Adaptive Stepping PTA for DC Analysis Based on Reinforcement Learning

Yichao Dong, *Graduate Student Member, IEEE*, Dan Niu, *Member, IEEE*, Zhou Jin, *Member, IEEE*, Chuan Zhang, *Senior Member, IEEE*, Qi Li, *Member, IEEE*, and Changyin Sun, *Senior Member, IEEE*

***Abstract*—Solving the DC operating point efficiently for large-scale nonlinear circuit is crucial and quite challenging. Pseudo transient analysis (PTA) is a widely-used and promising DC solver in the industry, in which the stepping policy is of great importance for PTA convergence and simulation efficiency. In this brief, a reinforcement learning (RL)-enhanced stepping policy is proposed. It designs dual Actor-Critic agents with stochastic policy and online adaptive scaling to intelligently evaluate PTA convergence status, and adaptively adjust forward and backward time-step size. Numerical examples demonstrate that a significant efficiency speedup and convergence improvement over the previous stepping methods is achieved by the proposed RL-enhanced stepping policy.**

***Index Terms*—Circuit simulation, DC analysis, pseudo transient analysis, reinforcement learning.**

I. Introduction

DC ANALYSIS is a vital and fundamental task in circuit simulation. It is also a precondition for further analyses in SPICE-like circuit simulators, including AC analysis, transient analysis [1], [2], [3]. With the scale of integrated circuit (IC) growing exponentially, how to efficiently solve a large set of nonlinear algebraic equations established by modified nodal analysis (MNA) [4] is quite challenging, and has emerged as a hot research topic [5], [6], [7].

The numerical iterative algorithms to solve nonlinear algebraic equations have been widely used, including the Newton-Raphson(NR)-based method, Gmin stepping, source stepping, pseudo-transient analysis (PTA) and homotopy methods [8], [9]. When solving a high-dimensional nonlinear system, the convergence of Gmin stepping and source stepping often have unsatisfactory performance [8]. In contrast, the globally convergent homotopy method is difficult to be implemented due to its high dependence on the device models [10], [11]. PTA and its variants (such as Pure PTA (PPTA), Compound element PTA (CEPTA) and Damped PTA (DPTA) [12], [13], [14]) are widely employed as alternative solutions in industrial simulators because of their ease of implementation and no discontinuation issues [15], [16].

The main principle of PTA method is to first modify the circuit by adding some pseudo elements, and then to carry out a transient analysis, starting from the initial states until a DC steady state is (hopefully) reached [13]. The PTA methods simply the DC analysis to solve the steady-state problem of a system of ordinary differential equations (ODE). The ODE is then be solved iteratively through numerical integration stepping towards the steady state. Therefore, a effective stepping policy is crucially important for the simulation efficiency and convergence performance [9], [15]. The previous works have proposed some heuristic stepping methods to speed up PTAs [16], [17]. However, they rely on manually setting formulas to tune the step size that are too general and do not consider the specificities of different circuits. Moreover, the device nonlinearity enhancement and the parasitic parameters increasing exponentially places greater demands on the PTA iterations. Designing a "intelligent" and "adaptive" stepping policy blending machine learning (ML) methods has emerged as a promising and hot research topic [18]. In [19], Bayesian optimization method has been introduced for initial parameters setting in PTA.

Reinforcement learning (RL), as an important branch of machine learning, has achieved remarkable results in many fields [20]. In the field of Electronic Design Automation (EDA), RL has been successfully applied to placement, and optimal device size selection [18], [21], [22]. In PTA iterations, the optimal step size in each PTA step is unknown and can not be manually labelled, so supervised machine learning methods do not fit it. Compared with unsupervised algorithm, RL is more suitable for solving the decision optimization problem like stepping policy in PTA iterations, which can be modeled as a Markov decision process (MDP) [23]. In this brief, we propose a RL enhanced stepping policy. It comprehensively evaluates the circuit simulation status, and generates more robust and online adaptive step size to enhance the PTA convergence and accelerate the iterative efficiency. The followings are the main novelty of this brief:

Manuscript received 5 August 2022; accepted 2 September 2022. Date of publication 16 September 2022; date of current version 22 December 2022. This work was supported in part by the Natural Science Foundation of Jiangsu Province of China under Grant BK20202006 and in part by the Commercialization of Scientific and Research findings of Jiangsu Province under Grant BA2021012. This brief was recommended by Associate Editor H. Yu. *(Corresponding authors: Dan Niu; Zhou Jin.)*

Yichao Dong and Qi Li are with the School of Automation, Southeast University, Nanjing 210096, China.

