Deep Reinforcement Learning applied to Analog Integrated Circuit Sizing

Tomás Bessa
Instituto Superior Técnico
Universidade de Lisboa

ABSTRACT

The following work resides in the scientific field of electronic design automation. In this paper, there is a particular emphasis on the automatic sizing of integrated circuits and how the process may be improved and optimized—mainly focusing on the application of Machine Learning and Deep Learning techniques to analog IC sizing, namely Reinforcement Learning methodology. This paper presents Asynchronous Actor-Critic, a Reinforcement Learning approach that can be applied to analog IC sizing. The proposed algorithm builds upon the AIDA-C tool and enhances its flow with an alternative RL-based sizing solution. The model is applied to two amplifier circuit topologies, VCOTA and Folded Cascode. For both cases, the agent can successfully perform the sizing of their components in a short time while fulfilling set target specifications.

Keywords

Actor-Critic, Analog Integrated Circuits, Artificial Neural Networks, Deep Learning, Electronic Design Automation, Reinforcement Learning

1. INTRODUCTION

Currently, markets are frenetic. Industries are developing and adapting to new patterns, and the market has become increasingly demanding. The electronics market, such as analog and digital circuits, is no exception. The overall projection of total IC sales growth in 2022 is unchanged and expected to rise 11% this year to a record-high $567.1 billion. The new 2Q22 update keeps the 2022 growth forecast unchanged in analog ICs (up 12%) and logic integrated circuits (up 11%)[1].

At the same time, the need for new functionalities, longer battery times, smaller (thinner) devices, more power efficiency, less production and integration costs, and less design cost makes the design of electronic systems a genuinely challenging task. IC designers are building increasingly complex systems, and the integration in modern systems is exceptionally high and is common to find devices where the whole system is integrated into a single chip [2].

The complexity of electronic systems, the extremely competitive markets, and the strict time-to-market impose the use of Computer-Aided Design (CAD) tools to support the design process. In digital IC design, several Electronic Design Automation (EDA) tools and design methodologies are available to help the designers keep up with the new capabilities the technology offers. Currently, almost all low-level phases of the process are automated. The level of automation is far from the push-button stage but is keeping up reasonably well with the complexity supported by the technology[3][4]. On the other hand, analog design automation (ADA) tools are not keeping up with new challenges created by technological evolution. Due to the lack of automation, designers keep exploring the solution space manually. This method causes long design times and is allied to the non-reusable nature of analog IC, making analog IC design a cumbersome task. [3][4]. The difference between analog and digital design automation is because analog design generally is less systematic, more heuristic, and knowledge intensive than the digital counterpart and becomes critical when digital and analog circuits are integrated.

This work was hosted at Instituto de Telecomunicações, funded by Fundação para a Ciência e Tecnologia–Ministério da Ciência, Tecnologia e Ensino Superior (FCT/MCTES) through national funds and, when applicable co-funded European Union (EU) funds under the project UIDB/50008/2020.

This paper presents a methodology and tool for automatic analog IC sizing approach based on reinforcement learning (RL). This paper is organized as follows: section 2 overviews analog IC design with particular emphasis on analog IC sizing using RL; section 3 highlights previous work in Deep Reinforcement Learning applied to Analog IC Sizing; section 4 explains the proposed architecture; section 5 presents case studies; and finally, in section 6 some conclusions are drawn, and future work proposed.

2. ANALOG IC DESIGN: OVERVIEW

In order to locate analog IC sizing, a brief presentation of a typical analog IC design flow is shown, and the analog IC sizing task is described.

2.1 Design Flow

The specific design flow of Integrated Circuits is often unique and specific to each designer or company producing the IC, like the concept of human fingerprints. Each designer/company has its style of design. However, the more significant part of the work related to analog design flow can be generally mapped into the model proposed by Gielen and Rutenbar [10], which consists of a series of top-down topology selection and specifications translation steps and bottom-up layout generation and extraction steps including several verification stages along the way.

Adopting the top-down strategy proved advantageous since it allowed exploring system architectures of higher complexity, which led to improved system optimization at a higher level of abstraction. This can be achieved before the beginning of more specific and intricate implementations at lower levels. This way, finding problems at the beginning of the design process is possible, leading to an increase in first-time success rate and a decrease in the necessary time it takes to conclude the entire process.

The system complexity influences the number of hierarchy levels in the design flow. A generally accepted representation of design architecture consists of two prominent design paths:

- Top-down electric synthesis: Includes topology selection, design verification, and specification translation. The latter is also known as circuit sizing at the lowest level;
- Bottom-up physical synthesis: Includes detailed design verification after layout extraction and generation;

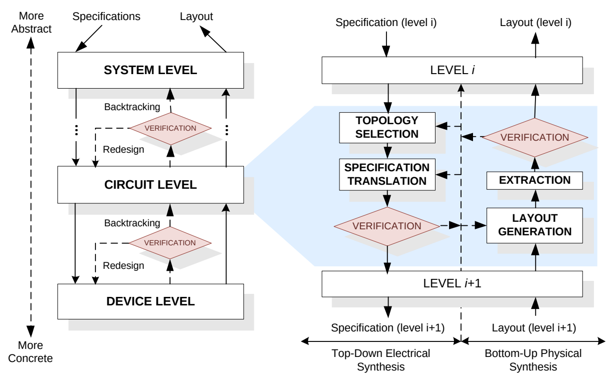

Figure 1. Hierarchical level and design tasks of analog design flow architecture

The process of determining the circuit topology best suited for a given problem is called topology selection. When selecting the topology that better suits a given problem, it is crucial to meet the specifications at the current hierarchy level, where the available topology can either be chosen from a determined set or synthesized. Each block carries its specifications to the next level down the hierarchy, and the process is repeated until the top-down electric synthesis flow is complete.

