Herramientas para el procesamiento de textos en Python

Profesor Titular

Martín Kondratzky

Colaboradores

Fernando Carranza

Macarena Fernández Urquiza

Fernando Schiaffino

Colaboradores invitados

Julia Milanese, Federico Alvarez

Catalina Rubio

Victoria Colombo

Pablo Ceballos

La clase de hoy

Número de clase: Clase 2
Profesor: Fernando Carranza
Contacto: fernandocarranza86@gmail.com
Contenidos

- Problemas y lenguajes formales. Jerarquía de lenguajes formales de Chomsky.
- Teoría de la complejidad. Costo computacional: tiempo lineal, polinómico y exponencial

Una introducción

A lo largo de esta cursada vamos a lidiar con textos escritos en lenguaje natural. A diferencia de lo que ocurre en un formulario estandarizado, en un texto escrito en lenguaje natural no es posible saber *a priori* en dónde encontrar la información que uno busca. Por esta razón, los textos representan fuentes de datos no estructurados. Para poder enfrentar al problema de buscar información en esta clase de datos, es sumamente necesario primero conocer la naturaleza de estos datos.

La doble articulación de lenguaje

"La primera articulación del lenguaje es aquella con arreglo a la cual todo hecho de experiencia que se vaya a transmitir, toda necesidad que se desee hacer conocer a otra persona, se analiza en una sucesión de unidades, dotadas cada una de una forma vocal y de un sentido".

(Martinet: 22)

"Cada una de estas unidades de la primera articulación presenta, como hemos visto, un sentido y una forma vocal (o fónica). Pero no puede ser analizada en unidades sucesivas más pequeñas dotadas de sentido. El conjunto *cabeza* quiere decir "cabeza" y no se puede atribuir a ca-, -be-, -za, sentidos distntos cuya suma sea equivalente a "cabeza". Pero la forma vocal es analizable en una sucesión de unidades, cada una de las cuales contribuye a distinguir *cabeza* de otras unidades como *cabete*, *majeza* o *careza*. Es a esto a lo que se designará como la segunda articulación del lenguaje."

(Martinet 1991 [1960]: 24)

En términos más familiares

- Léxico = Primera articulación
- Fonología = Segunda articulación.

"En el hablar corriente, 'el lenguaje' designa propiamente la facultad que tienen los hombres de entenderse por medio de signos vocales. Merece la pena detenerse en este carácter vocal del lenguaje. En los países civilizados, desde hace algunos milenios se hace uso con mucha frecuencia de signos pictóricos o gráficos que corresponden a los signos vocales del lenguaje. Esto es lo que se llama escritura. Hasta la invención del fonógrafo, todo signo vocal emitido era percibido inmediatamente o quedaba perdido para siempre. Por el contrario, un signo escrito, duraba tanto cuanto durara su soporte: piedra, pergamino o papel, y los rasgos dejados sobre este soporte por el buril, el estilo o la pluma."

Si nos centramos en la segunda articulación del lenguaje en la modalidad oral, vemos que las unidades relevantes son los llamados fonemas, esto es, los sonidos distintivos que utiliza cada lengua.

Si nos centramos en la segunda articulación del lenguaje en la modalidad oral, vemos que las unidades relevantes son los llamados fonemas, esto es, los sonidos distintivos que utiliza cada lengua. Ahora bien, como vamos a trabajar con textos escritos en lenguas de escritura alfabética (no vamos a hacer síntesis de habla), las unidades relevantes serán los grafemas. Denominaremos alfabeto al conjunto no vacío de símbolos que constituya esta segunda articulación del lenguaje. El alfabeto se designa convencionalmente con la letra Σ

Por ejemplo:

1. ALFABETO-LATINO = {a, b, c, d...}
2. NÚMEROS-NATURALES={1, 2, 3, 4...}

Por ejemplo:

1. ALFABETO-LATINO = {a, b, c, d...}
2. NÚMEROS-NATURALES={1, 2, 3, 4...}

Puesto que todos los textos con los que vamos a trabajar están en computadora, tenemos que prestar atención a cuál es el sistema de codificación de caracteres frente al cual nos estamos enfrentando:

- ASCII
- UTF-8

Cadenas

La concatenación de símbolos (iguales o diferentes) de un alfabeto Σ se conoce con el nombre de cadena. En términos de Martinet, la cadena equivale a la primera articulación del lenguaje humano.

Cadenas

- Todas las cadenas de determinada longitud k que se pueden construir con un alfabeto Σ se representan convencionalmente Σ^k

  Por ejemplo, dado el alfabeto $\Sigma = \{a, b\}$, se dan las siguientes extensiones:

  $\Sigma^0 = \{\emptyset\}$

  $\Sigma^1 = \{a, b\}$

  $\Sigma^2 = \{aa, ab, ba, bb\}$

  $\Sigma^3 = \{aaa, aab, abb, aba, bbb, bba, baa, bab\}...$

Cadenas

- Todas las cadenas de determinada longitud k que se pueden construir con un alfabeto Σ se representan convencionalmente Σ^k

  Por ejemplo, dado el alfabeto $\Sigma = \{a, b\}$, se dan las siguientes extensiones:
  $\Sigma^0 = \{\emptyset\}$
  $\Sigma^1 = \{a, b\}$
  $\Sigma^2 = \{aa, ab, ba, bb\}$
  $\Sigma^3 = \{aaa, aab, abb, aba, bbb, bba, baa, bab\}...$

- Para representar el conjunto de todas las cadenas posibles que se pueden obtener a partir de un alfabeto Σ se usa la notación Σ^*.