Dan Niu and Changyin Sun are with the School of Automation and the Key Laboratory of Measurement and Control of CSE, Ministry of Education, Southeast University, Nanjing 210096, China (e-mail: danniu1@163.com).

Zhou Jin is with the Super Scientific Software Laboratory, China University of Petroleum (Beijing), Beijing 102249, China (e-mail: jinzhou@cup.edu.cn).

Chuan Zhang is with LEADS and the National Mobile Communications Research Laboratory, Southeast University, Nanjing 210096, China, and also with Purple Mountain Laboratories, Nanjing 211189, China.

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCSII.2022.3207356.

Digital Object Identifier 10.1109/TCSII.2022.3207356

1) A RL-enhanced stepping policy for PTAs was proposed, which can output the adaptive time-step size by intelligently evaluating the simulation status of the pseudo circuit. It achieved a remarkable convergence enhancement and efficiency speedup over the previous stepping methods.
2) Dual Actor-Critic agents with a stochastic time-step distribution were designed. Dual agents deal with the NR convergence and non-convergence situation separately with different forward and backward stepping policies. Stochastic stepping policy achieves stronger stepping space exploration ability and helps damp out the oscillations to achieve better convergence.
3) Online adaptive scaling based on momentum was proposed to deal with the circuit differences between offline training and online prediction. We employ first and second moment estimates to extract history step size features and online dynamically adjust the step size scaling to further improve the simulation efficiency.

The proposed RL-enhanced stepping policy is compatible to kinds of PTA solvers and easy to be implemented.

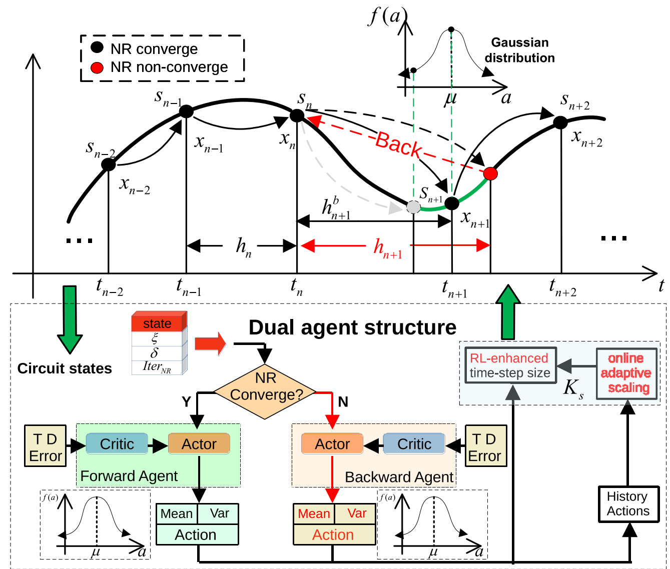

Fig. 1. Dual Actor-Critic agents with stochastic output and online adaptive scaling for PTA stepping.

II. PRELIMINARY

A. PTA and Time-Step Control Method

PTA is a quite practical and principal DC solver in industry. Describing a DC circuit, the resistive portion can be described by MNA as $F(x) = 0$, where $x = (v, i)^T$. v denotes the node voltage to the datum node and i represents the branch currents of the independent voltage sources [13]. When the target circuit is modified to the PTA case, whatever various pseudo-elements are inserted, the set of ODEs is obtained as $P(x(t), \dot{x}(t), t) = F(x) + \mathbf{D} * \dot{x}(t) = 0$.

