Final Seminar

Y. Miyazaki

2020/03/06

Contents

- About Study (B3 ~ M2)
- About Job
- About others

My Timeline

	Study	Job	Others
B3	Lecture "Gravitational Waves"		
B4	Student experiment		
M1	Toy interferometer	Job hunting	Competitive Programming
	PEM injection test		
	Lock loss study		
	Reference cavity experiment		
	Caltech		
M2	Coupled WFS		
company employee		Will work	

Study Timeline (B3~B4)

B3 Sep. — Lecture "Gravitational Waves" by Kawamura-san

- got interested in GW & GW detectors
- Contents:
  - Saulson, chap 2 & 3
  - paper about GW detectors (link)
  - GW150914 paper
- At beginning of each lecture, a "quick game" was played.

B4 Apr. — Student experiment "lock loss study of iKAGRA" (report)

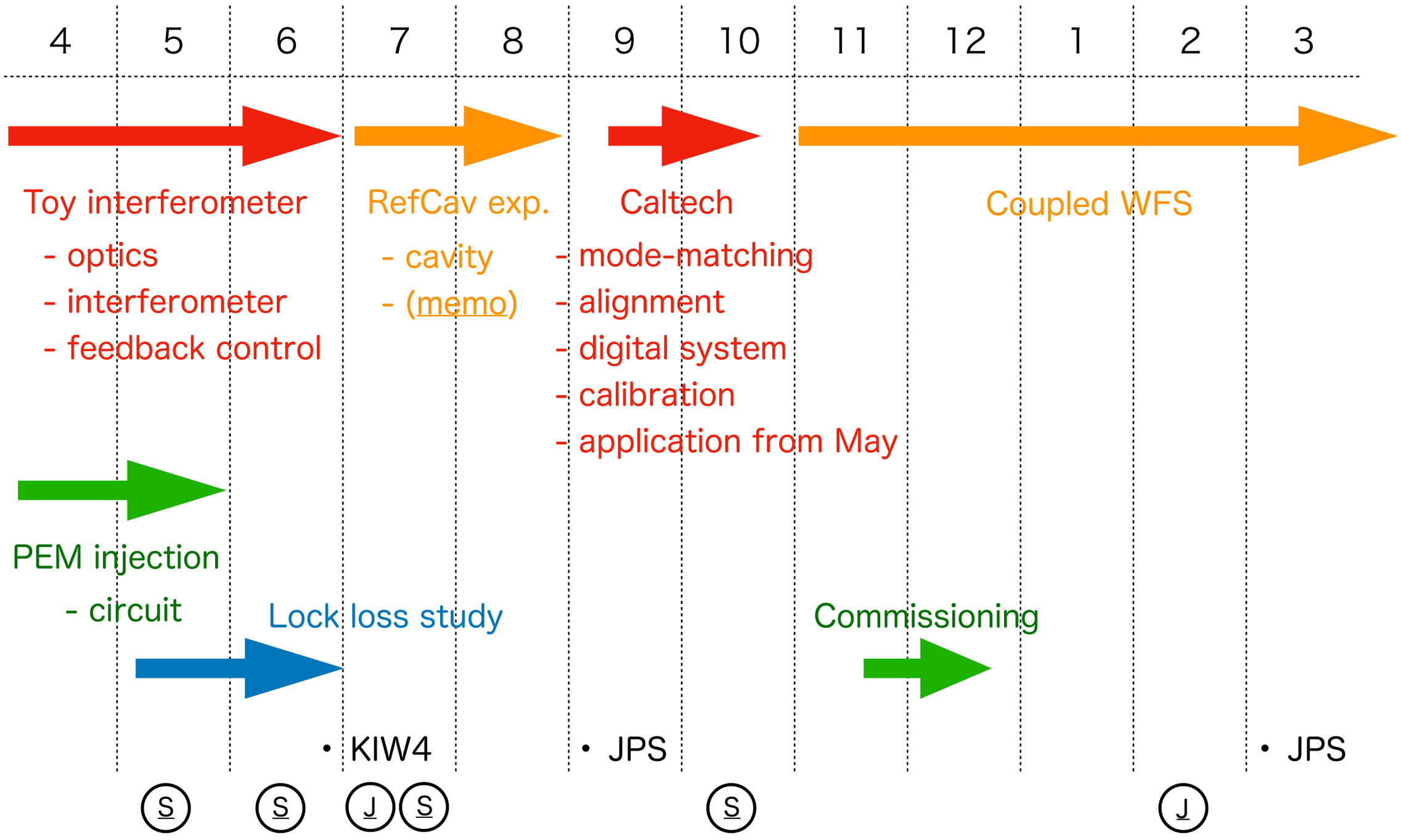
- learned & first visited KAGRA
- learned MATLAB