The task which ensures the mapping of high-level block specifications into independent specifications for each sub-block is named Specification Translation. This task can be narrowed down to circuit sizing when dealing with the lowest level, as the sub-blocks are single devices. Circuit sizing is an iterative process that determines a suitable set of lengths, widths, and multiplicities for each device in the topology to achieve desired specifications.

2.2 Circuit Sizing

Recent developments show that analog and RF cells, such as amplifiers or oscillators, can be automatically synthesized from specification to fabrication. The most common methodologies follow an optimization-based strategy with accurate circuit simulations in the loop to evaluate the circuit's performance. These tend to be time-consuming since the complex relationship between design parameters and circuit specifications plays a big part in complicating this task, but they can find usable results without user intervention. AIDA-C [3], developed at Instituto de Telecomunicações (IT), is one such optimization-based sizing approach. It is part of the AIDA Framework, an electronic design automation framework fully developed at IT, and appears as an Electronic Design Automation tool to aid designers in doing their job better and faster. AIDA multi-objective design methodology for automatic, analog IC sizing is based on NSGAII[5][6] multi-objective multi-constraint optimization, and the circuit's performance evaluation is done with circuit simulators, e.g., ELDO, Spectre, or NGSPICE, ensuring that the developed automatic circuit sizing is compliant with the accuracy requirements of the analog designers. In AIDA-C, whose core optimization engine is illustrated in Figure 2, circuit sizing and optimization are implemented as a multi-objective multi-constraint optimization problem defined as:

$$\begin{aligned} &\textit{find } x \textit{ that minimize } f_m(x) && m = 1,2,\ldots M \\ &\textit{subject to } g_j(x) \geq 0 && j = 1,2,\ldots J \\ &x_i^L \leq x_i \leq x_i^U && i = 1,2,\ldots N \end{aligned}$$

where, x is a vector of N optimization variables, $g_j(x)$ one of the J constraints on the circuit performances and $f_m(x)$ is one of the M circuit performances being optimized. However, these optimization-based sizing approaches require many circuit simulations, which take time and consume energy. Hence the search for new and more effective approaches to automatically size analog ICs. Here is where Machine Learning advances come in handy, presenting significant innovation potential.

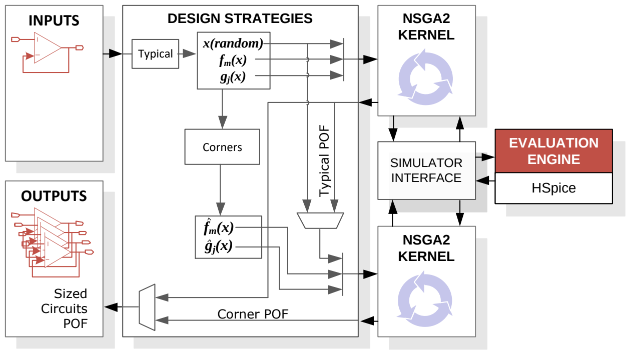

Figure 2. AIDA-C optimization loop [2]

3. PREVIOUS WORK

Automatic circuit sizing using reinforcement learning is a recent research area, and some of the most relevant works are described below.

In [7], a Circuit Attention Network is proposed. This approach is built on an Actor-Critic framework, and its primary goal is addressing the robust circuit sizing problem. Actor-Critic methods are temporal difference methods that have a separate memory structure to represent the policy value function. The policy structure may also be known as the actor since it is used to select actions. On the other hand, another model estimates the value (the expected reward of the state), also known as the critic, because it "criticizes" the actions previously performed by the actor. The CAN serves as both the Actor and Critic and is, at heart, a customized graph NN. It allows knowledge transfer between different topologies of the same circuit type, and its effectiveness has been proved through simulations. The stochastic technique is implemented to mitigate the layout effect. It tries to turn a sizing solution robust against layout uncertainty but requires a previously sized circuit. Result validation is done through post-layout simulation, showing significant improvement in circuit performance compared to Bayesian Optimization and GCN-RL.

A critical aspect of the CAN is its ability to transfer knowledge, i.e., it is applicable to several different topologies. Usually, there are two problems associated with knowledge transfer. The first one is letting a single GNN accommodate different input dimension sizes resulting from different graph structures. CAN successfully solves the first problem since graph pooling in CAN performs node/edge feature dimension reduction and effortlessly handles the input of different dimensions. The other problem is making a single GNN model provide different-sized outputs due to the difference in action spaces. This setback is solved by the actor network of target topology. It inherits the feature embedding layers from the source topology, replacing the MLP of source topology with an untrained MLP, thus needing very few data samples. Comparing the quality of the solution presented (the CAN-RL method with FOMR reward) with the GCN-RL method (which uses regular FOM reward). All in all, CAN-RL is better regarding circuit performance and robustness. Additionally, CAN-RL converges faster than the both GCN-RL and BO in most cases. CAN-RL is the fastest one to converge. It converges faster than both GCN-RL and BO. Relatively to BO, CAN-RL converges 5.6 times faster for *FOM* rewards and 6.8 times faster for FOMR rewards. When the reward function is FOM^R, the transfer makes CAN-RL approximately 2.8 times faster, which proves knowledge transfer facilitates further speedup.

