ISIS 211 Diseño de Algoritmos
Semestre 2010-2 - Parcial 3 - Solución
Noviembre 22, 2010
Prof. Rodrigo Cardoso

Cuadrados mágicos

Un `n`-*cuadrado mágico* es un arreglo `c[1..n,1..n]` cuyas entradas contienen los números en $1..n^2$, de modo que los elementos de cada fila y cada columna suman igual. Por ejemplo, el siguiente es un `3`-cuadrado mágico:

8	1	6
3	5	7
4	9	2

Dado un `n>0`, se quiere encontrar un `n`–cuadrado mágico, o determinar que éste no exista, usando un algoritmo de agenda. La idea que se quiere implementar considera como estados de búsqueda arreglos de `n×n` parcialmente llenos con números en $1..n^2$.

Para facilitar la expresión de los cambios de estado, use

`(b;(i,j):d)`

para denotar un arreglo igual al arreglo `b` en todas las posiciones, excepto quizás en `(i,j)`, donde vale `d`.

1 **[30/100]** Exprese los diferentes elementos de una solución con un algoritmo de agenda (`SOLPOS`, `sat`, ...).

Sea $A_0[n]$ el conjunto de los arreglos de `n×n` con entradas en $0..n^2$, donde si hay entradas repetidas, éstas deben ser `0`'s. Sea `A[n]` el conjunto de elementos de $A_0[n]$ que no tienen entradas con `0`'s.

SOLPOS = A[n]

[5/30]

sat: SOLPOS → **bool**
sat.x ≡ (∀i,j| 1≤i,j≤n : (+k| 1≤i≤n : x[i,k]) = (+k| 1≤i≤n : x[k,j]))

[5/30]

SOL = {x:A[n] | sat.x}

[5/30]

BUSQ = $A_0[n]$.

[5/30]

s = **0** // matriz llena de `0`'s, en $A_0[n]$.

[5/30]

Los arcos del grafo se definen cambiando un `0` por un valor $d \in 1..n^2$ que no esté asignado a alguna entrada:

x → (x;(i,j):d)
si x[i,j]=0, d∉x, 1≤i,j≤n

[5/30]

2 **[10/100]** Muestre que el grafo de búsqueda que defina es de ramificación finita.

Se puede determinar un $r \in \mathbf{nat}$, tal que $\#(SUC.x) \le r$, para todo estado x y, además, $\#(SUC.x_0)=r$, para algún estado x_0.

Se puede elegir $r=n^2$, ya que una configuración x tiene, a lo sumo, este número de sucesores. De hecho, se tendrá que $\#SUC.\mathbf{0}=n^2$ (y, en realidad, todos los demás estados tienen menos sucesores).

[10/10]

3 **[10/100]** Argumente si su algoritmo amerita o no manejo de nodos marcados.

El grafo no puede tener ciclos, porque el sucesor de un nodo tiene más entradas no nulas que el nodo, de manera estricta. No se necesitan nodos marcados, porque no puede haber ciclos en el grafo.

[10/10]

4 **[10/100]** Muestre que el algoritmo termina.

Cada nodo puede tener, a lo sumo r sucesores inmediatos (v. **2**). Además, el grafo no puede tener más de n^2+1 niveles desde la fuente s , ya que en cada paso se cambia un 0 por un valor no nulo aún no incluido.

El número de estados es, por tanto, finito. Como el espacio de búsqueda es finito, la búsqueda exhaustiva debe terminar.

5 **[10/100]** Estime el número de nodos del grafo de agenda que tienen asignados k números de $1..n^2$ en algunas de las casillas del arreglo. Use este resultado para estimar, en el peor caso, el tiempo de ejecución del algoritmo en términos de nodos explorados.

$$\begin{aligned} f.k \;=\; & \text{"Maneras de elegir k de } n^2 \text{ números"} \\ * \; & \text{"Maneras de elegir k de } n^2 \text{ casillas"} \\ * \; & \text{"Maneras de permutar k números"} \\ =\; & \binom{n^2}{k}\binom{n^2}{k} k! \end{aligned}$$

[7/10]

Para estimar el tiempo en peor caso, se considera el tamaño de BUSQ:

$$\begin{aligned} \#BUSQ &= (+k \mid 0 \le k \le n^2 : f.k) \\ &= (+k \mid 0 \le k \le n^2 : \binom{n^2}{k}\binom{n^2}{k} k!) \\ &= (+k \mid 0 \le k \le n^2 : (n^2!)^2/(k!(n^2-k)!)) \\ &= (n^2!)^2 \; (+k \mid 0 \le k \le n^2 : 1/(k!(n^2-k)!)) \end{aligned}$$

[3/10]

Se puede mostrar que[1]

$$(+k \mid 0 \le k \le n^2 : 1/(k!(n^2-k)!)) = O(e^2)$$

de modo que

$$\#BUSQ = O((n^2!)^2)$$

[+3/10]

---

[1] Se sabe que $e = (+k \mid 0 \le k : 1/k!)$. Al considerar cuadrados de sumas parciales de esta serie hasta n^2, se obtiene el resultado.

