Why Computer Science is Uglier and Prettier than Mathematics

Owen Astrachan
Duke University

http://www.cs.duke.edu/~ola
What is Computer Science?

What is it that distinguishes it from the separate subjects with which it is related? What is the linking thread which gathers these disparate branches into a single discipline? My answer to these questions is simple --- it is the art of programming a computer. It is the art of designing efficient and elegant methods of getting a computer to solve problems, theoretical or practical, small or large, simple or complex.

C.A.R (Tony) Hoare
What can be programmed?

- **What class of problems can be solved?**
  - G4, 500Mhz Pentium III, Cray, pencil?
  - Alan Turing proved somethings, hypothesized others
    - Halting problem, Church-Turing thesis

- **What class of problems can be solved efficiently?**
  - Problems with no practical solution
    - what does practical mean?
  - Problems for which we can’t find a practical solution
    - solving one solves them all
Schedule students, minimal conflicts

- **Given student requests, available teachers**
  - write a program that schedules classes
  - Minimize conflicts

- **Add a GUI too**
  - Web interface
  - ...
  - ...

I can’t write this program because I’m too dumb
One better scenario

I can’t write this program because it’s provably impossible
Another possible scenario

I can’t write this program but neither can all these famous people
The halting problem: writing `DoesHalt`

```cpp
bool DoesHalt(const string& progname,  
               const string& s)  
// post: returns true if progname halts given s  
//       as input, false otherwise  

int main()  
{  
    string f = PromptString("enter filename ");  
    string s = PromptString("input for "+filename);  
    if (DoesHalt(f,s)) cout << "does halt" << endl;  
    else cout << "does not halt" << endl;
}  
```

- A compiler is a program that reads other programs as input  
  - Can a word counting program count its own words?  
- The `DoesHalt` function might simulate, analyze, ...  
  - One program/function that works for any program/input
Consider the program *confuse.cpp*

```cpp
#include "halt.h"
int main()
{
    string f = PrompString("enter filename ");
    if (DoesHalt(f,f))
    {
        while (true)
        {
            // do nothing forever
        }
    }
    return 0;
}
```

- **We want to show writing **DoesHalt** is impossible**
  - Proof by contradiction:
  - Assume possible, show impossible situation results
Not impossible, but impractical

- Towers of Hanoi
  - How long to move n disks?

- What combination of switches turns the light on?
  - Try all combinations, how many are there?
  - Is there a better way?
Travelling Salesperson

- Visit every city exactly once
- Minimize cost of travel or distance
- Is there a tour for under $2,000? less than 6,000 miles?
- Is close good enough?

Try all paths, from every starting point -- how long does this take?

a, b, c, d, e, f, g
b, a, c, d, e, f, g ...
Complexity Classifications

- This route hits all cities for less than $2,000 --- \textit{verify} properties of route efficiently.
- Hard to \textit{find} optimal solution

Pack trucks with barrels, use minimal \# trucks

Ideas?

Problems are the “same hardness”: solve one efficiently, solve them all
Are hard problems easy?

- **P = easy problems, NP = “hard” problems**
  - P means solvable in polynomial time
    - Difference between $N$, $N^2$, $N^{10}$?
  - NP means non-deterministic, polynomial time
    - guess a solution and verify it efficiently

- **Question: P = NP?**
  - if yes, a whole class of difficult problems can be solved efficiently---one problem is reducible to another
  - if no, none of the hard problems can be solved efficiently
  - showing the first problem was in NP was an exercise in intellectual bootstrapping (1971)
Theory and Practice

- **Number theory: pure mathematics**
  - How many prime numbers are there?
  - How do we factor?
  - How do we determine primeness?

- **Computer Science**
  - Primality is “easy”
  - Factoring is “hard”
  - Encryption is possible
Shafi Goldwasser

- RCS professor of computer science at MIT
  - Co-inventor of zero-knowledge proof protocols

How do you convince someone that you know something without revealing “something”

- Consider card readers for dorms
  - Access without tracking

Work on what you like, what feels right, I now of no other way to end up doing creative work
Why is programming fun?

What delights may its practitioner expect as a reward?
First is the sheer joy of making things

Second is the pleasure of making things that are useful
Third is the fascination of fashioning complex puzzle-like objects of interlocking moving parts

Fourth is the joy of always learning
Finally, there is the delight of working in such a tractable medium. The programmer, like the poet, works only slightly removed from pure thought-stuff.