  En términos de teoría de conjuntos,
  $\Sigma^* = \{\Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \cup \Sigma^4 \cup ...\}$

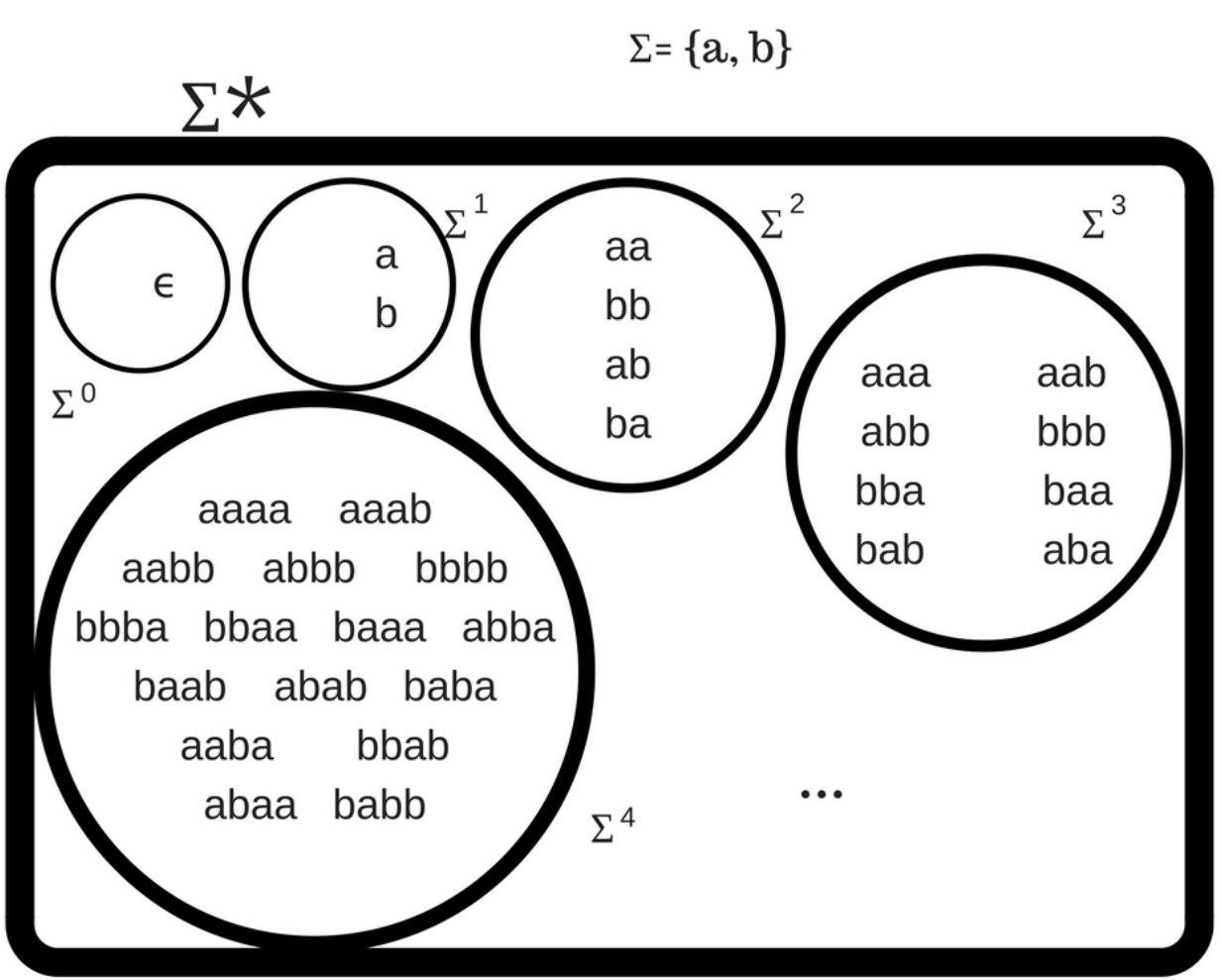

Figura: Imagen de ejemplo de las cadenas posibles generadas por un alfabeto $\Sigma = \{a, b\}$

Si contamos la u con diéresis y las vocales acentuadas como caracteres distintos de las no acentuadas, el español tiene 33 caracteres (solo minúsculas y sin contar signos de puntuación). Supongamos que estos 33 caracteres conforman el alfabeto Σ. Ahora bien, Σ^* incluye una infinita cantidad de cadenas que no forman parte del español, como por ejemplo dkfjhg o tuqpeigh

Un lenguaje L es un conjunto de cadenas particularmente relevante que está incluido en Σ^*.

