CS 173
August 28, 2018

Professor Tandy Warnow

Key Objectives

- Teach you to do proofs by induction and by contradiction
- Teach you to think inductively, and use this approach to design provably correct algorithms
- Teach you think think logically
- Teach you to communicate clearly without too much notation (e.g., to write good pseudo-code) about algorithms, and why they are correct
- Teach you to recognize common graph-theoretic problems and use existing methods to solve real world problems
- Teach you to design new algorithms using basic algorithm design techniques (e.g., dynamic programming)
- Teach you to analyze running time and communicate this with Big-O notation

Why is this course difficult?

- This course is a pre-requisite for CS 225 and CS 374, two classes that are difficult because of the requirement that you know how to *prove* things and write clearly using mathematical ideas and notation.
- This course will prepare you well for these classes, so that you'll be comfortable when you take them.
- In addition, this course will help you prepare for being a strong programmer and also for getting into graduate school!
- But don't worry – all of us will help you do well in the course. And although the class is difficult, the grading policy is generous.

Grading

- Participation in discussion section: 10 pts
- Reading Quizzes: 10 pts
- Homework: 20 pts (due Tuesdays at 10PM on Moodle, bottom homework dropped)
- Midterms (October 11 and November 13, evenings): 40 pts
- Final exam: 20 pts

Please note:

- I will grade on a curve so that at least 25% of the class gets an A and at least 60% get B or better.
- The last time I taught this course, **33% got A's, 38% got B's**, so more than 70% got As or Bs, several received A+.
- My goal is for everyone to do well. So if everyone deserves an A, then everyone will get an A!
- The best way to do well in the class is to not fall behind!

Websites

- http://tandy.cs.illinois.edu/cs173-warnow.html - this is the **Course Webpage**, for nearly everything
- Piazza – really just for you
- Moodle – for homework and reading quiz assignments and submissions.

Please look at the course webpage the day before class for the PDF/PPT of the upcoming lecture, announcements, and reading assignments.

Always check Moodle for homework and reading quiz assignment due dates.

Don't use Piazza to get solutions to your homework (nor to post solutions).

Differences between Lectures A and B

Similarities:

- We will both use cover much of the same material
- We will both have homework submitted through Moodle

Differences:

- Different textbooks
- I will not cover number theory or state diagrams (B lecture may)
- I will cover “trees” differently (as handled by Rosen)
- I will give examples from computational biology to illustrate techniques and concepts
- The B lecture will have in-class examlets, the A lecture will have two midterms (evenings)
- The A lecture will have an honors section

