Algorítmica

Tema 1. Planteamiento General

Tema 2. La Eficiencia de los Algoritmos

Tema 3. Algoritmos "Divide y vencerás"

Tema 4. Algoritmos Voraces ("Greedy")

Tema 5. Algoritmos basados en Programación Dinámica

Tema 6. Algoritmos para la Exploración de Grafos ("Backtracking", "Branch and Bound")

Tema 7. Otras metodologías algorítmicas

Algorítmica

TEMA 1: PLANTEAMIENTO GENERAL

1. CONCEPTO DE ALGORITMO
2. RESOLUCIÓN DE PROBLEMAS
3. CLASIFICACIÓN DE PROBLEMAS
4. ALGORÍTMICA
5. INTRODUCCION A LA EFICIENCIA DE ALGORITMOS
6. DISEÑO DE ALGORITMOS

Bibibliografía:

- **G. BRASSARD, P. BRATLEY. Fundamentos de Algoritmia. Prentice Hall (1997)**

CONCEPTO DE ALGORITMO

Definición de Algoritmo

Secuencia ordenada de pasos exentos de ambigüedad y determinísticos tal que al llevarse a cabo con fidelidad dará como resultado que se realice la tarea para la que se ha diseñado en un tiempo finito (se obtiene la solución del problema planteado)

Propiedades de un ALGORITMO

a) **Finitud**: terminar después de un número finito de etapas

b) **Precisión**: cada etapa debe estar definida de forma precisa; las acciones que hay que llevar a cabo deben estar rigurosamente especificadas para cada caso

c) **Entrada**:

d) **Salida**:

e) **Efectividad**: todas las operaciones que hay que realizar deben ser tan básicas como para que se puedan hacer exactamente y en un periodo finito de tiempo

RESOLUCIÓN DE PROBLEMAS

Resolución de un problema con un ordenador

- Diseñar un algoritmo para el problema
- Expresar el algoritmo como un programa
- Ejecutar el programa correctamente

ESTUDIO/CLASIFICACIÓN DE PROBLEMAS

Década 30: Problemas Computables y No Computables

Década 50: Complejidad de los problemas computables (búsqueda de algoritmos más eficaces)

Década 70: Clasificación de los problemas computables: P y NP

ESTUDIO/CLASIFICACIÓN DE PROBLEMAS

Problemas P y NP

- ¿Qué es la clase P?
- ¿Qué es la clase NP?
- ¿Qué hace que un problema sea NP?

Clase P de problemas

- Se consideran problemas con tiempo polinomial porque sus tiempos de ejecución son funciones polinomiales
- Se dice que todos esos problemas están en la Clase P
- Se asume que un tiempo polinómico es un tiempo *eficiente*

Clase NP de problemas

- Hay problemas para los que la única forma de resolverlos en tiempo polinomial es realizando una etapa aleatoria (incluyendo el azar de alguna manera).
- Típicamente, la solución incluye una primera etapa que de forma no determinista elige una posible solución, y entonces en etapas posteriores comprueba si esa solución es correcta.

Para esto, haría falta una máquina de turing no determinista. Podemos suponer que una máquina no determinista es una máquina "adivina" o aquella que podría clonarse y realizar tantas operaciones en paralelo como se quiera. Es lo que se busca con la computación cuántica.

La empresa canadiense D-Wave System había supuestamente presentado el 13 de febrero de 2007 en Silicon Valley, una primera computadora cuántica comercial de 16-qubits de propósito general; luego la misma compañía admitió que tal máquina llamada Orion no es realmente una Computadora Cuántica, sino una clase de máquina de propósito general que usa algo de mecánica cuántica para resolver problemas. Aún seguimos esperando…

Relación entre las clases P y NP

Clase P $\subseteq$ Clase NP

- La clase P es un subconjunto de la clase NP ya que podríamos construir un algoritmo que resolviera los problemas de la clase P con las mismas dos etapas que se usan en los problemas de la clase NP
- La diferencia es que tenemos soluciones en tiempo polinomial para ***todos*** los problemas de la clase P, pero no los tenemos para ***todos*** los de la clase NP.

Reducción de problemas y complejidad

- Si reducimos un problema (A) a otro (B) y obtenemos una solución con un algoritmo para el problema B, podemos transformar esa solución para convertirla en una solución para el problema A
- Si podemos realizar las transformaciónes de un problema en tiempo polinomial ($A \rightarrow B$ y $B \rightarrow A$), y también podemos resolverlo en tiempo polinomial, sabemos que el problema original es resoluble en tiempo polinomial

$$(A \rightarrow B + Res(B) + B \rightarrow A) \Rightarrow O(n^i) + O(n^j) + O(n^k) \Rightarrow O(n^f)$$
$$\text{tq. } f = \max(i, j, k)$$

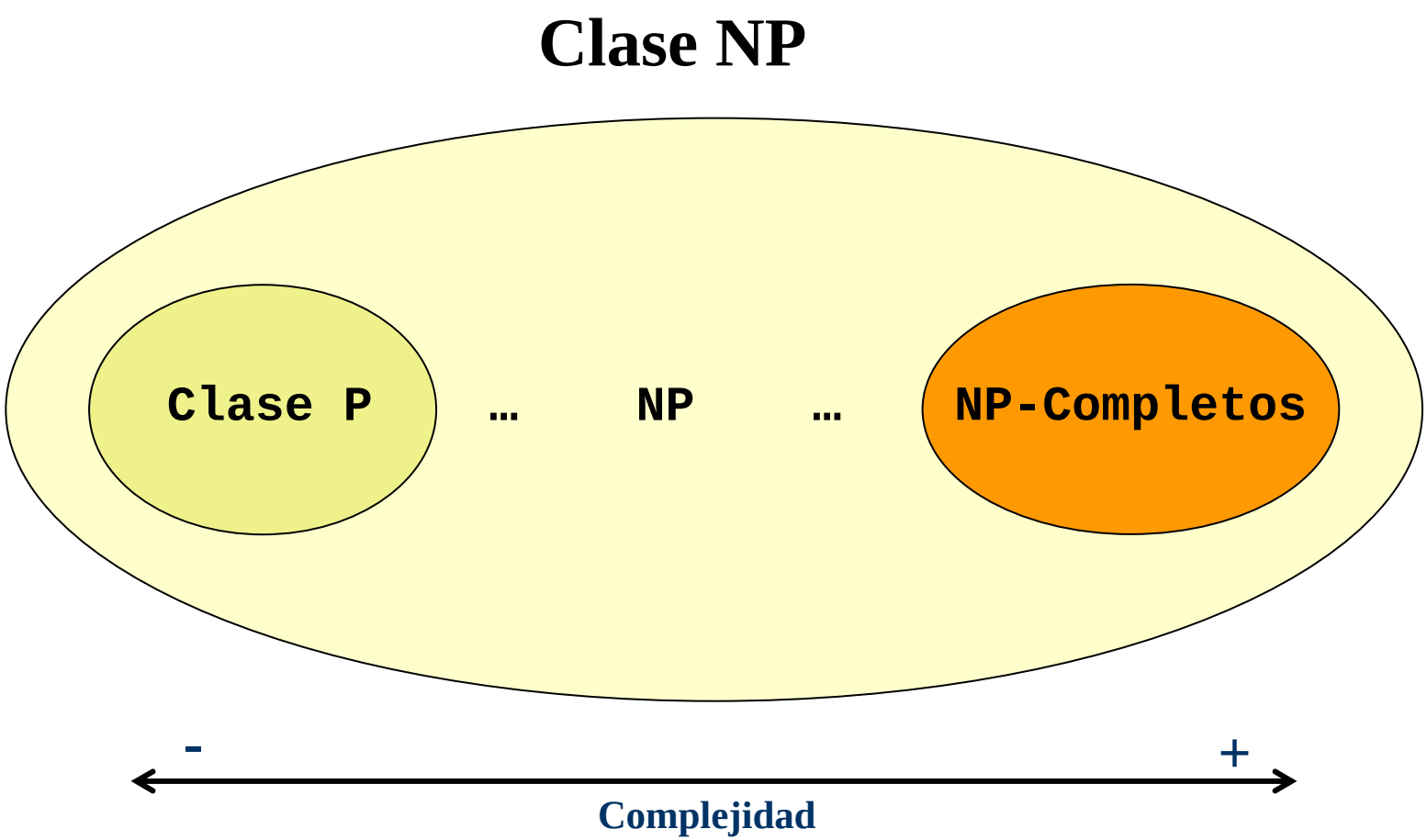
Problemas NP-Completos

- Se dice que un problema es NP-completo si todos los problemas de la clase NP pueden reducirse a él
- Los problemas NP-completos son los problemas mas difíciles de la clase de problemas NP
- Debido a la reducción, si encontráramos un algoritmo en tiempo polinomial para un problema NP-completo, sabríamos que todos los problemas de la clase NP requieren un tiempo polinómico

¿Es P = NP?

