PCB Climber

Mackenzie Norman

January 27, 2025

"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"

PCB Layout is a very difficult problem to automate, not only is it in it's self intractable, any heuristic used to asses quality is also intractable.

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 Perimiter 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 implent cannot create a placement with overlap. This will ensure all generated placements are valid.

Most placement algoritihims 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".

Measurable - Steps:

Must Achieve

- Implement GA PCB using operations from paper.
- Add concurrent/multithreaded 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/multithreaded support to our SA

Probably won't complete but someday would like to

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