# 2 Complete Search

Paul Grigoras
paul.grigoras09@imperial.ac.uk

# Simple Equations UVA 11565

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

$$x + y + z = A$$

$$xyz = B$$

$$x^2 + y^2 + z^2 = C$$

| | |
|---|---|
| 2<br>1 2 3<br>6 6 14 | No solution.<br>1 2 3 |

# Simple Equations - solution

- xyz = B => x, y, z in [-10000 …10000]

- Just *three* integers. Check all possible combinations?
    - too slow - this is O(n^3) with n = 10^4

- Can we reduce the search space?

# Reduce Search Space

- Rewrite equations (3 => 2 variables)

  $z = A - x - y$

  $xy(A - x - y) = B$

  $x^2 + y^2 + (A - x - y)^2 = C$

- Only need to search for x, y => O(n^2)
  - The number of tests is large (~20) => further pruning

# Simple Equations - Optimisations

- we want distinct and non-repeating pairs, hence $x < y$ (further halves execution time)
- actually since $xyz = B$

  => |x| <= B^1/3

# Simple Equations - Checks

Correct?

- We check all possibilities => yes
- *distinct* (make sure we catch that…)

Fast?

- O(n^2) + opts ⇒ yes (n ~ 60)

Time?

- About 10 mins

# Simple Equations II

Same text but input sizes change:

# 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 integers, find all distinct subsets that add up to T.*

```
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?

- Careful with generation order:
  1. sorted in decreasing order based on the numbers appearing in the sum
  2. all sums must be distinct

# Sum It Up - Checks

Correct?

- Complete search is always correct (modulo implementation bugs)

Fast?

- O(2^n) ⇒ Fast enough for (n < 12)

Time?

- About 10 mins

# 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 algorithm for complete search:

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