Proceedings of the ASME 2022 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference
IDETC/CIE 2022
August 14-August 17, 2022, St. Louis, USA

DETC2022/CIE-90046

AUTOMATIC POWER PLANE GENERATION WITH GENETIC OPTIMIZATION AND MULTILAYER PERCEPTRON

HAIGUANG LIAO[1] **VINAY PATIL**[1] **XULIANG DONG**[1] **DEVIKA SHANBHAG**[1]
ELIAS FALLON[2] **TAYLOR HOGAN**[2] **MIRKO SPASOJEVIC**[2] **LEVENT BURAK KARA**[1*]
1. Carnegie Mellon University
Pittsburgh, PA 15213
2. Cadence Design Systems
San Jose, CA 95134

ABSTRACT

We present an automatic power plane generation method to accelerate the design of printed circuit boards (PCB). In PCB design, while automatic solvers have been developed to predict important indicators such as the IR-drop, power integrity, and signal integrity, the generation of the power plane itself still largely relies on laborious manual methods. Our automatic power plane generation approach is based on genetic optimization combined with a multilayer perceptron and is able to automatically generate power planes across a diverse set of problems with varying levels of difficulty. Our method consists of an outer loop genetic optimizer (GO) and an inner loop multi-layer perceptron (MLP) that generate power planes automatically. The critical elements of our approach include contour detection, feature expansion, and a distance measure to enable island-minimizing complex power plane generation. We compare our approach to a baseline solution based on A. The A* method consisting of a sequential island generation and merging process which can produce less than ideal solutions. Our experimental results show that our method outperforms A* in 71% of the design problems with varying levels board layout difficulty. We also present ablation studies demonstrating the influence of various algorithmic choices. Finally, we provide insights into how the power planes evolve with our model parameters to form feasible and desirable space partitions.*

*Address all correspondence to this author.

1 Introduction

Following Moore's Law [1], modern electronics designs are getting increasingly more complex. Printed Circuit Boards (PCB) design, as one of the critical steps in electronics design is also getting increasingly more complex and time-consuming with the increasing size of boards, more complex design rule constraints (DRCs), as well as signal integrity (SI) and power integrity (PI) requirements. One of the most critical steps that hamper PCB design development is the generation of a power plane [2], which aims to configure the various metal planes' layout on a PCB. Although there has been significant development in automatic solvers for the critical objectives of PCB design related to power plane generation such as IR-drop solver [3,4,5], power integrity solver [6,7], and signal integrity [8,9,10], the generation of power plane itself still relies on the manual effort of electronics designers. Thus, automatic power plane generation is needed to address an unmet need in PCB design.

In this work, we present an evolutionary automatic power plane generation method based on a multilayer perceptron (MLP) and genetic optimization (GO), which we refer to as **GOMLP**. This method is based on a nested combination of a multilayer perceptron and genetic optimization, with critical components including contour detection, feature expansion and customized distance metrics. Systematic experiments are conducted to study the contributions of the critical components and also stress test the method on a diverse set of problems with varying difficulties. The method is compared against a baseline solution based on A* [11].

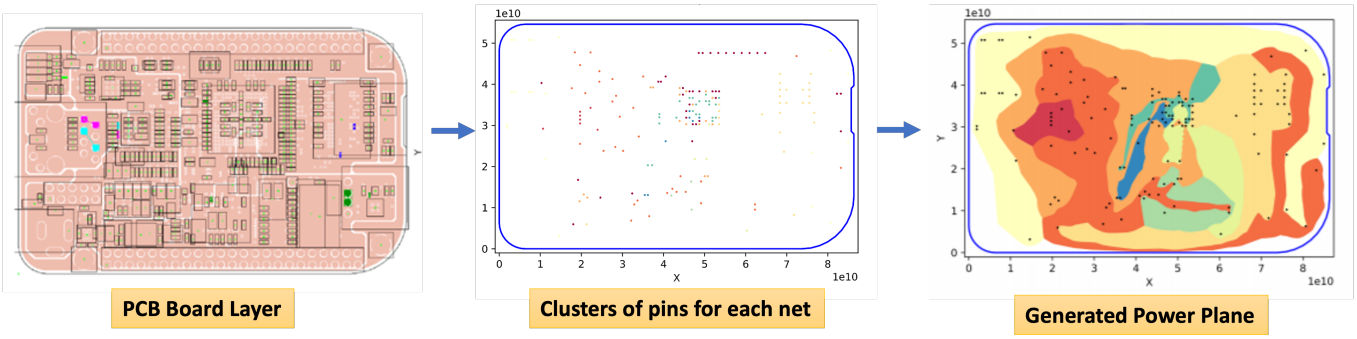

FIGURE 1. Power plane generation on a PCB board. Middle: Pins belonging to the same net are colored similarly. Right: Planes belonging to the same net are colored similarly. Note that some nets are split into multiple islands.

To the best of our knowledge, this work is the first effort to solve power plane generation with fully automated methods.

2 Background

2.1 Problem Formulation: Power Plane Generation

Power plane generation is a critical step PCB design, where the layout of metal planes connecting critical components is configured on each layer of the PCB board [12, 2]. As shown in Fig 1, a layer of the PCB board contains a set of m nets $\{N_1, N_2, N_3, ..., N_m\}$. Each net is composed of a potentially large number of pins (the I/O ports of the components on the board) $P_{N_i} = \{p_1, .., p_{q_i}\}$, where q_i is the total number of pins for N_i. $p_j \in \mathbb{R}^2$ encodes the x and y coordinates of a pin. Net information and corresponding pin locations are determined in a previous step of board design and component placement. As such, all pin locations are fixed and are not subject to change. The objective is to group together all the pins with the same label (net assignment) through a metal plane, while excluding all other pins.

Power plane generation can be formulated as a **mutually exclusive space partitioning** of the board Ω into m distinct power planes $\Omega_1, .., \Omega_m$ such that:

$$\Omega = \Omega_1 \cup \Omega_2 \cup ... \cup \Omega_m \quad and \quad \Omega_i \cap \Omega_j = \emptyset \tag{1}$$

Plane Ω_i corresponds to net N_i, which means the number of planes is equivalent to the number of nets. Each plane Ω_i may consists of several **islands**: $\Omega_i = I_i^1 \cup I_i^2 ... \cup I_i^{s_i}$, where s_i is the total number of islands of plane (or net) i.

Design Constraint: All pins of Net N_i must be located inside Ω_i (possibly spread over multiple islands) and no pin from another net can appear in Ω_i:

$$P_{N_i} = \{p_1, .., p_{q_i}\} \in \Omega_i \tag{2}$$

Design Objective: The power plane generation problem formulation described above is the basic form. For each plane Ω_i, it is preferable to generate a singly connected plane consisting of a single island. Otherwise, disconnected islands belonging to the same plane will have to be connected on a separate layer of the PCB using extraneous connections and vias. Mathematically, this can be stated as finding an optimized space partitioning Ω^* which minimizes the total number of split islands:

$$\Omega^* = argmin_\Omega \sum_{i=1}^{m} s_i \tag{3}$$

Given a board, we denote a set of power planes that satisfy the design constraint as **feasible**, and for each net, those that have a minimum number of islands as **desirable**. Fig. 2 shows examples. For the example design, there are two nets $\{N_1, N_2\}$, where N_1 has two pins and N_2 has one pin. The desirable designs (E and F) are with singly connected partitions that are feasible. Designs C and D, while feasible, split one of the nets into two separate islands. Compared to E and F, these designs are not desirable.