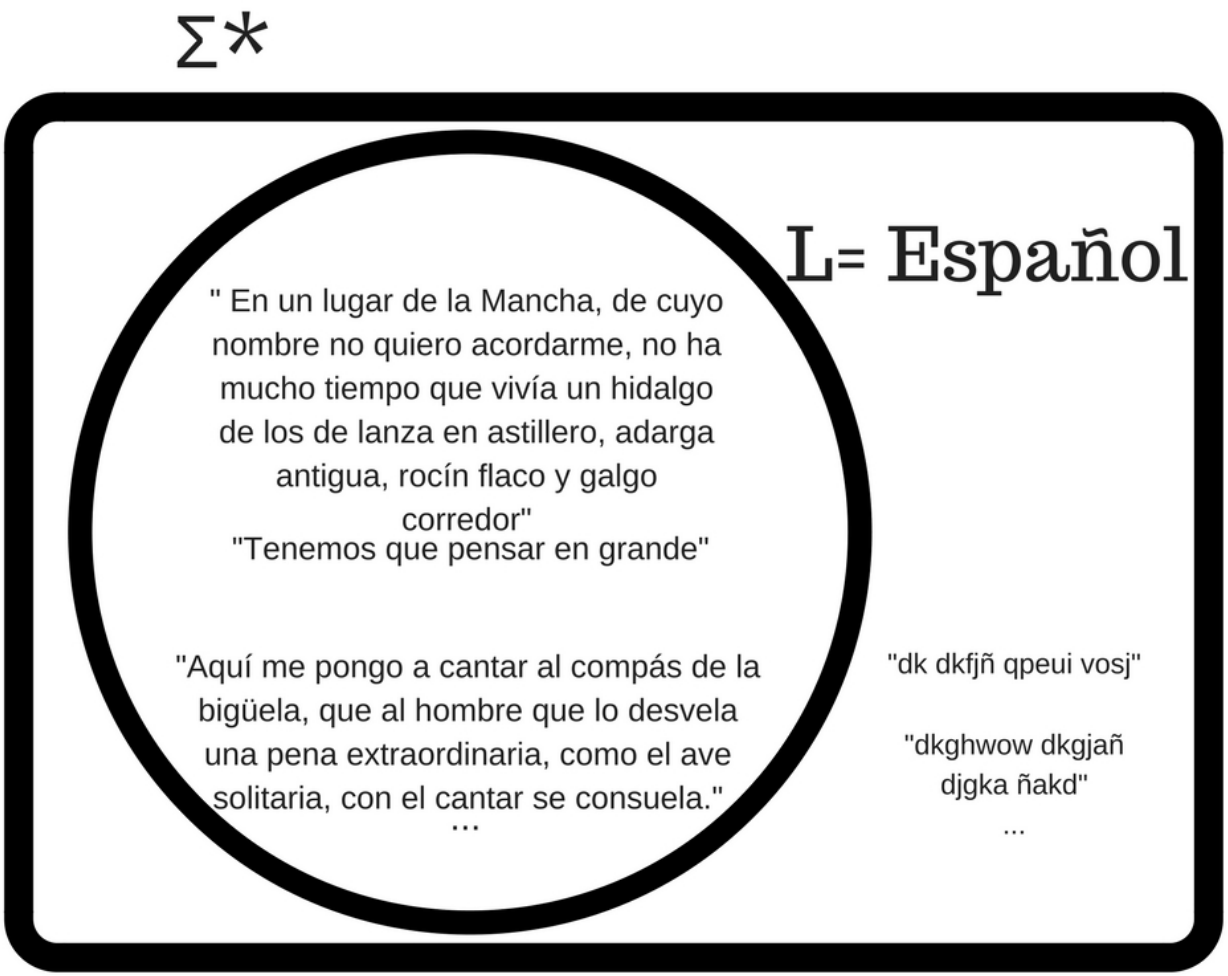

Figura: Lenguaje español como subconjunto de Σ^* para el alfabeto $\Sigma = \{a, b\}$

Supongamos el conjunto Σ^2.

Supongamos el conjunto Σ^2.

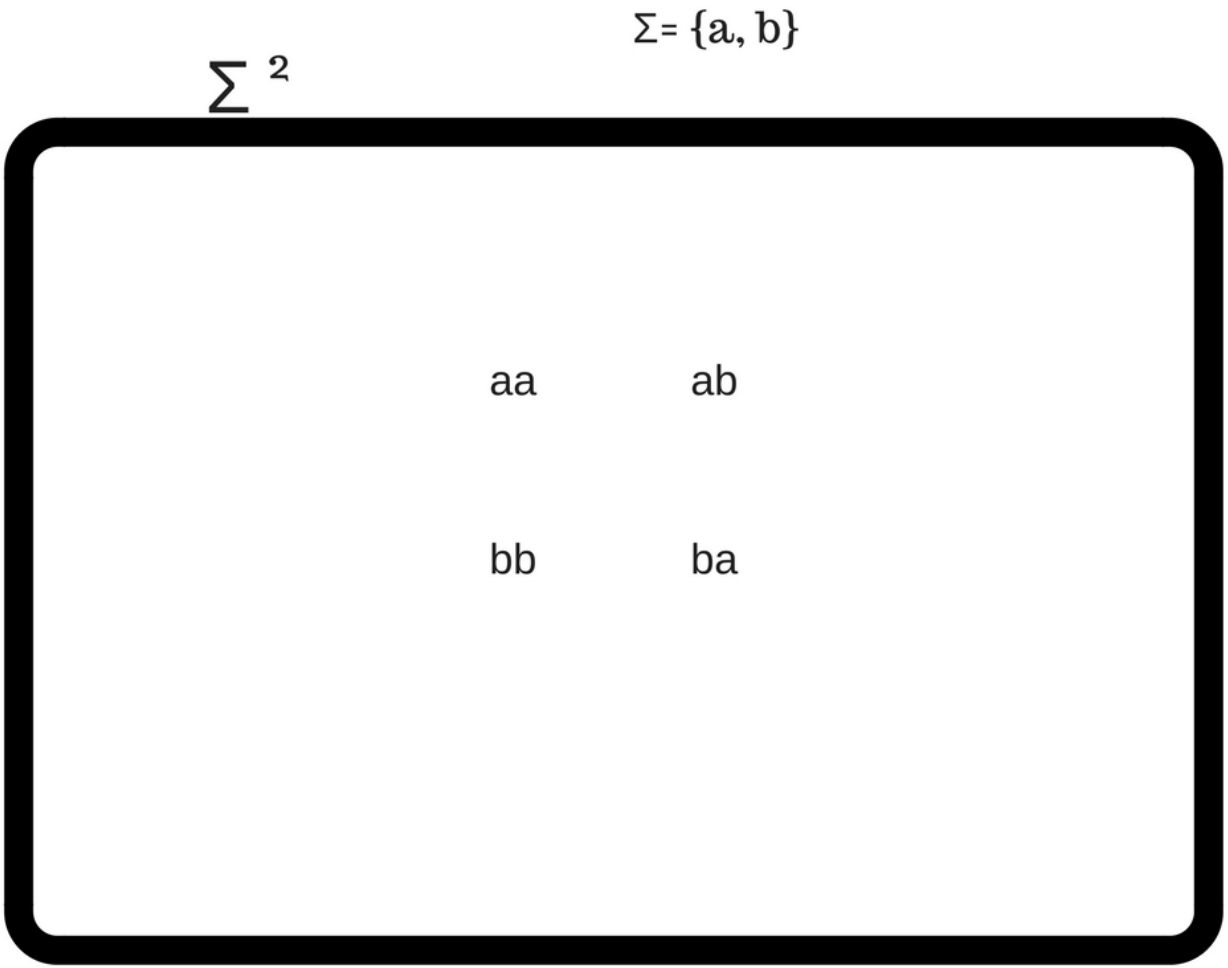

Figura: Σ^2 para el alfabeto $\Sigma = \{a, b\}$

El conjunto de todos los lenguajes posibles de Σ^2 es igual al superconjunto de Σ^2, o sea, $P(\Sigma^2)$

