HKOI 2004 Final

Traffic Lights
Solution

General Problem Solving

- Defining the Problem
- Developing a Model
- Acquiring Input Data
- Developing a Solution
- Testing the Solution
- Analyzing the Results
- Implementing the Results

Solving HKOI Problems

- Understand the Problem Statement
- Observe the Hints
- Identify Possible Solutions
- Evaluate the Candidate Solutions
- Implement a Solution
- Test the Implementation
- Hand-in the Implementation

Traffic Lights

- Phoebe is on a helicopter flying over a city. Suddenly she finds that all the traffic lights in the city turn from Green to Red at the same time. She thinks that this happens very rarely. Her problem is that how long it will take until this happens again.
- She does some research on all the *N* traffic lights in the city. The traffic light *i* ($1 \leq I \leq N$) stays at green for G_i seconds, then it switches to red immediately and stays for R_i seconds. Then it switches to green immediately again. Being a good friend of her, you are asked to solve the problem.

Traffic Lights

- Phoebe is on a helicopter flying over a city. Suddenly she finds that all the traffic lights in the city turn from Green to Red at the same time. She thinks that this happens very rarely. Her problem is that how long it will take until this happens again.
- She does some research on all the N traffic lights in the city. The traffic light i ($1 \leq I \leq N$) stays at green for G_i seconds, then it switches to red immediately and stays for R_i seconds. Then it switches to green immediately again. Being a good friend of her, you are asked to solve the problem.

Simplified Traffic Lights

- All turn from green to red same time.
- How long happens again.
- N traffic lights.
- Green for G_i seconds.
- Red for R_i seconds.

What is the problem about?

- Lowest Common Multiple
- Of what?
- Max=? How many?
- The fact is: many of you demonstrated me that you know it is a problem of LCM!

How to solve LCM?

- Try all!

Algorithm 1

```
read(v[1..n])
lcm ← 0
do
    lcm ← lcm + 1
    divisible ← true
    for i ← 1..n do
        if lcm mod v[i] ≠ 0 then
            divisible ← false
while not divisible
write(lcm)
```

What is largest possible LCM?

How large is *n*?

Runtime Analysis

- Number of Innermost Loops = $2^{31} \times 20 \approx 42G$
- The current fastest CPU for PC runs at 3.2GHz

How to find LCM?

- Try all multiples of the smallest number
- Will it work better?

How to find LCM?

- Try all multiples of the largest number
- Will it work better?
- Yes!
- We have to learn what is worst case scenario.

Algorithm 2

```
read(v[1..n])
max ← v[1]
for i ← 2..n do
        if v[i] > max then
                max ← v[i]
lcm ← 0
do
        lcm ← lcm + max
        divisible ← true
        for i ← 1..n do
                if lcm mod v[i] ≠ 0 then
                        divisible ← false
while not divisible
write(lcm)
```

Algorithm 2 Worst Case

- When will there be a large LCM but small “largest input”?
- LCM = $2^4 \times 3 \times 5 \times 7 \times 11 \times 13 \times 17 \times 19 \times 23 = 1\ 784\ 742\ 960 > 2^{30}$
- LCM $\div$ 23 $\times$ 20 $\approx$ 1.5G
- How many Hz for each loop?
- Probably the algorithm is about 10 times too slow.

How to solve LCM?

4	12	8	16
2	3	2	4
	3	1	2

But, what numbers to try?

LCM = $4 \times 2 \times 3 \times 1 \times 2 = 48$

*If you have a formal procedure, your computer can do **it**. **It** is only a problem of how fast you can code **it**, and how fast your computer can run **it**.*

Algorithm 3

It is fast, but
is it correct?

```
read(v[1..n])
product ← 1
for p ← 2..200 do
    …
        divisible ← false
        for i ← 1..n
            if v[i] mod p = 0 then
                divisible ← true
                v[i] ← v[i] / p
        if divisible then
            product ← product × p
    …
write(product)
```

How large is *n*?

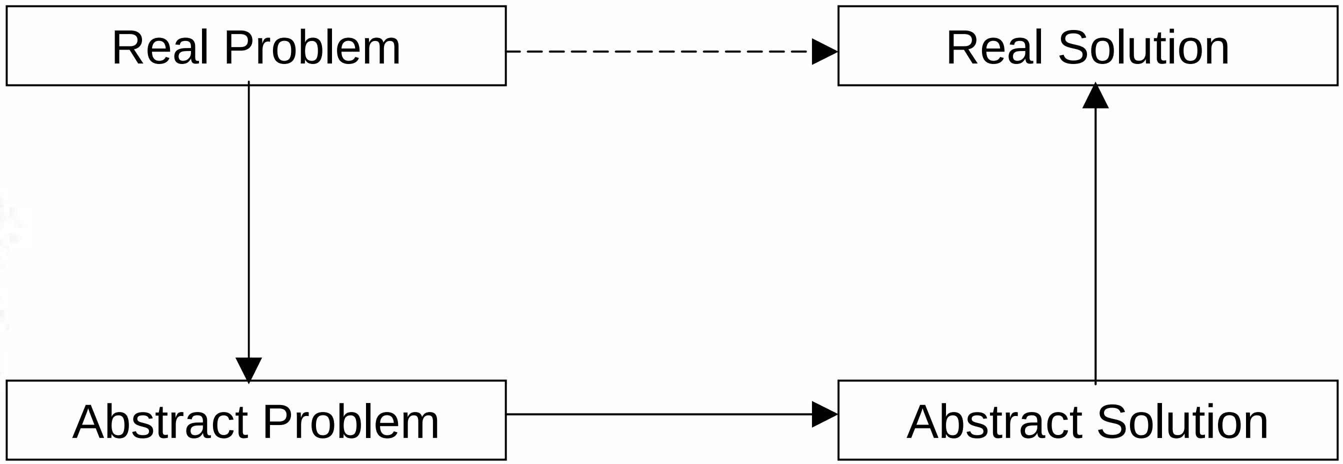
How to solve LCM?

- Why LCM is more difficult than GCD/HCF?
- LCM is as easy as GCD if …
- there are only 2 numbers
- $\mathrm{LCM}(\ldots(\mathrm{LCM}(\mathrm{LCM}(v_1, v_2), v_3),\ldots),v_n)$
- How to do LCM for big numbers?
- Utilize the asymmetry.

Algorithm 4

```
read(v[1..n])
lcm ← v[1]
for i ← 2..n do
    for j ← 1..v[i] do
        if lcm × j mod v[i] = 0 then
            lcm ← lcm × j
            break
write(lcm)
```

Problem Solving



How to find LCM?

- A fact that you may not observe:
- LCM(A, B) $\times$ GCD(A, B) = A $\times$ B
- What does this mean?
- We can use solution to GCD to solve LCM.
- How?
- Euclidean Algorithm

Euclidean Algorithm

1	96	60	1
	60	36	
1	36	24	2
	24	24	
	12		

Algorithm 5

```
read(v[1..n])
lcm ← v[1]
for i ← 2..n do
    p ← lcm
    q ← v[i]
    while p ≠ 0 and q ≠ 0 do
        r ← p
        p ← q mod p
        q ← r
    lcm ← lcm × v[i] / (p + q)
write(lcm)
```

How to find GCD?

- One more way to find GCD (For Your INTEREST Only)
- If both p and q are even, GCD(p, q) = 2 $\times$ GCD(p/2, q/2)
- If p is odd and q is even, GCD(p, q) = GCD(p, q/2)
- If both p and q are odd, GCD(p, q) = GCD(|p – q|, p) = GCD(|p – q|, q)
- So? Left as an exercise for you.