The objective of minimizing the number of islands for each net makes the problem non-trivial since planes corresponding to different nets have to compete for the shared design space while satisfying the design constraint. Intuitively, power plane generation is analogous to a 2D multi-class image segmentation problem aimed at minimizing the number of islands for all nets while ensuring that no pin is misclassified. While approaches such as image segmentation [13, 14, 15] and space partitioning [16, 17] have been extensively studied, these approaches do not put emphasis on split island minimization. To the best of our knowledge the power plane generation is an open problem not only in the scope of PCB design but also in the broader context of space partitioning problems.

2.2 Scope

Our approach seeks to find a feasible solution that minimizes the number of islands. In this work, the priority is on identifying power planes that match this desire. However, we do not optimize the outer contour geometries of the planes and reserve this consideration for future work. As such, in the presented work, designs E and F in Fig. 2 are identical in quality.

In industrial PCB design, there exists other constraints and objectives for space partitioning. These include considerations around IR-drop, power integrity, and signal integrity of the partitioned board. For instance, it is common to impose an upper bound on the admissible IR-drop for each source-drain pin pair within a plane, which favors planes with the least number of thin regions. For each power plane candidate, the evaluation of the IR drop involves a physical simulation that can be prohibitive. In this work, we thus exclude such considerations in favor of establishing the foundations for island minimizing partitioning. Future work will have to account for such criteria.

Additionally, current work only considers a single layer of the PCB, while in practical settings the nets and associated pins can be assigned to different layers in a stacked PCB. From this perspective, the presented work aims to identify an optimized solution to the nets and pins already assigned to a specific layer (i.e., net and pin assignment is assumed to have been completed).

3 Methods

We parameterize the problem using a set of **handles** defined for each plane Ω_i as $H_{N_i} = \{h_1, .., h_{k_i}\}$, where $h_j \in \mathbb{R}^2$ encodes the x and y coordinates of the handle on the board. These handles serve as labelled movable points on the board such that an optimized placement of the entire set of handles $H_{N_i}, i : 1..m$ produce a partitioning of the board where all original (and static) pin sets $P_{N_i}, i : 1..m$ are correctly grouped, and the number of power plane islands is minimized.

We refer to our method as GOMLP, which is summarized in Fig. 3. Our method consists of an outer loop genetic optimizer (GO) that identifies the best handle placements. To evaluate a candidate solution, we use an inner loop multi-layer perceptron (MLP) that produces a 2D segmentation of the board using H_{N_i} and $P_{N_i}, i : 1..m$ as the labelled data to be classified. The MLP uses an expanded feature space and, is allowed to overfit to the data so as to ensure all pins are correctly classified (i.e., satisfying the design constraint). A separate module then identifies the total number of islands together with the pairwise distances between the islands of the same net. These quantities are appended to GO's fitness function to drive the optimal handle placement.

3.1 Genetic optimization for handle placement

Genetic Optimization (GO) has found extensive use in problems where the gradients of the objective function and constraint

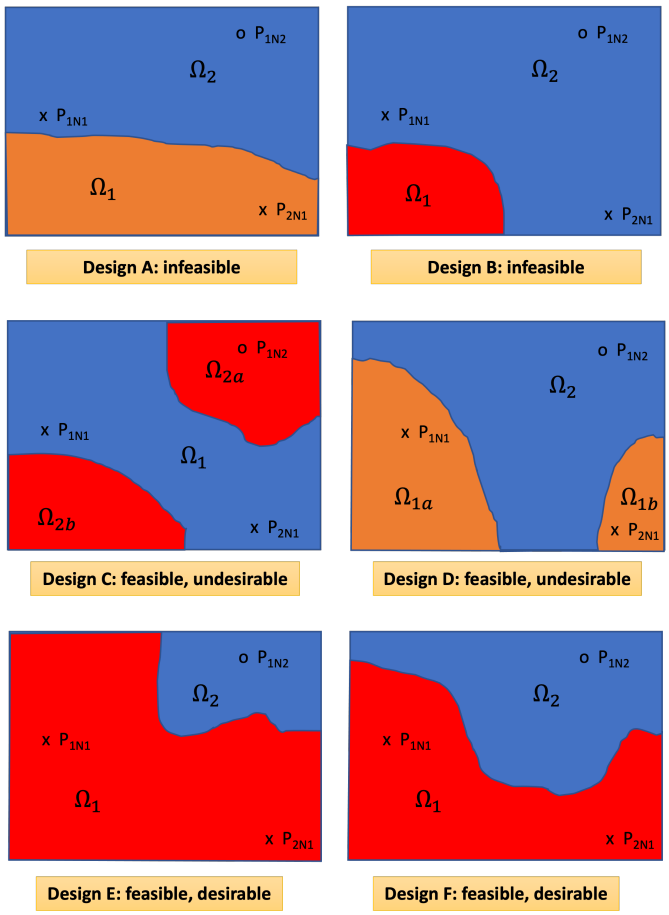

FIGURE 2. Plane feasibility vs. desirability.

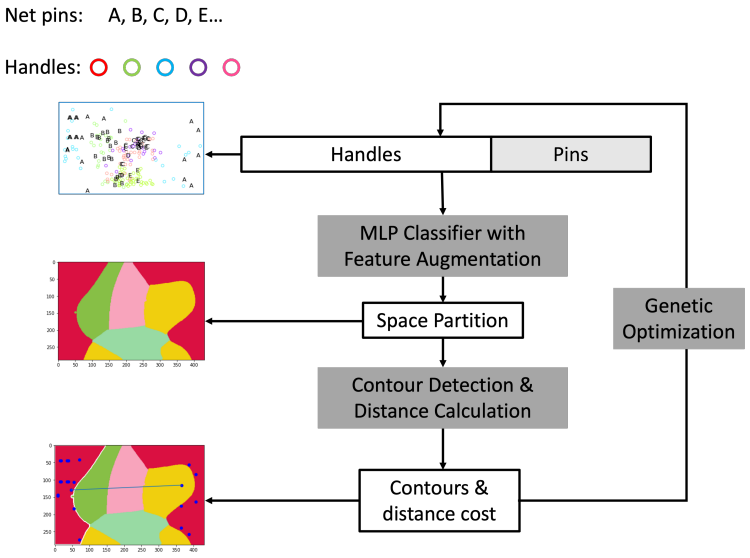

FIGURE 3. Proposed method.

are not readily available [18, 19, 20]. In this work, we use GO to optimize the handles' locations, which then are used as input to the MLP to produce the space partitions . The primary motivation for applying GO instead of other gradient-based optimizers is that the objective function in the power plane generation problem is not readily differentiable.

