Computación y Estructuras Discretas III

Andrés A. Aristizábal P.
aaaristizabal@icesi.edu.co
Ángela Villota
apvillota@icesi.edu.co

Departamento de Computación y Sistemas Inteligentes

2024-2

Agenda del día

1. **Presentación**
   - Información del curso
   - Estrategias Pedagógicas
   - Desarrollo del curso
   - Evaluaci?n
   - Aplicaciones
   - Bibliografía

2. **Regular languages and Automata theory**
   - Alphabets, strings and languages
   - Exercises

Agenda del día

Información del curso

- Código asignatura: 09772
- Programas: Ingeniería de Sistemas
- Intensidad semanal: 4 horas
- Apoyo al currículo central: I Plus
- Créditos: 3
- Días de clase: Martes-Jueves 16:00-18:00
- Salones: Martes 402E, Jueves 207C.

Descripción general del curso

- Este curso ofrece a los estudiantes una sólida base en informática teórica y procesamiento de lenguaje.
- A través de la aplicación de expresiones regulares, teoría de autómatas, gramáticas generativas y Máquinas de Turing, los estudiantes explorarán la resolución de problemas complejos en el procesamiento de lenguaje y reconocimiento de patrones.
- El curso tiene una gran componente de aplicación pero en cada unidad se hará énfasis en la comprensión de los conceptos y las técnicas formales que soportan cada una de las aplicaciones.
- Al finalizar el curso, los estudiantes estarán equipados con conocimientos teóricos y habilidades prácticas para enfrentar desafíos en la informática teórica y el procesamiento de lenguaje en un contexto académico y profesional.

Formación en competencias

- SO-1. Solución de problemas: Identificar, formular y resolver problemas complejos de ingeniería aplicando pensamiento crítico y principios de las ciencias, las matemáticas, la ingeniería y, en particular, de las Ciencias de la Computación y de la Ingeniería de Software.
- SO-2. Diseño de ingeniería: Diseñar soluciones y procesos basados en software que satisfagan necesidades específicas y generen valor a sus usuarios, considerando la salud pública, la seguridad y el bienestar de las personas, así como factores globales, culturales, sociales, ambientales y económicos.
- SO-3. Comunicación efectiva: Comunicarse efectivamente de forma oral y escrita, tanto en español como en inglés.

Objetivo General

Reconocer y aplicar apropiadamente distintos modelos computacionales de variado poder expresivo, reconociendo sus propiedades y limitaciones, para la solución de problemas asociados a lenguajes, gramáticas, aprendizaje automático y decidibilidad.

Objetivos terminales

- Aplicar las expresiones regulares y teoría de autómatas para resolver problemas relacionados al procesamiento de lenguaje y reconocimiento de patrones.
- Comprender y aplicar los conceptos de gramáticas generativas para implementar sistemas de procesamiento de lenguajes especializados para dominios específicos o aplicaciones.
- Reconocer la importancia de simplificar gramáticas en sus formas normales para poder procesar y analizar lenguajes de manera eficiente.
- Diseñar e implementar Máquinas de Turing para resolver problemas computacionales, en especial aquellos relacionados con redes neuronales.
- Entender la importancia de las técnicas formales para determinar la decidibilidad de problemas computacionales utilizando Máquinas de Turing.
- Expresar o comunicar con el vocabulario y lenguaje adecuado/especializado las ideas principales sobre los modelos computacionales estudiados en el curso y sus aplicaciones.

Agenda del día

Estrategias pedagógicas

Enfoque

En concordancia con la misión de la Universidad, el aprendizaje de los temas de este curso será el resultado del proceso de construcción del conocimiento, adelantado por el estudiante y guiado por el profesor. Parte fundamental de este proceso es el aprovechamiento del estudio previo hecho por los estudiantes, como elemento generador de preguntas, discusiones y conclusiones.

Estrategias pedagógicas

E-learning

La herramienta de E-learning (moodle) es el medio que contiene la información oficial del curso y es responsabilidad del estudiante consultar en ella todo lo referente al curso, especialmente las actualizaciones del material, actividades y calificaciones.

