# NOI.PH Training: Extras

## Sums on a Triangle

Kevin Charles Atienza

## Contents

# 1 Introduction

This module teaches you techniques to solve a certain kind of problem involving sums and the floor function. The motivating/prototypical problem is as follows:

**Problem 1.1.** Given integers $L$, $R$, $a$, $b$ and $c$.

$$\sum_{n=L}^{R} \left\lfloor \frac{an+b}{c} \right\rfloor.$$

You may want to try to solve this problem for yourself first. Your goal is to find something polynomial in the sizes of the input numbers, that is, (poly)logarithmic in the input numbers.

A more general version is as follows:

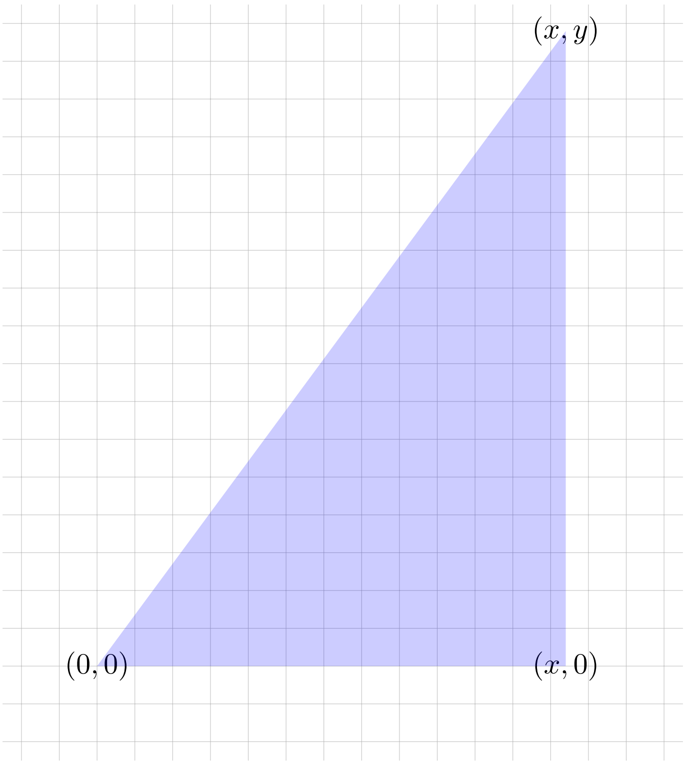
**Problem 1.2.** Given integers $L$, $R$, $a$, $b$ and $c$.

$$\sum_{n=L}^{R} \sum_{m=0}^{\left\lfloor \frac{an+b}{c} \right\rfloor} n^s m^t.$$

**Important Tip:** Don't read this in one go. At every point, stop and try to think and simplify things for yourself before moving on. (So the point of reading on is merely to confirm that you got it right.) Only proceed a bit if you get stuck after repeated attempts. (Then stop again to try again by yourself.) I believe it will be more instructive this way.

## 2 Points inside a triangle

Suppose we want to compute the number of lattice points inside or on the boundary of the triangle with vertices $\{(0,0),(x,0),(x,y)\}$. Note that $x$ and $y$ are not necessarily integers.




An easy way to do so is with Pick's theorem, but it only works when $x$ and $y$ are integers. Let's try to find a more general solution.

Let $n = \lfloor x \rfloor$. Then we can express the answer we are looking for as

$$\sum_{\substack{0 \le i \le n \\ j \ge 0 \\ j/i \le y/x}} 1.$$

Let's denote this sum by $F_{x,y}(n)$. It can be rewritten as

$$\sum_{i=0}^{n} \left( \left\lfloor \frac{iy}{x} \right\rfloor + 1 \right).$$

Now, $x$ and $y$ are not necessarily integers, but if they're rational, then $\frac{y}{x}$ is also rational. Therefore it can be expressed as a ratio of two coprime integers. Therefore, we can assume that $y$ and $x$ are coprime integers. (But we don't update $n$; we'll allow $n$ to be anything, not just $\lfloor x \rfloor$.)

You can interpret $F_{x,y}(n)$ as the number of points in the triangle $\{(0,0),(n,0),(n,\frac{ny}{x})\}$.

