Decoupling Optimization for Complex PDN Structures Using Deep Reinforcement Learning

Ling Zhang, *Member, IEEE*, Li Jiang, *Student Member, IEEE*, Jack Juang, *Student Member, IEEE*, Zhiping Yang, *Fellow, IEEE*, Er-Ping Li, *Fellow, IEEE*, and Chulsoon Hwang, *Senior Member, IEEE*

***Abstract*— This article presents a new optimization method for complex power distribution networks (PDNs) with irregular shapes and multilayer structures using deep reinforcement learning (DRL), which has not been considered before. A fast boundary integration method is applied to compute the impedance matrix of a PDN structure. Subsequently, a new DRL algorithm based on proximal policy optimization (PPO) is proposed to optimize the decoupling capacitor (decap) placement by minimizing the number of decaps while satisfying the desired target impedance. In the proposed approach, the PDN structure information is encoded into matrices and serves as the input of the DRL algorithm, which increases the flexibility of the method to be extended and generalized to different PDN configurations. Also, the output of the algorithm determines the selection of decap types and locations collaboratively, making it easier to find the optimal solution in a huge search space. The proposed method is compared with the state-of-the-art approaches and shows consistent advantages in reducing the number of decaps in different testing cases.**

***Index Terms*— Boundary integration, decoupling capacitor (decap), deep reinforcement learning (DRL), machine learning, power distribution network (PDN).**

I. Introduction

DECOUPLING capacitors (decaps) play a significant role in lowering the alternating current (ac) impedance and reducing the consequent transient voltage noise in power distribution networks (PDNs) [1]. Optimizing the locations and types of decaps for complex printed circuit boards (PCBs) to minimize the number of decaps is desired but troublesome, because the impedance simulation is usually time-consuming, and the search space for decap optimization is commonly enormous. As power integrity is increasingly becoming a bottleneck for modern integrated circuits (ICs) with lower and lower power supply voltages, various decap optimization methods have been proposed and investigated in recent years.

Optimization algorithms, such as the genetic algorithm (GA) [2], [3], [4], [5], [6] and particle swarm optimization (PSO) algorithm [7], [8], were adopted for decap optimization. GA was employed in [2] and [3] to optimize decaps to satisfy a target impedance, but the cost functions do not ensure the minimum number of decaps. Cecchetti et al. [4] and de Paulis et al. [5] proposed a GA by adding decaps sequentially and selecting the best decap type and location by minimizing the area where the PDN impedance violates the target impedance for each new decap placement. However, this method might miss the optimal solution with the minimum number of decaps, since different decaps may contribute collaboratively rather than individually. A GA was proposed to minimize the number of decaps by considering the collaborative contribution of decaps and defining an objective function associated with the number of placed decaps to satisfy a given target impedance [6], but the method might need a considerably long time to converge to the optimal solution. Besides, the PSO algorithms [7], [8] do not exhibit clear advantages over GA regarding the solution quality and speed. Thus, the biggest challenge with these optimization algorithms is that the optimum solution cannot be guaranteed, and the optimization time is problematic for a large number of decap locations.

There have also been some other optimization methods that are worthwhile noticing [9], [10], [11]. A physics-based approach was developed [9] with an iterative procedure and complex decision-making conditions. Han et al. [10] recently proposed a knowledge-based method and claimed that their method could find the optimized decap design using much fewer PDN simulations than other methods. Moreover, a Newton–Hessian minimization method was proposed for the fast optimization of decap locations [11]. However, the capability of these methods [9], [10], [11] to find the optimal solution for complex PDN scenarios was not demonstrated.

Manuscript received 12 January 2023; accepted 14 February 2023. Date of publication 3 March 2023; date of current version 5 September 2023. The work of Ling Zhang, Li Jiang, and Er-Ping Li was supported in part by the Fellowship of China Postdoctoral Science Foundation under Grant 2022M722718; in part by the Postdoctoral Science Preferential Funding of Zhejiang Province, China, under Grant ZJ2022071; in part by the Zhejiang Provincial Natural Science Foundation of China (ZPNSFC) under Grant LQ23F010020 and Grant LD21F010002; and in part by the Natural Science Foundation of China (NSFC) under Grant 62071424 and Grant 62027805. The work of Jack Juang and Chulsoon Hwang was supported in part by the National Science Foundation under Grant IIP-1916535 and in part by the Google Faculty Research Award. *(Corresponding authors: Er-Ping Li; Chulsoon Hwang.)*

Ling Zhang is with the College of Information Science and Electronic Engineering, Zhejiang University, Hangzhou 310027, China, and also with the Electromagnetic Compatibility Laboratory, Missouri University of Science and Technology, Rolla, MO 65409 USA (e-mail: lingzhang_zju@zju.edu.cn).

Li Jiang and Er-Ping Li are with the College of Information Science and Electronic Engineering, Zhejiang University, Hangzhou 310027, China (e-mail: lijiang_zjuisee@zju.edu.cn; liep@zju.edu.cn).

Jack Juang and Chulsoon Hwang are with the Electromagnetic Compatibility Laboratory, Missouri University of Science and Technology, Rolla, MO 65409 USA (e-mail: jjryb@mst.edu; hwangc@mst.edu).

Zhiping Yang is with Google Inc., Mountain View, CA 94043 USA (e-mail: zhipingyang@google.com).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TMTT.2023.3248237.

Digital Object Identifier 10.1109/TMTT.2023.3248237

Deep reinforcement learning (DRL) has been attracting growing interest due to its tremendous success in handling various complicated tasks [12], [13]. Therefore, researchers have attempted to optimize decaps using DRL [14], [15], [16], [17], [18] in recent years. Park et al. [14], [15] adopted DRL for decap placement on silicon interposer-based ICs and high bandwidth memory [16]. However, in their application scenarios, the silicon interposer has a rectangular shape, and there is only one decap type in the decap library. This makes it difficult to apply their approach to complex PDN structures with irregular shapes and multiple decaps types. Another DRL algorithm was proposed to optimize decap selection given the S-parameters of a PDN system [17], [18], but the algorithm does not consider the geometry information of the PDN system. Moreover, in [17] and [18], the decap locations and types are separately optimized. The priority of decap locations is determined by calculating the physical inductances; then, DRL is applied to select the appropriate decap types to be sequentially placed on the prioritized decap locations. The capability of the proposed algorithm in [17] and [18] to converge to high-quality solutions with fewer decaps than other methods was not demonstrated.

This article presents a new DRL algorithm that considers the PDN information in the algorithm input and can handle complex PDN structures. The impedance matrix of a complex PDN structure is rapidly calculated using a boundary integration method [19], [20]. Subsequently, the decap optimization is accomplished through an automatic self-exploration process using DRL. The proposed algorithm, which considers the collaborative contribution of different decaps, is compared with the GA [5], [6] and the Newton–Hessian minimization method [11], and demonstrates efficient and more reliable convergence to the optimal solutions in different test cases. Also, the algorithm output determines the selection of both decap locations and decap types and is more expandable than the method in [14] and [16] that only supports one decap type. Moreover, the proposed method incorporates the PCB shape, PCB stack up, IC location, and decap locations into the algorithm input, which enhances the flexibility of the algorithm to be extended and generalized to various PDN configurations compared with the existing DRL approaches [14], [15], [16], [17], [18].

The rest of this article is organized as follows. The boundary integration approach for impedance calculation and the other methods for decap optimization are briefly introduced in Section II. The proposed DRL method is elaborated in Section III and validated through several test cases in Section IV. Finally, a conclusion is drawn in Section V.

II. Problem Description

The boundary integration method [20] used for the impedance calculation of complex PDN systems is briefly reviewed in this section. Also, the mathematical formulations of the decap optimization scenario studied in this article are clarified. Moreover, the core concepts of several other decap optimization methods [5], [6], [11] are briefly summarized.

