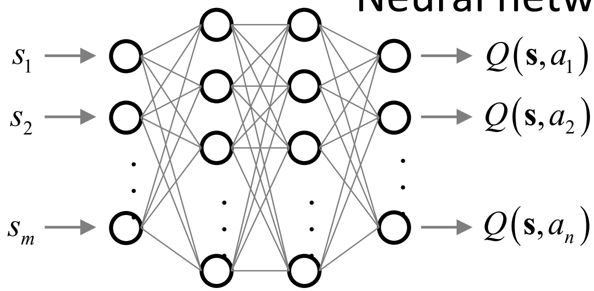
Deep-Q network for truss topology optimization with stress constraints

Kazuki Hayashi (Kyoto University)

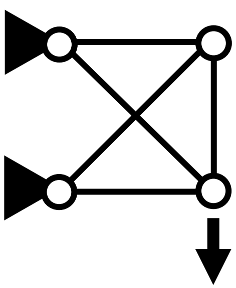
Makoto Ohsaki (Kyoto University)

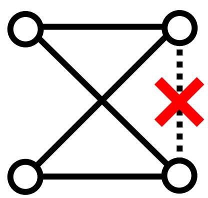
Ingredients of RL

- 

: agent

1. status 2. action 3. reward 4. environment





$+1$ if meet constraint

-1 else

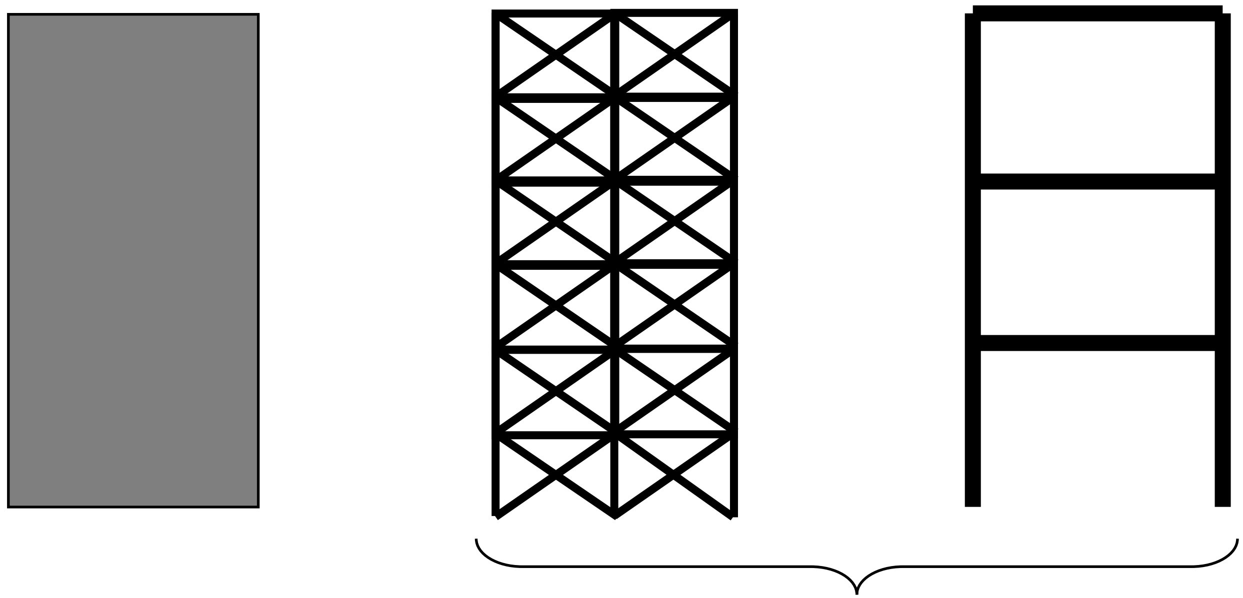
Python

s a r $F(s,a)=\{s',r\}$

- Objective is to maximize total reward

Problem is NOT always pixel-wise

continuous | truss | frame



CNN available | NOT available

→ How to apply RL to discrete structures?

