Great Theoretical Ideas in Computer Science

Course Staff
Instructor
Luis von Ahn
TAs
Anton Bachin
Jeremiah Blocki
Matt Bonakdarpour
Severin Hacker
Daniel Schafer
Jonah Sherman
Ben Wolf

Grading
Homework 40%
Final 25%
In-Class Quizzes 5%
Participation 5%

If Suzie gets 60, 90, and 80 in her tests, how many total test points will she have in her final grade?

\[
(0.05)(60) + (0.10)(90) + (0.10)(80) = 20
\]
Weekly Homework
Homework will go out every Tuesday and is due the Tuesday after
Ten points per day late penalty
No homework will be accepted more than three days late
Homework MUST be typeset

Collaboration + Cheating
You may NOT share written work
You may NOT use Google, or solutions to previous years’ homework
You MUST sign the class honor code

Textbook
There is NO textbook for this class
We have class notes in wiki format
You too can edit the wiki!!!

Feel free to ask questions

Bits of Wisdom on Solving Problems, Writing Proofs, and Enjoying the Pain: How to Succeed in This Class
Lecture 1 (January 15, 2008)

What did our brains evolve to do?
What were our brains “intelligently designed” to do?
What kind of meat did the Flying Spaghetti Monster put in our heads?

Our brains did NOT evolve to do math!
Over the last 30,000 years, our brains have essentially stayed the same!
The human mind was designed by evolution to deal with foraging in small bands on the African Savannah... faulting our minds for succumbing to games of chance is like complaining that our wrists are poorly designed for getting out of handcuffs.

Steven Pinker
“How the Mind Works”

Our brains can perform simple, concrete tasks very well.

And that’s how math should be approached!

Substitute concrete values for the variables: x=0, x=100, ...

Draw simple pictures

Try out small examples of the problem: What happens for n=1? n=2?

“I don’t have any magical ability...I look at the problem, and it looks like one I’ve already done. When nothing’s working out, then I think of a small trick that makes it a little better. I play with the problem, and after a while, I figure out what’s going on.”

Terry Tao (Fields Medalist, considered to be the best problem solver in the world)
The better the problem solver, the less brain activity is evident. The real masters show almost no brain activity!

Simple and to the point

Use a lot of paper, or a board!!!

Quick Test...

Count the green squares (you will have three seconds)
How many were there?

Hats with Consecutive Numbers

Alice starts: ...

\[ |A - B| = 1 \]

Alice, Bob

Hats with Consecutive Numbers

Alice

Bob

I don’t know what my number is

(round 1)

\[ |A - B| = 1 \text{ and } A, B > 0 \]

Alice starts: ...

Alice
Hats with Consecutive Numbers

Alice starts: …

| A - B | = 1 and A, B > 0

I don’t know what my number is
(round 2)

Alice

Bob

I don’t know what my number is
(round 3)

Alice

Bob

| A - B | = 1 and A, B > 0

Alice starts: …

I don’t know what my number is
(round 4)

Alice

Bob

| A - B | = 1 and A, B > 0

Alice starts: …
Hats with Consecutive Numbers

Alice starts: …

| A - B | = 1 and A, B > 0

I know what my number is!!!!!!!!

(round 251)

Alice

Bob

Hats with Consecutive Numbers

Alice starts: …

| A - B | = 1 and A, B > 0

I know what my number is!!!!!!!!

(round 252)

Alice

Bob

Question: What are Alice and Bob’s numbers?

Imagine Alice Knew Right Away

Alice

Bob

| A - B | = 1 and A, B > 0
Then A = 2 and B = 1

I know what my number is!!!!!!!!

(round 1)
1,2  N,Y
2,1  Y
2,3  N,Y
3,2  N,N,Y
3,4  N,N,N,Y
4,3  N,N,Y
4,5  N,N,N,Y

Inductive Claim

Claim: After 2k NOs, Alice knows that her number is at least 2k+1.

After 2k+1 NOs, Bob knows that his number is at least 2k+2.

Hence, after 250 NOs, Alice knows her number is at least 251. If she says YES, her number is at most 252.

If Bob’s number is 250, her number must be 251. If his number is 251, her number must be 252.

Exemplification:
Try out a problem or solution on small examples. Look for the patterns.

A volunteer, please
Relax
I am just going to ask you a Microsoft interview question

Four guys want to cross a bridge that can only hold two people at one time. It is pitch dark and they only have one flashlight, so people must cross either alone or in pairs (bringing the flashlight). Their walking speeds allow them to cross in 1, 2, 5, and 10 minutes, respectively. Is it possible for them to all cross in 17 minutes?

