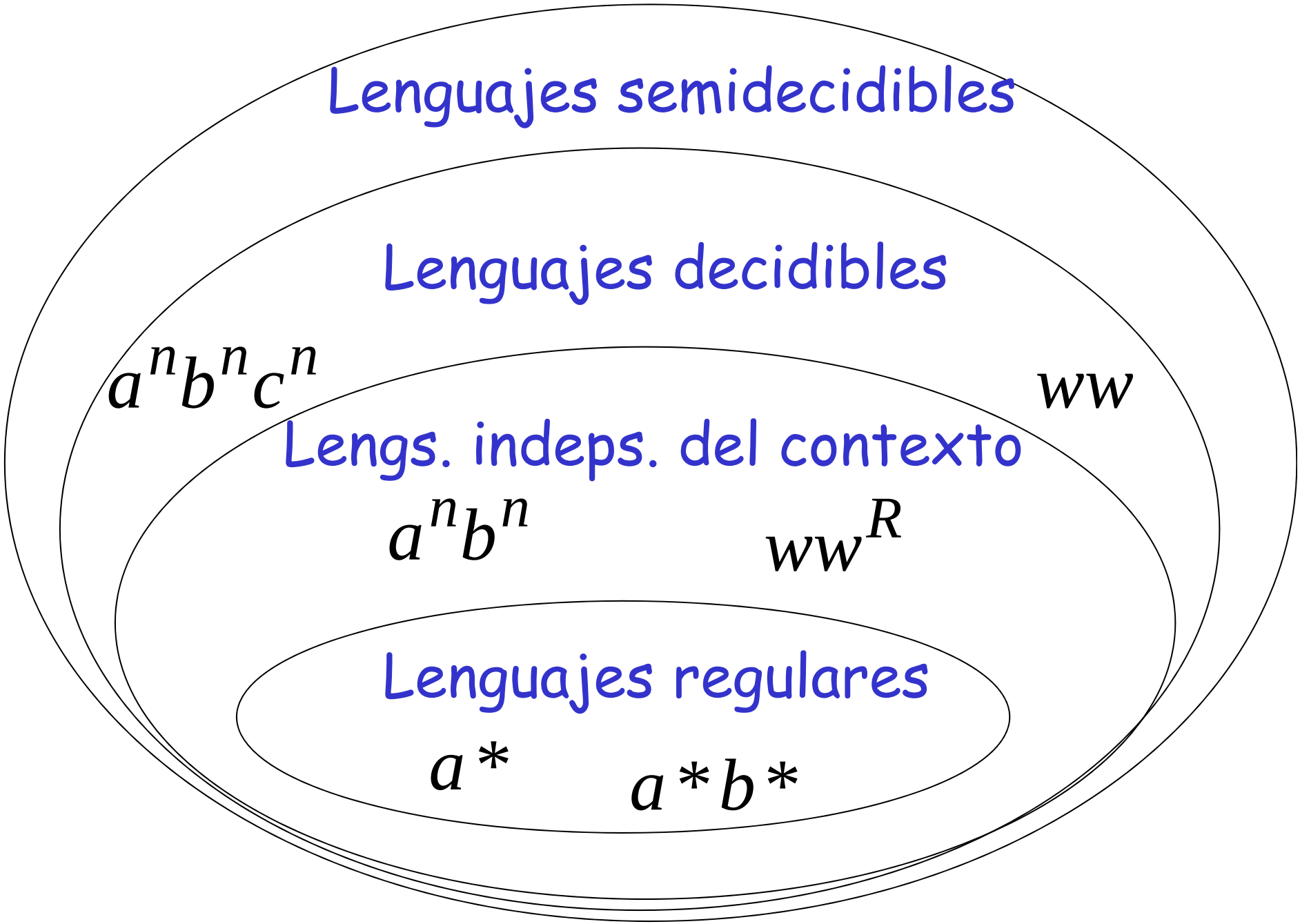
NP-completos

Elvira Mayordomo
Universidad de Zaragoza


Lenguajes semidecidibles
Lenguajes decidibles
$a^n b^n c^n$
ww
Lengs. indeps. del contexto
$a^n b^n$
ww^R
Lenguajes regulares
$a*$
$a*b*$

De la semana pasada: P y EXP

- P y EXP son **clases de lenguajes**.

- P es el conjunto de lenguajes (problemas) resolubles en tiempo polinómico.

- EXP es el conjunto de lenguajes (problemas) resolubles en tiempo exponencial.

