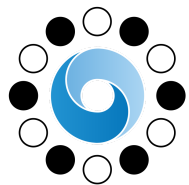# Robust probabilistic target-oriented exploration with reliability approximation

Tokyo Denki University

Moto Shinriki, Yu Kono, Tatsuji Takahashi

Background

Reinforcement learning (RL) agents have reached superhuman levels in games such as Go and chess.

Task complexity

In terms of the complexity of the environment,
real-world tasks are more challenging than games.

Reinforcement learning and human learning

[Problem] RL is still too costly to use in the real-world tasks.

- The required amount of sampling and exploration for <u>optimization</u> is not feasible in a realistic time frame.

[Idea] Can we solve this by imitating human learning?

- We focus on **satisficing**, a learning tendency of humans.
- We introduce the concept of an **aspiration level** into reinforcement learning.
- We generalize the goal of reinforcement learning from optimization into satisficing.

<div align="center">

We implemented **target-oriented exploration**.

</div>

The goal of this study

[Our main goal]

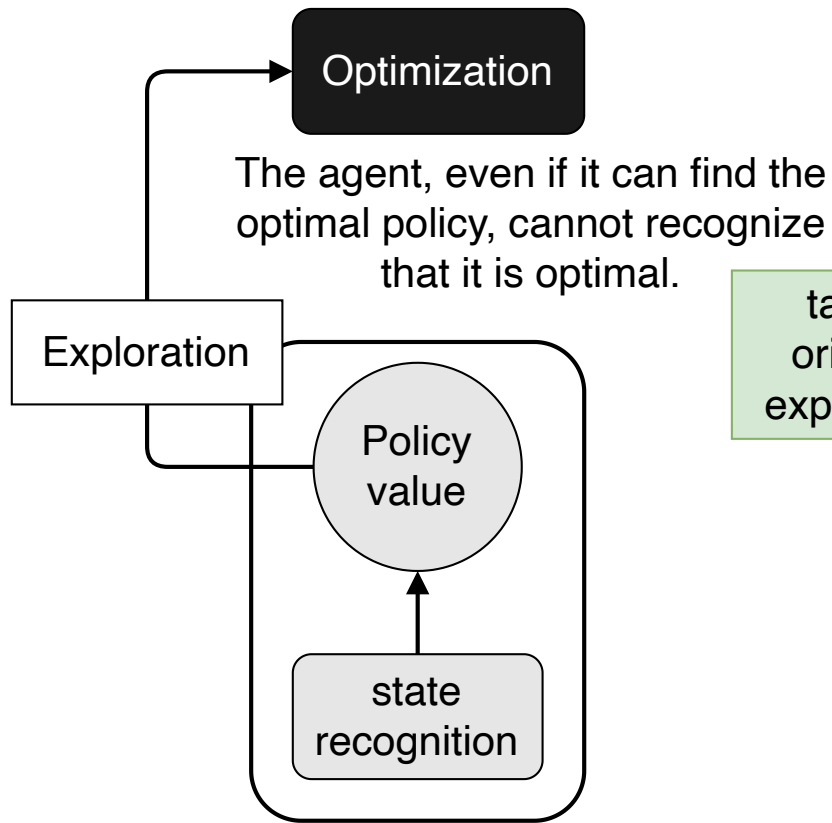Application of target-oriented exploration to deep reinforcement learning

[Our sub goal]

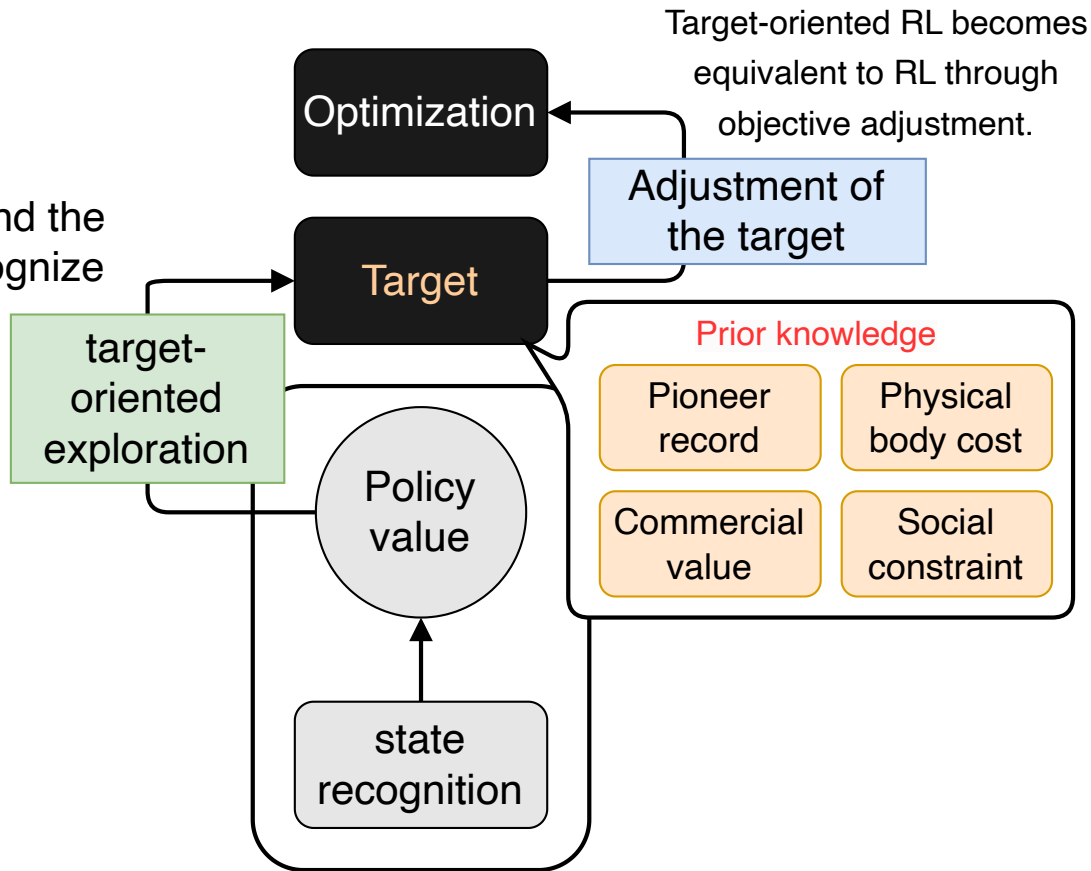Stochastic generalization of the action selection of target-oriented exploration methods

Milestones of our sub goal

1. In this study, we generalize the existing state approximation methods.

2. We show that our new method works without performance degradation.

3. Our goal is to show that the new method performs better than representative methods.

Conventional RL

Target oriented RL

**Optimization**

The agent, even if it can find the optimal policy, cannot recognize that it is optimal.

