CSC 225

Algorithms and Data Structures I
Spring 2013
Venkatesh Srinivasan

Lectures and Labs

Venkatesh Srinivasan

- E-mail: venkat@cs.uvic.ca
- Voice: 472-5731
- Office: ECS 626
- Office hours:
  TF 1:00 pm – 2:30 pm

- Lectures
  - A01/A02 TWF 11:30 – 12:20 pm ECS 116
- Labs
  - Bill Bird
  - Labs start week of Jan 7, 2013
  - B01 Thursday 2:30 – 3:20 pm ECS 104
  - B02 Mon 3:30 – 4:20 pm ECS 104
  - Check UVic website

- Course Web pages
  - Official Webpage on the Department Website
  - Detailed Course Website on ConneX

Administrative Officer Announcements

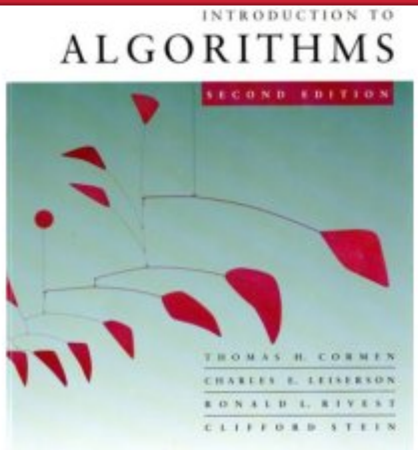
- CSC Administrative Officer is Jane Guy
  E-mail: jguy@csc.uvic.ca   Office: ECS 512
- Any student who has registered in CSC 225 and **does not** have the required pre-requisites and no waiver **must drop the class**. Otherwise: student will be dropped and a pre-requisite drop is recorded on the student's record.
- Taking the course more than twice:

  you must request, in writing, permission from the Chair of the Department and the Dean of the Faculty to be allowed to stay registered in the class (University Rule). The letter should be given to Jane Guy, Undergraduate Advisor. Otherwise: student will be dropped from class.
- Always use and check your UVic e-mail account and use CSC 225 as part of the subject line.
- Do not send messages from hotmail accounts (hotmail messages are filtered and discarded).
- Register for labs!

Books

- **Required Textbook**
  R. Sedgewick and K. Wayne
  Algorithms, Fourth Edition
  Addison-Wesley, Toronto, 2011
  ISBN: 0-321-57351-X
- http://algs4.cs.princeton.edu/home/

- **Optional Textbook**
  T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein. *Introduction to Algorithms.*
  MIT Press (2001), 2^{nd} edition.



Lectures and Labs

- **Attendance of Lectures**
  - Essential for doing well on assignments and exams
- **Labs**
  - Extra details and hints on assignments

Evaluation

Assignments & Quizzes	40%
Two Midterms	20%
Final	40%

- Marks will be posted on the web by student id
  - If you do not want your marks to be posted in this manner, notify the instructor by e-mail (venkat@cs.uvic.ca) before Jan 14, 2013
- Two midterm exams will be in-class, one hour, closed books, closed notes, no calculators, no gadgets
  January 30, 2013 and March 6, 2013
- The final exam will be three hours, closed books, closed notes, no calculators, no gadgets scheduled by the registrar

Assignment Schedule

A1	January 25, 2013 (due date)
A2	February 8, 2013
A3	March 1, 2013
A4	March 15, 2013
A5	April 2, 2013

Reading Assignment

- **Chapter 1 – 1.1, 1.2 in Sedgewick and Wayne**
- Algorithm wiki
  - http://en.wikipedia.org/wiki/Algorithm
  - History: Development of the notion of "algorithm"
- Data structures wiki
  - http://en.wikipedia.org/wiki/Data_structure
  - http://en.wikipedia.org/wiki/List_of_data_structures
  - http://en.wikibooks.org/wiki/Data_Structures

Assignments

- **5 assignments during the course**
- **Late submissions are <u>not</u> accepted**
  - if valid excuse (e.g., doctor's statement), raise weight of other assignments to compensate
- **Programming: work in the labs or at home**
  - use your favorite Java environment
- **Cheating: zero-tolerance policy**
  - first time fail assignment, second time fail course

Prerequisites

- **CSC 115**
  - ➢Basic Java knowledge and programming skills
  - ➢Object-oriented programming
  - ➢Basics in fundamental algorithms and data structures as discussed in CSC 115