Jul. — Applied Ando lab.

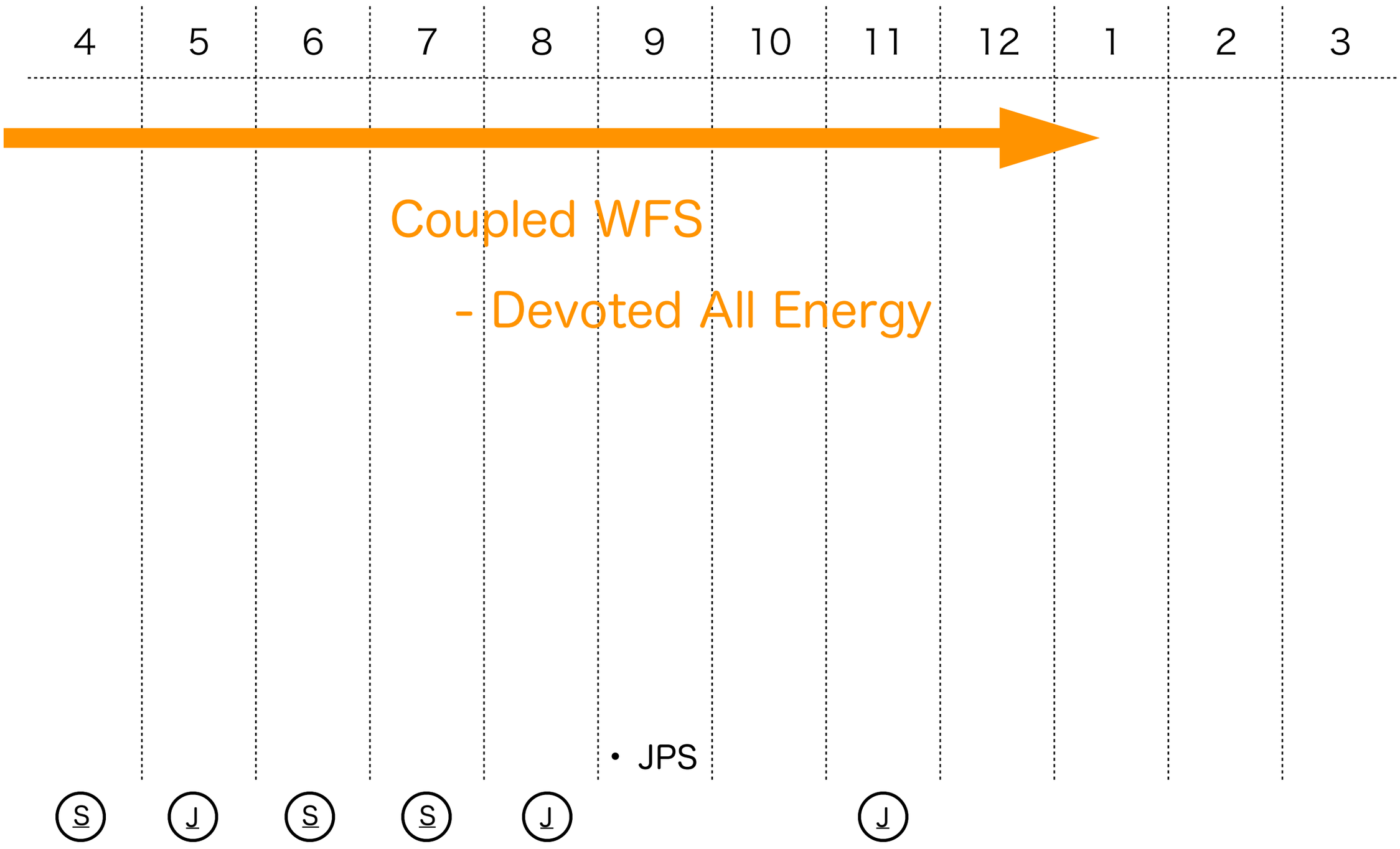
Dec. — Detchar wrapper (note)