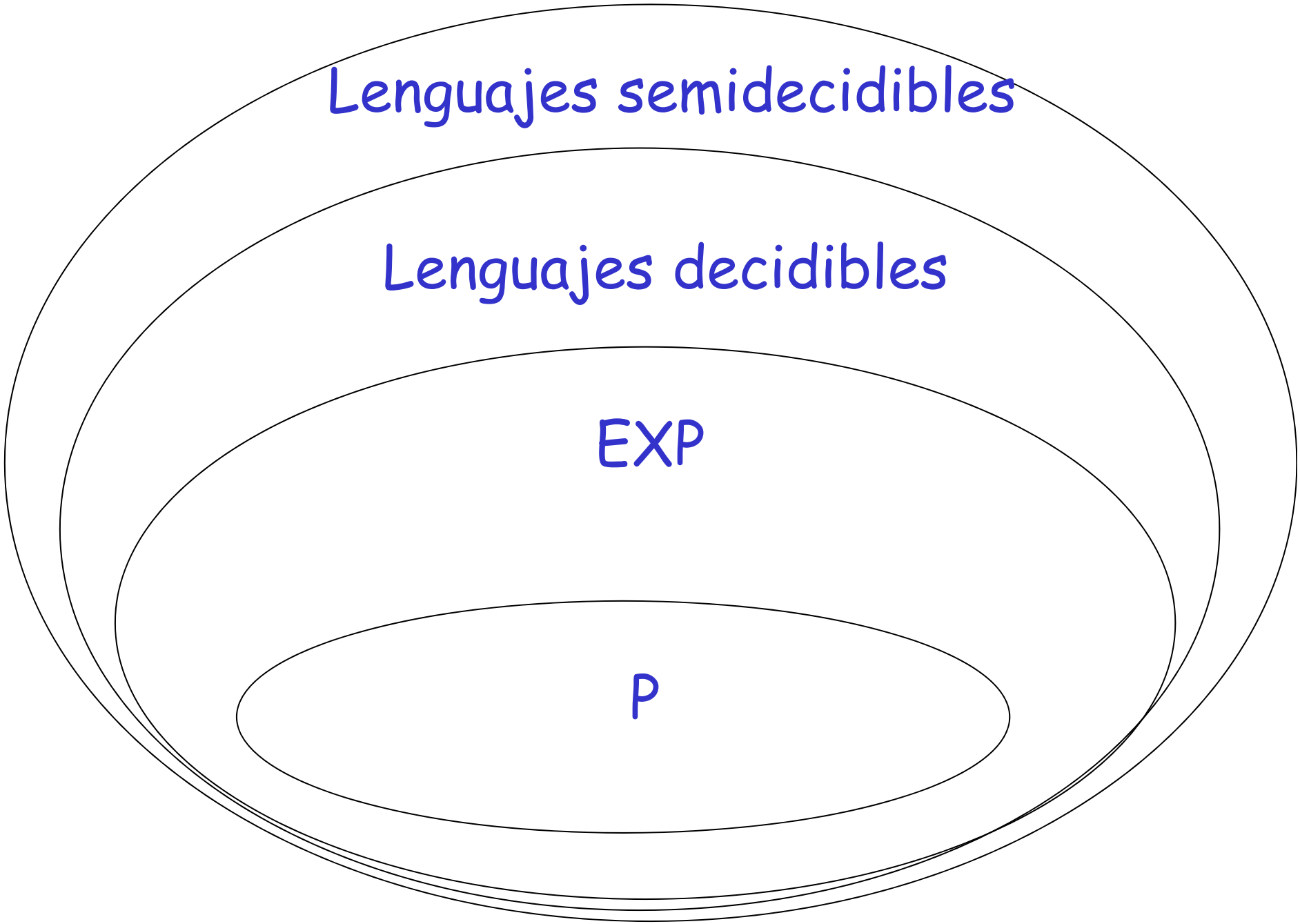
- $P \subseteq EXP$

Lenguajes decidibles