Structure2Vec (Hanjun et al., 2016)

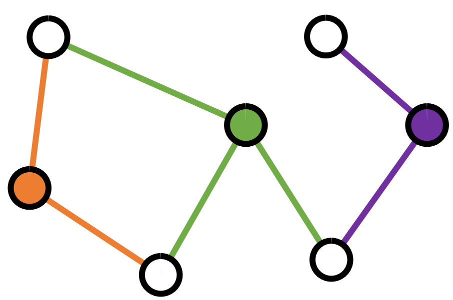
- Reinforcement learning + graph embedding
- Train for MVC and TSP with 50-100 nodes

→ Apply the trained model to 1000-1200 nodes

→ Solution comparable to CPLEX solver's one

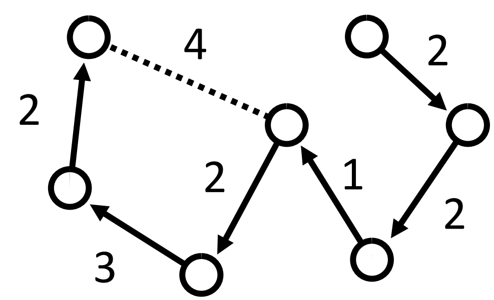
12 seconds

1 hour



Minimize (*no. of nodes selected*)

Minimum vertex cover (MVC)

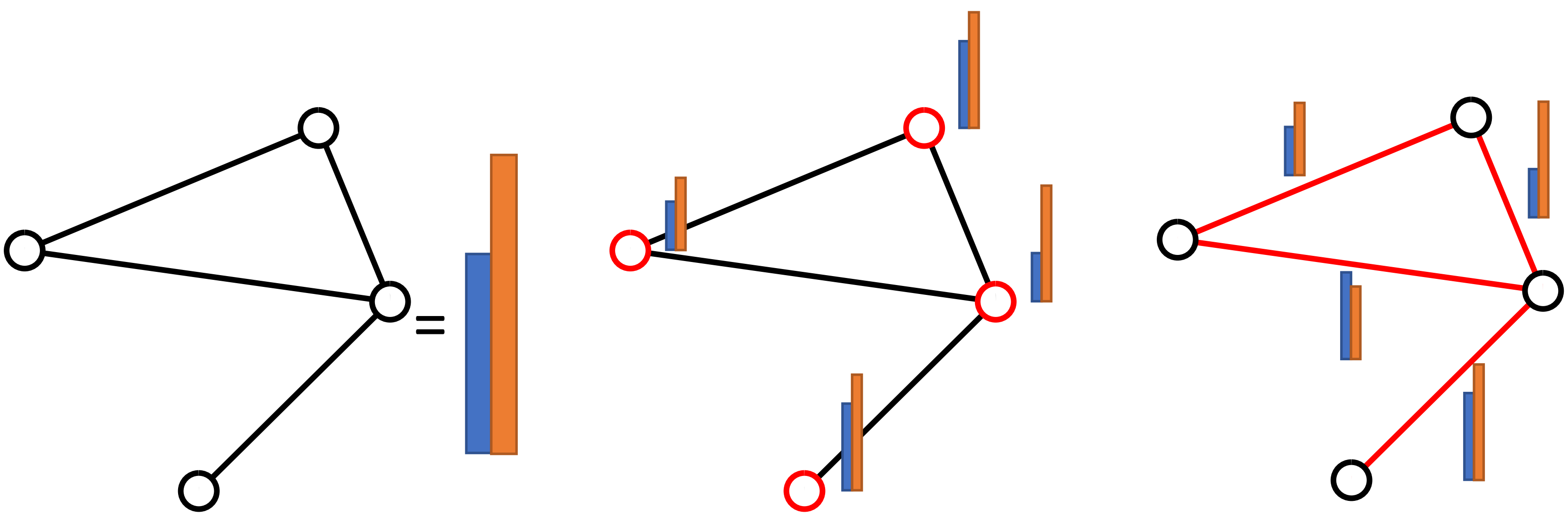


Minimize (*total path*)

Traveling salesman problem(TSP)

Graph embedding

- Convolutional NN : Image → Feature Vector
- Graph embedding : Graph → Feature Vector



Whole graph

Node embedding

ex) Structure2Vec
(Hanjun et al., 2016)

