

Coordinación de Ciencias Computacionales – INAOE

Teoría de Automatas y Lenguajes Formales

Temario detallado para examen de ingreso 2012

1. Automatas
 - 1.1. ¿Por qué estudiar la teoría de autómatas?
 - 1.1.1. Introducción al autómata finito
 - 1.1.2. Representaciones estructurales
 - 1.1.3. El autómata y su complejidad
 - 1.2. Introducción a las pruebas formales
 - 1.2.1. Pruebas deductivas
 - 1.2.2. La reducción de las definiciones
 - 1.2.3. Otras formas de teoremas
 - 1.2.4. Teoremas que parecen no ser sentencias If-Then
 - 1.3. Formas adicionales de prueba
 - 1.3.1. Pruebas de equivalencias en conjuntos
 - 1.3.2. El contra positivo
 - 1.3.3. Prueba por contradicción
 - 1.3.4. Contraejemplos
 - 1.4. Pruebas Inductivas
 - 1.4.1. Inducciones con enteros
 - 1.4.2. Formas más generales de inducciones con enteros
 - 1.4.3. Inducciones estructurales
 - 1.4.4. Inducciones mutuas
 - 1.5. Conceptos centrales de la teoría de autómatas
 - 1.5.1. Alfabetos
 - 1.5.2. Cadenas
 - 1.5.3. Lenguajes
 - 1.5.4. Problemas
2. Automatas finitos
 - 2.1. Una visión informal del autómata finito
 - 2.1.1. Las reglas base
 - 2.1.2. El protocolo
 - 2.1.3. Habilitación del autómata para ignorar acciones
 - 2.1.4. El sistema completo como un autómata
 - 2.1.5. Uso del autómata producto para validar el protocolo
 - 2.2. Automatas finitos deterministas
 - 2.2.1. Definición de un autómata finito determinista (DFA, por sus siglas en Inglés)
 - 2.2.2. Como un DFA procesa cadenas
 - 2.2.3. Notaciones simples para DFA
 - 2.2.4. Extensión de la función de transición a cadenas
 - 2.2.5. El lenguaje de un DFA
 - 2.3. Autómata finito no determinista (NFA, por sus siglas en Inglés)

- 2.3.1. Una mirada informal al NFA
 - 2.3.2. Definición del NFA
 - 2.3.3. La función de transición extendida
 - 2.3.4. El lenguaje de un NFA
 - 2.3.5. Equivalencia entre autómatas finitos deterministas y no deterministas
 - 2.3.6. Un caso malo para la construcción de subconjuntos
- 2.4. Una aplicación: búsqueda en texto
 - 2.4.1. Encontrar cadenas en textos
 - 2.4.2. Autómatas finitos no deterministas para búsqueda en texto
 - 2.4.3. Un DFA para reconocer un conjunto de palabras clave
- 2.5. Autómatas finitos con Transiciones Epsilon
 - 2.5.1. Usos de las transiciones epsilon
 - 2.5.2. La notación formal para ϵ -NFA
 - 2.5.3. Cerraduras-epsilon
 - 2.5.4. Transiciones extendidas y lenguajes para ϵ -NFA
 - 2.5.5. Eliminando transiciones- ϵ
- 3. Expresiones regulares y lenguajes
 - 3.1. Expresiones regulares
 - 3.1.1. Los operadores de las expresiones regulares
 - 3.1.2. Construcción de expresiones regulares
 - 3.1.3. Precedencia de operaciones de las expresiones regulares
 - 3.2. Autómatas finitos y expresiones regulares
 - 3.2.1. De DFA a expresiones regulares
 - 3.2.2. Conversión de DFA a expresiones regulares por eliminación de estados
 - 3.2.3. Conversión de expresiones regulares a autómatas
 - 3.3. Aplicaciones de expresiones regulares
 - 3.3.1. Expresiones regulares en UNIX
 - 3.3.2. Análisis léxico
 - 3.3.3. Encontrando patrones en texto
 - 3.4. Reglas algebraicas para expresiones regulares
 - 3.4.1. Asociatividad y conmutatividad
 - 3.4.2. Identidades y “aniquiladores”
 - 3.4.3. Ley distributiva
 - 3.4.4. Ley de idem-potencia
 - 3.4.5. Leyes relacionadas con la propiedad de cerradura
 - 3.4.6. Descubrir leyes para expresiones regulares
 - 3.4.7. La prueba de una ley algebraica para expresiones regulares
- 4. Propiedades de los lenguajes regulares
 - 4.1. Demostración de lenguajes no regulares
 - 4.1.1. Lema de bombeo para lenguajes regulares
 - 4.1.2. Aplicaciones del lema de bombeo
 - 4.2. Propiedades de cerradura de los lenguajes regulares
 - 4.2.1. Cerradura de lenguajes regulares bajo operaciones booleanas
 - 4.2.2. Inversión