In our GO, a population consists of M chromosomes, where each chromosome encodes the handle coordinates as a vector of genes. Thus, each chromosome is a candidate solution for Ω. Each gene has two scalars x, y encoding the 2D coordinates of a handle. For each $\Omega_i, i : 1..m$, we instantiate k handles, where k is calculated as follows:

$$k = \frac{2 \cdot \sum_{i=1}^{m} q_i}{m} \tag{4}$$

where q_i is the total number of pins for net i. Each chromosome is thus a $2 \cdot k \cdot m \times 1$ real vector, with the handles belonging to each net occupying a $2 \cdot k \times 1$ block. Handles are randomly initialized in the first generation of GO. We use a fitness-proportional parent selection, uniform mutation and probabilistic random swap for crossover operators [21, 22]. Fig. 3-top shows the collection of pins (letters) and handles (circles) for a 5-net design problem.

3.2 Power plane generation using MLP

Given a chromosome, we use an MLP to construct the partitioned power planes. The MLP takes as input the labelled training set $P_{N_i} \cup H_{N_i}, i : 1..m$ and creates a region segmentation of the board using the multi-class cross entropy loss. Fig. 3-middle shows the current space partitioning result. For each item in the training set, we expand its x, y coordinates into a 15×1 vector: $\{x, y, x \cdot y, \sin(2\pi \cdot x), \cos(2\pi \cdot x), x^2, \sin(2\pi \cdot y), \cos(2\pi \cdot y), y^2, \sin(3\pi \cdot x), \cos(3\pi \cdot x), x^3, \sin(3\pi \cdot y), \cos(3\pi \cdot y), y^3\}$ to enable complex boundaries. The MLP is trained with these features as input and the net classification as output is then used to generate space partition Ω by predicting the plane assignment for every coordinate from the meshgrid on the PCB board layer.

The MLP consists of three hidden layers, with 50 neurons in each layer and the *tanh* activation function. Adam optimizer with a learning rate of 0.002 is used. It is desirable to overfit to the training data, so as to favor all pins to satisfy the feasibility constraint of Eqn. 2. Nonetheless, the resulting partitioning may result in disconnected islands, and may not satisfy Eqn. 2.

3.3 Fitness calculation

To evaluate each solution's fitness, we determine the total number of islands s_i of each plane Ω_i generated by the MLP using a contour detection algorithm [23].

$$F^{island} = -\sum_{i=1}^{m} s_i \tag{5}$$

Additionally, we append the fitness function with a measure to distinguish the islands of a plane Ω_i that are far apart from those that are proximate. This helps GO acquire guidance in handle placement for the subsequent generation (Fig. 3-bottom). For Ω_i, we define:

$$F_i^{d_{min}} = -\sum_{a,b}^{s_i} ||I_i^a - I_i^b||_{min} \tag{6}$$

where I_i^a, I_i^b denote islands a and b belonging to Ω_i, and $||.||$ is the Euclidean norm. $F_i^{d_{min}}$ captures the sum of minimum distances between the pairs of the disjoint islands of Ω_i.

While min. distances are useful in signaling nearly connected islands, they typically favor the generation of elongated thin connections. To additionally drive the islands toward one another, we define a centroid distance as a part of the fitness function:

$$F_i^{d_{cent}} = -\sum_{a,b}^{s_i} ||I_i^a - I_i^b||_{cent} \tag{7}$$

Fig. 4 illustrates the above two islands distance calculations. For a segmentation solution produced by the MLP, the final fitness maximized by the GO is:

$$F = F^{island} + \sum_{i}^{m} F_i^{d_{min}} + \sum_{i}^{m} F_i^{d_{cent}} \tag{8}$$

4 Method: A* Baseline Method

A* is one of the most frequently used search algorithms in graph traversal and path finding. In this work, A* search is used as the backbone of our baseline power plane generation algorithm, which we refer to as **A***. It has been proven that with admissible heuristics, the A* can find the shortest path between two vertices in a graph [24]. Fig. 6 shows the flowchart of the A* baseline. The reason for selecting the A* as our baseline is that it has been the most frequently used algorithm practice to solve routing problem [25, 26, 27]. In the baseline method, the solution consists

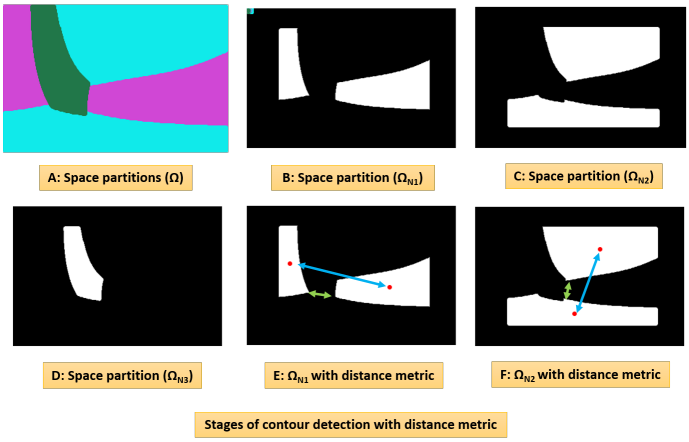

FIGURE 4. Contour detection and distance metric calculations of space partitions.

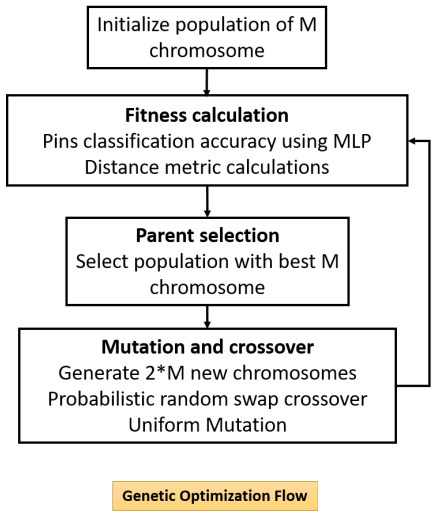

FIGURE 5. Details of GO structures and flow

of multiple steps and one of the key steps (Island Connection) can be formulated as a routing problem.

The input of the power plane generation problem is clusters of pins belonging to each net $\{N_1, N_2, ..., N_m\}$. For A*, the first step is the **Island Generation**. At this step, for each net N_i, a minimum spanning tree (MST_{N_i}) among all the pins P_{N_i} belonging to net N_i is generated with using Kruskal's algorithm [28]. After the MST for all nets $\{MST_{N_1}, MST_{N_2}, ..., MST_{N_m}\}$ are generated, edge pruning is applied to all the MSTs where any edges of the MSTs that intercept edges of other nets are pruned. After pruning, there is a set of disconnected trees (pins connected with edges) for each net, which is represented by an island that includes all the pins belonging to the tree while excluding all other pins not belonging to the tree. For instance, if island generation produces s_i disconnected trees $\{T_1, T_2, ..., T_{s_i}\}$ for Net N_i, then the disconnected trees are represented by a corresponding set of islands $\{I_i^1, I_i^2, ..., I_i^{s_i}\}$, where the area of I_j include all the pins of tree T_j but not include pins belonging to other trees of net N_i or other nets. Based on the Island Generation step, each net is represented as a set of islands. The islands for all the nets are fed into the downstream **Island Connection** step, where A* algorithm is applied to route all the disconnected islands for each net to generate a connected tree for each net. Specifically, given an order of the nets, for each net N_i with disconnected island set $\{I_i^1, I_i^2, ..., I_i^{s_i}\}$, A* works on connecting the disconnected island set. For the disconnected island set with s islands, $(s-1)$ pairs of islands are generated, and then A* will sequentially connect all the pairs of islands by searching through a graph consisting of upsampling nodes distributed on the board. The Island Connection is finished when all the island pairs belonging to all the nets have been attempted to be connected with A*.