A. Impedance Calculation

The approach proposed in [20] can handle irregular PCB shapes by approximating the PCB boundary into a perfect magnetic conductor (PMC), dividing the boundary into segments, and calculating the quasi-static inductances through 1-D boundary integration. For multilayer PDN structures, an equivalent circuit is constructed and rapidly solved using a matrix reduction strategy [20]. Eventually, a Z-parameter matrix containing the IC and decap ports can be efficiently computed. The boundary integration method in [20] can tackle complex PCB shapes, which is more potent than the cavity model method [21] for rectangular shapes only and much more efficient than the plane-pair PEEC (PPP) approach [22] for complex board geometries. The details of the method are explained in [20].

A segmentation method [23] is adopted to evaluate the PDN impedance after placing decaps. Since the Z-parameter matrix of the PCB board with the decap ports can be calculated through the boundary integration method, the total impedance after placing decaps can be obtained by connecting the Z-parameter matrices of the PCB board and the decaps. This process is much more efficient than repeatedly solving the entire equivalent circuit during the decap optimization process. The details of the segmentation method for decap connection can be found in [18] and [23].

B. Decap Optimization

The available decap locations are treated as discrete locations in this article. In [6] and [11], the decap locations are considered as continuous variables to be optimized. However, this is not applicable when the placement of decaps needs to compromise with other PCB components.

Therefore, in this article, we assume that there are N available decap locations represented by variables $L_1, L_2, \ldots, L_N$ and M decap types represented by $C_1, C_2, \ldots, C_M$. The number of available decap locations N is determined by users and depends on several factors, including the available layout areas on a PCB and the difficulty of satisfying the target impedance (the lower the target impedance is, the more the decaps are required). Each decap location can be regarded as a variable with discrete values

$$L_i = 0, C_1, C_2, \ldots, C_M \quad (i = 1, 2, \ldots, N) \tag{1}$$

where $L_i = 0$ means that no decap is placed on the location L_i.

In the industry, minimizing the number of decaps to save cost while satisfying a target impedance is pursued. Thus, the decap selection is converted to an optimization problem to achieve this objective. Namely, the cost function of the optimization process becomes

$$\begin{aligned} &\underset{L_1, L_2, \ldots, L_N}{\arg\min} \; N_{\text{decap}} \\ &\text{s.t. } Z_{\text{pdn}}(f) \leq Z_{\text{target}}(f) \end{aligned} \tag{2}$$

where N_{decap} represents the number of used decaps, namely, the number of nonempty decap locations.

Since the available decap positions are predefined in discrete locations, an impedance Z-parameter matrix can be calculated using the boundary integration method [20] by treating each candidate decap location as a Z-parameter port. Afterward, this Z-parameter matrix can be repeatedly utilized to connect with different decaps for optimization purposes by adopting the segmentation method [18], [23], which is highly time-efficient.

C. Other Decap Optimization Methods

The GAs proposed in [5] and [6] for decap optimization are denoted as full search and GA, respectively. The Newton–Hessian minimization method [11] is called the Newton–Hessian method for simplicity. These methods are slightly modified for more straightforward implementation and fair comparison with the proposed DRL method in this article, but the core concepts of the three methods are maintained. The details are explained as follows.

1) Full Search Method [5]: The full search method proposed in [5] decomposes the decap optimization problem into placing the decaps sequentially. In each decap selection, GA is used to search and select the best decap location and decap type that can minimize the areas where the PDN impedance violates the target impedance. The original optimization problem has $(M+1)^N$ possible cases in the search space, which exponentially increase with the number of decap locations and decap types. The full search method reduces this large search space into a much smaller search space with a maximum number of nearly MN^2 possible cases. Even though the optimization speed of the full search method is fast, it is hard to find the optimal solution, since it optimizes the selection and placement of each decap individually without considering the collaborative contribution of different decaps. In other words, even though the full search method is fast, it might miss the optimal solution, since it explores a subset of the whole search space.

The full search method utilizes a GA to optimize the decap location and decap type for adding each decap. Since the maximum number of possible cases using the full search method is only approximately MN^2, full search is also feasible and expected to behave better than or at least the same as using GA. By using the boundary integration method [20] and the segmentation method [18], [23], the full search is still acceptably time-efficient. Therefore, the full search strategy is adopted in this article to reproduce the concept of the full search method.

2) GA Method [6]: Another GA was proposed in the GA method to minimize the number of decaps while satisfying a target impedance or power supply induced jitter. Since this article focuses on optimizing PDN frequency-domain impedance, the method developed in [6] to satisfy a target impedance is reproduced and compared with the proposed DRL method. Unlike the full search method, which sequentially and individually optimizes the selection of each decap, the GA method optimizes the decap selection at different locations simultaneously. Also, the GA method considers the decap coordinates as continuous variables to be optimized. As mentioned earlier, the proposed DRL method considers discrete decap locations. Therefore, in this article, the objective function of the GA method is modified as follows:

$$\begin{aligned} &\underset{L_1,L_2,\ldots,L_N}{\arg\min}\ \left[\max\left(Z_{\text{pdn}}(f)\right)\cdot N_{\text{decap}}\right] \\ &\text{s.t. } Z_{\text{pdn}}(f)\le Z_{\text{target}}(f). \end{aligned} \tag{3}$$

Since the GA method optimizes the decap selection at different locations with a collaborative contribution to the impedance, the GA method is expected to find better solutions than the full search method, given enough exploration time. However, the GA method explores a search space with $(M+1)^N$ possible cases, which might cause a much longer time to find the optimal solution.

3) Newton–Hessian Method [11]: The Newton–Hessian method also regards decap locations as continuous variables and adopts the Newton–Hessian minimization method to optimize the decap locations to minimize the maximum impedance value. In the Newton–Hessian method, the decaps are placed sequentially, and the decap types and decap locations are optimized separately. The selection and placement of each decap are divided into two steps. First, the decap type with a resonance frequency closest to the frequency where the maximum PDN impedance occurs is selected. Second, the Newton–Hessian minimization method is applied to minimize the maximum PDN impedance by using the selected decap type of the first step and optimizing the decap location (x and y values). Thus, the decaps are added to the PDN sequentially until the target impedance is satisfied.

To compare the Newton–Hessian method with the proposed DRL method in this article, we modified the original method in [11] by assuming discrete decap locations. Namely, in the Newton–Hessian method, instead of optimizing the decap locations using the Newton–Hessian method, the best decap location is determined by inspecting all the discrete decap locations and finding the location that can minimize the maximum PDN impedance.

The Newton–Hessian method has a similar problem as the full search method in finding the optimal solution, since the Newton–Hessian method also optimizes the selection and placement of each decap individually and sequentially. In the full search method, the type and location for each decap are optimized simultaneously. However, in the Newton–Hessian method, the best decap type is selected first, and then, the best decap location is determined. Therefore, it is expected that the Newton–Hessian method behaves worse than the full search method, since the Newton–Hessian method explores a subset of the search space in the full search method.

It is worth mentioning that the Newton–Hessian method is based on a flat PDN target impedance, and the maximum PDN impedance serves as the criterion for selecting the best decap types and locations. Therefore, a flat target impedance will be used to compare the Newton–Hessian method and the proposed DRL method. The full search and GA methods allow using an arbitrary-shape target impedance. Hence, a typical R–L shape target impedance will be used when comparing the full search method and the GA method with the proposed DRL approach, which will be shown in Section IV.

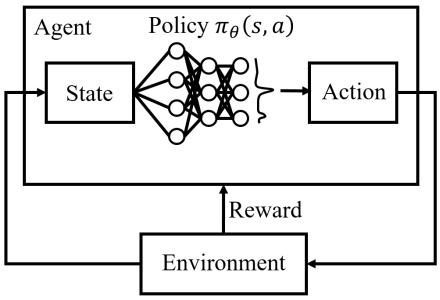


Fig. 1. Basic procedure of a DRL algorithm.

III. Proposed Method

The proposed DRL approach, including the detailed DRL algorithm, input and output matrix definition, deep neural network (DNN) structure, and reward definition, is elaborated in this section.

A. Brief Introduction of DRL

Reinforcement learning (RL) is a machine learning algorithm that trains an intelligent agent to make beneficial decisions in a dynamic environment. In DRL [12], [13], DNNs are embedded in the RL algorithms to enhance the capability of tackling complex environments. The basic procedure of a DRL algorithm is depicted in Fig. 1.

