One-Slide Summary

- **Recursive transition networks** and Backus-Naur Form context-free grammars are equivalent formalisms for specifying formal languages.
- Musical harmony contains an explicit notion of a stack. Music starts on the tonic, adds elements in a structured way, and returns to the tonic.
- **find-closest** is quite powerful. Problem sets?
- **L-system fractals** are based on a rewriting system that is very similar to BNF grammars.
- We can use our CS knowledge up to this point to defeat the evil scourge of Cracker Barrel pegboard puzzles! (More next time.)

Outline

- Recursive Transition Networks
  - vs. Backus-Naur Form Grammars
- Stacks and Musical Harmony
- Playing Poker
  - Revenge of find-closest
- That Cursed Pegboard!
  - Problem Representation
  - Important Functions

Recursive Transition Networks

Can we describe this using Backus Naur Form?

With Special Guests: The CS 150 Chorus!
Recursive Transition Networks

ORNATE NOUN ::= ARTICLE ADJECTIVE NOUN
ORNATE NOUN ::= ARTICLE ADJECTIVE ADJECTIVE NOUN
ORNATE NOUN ::= ARTICLE ADJECTIVE ADJECTIVE ADJECTIVE ADJECTIVE NOUN
ORNATE NOUN ::= ARTICLE ADJECTIVE ADJECTIVE ADJECTIVE ADJECTIVE ADJECTIVE NOUN

Recursive Transition Networks

ORNATE NOUN ::= OPTARTICLE ADJECTIVES NOUN
ADJECTIVES ::= ADJECTIVE ADJECTIVES
ADJECTIVES ::= ε

Which notation is better?

Music Harmony

Kleines Harmonisches Labyrinth
(Little Harmonic Labyrinth)

Hey Jude

• John Lennon and Paul McCartney, 1968

Breakdown of Lyrics to "Hey Jude"

Tonic: Hey Jude, don't make it
V: bad. take a sad song and make it
Tonic: better Re-
IV: member to let her into your
Tonic: heart, then you can
V: start to make it bet-
Tonic: -ter.

Hey Jude
Music

- Almost All Music Is Like This
  - Pushes and Pops the listener's stack, but doesn't get too far away from it
  - Repeats similar patterns in a structured way
  - Keeps coming back to the Tonic, and ends on the Tonic
- Any famous Beatles song that doesn't end on the tonic?

HeyJude ::= Verse VBBN VBBN Verse Verse Better Coda
VBBN ::= Verse Bridge Bridge Nanana (ends on C)
Coda ::= F Eb Bb F

Liberal Arts Trivia: Media Studies

- This 1988 book by Herman and Chomsky presented the seminal "propaganda model", arguing that as news media outlets are run by corporations, they are under competitive pressure. Consider the dependency of mass media news outlets upon major sources of news, particularly the government. If a particular outlet is in disfavor with a government, it can be subtly 'shut out', and other outlets given preferential treatment. Since this results in a loss in news leadership, it can also result in a loss of viewership. That can itself result in a loss of advertising revenue, which is the primary income for most of the mass media (newspapers, magazines, television). To minimize the possibilities of lost revenue, therefore, outlets will tend to report news in a tone more favorable to government and business, and giving unfavorable news about government and business less emphasis.

Problem Sets

- Not just meant to review stuff you should already know
  - Get you to explore new ideas
  - Motivate what is coming up in the class
- The main point of the PSs is learning, not evaluation
  - Don’t give up if you can’t find the answer in the book (you won’t solve many problems this way)
  - Do discuss with other students

PS2: Question 4

Why is
   (define (higher-card? card1 card2)
      (> (card-rank card1) (card-rank card2))
   )
better than
   (define (higher-card? card1 card2)
      (> (car card1) (car card2))
   )
?
PS2: Question 9, 10

- Predict how long it will take
- Identify ways to make it faster

Most of next week and much of many later classes will be focused on how computer scientists predict how long programs will take, and on how to make them faster.

Can we do better?

This is what we used in PS2 for our Poker-Bot:

```
(define (find-best-hand hole-cards community-cards)
  (car (
    sort (possible-hands hole-cards community-cards)
    higher-hand?))
```

Hmmm....

```
(define (find-closest goal lst closeness)
  (if (= 1 (length lst))
    (car lst)
    (pick-closest closeness goal (car lst)
     (find-closest goal (cdr lst) closeness))))

(define (pick-closest closeness goal num1 num2)
  (if (< (closeness goal num1)
        (closeness goal num2))
    num1
    num2))
```

We could use these to find the best hand!

find-bestest

```
(define (find-bestiest lst bestiness)
  (if (= 1 (length lst))
    (car lst)
    (pick-bestier bestiness...
     num2)
   (if (bestiness num1 num2)
       num1
       num2))
```

This used to be

```
(< (dist num1 goal)
  (dist num2 goal))
```

find-best-hand

```
(define (find-bestiest lst bestiness)
  (if (= 1 (length lst))
    (car lst)
    (pick-bestier bestiness...
     num2))

(define (find-best-hand lst)
  (find-bestest lst higher-hand?))
```

Next week: how much better is this?

