Final Presentation @ Frontiers in Artificial Intelligence
Jan 11th, 2022

Job-Scheduling by RL

Dept. of Computer Science / Sugiyama-Yokoya-Ishida Lab.

Toshiki Kodera

Problem: Job-Scheduling

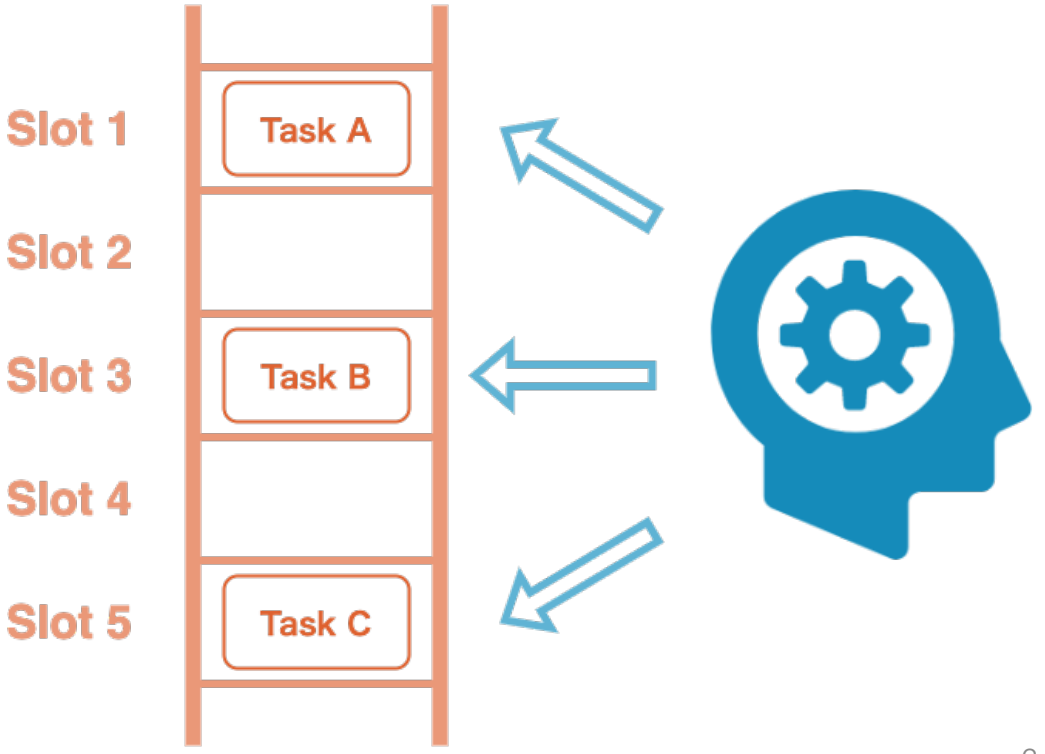
Goal: Design strategy that **maximizes task throughput**

Slot & Task

- Each task has these properties:
  - Required effort (to get done)
  - Remaining time (by deadline)
- Each task is assigned to one slot
- Each slot can store only one task