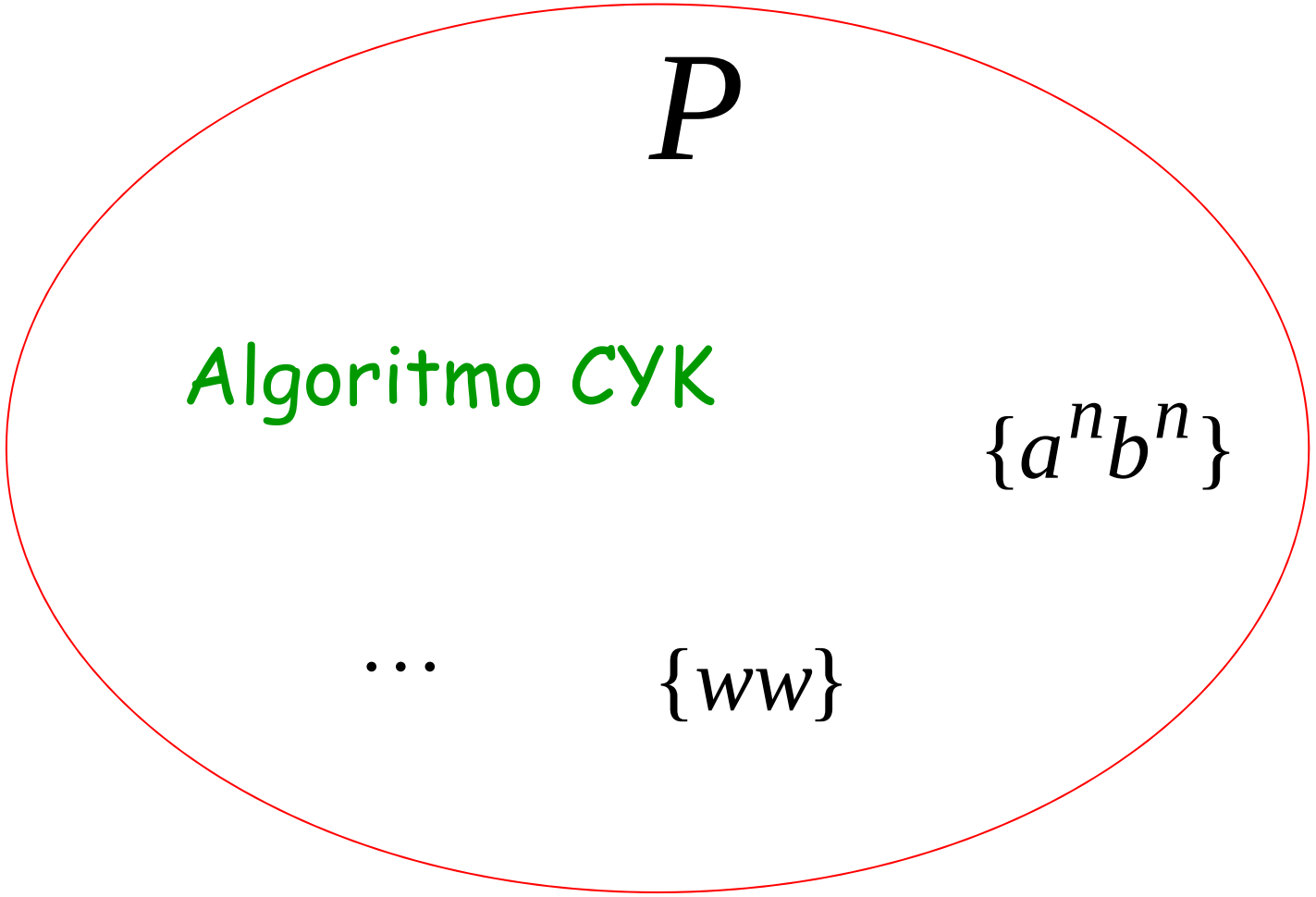
Un lenguaje L es decidible si es el aceptado por un programa que para siempre

