From Contest Problem to Course Unit

Workshop at the

Schweizer Tag für Informatik Unterricht

15 January 2010, 16:00–17:30

Tom Verhoeff
Eindhoven University of Technology
Department of Mathematics & Computer Science
Software Engineering & Techology Group
The Netherlands
www.win.tue.nl/~wstomv

Workshop Programme

1. Introduction
2. Step-by-Step Development Plan
3. Apply to Your Problem
4. Helpful Tools
5. Play with Tools

Algorithmic Problem Solving in Teaching Informatics

- Problems and puzzles motivate and activate students
- Problems must be perceived as "demanding a solution"
- Danger of presenting a disconnected "bag of tricks"
- Problems in (international) informatics contests are challenging
- Helps distinguish theoretical analysis (algorithmic thinking) and practical implementation work (programming)

ACM ICPC World Finals 1990: Rare Order

A rare book collector recently discovered a book written in an unfamiliar language that used the same characters as the English language.

The book contained a short index, but the ordering of the items in the index was different from what one would expect if the characters were ordered the same way as in the English alphabet.

The collector tried to use the index to determine the ordering of characters (i.e., the *collating sequence*) of the strange alphabet. . . .

Sample Input	Sample Output
XWY	XZYW
ZX	
ZXY	
ZXW	
YWWX	

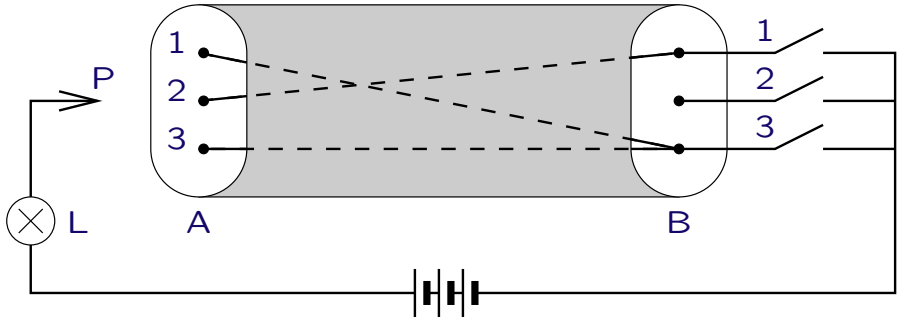
Background

- Non-standard problem
- Involves lexicographic order on strings
- Used as example in my
  Guidelines for Producing a Programming-Contest Problem Set
- Understanding, analysis, algorithm design, and implementation all play a role and can be clearly separated

International Olympiad in Informatics: IOI 1994

- 6 programming problems of varying difficulty level
- Documented solutions: `olympiads.win.tue.nl/ioi/ioi94/contest`

IOI 1995: Wires and Switches (first reactive task at IOI)



Secret mapping $f : A \to B$, wire i at A connected to switch $f(i)$ at B

Operations:

Probe wire i at side A: light L goes on iff switch $f(i)$ conducts

Change switch j at side B: toggles conductivity off $\leftrightarrow$ on
initially all off

Goal: reconstruct f efficiently (restricted number of operations)

Background

- Origin: moved into new home, electricity group chart lost
- Allows very diverse simple algorithms (quadratic)
- Efficient solution not too difficult
- Interactive Java applet for experimenting
- Evaluation used demonic adversary (minimizing information)
- Publication: "The Lost Group Chart and Related Problems"

  `www.win.tue.nl/~wstomv/publications/kruseman.pdf`
  `www.win.tue.nl/~wstomv/applets/wires/WiresApplet.html`

IOI 2000: Median (reactive task)

Given: an odd number of objects, all of distinct secret *weight*

Function $Med3$ returns the object of median (middle) weight among three distinct objects:

$$\{a,b,c\} \;=\; \Big\{ \min\{a,b,c\}\,,\; Med3(a,b,c)\,,\; \max\{a,b,c\} \Big\}$$

Goal: determine the object of median weight among all given objects, using only function $Med3$ (restricted number of calls)

Background

- Origin: during a brainstorm session in a bar with Gyula Horváth
- Allows very diverse simple algorithms (quadratic)
- Efficient (worst-case) solution involves interesting data structure
- Randomized algorithms can do well ('on average')
- Optimal solution unknown
- Publication: "Finding the Median under IOI Conditions" , *Inf. Edu.*
  `www.win.tue.nl/~wstomv/publications/INFO360.pdf`

Settling Multiple Debts Efficiently

A group of friends lend each other money throughout the year.