$\dot{x}(t) = (\dot{v}(t), \dot{i}(t))$, and $\mathbf{D}$ is the incidence matrix to represent the inserted pseudo elements [13]. When the implicit DDF-k numerical integration $\dot{\boldsymbol{x}}(t)|_{t=t_{n+1}} = (\boldsymbol{x}_{n+1} - \boldsymbol{x}_{n+1-k})/\sum_{j=0}^{k-1}(\boldsymbol{h}_{n+1-j})$ [15] is employed at the discrete time point t_{n+1}, the ODEs will converge discretely to the steady-state point under numerical iterative operation. It is clear that the choice of time-step size affects the rate at which the steady state is reached or even makes the iteration non-convergence.

The traditional PTA methods adopt a simple iterative counting stepping method [15], [16], which enlarges or reduces the step size according to the NR iteration number in previous time-step or NR non-convergence. This policy is simple and fast, but how to select appropriate parameters (IMAX, IMIN, change rate, etc.) is difficult. Moreover, a adaptive time-step method based on Switched Evolution/Relaxation (SER) was proposed [17], but it is a heuristic method that relies on human expertise and has weak generalization ability.

B. Reinforcement Learning

In reinforcement learning, the agent learns from the interacting with the environment to achieve the maximum reward or special goals [20]. The PTA iterations can be treated as a MDP, which consists of five elements $\mathcal{S}$(set of states), $\mathcal{A}$(set of actions), $\mathcal{P}$(state transition probability matrix), $\mathcal{R}$(reward function) and γ(discount factor). In a MDP with policy π, the score of a certain state s can be evaluated by the state value function. The Bellman expectation function of the state value function can be obtained [20]:

$$V^{\pi}(s, a) = \mathbb{E}\left[r' + \gamma V^{\pi}(s', a') \middle| \mathcal{S}_t = s, \mathcal{A}_t = a\right], \quad (1)$$

where s, a are the current state and action. s', a', r' represent the state, action and reward of the next moment. In this brief, RL is employed to "intelligently" output an optimal stepping policy by interaction with the PTA iterations.

III. PROPOSED RL-ENHANCED STEPPING POLICY

In this section, we transform the PTA iteration process into a MDP problem. As for the RL state RL_s, it should reflect whether and how difficult the PTA iterations tends toward gradually convergence. In this brief, five states including $Iter_{NR}$, ξ, δ, C_{NR} and C_{PTA}, are employed.

$Iter_{NR}$ is the NR iteration number at each time-step. It evaluates the difficulty of NR iterations. ξ represents whether the equation is close to final solution, defined as $\xi = C_\xi \frac{\|x_n - x_{n-1}\|}{\|t_n - t_{n-1}\|}$. C_ξ is the residual coefficient. δ is the relative change rate of solution to indicate whether the solution tends toward the steady state or still changes drastically, that is $\delta = C_\delta \frac{\|x_n - x_{n-1}\|}{\|x_{n-1} - x_{n-2}\|}$, $(n \geq 2)$. C_δ is the constant coefficient.

Besides, C_{NR} and C_{PTA} are two bool flags. C_{NR} represents whether NR iterations converge. It is the key for two agents working alternately. C_{PTA} denotes whether the PTA reaches the steady state and it is a successful ending flag.

As shown in Fig. 1. The solution x_n illustrated in the top subgraph is used to obtain the simulation state s_n, which is input to the proposed dual agents network in the bottom subfigure. Then the Actor outputs stochastic action a_n and combines with the online adaptive scaling module to generate the next step size h_{n+1} or h^b_{n+1} for PTA forward or rollback steppings, by which the next time-step solution x_{n+1} shown in the top subgraph is obtained.

In this brief, conventional single agent structure of RL is not suitable for PTA iterative process. First, both forward steppings and backward steppings exist in circuit simulation, especially in DC analysis of "difficult" circuits. Forward stepping when NR iterations converge, aims to take the step size as large as possible to enhance convergence efficiency. Backward stepping needs to rollback to the previous time step and chooses smaller step size carefully to solve NR non-convergence, which is different from many RL tasks (e.g., "Undo" in most games is not allowed). Second, the proportion of training samples at forward steppings and backward steppings is highly imbalanced. Most training data are collected at NR convergence situation. Single agent hardly learns valuable backward stepping information and tends to be unstable due to the imbalanced samples, which is an obstacle for the update of Actor network. As shown in Fig. 1, two agents, that is, forward agent and backward agent, are proposed to deal with the NR convergence and non-convergence and output the different stepping policies for different situations, respectively.