- La clase P es un subconjunto de la clase NP
- Para que ambas clases sean iguales, todos los problemas en la clase NP debería tener algoritmos en tiempo polinomial
- Hasta ahora nadie ha sido capaz de encontrar un algoritmo en tiempo polinomial ni de demostrar lo contrario
- Por tanto, la cuestión de si “P = NP” es actualmente un problema abierto

¿Es P = NP?



ALGORÍTMICA

El estudio de los algoritmos incluye el de diferentes e importantes áreas de investigación y docencia, y se suele llamar Algorítmica

La Algorítmica considera:

1. **La construcción**
2. **La expresión**
3. **La validación**
4. **El análisis, y**
5. **Estudio empírico**

1. La construcción de algoritmos

- Creación de algoritmos: parte creativa y sistemática
- El acto de crear un algoritmo no se puede dominar si no se conocen a la perfección las técnicas de diseño de los algoritmos
- El área de la construcción de algoritmos engloba el estudio de los métodos que, facilitando esa tarea, se han demostrado en la práctica mas útiles

2. La expresión de algoritmos

- Los algoritmos deben expresarse de forma precisa, clara, e independientes de un lenguaje de programación en concreto

3. Validación de algoritmos

- Validación: Demostrar que las respuestas que da son correctas para todas las posibles entradas
- Único método válido: demostración formal
- En muchos casos no es factible: pruebas empíricas o parciales

4. Análisis de algoritmos

- Análisis de algoritmos: Determinar los recursos (tiempo, espacio) que consume un algoritmo en la resolución de un problema
- Permite comparar algoritmos con criterios cuantitativos
- Elección de un algoritmo entre varios para resolver un problema

5. Estudio empírico

- Se trata de generar e implementar el algoritmo mediante un programa: codificación, prueba y validación (con depuración en su caso)

- Evaluación empírica del tiempo y espacio que requiere el algoritmo implementado para resolver problemas de distinto tamaño.

INTRODUCCIÓN A LA EFICIENCIA DE LOS ALGORITMOS

Elección de un Algoritmo

En la práctica no solo queremos algoritmos, sino que queremos buenos algoritmos en algún sentido propio de cada usuario.

INTRODUCCIÓN A LA EFICIENCIA DE LOS ALGORITMOS

Elección de un Algoritmo

A menudo tendremos varios algoritmos para un mismo problema, y deberemos decidir cual es el mejor, o cual es el que tenemos que escoger según algún criterio, para resolverlo.

Esto nos centra el estudio en el campo del Análisis de los Algoritmos.

Problema Central:

Determinar ciertas características que sirvan para evaluar su rendimiento.

INTRODUCCIÓN A LA EFICIENCIA DE LOS ALGORITMOS

Elección de un Algoritmo

Un criterio de bondad es la longitud del tiempo consumido para ejecutar el algoritmo; esto puede expresarse en términos del número de veces que se ejecuta cada etapa.

INTRODUCCIÓN A LA EFICIENCIA DE LOS ALGORITMOS

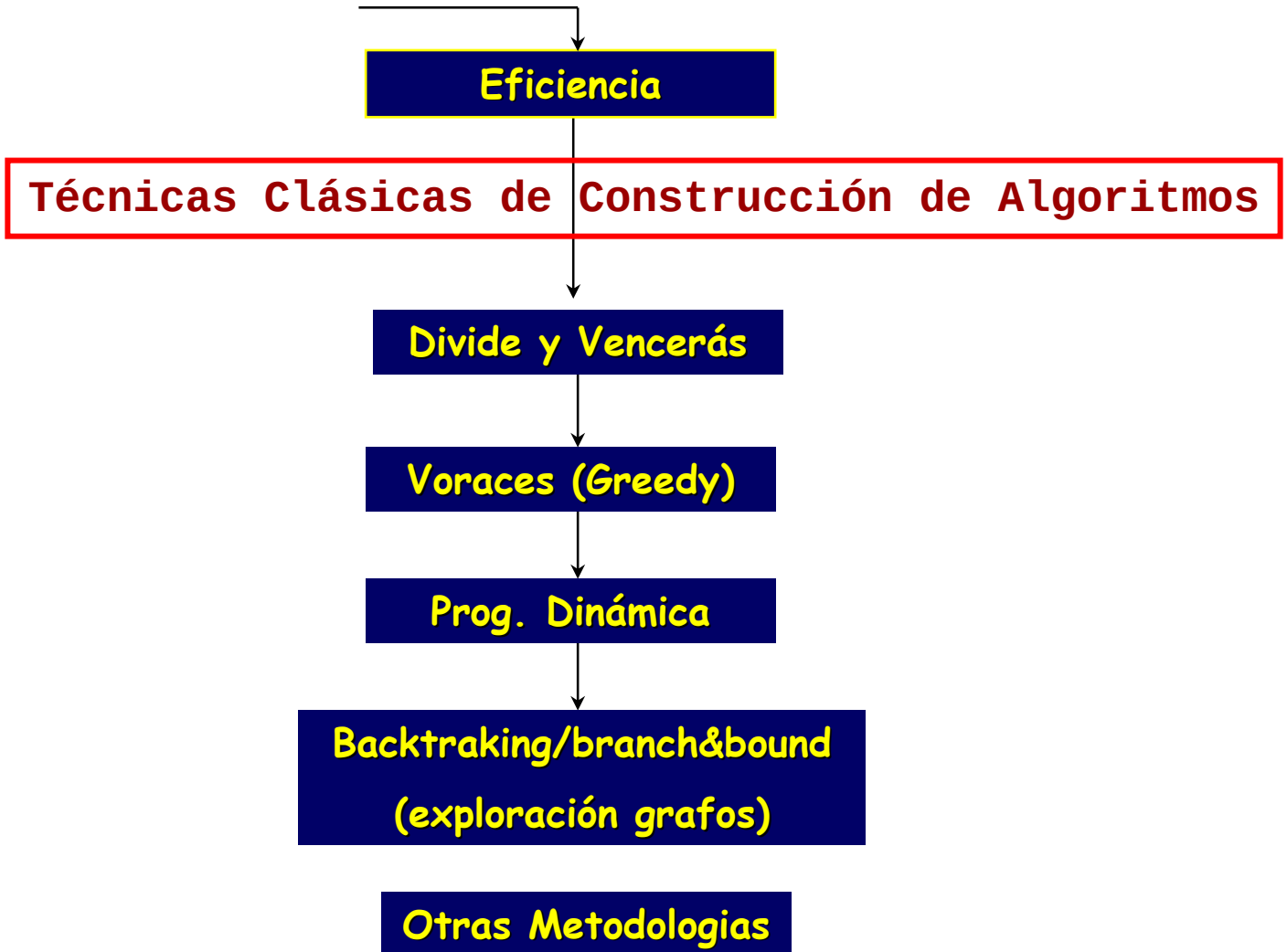
Modelos para el análisis de la eficiencia:

- **Enfoque Empírico.**
- **Enfoque teórico.**
- **Enfoque Híbrido.**

SOBRE EL DISEÑO DE ALGORITMOS

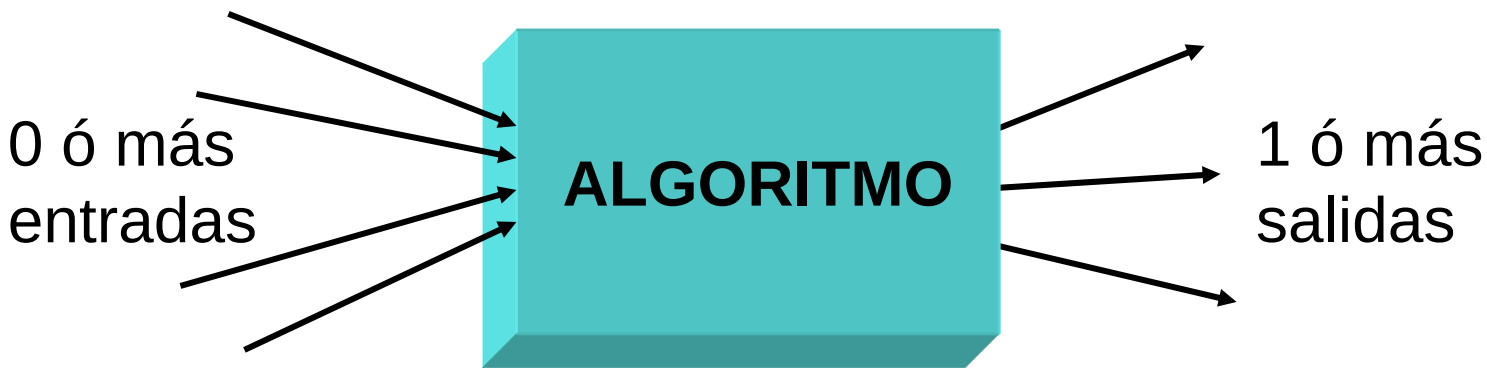
Se necesitan técnicas para la Construcción de Algoritmos Eficaces y Eficientes.

SOBRE EL DISEÑO DE ALGORITMOS



Resumen

- **Algoritmo:**
  Conjunto de reglas para resolver un problema.
- **Propiedades**
  - **Definibilidad:** El conjunto debe estar bien definido, sin dejar dudas en su interpretación.
  - **Finitud:** Debe tener un número finito de pasos que se ejecuten en un tiempo finito.



Resumen

- **Algoritmos deterministas:** Para los mismos datos de entrada se producen los mismos datos de salida.
- **Algoritmos no deterministas:** Para los mismos datos de entrada pueden producirse diferentes de salida.

- **ALGORÍTMICA:** Ciencia que estudia técnicas para construir algoritmos eficientes y técnicas para medir la eficacia de los algoritmos.
- **Objetivo**: Dado un problema concreto encontrar la mejor forma de resolverlo.

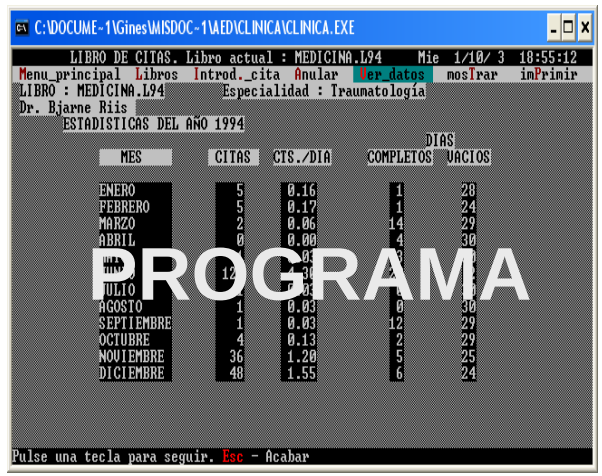
Objetivo de la asignatura

Ser capaz de **analizar**, **comprender** y **resolver** una amplia variedad de **problemas** de programación, diseñando soluciones **eficientes** y de **calidad**.

Pero **ojo**, los algoritmos no son el único componente en la resolución de un problema de programación.

Resumen

Algoritmos
+
Estructuras de datos



Algoritmos + Estructuras de Datos = Programas

- **Estructura de datos:** Parte estática, almacenada.
- **Algoritmo:** Parte dinámica, manipulador.

Resumen

ALGORITMICA = ANÁLISIS + DISEÑO

- **Análisis de algoritmos:** Estudio de los recursos que necesita la ejecución de un algoritmo.
- No confundir con análisis de un problema.

- **Diseño de algoritmos:** Técnicas generales para la construcción de algoritmos.
- Por ejemplo, divide y vencerás: dado un problema, divídelo, resuelve los subproblemas y luego junta las soluciones.

Resumen

- **Análisis de algoritmos.** Normalmente estamos interesados en el estudio del tiempo de ejecución.
- Dado un algoritmo, usaremos las siguientes notaciones:
  - $t(..)$: Tiempo de ejecución del algoritmo.
  - $O(..)$: Orden de complejidad.
  - $o(..)$: O pequeña del tiempo de ejecución.
  - $\Omega(..)$: Cota inferior de complejidad.
  - $\Theta(..)$: Orden exacto de complejidad.

Resumen: Análisis de algoritmos.

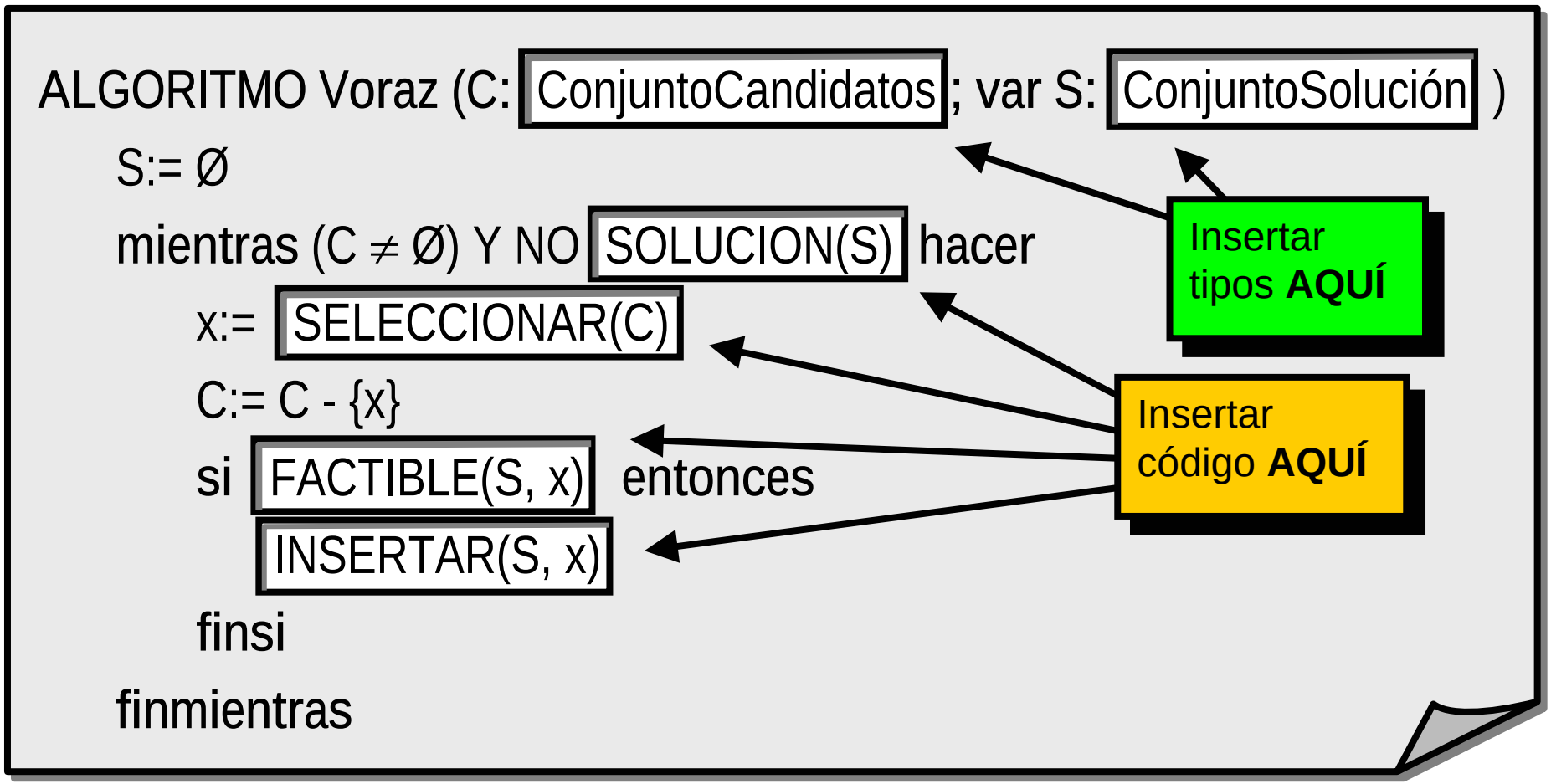
- **Ejemplo.** Analizar el tiempo de ejecución y el orden de complejidad del siguiente algoritmo.

```
Hanoi (N, A, B, C: integer)
    if  N=1 then
        Mover (A, C)
    else begin
        Hanoi (N-1, A, C, B)
        Mover (A, C)
        Hanoi (N-1, B, A, C)
    end
```