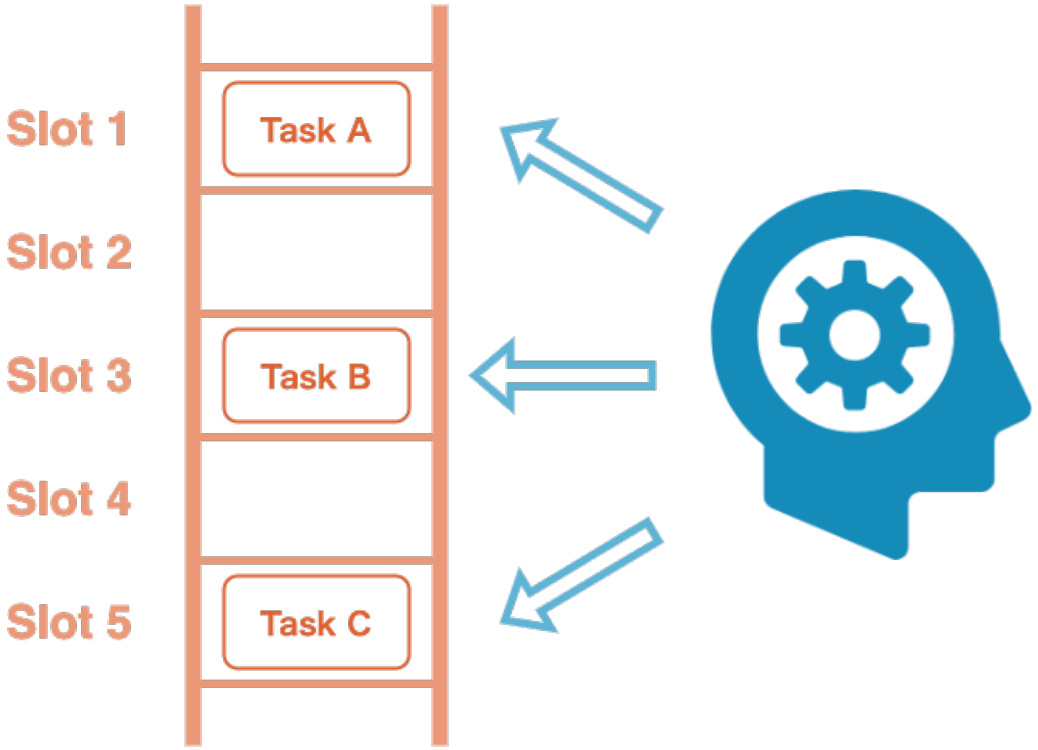
Worker

- Worker chooses *slots* to deal with



Solution: RL

Use **reinforcement learning (RL)** to solve job-scheduling problem!



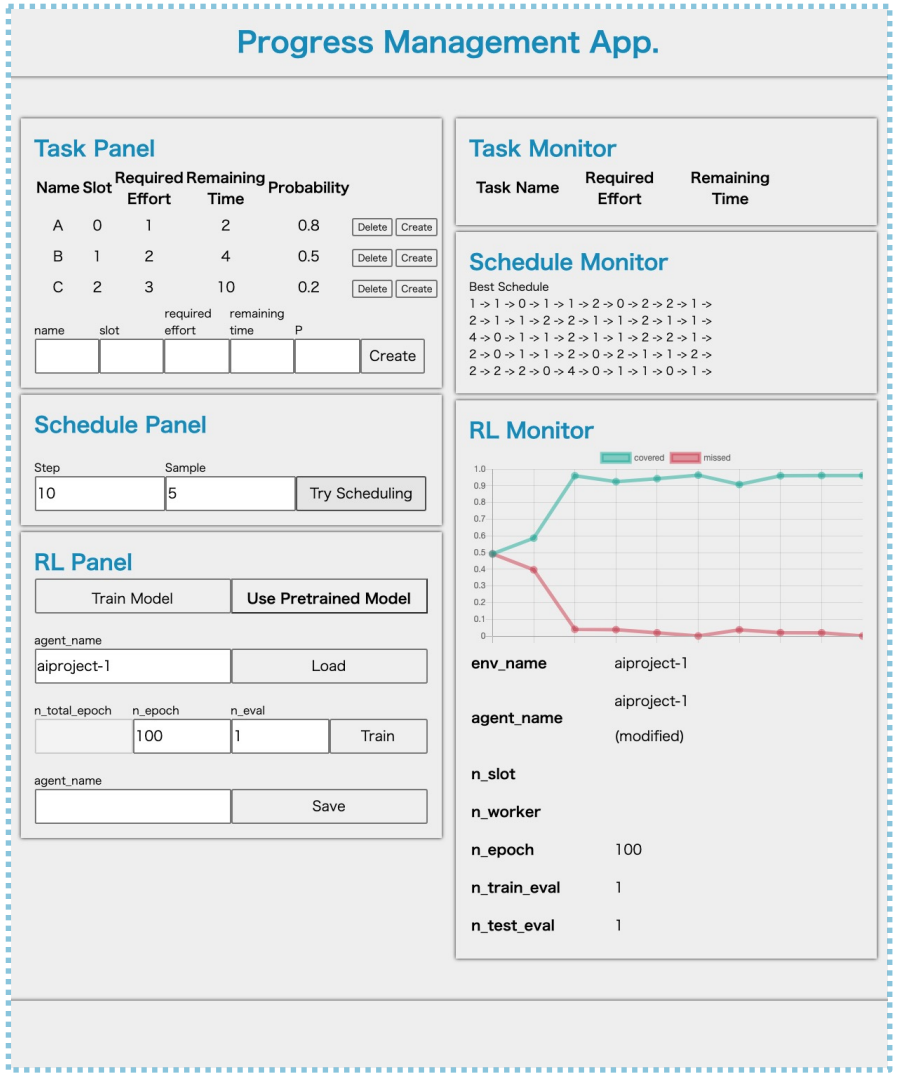
Reinforcement Learning

- Agent tries to find the best strategy which maximizes cumulative reward (= # of completed tasks)

