Universidad Rey Juan Carlos

# Práctica 3: Backtracking

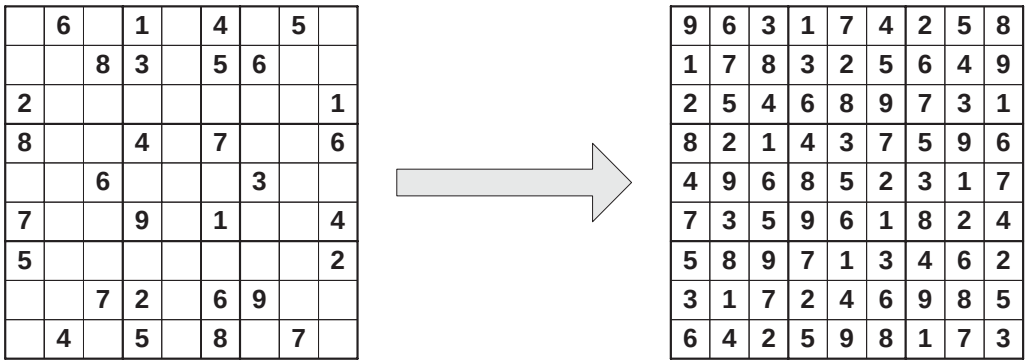
## Diseño y Análisis de Algoritmos

## Grado en Ingeniería Informática

- Valor: 10 % de la nota final
- Los códigos tendrán que probarse con **Mooshak**
  - `http://gibson.escet.urjc.es/~mooshak`
  - Registrarse en Mooshak:
    - El nombre debe tener el formato "NombreApellido1Apellido2", por ejemplo: `ManuelMunozSanchez` (todo junto, con iniciales en mayúsculas, sin tildes ni eñes)
    - El grupo es el asociado a la titulación y número de expediente del alumno
- Grupos: individual
- Debéis subir los códigos fuente tanto a Mooshak como al campus virtual
- Los ejercicios deben ser aceptados por Mooshak para poder puntuar
- Fecha límite: se especificará en el aula virtual

## Índice

# 1. Verificador de Sudokus [1.0 puntos]



| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | 6 | | 1 | | 4 | | 5 | |
| | | 8 | 3 | | 5 | 6 | | |
| 2 | | | | | | | | 1 |
| 8 | | | 4 | | 7 | | | 6 |
| | | 6 | | | | 3 | | |
| 7 | | | 9 | | 1 | | | 4 |
| 5 | | | | | | | | 2 |
| | | 7 | 2 | | 6 | 9 | | |
| | 4 | | 5 | | 8 | | 7 | |

