PCB Climber

Mackenzie Norman
mnor2@pdx.edu

February 5, 2025

1 Project Topic

For my term project I hope to implement an early genetic algorithm used to generate rudimentary layouts for Printed Circuit Boards. This was described in the paper by Jain (see citations)

Note: since the paper is paywalled I summarized it in GA_breakdown.pdf

My reason for wanting to do this is summarized well in this qoute: (I also just think it seems fun, I implemented this in python but it was pretty rudimentary and very slow) "The rapid progress in PCB capability has been driven by a few factors, including more capable components and improved manufacturing techniques. Nevertheless, while the components have gotten smaller and faster over decades, the process of designing a PCB has not changed significantly. Until the late 1980's, PCB designers would labor over large schematics and models for weeks if not months, trying to place components and calculate routing paths. Today, pen and paper have been replaced with highly functional Electronic Design Automation (EDA) software. Even still, most component placement and routing done today still relies on the experience and skill of the designer, just as it did 50 years ago. Through the age of automation, PCB component placement and routing remain in the technological stone age" [Crocker]

PCB Layout - done optimally - is pretty hard. Actually nearly impossible, especially as layouts and components get smaller and routing gets more complex.

2 Project Vision

When optimizing an arbitrary placement there are three basic constraints that must be optimized for.

Placement area: either in the form of a bounding box or (depending on computing expense) the convex-hull of the placement.

Net Length: By far the most important optimization metric - traditional metrics are Half Perimeter Wire Length (HPWL), this is a holdover from VLSI

design (as is most of the research) or some combination of manhattan or euclidean distance. It is currently seen as infeasible to use an auto-router to asses wire length

No Overlap: some approaches to placement allow for some overlap - especially simulated-annealing based ones. The electrostatic VLSI placer - rePlacer allows for overlap at the beginning. The operators we plan to implement cannot create a placement with overlap. This will ensure all generated placements are valid.

Most placement algorithms combine these into one all encompassing heuristic - which is the initial approach our GA based placer will take. However if time allows for implementing PSA/MOSA these metrics will create a Pareto front of which all points are considered "valid".

2.1 Measurable - Steps

Minimum

- Implement GA using operations from GA_breakdown.pdf

Should Achieve (Would be dissapointed if not)

- Add concurrent/multi-threaded support
- Ability to parse Kicad input and output to Kicad (this ideally will be handled via a crate but haven't tried it yet)

Stretch

- Utilizing mutations created above, implement PSA/MOSA
- Add concurrent/multi-threaded support to our SA

Looking Forward

- improve upon operators and try other simulated annealing based approaches
- implement a better wire length calculation (maybe A*?)

3 Concerns

Pretty minimal concern in terms of my initial scope, I am unsure if it makes sense to use the genome representation as described in the paper. I also am not sure how exactly I plan to set up the GA for concurrent processing.

4 Citations

Crocker, Peter, https://dspace.mit.edu/bitstream/handle/1721.1/139247/Crocker-pcrocker-meng-eecs-2021-thesis.pdf?sequence=1isAllowed=y Jain, Sait, https://asmedigitalcollection.asme.org/abstract/118/1/11/404553/PCB-Layout-Design-Using-a-Genetic-Algorithm?redirectedFrom=fulltext