Agent

- Job-scheduler uses **Deep Q-Network (DQN)** algorithm

Job-Scheduling App



Components

- Task Panel & Monitor
  - Create/delete tasks
- Schedule Panel & Monitor
  - Try job-scheduling
- RL Panel & Monitor
  - Train/load/save RL models

Job-Scheduling App

1. INIT → 2. TRAIN → 3. USE

Environment

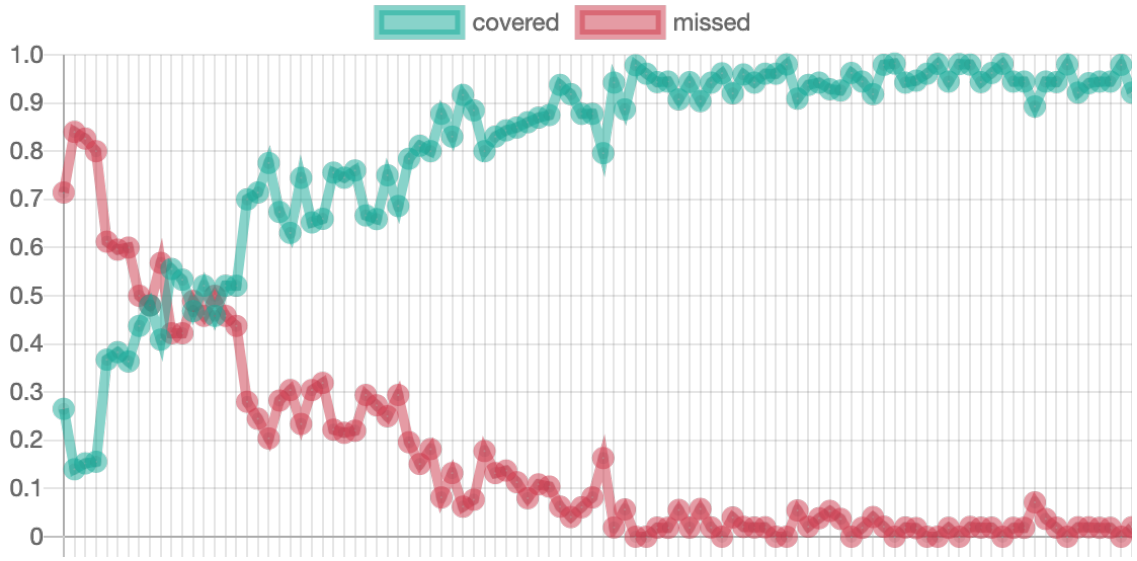
Task	Slot	Duration	Due	Prob.
A	0	1	2	0.8
B	1	2	4	0.5
C	2	3	10	0.2

Agent

Epoch	# of Evaluation
1000	1

Best
1 → 2 → …

Example (1)



Covered: Ratio of completed tasks

Missed: Ratio of unfinished (& overdue) tasks

Meta-data

# of slot	# of worker
5	1

Environment

Task	Slot	Duration	Due	Prob.
A	0	1	2	0.8
B	1	2	4	0.5
C	2	3	10	0.2

Agent

Epoch	# of Evaluation
1000	1

Example (2)

Covered: Ratio of completed tasks

Missed: Ratio of unfinished (& overdue) tasks

Meta-data

# of slot	# of worker
5	2

Environment

Task	Slot	Duration	Due	Prob.
A	0	2	3	0.8
B	1	3	4	0.8
C	2	4	5	0.8

Agent

Epoch	# of Evaluation
1000	1

References

Repository:

- **Job-Scheduling App.**
  https://github.com/Sharkkii/Aiproject
- **Reinforcement Learning framework**
  https://github.com/Sharkkii/shaRL