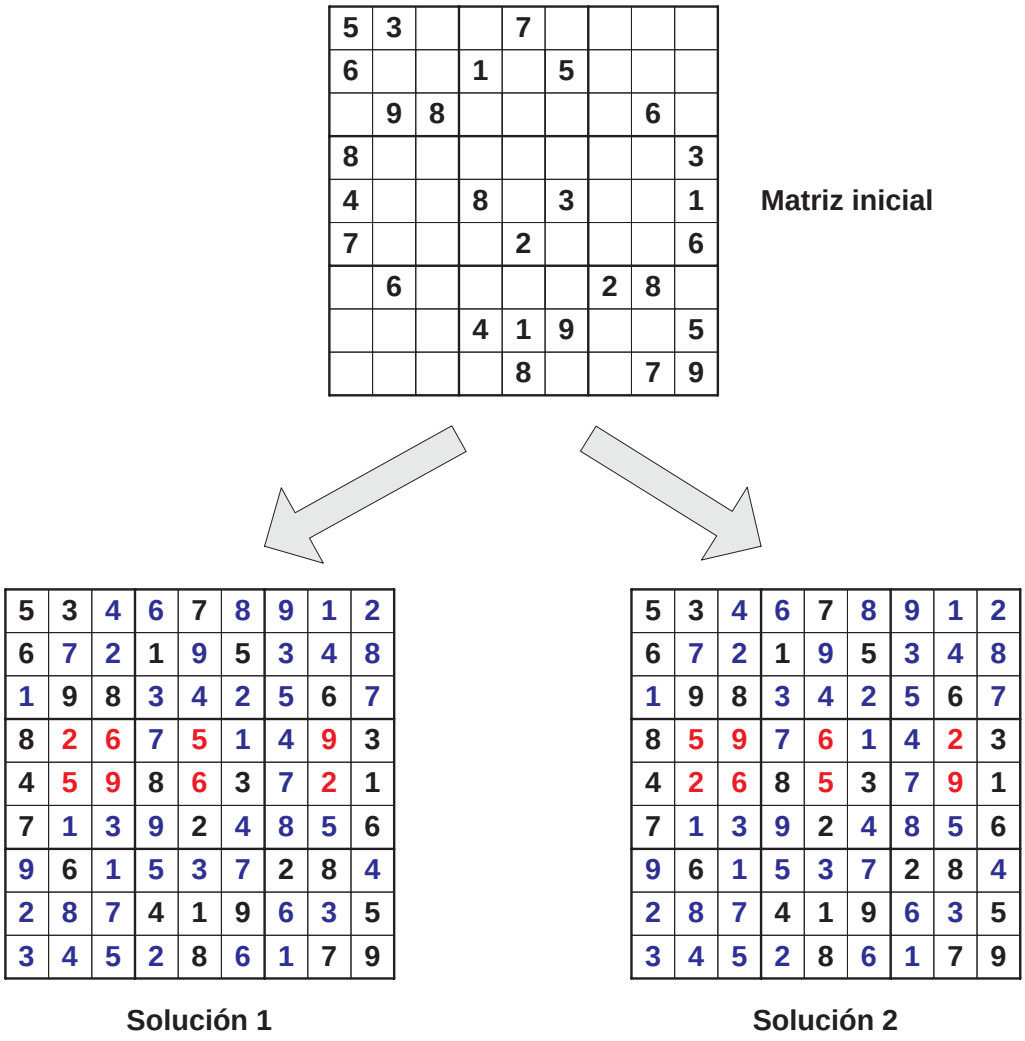
**Matriz inicial**

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 9 | 6 | 3 | 1 | 7 | 4 | 2 | 5 | 8 |
| 1 | 7 | 8 | 3 | 2 | 5 | 6 | 4 | 9 |
| 2 | 5 | 4 | 6 | 8 | 9 | 7 | 3 | 1 |
| 8 | 2 | 1 | 4 | 3 | 7 | 5 | 9 | 6 |
| 4 | 9 | 6 | 8 | 5 | 2 | 3 | 1 | 7 |
| 7 | 3 | 5 | 9 | 6 | 1 | 8 | 2 | 4 |
| 5 | 8 | 9 | 7 | 1 | 3 | 4 | 6 | 2 |
| 3 | 1 | 7 | 2 | 4 | 6 | 9 | 8 | 5 |
| 6 | 4 | 2 | 5 | 9 | 8 | 1 | 7 | 3 |

**Solución**

El sudoku es un pasatiempos-juego-puzzle basado en el uso de la lógica, que se popularizó en el 2005. El objetivo del puzzle es rellenar con símbolos – generalmente números comprendidos entre el 1 y el 9 – una matriz de $9 \times 9$ celdas, subdividida en 9 submatrices (cajas) de tamaño $3 \times 3$, a partir de unos valores iniciales para un subconjunto de celdas. Existen tres restricciones a la hora de rellenar las celdas: un determinado símbolo sólo puede aparecer una vez en cada fila, columna y caja.

Un sudoku está bien planteado si la solución es única para los valores iniciales de las celdas, como ocurre en la figura anterior. Si no está bien planteado puede que no tenga ninguna solución, o puede que tenga múltiples soluciones. Por ejemplo, el siguiente sudoku tiene dos soluciones posibles (en rojo se señalan las diferencias):




## 1.1. Problema a implementar

Se pide implementar un programa que compute el número de soluciones que admite un sudoku, esté bien planteado o no, usando la técnica de *backtracking*.

### 1.1.1. Descripción de la entrada

La entrada contiene 9 líneas. Cada línea representará una fila del sudoku inicial. Por tanto, contendrá dígitos enteros del 0 al 9, separados por espacios, donde el 0 denota una celda vacía, y el resto un símbolo particular. Habrá un salto de línea después del último dígito de cada línea.

### 1.1.2. Descripción de la salida

La salida contendrá un número entero, seguido de un salto de línea. El entero será el número de soluciones posibles que admita el sudoku (si no admite soluciones la salida será 0).

### 1.1.3. Ejemplo de entrada

```
5 3 0 0 7 0 0 0 0
6 0 0 1 0 5 0 0 0
0 9 8 0 0 0 0 6 0
8 0 0 0 0 0 0 0 3
4 0 0 8 0 3 0 0 1
7 0 0 0 2 0 0 0 6
0 6 0 0 0 0 2 8 0
0 0 0 4 1 9 0 0 5
0 0 0 0 8 0 0 7 9
```

### 1.1.4. Salida para el ejemplo de entrada

```
2
```