Exploration

Policy value

state recognition

Target-oriented RL becomes equivalent to RL through objective adjustment.

Optimization

Adjustment of the target

Target

target-oriented exploration

Policy value

state recognition

Prior knowledge

Pioneer record

Physical body cost

Commercial value

Social constraint

Related research

Method	Overview	Mechanism	Features
Risk-sensitive Satisficing (RS)	Combination aspiration level with RL	▪ Off-policy ▪ Deterministic	Better performance than other methods in bandit problems and RL problems
Regional Linear RS (RegLinRS)	State-distinguishable RS	▪ Off-policy ▪ Deterministic	Better performance than other methods in contextual bandit problems

We want to generalize deterministic action selection into stochastic action selection.
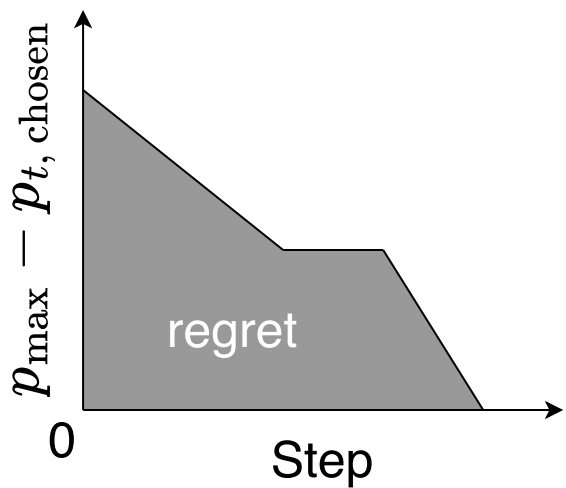
Contextual Bandit Problems

$$\hat{p}_{t,i} = \mathbf{x}_t^T \boldsymbol{\theta}_i + \epsilon_t$$
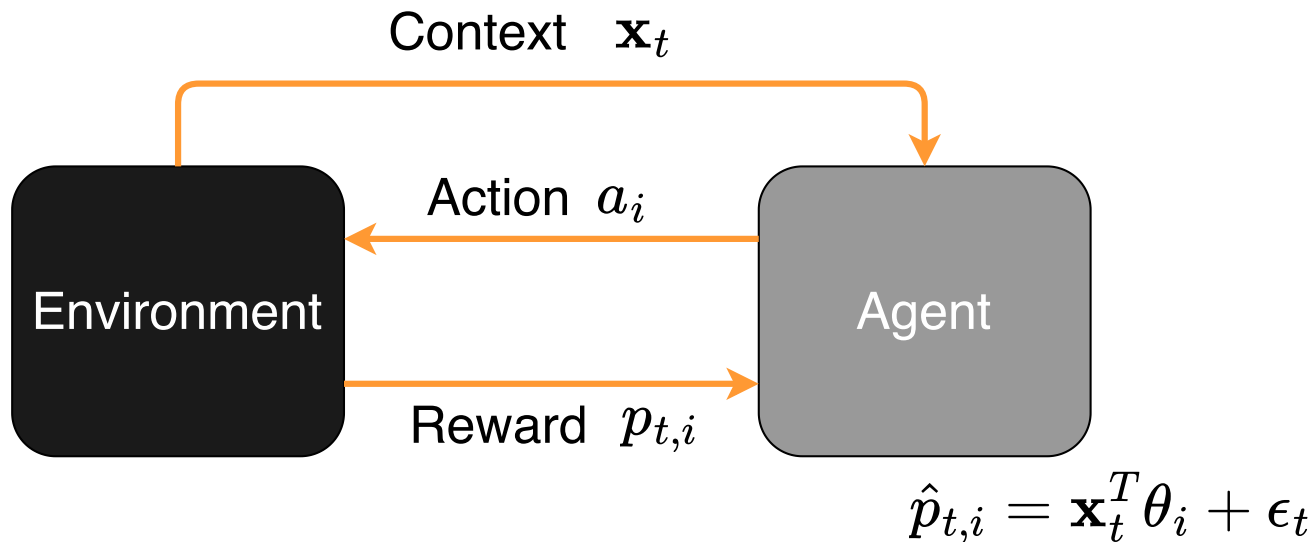
- $p_{t,i}$: Reward expectation value of action a_i
- $\boldsymbol{\theta}_i$: The parameter of the reward expectation value
- ϵ_t: The error term with an expected value of 0
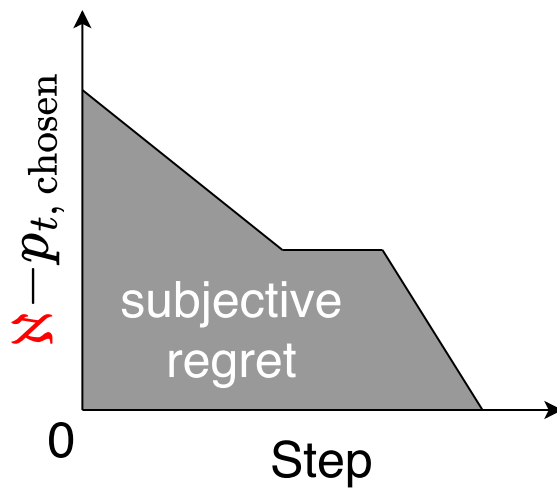
Regret

Evaluation index

$$\text{regret} = \sum_{t=1}^{T}(p_{\max} - p_{t,\,\text{chosen}})$$

Subjective regret

Implementation of target-oriented exploration

$$I_i^{\text{SR}} = \sum_{t=1}^{T}(\aleph - p_{t,\,\text{chosen}})$$

- $\aleph$: Aspiration level

Risk-sensitive Satisficing (RS)

Implementation of target-oriented exploration

- We define the RS value function as $I_i^{\mathrm{RS}} := -I_i^{\mathrm{SR}}$.
- The agent chooses an action by taking the argmax from I_i^{RS}.

$$I_i^{\mathrm{RS}} = \frac{n_i}{N}(p_i - \aleph) = \frac{n_i}{N}\delta_i$$

- n_i: The number of times the agent chose action a_i
- N: The total number of times the agent chose an action
- n_i/N: Reliability (Choice probability) of action a_i

- δ_i: Reflection effect of prospect theory → Difference between aspiration level $(p_i - \aleph)$
  - By multiplying the reliability and δ_i, the agent makes optimistic or pessimistic action choices depending on the situation.

