Engineering with Computers manuscript No.
(will be inserted by the editor)

Submitted to Engineering with Computers

Reinforcement Learning for Optimum Design of a Plane Frame under Static Loads

Kazuki Hayashi · Makoto Ohsaki

Received: date / Accepted: date

Abstract A new method is presented for optimum cross-section design of planar frame structures combining reinforcement learning (RL) and metaheuristics. The method starts from RL jointly using artificial neural network (ANN) so that the action taker, or the agent, can choose a proper action on which members to increase, reduce or keep their size. The size of the neural network is compressed into small numbers of inputs and outputs utilizing story-wise decomposition of the frame. The trained agent is used in the process of generating a neighborhood solution during optimization with simulated annealing (SA) and particle swarm optimization (PSO). Because the proposed method is able to explore the solution space efficiently, better optimal solutions can be found with less computational cost compared with those obtained solely by metaheuristics. Utilization of RL agent also leads to high-quality optimal solutions regardless of variation of parameters of SA and PSO or initial solution. Furthermore, once the agent is trained, it can be applied to optimization of other frames with different numbers of stories and spans.

Keywords Steel frame · Cross-section optimization · Machine learning · Reinforcement learning · Q-learning · Artificial neural network

Kazuki Hayashi
Department of Architecture and Architectural Engineering, Graduate School of Engineering, Kyoto University, Kyoto-Daigaku Katsura, Nishikyo, Kyoto, 615-8540, Japan.
Tel.: +81-75-383-2925
Fax: +81-75-383-2966
E-mail: hayashi.kazuki.55a@st.kyoto-u.ac.jp

Makoto Ohsaki
Department of Architecture and Architectural Engineering, Graduate School of Engineering, Kyoto University, Kyoto-Daigaku Katsura, Nishikyo, Kyoto, 615-8540, Japan.

1 Introduction

Attempt to find mechanically efficient structures is called structural optimization. One of the main issues in the field of optimization of building frames is size optimization, where cross-sections of members are optimized as design variables [25]. If the sizes can vary continuously, the optimization problem can be formulated as a linear or nonlinear programming problem, and various approaches based on mathematical programming such as simplex method [19] and sequential quadratic programming [9] are available. However, for practical applications to steel frame structures, the sizes are preferred to be discrete variables, because cross-sections are usually chosen from a prescribed list of standard sections.

If some or all of the design variables are supposed to have discrete values, the design problem becomes combinatorial optimization problem where integer programming, such as cutting plane method and branch-and-bound method are available. However, the computational cost grows exponentially as the problem size increases, and it becomes difficult to obtain the global optimal solution.

Alternatively, many heuristic approaches have been proposed to obtain approximate optimal solutions, although the solutions do not satisfy any theoretically defined optimality criteria. Balling [2] applied simulated annealing (SA), one of metaheuristic methods, to select optimum sections of columns and beams from discrete standard sections. The SA-based strategy requires less computation time to obtain promising results compared with the branch-and-bound method; however, quality of the solution is strongly dependent on the initial solution. Besides SA, particle swarm optimization (PSO) is also a quite popular heuristic approach, which solves a problem by improving a population of candidate solutions. PSO has been used to optimize either or all of size, topology and shape of trusses [8, 20]. Although

PSO ensures fast convergence than SA, escaping from a local optimal solution is difficult with PSO [13].

Metaheuristics also require appropriate tuning of the parameters such as initial temperature or cooling rate in SA. Although there are many methods for determining the parameters for some benchmark problems such as maximum clique problem [3] and graph matching problem [29], each optimization problem has its own suitable parameters that are unknown beforehand. Hence, the validity of the parameter values can be confirmed only by trial-and-errors, which needs substantial effort and computation time.

In the past decades, there are increasing number of applications of machine learning in the field of civil engineering. The previous researches shows remarkable performance in predicting structural properties or behavior which requires complex structural analysis, such as shear capacity of reinforced concrete beams [27] and ground vibration induced by blasting [15]. Nevertheless, there are relatively few number of researches which focus on how to utilize the prediction results in the design process.

Reinforcement learning (RL) is a branch of machine learning, which trains an agent to take actions that are expected to maximize reward in an environment. The environment is usually formulated as the Markov decision process (MDP) in which next state and reward depend on the state and the action taken at the current step only. RL has shown a remarkable success in game playing [23]; the games are easily fit into MDP framework so that the state is represented by the game screen and the action is represented by controlling buttons and a joystick. In the context of structural engineering, MDPs have been extensively used for the modeling of infrastructure management [21, 33] in order to estimate maintenance cost. Still, few studies focus on structural design process itself.

Structural design and optimization are essentially homogeneous because that they are intended to maximize their structural performance within structural cost. If the process of solving a structural optimization problem can be converted to a series of decision making process by a qualified engineer, the trained model by RL can be utilized to boost the efficiency of structural design process, which is expected as a strong tool to support structural engineers.

Another interplay of machine learning and structural optimization is classification of solutions based on objective function value for reduction of computation time in the searching process. Tamura et al. [32] applied binary decision tree and support vector machine to classify approximate optimal and non-optimal brace placements of steel frames. Approximation of response function value is a standard application of machine learning to structural optimization. Papadrakakis et al. [26] substituted structural analysis to neural network (NN) to obtain objective function value in the iterative process of a size optimization and a shape optimization using an evolutionary strategy. Function approximation with NN is a vital technique in the field of deep learning, which has been extensively applied in the field of image recognition [18]. Combination of deep learning and RL results in deep reinforcement learning, which ignited a boom of artificial intelligence.

In this paper, the design process of steel frames under static loads is formulated as MDP in order to simulate the sequential process of design change by structural engineers and apply RL. The RL agent is trained using NN to minimize the total structural volume while satisfying constraints on allowable stress and strong column-weak beam design. To verify the improvement of searching process by the proposed method, we further combine the trained RL agent with SA and PSO. The random process of neighborhood search in SA and PSO is supported by RL agent to modify variables in a better direction. The combination of trained RL and metaheuristics is demonstrated through two different frame models. Note that the second example uses the NN which is trained for the first example model without re-training.

It is shown that combination of RL and SA and that of RL and PSO requires fewer iterations to reach high-quality solutions than using SA or PSO only. In addition, the proposed method is robust with respect to variation of hyperparameters of these metaheuristics. Therefore, it does not require a precise tuning for determining the parameters, which contribute to alleviate the effort for the parameter setting before optimization.

2 Boosting simulated annealing with deep reinforcement learning

2.1 Reinforcement learning

RL is one of the machine learning methods [31], which is applied to a sequential decision making process. Let s and a represent state and action of the learning problem. The goal of RL is to acquire a policy $\pi(s) = a$ that maximizes the expected future rewards, and such policy is called an optimal policy π^*. As a prerequisite, the task must be modeled as a Markov decision process, where the next state and expected reward are solely dependent on the current state and action [4]. The model of RL is defined by four elements (S, A, P, R); S is the state space; A is the action space; $P(s, a, s')$ is the probabilistic model of the transition from state s to s' by taking action a; $R(s, a, s')$ is the space of reward function r which returns the reward when taking action a at a state s resulting in the next state s'. In addition, some states in S are classified as terminal state.