- 4.2.3.Homomorfismos
 - 4.2.4.Homomorfismos inversos
 - 4.3. Propiedades de decisión de los lenguajes regulares
 - 4.3.1.Conversión entre representaciones
 - 4.3.2.Prueba de vacuidad de lenguajes regulares
 - 4.3.3.Prueba de pertenencia en lenguajes regulares
 - 4.4. Equivalencia y minimización de autómatas
 - 4.4.1.Prueba de equivalencia de estados
 - 4.4.2.Prueba de equivalencia de lenguajes regulares
 - 4.4.3.Minimización de DFA
 - 4.4.4.Porque no se puede vencer a un DFA minimizado
- 5. Gramáticas libres de contexto y lenguajes
 - 5.1. Gramáticas libres de contexto
 - 5.1.1.Un ejemplo informal
 - 5.1.2.Definición de las gramáticas libres de contexto
 - 5.1.3.Derivaciones utilizando gramáticas
 - 5.1.4.Derivaciones por la izquierda y por la derecha
 - 5.1.5.El lenguaje de una gramática
 - 5.1.6.Formas enunciativas
 - 5.2. Análisis sintáctico de árboles
 - 5.2.1.Construcción de analizadores sintácticos de árboles
 - 5.2.2.El rendimiento de un analizador sintáctico de árboles
 - 5.2.3.Inferencia, derivaciones y analizadores sintácticos de árboles
 - 5.2.4.De inferencias a árboles
 - 5.2.5.De árboles a derivaciones
 - 5.2.6.De derivaciones a inferencias recursivas
 - 5.3. Aplicaciones de las gramáticas libres de contexto
 - 5.3.1.Analizadores sintácticos
 - 5.3.2.El analizador sintáctico-generator: YACC
 - 5.3.3.Lenguajes de marcado
 - 5.3.4.XML y definiciones documento-tipo
 - 5.4. Ambigüedad en gramáticas y lenguajes
 - 5.4.1.Gramáticas ambiguas
 - 5.4.2.Eliminación de la ambigüedad en las gramáticas
 - 5.4.3.Derivaciones por la izquierda como una forma de expresar ambigüedad
 - 5.4.4.Ambigüedad inherente
- 6. Autómata de pila
 - 6.1. Definición del autómata de pila
 - 6.1.1.Introducción informal
 - 6.1.2.La definición formal del autómata de pila
 - 6.1.3.Notación gráfica para el PDA (por sus siglas en inglés)
 - 6.1.4.Descripciones instantáneas del PDA
 - 6.2. Los lenguajes de un PDA
 - 6.2.1.Aceptación por estado final