- Mecanismos:
  - Conteo de instrucciones.
  - Uso de ecuaciones de recurrencia.
  - Medida del trabajo total realizado.

Resumen:Diseño de algoritmos.

- **Diseño de Algoritmos.** Técnicas generales, aplicables a muchas situaciones.
- **Esquemas algorítmicos.** Ejemplo:



Y para terminar..algunos ejemplos...

¿Qué clase de problemas pueden estudiarse?

El Sudoku

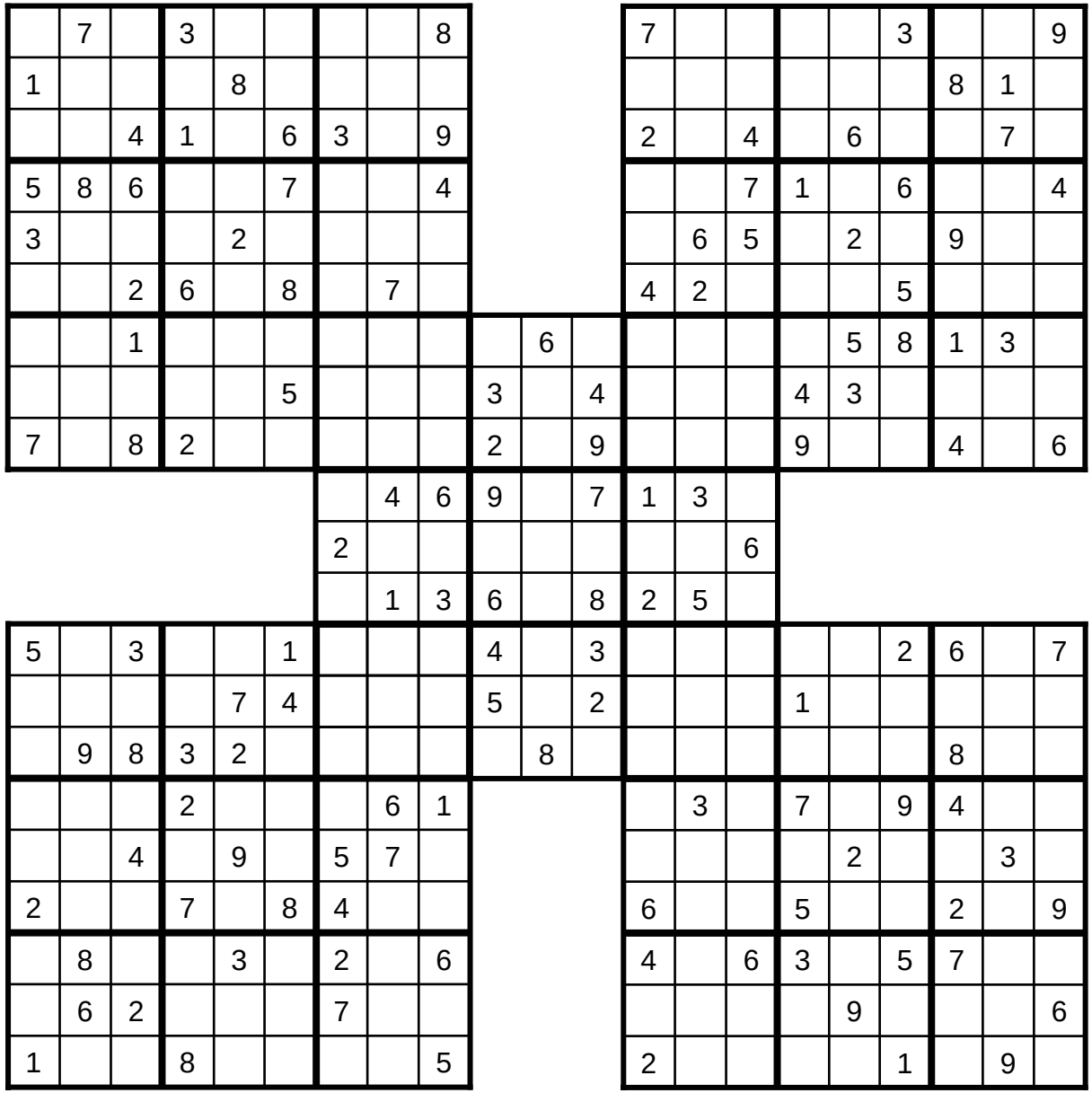
	8	1		2	4			
2					8			5
			1	3	5	2	4	
9	2	7		4				
4								7
				1		8	9	4
	4	5	6	7	9			
1			3					9
			4	5		7	6	