Estrategias pedagógicas

Discusión

- La discusión, orientada por el profesor es el elemento central en la metodología del curso. Se fundamenta en el estudio preliminar de las secciones asignadas, en las preguntas de los estudiantes y en sus respuestas a sus preguntas y a las del profesor, que alimenten el proceso de aprendizaje activo.
- El profesor interviene esencialmente como guía y moderador de las discusiones, y se encarga de hacer la síntesis final para socializar el conocimiento consolidado en clase y de indicar al estudiante la labor que debe realizar como preparación para la clase siguiente.

Agenda del día

Espacios

Este curso cuenta semanalmente con dos momentos (teórico-práctico y práctico) que son utilizados de la siguiente forma:

- Teórico-práctico: los estudiantes y el profesor se encontrarán en este espacio, en dos sesiones de una hora y media en las cuales se llevará a cabo la discusión de los diferentes temas y la realización de ejercicios que permitan ponerlos en práctica.
- Práctico: los estudiantes, el profesor y el monitor comparten este momento para llevar a cabo los seguimientos de evaluación, la solución de problemas propuestos y su implementación en Python y las librerías indicadas. Para este componente se han destinado dos horas cada 15 días.

Lenguaje y entorno de desarrollo

Este curso utilizará Python como lenguaje de programación y Jupyter notebooks como entorno de desarrollo. No obstante los objetivos de aprendizaje son independientes del lenguaje y el entorno de programación seleccionado.

Contenido I

1. **Lenguajes Regulares y Autómatas** [6.5 sesiones]
   - Explicar los conceptos fundamentales de alfabetos, cadenas y lenguajes.
   - Definir formalmente los distintos tipos de autómatas finitos (deterministas, no deterministas, no deterministas con transiciones lambda, transductores), dar ejemplos, identificar sus elementos y conocer sus aplicaciones.
   - Reconocer la equivalencia entre expresiones regulares y autómatas finitos.
   - Entender y aplicar, tanto las nociones de lenguajes y expresiones regulares, como de autómatas de estado finito, para el reconocimiento de patrones, procesamiento, validación y extracción de texto usando un lenguaje de programación.
2. **Gramáticas y Lenguajes** [6.5 sesiones]
   - Definir una gramática independiente del contexto y construir GICs para lenguajes dados.
   - Explicar el concepto de ambigüedad lingüística y aplicarlo a GIC ambiguas.
   - Reconocer la importancia de las formas normales para simplificar GICs.
   - Aplicar los conceptos de gram?áicas generativas para el diseño e implementación de DSLs usando un lenguaje de programación.
   - Reconocer la importancia de los distintos algoritmos de decisión sobre GIC.

Contenido II

3. **Máquinas de Turing y Aplicaciones** [11 sesiones]
   - Definir formalmente una Máquina de Turing y cada uno de sus componentes.
   - Construir Máquinas de Turing como reconocedoras de lenguajes y calculadoras de funciones usando un lenguaje de programación.
   - Comprender los conceptos básicos de aprendizaje automático, en especial, el procesamiento de lenguaje natural, el aprendizaje supervisado y las redes neuronales.
   - Aplicar los conceptos de redes neuronales recurrentes, LSTM y Redes Neuronales de Turing para resolver tareas de NLP usando un lenguaje de programación.
   - Codificar y enumerar Máquinas de Turing con el fin de establecer un marco estandarizado y riguroso para describir la estructura, comportamiento y límites de los distintos dispositivos computacionales.
   - Reconocer la importancia de lenguajes que no son RE, RE que no son Recursivos para representar problemas complejos, explorar la computabilidad y así proporcionar una base teórica para la teoría del lenguaje formal.

Agenda del día

1. **Presentación**
   - Información del curso
   - Estrategias Pedagógicas
   - Desarrollo del curso
   - Evaluaci?n
   - Aplicaciones
   - Bibliografía
2. **Regular languages and Automata theory**
   - Alphabets, strings and languages
   - Exercises

Evaluaci?n

Evaluaciones individuales

El curso cuenta con 2 momentos de evaluaci?n individual en el sal?n de clase:

- Controles de lectura y quices: corresponde a todas las comprobaciones de lectura y de aprendizaje que se hagan durante el curso. Estas comprobaciones pueden realizarse con o sin previo aviso por parte del profesor.
- Seguimiento de Aprendizaje: evaluación corta durante los 30 - 60 primeros minutos del horario de la sesión práctica.