Local approximation of reliability

- The agent approximates and estimates reliability using episodic memory and k-nearest neighbor.
- **Regional Linear RS** (RegLinRS)
  - Tsuboya et al., 2023
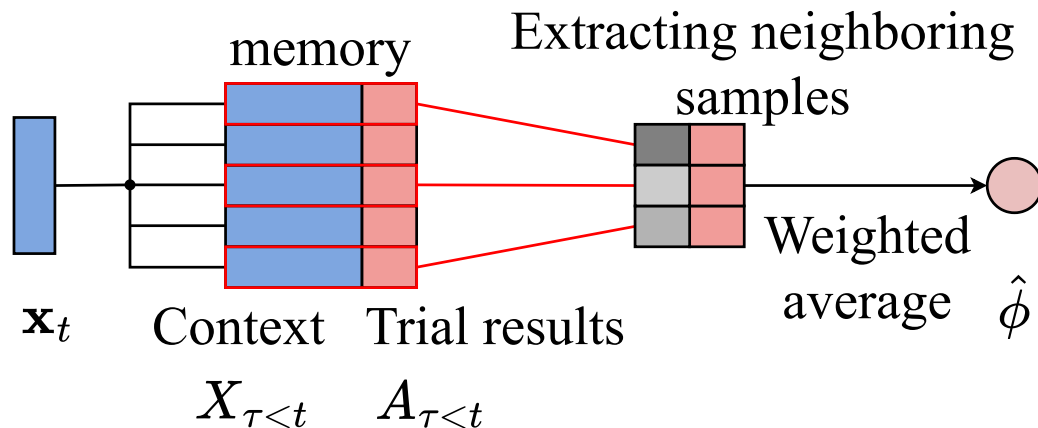
Stochastic RS (SRS)

- We can estimate the reliability of the actions that the agent has.
- We generate a probability distribution from the difference between the estimated reliability and the actual reliability.

$$I_i^{\mathrm{RS}} = -Z$$

$$\rho_i^z = \frac{Z}{\aleph - p_i}$$

$$I_i^{\mathrm{SRS}} = \left\{ \max_i \left(\frac{\rho_i}{\rho_i^z} \right) + \epsilon \right\} \rho_i^z - \rho_i$$

$$\rho_i = \frac{n_i}{N}$$

$$b_i = \rho_i / \rho_i^z - 1 + \epsilon$$

$$\pi_i = I_i^{\mathrm{SRS}} / \sum_{j=1}^{K} I_j^{\mathrm{SRS}}$$

RS (Diterministic)

| 100% | 0% | 0% |

SRS (Stochastic)

| 80% | 15% | 5% |

Regional Linear SRS (RegLinSRS)

- **Our new method**

- With this method, the reliability estimation part of RegLinRS is combined with SRS.

Summary so far

The features of each method

Method	Reliability estimation	Action selection
RS	-	Deterministic
SRS	-	Stochastic
RegLinRS	Approximation using episodic memory and k-nearest neighbor	Deterministic
RegLinSRS	Approximation using episodic memory and k-nearest neighbor	Stochastic

Summary so far

The features of each method

Method	Reliability estimation	Action selection
RS	-	Deterministic
SRS	-	Stochastic
RegLinRS	Approximation using episodic memory and k-nearest neighbor	Deterministic
RegLinSRS	Approximation using episodic memory and k-nearest neighbor	Stochastic

Artificial dataset

- We created an artificial dataset in which the aspiration level $\aleph$ is always constant.
  - The purpose is to compare RS methods (RegLinRS and RegLinSRS) with other methods using the same evaluation index.

Configuration Item	Configuration Value	
Feature vector dimension d	128	
Number of actions K	8	※ $\aleph_{opt}$ is the reference value set between the optimal and suboptimal.
Optimal Aspiration level $\aleph_{opt}$	0.7	

Experiment 1

RL methods

LinUCB, LinTS, and RS methods (RegLinRS, RegLinSRS)
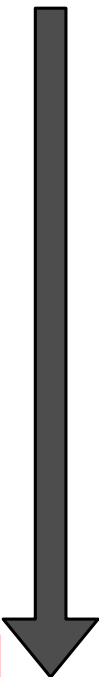
Experimental settings

