NANYANG TECHNOLOGICAL UNIVERSITY
SINGAPORE

Learning to Run a Power Network – A possible solution

Ziming Yan, Yan Xu
School of Electrical & Electronic Engineering
Nanyang Technological University (NTU)
Web: https://eexuyan.github.io/soda/index.html
Code available: GitHub/L2RPN_WCCI_a_Solution

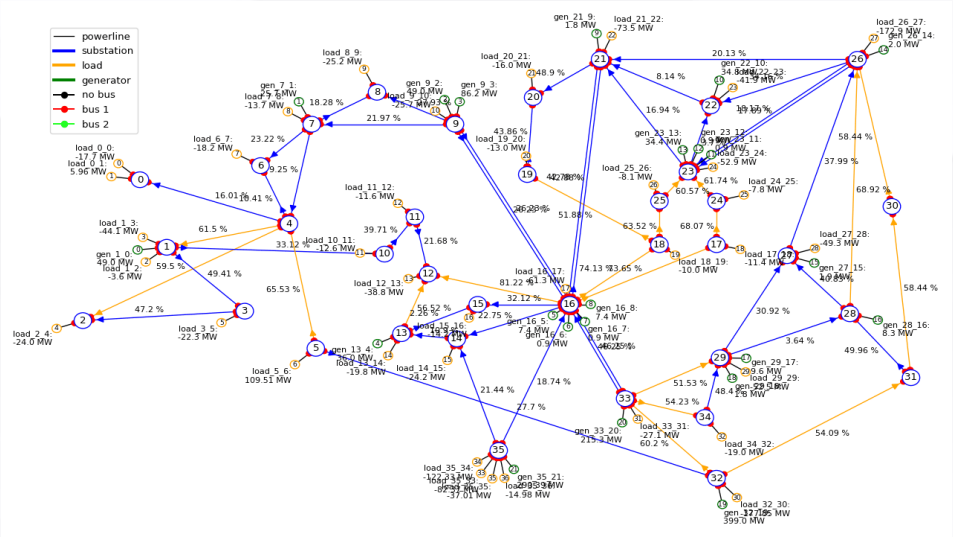
Problem description

Methodology

Tricks

Results

Objective



➢ Goal

The goal for L2RPN WCCI competition is to train an agent that will be able to learn a policy that could overcome obstacles, such as congestions, while optimizing the overall operation cost.

➢ Operation Cost

The operation cost to be minimized includes powerlines losses, redispatch cost and blackout cost. The participants are encouraged to operate the grid for as long as possible, and will be penalized for a blackout even after the game is over.

In the following pages, we presume the readers have basic knowledge for reinforcement learning and power system operation.

NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

Problem description

Methodology

Tricks

Results

Action

Topological action

Modify the structure of the grid, including line switching and connection of substations' busbar. The substations have a "double busbar layout". Eeach connection between a substation and an object (generator, load or end of a powerline) can be made on the first bus, or the second bus, or not at all if there is a disconnection.

Generation redispatch

Generation redispatch for several generators is allowed: have some generators produce more energy and others produce less. The grid operator will pay both producers for the redispatched energy at an additional cost. Load shedding is not allowed.

State

Available state

The available operational state includes time, generation, voltage, load, line flows, topology, line status, scheduled maintenance, dispatch results.

Grid parameters

The parameters of grid including line reactance, resistance, generator cost functions are not accessible by the agent.

Problem description

Methodology

Tricks

Results

Environment

Lines have thermal limits. If violated for a given amount of time, the powerline is automatically disconnected form the powergrid without any human intervention. To prevent the disconnection of more and more powerlines (phenomenon called "**cascading failure**") operators often prefer to act before the powerline is disconnected and try to re-route the flow to other powerlines.

Fluctuations of load and renewable energy, line maintenance and hazards are also considered by the competition environment. The status of grid is calculated with power flow solver.

Score

The introduction of score is in “2_Develop_And_RunLocally_An_agent.ipynb”.
The agent with less blackouts and less operation costs will be given higher score.

$$c_{\text{operations}}(t) = c_{\text{loss}}(t) + c_{\text{redispatching}}(t)$$

$$c_{\text{blackout}}(t) = \text{Load}(t) * \beta * p(t),\ \beta \geqslant 1$$

$$c(e) = \sum_{t=1}^{t_{\text{end}}} c_{\text{operations}}(t) + \sum_{t=t_{\text{end}}}^{T_e} c_{\text{blackout}}(t)$$

$$Score = \sum_{i=1}^{N} c(e_i)$$

NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

Problem description

Methodology

Tricks

Results

Summary

We present a policy-based DRL agent for the competition. Two agents are trained with different strategies/tricks and (randomly selected) datasets. To improve the control performance, the agents will backup each other during the test phase. The proposed method is inspired by the works of GEIRINA (https://github.com/shidi1985/L2RPN) and Amar (https://github.com/amar-iastate/L2RPN-using-A3C).