Evaluaciones grupales

El curso cuenta con 1 momento de evaluaci?n grupal en el sal?n de clase:

- Tareas Integradoras: durante el semestre se llevarán a cabo 2 Tareas Integradoras que los estudiantes deben entregar utilizando GitHub classroom.

Evaluaci?n

Porcentaje de evaluaciones

Tarea Integradora 1	20 %
Tarea Integradora 2	20 %
Seguimientos de Aprendizaje	45 %
Controles de lectura y quices	15 %

Agenda del día

1. **Presentación**
   - Información del curso
   - Estrategias Pedagógicas
   - Desarrollo del curso
   - Evaluaci?n
   - Aplicaciones
   - Bibliografía
2. **Regular languages and Automata theory**
   - Alphabets, strings and languages
   - Exercises

Aplicaciones

- Lenguajes regulares: validación de datos, data scraping, preprocesamiento de datos, data wrangling.
- Autómatas finitos: correctores ortográficos, editores de texto, analizadores léxicos.
- Transductores: traductores, analizadores fonológicos y morfológicos.
- Gramáticas libres de contexto: analizadores.
- Máquinas de Turing: Turing neural networks.

Agenda del día

Bibliografía

- John E. Hopcroft Rajeev Motwani Jeffrey D. Ullman. Introduction to Automata Theory, Languages, and Computation. Prentice Hall (2006)
- Ralf Lämmel. Software Languages, Syntax, Semantics, and Metaprogramming. Springer 2018
- Rodrigo de Castro. Teoría de la Computación, Lenguajes, autómatas, gramáticas (Notas de Clase). Editorial de la Universidad Nacional (2004)
- Lecturas complementarias disponibles en la herramienta de E-learning (Moodle) para los estudiantes de la materia.

Agenda del día

1. Presentación
   - Información del curso
   - Estrategias Pedagógicas
   - Desarrollo del curso
   - Evaluaci?n
   - Aplicaciones
   - Bibliografía

2. **Regular languages and Automata theory**
   - Alphabets, strings and languages
   - Exercises

Alphabets, strings and languages

What is an alphabet?

Alphabets, strings and languages

What is an alphabet?

Definition

An alphabet is a finite, noneempty set of symbols. We use the symbol Σ to define it.

Alphabets, strings and languages

What is an alphabet?

Definition

An alphabet is a finite, noneempty set of symbols. We use the symbol Σ to define it.

Which are some examples of alphabets?

Alphabets, strings and languages

What is an alphabet?

Definition

An alphabet is a finite, noneempty set of symbols. We use the symbol Σ to define it.

Which are some examples of alphabets?

- The binary alphabet: $\Sigma = \{0, 1\}$
- The set of all lower case vowels: $\Sigma = \{a, e, i, o, u\}$.
- The set of all ASCII characters, or the set of all printable ASCII characters.

Alphabets, strings and languages

What is a string?

What is a string?

Definition

A string or a word is a finite sequence of symbols chosen from some alphabet.

Alphabets, strings and languages

What is a string?

Definition

A string or a word is a finite sequence of symbols chosen from some alphabet.

Which are some examples of strings?

Alphabets, strings and languages

What is a string?

Definition

A string or a word is a finite sequence of symbols chosen from some alphabet.

Which are some examples of strings?

- 01101 is a string from the binary alphabet $\Sigma = \{0, 1\}$
- *banana* is a string from the all lower case letters‘ alphabet $\Sigma = \{a, b, c, ..., z\}$.

Example

Let $\Sigma = \{a, b\}$.

Example

Let $\Sigma = \{a, b\}$.
Here we have strings from Σ :

aba
ababaaa
aaaab

Example

Let $\Sigma = \{a, b\}$.
Here we have strings from Σ :

$$aba$$
$$ababaaa$$
$$aaaab$$

We can see that $aba \neq aab$, *due to the fact that the order of symbols is important when dealing with strings, since strings work as sequences.*

Alphabets, strings and languages

What is the empty string?