- We ran 1,000 simulations, with 100,000 steps per simulation.
  - We calculated the average values and used them as the result.
- The agent initially selects each action 10 times.
  - This setting is necessary for parameter initialization.
- We set the batch size to 20 for all methods.
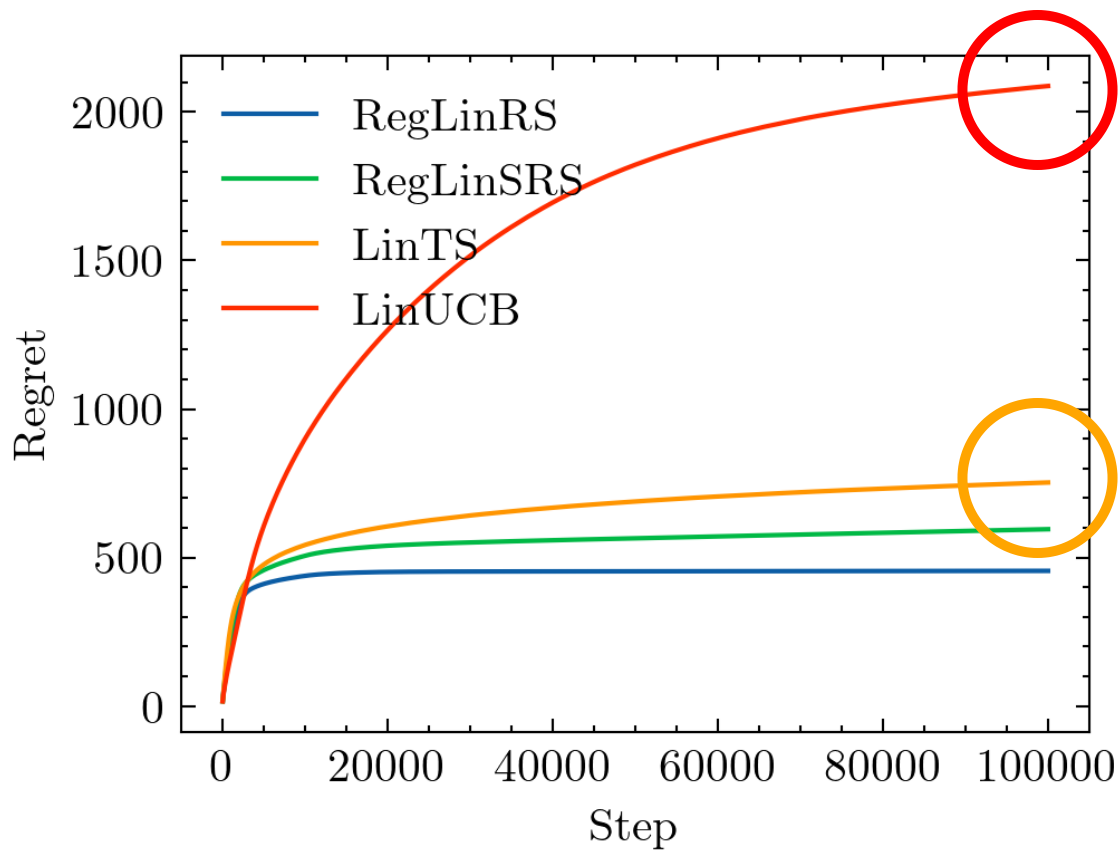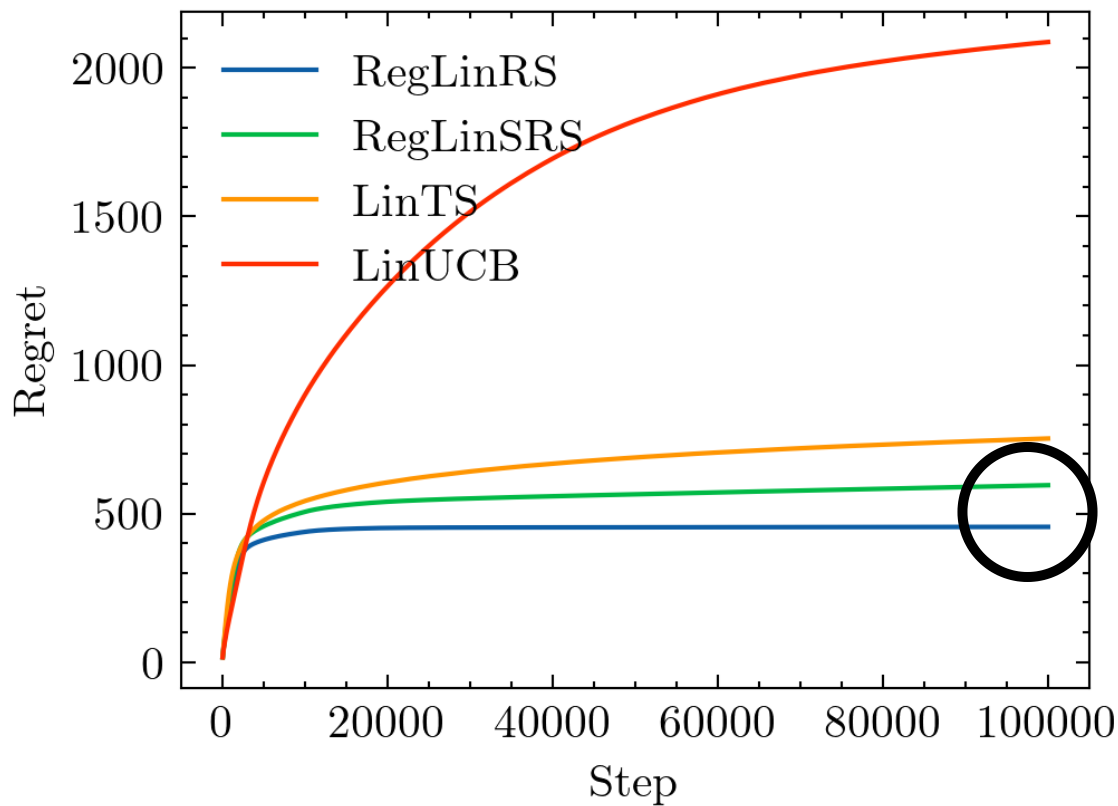
Result

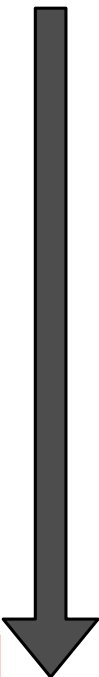Experiment 1
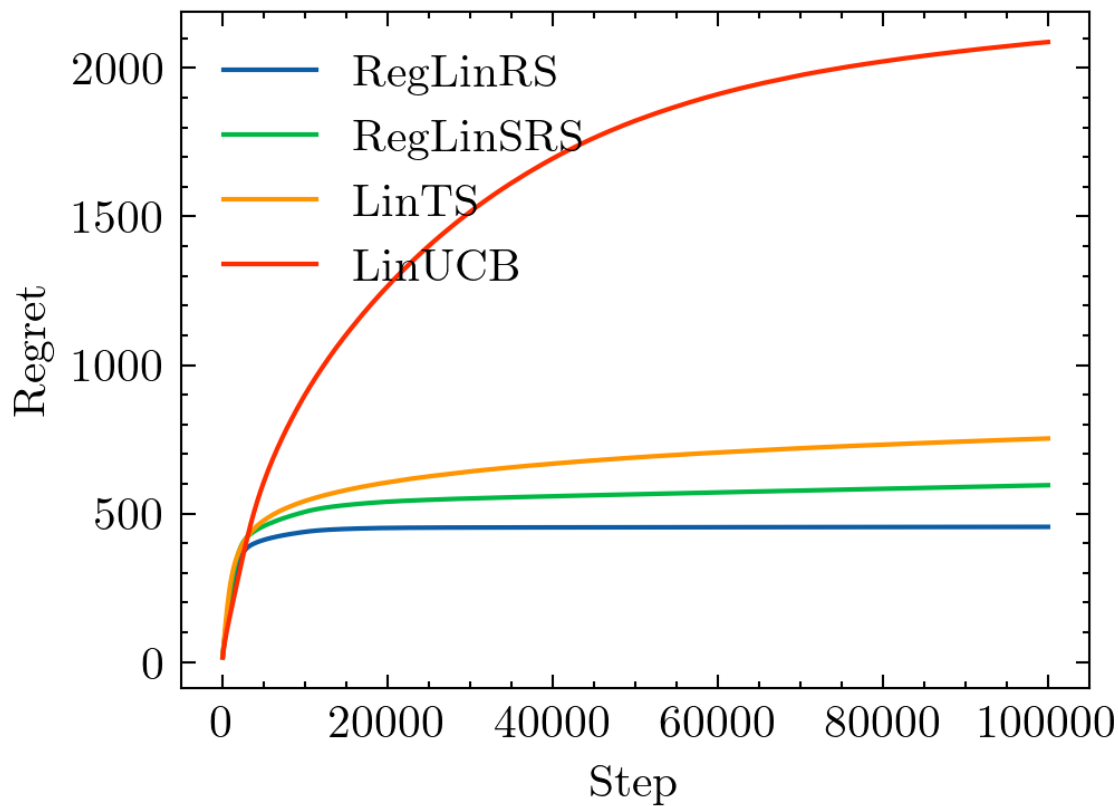
Good

