Programación 3

Cursada 2023

Prof. Alejandra Schiavoni (ales@info.unlp.edu.ar)

Prof. Laura Fava (lfava@info.unlp.edu.ar

Prof. Pablo Iuliano (piuliano@info.unlp.edu.ar)

Objetivos de la materia

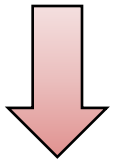
- Analizar algoritmos y evaluar su eficiencia
- Estudiar estructuras de datos avanzadas: su implementación y aplicaciones

Estructuras de Datos: Qué, Cómo y Por qué?

- Los programas reciben, procesan y devuelven datos
- Necesidad de organizar los datos de acuerdo al problema que vamos a resolver

Estructuras de Datos: Qué, Cómo y Por qué?

- Los programas reciben, procesan y devuelven datos
- Necesidad de organizar los datos de acuerdo al problema que vamos a resolver



Las estructuras de datos son formas de organización de los datos

Objetivos del curso respecto de las Estructuras de Datos

- Aprender a implementar las estructuras de datos usando abstracción
- Estudiar diferentes representaciones e implementaciones para las estructuras de datos
- Aprender a elegir la “mejor” estructura de datos para cada problema

Algoritmos y su Análisis

- ¿Qué es un algoritmo?
  - Es una secuencia de pasos que resuelven un problema
  - Es independiente del lenguaje de programación
- Existen varios algoritmos que resuelven correctamente un problema
- La elección de un algoritmo particular tiene un enorme impacto en el tiempo y la memoria que utiliza

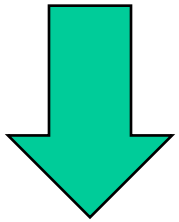
La elección de un algoritmo y de la estructura de datos para resolver un problema son interdependientes

Objetivos del curso respecto del Análisis de los Algoritmos

- Entender los fundamentos matemáticos necesarios para analizar algoritmos
- Aprender a comparar la eficiencia de diferentes algoritmos en términos del tiempo de ejecución
- Estudiar algunos algoritmos estándares para el manejo de las estructuras de datos y aprender a usarlos para resolver nuevos problemas

Problemas y algoritmos

- Problemas:
  - Buscar un elemento en un arreglo
  - Ordenar una lista de elementos
  - Encontrar el camino mínimo entre dos puntos



Encontrar **el algoritmo** que lo resuelve

Caso: Buscar un elemento en un arreglo

El arreglo puede estar:

- desordenado
- ordenado

Si el arreglo está **desordenado**

64	13	93	97	33	6	43	14	51	84	25	53	95
0	1	2	3	4	5	6	7	8	9	10	11	12

Caso: Buscar un elemento en un arreglo

El arreglo puede estar:

- desordenado
- ordenado

Si el arreglo está **desordenado** → **Búsqueda secuencial**

64	13	93	97	33	6	43	14	51	84	25	53	95
0	1	2	3	4	5	6	7	8	9	10	11	12

Algoritmo: Búsqueda secuencial

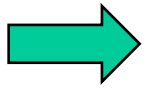
```
public static int seqSearch(int[] a, int
   key)
{
   int index = -1;
   for (int i = 0; i < N; i++)
       if (key == a[i])
              index = i;
   return index;
}
```

¿Cuántas comparaciones hace?

Caso: Buscar un elemento en un arreglo

El arreglo puede estar:

- desordenado
- ordenado

Si el arreglo está **ordenado** 

6	13	14	25	33	43	51	53	64	72	84	93	95	96	97
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14

Caso: Buscar un elemento en un arreglo

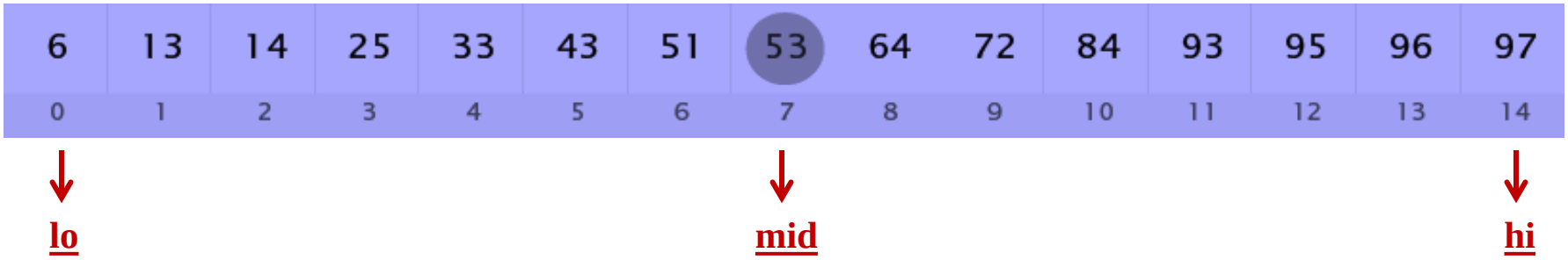
El arreglo puede estar:

- desordenado
- ordenado

Si el arreglo está **ordenado** ➡

Búsqueda binaria: Comparo la clave con la entrada del centro

- Si es menor, voy hacia la izquierda
- Si es mayor, voy hacia la derecha
- Si es igual, la encontré



Algoritmo: Búsqueda binaria

Adivinar número - Búsqueda lineal o binaria

https://es.khanacademy.org/computing/computer-science/algorithms/intro-to-algorithms/a/a-guessing-game

Algoritmo: Búsqueda binaria

```
public static int binarySearch(int[] a, int key)
{
   int lo = 0, hi = a.length-1;
   while (lo <= hi)
   {
       int mid = lo + (hi - lo) / 2;
       if (key < a[mid]) hi = mid - 1;
       else if (key > a[mid]) lo = mid + 1;
       else return mid;
   }
return -1;
}
```

¿Cuántas comparaciones hace?

¿Cuántas operaciones hace cada algoritmo?

Búsqueda secuencial

N	Cantidad de operaciones
1000	1000
2000	2000
4000	4000
8000	8000
16000	16000

Búsqueda binaria

N	Cantidad de operaciones
1000	~10
2000	~11
4000	~12
8000	~13
16000	~14

¿Cuántas operaciones hace cada algoritmo?

Búsqueda secuencial

N	Cantidad de operaciones
1000	1000
2000	2000
4000	4000
8000	8000
16000	16000

Hace N operaciones

Búsqueda binaria

N	Cantidad de operaciones
1000	~10
2000	~11
4000	~12
8000	~13
16000	~14

Hace log(N) operaciones

¿Cómo medir el tiempo?

✓**En forma empírica**

Se realiza a posteriori

✓**En forma teórica**

Se realiza a priori

Análisis empírico

Correr el programa para varios tamaños de la entrada y medir el tiempo. Suponemos que cada comparación tarda 1 seg.