AutoCkt [8]is a ML optimization framework which is trained by using RL. Firstly, there is a netlist of *N* design parameters and *M* target specifications that need to be optimized. The parameters are initialized, and the ANN uses the performance, target specifications, and current parameters to act. Afterwards, *L* trajectories are generated, and the targets are chosen among the target specifications. The rewards are cumulative as they depend on previous actions. Aiming to demonstrate AutoCkt's capabilities, three different simulation environments as well as three circuit topologies are considered: transimpedance amplifier, two-stage operational amplifier, two stage OTA. In order to test its performance, AutoCkt is tested and compared to Genetic Algorithm as well as a random RL agent. Compared to every algorithm. AutoCkt is faster and can fulfill many more target specifications than the other algorithms.

Robust Analog [9] is proposed as an efficient variation-aware optimization framework for automatic analog circuit design. Its primary goal is reducing simulation cost to design a robust circuit which is not susceptible to variations. When in the presence of any circuit topology, a search is conducted with the objective of finding a circuit sizing vector which can satisfy the constraints across all variations. The problem can, therefore, be formulated as a constraint satisfaction problem. The prominent goal is to find a sizing vector which satisfies any constraints under any corner task. This approach possesses an actor-critic model which can predict the value of task-conditioned action-state pairs. The aim of the actor-critic network is looking for sizing that fits all specified tasks. The reward is defined as the measurement of the relative distance between the current performance metrics and the design targets.

RobustAnalog is tested on three real-world/analog mixed signal circuits: Two-stage OTA, the second a Folded-Cascode OTA and the last one is a strongARM Latch. The results are compared to those of Bayesian Optimization (BO), Evolutionary Strategy (ES), and single-task RL algorithm (DDPG). Each method is compared to the others by the average reward. Considering all three circuit benchmarks, RobustAnalog achieves the smallest simulation cost to accomplish all corner tasks. As far as simulation costs go, RobustAnalog outperforms the other methods by a significant amount. The reductions in simulation cost are roughly 26 times in Two-Stage OTA, 30 times in storngARM Latch and 14 times in Folded-Cascode OTA. RobustAnalog shows a very significant improvement in efficiency. RobustAnalog is, therefore, a fast variation-aware optimization framework that is based on multitask RL. Its key property involves the ability to conduct efficient multitask learning with pruned training task space. Hence, it can design circuits effectively for variations. It is shown to significantly reduce simulation cost and scale to many variation cases. RobustAnalog is, thus, a promising approach to drastically shorten the circuit design cycle and reduce the cost.

4. Deep RL-Based Analog IC Sizing

The Analog IC Design Automation (AIDA) [3] software tool is an automation solution developed and maintained at IT. The AIDA framework implements an analog IC design flow from circuit-level specifications to a physical layout description, focusing on design optimizing and porting. It uses highly efficient searching methods combined with accurate circuit-level simulation, layout design rules, and parasitic extraction engines. Although not directly integrated into AIDA's tool, this work follows its flow, providing an alternative Deep RL-based sizing approach. The circuit setup, including netlists (circuit and test benches), circuit parameterization and design variables, and target specifications, is the one from AIDA. To that end, the interface with the simulator and measurement processing are wrapped in an environment suitable for RL. An agent is trained to replace the optimizer, as illustrated in Figure 3.

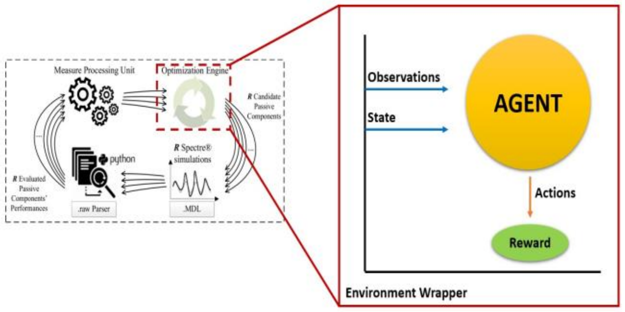

Figure 3. Proposed method illustrative diagram

4.1 State Space, Action Space, and Reward

The state is defined by design variables. The ranges of the design variables define the state space. Each variable range is encoded in the tuple (*min*, *max*, *step*). The *min* corresponds to the minimum size value a design variable can reach, whereas the *max* is the maximum size value the same variable can reach, and finally, the *step* is the minimum amount an action can alter the value of that design variable. The target specifications, however, are set in the environment wrapper and remain unchanged during the sizing process. The specifications establish limits on the measured performances and are used to compute the reward. An action in this framework is an alteration in the state. These changes only apply to design variables since the target specifications remain unchanged.

Each action is defined by a tuple, which encompasses the design variable that needs to be altered and the amount it can change (called the "step size"). For this work, the step size defined for each action is 1 or 10, meaning that an action can increase/decrease by one or ten times the step of each design variable.

Finally, any RL model needs a reward function. The RL process is divided into episodes and timesteps. An episode comprises time steps, and each episode in this work comprises 50 timesteps. Upon each time step, an action is performed, resulting in a reward assignment. If the action brings the circuit performance measures closer to the target specifications, it means the circuit performance was improved, resulting in a positive reward. On the other hand, if the action leads the circuit performance away from the target specifications, the assigned reward is negative.

The reward for each action is the difference between the target specifications that are not fulfilled and the objective target specifications, similarly to what is defined in AIDA. An action results in an alteration of the state and, if the agent gets closer to the target specification, it results in a reward increase. Otherwise, the reward value decreases. This calculation is performed for every target constraint (IDD, DC Gain, GBW, Phase Margin, and FoM). The resulting reward for each action is obtained by adding these results. If a solution is deemed impossible to simulate, the episode is immediately terminated and a reward of -100 is assigned to the episode. Once a feasible solution has been found, the episode terminates and is given a reward of +2000.