Moreover, different from the previous PTA methods where deterministic stepping policy is employed, a stochastic stepping policy is proposed to achieve stronger stepping space exploration ability and helps damp out the oscillations to obtain better convergence. The output action a is a Gaussian distribution with mean and covariance given by the Actor network, where improved proximal policy optimization (PPO) [24] with online adaptive scaling is introduced. The output Gaussian action a is then normalized to the interval $(-1, 1)$ by the activation function tanh.

The policy update range is limited to make the algorithm training more stable. The objective function is as follows:

$$J_{\phi}^{\phi'}(\phi) \approx \sum_{(s_t,a_t)} \min\left(\frac{p_{\phi}(a_t \mid s_t)}{p_{\phi'}(a_t \mid s_t)} A^{\phi'}(s_t, a_t),\right.$$
$$\left.\operatorname{clip}\left(\frac{p_{\phi}(a_t \mid s_t)}{p_{\phi'}(a_t \mid s_t)}, 1-\lambda, 1+\lambda\right) A^{\phi'}(s_t, a_t)\right), \quad (2)$$

where the ϕ' and ϕ are the updated and original Actor parameters, λ is the changing limitation of the Actor network. This function modifies the surrogate objective by clipping the probability ratio. We take the minimum of the clipped and unclipped objective, so the final objective is a lower bound on the unclipped objective which can make the policy update in a limited range [24]. Note that the probability ratio is clipped at $1-\lambda$ or $1+\lambda$ depending on whether the advantage A is positive or negative. The function A is the advantage function estimated by Critic.

Next, the Gaussian action output a based the current five RL states is employed to determine the next time-step size of the PTA iteration. We design two exponential transformation equations. In the forward agent, that is

$$h_{n+1} = h_n \rho_f(a_n) = h_n \frac{m_f}{1 - e^{a_n + n_f}}. \quad (3)$$

It is noted that the equation in the backward agent is designed as:

$$h_{n+1}^b = h_n \rho_b(a_n) = h_n \prod_{i=1}^{l} \frac{m_b}{1 + e^{a_{n,i} + n_b}}, \quad (4)$$

where m_f, n_f, m_b, n_b are constant parameters, which are selected according to the set maximum and minimum change rates. l is the number of continuous NR non-convergence. In the backward stepping stage, when continuous NR non-convergence occurs, $\frac{m_b}{1+e^{a_{n,i+1}+n_b}} < 1$ holds and then a smaller time-step size than previous non-convergent step size $\prod_{i=1}^{l} \frac{m_b}{1+e^{a_{n,i}+n_b}} < \prod_{i=1}^{l-1} \frac{m_b}{1+e^{a_{n,i}+n_b}}$ can be achieved.

At last, for the reward function, the weighted sum of normalized simulation state variables and additional bias are employed. The reward function of the forward agent is

$$r_f = \sum_{i=1}^{3} C_{fi} \left\| \tilde{RL}_s \right\| + R_{fend} + C_f, \quad (5)$$

where $\tilde{RL}_s$ represents the normalized state variables, C_{fi} is the weight coefficient, R_{fend} is the additional reward value when the current PTA iteration converges, and C_f is a constant bias to make the reward value be negative.

Similarly, the reward function of the backward agent is

$$r_b = \sum_{i=1}^{3} C_{bi} \left\| \tilde{RL}_s \right\| + R_{bend} + C_b, \quad (6)$$

where R_{bend} is also the additional reward value when the current PTA iteration converges, and C_b is a constant bias to achieve the reward value be negative.

The detailed pseudo-code of the RL-enhanced PTA stepping policy is shown in the following Algorithm 1.