Computing $F_{x,y}(n)$ quickly involves three things:

- Finding a relationship between $F_{x,y}$ and $F_{x,(y \bmod x)}$.
- Finding a relationship between $F_{x,y}$ and $F_{y,x}$.
- Solving the case $F_{x,0}$.

By using these relationships, we can perform some sort of "Euclidean gcd algorithm" over the arguments $x$ and $y$. If each reduction can be done in $\mathcal{O}(1)$, then the overall algorithm runs in $\mathcal{O}(\log y)$.

**Exercise 2.1.** Assume that for $x, y > 0$, $F_{x,y}(n)$ can be computed in $\mathcal{O}(1)$ from a single call to $F_{x,(y \bmod x)}$ or $F_{y,x}$, and assume that $F_{x,0}(n)$ can be computed in $\mathcal{O}(1)$. Give an $\mathcal{O}(\log y)$ time algorithm to compute $F_{x,y}(n)$.

*Hint:* Use Euclid's gcd algorithm.

As a side note, the third thing is trivial:

**Exercise 2.2.** What is $F_{x,0}(n)$?

## 2.1 $F_{x,y}$ and $F_{x,(y \bmod x)}$

Let's find a relationship between $F_{x,y}$ and $F_{x,(y \bmod x)}$. Let $y = qx + r$ with $0 \le r < x$, so that $r = (y \bmod x)$. Then

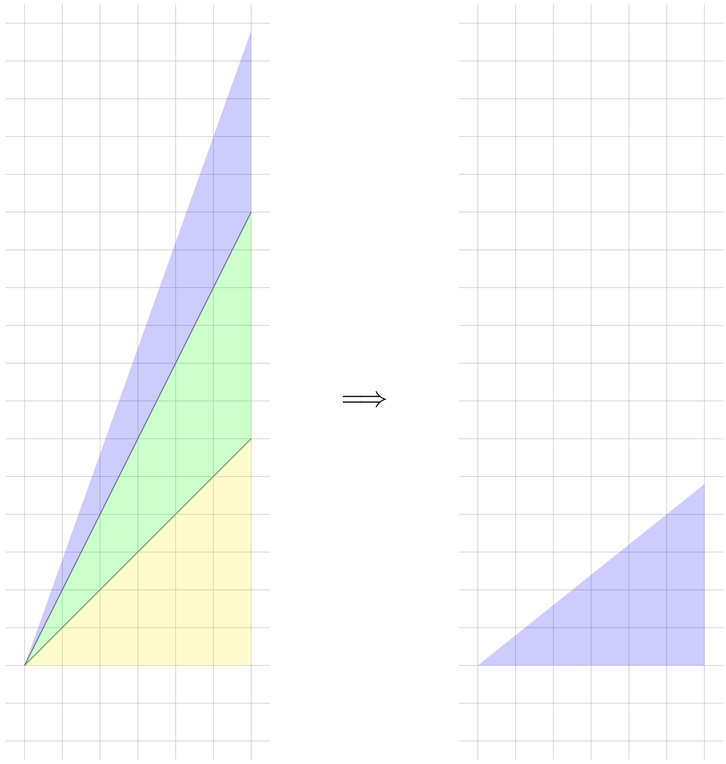
$$\begin{aligned} F_{x,y}(n) &= \sum_{i=0}^{n} \left( \left\lfloor \frac{iy}{x} \right\rfloor + 1 \right) \\ &= \sum_{i=0}^{n} \left( \left\lfloor \frac{i(qx+r)}{x} \right\rfloor + 1 \right) \\ &= \sum_{i=0}^{n} iq + \sum_{i=0}^{n} \left( \left\lfloor \frac{ir}{x} \right\rfloor + 1 \right) \\ &= q\frac{n(n+1)}{2} + F_{x,r}(n). \end{aligned}$$

This is the desired relationship between $F_{x,y}$ and $F_{x,(y \bmod x)}$ :)

**Theorem 2.3.** If $x, y > 0$ and $y = qx + r$ with $0 \le r < x$, then

$$F_{x,y}(n) = q\frac{n(n+1)}{2} + F_{x,r}(n).$$

Geometrically, you can think of this as reducing the triangle's height by $n$ units, $q$ times. Reducing the triangle by a height of $n$ is the same as removing a triangle $\{(0,0),(n,0),(n,n)\}$, and so easy to figure out how many points were removed: $\frac{n(n+1)}{2}$.