Since all the island pairs of one design have to share the same space when connected with A*, it is not guaranteed that all of them can be connected. If the Island Connection step is successful, then a set of trees $\{T_{N_1}, T_{N_2}, ..., T_{Nm}\}$ will be generated where T_{N_i} connected all the pins belonging to net N_i. If some islands are not fully connected, some nets will end up having more than one tree. Based on the generated tree set from the Island Connection step, the final power plane is generated by inflating the tree edges of each tree. Specifically, densely located nodes are sampled on all edges of the trees so that each tree T_{N_i} can be represented as a set of nodes. Then the space partition is generated by running K-nearest Neighbors (KNN) [29] algorithms based on the nodes from all trees with the number of neighbor k set as 1.

The A* baseline approach provides a practical solution flow to generate a power plane. However, a downside of A* is its greedy manner [30,31]: it focuses on finding out the optimal solu-

Algorithm 1: GOMLP power plane generation

Input : Netlist of designs: $\{N_1, N_2, ..., N_m\}$
Output : Space partitions: $\{\Omega_1, \Omega_2, ..., \Omega_m\}$
Initialize handles $\{h_1, h_2, ..., h_{k_i}\}$ for each net N_i as the first population for GO;
for *generation=1,...,N* **do**
  MLP fitting all fixed points $\{P_1, P_2, ..., P_{q_i}\}$ and handles $\{H_1, H_2, ..., H_{k_i}\}$ for each net N_i with augmented features;
  Contour detection and distance metric calculations for all planes $\{\Omega_1, \Omega_2, ..., \Omega_m\}$;
  Calculating fitness scores for all chromosomes based on MLP fitting results;
  Generating new population with crossover and mutation based on selected elites;
end

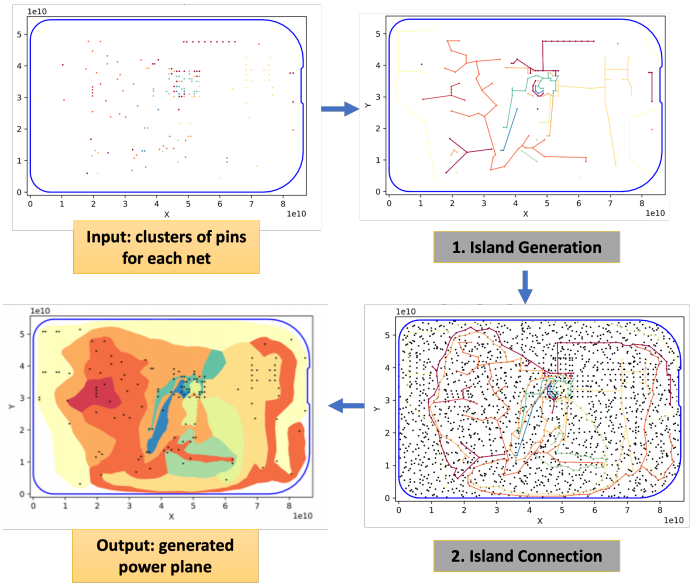

FIGURE 6. Flowchart of A* Power Plane Generation Method.

tion for only one search problem. In power plane generation, there are cases where multiple search problems that need to be solved on the same graph sharing limited graph space. Under such case, A* would struggle to consider all the problems simultaneously. Thus, there are at least the following combinatorial optimization challenges to be addressed to make it an applicable approach:

Sequence of nets: in the Island Connection step, the A* has to be applied to the nets sequentially. Thus, an optimal sequence of nets needs to be determined. Otherwise, the method will be unfavorable to the nets that are routed later because the nets routed earlier have occupied part of the search graph.

Sequence of island pairs: similar to the sequence of nets challenge, within each net, the island pairs also need to be sequenced before A* is applied. The sequence is a non-trivial combinatorial optimization problem.

Design of upsampling nodes: to run the A* algorithm efficiently, the search graph size need to be reasonably small, and thus designing the distribution of upsampling nodes is another combinatorial optimization challenge that needs to be solved.

5 Evaluation

To systematically study the performance of our approach, extensive experiments are performed, which can be divided into two parts. In part one, GOMLP is applied to solve problems with and without the key components to show the effect of the key features of our method, which include the following:

Algorithm 2: A* power plane generation

Input : Netlist of designs: $\{N_1, N_2, ..., N_m\}$
Output : Space partitions: $\{\Omega_1, \Omega_2, ..., \Omega_m\}$
Tree generatoin $\{MST_{N_1}, MST_{N_2}, ..., MST_{N_m}\}$ with MST for each net;
Island generation by tree pruning for each net N_i: $\{T_1, T_2, ..., T_{s_i}\}$;
for net **for** i=1,...,m **do**
  Running A* to connect the isolated islands $\{I_i^1, I_i^2, .., I_i^{s_i}\}$ for net N_i, forming complete tree edges for each net
end
Generating final space partitions by running KNN on the upsampled nodes from edges of different net

Effect of GO on MLP: Given the same design problem, power planes are generated with standalone MLP and MLP with GO to investigate the effect of GO upon MLP.

Effect of feature expansion on GOMLP : Given the same design problem, GOMLP is applied with and without expanded features to investigate the effect of feature expansion.

Effect of distance metric on the GOMLP: Given the same design problem, GOMLP is applied with and without the distance metric to investigate its effects on performance of the GOMLP.

In the second part of the experiments, we apply GOMLP as well as the baseline A* method to a set of power plane generation problems with different configurations and difficulties to systematically study the performance of our method.

5.1 Dataset

To generate the problems, given a layer of design on a PCB board, we vary the number of nets from 6 to 8, with more nets corresponding to more challenging problems. For each net number, combinations of nets are selected to generate distinct problems. With the variations, there are altogether 129 different problems with varying difficulties. The GOMLP as well as the baseline A* are applied to each of the problems with maximum run time for solving each problem set as 30 minutes for a fair comparison.