A. Online Adaptive Scaling

In actual circuit simulation, the structures and scales of the test circuits are different from those in the training circuit dataset. Therefore online learning and adaptively adjusting the step size for the new test circuits is quite important. Moreover, as shown in Eqs. (4) and (5), the change rate of the stepping policy trained by the offline circuit dataset is limited. For example, the change rate interval in the forward stepping stage is $(\frac{m_f}{1-e^{n_f-1}}, \frac{m_f}{1-e^{n_f+1}})$. This will bring down the simulation efficiency when continuous NR convergences occur in the test circuits, where more aggressive step size can be adopted.

In this brief, online adaptive scaling for the step size by the scaling parameter K_s is designed using the concept of momentum [25]. In the online step prediction stage for the actual test circuits, Eqs. (5) and (6) are modified as

$$h_{n+1} = K_s h_n \frac{m_f}{1 - e^{a_n + n_f}}, \quad (7)$$

$$h_{n+1}^b = K_s h_n \prod_{i=1}^{l} \frac{m_b}{1 + e^{a_{n,i} + n_b}}, \quad (8)$$

$$K_s = \frac{M_n}{\sqrt{V_n} + \epsilon}, \quad (9)$$

where ϵ is a value to make the denominator be a positive value.

The M_n is the momentum of the maximum magnification which is related with the weighted mean of first order moment, and the V_n is the weighted mean of second order moment.

Algorithm 1 RL-Enhanced PTA Stepping

Require: Important parameters setting:
Reward function r_f, r_b, step function ρ_f, ρ_b;
Learning rate α_θ, α_ϕ;
Ensure:
Connect SPICE software;
Build Critic network V_{θ_f}, V_{θ_b}, Actor network π_{ϕ_f}, π_{ϕ_b};
Simulate with initial time step h, scaling K_s;
Record state s, C_{NR}, C_{PTA}, C'_{NR};
while $C_{PTA} \neq$ **True do**
  if $C_{NR} \neq$ **False then**
    get action $a \sim \pi_{\phi_f}(a \mid \mathbf{s})$; ▷ Forward agent
    calculate next time step $h' \leftarrow \rho_f(a)h$;
    get next state s', update C'_{NR}, C_{PTA};
    calculate reward $r \leftarrow r_f(s, a)$;
    update forward trajectory $\tau_f \leftarrow \tau_f \cup \{(s, a, r)\}$;
  else
    get action $a \sim \pi_{\phi_b}(a \mid \mathbf{s})$; ▷ Backward agent
    calculate next time step $h' \leftarrow \rho_b(a)h$;
    get next state s', update C'_{NR}, C_{PTA};
    calculate reward $r \leftarrow r_b(s, a)$;
    update forward trajectory $\tau_b \leftarrow \tau_b \cup \{(s, a, r)\}$;
  end if
  if $C_{NR} \oplus C'_{NR}$ **then**
    gradient update:
    $\theta_i \leftarrow \theta_i - \alpha_\theta \nabla_{\theta_i} J_V(\theta_i)$ for $i \in \{f, b\}$;
    $\phi_i \leftarrow \phi_i - \alpha_\phi \nabla_{\phi_i} J_\pi(\phi_i)$ for $i \in \{f, b\}$;
    trajectories reset;
  end if
  update the action trajectory $\tau_a \leftarrow \tau_a \cup \{a\}$
  update the K_s by τ_a; ▷ Online adaptive scaling
  calculate the true time-step size $h' \leftarrow K_s * h'$;
  Iterate to the next step:
  $s \leftarrow s'$, $h \leftarrow h'$, $C_{NR} \leftarrow C'_{NR}$;
end while

They are designed as

$$\begin{cases} M_n = \varpi_1 M_{n-1} + (1 - \varpi_1) K_{n-1}, \\ V_n = \varpi_2 V_{n-1} + (1 - \varpi_2) \frac{1}{T} \sum_{j=n-T}^{n-1} \bar{K}_j^2, \end{cases} \tag{10}$$

where ϖ_1, ϖ_2 are the filter coefficients, the T is the filtering period. We calculate the discrete degree of the actions to represent the stability of NR iterations. In the forward stepping case, K_n and $\bar{K}_n$ are set as $\frac{m_f}{1-e^{a_n+n_f}}$ and $\frac{m_f}{1-e^{\mu_n+n_f}}$, respectively. Otherwise, K_n and $\bar{K}_n$ are $\prod_{i=1}^{l} \frac{m_b}{1+e^{a_{n,i}+n_b}}$ and $\prod_{i=1}^{l} \frac{m_b}{1+e^{\mu_{n,i}+n_b}}$, respectively. μ is the mean of the Gaussian action a. M_n and V_n online learn the history time-step size experiences of the test circuit environment.