Fred Brooks
What is computer science?

- **What is a computation?**
  - Can formulate this precisely using mathematics
  - Can say “anything a computer can compute”
  - Study both theoretical and empirical formulations, build machines as well as theoretical models

- **How do we build machines and the software that runs them?**
  - Hardware: gates, circuits, chips, cache, memory, disk, ...
  - Software: operating systems, applications, programs

- **Art, Science, Engineering**
  - How do we get better at programming and dealing with abstractions
  - What is hard about programming?
Simple, Elegant, Deep

\[(a+b)^2 = c^2 + 4\left(\frac{1}{2}ab\right)\]

\[(a+b)^2 = c^2 + 4\left(\frac{1}{2}ab\right)\]

\[a^2 + 2ab + b^2 = c^2 + 2ab\]

\[a^2 + b^2 = c^2\]
The selection problem

- In a list of $N$ ordered items, find the $k^{th}$ smallest
  - Highest salary, median, 90%, ...
  - Solve the problem, then solve it efficiently

- Suppose we can re-arrange the items

- Where is $k^{th}$ smallest?
  - It’s $x$
  - It’s before $x$
  - It’s after $x$
The Selection Problem

- In a list of N ordered items, find the k\textsuperscript{th} smallest
  - Highest salary, median, 90\%, ...
  - Solve the problem, then solve it efficiently

```cpp
double Select(tvector<double> & a,  
               int left, int right, int k)  
// pre: left <= right  
// post: return k-th smallest in a[left..right]  
{
    if (left == right) return a[left];
    int p = Partition(a, left, right); // re-arrange a  
    int size = p - left + 1; // left sublist  
    if (k <= size) return Select(a, left, p, k);  
    else return Select(a, p+1, right, k-size);
}
```
Partition, the picture and invariant

The desired state, where we want to go

The initial state

The intermediate state, and invariant
Partition: the code

```c
int Partition(tvector<int> & a, int left, int right)
// pre: left <= right, and legal indices for a
// post: return index and re-arrange elements in a
// so a[left..index] <= a[index+1..right]
{
    int p=left;
    int k;
    for(k=left+1; k <= right; k++)
    {
        if (a[k] <= a[left])
        {
            p++;
            Swap(a[k], a[p]);
        }
    }
    Swap(a[left],a[p]);
    return p;
}
```

Duke University Computer Science
void QuickSort(tvector<int> & a, int left, int right)
// pre: left <= right, and legal indices for a
// post: a[left] <= … <= a[right]
{
    if (left <= right)
    {
        int p = Partition(a, left, right);
        QuickSort(a, left, p-1);
        QuickSort(a, p+1, right);
    }
}

● Why is this fast? When is it slow?
  ➢ Invented in 1960, hard to understand, why?
  ➢ Usable in mission critical applications? INTROSORT in 1998
C.A.R. (Tony) Hoare (b. 1934)

- Won Turing award in 1980
- Invented quicksort, but didn’t see how simple it was to program recursively
- Developed mechanism and theory for concurrent processing
- In Turing Award speech used “Emporer’s New Clothes” as metaphor for current fads in programming

“Beginning students don’t know how to do top-down design because they don’t know which end is up”
Practice with invariants

- Remove zeros from an array, leave order of non-zeros unchanged (AP exam, 1987)

2 1 0 5 0 0 8 4

- Sketch/write a solution
  
  ➢ Make it run, make it right, make it fast

  non-zeros (ignored) ? ? ?

  lnz k
Shoulders of Giants

Newton: “If I have seen farther it is because I have stood on the shoulders of giants.”

Robert Burton: “a dwarf standing on the shoulders of a giant may see farther than the giant himself.”
Richard Stallman (born 1953)

- Described by some as “world’s best programmer”
  - Wrote/developed GNU software tools, particularly g++
  - Believes all software should be free, but like “free speech”, not “free beer”
  - Won MacArthur award for his efforts and contributions
  - League for Programming Freedom

- Gnu/Linux is a free operating system and computing environment
  - Heavy industry/web use
  - Wintel killer??

• Local tie-in: Red Hat Linux,
  • headquartered in Durham, NC
  • IPO in 1999 at $14
  • One month later at $110+
  • Markets “free” product