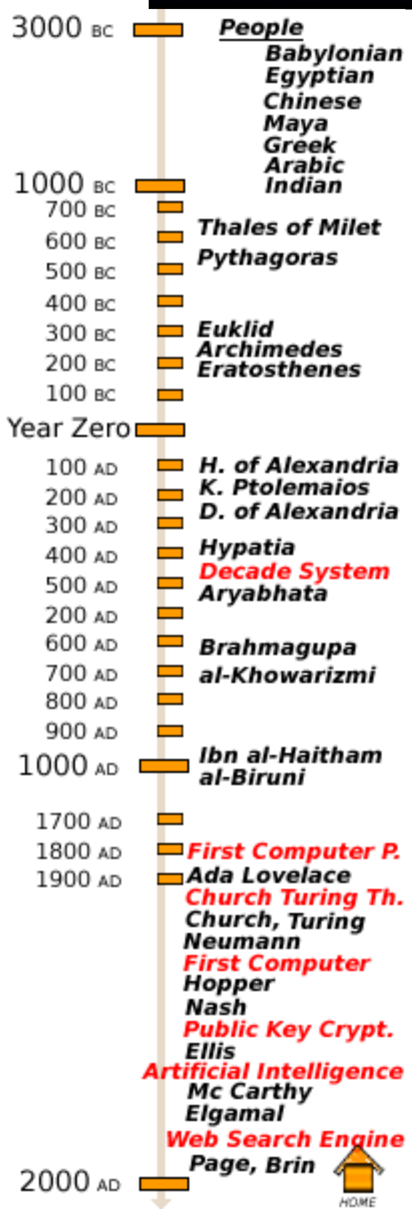
Lecture Notes

- **Acknowledgments**
  - Most of the slides for this course were prepared by Dr. Ulrike Stege. Thank you!!
- Consider posted lecture notes as *additional* information
- **Note**
  - Not all materials required for the midterm and final exams are on the lecture notes

Questions?

- Regarding questions on lectures, assignments, algorithms, data structures, programming, Java, etc. consult in the following order:
  - Study group
  - Course web page
  - Lab instructor
  - Instructor

History of Algorithms

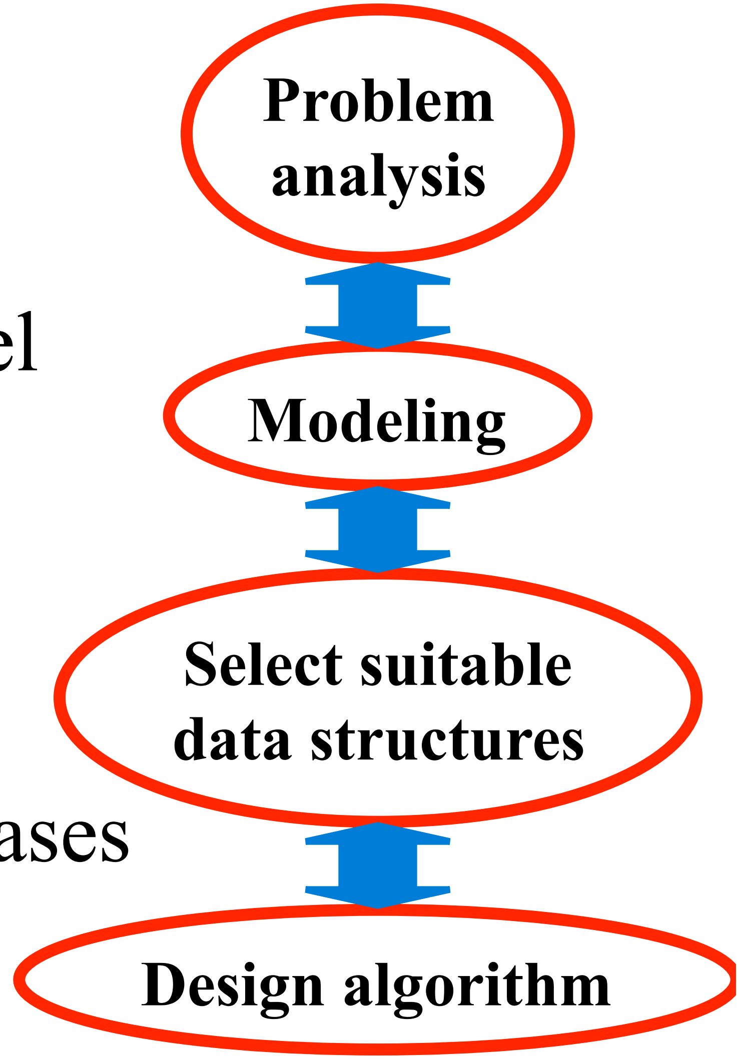


- The word algorithm can be traced back to the 9th century to the Persian scientist, astronomer and mathematician Abdullah Muhammad bin Musa al-Khwarizmi, often cited as "The father of Algebra"
- In the 12th century one of his books was translated into Latin, where his name was rendered as "Algorithmi"
- Algorithms are everywhere. They have been developed to ease our daily life from calculating algorithms, to artificial intelligence and molecular biology. The searching and sorting algorithms embodied in Google are a good example of our daily use of algorithms.
- In the age of information, people are inundated with data. With the aid of powerful algorithms and data structures, we can make sense of volumes of data that come in many forms: text, numbers, images, video, audio.
- **History of Algorithms** http://cs-exhibitions.uni-klu.ac.at/index.php?id=193

Algorithmics

- Problem analysis
- Design an appropriate model
- Select data structures
- Select algorithms

