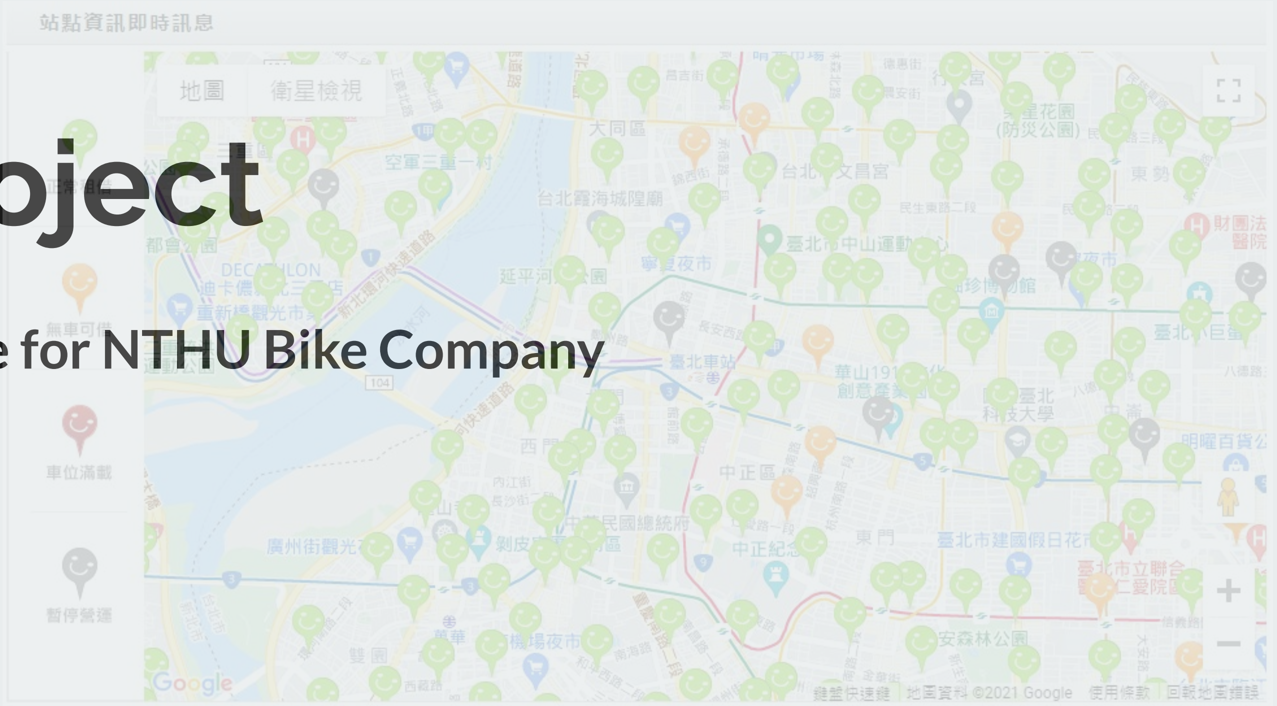


Final Project

Maximize Revenue for NTHU Bike Company

2022 Fall,
Data Structure

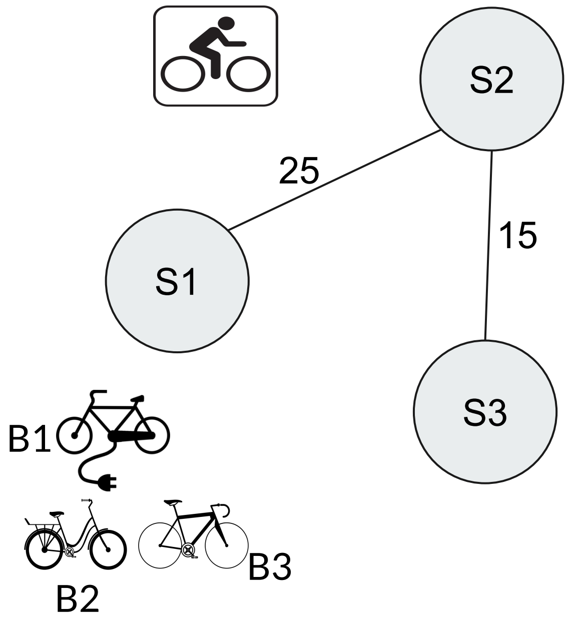
Introduction

rate = distance / time
given rate = 1

User rental request

U1 B1 23 89 S1 S3

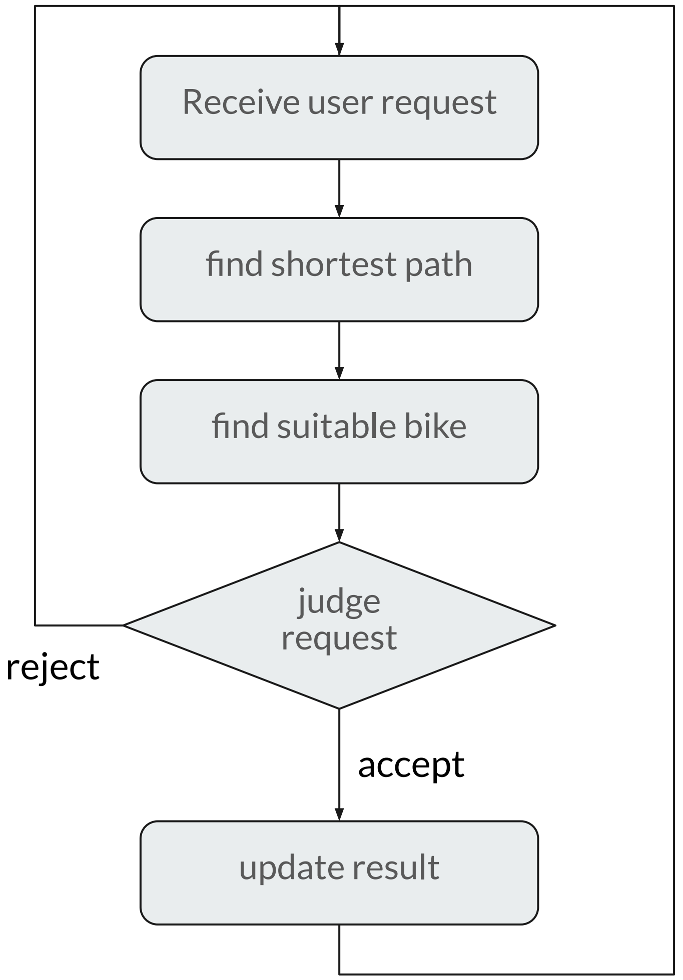
I am the user U1
I will arrive station S1 at time 23
I want to arrive station S3 before time 89
I want to rent bike type B1



Rental_fee
= rental_price * distance