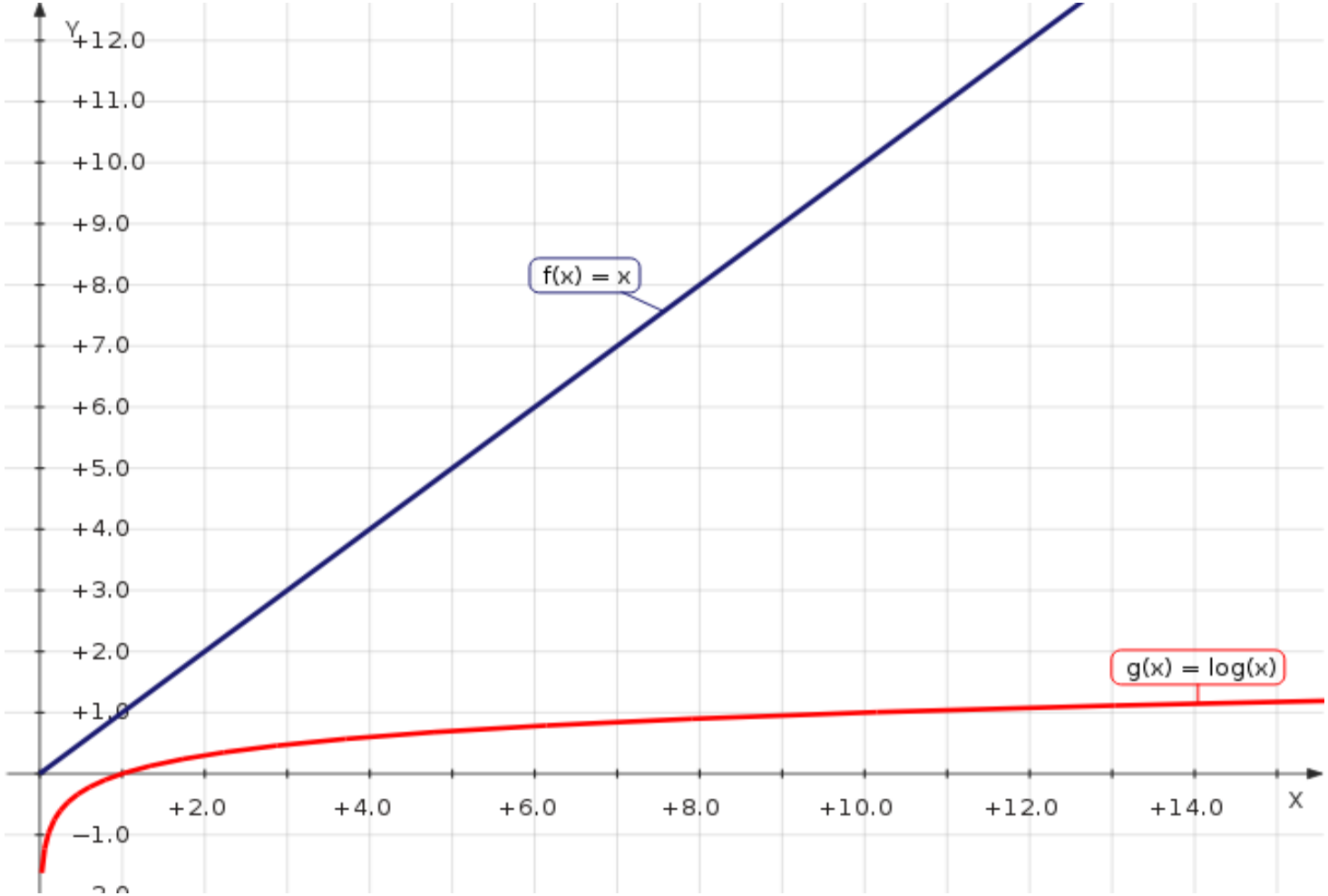
Búsqueda secuencial

N	Tiempo (seg)
1000	1000
2000	2000
4000	4000 ~ 1 hs.
8000	8000 ~ 2 hs
16000	16000 ~ 4 hs.

Búsqueda binaria

N	Tiempo (seg)
1000	~10
2000	~11
4000	~12
8000	~13
16000	~14

Análisis de Algoritmos



Cantidad de operaciones de la búsqueda binaria

Ejercitación

https://es.khanacademy.org/computing/computer-science/algorithms/binary-search/e/running-time-of-binary-search

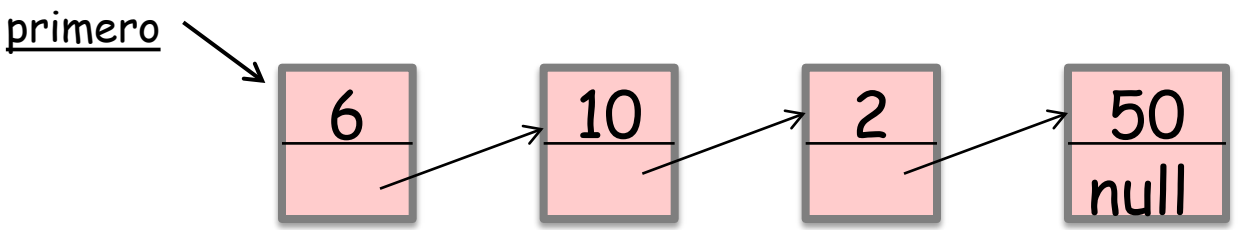
Caso: Buscar un elemento en una lista dinámica

Si los elementos están almacenados en una lista dinámica

La lista puede estar:

- desordenada
- ordenada

¿Cómo sería el algoritmo de búsqueda?



