CS 5 finale

**Looking back!** Evals, Ideas

I'll be back...

That's my line!

CS 5, on the verge of termination

**Looking ahead?** Options...

CS 5 Final Projects

due this Fri. 5:00 pm  (or later w/ Euro)
Friday hours at the LAC !!!

CS 5 Review Session

7-8+ pm  Sunday, 12/15  in big Shanahan

CS 5 Final Exam

2pm Mon. 12/16 or 2pm Wed. 12/18, in big Shan
Two pages of notes are OK
practice probs, hwks, quizzes to review
The CS view of the world...

Input → Algorithm → Output

CS's challenge
Uncomputable functions

Sometimes an algorithm simply does not exist...

more precisely: every possible algorithm contains bugs!
Haltchecking is uncomputable.

\[ \text{hc}(f) \]

\[ \sim \, \text{return whether } f() \text{ halts or not} \]

\textbf{Halt Checking}

\textbf{hc} can't be written well, not without bugs
In computability theory, the **halting problem** can be stated as follows: Given a description of a computer program, decide whether the program finishes running or continues to run forever. This is equivalent to the problem of deciding, given a program and an input, whether the program will eventually halt when run with that input, or will run forever.

**Alan Turing** proved in 1936 that a general algorithm to solve the halting problem for all possible program-input pairs cannot exist. A key part of the proof was a mathematical definition of a computer and program, what became known as a **Turing machine**; the
the iPhone's icon for Google Maps …
Haltchecking is uncomputable.

It is impossible to write a bug-free function \( \text{hc}(f) \) that determines if function \( f \) halts or loops forever:

1. \( \text{hc}(f) \) returns \text{True} if \( f() \) halts and
2. \( \text{hc}(f) \) returns \text{False} if \( f() \) loops infinitely
Haltchecking a few functions...

```python
def f():
    while True:
        print 'Ha!

def g():
    return 42

def BFF():
    if hc(BFF) == True:
        while True:
            print 'Ha!
    else:
        return 42
```

Should `hc` return `True` or `False` in each of these cases?
Haltchecking is uncomputable.

Suppose $hc(f)$ works for all $f$...

Then write this f'n, BFF:

```python
def BFF():
    if hc(BFF) == True:
        while True: print 'Ha!'
    else:
        return 42
```

What is $hc(BFF)$ True? False?
Why are these uncomputable?

Where did \textbf{kc} and \textbf{hc} go wrong?

They involved computing what other programs were going to do.

**Rice's Theorem:** programs \textbf{can not} determine the results of other programs. If they try, they always have bugs.
Meaningful functions?

Input → Algorithm → Output

but nearly all meaningful functions are computable...
CS 5's examples...

Input → Algorithm → Output

Connect 4 Board → intelligent move
Input text → Markov model + new text
current gener. of “life” cells → next gener. of “life” cells
CS 5's examples...

Input → Algorithm → Output

Connect 4 Board

Input

current generation of “life”

Intelligent move

next generation of “life” cells

new text

balance of utility & creativity
Connect Four results...
"Science without religion is lame, religion without science is blind."
"Two things are infinite: the universe and human stupidity; and I'm not sure about the universe."
"Duct tape is like the force, it has a light side, a dark side, and it holds the world together"
"If you die in an elevator, be sure to push the Up button."
"All generalizations are false, including this one."
"Clearly you've never been to Singapore!"
"Luke, I am your father."
"To be, or not to be."
"You shall not pass!"
(... and many others ...)