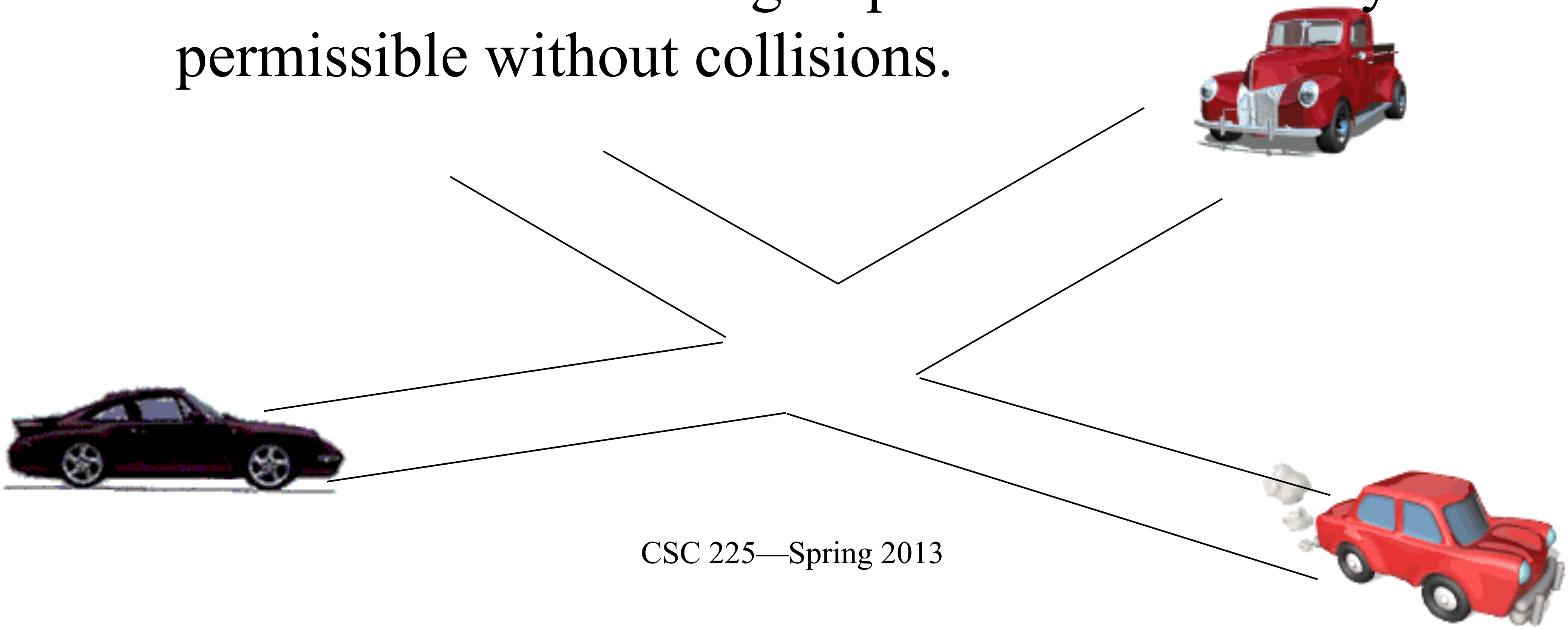
- Iterate over these design phases



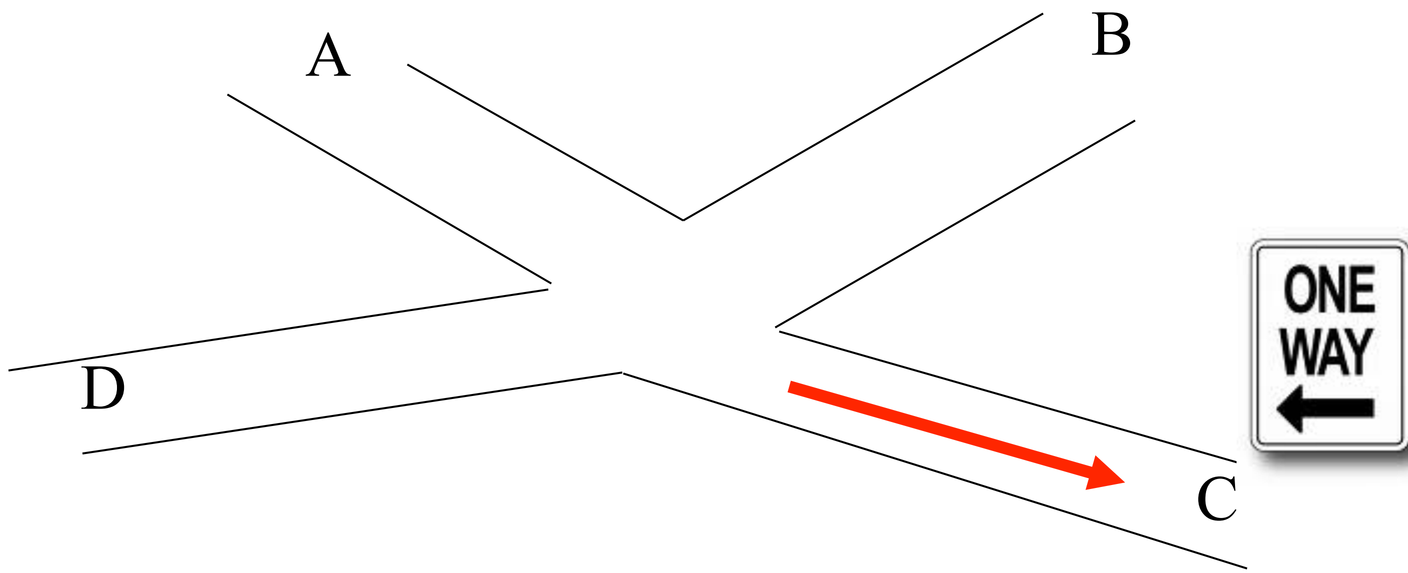
Design of traffic light phases for an intersection of roads

Given: A set of permitted turns of intersections

Goal: Partition the set into as few groups as possible such that all turns in a group are simultaneously permissible without collisions.