4.2 Model Structure

The Actor-Critic model is defined as a large Neural Network. Like all ANNs, the proposed network has an input layer, an output layer, and a set of hidden layers. This work makes use of two NNs. The first network is composed as follows: 1 input layer, 2 common hidden layers, 1 output relative to the actor, and 1 output relative to the critic (Figure 4). The second network has: 1 input layer, 2 hidden layers for the actor, 2 hidden layers for the critic, 1 output layer relative to the actor, 1 output layer relative to the critic.

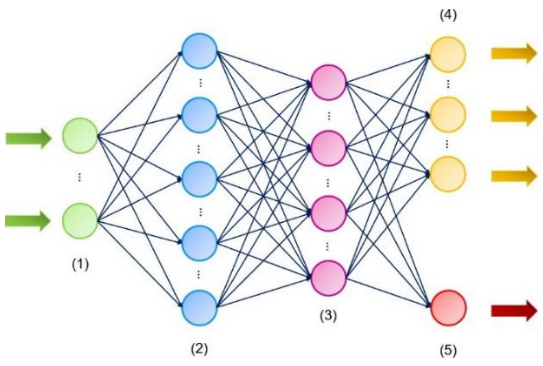

Figure 4 Model 1 representation

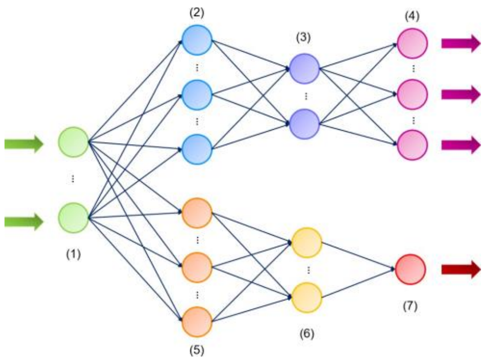

Figure 5. Model 2 representation

The chosen optimizer is the Stochastic Gradient Descent. One of its parameters is the learning rate. The nature of this problem caused the implementation of an adaptive learning rate. An adaptive learning rate is a value of the learning rate which decreases over time. To achieve that, it benefits from two main parameters: gamma and step size. The first parameter controls the decay rate, i.e., how much the learning rate will decrease. The second parameter defines how frequently the learning rate is updated (for example: if step size= 2: the learning rate is updated every two episodes). In this work, the Leaky ReLU activation function is applied to every layer except the last hidden layer of the actor network on both models, where the sigmoid function is applied. The output layer of the actor (for both models) is subjected to a SoftMax activation function.

4.3 Action Selection

Exploration and exploitation are two possible approaches to face in a decision-making problem. Exploitation consists of preferring the decision assumed to be optimal regarding the observed data. The main goal of this behavior is to avoid poor decisions as much as possible. However, it may prevent the algorithm from finding potentially better solutions. Exploration involves avoiding decisions that may seem optimal. This process relies on disregarding observed data, which is not considered sufficient to identify the best possible option. The main goal is to find a strategy with a favorable tradeoff between these decisions. The exploration-exploitation tradeoff implies a decision regarding the current model's state. Hence, the actor's output consists of a probability distribution of the best action to take. The best distribution, in this case, is the Categorical distribution since selecting the action is a multi-class problem.

4.4 Circuit Sizing using Deep Reinforcement Learning

Contrary to most ML methods, RL algorithms are not divided into the typical training, validation, and test stages. In model training, a given model is trained to be later tested and validated. This is not the case for this work. The learning process in RL, precisely the actor-critic method, is finding a feasible solution given the environment with the constraints. Therefore, the principal objective of the learning process is finding a feasible solution for the environment and automating the sizing of the circuit. At first, the neural network's weights are initialized randomly. During the beginning of the process, the quality of the results is low because the agent is learning what actions result in better rewards. As training progresses, the objective is to transform a poor-performing agent into an agent that can size the circuit's components, meeting the set target specifications. The agent maintains a policy (actor) π ($a_t|s_t$; θ) and a value function (critic) V (s_t; θ_v). Both policy and value functions are updated after 50 timesteps or if a terminal state is reached.
The TD term is multiplied by the probability assigned by the policy for the action at time t. This way, certain policies are more heavily penalized if they incorrectly estimate the value function.

The entire process is represented in Figure 6, It is important to highlight how it progresses over time. First, it is divided into episodes and time steps. Each episode is composed of time steps and in each time step, an action is performed. The selection of the action is as it is described in Chapter 4.2.4. The result of the action proceeds to influence the current state and reward. Afterwards, the episode reward is updated as the last reward of the episode. After each episode is finished, the reward function is updated, leading to an update in the solving condition. Once the reward function is updated, the training process enters in the finishing step.

