

Welcome to
CS E 589 -- Applied Algorithms

Anna R. Karlin (karlin@cs)
Office Hours: Wed 9:20 - 10:00 pm
and by appointment

TA: Ashish Sabharwal (ashish@cs)
Office Hours: W 5:30-6:20pm

Administrivia

Books, etc.

- The Algorithm Design Manual
 - Steven Skiena

Grading:

- Weekly problem sets 50%
- Final 50%

Course Goals

An appreciation for applications of algorithmic techniques in the real world.

A larger toolbox.

A better sense of how to model problems you encounter as well-known algorithmic problems.

A deeper understanding of the issues and tradeoffs involved in algorithm design.

FUN!!!!!!!!!!!!!! BEAUTY!!!!!!!!!!!!

Other

Interested in suggestions, feedback, constructive criticism. (E.g., topics/depth vs. breadth)

Ask lots of questions! Participate! Offer opinions!

Plan For Today

Intro -- Skiena, chapter 1

Cool lectures from CMU

- intro to "great theoretical ideas in computer science"
- the stable marriage problem

Warning: very different from the typical lecture in this course

Algorithm Design Goals

correctness
efficiency

According to the *Oxford English Dictionary*

algorist "one skillful in reckonings or figuring"

Keeping Score:
The RAM Model of Computation

Each simple operation takes 1 time step.
Loops and subroutines are not simple operations.

Each memory access takes one time step, and there is no shortage of memory.

For a given problem instance:

Running time of an algorithm = # RAM steps (ops)

Space used by an algorithm = # RAM memory cells

Types of complexity

Function from size of instance to real numbers

Worst-case complexity

Best-case complexity

Average-case complexity

Big Oh Notation

Goals :

- ignore details that do not impact comparisons of algorithms
- ignore constant factors

$f(n) = O(g(n)) \Leftrightarrow cg(n)$ is upper bound on $f(n)$

\Leftrightarrow There exist c, N s.t. for $n \geq N$, $f(n) \leq cg(n)$

$f(n) = \Omega(g(n)) \Leftrightarrow cg(n)$ is lower bound on $f(n)$

\Leftrightarrow There exist c, N s.t. for $n \geq N$, $f(n) \geq cg(n)$

$f(n) = \Theta(g(n)) \Leftrightarrow f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$

\Leftrightarrow There exist c_1, c_2, N s.t. for $n \geq N$,
 $c_1 g(n) \leq f(n) \leq c_2 g(n)$

Growth Rates, etc.

Even by ignoring constant factors, can get an excellent idea of whether a given algorithm will be able to run in a reasonable amount of time on a problem of a given size.

The "big Oh" notation and worst-case analysis are tools that greatly simplify our ability to compare the efficiency of algorithms.

Next Bunch of Slides Taken From:

Great Theoretical Ideas In Computer Science

CS 15-251

?

Professors
Steven Rudich and Bruce Maggs
Carnegie Mellon University

Lecture 1

CS 15-251

The future belongs to the computer scientist who has

- Content: An up to date grasp of fundamental problems and solutions
- Method: Principles and techniques to solve the vast array of unfamiliar problems that arise in a rapidly changing field

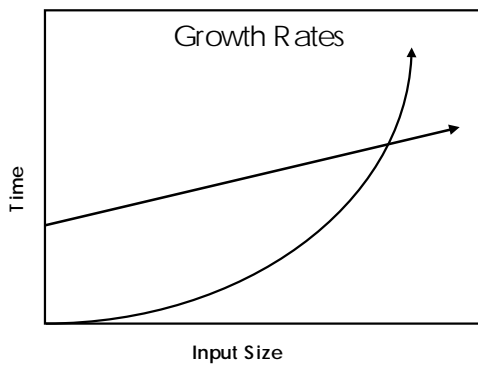


Course Content

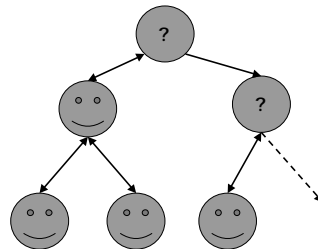
- A survey of fundamental theoretical ideas in Computer Science as they occur in a variety of important applications
- An introduction to discrete mathematics
- Effective problem solving, learning, and communication techniques

A survey of fundamental theoretical ideas in Computer Science as they occur in a variety of important applications

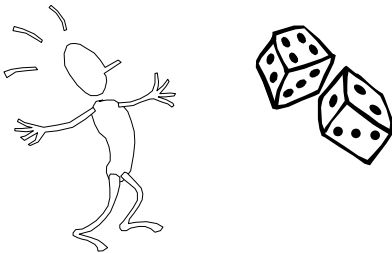
For example . . .