Q-Learning [34] is a well established method of RL. In Q-Learning, the agent updates its policy based on Q-function $Q(s, a)$, which is the expected future discounted reward for taking action a in the current state s and then following the

specified policy afterwards. When the reward r and the next state s' are observed, estimated value of Q-function, which is called Q-value for brevity, is updated by the following equation:

$$Q(s,a) \leftarrow Q(s,a) + \alpha \left(r(s') + \gamma \max_a Q(s',a) - Q(s,a) \right) \quad (1)$$

where $\alpha \in (0,1]$ is a learning rate to control the convergence of learning process, and $\gamma \in [0,1]$ is a discount factor to control the importance of long-term rewards.

2.2 Neural network to approximate Q-function

When the state space or the action space is too large and it is unrealistic to estimate and store the exact Q-values for all the state-action pairs, it is possible to use an approximator to estimate the Q-values. Deep Q-network (DQN) [23] is a method combining conventional Q-Learning and deep learning: a method to estimate the relation between inputs and outputs using a deep NN architecture. In DQN, the agent learns by tuning NN parameters to find the ideal values θ^* so that it can output proper Q-function approximation $Q(s,a;\theta)$ from the input states as

$$Q(s,a;\theta^*) \approx Q^*(s,a) \quad (2)$$

where $Q^*(s,a)$ is an Q-value, which is the highest expected discounted rewards obtained by an optimal sequence of actions. In Eq. (1), it is intended that the difference is minimized between current estimation of Q-value $Q(s,a)$ and sum of the observed reward and the estimated Q-value thereafter $r(s') + \gamma \max_a Q(s',a)$. Similarly, the NN parameters θ are optimized to minimize the mean squared error L defined as

$$L(\theta) = \left(r(s') + \gamma \max_a Q(s',a;\theta) - Q(s,a;\theta) \right)^2 \quad (3)$$

The optimal target values $r(s') + \gamma \max_a Q(s',a;\theta)$ are substituted by $r(s') + \gamma \max_{a^-} Q(s',a^-;\theta^-)$, using θ^- from previous θ to stabilize the algorithm, where a^- is distinguished from a because they are derived from different Q-values [23]. Therefore, the NN parameters are tuned so as to minimize the following equation:

$$\text{minimize } \tilde{L}(\theta) = \left(r(s') + \gamma \max_{a^-} Q(s',a^-;\theta^-) - Q(s,a;\theta) \right)^2 \quad (4)$$

The gradient of $\tilde{L}$ with respect to θ is obtained as

$$\nabla \tilde{L}(\theta) = 2 \left(r(s') + \gamma \max_{a^-} Q(s',a^-;\theta^-) - Q(s,a;\theta) \right) \nabla Q(s,a;\theta) \quad (5)$$

Since the gradient of the objective function is analytically obtained, the tuning problem (4) is solved with a gradient-based method. See more details about DQN in Mnih et al. [23].

3 Metaheuristics boosted by reinforcement learning

Two representative metaheuristics are first introduced and their algorithms are strengthen by RL agent in this section. SA is one of metaheuristics to obtain approximate global optimal solutions of a given optimization problem, whose name is originated from annealing process to control crystalization of materials by heating and cooling [17]. The algorithm accepts worse solutions in the beginning of the optimization process with high probability, and gradually reduces the probability as optimization proceeds. PSO is also a popular heuristic approach, which utilizes a population of candidate solutions [14]. Each particle's position is repeatedly updated during the optimization based on its position, velocity and the best known positions in the search space.

However, metaheuristics are generally stochastic methods that cannot adapt to the intrinsic structure of the deterministic optimization problem, and require an enormous computational effort to obtain high-quality solutions for large-scale optimization problems. For an optimization problem with complex constraints, it is even difficult to reach the good solutions through completely random search of the neighborhood solutions. Therefore, in this paper, SA and PSO are modified so that RL agent can be applied. In other words, RL agent is utilized in selection process of neighborhood solutions to develop a more intelligent algorithm.

The customized SA algorithm involving RL agent is described in Algorithm 1. Note that the number of neighborhood solutions is set to be just one in this study to observe convergence property of SA. As shown in the line 4, RL agent is utilized to generate neighborhood solutions. To balance the exploitation of RL agent and exploration, a ϵ-greedy policy with $\epsilon = 0.2$ is utilized here. Similarly, the customized PSO algorithm is described in Algorithm 2. Since the position and velocity of each particle vary continuously, each particle's position is modified continuously in line 17; a greedy policy is taken and the observed particle's position and the previous position is randomly interpolated.

4 Optimization of plane steel frame

In this section, the proposed algorithm is applied to cross-section optimization of steel frame structures to minimize total structural volume under constraints on maximum stresses and strength ratios between beams and columns. Figure 1 shows the prescribed configuration of the frame model; the numbers of stories and spans are eight and three, respectively.

Algorithm 1: SA with RL agent

input: x: initial solution, T: initial temperature, c: cooling rate, n_c: cooling interval, n_i: maximum number of iteration
output: x_b: best solution

$x_\mathrm{b} \longleftarrow x$
$j \longleftarrow 0$
for $i \leftarrow 1$ **to** n_i **do**
  $y \longleftarrow$ neighborhood solution of x generated by RL agent
  if $f(y) < f(x_\mathrm{b})$ **then**
    $x_\mathrm{b} \longleftarrow y$
  else if $f(y) < f(x)$ **then**
    $x \longleftarrow y$
  else
    if $\exp\left(\frac{f(x)-f(y)}{T}\right) \geq \mathrm{random}\,[0,1]$ **then** $x \longleftarrow y$
  $j \longleftarrow j + 1$
  if $j = n_\mathrm{c}$ **then**
    $T \longleftarrow cT$
    $j \longleftarrow 0$
return x_b

Algorithm 2: PSO with RL agent

input: x_j: initial solution of particle j, v_j: initial velocity of particle j, n_p: number of particles, n_i: maximum number of iteration, ξ: momentum factor, c_1: social coefficient, c_2: cognitive coefficient
output: x_b: best solution

$i \longleftarrow 0$
$x_\mathrm{b} \longleftarrow x_1$
for $j \leftarrow 1$ **to** n_p **do**
  $x_{\mathrm{pb},j} \longleftarrow x_j$
  if $f(x_j) < f(x_\mathrm{b})$ **then**
    $x_\mathrm{b} \longleftarrow x_j$
  $i \longleftarrow i + 1$
while $i \leq n_\mathrm{i}$ **do**
  for $j \leftarrow 1$ **to** n_p **do**
    $x_j \longleftarrow x_j + v_j$
    if $f(x_j) < f(x_\mathrm{b})$ **then**
      $x_\mathrm{b} \longleftarrow x_j$
    if $f(x_j) < f(x_{\mathrm{pb},j})$ **then**
      $x_{\mathrm{pb},j} \longleftarrow x_j$
    $i \longleftarrow i + 1$
  for $j \leftarrow 1$ **to** n_p **do**
    $x_j \longleftarrow$ modified x_j by RL agent
    $v_j \longleftarrow \xi v_j + c_1(x_\mathrm{b} - x_j) + c_2(x_{\mathrm{pb},j} - x_j)$
return x_b

The frame has uniform spans and story heights, and the bottom four nodes are rigidly supported. Note that the column span in the perpendicular direction to the plane is also 10.0 m. The column section is a square hollow structural section and the beam section is a wide-flange section. The section dimensions of beams and columns are selected from the list in Table1. Note that sectional dimensions of columns are expressed in the form of (width)×(height)×(thickness), and beams are (web height)×(flange width)×(web thickness)×(flange thickness). In this example, columns and beams in the same story are assumed to have the same cross-sections, respectively, to simplify the problem; still, about 10^{16} combinations of variables are possible.