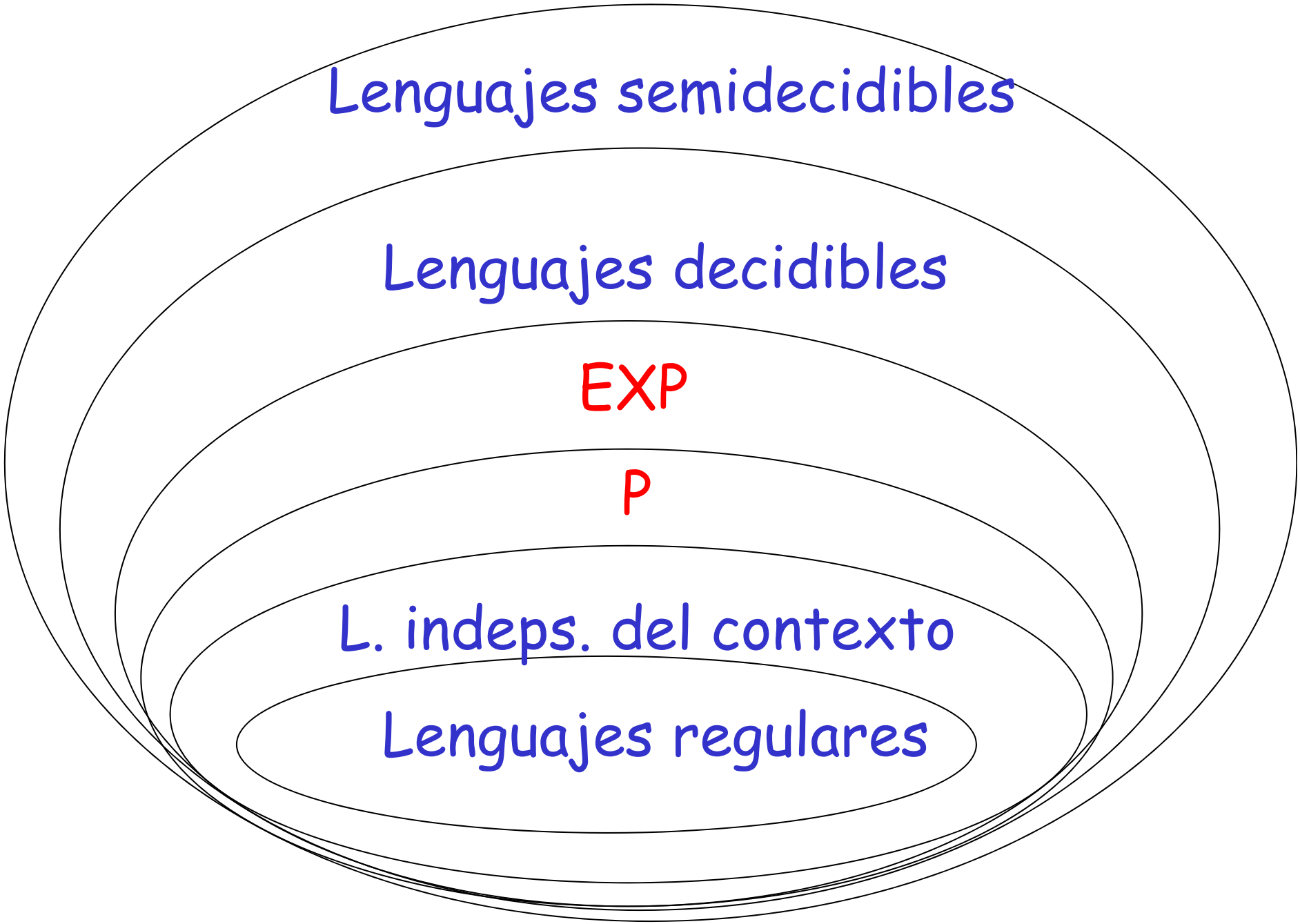
En otras palabras:

Un lenguaje L es decidible si existe un algoritmo que resuelve completamente el problema de pertenencia


Lenguajes semidecidibles
Lenguajes decidibles
EXP
P

Por otro lado




Lenguajes semidecidibles
Lenguajes decidibles
EXP
P
L. indeps. del contexto
Lenguajes regulares

Hoy veremos

- NP
- NP-completos

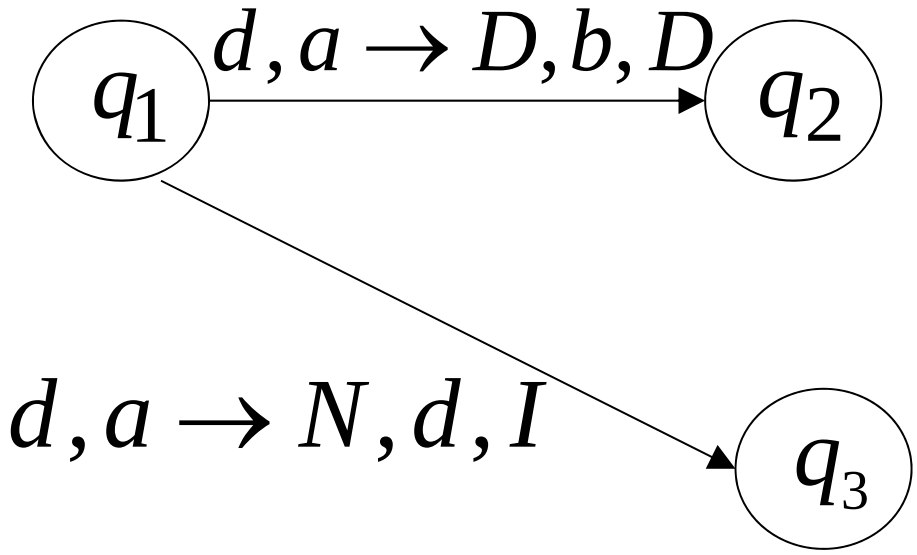
Muy poco formalizado …

NP

- NP es el conjunto de lenguajes resolubles en tiempo polinómico no determinista.