What is the empty string?

Definition

The empty string is the string with zero occurrences of symbols. This string is denoted by λ or ϵ. This string may be chosen from any alphabet whatsoever.

Alphabets, strings and languages

What about the length of a string?

Alphabets, strings and languages

What about the length of a string?

- It is useful to classify strings by their length.
- It is common to say that the length of a string is the number of symbols in the string.
- Not strictly correct. 10101 has only two symbols but there are five positions for symbols (its length is 5).
- Generally expect that the number of symbols can be used when. number of positions is meant.
- The standard notation for the length of a string w is $|w|$.

Alphabets, strings and languages

How could we formally define the length of a string?

How could we formally define the length of a string?

Definition

$$|u| = \begin{cases} 0 & \text{if } u = \lambda \\ n & \text{if } u = a_1 a_2 \cdots a_n. \end{cases}$$

How could we formally define the length of a string?

Definition

$$|u| = \begin{cases} 0 & \text{if } u = \lambda \\ n & \text{if } u = a_1 a_2 \cdots a_n. \end{cases}$$

Example

$|aba| = 3$, $|baaa| = 4$, $|\lambda| = 0$.

Alphabets, strings and languages

What are the powers of an alphabet?

Alphabets, strings and languages

What are the powers of an alphabet?

- Let Σ be an alphabet.
- We can express the set of all strings of a certain length from that alphabet by using an exponential notation.
- We define Σ^k to be the set of strings of length k, each of whose symbols is in Σ.

Alphabets, strings and languages

What are the powers of an alphabet?

- Let Σ be an alphabet.
- We can express the set of all strings of a certain length from that alphabet by using an exponential notation.
- We define Σ^k to be the set of strings of length k, each of whose symbols is in Σ.

Example

- $\Sigma^0 = \{\lambda\}$ *regardless of what alphabet* Σ *is.*
- *If* $\Sigma = \{0, 1\}$, *then* $\Sigma^1 = \{0, 1\}$, $\Sigma^2 = \{00, 01, 10, 11\}$.

Alphabets, strings and languages

What is the Kleene star of an alphabet?

Alphabets, strings and languages

What is the Kleene star of an alphabet?

- The set of all possible strings over an alphabet Σ.
- It is denoted as Σ^*.
- $\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \ldots$

Alphabets, strings and languages

What is the Kleene star of an alphabet?

- The set of all possible strings over an alphabet Σ.
- It is denoted as Σ^*.
- $\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \ldots$

Example

- *If* $\Sigma = \{0, 1\}$, *then* $\Sigma^* = \{\lambda, 0, 1, 00, 01, 10, 11, 000, ...\}$.

Alphabets, strings and languages

What is the positive closure of an alphabet?

Alphabets, strings and languages

What is the positive closure of an alphabet?

- The set of all possible strings over an alphabet Σ without the empty string.
- It is denoted as Σ^+.
- $\Sigma^+ = \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \cup \cdots$

What is the positive closure of an alphabet?

- The set of all possible strings over an alphabet Σ without the empty string.
- It is denoted as Σ^+.
- $\Sigma^+ = \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \cup \cdots$

Example

- *If* $\Sigma = \{0, 1\}$, *then* $\Sigma^* = \{0, 1, 00, 01, 10, 11, 000, ...\}$.

Alphabets, strings and languages

What is string concatenation?

What is string concatenation?

Definition

Given an alphabet Σ and two strings $u, v \in \Sigma^*$, the concatenation of u and v is expressed as $u \cdot v$ or simply uv.

What is string concatenation?

Definition

Given an alphabet Σ and two strings $u, v \in \Sigma^*$, the concatenation of u and v is expressed as $u \cdot v$ or simply uv. We can define it as

What is string concatenation?

Definition

Given an alphabet Σ and two strings $u, v \in \Sigma^*$, the concatenation of u and v is expressed as $u \cdot v$ or simply uv. We can define it as If $u, v \in \Sigma^*$, $a \in \Sigma$, then

What is string concatenation?

Definition

Given an alphabet Σ and two strings $u, v \in \Sigma^*$, the concatenation of u and v is expressed as $u \cdot v$ or simply uv. We can define it as If $u, v \in \Sigma^*$, $a \in \Sigma$, then

1. $u \cdot \lambda = \lambda \cdot u = u$

What is string concatenation?