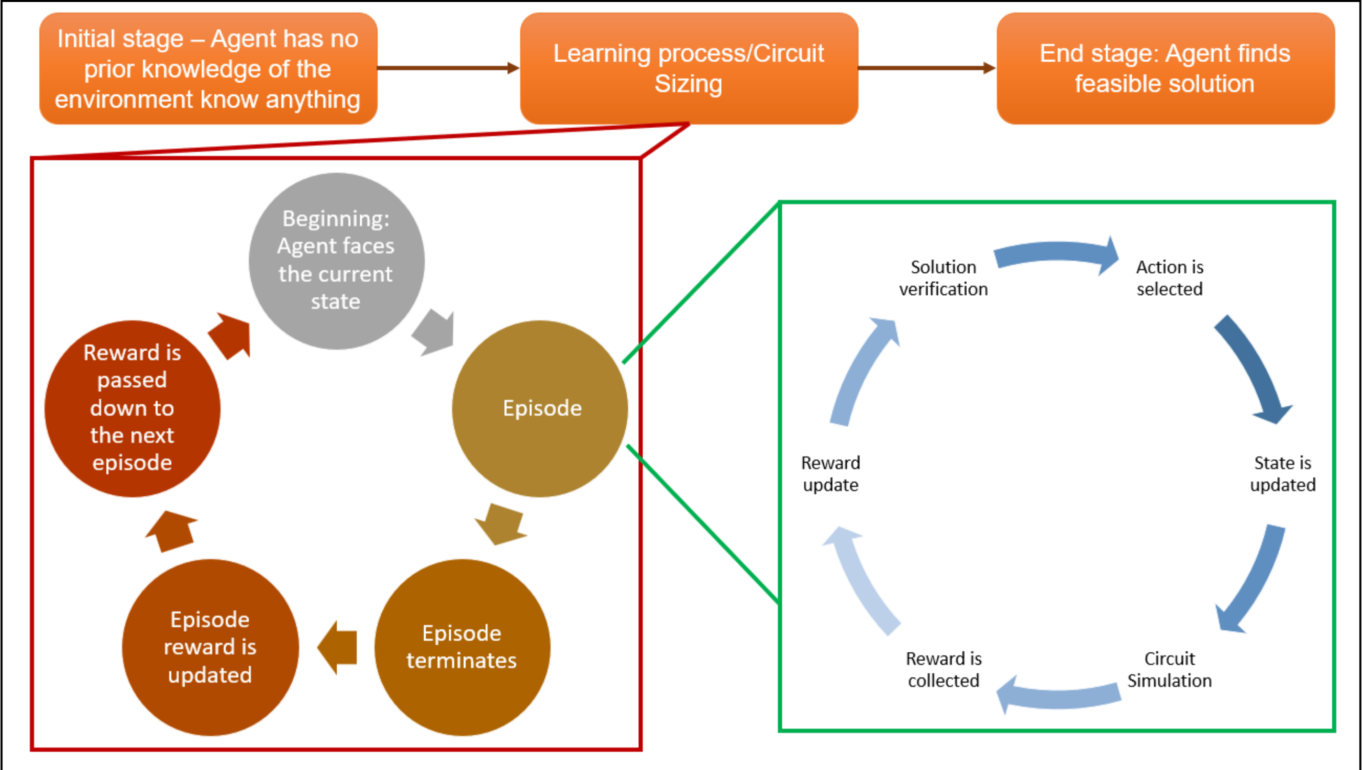
Initial stage – Agent has no prior knowledge of the environment know anything
Learning process/Circuit Sizing
End stage: Agent finds feasible solution
Beginning: Agent faces the current state
Episode
Episode terminates
Episode reward is updated
Reward is passed down to the next episode
Solution verification
Action is selected
State is updated
Circuit Simulation
Reward is collected
Reward update

Figure 6. Analog IC Sizing

The finishing step is the final step in an episode, incorporating every update the model needs to progress to the next episode. The final step ensures the model is ready to progress to the following episode and improve, by applying prior knowledge. The present rewards are updated considering the past rewards multiplied by the discounting factor and total episode rewards. The result is used to calculate the advantage function. The advantage function factors how beneficial an action is when compared to other actions for a particular state. Following the advantage function calculation, both policy losses and value losses are calculated. Once they are calculated, the gradients are all set to 0. (This step is important because PyTorch accumulates gradients, compromising the backpropagation and weight adjustment stages). The policy and value losses are added to achieve the total loss, which contributes to updating the weights in the Deep NN. Once the loss is fully calculated, it is possible to perform backpropagation, thus updating the weights and improving the network.

5. CASE STUDY

This Chapter is destined to the presentation of the results derived from the implementation. Every model in this work, both agent and environment, are implemented in Python 3.8.8, using PyTorch as backend. The code is run on an AMD Ryzen 7 5700U with Radeon Graphics CPU 1.80 GHz with 16GB of RAM.

5.1 Circuit Description

For proof of concept, the amplifier using voltage combiners for gain enhancement (VCOTA), presented in [42], is considered as well as the folded cascade amplifier (FCA), used in [43]. Voltage-combiners are typically used in radio frequency, due to their ability to convert fully differential signals into a single-ended one, for 50 and 75-Ohm impedance matching. The electrical scheme of a VC is shown in Figure 35. It employs a combination of a NMOS in a common-drain configuration and an NMOS in a common-source configuration.

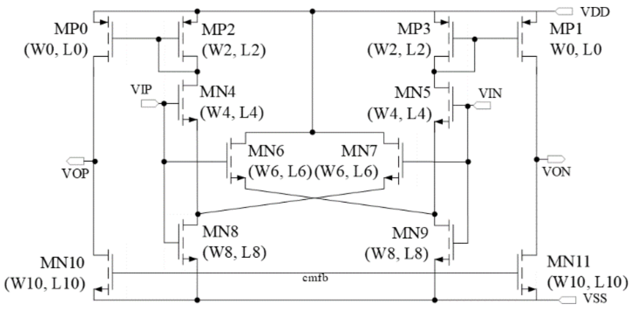

Figure 7. Circuit Schematic - VCOTA

5.2 VCOTA

The selection of the best solutions is organized by targets specifications. Each target has different specification, which needs to be met for the agent to find a valid solution. To test the first circuit(VCOTA), various target constraints are specified. The circuit performances which are considered as a way of evaluating the performance of the RL algorithm are: DC Gain, I_{DD}, GBW, Phase Margin (PM) and figure of merit (FoM). Their ranges of values are shown in the table below´:

Table 2 VCOTA: Table displays constraints for each Target

	IDD (μA)	GBW (MHz)	DC Gain (dB)	PM (º)
Target 0:	< 350	> 35	> 50	45 < PM < 90
Target 1:	< 300	> 40	> 40	45 < PM < 90
Target 2:	< 700	> 120	> 50	45 < PM < 90
Target 3:	< 210	> 25	> 40	45 < PM < 90
Target 4:	< 130	> 2	> 40	45 < PM < 90

Table 1 VCOTA: Model 1

Seed	#		IDD (μA)	DC Gain (dB)	GBW (MHz)	PM (º)	FoM² (MHz×pF/mA)
12	4	Target 0	347.2	55,3	49.45	58,9	854,7
72	50		347.0	58,8	45.57	49,1	787,8
44	124		308.9	58,4	43.67	46,1	848,2
56	70		339.5	55,1	45.72	74,4	807,9
80	21		345.9	54,4	62.64	62,8	1086,4
21	15		261.8	58,7	41.28	49,1	946,1
13	5		345.9	56,2	46.92	55,4	814,0
Average	41,3		---------	---------	---------	---------	877,9
12	34	Target 1	285,6	53,750	42,24	79,644	887,283
15	46		260,4	58,449	42,72	48,024	984,351
16	77		298,7	55,889	49,46	45,718	993,533
19	57		279,8	53,053	45,45	46,475	974,619
24	32		297,5	53,412	47,71	58,052	962,313
Average	49,2		---------	---------	---------	---------	960,420
12	34	Target 2	615.0	48,429	143,8	73,789	1403,415
14	184		666.4	53,370	124,4	46,399	1120,308
16	201		679.9	52,370	130,7	62,829	1153,398
20	268		593.1	50,920	124,3	63,475	1257,840
80	209		559.1	54,380	120,4	57,045	1292,535
Average	179,2		---------	---------	---------	---------	1245,499
12	34	Target 3	209,9	55,771	34,44	72,784	984,773
15	247		207,1	52,864	35,52	63,396	1028,833
24	54		187,2	48,983	26,04	69,794	834,732
21	143		207,1	57,824	30,16	63,036	873,573
22	49		202,0	47,200	26,33	46,743	782,190
Average	105,4		---------	---------	---------	---------	900,820
12	39	Target 4	117,6	51,068	4,696	89,493	239,572
15	5		128,7	46,519	2,201	88,901	102,616
22	24		129,2	52,674	15,54	53,254	721,798
24	41		122,9	45,109	4,265	86,979	208,175
21	143		108,9	54,185	4,591	88,005	252,906
Average	50,4		---------	---------	---------	---------	305,013

Target 0 is notably straightforward since the pre-determined conditions are easier to meet. On the other hand, the constraints on Target 1 are more complicated to fulfil. The IDD minimum limit value is lower than that of Target 0, which, when conjugated to the maximum limit set for GBW, make these constraints more complicated to meet. Target 2 aims to set a higher GBW value by sacrificing the maximum limit of IDD. Targets 3 and 4 are designed with the goal of limiting the maximum IDD value while not sacrificing DC Gain in the process. The main difference between Target 3 and Target 4 is that target 4 handles the trade off by lowering the lower limit of GBW while Target 3 does aims to achieve a lower IDD value while keeping the GBW at relatively standard levels.
For the VCOTA topology, the results show that the actor-critic network learned the design patterns and found solutions in few episodes. Moreover, it successfully finds solutions within difficult constraints. Given the FoM in every result, it indicates that the solutions it found are efficient, since the FoM value is over 850 in most cases. The FoM is calculated based on Equation 20, indicating that the solutions are efficient.

$$FoM = \frac{GBW \ \times C}{Idd} \ [MHz.pF/mA]$$

For target 0, 1, 2 and 3, the average FoM is over 850, which indicates quality in the solutions that the algorithm found. However, for target 4, the average value of FoM is slightly above 300. This value is considerably low when compared to the other targets. Target 4 compromises GBW in order to achieve a lower and feasible IDD value. Therefore, since GBW is sub optimal, that change will reflect on the FoM value, resulting in a poor FoM for Target 4. The second model (Chapter 4.2.6) produced the following results: ure of the model changed.

The number of episodes necessary to reach a valid solution for target 0 decreases on average when compared to its model counterpart. However, for the remaining Target specifications, the second model needs on average more time to reach a valid solution. This way, the second model seems to perform better for simpler solutions whereas the first model outperforms its counterpart for more complex solutions, although the difference is small. One aspect worth mentioning is that the number of episodes needed to reach a solution varies greatly depending on the utilized ”seed” value, leading to the conclusion that the seed value greatly influences the number of episodes needed to reach a solution. As for the FoM value, all the results are above 800 (except for Target 4, which is expected). There is a slight decrease in average FoM value for Target 0 when compared to the one obtained by model 1, although the difference is minimal. This observation is also applicable for targets 1 and 3. The difference between these values is not significant enough. Surprisingly, the average FoM value increases for target 2 and 4, surpassing the one obtained by the first model. It is important to illustrate how the agent evolves and learns over each episode on different occasions. With that goal in mind, several graphs were taken which show how each target specification evolves in an episode. To get a clearer view of it, different episodes in distinct stages of the sizing process are depicted. The first set is taken for one of the first episodes, the second for one for a later stage, and the third set represents how the different target parameters evolve during the episode for which the agent finds a solution. This process was done for both models and topologies.

