2019 IEEE Symposium Series on Computational Intelligence (SSCI)
December 6-9 2019, Xiamen, China

Parallel Adaptive Critic Designs of Optimal Control for Ice-Storage Air Conditioning Systems

Zehua Liao, Qinglai Wei
The State Key Laboratory of Management and Control for Complex Systems
Institute of Automation, Chinese Academy of Sciences
Beijing, China
Email: {liaozehua2017, qinglai.wei}@ia.ac.cn

Ruizhuo Song
School of Automation and Electrical Engineering
University of Science and Technology Beijing
Beijing, China
Email: ruizhuosong@ustb.edu.cn

***Abstract*—As a kind of distributed energy storage system, ice-storage air conditioning (IAC) system is an effective way for transferring cooling load. A parallel adaptive critic design method is constructed for solving the optimal control problem of IAC systems. Adaptive critic design is also called adaptive dynamic programming (ADP). The initial value function of the parallel ADP method is obtained by using particle swarm optimization (PSO) method to pre-train the weights of the neural networks. Another PSO algorithm is utilized in each iteration to achieve the target control, which avoids solving a nonlinear equation in traditional ADP algorithms. Finally, the efficiency of the parallel ADP algorithm will be illustrated through the experiment results.**

***Index Terms*—Parallel adaptive critic design, adaptive dynamic programming, particle swarm optimization, ice-storage air conditioning.**

I. Introduction

In recent years, IAC systems have been widely used over the world due to its outstanding characteristics that, it can store cooling by making ice during off-peak period and melt it in on-peak period to release cooling [1]. IAC is of great significance for the demand side management.

As IAC system has been widely implemented in the real-world applications [2]–[4]. The optimization of IAC system has become the mainstream research object. Some typical optimization methods for IAC have been proposed [5]–[7]. The difficulty for the optimization of IAC lies in the controller design of the system. It is difficult to built the precise models for the IAC system. Therefore, a heuristic algorithm is needed.

With the development of control technology, parallel control theory has become the guiding ideology for solving complex control problems by combining theoretical research, scientific experiment and computational technology [8]–[10]. Adaptive critic design is in the category of parallel control [8]. ADP is another name of adaptive critic design, which is proposed in [11], [12]. Now ADP has been introduced in the field of energy management [13]–[16], and performs well. However, the traditional ADP algorithms has the problem of huge amount of iterations. Inspired by references [8] and [17], we consider designing a parallel self-learning optimal control method.

For solving the optimal control problem of IAC systems, a new parallel self-learning algorithm is developed. The algorithm is formed by the combination of ADP and parallel control theory. In the developed algorithm, the ADP algorithm and the PSO algorithm are implemented in parallel. Next, the detailed iteration structure is displayed.

II. Problem Formulations

A. IAC System Description

The IAC system is composed of an air conditioning refrigerator, a cold storage equipment, a cooling converter and a cooling management unit, which makes cooling inter-exchange available. In the IAC system, the cold storage equipment adopts different control strategies to meet the cooling load demand. There are three operational modes for the IAC system under consideration: store mode, idle mode and release mode.

Based on the main operation patters, the IAC model is formulated as

$$R_I(\tau+1) = R_I(\tau) - r_I(\tau) \times 1[\mathrm{h}], \tag{1}$$

where τ is the time index, $R_I(\tau)$(kWh) is the residual cooling capacity of the cold storage equipment, and $r_I(\tau)$(kW) is the cooling capacity output of the cold storage equipment. Here, $r_I(\tau) < 0$ means storing cooling, $r_I(\tau) > 0$ means releasing cooling, and $r_I(\tau) = 0$ means the cold storage equipment is idle.

For the safe use of the cold storage equipment, the following physical constraints should be taken into consideration. The residual cooling capacity $R_I(\tau)$ satisfies $R_{I\min} \le R(\tau) \le R_{I\max}$, the cooling output $r_I(\tau)$ satisfies $r_{I\min} \le R(\tau) \le r_{I\max}$. The cooling load demand is met by the air conditioning refrigerator and the cold storage equipment. Then, we have

$$r_L(\tau-1) = r_I(\tau-1) + r_C(\tau). \tag{2}$$

where $r_L(\tau)$ is the cooling load demand, and $r_C(\tau)$ is the cooling capacity output of the refrigerator. Here, we introduce delays in $r_L(\tau)$ and $r_I(\tau)$, for convenience of analysis [18]. According to the actual situation, we let $r_C(\tau) \ge 0$.