Several critical elements of DRL are state, action, and reward. State s is a situation or configuration the agent observes from the environment. Action a is the movement that the agent takes in response to the observed state. Reward r is the feedback from the environment by taking the corresponding actions. Policy $\pi_\theta(s,a)$ maps the states to the actions and selects the actions that promise the highest reward. In DRL, the policy $\pi_\theta(s,a)$ is represented by a DNN. The agent continuously interacts with the environment by observing the state, taking an action, obtaining a reward, observing the next state, and so on. In such an iterative exploration process, the parameter θ of the DNN policy is trained through the exploration experiences, such that the actions with higher rewards have a higher priority of being selected. The goal is to optimize the policy after training, so the agent can automatically take a sequence of actions in the environment with a maximized reward.

B. PPO Algorithm

The policy-gradient algorithm [24] is a popular RL technique that has been applied to a wide range of problems. The output of the policy network in policy-gradient algorithms is the probability prediction for different actions. The general objective function of policy-gradient algorithms is

$$\text{Maximize}\hat{E}_t[\ln \pi_\theta(a_t|s_t) R_t] \tag{4}$$

where $\ln\pi_\theta(a_t|s_t)$ is the log probability for action a_t under state s_t, R_t is the discounted reward, and $\hat{E}_t$ represents the mathematical expectation value. This objective function encourages the algorithm to increase the probabilities for the actions with high rewards.

One possible problem with the objective function (4) is that when R_t is always positive, the probability $\pi_\theta(a_t|s_t)$ may be raised up even if a_t is a bad action, which increases instability of the training. One common trick to mitigate this issue is to subtract a baseline value function V_t, an estimation of the discounted reward from the current state onward, from the discounted reward R_t [25]. Hence, besides the policy network predicting the probabilities of different possible actions, a value function neural network outputting the value function V_t of the input state also needs to be trained as an estimation of the discounted reward starting from the input state.

Another issue with the objective function (4) is that the policy may move far away from the old policy in the exploration, which may result in a locally bad policy. The proximal policy optimization (PPO) algorithm was proposed in 2017 [26] to mitigate this problem by adding constraints to the parameter change of the policy and networks. Implicitly, a clipping objective function is defined as follows:

$$L^{\text{CLIP}}(\theta) = \hat{E}_t\left\{\min\left[r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\varepsilon, 1+\varepsilon)\hat{A}_t\right]\right\} \tag{5}$$

where $r_t(\theta)$ is the ratio of the output probabilities to the old output probabilities

$$r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)} \tag{6}$$

and the clipping function $\text{clip}(r_t(\theta), 1-\varepsilon, 1+\varepsilon)$ constrains the network parameter change $r_t(\theta)$ between $1-\varepsilon$ and $1+\varepsilon$, where ε is set to 0.2 as recommended in [26]. $\hat{A}_t$ is called the advantage function. When $\hat{A}_t$ is positive, the clipping objective function (5) prevents the action probability from being overly increased; when $\hat{A}_t$ is negative, the corresponding action probability is inhibited from being overly decreased. Therefore, the PPO approach can prevent large policy updates and ensure a more stabilized convergence.

A generalized advantage estimation (GAE) was proposed in [26] that includes the summation of discounted rewards and value functions in the subsequent steps

$$\hat{A}_t = \delta_t + (\gamma\lambda)\delta_{t+1} + \cdots + (\gamma\lambda)^{T-t+1}\delta_{T-1} \tag{7}$$

where

$$\delta_t = r_t + \gamma V_\theta(s_{t+1}) - V_\theta(s_t)$$

where T is the total number of steps; r_t is the reward of action a_t under state s_t; γ is a discount factor; λ is a smoothing parameter; $V_\theta(s_t)$ and $V_\theta(s_{t+1})$ are the predicted values for states s_t and s_{t+1} from the value network, respectively.

According to [26], the total objective function of the PPO algorithm is expressed as follows:

$$L_t^{\text{CLIP+VF+}S}(\theta) = \hat{E}_t\left[L_t^{\text{CLIP}}(\theta) - c_1 L_t^{\text{VF}} + c_2 S[\pi_\theta](s_t)\right] \tag{8}$$

where $L_t^{\text{CLIP}}(\theta)$ is defined in (5), and L_t^{VF} is the square error loss of the value network output

$$L_t^{\text{VF}} = \left[V_\theta(s_t) - \hat{A}_t\right]^2. \tag{9}$$

$S[\pi_\theta](s_t)$ denotes the entropy of the output probability to enhance exploration, and c_1 and c_2 represent two coefficients. In this article, c_1 is 0.5, and c_2 is 0.01.

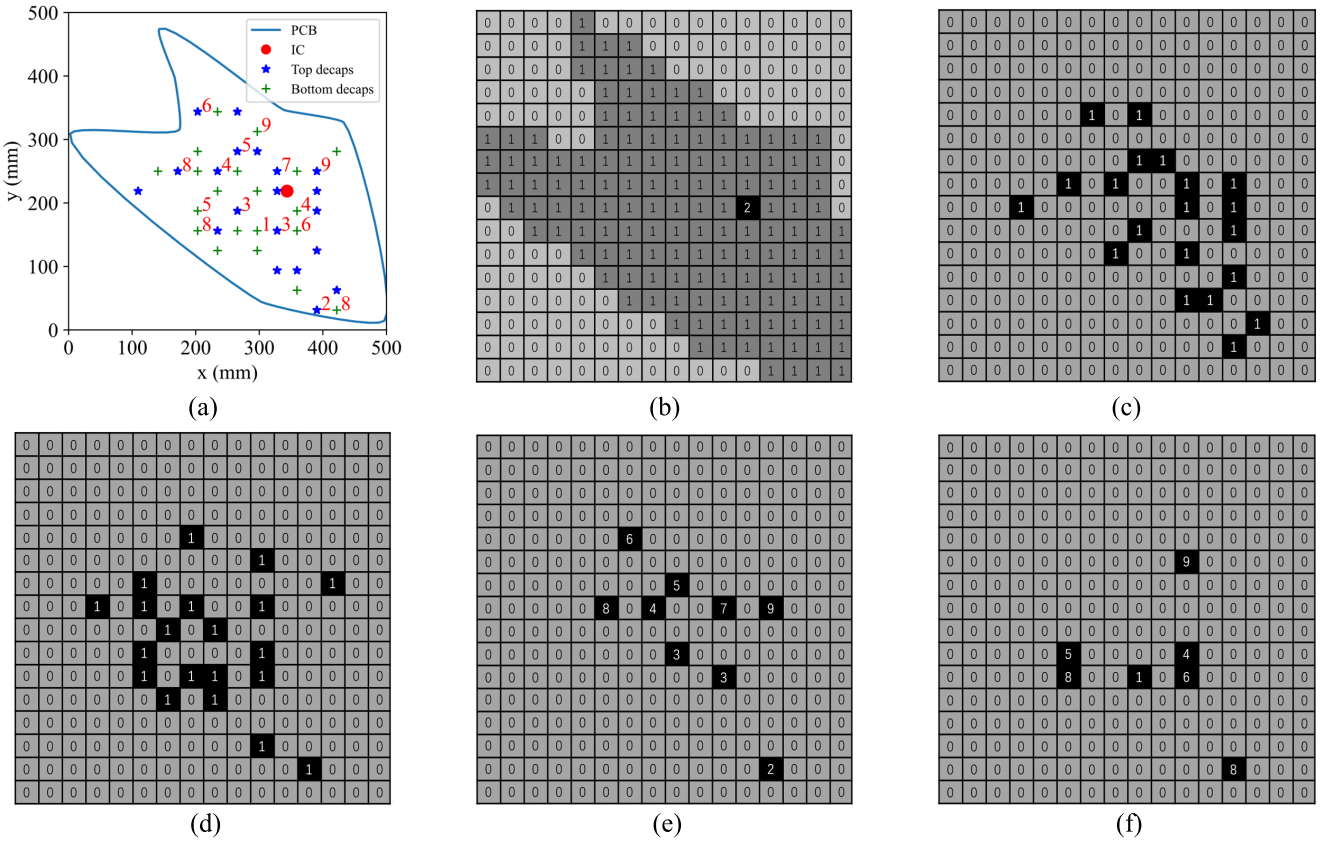


Fig. 2. Encoding the board shape, IC location, and decap locations into 2-D matrices. (a) PCB information, including the PCB shape, IC location, decap locations, and placed decaps represented by different numbers. Matrix representation of (b) PCB shape and IC location, (c) available top decap locations, (d) available bottom decap locations, (e) placed top decaps, and (f) placed bottom decaps.