"I have a dream! Duct tape is written on. Luke, I am your thoughts and what lies within us."
    ---- Audrey Rooney

"Your work is lame, religion is lame, religion is nearly the Up button."
    ---- Abraham Marx

"Two things are false, including this one."
    ---- Captain_Jack Truman
nearly all meaningful functions are computable...

but this doesn't mean we know how to solve them (yet)!

for example, ...
the "computer vision" problem
computer vision: what's the input and output?
Input → Algorithm → Output

2d array of numbers

020 067 073 058 055 076 069 050 074 064 065 066 066 059 023
047 109 107 118 107 115 110 120 120 128 124 132 131
047 125 130 130 122 121 117 142 131 133 134 134 141 149 144 135
051 139 143 139 147 134 149 069 127 144 139 144 150 161 149
054 136 161 148 147 158 055 052 034 030 158 156 165 163 156
043 144 165 159 154 171 224 191 104 070 30 028 120 169 173 177 173
011 091 101 105 177 039 078 060 041 026 073 102 167 208 121
011 091 094 066 094 033 199 184 139 024 060 094 125 152 134
009 068 072 072 065 031 151 171 075 028 035 072 083 109 063
013 068 074 059 057 037 161 129 062 028 035 071 072 078 056
012 042 063 055 072 033 020 067 031 022 027 082 070 073 060
011 037 064 094 091 026 025 080 066 026 023 071 070 080 060
011 060 077 082 037 023 024 147 140 038 023 037 043 076 037
013 049 076 059 032 028 174 197 182 060 021 021 121 101 062
013 059 111 072 020 078 200 211 182 061 059 043 086 106
007 053 057 092 023 105 189 230 210 084 034 021 017 033 091
011 061 072 018 027 054 069 068 062 023 045 011 016 042 044
014 041 047 025 018 040 040 065 039 024 021 036 041 013 030 022
013 093 106 017 019 027 030 042 012 021 043 013 014 020 027
019 040 029 023 016 024 015 026 011 010 026 017 012 013 014
022 042 030 040 019 015 016 011 012 009 008 012 009 017 019
022 026 018 030 020 012 017 010 008 011 007 015 008 016 034
019 018 048 029 012 054 012 008 009 008 012 007 016 005
022 015 057 043 126 135 122 006 005 008 007 019 010 011 008
018 008 009 019 023 093 109 128 063 052 031 010 012 009 006
017 010 010 007 067 054 106 116 067 056 011 028 005 009 006
015 010 012 014 062 076 057 055 019 024 020 006 005 013 004
016 010 008 011 039 025 020 016 011 007 008 007 006 010 003
015 009 010 010 012 011 014 009 008 007 007 005 005 008 002
014 007 008 011 007 012 010 009 007 008 007 005 007 003
020 011 015 019 013 017 017 013 019 013 012 013 011 009 005
020 067 073 058 055 076 069 050 074 064 065 066 066 059 023
025 161 174 172 167 049 200 193 112 028 120 169 173 177 173
<table>
<thead>
<tr>
<th>Input</th>
<th>Algorithm</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>2D array of numbers</td>
<td>an image?</td>
<td>an image?</td>
</tr>
</tbody>
</table>
Why does this input/output example definitely **not** match?

*an image?*

2d array of numbers
Actual image

Actual input: **pixels**

Actual output: **contents**

it's a woman, smiling?
What's red?

Goal: a coke-can collecting robot...
What's red?

not good enough...
What's red?

| hue | < 25

| hue | < 25
Hue and saturation

- hue = 25°
- hue = -25°
What's red?

not good enough...

| hue | < 25
What's red?

| hue  | < 25
saturation | > 0.75
Hue and saturation

- **sat < 0.75**
- **sat > 0.75**

- **hue = 25°**
- **sat > 0.75**
- **hue = -25°**
What's red?

Aargh!

| hue | < 25

| saturation | > 0.75

The door is still matched, too... why?
What's red?

Remarkably, this problem is, in a sense, *our own computers' fault*...!

Aargh!

Door is still matched, too...  *why?*
Quiz

Illusions? What **computations** is your brain doing to cause them?  

Why?

two tones

two towers

two tables

two segments
2 layers
2 layers
2 layers

Edward H. Adelson
Are these lines parallel?
Quiz

Illusions? What computations is your brain doing to cause them?

Vision is more challenging than it might seem on first "glance"!
we don't always give our own vision system credit for everything it's doing...
we don't always give our own vision system credit for everything it's doing...
What's red?

<table>
<thead>
<tr>
<th>hue</th>
<th>&lt; 25</th>
</tr>
</thead>
<tbody>
<tr>
<td>saturation</td>
<td>&gt; 0.75</td>
</tr>
</tbody>
</table>

Aargh!