The momentum helps accelerate the time-step size in the relevant direction and dampens oscillations by adding a fraction ϖ_1, ϖ_2 of the update vector of the past time step to the current update vector. The momentum term increases step size whose previous step magnifications are in the similar large scope and reduces step size where previous step magnifications are not stable. As a result, we obtain faster convergence and reduce oscillation.

IV. Experiments and Results

A. Experimental Environment

In this experiment, the proposed RL-enhanced stepping policy for PTAs is implemented in SPICE-like simulator, and its performance is evaluated by dozens of benchmark circuits. We conduct experiments on a 64-bit Ubuntu 18.04 computer with Intel i7-10750H CPUs, 32 GB memory. GeForce RTX 2060 GPU is used to accelerate the dual agent network computing and more computing resources are needed for the proposed stepping policy.

First, seven typical circuits (two MOS circuits and five BJT circuits) are selected as the training circuit dataset to achieve the offline pre-training of model. The DPTA with the proposed RL-enhanced stepping method conducts the DC analysis for the seven circuits. The obtained sample at each time-step is utilized to update the dual Actor-Critic network.

B. Results and Comparisons

For comprehensively evaluating the performance of the proposed RL-enhanced stepping policy, it is implemented in typical PTA method (DPTA [17], usually regarded as SOTA PTA) and compared with two widely-used and effectively time-step control methods (simple iteration counting (iter-based) stepping method [15] and SER-based adaptive stepping method [17]), which are also implemented in DPTA, respectively. 50 benchmark circuits [26] are employed as test circuit dataset, in which 17 "difficult to converge" circuits are used for convergence comparisons. Moreover, both convergence and simulation efficiency are compared.

For the simulation efficiency, the NR iteration number with the three stepping methods in DPTA for 33 test circuits are shown in Table I. From Table I, it is clear that the DPTA with the proposed RL-enahnced stepping policy has highest simulation efficiency. The DPTA with the proposed RL-enhanced stepping policy outperforms the DPTA with simple iter-based stepping method (Speedup: maximum 257.56x, average 17.90x) and the DPTA with SER-based adaptive stepping method (Speedup: maximum 257.43x, average 17.58x) in terms of NR iteration number. In general, large speedup occurs in some convergence difficult circuits, where small forward stepping sizes and large number of backward steppings usually exist by the traditional stepping methods.

Apart from simulation efficiency enhancement, convergence guarantee is actually more important especially for the large-scale circuits and some "difficult" circuits. It is highly desirable to make non-convergence cases converge. Table II gives the test results of 17 "difficult" circuit cases and the DPTA with three stepping methods are compared.

From Table II, for some circuits that the DPTA with "iter-based" and "SER-based" stepping methods do not converge, but PTA convergence can be achieved and DC solution can be obtained with the proposed RL-enhanced stepping policy. It is demonstrated that the proposed RL-enhanced stepping strategy with stochastic stepping policy and online adaptive scaling can noteworthily improve the convergence performance of the DPTA (actually almost PTAs) solvers.