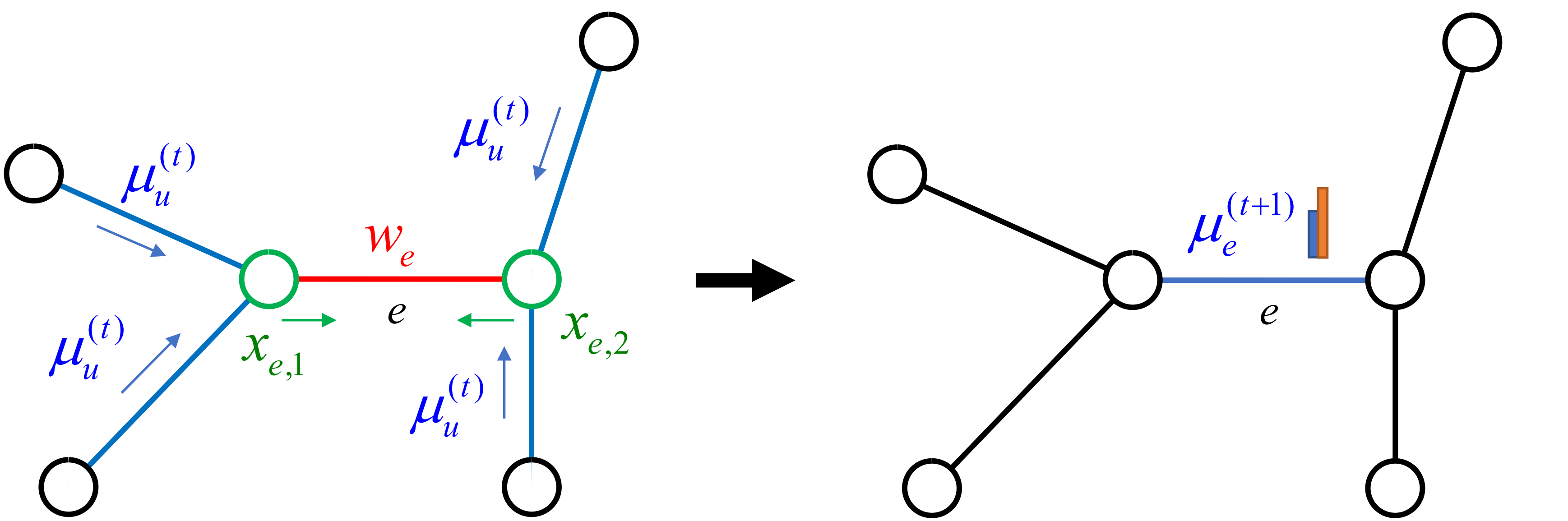
Edge embedding

We formulate a new method
for truss topology optimization

Edge embedding

- μ_e: feature vector of edge e

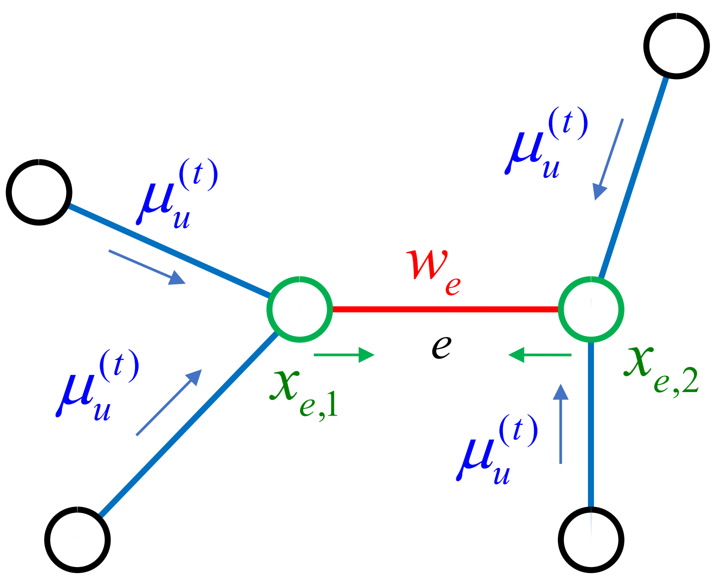
$$\mu_e^{(t+1)} \leftarrow \mathrm{relu}\left(\theta_1 w_e + \sum_{i=1}^{2} \mathrm{relu}\left(\theta_2\left(\left(\sum_{u\in\aleph(v_{e,i})}\mu_u^{(t)}\right)-\mu_e^{(t)}\right)\right)+\theta_3\sum_{i=1}^{2}\mathrm{relu}\left(\theta_4 x_{e,i}\right)\right)$$