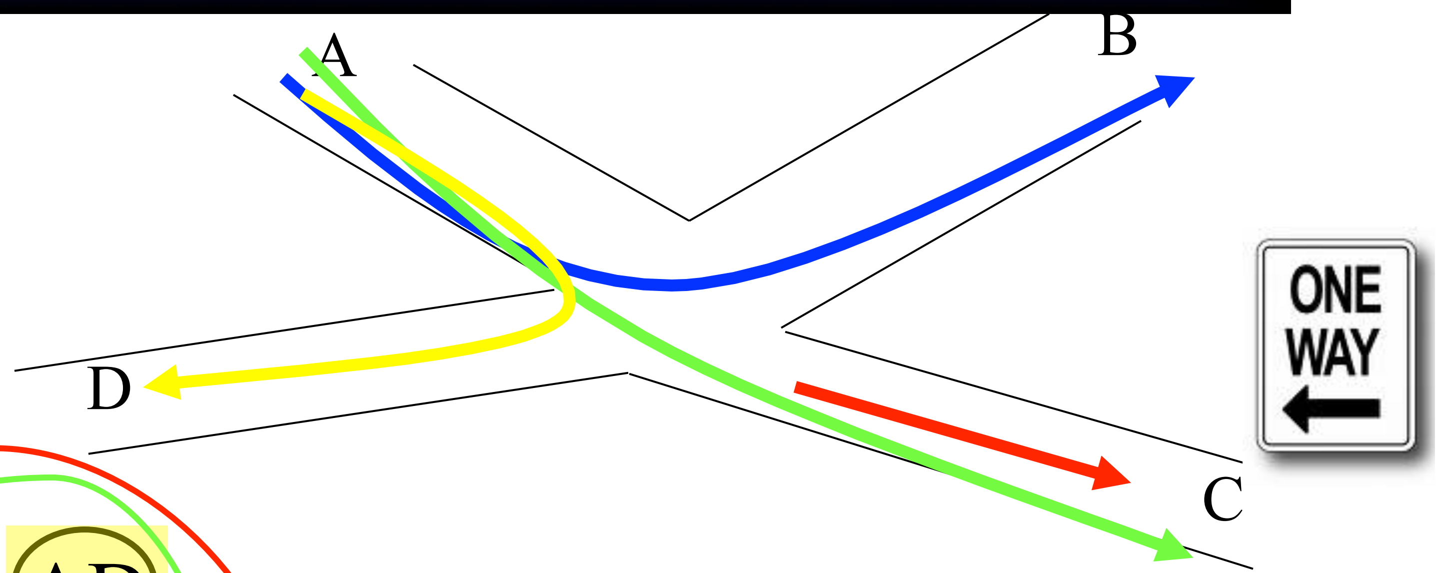


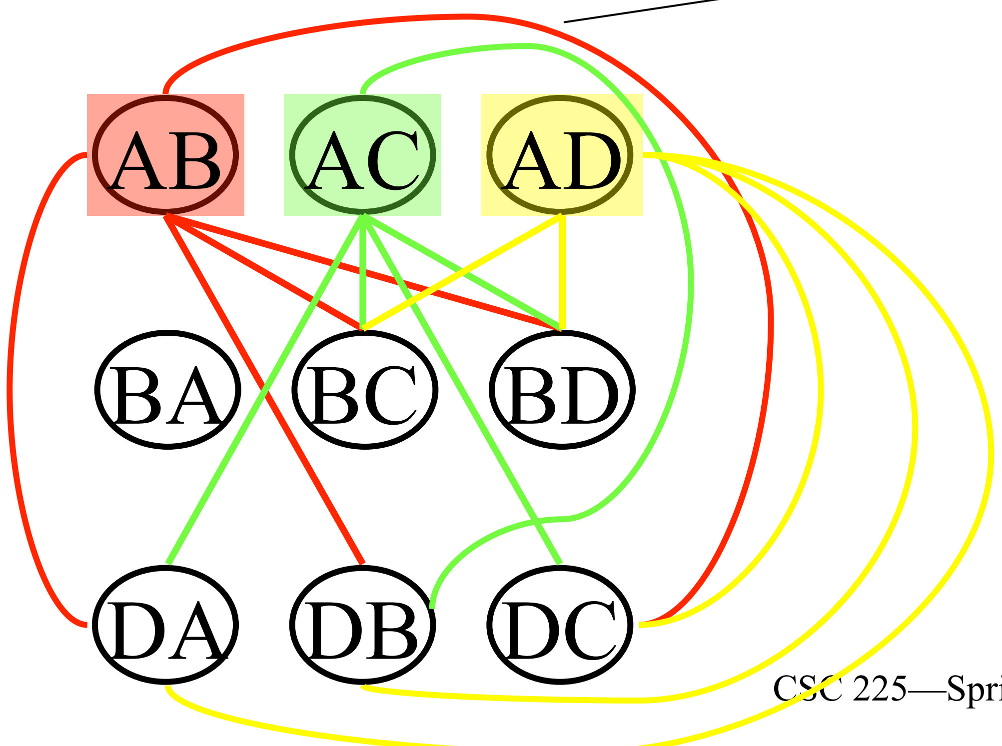
Possible Turns



- AB, AC, AD
- BA, BC, BD
- DA, DB, DC

The Model: A Conflict Graph





Possible turns ≡ *vertices*
Incompatible turns ≡ *edges*

The Model: A Conflict Graph

- Complete the conflict graph for the other two rows of the graph (done in class)

- Find a traffic phase assignment such that no two turns that have a possible collision have a green light at the same time
- Minimize the number of phases
- Implement domain-specific optimization

Problem Solving

- **Model:** Conflict Graph
- **Data Structure:** graph
- **Algorithm:** graph colouring

Graph Colouring Problem

Input: A *graph G* = (*V*, *E*) with *vertices* and *edges E*.