El conjunto de todos los lenguajes posibles de Σ^2 es igual al superconjunto de Σ^2, o sea, $P(\Sigma^2)$

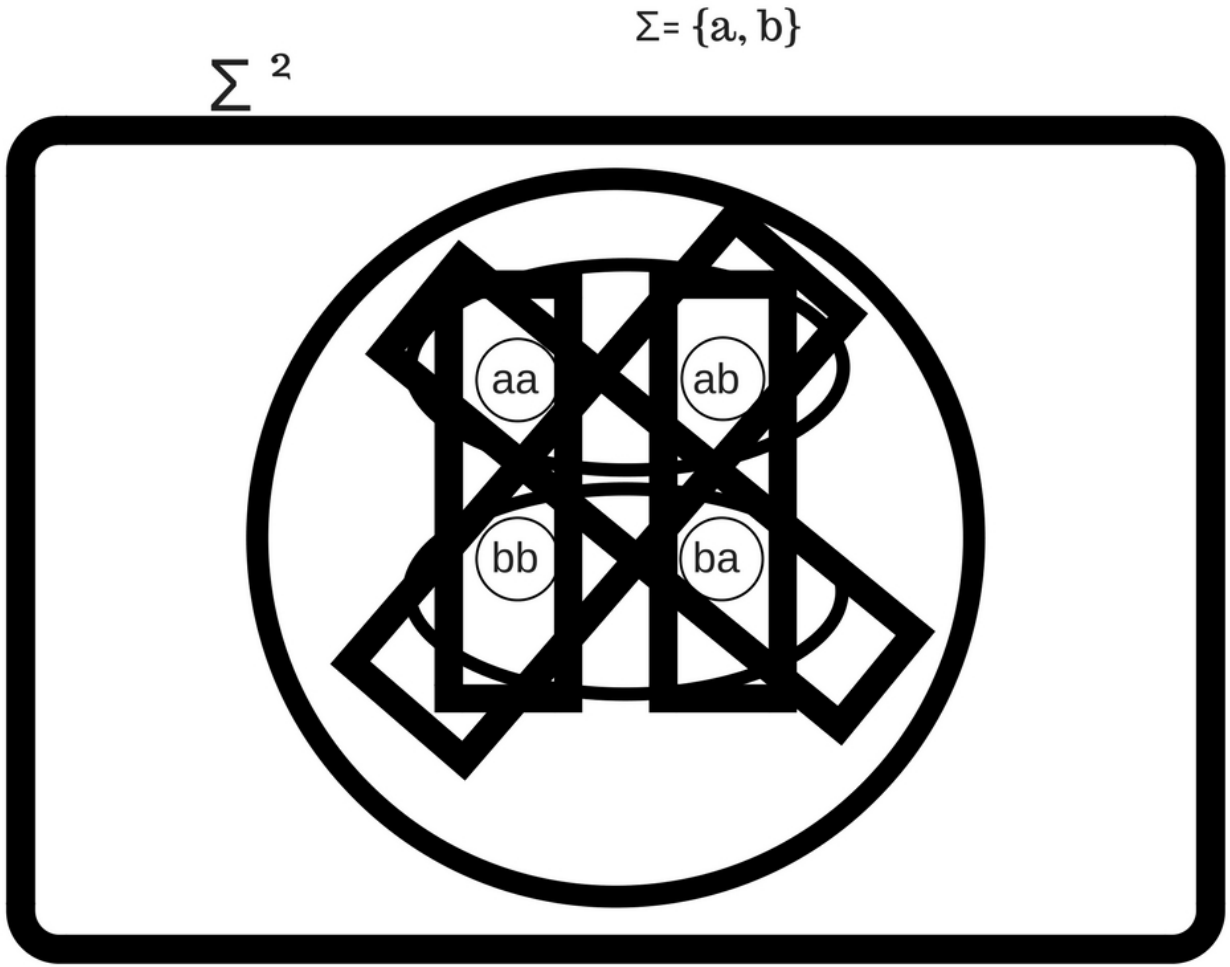

Figura: Σ^2 para el alfabeto $\Sigma = \{a, b\}$

Esto se puede representar en forma de retículo de la siguiente forma.