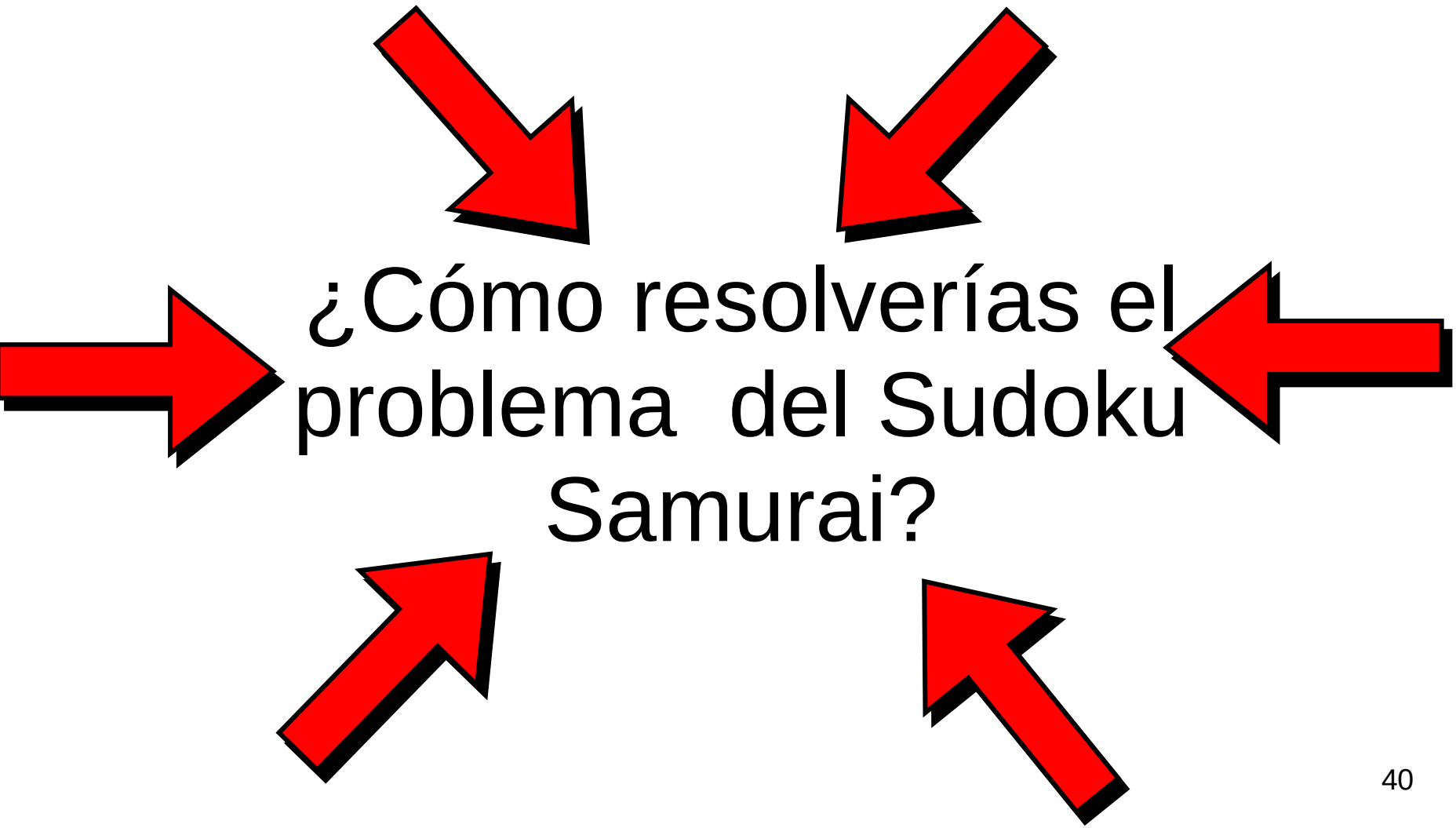
Se puede diseñar un algorítmo que lo resuelva de forma sencilla

El Sudoku

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

El Sudoku Samurai

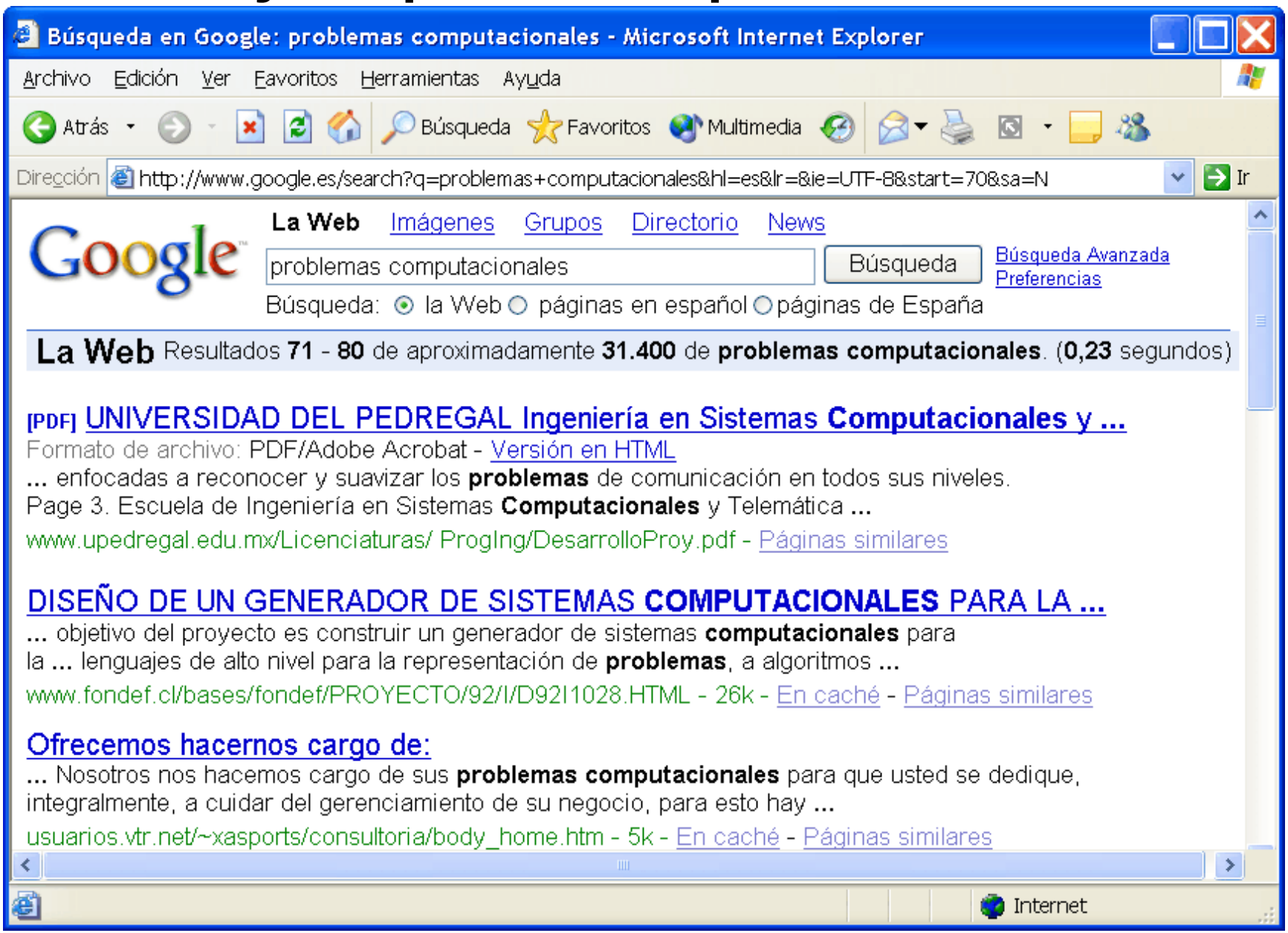



¿Cómo resolverías el problema del Sudoku Samurai?