Two-person games

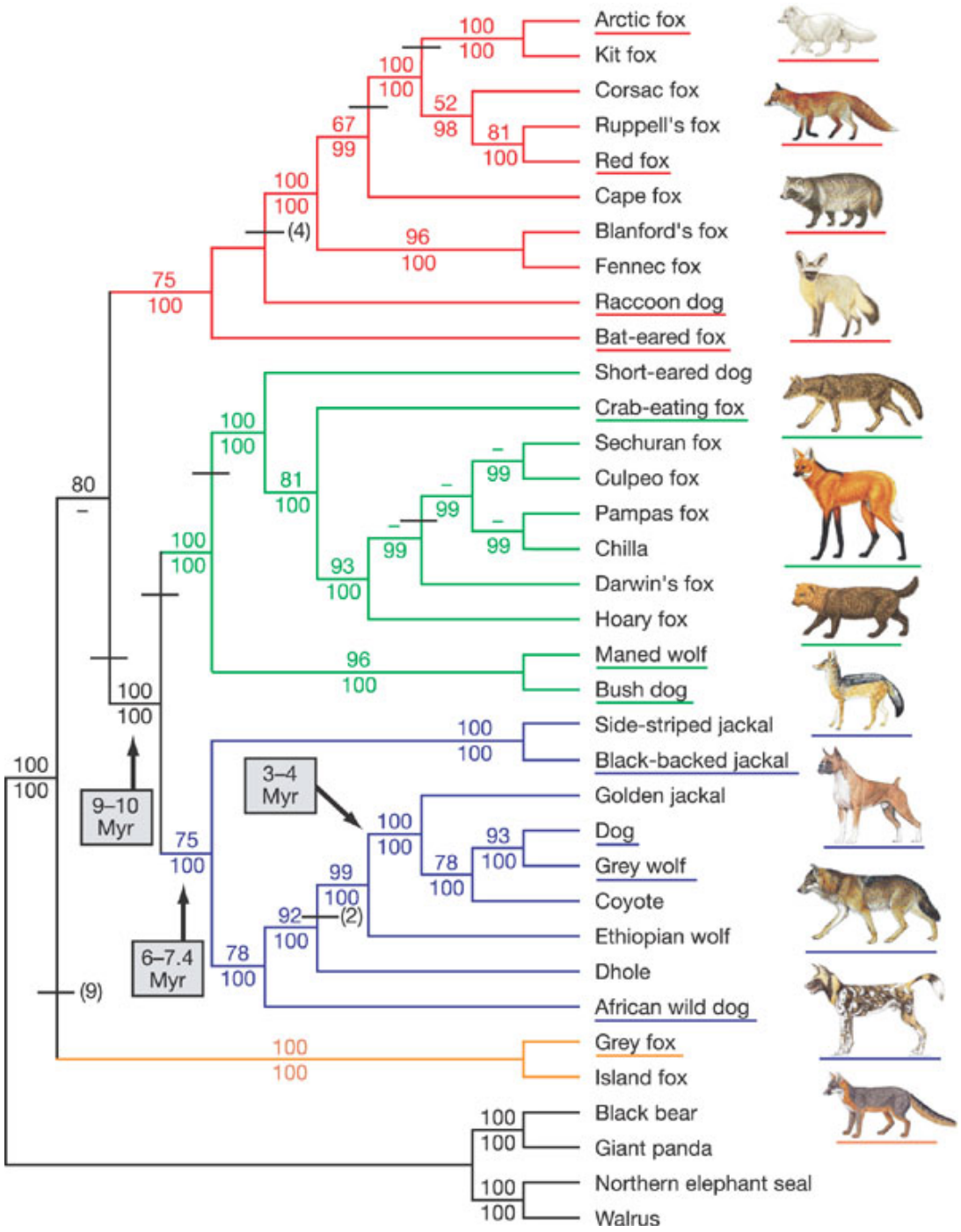
- Two players, A and B. A starts.
- In the beginning there are two piles of stones, with K and L stones respectively.
- During a turn, a player must take at least one stone – the choice is between one stone off of both piles, or one stone off of one of the two piles. The person who takes the last stone wins.
- Who wins when
  - K = 1 and L = 1?
  - K = 2 and L = 1?
  - K = 3 and L = 3?
  - K = 4 and L = 16?
- You can probably figure out a pattern here… but see if you can try to *prove* that you are right. (This is something you'll learn how to do in this class.)
- Spoiler: this can be solved using dynamic programming, and the proof of correctness uses induction

Another two-person game

- Again two players, A and B. A begins. The starting position has two piles of stones, with K and L stones.
- During a turn, the player can take 1 or 2 stones off in total, and these can be from the same pile, or from different piles.
- Who wins
  - K=2 and L=1?
  - K=2 and L=2?
  - K=101 and L =47?
- Figuring out who has a winning strategy is harder here, but still feasible. You'll learn how to do this, and prove you are correct, in this class.
- Spoiler: this can be solved using dynamic programming and the proof of correctness uses induction.

Something perhaps more realistic

- Biologists often try to infer how evolution occurred.