Get The Problem Right!
Given any context you should double check that you read/heard it correctly!
You should be able to repeat the problem back to the source and have them agree that you understand the issue

Four guys want to cross a bridge that can only hold two people at one time. It is pitch dark and they only have one flashlight, so people must cross either alone or in pairs (bringing the flashlight). Their walking speeds allow them to cross in 1, 2, 5, and 10 minutes, respectively. Is it possible for them to all cross in 17 minutes?
**Intuitive, But False**

“$10 + 1 + 5 + 1 + 2 = 19$, so the four guys just can’t cross in $17$ minutes”

“Even if the fastest guy is the one to shuttle the others back and forth – you use at least $10 + 1 + 5 + 1 + 2 > 17$ minutes”

**Vocabulary Self-Proofing**

As you talk to yourself, make sure to tag assertions with phrases that denote degrees of conviction

---

If it is possible, there must be more than one guy doing the return trips: it must be that someone gets deposited on one side and comes back for the return trip later!

---

Keep track of what you actually know – remember what you merely suspect

“$10 + 1 + 5 + 1 + 2 = 19$, so it would be weird if the four guys could cross in $17$ minutes”

“even if we use the fastest guy to shuttle the others, they take too long.”
Suppose we leave 1 for a return trip later.

We start with 1 and X and then X returns.
Total time: 2X

Thus, we start with 1,2 go over and 2 comes back....
5 and 10
“Load Balancing”:
Handle our hardest work loads in parallel!
Work backwards by assuming 5 and 10 walk together

In this course you will have to write a lot of proofs!
<table>
<thead>
<tr>
<th>MINESWEEPER</th>
<th>Mines Left: 7</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3   * 2 0 0 0 0 0</td>
</tr>
<tr>
<td></td>
<td>*  * 3 1 1 1 0 0</td>
</tr>
<tr>
<td></td>
<td>2   1 1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MINESWEEPER</th>
<th>Mines Left: 6</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3   * 2 0 0 0 0 0</td>
</tr>
<tr>
<td></td>
<td>*  * 3 1 1 1 0 0</td>
</tr>
<tr>
<td></td>
<td>2   1 1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MINESWEEPER</th>
<th>Mines Left: 6</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3   * 2 0 0 0 0 0</td>
</tr>
<tr>
<td></td>
<td>*  * 3 1 1 1 1 0</td>
</tr>
<tr>
<td></td>
<td>2   1 0</td>
</tr>
<tr>
<td></td>
<td>1   0</td>
</tr>
<tr>
<td></td>
<td>1   1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MINESWEEPER</th>
<th>Mines Left: 5</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3   * 2 0 0 0 0 0</td>
</tr>
<tr>
<td></td>
<td>*  * 3 1 1 1 0 0</td>
</tr>
<tr>
<td></td>
<td>4   3   2 1 1 0 0</td>
</tr>
<tr>
<td></td>
<td>1   0</td>
</tr>
<tr>
<td></td>
<td>1   0</td>
</tr>
<tr>
<td></td>
<td>1   1</td>
</tr>
</tbody>
</table>
MINESWEEPER
Mines Left: 3

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>*</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>*</td>
<td>*</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>3</td>
<td>*</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>*</td>
<td>2</td>
<td>*</td>
<td>2</td>
<td>2</td>
<td>*</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>*</td>
<td>1</td>
<td>*</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

“Dad, it’s INDIAN-PENDANT like you said.” Isaac Rudich

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>*</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>*</td>
<td>*</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>3</td>
<td>*</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>*</td>
<td>2</td>
<td>*</td>
<td>2</td>
<td>2</td>
<td>*</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>?</td>
<td>?</td>
<td>1</td>
<td>?</td>
<td>?</td>
<td>1</td>
<td>?</td>
</tr>
</tbody>
</table>

Think of Yourself as a (Logical) Lawyer

Your arguments should have no holes, because the opposing lawyer will expose them

There is no sound reason to go from Statement₁ to Statement₂

Prover

Statement₁
Statement₂
...
Statementₙ

Verifier
The verifier is very thorough, (he can catch all your mistakes), but he will not supply missing details of a proof

A valid complaint on his part is: I don’t understand

The verifier is similar to a computer running a program that you wrote!

Writing Proofs Is A Lot Like Writing Programs

You have to write the correct sequence of statements to satisfy the verifier

Errors than can occur with a program and with a proof!

Syntax error
Undefined term
Infinite Loop
Output is not quite what was needed

Good code is well-commented and written in a way that is easy for other humans (and yourself) to understand

Similarly, good proofs should be easy to understand. Although the formal proof does not require certain explanatory sentences (e.g., “the idea of this proof is basically X”), good proofs usually do

Writing Proofs is Even Harder than Writing Programs

The proof verifier will not accept a proof unless every step is justified!

It’s as if a compiler required your programs to have every line commented (using a special syntax) as to why you wrote that line
A successful mathematician plays both roles in their head when writing a proof.