C. Matrix Definition and DNN Structure

As stated in Section I, this article considers the PDN structure information in the algorithm input, including PCB shapes, stack up, IC location, and decap locations. The PCB shape, IC location, and decap locations on the top and bottom layers are encoded into 2-D matrices, as illustrated in Fig. 2. A maximum board area of 200 × 200 mm is defined, which is enough for most decap placement applications. Fig. 2(a) shows a PDN example with an irregular board shape, 20 decap locations on the top and bottom layers, respectively, and some decaps placed on the available locations. The decap values are denoted by numbers ranging from 1 to 10, representing ten different decap types. The R–L–C parameters for these decap types can be found in [27]. The board shape and the IC location are encoded into a 2-D matrix, e.g., a 16 × 16 matrix, as shown in Fig. 2(b). This matrix size can be adjusted if a different decap placement density is needed. If the required decap density is high, the matrix size needs to be appropriately enlarged. But on the other hand, enlarging the matrix size will slow down the training speed and increase the computational burden. Hence, a trade-off is needed to determine the optimal size of the 2-D matrices. The available decap locations on the top and bottom layers are also encoded into 2-D matrices, as shown in Fig. 2(c) and (d). The 2-D matrices in Fig. 2(e) and (f) illustrate the partial decap placement on the top and bottom, representing the decaps that have been placed. These two 2-D matrices are updated once new decaps are added to the board. Similar to the strategy in [27], the PCB stack up is encoded into a 1-D matrix, as shown in Fig. 3. A maximum number of nine layers is assumed in this article. Hence, the PCB stack-up information is encoded into a 1 × 17 matrix, where the first nine elements represent the layer type, and the last eight elements represent the dielectric thickness. This matrix size can also be enlarged if more layers are needed.

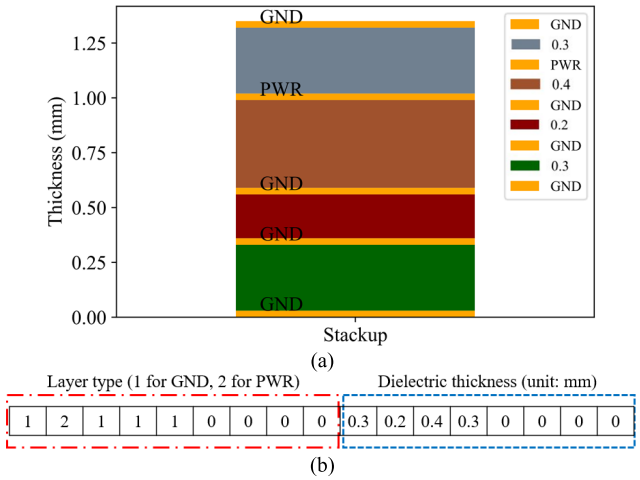


Fig. 3. (a) PCB stack-up example. (b) One-dimensional matrix representation of the stack up.

The DNN structure employed for the proposed DRL algorithm is depicted in Fig. 4. Besides the PDN structure

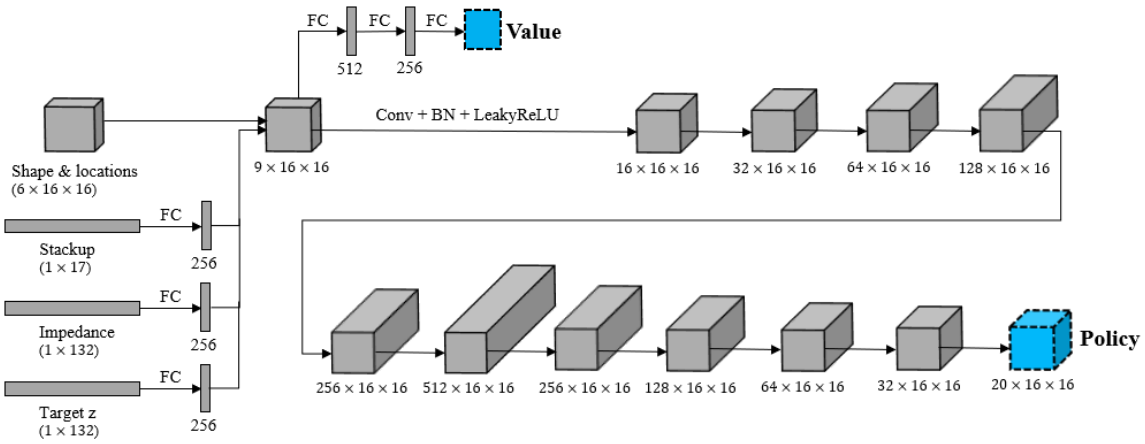


Fig. 4. DNN structure for the policy and value networks used in the proposed DRL approach.

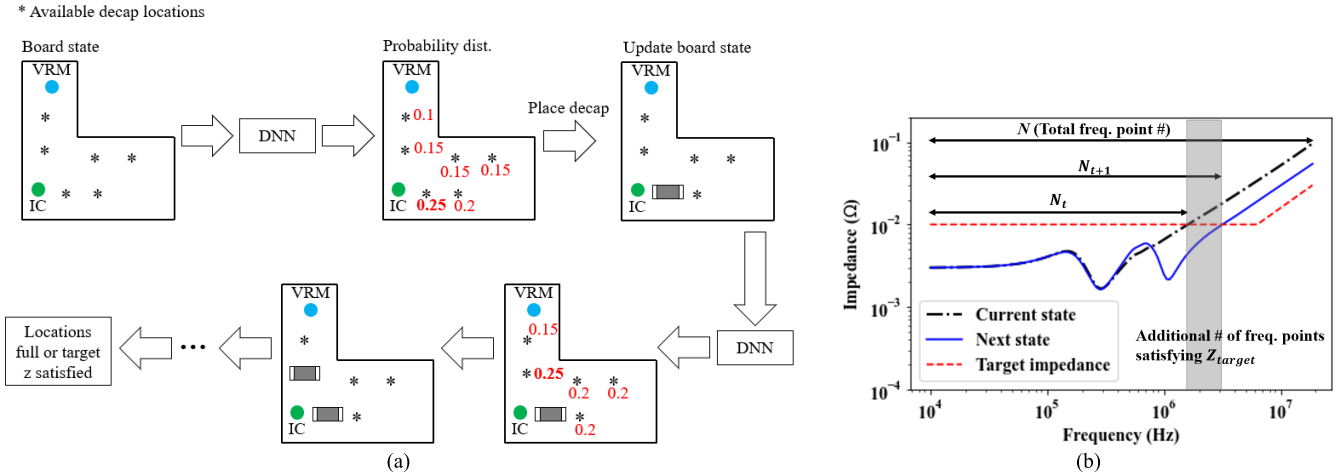


Fig. 5. Illustration of (a) placing decaps sequentially according to the output probability distribution of the policy network and (b) reward definition.

information, the impedance and the target impedance are also included in the input state matrices as 1-D matrices. The impedance needs to be updated if new decaps are placed and the impedance changes. As shown in Fig. 4, the board shape, IC location, and decap locations are represented by a 5 × 16 × 16 matrix, while the other input parameters are 1-D matrices. To combine all the input parameters into one single matrix, each input 1-D matrix is converted to a 1 × 256 matrix through a fully connected (FC) layer, reshaped into a 16 × 16 matrix, and then cascaded with the 5 × 16 × 16 matrix. Consequently, an 8 × 16 × 16 matrix is constructed and followed by a series of convolutional layers [28]. In each convolutional layer, batch normalization [29] and the LeakyReLU activation function [30] are adopted.