Result

Experiment 1

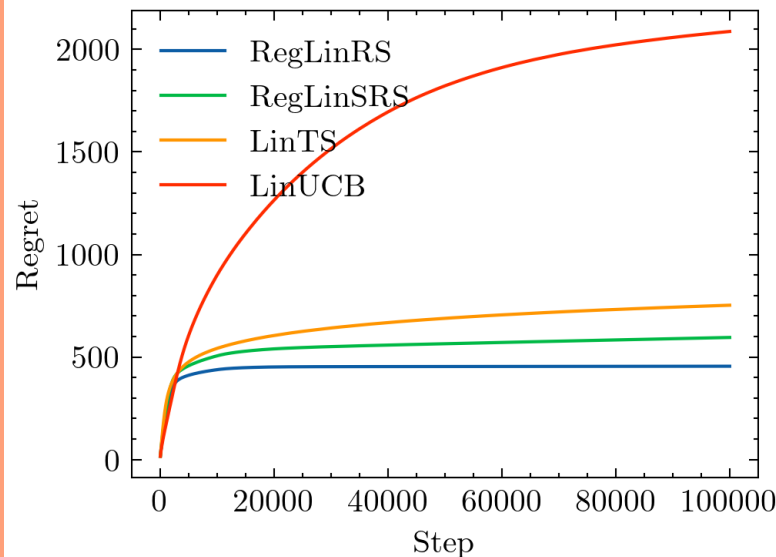Good

Result

Experiment 1

Good

Result

- LinTS is one of the state-of-the-art methods (Agrawal et al., 2019).
- However, from the early steps, the regret of LinTS is larger than that of RegLinRS and RegLinSRS.

A question:

RegLinRS, RegLinSRS can select the truly optimal action even in situations where accurate approximation has not yet been achieved?
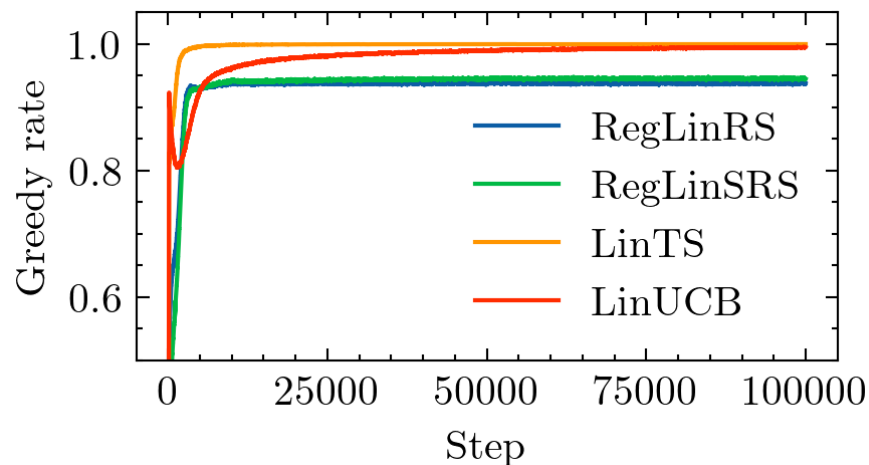
Discussions

Experiment 1

- LinTS and LinUCB have reached a Greedy rate of 1.0.
- RegLinRS and RegLinSRS have stopped at a Greedy rate of over 0.9.
    - About once in 10 times, they were not greedy.

→ RegLinRS and RegLinSRS can choose the truly optimal action even if the action is overestimated.

Hypothesis

Experiment 1

[Fact] RegLinRS and RegLinSRS can partially mitigate the effects of approximation errors.

- These methods can choose the truly optimal action even if the action is overestimated due to approximation errors.

[Hypothesis] Are RegLinRS and RegLinSRS achieving this using reliability?

- RS does not necessarily make greedy action selection, because it uses the reflection effect of reliability.
- This property is effective in the sense that the agent can choose a satisfactory action with certainty.

$\rightarrow$ We conducted an experiment to verify this property
by intentionally adding noise to the reward expectation values.

Experiment 2

Purpose of Experiment

- **To verify the robustness of RS against approximation errors**
    - We added noise to the estimated reward expectation value.
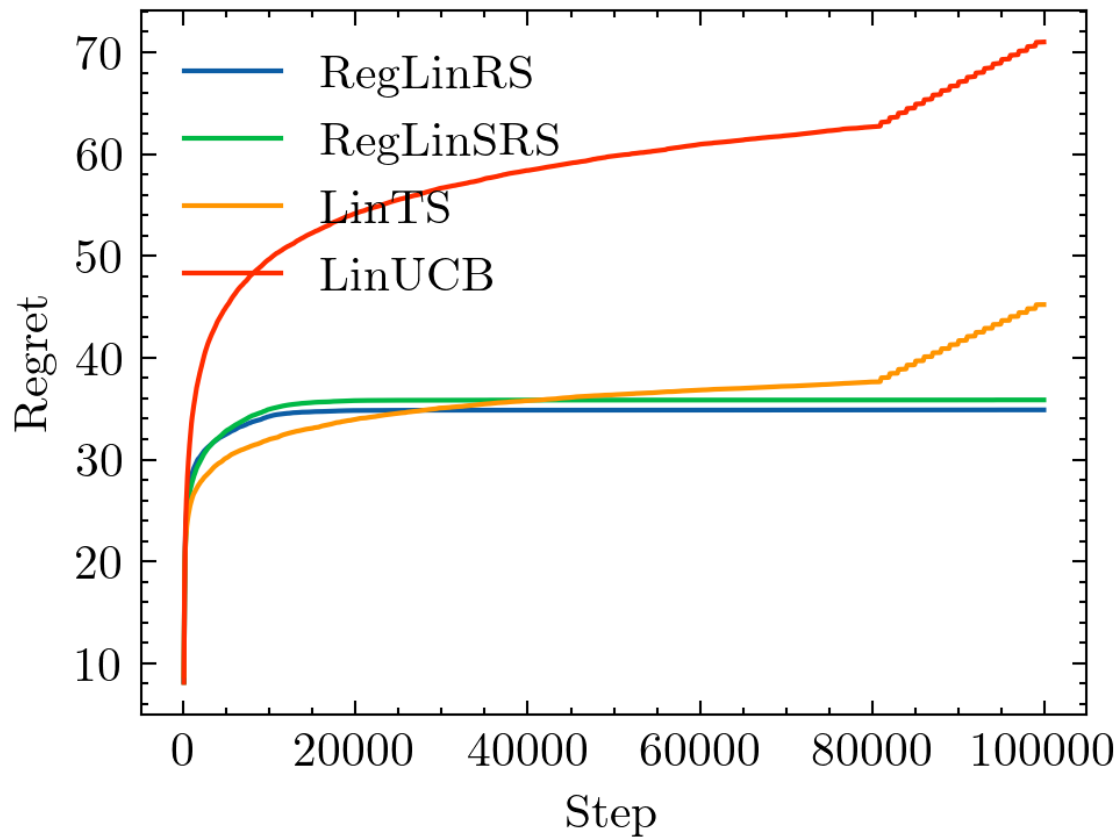   - We intentionally created a situation where it is easy to select a non-optimal action.