This work was supported in part by the National Natural Science Foundation of China under Grants 61722312, 61673054, 61533017, and in part by National Key Research and Development Program of China under Grant 2018YFB1702300.

B. Optimization Objectives and Optimality Principles

Then, we let $x_1(\tau) = r_C(\tau)$, $x_2(\tau) = R_I(\tau)$. Let $x(\tau) = [x_1(\tau), x_2(\tau)]^T$, $u(\tau) = r_I(\tau)$. The discrete nonlinear system is given according to the air conditioning model as

$$x(\tau+1) = F\left[x(\tau), u(\tau), \tau\right] = \begin{pmatrix} r_L(\tau) - u(\tau) \\ x_2(\tau) - u(\tau) \end{pmatrix}. \quad (3)$$

The utility function is defined as

$$U[x(\tau), u(\tau), \tau] = n_1 \times \left(\frac{x_1(\tau)}{\eta} \times C(\tau)\right)^2 + n_2 \times \left(R_{I\max} - x_2(\tau)\right)^2. \quad (4)$$

where $F(\cdot)$ is the system function, $C(\tau)$(CNY/kWh) is the electricity rate, n_1 and n_2 are constants. Based on the utility function, the performance index function is given by

$$J\left[x(\tau), \tau\right] = \sum_{\kappa=\tau}^{\infty} \gamma^{\kappa-\tau} U\left[x(\kappa), u(\kappa), \kappa\right], \quad (5)$$

where γ is the discount factor. The research object is to find the optimal control which minimize (5).

III. Parallel Self-Learning Optimal Control for IAC System

A. PSO Algorithm

In the work of Eberhart and Kennedy [20], [21], Particle swarm optimization (PSO) is proposed. For the PSO algorithm used here, we let θ be the swarm size, $l = 1, 2, \ldots, \theta$. Let $p(l, \tau)$ and $v(l, \tau)$ denote the position and the velocity at time τ of the particles, respectively. Here, the position vector corresponds to $u(\tau)$ in system (3). We let p_g denote the best position, p_l denote the best position of particles. Then the update rule is shown as

$$p(l, \tau) = p(l, \tau-1) + v(l, \tau) \quad (6)$$

and

$$v(l, \tau) = \omega v(l, \tau-1) + \rho_1 \varphi_1 \left(p_l - p(l, \tau-1)\right) + \rho_2 \varphi_2 \left(p_g - p(l, \tau-1)\right), \quad (7)$$

where ω is the inertia factor, ρ_1 and ρ_2 are the correction factors, φ_1 and φ_2 are the random numbers.

The utility function of the PSO algorithm to calculate the fitness value is defined as

$$U_{PSO}(\tau) = \sqrt{n_1 \times \delta_1^2 + n_2 \times \delta_2^2}, \quad (8)$$

where

$$\delta_1 = \frac{r_L(\tau) - p(l, \tau)}{\eta} \times \frac{C(\tau)}{\min C(\tau)}$$

and

$$\delta_2 = R_{I\max} - \left(R_I(\tau) - p(l, \tau)\right).$$

B. Parallel ADP Algorithm

Inspired by [18], the parallel ADP algorithm is developed based on action-dependent heuristic dynamic programming (ADHDP), which is one of the ADP approaches.

We defined an optimal action-dependent value function $V^*\left[x(\tau), u(\tau), \tau\right]$, such that

$$\min_{u(\tau)} V^*\left[x(\tau), u(\tau), \tau\right] = J^*\left[x(\tau), \tau\right]. \quad (9)$$

where $J^*\left[x(\tau), \tau\right]$ denotes the optimal performance index function. Based on [22], [23], we can obtain

$$V^*\left[x(\tau), u(\tau), \tau\right] = U\left[x(\tau), u(\tau), \tau\right] + \gamma \min_{u(\tau+1)} V^*\left[x(\tau+1), u(\tau+1), \tau+1\right]. \quad (10)$$

From (10), it can be found that the optimal action-dependent value function depends on the control input. Thus, we can obtained the optimal control without the mathematical expressions of the IAC system. However, it can be found that $V^*\left[x(\tau), u(\tau), \tau\right]$ is difficult to get from (10), because it is time-varying. Hence, the following derivation is necessary.

