Adding Nondeterminism

Regular Languages

Configuration: one state

DFA

Adding Nondeterminism to DPDA

\[ \delta : Q \times \Sigma \times \Gamma_e \rightarrow Q \times \Gamma_e \]

Languages recognized: ?

Configuration: one state + one stack

DPDA

NPDA

Adding Nondeterminism to DPDA

\[ \delta : Q \times \Sigma \times \Gamma_e \rightarrow Q \times \Gamma_e \]

Languages recognized: ?

Configuration: one state + one stack

DPDA

NPDA
Example

Acceptance: NPDA accepts $w$ when:

$$\delta^*(q_0, w, \epsilon) \to (q_f, s) \land q_f \in F$$

Accepting State Model

Empty Stack Model

Is the set of languages accepted by NPDAs with each model the same?

$L(\text{NPDA/Empty Stack}) \subseteq L(\text{NPDA/Accepting})$

$L(\text{NPDA/Accepting}) \subseteq L(\text{NPDA/Empty Stack})$

$L(\text{NPDA/Accepting}) \subseteq L(\text{NPDA/Empty Stack})$

$L(\text{NPDA/Accepting}) \subseteq L(\text{NPDA/Empty Stack})$
Open (for us) Questions

L(DPDA/Accepting) =? L(DPDA/Empty Stack)
Why don’t the proofs for NPDAs work for DPDAs?
Are NPDAs more powerful than DPDAs?
(Will answer next week)

What languages cannot be recognized by an NDPDA?
(Will answer next week)

Instead of answering these now, we’ll introduce a different model and show it is equivalent to NPDA

Machine Models

“Kleene Machine”

Modeling Human Intellect

Stephen Kleene* (1909-1994)

“Kleeneliness is next to Gödeliness”

Turing Machine (Alan Turing, 1936)
Modeling Human Computers

Finite Automata
McCulloch and Pitts, A logical calculus of the ideas immanent in nervous activity, 1943
S. C. Kleene, Representation of Events in Nerve Nets and Finite Automata, 1956
Claude Shannon and John McCarthy, Automata Studies, 1956

Our theoretical objective is not dependent on the assumptions fitting exactly. It is a familiar strategem of science, when faced with a body of data too complex to be mastered as a whole, to select some limited domain of experiences, some simple situations, and to undertake to construct a model to fit these at least approximately. Having set up such a model, the next step is to seek a thorough understanding of the model itself.

S. C. Kleene, Representation of Events in Nerve Nets and Finite Automata, 1956

Noam Chomsky
I don’t know anybody who’s ever read a Chomsky book. He does not write page turners; he writes page stoppers. There are a lot of bent pages in Noam Chomsky’s books, and they are usually at about Page 16.

Alan Dershowitz

“I must admit to taking a copy of Noam Chomsky’s Syntactic Structures along with me on my honeymoon in 1961. During odd moments, while crossing the Atlantic in an ocean liner and while camping in Europe, I read that book rather thoroughly and tried to answer some basic theoretical questions. Here was a marvelous thing: a mathematical theory of language in which I could use a computer programmer’s intuition! The mathematical, linguistic, and algorithmic parts of my life had previously been totally separate. During the ensuing years those three aspects became steadily more intertwined; and by the end of the 1960s I found myself a Professor of Computer Science at Stanford University, primarily because of work that I had done with respect to languages for computer programming.”

Donald Knuth
Recursion \equiv \text{Human?}

We hypothesize that faculty of language in the narrow sense (FLN) only includes recursion and is the only uniquely human component of the faculty of language. We further argue that FLN may have evolved for reasons other than language, hence comparative studies might look for evidence of such computations outside of the domain of communication (for example, number, navigation, and social relations).


Steven Pinker and Ray Jackendoff (2004): its not just recursion...

Kanzi’s Language

Languages Modeling Computing

**Machines**: String is in the language if machine accepts that input string.

Power of a *machine type* is defined by the set of languages it can recognize.

**Generative grammars**: String is in the language if grammar can produce that input string.

Power of a *grammar type* is defined by the set of languages it can recognize.

Finite Languages

**Machine**: simple lookup table

DFA with no cycles

**Grammar**: grammar with no cycles

\[ A \rightarrow \text{terminals} \]
Regular Languages

**Machine:** DFA

**Grammar:** Regular Grammar

\[ A \rightarrow aB \]
\[ A \rightarrow a \]

*Hint: PS2, Problem 9*

---

$L(DFA) \equiv L(Regular\ Grammar)$

---

Context-Free Grammars

\[ A \rightarrow BCD \]

**Match:** one nonterminal

**Replacement:** any sequence of terminals and nonterminals

Can a CFG generate the language \( a^n b^n \)?

\[ S \rightarrow aSb \]
\[ S \rightarrow \epsilon \]

---

\( \{ w \mid w \text{ contains more } a\text{s than } b\text{s} \} \)

---

Context-Free Languages

**Machine:** NPDA

**Grammar:** Context-Free Grammar

\[ A \rightarrow BCD \]

*Left side: one nonterminal
Replacement: any sequence of terminals and nonterminals*

\[ L(NDPDA) \equiv L(CFG) \]

1. \( L(NDPDA) \subseteq L(CFG) \)

\[ A \rightarrow BCD \quad \Rightarrow \quad \text{pop } A, \text{ push } D, C, B \text{ on stack} \]

\[ A \rightarrow a \quad \Rightarrow \quad \text{ } \]

Accept: empty stack

(Detailed Proof: Sipser, Section 2.2)
\( L(NDPDA) \equiv L(CFG) \)

2. \( L(CFG) \subseteq L(NDPDA) \)

Need to model states also!

More Powerful Grammars

- **Context-Free Grammar**
  \[ A \rightarrow BCD \]
  \[ A \rightarrow a \]

- **Context-Sensitive Grammar**
  \[ XAY \rightarrow XBCDY \]

- **Unrestricted Grammar**
  \[ XAY \rightarrow BCD \]

Recap Questions
- How can you prove a grammar is regular?
- How can you prove a language is regular?
- How can you prove a language is not regular?
- How can you prove a grammar is context-free?
- How can you prove a language is context-free?
- How can you prove a language is not context-free?

Charge
- PS2 is due Tuesday
- **Human Languages**
  - Are they finite, regular, context-free, context-sensitive? (Is the human brain a DFA, PDA, etc. or something entirely different?)
- Next week:
  - Non-Context-Free Languages
  - Parsing, Applications of Grammars