A Multi-Objective Design for PEBB-Based Power Corridors in Shipboard Applications

S. Lohier, J. S. Chalfant
Design Laboratory, MIT Sea Grant College Program, Massachusetts Institute of Technology (MIT), Cambridge, MA, USA
email: slohier@mit.edu, chalfant@mit.edu

***Abstract*—This paper presents a novel methodology for the automatic placement of Power Electronics Building Blocks (PEBBs) in modular, integrated power corridor designs. These building blocks are currently arranged manually during the design process, a method that is time-consuming and suboptimal. To address this challenge, the placement problem is reduced to a 2D bin-packing problem, leveraging a hybrid approach combining Genetic Algorithms and Simulated Annealing. This approach enables the generation of optimized placements that find the extremes of arbitrary heuristics, including minimizing routing distance and power density, effectively improving both design efficiency and system performance. The proposed methodology offers a significant step toward automating and optimizing the layout of power electronic components in complex systems.**

***Index Terms*—Power Electronic Building Block, Power Corridor, Optimization Algorithm.**

I. Introduction

In an era of increasingly sophisticated naval warfare, the design and efficiency of ship power systems have become critical factors in maintaining military readiness and operational effectiveness. The modular integrated power corridor design for ships represents a significant advancement in naval engineering, offering a cost-effective yet robust solution to meet the intensive power needs of future warships [1]. This innovative approach integrates power transfer, conversion, isolation, and storage functionalities into standardized, off-hull fabricated modules that can be easily integrated onboard. The

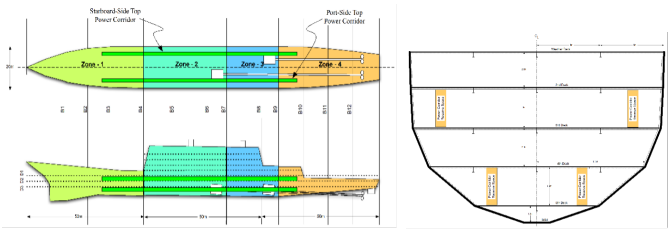

Fig. 1: Extent of power corridor within a representative ship. This example is a highly redundant four-corridor example. A minimum of two corridors is envisioned for redundancy.

potential benefits of this design are substantial. By leveraging modularity, standardization, and off-hull construction, the power corridor system promises to significantly reduce both construction and life-cycle costs. Moreover, the standardized

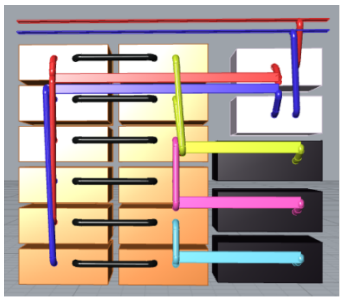

Fig. 2: Example manually arranged design

functionality, hardware, and control interfaces of these modules enhance maintainability and upgradability, crucial factors in the long-term operational viability of naval vessels. An example power corridor layout is shown in Fig. 1.

However, to fully realize the advantages of this modular integrated power corridor design, an optimal placement of electrical components is essential. This placement must be both routing-efficient and power-dense, a complex task that, when performed manually, is time-consuming and prone to suboptimal outcomes. In early-stage ship design, large design space exploration can result in thousands of individual placements, necessitating automatic placements. Therefore, this project proposes to develop a time-efficient algorithmic solution capable of producing near-optimal component placements within the power corridor framework.

A. Problem

The core challenge addressed in this project is the optimal placement of electrical components within the modular integrated power corridor of naval vessels. This challenge can be primarily conceptualized as a variant of the 2D bin-packing problem, a well-known optimization problem in operations research.

In its classical form, the 2D bin-packing problem aims to efficiently pack a set of rectangular items into a minimum number of rectangular bins, ensuring no overlap and adherence to bin boundaries. Our specific scenario adapts this problem to a single defined bin representing the power corridor.

It is crucial to note that the 2D bin-packing problem is NP-hard [2], indicating the absence of known efficient algorithms

for optimal solutions. Consequently, our approach necessitates the development and application of heuristic methods to achieve reasonable, near-optimal solutions within practical time constraints.

However, the problem extends beyond the traditional bin-packing paradigm. In addition to efficient placement within spatial constraints, additional heuristics such as the minimization of routing distances between interconnected components must also be considered. This additional objective significantly increases the complexity of the problem, requiring a more sophisticated approach than standard bin-packing algorithms. To solve this, a hybrid meta-heuristic technique which utilizes both a Genetic Algorithm and Simulated Annealing is proposed.

B. Genetic Algorithm

Genetic Algorithms are a class of optimization algorithm inspired by the process of natural selection. They have shown practical success in finding near optimal solutions to complex problems with large search spaces. This is done through a repeated cycle of initialization, crossover, mutation, fitness computation, selection, and termination. The following description is primarily based on Kramer (2017) [3].

1) Basic Principles: To understand a Genetic Algorithm, one must first understand the several core definitions.

- Population: A set of candidate solutions to the problem. Each individual is a possible solution, generally encoded as a string of values known as chromosomes.
- Fitness Function: An arbitrary function which, when given a possible solution, outputs the quality of the given solution, allowing the algorithm to identify good solutions.
- Selection: A method of deciding which members of the population will be used to generate the next population.
- Crossover: A genetic operator which combines two parent individuals to generate an offspring. It is meant to simulate genetic recombination and results in a set of new individuals which have attributes of both "parents". This is generally performed by some method of exchanging genes between two individuals.
- Mutation: A genetic operator which introduces small random changes to the child chromosomes. This allows the population to maintain genetic diversity and avoid premature convergence to suboptimal solutions. This is a computational analog of biological mutations.

The algorithm employs an iterative optimization approach that begins with the generation of an initial population comprised of randomly instantiated individuals. Following population initialization, each solution is evaluated through a fitness function that quantifies its performance relative to the optimization objectives. A selection mechanism then identifies two parent solutions based on their respective fitness values. These selected parents undergo genetic recombination operations to produce offspring that constitute the subsequent generation's population. This evolutionary process — encompassing fitness evaluation, parent selection, and reproductive operations — continues iteratively until predetermined termination criteria are satisfied. The best solution seen thoughout the process is then returned.