- 6.2.2. Aceptación por pila vacía
- 6.2.3. De la pila vacía al estado final
- 6.2.4. Del estado final a la pila vacía
- 6.3. Equivalencia de los autómatas de pila y las gramáticas libres de contexto
 - 6.3.1. De las gramáticas a los autómatas de pila
 - 6.3.2. De los autómatas de pila a las gramáticas
- 6.4. Autómatas de pila deterministas
 - 6.4.1. Definición de un PDA determinista
 - 6.4.2. Lenguajes regulares y PDA deterministas
 - 6.4.3. PDA deterministas y lenguajes libres de contexto
 - 6.4.4. PDA deterministas y gramáticas ambiguas
- 7. Propiedades de los lenguajes libres de contexto
 - 7.1. Formas normales de las gramáticas libres de contexto
 - 7.1.1. Eliminación de símbolos innecesarios
 - 7.1.2. Cálculo de los símbolos generadores y alcanzables
 - 7.1.3. Eliminación de producciones ϵ
 - 7.1.4. Eliminación de producciones unitarias
 - 7.1.5. Forma normal de Chomsky
 - 7.2. Lema de bombeo para los lenguajes libres de contexto
 - 7.2.1. El tamaño de los analizadores sintácticos de árboles
 - 7.2.2. Declaración del lema de bombeo
 - 7.2.3. Aplicación del lema de bombeo para lenguajes libres de contexto
 - 7.3. Propiedades de cerradura para los lenguajes libres de contexto
 - 7.3.1. Sustituciones
 - 7.3.2. Aplicaciones del teorema de sustitución
 - 7.3.3. Inversión
 - 7.3.4. Intersección con un lenguaje regular
 - 7.3.5. Homomorfismo inverso
 - 7.4. Propiedades de decisión de los lenguajes libres de contexto
 - 7.4.1. Complejidad de conversión entre gramáticas libres de contexto y autómatas de pila deterministas
 - 7.4.2. Tiempo de ejecución para conversión a la forma normal de Chomsky
 - 7.4.3. Prueba de vacío de los lenguajes libres de contexto
 - 7.4.4. Prueba de pertenencia a un lenguaje libre de contexto
 - 7.4.5. Vista previa a problemas de los lenguajes libres de contexto indecidibles
- 8. Introducción a las máquinas de Turing
 - 8.1. Problemas que las computadoras no pueden resolver
 - 8.1.1. Programas que imprimen "Hello, World"
 - 8.1.2. El evaluador "Hello, World" hipotético
 - 8.1.3. Reduciendo un problema a otro
 - 8.2. La máquina de Turing
 - 8.2.1. La búsqueda para decidir todas las preguntas matemáticas
 - 8.2.2. Notación para la máquina de Turing
 - 8.2.3. Descripciones instantáneas para la máquina de Turing

- 8.2.4. Diagramas de transición para las máquinas de Turing
- 8.2.5. El lenguaje de una máquina de Turing
- 8.2.6. Máquinas de Turing e interrupción
- 8.3. Técnicas de programación para las máquinas de Turing
 - 8.3.1. Almacenamiento en el estado
 - 8.3.2. Rutas múltiples
 - 8.3.3. Subrutinas
- 8.4. Extensiones a la máquina de Turing básica
 - 8.4.1. Máquinas de Turing con múltiples cintas
 - 8.4.2. Equivalencia entre máquinas de Turing de una y múltiples cintas
 - 8.4.3. Tiempo de ejecución y la construcción muchas cintas a – una cinta
 - 8.4.4. Máquinas de Turing no deterministas
- 8.5. Máquinas de Turing restringidas
 - 8.5.1. Máquinas de Turing con cintas semi-infinitas
 - 8.5.2. Máquinas con múltiples pilas
 - 8.5.3. Máquinas
 - 8.5.4. El poder de las máquinas
- 8.6. Máquinas de Turing y las computadoras
 - 8.6.1. Simulación por computadora de una máquina de Turing
 - 8.6.2. Simulación por máquina de Turing de una computadora
 - 8.6.3. Comparación de los tiempos de operación de computadoras y máquinas de Turing

Bibliografía

1. John E. Hopcroft, Rajeev Motwani and Jeffrey D. Ullman (2001). *Automata Theory, Language and Computation*. Addison-Wesley Publishing.
2. Michael Sipser (1997). *Introduction to the Theory of Computation*. PWS publishing company.
3. Harry R. Lewis, Christos H. Papadimitriou (1997). *Elements of the Theory of Computation*. Prentice Hall.