They carefully record each transaction.

When Alice lends 10 euro to Bob, this is recorded as Alice $\xrightarrow{10}$ Bob.

At the end of the year they wish to settle their debts.

How should they transfer money so as to settle all debts?

1. Minimize the total amount transferred.

2. Minimize the number of transfers.

Background

- Origin: bookkeeping at my wife's physical therapy practice
- Gives rise to various discoveries, including tables and labeled graphs
- First variant allows elegant linear solution
- Second variant is NP-hard
- Publication: "Settling Multiple Debts Efficiently", *Inf. Edu.*
  `www.win.tue.nl/~wstomv/publications/INFE023.pdf`
  `www.win.tue.nl/~wstomv/publications/lesson-plan.pdf`
  `www.win.tue.nl/~wstomv/publications/settling-debts-problems.pdf`

From Contest Problem to Course Module

- Select contest problem
- Prepare teaching plan and material
- Teach class
- Revise

Polya's Approach to Problem Solving

George Polyá. *How to Solve It*. Princeton Univ. Press, 1945, 1957.

1. Understand the problem
2. Devise a plan
3. Carry out the plan
4. Look back (reflect)

Polya offers many heuristic strategies:

`en.wikipedia.org/wiki/How_to_Solve_It`

Prepare Teaching Material

Your teaching material

- should support Polya's approach;
- should encourage students to do most of the work themselves;
- should set a good example for style;
- could support alternative pathways.

Step-by-Step Development Plan

1. Problem statement and problem-related artifacts (library, test data)
2. Investigate problem domain, notions involved, definitions
3. Understand the problem, more examples and exercises
4. Analyze, experiment
5. Simplify, break down/decompose, develop theory
6. Algorithm specification, selection, design
7. Program design, coding, verification
8. Reflect, problem variants

Step 5: Decompose

- input, output, parsing, preprocessing
- data (information) representation, transformation
- explicitly express and write down design decisions
- invent new notions (creativity)
- divide & conquer, in small steps
- define separate programming subproblems, solve them, verify them
- strenghten precondition, weaken postcondition, lift restrictions
- can provide (implemented) solutions to subproblems

Step 8: Problem Variants

- ...
- Other (related) computations on same input
- Check input format, validity
- Check output format, validity

peach^3

- Open-source web-based client-server system: `peach3.nl`
- Various user categories: student, grader, teacher, admin, observer
- Collect, store, evaluate submitted work, feedback, and results
- Enforce deadlines, fraud detection
- Supports multiple courses, with groups, over multiple years
- Evaluation configurable per assignment
- Supports multiple (programming) languages

What peach^3 Is (Not)

peach^3 is not intended as a full-blown generic

- student administration system
- course management system (cf. Moodle.org)
- web content management system (WCMS)
- workflow management system
- program development environment (IDE)
- version management system (cf. Subversion)

Low-threshold Facility for Introduction to Programming

- Tom's JavaScript Machine:

  `www.win.tue.nl/~wstomv/edu/javascript`

- Zero install: runs in any (modern) browser
- Easy to make teaching material with embedded programs
- Adaptable: Event-driven GUI, web apps (DOM), turtle graphics, ...

Conclusion

- Programming contests are a rich source of algorithmic problems
- They need (re)work to use in regular education
- peach^3 is a management tool for programming education
- *Tom's JavaScript Machine* can serve as low-threshold introduction

Links

www.win.tue.nl/~wstomv

olympiads.win.tue.nl/ioi/ioi94/contest

Wires: www.win.tue.nl/~wstomv/publications/kruseman.pdf
www.win.tue.nl/~wstomv/applets/wires/WiresApplet.html

Find Median: www.win.tue.nl/~wstomv/publications/INFO360.pdf

Settling Debts: www.win.tue.nl/~wstomv/publications/INFE023.pdf
www.win.tue.nl/~wstomv/publications/lesson-plan.pdf
www.win.tue.nl/~wstomv/publications/settling-debts-problems.pdf

en.wikipedia.org/wiki/How_to_Solve_It

peach3.nl demo.peach3.nl

www.win.tue.nl/~wstomv/edu/javascript