C. Simulated Annealing

Simulated Annealing (SA) is a probabilistic metaheuristic optimization algorithm that has gained significant attention in various fields of study, including operations research, artificial intelligence, and computational physics. Its foundation lies in the Metropolis-Hastings algorithm, a Monte Carlo method used to generate sample states of a thermodynamic system.

The algorithm's name and inspiration derive from the annealing process in metallurgy, where controlled cooling of a material leads to the formation of large crystals with minimal defects. In the context of optimization, this process is analogous to finding a global minimum in a complex search space. These definitions are based on material produced by Henderson [4]

1) Basic Principles: Similarly to Genetic Algorithms, Simulated Annealing (SA) is built upon a few core definitions.

- State Space: Let S be the set of all possible solutions.
- Energy Function: E(s), where s $\in$ S, quantifies the quality of a solution.
- Neighborhood Function: N(s) generates a neighboring solution.
- Temperature Schedule: T(k), where k is the iteration number.
- Acceptance Probability: P(ΔE, T) determines the likelihood of accepting a worse solution based on the energy difference and a temperature parameter

SA, which also utilizes an iterative optimization approach, begins with an initial solution and high "temperature" value. At each iteration, it evaluates a neighboring solution and decides whether to accept or reject it based on its Energy value. If the new solution represents an improvement, it is automatically accepted; however, if the new solution is inferior, it may still be accepted with a probability given by P(ΔE, T), which is typically defined as:

$$P(\Delta E, T) = e^{\frac{-\Delta E}{T}} \tag{1}$$

The acceptance probability formula, derived from the Boltzmann distribution in statistical mechanics, is a key component that allows the algorithm to accept suboptimal solutions with decreasing probability as the temperature drops. This mechanism is intricately linked to the temperature schedule, which plays a crucial role in the algorithm's performance. The temperature schedule dictates how quickly the algorithm transitions from exploration to exploitation. Common approaches to temperature scheduling include:

- Linear cooling: T(k) = $T_0 - \alpha * k$
- Geometric cooling: T(k) = $T_0 * \alpha^k$
- Logarithmic cooling: T(k) = $\frac{\alpha}{log(k+1)}$

In this case, k is the step number and α is a hyper-parameter to be tuned for optimal results. Each schedule has its advantages and is suited to different types of problems.

Under certain conditions, SA can be proven to converge to the global optimum with probability 1. These conditions being a sufficiently slow cooling schedule, the ability to reach any state from any other state in a finite number of moves, and the probability of selecting any neighbor being greater than zero. However, in practice, these conditions often conflict with computational feasibility, leading to the use of faster cooling schedules that sacrifice theoretical guarantees for practical performance.

II. Related Works

This problem bears striking similarities to the place-and-route (P&R) challenges encountered in Very Large Scale Integration (VLSI) chip design. In VLSI design, components (gates, memory blocks, and other circuit elements) must be efficiently placed while considering both spatial constraints and the optimization of interconnect routing. Over the past several decades, researchers have developed numerous approaches to tackle this complex optimization problem. These techniques range from deterministic algorithms like force-directed placement and quadratic optimization, to meta-heuristic methods such as simulated annealing, genetic algorithms, and hybrid approaches. The success of these methods in VLSI design provides valuable insights for our problem, as both domains share core challenges in optimizing component placement while minimizing interconnection distances and managing thermal constraints.

A. Deterministic Algorithms

Several researchers have explored deterministic algorithms for solving packing and placement problems, aiming to provide consistent and reproducible results [5]–[7]. These approaches often leverage mathematical programming techniques, heuristics, or specialized data structures to systematically explore the solution space. Deterministic methods can offer advantages in terms of predictability and, in some cases, guaranteed optimality, though they may struggle with scalability for larger problem instances. These works demonstrate the potential of deterministic algorithms in tackling complex packing and placement challenges, although they often need to balance solution quality with computational efficiency, especially for real-world applications involving numerous objects or intricate constraints.

1) Hybrid First Fit: Chung et al. [8] presented a deterministic placement algorithm to find near-optimal packings of rectangular 2D shapes. In their First Fit by Decreasing Height (FFDH) approach, all the rectangles to be packed are sorted in descending order based on their height. The algorithm then places these rectangles into the alloted space one by one, always positioning the current rectangle as far left as possible in the lowest strip that can accommodate it without overlapping previously placed rectangles. If there is no possible strip to hold it, a new strip is created with the same height as the block, above the current top block. This process effectively creates horizontal levels or "blocks" within the strip. The researchers innovatively extended this approach by considering these resulting blocks as larger, composite rectangles. They then subjected these new, larger blocks to a second round of packing. This two-phase approach allows for potentially more efficient use of space, as the second packing phase can exploit gaps left between the initial blocks. By repacking these larger units, Chung et al. aimed to reduce the overall height of the strip and improve the packing density, potentially overcoming some of the limitations of the initial FFDH placement.

2) Force-based approach: Liao et al. [9] introduced a novel physical force-driven packing optimization method for solving Strip Packing Problems (SPP). Their approach begins with a mathematical optimization model for SPP, focusing on minimizing the strip length while ensuring all objects fit without overlap.

The core of their method is the convex hull plus rubber band compact layout technique. This innovative approach uses a virtual rubber band to bind all layout objects from the periphery, causing them to move closer together under elastic forces. The authors provide a detailed physical analysis of the forces acting on objects during the layout process, including rubber band forces and interaction forces between objects, and implement a time-based simulation of this process.

To enhance the packing process, Liao et al. propose an improved version of the mate algorithm for rectangle packing. This algorithm uses minimal rectangles to replace polygon objects and employs score decision rules to choose the next packing object, allowing for dynamic adjustment of the packing sequence.

The packing process in their method unfolds in four stages: initial packing, rubber band force-driven, uniform force-driven, and packing adjustment. This multi-stage approach allows for progressive refinement of the packing arrangement.

To validate their method, the authors conducted computational experiments, comparing their results with other approaches in the literature. These experiments demonstrated the feasibility and effectiveness of their method, showing promising results in terms of packing density and computational efficiency.