Here we assume that $r_L(\tau)$ and $C(\tau)$ are periodic functions with the period $\mathcal{T} = 24$h. According to the assumption, for $\tau = 0, 1, \ldots$, we define $\alpha = 0, 1, \ldots$ and $\beta = 0, 1, \ldots, \mathcal{T}-1$, that satisfy $\tau = \alpha\mathcal{T} + \beta$. Define $\kappa = \alpha\mathcal{T}$. Then, there are $r_L(\tau) = r_L(\kappa+\beta) = r_L(\beta)$ and $C(\tau) = C(\kappa+\beta) = C(\beta)$. Let $\Gamma(\kappa) = (u(\kappa), u(\kappa+1), \ldots, u(\kappa+\mathcal{T}-1))$ denote the control sequence. Then, for all $\kappa \in \{0, \mathcal{T}, 2\mathcal{T}, \ldots\}$, the utility function can be expressed as

$$\Lambda\left[x(\kappa), \Gamma(\kappa)\right] = \sum_{\beta=0}^{\mathcal{T}-1} \gamma^{\beta} U\left[x(\kappa+\beta), u(\kappa+\beta), \beta\right]. \quad (11)$$

Hence, combined with (10), there is

$$V^*\left[x(\kappa), \Gamma(\kappa)\right] = \Lambda\left[x(\kappa), \Gamma(\kappa)\right] + \gamma^{\mathcal{T}} \min_{\Gamma(\kappa+\mathcal{T})} V^*\left[x(\kappa+\mathcal{T}), \Gamma(\kappa+\mathcal{T})\right], \quad (12)$$

where $V^*\left[x(\kappa), \Gamma(\kappa)\right]$ is called optimal value iteration. Hence, the optimal control sequence can be obtained by

$$\Gamma^*(\kappa) = \arg\min_{\Gamma(\kappa)} V^*\left[x(\kappa), \Gamma(\kappa)\right]. \quad (13)$$

According to the above analysis, the parallel ADP algorithm is designed. In the developed algorithm, there are two iteration processes, including external iteration and internal iteration. The first iteration is the external iteration. Here, we define $i = 0, 1, \ldots$ as the index of the external iteration. Let $\phi\left[x(\kappa), u(\kappa)\right]$ denote the initial iterative function for the parallel ADP algorithm, which is usually set as zero in traditional iterative ADP algorithms [18], [24], [25]. Here, the PSO method is used to obtain a suboptimal value function. For $D = \alpha\mathcal{T} + \beta$, implementing PSO algorithm to get the control sequence. For

$\beta = 0, 1, \ldots, D-1$, letting $u(\kappa+\beta) = p_g(\kappa+\beta)$, then we obtain

$$\phi\left[x(\kappa), u(\kappa)\right] = \sum_{\beta=0}^{D-1} \gamma^{\beta} U\left[x(\kappa+\beta), u(\kappa+\beta), \beta\right]. \quad (14)$$

Based on the initial iterative function, the initial iterative value function $V_0\left[x(\kappa), \Gamma(\kappa)\right]$ is defined as

$$V_0\left[x(\kappa), \Gamma(\kappa)\right] = \Lambda\left[x(\kappa), \Gamma(\kappa)\right] + \gamma^{\mathcal{T}} \min_{\Gamma(\kappa+\mathcal{T})} \phi\left[x(\kappa+\mathcal{T}), \Gamma(\kappa+\mathcal{T})\right]. \quad (15)$$

Based on (13), the iterative control sequence is computed as

$$\Gamma_0\left(x(\kappa)\right) = \arg\min_{\Gamma(\kappa)} V_0\left[x(\kappa), \Gamma(\kappa)\right]. \quad (16)$$