- developed MATLAB wrapper of command-line-interface software

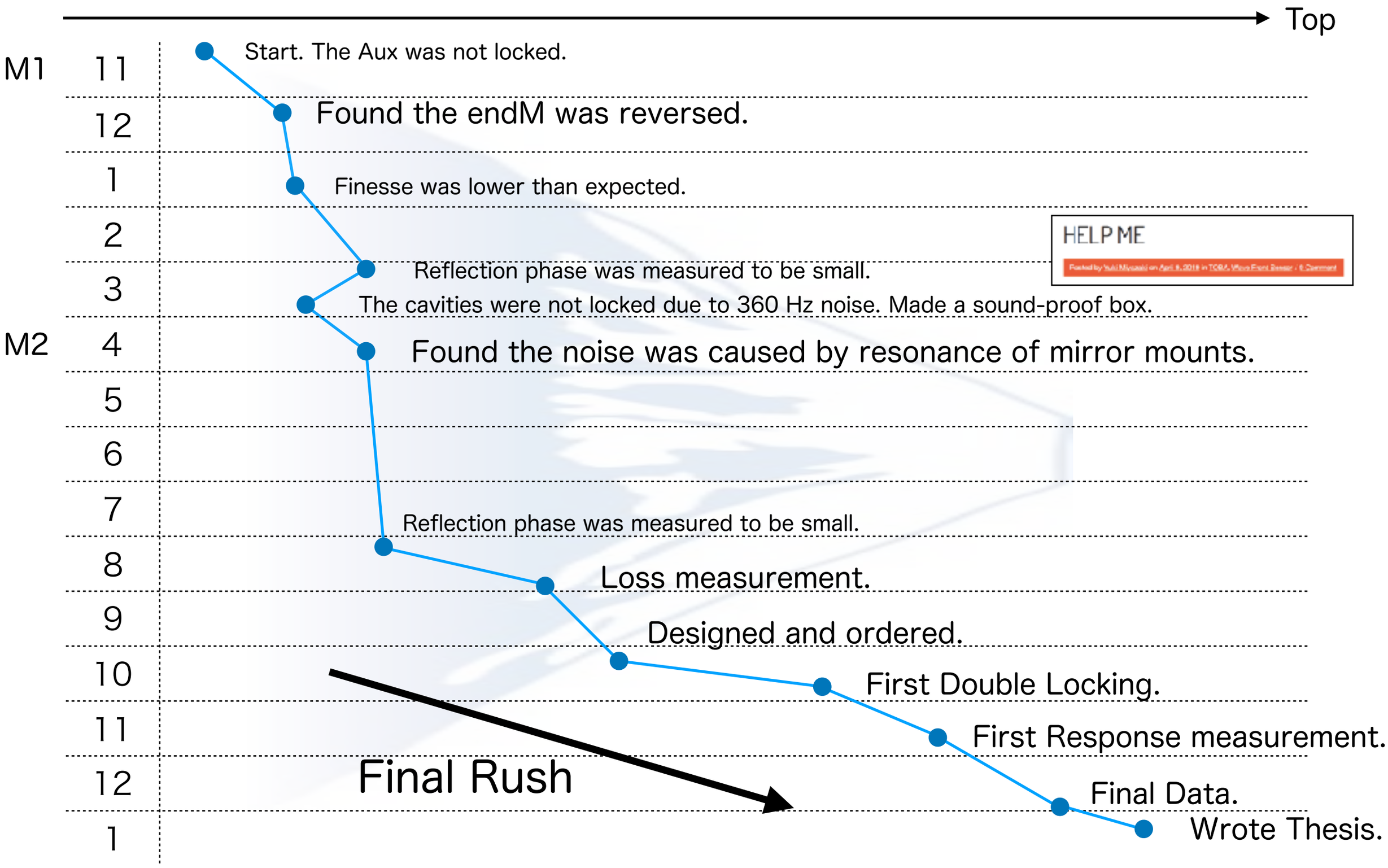
Study Timeline (M1)