Experimental settings

- We set the number of actions to $K = 2$, to simplify Experiment 1.
- We added noise to the estimated reward expectation value at equal intervals after 80,000 steps.
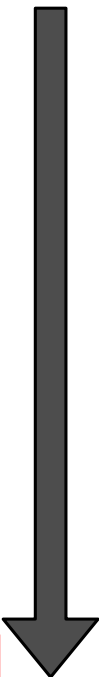- We set the other settings the same as in Experiment 1.
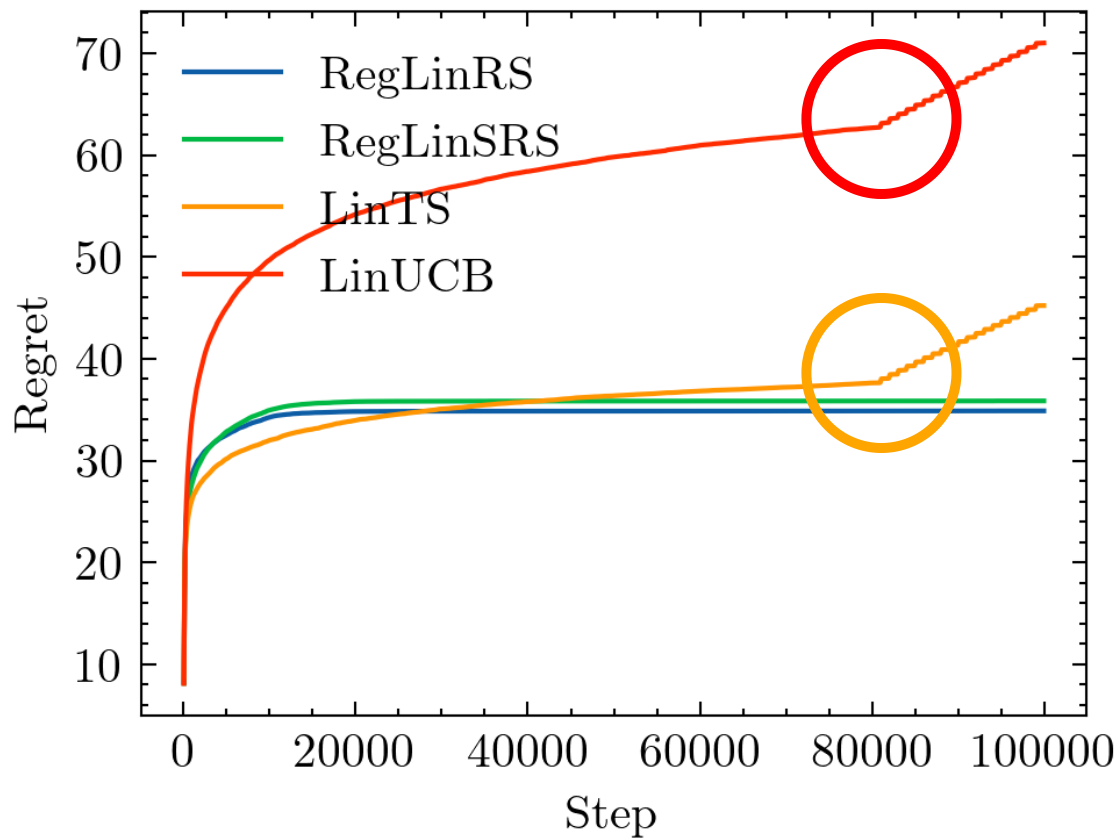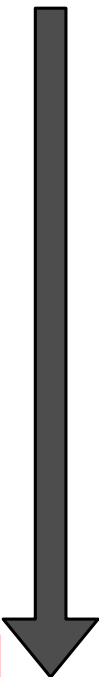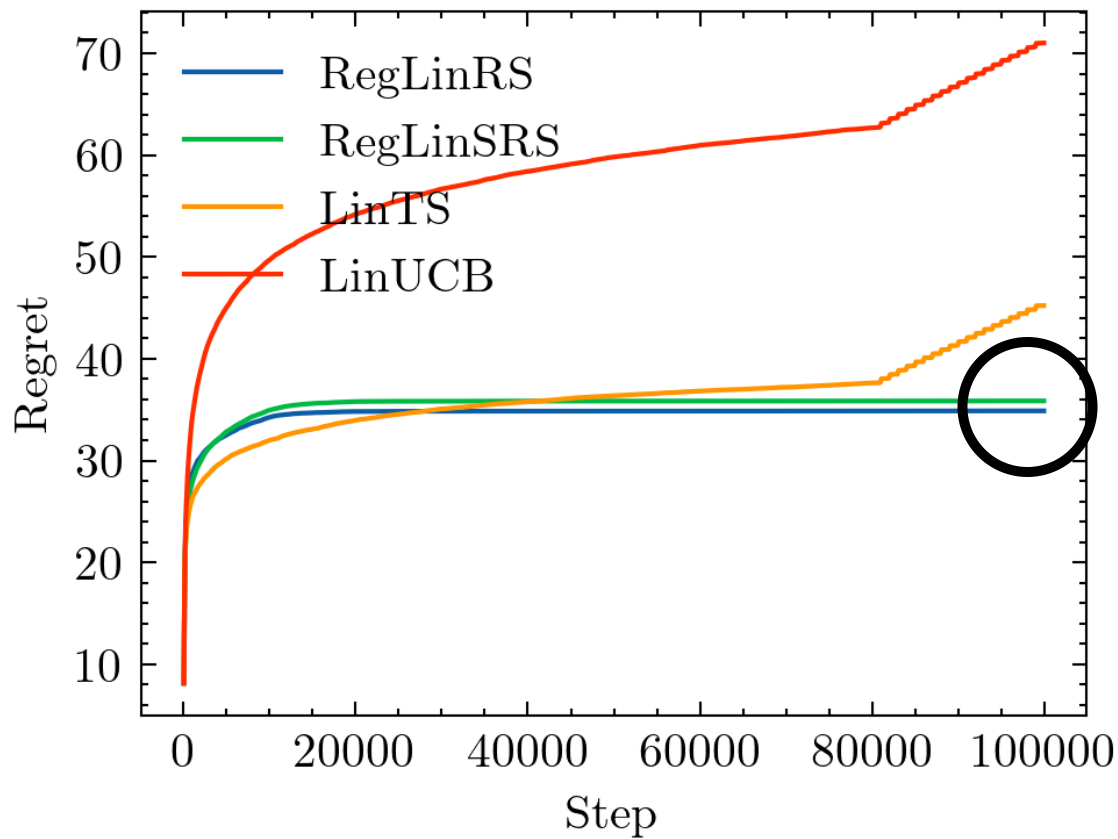
Result

Experiment 2

Good

Result

Good
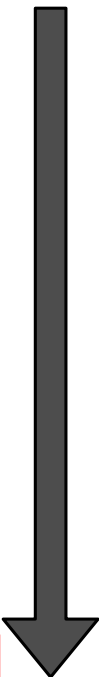
Result

Experiment 2

Good

Result

Experiment 2

- RegLinRS and RegLinSRS have no increase in regret.
    - The latter two methods can partially mitigate
      the effects of approximation errors
      by the reflection effect of reliability.

