Computational Geometry **Module 4-6** **NAME:**
PROBLEM SET 2

ScoringRubric:

Points Earned on Problem	Criteria
4	Excellent. All work is correct. Neatly written and legible. Clearly presented and worked through without a fault.
3	Good. Only slight error in work. Neatly written and legible. Clearly presented and clear understanding of the concepts. Only possesses mistakes in calculations.
2	Fair. Errors in work. Illegible and not written neatly. Work is not presented in a way that shows understanding of the concepts. Calculation is not clear or correct.
1	Poor. Work is filled with errors, work is not clear and concepts are confused. Needs work to bring their work up to this courses expectations.
0	Immediate action needs to be taken to correct your understanding and/or effort in this class.

Directions: Each problem from the Problem Set will be worked out in its entirety. Your solution method may vary and will not be used against you if you get the right answer. You are expected to honor the following rules:

- Do not read any homework solutions or "sample solutions", including those produced by students (except yourself), instructors, or TAs, related to this or any other course on Computational Geometry or a closely related topic. Some of the homework questions I'll use this semester have been used before by me or other teachers at other institutions, and you're on your honor not to consult the solutions.
- Do not copy solutions or work from classmates or any work you've seen online. The purpose of the problems is to grow your understanding of the subject manner and refine your skills at writing and implementing algorithms. • Do not give solutions to each other. Learning is not done by copying solutions.

You may consult any books, papers, web pages, or other inanimate information sources you please, so long as they're not other people's homework solutions or sample solutions. If you use information from such a source, please cite it. I don't believe that the solutions to many of the problems I'll give are available in easy-to-find references, but if you do find one, it's fair game as long as you cite it. After all, a good literature search should be rewarded. You may not, however, let your classmates know about your discovery or sources until the deadline has passed.

You are welcome to ask me to clarify anything in the readings, the lectures, or this homework.

I prefer typeset solutions, on paper. I will accept handwritten solutions with the proviso that I reserve the right to give a zero to a solution if there is any word or variable I can't read clearly. (Sorry, I just don't have time to infer from context whether that vague squiggle is an i or a j.) Any such decision is final and irrevocable. By contrast, typeset solutions reserve the right to argue with my grading after the fact.

On the other hand, handwritten figures will be gratefully accepted. Anything to encourage you to draw figures! A simple figure can often mean the difference between my spending five minutes or two hours grading an answer. Thank you in advance for every figure you draw.

Computational Geometry **Module 4-6** **NAME:** _____
PROBLEM SET 2

---

1. (Point Location) Draw the graph of the search structure D for the set of segments depicted in Figure 1, for some insertion order of the segments.

