Lecture 12 Notes

• **Goals for this week**
  – Conditional probability and Bayes’ rule
  – Induction and recursion
  – For **quiz**: Today’s material

• **Feedback survey posted** (anonymous google doc)
  – Please give your comments
  – Small appreciation for class response rate of 85% by Thursday night ⇒ lowest quiz score will be dropped

• **Grading principles and policy** (see Piazza post)
  – Guarantees: 90 = A-, 80 = B-, 70 = C-, 60 = D
  – “Midterm consolidation” can recover up to half of the gap between your MT score and 90 points
    • 114 students with scores < 90. 75 students with scores ≥ 90.
  – Sunday, May 18, 6pm in CSE building Rooms 1202, 2154, 4140

• **Poker chips**
  – Today’s Px.y problems are on **induction**
  – *Can trade your current poker chips for nicer CSE 21 chips …*
Main Points From Last Time

• Baseball card collector: $n \log n$ packs of cards
  – Defined $X_i =$ number of packs needed to get the $i^{th}$ distinct baseball card
  – Used two facts
    • Linearity of expectation $E(X) = E(\sum X_i) = \sum E(X_i)$
    • If $P(\text{success}) = r$, expect $1/r$ trials up to and including first success

• Decision trees
  – Sorting: comparison tree = a sorting algorithm
    • Worst-case and average-case complexities of sorting by comparisons
  – Counting: non-increasing functions
  – Implicit enumeration (“smart brute-force”) for hard problems: traveling salesman problem, N-queens, graph coloring
  – Next: recursion and induction
Definition 4: Conditional Probability

- Let $U$ be a sample space with probability function $P$.
- For events $A, B \subseteq U$, the conditional probability of $B$ given $A$ is:

$$P(B \mid A) = \frac{P(A \cap B)}{P(A)} \quad \text{if } P(A) \neq 0$$

$$= \text{undefined} \quad \text{if } P(A) = 0$$

- Notice:

$$P(A \cap B) = P(B \mid A) \cdot P(A)$$

$$= P(A \mid B) \cdot P(B)$$

- $P(A_1 \cap \ldots \cap A_n) = P(A_1)P(A_2 \mid A_1)P(A_3 \mid A_1 \cap A_2)\ldots P(A_n \mid A_1 \cap \ldots \cap A_{n-1})$

Intuition: cascade
Conditional Probability, Bayes’ Rule

• Probability of A and B, i.e., $P(A \cap B)$

\[ P(A \cap B) = P(B \mid A) \cdot P(A) \]  
probability of B given A, times probability of A

\[ P(A \cap B) = P(A \mid B) \cdot P(B) \]  
symmetrical observation

\[ P(A \mid B) \cdot P(B) = P(B \mid A) \cdot P(A) \]

\[ \Rightarrow P(A \mid B) = \frac{P(B \mid A) \cdot P(A)}{P(B)} \]  
Bayes’ Rule / Theorem
Properties of Conditional Probability

• **Fact:** \( P(B|U) = P(B) \)
  Conditioned on everything, we don’t get any new knowledge

• **Fact:** \( A \) and \( B \) are independent iff \( P(B \mid A) = P(B) \)

By definition (Lecture 10):

\( A \) and \( B \) are independent iff \( P(A \cap B) = P(A) \cdot P(B) \)

Also: \( P(A \cap B) = P(B \mid A) \cdot P(A) \)

\[ \Rightarrow \] Divide by \( P(A) \) to get **Fact**

Independence means that conditioning on new information doesn’t change probability
A student knows 80% of the material on an exam. What is the probability that he answers a given true-false question correctly?

Knows material: correct with $P = 1$

Doesn’t know material: correct with $P = 0.5$
PED Testing in the NFL

• A drug test detects PED use 95% of the time. However, 15% of PED-free individuals also test positive (= false-positive rate). 10% of NFL players use PEDs. Your NFL player friend tests positive. What is the probability that he actually did use PEDs?