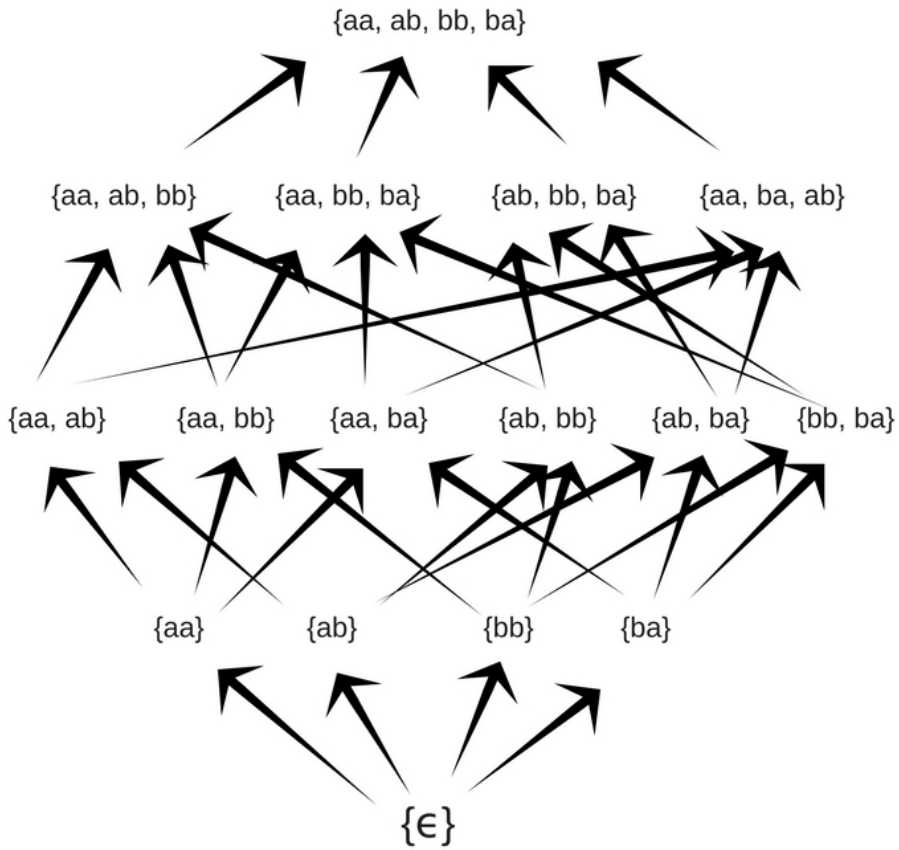

Figura: Σ^2 para el alfabeto $\Sigma = \{a, b\}$

El conjunto de todos los lenguajes posibles de Σ^* es igual al superconjunto de la unión de todas las cadenas posibles a partir del alfabeto Σ. Es decir, el conjunto de todos los lenguajes posibles es igual a $P(\Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 ...)$.

En lingüística formal, se asume generalmente que una lengua es un conjunto de oraciones formadas a partir de un vocabulario.

Como el conjunto total de todas las oraciones no puede definirse por extensión, el desafío de la lingüística formal consiste en encontrar una forma de definirlo por intensión.

- **Lenguaje**: Conjunto de oraciones gramaticales incluido en el conjunto total de oraciones posibles. Se supone que es un conjunto recursivo (por cada oración es posible determinar si pertenece o no al conjunto). Dado que es potencialmente infinito, no puede ser definido por extensión.
- **Lengua-I**: Es el sistema intensional que posee cada hablante y que produce todas las oraciones gramaticales de una lengua y ninguna de las agramaticales. Reconstruir ese algoritmo es el objetivo principal de la lingüística formal.
- **Lengua-E**: Es el conjunto de las oraciones exteriorizadas. Existe cierta ambigüedad respecto de si coincide con la noción de lenguaje, si se trata del subconjunto L-E incluido en el lenguaje L formado por las oraciones que pertenecen al conjunto de las oraciones efectivamente exteriorizadas o si es un conjunto L-E cuya intersección con L es el conjunto de las oraciones gramaticales efectivamente exteriorizadas y el complemento son las oraciones agramaticales exteriorizadas ya sea por errores de actuación o por el motivo que fuere.

La noción de lenguaje se extiende no solamente a los lenguajes naturales sino también a cualquier conjunto de cadenas formadas a partir de un alfabeto Σ.

El mundo está plagado de problemas

- Hay desigualdad
- Me quedé pelado
- Necesito saber cuántos huevos tengo que comprar para hacer nueve panqueques.

Algunos de estos problemas pueden ser resueltos mediante una computadora.

Algunos de estos problemas pueden ser resueltos mediante una computadora.
Supongamos que queremos resolver el problema de la cantidad de panqueques. Al respecto, sé que por cada docena de panqueques se gastan tres huevos, dos tazas de leche y una taza de harina.
Puedo usar la computadora para resolver esto a la manera de una calculadora:

1. 9*3/12

Puedo hacer lo mismo usando variables en lugar de los números a secas:

1. cantidadhuevospordocena = 3
2. docena = 12
3. panqueques = 9
4. cantidadhuevosdeseada = panqueques*cantidadhuevospordocena/docena
5. print(cantidadhuevosdeseada)

El problema de cuántos huevos necesito para hacer nueve panqueques puede traducirse a una función f= cantidadhuevosdeseada.

El problema de cuántos huevos necesito para hacer nueve panqueques puede traducirse a una función f= cantidadhuevosdeseada.

- def funcionhuevos():
  y = 9*3/12
  print(y)

Ahora bien, esta función tiene el problema de que es poco útil si yo quiero saber cuántos huevos necesito para hacer cualquier número de panqueques distinto de 9. Resulta mucho más útil y portable una función en la que la cantidad de panqueques sea un parámetro. Podemos reescribir la función de la siguiente forma:

- def funcionhuevos(x):
  y = x*3/12
  return(y)

Y llamarla como print(funcionhuevos(9))

Según la terminología, el número 9 de panqueques en la primera función está "hardcodeado", es decir, está dado por supuesto dentro del código en lugar de estar sujeto a parametrización. El hardcodeado está visto en programación como una mala práctica.

Dado que funcionhuevos(x) tiene un solo parámetro, es una función unaria. Toda función unaria puede representarse en términos de un diagrama cartesiano con dos ejes y = dominio y x = imagen. Para que algo sea una función, a cada elemento del dominio le debe corresponder un solo elemento de la imagen (lo inverso no es necesario). Cuando un elemento del dominio se corresponde con más de un elemento de la imagen hablamos de relación en lugar de función.

Toda función unaria f(x)=y en la cual x $\in$ A e y $\in$ B equivale también a un conjunto de pares ordenados incluido en el producto cartesiano A X B.

Toda función unaria f(x)=y en la cual x $\in$ A e y $\in$ B equivale también a un conjunto de pares ordenados incluido en el producto cartesiano A X B.
En el caso de la función funcionhuevos(x), el producto cartesiano sería cantidaddehuevos X cantidaddepanqueques. Este conjunto de pares ordenados incluye el conjunto $\{<3, 0.75>, <6, 1.5>, <9, 2.25>, <12, 3>, <15, 3.75>, \ldots\}$.