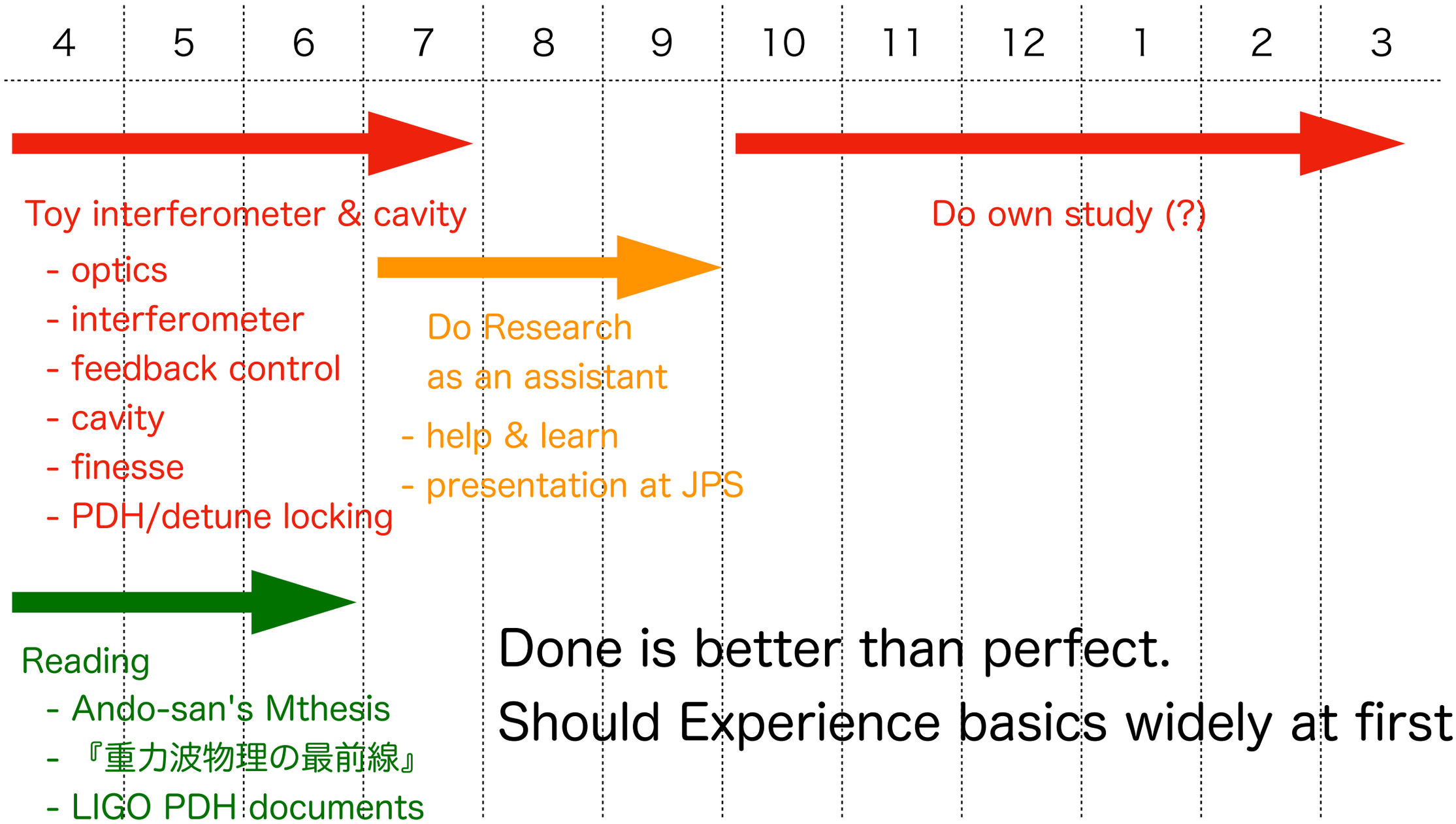
Study Timeline (M2)