Definition

Given an alphabet Σ and two strings $u, v \in \Sigma^*$, the concatenation of u and v is expressed as $u \cdot v$ or simply uv. We can define it as If $u, v \in \Sigma^*$, $a \in \Sigma$, then

1. $u \cdot \lambda = \lambda \cdot u = u$
2. $u \cdot (va) = (u \cdot v)a$

Alphabets, strings and languages

What is the reverse or inverse of a string?

What is the reverse or inverse of a string?

Definition

The reverse or inverse of a string $u \in \Sigma^*$ is expressed as u^R and defined as:

What is the reverse or inverse of a string?

Definition

The reverse or inverse of a string $u \in \Sigma^*$ is expressed as u^R and defined as:

$$u^R = \begin{cases} \lambda & \text{if } u = \lambda \\ a_n \cdots a_2 a_1 & \text{if } u = a_1 a_2 \cdots a_n. \end{cases}$$

Alphabets, strings and languages

What is a substring?

Alphabets, strings and languages

What is a substring?

Definition

A string v is a substring or subword of u if there exist strings x, y such that $u = xvy$. Let's point out that x or y might be λ and therefore, the empty string is a substring of any string and any string is a substring of itself.

Alphabets, strings and languages

What is a substring?

Definition

A string v is a substring or subword of u if there exist strings x, y such that $u = xvy$. Let's point out that x or y might be λ and therefore, the empty string is a substring of any string and any string is a substring of itself.

What is a prefix?

Alphabets, strings and languages

What is a substring?

Definition

A string v is a substring or subword of u if there exist strings x, y such that $u = xvy$. Let's point out that x or y might be λ and therefore, the empty string is a substring of any string and any string is a substring of itself.

What is a prefix?

Definition

A prefix of u is a string v such that $u = vw$ for any string $w \in \Sigma^*$. We say that v is a proper prefix if $v \neq u$.

Alphabets, strings and languages

What is a substring?

Definition

A string v is a substring or subword of u if there exist strings x, y such that $u = xvy$. Let's point out that x or y might be λ and therefore, the empty string is a substring of any string and any string is a substring of itself.

What is a prefix?

Definition

A prefix of u is a string v such that $u = vw$ for any string $w \in \Sigma^*$. We say that v is a proper prefix if $v \neq u$.

What is a suffix?

Alphabets, strings and languages

What is a substring?

Definition

A string v is a substring or subword of u if there exist strings x, y such that $u = xvy$. Let's point out that x or y might be λ and therefore, the empty string is a substring of any string and any string is a substring of itself.

What is a prefix?

Definition

A prefix of u is a string v such that $u = vw$ for any string $w \in \Sigma^*$. We say that v is a proper prefix if $v \neq u$.

What is a suffix?

Definition

A suffix of u is a string v such that $u = wv$ for any string $w \in \Sigma^*$. We say that v is a proper suffix if $v \neq u$.

See that λ is a prefix and suffix of u since $u\lambda = \lambda u = u$. By the same reasoning u is a prefix and suffix of itself.

See that λ is a prefix and suffix of u since $u\lambda = \lambda u = u$. By the same reasoning u is a prefix and suffix of itself.

Example

Given $\Sigma = \{a, d, c, d\}$ *and* $u = bcbaadb$.

Prefixes of u :	*Suffixes of* u :
λ	λ
b	b
bc	db
bcb	adb
$bcba$	$aadb$
$bcbaa$	$baadb$
$bcbaaad$	$cbaadb$
$bcbaadb$	$bcbaadb$

Alphabets, strings and languages

What is a language?

Alphabets, strings and languages

What is a language?

Definition

A language L over an alphabet Σ is a subset of Σ^*, that is $L \subseteq \Sigma^*$

What is a language?

Definition

A language L over an alphabet Σ is a subset of Σ^*, that is $L \subseteq \Sigma^*$
Border cases:

$L = \emptyset$, *the empty language.*
$L = \Sigma^*$, *the language of every string over* Σ.

What is a language?