PS3: Lindenmayer System Fractals
L-Systems

CommandSequence ::= ( CommandList )
CommandList ::= Command CommandList
CommandList ::= Command
Command ::= F
Command ::= RAngle
Command ::= OCommandSequence

L-System
Rewriting
Start: (F)
Rewrite Rule: 
F →  (F O(R30 F) F O(R-60 F) F)
Work like BNF replacement rules, except replace all instances at once!

Why is this a better model for biological systems?

The Great Lambda Tree of Ultimate Knowledge and Infinite Power
(Level 5 with color)

Previous CS 150 Students:
Rose Bush by Jacintha Henry and Rachel Kay
Tie Dye by Bill Ingram
Liberal Arts Trivia: Medicine

- This vector-borne infectious disease is caused by protozoan parasites. It is widespread in tropical regions, such as sub-Saharan African. Each year there are about 515 million cases of it, killing between one and three million people. No formal vaccine is available. Classic symptoms include sudden coldness followed by rigor and then fever and sweating.

Liberal Arts Trivia: Accounting

- In this bookkeeping system, each transaction is recorded in at least two accounts. Each transaction results in one account being debited and another account being credited, with the total debits equal to the total credits. Luca Pacioli, a monk an collaborator of Leonardo da Vinci, is called the “father of accounting” because he published a usable, detailed description of this system.

Pegboard Puzzle

- We’ll use an (x,y) notation to represent positions on the pegboard.

```
1,1
2,1 2,2
3,1 3,2 3,3
4,1 4,2 4,3 4,4
5,1 5,2 5,3 5,4 5,5
```

Solving the Pegboard Puzzle

- How to represent the state of the board?
  - Which holes have pegs in them
- How can we simulate a jump?
  - board state, jump positions \( \rightarrow \) board state
- How can we generate a list of all possible jumps on a given board?
- How can we find a winning sequence of jumps?

Data Abstractions

```
(define (make-board rows holes)
  (cons rows holes))

(define (board-holes board) (cdr board))
(define (board-rows board) (car board))

(define (make-position row col) (cons row col))

(define (get-row posn) (car posn))
(define (get-col posn) (cdr posn))

(define (same-position pos1 pos2)
  (and
   (= (get-row pos1) (get-row pos2))
   (= (get-col pos1) (get-col pos2))))
```

We are defining our data structures!
Example Board: “Grey”

(define grey-rows 5)
(define grey-holes (list (make-position 1 1) (make-position 2 1) (make-position 2 2) (make-position 3 2))
(define grey-board (make-board grey-rows grey-holes))

Removing a Peg

;;; remove-peg evaluates to the board you get by removing a peg at posn from the passed board
;;; (removing a peg adds a hole)

(define (remove-peg board posn)
  (make-board (board-rows board)
              (cons posn (board-holes board))))

Adding a Peg

;;; add-peg evaluates to the board you get by adding a peg at posn to board
;;; (adding a peg removes a hole)

(define (add-peg board posn)
  (make-board (board-rows board)
               (remove-hole (board-holes board) posn)))

Remove Hole

(define (remove-hole lst posn)
  (if (same-position (car lst) posn)
      (cdr lst)
      (cons (car lst) (remove-hole (cdr lst) posn))))

Could we instead define remove-hole using map?

No. \((\text{length} \ (\text{map} \ f \ \text{lst}))\) is always the same as \((\text{length} \ \text{lst})\), but remove-hole needs to remove elements from the list.

What if we had a procedure \((\text{filter} \ \text{proc} \ \text{lst})\) that removes from \text{lst} all elements for which \text{proc} (applied to that element) is false?

Oh, wait! We do!

Filter

(define (filter pred lst)
  (if (null? lst)
      null
      (if (pred (car lst)); pred is true, keep it
          (cons (car lst) (filter proc (cdr lst)))
          (filter pred (cdr lst))))))

> (filter (lambda (x) (> x 0)) (list 1 4 -3 2 -8))

\((1 \ 4 \ 2)\)
Filter Remove

(define (filter pred lst)
  (if (null? lst)
      null
      (if (pred (car lst)) ; pred is true, keep it
        (cons (car lst) (filter proc (cdr lst)))
        (filter pred (cdr lst)))) ; pred is false, drop it)

(define (remove-hole lst posn)
  (filter (lambda (pos)
            (not (same-position pos posn)))
           lst))

Solving the Peg Board Game

• Try all possible moves on the board
• Try all possible moves from the positions you get after each possible first move
• Try all possible moves from the positions you get after trying each possible move from the positions you get after each possible first move
• …

Jumps

;;; move creates a list of three positions: a start (the posn that the jumping peg starts from), a jump (the posn that is being jumped over), and end (the posn that the peg will end up in)

(define (make-move start jump end) (list start jump end))
(define (get-start move) (first move))
(define (get-jump move) (second move))
(define (get-end move) (third move))

;;; execute-move evaluates to the board after making move on board.
(define (execute-move board move)
  (add-peg (remove-peg (remove-peg board (get-start move)))
            (get-jump move))
  (get-end move)))

Homework

• Read Course Book Chapter 6 before Monday
• Start on reading Chapter 7
  - As soon as it’s available …