Table 3 VCOTA: Model 2

Seed	#		IDD (µA)	DC Gain (dB)	GBW (MHz)	PM (º)	FoM² (MHz×pF/mA)
12	15	Target 0	282,1	50,073	37,26	75,367	792,401
72	4		325,6	58,129	45,96	60,478	846,799
44	15		336,4	58,144	48,12	58,541	858,187
56	27		314,7	56,318	37,02	77,097	705,892
80	23		338,3	51,148	47,62	73,330	844,450
21	14		260,0	58,465	42,65	48,043	984,204
13	32		317,8	58,128	37,79	49,306	713,579
Average	18,6		---------	---------	---------	---------	820,787
12	91	Target 1	286,5	50,966	45,46	67,870	951,973
14	639		280,1	53,292	45,05	46,934	965,030
16	36		298,8	58,106	43,00	57,185	863,319
20	43		295,1	53,818	63,35	58,740	1287,919
80	216		299,8	52,984	44,42	86,082	888,991
Average	205		---------	---------	---------	---------	991,446
12	72	Target 2	678,6	55,624	121,1	68,230	1070,738
15	431		651,4	54,481	133,1	48,729	1225,866
16	251		689,8	50,249	129,9	69,897	1130,191
19	231		593,1	50,948	124,2	52,838	1256,638
24	318		660,0	48,621	142,9	80,245	1298,823
Average	260,6		---------	---------	---------	---------	1196,451
12	353	Target 3	197,3	55,707	26,79	61,690	814,704
15	272		204,3	45,247	29,24	74,400	858,994
24	67		189,5	52,866	25,86	47,798	818,726
21	318		200,4	57,605	26,99	54,492	807,975
22	31		176,8	56,651	29,02	66,046	984,639
Average	208,2		---------	---------	---------	---------	857,007
12	353	Target 4	127,6	50,243	4,672	86,658	219,670
15	87		113,4	43,590	5,345	85,781	282,829
22	31		121,3	46,415	10,76	74,667	532,363
24	30		129,5	42,906	2,835	87,062	131,367
21	485		121,4	56,448	1,169	84,976	577,780
Average	197,2		---------	---------	---------	---------	348,802