Recursion

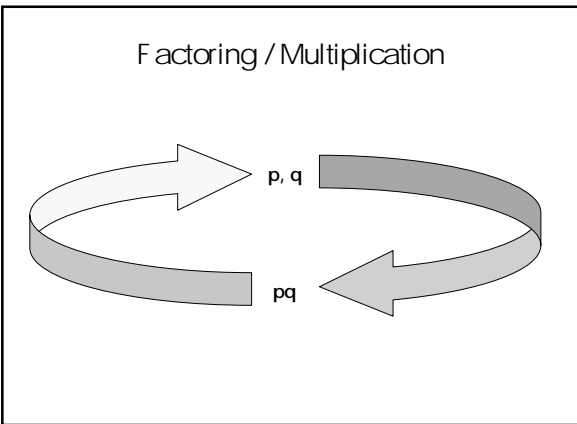
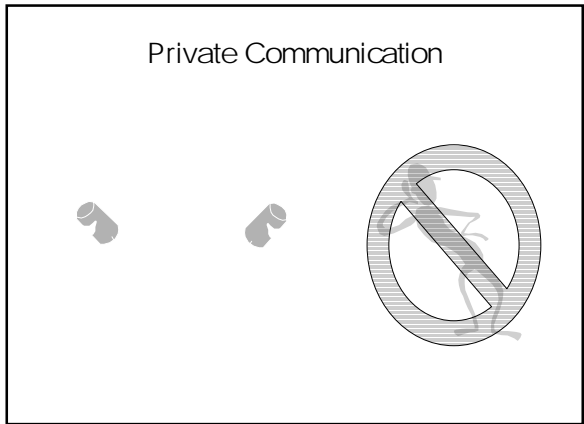
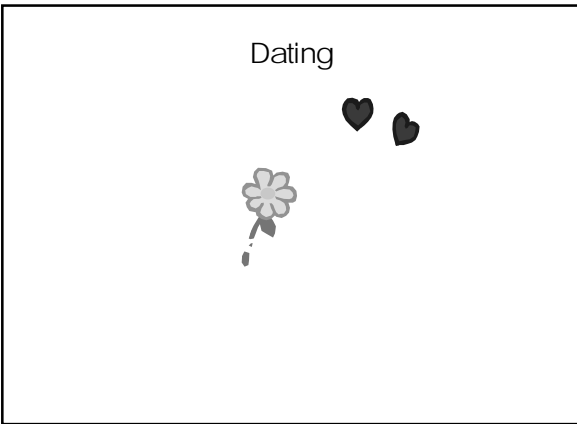
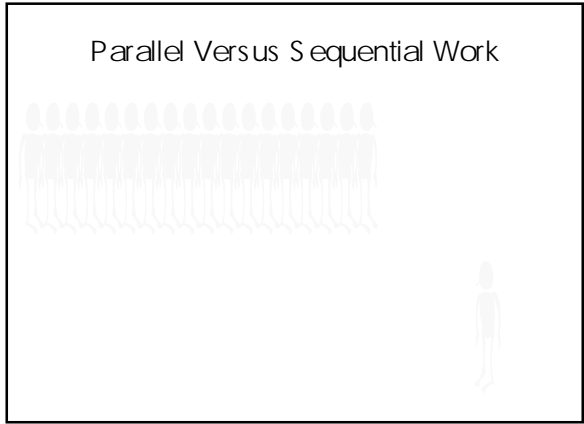
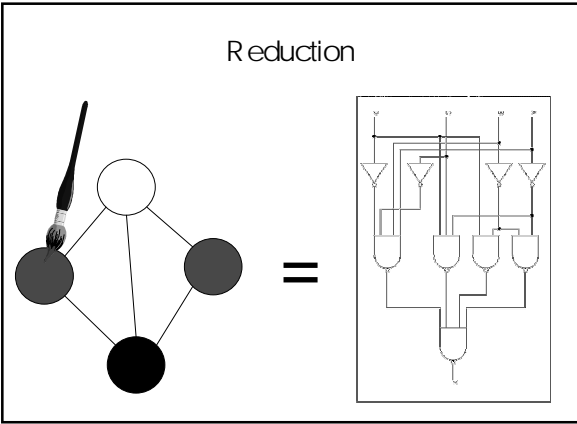


Randomization



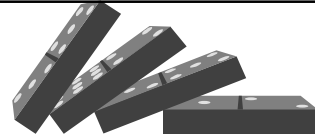
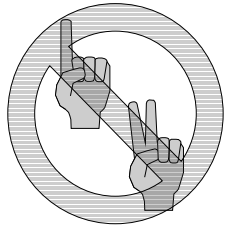
Hashing





An introduction to discrete mathematics

Count without counting:
Estimate, Calculate, Analyze



Induction has many guises.
Master their interrelationship.