Dado que todo subconjunto relevante de cadenas incluido en el conjunto total de cadenas posibles califica como lenguaje, todo problema puede ser traducido en términos de la pregunta por la pertenencia o no de un elemento a un lenguaje.

Por supuesto, salvo en casos triviales, los lenguajes generalmente son o bien muy vastos o directamente infinitos. Por eso, no es practicable definirlos por extensión. Queda entonces definirlos por intensión mediante alguna clase de función o algoritmo.

Existen distintas formas de definir lenguajes. Dos formas corrientes son las siguientes:

- Autómatas
- Gramáticas

Los lenguajes

Dado un vocabulario $\{0, 1\}$, consideren los siguientes lenguajes posibles:

Los lenguajes

Dado un vocabulario $\{0, 1\}$, consideren los siguientes lenguajes posibles:

1. L1: cualquier número de unos y ceros en cualquier orden
2. L2: un número cualquiera de unos seguidos de un número cualquiera de ceros
3. L2: un número cualquiera de unos seguidos del mismo número de ceros.
4. L3: un número de unos equivalente al cuadrado del número de ceros que haya
5. L4: un par ordenado formado por una cadena de ceros y unos que conformen un programa que haga operaciones con cadenas de ceros y unos y una cadena de ceros y unos que sea un input válido para ese programa

No todos los lenguajes pueden definirse por el mismo tipo de función, autómata, gramática.

Los lenguajes se caracterizan por su poder discriminatorio en distintos tipos.

- Lenguajes regulares
- Lenguajes independientes de contexto
- Lenguajes sensibles al contexto
- Lenguajes irrestrictos

Los lenguajes se caracterizan por su poder discriminatorio en distintos tipos.

- Lenguajes regulares
- Lenguajes independientes de contexto
- Lenguajes sensibles al contexto
- Lenguajes irrestrictos

Esto es lo que se conoce como **Jerarquía de Chomsky**

La jerarquía de Chomsky está definida en términos de inclusión: Lenguajes regulares $\subset$ Lenguas independientes de contexto $\subset$ Lenguajes sensibles al contexto $\subset$ Lenguajes irrestrictos.

Otra manera de denominar a los lenguajes es según la posibilidad de decidir si una cadena w pertenece o no a él. Según este criterio, se obtienen los siguientes tipos de lenguajes:

- **Lenguajes recursivos**: Es posible decidir si un elemento pertenece a L o a –L
- **Lenguajes recursivamente enumerables**: Es posible decidir si un elemento pertenece a L.
- **Lenguajes no recursivamente enumerables**: No es posible decidir si un elemento pertenece a L.

La máquina de Turing

- La máquina de Turing (MT) es el tipo de autómata más poderoso. Puede generar lenguajes irrestrictos/recursivamente enumerables, y, por ende, todos los lenguajes incluidos en ellos.
- Una MT consta de una cinta con símbolos, un escaner que lee esos símbolos y un conjunto de instrucciones que definen qué debe hacer en cada momento al ver un símbolo.
- Al correr una MT, cada aplicación de una instrucción es un paso.
- Toda función que pueda ser resuelta mediante una MT en una cantidad finita de pasos es una función computable.
- Puede optimizarse la cantidad de pasos agregando mayor cantidad de cintas con sus correspondientes escaners. Por supuesto, esto hará que cada instrucción sea más compleja. Una MT con más de una cinta se conoce como Multitape Turing Machine, y define exactamente los mismos lenguajes que las máquinas con una cinta.

Las computadoras reales tienen los siguientes elementos:

- Un conjunto de “palabras” (de 32 o 64 bits) junto con un número que las identifica, denominado Instruction Pointer o Address.
- El programa de la computadora también está incluido entre este conjunto de “palabras”. Estas palabras permiten “indirect addressing”, es decir, pueden operar con otras palabras.
- Toda instrucción opera sobre un conjunto finito de palabras y altera el valor de al menos una.
- Los registros son palabras de acceso rápido.

Toda MT puede ser traducida a una computadora y toda computadora puede ser traducida a una MT. Es decir, las computadoras y las MT son equivalentes (definen exactamente los mismos lenguajes). Puesto que una máquina de Turing es un sistema abstracto que tiene todas las limitaciones formales de una computadora pero ninguna de las físicas, estudiar qué cosas pueden resolverse en una máquina de Turing y cuáles no y qué cosas pueden resolverse eficientemente y cuáles no nos permite saber qué clase de problemas pueden resolver las computadoras reales y cuáles no.