- ¿ Y qué es tiempo polinómico no determinista?
  - Usando máquinas de Turing no deterministas
  - Usando programas no deterministas

Máquina de Turing no determinista



Programa no determinista

- Incluye instrucciones "nodeterministas" como:

Para algún $x \in \{0,1\}^n$ hacer
  <secuencia de acciones>
fpalgun

NP-completos

- Son **los más difíciles** de NP

- No veremos la definición formal, sólo ejemplos y qué significa que un problema sea NP-completo.

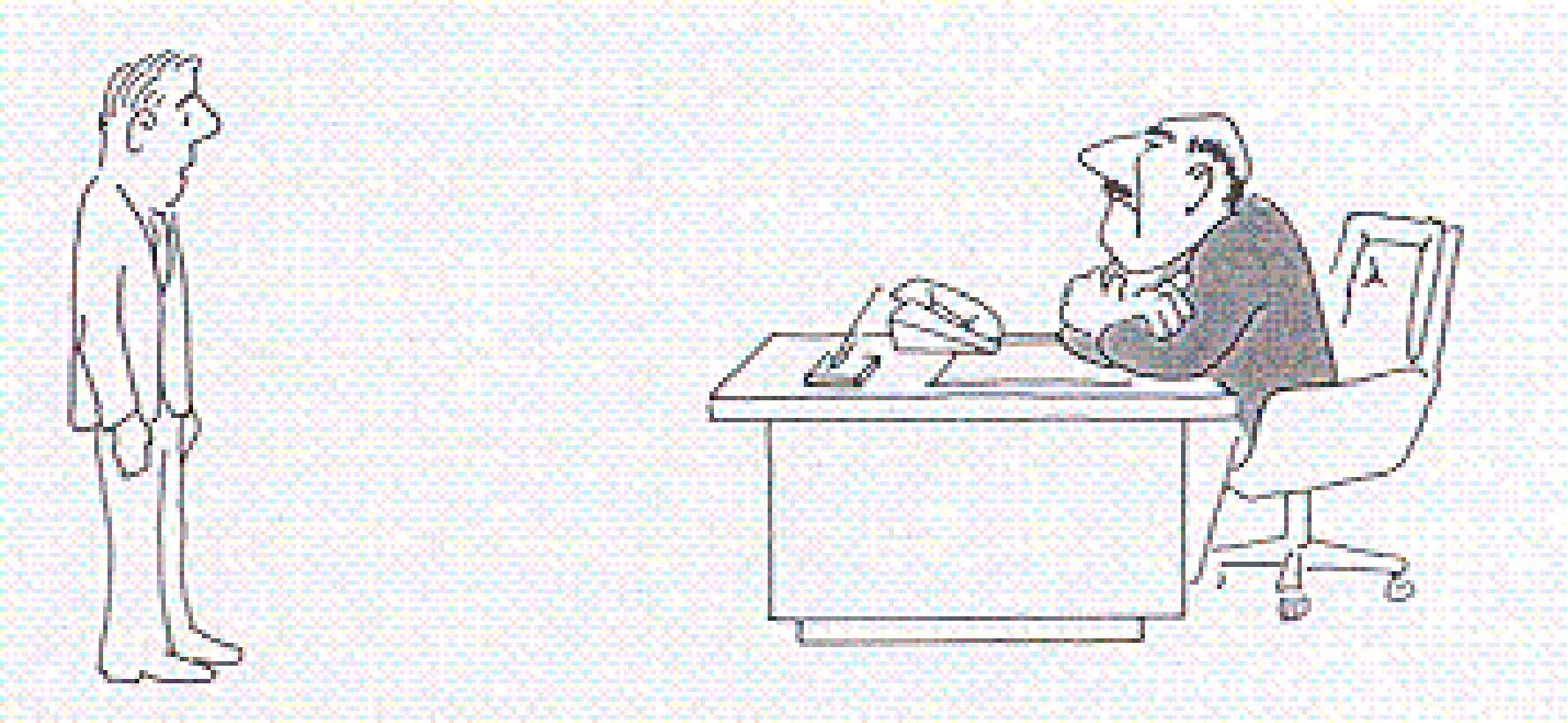
NP-completos

Una aplicación práctica

- Supón que tu jefe te pide que escribas un algoritmo eficiente para un problema extremadamente importante para tu empresa.

- Después de horas de romperte la cabeza sólo se te ocurre un algoritmo de "fuerza bruta", que analizándolo ves que cuesta tiempo exponencial.