Input features x and w

- μ_e: feature vector of edge e

$$\mu_e^{(t+1)} \leftarrow \mathrm{relu}\left(\theta_1 w_e + \sum_{i=1}^{2}\mathrm{relu}\left(\theta_2\left(\left(\sum_{u\in\aleph(v_{e,i})}\mu_u^{(t)}\right)-\mu_e^{(t)}\right)\right)+\theta_3\sum_{i=1}^{2}\mathrm{relu}(\theta_4 x_{e,i})\right)$$



$\mathbb{R}^5$
1) Absolute value of cosine direction
2) Absolute value of sine direction
3) Member length
4) Binary feature if
5) Axial stress over admissible stress

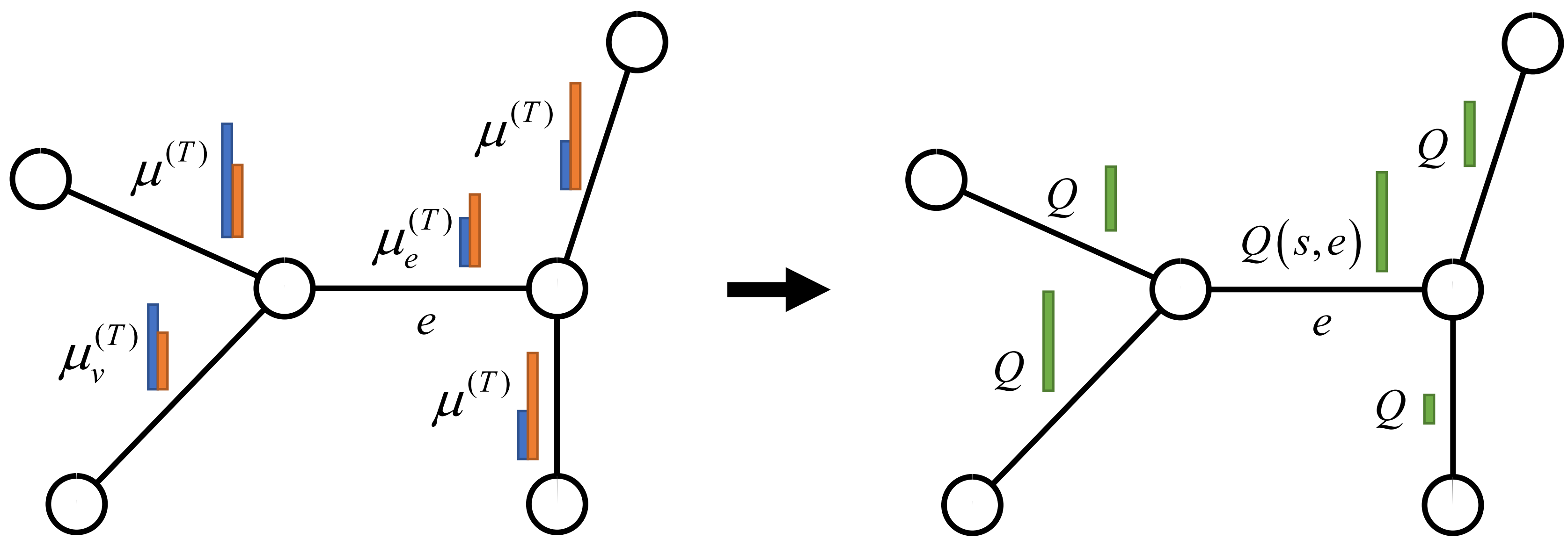
$\mathbb{R}^3$
1) Binary feature if node is pin-supported
2) Load in x direction
3) Load in y direction

$\mathbb{R}^{20}$ Embedded edge features

Q-value using embedded value

- $Q(s,e)$: value to remove edge e in current state s

$$Q(s(\mu),e) = \theta_5^{\mathrm{T}} \mathrm{relu}[\theta_6 \sum_{u \in V} \mu_u^{(T)}, \theta_7 \mu_e^{(T)}]$$