Lindblad-Toh et al., Nature 2005

Maximum Parsimony

- **Input**: Set *S* of *n* aligned sequences of length k
- **Output**:
  - A phylogenetic tree T leaf-labeled by sequences in *S*
  - additional sequences of length *k* labeling the internal nodes of *T*

such that $\sum_{(i,j)\in E(T)} H(i,j)$

is minimized, where H(i,j) denotes the Hamming distance between sequences at nodes i and j

Maximum Parsimony

- **Input**: Set *S* of *n* aligned sequences of length k
- **Output**:
  - A phylogenetic tree T leaf-labeled by sequences in *S*
  - additional sequences of length *k* labeling the internal nodes of *T*

such that $\sum_{(i,j)\in E(T)} H(i,j)$

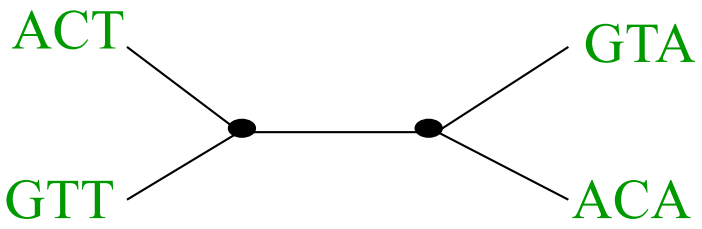
Note: E(T) is a set, denoting the edges of a tree.

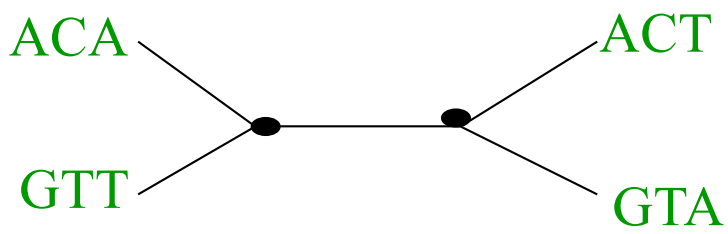
is minimized, where H(i,j) denotes the Hamming distance between sequences at nodes i and j

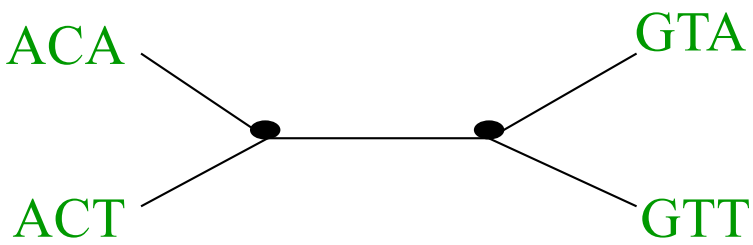
Maximum parsimony (example)

- **Input**: Four sequences
  - ACT
  - ACA
  - GTT
  - GTA
- **Question**: which of the three trees has the best MP scores?

