A Mental Game as a Source of CS Case Studies

Chris Healy
Department of Computer Science
Furman University

Starting with a game

- Apterous: implemented by students at Cambridge University
  - Based on *Countdown* in the UK and *Des Chiffres et des Lettres* in France
  - Letters game

    e.g. Find a word from GLAEITDTA
  - Numbers game

    e.g. 2 3 7 8 9 10 → 403
- Analysis questions about the games lend themselves to many possible projects in computer science classes.

Opportunities

Courses where I have used Apterous material

- Introduction to Python
- Parallel programming
- Discrete structures
- Data structures and algorithms
- Independent research project

Letters game assignments

- Scan the online “dictionary” and see how many words have n letters. (c consonants and v vowels)
- Simulate game to find optimal solutions for each. Do this many times to detect patterns…
  - Which words appear most often as the longest possible solution (i.e. words guaranteeing points) ?
  - How long is the longest possible word: how often does it have k letters? Average max score per round.
  - Finding words that are most often the unique longest word.
- Random sampling as opposed to brute force

Quantitative analysis

Discrete math class…

- A selection of letters must have 3, 4, or 5 vowels.
- Can calculate the number of possible number of distinct letter selections for a game.
- Probability of a letter appearing mirrors actual frequency analysis
  - Analysis of words in a dictionary (q is least common)
  - Analysis of a corpus of text (z is least common)
- Which of the 13+ billion letter selections are most likely to appear?

Numbers game

- Mapping between
  - Game instances (selection of numbers)
  - Mathematical expressions
- In discrete math, we can calculate the total number of both sets
  - Catalan number: How many ways are there to draw a binary tree having (6) nodes?
  - Generating functions: How many ways can we select 6 numbers from a set? And similar questions
  - Total number of expressions > 424 billion.

Extended project

- Reducing the number of expressions, due to the inherent redundancy.
- Incremental development
  - Removing permutations that are not distinguishable (from 410 billion to 227 billion)
  - Discard expressions that have intermediate result of 0, or whose final value is not 101..999 (now 7.5 billion)
  - Exploit commutativity of + and * (now 685 million)
  - Enforce left associativity of + and * (now 559 million)
  - For 2 operators at same precedence, make first number greater than second (now 341 million)
  - Look for “wasted number” such as “+ 2 – 2” or “4 * 4 / 2” (now 329 million)

Parsing game history

- Apterous.org contains results of games played on television (>50,000 of letters, 15,000 of numbers)
- Write a Web robot to download the history
- Parse the HTML (regular expressions)
- Analyze results of letters games
  - Common words
  - How often players found longest length word
  - How many vowels contestants desired
  - Does the letter frequency match what we'd theoretically expect?
  - Player performance over time

continued

Individual student project

- Analysis of numbers game results
- How difficult is a mathematical expression?
- Difficulty level of game
  - How many contestants solved it?
  - Did the TV mathematician solve it?
  - Does it even have a solution? If not, how close?

Conclusion

- Programming projects based on Apterous games can illustrate or reinforce several CS motifs such as
  - Binary trees
  - Postfix notation and evaluation with stack
  - File I/O and regular expression parsing
  - Cryptanalysis
- Look at other “game shows”
  - *Price is Right*’s bidding game
  - What words are used as clues on *Password*?