## 2.2 $F_{x,y}$ and $F_{y,x}$

Here, we assume without loss of generality that $0 < y < x$.

Let's now find a relationship between $F_{x,y}$ and $F_{y,x}$. Let's define $n' = \lfloor \frac{ny}{x} \rfloor$, which is the maximum possible value of $j$ in the sum $F_{x,y}(n)$.

Finding a symmetric relationship between $F_{x,y}$ and $F_{y,x}$ begins with the following sum:

$$(n+1)(n'+1) = \sum_{\substack{0 \le i \le n \\ 0 \le j \le n'}} 1.$$

We can decompose this into three sums, depending on the slope $j/i$:

$$(n+1)(n'+1) = \sum_{\substack{0 \le i \le n \\ 0 \le j \le n' \\ j/i \le y/x}} 1 + \sum_{\substack{0 \le i \le n \\ 0 \le j \le n' \\ j/i \ge y/x}} 1 - \sum_{\substack{0 \le i \le n \\ 0 \le j \le n' \\ j/i = y/x}} 1.$$

The nice thing about this is that the first two sums are just $F_{x,y}(n)$ and $F_{y,x}(n')$, respectively! So this gives us the desired relationship.

**Exercise 2.4.** Show that

$$F_{y,x}(n') = \sum_{\substack{0 \le i \le n \\ 0 \le j \le n' \\ j/i \ge y/x}} 1.$$

*Be careful*: $n'$ is equal to $\lfloor ny/x \rfloor$, but $n$ is not necessarily equal to $\lfloor n'x/y \rfloor$.

Note that the third sum is quite easy to compute. We assumed $x$ and $y$ are coprime, so that's just the number of multiples of $\langle x, y \rangle$ in the rectangle $\{(0,0),(n,0),(n,n'),(0,n')\}$, i.e., $\lfloor \min(n/x, n'/y) \rfloor + 1$. Combining everything, we get:

**Theorem 2.5.** If $0 < y < x$, and $n' = \lfloor ny/x \rfloor$, then

$$F_{x,y}(n) + F_{y,x}(n') = (n+1)(n'+1) + (\lfloor \min(n/x, n'/y) \rfloor + 1).$$

Geometrically, we are computing the number of points in the triangle

$$\{(0,0), (n,0), (n, ny/x)\}$$

by first computing the number of points in the rectangle

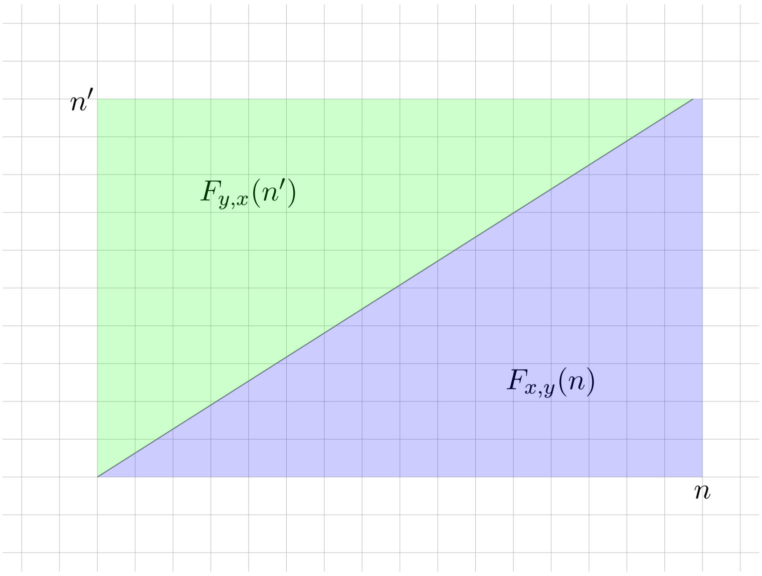
$$\{(0,0), (n,0), (n, ny/x), (0, ny/x)\}$$