• Estimate:

  A. Close to 100%
  B. Close to 50%
  C. Close to 15%
  D. Close to 10%
  E. None of the above
PED Testing in the NFL

• A drug test detects PED use 95% of the time. However, 15% of PED-free individuals also test positive (= false-positive rate). 10% of NFL players use PEDs. Your NFL player friend tests positive. What is the probability that he actually did use PEDs?

Define: Event A = your friend used PEDs

Event B = your friend tests positive

We want: $P(A | B)$

We know: $P(B | A) =$  
$P(B | A^c) =$  
$P(A) =$
PED Testing in the NFL

Define: Event A = your friend used PEDs
Event B = your friend tests positive

We know:
- \( P(B \mid A) = 0.95 \)
- \( P(B \mid A^c) = 0.15 \)
- \( P(A) = 0.1 \)

We want: \( P(A \mid B) = \frac{P(B \mid A) \cdot P(A)}{P(B)} \) Bayes’ Rule

\[
= \frac{P(B \mid A) \cdot P(A)}{P(B)} = \frac{0.95 \cdot 0.1}{0.95 \cdot 0.1 + 0.15 \cdot 0.9}
= \frac{0.095}{0.095 + 0.135} = 0.413
\]
Bayesian Spam Filters

• The word “Rolex” occurs in 250 of 2000 messages known to be spam and in 5 out of 1000 messages known not to be spam.

• (1) Estimate the probability that an incoming message containing the word “Rolex” is spam, assuming that it is equally likely that an incoming message is spam or not spam.

• (2) If the threshold for rejecting a message as spam is 0.9, will this message be rejected?

A. About 100%, will be rejected
B. About 80%, will be rejected
C. About 50%, will not be rejected
D. About 20%, will not be rejected
E. None of the above.
Bayesian Spam Filtering

• The word “Rolex” occurs in 250 of 2000 messages known to be spam and in 5 out of 1000 messages known not to be spam. Estimate the probability that an incoming message containing the word “Rolex” is spam, assuming that it is equally likely that an incoming message is spam or not spam.

Define: Event A =

Event B =

We want:
Bayesian Spam Filtering

Define: Event $A$ = the message is spam
        Event $B$ = it contains the word “Rolex”

We know:

$P(B \mid A) = \frac{250}{2000} = 0.125$
$P(B \mid A^c) = \frac{5}{1000} = 0.005$
$P(A) = 0.5$

We want: $P(A \mid B) = \frac{P(B \mid A) \cdot P(A)}{P(B)}$ Bayes’ Rule

$= \frac{P(B \mid A) \cdot P(A)}{P(B)}$

$= \frac{0.125 \cdot 0.5}{0.125 \cdot 0.5 + 0.005 \cdot 0.5}$

$= 0.962$
Market Research

• An electronics company commissions a marketing report for each new product that predicts the success or failure of the product. Of the new products that have been introduced by the company, 60% have been successes. Furthermore, 70% of the successful products were predicted to be successes, while 40% of the failed products were predicted to be successes. Find the probability that a new smart phone will be successful, given that success is predicted.

• Define: Event A =
  Event B =

Exercise: Should get $P(A \mid B) \approx 0.724$
“Lets Make A Deal”: The Monty Hall Puzzle

• TV game show

- Car hidden behind one of three doors. Goats behind other two.
- Player chooses a door.
- Host opens another door, reveals a goat.
- Player can choose whether to switch doors or stay.

**Optimal strategy:**

A. The player should always switch.
B. The player should always stay.
C. It doesn’t matter, same chance of winning.
Analyze Winning Probabilities

Decision tree!