Table 1 Dimension list of column and beam sections [mm].

index	column	beam
200	$200\times200\times12$	$194\times150\times6\times9$
250	$250\times250\times12$	$244\times175\times7\times11$
300	$300\times300\times16$	$294\times200\times8\times12$
350	$350\times350\times19$	$340\times250\times9\times14$
400	$400\times400\times22$	$400\times200\times9\times19$
450	$450\times450\times22$	$450\times200\times9\times22$
500	$500\times500\times25$	$500\times250\times9\times22$
550	$550\times550\times25$	$550\times250\times9\times22$
600	$600\times600\times25$	$600\times250\times12\times25$
650	$650\times650\times28$	$650\times250\times12\times25$

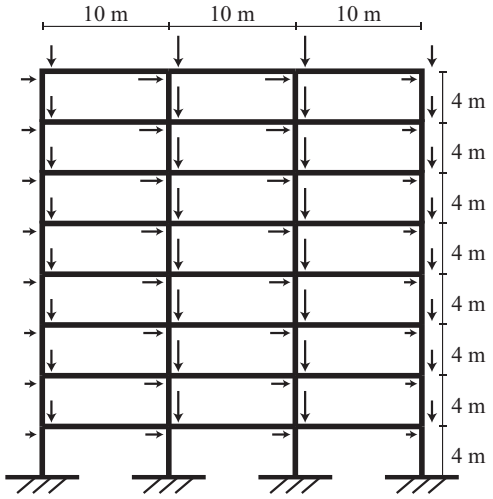

Fig. 1 First example of steel frame.

4.1 Allowable stress design

Design strength F is the reference value for determining allowable stresses of the material against tension, compression, bending and shear forces. For steel material, the yield stress is generally equal to its design strength under short-term loads. The value of F for columns and beams are both equal to 235 $\mathrm{N/mm^2}$ in this example.

Based on the design strength F, allowable stresses of the steel against tension, compression and bending forces, denoted as f_t, f_c and f_b, respectively, are determined as shown in Table 2, where λ is the effective slenderness ratio and Λ is the critical slenderness ratio. Note that Table 2 is a simplified form of Japanese building standards. Usually

allowable stress for long-term loads is used for self weight and live load, and allowable stress for short-term loads is for earthquake load and wind load. The frame structure is assumed to be subjected to the latter loading case, and we only use short-term allowable stresses in this paper.

Table 2 Allowable stresses of steel material

		long-term allowable stress *
tension f_t		$F/1.5$
compression f_c	$\lambda \le \Lambda$	$\frac{1-0.4(\lambda/\Lambda)^2}{3/2+(2/3)(\lambda/\Lambda)^2}F$
	$\lambda > \Lambda$	$\frac{18}{65(\lambda/\Lambda)^2}F$
bending f_b		$F/1.5$

∗ Short-term allowable stresses are 1.5 times the long-term ones.

To calculate stresses of members, the static nodal loads are computed in the following way. Self weight of ith story W_i^S is a product of structural volume of ith story V_i $[\mathrm{m}^3]$ and unit weight of steel $\rho = 77.0$ $[\mathrm{kN/m}^3]$. Distributed load of $w_\mathrm{L} = 10.0$ $[\mathrm{kN/m}^2]$ is assigned to compute W_i^L, the live load of story i. W_i^S and W_i^L are re-distributed to ith story's nodes so that the loads at the exterior nodes are half of those at the interior nodes.

Let W_T and α_i denote the total weight of the structure and the ratio of the weight above ith story to W_T, respectively. The shear force coefficient which defines intensity and vertical variation of the seismic load is denoted by C_i. In the current Japanese seismic design code [22], the design shear force of ith story Q_i is expressed as

$$Q_i = C_i W_\mathrm{T} \alpha_i \tag{6}$$

Note that distributed load of $w_\mathrm{E} = 5.0$ $[\mathrm{kN/m}^2]$ is assigned instead of w_L on each story for computing Q_i in Eq. (6). C_i is expressed as a product of base shear coefficient C_B and shear coefficient variation factor which is simply given as $1/\sqrt{\alpha_i}$ [12]. Hence, Eq. (6) is rewritten as

$$Q_i = C_\mathrm{B} W_\mathrm{T} \sqrt{\alpha_i} \tag{7}$$

Seismic load in story i is computed from $Q_i - Q_{i-1}$ as a difference between shear forces of ith and $(i-1)$th story with $Q_0 = 0.0$. Horizontal seismic load is also distributed to the nodes in each story so that the loads at the exterior nodes are half of those at the interior nodes. The nodal loads explained above are illustrated in Fig. 1.

4.2 Evaluation of member stress

The frame is assumed to be linear elastic with Young's modulus E and shear modulus G. The standard 6-degree-of-freedom 2D beam element is used. Let $\bar{\mathbf{p}}_i$ denote member force vector. The local stiffness equation about member i is described with local stiffness matrix $\bar{\mathbf{K}}_i$, local displacement vector $\bar{\mathbf{u}}_i$ and $\bar{\mathbf{p}}_i$ as

$$\bar{\mathbf{K}}_i \bar{\mathbf{u}}_i = \bar{\mathbf{p}}_i \tag{8}$$

Using transformation matrix, elements of $\bar{\mathbf{K}}_i$ are converted to those in the global coordinate system to be assembled into the global stiffness matrix $\mathbf{K}$. Let $\mathbf{u}$ and $\mathbf{p}$ denote displacement and load vectors about all degrees of freedom in a global coordinate system. Then stiffness equation is constructed as

$$\mathbf{K}\mathbf{u} = \mathbf{p} \tag{9}$$

Since $\mathbf{p}$ is already specified by the process introduced in the previous subsections, Eq. (9) is a set of simultaneous linear equations with respect to $\mathbf{u}$. After solving Eq. (9), the axial force N_i and the bending moments $M_{i,1}$ and $M_{i,2}$ at the two ends of member i are computed from $\bar{\mathbf{p}}_i$. Using the cross-sectional area A_i and the section modulus Z_i of ith member, the axial stress σ_i^a and the bending stresses $\sigma_{i,1}^\mathrm{b}$ and $\sigma_{i,2}^\mathrm{b}$ at the two ends are computed as

$$\sigma_i^\mathrm{a} = \frac{N_i}{A_i} \tag{10a}$$

$$\sigma_{i,1}^\mathrm{b} = \frac{M_{i,1}}{Z_i} \tag{10b}$$

$$\sigma_{i,2}^\mathrm{b} = \frac{M_{i,2}}{Z_i} \tag{10c}$$

Note that σ_i^a is negative for compression and positive for tension. Then, these stresses need to satisfy the following equation using the stress ratio $\bar{\sigma}_i$ and prescribed parameter τ:

$$\bar{\sigma}_i \le \tau \tag{11a}$$

$$\bar{\sigma}_i = \begin{cases} -\dfrac{\sigma_i^\mathrm{a}}{f_c} + \dfrac{\max\{|\sigma_{i,1}^\mathrm{b}|, |\sigma_{i,2}^\mathrm{b}|\}}{f_b} & (\text{if } \sigma_i^\mathrm{a} < 0) \\ \dfrac{\sigma_i^\mathrm{a}}{f_t} + \dfrac{\max\{|\sigma_{i,1}^\mathrm{b}|, |\sigma_{i,2}^\mathrm{b}|\}}{f_b} & (\text{if } \sigma_i^\mathrm{a} \ge 0) \end{cases} \tag{11b}$$

A value below 1.0 is usually specified for τ. In this paper, $\tau = 0.8$ to ensure the safety.