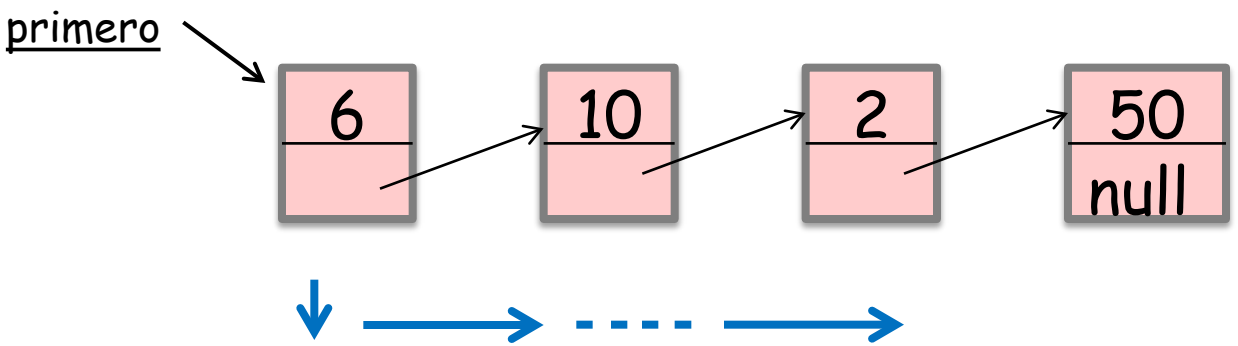
Caso: Buscar un elemento en una lista dinámica

Si los elementos están almacenados en una lista dinámica

La lista puede estar:

- desordenada
- ordenada

¿Cómo sería el algoritmo de búsqueda?



Caso: Buscar un elemento en una lista dinámica

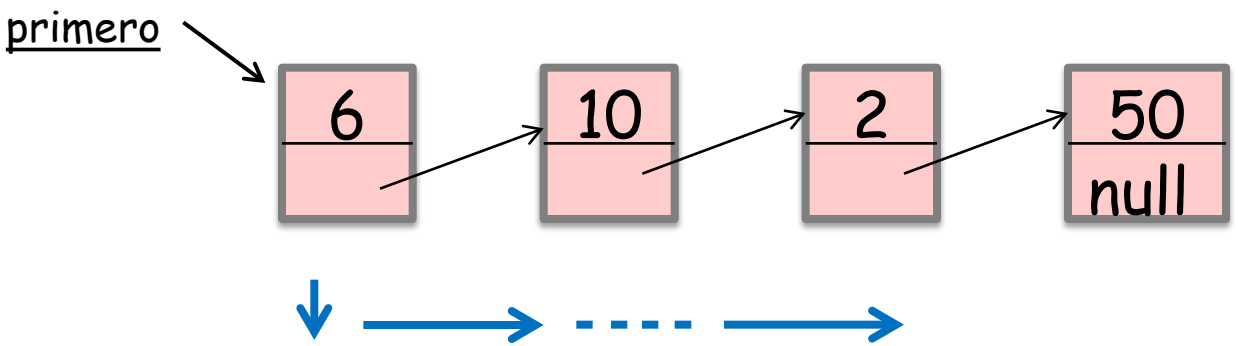
Si los elementos están almacenados en una lista dinámica

La lista puede estar:

- desordenada
- ordenada

¿Cómo sería el algoritmo de búsqueda?

¿Cuántas comparaciones hace?



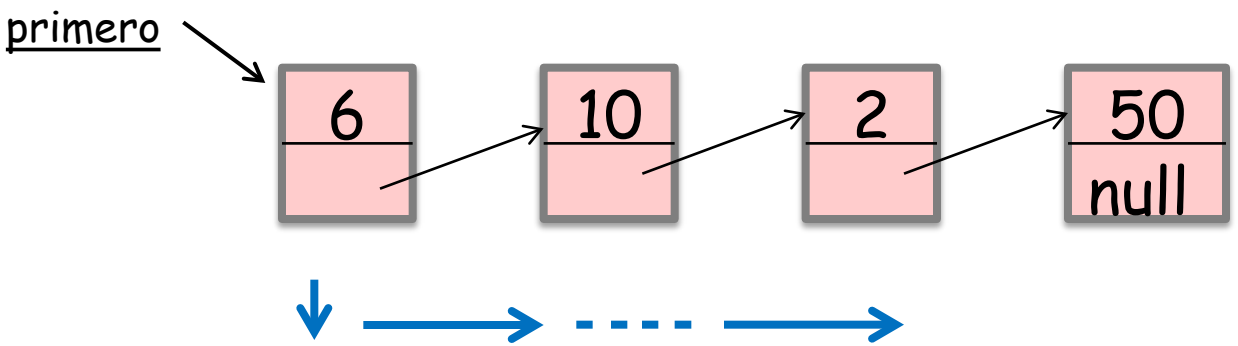
Caso: Buscar un elemento en una lista dinámica

Si los elementos están almacenados en una lista dinámica

La lista puede estar:

- desordenada
- ordenada

¿Cómo sería el algoritmo de búsqueda?