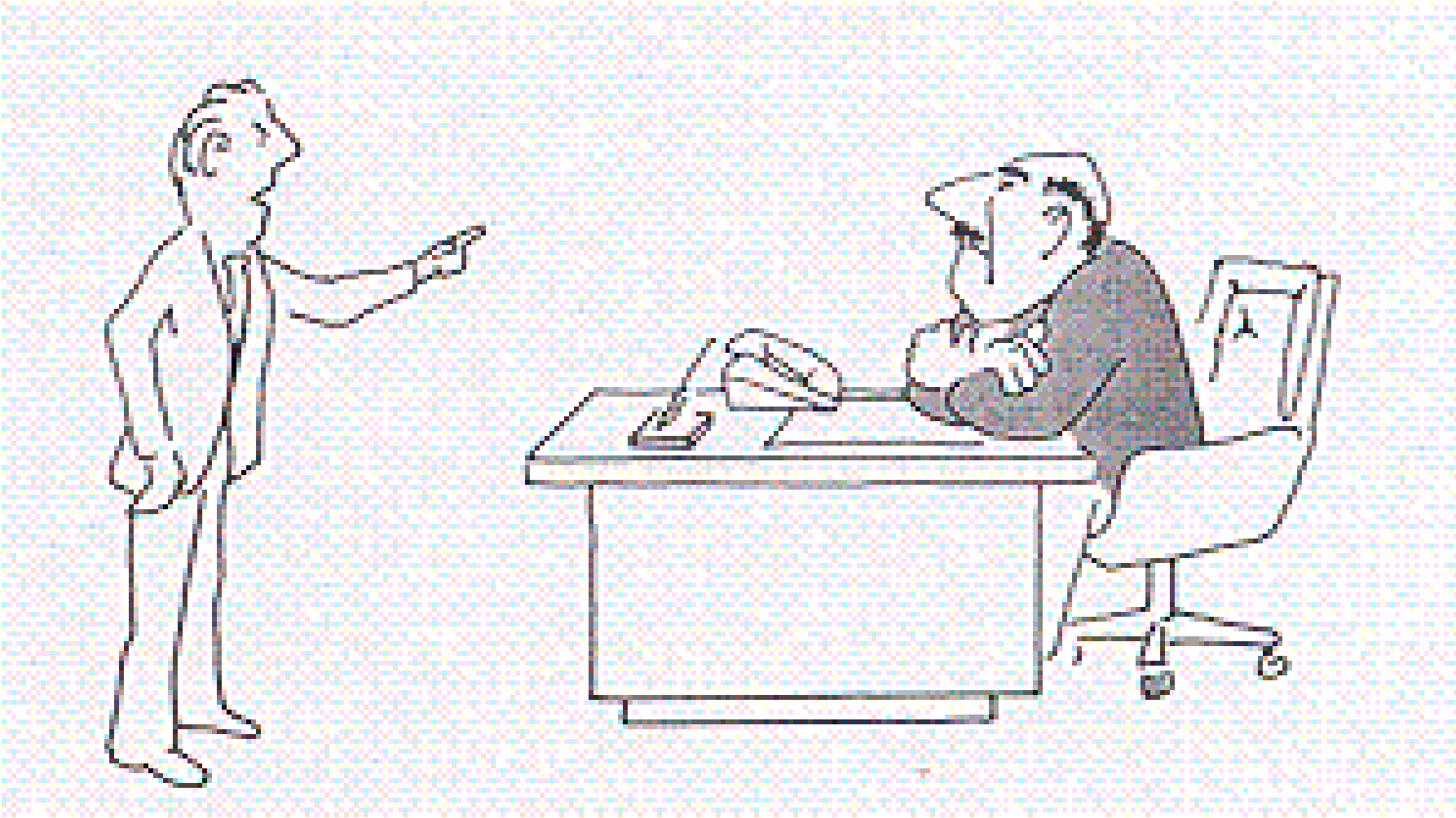
Una aplicación práctica



Te encuentras en una situación muy embarazosa:

"No puedo encontrar un algoritmo eficiente, me temo que no estoy a la altura"

Te gustaría poder decir …



"No puedo encontrar un algoritmo eficiente porque no existe"