Contestant chooses: Goat 1 | Goat 2 | Car
1/3 | 1/3 | 1/2

Host shows: Goat 2 | Goat 2 | Goat 2 | Goat 1
1/3 | 1/3 | 1/3 | 1/3

Contestant switches? Y | N | Y | N | Y | N | Y | N
1/3 | 1/3 | 1/3 | 1/3 | 1/3 | 1/3 | 1/3 | 1/3

Contestant gets: C | G1 | C | G2 | G2 | C | G1 | C
1/3 | 1/3 | 1/3 | 1/3 | 1/6 | 1/6 | 1/6 | 1/6

Switch: Car: 2/3 | Goat 1: 1/6 | Goat 2: 1/6
No Switch: Car: 1/3 | Goat 1: 1/3 | Goat 2: 1/3
Definition 2: Recursive Approach

A recursive approach to a problem consists of two parts:

- The problem is reduced to one or more problems of the same kind which are simpler in some sense.
- There is a set of simplest problems to which all others are reduced after one or more steps. Solutions to these simplest problems are given.

Examples

- CSE11: “ReverseRecurse”
- CSE8B: “printReverseDigits”, longest common subsequence
- Fibonacci numbers, matrix determinant, Tower of Hanoi, Mergesort, Matrix multiplication, fast Fourier transform, …
- Combinations, Sterling numbers, Shortest paths, …

⇒ Divide-and-conquer, dynamic programming algorithms
Divide and Conquer ("DQ")

- Framework for solving problems recursively

\[
\text{DQ}(S) \\
\text{if } S \text{ is small return ADHOC}(S) \\
\text{else} \\
\quad \text{decompose } S \text{ into subproblems } S_1, \ldots, S_k \quad \text{// divide} \\
\quad \text{for } i = 1 \text{ to } k \text{ do } y_i = \text{DQ}(S_i) \quad \text{// conquer} \\
\quad \text{recombine } y_i \text{ into solution } y \quad \text{// compose} \\
\quad \text{return } y
\]

- "Universal method": Mergesort, Quicksort, FFT, Matrix/Integer arithmetic are classic examples
DQ: Mergesort

• Sorting a list of numbers
  – Divide (into two equal parts)
  – Conquer (solve for each part separately)
  – Combine separate solutions

If list has length one, it’s already sorted (return list)
Else
  Sort first half of list (recursively)
  Sort second half of list (recursively)
  Merge the two sorted lists
Merging Two Sorted Subsequences

Sequence #1: \( x[1], x[2], \ldots, x[m] \)
Sequence #2: \( y[1], y[2], \ldots, y[n] \)

if \( y[i] < x[j] \) then compare \( y[i+1] \) and \( x[j] \),
else compare \( y[i] \) and \( x[j+1] \)

After comparing \( x[1] \) and \( y[1] \), advance in \( x[.] \) and \( y[.] \)

total of \((m-1) + (n-1)\) times \(\Rightarrow m + n - 1\) comparisons
(edges between red and blue) = linear time
Tree of Recursions in Mergesort

Check out:  https://www.youtube.com/watch?v=XaqR3G_NVoo

• n comparisons per level
• log n levels
• total runtime looks to be around .... n \cdot \log n
More Precisely ...

- Let $M(n)$ denote Mergesort runtime on list of $n$ numbers

- $M(n) = 1$ if $n = 1$
  
  $= M(n/2) + M(n/2) + n$ if $n > 1$

- Claim: $M(n) = n \cdot \log_2 n + n \quad \forall n \geq 1$

How can we prove this?
Mathematical Induction

• M(n) = Mergesort runtime on list of n numbers
• M(n) = 1 if n = 1
  = M(n/2) + M(n/2) + n if n > 1

• Claim: M(n) = n \cdot \log_2 n + n \quad \forall n \geq 1

• Base case: M(1) = 1 \cdot \log_2 1 + 1 = 1 \cdot 0 + 1 = 1 \checkmark

• Induction step: Let n be a fixed integer and suppose M(k) = k \cdot \log_2 k + k for all k < n. We want to show that M(n) = n \cdot \log_2 n + n.