→ It shows the robustness
of RS to approximation errors.
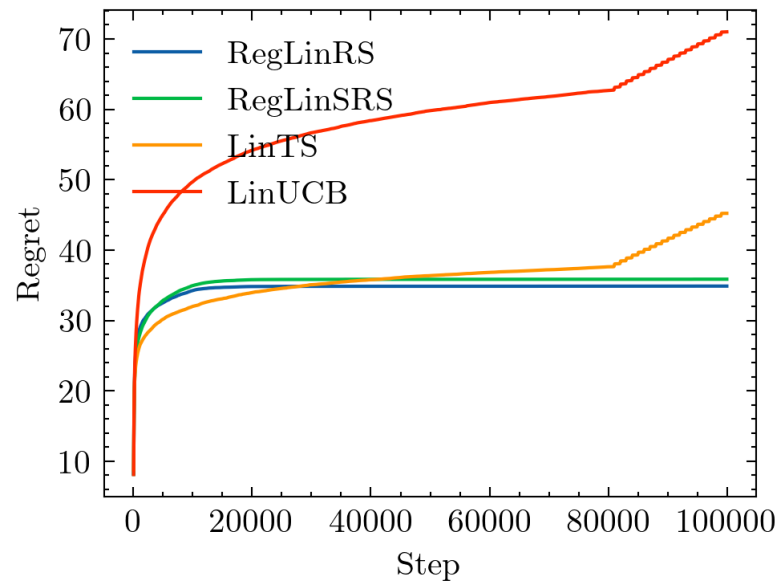
Conclusion

Goals	Contents	Achievement
1	Generalization of RegLinRS	✅
2	Performance comparison with RegLinRS	✅
3	Performance comparison with LinTS and LinUCB	✅
	Additional	
4	Robustness of RS to approximation errors	✅

→ We are now prepared for the application of RS to deep RL.

EOP

Appendix

Reason of generalization diterministic to stochastic
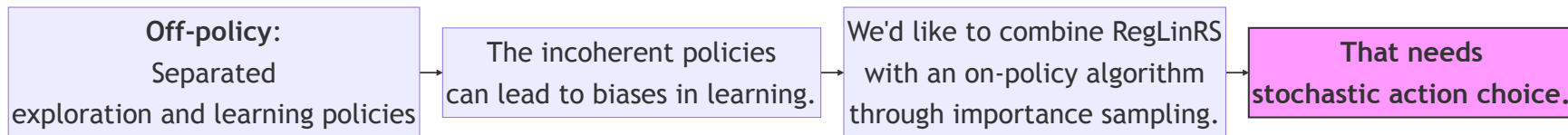
RegLinRS to RegLinSRS

We want to generalize deterministic action selection into stochastic action selection.

Because we want to combine RegLinRS with an on-policy algorithm through importance sampling. [1]

| Off-policy: Separated exploration and learning policies | → | The incoherent policies can lead to biases in learning. | → | We'd like to combine RegLinRS with an on-policy algorithm through importance sampling. | → | That needs stochastic action choice. |

1. L. Espeholt, H. Soyer, R. Munos et al. (2018) IMPALA: Scalable Distributed Deep-RL with Importance Weighted Actor-Learner Architectures
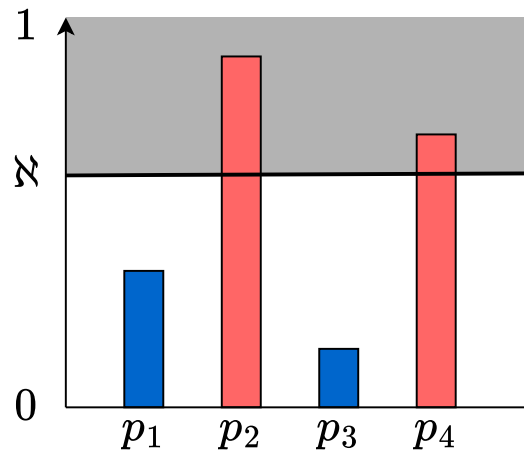
Under-archieved and Over-archieved situations

Under-archieved situation

All reward expectation values are less than ℵ.
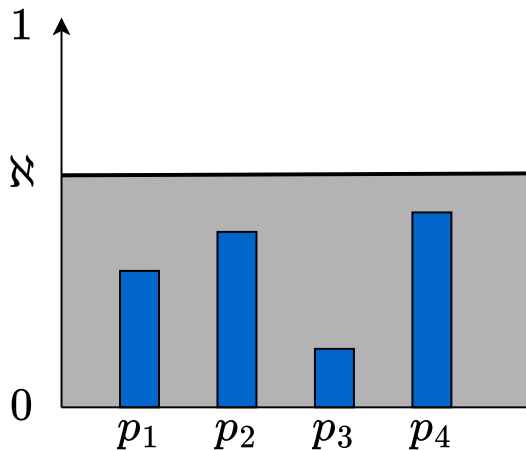
Over-archieved situation

At least one reward expectation value is greater than or equal to ℵ.

Exploration area

Reflection effect of reliability

Difference in selection tendency between under-archieved and over-archieved situations

$$I_i^{\mathrm{RS}} = \frac{n_i}{N}(p_i - \aleph)$$

Over-archieved situation

- The higher the reliability, the higher the I_i^{RS}.