Definition

A language L over an alphabet Σ is a subset of Σ^*, that is $L \subseteq \Sigma^*$
Border cases:

$L = \emptyset$, *the empty language.*
$L = \Sigma^*$, *the language of every string over* Σ.

Any language L fulfills $\emptyset \subseteq L \subseteq \Sigma^*$, and it can be either finite or infinite.
We represent languages with upper case letters A, B, C, ..., L, M, N, ...

Example

The following are examples of languages over given alphabets.

Example

The following are examples of languages over given alphabets.

- $\Sigma = \{a, b, c\}$. $L = \{a, aba, aca\}$.

Example

The following are examples of languages over given alphabets.

- $\Sigma = \{a, b, c\}$. $L = \{a, aba, aca\}$.
- $\Sigma = \{a, b, c\}$. $L = \{a, aa, aaa, ...\} = \{a^n \mid n \geq 1\}$.

Example

The following are examples of languages over given alphabets.

- $\Sigma = \{a, b, c\}$. $L = \{a, aba, aca\}$.
- $\Sigma = \{a, b, c\}$. $L = \{a, aa, aaa, ...\} = \{a^n \mid n \geq 1\}$.
- $\Sigma = \{a, b, c\}$. $L = \{\lambda, aa, aba, ab^2a, ab^3a, ...\} = \{ab^na \mid n \geq 0\} \cup \{\lambda\}$.

Alphabets, strings and languages

Which are the operations between languages?

Which are the operations between languages?

Definition

Since languages over Σ are subsets of Σ^*, the usual operations between sets are also valid operations between languages. Thení, if A and B are languages over Σ ($A, B \subseteq \Sigma^*$), then the following are also languages over Σ :

$A \cup B$ *Union*

Which are the operations between languages?

Definition

Since languages over Σ are subsets of Σ^*, the usual operations between sets are also valid operations between languages. Thení, if A and B are languages over Σ ($A, B \subseteq \Sigma^*$), then the following are also languages over Σ :

$A \cup B$	*Union*
$A \cap B$	*Intersection*

Which are the operations between languages?

Definition

Since languages over Σ are subsets of Σ^*, the usual operations between sets are also valid operations between languages. Thení, if A and B are languages over Σ ($A, B \subseteq \Sigma^*$), then the following are also languages over Σ :

$A \cup B$	*Union*
$A \cap B$	*Intersection*
$A - B$	*Diference*

Which are the operations between languages?

Definition

Since languages over Σ are subsets of Σ^*, the usual operations between sets are also valid operations between languages. Thení, if A and B are languages over Σ ($A, B \subseteq \Sigma^*$), then the following are also languages over Σ :

$A \cup B$	*Union*
$A \cap B$	*Intersection*
$A - B$	*Diference*
$\overline{A} = \Sigma^* - A$	*Complement*

Which are the operations between languages?

Definition

Since languages over Σ are subsets of Σ^*, the usual operations between sets are also valid operations between languages. Thení, if A and B are languages over Σ ($A, B \subseteq \Sigma^*$), then the following are also languages over Σ :

$A \cup B$	*Union*
$A \cap B$	*Intersection*
$A - B$	*Diference*
$\overline{A} = \Sigma^* - A$	*Complement*

These operations between languages are called boolean operations in order to differentiate them from the linguistic operations, which are extensions to the languages of the operations between strings.

Alphabets, strings and languages

What is the concatenation between languages?

What is the concatenation between languages?

Definition

The concatenation of two languages A and B over Σ, represented as $A \cdot B$ or simply AB is defined as:

What is the concatenation between languages?

Definition

The concatenation of two languages A and B over Σ, represented as $A \cdot B$ or simply AB is defined as:

$$AB = \{uv \mid u \in A,\ v \in B\}$$

What is the concatenation between languages?

Definition

The concatenation of two languages A and B over Σ, represented as $A \cdot B$ or simply AB is defined as:

$$AB = \{uv \mid u \in A,\ v \in B\}$$

In general, $AB \neq BA$

Alphabets, strings and languages

Which are the properties of concatenation of languages?

Alphabets, strings and languages

Which are the properties of concatenation of languages?