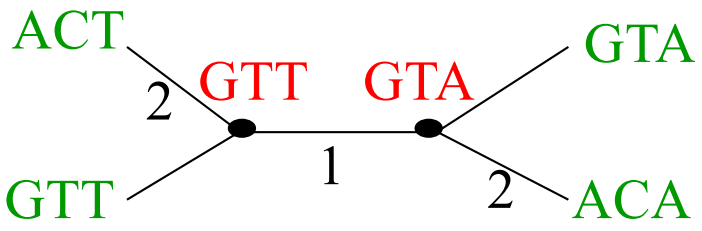
Maximum Parsimony



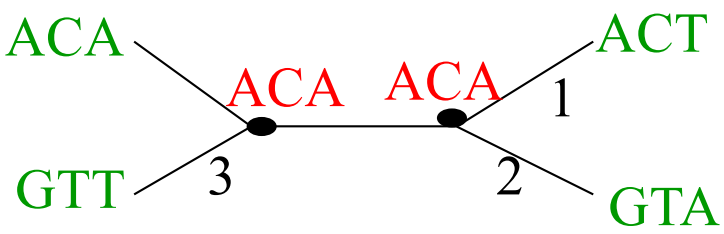




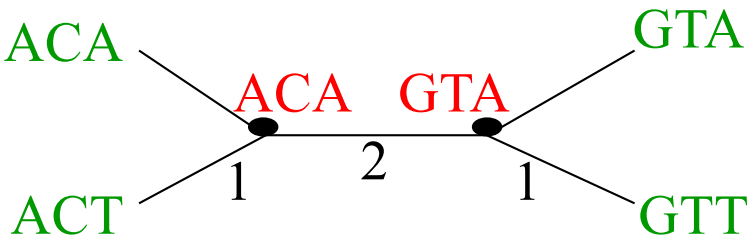
Maximum Parsimony



MP score = 5



MP score = 6

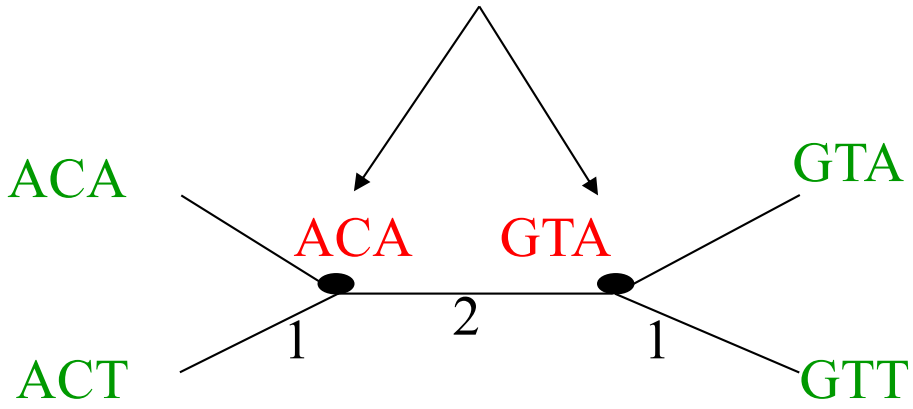


MP score = 4

Optimal MP tree

MP: computational complexity

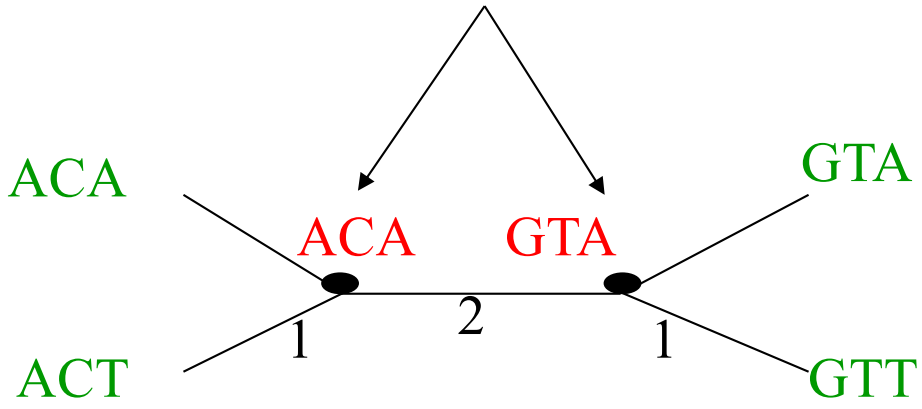
For four leaves, we can do this by inspection



MP score = 4

But finding the best tree is NP-hard!

Optimal labeling can be computed in polynomial time!



MP score = 4

NP-hardness and Algorithms

- Finding the best possible parsimony score of a given tree T (with leaves labelled by DNA sequences) can be computed in polynomial time using an algorithmic technique called “Dynamic Programming”. You will learn how to design dynamic programming algorithms in this course.
- But finding the best possible tree for the sequences is NP-hard. You will learn what that means, and how to design methods that address NP-hard problems.