Q-learning

- Q-learning (Watkins, 1989)

$$Q(\mathbf{s},a) \leftarrow Q(\mathbf{s},a) + \alpha\left(r(\mathbf{s}') + \gamma \max_a Q(\mathbf{s}',a) - Q(\mathbf{s},a) \right)$$

observed reward + estimated total reward at next state

estimated total reward at current state

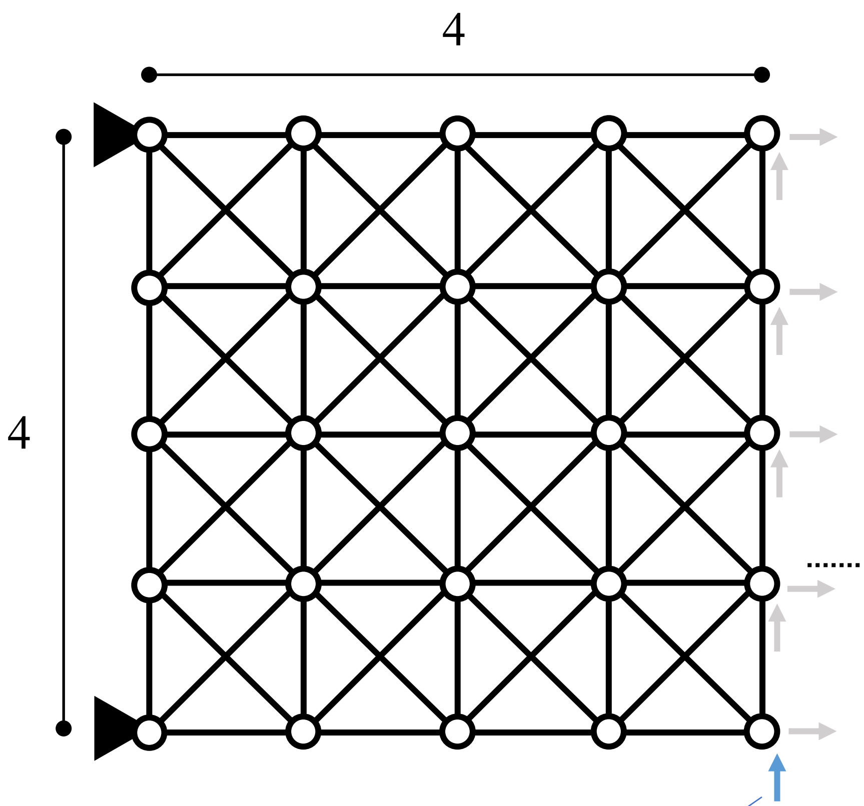
$$\rightarrow \quad \text{minimize } F(\{\theta_1,\cdots,\theta_7\}) = \left(r(\mathbf{s}') + \gamma \max_a Q(\mathbf{s}',a) - Q(\mathbf{s},a) \right)^2$$

$\mathbf{s}(\mu)$: expressed by edge embedding

a : remove an edge e with $\max_e Q(\mathbf{s}, a = e)$

r : +1 (if satisfy constraint) or -1 (else)

