSET SUM el problema de decisión se reduce polinomialmente al de búsqueda; si el primero no puede resolverse en tiempo polinomial el segundo tampoco puede serlo).

   b) Para completar la demostración hace falta demostrar que 3-PARTITION está en NP.

   c) El autor pretende reducir 3-PARTITION $\leq_p$ PARTITION, lo que solo demostraría que 3-PARTITION está en NP. Es incorrecto, debe demostrar PARTITION $\leq_p$ 3-PARTITION.

   Una demostración correcta, usando la notación que introduce el autor de la consulta, sería:

   Proposición. 3-PARTITION *es* NP*-completo.*

   Demostración. Un certificado para 3-PARTITION es una división de $\mathscr{A}$ en tres multisubconjuntos de la misma suma. Revisar esto es rápido, está en NP.

   Para demostrar que 3-PARTITION es NP-duro damos una reducción polinomial de PARTITION a 3-PARTITION. Dada una instancia $\mathscr{A} = \{a_1, a_2, \ldots, a_n\}$ de PARTITION, creamos una instancia de 3-PARTITION agregando un nuevo elemento $a_{n+1} = S(\mathscr{A})/2$. Si la instancia de PARTITION tiene solución, es claro que $S(\mathscr{A})$ es par. Este nuevo elemento únicamente puede ir solo en un subconjunto en una solución de 3-PARTITION, es una reducción. Es claro que la transformación puede efectuarse en tiempo polinomial. □

Puntajes

Total			40
a) Total		20	
Discusión	15		
Conclusión	5		
b) Falta 3-PARTITION $\in$ NP		5	
c) Total		15	
Reducción que intenta demostrar	10		
Es incorrecto, dirección correcta	5		