- Possible turns ≡ *vertices*
- Incompatible turns ≡ *edges*

Output: A minimum set of colours and an assignment from the colours to the vertices such that no two vertices that have an edge in common have the same colour.

- Colour ≡ *traffic light phase*

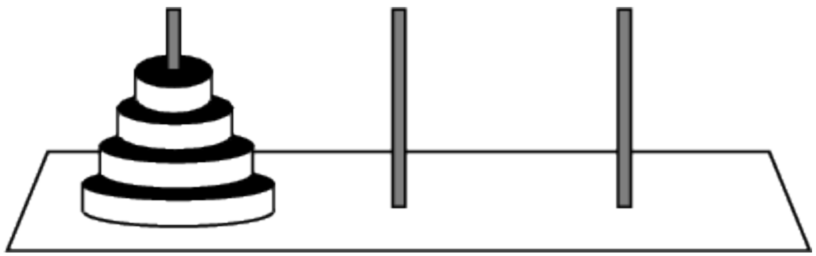
Algorithm Design Techniques

- **Algorithm Design Techniques**
  - Greedy algorithms
  - Divide and conquer
  - Backtracking
  - Dynamic programming

Design of traffic light phases for an intersection of roads

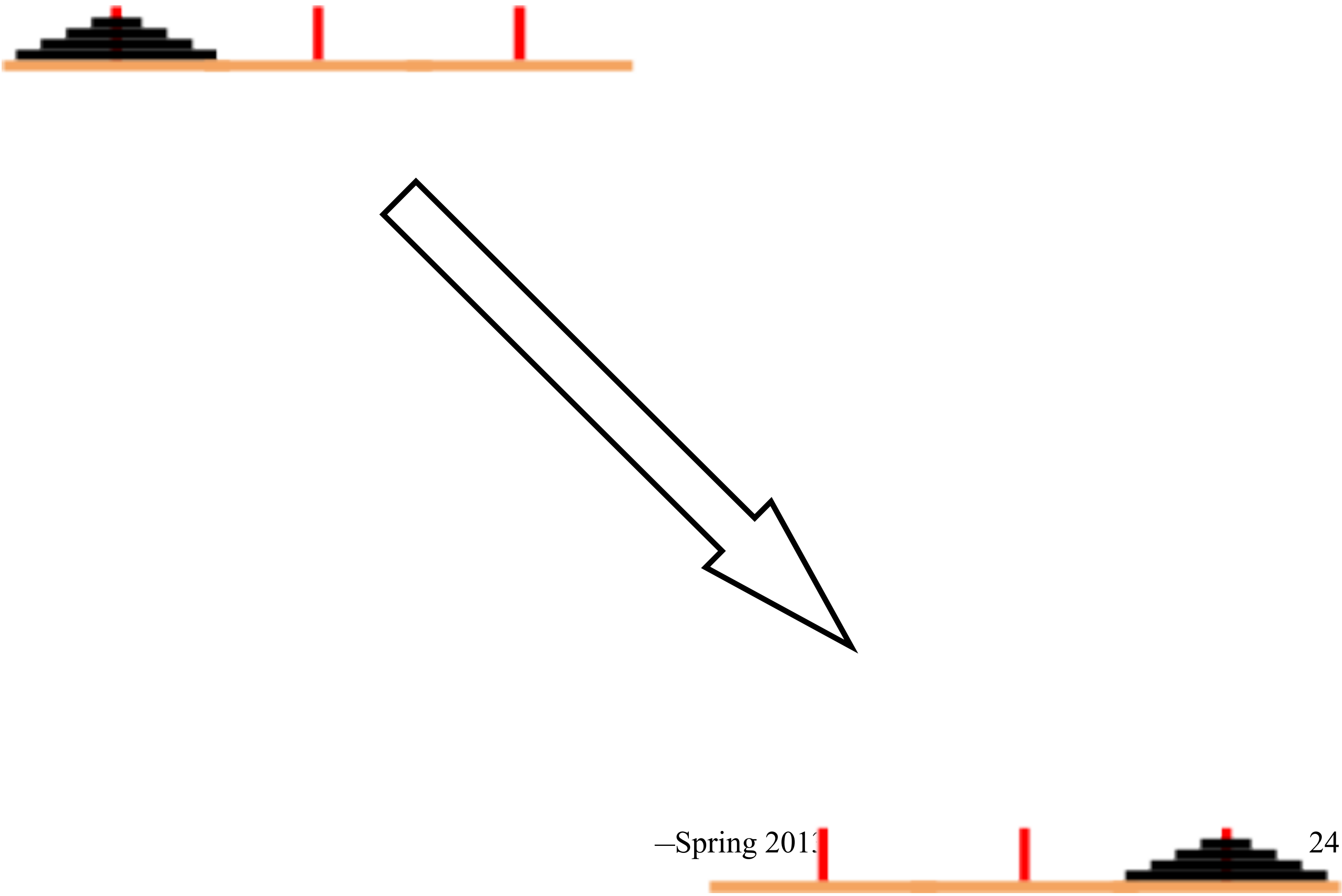
- Find optimal solution this problem
  - Hard
  - Graph colouring is NP-complete
- Find "a" solution using greedy approach
  - Heuristic: choose an arbitrary node as the first node to be coloured
  - Colour all the nodes that are not connected to this node with the same colour
  - Iterate
- Enumerate all solutions using greedy approach and select best or most appropriate solution
  - Works for small graphs

Which Type of Algorithm Solves the Towers of Hanoi Problem?