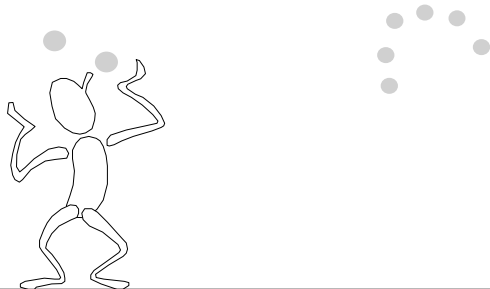
- Formal Arguments
- Loop Invariants
- Recursion
- Algorithm Design
- Recurrences

Map Coloring

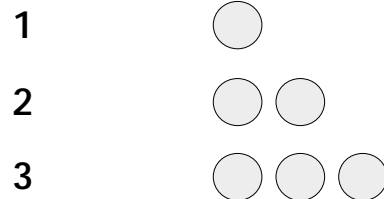


Effective problem solving, learning,
and communication techniques

Exemplification:
Try out a problem or
solution on small examples.



Representation:
Understand the relationship between
different representations of the same
information or idea



Modularity:
Decompose a complex problem into simpler subproblems

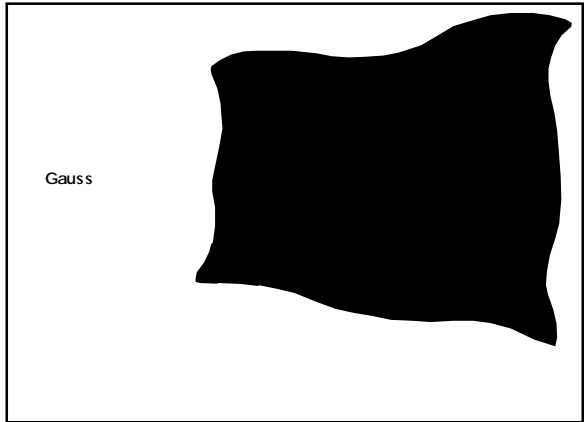
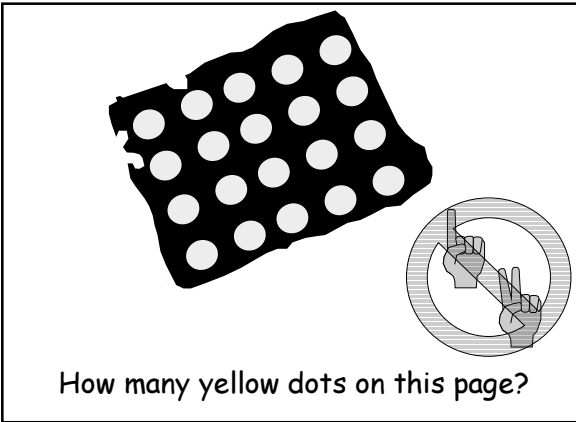
Abstraction:
Abstract away the inessential features of a problem

Refinement:
The best solution comes from a process of repeatedly refining and inventing alternative solutions

Build your toolbox of abstract structures and concepts. Know the capacities and limits of each tool.

Similarity:
A significant form of intellectual progress is to be able to classify and manipulate distinct objects with regard to a sense in which they are similar.

$13 = 21 \pmod{2}$



$$1 + 2 + 3 + \dots + n-1 + n = S$$

$$1 + 2 + 3 + \dots + n-1 + n = S$$

$$n + n-1 + n-2 + \dots + 2 + 1 = S$$

$$1 + 2 + 3 + \dots + n-1 + n = S$$

$$n + n-1 + n-2 + \dots + 2 + 1 = S$$


$$(n+1) + (n+1) + (n+1) + \dots + (n+1) + (n+1) = 2S$$

$$n(n+1) = 2S$$

$$1 + 2 + 3 + \dots + n-1 + n = S$$

$$n + n-1 + n-2 + \dots + 2 + 1 = S$$

$$(n+1) + (n+1) + (n+1) + \dots + (n+1) + (n+1) = 2S$$

$$n(n+1) = 2S$$


$n + n-1 + n-2 + \dots + 2 + 1 = S$
 $(n+1) + (n+1) + (n+1) + \dots + (n+1) + (n+1) = 2S$
 $n(n+1) = 2S$

Algebraic argument

Let's restate this argument using a geometric representation

$1 + 2 + 3 + \dots + n-1 + n = S$ = number of white dots.
 $n + n-1 + n-2 + \dots + 2 + 1 = S$
 $(n+1) + (n+1) + (n+1) + \dots + (n+1) + (n+1) = 2S$
 $n(n+1) = 2S$

= number of white dots
 $n + n-1 + n-2 + \dots + 2 + 1 = S$ = number of yellow dots
 $(n+1) + (n+1) + (n+1) + \dots + (n+1) + (n+1) = 2S$
 $n(n+1) = 2S$

= number of white dots
 $n + n-1 + n-2 + \dots + 2 + 1 = S$ = number of yellow dots
 $(n+1) + (n+1) + (n+1) + \dots + (n+1) + (n+1) = 2S$
 $n(n+1) = 2S$

There are $n(n+1)$ dots in the grid

= number of white dots
 $n + n-1 + n-2 + \dots + 2 + 1 = S$ = number of yellow dots
 $(n+1) + (n+1) + (n+1) + \dots + (n+1) + (n+1) = 2S$
 $n(n+1) = 2S$

Gauss

A first glance, these problems appeared unrelated, but a representational change revealed a correspondence. An extension of the similarity principle is that when we come to understand that seemingly unrelated things *are* related, we are making intellectual progress.