Climbing of CWFS experiment



Proposal: M1 educational plan



Done is better than perfect.

Should Experience basics widely at first.

Conpetitive Programming

- What's conpetitive programming (競技プログラミング、競プロ)？

Example:

Problem statement:

You are given a three-digit positive integer N.
Determine whether N is a palindromic number. (回文数)
Here, a palindromic number is an integer that reads the same backward as forward in decimal notation.

Constrains:

100 <= N <= 999.
N is an integer.

Time Limit:

2sec. (~10^8 calculations)

Input:

Input is given from Standard Input.
(ex) 575

Output:

If N is a palindromic number, print 'Yes'; otherwise, print 'No'.
(ex) Yes

Conpetitive Programming

Answer (C++)

```cpp
int main(void) {
    string N;
    cin >> N;
    if (N[0] == N[2])
        cout << "Yes" << endl;
    else
        cout << "No" << endl;
    return 0;
}
```

This problem is too easy for you.

Famous problems; Ants

Problem statement:

An army of ants walk on a horizontal pole of length L cm, each with a constant speed of 1 cm/s. When a walking ant reaches an end of the pole, it immediatelly falls off it. When two ants meet they turn back and start walking in opposite directions. We know the original positions of ants on the pole $\{x_i\}$, unfortunately, we do not know the directions in which the ants are walking. Your task is to compute the earliest and the latest possible times needed for all ants to fall off the pole.

Constrains:

$1 <= L <= 10^6$
$1 <= N <= 10^6$
$0 <= x_i <= L$

Time Limit:

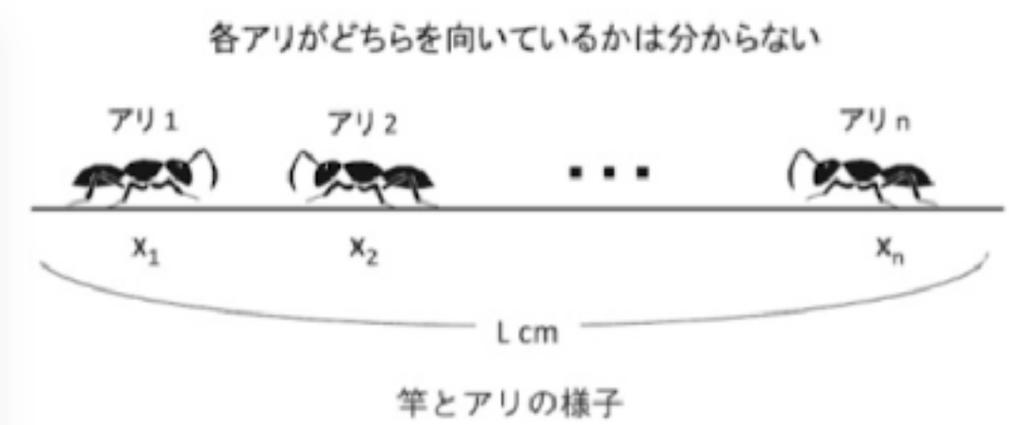
2sec. (~10^8 calculations)