The agent is trained mainly based on A3C (Asynchronous Advantage Actor-Critic).

- Rewards are based on L2RPNReward (provided by environment) and penalties on constraints violation.
- Action space is randomly selected. (Further improvement is entirely possible)
- Agent is forced to try out more actions when constraints are violated.
- Agent is forced to randomly explore with a small probability.
- Two agents with different setting are trained to backup each other.

Asynchronous Advantage Actor-Critic

Actor

$$d\theta = d\theta + \Delta_{\theta'} log(\pi(a_i|si;\theta')(R - V(s_i;\theta'_v)))$$

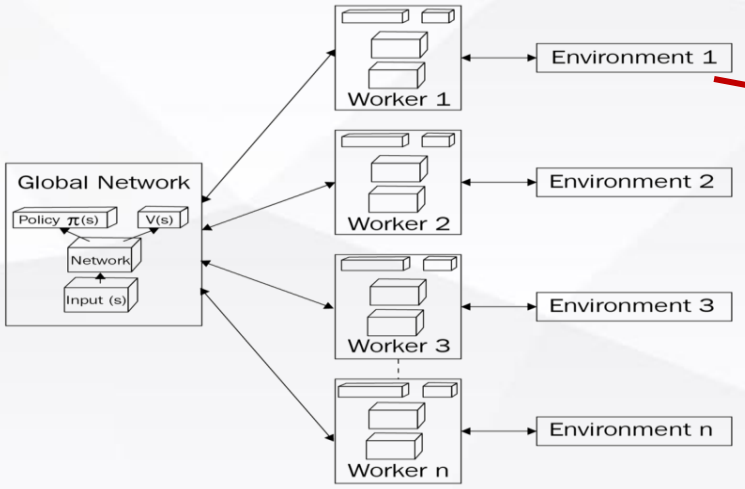
Advantages item: the agent learns how much better the rewards were than it's expectation; faster and more robust learning process

Critic

$$d\theta_v = d\theta_v + \frac{\partial((R - V(s_i;\theta'_v))^2)}{\partial\theta'_v}$$

Estimate state value (action is not input)

Asynchronous



The global network is constantly updated by each worker as they interact with its own environment

The overall experience for training is more diverse.

http://www.henrypan.com/blog/reinforcement-learning/2019/02/27/a3c-rl-layments-explanation.html

Rewards

L2RPNreward

Provided by environment, representing the performance of economic operation.

$$R_{L2RPN}(t)$$

Penalties on constraints violation

A common practice to consider constraints is penalties on constraints violation:

$$R_{violated}(t) = -\sum_i \beta_i ReLU(\frac{flow_i}{limit_i} - 1), \beta_i > 1$$

For simplify, the coefficients β_i are constant and not optimized/variable.

Avoid "fear"

If penalties are high, the agent may try to end the game early, which is called "fear"

$$R(t) = \max(-10, R_{L2RPN}(t) + R_{violated}(t))$$

End game penalty

If the game is ended due to blackout or power flow divergence, large penalty is given.

Selection of Action space

Agent 1 topology action:

1 no action +200 selected actions + 100 random actions + 295 line set

Substations

Size = 66525

Reason: at substation 16, the number of actions is about 2^16 = 65536

However, we find that the flow limits are still sometimes violated even when exhaustivity try out all these actions. The action space can be further improved.

Backup Agent topology action:

1 no action +200 selected actions + 100 random actions + 295 line set

Substations

Size = 66525

Selection of Action space

Redispatch action: modelled as discrete actions due to strict ramping limits.