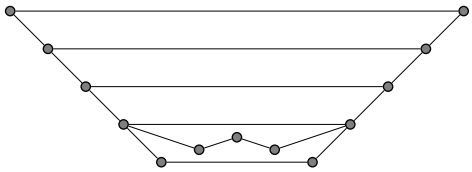


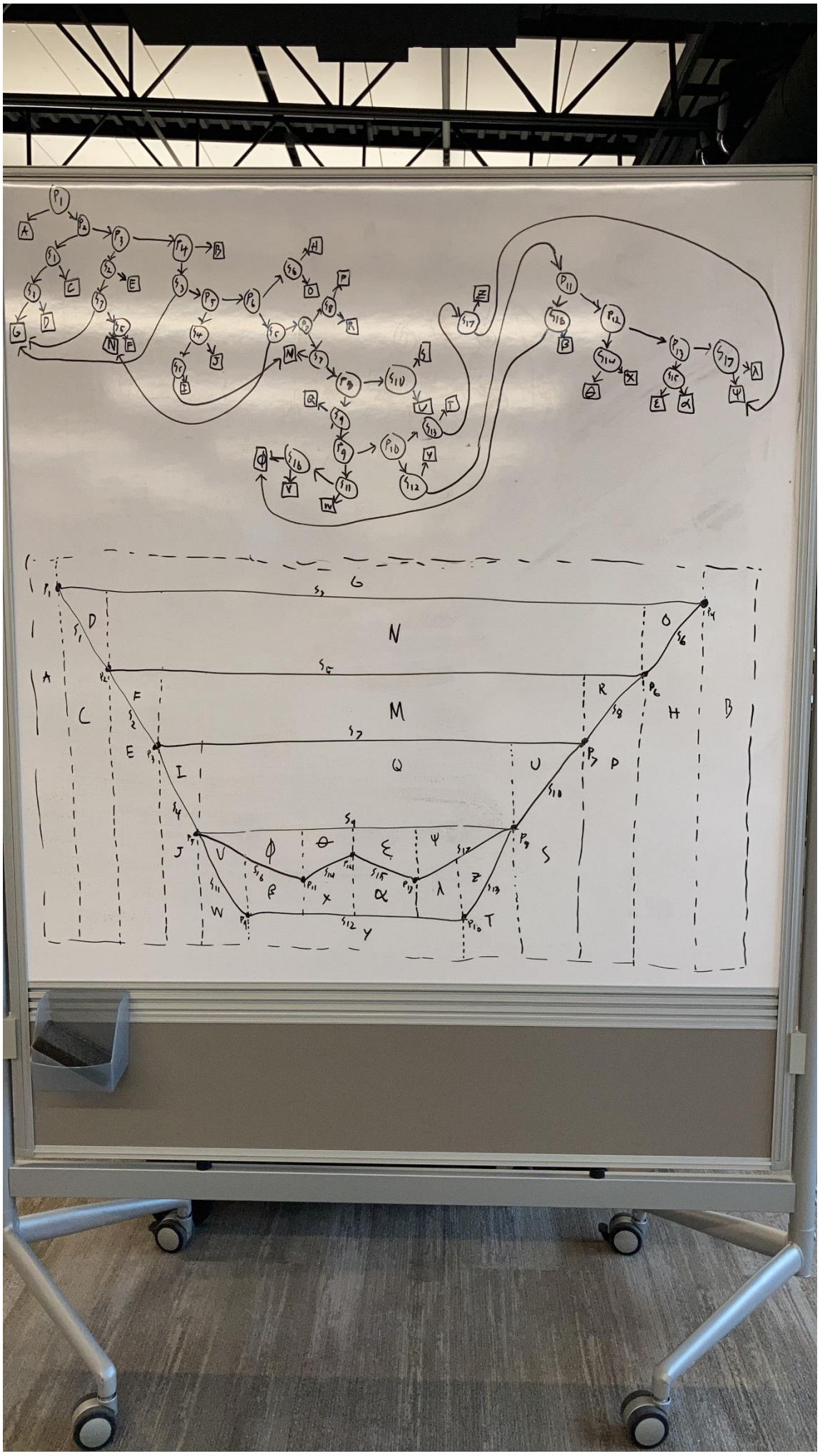
Figure 1: Collection of segments.