TABLE I
SIMULATION EFFICIENCY COMPARISONS WITH THREE STEPPING METHODS IN DPTA

Circuits	Transistors	DPTA			speed-up	
		iter-b	SER-b	RL-e	to iter-b	to SER-b
latch	14	108	100	70	1.54X	1.43X
nagle	23	2093	1948	377	5.55X	5.18X
rca	11	104	119	60	1.73X	1.98X
ab_ac	31	3961	3947	123	32.20X	32.09X
ab_integ	31	4540	4406	177	25.65X	24.89X
ab_opamp	31	2417	2536	208	11.62X	12.19X
cram	60	130	87	55	2.36X	1.58X
e1480	28	5553	5514	179	31.02X	30.80X
gm6	5	110	227	60	1.83X	3.78X
hussamp	16	209	227	153	1.37X	1.48X
mosrect	4	838	826	103	8.14X	8.02X
mux8	64	156	118	84	1.86X	1.40X
schmitfast	6	5681	5691	129	44.04X	44.12X
slowlatch	14	9382	9353	186	50.44X	50.29X
fadd32	288	1968	1859	141	13.96X	13.18X
gm1	46	85	47	40	2.13X	1.18X
gm3	30	91	83	68	1.34X	1.22X
mike2	12	90	152	64	1.41X	2.38X
todd3	13	9353	9341	444	21.07X	21.04X
6stageLimA	19	135	116	89	1.52X	1.30X
D30	3	72	64	53	1.36X	1.201X
mosamp	16	248	239	129	1.92X	1.85X
MOSBandgap	13	342	303	166	2.06X	1.83X
MOSMEM	12	26013	26000	101	257.56X	257.43X
RCA3040	11	104	119	60	1.73X	1.98X
SCHMITT	4	60	76	43	1.340X	1.77X
TADEG	4	164	143	86	1.91X	1.66X
TADEGLOW	6	145	102	90	1.61X	1.13X
THM5	9	5331	5324	124	42.99X	42.94X
TRISTABLE	6	82	77	54	1.52X	1.43X
UA709	19	2985	375	297	10.05X	1.26X
UA727	22	813	806	208	3.91X	3.88X
UA733	11	141	152	69	2.04X	2.20X
Average	-	-	-	-	17.90X	17.58X

TABLE II
CONVERGENCE COMPARISONS WITH THREE STEPPING METHODS

Circuits	Transistors	iter-b	SER-b	RL-e
bjtff	41	--	--	176
opampal	148	--	3807	3553
optrans	528	--	--	5968
nand	25	--	--	99
pump	1	22	--	14
add20	958	--	--	336
add32	1984	--	--	169
pchip	942	1049	--	375
sram	1008	--	--	398
gm17	56	--	--	230
gm19	162	--	--	2420
jge	348	--	--	568
MOSInvchain15	30	--	329	70
REGULATOR	24	--	--	467
UA741NEG	22	--	--	271
UA741POS	22	--	--	273

V. CONCLUSION

In this brief, we propose a RL-enhanced stepping policy to intelligently evaluate PTA convergence status and adaptively adjust time-step size. Dual Actor-Critic agents with stochastic action output and online adaptive scaling are designed to enhance the model robustness and convergence. Comparing with the widely-used iter-based and SER-based stepping methods, the proposed RL-enhance stepping policy achieves significant efficiency acceleration and convergence enhancement.

REFERENCES

[1] T. Nakura, "SPICE simulation," in *Essential Knowledge for Transistor-Level LSI Circuit Design*. Singapore: Springer, 2016, pp. 19–47.

[2] J. Zhao, Y. Wen, Y. Luo, Z. Jin, W. Liu, and Z. Zhou, "SFLU: Synchronization-free sparse LU factorization for fast circuit simulation on GPUs," in *Proc. 58th ACM/IEEE Design Autom. Conf. (DAC)*, 2021, pp. 37–42.

[3] J. Deng, K. Batselier, Y. Zhang, and N. Wong, "An efficient two-level DC operating points finder for transistor circuits," in *Proc. 51st ACM/EDAC/IEEE Design Autom. Conf. (DAC)*, 2014, pp. 1–6.

[4] I. N. Hajj, "Circuit theory in circuit simulation," *IEEE Circuits Syst. Mag.*, vol. 16, no. 2, pp. 6–10, 2016.

[5] S. Uatrongjit, B. Kaewkham-Ai, and K. Prakobwaitayakitt, "Finding all DC operating points of nonlinear circuits based on interval linearization and coordinate transformation," in *Proc. Int. Elect. Eng. Congr. (iEECON)*, 2022, pp. 1–4.