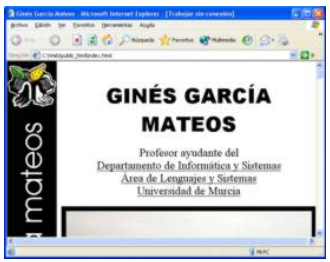
Ejemplos de problemas

Ejemplos de problemas

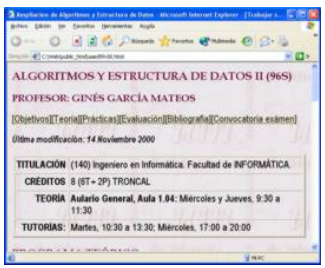
Ejemplos de problemas



Buscador de Internet



algoritmos, ayudante, curso, datos, estructuras, garcía, mateos, …

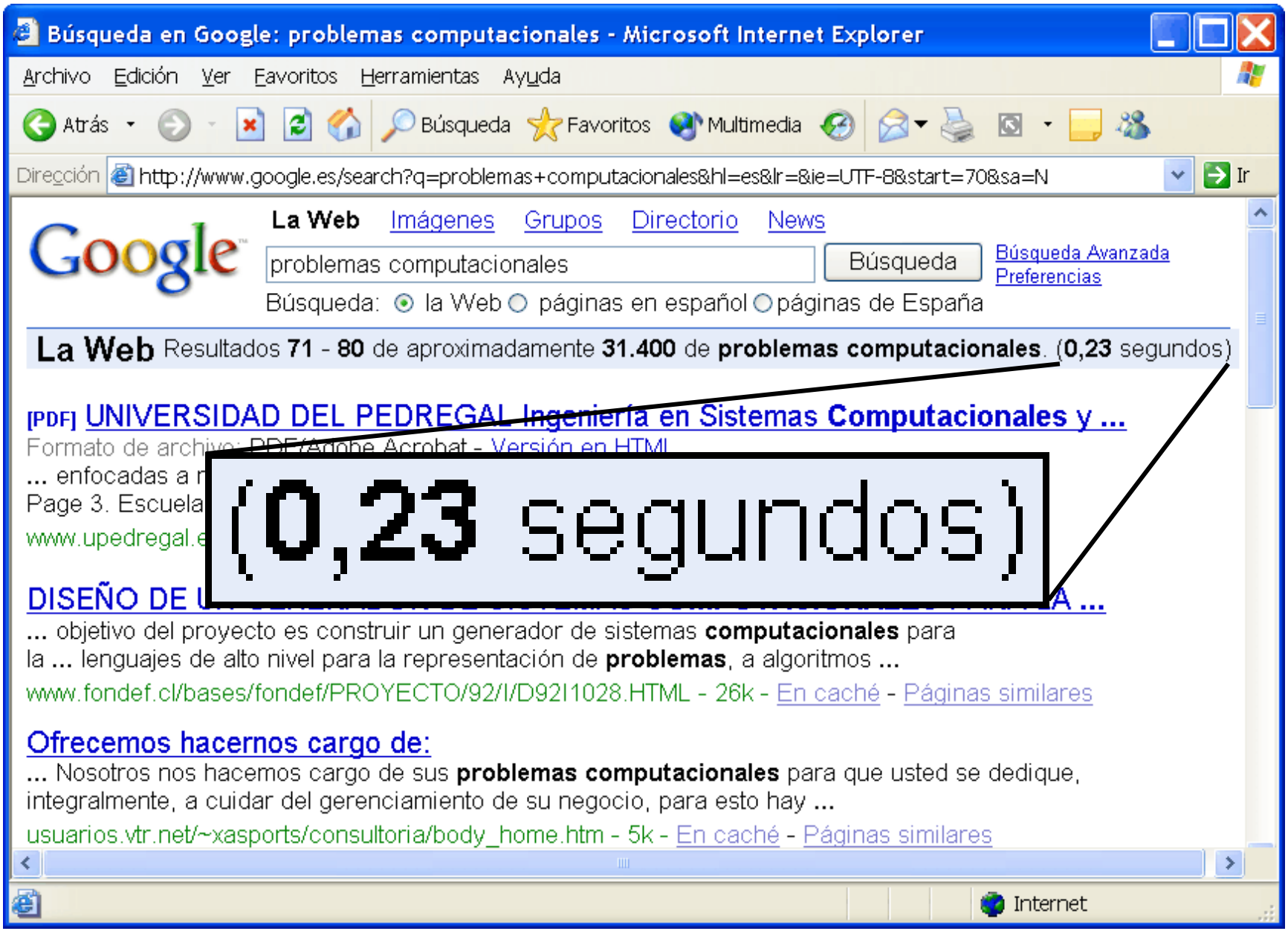


algoritmos, cosa, curso, datos, estructuras, evaluación, prácticas, …

agua, botavara, barco, confeccionar, las, velas, …

Buscador de Internet

Buscador de Internet



Buscador de Internet

- ¡¡¡Cuatro mil millones de páginas en un cuarto de segundo!!!

- **Problema:** ¿cómo estructurar la información necesaria para realizar las consultas rápidamente? ¿Qué algoritmos de búsqueda utilizar?

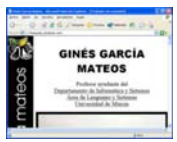
Buscador de Internet

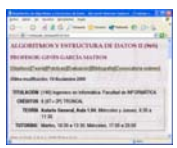
- Supongamos una red de 1024 ordenadores a 3 GHz.
- Supongamos que cada página tiene 200 palabras, de 8 letras cada una y en cada letra se tarda 2 ciclos de reloj.
- ¡¡El recorrido de todas las páginas tardaría 4,5 segundos!!

Buscador de Internet

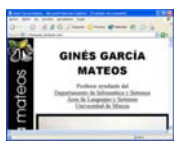
- **Solución:** Darle la vuelta al problema…

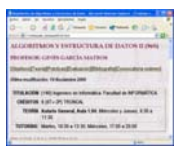
agua → 

ayudante → 

cosa → 

las → 





. . .


Búsqueda en Google: problemas computacionales - Microsoft Internet Explorer
Archivo Edición Ver Favoritos Herramientas Ayuda
Atrás Búsqueda Favoritos Multimedia
Dirección http://www.google.es/search?q=problemas+computacionales&hl=es&lr=&ie=UTF-8&start=70&sa=N
Ir
Google
La Web Imágenes Grupos Directorio News
problemas computacionales
Búsqueda
Búsqueda: la Web páginas en español
La Web Resultados 71 - 80 de aproximadamente 31.400
(0,23
[PDF] UNIVERSIDAD DEL PEDREGAL
Formato de archivo: PDF/Adobe Acrobat - Versión en
... enfocadas a reconocer y suavizar los
Page 3. Escuela de Ingeniería en Computación y Telemática
www.upedregal.edu.mx
Páginas similares
DISEÑO DE UN GENERADOR DE SISTEMAS COMPUTACIONALES PARA LA ...
... objetivo del proyecto es construir un generador de sistemas computacionales para la ... lenguajes de alto nivel para la representación de problemas, a algoritmos ...
www.fondef.cl/fondef/PROYECTO/92/I/D92I1028.HTML - 26k - En caché - Páginas similares
Ofrecemos hacernos cargo de:
... Nosotros nos hacemos cargo de sus problemas computacionales para que usted se dedique, integralmente, a cuidar del gerenciamiento de su negocio, para esto hay ...
usuarios.vtr.net/~xasports/consultoria/body_home.htm - 5k - En caché - Páginas similares
Internet
RESUELTO