4.3 "Strong column-weak beam" design

To avoid local story collapse without enough energy dissipation and ensure ductility of the frame, it is important to design the frame in accordance with "strong column-weak beam" principle. Figure 2 shows two typical collapse modes. If a frame is designed so that plastic hinges are formed at beam ends rather than in columns, its collapse mechanism is expected to form the whole collapse mode where hinges

at the beams are dominant as shown in the left figure of Fig. 2. On the other hand, if plastic moment capacity of the columns is smaller than that of the beams, there is a possibility of collapse at a specific story as shown in right figure of Fig. 2.

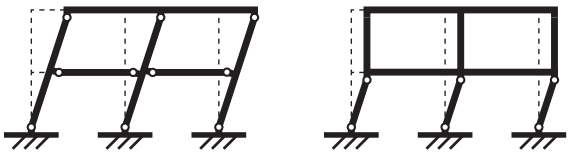

Fig. 2 Desirable mechanism (left) and mechanism to be avoided (right).

The column-to-beam strength ratio, or column overstrength factor (COF), is a strong column-weak beam criterion about a joint and calculated as

$$\beta = \frac{\sum M_{\mathrm{pc}}}{\sum M_{\mathrm{pb}}} \tag{12}$$

where $\sum M_{\mathrm{pc}}$ and $\sum M_{\mathrm{pb}}$ are the sum of full plastic moment of columns and beams, respectively, connecting to the joint.

Nakashima and Sawaizumi [24] investigated the magnitude of COF required for ensuring occurrence of plastic hinges only at the ends of beams and the first-story column bases. It was found that the required COFs increase steadily with the increase of the ground motion velocity, and reaches about 1.5 for the ground motion amplitude of 0.5 m/s, which is considered as the strong earthquake in Japanese seismic design code. Therefore, frames are designed to satisfy $\beta \geq 1.5$ in this study.

According to Japanese building standards law, M_{pb} in Eq. (12) is given as

$$M_{\mathrm{pb}} = F_{\mathrm{yb}} Z_{\mathrm{pb}} \tag{13}$$

where F_{yb} is the design strength and Z_{pb} is the plastic section modulus of beam element. Although M_{pc} for a column is given in the similar way using the design strength F_{yc} and the plastic section modulus Z_{pc} of column element, the value is discounted using the axial force ratio η_{c}, the ratio of axial stress $|\sigma_i^{\mathrm{a}}|$ to the yield stress, as

$$M_{\mathrm{pc}} = \tau F_{\mathrm{yc}} Z_{\mathrm{pc}} \tag{14a}$$

$$\tau = \begin{cases} 1 - \frac{4\eta_{\mathrm{c}}^2}{3} & (\eta_{\mathrm{c}} \leq 0.5) \\ \frac{4(1-\eta_{\mathrm{c}})}{3} & (\eta_{\mathrm{c}} > 0.5) \end{cases} \tag{14b}$$

Note that hinges are allowed to exist at upper ends of the top story columns, because it is too difficult for only one column to outperform sum of full plastic moments of two beams. Similarly, hinges are allowed to exist at the bottom ends of the first-story columns, since there is no corresponding beam at the nodes.

4.4 Phase 1: learning

Although states primarily need to contain all the information about the system, structural information about one certain story and its upper and lower stories is included in the state to reduce the size of NN and utilize the learning results to a frame with different numbers of stories and spans. The beam connected to the top node of a column in the ith story is called "beam in the ith story". The state contains the following five types of features for the target ith story:

- $s_{\mathrm{a}} \in \mathbb{R}^{2\times 3}$:
  column and beam sections at the $(i-1)$th, ith and $(i+1)$th stories which are defined as the indices in Table1 normalized by their largest value, i.e., 650 in this example
- $s_{\mathrm{b}} \in \mathbb{R}^{2\times 2}$:
  binary features which are activated if the cross-section of columns or beams at the ith story is at its upper- or lower-bound
- $s_{\mathrm{c}} \in \mathbb{R}^{2\times 3}$:
  maximum stress ratios, respectively, in the columns and beams for the $(i-1)$th, ith and $(i+1)$th stories
- $s_{\mathrm{d}} \in \mathbb{R}^{4}$:
  minimum COFs within the nodes in the $(i-2)$th, $(i-1)$th, ith and $(i+1)$th stories, respectively
- $s_{\mathrm{e}} \in \mathbb{R}^{2}$:
  binary features which are activated if the ith story is the first or the top story, respectively

For features s_{b} and s_{c}, zero is assigned if there is no corresponding story to be focused. An example of extracting the feature values is illustrated in Fig. 3 for $i = 1$. The left and right numbers enclosed by rectangles show the indices of beams and columns in the stories, respectively. The numbers on the frame nodes with gray background show COFs at the nodes. The numbers enclosed by rounded rectangles with white background are the stress ratios of the members.

In addition to the state, five actions are defined as follows:

- increase size of the columns in the current target story by one level
- increase size of the beams in the current target story by one level
- reduce size of the columns in the current target story by one level
- reduce size of the beams in the current target story by one level
- keep the current sizes of beams and columns

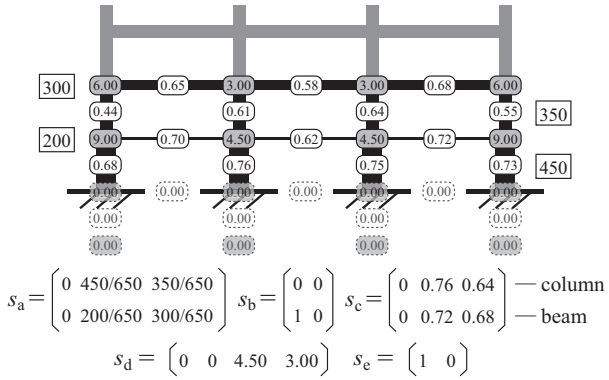

Fig. 3 Feature extraction example from the first story as the target story.

In general, the number of layers and units in each layer in NN cannot be determined analytically to address a specific problem, and they are usually determined by trial and error [11]. Therefore, systematic experiment is employed to discover the appropriate structure of NN and determined as follows; the NN has three fully-connected middle layers with 100 units. Therefore, a relatively simple 22-100-100-100-5 NN is defined as illustrated in Fig. 4. Sigmoid function is used as an activation function of all the units except for the output layer [28].

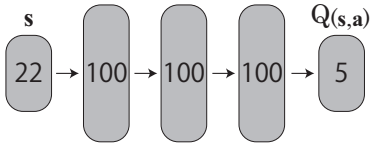

Fig. 4 Number of cells in each layer of NN for frame optimization.

