Summarization of A3

lots of question has been asked about A3, it is really annoying, I try to summarize the process

Normal condition

Under normal input, every thing works fine.

Data flow:

creating processes

First thing after the Driver start, it need to create some child processes, they are:

- rgen(generate valid input for A1's program)
- A1(receive output from rgen and redirected output to A2)
- A2(receive output from A2 and receive input from user's command)
- Driver or CommandReader has to read input of the user from the terminal and pass it to A2, and A2 need to output the result to the terminal.

pipes between process:

- rgen$\rightarrow$ A1:

This pipe will output valid command for A1, several valid command are:

1. add "A" (0,0) (0,1) (1,0)
2. gg
3. rm "A"

Note that according to A1's pdf, rm command need to have the street name as parameter, so rgen need to store all the command it passed to A1, so that, it can remove the street.

Before add command, A1 need to issue enough rm command to clear A1's storage.

- A1$\rightarrow$ A2

You need to change A1's code, because A2 need a command "V", which is simple, A1 gives the max id of the vertex. and pass "E" command to A2 when A1 receive the command "gg" from rgen

several valid command are:

1. V 13
2. E {<2,3>,<3,5>,<7,3>}

- Driver or CommandReader $\rightarrow$ A2

Every command from A1$\rightarrow$ A2 will be print on terminal, when user see the output. User could input only one valid command "s", and passed it to A2. Also when driver receive the EOF, it will terminate every child process and terminate itself, the program exit.

valid command are:

1. s 2 5

Error handles

Driver's parameter error

Driver has at most 4 kinds of parameters, All are optional:

- -s k — where k is an integer ≥ 2. The number of streets should be a random integer in [2, k]. If this option is not specified, you should use a default of k = 10; that is, the number of streets should be a random integer in [2, 10].
- -n k — where k is an integer ≥ 1. The number of line-segments in each street should be a random integer in [1, k]. Default: k = 5.
- -l k — where k is an integer ≥ 5. Your process should wait a random number w seconds, where w is in [5, k] before generating the next (random) input. Default: k = 5.
- -c k — where k is an integer ≥ 1. Your process should generate (x, y) coordinates such that every x and y value is in the range [−k, k]. For example, if k = 15, all of your coordinate values should be integers between −15 and 15. Default: k = 20.

All of the parameter are used by rgen, as you noticed, all the parameter has a valid range, when driver received those parameter, it should check whether they are valid for rgen, before pass them to rgen.

Handling the error:

If parameter is not valid, the program should The driver should **print an error** and **exit**. All processes should terminate. (Actually for this step, no child process is created yet, so, it just terminate Driver)

Cannot generate valid streets error

This error will appear in the rgen, if the range is small and there are too many street, it may never generate a valid streets. (Note that I use street**s** instead of street, may be it is easy to generate first street but it is impossible to generate second).

Handling the error:

To prevent this error, You should disallow this from happening by limiting the number of tries. That is, if your random generator fails to generate a valid specification for a continuous A number of attempts, it should exit() with an error message reported on **stderr**.

Note that **rgen** will exit **NOT** driver, so if there is a valid output of rgen before this error, it should be possible you can input "s" command and the program should gives a shortest path. If there is no vaild output, then nothing will show in the terminal, which means it just wait for user to give and EOF command, all other command is pointless.

The exit code could be non-zero, if you prefer others.

For example:

If you try to generate $n \in [1,5]$ First time the program wants to generate 3 streets, and it failed after 25 attempt. It will repick the number of the streets, the second time rgen wants to generate 4 streets, luckily, it successfully generate a street set contains 4 valid streets, it will output to A1 with the gg command in the end. If the rgen is so unlucky, after 25 attempts there is no valid streets set, it will exit. This means it has failed at least $25 * 25 = 625$ times, because for each attempt to generate a valid streets set, the fail time could be more than 25 Why is that?

- one valid set has 3 streets, to generate street A it failed 15 times, to generate street B it failed 5 times,to generate street C it failed 20 times, it failed 40 times in total)
- one invalid set has 3 streets, to generate street A it failed 15 times, to generate street B it failed 5 times,to generate street C it failed 25 times, it failed 45 times in total) and next 24

attempts have same result, total fail will be (45*25=1125)

Undetermined behavior

1. There is one undetermined behavior: program will have to read user's command all the time, for a special moment, user and A1 gives the input to the A2, which is a race case, and I am not sure whether different input buffer will handle this, and it could cause undetermined behavior. It is really unprofessional to write this code. The output could be different from what you expected because the graph changes. This case is very difficult to debug.
2. All the input should ended with "Enter" or "\n", the default setting of C++ is that when see a \n, it will clear the input buffer and push it to the program's var, but due to the A2's result, it is weird. Without clear the input buffer, it could read the buffer by fgets() and it can't read EOF from the file, which cause a infinity loop. And it could cause more tricky problem when you input in the terminal by hand instead of issuing <test01.in. Yes, I am struggling on this problem, it almost shred all my knowledge about stdin and basic skill about interaction with computer.