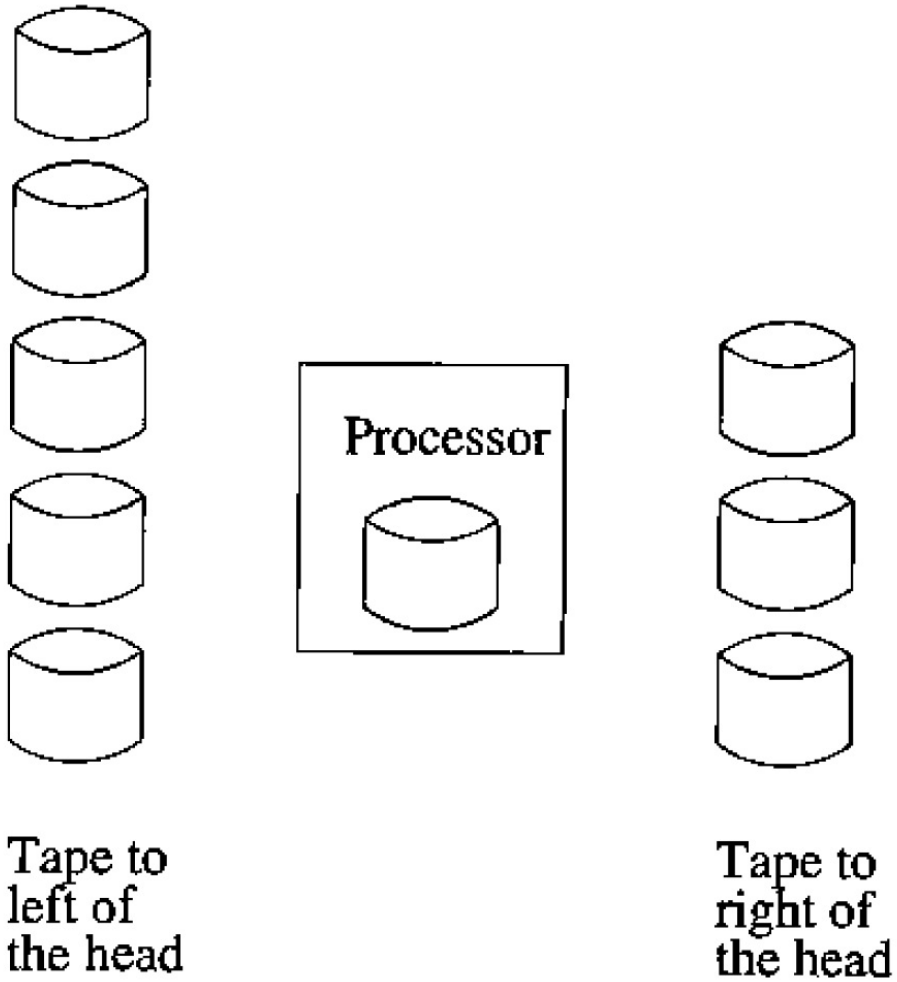

Figura: Computadora que imita una Máquina de Turing (tomado de Hopcroft, Motwani y Ullman 2001: 356)

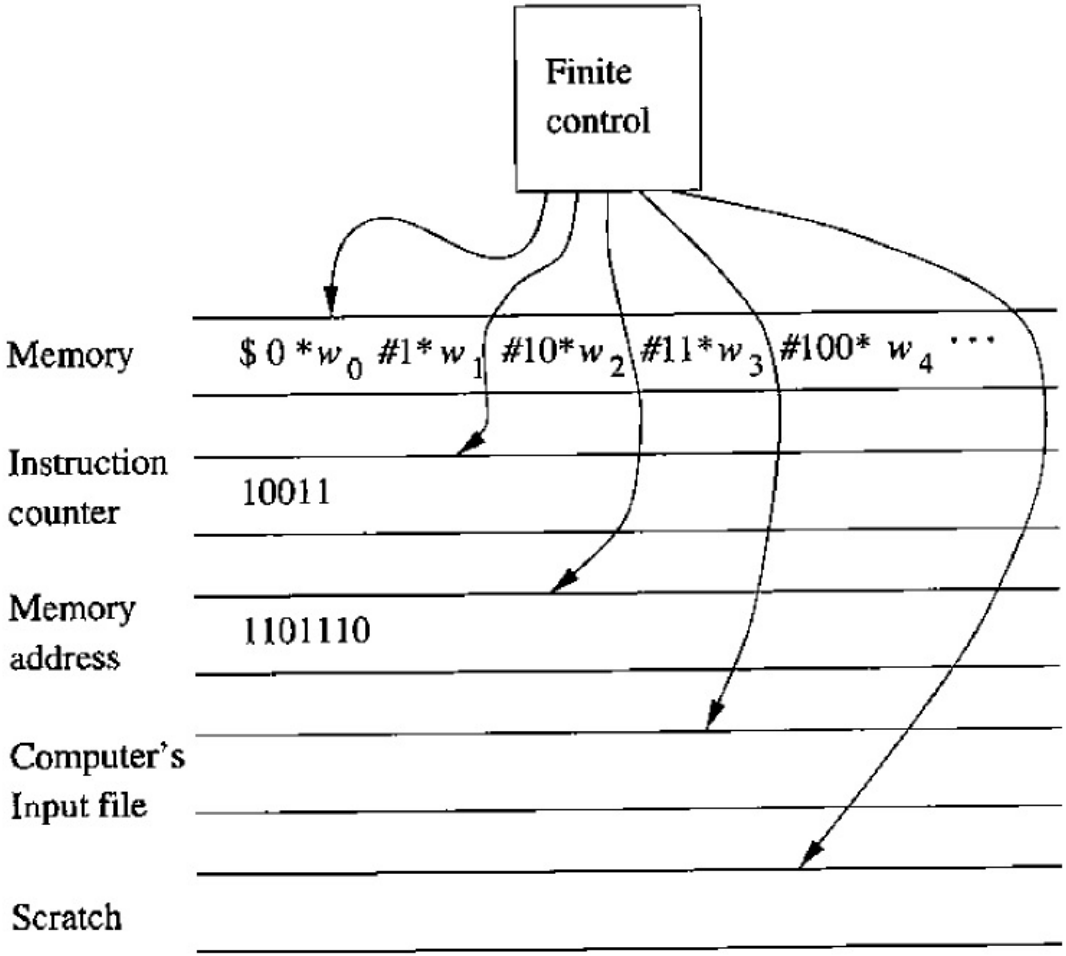

Figura: Máquina de Turing que imita una computadora (tomado de Hopcroft, Motwani y Ullman 2001: 359)

Complejidad

El principio de la **complejidad** mide la dificultad de resolver un problema computacional, medido en términos de recursos consumidos durante la computación. Normalmente se toma como referencia el espacio o el tiempo. (...) Cuanto más complejo sea el autómata permitido, tanto más complejas serán las lenguas reconocidas por él. (Moreno Sandoval 2001: 233)