(which is easy) and then subtracting from those the number of points in the triangle

$$\{(0,0), (0, ny/x), (n, ny/x)\},$$

then adjusting for double-counting along the line

$$\{(0,0), (n, ny/x)\}.$$




**Exercise 2.6.** Implement an $\mathcal{O}(\log y)$-time algorithm to compute $F_{x,y}(n)$.

**Note:** We're assuming that the input numbers fit in 64-bit integers, so we can assume that most arithmetic operations take constant time. Strictly speaking, when dealing with these kinds of problems, we should take into account the actual running times of those operations. (Read about bit complexity[a].)

[a] https://en.wikipedia.org/wiki/Computational_complexity#Bit_complexity

**Exercise 2.7.** Show that $\lfloor n/x \rfloor = \lfloor n'/y \rfloor$. (This simplifies the formula in Theorem 2.5 a bit.)

**Exercise 2.8.** Show that if $x$ and $y$ are coprime, then the number of multiples of $\langle x, y \rangle$ in the rectangle $\{(0,0),(n,0),(n,n'),(0,n')\}$ is $\lfloor \min(n/x, n'/y) \rfloor + 1$.

# 3 Sum on a triangle

Let's generalize the problem. Suppose we're now trying to find

$$F_{x,y}(s,t,n) = \sum_{\substack{0\le i\le n\\ j\ge 0\\ j/i\le y/x}} i^s j^t.$$

Here, we assume that we're working modulo some small prime to keep the numbers small.

If we define $S_k(n)$ as

$$S_k(n) = \sum_{i=0}^{n-1} i^k,$$

then we can rewrite the definition of $F_{x,y}$ as

$$F_{x,y}(s,t,n) = \sum_{0\le i\le n} i^s S_t\left(\left\lfloor \frac{iy}{x}\right\rfloor + 1\right).$$

Let's assume that $s$ and $t$ are reasonably small, so that $S_k(n)$ can be computed in $\mathcal{O}(1)$. For example, $S_0(n) = n$, $S_1(n) = \frac{n(n-1)}{2}$, $S_2(n) = \frac{n(n-1)(2n-1)}{6}$, etc. In general, each $S_k(n)$ is a polynomial on $n$ of degree $k+1$, i.e.,

$$S_k(n) = \sum_{i=0}^{k+1} \alpha_{k,i} n^i$$

for some coefficients $\alpha_{k,i}$. Since $k$ is small, all values $\alpha_{k,i}$ can just be hardcoded.

**Exercise 3.1.** Show that

$$n^{k+1} = \sum_{i=0}^{n-1}\left((i+1)^{k+1} - i^{k+1}\right) = \sum_{i=0}^{n-1}\sum_{j=0}^{k}\binom{k+1}{j} i^j = \sum_{j=0}^{k}\binom{k+1}{j} S_j(n).$$

Use this relationship to prove that $S_k(n)$ is a polynomial on $n$ of degree $k+1$.

Also, use it to derive an $\mathcal{O}(k^3)$ algorithm to compute the coefficients of $S_k(n)$.

To compute $F_{x,y}(s,t,n)$, the general pattern is similar to the above:

- Finding a relationship between $F_{x,y}$ and $F_{x,(y \bmod x)}$.
- Finding a relationship between $F_{x,y}$ and $F_{y,x}$.
- Solving the case $F_{x,0}$.

With this, a "Euclidean-like" algorithm is again possible.

And again, the third thing is trivial. If $F_{x,0}(s,t,n)$ is simply $0^t S_s(n+1)$, where $0^t = [t = 0]$.