As a practical application of their research, Liao et al. developed an auto-layout system using C++ and Box2D, based on their proposed method. This system demonstrates the potential for real-world implementation of their force-based packing approach.

This work represents a significant contribution to force-based approaches in packing problems, offering a novel perspective that combines physical principles with optimization techniques. The method shows promise in achieving good packing density with high time efficiency, particularly for complex 2D irregular polygon packing problems.

Given the need to address additional complexities beyond packing, such as wiring considerations and specific placement requirements, we decided to pursue a different approach altogether, rather than building upon the force-based method of Liao et al. and the strip-packing-based approaches of Chung et al. Our research necessitated the development of a more

versatile strategy to accommodate these diverse constraints and objectives.

B. Randomized Search Algorithms

Previous research in the field of optimization has demonstrated the efficacy of metaheuristic approaches in solving complex packing problems [10]. One particularly relevant study has shown promising results in tackling variations of the 2D bin-packing problem:

Soke and Bingul [11] explored the use of Genetic Algorithms and Simulated Annealing for two-dimensional non-guillotine rectangular packing problems. In this context, guillotine packing refers to a placement strategy where all components are arranged such that the entire layout can be subdivided into smaller rectangles through a series of straight cuts that extend from one edge to another. This constraint ensures that components can be physically separated by straight-line cuts. Their work demonstrated the potential of these meta-heuristic techniques in finding near-optimal solutions for complex packing scenarios.

Soke and Bingul's Improved Bottom Left (BL) placement algorithm begins by positioning each block in the top-right corner of the designated area. It then drops the block downward until it collides with a previously placed block. From there, the block slides to the left and continues to fall until it can no longer move. Thus a placement is defined by the order of blocks placed.

1) Genetic Algorithm: Soke and Bingul implemented a Genetic Algorithm (GA) approach as one of their primary methods for tackling the two-dimensional non-guillotine rectangular packing problem. Their GA representation is notable for its simplicity and effectiveness: each chromosome represents a specific ordering of the blocks to be placed.

The fitness function, a critical component of any GA, was carefully designed to optimize for placement density. For a given ordering π, the fitness F(π) is defined as:

$$F(\pi) = \frac{1}{\epsilon + T} \tag{2}$$

where T, the trim loss value, is

$$T = \frac{A_m - \sum A_p}{A_m}, \tag{3}$$

A_m is the area of the main object, and A_p is the area of each placed piece. The inclusion of ϵ prevents division by zero and facilitates fine-tuning of the selection pressure.

For the selection process, the authors employed the Roulette Wheel Selection operator. This probabilistic selection method assigns each individual a selection probability proportional to its fitness, calculated by dividing its fitness value by the sum of all fitness values in the population. This approach balances exploration and exploitation in the search space.

The breeding process to produce the next generation involved extensive experimentation with various crossover techniques. Notably, the Order-Based Crossover (OBX) operator emerged as the most effective. The OBX operation proceeds as follows:

1) Randomly sample half of the genes from the first parent and copy them to the child.
2) Fill the remaining genes from the second parent while preserving their original order.

This method effectively combines genetic material from both parents while maintaining the crucial ordering information inherent in the packing problem. This is because the relative ordering of the genes selected between each parent is the same.

Finally, to introduce further variability and prevent premature convergence, each child undergoes mutation. The mutation operator simply swaps a random pair of blocks in the ordering, providing a mechanism for small, potentially beneficial perturbations in the solution.

This GA approach demonstrates a thoughtful balance between preserving beneficial ordering information and introducing variability to explore the solution space. The authors' systematic evaluation of different components, particularly the crossover operators, provides valuable insights for researchers seeking to apply GA to similar combinatorial optimization problems.

2) Simulated Annealing: Soke and Bingul also explored a Simulated Annealing (SA) approach as a comparative method for solving the two-dimensional non-guillotine rectangular packing problem. Their implementation of SA showcases several key design choices and parameter considerations.

- Initial Solution and Representation: The authors used the same representation as in their GA approach, where a solution is encoded as an ordering of blocks to be placed. The initial solution was generated randomly.
- Neighborhood Structure: Two different neighborhood exploration types were investigated
  - Swapping move: Exchanges the positions of two randomly selected blocks.
  - Shifting move: Removes a randomly selected block and reinserts it at a random position.

  After extensive testing, the swapping move was found to be more effective and was adopted for the final implementation.
- Cooling Schedule: Two cooling schedules were evaluated:
  - Lundy and Mees schedule: $T_{k+1} = T_k/(1 + \alpha T_k)$
  - Proportional decrement schedule: $T_{k+1} = \alpha T_k$

  The Lundy and Mees schedule demonstrated superior performance and was chosen for the final algorithm.
- Acceptance Probability: The standard Metropolis criterion was employed for accepting or rejecting new solutions: P(accept) $= e^{\frac{-\Delta E}{T}}$, where ΔE is the change in energy (trim loss in this case) and T is the current temperature.
- Inner Loop Criterion: The authors experimented with different inner loop termination criteria, testing values of 1, 3, and 5. They found that an inner loop criterion of 3 produced the best results, striking a balance between exploration at each temperature and overall runtime.

- Termination Condition: The SA algorithm terminated when the temperature reached a predefined minimum value or when no improvement was observed for a specified number of consecutive iterations.
- Parameter Tuning: Considerable effort was devoted to tuning the SA parameters, including the initial temperature, cooling rate, and termination conditions. The authors emphasize the importance of this tuning process in achieving competitive results.

3) Analysis: This paper's careful consideration of different neighborhood structures, cooling schedules, and termination criteria provides valuable insights in solving our problem. The comparative analysis between SA and GA reveals distinct advantages and limitations of each approach in the context of two-dimensional packing problems. Genetic Algorithms excel at exploring diverse solution spaces through population-based evolution and can effectively handle multiple objectives through fitness function design. However, they can be computationally intensive and may struggle with fine-tuning solutions in the later stages of optimization. In contrast, Simulated Annealing offers superior local search capabilities and can efficiently escape local optima through its temperature-based acceptance probability, but may miss global optima due to its single-solution nature and is highly sensitive to parameter tuning. Understanding these complementary characteristics allowed us to develop a hybrid approach that leverages GA's global exploration capabilities with SA's local search precision, enabling optimization beyond mere packing density to include routing efficiency and thermal distribution considerations.