The policy and value networks are embedded in the DNN shown in Fig. 4. The output after the convolutional layers is a 20 × 16 × 16 matrix followed by a Softmax function [31], so that the output is converted to probabilities between 0 and 1 representing different action probabilities, which is the final output of the policy network. The second and third dimensions of this 20 × 16 × 16 matrix correspond to different decap locations on the PDN, while the first dimension covers ten different decap types and two decap placement layers (top and bottom). The 8 × 16 × 16 matrix in Fig. 4 is followed by two FC layers and one value output, which is the output of the value network $V_\theta(s_t)$ and an estimation of the GAE in (7).

D. Reward Definition

The proposed DRL approach trains the agent and learns the optimal decap placement strategy through an iterative exploration process. Fig. 5(a) illustrates the iterative decap placement process according to the output probability distribution of the policy network. As shown in Fig. 4, the output probabilities of the proposed DRL method include the probabilities for decap location selection (on the top and bottom layers) and decap type selection. In Fig. 5(a), only the probability distribution of different decap locations is plotted for simplicity. In the training and exploration process, the current board state is input to the DNN, which outputs the probability for different actions. Subsequently, one action is selected according to the probability distribution, the board state is updated, and the next action is determined. Such an iterative procedure is continued until the decap locations are full or the target impedance is satisfied. For each action, a reward is assigned to evaluate the quality of the action and guide the DNN training toward reward maximization.

The reward is essential to the effective training of the DRL agent. The reward defined in the proposed DRL method is depicted in Fig. 5(b), as expressed by

$$r_t = \left(\frac{N_{t+1} - N_t}{N}\right) + T \tag{10}$$

where N_t and N_{t+1} are the number of frequency points at which the target impedance is satisfied before and after the current action is taken, respectively, and T is an extra term determined by whether the target impedance is satisfied and the number of placed decaps, expressed by

$$T = \begin{cases} 1 + N - N_{\text{decap}}, & \text{if target z is satisfied} \\ -p, & \text{if locations are full and target z is not satisfied} \end{cases} \tag{11}$$

where N is the total number of available decap locations, N_{decap} is the number of placed decaps, and p is a penalty term representing the maximum percentage for which the target impedance is violated.

This reward definition in (10) is similar to [15], but the extra term T is slightly different. In [15], the term T is equal to 1 if the target impedance is satisfied, and 0 otherwise. However, in this article, a larger value for T is assigned if fewer number of decaps are used to satisfy the target impedance, which encourages the algorithm to converge faster to the optimal solution. Moreover, the penalty term p in (11) is used to better guide the algorithm toward a possible solution even if the target impedance is hard to be satisfied.

The reward definition (11) aims to minimize the number of decaps to satisfy a target impedance. Users can flexibly adjust this reward definition to meet different requirements, such as reducing costs or occupying layout areas.

E. DNN Training and Testing

Algorithm 1 briefly summarizes the training and testing procedure of the proposed algorithm. The policy and value networks are trained through exploration experiences by maximizing the loss function (8) (or minimizing it by adding a negative sign). Every epoch means starting decap placement on an empty PDN until all decap locations are full or the target impedance is met. In each epoch, eight different environments are used (i.e., $N_{\text{env}} = 8$ in Algorithm 1), which means that the agent explores the decap placement strategy independently eight times. Since each decap placement is determined from the probability distribution, the eight environments are different from each other due to randomness. The experiences from the eight environments are stored and used to train the policy and value networks after each epoch is finished.

The PPO algorithm introduced in Section III-B is adopted as the DRL algorithm in this article. To accelerate the convergence of the training process, this article proposes a modified PPO algorithm as described in Algorithm 1. The improvement is that another memory B is defined to store the best experiences with the fewest decaps during the training phase. After each epoch, the exploration experiences stored in memory A and the best experiences stored in memory B are used collectively to train the DNN. In contrast, the original PPO algorithm only uses the experiences in memory A for training.

Algorithm 1 Proposed Modified PPO Algorithm

Input: N_epoch, number of epochs for training;
N_env, number of environments for training;
Input PDN information, including board shape, stackup, decap locations, and target z;
N_loc, number of decap locations in total;
n_decap_best, initial value is N_loc;
Define a policy and value DNN;
Define hyperparameters, including learning rate, discount factor γ, smoothing parameter λ;
Define memory A to store training experiences;
Define memory B to store the best experiences with the fewest decaps;
for $n\_epoch \in \{1, \ldots, N\_epoch\}$ **do**:
  // Training the DNN
  for $n\_env \in \{1, \ldots, N\_env\}$ **do**:
    Obtain the initial board state s_t;
    for $n\_decap \in \{1, \ldots, N\_loc\}$ **do**:
      Input board state s_t into DNN;
      Obtain output value V_t and action probability distribution $\pi_\theta(a_t|s_t)$;
      Choose one action a_t according to the probability distribution;
      Obtain reward r_t and the next state s_{t+1};
      $s_t = s_{t+1}$;
      if target z satisfied or $n\_decap = N\_loc$:
        if $n\_decap < n\_decap\_best$:
          $n\_decap\_best = n\_decap$;
          Empty memory B;
          Store experiences in B;
        else:
          Store experiences in A;
        end if
        break;
      end if
    end for
  end for
  Use the experiences in both A & B and the objective function in Eqn. (8) to train DNN;
  Empty memory A;
  // Testing the DNN
  Obtain the initial board state s_t;
  for $n\_decap \in \{1, \ldots, N\_loc\}$ **do**:
    Input board state s_t into DNN;
    Obtain action probability distribution;
    Choose action a_t with the highest probability;
    Obtain the next state s_{t+1};
    $s_t = s_{t+1}$;
    Record the best solution;
  end for
end for
Obtain the best solution with the fewest number of decaps during the training and testing phases

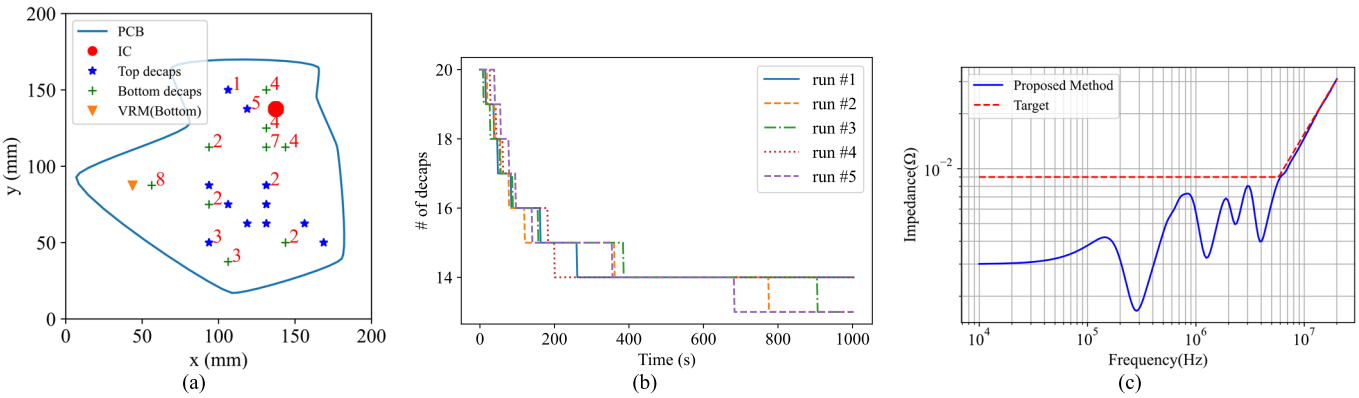


Fig. 6. Case 1. The board layer types (from the top layer to the bottom layer) are GND, PWR, GND, GND, and GND; the dielectric thickness (from top to bottom) is 0.3, 0.2, 0.4, and 0.3 (unit: mm). (a) Decap placement solution using the proposed DRL method (13 decaps). (b) Convergence of the best solution for five independent runs during the training and testing process. (c) Impedance of the proposed DRL method and the target impedance.

