Jain, Gea paper summary

Mackenzie Norman
mnor2@pdx.edu

February 1, 2025

1 Algorithm details and rough implementation vision

Since the paper is paywalled (why a paper from the mid 90's is paywalled is beyond me) I will attempt to quickly summarize the operators and how a 1d array is used to represent a chromosome.

1.1 Genomic Representation

The design space is initially discretized into a finite number of NxN cells. The paper recommends setting N to the LCM of the lengths of the sides of the chips. In my experience, with a more modern machine it is feasible to discretize the space to even smaller units. (In the real world PCB's are often designed with a grid that components snap to, ideally the units of this grid would be the discretization, but it can truly be arbitrary). With the grid discretized, the space a chip takes up is represented by a number on a list.

0	0	0	0	0	0
0	2	0	1	0	0
0	2	0	1	0	0
0	3	3	3	0	0
0	3	3	3	0	0
0	3	3	3	0	0

Table 1: Discretized Layout

This is then flattened to a one-dimensional array (another point of improvement may be using a 2 dimensional array)

[0,0,0,0,0,0,0,2,0,1,0,0,0,2,0,1,0,0,0,3,3,3,0,0,0,3,3,3,0,0,0,3,3,3,0,0]

Figure 1: Flattened Array

1.2 Genetic Operators

Because of the problem, it is rightly noted that using traditional mutation and crossover operators would often times result in unfeasible or impossible placements. Additionally a new operator is suggested: Compaction.

1.2.1 Mutation

The mutation operator has three different options. The first begins with selecting a component from a parent chromosome, removing and randomly selecting a new position in the chromosome where it can be placed. (Note can is the operative phrase here since it is possible there are no locations for it to be placed.)

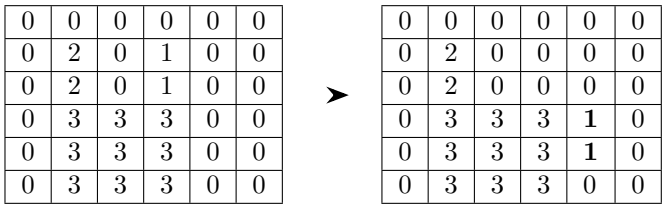

Figure 2: Move Mutation operator

The second mutation swaps 2 components. If either component cannot has overlap/out of bounds issues. these are attempted to remedied using a rotation.

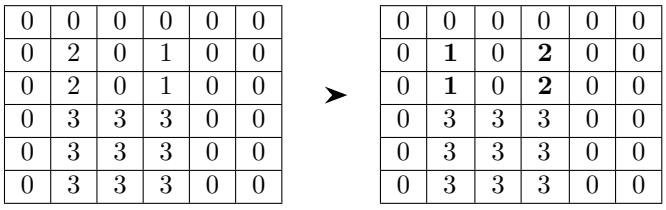

Figure 3: Swap Mutation operator

Third, a component is rotated at random. (In this algorithm, we simplify the problem by only allowing 90, 180,270 degree rotations.)

With all three mutations, if a chip does not "fit" then it is first rotated, then shifted to nearby cells, and finally moved to a random location.

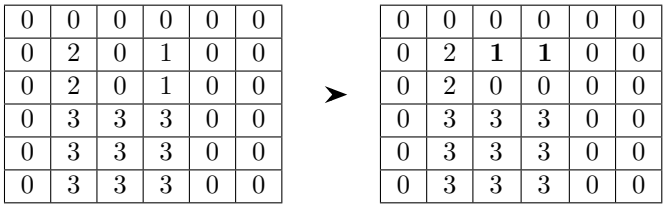

Figure 4: Rotation Mutation operator

1.2.2 Crossover

The crossover operator is relatively simple. Two parents are selected A and B. A rectangular region of random size is selected, and expanded to ensure no components are cut off, then in the child, the components from parent A are first placed, then all remaining components that fit from parent B are placed. The ones that do not fit are first checked to see if their locations in parent A would be feasible and if not, a random location is selected. Parents A and B are then swapped for child 2.

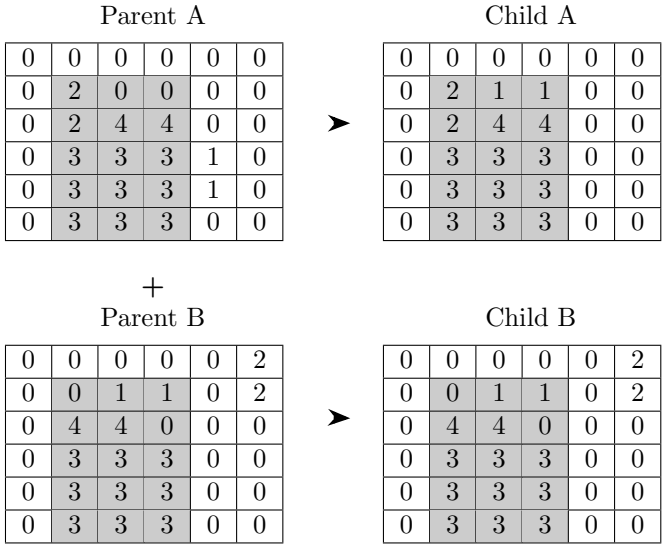

Figure 5: Crossover operator

1.2.3 Compaction

No details are given on the specific implementation of this operator. In my wildest dreams I would implement this with an FP approach that was encoded in the genome. A naive implementation of this is find the center of the placement, move components towards that going component-wise from the center out.

1.2.4 Evaluation

The evaluation is a normalized function of the three heuristics described in the intro of section 2. Plus any other penalty functions. this is described in (1) as

$$f = f_1 + f_2 + P * f_3 \tag{1}$$

with f_1 being placement area, f_2 as net length, f_3 being all other penalty functions.

1.2.5 Selection

The paper recommends two selection methods both stemming from Goldberg. An "Expected Value(EV)" Plan and the "Elitist" plan. They differ in that the EV plan uses the function to determine how many reproductions of an individual are in the next generation while in the Elitist plan the fitter individuals will persist onto the next generation.