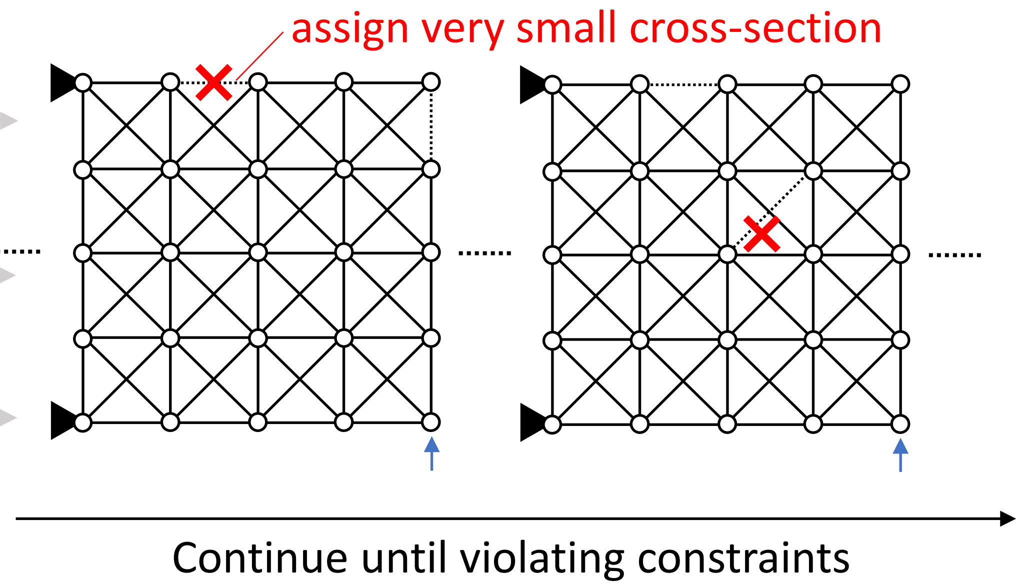
Training model



maximize *number of edges eliminated*

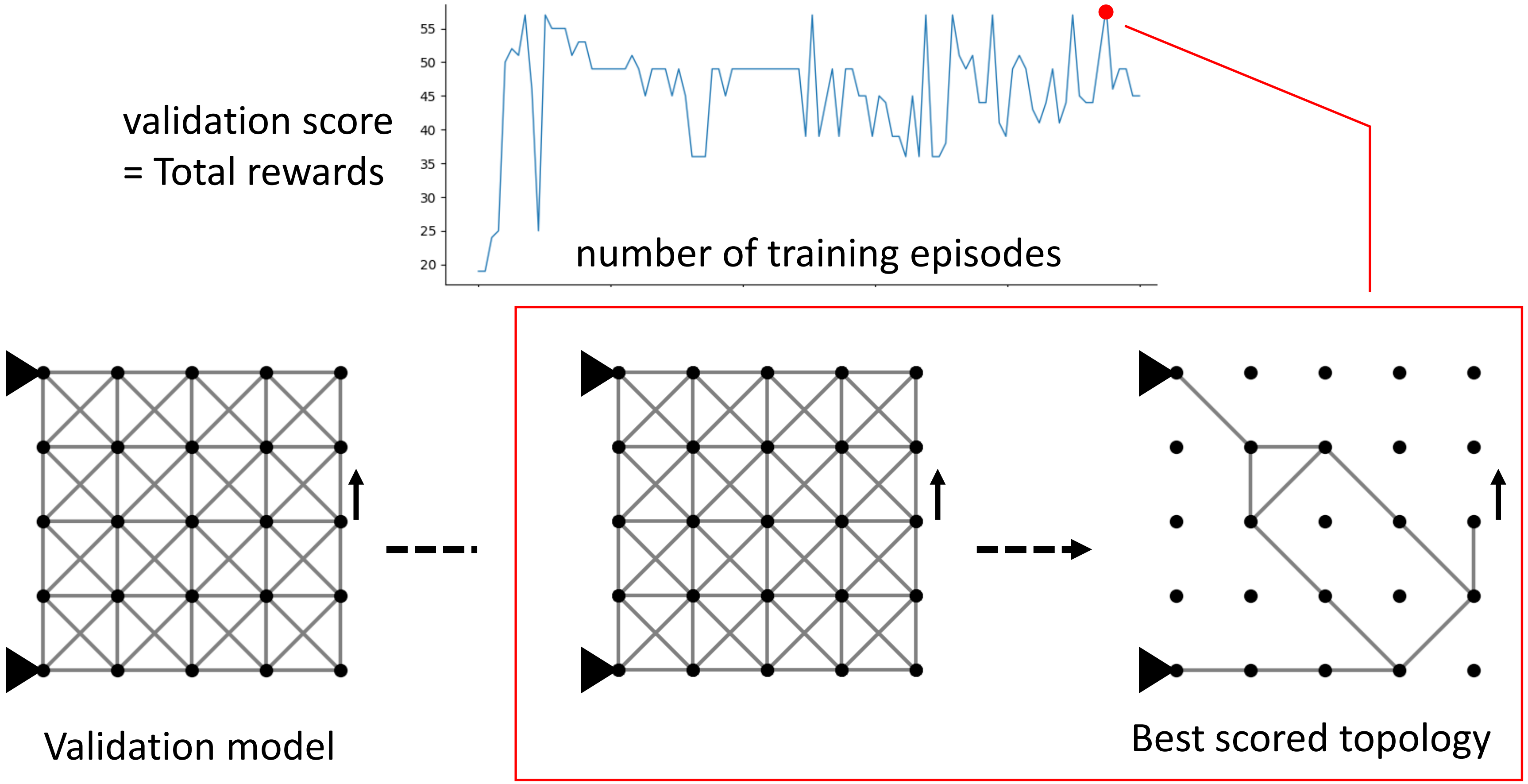
subject to $\left\|\sigma_i\right\| \leq \bar{\sigma} = 2.0 \ (i = 1, \cdots, m)$