III. Novel Hybrid Optimization Method

A. Full Problem Statement

The optimization problem addressed in this project focuses on the automated placement of power electronic components within a rectangular area of fixed height. The system is comprised of three primary component types: DC switches, Power Electronic Building Blocks (PEBBs), and AC switches, each with specified rectangular dimensions that must be maintained. The power flow follows a hierarchical structure, beginning at DC switches, passing through chains of PEBBs for power conversion, and terminating at AC switches. The number of PEBBs in each chain is determined by the system's voltage and power requirements, with no predetermined maximum limit on chain length. The placement area's physical constraints require all components to be positioned within the bounded region without overlap as shown in Figure 3. The electrical architecture demands specific connection patterns: both DC and AC switches connect to their respective horizontal bus bars, which in turn connect to the PEBBs via strictly vertical connections. These bus bars, which can be positioned at any height coinciding with their respective switches, may share the same vertical position and extend horizontally as needed to reach their connected components. PEBBs within a chain must maintain sequential connectivity, connecting from DC bus bars through the PEBB chain to the AC bus bars.

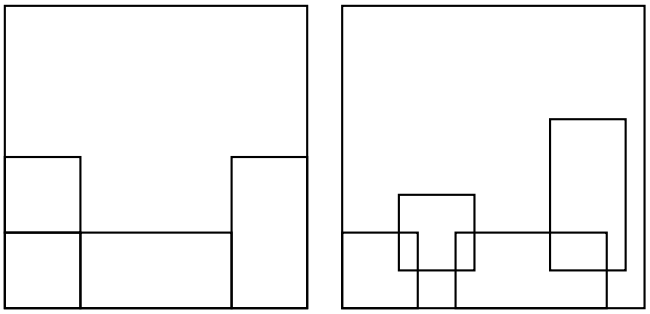

Fig. 3: The left image showcases a valid placement of components withing the placement area. The right image is an invalid placement as evident by the overlapping component placements.

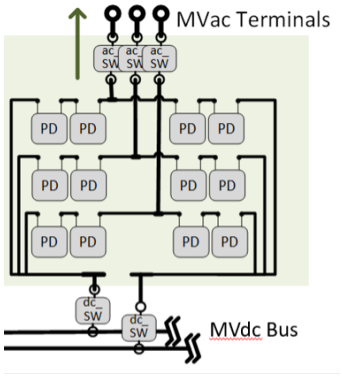

Fig. 4: Example electrical design to be implemented [12]

B. Proposed Solution

In this paper a new hybrid optimization method is proposed. Due to its closeness to the 2D bin-packing problem, the new approach is motivated by work done by Soke and Bingul [11]. A Genetic Algorithm is first used to optimize for all of the spatial constraints of the problem. These include, but are not limited to, packing density and electromagnetic interference. After achieving a terminating solution in the GA, the solution is passed through a constrained Simulated Annealing process, which aims to optimize for routing distance. Finally, several post-processing mechanisms are applied for ease of implementation.

An example system is shown in Figure 4.

1) Input: Inputs to the algorithm include input voltage and power of the DC bus, the required output voltage of the AC switches and the number of AC phases. Additionally, a user must describe the dimensions of the room in which the blocks will be paced. These parameters give the algorithm an idea of the system being designed.

2) Pre-processing: Before the algorithm begins optimizing it first uses the input parameters to determine the arrangement of electrical components. The required voltage determines the number and length of the PEBB chains between the AC and DC switches. The required power determines the number of parallel PEBB chains. Lastly, the required number of phases

determines the number of sets of parallel PEBB chains (one set for each phase).

3) Optimization: The optimization objectives follow a strict hierarchy. The primary goal is to minimize the total volume of the placement area while maintaining all connectivity requirements. Given a satisfactory volume, secondary objectives are considered. These include minimizing the distance between dc+ and dc- bus bars to reduce electromagnetic interference (EMI), keeping AC switches and their associated bus bars of the same phase in close proximity for system simplicity, and minimizing the total wire length with enhanced weight given to PEBB-to-PEBB connections. Additionally, the placement algorithm strives to position DC switches as high as possible within the placement area when feasible.

4) Output: The algorithm generates a comprehensive visual representation of the optimized component layout. DC components are rendered in green. AC components are displayed in blue. PEBBs that form functional chains are assigned matching colors, with each distinct chain receiving a unique color identifier. This visual grouping immediately conveys the functional relationships between interconnected components. The connection infrastructure is differentiated by color as well, with busbars depicted in green, vertical connections to busbars depicted in red, and PEBB-to-PEBB interconnections are shown in black. This visualization serves as both a technical blueprint for implementation and an analytical tool for evaluating the effectiveness of the algorithm's placement strategy. The color-coded representation allows for intuitive assessment of component grouping, connection efficiency, and overall system organization.

5) Placement Algorithm: As a basis on which the rest of the implementation rests, a new modified BL placement algorithm is developed. In this placement algorithm, blocks are initially placed in the top-right corner of the placement area and moved down and to the left until they can no longer be moved.

Unlike the original algorithm, the highest-placed block is tracked. Before a block is permanently placed, a check is performed to determine if this block will be the new highest-placed block. If this is the case, the block is repositioned by first placing it in the top left corner and then moved down and to the right until it can no longer move. All following blocks will be placed the same way until a new highest block is found and the direction is reversed again. This helps ensure two blocks near each other in ordering would also be located near each other in space. Previously, two connected blocks that fall across the border of the placement area would be placed on opposite ends of the structure.

Additionally, the x and y directions are flipped to ensure the placements utilize the height fully before increasing the width of the placement. This is due to the practical use of the placement algorithm: since this placement will be used on a ship it is more desirable to fill the unoccupied height of the compartment rather than create a larger horizontal footprint.