After taking the action, linear structural analysis is conducted and reward r and the next state s' are observed. If the columns and beams in the current target story and its upper and lower stories satisfy allowable stress constraint and the nodes in the target story satisfy COF constraint, the value of $200 + 200$ over sum of column and beam indices at current target story is given to the agent as a reward; otherwise, zero reward is given. For instance, if the constraints are satisfied and column and beam indices in the target story are 400 and 600, respectively, $(200 + 200)/(400 + 600) = 0.4$ is provided as the reward. This way, reward is clipped within the range of $[0, 1]$. After observing the reward, the target story moves up by one level and the state values are updated for the current target story, which becomes the next state s' in Eqs. (1)–(5). Note that the target story is reset to the first story when it tries to move above the top story.

Initial indices of sections of the frame are set as 400 in Table 1, and the initial state to be evaluated is assigned at the first story of the frame. Terminal state is not explicitly provided; instead, each episode is terminated at the 100th step.

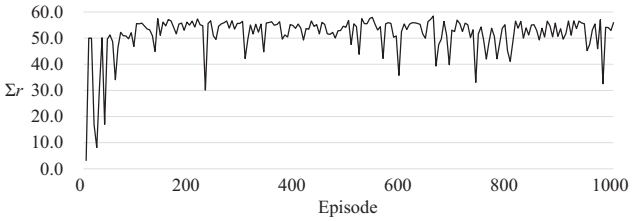

Fig. 5 History of total reward during training process of NN.

The memory size for storing a set of experience (s, a, r, s') is 1000 to secure enough memory space, and randomly permutated 100 mini-batch data is constructed from this memory for updating the NN parameters. Synchronizing frequency of NN parameters of θ and θ^- is 5 steps [23]. Adam [16] is used to solve the problem (4). Discount factor γ is set to be 0.99, which is a widely accepted value for tasks where long-term rewards are important [1, 10]. "Dropout" [30] is not used in this study.

Every after ten episodes of learning, performance of the NN is tested by simulating one episode with greedy policy (i.e., ϵ-greedy policy with $\epsilon = 0.0$) and observing the total rewards. History of the test results is illustrated in Fig. 5. It is observed that obtained total rewards increases as learning proceeds, which implies that the agent acquired a policy of section design to reduce the total structural volume while satisfying requirements for allowable stresses and COFs. To demonstrate the process of acquiring intelligence, three test simulation results are illustrated; Figs. 6, 7 and 8 are the best test simulations after 10-, 100- and 1000-episode training, respectively. The resulting cross-sections after 10 episodes is the same as the initial solution, because the NN only choose the conservative action of keeping the size of members to avoid violating allowable stress and COF constraints. On the other hand, the best test simulation result after the training of 100 and 1000 episodes successfully reduced the cross-sections while satisfying the constraints. It should be noticed that the latter result reached smaller cross-sections. The NN parameters which showed the best test performance are retrieved in order to be utilized for solving the optimization problem described in the next section.

It should be noted that the agent focuses on four stories of the frame structure and fundamentally cannot handle the whole structural design by itself, although the simulation reached a section design satisfying the constraints. Moreover, the policy in the test simulation is deterministic and is not capable of finding better neighborhood solutions. Thus, it is necessary to use a stochastic optimization method that can consider performance of the whole structure.

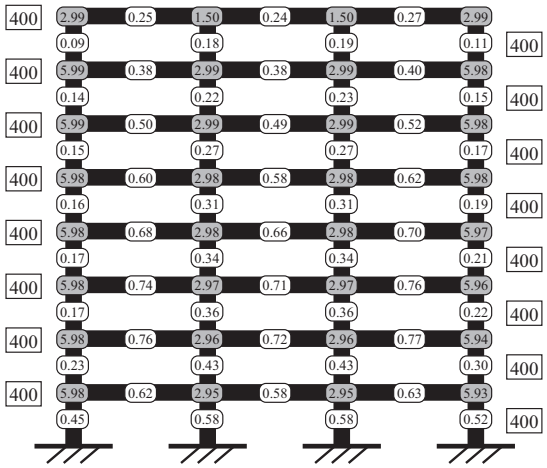

Fig. 6 Performance of frame after 10 episodes trained (V = 6.579 [m^3]).

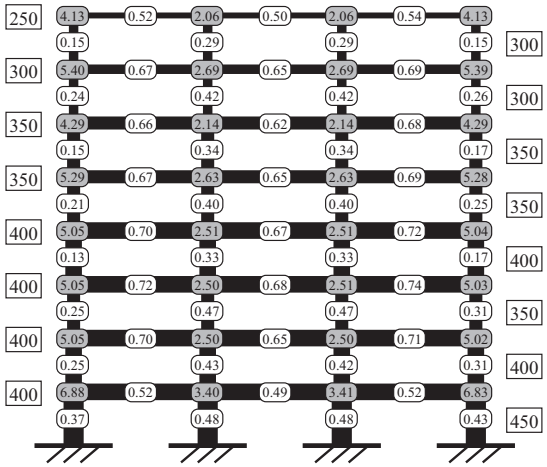

Fig. 7 Performance of frame after 100 episodes trained (V = 5.546 [m^3]).

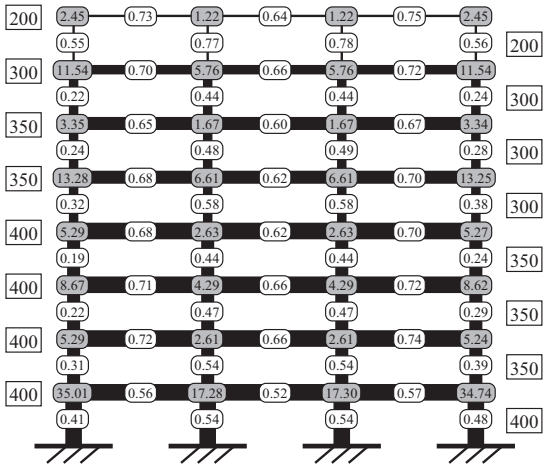

Fig. 8 Performance of frame after 1000 episodes trained (V = 4.852 [m^3]).

4.5 Phase 2: optimization

The objective of optimization is the same as the learning task in Phase 1: minimizing total structural volume while satisfying allowable stress and COF constraints. Therefore, the optimization problem is formulated as

$$\text{minimize} \quad V(\mathbf{t}) \tag{15a}$$

$$\text{subject to} \begin{cases} t_i \in \{200, 250, \cdots, 650\} & (i = 1, \cdots, n_{\mathrm{g}}) \\ \bar{\sigma}_j \le 0.8 & (j = 1, \cdots, m) \\ \beta_k \ge 1.5 & (k \in \mathbf{\Omega}) \end{cases} \tag{15b}$$

where $V(\mathbf{t})$ is the total structural volume, $\mathbf{t} \in \{t_1, t_2, \cdots, t_{n_{\mathrm{g}}}\}$ is the cross-section indices of n_{g} member groups, $\bar{\sigma}_j$ is the stress ratio of jth member computed by Eq. (11b), β_k is the COF of node k, and $\mathbf{\Omega}$ is the set of indices of all nodes except for the bottom and the top ones. Initial solution is a frame with uniform "400" sections in Table 1. To consider the constraints for SA and PSO implementation, a neighborhood solution which violates any of constraints in Eq. (15b) is rejected regardless of its objective function value. In the same manner as the learning process of Phase 1, the target story is introduced to change cross-sections of the members in the specific story. The target story moves up by one level every step and is reset to the first story after finishing the top story.