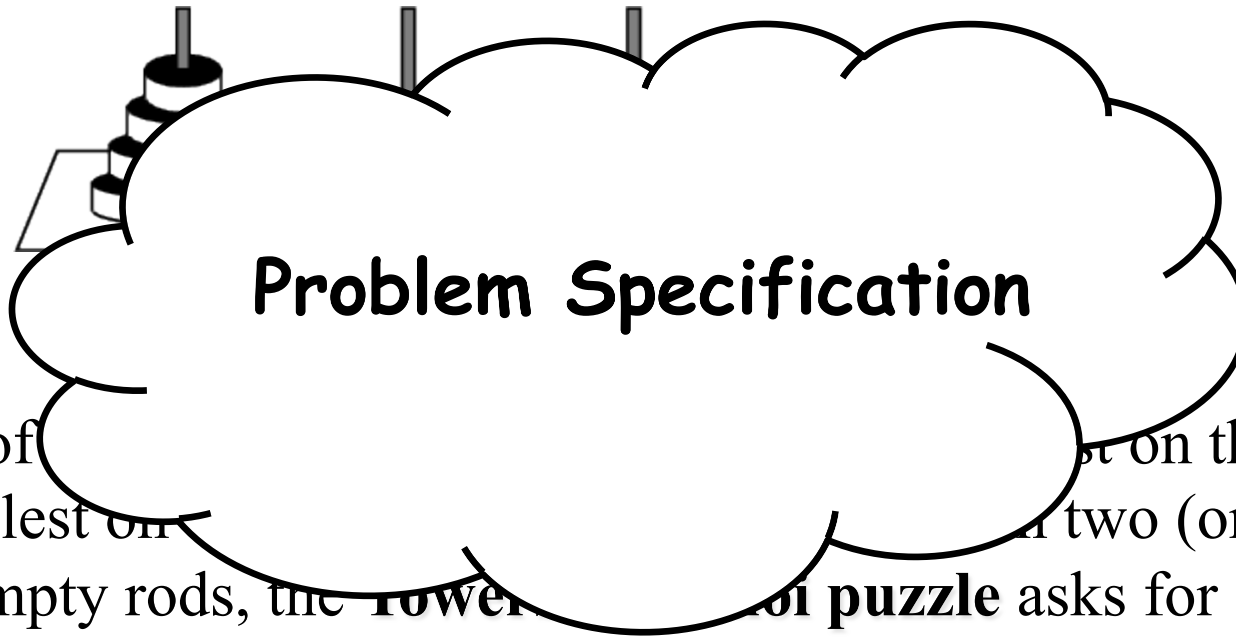


Recursion and Backtracking

Towers of Hanoi



Towers of Hanoi

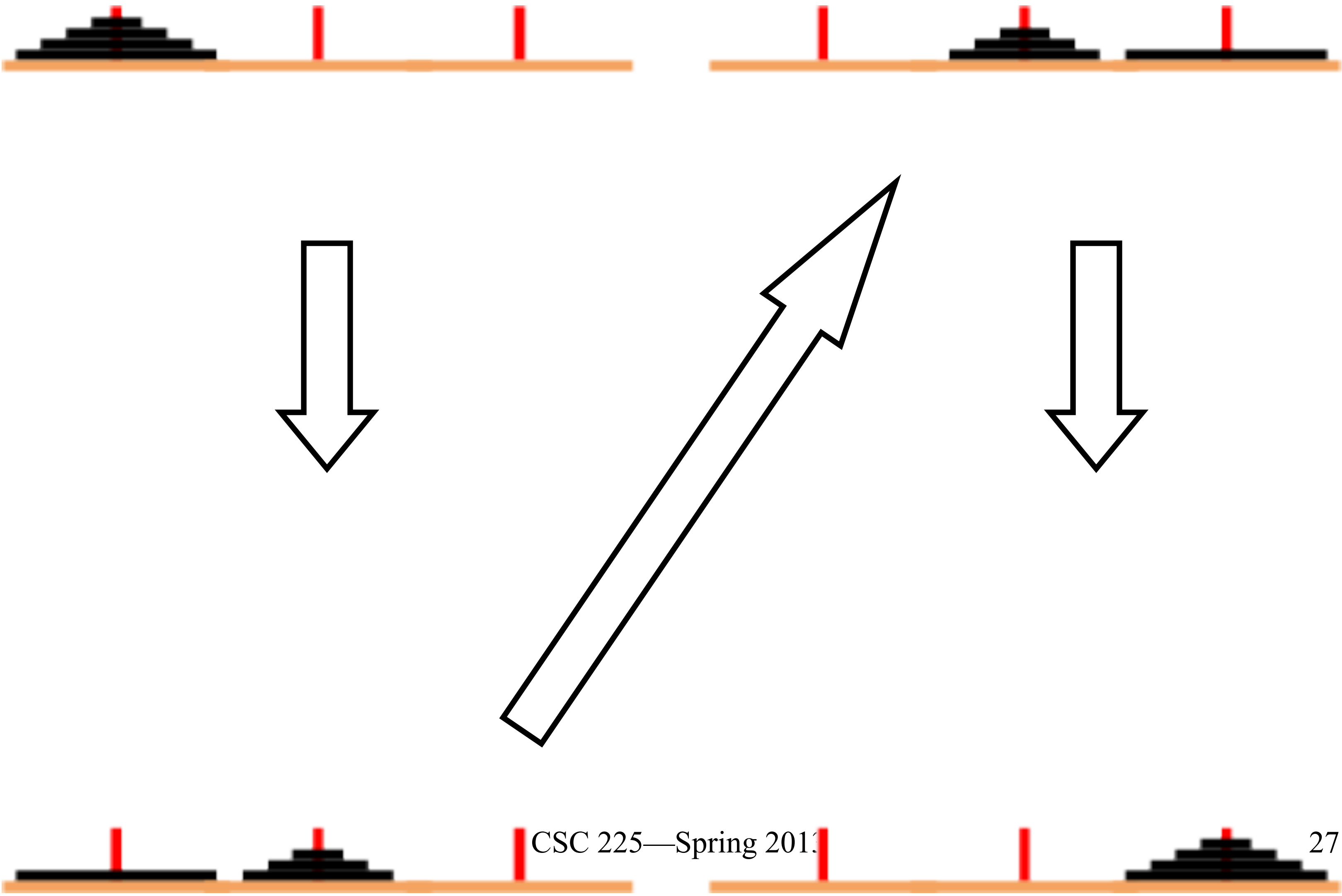


Given a stack of … on the bottom to smallest on … two (or, in general *r*) empty rods, the Tower… **…oi puzzle** asks for the *minimum number* of