6 **[10/100]** Defina un predicado dominó (que no sea equivalente a `sat`) y muestre que cumple las condiciones para serlo.

Sea $m_n = n(n^2+1)/2$. Si existe un n−cuadrado mágico todas las filas y columnas deben sumar m_n.

Se define:

$$d.x \equiv (\forall i,j \mid 1\le i,j\le n : (+k \mid 1\le i\le n : x[i,k])\le m_n \land (+k \mid 1\le j\le n : x[k,j])\le m_n))$$

[5/10]

Se muestra que `d` es un predicado dominó:

[d1]

$$
\begin{array}{ll}
 & sat.x \\
\equiv & \\
 & (\forall i,j \mid 1\le i,j\le n : (+k \mid 1\le i\le n : x[i,k]) = (+k \mid 1\le i\le n : x[k,j])) \\
\equiv & \\
 & (\forall i,j \mid 1\le i,j\le n : (+k \mid 1\le i\le n : x[i,k])=m_n \land (+k \mid 1\le i\le n : x[k,j])=m_n)) \\
\Rightarrow & \\
 & (\forall i,j \mid 1\le i,j\le n : (+k \mid 1\le i\le n : x[i,k])\le m_n \land (+k \mid 1\le j\le n : x[k,j])\le m_n)) \\
\equiv & \\
 & d.x
\end{array}
$$

[3/10]

[d2]

Sea $y\in SUC^+.x$ (y tiene los números de x y algunos números no nulos más)

$$
\begin{array}{ll}
 & \neg d.x \\
\Rightarrow & \\
 & (\exists i,j \mid 1\le i,j\le n : (+k \mid 1\le i\le n : x[i,k])>m_n \lor (+k \mid 1\le j\le n : x[k,j])>m_n)) \\
\Rightarrow & \langle \text{se añaden siempre números positivos} \rangle \\
 & (\exists i,j \mid 1\le i,j\le n : (+k \mid 1\le i\le n : y[i,k])>m_n \lor (+k \mid 1\le j\le n : y[k,j])>m_n)) \\
\equiv & \\
 & \neg d.y
\end{array}
$$

[2/10]

Recubrimiento de vértices

Dado un grafo `G(V,E)`, un *recubrimiento de vértices* es un subconjunto de vértices $C\subseteq V$ tal que todo arco de `E` incide en alguno de los vértices de `C`.

El *problema del recubrimiento de vértices* (`RC`) consiste en que, dado un grafo `G(V,E)` y un entero positivo `k`, se decida si hay un recubrimiento de vértices `C`, con $\#C\le k$.

El problema del *recubrimiento de vértices óptimo* (`RCO`) consiste en determinar, para un grafo `G(V,E)`, el tamaño de un recubrimiento de vértices de tamaño mínimo.

8 **[10/100]** Muestre que $RC \in$ **NP**.

Basta considerar un $C \subseteq V$ y verificar que tenga `k` o menos vértices y, para cada uno de los arcos (a lo sumo, n^2), verificar que alguno de los extremos tiene un vértice en `C`. Estos chequeos son polonomiales, de modo que $RC \in$ **NP**.

[10/10]

9 **[10/100]** Muestre que si $RC \in \mathbf{P}$, entonces RCO es solucionable en tiempo polinomial.

Si RC es solucionable en tiempo polinomial, sea $T_{RC}(n,k)$ un algoritmo que resuelve polinomialmente el problema para un grafo de n vértices y una cota k. Sin perdida de generalidad se puede suponer que $0<k\leq n$. Así, la entrada de T_{RC} es $\theta(n^2)$.

Si la entrada de RCO es un grafo de n vértices, i.e., también tiene un tamaño $\theta(n^2)$. La respuesta debe ser un k, con $0<k\leq n$, porque todo V es un recubrimiento y un recubrimiento no puede ser vacío (los arcos deben incidir en los elementos del recubrimiento).

Entonces se puede hacer una búsqueda binaria sobre $1..n$, usando un algoritmo polinomial que resuelva el problema de decisión RC para G y k. Es decir la complejidad de este algoritmo debe ser $T_{RCO}(n)=O(T_{RC}(n,n) \log n)$. Como T_{RC} es polinomial, $_{RCO}$ también debe serlo.

[10/10]