Sample Input:

L = 10
N = 3
x = (2, 6, 7)

Sample Output:

min = 4
max = 8



竿とアリの様子

Ants; bad solution

Try all posibility of ants' direction

- direction of each ant: 2 ways → all posibility: 2^N ways
- when N=10^6, it takes 2^(10^6) times calculations (too many)

Constrains:

$1 <= L <= 10^6$
$1 <= N <= 10^6$
$0 <= x_i <= L$

Time Limit:

2sec. (~10^8 calculations)

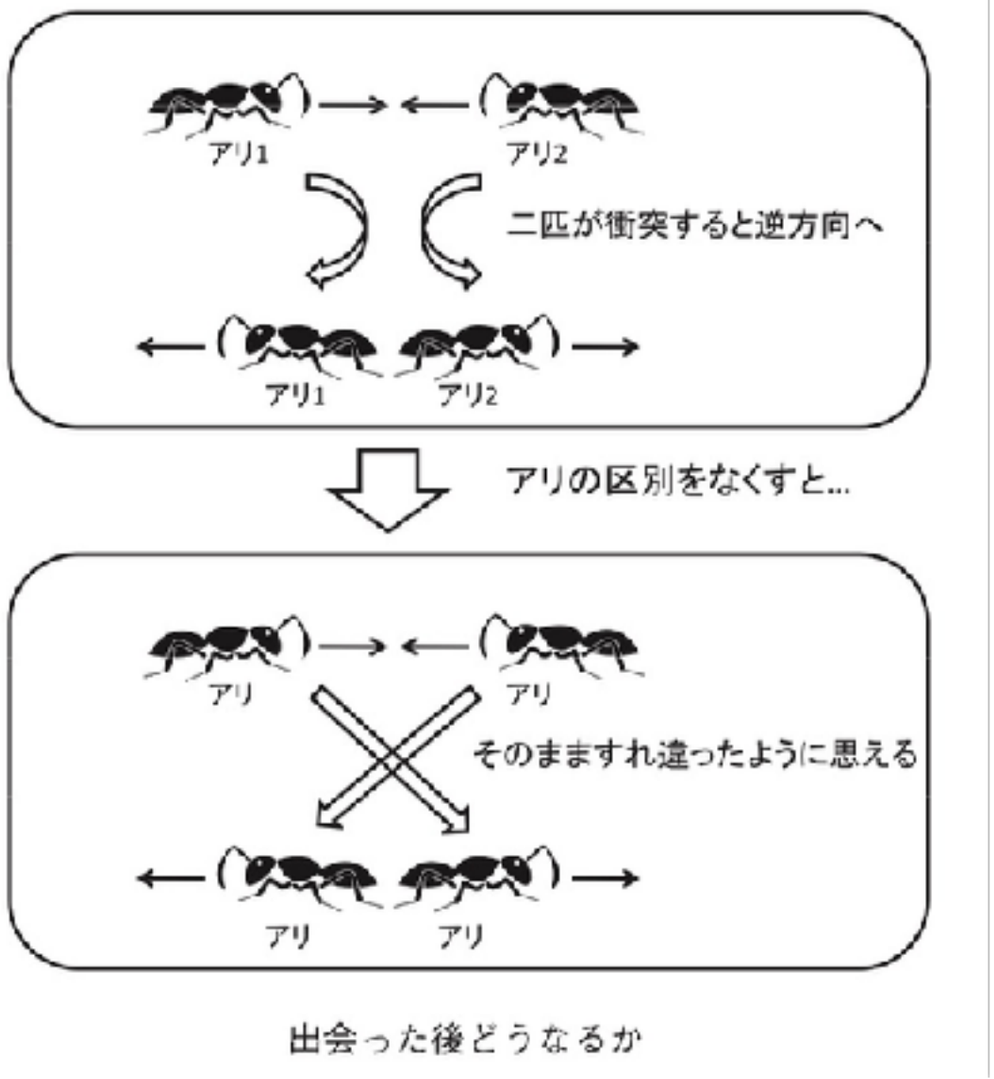
How to solve the problem effeciently?