Adjustable generator	Bus	Upper limits	Ramping limits
0	1	50	1.39
2	9	50	1.39
3	9	250	10.39
4	13	50	1.39
10	22	100	2.79
13	23	100	2.79
16	28	150	4.30
19	32	400	2.79
20	33	300	8.5
21	35	350	9.89

Problem description

Methodology

Tricks

Results

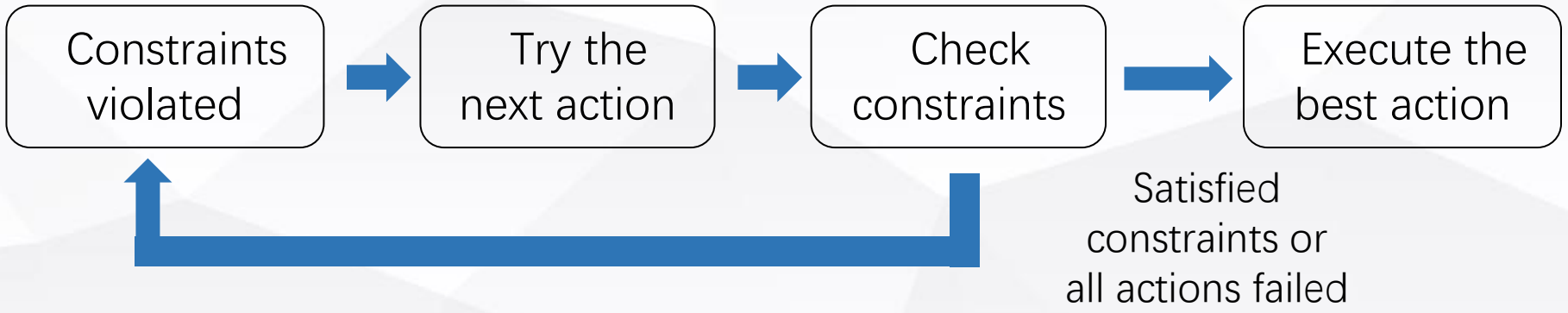
Tricks

Forced exploration

For more robust critic/actor, the agent is forced to randomly explore with a small probability.

Forced verification of constraints

In the backup agent, if constraints are violated, the agent will be forced to try out all actions, to see if any action can satisfy all the constraints.