Then, for $i = 1, 2, \ldots$, the update rule of external iteration process are expressed as the following iteration equations, which are

$$\begin{aligned} & V_i\left[x(\kappa), \Gamma(\kappa)\right] \\ & = \Lambda\left[x(\kappa), \Gamma(\kappa)\right] + \gamma^{\mathcal{T}} \min_{\Gamma(\kappa+\mathcal{T})} V_{i-1}\left[x(\kappa+\mathcal{T}), \Gamma(\kappa+\mathcal{T})\right] \\ & = \Lambda\left[x(\kappa), \Gamma(\kappa)\right] + \gamma^{\mathcal{T}} V_{i-1}\left[x(\kappa+\mathcal{T}), \Gamma_{i-1}\left(x(\kappa+\mathcal{T})\right)\right] \end{aligned} \quad (17)$$

and

$$\Gamma_i\left(x(\kappa)\right) = \arg\min_{\Gamma(\kappa)} V_i\left[x(\kappa), \Gamma(\kappa)\right], \quad (18)$$

respectively.

The second iteration process is the internal iteration. Here, we define $j = 0, 1, \ldots, \mathcal{T}-1$, as the index of the internal iteration. For $j = 0$ and $i = 0$, let the initial iterative performance index be

$$V_0^0\left[x(\kappa), u(\kappa)\right] = \phi\left[x(\kappa), u(\kappa)\right]. \quad (19)$$

Then, for $i = 0$ and $j = 0, 1, \ldots, \mathcal{T}-1$, the internal iteration is proceeded between

$$u_0^j\left(x(\kappa)\right) = \arg\min_{u(\kappa)} V_0^j\left[x(\kappa), u(\kappa)\right] \quad (20)$$

and

$$\begin{aligned} & V_0^{j+1}\left[x(\kappa), u(\kappa)\right] \\ & = U\left[x(\kappa), u(\kappa), j\right] + \gamma \min_{u(\kappa+1)} V_0^j\left[x(\kappa+1), u(\kappa+1)\right] \\ & = U\left[x(\kappa), u(\kappa), j\right] + \gamma V_0^j\left[x(\kappa+1), u_0^j\left(x(\kappa+1)\right)\right], \end{aligned} \quad (21)$$

where we let $x(\kappa+1) = \begin{pmatrix} r_L(\mathcal{T}-1-j) - u(\kappa) \\ x_2(\kappa) - u(\kappa) \end{pmatrix}$. Here the utility function is expressed as

$$U\left[x(\kappa), u(\kappa), j\right] = n_1 \times \left(\frac{x_1(\kappa)}{\eta} \times C(\mathcal{T}-1-j)\right)^2 + n_2 \times \left(R_{I\max} - x_2(\kappa)\right)^2. \quad (22)$$

For $i = 1, 2, \ldots$, let $V_i^0\left[x(\kappa), u(\kappa)\right] = V_{i-1}^{\mathcal{T}}\left[x(\kappa), u(\kappa)\right]$. Then, for $j = 0, 1, \ldots, \mathcal{T}-1$, the internal iteration is proceeded between

$$u_i^j\left(x(\kappa)\right) = \arg\min_{u(\kappa)} V_i^j\left[x(\kappa), u(\kappa)\right] \quad (23)$$

and

$$\begin{aligned} & V_i^{j+1}\left[x(\kappa), u(\kappa)\right] \\ & = U\left[x(\kappa), u(\kappa), j\right] + \gamma \min_{u(\kappa+1)} V_i^j\left[x(\kappa+1), u(\kappa+1)\right] \\ & = U\left[x(\kappa), u(\kappa), j\right] + \gamma V_i^j\left[x(\kappa+1), u_i^j\left(x(\kappa+1)\right)\right]. \end{aligned} \quad (24)$$

Hence, the iterative control sequence is obtained by

$$\Gamma_i\left(x(\kappa)\right) = \left\{u_i^{\mathcal{T}-1}(\kappa), u_i^{\mathcal{T}-2}(\kappa), \ldots, u_i^0(\kappa)\right\}. \quad (25)$$

IV. Neural Network Training for the Parallel ADP Algorithm

A. Action Network

The developed ADP algorithm is implemented by using two neural networks, Action Network and Critic Network, to provide the control signals and criticize the action network performance, respectively. Both networks adopt three-layer BP network model.