The maximum number of structural analyses for one optimization is set to be 1000 for both SA and PSO. The optimization is conducted 10 times using different random seeds to choose the best solution out of the 10 local optimal solutions. In the implementation of SA, cooling rate c is 0.9, and cooling interval n_{c} is 20. Initial temperature is set so that the acceptance probability p_0 of worse transition at the initial step is 0.8 on average, by sampling a number of random transitions from an initial solution, based on the algorithm proposed by [5]. In the implementation of PSO, the number of particles n_{p} is 5, momentum factor ξ is 0.7, and social and cognitive coefficients c_1 and c_2 are both set to be 2.0. The initial velocity is randomly generated within 10% of the range of variables.

The maximum, median, minimum, average values, and standard deviation of the objective function values of 10 different solutions are listed in Table 3. The combination of metaheuristics and RL outperforms metaheuristics in maximum, median, minimum and average values of objective function. It should also be noted that the increment of computational cost by introducing RL agent is relative small; the average CPU time for yielding one optimal solution is 2.99[s] for (SA+RL), 2.61[s] for SA, 3.08[s] for (PSO+RL), and 2.71[s] for PSO. The reason of the outperformance can be explained by tracing the iteration history of the objective function in Figs. 9 and 10, where the histories of the best and the worst optimal solutions out of the 10 solutions obtained

Table 3 Maximum, median, minimum, average values, and standard deviation of total structural volume for 10 solutions of the eight-story model for proposed methods (SA+RL) and (PSO+RL) and conventional methods (SA only and PSO only). $c = 0.9$ and $p_0 = 0.8$ for SA implementation, and $c_1 = 2.0$, $c_2 = 2.0$ and $n_\mathrm{p} = 5$ for PSO implementation.

	SA+RL	SA only	PSO+RL	PSO only
max.	4.935	5.334	4.892	5.297
median	4.701	4.860	4.824	5.111
min.	4.468	4.657	4.605	4.680
average	4.676	4.920	4.794	5.067
std. dev.	0.155	0.207	0.103	0.194

Table 4 Medians of objective functions with a variety of SA parameters.

	SA+RL			SA only		
	$c = 0.8$	$c = 0.9$	$c = 0.95$	$c = 0.8$	$c = 0.9$	$c = 0.95$
$p_0 = 0.2$	4.725	4.672	4.782	4.861	4.794	4.808
$p_0 = 0.8$	4.677	4.701	4.697	5.023	4.860	4.804
$p_0 = 0.99$	4.782	4.762	4.815	5.141	5.695	6.453

by the proposed hybrid methods and metaheuristics only are plotted. The hybridized method steadily and rapidly reduces the objective function in the early stage of optimization process, which implies the intelligence of the agent in generating neighborhood solutions. However, the iteration histories of SA+RL fluctuate after the rapid reduction of optimization process above $V = 4.8$, which is almost equal to $V = 4.852$ as observed in the best test simulation result. Therefore, it is better to set a certain positive value on ϵ to complement the agent's knowledge and prevent premature convergence.

The best optimization result obtained by the proposed method is illustrated in Fig. 11. The first observation to be noticed is that the total structural volume of the solution is better than that of the test result which is obtained just by agent's deterministic policy. This implies that the optimizer using SA was able to search a wider range of solution space while utilizing agent's knowledge. Almost all the column and beam sections in upper stories have smaller values than lower stories; however, some column and beam sections in the lower stories are relatively small, because the columns are rigidly supported on the ground, and the first story has enough horizontal stiffness.

We further investigate the robustness against variation of parameters of metaheuristics. Table 4 shows medians of objective functions with different cooling rates $c = 0.8, 0.9$ and 0.95 and initial temperatures with the acceptance probability $p_0 = 0.5, 0.8$ and 0.99. Similarly, Table 5 shows medians of objective functions with different social and cognitive coefficients $c_1 = c_2 = 1.0, 2.0$ and 3.0 and the number of particles $n_\mathrm{p} = 2, 5$ and 10. It was found that introducing RL agent alleviates the dependency on parameters of SA and PSO, and stabilize the process of optimization.

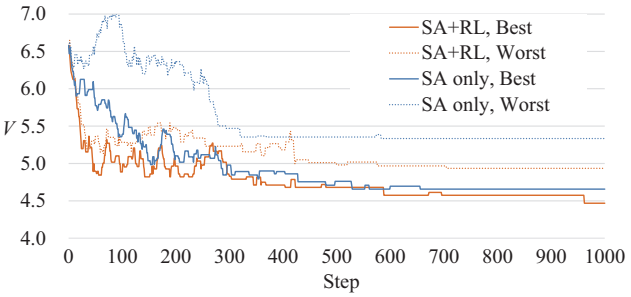

Fig. 9 Iteration history of objective values of the best and the worst solutions (SA).

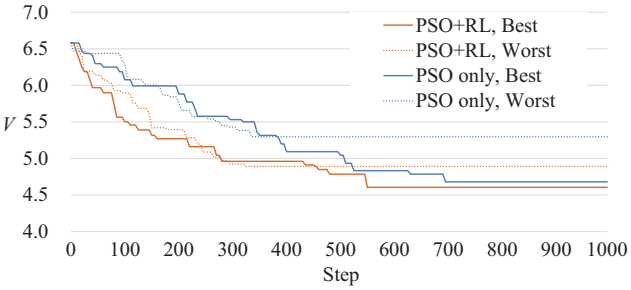

Fig. 10 Iteration history of best known objective values $f(x_\mathrm{b})$ of the best and the worst solutions (PSO).

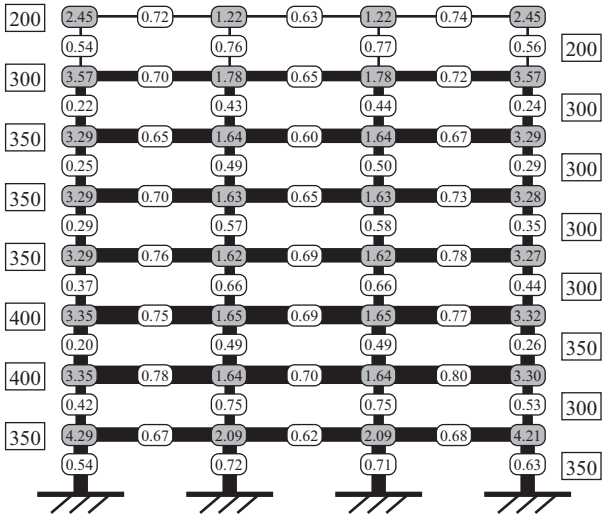

Fig. 11 The best solution of the eight-story frame obtained by the proposed method ($V = 4.468\ [\mathrm{m}^3]$).

Table 5 Medians of objective functions with a variety of PSO parameters.