Planificador de rutas

Planificador de rutas

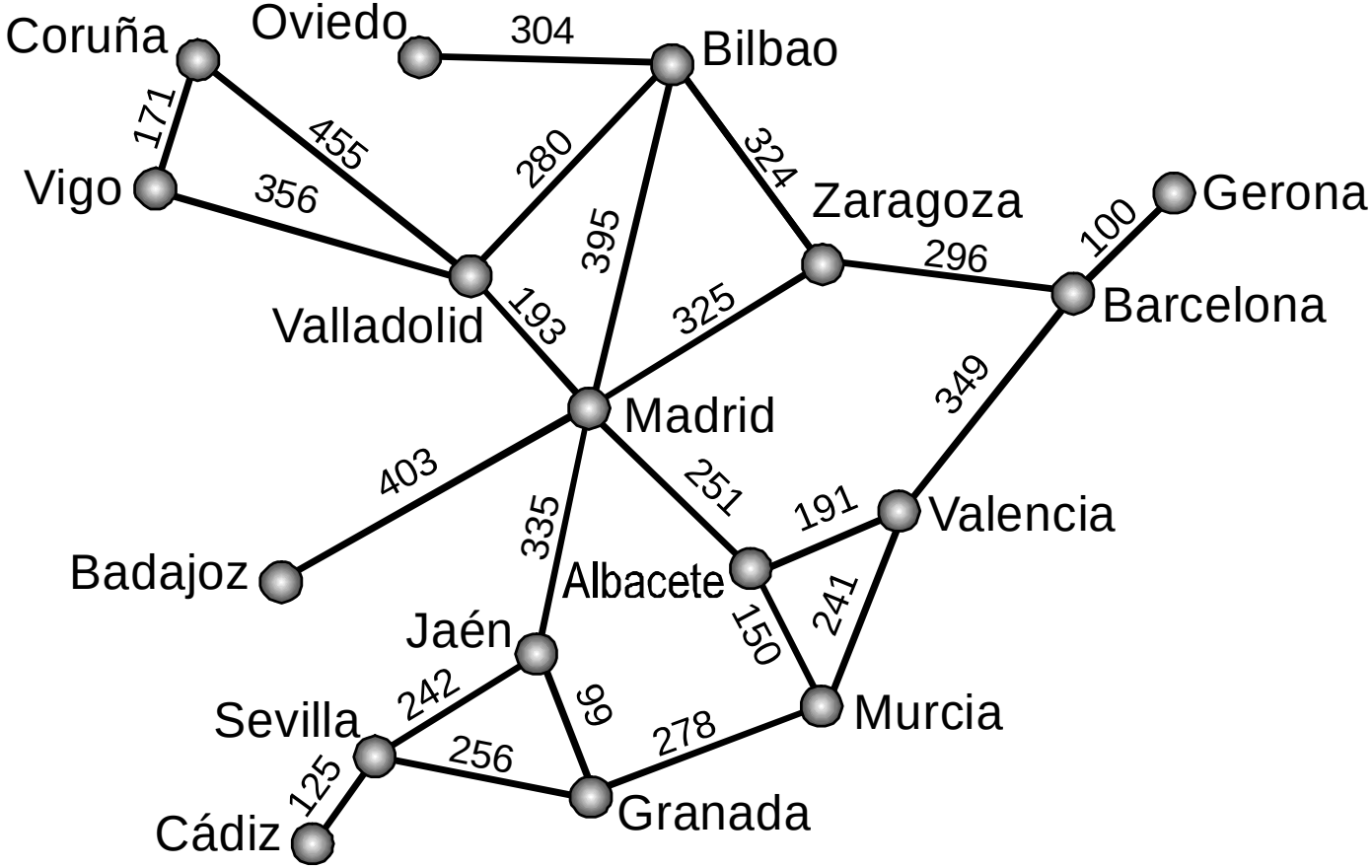
Planificador de rutas

Planificador de rutas

- ¿Cómo representar la información (lugares y carreteras)?
- ¿Cómo calcular el camino más corto entre dos lugares?

Planificador de rutas

- Representación mediante un **grafo**:
  - Lugares = nodos.
  - Carreteras = arcos entre nodos.



Planificador de rutas

- ¿Cómo calcular los caminos mínimos en el mapa?
- Fuerza bruta: empezar por un sitio y probar todos los caminos hasta llegar a otro. NO!!!!
- Algorítmos de camino mínimo

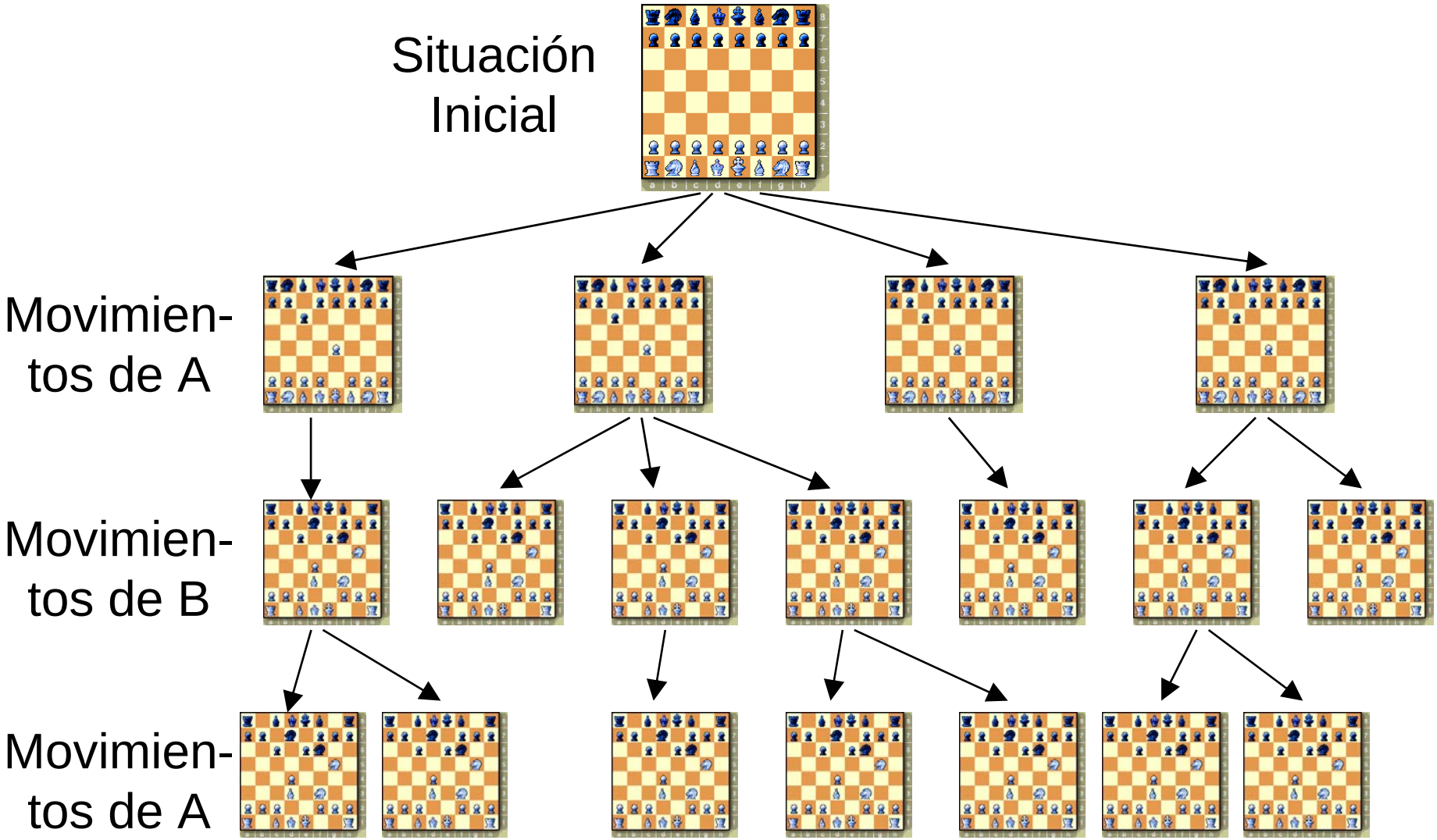
Planificador de rutas

Jugador de Ajedrez

Jugador de Ajedrez

- ¿Cómo representar el problema?
- ¿Cómo decidir el siguiente movimiento de forma “inteligente”? ¿?

Jugador de Ajedrez



Jugador de Ajedrez

- El **árbol de juego** del ajedrez representa todas las posibles partidas del juego.
- **Solución:** encontrar un camino en el árbol que llegue hasta la victoria.
- ¿Qué tamaño tiene el árbol de juego del ajedrez?

