Práctica 3

De recursivo a iterativo

ALGORÍTMICA Y COMPLEJIDAD
Grados en Ing. Informática y Matemáticas
Universidad de Cantabria

Camilo Palazuelos Calderón

Objetivos de la práctica

1. Transformar, de recursivo a iterativo, el diseño de un algoritmo
2. Profundizar en el análisis de algoritmos recursivos e iterativos
3. Evaluar los costes temporal y espacial empíricos de un algoritmo

Duración de la práctica y fecha de entrega

- Las dos sesiones destinadas a la práctica son las de los días 12/13 de marzo y 19/20 de marzo
- La práctica debe entregarse, a través de la plataforma Moodle, antes del 24 de marzo a las 23:59

Qué debéis entregar

- Memoria con respuestas a las preguntas formuladas en este documento
- Código desarrollado y material suplementario que se considere oportuno

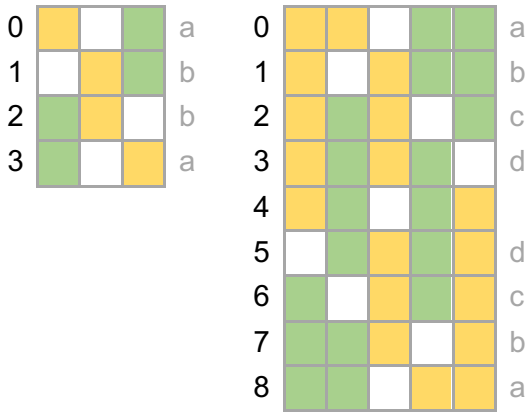
---

Juego de peones

El Museum der Illusionen de Hamburgo está plagado de referencias a la obra de Édouard Lucas, matemático francés del siglo XIX que inventó, entre otros juegos, el rompecabezas de las Torres de Hanói. A la entrada del museo, hay un puesto con once huecos alineados y un cartel que, bajo las palabras SWITCH 10, dice:

> *Antes de empezar, debes colocar las cinco fichas amarillas en un lado del tablero* [se refiere a los once huecos alineados] *y las cinco verdes en el lado opuesto. El hueco central permanece vacío por el momento. La tarea consiste en intercambiar las fichas de modo que, al final, las amarillas ocupen el lugar de las verdes y viceversa. Hay que tener en cuenta, sin embargo, que las fichas solo pueden avanzar un espacio cada vez o pueden saltarse otra ficha que tengan delante si el hueco de detrás está libre.*

El SWITCH 10 está basado en Juego de peones, un rompecabezas publicado en el segundo volumen de las *Récréations mathématiques* (1883) de Lucas. Se trata del caso $n = 5$, donde n es el número de fichas de cada color en el tablero. En la figura siguiente, se muestran los movimientos necesarios para resolver los casos $n = 1$ y $n = 2$. Cada fila se corresponde con una configuración del tablero, que representaremos mediante un array $A[1..2n + 1]$ cuyos elementos pueden tomar los valores `-1` (ficha amarilla), `0` (agujero) o `1` (ficha verde).



Observad que las configuraciones $0..\lfloor m/2 \rfloor$ son simétricas a las configuraciones $\lceil (m + 1)/2 \rceil..m$ en sentido inverso, donde m es el número de movimientos necesario para resolver el juego: para $n = 1$, los pares de filas (a) 0-3 y (b) 1-2 son simétricos; para $n = 2$, lo son (a) 0-8, (b) 1-7, (c) 2-6 y (d) 3-5. El algoritmo RECURSIVO devuelve el número de movimientos iniciales $\lfloor m/2 \rfloor$ necesario para resolver Juego de peones a partir de la configuración 0 de un tablero $A[1..2n + 1]$ (asumiremos que la primera llamada es RECURSIVO($A[1..2n + 1]$, $n + 1$, n, 0)):

```
RECURSIVO(A[1..2n + 1], hueco, ficha, m):
    if m ≥ 2n – 1
        print A[1..2n + 1]
        return 0
    else
        print A[1..2n + 1]
        A[hueco] ↔ A[ficha]
        if ficha = n + 1
            m ← m + 1
        if n ≤ ficha ≤ n + 2
            m ← m + 1
        if hueco – 1 ≤ ficha ≤ hueco + 1
            return RECURSIVO(A[1..2n + 1], ficha, hueco – A[hueco], m) + 1
        else if A[hueco] = A[ficha + A[hueco]]
            return RECURSIVO(A[1..2n + 1], ficha, ficha + A[hueco], m) + 1
        else
            return RECURSIVO(A[1..2n + 1], ficha, ficha + A[hueco] · 2, m) + 1
```

Preguntas en grupos de 2 o 3 alumnos

Pregunta 1 [1 PUNTO]. Diseñad, siguiendo el esquema de transformación de diseños recursivos a iterativos más adecuado, un algoritmo ITERATIVO que devuelva el número de movimientos iniciales $\lfloor m/2 \rfloor$ necesario para resolver Juego de peones a partir de la configuración 0 de un tablero $A[1..2n + 1]$ y muestre por pantalla la configuración de $A[1..2n + 1]$ tras cada uno de los movimientos.

Pregunta 2 [1 PUNTO]. Modificad vuestro diseño del algoritmo ITERATIVO para que, aprovechando las estructuras de datos requeridas para la transformación de la pregunta 1, devuelva el número de movimientos m necesario para resolver Juego de peones a partir de la configuración 0 de un tablero $A[1..2n + 1]$ y muestre por pantalla la configuración de $A[1..2n + 1]$ tras cada uno de los movimientos.

Pregunta 3 [1 PUNTO]. Calculad el coste temporal de vuestro diseño del algoritmo ITERATIVO de la pregunta 2. Eliminad las instrucciones **print** $A[1..2n + 1]$ del diseño del algoritmo y recalculad su coste temporal.

Pregunta 4 [1 PUNTO]. Calculad el coste espacial de vuestro diseño del algoritmo ITERATIVO de la pregunta 2. ¿Qué relación hay entre este y el coste espacial del algoritmo RECURSIVO?

Preguntas en grupos de 4 o 5 alumnos

Pregunta 5 [2 PUNTOS]. Implementad vuestro diseño del algoritmo ITERATIVO de la pregunta 2 en Python. Mostrad que la implementación es correcta proporcionando las salidas correspondientes a los casos $n = 1, \ldots, n = 5$, así como a algún otro tal que $n > 5$.

Pregunta 6 [2 PUNTOS]. Dibujad una gráfica con los tiempos de ejecución de vuestra implementación del algoritmo ITERATIVO, de la que previamente deberéis haber eliminado las instrucciones **print** $A[1..2n + 1]$, en función de n (para, por ejemplo, n entre 10 y 1000 de 10 en 10). ¿Es coherente la curva obtenida con el coste temporal calculado en la pregunta 3?

Pregunta 7 [2 PUNTOS]. Dibujad una gráfica con los recursos de memoria utilizados por vuestra implementación del algoritmo ITERATIVO en función de n (para, por ejemplo, n entre 10 y 1000 de 10 en 10). ¿Es coherente la curva obtenida con el coste espacial calculado en la pregunta 4?

Resolución de los casos $n = 3$, $n = 4$ y $n = 5$