  \begin{align*}
  M(n) &= 2 \cdot M(n/2) + n \\
  &= 2 \cdot [(n/2) \cdot \log_2 (n/2) + (n/2)] + n \quad \text{by I.H.}
  \end{align*}
Mathematical Induction

- \( M(n) = \text{Mergesort runtime on list of } n \text{ numbers} \)
- \[ M(n) = \begin{cases} 1 & \text{if } n = 1 \\ M(n/2) + M(n/2) + n & \text{if } n > 1 \end{cases} \]
- **Claim:** \( M(n) = n \cdot \log_2 n + n \quad \forall n \geq 1 \)
- **Base case:** \( M(1) = 1 \cdot \log_2 1 + 1 = 1 \cdot 0 + 1 = 1 \)
- **Induction step:** Let \( n \) be a fixed integer and suppose \( M(k) = k \cdot \log_2 k + k \) for all \( k < n \). We want to show that \( M(n) = n \cdot \log_2 n + n \).

\[
M(n) = 2 \cdot M(n/2) + n \\
= 2 \cdot [(n/2) \cdot \log_2(n/2) + (n/2)] + n \quad \text{by I.H.} \\
= n(\log_2 n - \log_2 2) + n + n = n \cdot \log_2 n - n + n + n \\
= n \cdot \log_2 n + n \quad \checkmark
\]
Another Recursion: Tower of Hanoi

• One disk moves at a time
• Never put larger disk onto smaller disk

What is minimum #moves needed to transfer a stack of n disks?

• Notation
  – For a stack of n disks, call this number $T_n$

• Small Cases
  – $T_0 = 0$, $T_1 = 1$, $T_2 = 3$, $T_3 = 7$, …
Tower of Hanoi (cont.)

• Can we “reduce to a known problem”?
  – Note that this is the idea of recursion!
  – \( T_n \leq 2T_{n-1} + 1, \quad n > 0 \)
  – Why?
    Shift (n-1), move largest disk, shift again
  – Why the \( \leq \) inequality? = an **upper** bound
    \( 2T_{n-1} + 1 \) suffices, but maybe can do better
Tower of Hanoi (cont.)

- Can we “reduce to a known problem”?  
  - Note that this is the idea of recursion!
  - $T_n \leq 2T_{n-1} + 1$, $n > 0$
  - Why?
    - Shift (n-1), move largest disk, shift again
  - Why the $\leq$ inequality? = an upper bound
    - $2T_{n-1} + 1$ suffices, but maybe can do better

- Why does a lower bound $T_n \geq 2T_{n-1} + 1$ hold?  
  - Must move largest disk sometime
  - At this instant, have (n-1) disks on a single peg
  
  \[ \Rightarrow T_n = 2T_{n-1} + 1, \quad T_0 = 0 \]

- What is a general solution?
Tower of Hanoi (cont.)

• General solution looks like … \( T_n = 2^n - 1 \)
• Let’s guess this answer and try to prove it

• **Claim**: \( T_n = 2^n - 1 \)
• **Proof**: (by mathematical induction)
  
  Base case (\( n = 0 \)): \( T_0 = 0 = 2^0 - 1 \) holds
  
  I.H.: \( T_k = 2^k - 1 \) \( \forall k = 0, 1, \ldots, n - 1 \)
  
  I.S.: \( T_n = 2T_{n-1} + 1 = 2(2^{n-1} - 1) + 1 = 2^n - 1 \) (I.H.)
Another Recursion: Matrix Multiplication

- $A = [...]$, $B = [...]$ are $n \times n$ matrices
- $a_{11}, a_{12}, \text{etc.}$ are $n/2 \times n/2$ submatrices
- $M = A \cdot B = [...]$
  - where $m_{11} = a_{11}b_{11} + a_{12}b_{21}$ etc.
  - Evaluation requires 8 multiplies, 4 adds
- Divide and Conquer: $T(n) = 8T(n/2) + n^2$

$$
\begin{bmatrix}
a_{11} & a_{12} \\
a_{21} & a_{22}
\end{bmatrix}
\begin{bmatrix}
b_{11} & b_{12} \\
b_{21} & b_{22}
\end{bmatrix}
= 
\begin{bmatrix}
m_{11} & m_{12} \\
m_{21} & m_{22}
\end{bmatrix}
$$
Another Recursion: Long Multiplication