	PSO+RL			PSO only		
	$c_1 = c_2 =$			$c_1 = c_2 =$		
	1.5	2.0	2.5	1.5	2.0	2.5
$n_\mathrm{p} = 2$	4.926	4.806	4.920	6.579	5.850	4.901
$n_\mathrm{p} = 5$	4.845	4.824	4.846	6.434	5.111	4.986
$n_\mathrm{p} = 10$	4.788	4.798	4.934	6.125	5.006	4.946

4.6 Utilization of learning results to a different frame model

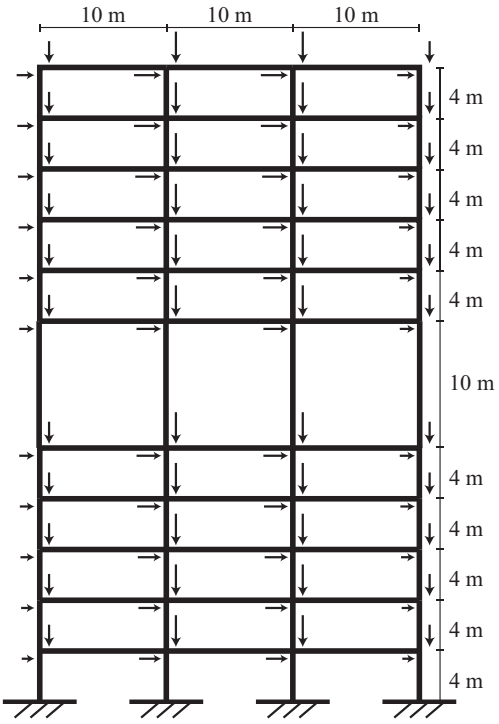

Fig. 12 Second example of steel frame.

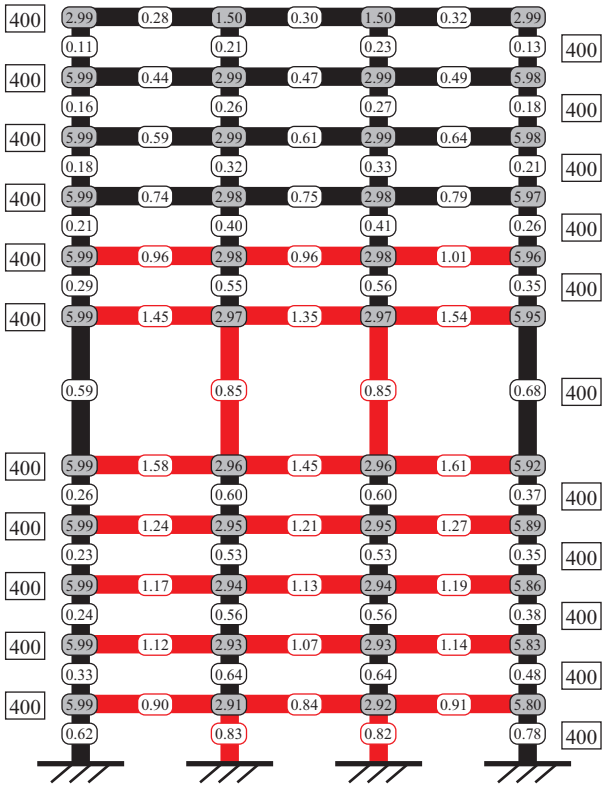

Fig. 13 Initial solution with uniform "400" cross-sections violating the allowable stress constraint (indicated with red).

The same NN architecture used for the eight-story frame model can be applied to different frame models, as long as columns or beams at each story have the same cross-sections, respectively. Performance of the NN is tested here for a more complex eleven-story frame model, as illustrated in Fig. 12. The number of spans and pitches of columns are the same as the previous model; however, there is a story with large height in the middle. The number of possible combinations of cross-sections is 10^{22} in this example.

For optimization of this model, NN trained for the eight-story model is applied to SA without further tuning. Optimization is conducted from two different initial solutions: a frame with uniform "400" cross-sections and that with "600" cross-sections in Table 1. Because the former initial solution does not satisfy the allowable stress constraint as shown in Fig. 13, we further add a rule to the SA algorithm; if the current solution is infeasible, the optimizer accepts a neighborhood solution regardless of the objective function value. That is, the change from an infeasible solution to another infeasible or feasible solution is accepted unconditionally. This time, the number of steps is set to be 2000 and cooling interval is 40 accordingly. Temperature parameters are fixed as $c = 0.9$ and $p_0 = 0.8$.

The maximum, median, minimum, average values, and standard deviation of the objective function values of 10 solutions are summarized in Table 6. The key point to notice is that nine feasible solutions were obtained by the proposed method starting from "400" cross-sections. This implies that the RL agent is certainly capable of finding feasible solutions. The total structural volumes of the solutions obtained by (SA+RL) are almost equal to those of SA only when starting from a feasible solution and the best value by (SA+RL) is slightly better than that by SA only. Optimal solution with $V = 9.775\ [\mathrm{m}^3]$ is illustrated in Fig. 14. Despite the irregular shape of the frame, the structure is successfully stiffened by assigning large cross-section to members in the first and middle stories.

Table 6 Maximum, median, minimum, average values and standard deviation of total structural volume of feasible solutions of the eleven-story model with two different initial solutions.

	SA+RL		SA only	
	$t = 400$	$t = 600$	$t = 400^*$	$t = 600$
max.	11.000	10.975	—	10.850
median	10.482	10.333	—	10.504
min.	10.175	9.775	—	10.084
average	10.505	10.349	—	10.450
std. dev.	0.293	0.303	—	0.206

$*$ No feasible solution was obtained.

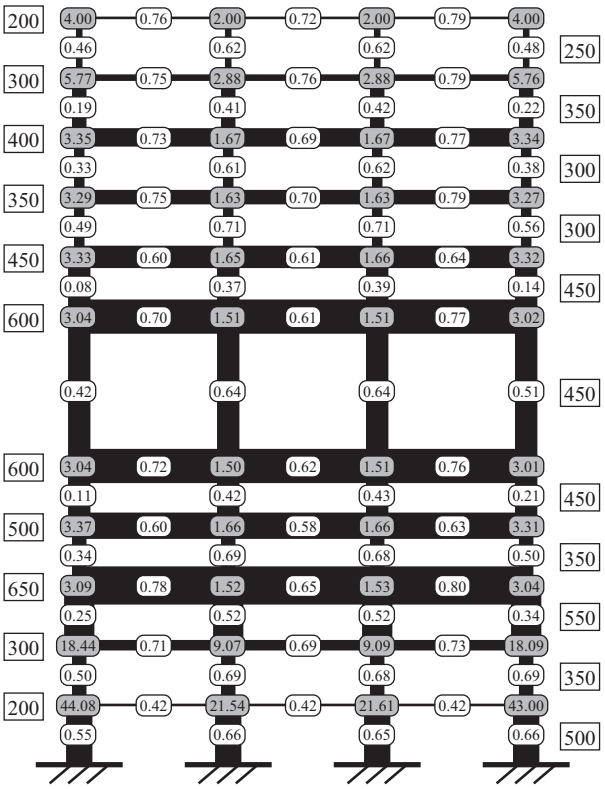

Fig. 14 The best solution of the eleven-story frame obtained by the proposed method ($V = 9.775\ [\mathrm{m}^3]$).

5 Concluding remarks

The sequential process of design change of plane frame cross-sections by structural engineer is formulated in this study. RL is further introduced for optimal design of the cross-sections to minimize total structural volume under allowable stress and strong column-weak beam constraints. The method aims at searching the optimal solution intelligently using RL agent, not just relying on the number of iterations in the optimization process. We trained an agent that focuses on a specific story and its upper and lower ones, instead of the whole structure. In this way, not only the size of the NN is reduced, but also the agent is capable of handling any plane frame models regardless of the shape or the number of spans and stories.