Cross entropy is used as the loss function. For the GO, the maximum number of generations is set as 20, population size is 30 and elite size is 10. An industry real PCB board design **Beaglebone** is used, and different design problems in the experiment come from different design problems on different layers. Since the objective in this work is to minimizes the total number of split islands, in evaluating the quality of generated power planes, the primary metric used is **Extra Islands (EI)**, which is defined as follows:

$$EI = \sum_{i=1}^{m} s_i - m \tag{9}$$

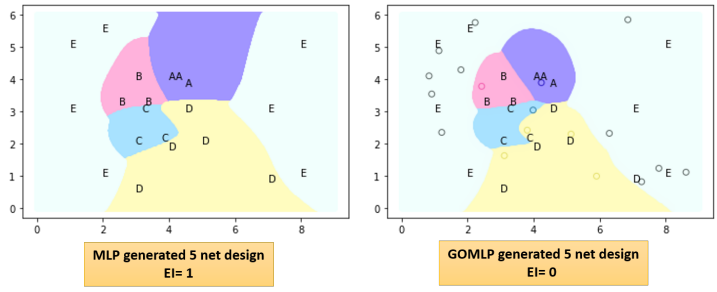

FIGURE 7. Power plane generation on 5 nets PCB layer design case using MLP vs GOMLP. MLP: 6 partitions with 2 disjoint partitions belonging to net E , evolutionary MLP : 5 partitions with no disjoint partitions

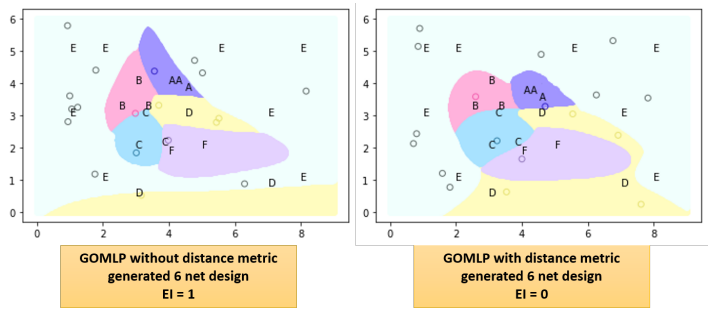

FIGURE 8. Power plane generation on 6 nets PCB layer design using GOMLP with and without distance metric.

6 Results and Discussions

6.1 Comparison between MLP and GOMLP

Fig. 7 , shows the results of MLP and GOMLP on a 5 net case. On the left, the partitions are generated using only MLP. Although the MLP can generate a power plane design that correctly classifies the nets. It fails to converge to the most desired power plane partitioning where the partition for each net is connected. As a result, the partition corresponding to net E is divided into two islands. On the right side, the partition is generated using GOMLP. It was executed for just 2 generations and generated the desired result which satisfied constraints. Also, more importantly, the space partitions for each net are all single connected planes. The result demonstrates that the power plane generation is a nontrivial space partition problem that can not be solved directly with a classifier. The challenge for a single classifier to generate a feasible power plane is due to the difficulty of encoding the design objective of overall island minimization as a differentiable function into the loss function of MLP. While Euler number [32] allows the determination of the number of disconnected islands, connecting that loss to the handle parameters is not readily achievable.

In our GOMLP, the combination of MLP and GO solves this challenge by optimizing the handle locations to control the shape of the MLP generated space partitions. Since handle optimization seeks to minimize the number of islands, GO will work on connecting isolated islands of the MLP-generated partitions implicitly.

6.2 Comparison between GOMLP with and without distance based cost

Fig. 8 shows the experiment results of the GOMLP with and without the distance metric described in the Method part. A 6-net PCB board layer design is given and the GOMLP is applied twice to solve the problem: the first time with the distance metric and the second time without the distance metric. The distance metric describes the proximity of isolated islands for each net. On the left of Fig. 8, the generated power plane is based on GOMLP without including distance metric, which means only the number of isolated islands are used to describe the layout of the space partition. On the power plane, the space partition of net E is divided into two isolated islands on the left and right sides of the plane. While on the right of Fig. 8, the generated power plane is based on GOMLP with the distance metric included. In this case, the space partition corresponding to net E is connected as a single U-shaped plane during the iterative optimization procedure of the GOMLP.

The improved performance with the introduction of the distance metrics compared to cases where the only number of isolated islands are used in cost function can be explained as follows: in the optimization process of the GOMLP, the space partition is controlled by the layout of the handles. The handles are iteratively optimized to generate higher fitness scores through the GO operation. This mechanism means that the GO is working on optimizing the planes implicitly through the handles' movement. As a result, in cases where the handles for one net are distributed as distant clusters, for instance, the case in Fig. 8, it would be challenging for the GOMLP to provide the handles with a good moving direction without the introduction of distance metric. In other words, the number of isolated islands is not enough to provide enough guidance for driving the handles to gradually move and form connected space partitions for each net. The distance metric provides something similar to gradient information to guide the handles gradually move in order to connect isolated islands for each net.

6.3 Comparison of GOMLP with and without feature expansion

Fig. 9 shows the results GOMLP with and without feature expansion on a 6 net PCB layer design. On the left, the partitions are generated with GOMLP without the use of feature expansion. Without the expanded features, the model is unable to generate complex space partitions that are feasible and desirable. In the generated power plane, Ω_D is separated into two isolated islands, which makes it undesirable. Whereas on the right of Fig. 9, the

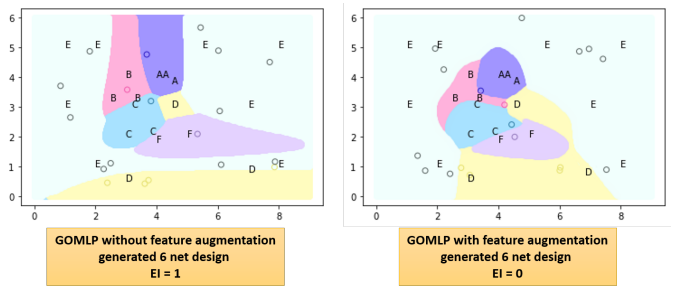

FIGURE 9. Power plane generation on 6 nets PCB layer design using GOMLP with and without feature expansion. GOMLP: 7 partitions with 1 disjoint partitions belonging to net D, with feature expansion: 6 partitions with no disjoint partitions.

partitions are generated by GOMLP with feature expansion. With the additional higher order features as input, the MLP can learn more complex space partitions and curved space partitions which leads to desirable power plane generation. In the generated power plane, the partition for Ω_D is a singly connected plane with a thin channel connecting the two isolated regions of net D. The comparative results demonstrate the effect of feature expansion in our method. The introduction of high dimensional features makes it easier for the MLP to generate complex space partitions such as zig-zag-shaped planes.

6.4 Scalability of GOMLP

As described in the Experiment part, the second part of the experiments includes the applications of both the GOMLP and A* method to problems with different configurations and difficulties to systematically assess the performance and scalability of the GOMLP and A*. For the 129 problems experimented, the number of nets ranged from 6 to 8, and problems with more tend nets problems tend to be more difficult. Given a solution, EI is used as the primary metric. Only when EI equals 0, the generated power plane is desirable.

Fig. 10 shows the results of GOMLP and A* on the set of 129 problems. The plot shows the number of extra islands for solutions of GOMLP and A* in the ascending order of evolutionary MLP results, as well as their gaps. The plot shows that GOMLP is significantly better in generating high quality power planes with fewer extra islands. Among all the 129 experiments, GOMLP outperforms or has the number of extra islands as A* in 71% of cases. For problems where A* and GOMLP perform similarly, both methods tend to identify the optimal power planes ($EI = 0$). A paired t-test of GOMLP and A* on our 129 board dataset indicates that GOMLP is statistically better than A* ($p<0.001$).