• Multiplying Large Integers
  \[ A = r^{n-1}a_{n-1} + \ldots + r^1a_1 + a_0, \quad r = \text{radix} \]

• Method that you learned 10 years ago:
  \[ \begin{array}{c}
  a_{n-1}a_{n-2}\ldots a_1a_0 \\
  \times \phantom{a_{n-1}a_{n-2}\ldots a_1a_0} \\
  b_{n-1}b_{n-2}\ldots b_1b_0
  \end{array} \]
  
  -----------------------------------------------

• What would “Divide and Conquer” look like?
Another Recursion: Determinant

• det (2 × 2 matrix) = ad – bc

• Recursively defined det(M) = (…)
  – M_{1j} is the (1, j) cofactor matrix (or, minor) of n x n matrix M

• det(M) = m_{11} \cdot \text{det}(M_{11}) + … + (-1)^{1+j} m_{1j} \cdot \text{det}(M_{1j}) + …

• With this recursive definition, need ~ n! multiplications to calculate the determinant of an n x n matrix

• What would be a “better method”?
Problems 12 – Induction !!!

• P12.1 In Transylvania, every road is one-way. And, every pair of cities is connected by exactly one direct road. Show that there exists a city which can be reached from every other city either directly or via at most one other city.

• P12.2 A planar map consists of N vertices, along with some number of edges drawn between pairs of vertices, with no two edges crossing. A planar map dissects the plane into disjoint regions, one of which is the infinite region. A map is properly colored if each region is assigned a color, and no two regions whose boundaries share an edge have the same color. Show that the regions of a map can be properly colored with two colors if and only if all of its vertices have even degree.
Problems 12 – Induction !!!

• P12.3 Let $n = 2k$. Prove that one can select $n$ integers from any $(2n – 1)$ integers such that their sum is divisible by $n$.

• P12.4 A knight is located at the origin of an infinite chessboard. State, and prove by induction, an expression for the number of squares that the knight can reach after exactly $n$ moves. (Note: If $f(n)$ is the number of squares on which the knight could be after exactly $n$ moves, then $f(0) = 1$, $f(1) = 8$, $f(2) = 33$, $f(3) = 76$, etc.)
Hashing

• **Definition**: A hash function maps data of variable length to data of a fixed length. The values returned by a hash function are called *hash values* or simply *hashes*.

• **Open Hashing**: Each item hashes to a particular location in an array, and locations can hold more than one item. Multiple input data values can have the same hash value. These are called *collisions*. 
Number of Items Per Location

• n items, hash to table of size k

• What is expected number of items per location?

• n independent trials with outcome = location in table (i.e., hash value)

• Let r.v. \( X = \# \) inputs that hash to location 1
  – Let’s write \( X \) as the sum of \( X_i \)'s:
    • \( X_i = 1 \) if the \( i^{th} \) input hashes to location 1
    • \( X_i = 0 \) otherwise
  \( \Rightarrow X = X_1 + X_2 + \ldots + X_n \)
  \( \Rightarrow E(X) = E(X_1) + E(X_2) + \ldots + E(X_n) \)
  \( = n \cdot 1/k = n/k \)
Number of Empty Locations

• What is expected number of empty locations?

Define r.v. $Y = \# \text{ empty locations}$. We want $E(Y)$

Example: 20 items into table of size 5: $E(Y) = 5 \cdot (4/5)^{20} \approx 0.06$
Example: 20 items into table of size 10: $E(Y) = 10 \cdot (9/10)^{20} \approx 1.22$