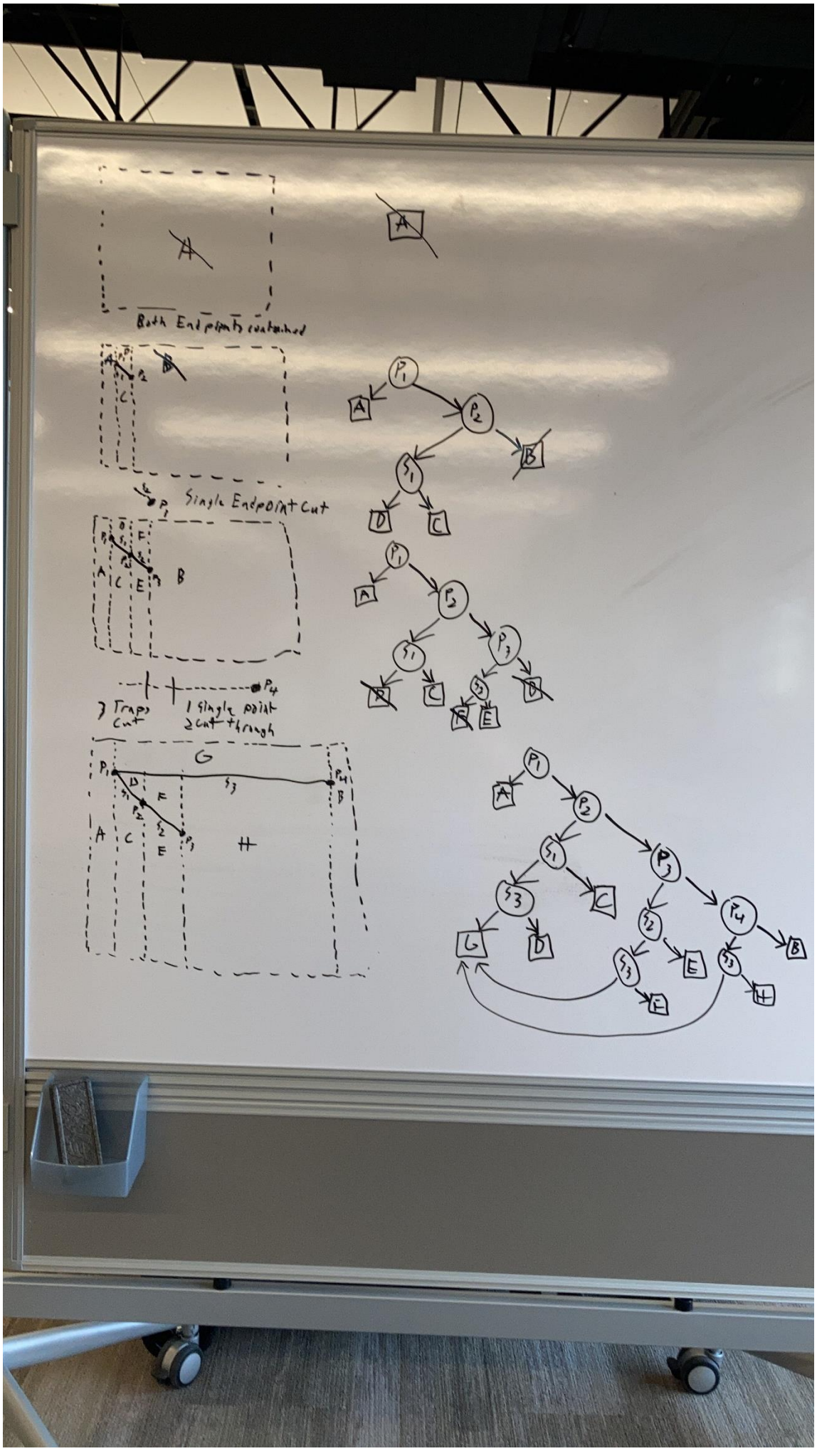
This pdf from CMU helped with understanding the iterative generation:
https://www.cs.umd.edu/class/spring2020/cmsc754/Lects/lect09-pt-loc.pdf
They broke down the iteration into three main rules illustrated in fig 4. The only difference for me was their examples go right to left.

I thought about writing a script to illustrate this but only briefly. In order to demonstrate the creation of the structure I included some initial iteration steps along with the final structure. I did not add segments randomly which naturally makes the structure less ideal. I included the first few iterations to demo a limited understanding because I think there is a good chance my final structure is not valid.

The cool part for me with this assignment was seeing the search working when checking the structure during iterations. Searching for a point in epsilon actually showed a mistake I made.

Something to note, I maintained the left right relationship by making it counterclockwise for when I swing the child nodes to the side.




Both Endpoints contained
Single Endpoint Cut
Trap Cut
1 single point 2 cut through