6) Genetic Algorithm:

a) Fitness Function.: The Genetic Algorithm utilizes a fitness function, $F(\pi)$, which optimizes for all of the stated

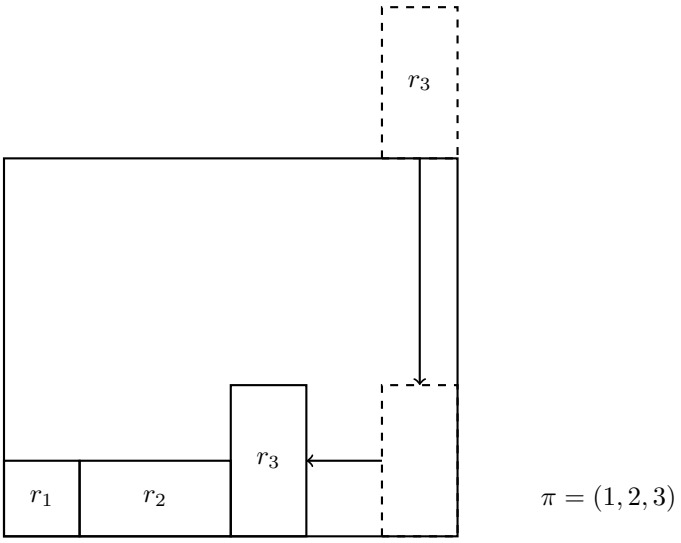

Fig. 5: Simple placement of three blocks. In this case the blocks slide to the bottom and slide to the left.

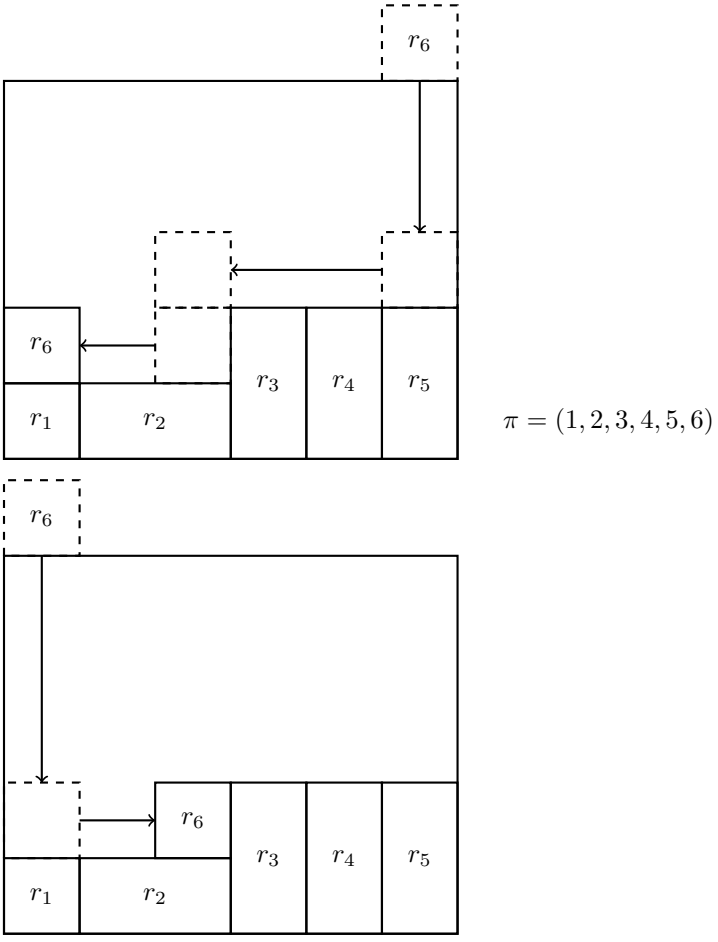

Fig. 6: A continued placement up to 6 blocks. In this case a the placement algorithm first tries an initial placement. It then sees that the r_6 will be placed at a new height. In response it retries a placement from the left and continues to do so until it reaches a new height.

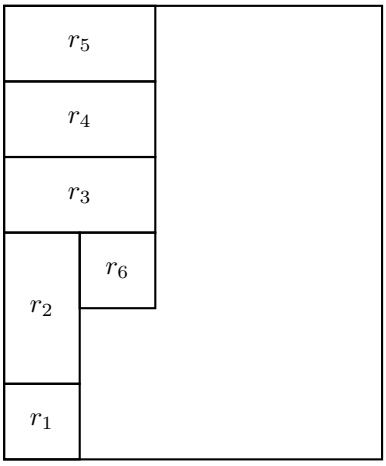

Fig. 7: At the end, the x and y coordinates are flipped in order to maximize the height used rather than the width.

goals by taking the product of each of the individual goals weighted by their respective exponents.

$$F(\pi) = (\frac{1}{V})^{\mu}(\frac{1}{D_{bb}})^{\tau}(\frac{1}{D_w})^{\alpha}(\frac{1}{D_{dc}})^{\beta}(\frac{1}{D_{AC}})^{\epsilon}(\frac{1}{W_s})^{\lambda} * (H_{dc})^{\gamma} * 1.1^{-(PSO)\theta} \tag{4}$$

where the overall volume, V, is the total volume of the NiPEC segment; the busbar routing distance, D_{bb}, is length of the connections between the DC switches and PEBBS; the wire routing distance, D_w, is the length of the PEBB-to-PEBB and PEBB-to-AC switch connections; the DC block distance, D_{dc}, is the distance between DC switches; the DC height, H_{dc}, is the height of the DC switches; the AC block distance, D_{AC} is the distance between AC switches; the switch width, W_s is the horizontal range of switch blocks, measuring the distance between the rightmost edge of the rightmost switch and the leftmost edge of the leftmost switch; and PSO is the PEBB-switch overlap. The PSO is calculated by finding the horizontal range of the PEBB block placements and comparing it to the horizontal range of the switch locations; the overlap of these ranges are calculated by

$$PSO = max(S_{le} - P_{re}, P_{le} - S_{re}) \tag{5}$$

where S_{le} and S_{re} are the left and right edges of the leftmost and rightmost switches, respectively, and P_{le} and P_{re} are the left and right edges of the leftmost and rightmost PEBBs, respectively. This value is positive when there is no overlap, and negative with some overlap. To make this value suitable as a fitness parameter, we take its exponential, making its range only positive.