[6] Y. Chen, H. Pei, X. Dong, Z. Jin, and C. Zhuo, "Application of deep learning in back-end simulation: Challenges and opportunities," in *Proc. 27th Asia South Pac. Design Autom. Conf. (ASP-DAC)*, 2022, pp. 641–646.

[7] Z. Jin *et al.*, "PALBBD: A parallel arclength method using bordered block diagonal form for DC analysis," in *Proc. Great Lakes Symp. VLSI*, 2021, pp. 327–332.

[8] E. Yilmaz and M. M. Green, "Some standard SPICE DC algorithms revisited: Why does SPICE still not converge?" in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, vol. 6, 1999, pp. 286–289.

[9] T. Najibi, "Continuation methods as applied to circuit simulation," *IEEE Circuits Devices Mag.*, vol. 5, no. 5, pp. 48–49, Jan. 1989.

[10] L. T. Watson, "Globally convergent homotopy methods: A tutorial," *Appl. Math. Comput.*, vol. 31, pp. 369–396, May 1989.

[11] D. Niu, K. Sako, G. Hu, and Y. Inoue, "A globally convergent nonlinear homotopy method for MOS transistor circuits," *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.*, vol. 95, no. 12, pp. 2251–2260, 2012.

[12] W. Weeks, A. Jimenez, G. Mahoney, D. Mehta, H. Qassemzadeh, and T. Scott, "Algorithms for ASTAP—A network-analysis program," *IEEE Trans. Circuit Theory*, vol. CT-20, no. 6, pp. 628–634, Nov. 1973.

[13] H. Yu, Y. Inoue, K. Sako, X. Hu, and Z. Huang, "An effective SPICE3 implementation of the compound element pseudo-transient algorithm," *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.*, vol. 90, no. 10, pp. 2124–2131, 2007.

[14] X. Wu, Z. Jin, D. Niu, and Y. Inoue, "A PTA method using numerical integration algorithms with artificial damping for solving nonlinear DC circuits," *Nonlinear Theory Appl.*, vol. 5, no. 4, pp. 512–522, 2014.

[15] F. N. Najm, *Circuit Simulation*. Hoboken, NJ, USA: Wiley, 2010.

[16] R. M. Kielkowski, *Inside Spice*, vol. 2. New York, NY, USA: McGraw-Hill, 1998.

[17] X. Wu, Z. Jin, D. Niu, and Y. Inoue, "An adaptive time-step control method in damped pseudo-transient analysis for solving nonlinear DC circuit equations," *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.*, vol. 100, no. 2, pp. 619–628, 2017.

[18] G. Huang *et al.*, "Machine learning for electronic design automation: A survey," *ACM Trans. Design Autom. Electron. Syst.*, vol. 26, no. 5, pp. 1–46, 2021.

[19] W. W. Xing, X. Jin, Y. Liu, D. Niu, W. Zhao, and Z. Jin, "BOA-PTA, a Bayesian optimization accelerated error-free spice solver," 2021, *arXiv:2108.00257*.

[20] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2018.

[21] H. Wang *et al.*, "GCN-RL circuit designer: Transferable transistor sizing with graph neural networks and reinforcement learning," in *Proc. 57th ACM/IEEE Design Autom. Conf. (DAC)*, 2020, pp. 1–6.

[22] A. Hosny, S. Hashemi, M. Shalan, and S. Reda, "DRILLs: Deep reinforcement learning for logic synthesis," in *Proc. IEEE 25th Asia South Pac. Design Autom. Conf. (ASP-DAC)*, 2020, pp. 581–586.

[23] R. Bellman, "A Markovian decision process," *J. Math. Mech.*, vol. 6, no. 5, pp. 679–684, 1957.

[24] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017, *arXiv:1707.06347*.

[25] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.

[26] J. Barby and R. Guindi, "CircuitSim93: A circuit simulator benchmarking methodology case study," in *Proc. IEEE 6th Annu. Int. ASIC Conf. Exhibit*, 1993, pp. 531–535.