For the action network, the input vector $x(\kappa)$ is the state vector in (3). We define W_{a_1} and W_{a_2} as the input-hidden weight matrix and the hidden-output weight matrix, respectively. Let b_a be the threshold value. Define the training step as $\lambda = 0, 1, \ldots$. Then the output of the network can be obtained by

$$\hat{u}_i^j\left(\lambda, x(\kappa)\right) = \left(W_{a_2 i}^j(\lambda)\right)^T f\left(Y_a\left(x(\kappa)\right)\right), \quad (26)$$

where $Y_a\left(x(\kappa)\right) = (W_{a_1})^T x(\kappa) + b_a$, and the activate function $f\left(x\right) = \frac{1}{1+e^{-x}}$. Based on the work in [27], the update rule of the network weight is displayed as follows.

$$W_{a_2 i}^j(\lambda+1) = W_{a_2 i}^j(\lambda) - \mu_a \frac{\partial E_{ai}^j(\lambda)}{\partial W_{a_2 i}^j(\lambda)}, \quad (27)$$

where

$$E_{ai}^j(\lambda) = \frac{1}{2}\left(\hat{u}_i^j\left(\lambda, x(\kappa)\right) - u_i^j\left(x(\kappa)\right)\right)^2, \quad (28)$$

and μ_a denotes the learning rate.

In (28), a target control $u_i^j\left(x(\kappa)\right)$ is necessary to train the network, while the target control is expressed in (23). For traditional ADP algorithms [13]–[16], [18], [24], [25], the target control $u_i^j\left(x(\kappa)\right)$ in each iteration can be obtained by

$$\frac{\partial V_i^j\left[x(\kappa), u(\kappa)\right]}{\partial u(\kappa)} = 0. \quad (29)$$

As $V_i^j\left[x(\kappa), u(\kappa)\right]$ is approximated by critic network, which is generally a nonlinear function, the analytical solution of $u_i^j\left(x(\kappa)\right)$ is nearly impossible. Traditional numerical solution in [26] for the equation (29) will expend large amount of computation resource. In this section, PSO algorithm is employed for the first time to obtain the target control $u_i^j\left(x(\kappa)\right)$.

B. Critic Network

Based on updated control law by the action network training, the critic network can be improved. For the critic network, we let $X_c(\kappa) = \left[(x(\kappa))^T, u(\kappa)\right]^T$ be the input vector. Define W_{c_1} as the input-hidden weight matrix, and define W_{c_2} as the hidden-output weight matrix. Let b_c be the threshold value. Then the output of the network can be obtained by

$$\hat{V}_i^{j+1}(\lambda, x(\kappa), u(\kappa)) = \left(W_{c_2 i}^j(\lambda)\right)^T f\left(Y_c\left(x(\kappa)\right)\right), \tag{30}$$

where $Y_c\left(x(\kappa)\right) = (W_{c_1})^T X_c(\kappa) + b_c$. The update rule of the network weight is

$$W_{c_2 i}^j(\lambda+1) = W_{c_2 i}^j(\lambda) - \mu_c \frac{\partial E_{ci}^j(\mathcal{T})}{\partial W_{c_2 i}^j(\mathcal{T})}, \tag{31}$$

where

$$E_{ci}^j(\lambda) = \frac{1}{2}\left(\hat{V}_i^{j+1}(\lambda, x(\kappa), u(\kappa)) - V_i^{j+1}(x(\kappa), u(\kappa))\right)^2,$$

and μ_c denotes the learning rate.

V. Numerical Example

A. Simulation Preparation

Before the simulation, the following settings are required.

- The IAC system should meet the demand of cooling load at any time.
- The simulation period is 4 days (96 h).
- $R_{I\max} = 50000$ kWh, $R_{I\min} = 10000$ kWh, $R_I(0) = 20000$ kWh.
- $r_{I\max} = 8000$ kW, and $r_{I\min} = -8000$ kW.
- $n_1 = 1$, and $n_2 = 0.1$.
- $\eta = 2.5$.
- $\gamma = 0.95$.
- $\varepsilon = 10^{-5}$.
- $\mu_a = \mu_b = 0.01$.