The door is still matched, too...  why?
The coke-can collector: *wall-avoidance*
The coke-can collector: *seeking*...
The coke-can collector: seeking...
The coke-can collector: *success!*

![Image of a robot collecting a coke can](image-url)
State of the "art" ...

Actual output: contents people, walking...
State of the "art" ...

3-Sweep: Extracting Editable Objects from a Single Photo, SIGGR...

Actual output: **models**

a tap against a wall...
3d reconstruction?

Incredibly, our vision systems can do this.
An elegant example of the 2d and 3d geometry of vision
Fukuda's vision atop vision's geometry
16 January 2009

SHIGEO FUKUDA Passes Away At 76

Tokyo (Japan) - At the age of 76 years old, Mr. Shigeo Fukuda passed away suddenly in Tokyo on Sunday, 11 January 2009 as the result of a subarachnoid hemorrhage. As one of the founding members and directors of the Japan Graphic Designers Association (JAGDA), Mr. Fukuda contributed profoundly to the development of the Association. The whole creative industry and society of Japan was struck with this sad and sudden news. His funeral has already been held with family members.

Fukuda's vision atop vision's geometry
Incredibly, our vision systems can do this.
State of the "art" ...

Actual output: contents 3d world model
CS in surprising places...

Automated Material Handling Order Fulfillment System

Kiva Systems, M. Mountz
CS 5: the past...

Functions & variables

Recursion

Representations (binary, ascii)

Circuit design & Hmmm

Loops, 2d arrays

Dictionaries

Objects and Classes

Computability

Caesar cipher

4-bit multiplier

Mandelbrot, Life

Markov Text Gen.

Date, C4, Project

Finite state machines

Uncomputable functions
CS 5: the past...

Functions & variables

Recursion

Representations (binary, ascii)

Circuit design & Hmmm

Loops

Dictionaries

Objects and Classes

Computability

Future CS?

- Caesar cipher
- 4-bit multiplier
- Mandelbrot, Life
- Markov Text Gen.
- Date, C4, Project
- Finite state machines
- Uncomputable functions
**Interfacing**

*User Interfaces, Graphics, Animation*

CS 124, 155, 157

**AI**

*AI, Neural Networks, Computer Vision, Robotics*

CS 151, 152, 153, 154

**Systems**

*Compilers, Programming Languages, Networking, Operating Systems, Computer Architecture*

Eng 85, 155, 158 & CS 125, 131, 132, 134, 136

**Theory**

*Theory of Computation, Advanced Algorithms*

Math 167, 168 & CS 141, 142

---

**SW Engineering & Data structures**

CS 70, 121

**Logic & Computability**

CS 81

**Principles of CS**

CS 60
Summer CS

Project pitches: *First Thursday of the spring term...*

but what if I'm nostalgic for CS 5 itself?
Summer CS

Project pitches:  First Thursday of the spring term...

but what if I'm nostalgic for CS 5 itself?

you might also consider grading for CS5 next fall – all schools welcome...
I AM THE WINNER AND IT ALL WORKS!!!
GWAHAHAHAHAHAHAHAHAHAH!!

I started at noon, I'm done at 11:11

minus a brief break to watch shawshank redemption and eat dinner,
I've been working solid

I AM TOTALLY THE WINNER!
Final Projects: due Friday @ 5pm

Exam: next Mon. or Wed. at 2 pm  (Big Shanahan)

REVIEW: Optional review session covering the practice final and any other questions...

Sunday evening, 7-8+ pm  (Big Shanahan)
Sunday evening, 7-8+ pm   (Big Shanahan)

Exam: next Mon. or Wed. at 2 pm   (Big Shanahan)

Optional review session covering the practice final and any other questions…

Final Projects:  due Friday @ 5pm

REVIEW:

Thank you for joining CS5!

Sincerely,

(Signed Shanahan)
we don't always give our own vision system credit for everything it's doing...
human vision can be deceived, too!
2 layers
2 layers

Edward H. Adelson
Are these lines parallel?
Quiz

Illusions? What computations is your brain doing to cause them?