disk-moves required to move the stack from one rod to another. **…e moves are not allowed only if they place larger disks on top of smaller disks.**

$n = 4$

Solution:
high level description

Solving for n = 3 helps!



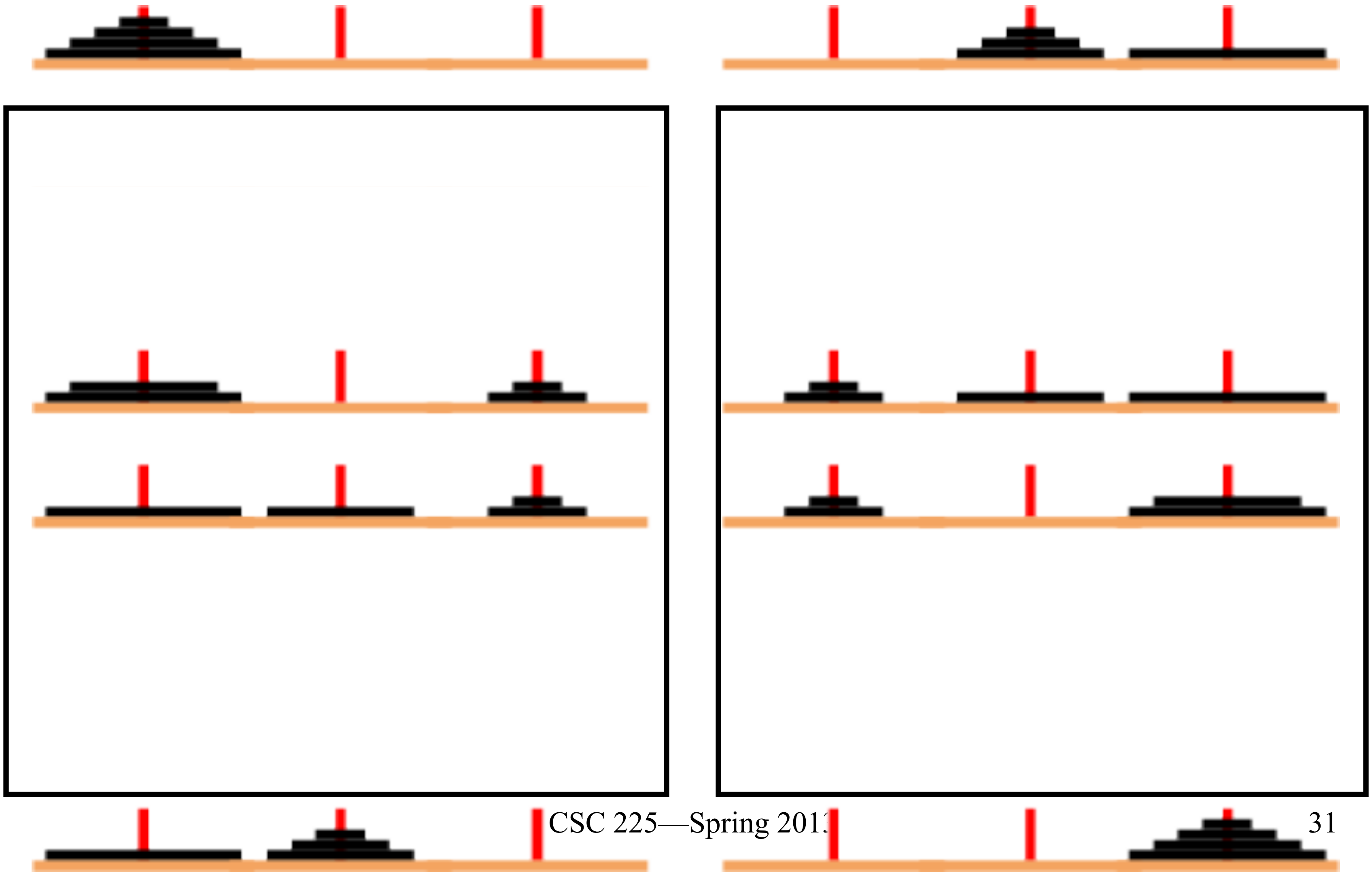
$n = 3$

- Solving for $n = 2$ helps!