Solving NP-hard problems exactly is … unlikely

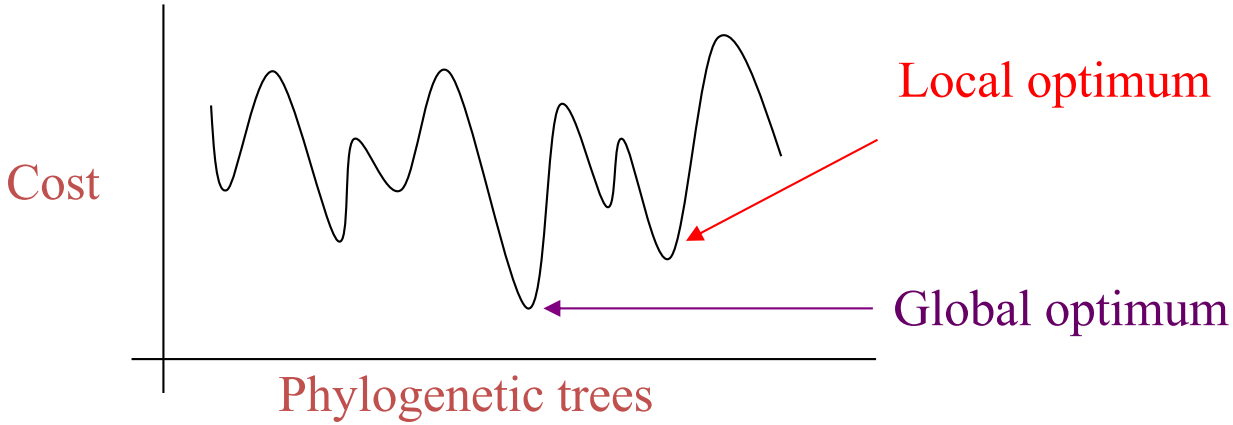
- Number of (unrooted) binary trees on *n* leaves is (2n-5)!!
- If each tree on **1000** taxa could be analyzed in **0.001** seconds, we would find the best tree in

  2890 millennia

#leaves	#trees
4	3
5	15
6	105
7	945
8	10395
9	135135
10	2027025
20	2.2×10^{20}
100	4.5×10^{190}
1000	2.7×10^{2900}

Approaches for "solving" MP

1. Hill-climbing heuristics (which can get stuck in local optima)
2. Randomized algorithms for getting out of local optima
3. Approximation algorithms for MP (based upon Steiner Tree approximation algorithms).



Logic Problem

- Suppose that in the village of LALA, everyone is either a truth teller (and always tells the truth) or a liar (and so always lies).
- You meet two LALA villagers -- Henry and Allen.
- Henry says “Allen is a truth teller”
- Allen says “Only one of us is a truth teller”

Is either a truth teller? If so, who?

Two more problems

- Suppose A is a finite set of integers, and satisfies:
  - Whenever x is an element of A, then 2x is an element of A
  - What is A?
- Suppose B is a set of integers and satisfies:
  - Whenever x is an element of B, then x+1 is an element of B.
  - Can B be finite?

Concepts (so far)

- Two-person games
- Sets
- Trees (a special kind of graph)
- Running time – and “Big-O” notation
- NP-hardness
- Dynamic programming
- Proofs by induction
- Recursive definitions
- Computational biology problems
- Logic problems

Other

- My office hours are Tuesdays, 11 AM-12 PM, in Siebel 3235 (starting today).

- If you are interested in doing research with me, please come see me. I have NSF funding for undergraduate research for qualified students.

Upcoming Assignments

- Reading Assignments: see course webpage http://tandy.cs.illinois.edu/cs173A-2018-lectures.html
- Homework assignments and Reading Quizzes must be submitted on Moodle, and start next week. Check Moodle for the assignments and their deadlines. (Submitting early and then resubmitting is fine – but don't miss the deadline.)
- If you aren't yet registered for the course but are hoping to get into the course, please add yourself to Moodle – the password will be given in class.