When presented with multiple solutions many students remember only the one that they find easiest to understand. The expert student takes the opportunity to think through the similarities and differences between the approaches. Such meditations lay the foundations for effective learning and problem solving.

Martial Arts 101

- The novice makes a **huge** motion
- The black belt makes a small motion
- The master makes a tiny motion

Scanning the brains of master problem solvers



The better the problem solver, the less brain activity is evident. The real masters show almost no brain activity!



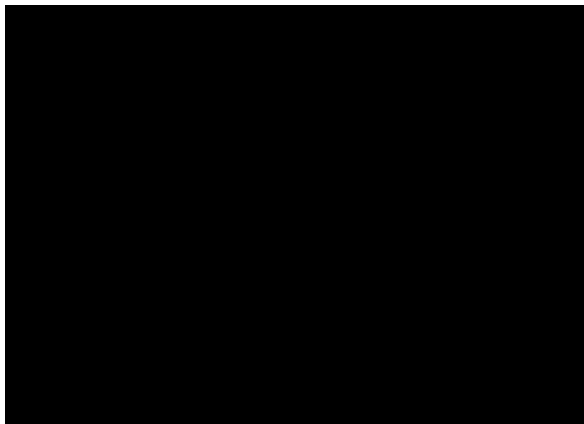
Aside to university students

It is natural to think that you are really getting your money's worth when the professor says very complicated things, but the real value is when the professor makes complex things simple.



Aside to university students

A good martial arts student will attentively repeat each fundamental technique many times. In contrast, many college students tune out when a concept (e.g., depth first search) is taught more than once. The better students listen carefully in order to refine and develop their understanding.



The Master Programmer

The master seeks an algorithm that will use as small an amount of computer resources (e.g., time, space, communication) as possible. Most "expert" programmers (black belts) will miss the best program because they did not have the patience to refine their solutions further.

A case study.

Anagram Programming Task.

You are given a 70,000 word dictionary. Write an anagram utility that given a word as input returns all anagrams of that word appearing in the dictionary.

Examples

Input: CAT

Output: ACT, CAT, TAC

Input: SUBESSENTIAL

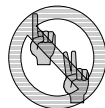
Output: SUITABLENESS

Impatient Hacker (Novice Level Solution)

Loop through all possible ways of rearranging the input word

- Use binary search to look up resulting word in dictionary.
- If found, output it

Performance Analysis: Counting Without Executing



"Expert" Hacker (Black Belt Level)

Subroutine ANAGRAM(X,Y) returns TRUE exactly when X and Y are anagrams. It works by sorting the letters in X and Y

Input X

- Loop through all dictionary words Y
- If ANAGRAM(X,Y) output Y

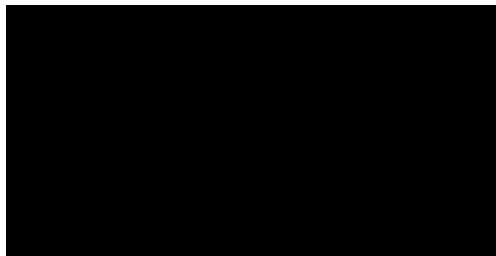
Comparing an input word with each of 70,000 dictionary entries takes about 15 seconds.



The master keeps trying to refine the solution.

The master's program runs in less than 1/1000 seconds.

Master Solution



Suppose the dictionary was the list below.

ASP
DOG
LURE
GOD
NICE
RULE
SPA

After each word, write its "signature"
(sort its letters)

ASP	APS
DOG	DGO
LURE	ELRU
GOD	DGO
NICE	CEIN
RULE	ELRU
SPA	APS

Sort by the signatures

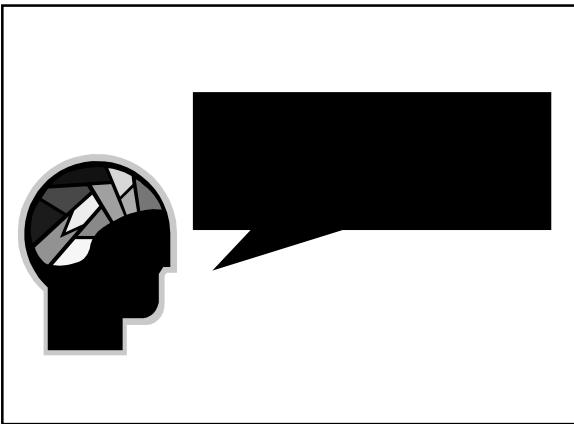
ASP	APS
SPA	APS
NICE	CEIN
DOG	DGO
GOD	DGO
LURE	ELRU
RULE	ELRU

Master Program

Input word W
 $X :=$ signature of W
Use binary search to find the anagram class of W and output it.

About $\log_2(70,000) \times 25$ microseconds
 $\approx .0004$ seconds

Of course, it takes about 30 seconds to create the dictionary, but it is perfectly fair to think of this as programming time. The building of the dictionary is a one-time cost that is part of writing the program.