Forced verification of constraints during learning process

Problem description

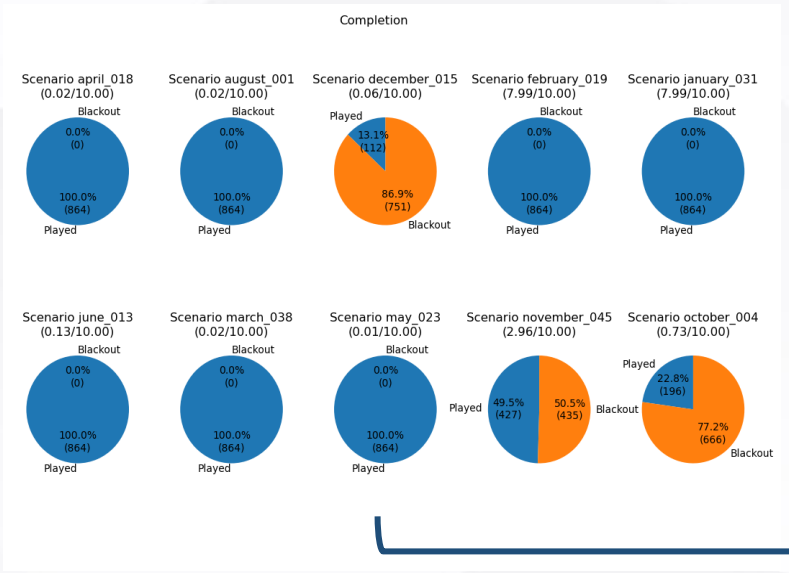
Methodology

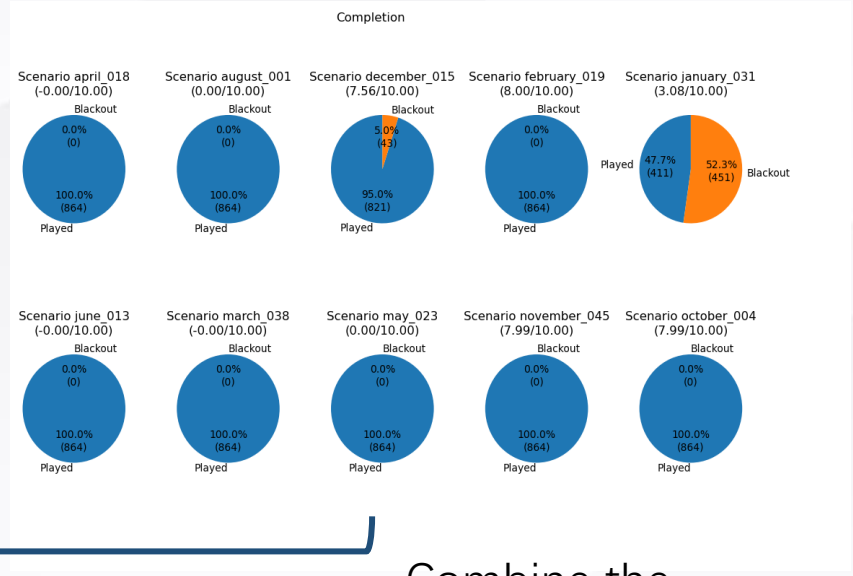
Tricks

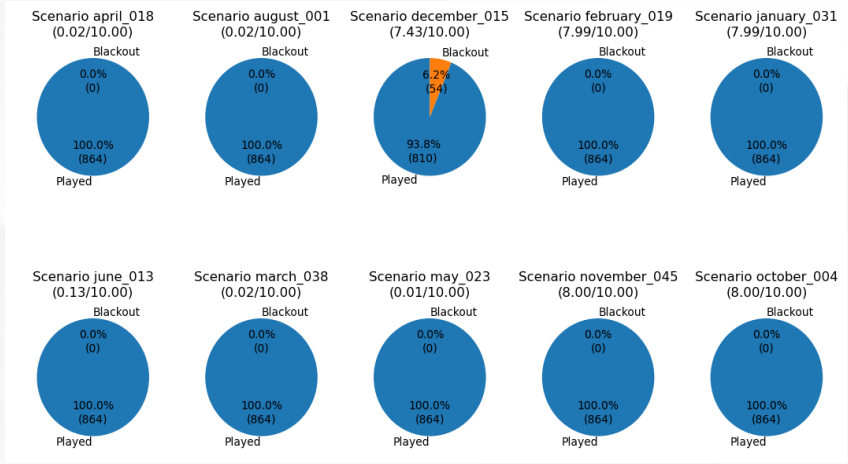
Results

Two different agents backup each other

If the performance of agents are different, they may backup each other: when one agent fails, another may work.







Combine the two agents

The submitted agent can perform well on more datasets.

NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

Problem description

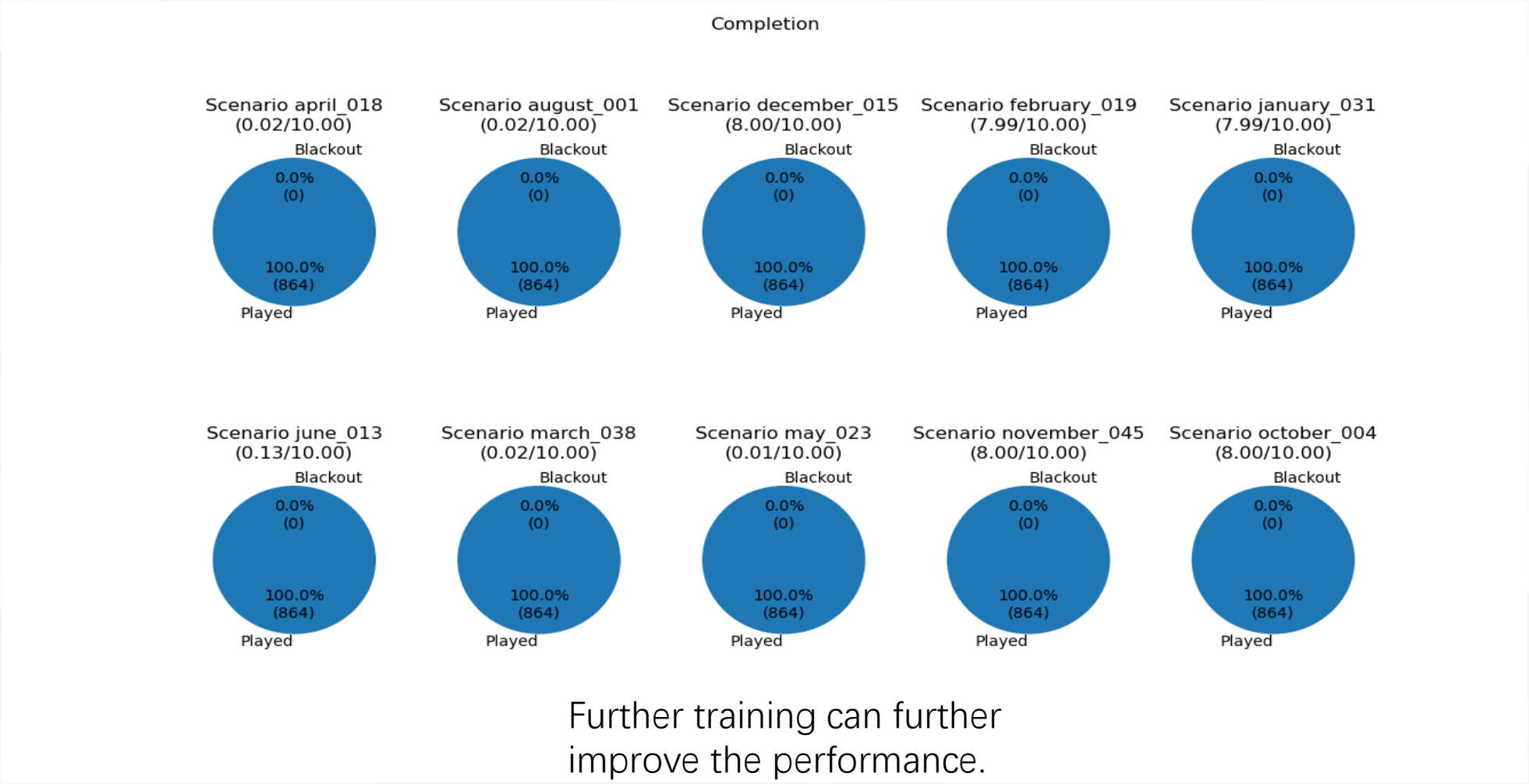
Methodology

Tricks

Results

After the competition: further training improves the performance

Due to time/PC limitations, the agent was not fully trained for the competition. We find that further training on more datasets can improve the performance of our agent:



Further training can further improve the performance.

NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

Problem description

Methodology

Tricks

Results

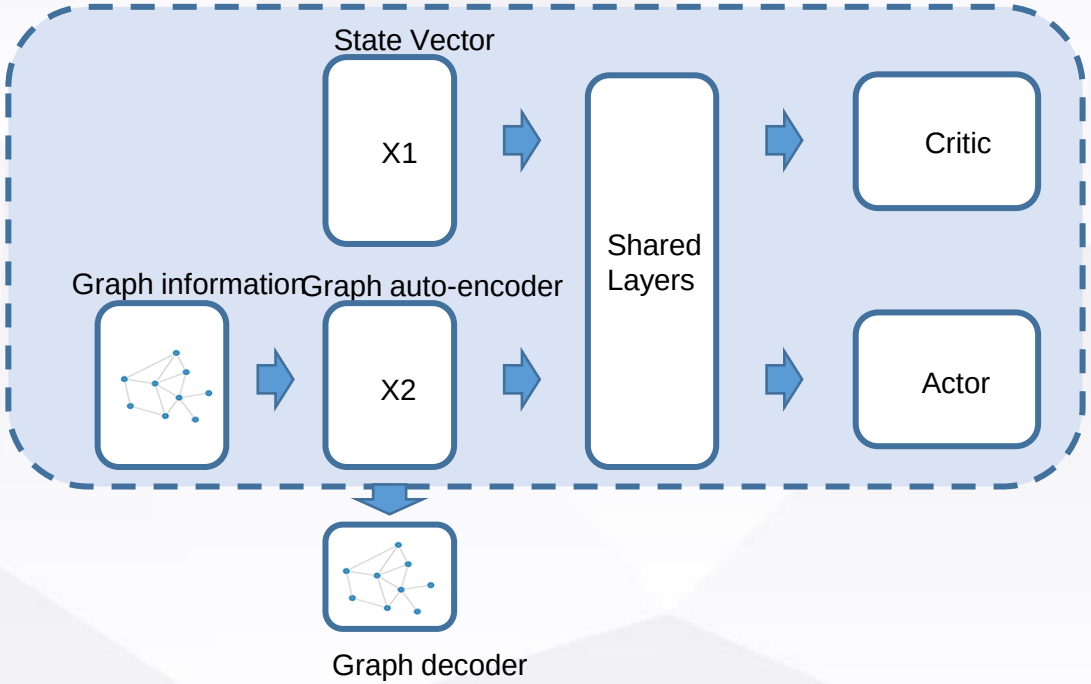
Future chance

In the future, other ideas are also worth trying:

1. Graph-based neural network may be employed into input layers

2. Better consideration of constraints.

3. Experience replay.



THANKS

Ziming Yan | Student
Dr Yan Xu | Assistant Professor
School of Electrical & Electronic Engineering
Nanyang Technological University, Singapore
Email: yanzmics@gmail.com, xuyan@ntu.edu.sg