Gratuitous Induction Proof

\[ S_n = \text{"sum of first } n \text{ integers } = n(n+1)/2" \]

Want to prove: \( S_n \) is true for all \( n > 0 \)

Base case: \( S_1 = 1 = 1(1+1)/2 \)

I.H. Suppose \( S_k \) is true for some \( k > 0 \)

Induction step:

\[ 1 + 2 + \ldots + n + (n+1) = k(k+1)/2 + (k+1) \text{ (by I.H.)} = (k + 1)(k+2)/2 \]

Thus \( S_{k+1} \)

Wrong variable
<table>
<thead>
<tr>
<th>Proof by Throwing in the Kitchen Sink</th>
</tr>
</thead>
<tbody>
<tr>
<td>The author writes down every theorem or result known to mankind and then adds a few more just for good measure.</td>
</tr>
<tr>
<td>When questioned later, the author correctly observes that the proof contains all the key facts needed to actually prove the result.</td>
</tr>
<tr>
<td>Very popular strategy on 251 exams</td>
</tr>
<tr>
<td>Believed to result in partial credit with sufficient whining</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Proof by Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>The author gives only the case $n = 2$ and suggests that it contains most of the ideas of the general proof.</td>
</tr>
<tr>
<td>Like writing a program that only works for a few inputs</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Proof by Cumbersome Notation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Best done with access to at least four alphabets and special symbols.</td>
</tr>
<tr>
<td>Helps to speak several foreign languages.</td>
</tr>
<tr>
<td>Like writing a program that’s really hard to read because the variable names are screwy</td>
</tr>
</tbody>
</table>
Proof by Lengthiness

An issue or two of a journal devoted to your proof is useful. Works well in combination with Proof strategy #10 (throwing in the kitchen sink) and Proof strategy #8 (cumbersome notation).

Like writing 10,000 lines of code to simply print “hello world”

Switcharoo Example

$S_n = \text{“sum of first } n \text{ integers } = n(n+1)/2\text{”}$
Want to prove: $S_n$ is true for all $n > 0$

Base case: $S_1 = 1 = 1(1+1)/2$

I.H. Suppose $S_k$ is true for some $k > 0$

Induction step: by $S_{k+1}$

$1 + 2 + \ldots + k + (k+1) = (k + 1)(k+2)/2$

Hence blah blah, $S_k$ is true

Proof by Switcharoo

Concluding that $p$ is true when both $p \Rightarrow q$ and $q$ are true

Makes as much sense as:

```plaintext
If (PRINT “X is prime”) {
PRIME(X);
}
```

Proof by “It is Clear That…”

“It is clear that that the worst case is this:”

Like a program that calls a function that you never wrote
Proof by Assuming The Result

Assume X is true

Therefore, X is true!

Like a program with this code:

```c
RECURSIVE(X) {
  :
  return RECURSIVE(X);
}
```

"Assuming the Result" Example

\( S_n = \text{"sum of first } n \text{ integers} = n(n+1)/2" \)

Want to prove: \( S_n \) is true for all \( n > 0 \)

Base case: \( S_1 = "1 = 1(1+1)/2" \)

I.H. Suppose \( S_k \) is true for all \( k > 0 \)

Induction step:

\[
1 + 2 + \ldots + k + (k+1) = k(k+1)/2 + (k+1) \quad \text{(by I.H.)}
\]

\[
= (k + 1)(k+2)/2
\]

Thus \( S_{k+1} \)

3 Not Covering All Cases

Usual mistake in inductive proofs: A proof is given for \( N = 1 \) (base case), and another proof is given that, for any \( N > 2 \), if it is true for \( N \), then it is true for \( N+1 \)

Like a program with this function:

```c
RECURSIVE(X) {
  if (X > 2) { return 2*RECURSIVE(X-1); }
  if (X = 1) { return 1; }
}
```

"Not Covering All Cases" Example

\( S_n = \text{"sum of first } n \text{ integers} = n(n+1)/2" \)

Want to prove: \( S_n \) is true for all \( n > 0 \)

Base case: \( S_0 = "0 = 0(0+1)/2" \)

I.H. Suppose \( S_k \) is true for some \( k > 0 \)

Induction step:

\[
1 + 2 + \ldots + k + (k+1) = k(k+1)/2 + (k+1) \quad \text{(by I.H.)}
\]

\[
= (k + 1)(k+2)/2
\]

Thus \( S_{k+1} \)
Incorrectly Using “By Definition”

“By definition, \( \{ a^n b^n \mid n > 0 \} \) is not a regular language”

Like a program that assumes a procedure does something other than what it actually does.

Solving Problems

- Always try small examples!
- Use enough paper

Writing Proofs

- Writing proofs is sort of like writing programs, except every step in a proof has to be justified
- Be careful; search for your own errors