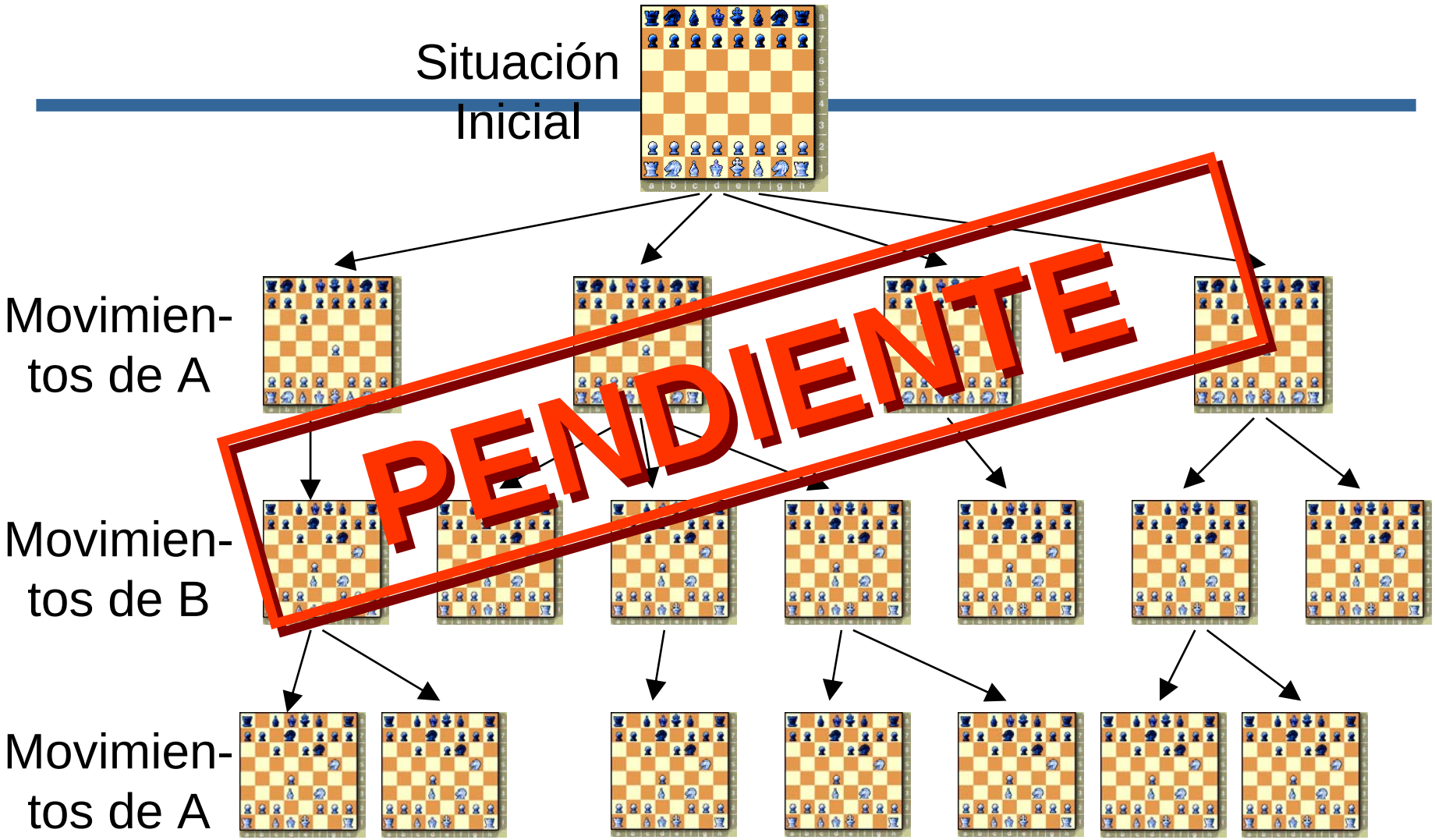
Jugador de Ajedrez

- Suponiendo que cada jugador hace unos 50 movimientos, el factor de ramificación medio es de 35 posibles movimientos.

- Tamaño del árbol: $35^{100} = 2{,}5 \cdot 10^{154}$

- ¡¡Sólo existen 10^{87} partículas subatómicas en el universo!!

Jugador de Ajedrez

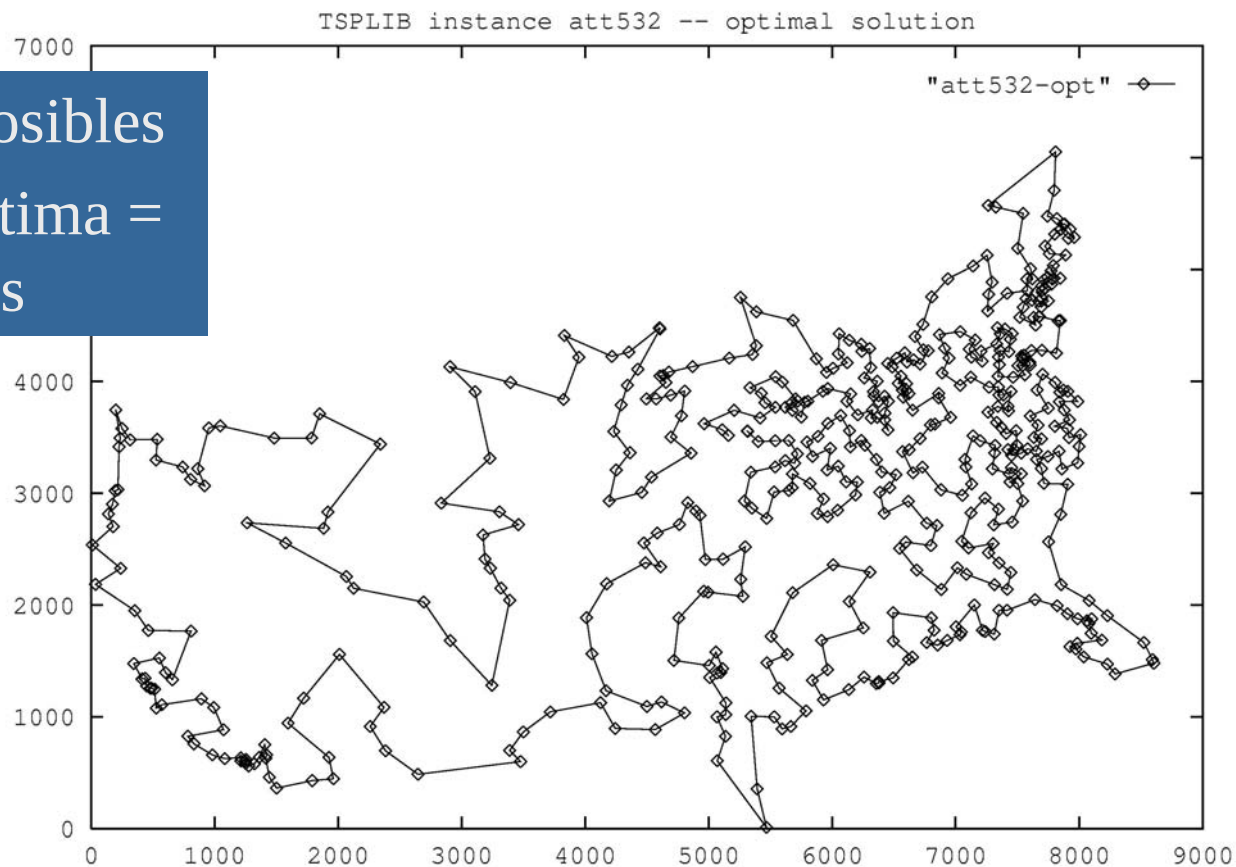


El problema del viajante

Ejemplo de Viajante de Comercio con 532 ciudades

532! soluciones posibles

Coste solución óptima = 27.686 millas



Problema del viajante

Resumen

- **Técnicas de diseño** de algoritmos:
  1. Divide y vencerás.
  2. Algoritmos voraces.
  3. Programación dinámica.
  4. Backtracking. Ramificación y poda.

- **Dado un problema:** seleccionar la técnica, seguir el proceso/esquema algorítmico, obtener el algoritmo y comprobarlo.

- **Recordar:** No empezar tecleando código como locos.

Algorítmica

Tema 1. Planteamiento General

Tema 2. La Eficiencia de los Algoritmos

Tema 3. Algoritmos "Divide y vencerás"

Tema 4. Algoritmos Voraces ("Greedy")

Tema 5. Algoritmos basados en Programación Dinámica

Tema 6. Algoritmos para la Exploración de Grafos ("Backtracking", "Branch and Bound")

Tema 7. Otras metodologías algorítmicas