Complejidad medida en términos de tiempo

Tipos de tiempos	**Notación O mayúscula**
Tiempo constante	$O(1)$
Tiempo lineal	$O(n)$
Tiempo polinómico	$O(n^c)$
Tiempo exponencial	$O(c^n)$
tiempo factorial	$O(n!)$

Complejidad medida en términos de tiempo

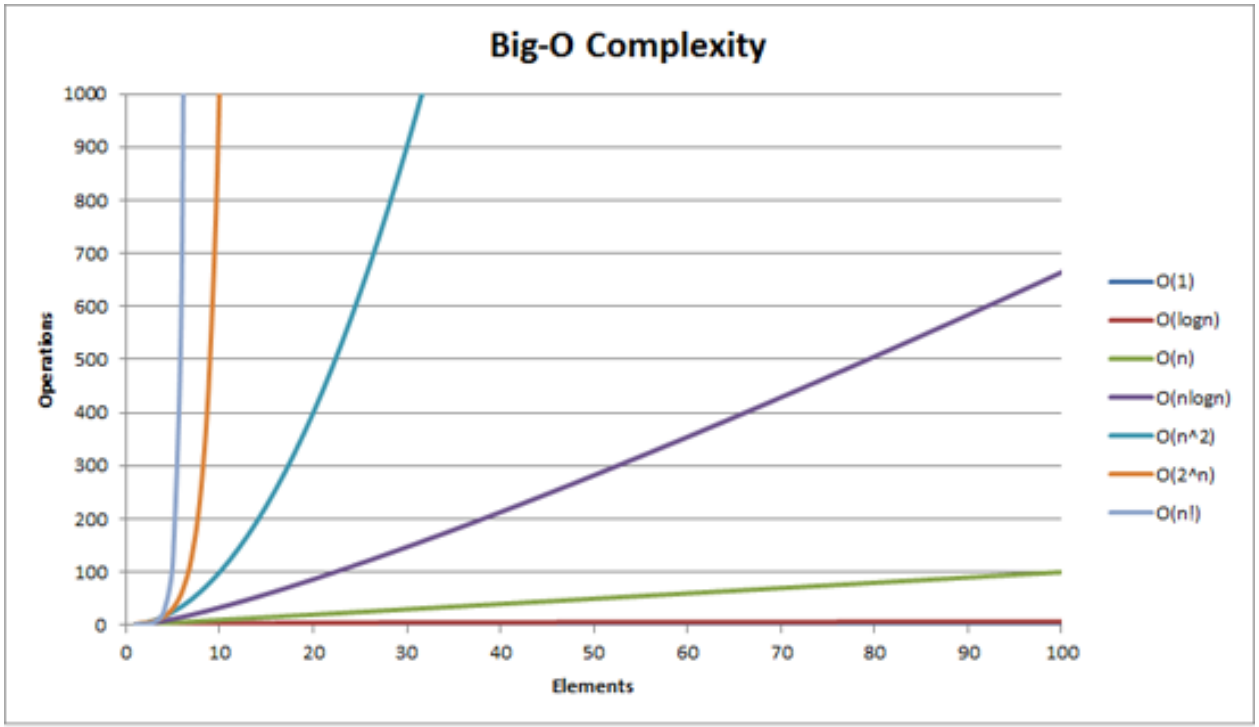

Figura: Comparación de curvas de los distintos costos de procesamiento (tomado de https://i.stack.imgur.com/Aq09a.png)

Complejidad medida en términos de tiempo

Se sabe que el procedimiento para decidir si un elemento pertenece a un lenguaje tiene un costo de procesamiento según la siguiente tabla:

Tipos de tiempos	**Notación O mayúscula**
Lenguajes regulares	tiempo lineal
Lenguajes independientes de contexto	tiempo polinómico
Lenguajes sensibles al contexto	tiempo exponencial (intratable)
Lenguajes irrestrictos	indecidible

La escala de complejidad nos permite saber qué clase de problemas podemos tratar de resolver en una computadora, cuáles solo pueden ser resueltos para números pequeños de datos. En consecuencia, siempre conviene reducir los problemas a la clase de problemas más simples que podamos, aun a costa de perder efectividad.

Por ejemplo, si queremos parsear oraciones del lenguaje natural, sabemos que una gramática cualquiera que genere lenguajes independientes de contexto es insuficiente, mientras que una gramática que genere lenguajes sensibles al contexto tiene poder suficiente. No obstante, una gramática sensible al contexto es intratable, puesto que tiene un costo de procesamiento que crece exponencialmente a medida que crece la longitud de la cadena. Por esta razón, es preferible muchas veces, en todo caso, utilizar una gramática independiente de contexto y, o bien renunciar a hacer un buen análisis de aquellas cadenas que desafían esta clase de lenguajes, o bien utilizar estrategias de posprocesamiento.

Anteriormente consideramos el siguiente lenguaje:

- Un par ordenado formado por una cadena de ceros y unos que conformen un programa que haga operaciones con cadenas de ceros y unos y una cadena de ceros y unos que sea un input válido para ese programa

Anteriormente consideramos el siguiente lenguaje:

- Un par ordenado formado por una cadena de ceros y unos que conformen un programa que haga operaciones con cadenas de ceros y unos y una cadena de ceros y unos que sea un input válido para ese programa

Este lenguaje se conoce con el nombre de **Lenguaje Universal**.

Anteriormente consideramos el siguiente lenguaje:

- Un par ordenado formado por una cadena de ceros y unos que conformen un programa que haga operaciones con cadenas de ceros y unos y una cadena de ceros y unos que sea un input válido para ese programa

Este lenguaje se conoce con el nombre de **Lenguaje Universal**. La pregunta por la pertenencia al lenguaje universal equivale en términos de lenguaje al problema de si existe un algoritmo general que pueda determinar si una Máquina de Turing puede aceptar o no determinado input. Se sabe que este problema es indecidible. Por lo tanto, cualquier intento de escribir un programa que resuelva este problema es desde ya una tarea vana.