The best decap solution during the training process is recorded. After the training of each epoch, the DNN is also tested by choosing the highest probability for each action. The best decap solution for the training and testing process in each epoch is recorded, which is expected to converge to the optimal solution as the training proceeds. The Adam optimizer [32] was used for training, and the learning rate was 0.0001 in this article. The training was performed on an NVIDIA Tesla K80 GPU.

The proposed DRL method can learn from the exploration experiences with the corresponding reward and is expected to be more effective than the GA method [6], which is closer to random searching. Also, the sequence of actions contributes collaboratively to the reward in DRL rather than individually, rendering the proposed method more promising than the full search method [5] and the Newton–Hessian method [11] in finding the optimal solutions.

IV. Method Validation

Three testing cases with different PDN configurations and target impedances are employed in this section to compare the proposed method with the full search method [5], the GA method [6], and the Newton–Hessian method [11]. The decap placement solutions of different approaches are compared. The detailed parameters of the decap types represented by numbers 1–10 can be found in [18]. The solution comparison and time consumption of different approaches for five independent runs are listed in Table I. The full search method and the Newton–Hessian method, which explores a much smaller search space, took less time than the proposed DRL and the GA methods. Therefore, it is less meaningful to compare the time consumption of the full search and Newton–Hessian methods in Table I, but only the numbers of decaps are listed for the two approaches. Moreover, since these two methods implemented in this article are deterministic without randomness, only one solution is obtained for each testing case.

A. Case 1

In Case 1, the PCB has five layers, and the PDN configuration is shown in Fig. 6. There are 20 available decap locations.

TABLE I
Solution Comparison of Different Methods (Time–Decap Number)

Case #	Run #	Proposed DRL	GA	Newton-Hessian	Full Search
1	1	259s – 14	1032s – 13	/	/ – 15
	2	773s – 13	422s – 14		
	3	903s – 13	387s – 14		
	4	201s – 14	887s – 13		
	5	682s – 13	259s – 14		
2	1	1373s – 27	1768s – 35		/ – 33
	2	3017s – 26	3188s – 38		
	3	1172s – 28	2484s – 37		
	4	4252s – 26	3385s – 33		
	5	1442s – 27	1900s – 36		
3	1	688s – 7	391s – 9	/ – 10	/ – 8
	2	142s – 8	763s – 8		
	3	872s – 7	442s – 8		
	4	201s – 8	1082s – 7		
	5	924s – 7	353s – 9		

An R–L shape target impedance is used. In the proposed DRL, the discount factor γ is 0.99, and the smoothing parameter λ is 0.99. The proposed DRL converged to a solution of 13 or 14 decaps after searching for several minutes. The full search method took less computation time, as it explored the solution in a much narrower search space, but it only found a solution of 15 decaps. The GA method, on average, consumed a comparable time to the proposed method and also converged to 13 or 14 decaps. The Newton–Hessian method can only be applied to a flat target impedance, so it is not compared with the other approaches in this case. In this case with 20 decap locations, it can be concluded that the proposed DRL can find slightly better solutions than the full search and has a comparable performance with the GA approach regarding solution quality and convergence speed.

B. Case 2

In Case 2, the PCB has five layers, and the PDN configuration is shown in Fig. 7. There are 60 available decap

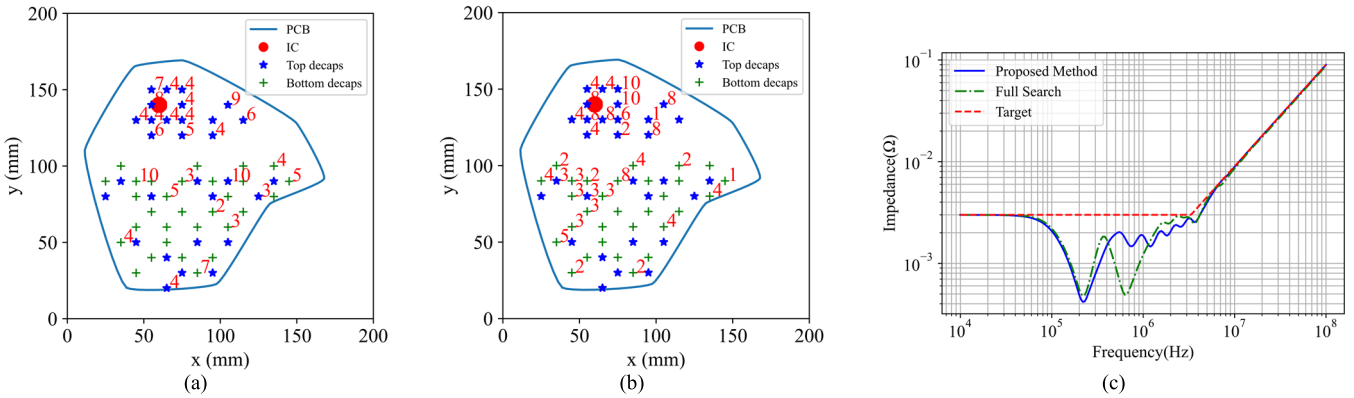


Fig. 7. Case 2. The board layer types (from the top layer to the bottom layer) are GND, PWR, GND, GND, and GND; the dielectric thickness (from top to bottom) is 0.3, 0.5, and 0.3 (unit: mm). (a) Decap placement solution using the proposed DRL method (26 decaps). (b) Decap placement solution using the full search method (33 decaps). (c) Target impedance, impedance of the proposed DRL method and the full search method.

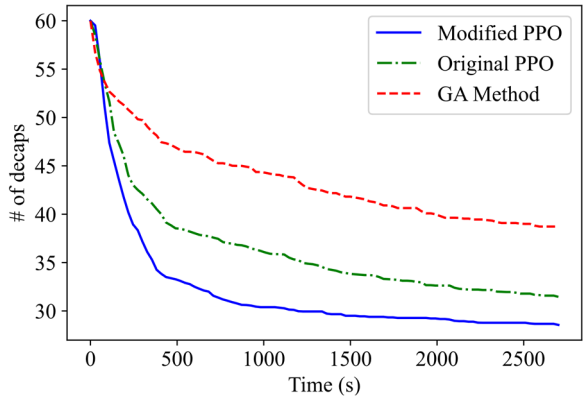


Fig. 8. Convergence of decap number for the modified PPO (proposed), original PPO, and GA method by taking the average result of 18 independent runs for Case 2.

locations, and an R–L target impedance is used. Hence, the Newton–Hessian method is not included in the comparison. In the proposed DRL, the discount factor γ is 0.99, and the smoothing parameter λ is 1. The proposed DRL converged to a solution of 26–28 decaps after training for 20–70 min. Since the search space becomes much larger than Case 1 due to a much larger number of decap locations, searching for the optimal solution becomes more challenging. However, in Case 2, the full search method could only find 33–38 decaps even after searching for about 30–60 min. Fig. 7 compares the decap distribution and impedance result for one solution of 26 decaps for the proposed DRL and one solution of 33 decaps for the GA method. Moreover, in Case 2, the full search method could only find a solution of 33 decaps. The result comparison demonstrates that when the search space becomes much larger due to a larger number of decap locations, the proposed DRL method shows more remarkable advantages over the full search and GA methods.

Moreover, to demonstrate the advantage of the modified PPO algorithm over the original PPO and GA method, the convergence of the number of decaps for the three methods is compared in Fig. 8 by taking the average results of 18 independent runs. As explained earlier, the original PPO method denotes the PPO algorithm without memory B to store the best experiences. It can be concluded from Fig. 8 that the modified PPO algorithm can noticeably accelerate the convergence compared with the original PPO algorithm and can also converge much faster than the GA method.