A* outperforms GOMLP in only 4 out of the 129 experiments. Detailed studies are conducted on those rare cases where A* outperforms GOMLP. The primary reason that GOMLP did not perform as well as A* in these cases is due to the presence of

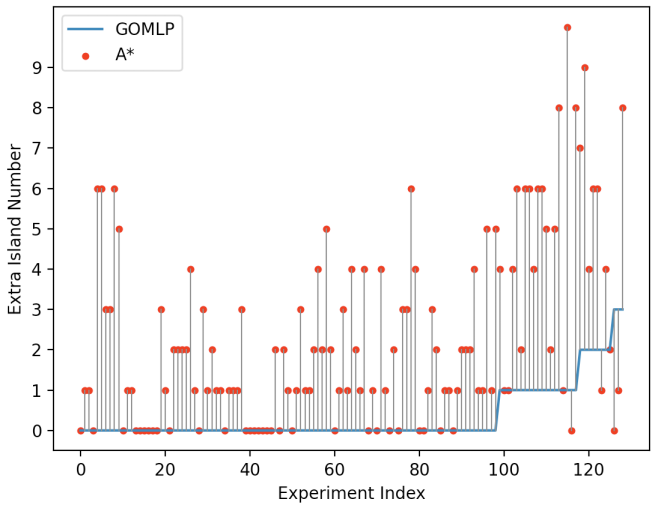

FIGURE 10. GOMLP vs A* for the 129 different board problems. Vertical axis shows the extra islands. GOMLP is better than A* in 71% of the cases.

very near-by pins that belong to different nets, challenging the MLP to correctly classify these pins. However, these cases will not likely be encountered in realistic PCB board designs given the multilayer flexibility of the problem: designers would move some of the very close pins belonging to different layers of the PCB board to avoid such corner cases. To sum up, GOMLP has good scalability to non-trivial power plane generation designs and performs significantly better than A* baseline.

Finally, to give a more intuitive demonstration of how the GOMLP works, Fig. 11 and Fig. 12 show the evolution of space partitions and the handles across generations. In Fig. 11, the pins (annotated with alphabets) are correctly classified in the very first few generations, however, the space partition for net E is divided into two separated islands in the left and right side of the board. As the evolutionary MLP keeps optimizing the space partitions, the two islands for net E start to approach and then finally merged into a desired one single plane. The handles are the implicit driving force for the evolution, as shown in Fig. 12, where the handles, pins, and the planes are plotted together. The handles locations changed across generations to implicitly change the space partitions generated by the MLP. It is worth mentioning that the moving patterns of the handles are not as intuitive as we expected. For instance, during the merge of two islands for net E, the handles for net E on two different islands are not directly moving towards the other islands. This is because the relationship between handles and the space partitions is highly nonlinear and it is part of our future work to further study the evolutionary patterns of the handles. For design C,D,E and F in Fig. 12, they are all converged solutions for the same problem, showing another advantage for GOMLP: generating multiple solutions for the same

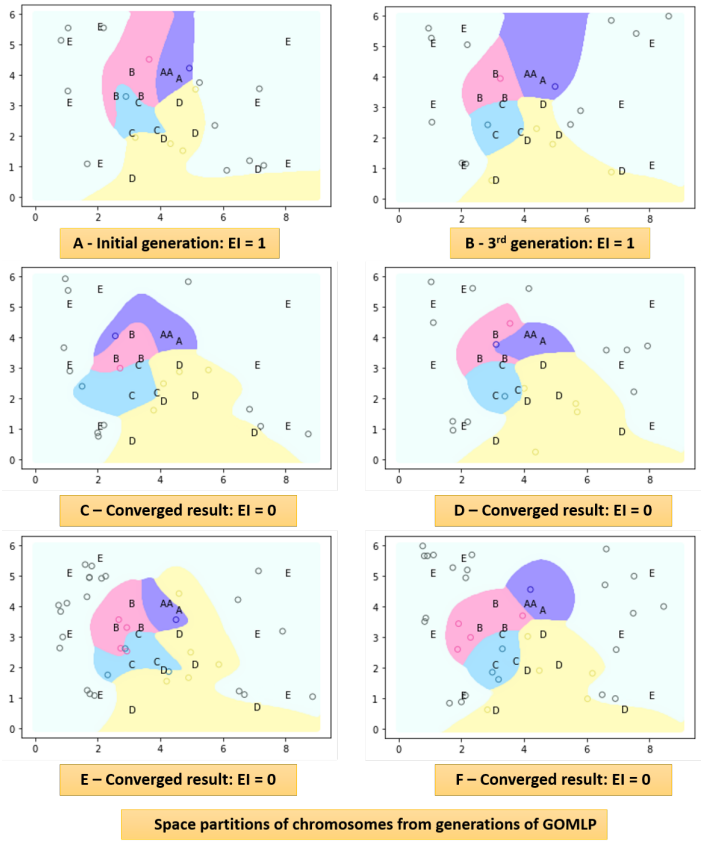

FIGURE 11. Resulting power planes for a 5-net design problem

problem. Fig. 13 shows the generated power planes by GOMLP and A* on the same set of the problems with 6, 7, and 8 number of nets. For the three problems, GOMLP is able to generate desirable power planes, while A* method generates power planes that have more EI numbers.

6.5 Discussions

In comparing GOMLP and A*, one obvious advantage that contributes to the better performance of GOMLP is that it can avoid the combinatorial challenges that A* faces: net sequencing, sequence of islands pairs as well as the design of upsampling nodes. It has been shown [33] that there is no heuristic for net ordering that is universally better for all possible problems. This means that methods like A* need to solve the NP-hard combinatorial problem. Even if optimized net sequences or upsamplings are given, the sequential nature of such a method means that the subproblems solved first are prioritized over those solved later, which is not desirable in partitioning problems of this sort [34]. GOMLP provides a good alternative to avoid such combinatorial challenges. Another advantage of the GOMLP over A* is the flexibility to add complex objectives into the method. The A* method is based on generating skeletons of each net using search, which solely focuses on finding the shortest paths, this makes adding more complex objectives such as IR-drop to A* challenging. On the other hand, GOMLP optimizes over the fitness score in GO, which means the objectives do not have to be explicit and differentiable. This will make sure that the GOMLP can be easily extended to more complex objectives.

7 Conclusion

In this work, we present a GOMLP automatic power plane generation method based on the combination of MLP and GO. The method is compared against a baseline A* solution. The GOMLP is based on a combination of multilayer perceptron and genetic optimization, with critical components including contour detection, feature expansion as well as the customized distance metric. The GOMLP demonstrates the ability to automatically generate power planes across a diverse set of non-trivial problems with different levels of difficulty. Comparative study also shows the GOMLP is significantly better (in 71% cases) than A* and those critical components including feature expansion, distance metric as well as the combination of multilayer perceptron and genetic optimization contribute to the competitive capabilities of the GOMLP in automatically generating power planes. Future works include extending the GOMLP to multilayer PCB power plane generations and adding more complex objectives such as IR-drop.

Acknowledgment

This work is partially funded by the DARPA IDEA program (HR0011-18-3-0010).

