2 Complete Search

Paul Grigoras
paul.grigoras09@imperial.ac.uk

Simple Equations UVA 11565

Given A, B, C (integers 1 … 10000), find integers x < y < z that satisfy:

$$\begin{cases} x + y + z = A \\ xyz = B \\ x^2 + y^2 + z^2 = C \end{cases}$$

2 1 2 3 6 6 14	No solution. 1 2 3

Simple Equations - solution

- Limited range. Just *three* integers. Check all?

Simple Equations - solution

- Limited range. Just *three* integers. Check all?

- xyz = B => *x* in [-10000 .. 10000]
  - 20000^3 too slow… (about 16 hours)

Simple Equations - solution

- Limited range. Just *three* integers. Check all?
- xyz = B => *x* in [-10000 .. 10000]
  - 20000^3 too slow… (about 16 hours)

- $x^2 + y^2 + z^2 = C$ => $3x^2 <= C$ => **$|x| < 34$**
- search space size = 68 x 68 x 68

Reduce Search Space

- $x + y + z = A$ => $z = A - x - y$
- Only need to search for x, y => 68 x 68

Reduce Search Space

- $x + y + z = A$ => $z = A - x - y$
- Only need to search for x, y => 68 x 68

- Distinct, x < y => 34 x 69

Reduce Search Space

- $x + y + z = A$ => z = A - x - y
- Only need to search for x, y => 68 x 68

- Distinct, x < y => 34 x 69

- Much faster: 16 hours -> 0.1 s

Reduce Search Space

- $x + y + z = A$ => $z = A - x - y$
- Only need to search for x, y => 68 x 68

- Distinct, x < y => 34 x 69

- Much faster: 16 hours -> 0.1 s -> 0.003s (C++)

JVM Startup Time….

Lessons Learned

- CS only works if the input is small
- Preliminary work can make CS feasible
- Better solutions may exist, but if your approach is just good enough, go for it!

Sum It Up [UVA 574]

Given N (<13) integers, find all distinct subsets that add up to T (<1000).

```
4 6 4 3 2 2 1 1
5 3 2 1 1
400 12 50 50 50 50 50 50 25 25 25 25 25 25
0 0
```

```
Sums of 4:
4
3+1
2+2
2+1+1
Sums of 5:
NONE
Sums of 400:
50+50+50+50+50+50+25+25+25+25
50+50+50+50+50+25+25+25+25+25+25
```

Sum It Up - Solution

- Generate all sets and check their sum?
  - 13 loops is a bit much… use **backtracking**

Sum It Up - Solution

- Generate all sets and check their sum?
  - 13 loops is a bit much… use **backtracking**

- Careful with generation order:
  1. sorted in decreasing order:
     a. 50+50 *before* 25+25+25+25
  2. all sums must be distinct

Trick - Subset Generation with Bits

```
0000 -> Empty set
0000 + 1 = 0001 -> Last element
0001 + 1 = 0010 -> Penultimate element
0010 + 1 = 0011 -> Two elements
…
1110 + 1 = 1111 -> All elements
```

Just add one, and check which bits are set!

NB! Can’t control order as we did previously.

Trick - Subset Generation with Bits

```
for (int i = 0; i <= 1 << vals.length; i++) {
  int sum = 0;
  for (int j = 0; j < vals.length; j++) {
    int mask = 1 << j;
    if ((i & mask) == mask)
      sum += vals[j];
  }
  if (sum == total) { (print values) }
}
```

Lessons Learned - Backtracking

Backtracking - general purpose CS:

```
backtrack(solution)
  if reject(solution) return
  if check(solution) print(solution)
  for (node : valid_next_nodes(solution))
     backtrack(solution + node)
```