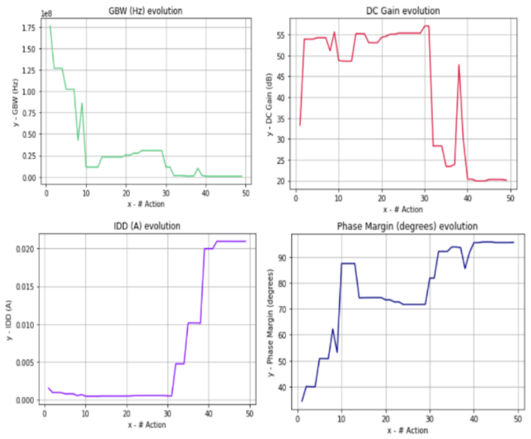

Figure 8. VCOTA Model 1 – Early Episode

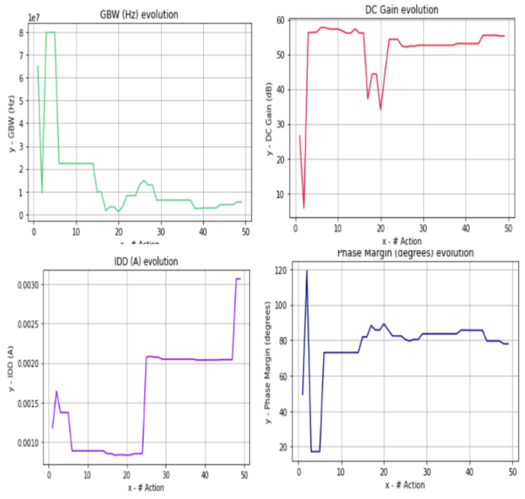

Figure 9. VCOTA Model 1 – Intermediate Episode

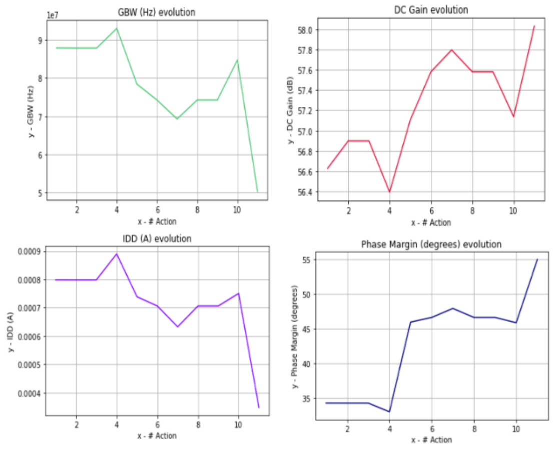

Figure 10.VCOTA Model 1 – Final Episode

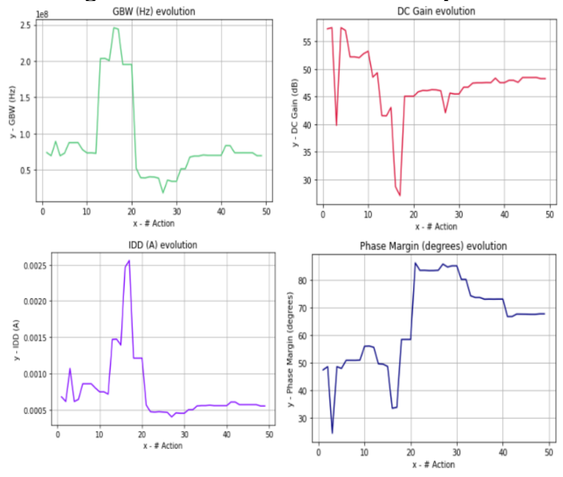

Figure 11.VCOTA Model 2 – Early Episode

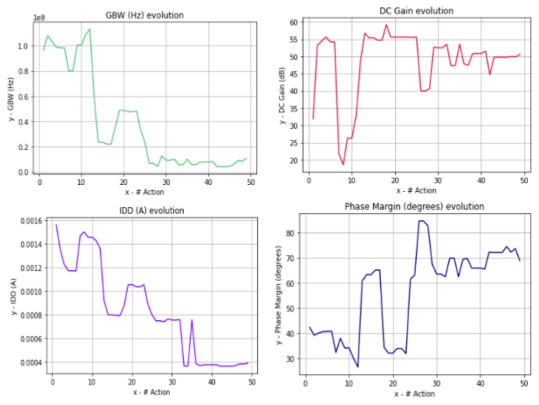

Figure 12. VCOTA Model 2 – Intermediate Episode

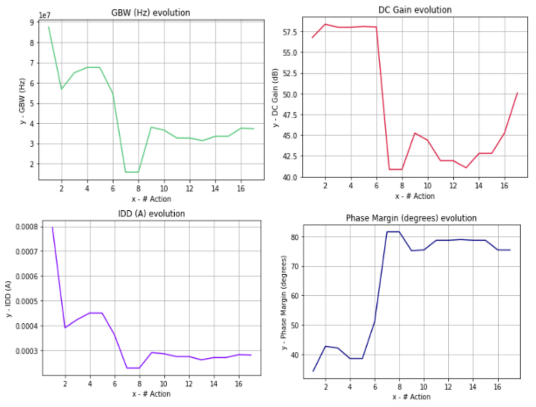

Figure 13.VCOTA Model 2 – Final Episode

Both models obtained successful results, fulfilled the target specifications and within a reasonable time. Overall, the first model performed better than the second one regarding the number of episodes it needed to reach a solution and the average FoM value. However, the difference between both models is not large and greatly depends on the seeds, leading to the conclusion that both model structures perform well when applied to the VCOTA topology.

6. CONCLUSIONS

This work presents a reinforcement learning approach – Advantage Actor-Critic - that successfully contributed to the analog IC sizing for two amplifiers: VCOTA and FCA, given their intended target performances. There has not been much research on the topic at hand. Reinforcement Learning is still being tested on analog IC sizing. The available information is not widely spread as there is little research on the subject.

The Actor-Critic model proved to be very flexible, fast, and capable of satisfying complex constraints with relative ease. This technique can prove itself extremely useful in the future as it possesses a tremendous potential. This work shows that the actor-critic approach can learn design patterns and generate circuit sizing that are correct for specification tradeoffs.

The otential of the actor-critic algorithm shows considerable promises to future applications and could prove itself extremely useful in future research. This work showed that it can adapt to different analog IC environments, namely VCOTA and FCA, proving it generalizes quite well to other more complex

7. REFERENCES

[1] The McClean Report - A complete analysis and forecast of the semicondutor industry. (2022). IC Insights. https://www.icinsights.com/data/reports/6/1/brochure.pdf?parm=1666909705.

[2] Lourenc¸o, Nuno, Ricardo Martins, and Nuno Horta. Automatic analog IC sizing and optimization constrained with PVT corners and layout effects. Cham: Springer International Publishing, 2017.

[3] Martins, Ricardo, et al.” AIDA: Robust layout-aware synthesis of analog ICs including sizing and layout generation.” 2015 International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design (SMACD). IEEE, 2015.

[4] R. F. Badaoui and R. Vemuri,” Analog VLSI circuit-level synthesis using multi-placement structures,” 2005 IEEE International Symposium on Circuits and Systems, 2005, pp. 5978-5981 Vol. 6, doi: 10.1109/ISCAS.2005.1466001.

[5] K. Deb, A. Pratap, S. Agarwal, T. MeyarivanA fast and elitist multiobjective genetic algorithm: NSGA-II IEEE Trans. Evol. Comput., 6 (2) (2002), pp. 182-197, 10.1109/4235.996017

[6] K. Deb, H. BeyerSelf-adaptive genetic algorithms with simulated binary crossover Evol. Comput., 9 (2) (2001), pp. 197-221, 10.1162/106365601750190406

[7] Li, Yaguang, et al.” A Circuit Attention Network-Based Actor-Critic Learning Approach to Robust Analog Transistor Sizing.” 2021 ACM/IEEE 3rd Workshop on Machine Learning for CAD (MLCAD). IEEE, 2021

[8] Settaluri, Keertana, et al.” AutoCkt: deep reinforcement learning of analog circuit designs.” 2020 Design, Automation & Test in Europe Conference & Exhibition (DATE). IEEE, 2020.

[9] Shi, Wei, et al.” RobustAnalog: Fast Variation-Aware Analog Circuit Design Via Multi-task RL.” arXiv preprint arXiv:2207.06412 (2022).

[10] J. Domingues, A. Gusmão, N. Horta, N. Lourenço and R. Martins, “Accelerating Voltage-Controlled Oscillator Sizing Optimizations with ANN-based Convergence Classifiers and Frequency Guess Predictors,” 2022 18th International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design (SMACD), 2022, pp. 1-4, doi: 10.1109/SMACD55068.2022.9816265.

[11] Zhao, Zhenxin, and Lihong Zhang.” Deep Reinforcement Learning for Analog Circuit Sizing.” 2020 IEEE International Symposium on Circuits and Systems (ISCAS). IEEE, 2020.

[12] Chen, Po-Yan, et al.” A Reinforcement Learning Agent for Obstacle-Avoiding Rectilinear Steiner Tree Construction.” Proceedings of the 2022 International Symposium on Physical Design. 2022.

[13] K.-W. Lin et al., “A maze routing-based methodology with bounded exploration and path-assessed retracing for constrained multilayer obstacle-avoiding rectilinear Steiner tree construction,” ACM TODAES, Vol. 23, Iss. 4, Article 45. May, 2018