¿Cuántas comparaciones hace?

Hace N comparaciones

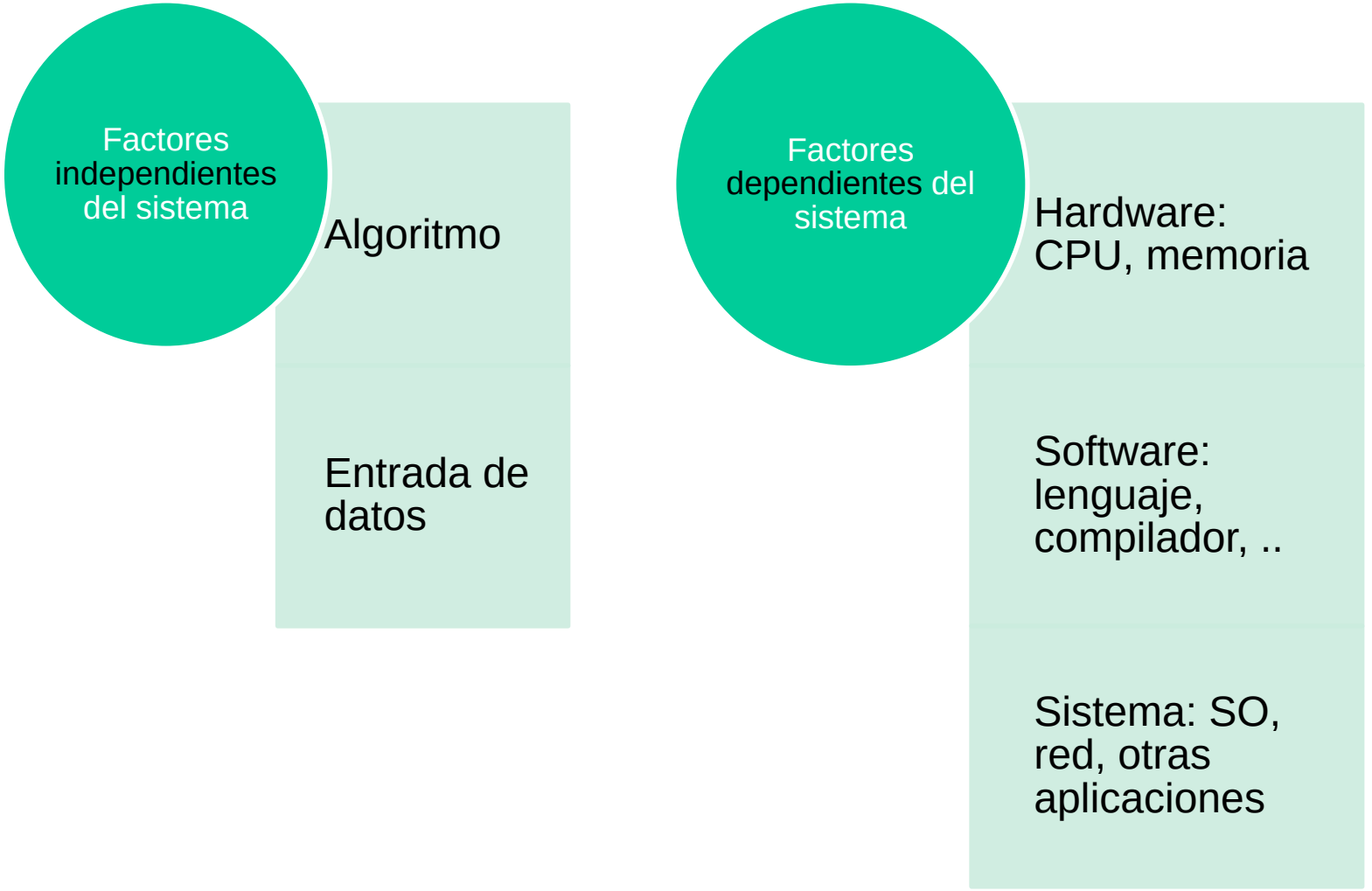
Análisis de Algoritmos

Marco para predecir la performance y comparar algoritmos

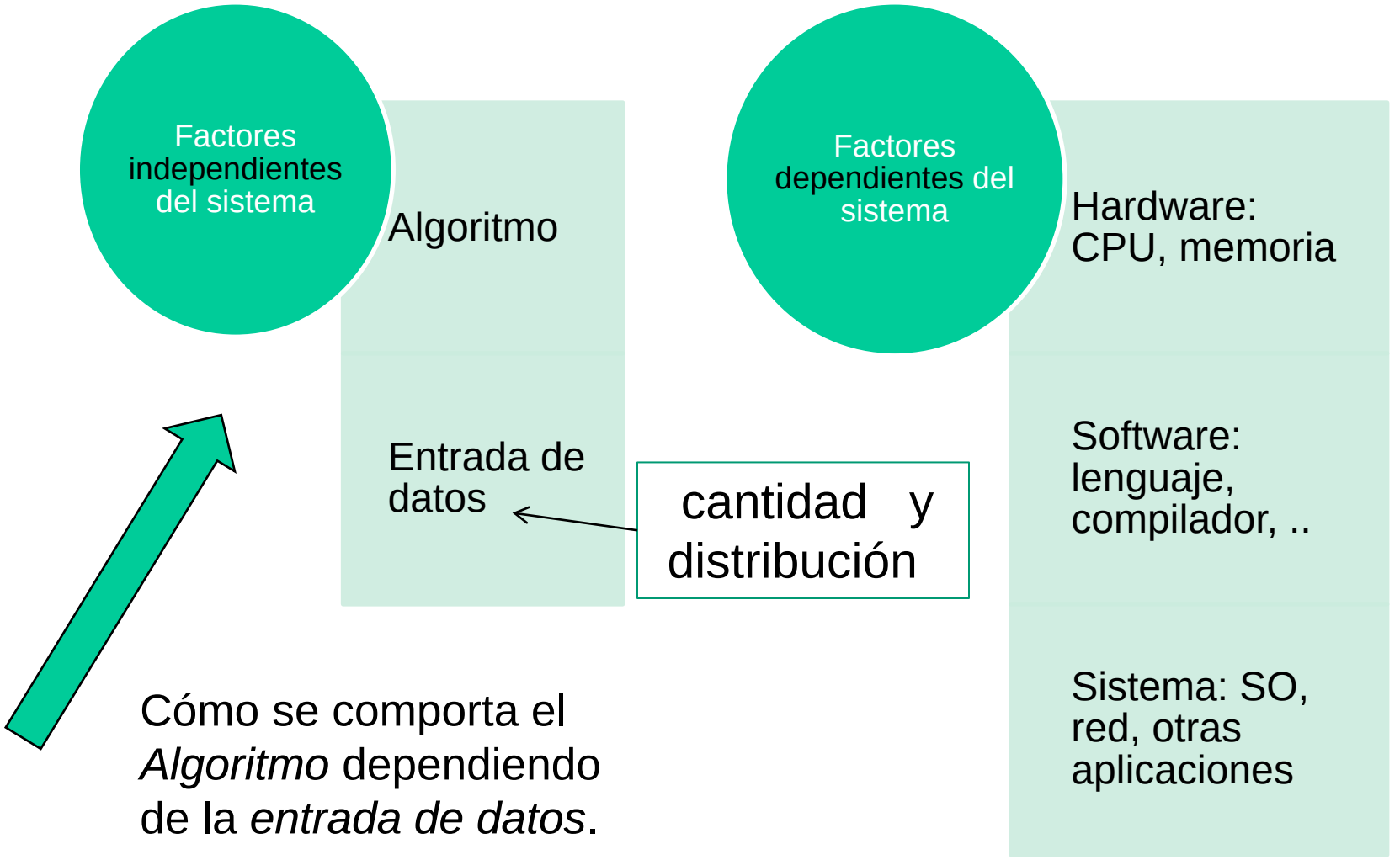
Desafío:

Escribir programas que puedan resolver en forma eficiente problemas con una gran entrada de datos

Análisis de Algoritmos



Análisis de Algoritmos



Análisis de Algoritmos

Existe un modelo matemático para medir el tiempo

Tiempo total:

*Suma del **costo** x **frecuencia** de todas las operaciones*

• Es necesario analizar el algoritmo para determinar el conjunto de operaciones

- **Costo** depende de la máquina, del compilador, del lenguaje
- **Frecuencia** depende del algoritmo y de la entrada