### 3.1 $F_{x,y}$ and $F_{x,(y \bmod x)}$

Let's find a relationship between $F_{x,y}$ and $F_{x,(y \bmod x)}$. We'll proceed as before. Let $y = qx+r$ with $0 \le r < x$, so that $r = (y \bmod x)$. Then:

$$\begin{aligned}
F_{x,y}(s,t,n) &= \sum_{0\le i\le n} i^s S_t\left(\left\lfloor \frac{iy}{x} \right\rfloor + 1\right) \\
&= \sum_{0\le i\le n} i^s S_t\left(\left\lfloor \frac{i(qx+r)}{x} \right\rfloor + 1\right) \\
&= \sum_{0\le i\le n} i^s S_t\left(iq + \left\lfloor \frac{ir}{x} \right\rfloor + 1\right).
\end{aligned}$$

To proceed, we will rely on a certain property of $S_t$:

$$\begin{aligned}
S_t(a+b) &= \sum_{i=0}^{a+b-1} i^t \\
&= \sum_{i=0}^{a-1} i^t + \sum_{i=0}^{b-1} (i+a)^t \\
&= S_t(a) + \sum_{i=0}^{b-1} \sum_{j=0}^{t} \binom{t}{j} i^{t-j} a^j \\
&= S_t(a) + \sum_{j=0}^{t} \binom{t}{j} a^j \sum_{i=0}^{b-1} i^{t-j} \\
&= S_t(a) + \sum_{j=0}^{t} \binom{t}{j} a^j S_{t-j}(b).
\end{aligned}$$

Now, back to $F_{x,y}(s,t,n)$:

$$\begin{aligned}
F_{x,y}(s,t,n) &= \sum_{0\le i\le n} i^s S_t\left(iq + \left\lfloor \frac{ir}{x} \right\rfloor + 1\right) \\
&= \sum_{0\le i\le n} i^s \left(S_t(iq) + \sum_{j=0}^{t} \binom{t}{j} (iq)^j S_{t-j}\left(\left\lfloor \frac{ir}{x} \right\rfloor + 1\right)\right) \\
&= \sum_{0\le i\le n} i^s S_t(iq) + \sum_{0\le i\le n} i^s \sum_{j=0}^{t} \binom{t}{j} (iq)^j S_{t-j}\left(\left\lfloor \frac{ir}{x} \right\rfloor + 1\right) \\
&= \sum_{0\le i\le n} i^s \sum_{j=0}^{t+1} \alpha_{t,j} (iq)^j + \sum_{j=0}^{t} \binom{t}{j} q^j \sum_{0\le i\le n} i^{s+j} S_{t-j}\left(\left\lfloor \frac{ir}{x} \right\rfloor + 1\right) \\
&= \sum_{j=0}^{t+1} \alpha_{t,j} q^j S_{s+j}(n) + \sum_{j=0}^{t} \binom{t}{j} q^j F_{x,r}(s+j, t-j, n).
\end{aligned}$$

This is the relationship we want! This sum should be small since $t$ is small.

**Theorem 3.2.** If $x, y > 0$ and $y = qx + r$ with $0 \le r < x$, then

$$F_{x,y}(s,t,n) = \sum_{j=0}^{t+1} \alpha_{t,j} q^j S_{s+j}(n) + \sum_{j=0}^{t} \binom{t}{j} q^j F_{x,r}(s+j, t-j, n).$$

## 3.2 $F_{x,y}$ and $F_{y,x}$

Again, assume that $0 < y < x$.

Let's now find a relationship between $F_{x,y}$ and $F_{y,x}$. Let's define $n' = \lfloor \frac{ny}{x} \rfloor$.

Similar to before, finding a symmetric relationship between $F_{x,y}$ and $F_{y,x}$ begins with the following sum:

$$S_s(n+1)S_t(n'+1) = \sum_{\substack{0 \le i \le n \\ 0 \le j \le n'}} i^s j^t.$$

We can decompose this into three sums, depending on the slope $j/i$:

$$S_s(n+1)S_t(n'+1) = F_{x,y}(s,t,n) + F_{y,x}(t,s,n') - \sum_{\substack{0 \le i \le n \\ 0 \le j \le n' \\ j/i = y/x}} i^s j^t.$$

This gives us the desired relationship! The only thing remaining is computing the last sum quickly, but notice that it is equal to:

$$\begin{aligned}
\sum_{\substack{0 \le i \le n \\ 0 \le j \le n' \\ j/i = y/x}} i^s j^t &= \sum_{k=0}^{\lfloor \min(n/x, n'/y) \rfloor} (xk)^s (yk)^t \\
&= x^s y^t \sum_{k=0}^{\lfloor \min(n/x, n'/y) \rfloor} k^{s+t} \\
&= x^s y^t S_{s+t}(\lfloor \min(n/x, n'/y) \rfloor + 1).
\end{aligned}$$