C. Case 3

In Case 3, a flat target impedance of 0.012 Ω is adopted, so that the Newton–Hessian method can be applied and compared with the other approaches. There are 20 available decap locations. In the proposed DRL method, the discount factor γ is 0.99, and the smoothing parameter λ is 0.99. The proposed DRL achieved a solution of 7–8 decaps in several minutes. The full search method and the GA method behave similar to the proposed method in this case, but the Newton–Hessian method only found a solution of ten decaps. The result comparison is shown in Fig. 9. As explained in Section II, the Newton–Hessian method selects the decap type with a resonance frequency closest to the frequency where the maximum impedance occurs. As observed in Fig. 9(b), the Newton–Hessian method chooses the same kind of decap, which causes an overdesign issue in the impedance result shown in Fig. 9(c). However, the proposed DRL method achieved the same target impedance using fewer decaps and mitigated the overdesign problem. This test case indicates that the decap selection and placement method in the Newton–Hessian method is not an optimal strategy.

V. Conclusion

This article presents a new DRL methodology for decap optimization on complex PDN structures. The method encodes the PDN structure information into the input matrices, including the PCB shape, stack up, IC location, and decap locations, and outputs the probabilities for different decap locations and types. The impedance matrix of a complex PDN is fast computed using a boundary integration method. Subsequently, the DRL algorithm is trained through self-exploration and records the best training and testing solution as the training proceeds. Compared with the state-of-the-art approaches, including the full search method [5], the GA method [6], and the Newton–Hessian method [11], the proposed algorithm demonstrates a more reliable convergence to the optimal solution by considering the collaborative contribution of a

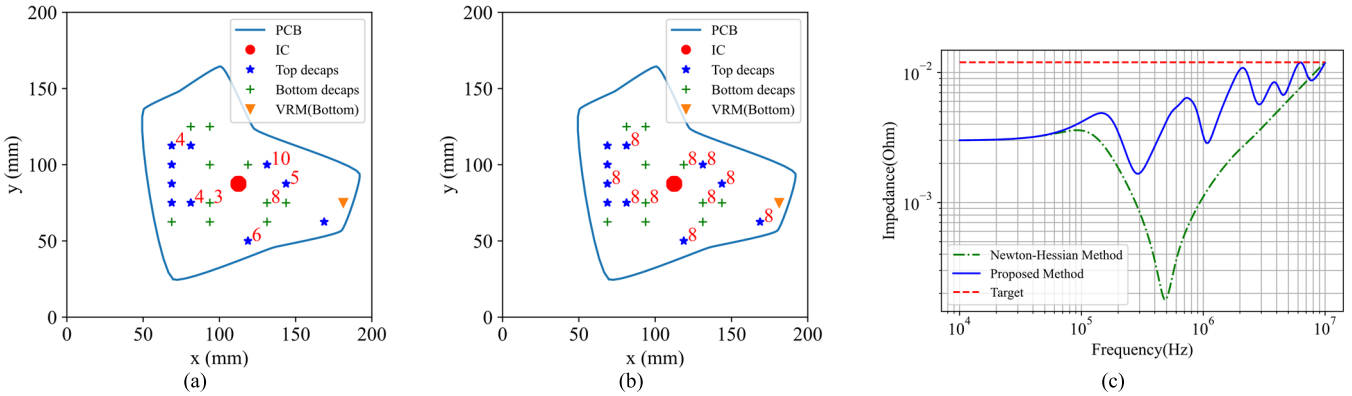


Fig. 9. Case 3. The board layer types (from the top layer to the bottom layer) are GND, PWR, GND, GND, and GND; the dielectric thickness (from top to bottom) is 0.3, 0.4, 0.2, and 0.3 (unit: mm). (a) Decap placement solution using the proposed DRL method (seven decaps). (b) Decap placement solution using the Newton–Hessian method (ten decaps). (c) Target impedance, impedance of the proposed DRL method and the Newton–Hessian method.

sequence of decap location and type selections. Also, in the proposed DRL method based on the PPO algorithm, this article presents a modified PPO algorithm that converges noticeably faster than the original PPO algorithm and the GA method. Moreover, the proposed DRL method can find better solutions than the full search and Newton–Hessian methods, since these two approaches search in a much smaller space.

The proposed method exhibits the capability of converging to high-quality decap solutions with complex PDN input information. By varying the input parameters and generating large amounts of PDN data [27], it is promising to pretrain a DRL algorithm that can predict a high-quality decap solution for a new PDN. Even though Park et al. [16] recently showed the feasibility of constructing a pretrained DRL model, however, their model can only be generalized to different IC and decap locations, with the other parameters being the same, making their algorithm impractical in real applications. Based on the work of this article, a DRL model can be pretrained and generalized to different PDN configurations by adopting the transfer learning [13] or meta-learning technique [33], which will be addressed and presented in our future publications.

ACKNOWLEDGMENT

The authors would like to sincerely thank Dr. James Drewniak, Dr. Jun Fan, Dr. Albert Ruehli, Dr. Brice Achkir, Dr. Bo Pu, and Dr. Zhifei Xu for their valuable discussions on this project.

REFERENCES

[1] M. A. P. Mezhiba and E. G. Friedman, *Power Distribution Networks With On-Chip Decoupling Capacitors*. Cham, Switzerland: Springer, 2007.

[2] J. Y. Choi and M. Swaminathan, "Decoupling capacitor placement in power delivery networks using MFEM," *IEEE Trans. Compon., Packag., Manuf. Technol.*, vol. 1, no. 10, pp. 1651–1661, Oct. 2011.

[3] S. Piersanti, F. de Paulis, C. Olivieri, and A. Orlandi, "Decoupling capacitors placement for a multichip PDN by a nature-inspired algorithm," *IEEE Trans. Electromagn. Compat.*, vol. 60, no. 6, pp. 1678–1685, Dec. 2018.

[4] R. Cecchetti, F. de Paulis, C. Olivieri, A. Orlandi, and M. Buecker, "Effective PCB decoupling optimization by combining an iterative genetic algorithm and machine learning," *Electronics*, vol. 9, no. 8, p. 1243, Aug. 2020.

[5] F. de Paulis, R. Cecchetti, C. Olivieri, and M. Buecker, "Genetic algorithm PDN optimization based on minimum number of decoupling capacitors applied to arbitrary target impedance," in *Proc. IEEE Int. Symp. Electromagn. Compat. Signal/Power Integrity (EMCSI)*, Jul. 2020, pp. 428–433.

[6] Z. Xu, Z. Wang, Y. Sun, C. Hwang, H. Delingette, and J. Fan, "Jitter-aware economic PDN optimization with a genetic algorithm," *IEEE Trans. Microw. Theory Techn.*, vol. 69, no. 8, pp. 3715–3725, Aug. 2021.

[7] S. Hemaram and J. N. Tripathi, "Optimal design of a decoupling network using variants of particle swarm optimization algorithm," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2021, pp. 1–5.

[8] P. Kadlec, M. Marek, M. Stumpf, and V. Sedenka, "PCB decoupling optimization with variable number of capacitors," *IEEE Trans. Electromagn. Compat.*, vol. 61, no. 6, pp. 1841–1848, Dec. 2019.

[9] K. Koo, G. R. Luevano, T. Wang, S. Özbayat, T. Michalka, and J. L. Drewniak, "Fast algorithm for minimizing the number of decap in power distribution networks," *IEEE Trans. Electromagn. Compat.*, vol. 60, no. 3, pp. 725–732, Jun. 2018.

[10] S. Han, O. W. Bhatti, and M. Swaminathan, "A knowledge based method for optimization of decoupling capacitors in power delivery networks," *IEEE Trans. Compon., Packag., Manuf. Technol.*, vol. 12, no. 5, pp. 828–838, May 2022.

[11] J. Wang, Z. Xu, X. Chu, J. Lu, B. Ravelo, and J. Fan, "Multiport PDN optimization with the Newton–Hessian minimization method," *IEEE Trans. Microw. Theory Techn.*, vol. 69, no. 4, pp. 2098–2109, Apr. 2021.

[12] D. Silver et al., "Mastering the game of go without human knowledge," *Nature*, vol. 550, no. 7676, pp. 354–359, 2017.

[13] A. Mirhoseini et al., "A graph placement methodology for fast chip design," *Nature*, vol. 594, no. 7862, pp. 207–212, 2021.

[14] H. Park et al., "Reinforcement learning-based optimal on-board decoupling capacitor design method," in *Proc. IEEE 27th Conf. Elect. Perform. Electron. Packag. Syst. (EPEPS)*, Oct. 2018, pp. 213–215.