V is the most critical measure, allowing the algorithm to optimize for densely packed solutions. Minimizing D_{bb} and D_w reduces the physical weight of connecting hardware such as cables and busbars. To minimize electromagnetic interference, the bus bars coming from the positive and negative DC switches should be near each other. The distance of the DC switches from each other, D_{dc}, is used as a proxy. The DC distribution bus runs along the top of the NIPEC. To reduce the routing distance, the DC switches should be placed as high as possible; thus, minimizing H_{dc} is desired. Ideally it is preferable for AC switches to be near each other, so D_{AC} is included in the fitness function.

Taking the product of all metrics enabled the algorithm to optimize all of the metrics at once. Another common practice is to take the sum of the individual metrics in the fitness function multiplied by their respective weights, but that process requires analysis of the relative scale of each of the parameters, making parameter optimization problem-specific.

b) Breeding: The genetic recombination process implemented is the Order-Based Crossover (OBX) operator, which is particularly well-suited for permutation-based problems. This crossover method preserves the relative order of elements from both parents while ensuring the production of valid offspring.

The OBX procedure begins with the random selection of a subset of positions from the first parent. The genes in these positions are copied directly to the child chromosome, maintaining their original positions. The remaining genes in the child chromosome are then filled based on the order of genes in the second parent, excluding those already present from the first parent.

Formally, let $P_1 = (p_{11}, p_{12}, ..., p_{1n})$ and $P_2 = (p_{21}, p_{22}, ..., p_{2n})$ be the two parent chromosomes. These parents are chosen probabilistically, weighted by their relative fitness. Let S be a randomly chosen continuous subarray of indices from 1, 2, ..., n. The child chromosome C is constructed by first setting C[i] = $P_1[i]$ for all i $\in$ S, then filling the remaining positions in C with the elements of P_2 in order, skipping elements already present in C.

c) Mutation: Following the Order-Based Crossover (OBX) operation, a mutation operator is applied with a predefined probability to introduce additional diversity into the population. The mutation process involves selecting a random subarray of the child chromosome and shifting it to the right by a random amount, but only up to the end of the chromosome.

This mutation can be more accurately described as follows:

Let $C = (c_1, c_2, ..., c_n)$ be the child chromosome after crossover. Let i and j be randomly chosen indices such that $1 \leq i < j \leq n$, defining the subarray to be shifted. Let k be a random integer where $1 \leq k \leq n - j + 1$, determining the amount of right shift.

The mutated chromosome C′ is then constructed as:
C′ = $(c_1, ..., c_{i-1}, c_{j+1}, ..., c_{j+k-1}, c_i, ..., c_j, c_{j+k}, ..., c_n)$ for $j + k \leq n$. This mutation operator shifts the selected subarray $(c - i, ..., c_j)$ k positions to the right, moving any intervening elements to the left to fill the gap. This mutation strategy allows for local rearrangements within the chromosome, potentially discovering new, beneficial orderings of genes. It provides a mechanism for fine-tuning solutions and exploring nearby regions of the solution space, complementing the more substantial changes that can occur through crossover.

d) Checkpoints: Due to the sensitivity of the problem, once the genetic algorithm generates a weak generation, it is difficult for the algorithm to return. Any slight mistake in an ordering can lead to a series of sub-optimal results. Though the randomized nature allows the algorithm to make sub-optimal movements to avoid local minima and reach local maxima and reach global ones, this does not always solve local minima difficulties. To avoid this, a checkpoint system is implemented. As the algorithm runs it saves the best individual it has seen during the entire process. If the algorithm goes through 100 generations without seeing an improvement, it will generate a new generation based on the best individual. This allows it to not only explore the search space, but to return to previous optima if it has gone too far in the wrong direction.

7) Simulated Annealing: Soke and Singul showed that genetic algorithms showed good results in finding-well packed solutions. In this application, although the solutions from the genetic algorithm were well packed, the wiring was suboptimal. To remedy this, Simulated Annealing was considered. During the simulated annealing portion of the algorithm, the packing locations are held constant; instead, only the ordering of the PEBBs is swapped within this framework, while leaving the switches in place. This means each placement will have the exact same volume, allowing us to consider only wiring distance in the energy function.

a) Energy Function:

$$E(\pi) = \alpha D_w + \beta D_{bb} \tag{6}$$

Two tunable parameters, α and β, are used to optimize for specific results. Increasing α produced more organized layouts.

b) Sorted Order: To enhance the efficiency of the simulated annealing process, a strategic pre-processing step is implemented that focuses on the arrangement of PEBBs. This approach involves sorting the PEBB placements prior to the annealing process, while maintaining the order of all other components. The sorting is done primarily by phase, and secondarily by the PEBB's position within its respective daisy-chain configuration.

This sorting strategy is designed to ensure that PEBBs that are physically connected via wiring are placed in sequential order. The rationale behind this approach is to provide the annealing process with a more favorable starting point, particularly in terms of wiring efficiency. Grouping connected PEBBs together in the initial configuration aims to reduce the likelihood of suboptimal wiring arrangements early in the optimization process. However, it is important to note that while this sorting provides a structured initial state, the stochastic nature of simulated annealing still allows for exploration of the full solution space, maintaining the algorithm's ability to escape local optima.

c) Exploration: The state transition mechanism in adopted in the simulated annealing algorithm is the same mutation function that was previously implemented in the genetic algorithm approach, described in Section III-B6c.

8) Post-Optimization: Upon closer examination, it becomes apparent that the initial placement algorithm does not inherently account for gravitational constraints, which are crucial in practical implementations. To address this limitation, gravity simulation was implemented. This step adjusts the vertical positions of the blocks to reflect realistic placement, thus ensuring no floating blocks.