Thus it can be computed quickly :)

Overall, we have:

**Theorem 3.3.** If $0 < y < x$, and $n' = \lfloor ny/x \rfloor$, then

$$F_{x,y}(s,t,n) + F_{y,x}(t,s,n') = S_s(n+1)S_t(n'+1) + x^s y^t S_{s+t}(\lfloor \min(n/x, n'/y) \rfloor + 1).$$

**Exercise 3.4.** Unlike in the previous section, it's not clear how to do the "Euclid-like algorithm" here because $F_{x,y}(s,t,n)$ requires multiple calls to $F_{x,(y \bmod x)}$.

Show that you can do it in $\mathcal{O}((s+t)^3 \log y)$ by defining $\mathbf{F}_{x,y}(a,n)$ to be the vector of all $F_{x,y}(s,t,n)$ all pairs $(s,t)$ such that $s+t=a$.

*Bonus:* Show how to do it in $\mathcal{O}\left((s+t)^2 \log y\right)$.

**Exercise 3.5.** Implement the algorithm described in Exercise 3.4.

## 4 Problems

**N1** [20★] Derive an $\mathcal{O}(\log c)$-time algorithm to compute the following sum, for three integers $a$, $b$, $c$ with nonzero $c$:

$$\sum_{i=L}^{R} \left\lfloor \frac{ai+b}{c} \right\rfloor.$$

**N2** [15★] Derive an algorithm to compute the number of lattice points inside or on the boundary of any triangle with *rational* coefficients, in time logarithmic in the coordinates.

**Hint:** Decompose into "axis-aligned right triangles".

**N3** [10★] Derive an algorithm to compute the number of lattice points inside or on the boundary of any *polygon* with rational coefficients, in time logarithmic in the coordinates and linear in the number of sides.

**Hint:** Triangulation.

**Hint 2:** (Optional but may simplify implementation a bit) Signed area.

**N4** [10★] Derive an $\mathcal{O}((s+t)^3 \log c)$-time algorithm to compute the following sum, for three integers $a$, $b$, $c$ with nonzero $c$:

$$\sum_{i=L}^{R} i^s \sum_{j=0}^{\left\lfloor \frac{ai+b}{c} \right\rfloor} j^t.$$

**N5** [20★] Derive an $\mathcal{O}((s+t)^3 \log c)$-time algorithm to compute the following sum, for three integers $a$, $b$, $c$ with nonzero $c$:

$$\sum_{i=L}^{R} i^s \left\lfloor \frac{ai+b}{c} \right\rfloor^t.$$

**N6** [10★] Derive an $\mathcal{O}((s+t)^3 \log c)$-time algorithm to compute the following sum, for three integers $a$, $b$, $c$ with nonzero $c$:

$$\sum_{i=L}^{R} i^s \left((ai+b) \bmod c\right)^t.$$

**N7** [15★] Describe a fast solution to compute the following sum quickly for some integer $n$:

$$\sum_{i=L}^{R} \lfloor i\sqrt{n} \rfloor.$$

**Hint:** Continued Fraction

**Hint:** (For an alternative, but related, solution) Rational Approximation

**N8** [15★] Describe a fast solution to compute the following sum quickly for some *real* $r$:

$$\sum_{i=L}^{R} \lfloor ir \rfloor.$$

For Project Euler problems, your program must be able to compute the answer) in $< 10$ seconds. It must also be sufficiently general and be able to answer the problem for inputs of similar magnitude (within a 10 second time limit).

**C1** [30★] **Pencils of rays:** https://projecteuler.net/problem=372

**C2** [50★] **Chef has a Spaghetti Garden:** https://www.codechef.com/problems/CHEFSPAG

**C3** [30★] **KM xor M:** https://codebreaker.xyz/problem/kmxorm

**C4** [80★] **Lattice Quadrilaterals:** https://projecteuler.net/problem=453