$\left\|u_j\right\| \leq \bar{u} = 16.0 \ (j = 1, \cdots, n_p)$



Training result

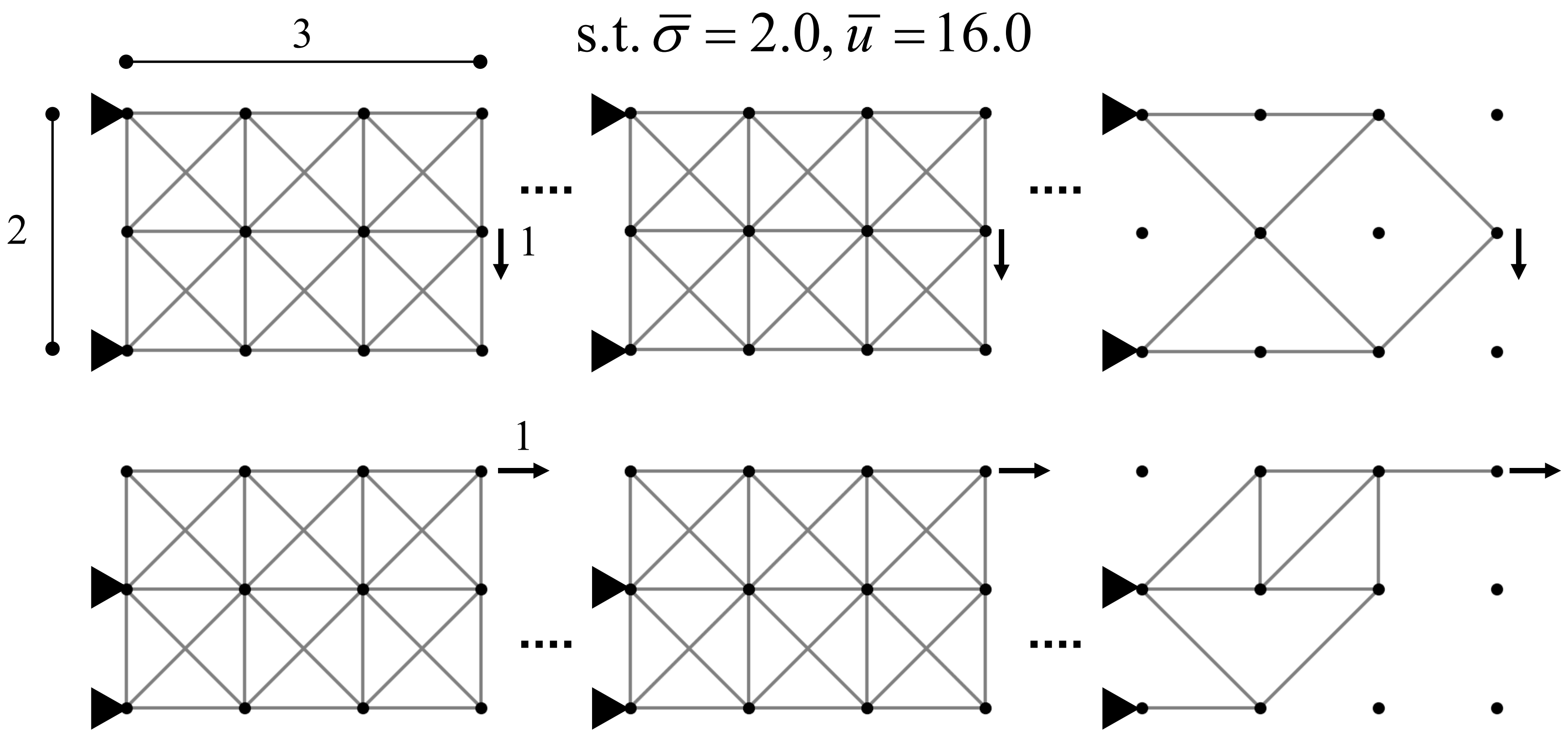
Agent learned how to remove unnecessary members