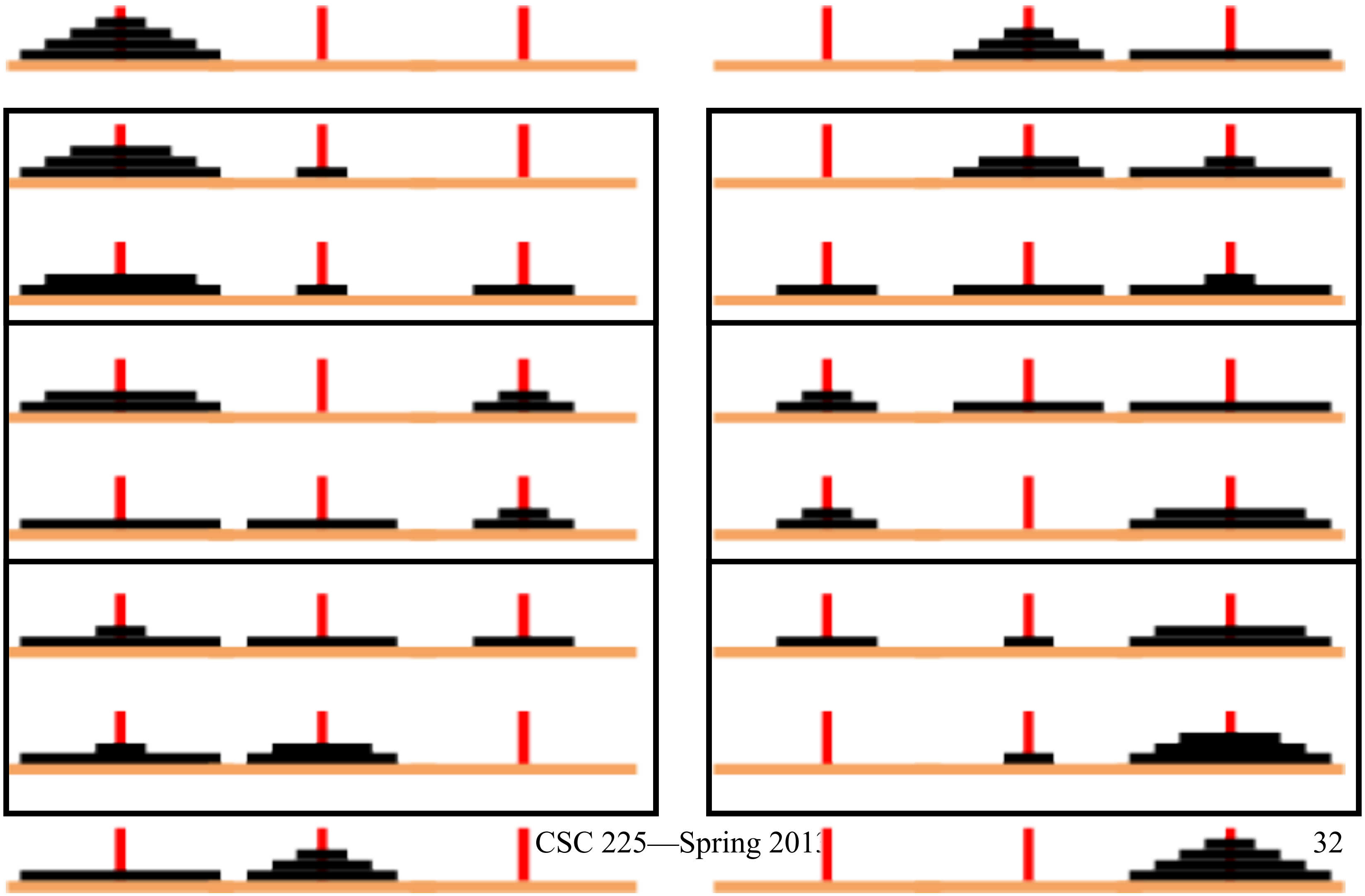
$n = 2$

$n = 4$

Solution for $n = 4$
(15 moves required)



Solution for $n = 4$
(15 moves required)



Still missing

Solution as a high-level description

Still missing

Proof of Correctness

-does the suggested solution solve the Tower of Hanoi puzzle for **any** given instance?

-Does the suggested solution use as few steps as possible?

Quiz

- How many moves are required for $n = 5$?
- How many moves are required for n discs?
- Does the number of moves change if 4 rods (instead of 3 rods) are allowed?
  - $n = 1$
  - $n = 2$
  - $n = 3$
  - $n = 4$
  - $n = 5$

What is an Algorithm?

- An ***algorithm*** is a sequence of unambiguous instructions for solving a problem for obtaining the desired output for any legitimate input in a finite amount of time.

- An ***algorithm*** is a finite procedure, written in a fixed symbolic vocabulary, governed by precise instructions, moving in discrete steps, 1, 2, 3, …, whose execution requires no insight, cleverness, intuition, intelligence, or perspicuity, and that sooner or later comes to an end.

- The nonambiguity requirement is critical
- The range of inputs has to be carefully specified
- An algorithm can be implemented in several different ways
- Algorithms for the same problem can be based on very different ideas and can solve the problem with very different speeds