Learning Advice

Whenever you see something you wish you had thought of, try and formulate the minimal and most general lesson that will insure that you will not miss the same thing the next time. Name the lesson to make it easy to remember.

NAME: Preprocessing

It is sometimes possible to pay a reasonable, one-time preprocessing cost to reorganize your data in such a way as to use it more efficiently later. The extra time required to preprocess can be thought of as additional programming effort.

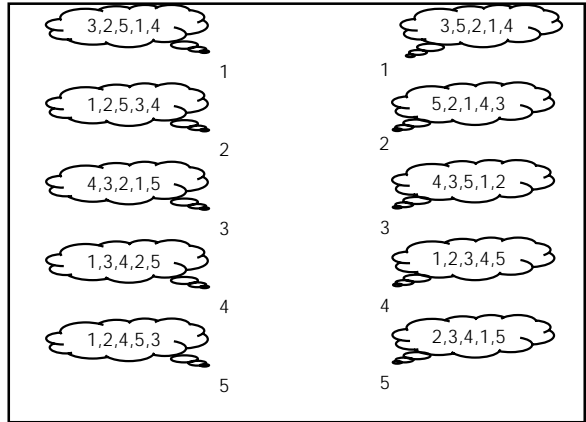
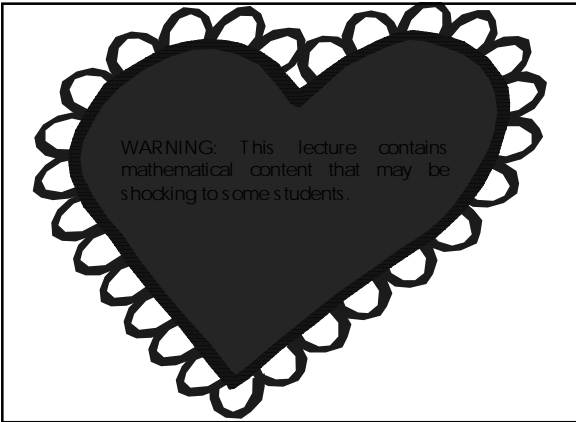
Great Theoretical Ideas In Computer Science

**The Mathematics Of 1950's Dating:
Who wins the battle of the sexes?**



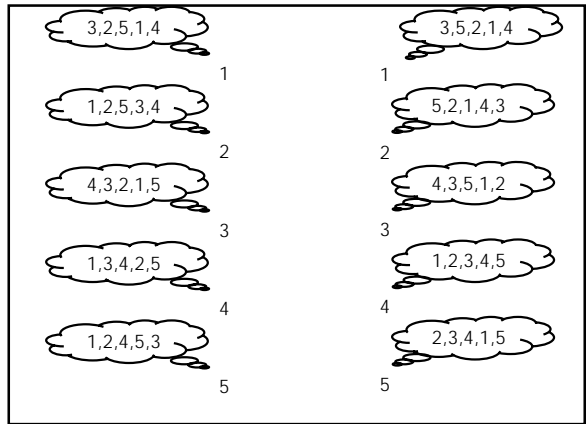
Lecture 2

CS 15-251



Dating Scenario

- There are n boys and n girls
- Each girl has her own ranked preference list of all the boys
- Each boy has his own ranked preference list of the girls
- The lists have no ties



There is more than one notion of what constitutes a "good" pairing.

- Maximizing total satisfaction
 - Hong Kong and to an extent the United States
- Maximizing the minimum satisfaction
 - Western Europe
- Minimizing the maximum difference in mate ranks
 - Sweden
- Maximizing the number of people who get their first choice
 - Barbie and Ken Land

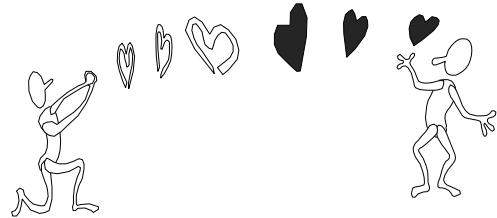
We will ignore the issue of what is "equitable"!

Rogue Couples

Suppose we pair off all the boys and girls. Now suppose that some boy and some girl prefer each other to the people to whom they are paired. They will be called a rogue couple.



Why be with them when we can be with each other?



Stable Pairings

A pairing of boys and girls is called stable if it contains no rogue couples.

Stability is primary.

Any list of criteria for a good pairing must include **stability**. (A pairing is doomed if it contains a rogue couple.)

Any reasonable list of criteria must contain the stability criterion.

The study of stability will be the subject of the entire lecture.


We will:

- Analyze various mathematical properties of an algorithm that looks a lot like 1950's dating
- Discover the **naked mathematical truth** about which sex has the romantic edge
- Learn how the world's largest, most successful dating service operates

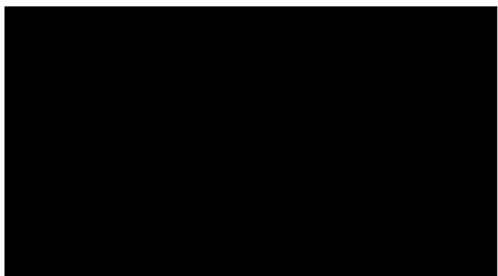
Given a set of preference lists, how do we find a stable pairing?