Process

- Judge request
    - If suitable bike exist
    - Can reach before end time
- Update result
    - update bike status
        - Station
        - Rental_Count
        - Rental_Price
    - update revenue



Bike

Each bike has its Bike_ID, along with Bike_Type, Rental Count and Rental Price.

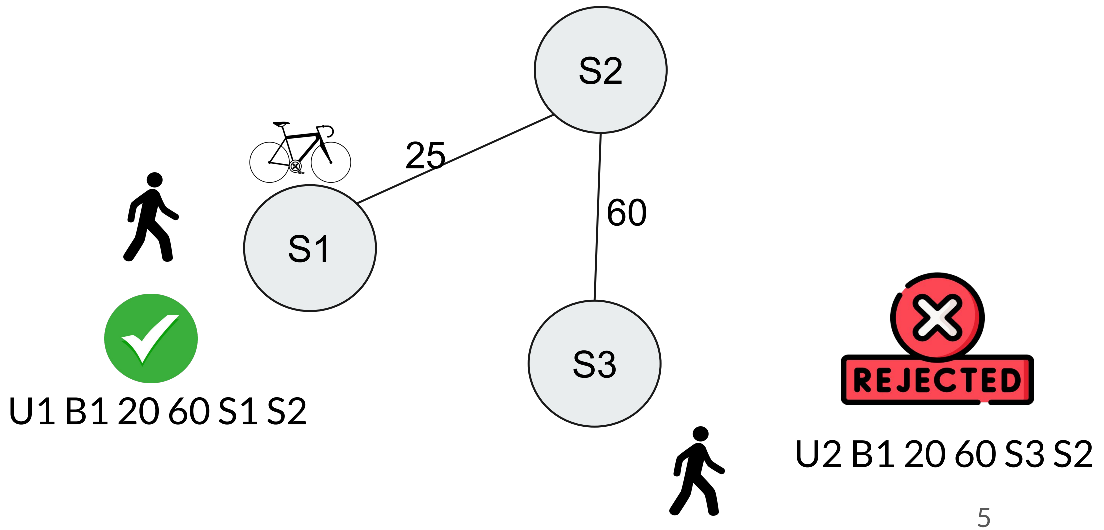
Bike price is related to the bike rental count

If the bike reaches maximum rental count, it will be retired.