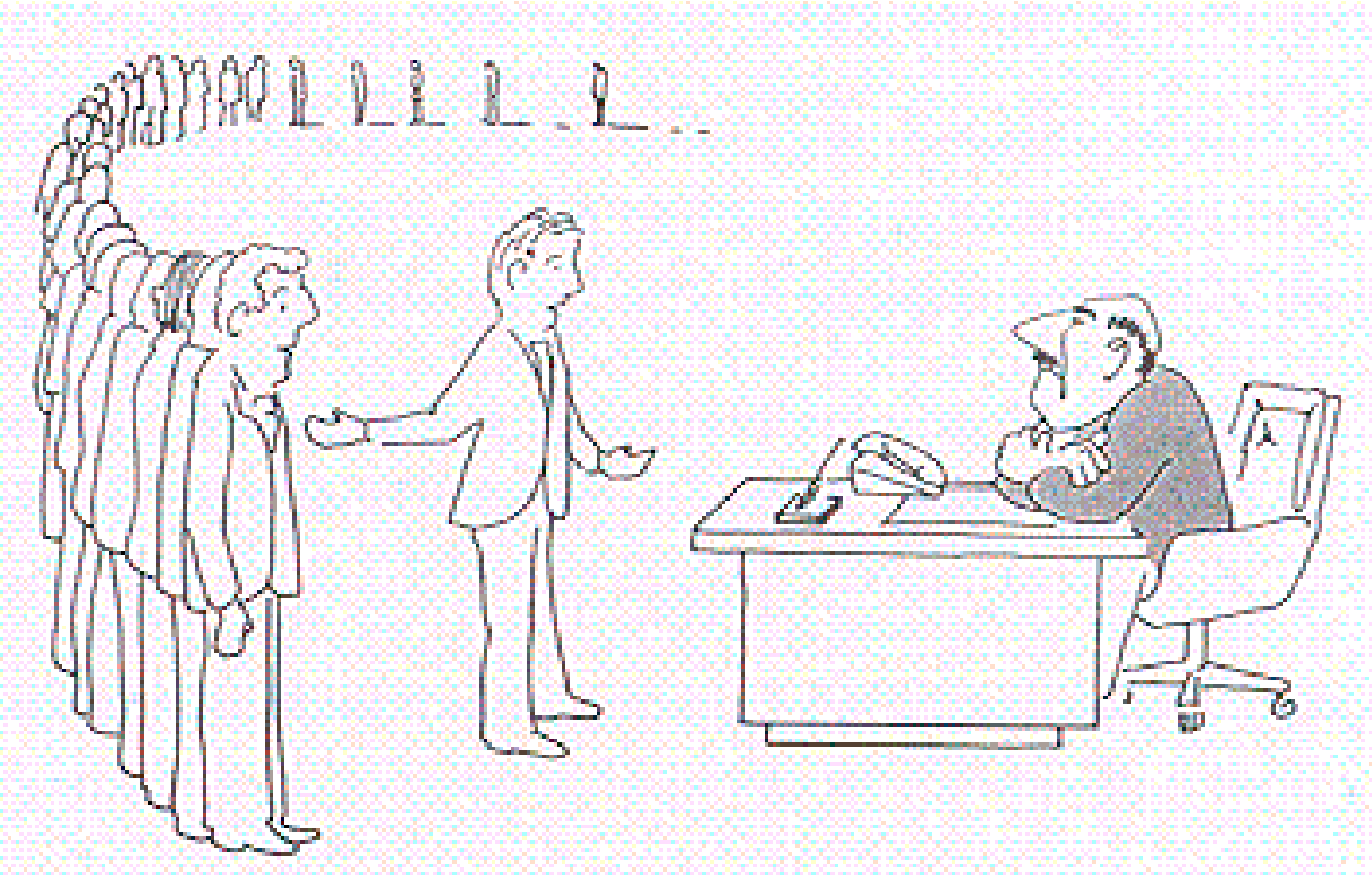
"No puedo encontrar un algoritmo eficiente porque no existe"

Eso es decir que el problema es intratable

- En realidad es muy poco frecuente poder decir algo tan tajante, para la mayoría de los problemas, es **muy difícil demostrar que son intratables**

- Pero la teoría de los NP-completos te puede ayudar a no perder tu trabajo diciendo …

Usando la teoría de los NP-completos …



"No puedo encontrar un algoritmo eficiente pero tampoco pueden ninguno de estos informáticos famosos"

O todavía mejor

"Si pudiera diseñar un algoritmo eficiente para este problema, ¡no estaría trabajando para usted! Me habría ganado el premio de un millón de dólares que da el Instituto Clay."

NP-completos

- Los NP-completos parecen intratables
- Nadie ha sabido demostrar que los NP-completos son intratables

- Son todos equivalentes, es decir:
  - Si se encuentra un algoritmo eficiente para un NP-completo entonces tenemos un algoritmo eficiente para cualquiera de ellos
  - Si probamos que un NP-completo no tiene algoritmos eficientes entonces ninguno los tiene.