Given a set of preference lists, how do we find a stable pairing?

Wait! There is a more primary question!




Existence Question

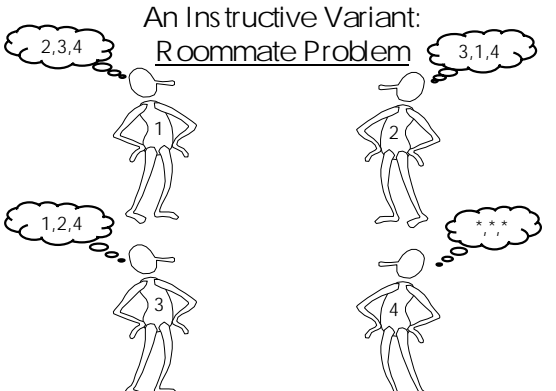


3,2,5,1,4	1	3,5,2,1,4	1
1,2,5,3,4	2	5,2,1,4,3	2
4,3,2,1,5	3	4,3,5,1,2	3
1,3,4,2,5	4	1,2,3,4,5	4
1,2,4,5,3	5	2,3,4,1,5	5

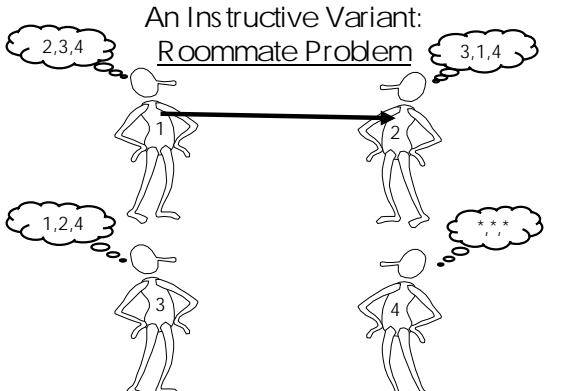
Can you argue that the couples will not continue breaking up and reforming forever?

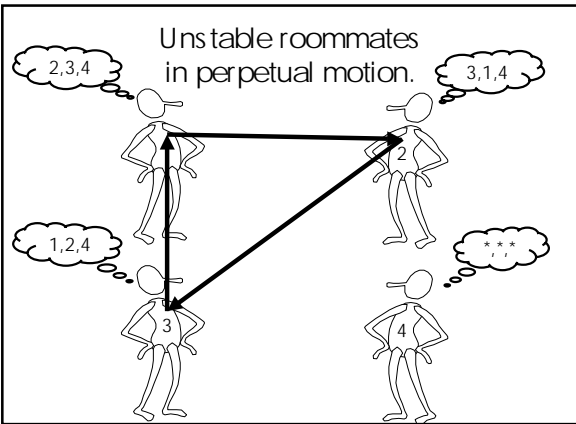
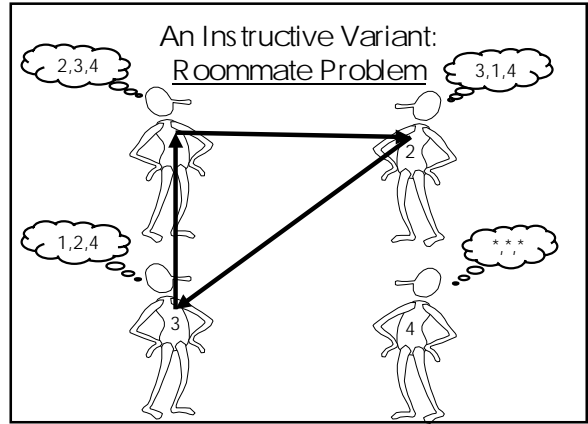
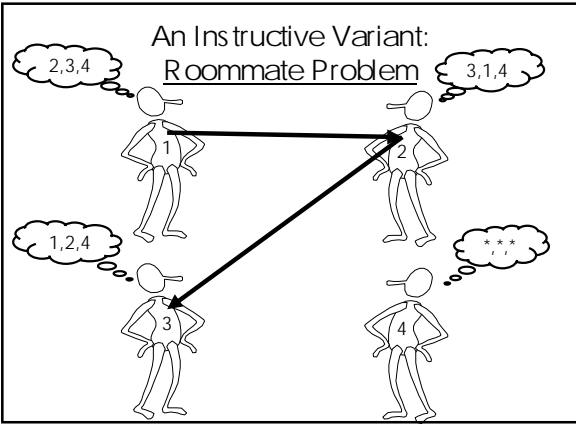