The agent is first trained within the same framework as DQN in order to learn a desirable rule of updating cross-section. Although the hyper-parameters are determined by following previous studies here, the performance of NN might be improved by adjusting them using auto-tuning scheme [6, 7]. By utilizing the trained agent for generating neighborhood solutions during SA and for modifying particle positions in PSO, quality of the optimal solutions is improved rather than modifying the cross-section in accordance with the heuristic algorithm. Since the proposed method especially boosts the improvement in early stage of optimization, it is expected to obtain an feasible and approximate optimal solution with less computation time.

The methodology presented in this study is not only efficient, but also robust in variation of user-specified parameters of metaheuristics. It is confirmed from the numerical examples that this method is able to mitigate the deterioration of the optimization results obtained by SA due to the variation of initial temperature (i.e., initial acceptance probability) and cooling rate, because the "intelligent" RL agent takes the role of efficient selection of neighborhood solutions, which is randomly generated in the standard SA algorithm. As a result, combination of RL and SA is capable of obtaining good approximate optimal solutions regardless of the temperature parameters. Similarly, it is confirmed that combination of RL and PSO is also capable of alleviating the effect of variation of the number of particles, social and cognitive coefficients of PSO, and effective for obtaining good solutions regardless of these parameters. Furthermore, introduction of the RL agent is able to alleviate malfunction of the optimization due to "bad" initial solutions, and can enhance ability of finding a feasible solution. It is also noteworthy that this was achieved just by reusing the NN parameters from the previous training data, without tuning them for the new frame model. This implies the adaptability of the agent to frame models with different numbers of stories and spans.

Acknowledgements The authors are grateful to Prof. Senda for insightful comments on RL. This study was supported by Grant-in-Aid for JSPS Research Fellow No.18J21456 and JSPS KAKENHI No.18K18898.

Conflict of interest

The authors declare that they have no conflict of interest.

References

1. Abbeel P, Ng AY (2004) Apprenticeship learning via inverse reinforcement learning. In: Proceedings of the Twenty-first International Conference on Machine Learning, ACM, New York, NY, USA, ICML '04, pp 1–8
2. Balling RJ (1991) Optimal steel frame design by simulated annealing. Journal of Structural Engineering 117(6):1780–1795
3. Battiti R, Protasi M (2001) Reactive local search for the maximum clique problem. Algorithmica 29(4):610–637, DOI 10.1007/s004530010074
4. Bellman R (1957) A markovian decision process. Indiana Univ Math J 6:679–684

5. Ben-Ameur W (2004) Computing the initial temperature of simulated annealing. Comput Optim Appl 29(3):369–385, DOI 10.1023/B:COAP.0000044187.23143.bd
6. Bergstra J, Bengio Y (2012) Random search for hyper-parameter optimization. J Mach Learn Res 13:281–305
7. Claesen M, Moor BD (2015) Hyperparameter search in machine learning. CoRR abs/1502.02127
8. Fourie P, Groenwold A (2002) The particle swarm optimization algorithm in size and shape optimization. Structural and Multidisciplinary Optimization 23(4):259–267
9. Gill PE, Murray W, Saunders MA, Wright MH (1984) Sequential quadratic programming methods for nonlinear programming. In: Haug EJ (ed) Computer Aided Analysis and Optimization of Mechanical System Dynamics, Springer Berlin Heidelberg, Berlin, Heidelberg, pp 679–700
10. Gron A (2017) Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems, 1st edn. O'Reilly Media, Inc.
11. Heaton J (2008) Introduction to Neural Networks for Java, 2nd Edition, 2nd edn. Heaton Research, Inc.
12. Housner G (1960) In proceedings of second world conference on earthquake engineering. In: The plastic failure of structures during earthquakes, pp 997–1012
13. Idoumghar L, Melkemi M, Schott R, Aouad MI (2011) Hybrid PSO-SA type algorithms for multimodal function optimization and reducing energy consumption in embedded systems. Appl Comp Intell Soft Comput 2011:3:1–3:12
14. Kennedy J, Eberhart R (1995) Particle swarm optimization. In: Proceedings of ICNN'95 - International Conference on Neural Networks, vol 4, pp 1942–1948
15. Khandelwal M (2011) Blast-induced ground vibration prediction using support vector machine. Engineering with Computers 27(3):193–200
16. Kingma DP, Ba J (2014) Adam: A method for stochastic optimization. CoRR 1412.6980
17. Kirkpatrick S, Gelatt CD, Vecchi MP (1983) Optimization by simulated annealing. Science 220(4598):671–680, DOI 10.1126/science.220.4598.671
18. LeCun Y, Boser B, Denker JS, Henderson D, Howard RE, Hubbard W, Jackel LD (1989) Backpropagation applied to handwritten zip code recognition. Neural Computation 1(4):541–551
19. Luenberger D (1984) Linear and Nonlinear Programming. Addison-Wesley
20. Luh GC, Lin CY (2011) Optimal design of truss-structures using particle swarm optimization. Computers and Structures 89(23):2221 – 2232
21. Meidani H, Ghanem R (2015) Random markov decision processes for sustainable infrastructure systems. Structure and Infrastructure Engineering 11(5):655–667
22. MILT (2008) Technical Standards Manual on Building Structures: 2007 Edition"(in Japanese), 2nd edn. National Marketing Cooperative of Official Gazettes
23. Mnih V, Kavukcuoglu K, Silver D, Rusu AA, Veness J, Bellemare MG, Graves A, Riedmiller M, Fidjeland AK, Ostrovski G, Petersen S, Beattie C, Sadik A, Antonoglou I, King H, Kumaran D, Wierstra D, Legg S, Hassabis D (2015) Human-level control through deep reinforcement learning. Nature 518(7540):529–533
24. Nakashima M, Sawaizumi S (2000) Column-to-beam strength ratio required for ensuring beam-collapse mechanism in earthquake responses of steel frames. In: Proceedings of the 12 th World Conference on Earthquake Engineering
25. Ohsaki M (2010) Optimization of finite dimensional structures. CRC Press
26. Papadrakakis M, Lagaros ND, Tsompanakis Y (1998) Structural optimization using evolution strategies and neural networks. Computer Methods in Applied Mechanics and Engineering 156(1):309 – 333, DOI https://doi.org/10.1016/S0045-7825(97)00215-6
27. Prayogo D, Cheng MY, Wu YW, Tran DH (2019) Combining machine learning models via adaptive ensemble weighting for prediction of shear capacity of reinforced-concrete deep beams. Engineering with Computers
28. Rumelhart DE, Hinton GE, Williams RJ (1986) Learning representations by back-propagating errors. Nature 323:533–536
29. Sammoud O, Sorlin S, Solnon C, Ghédira K (2006) A comparative study of ant colony optimization and reactive search for graph matching problems. In: Evolutionary Computation in Combinatorial Optimization, Springer Berlin Heidelberg, Berlin, Heidelberg, pp 234–246
30. Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R (2014) Dropout: A simple way to prevent neural networks from overfitting. Journal of Machine Learning Research 15:1929–1958
31. Sutton RS, Barto AG (1998) Introduction to Reinforcement Learning, 1st edn. MIT Press, Cambridge, MA, USA
32. Tamura T, Ohsaki M, Takagi J (2018) Machine learning for combinatorial optimization of brace placement of steel frames. Japan Architectural Review 1(4):419–430
33. Tao Z, Corotis RB, Ellis JH (1995) Reliability-based structural design with markov decision processes. Journal of Structural Engineering 121(6):971–980
34. Watkins CJCH (1989) Learning from delayed rewards. PhD thesis, King's College, Cambridge, UK