The cooling load demand and the real-time electricity rate are shown in Fig. 1 and Fig. 2, respectively.

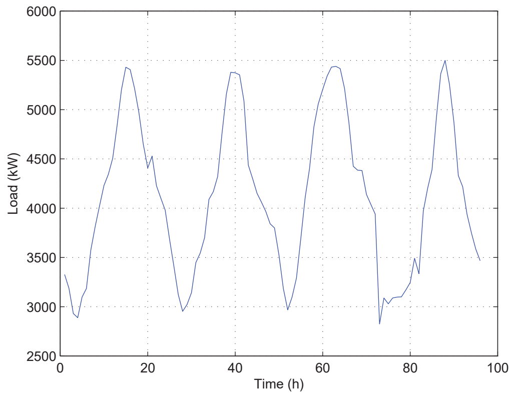

Fig. 1. The cooling load demand in 96 h

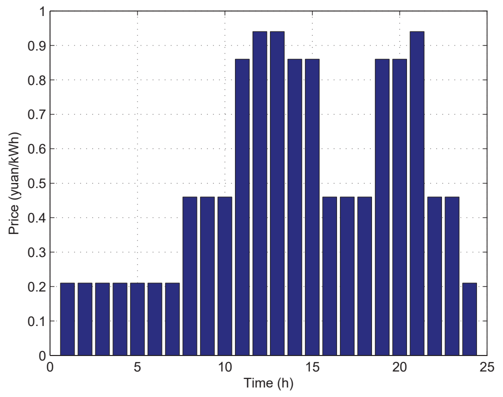

Fig. 2. The daily real-time electricity rate

B. Results and Analysis

The convergence of the V-function is shown in Fig. 3, where "optimal V-function" denotes the optimal iterative value function. We can find that the V-function converges after 5 external iterations.

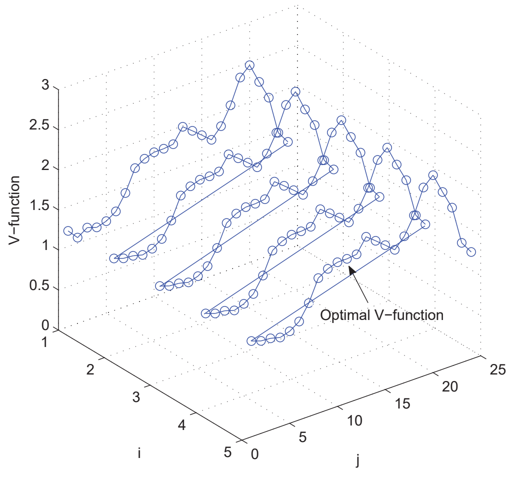

Fig. 3. Convergence trajectory of parallel ADP algorithm

The optimal control scheme of the IAC system is shown in Fig. 4. We can see that when the electricity price is low and cooling load demand is low, such as the midnight time, the IAC stores the cooling. While the cooling load demand and the electricity rate are both high, the IAC releases the cooling to meet demand, which reduces the cost from the power grid. For each hour in a day, the cooling storage/releasing power can also be obtained. From the optimal scheduling shown in Fig. 4, the correctness of the theoretical results can be verified.

To illustrate the superiority of the parallel ADP algorithm, a traditional ADP algorithm is employed for comparison,

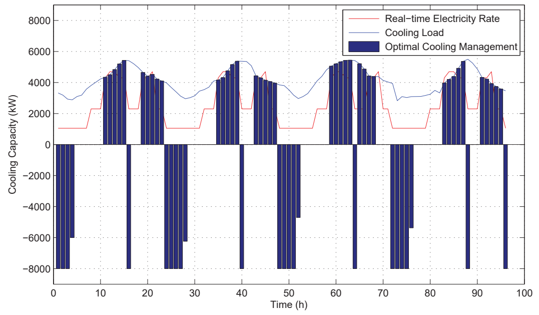

Fig. 4. Optimal scheduling of IAC system in 96 h

which is the dual Q-learning algorithm [18]. The convergence trajectory of the dual Q-learning algorithm is displayed in Fig. 5. The numerical results are shown in Table I, where "Dual QL" denotes the dual Q-learning algorithm and "PADP" denotes the developed parallel ADP algorithm. From Fig. 5, we can see that the dual Q-learning algorithm converges after 20 iterations, while the parallel ADP algorithm converges after 5 iterations. The numerical results in Table I show that the developed algorithm greatly reduces the amount of computation. The pre-train process by using PSO increases the algorithm performance.