An Instructive Variant:
Roommate Problem



An Instructive Variant:
Roommate Problem





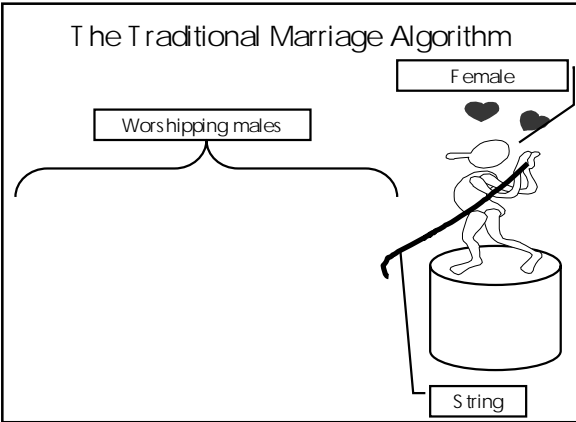
Insight

Any proof that couples do not break up and reform forever must contain a step that fails in the case of the roommate problem

Insight

If you have a proof idea that works equally well in the marriage problem and the roommate problem, then your idea is not adequate to show the couples eventually stop.

The Traditional Marriage Algorithm



Traditional Marriage Algorithm

For each day that some boy gets a "No" do:

- **Morning**
 - Each girl stands on her balcony
 - Each boy proposes under the balcony of the best girl whom he has not yet crossed off
- **Afternoon (for those girls with at least one suitor)**
 - To today's best suitor: "Maybe, come back tomorrow"
 - To any others: "No, I will never marry you"
- **Evening**
 - Any rejected boy crosses the girl off his list

Each girl marries the boy to whom she just said "maybe"

Does the Traditional Marriage Algorithm always produce a stable pairing?

Does the Traditional Marriage Algorithm always produce a stable pairing?

Wait! There is a more primary question!

Does TMA always terminate?

- It might encounter a situation where algorithm does not specify what to do next (core dump error)
- It might keep on going for an infinite number of days

Traditional Marriage Algorithm

For each day that some boy gets a "No" do:

- **Morning**
 - Each girl stands on her balcony
 - Each boy proposes under the balcony of the best girl whom he has not yet crossed off
- **Afternoon (for those girls with at least one suitor)**
 - To today's best suitor: "Maybe, come back tomorrow"
 - To any others: "No, I will never marry you"
- **Evening**
 - Any rejected boy crosses the girl off his list

Each girl marries the boy to whom she just said "maybe"

Lemma: No boy can be rejected by all the girls

Proof by contradiction.

Suppose boy b is rejected by all the girls. At that point:

- Each girl must have a suitor other than b (Once a girl has a suitor she will always have at least one)
- But there are n girls and only n-1 boys besides b



Theorem: The TMA always terminates in at most n^2 days

- A "master list" of all n of the boys lists starts with a total of $n \times n = n^2$ girls on it.
- Each day that at least one boy gets a "No", at least one girl gets crossed off the master list
- Therefore, the number of days is bounded by the original size of the master list

Great! We know that TMA will terminate and produce a pairing.

But is it stable?

MAYBE Lemma: In TMA if on day i a girl says "maybe" to boy b, she is guaranteed to marry a husband that she likes at least as much as b

- She would only let go of him in order to "maybe" someone better
- She would only let go of that guy for someone even better
- She would only let go of that guy for someone even better
- AND SO ON

Informal Induction



MAYBE Lemma: In TMA if on day i a girl says "maybe" to boy b, she is guaranteed to marry a husband that she likes at least as much as b

(*) For all $k \geq 0$, on day $i+k$ the girl will say "maybe" to a boy she likes as much as b.

BASE: $k=0$ (true by assumption)

Assume (*) is true for $k-1$. Thus she has a boy as good as b on day $i+k-1$. The next day she will either keep him or reject him for some better. Thus (*) is true for k.

Formal Induction



Corollary: Each girl will marry her absolute favorite of the boys who visit her during the TMA



Theorem: Let T be the pairing produced by TMA. T is stable.

- Let b and g be any couple in T .
- Suppose b prefers g^* to g . We will argue that g^* prefers her husband to b .
- During TMA, b proposed to g^* before he proposed to g . Hence, at some point g^* rejected b for someone she preferred. By the MAYBE lemma, the person she married was also preferable to b .
- Thus, every boy will be rejected by any girl he prefers to his wife. T is stable.

Opinion Poll



Forget TMA for a moment

How should we define what we mean when we say "the optimal girl for boy b "?

Flawed Attempt:
"The girl at the top of b 's list"

The Optimal Girl

A boy's optimal girl is the highest ranked girl for whom there is some stable pairing in which the boy gets her.

She is the best girl he can conceivably get in a stable world. Presumably, she might be better than the girl he gets in the stable pairing output by TMA.

The Pessimist Girl

A boy's pessimist girl is the lowest ranked girl for whom there is some stable pairing in which the boy gets her.

She is the worst girl he can conceivably get in a stable world.

Dating Heaven and Hell

A pairing is male-optimal if every boy gets his optimal mate. This is the best of all possible stable worlds for every boy simultaneously.