Secondly, to facilitate the organization of blocks into well-defined racks or columns, a guillotine cutting algorithm was employed. This process is crucial for applications where equipment must be arranged in distinct, vertically aligned groups. The algorithm proceeds as follows:

Blocks are initially sorted based on their left-edge coordinates. The algorithm identifies the largest block with the minimum left-edge coordinate, using its width to define the width of the rack. Within this defined width, the algorithm packs as many blocks as possible, adhering to the rack's dimensional constraints; thus, if a block is less than half the width of another, you may end up with two blocks side-by-side in a rack. Any blocks that cannot fit within the current rack are allocated to subsequent racks. This process is iteratively applied until all blocks are assigned to racks, resulting in a columnar organization of the placement.

IV. Parameter Exploration

A. Evaluation Methodology

The parameter optimization process relied on dual evaluation criteria. The primary quantitative metrics were total volume and routing distance minimization, while qualitative assessment was performed through visual inspection of component placement. This comprehensive evaluation approach ensured that the solutions were not only mathematically optimal but also practically implementable in real-world scenarios.

B. Experimental Process

The algorithm's performance parameters were systematically refined through extensive testing on representative test cases. The optimization process focused on configurations featuring 1-4 AC switches, 2-3 distinct paths to each AC switch, and PEBB chains of lengths 2-4. This balanced test environment provided sufficient complexity to evaluate parameter effectiveness while remaining computationally manageable. Multiple iterations with varying parameter combinations were executed to identify optimal weightings. The final parameter values emerged from this methodical tuning process, demonstrating superior performance in minimizing component distances, optimizing chain placements, and maintaining efficient electrical connections. Final values are shown in Table I for the genetic algorithm and Table II for the simulated annealing algorithm. This calibration approach ensured the algorithm could effectively handle the spatial constraints and connectivity requirements typical in electrical distribution systems, while maintaining reasonable computational efficiency.

Fig. 8: A mutation rate of 0.75 produced the highest peak results (red line). This example used chains of four PEBBs with two chains in parallel for each of four AC phases.

TABLE I: Genetic algorithm parameters.

Population Size	100
Number of Generations	800
Mutation Probability	0.75
Busbar Routing Distance	3
Wire Routing Distance	5
DC Block Distance	3
DC Height	1
AC Distance	3
Volume	1
PEBB/Switch Overlap	6.734
Switch Width	1

TABLE II: Simulated annealing parameters.

Iterations per Temperature	400
Start Temperature	50
End Temperature	1
Cooling Rate	0.992

C. Genetic Algorithm Parameters

A genetic algorithm's performance is primarily influenced by population size, generation count, mutation probability, and a fitness function. While the experiments extended to 1500 generations, the additional computational cost beyond 800 generations did not yield significant improvements in the solution quality. A population size of 100 individuals was maintained, which proved sufficient to maintain genetic diversity while keeping memory requirements manageable. As shown in Figure 8, a mutation rate of 0.75 provided the best results. This value helped prevent the algorithm from becoming too aggressive in its mutations while still maintaining the ability to escape local optima.

The parameters with the most impact on the results are the fitness weights. When solutions showed a bias toward certain characteristics, fine-tuning the parameters helped achieve more balanced results. This suggests that while generally optimal parameters were identified, slight adjustments might be beneficial for specific use cases or requirements.

D. Simulated Annealing Parameters

The SA component's performance was optimized through careful tuning of three critical parameters. A starting temperature of 50 was established, which was calibrated based on the quality of initial solutions from the GA phase. This higher starting temperature allowed for sufficient exploration of the solution space in the early stages. The final temperature was set to 1, and a cooling rate of 0.992 was implemented. This cooling rate was selected to provide a gradual decrease in temperature, allowing the algorithm to thoroughly explore promising regions of the solution space while still maintaining the ability to escape local optima.

E. Fitness Function Exploration

To verify the impact of various parameters, we present a comparative analysis of the impact of adjusting fitness function weights on the final layout solutions. The experiments focused on three configurations: a baseline algorithm with balanced parameters, a variant with reduced DC height weighting, and a variant with minimal AC proximity weighting. Results can be seen in Table III and Figure 9.

TABLE III: Metric values for comparison of weighting parameter impact

	Baseline	No DC	No AC
DC Height Weight β	1	0	1
AC Distance Weight ϵ	3	3	0
Busbar Routing Distance	216	370	365
Wire Routing Distance	157	126	178
DC Block Distance	12.25	12.25	12.25
DC Height	0.80	0.15	0.64
AC Distance	173.28	144	484
Volume	1193	2257	2257
PEBB/Switch Overlap	0.09	0.09	0.09
Switch Width	9.8	9.8	9.8

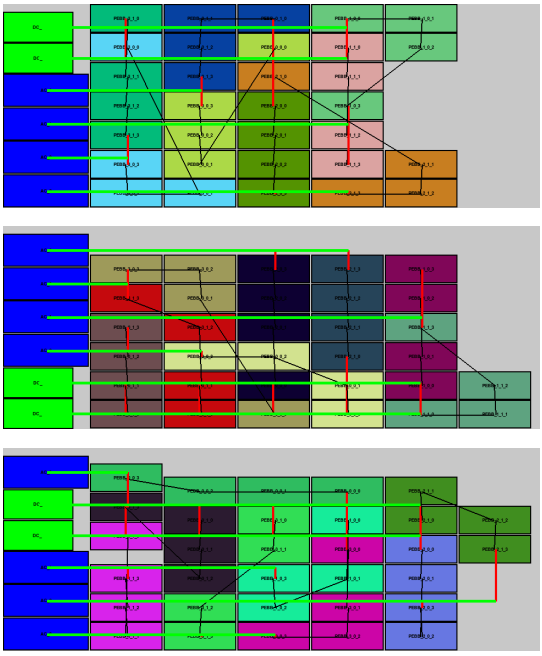

Fig. 9: Comparison of weighting parameter impact: baseline (top), DC height removed (middle), AC distance removed (bottom)

Each example was calculated using a population size of 100, 800 generations, and a simulated annealing starting temperature of 50 with a cooling rate of 0.992.