Ants; good solution

- It doesn't matter when we think ants can pass each other.

- Then,
  t_{min} = min{drop time}
  = min{distance to the end}
  t_{max} = max{drop time}
  = max{distance to the end}

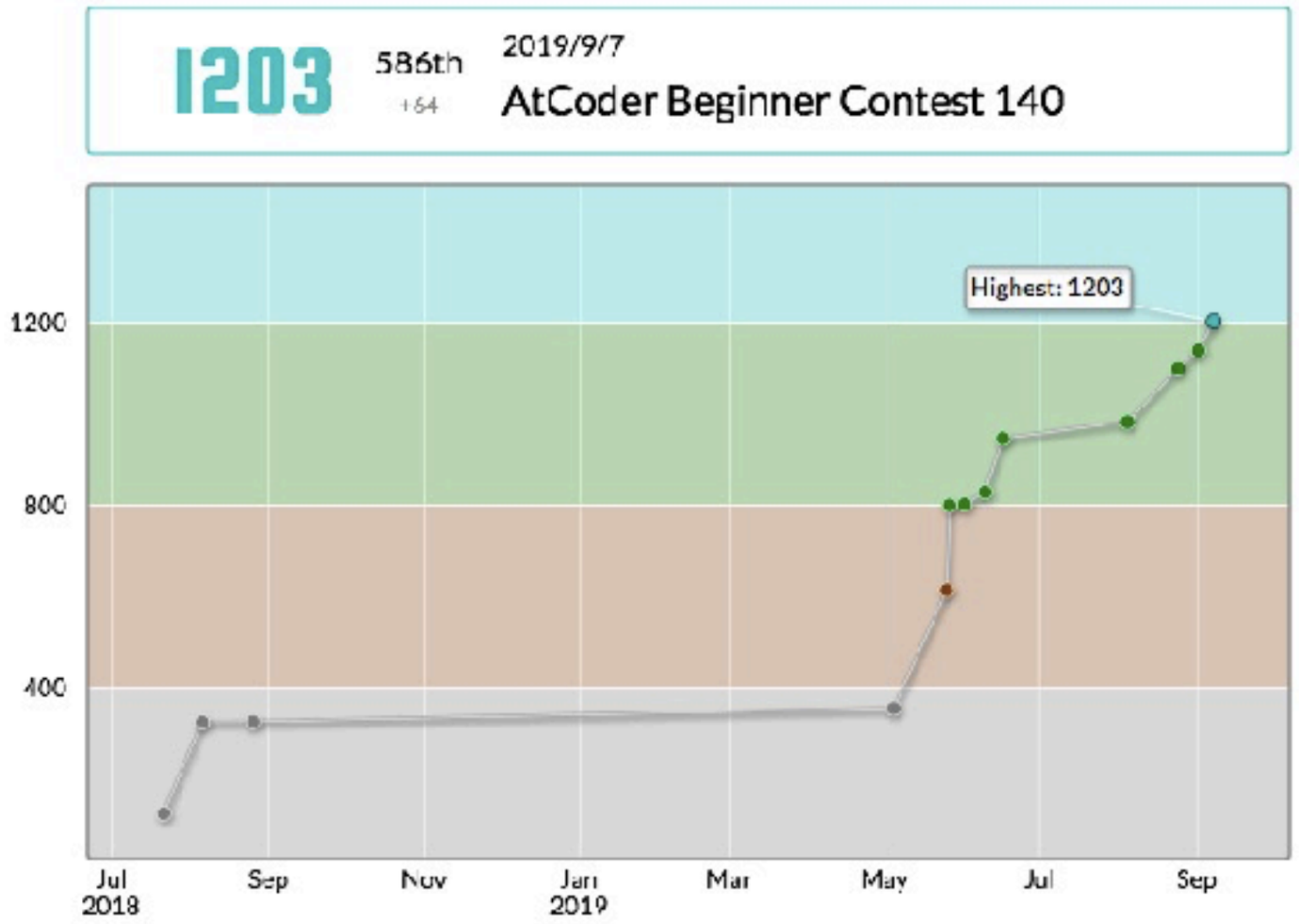
- Then, computational complexity
  = O(N)