REFERENCES

[1] Schaller, R. R., 1997. "Moore's law: past, present and future". *IEEE spectrum,* ***34***(6), pp. 52–59.

[2] Smith, L. D., Anderson, R., and Roy, T., 2001. "Power plane spice models and simulated performance for materials and geometries". *IEEE Transactions on Advanced Packaging,* ***24***(3), pp. 277–287.

[3] Zhong, Y., and Wong, M. D., 2005. "Fast algorithms for ir drop analysis in large power grid". In ICCAD-2005. IEEE/ACM International Conference on Computer-Aided Design, 2005., IEEE, pp. 351–357.

[4] Nithin, S., Shanmugam, G., and Chandrasekar, S., 2010. "Dynamic voltage (ir) drop analysis and design closure: Issues and challenges". In 2010 11th International Symposium on Quality Electronic Design (ISQED), IEEE, pp. 611–617.

[5] Zhang, M.-S., and Tan, H. Z., 2015. "Ir-drop modeling and reduction for high-performance printed circuit

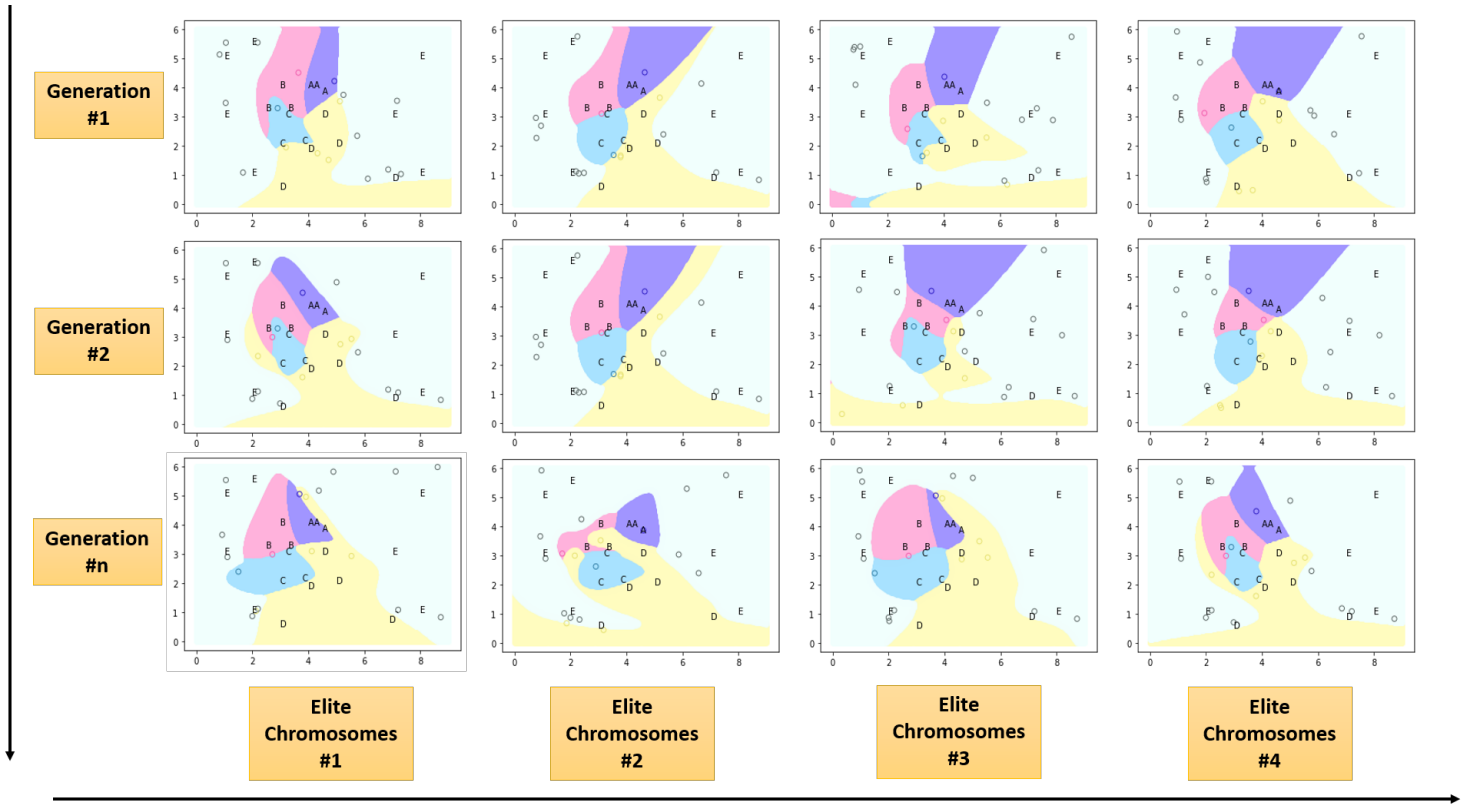

FIGURE 12. Evolution of the handles across the generations. Colored circles: handle locations, letters A-E: pins belonging to different nets. Elite#4 in generation#3 converges to the desired result.

boards". *IEEE Electromagnetic Compatibility Magazine,* ***4***(4), pp. 90–101.

[6] Wu, T.-L., Chuang, H.-H., and Wang, T.-K., 2010. "Overview of power integrity solutions on package and pcb: Decoupling and ebg isolation". *IEEE Transactions on electromagnetic compatibility,* ***52***(2), pp. 346–356.

[7] Fizesan, R., and Pitica, D., 2010. "Simulation for power integrity to design a pcb for an optimum cost". In 2010 IEEE 16th International Symposium for Design and Technology in Electronic Packaging (SIITME), IEEE, pp. 141–146.

[8] Kim, J., and Li, E., 2010. "Special issue on pcb level signal integrity, power integrity, and emc". *IEEE Transactions on Electromagnetic Compatibility,* ***52***(2), pp. 246–247.

[9] Chen, X.-P., 2010. "Analysis and application for integrity of pcb signal". In 2010 2nd IEEE International Conference on Information and Financial Engineering, IEEE, pp. 328–331.

[10] Eudes, T., Ravelo, B., and Louis, A., 2011. "Experimental validations of a simple pcb interconnect model for high-rate signal integrity". *IEEE transactions on electromagnetic compatibility,* ***54***(2), pp. 397–404.

[11] Tseng, F. H., Liang, T. T., Lee, C. H., Der Chou, L., and Chao, H. C., 2014. "A star search algorithm for civil uav path planning with 3g communication". In 2014 Tenth International Conference on Intelligent Information Hiding and Multimedia Signal Processing, IEEE, pp. 942–945.

[12] Cui, W., Fan, J., Ren, Y., Shi, H., Drewniak, J. L., and DuBroff, R. E., 2003. "Dc power-bus noise isolation with power-plane segmentation". *IEEE transactions on electromagnetic compatibility,* ***45***(2), pp. 436–443.

[13] Cheng, H.-D., Jiang, X. H., Sun, Y., and Wang, J., 2001. "Color image segmentation: advances and prospects". *Pattern recognition,* ***34***(12), pp. 2259–2281.