Ejemplos de NP completos

- El problema del viajante de comercio
- El problema de hacer cierto un circuito booleano (Circuit-SAT)
- Asignación de procesadores (multiprocessor scheduling)

El problema del viajante

Dados:

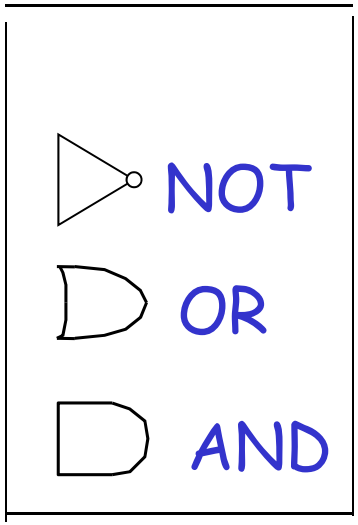
n el número de ciudades,

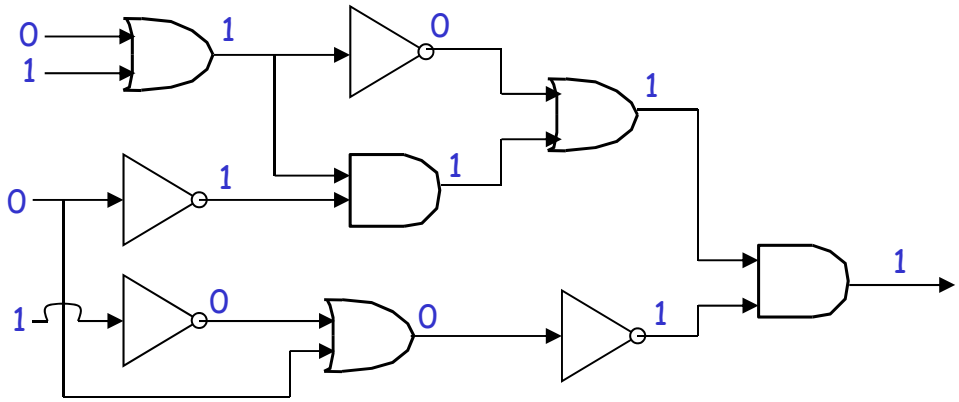
la matriz de distancias d(i,j) para 1<=i,j<=n,

una longitud máxima k

¿existe un camino de que pasa por todas las ciudades con longitud como máximo k?

Circuit-SAT





Dado: un circuito booleano con una única puerta de salida

¿Existe una asignación de valores a las entradas para la que la salida es "1"?

Multiprocessor scheduling (asignación de procesadores)

Dadas: n el número de tareas,

m número de procesadores

dur(i) la duración de la tarea I (i<=n)

T tiempo máximo (deadline)

¿Existe una asignación de las tareas a los procesadores **sin solapes** y cumpliendo el deadline?

Hay muchos más

Prácticamente de cualquier tema (redes, bases de datos, geometría, juegos)

Los problemas interesantes suelen ser NP-completos.

Una aplicación práctica

Después de la segunda respuesta (*No puedo encontrar un algoritmo eficiente pero tampoco pueden ninguno de estos informáticos famosos)*, tu jefe abandonará la búsqueda.

Pero la necesidad de una solución no desaparecerá.

Seguramente al demostrar que es NP-completo has aprendido mucho sobre el problema y ahora puedes:

- Olvidar lo de intentar encontrar un algoritmo en tiempo polinómico para el problema.
- Buscar un algoritmo eficiente para un problema diferente relacionado con el original.
- O bien intentar usar el algoritmo exponencial a ver qué tal funciona con las entradas que te interesan.

Referencias

- Lista de NP-completos:

  GAREY, M. y JOHNSON. D.: Computers and Intractability: *A Guide to the Theory of NP-Completeness*. Freeman. 1978.

- Premio del instituto Clay:

www.claymath.org/prizeproblems/pvsnp.htm
www.claymath.org/prizeproblems/milliondollarminesweeper.htm

Un momento de publicidad

Si os gusta pensar en algoritmos/programas divertidos/ocurrentes/diferentes

Si queréis saber más sobre NP-completos

Si tenéis curiosidad sobre alguna de las partes de esta asignatura …

Un momento de publicidad

Echadle un vistazo a las asignaturas de la especialidad de Computación:

- Algoritmia básica
- Algoritmia para problemas difíciles
- Bioinformática
- Procesadores de Lenguajes
- Robótica
- Videojuegos
- …