The baseline algorithm configuration with balanced parameter weights is shown in the top image of Figure 9. The resulting layout demonstrates a reasonable compromise between all optimization criteria, with components arranged to balance DC bus height, AC component proximity, and overall system volume. The baseline configuration's fitness function incorporated multiple weighted parameters, with each contributing proportionally to guide the optimization process. This balanced approach served as the reference point for subsequent parameter adjustments.

The middle image in Figure 9 demonstrates the effect of reducing the weight assigned to DC bus height in the fitness function. By decreasing the weight parameter for DC height, a change in component organization is observed. The algorithm produced a layout with DC blocks at the very bottom.

The third configuration, shown in the bottom image in Figure 9, demonstrates the impact of minimizing the AC proximity parameter on component placement. With reduced emphasis on AC component proximity, the algorithm generated a layout in which AC components were more distributed throughout the available space.

V. Conclusions

This project presented a novel hybrid optimization approach for the automated placement of electrical components within modular integrated power corridors of naval vessels. By combining the global exploration capabilities of Genetic Algorithms with the local search precision of Simulated Annealing, the developed method successfully addresses the complex challenge of optimizing component placement while satisfying multiple objectives and constraints.

The proposed algorithm demonstrated significant advantages over traditional manual placement methods, particularly in routing efficiency. By systematically optimizing the arrangement of DC switches, Power Electronic Building Blocks (PEBBs), and AC switches, the algorithm consistently produced well-organized layouts that matched or exceeded the quality of hand-designed solutions. The ability to generate these optimized placements in approximately one minute represents a substantial improvement over the time-intensive manual design process, enabling rapid design space exploration during early-stage ship design.

The effectiveness of the hybrid approach was explored through parameter tuning and testing. The optimal parameter configuration (population size of 100, 800 generations, mutation probability of 0.75, with specific weightings for various objectives) provided a balanced optimization framework that prioritized critical aspects of component placement while maintaining reasonable computational efficiency. The implementation of strategic features such as the modified Bottom-Left placement algorithm, checkpoint systems, and post-optimization processes (gravity simulation and guillotine cutting) further enhanced the practical applicability of the solution.

As expected with meta-heuristic approaches, the algorithm's performance exhibited some variability due to its stochastic nature, occasionally producing suboptimal results. This limitation was addressed by increasing the number of generations proportionally to the problem complexity, though multiple test runs may still be beneficial in practical implementation scenarios. Additionally, the algorithm's performance predictably decreased as the number of components increased, reflecting the exponential growth of the search space.

The primary contribution of this work lies in providing naval engineers with a powerful tool for quickly generating accurate design approximations of electrical component placements within power corridors. This capability will significantly enhance the efficiency of the ship design process, allowing for more comprehensive exploration of design alternatives and potentially leading to more optimized vessel configurations.

Future research directions could expand upon this foundation in several ways. The incorporation of thermal considerations would add another critical dimension to the optimization process, accounting for heat dissipation and cooling requirements. Additionally, extending the algorithm to accommodate new types of components would increase its versatility and applicability across a broader range of naval power system configurations. Further refinement of the algorithm's handling of larger component sets could also improve its scalability for more complex power corridor designs.

In conclusion, this research demonstrates the effectiveness of hybrid meta-heuristic approaches in solving the complex optimization problem of electrical component placement in

naval power corridors. The developed algorithm provides a practical, efficient solution that balances multiple competing objectives while maintaining the essential constraints of the system.

ACKNOWLEDGMENTS

The authors would like to express thanks to Prof. Robert Cuzner of the University of Wisconsin-Milwaukee and Prof. Herbert Ginn of the University of South Carolina for valuable discussions regarding the approach for PEBB and switch connectivity. These discussions were the genesis of Figure 4.

This work is supported by the Office of Naval Research Grant No. N00014-21-1-2124, and by the National Oceanic and Atmospheric Administration (NOAA) Grant No. NA22OAR4170126.

REFERENCES

[1] C. M. Cooke, C. Chryssostomidis, and J. Chalfant, "Modular integrated power corridor," in *2017 IEEE Electric Ship Technologies Symposium (ESTS)*, 2017, pp. 91–95.

[2] B. Guo, Y. Zhang, J. Hu, J. Li, F. Wu, Q. Peng, and Q. Zhang, "Two-dimensional irregular packing problems: A review," *Frontiers in Mechanical Engineering*, vol. 8, 2022.

[3] O. Kramer, *Genetic Algorithm Essentials*, 1st ed. Springer Publishing Company, Incorporated, 2017.

[4] D. Henderson, S. H. Jacobson, and A. W. Johnson, *The Theory and Practice of Simulated Annealing*. Boston, MA: Springer US, 2003, pp. 287–319.

[5] A. Lodi, S. Martello, and D. Vigo, "Recent advances on two-dimensional bin packing problems," *Discrete Applied Mathematics*, vol. 123, no. 1, pp. 379–396, 2002.

[6] ——, *Neighborhood Search Algorithm for the Guillotine Non-Oriented Two-Dimensional Bin Packing Problem*. Boston, MA: Springer US, 1999, pp. 125–139.

[7] ——, "Approximation algorithms for the oriented two-dimensional bin packing problem," *European Journal of Operational Research*, vol. 112, no. 1, pp. 158–166, January 1999.

[8] F. R. K. Chung, M. R. Garey, and D. S. Johnson, "On packing two-dimensional bins," *SIAM Journal on Algebraic Discrete Methods*, vol. 3, no. 1, pp. 66–76, 1982.

[9] X. Liao, S. Guo, and C. Ou, "An physical force-driven packing optimization design method in strip packing problems," in *Proceedings of the 2016 International Forum on Energy, Environment and Sustainable Development*. Atlantis Press, 2016/05, pp. 483–487.

[10] S. Chattopadhyay, D. W. Bouldin, and P. H. Dehkordi, *An Overview of Placement and Routing Algorithms for Multi-chip Modules*, pp. 3–23.

[11] A. Soke and Z. Bingul, "Hybrid genetic algorithm and simulated annealing for two-dimensional non-guillotine rectangular packing problems," *Engineering Applications of Artificial Intelligence*, vol. 19, no. 5, pp. 557–567, 2006.

[12] R. Cuzner and H. Ginn, "Mini-PEPDS example arrangements," 2024.