Vision is more challenging than it might seem on first "glance"!
**Principles of CS**

**Logic & Computability**

CS 60

**Data structures**

CS 70, 121

**SW Engineering &**

**Logic & Computability**

CS 81

**User Interfaces, Graphics, Animation**

CS 124, 155, 157

**AI**

**AI, Neural Networks, Computer Vision, Robotics**

CS 151, 152, 153, 154

**Compilers, Programming Languages, Networking,**

**Operating Systems, Computer Architecture**

Eng 85, 155, 158 & CS 125, 131, 132, 134, 136

**Theory**

**Theory of Computation, Advanced Algorithms**

Math 167, 168 & CS 141, 142

**Interfaces**

**User Interfaces, Graphics, Animation**

CS 124, 155, 157

**AI**

**AI, Neural Networks, Computer Vision, Robotics**

CS 151, 152, 153, 154

**Systems**

**Compilers, Programming Languages, Networking,**

**Operating Systems, Computer Architecture**

Eng 85, 155, 158 & CS 125, 131, 132, 134, 136

**Theory**

**Theory of Computation, Advanced Algorithms**

Math 167, 168 & CS 141, 142
Who is *whoami*?
Suppose your list has $N$ elements...

$L = [4, 7, 42, 1, \ldots, 0, 14, -5, 5]$

Find $L$'s max

Sort $L$

Will sorting or max-finding take more work?
Suppose your list has \( N \) elements...

\[ L = [4, 7, 42, 1, \ldots, 0, 14, -5, 5] \]

**Find \( L \)'s max** \( \leftrightarrow \) **Find \( L \)'s median** \( \leftrightarrow \) **Sort \( L \)**

*How much work will median-finding take?*
the iPhone's icon for Google Maps …
Future CS?

CS's Summer projects colloquium

you might consider grading for CS5 next fall – all schools welcome...

Most likely:
Th., January 19, 2012
~ place TBA

(probably right here)
CS 5 finale

CS 5 Practice Exam

CS 5 Review Session
4-5pm Sunday, 12/16 here in Pryne

CS 5 Final Exam  
2pm - on Thu. or Fri. 12/17-18 here in Pryne
2 pages of notes are OK
practice exam is a good review

cs 5 Final Projects

due Fri. 11:59 pm  (or Sun. w/ Euro)

Looking back! evals, ideas

Looking ahead? Other options
CS in the next few years...

the long-awaited emergence of Artificial Intelligence?

Natural speech interactions

Siri+ ?

Interpretation of images

Google + facebook recognition

Physical interactions with the world...

autonomous cars?
Color Tracking

relying on our perceptions of color can be deceiving
Other Courses?

More Python ~ PyGame

That's me!
Color definitions

matching based on pixel colors

Color-based matching...

red > 200

$|\text{hue}| < 0.1$

$|\text{hue}| < 0.1$

sat > 0.7
Future CS?

CS's Summer projects colloquium

Thursday, January 20, 2011 ~ place TBA

(but probably right here)
# Name: Sami Mourad. Height: 1m85cm. Weight: XXX. Color of eyes: brown.
# Color of hair: brown. Tel: 424-242-4242

# Dear HamsterBob, I am sorry to have deceived you with my foolishly
# negligent header comments. In an attempt to make up for my superciliousness,
# I have written a short story based on Goldie Locks and the three bears,
# and have incorporated it into my old connect four code to your (optional) enjoyment.

# Once upon a saturday morning...

# Big daddy bear wakes up from a dream filled with Cucurbita desserts.

```python
from random import *

class Board:
    """ a datatype representing a C4 board
    with an arbitrary number of rows and cols
    """

    def __init__( self, width, height ):
        """ the constructor for objects of type Board """
        self.width = width
        self.height = height
        self.data = [ [ ] ] * height # for row in range(n)
        # Bears love Cucurbita Maxima pies.
```

Grading CS5!

Sami
Summer CS

Missile launcher