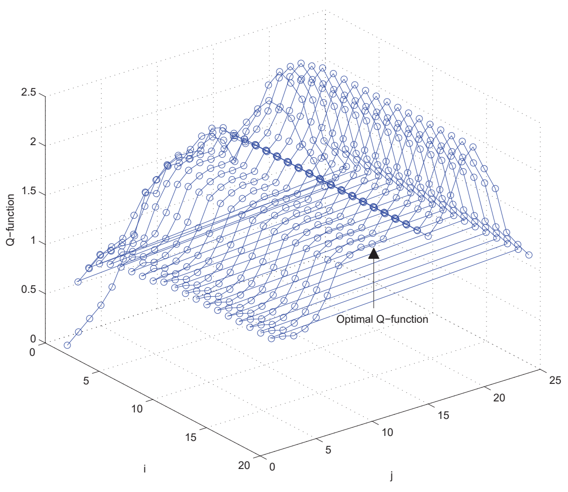

Fig. 5. Convergence trajectory of dual Q-learning algorithm

TABLE I
COST COMPARISON

	Original	Dual QL	PADP
Cost (CNY)	99033.09	58282.18	57602.91
Saving rate		41.15%	41.84%
Iteration No.		20	5

VI. CONCLUSION

A parallel ADP algorithm is constructed in this paper for the IAC system to find optimal scheduling, that minimizes the cost of the IAC system. For the first time, the idea of parallel control is applied to obtain the optimal control of IAC system. The results of the optimal control scheme indicate that the developed algorithm is effective in minimizing the overall cost of the IAC system, and the comparison results also prove the superiority.

REFERENCES

[1] Q. Wu, J. Zheng, and Z. Jing, "Coordinated scheduling of energy resources for distributed DHCs in an integrated energy grid," *CSEE Journal of Power and Energy Systems*, vol. 1, no. 1, pp. 95–103, Mar. 2015.

[2] B. H, Y. Tang, X. Liu, and W. Cheng, "Ice storage air-conditioning system design and evaluation for Taizhou electric power control center," in *Proceedings of the International Conference on Electric Technology and Civil Engineering*, Lushan, China, Apr. 2011, pp. 2296–2299.

[3] W. Hu, C. C. Chai, W. Yang, and R. Yu, "Comprehensive modeling and joint optimization of ice thermal and battery energy storage with provision of grid services," in *Proceedings of TENCON 2017: IEEE Region 10 Conference*, Penang, Malaysia, Nov. 2017, pp. 528–533.

[4] S. Rahnama, J. D. Bendtsen, J. Stoustrup, and H. Rasmussen, "Robust aggregator design for industrial thermal energy storages in smart grid," *IEEE Transactions on Smart Grid*, vol. 8, no. 2, pp. 902–916, Mar. 2017.

[5] S. Sanaye and A. Shirazi, "Four E analysis and multi-objective optimization of an ice thermal energy storage for air-conditioning applications," *International Journal of Refrigeration*, vol. 36, no. 3, pp. 828–841, May. 2013.

[6] W.-S Lee, Y.-T Chen, and T.-H Wu, "Optimization for ice-storage air-conditioning system using particle swarm algorithm," *Applied Energy*, vol. 86, no. 9, pp. 1589–1595, Sep. 2009.

[7] M. Zhang and Y. Gu, "Optimization of ice-storage air conditioning system with ASAGA," in *Proceedings of IEEE Workshop on Advanced Research and Technology in Industry Applications*, Ottawa, Canada, Sept. 2014, pp. 1042–1046.

[8] F.-Y. Wang, J. Zhang, and Q. Wei, "PDP: parallel dynamic programming," *IEEE/CAA Journal of Automatica Sinica*, vol. 4, no.1, pp. 1–5, Jan. 2017.