- The agent makes a pessimistic action selection.
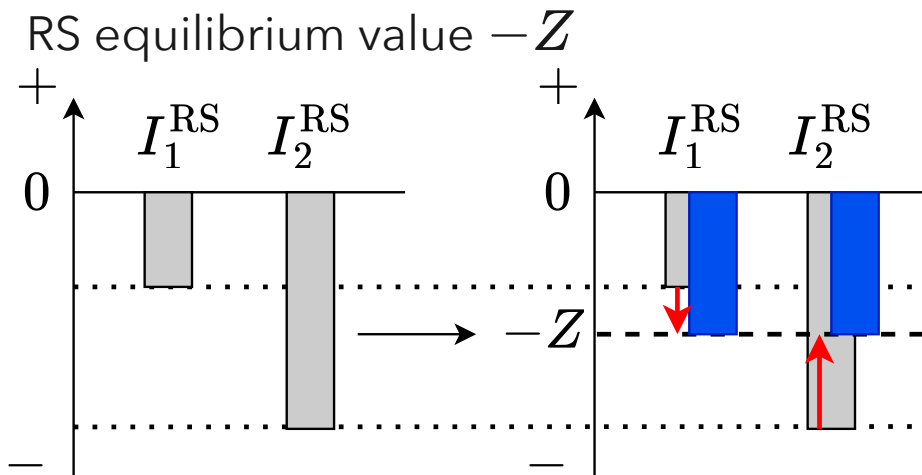
Under-archieved situation

- The higher the reliability, the higher the I_i^{RS}.

- The agent makes an optimistic action selection.

Over-archieved

I_i^{RS}

0

$\dfrac{n_i}{N}$
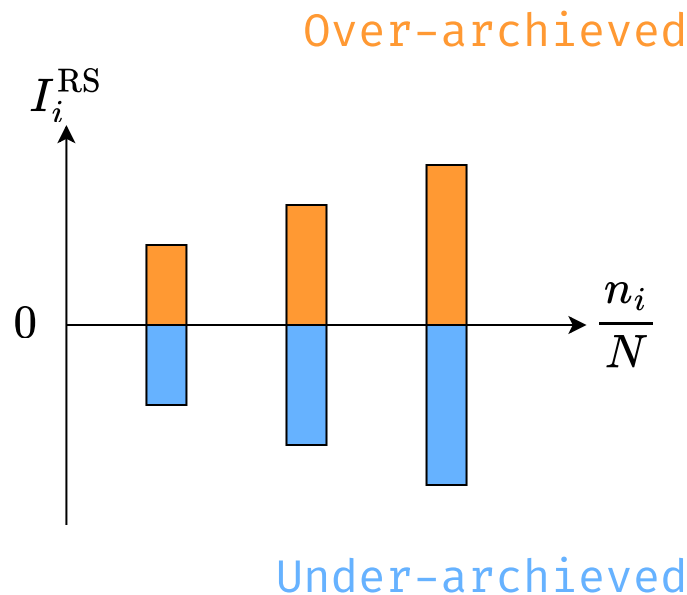
Under-archieved

→ In the over-archieved situation, the agent does not necessarily
select the action with the maximum reward expectation value.

Inverse calculation of the choice distribution of RS

- We can estimate the internal choice ratio of RS in the under-archieved situation.
- The agent generates a probability distribution from the difference between the estimated reliability and the actual reliability.
  - The agent generates the estimated reliability ρ_i^z using the RS equilibrium value $-Z$.
- **Stochastic RS** (SRS)

RS equilibrium value $-Z$

$$I_i^{\mathrm{RS}} = -Z$$

$$\rho_i = n_i/N = Z/(\aleph - p_i)$$

$$\sum_{i=1}^{K} \rho_i = \sum_{i=1}^{K} Z/(\aleph - p_i) = 1$$

$$Z = 1/\sum_{i=1}^{K} \frac{1}{\aleph - p_i}$$

How to calculate the policy of SRS

Under-archieved situation

- We can calculate $-Z$.

$$b_i = \frac{n_i}{\rho_i^z} - N + \epsilon$$

$$I_i^{\mathrm{SRS}} = (N + \max_i(b_i))\rho_i^z - n_i > 0$$

$$\pi_i = I_i^{\mathrm{SRS}} / \sum_{i=1}^{K} I^{\mathrm{SRS}_i}$$

- b: Adjustment parameter for preventing negative ratios
- ϵ: An extremely small value to prevent zero division

Over-achieved situation

- We cannot calculate $-Z$.
- Instead, we calculate the probability of choosing an action that has a reward expectation value greater than $\aleph$.

$$I_i^{\mathrm{RS}'} = \begin{cases} I_i^{\mathrm{RS}} + \epsilon, & \text{if } p_i \geq \aleph \\ 0, & \text{if } p_i < \aleph \end{cases}$$

$$\pi_i = I_i^{\mathrm{RS}'} / \sum_{j=1}^{K} I_j^{\mathrm{RS}'}$$

Regional Linear SRS

- **Our new method**
- With this method, the reliability estimation part of RegLinRS is combined with SRS.
  - The formula of SRS contains n_i and N.
  - We can approximate n_i/N, but we cannot approximate n_i and N.

Transformation of the equations in SRS

$$\frac{b_i}{N_x} = \frac{1}{\rho_i^z} \cdot \frac{n_i}{N_x} - 1 + \epsilon = \frac{\rho_i}{\rho_i^z} - 1 + \epsilon \qquad \frac{I_i^{\text{SRS}}}{N_x} = \left\{ \max\left(\frac{\hat{\phi}_i}{\rho_i^z}\right) + \epsilon \right\} \rho_i^z - \hat{\phi}_i$$

$$\frac{I_i^{\text{SRS}}}{N_x} = \left(\frac{N_x + \max_i(b_i)}{N_x} \right) \qquad\qquad \pi_i = \frac{I_i^{\text{SRS}}}{N_x} \bigg/ \sum_{j=1}^{K} \frac{I_j^{\text{SRS}}}{N_x} = I_i^{\text{SRS}} \bigg/ \sum_{j=1}^{K} I_j^{\text{SRS}}$$

$$= \left\{ \max_i\left(\frac{\rho_i}{\rho_i^z}\right) + \epsilon \right\} \rho_i^z - \rho_i$$

$\rightarrow$ We can extend SRS to RegLinSRS.

Settings for Experiment 1

Value Name	Value
ϵ	sys.float_info.epsilon in Python
episodic memory size	10,000
k of K-nearest neighbors	50
$\aleph$	0.6
α of LinUCB	0.1
λ of LinTS	0.25
α of LinTS	6

Settings for Experiment 1

Value Name	Value
β of LinTS	6
$\mathbf{b}_i$	All 0
$\mathbf{A}_i$	Identity matrix $\mathbf{I}$

Noise settings for Experiment 2

Adding noise to the estimated reward expectation value

- We added noise to the estimated reward expectation value at equal intervals after 80,000
  steps as follows.
  - This noise reverses the order of the estimated reward expectation values.
- We set γ to a constant of 0.01.
- We made sure that the noise was not added to the reward expectation value that is returned
  to the agent as a reward.

$$p'_{\min} \leftarrow p_{\min} + (p_{\max} - p_{\min}) + \gamma$$