[15] H. Park et al., "Deep reinforcement learning-based optimal decoupling capacitor design method for silicon interposer-based 2.5-D/3-D ICs," *IEEE Trans. Compon., Packag., Manuf. Technol.*, vol. 10, no. 3, pp. 467–478, Mar. 2020.

[16] H. Park et al., "Transformer network-based reinforcement learning method for power distribution network (PDN) optimization of high bandwidth memory (HBM)," *IEEE Trans. Microw. Theory Techn.*, vol. 70, no. 11, pp. 4772–4786, Nov. 2022.

[17] L. Zhang et al., "Decoupling capacitor selection algorithm for PDN based on deep reinforcement learning," in *Proc. IEEE Int. Symp. Electromagn. Compat.*, Jul. 2019, pp. 616–620.

[18] L. Zhang, W. Huang, J. Juang, H. Lin, B.-C. Tseng, and C. Hwang, "An enhanced deep reinforcement learning algorithm for decoupling capacitor selection in power distribution network design," in *Proc. IEEE Int. Symp. Electromagn. Compat. Signal/Power Integrity (EMCSI)*, Jul. 2020, pp. 245–250.

[19] M. Friedrich and M. Leone, "Boundary-element method for the calculation of port inductances in parallel-plane structures," *IEEE Trans. Electromagn. Compat.*, vol. 56, no. 6, pp. 1439–1447, Dec. 2014.

[20] L. Zhang et al., "Efficient DC and AC impedance calculation for arbitrary-shape and multilayer PDN using boundary integration," *IEEE Trans. Signal Power Integrity*, vol. 1, pp. 1–11, 2022.

[21] J. Kim, L. Ren, and J. Fan, "Physics-based inductance extraction for via arrays in parallel planes for power distribution network design," *IEEE Trans. Microw. Theory Techn.*, vol. 58, no. 9, pp. 2434–2447, Sep. 2010.

[22] L. Wei et al., "Plane-pair PEEC model for power distribution networks with sub-meshing techniques," *IEEE Trans. Microw. Theory Techn.*, vol. 64, no. 3, pp. 733–741, Mar. 2016.
[23] J. Kim et al., "Chip-package hierarchical power distribution network modeling and analysis based on a segmentation method," *IEEE Trans. Adv. Packag.*, vol. 33, no. 3, pp. 647–659, Aug. 2010.
[24] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Mach. Learn.*, vol. 8, nos. 3–4, pp. 229–256, May 1992.
[25] A. Ashok, N. Rhinehart, F. Beainy, and K. M. Kitani, "N2N learning: Network to network compression via policy gradient reinforcement learning," 2017, *arXiv:1709.06030*.
[26] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017, *arXiv:1707.06347*.
[27] L. Zhang et al., "Fast impedance prediction for power distribution network using deep learning," *Int. J. Numer. Model.*, vol. 35, no. 2, p. e2956, 2022.
[28] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 2, pp. 84–90, Jun. 2012.
[29] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 448–456.
[30] B. Xu, N. Wang, T. Chen, and M. Li, "Empirical evaluation of rectified activations in convolutional network," 2015, *arXiv:1505.00853*.
[31] B. Gao and L. Pavel, "On the properties of the softmax function with application in game theory and reinforcement learning," 2017, *arXiv:1704.00805*.
[32] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.
[33] T. Hospedales, A. Antoniou, P. Micaelli, and A. Storkey, "Meta-learning in neural networks: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 9, pp. 5149–5169, Sep. 2022.

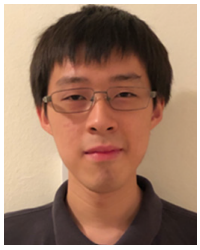
Ling Zhang (Member, IEEE) received the B.S. degree in electrical engineering from the Huazhong University of Science and Technology, Wuhan, China, in 2015, and the M.S. and Ph.D. degrees in electrical engineering from the Missouri University of Science and Technology (Missouri S&T), Rolla, MO, USA, in 2017 and 2021, respectively.

He is currently a Research Fellow at Zhejiang University, Hangzhou, China. His research interests include machine learning, power integrity, electromagnetic interference, radio frequency interference, and signal integrity.

Dr. Zhang was a recipient of the AP-EMC Young Scientist Reward in 2022.

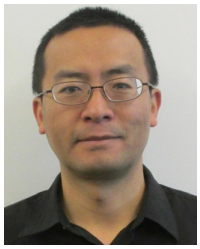
Li Jiang (Student Member, IEEE) received the B.S. degree from the College of Information Science and Electronic Engineering, Zhejiang University, Hangzhou, China, in 2022, where he is currently pursuing the Ph.D. degree in electronics science and technology.

His current research interests include modeling and optimization of PDN in PI problem.



Jack Juang (Student Member, IEEE) received the B.S. degree in electrical engineering from the Missouri University of Science and Technology (Missouri S&T), Rolla, MO, USA, in 2020, where he is currently pursuing the M.S. degree at the EMC Laboratory.

He has been involved in projects relating to optimizing decoupling capacitor placement and RF susceptibility of smart devices. His research interests include power distribution network modeling and optimization.



Zhiping Yang (Fellow, IEEE) received the B.S. and M.S. degrees in electrical engineering from Tsinghua University, Beijing, China, in 1994 and 1997, respectively, and the Ph.D. degree in electrical engineering from the Missouri University of Science and Technology (Missouri S&T), Rolla, MO, USA, in 2000.

From 2000 to 2005, he worked at Cisco Systems, San Jose, CA, USA, as a Technical Leader. From 2005 to 2006, he worked at Apple Computer, Cupertino, CA, USA, as a Principal Engineer. From 2006 to 2012, he worked at Nuova Systems (which was acquired by Cisco in 2008) and Cisco Systems as a Principal Engineer. From 2012 to 2015, he worked at Apple, Cupertino, as a Senior Manager. He is currently a Senior Hardware Manager at Google Consumer Hardware Group, Mountain View, CA, USA. His research interests include signal integrity and power integrity methodology development for die/package/board co-design, high-speed optical module, various high-speed cabling solutions, high-speed DRAM/storage technology, high-speed serial signaling technology, and RF interference.

Er-Ping Li (Fellow, IEEE) is currently a Qiushi-Distinguished Professor at the Department of Information Science and Electronic Engineering, Zhejiang University, Zhejiang, China. He served as a Founding Dean for the Institute of Zhejiang University, Zhejiang—University of Illinois at Urbana-Champaign, Champaign, IL, USA, in 2016. From 1993, he has served as a Research Fellow, an Associate Professor, a Professor, a Principal Scientist, and a Senior Director at the Singapore Research Institute and University. His research interests include electrical modeling and design of micro-/nano-scale integrated circuits and 3-D electronic package integration.

Dr. Li is a Fellow of the USA Electromagnetics Academy and the Singapore Academy of Engineering. He was a recipient of the IEEE EMC Technical Achievement Award in 2006, the Singapore IES Prestigious Engineering Achievement Award, the Changjiang Chair Professorship Award in 2007, the 2015 IEEE Richard Stoddard Award on EMC, the 2021 IEEE EMC Laurence G. Cumming Award, and the Zhejiang Natural Science 1st Class Award.

Chulsoon Hwang (Senior Member, IEEE) received the B.S., M.S., and Ph.D. degrees from the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea, in 2007, 2009, and 2012, respectively, all in electrical engineering.

He was with Samsung Electronics, Suwon, South Korea, as a Senior Engineer, from 2012 to 2015. In July 2015, he joined at the Missouri University of Science and Technology (Missouri S&T), Rolla, MO, USA, where he is currently an Assistant Professor. His research interests include RF desense, signal/power integrity in high-speed digital systems, EMI/EMC, hardware security, and machine learning.

Dr. Hwang was a recipient of the AP-EMC Young Scientist Award, the Google Faculty Research Award, and the Missouri S&T's Faculty Research Award. He was a co-recipient of the IEEE EMC Best Paper Award, the AP-EMC Best Paper Award, and a two-time co-recipient of the DesignCon Best Paper Award.