Definition

Given languages A, B, C *over* Σ, *i.e.* $A, B, C \subseteq \Sigma^*$. *Then*

Alphabets, strings and languages

Which are the properties of concatenation of languages?

Definition

Given languages A, B, C over Σ, i.e. $A, B, C \subseteq \Sigma^*$. Then

1. $A \cdot \emptyset = \emptyset \cdot A = \emptyset$.

Alphabets, strings and languages

Which are the properties of concatenation of languages?

Definition

Given languages A, B, C over Σ, i.e. $A, B, C \subseteq \Sigma^*$. Then

1. $A \cdot \emptyset = \emptyset \cdot A = \emptyset$.
2. $A \cdot \{\lambda\} = \{\lambda\} \cdot A = A$.

Alphabets, strings and languages

Which are the properties of concatenation of languages?

Definition

Given languages A, B, C over Σ, i.e. $A, B, C \subseteq \Sigma^*$. Then

1. $A \cdot \emptyset = \emptyset \cdot A = \emptyset$.
2. $A \cdot \{\lambda\} = \{\lambda\} \cdot A = A$.
3. *Associative property,*

$$A \cdot (B \cdot C) = (A \cdot B) \cdot C.$$

Alphabets, strings and languages

Which are the properties of concatenation of languages?

Definition

Given languages A, B, C over Σ, i.e. $A, B, C \subseteq \Sigma^*$. Then

1. $A \cdot \emptyset = \emptyset \cdot A = \emptyset$.
2. $A \cdot \{\lambda\} = \{\lambda\} \cdot A = A$.
3. *Associative property,*

$$A \cdot (B \cdot C) = (A \cdot B) \cdot C.$$

4. *Distributivity of concatenation with respect to union,*

$$A \cdot (B \cup C) = A \cdot B \cup A \cdot C.$$
$$(B \cup C) \cdot A = B \cdot A \cup C \cdot A.$$

Alphabets, strings and languages

Which are the properties of concatenation of languages?

Definition

Given languages A, B, C over Σ, i.e. $A, B, C \subseteq \Sigma^*$. Then

1. $A \cdot \emptyset = \emptyset \cdot A = \emptyset$.
2. $A \cdot \{\lambda\} = \{\lambda\} \cdot A = A$.
3. *Associative property,*
$$A \cdot (B \cdot C) = (A \cdot B) \cdot C.$$
4. *Distributivity of concatenation with respect to union,*
$$A \cdot (B \cup C) = A \cdot B \cup A \cdot C.$$
$$(B \cup C) \cdot A = B \cdot A \cup C \cdot A.$$
5. *Generalized distributivity property. If $\{B_i\}_{i \in I}$ is any family of languages over Σ, then*
$$A \cdot \bigcup_{i \in I} B_i = \bigcup_{i \in I} (A \cdot B_i),$$
$$\left(\bigcup_{i \in I} B_i \right) \cdot A = \bigcup_{i \in I} (B_i \cdot A).$$

Agenda del día

1. Presentación
   - Información del curso
   - Estrategias Pedagógicas
   - Desarrollo del curso
   - Evaluaci?n
   - Aplicaciones
   - Bibliografía
2. **Regular languages and Automata theory**
   - Alphabets, strings and languages
   - Exercises

Exercises

Exercise

Given $\Sigma = \{a, b, c, d\}$ *define* Σ^*

Exercises

Exercise

Given $u \in \Sigma^*$ and $n \in \mathbb{N}$, give a recursive definition of u^n.

Exercises

Exercise

Given $u \in \Sigma^*$ and $a \in \Sigma$ give a recursive definition of $|u|_a$.

Exercises

Exercise

Given $u \in \Sigma^*$ and $a \in \Sigma$ give a recursive definition of u^R.

Exercises

Exercise

If $u, v \in \Sigma^*$, *show that* $(uv)^R = v^R u^R$.

Exercises

Exercise

Show that $A \cdot (B \cdot C) = (A \cdot B) \cdot C$.

Exercises

Exercise

Show that $A \cdot \{\lambda\} = \{\lambda\} \cdot A = A$.

Exercises

Exercise

Show that $A \cdot \emptyset = \emptyset \cdot A = \emptyset$.