A pairing is male-pessimist if every boy gets his pessimist mate. This is the worst of all possible stable worlds for every boy simultaneously.

Dating Heaven and Hell

A pairing is male-optimal if every boy gets his optimal mate. Thus, the pairing is simultaneously giving each boy his optimal.

Is a male-optimal pairing always stable?

Dating Heaven and Hell

A pairing is female-optimal if every girl gets her optimal mate. This is the best of all possible stable worlds for every girl simultaneously.

A pairing is female-pessimal if every girl gets her pessimal mate. This is the worst of all possible stable worlds for every girl simultaneously.

The Naked Mathematical Truth!



Theorem: TMA produces a male-optimal pairing

- Suppose, for a contradiction, that some boy gets rejected by his optimal girl during TMA. Let t be the earliest time at which this happened.
- In particular, at time t , some boy b got rejected by his optimal girl g because she said "maybe" to a preferred b' . By the definition of t , b' had not yet been rejected by his optimal girl.
- Therefore, b' likes g at least as much as his optimal.

Some boy b got rejected by his optimal girl g because she said "maybe" to a preferred b' . b' likes g at least as much as his optimal girl.

There must exist a stable pairing S in which b and g are married.

- b^* wants g more than his wife in S
 - g is as at least as good as his best and he does not have her in stable pairing S
- g wants b^* more than her husband in S
 - b is her husband in S and she rejects him for b^* in TMA

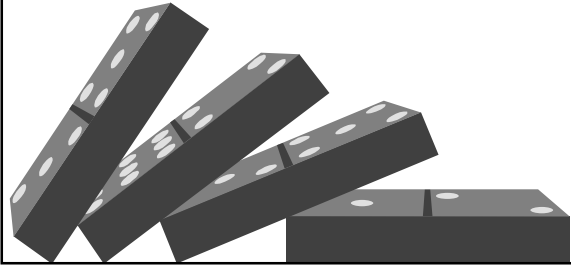
Some boy b got rejected by his optimal girl g because she said "maybe" to a preferred b' . b' likes g at least as much as his optimal girl.

There must exist a stable pairing S in which b and g are married.

- b^* wants g more than his wife in S
 - g is as at least as good as his best and he does not have her in stable pairing S
- g wants b^* more than her husband in S
 - b is her husband in S and she rejects him for b^* in TMA

What proof technique did we just use?

What proof technique did we just use?



Theorem: The TMA pairing, T , is female-pessimal.

We know it is male-optimal. Suppose there is a stable pairing S where some girl g does worse than in T .

Let b be her mate in T .

Let b^* be her mate in S .

- By assumption, g likes b better than her mate in S
- b likes g better than his mate in S
 - We already know that g is his optimal girl
- **Therefore, S is not stable.**



Advice to females

Learn to make the first move.

The largest, most successful dating service in the world uses a computer to run TMA!

"The Match":
Doctors and Medical Residencies

- Each medical school graduate submits a ranked list of hospitals where he/she wants to do a residency
- Each hospital submits a ranked list of newly minted doctors
- A computer runs TMA (extended to handle Mormon marriages)
- Until recently, it was hospital-optimal

History

1900

- Idea of hospitals having residents (then called "interns")

Over the next few decades

- Intense competition among hospitals for an inadequate supply of residents
 - Each hospital makes offers independently
 - Process degenerates into a race. Hospitals steadily advancing date at which they finalize binding contracts

History

1944 Absurd Situation. Appointments being made 2 years ahead of time!

- All parties were unhappy
- Medical schools stop releasing any information about students before some reasonable date
- Did this fix the situation?

History

1944 Absurd Situation. Appointments being made 2 years ahead of time!

- All parties were unhappy
- Medical schools stop releasing any information about students before some reasonable date
- **Offers were made at a more reasonable date, but new problems developed**

History

1945-1949 Just As Competitive

- Hospitals started putting time limits on offers
- Time limit gets down to 12 hours
- Lots of unhappy people
- Many instabilities resulting from lack of cooperation

History

1950 Centralized System

- Each hospital ranks residents
- Each resident ranks hospitals
- National Resident Matching Program produces a pairing

Whoops! The pairings were not always stable. By 1952 the algorithm was the TMA (hospital-optimal) and therefore stable.

History Repeats Its elf!

NY TIMES, March 17, 1989

- The once decorous process by which federal judges select their law clerks has degenerated into a free-for-all in which some of the judges scramble for the top law school students . . .
- The judge have steadily pushed up the hiring process . . .
- Offered some jobs as early as February of the second year of law school . . .
- On the basis of fewer grades and flimsier evidence . . .

NY TIMES

- "Law of the jungle reigns . . ."
- The association of American Law Schools agreed not to hire before September of the third year . . .
- Some extend offers from only a few hours, a practice known in the clerkship vernacular as a "short fuse" or a "hold up".
- Judge Winter offered a Yale student a clerkship at 11:35 and gave her until noon to accept . . . At 11:55 . . . he withdrew his offer