Applicability to another model

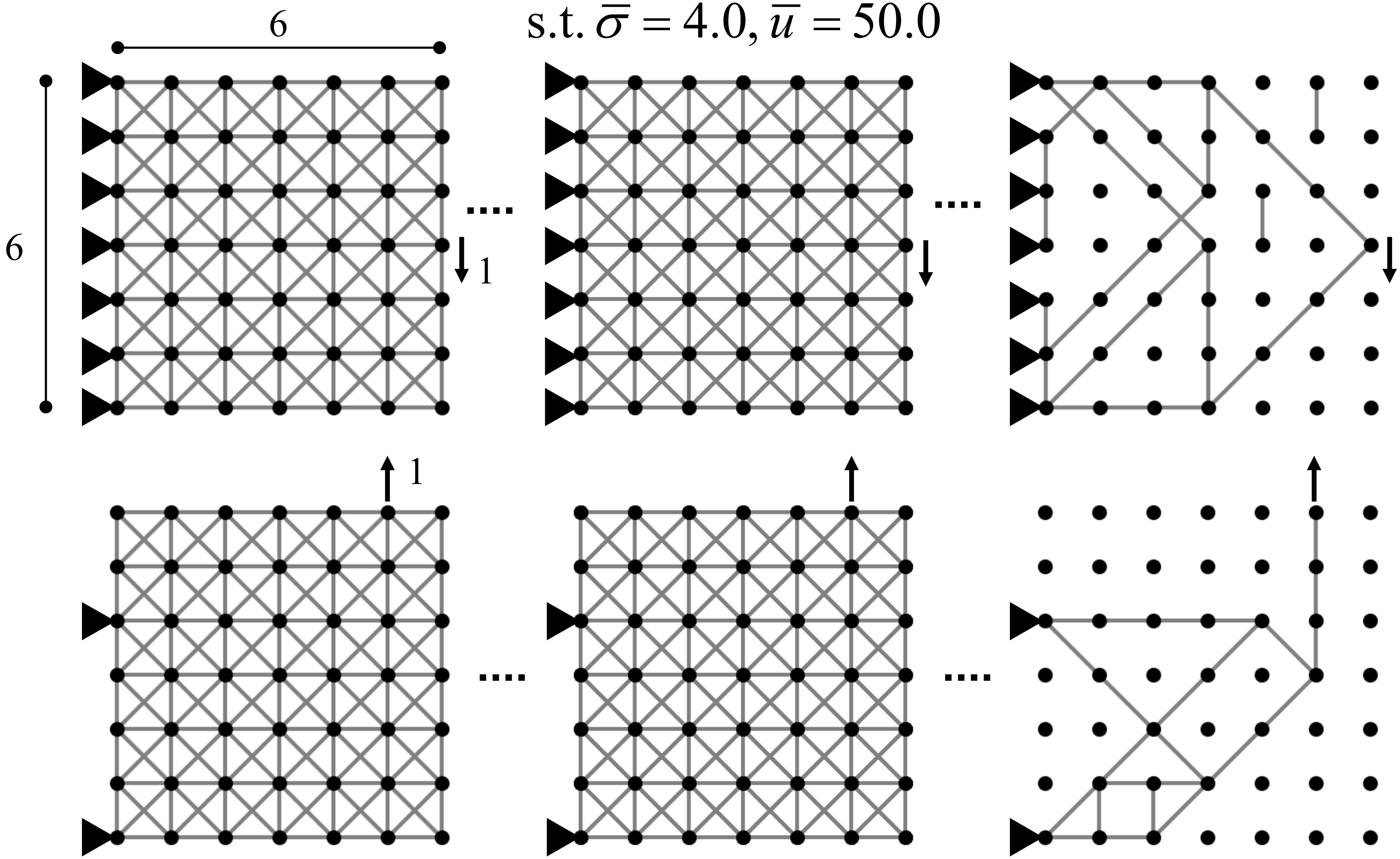
Trained agent can be used without re-training

$$\text{s.t. } \overline{\sigma} = 2.0, \overline{u} = 16.0$$



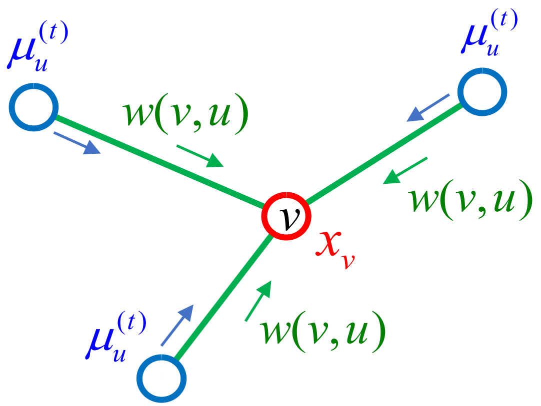
Applicability to another model

$$\text{s.t. } \overline{\sigma} = 4.0, \overline{u} = 50.0$$



Conclusion

- Graph embedding method is introduced to express features of truss members
- Conducted Q-learning based on the embedded features
- Trained agent is applicable to any topology and geometry of trusses



Email: hayashi.kazuki.55a@st.Kyoto-u.ac.jp