[14] Pham, D. L., Xu, C., and Prince, J. L., 2000. "Current methods in medical image segmentation". *Annual review of biomedical engineering,* ***2***(1), pp. 315–337.

[15] Haralick, R. M., and Shapiro, L. G., 1985. "Image segmentation techniques". *Computer vision, graphics, and image processing,* ***29***(1), pp. 100–132.

[16] Wang, L., Fonseca, R., and Tian, Y., 2020. "Learning search space partition for black-box optimization using monte carlo tree search". *arXiv preprint arXiv:2007.00708*.

[17] Buck, R. C., 1943. "Partition of space". *The American Mathematical Monthly,* ***50***(9), pp. 541–544.

[18] Whitley, D., 1994. "A genetic algorithm tutorial". *Statistics*

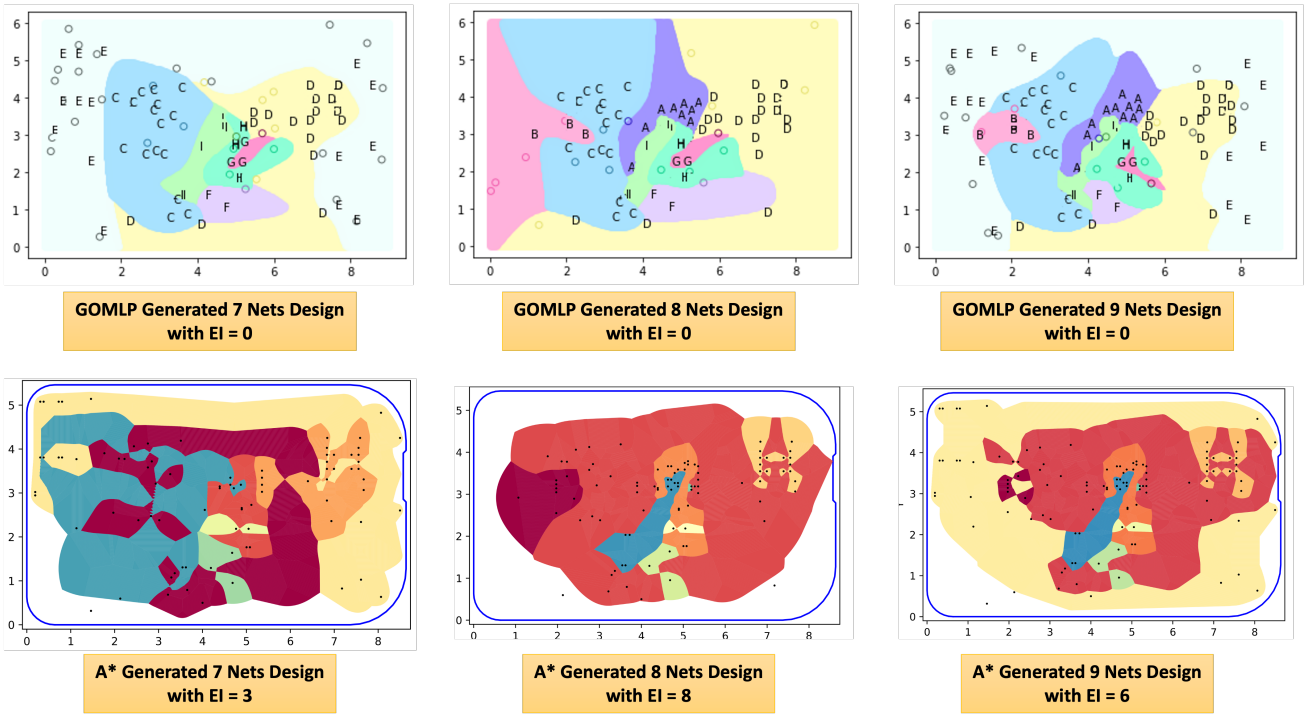

FIGURE 13. GOMLP (top) and A* (bottom) generated power planes for problems with 6,7 and 8 nets.

and computing, ***4***(2), pp. 65–85.

[19] Jing, T., Lim, M. H., and Ong, Y. S., 2003. "A parallel hybrid ga for combinatorial optimization using grid technology". In The 2003 Congress on Evolutionary Computation, 2003. CEC'03., Vol. 3, IEEE, pp. 1895–1902.

[20] Mühlenbein, H., 1992. "Parallel genetic algorithms in combinatorial optimization". In *Computer science and operations research*. Elsevier, pp. 441–453.

[21] Lipowski, A., and Lipowska, D., 2012. "Roulette-wheel selection via stochastic acceptance". *Physica A: Statistical Mechanics and its Applications,* ***391***(6), pp. 2193–2196.

[22] Hassanat, A., Alkafaween, E., Alnawaiseh, N., Abbadi, M., Alkasassbeh, M., and Alhasanat, M., 2016. "Enhancing genetic algorithms using multi mutations: Experimental results on the travelling salesman problem". *International Journal of Computer Science and Information Security,* ***14***, 09, pp. 785–801.

[23] Gong, X.-Y., Su, H., Xu, D., Zhang, Z., Shen, F., and Yang, H.-B., 2018. "An overview of contour detection approaches". *International Journal of Automation and Computing,* ***15***, 06, pp. 1–17.

[24] Gelperin, D., 1977. "On the optimality of a". *Artificial Intelligence,* ***8***(1), pp. 69–76.

[25] Hu, J., and Sapatnekar, S. S., 2001. "A survey on multi-net global routing for integrated circuits". *Integration,* ***31***(1), pp. 1–49.

[26] Ozdal, M. M., and Wong, M. D., 2007. "Archer: a history-driven global routing algorithm". In 2007 IEEE/ACM International Conference on Computer-Aided Design, IEEE, pp. 488–495.

[27] Liao, H., Zhang, W., Dong, X., Poczos, B., Shimada, K., and Burak Kara, L., 2020. "A deep reinforcement learning approach for global routing". *Journal of Mechanical Design,* ***142***(6), p. 061701.

[28] Kruskal, J. B., 1956. "On the shortest spanning subtree of a graph and the traveling salesman problem". *Proceedings of the American Mathematical society,* ***7***(1), pp. 48–50.

[29] Peterson, L. E., 2009. "K-nearest neighbor". *Scholarpedia,* ***4***(2), p. 1883.

[30] Wayahdi, M. R., Ginting, S. H. N., and Syahputra, D., 2021. "Greedy, a-star, and dijkstra's algorithms in finding shortest path". *International Journal of Advances in Data and Information Systems,* ***2***(1), pp. 45–52.

[31] Dogar, M. R., Koval, M. C., Tallavajhula, A., and Srinivasa, S. S., 2014. "Object search by manipulation". *Autonomous Robots,* ***36***(1), pp. 153–167.

[32] Dyer, C. R., 1980. "Computing the euler number of an image from its quadtree". *Computer graphics and image processing,* ***13***(3), pp. 270–276.

[33] Abel, L. C., 1972. "On the ordering of connections for automatic wire routing". *IEEE Transactions on Computers,* ***100***(11), pp. 1227–1233.

[34] Chen, H.-Y., and Chang, Y.-W., 2009. "Global and detailed routing". In *Electronic Design Automation*. Elsevier, pp. 687–749.