[9] F.-Y. Wang, "Control 5.0: From Newton to Merton in popper's cyber-social-physical spaces," *IEEE/CAA Journal of Automatica Sinica*, vol. 3, no. 3, pp. 233–234, Jun. 2016.

[10] Y. Song, X. He, Z. Liu, W. He, C. Sun, and F.-Y. Wang, "Parallel control of distributed parameter systems," *IEEE Transactions on Cybernetics*, vol. PP, no. 99, pp. 1–11, Early Access 2018.

[11] P. J. Werbos, "Advanced forecasting methods for global crisis warning and models of intelligence," *General Systems Yearbook*, vol. 22, pp. 25–38, 1977.

[12] P. J. Werbos, "A menu of designs for reinforcement learning over time," in: *Neural Networks for Control*, W. T. Miller, R. S. Sutton and P. J. Werbos, Eds. Cambridge: MIT Press, 1991, pp. 67–95.

[13] Q. Wei, D. Liu, F. L. Lewis, and Y. Liu, "Mixed iterative adaptive dynamic programming for optimal battery energy control in smart residential microgrids," *IEEE Transactions on Industrial Electronics*, vol. 64, no. 5, pp. 4110–4120, May. 2017.

[14] Q. Wei, D. Liu, Y. Liu, and R. Song, "Optimal constrained self-learning battery sequential management in microgrid via adaptive dynamic programming," *IEEE/CAA Journal of Automatica Sinica*, vol. 4, no. 2, pp. 168–176, Jun. 2017.

[15] Q. Wei, G. Shi, R. Song, and Y. Liu, "Adaptive dynamic programming-based optimal control scheme for energy storage systems with solar renewable energy," *IEEE Transactions on Industrial Electronics*, vol. 64, no. 7, pp. 5468–5478, July. 2017.

[16] X. Yang, H. He, and X. Zhong, "Adaptive dynamic programming for robust regulation and its application to power systems," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 7, pp. 5722–5732, Jul. 2018.

[17] D. Fuselli, F. D. Angelis, and M. Boaro, “Action dependent heuristic dynamic programming for home energy resource scheduling,” *International Journal of Electrical Power & Energy Systems*, vol. 48, no. 1, pp. 148–160, Jun. 2013.

[18] Q. Wei, D. Liu, and G. Shi, “A Novel Dual Iterative Q-Learning Method for Optimal Battery Management in Smart Residential Environments,” *IEEE Transactions on Industrial Electronics*, vol. 62, no. 4, pp. 2509–2518, Apr. 2015.

[19] R. E. Bellman, *Dynamic programming*. Princeton, NJ: Princeton University Press, 1957.

[20] J. Kennedy and R. Eberhart, “Particle swarm optimization,” in *Proceedings IEEE International Conference on Neural Networks*, Perth, Australia, Nov. 1995, pp. 1942–1948.

[21] R. Eberhart and J. Kennedy, “A new optimizer using particle swarm theory,” in *Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, Nagoya, Japan, Oct. 1995, pp. 39–43.

[22] C. Watkins, “Learning from Delayed Rewards,” Ph.D. dissertation, Cambridge Univ., Cambridge, U.K., 1989.

[23] C. Watkins and P. Dayan, “Q-learning,” *Machine Learning*, vol. 8, no. 3/4, pp. 279–292, May 1992.

[24] A. Al-Tamimi, F. L. Lewis, and M. Abu-Khalaf, “Discrete-time nonlinear HJB solution using approximate dynamic programming: Convergence proof,” *IEEE Trans. Syst., Man, Cybern. Part B, Cybern.*, vol. 38, no. 4, pp. 943–949, Aug. 2008.

[25] Q. Wei, D. Liu, and H. Lin, “Value iteration adaptive dynamic programming for optimal control of discrete-time nonlinear systems,” *IEEE Transactions on Cybernetics*, vol. 46, no. 3, pp. 840–853, Mar. 2016.

[26] J. J. Learder, *Numerical Analysis and Scientific Computaion*. Upper Saddle River, New Jersey: Prentice Hall Inc., 2008.

[27] J. Si and Y.-T. Wang, “On-line learning control by association and reinforcement,” *IEEE Trans. Neural Network*, vol. 12, no. 2, pp. 264–276, Mar. 2001.