1. Introducción e historia
- Antecedentes históricos: desde las primeras máquinas sumadoras de Pascal (1640), pasando por los cir
- Motivación: entender la computación como modelos abstractos permite analizar computabilidad, complej

2. Máquinas abstractas
- Definición: dispositivos formales (autómatas) que reaccionan a estímulos externos en tiempo discreto, ac
- Características comunes:
  - Tiempo discreto, estados finitos.
  - Función de transición (posible parcialidad) que define siguiente estado, movimiento de cabezal y, si pro
  - Dependencia de estado + estímulo para determinar la respuesta.
- Clasificaciones:
  - Según propósito: reconocedoras (aceptan/descartan cadenas) vs traductoras (transforman cadenas).
  - Según memoria: finitos, con pila (LIFO), linealmente acotados, Turing.
  - Según determinismo: deterministas (un único siguiente estado) vs no deterministas (múltiples, o transic
- Automatismos vs autonomía:
  - Automatismo: comportamiento completamente prediseñado.
  - Autonomía: capacidad de modificar dinámicamente su propia función de transición o alfabeto de entrad

3. Jerarquía de máquinas y lenguajes
- Jerarquía de Chomsky (gramáticas):
  1. Tipo 3: regulares
  2. Tipo 2: independientes de contexto
  3. Tipo 1: dependientes de contexto
  4. Tipo 0: sin restricciones ■filecite■turn0file2■.
- Jerarquía de máquinas (de menor a mayor poder):
  - Autómata finito → Autómata con pila → Autómata linealmente acotado → Máquina de Turing ■filecite■
- Vínculo isomórfico:
  - Regular ↔ Autómata finito
  - Independiente de contexto ↔ Autómata con pila
  - Dependiente de contexto ↔ Autómata linealmente acotado
  - Sin restricciones ↔ Máquina de Turing ■filecite■turn0file2■.

4. Vínculos y utilidad práctica
- Compiladores:
  - El análisis léxico usa AF para reconocer tokens; el sintáctico, AP para validar estructuras; la fase semá
- Procesamiento de lenguaje natural: analizadores morfológicos (AF), sintácticos (AP) y semánticos avanz
- Modelado de sistemas: UML (diagramas de estados/ secuencia) refleja máquinas de estados; aplica en
- Implementación de algoritmos: representar lógica compleja como máquinas finitas mejora claridad, eficie
- Sistemas industriales y embebidos: control de procesos, robots, electrodomésticos, etc., modelados com
- Identificación de patrones y virus: uso de gramáticas y autómatas para detección sintáctica de anomalías

5. Conceptos fundamentales de compiladores
- Definición: transforma programa fuente (alto nivel) en objeto (lenguaje máquina) mediante análisis (léxico
- Componentes del contexto (IDE): editor, preprocesador, compilador (múltiples pasadas), ensamblador, e
- Notación T: representa lenguajes Fuente (F), Objeto (O) e Implantación (I) del compilador; ejemplifica en

6. Estructura interna de un compilador
- Análisis
  1. Léxico (scanner/tokenización)
  2. Sintáctico (parser LL/LR, validación con AP)
  3. Semántico (comprobación de tipos, flujo, coherencia)
- Síntesis