Use free bike transfer service to maximize the revenue in advanced version.

Basic

I. Free bike transfer service is forbidden.
II. If there is no bike available at the **Start_time**, the rental request should be rejected with no charge. (User cannot wait the bike)
III. More detailed refer to spec



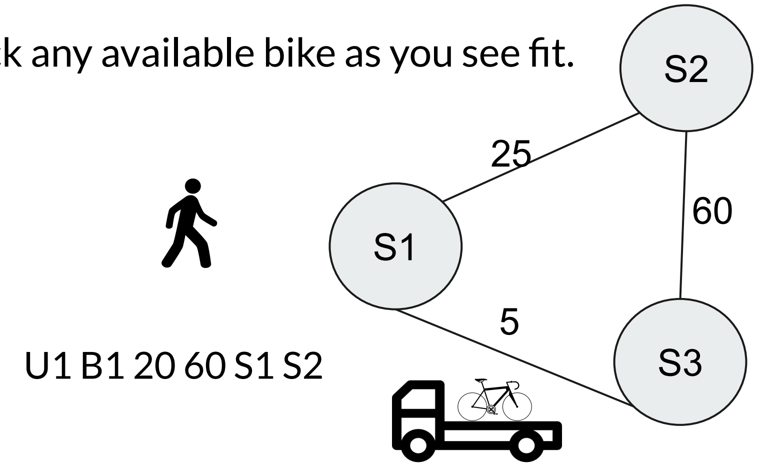
Advanced

I. With the same user requests, the final total revenue should be larger than the basic version.

II. Free bike transfer service is allowed.

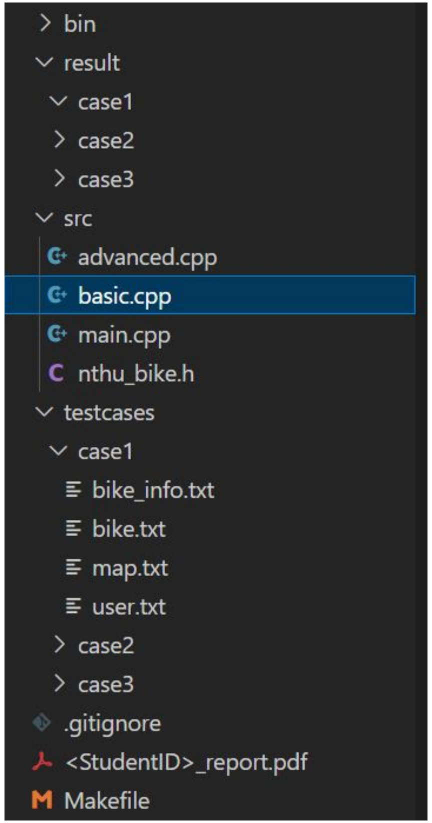
III. If no bike is available at **Start_time**, users could wait for bikes. Make sure that the user still needs to arrive at **End_Point** before **End_Time**.

IV. You could reject the request or pick any available bike as you see fit.



Submission

- Directory structure
  - <StudentID>_proj/
    - ↳ bin/
    - ↳ src/
      - ↳ main.cpp
      - ↳ basic.cpp
      - ↳ advanced.cpp
      - ↳ ...
    - ↳ testcases/
      - ↳ case1/
      - ↳ ...
    - ↳ result
      - ↳ case1/
      - ↳ ...
    - ↳ Makefile
    - ↳ <StudentID>_report.pdf



```
> bin
∨ result
  ∨ case1
  > case2
  > case3
∨ src
  advanced.cpp
  basic.cpp
  main.cpp
  nthu_bike.h
∨ testcases
  ∨ case1
    bike_info.txt
    bike.txt
    map.txt
    user.txt
  > case2
  > case3
.gitignore
<StudentID>_report.pdf
Makefile
```

Command

- Compiling command:
  - g++ -g -std=c++11 -o ./bin/main ./src/*.cpp
- Execution command:
  - Basic: ./bin/main case<i> basic
  - Advanced: ./bin/main case<i> advanced
- Or using makefile
  - make

Environment

- OS: Ubuntu 20.04
- Setup: Tutorial

Grading

- Correctness (60%)
  - There are 3 open test cases and 7 hidden test cases.
  - 6 points for each test case. TA will run the basic version to test your result.
- Performance (20%)
  - It is graded by the revenue of your program on 3 open test cases and 7 hidden test cases. Incorrect results will not get any points.
  - 2 points for each test case based on your perfomance. TA will run the advanced version to test your result.
- Report(10%)
- Demo(10%)

Q & A