出会った後どうなるか

Appealing points of 競プロ

- Inprove programming skills
- Learn algorithm
- Fun!



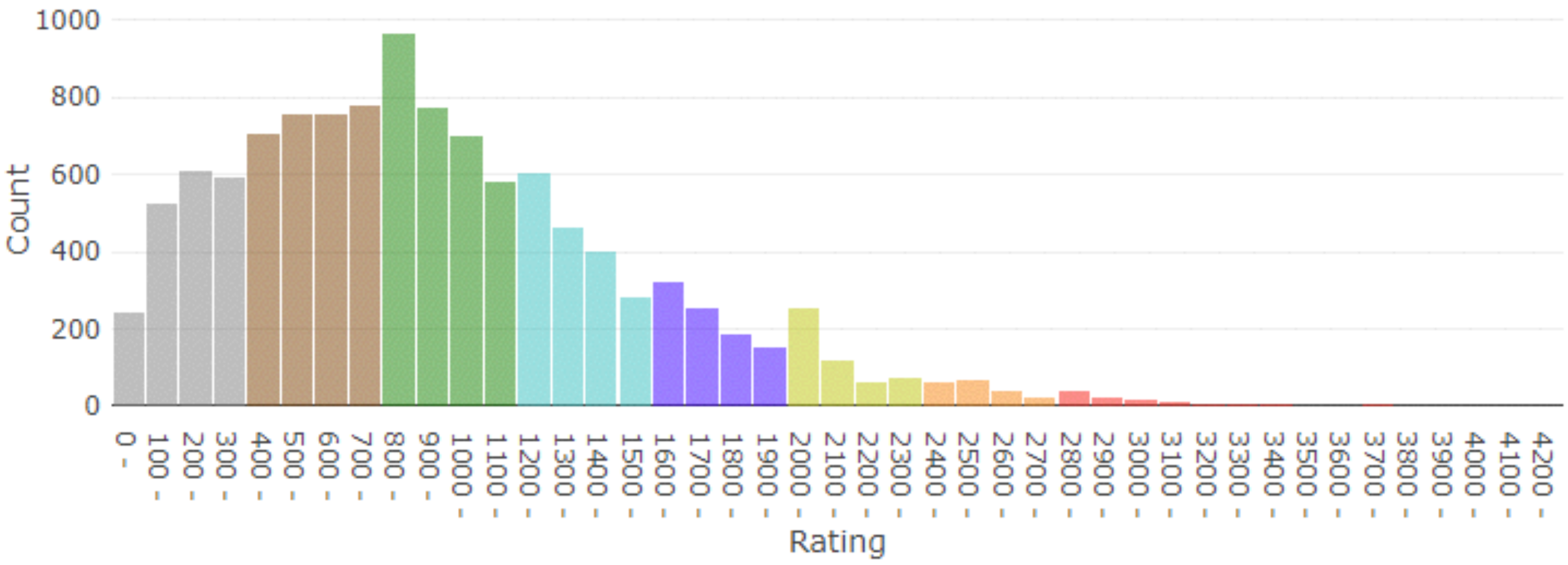
Start 競プロ！

- Search "AtCoder"
- Regist from here
- Python, C++, C, (more than 50 languages)

End.

Rating distrubution

only including players participating more than 10 contests



@chokudai