2023 INTERNATIONAL CONFERENCE
ON SYSTEM SCIENCE AND ENGINEERING

July 27-28, 2023 - Ho Chi Minh City, Vietnam

HYBRID EVENT

IRLS: An Improved Reinforcement Learning Scheduler for High Performance Computing Systems

Thanh Hoang Le Hai,
Luan Le Dinh, Dat Ngo Tien, Dat Bui Huu Tien and Nam Thoai

High Performance Computing Laboratory, Faculty of Computer Science & Engineering
Advanced Institute of Interdisciplinary Science and Technology
Ho Chi Minh City University of Technology (HCMUT)
Vietnam National University Ho Chi Minh City

Outline

- **Introduction**
  - Scheduling on HPC Systems
  - Deep Reinforcement Learning Schedulers
  - Research Goals
- **IRLS: An Improved Reinforcement Learning Scheduler**
  - IRLS Environment Representation
  - Scheduling Agent Rewards
- **Evaluation**
- **Conclusion and Future Work**

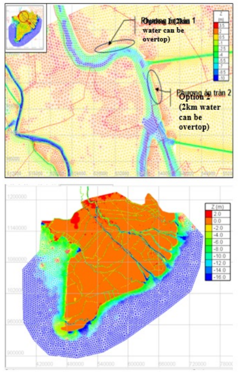
Scheduling on HPC Systems

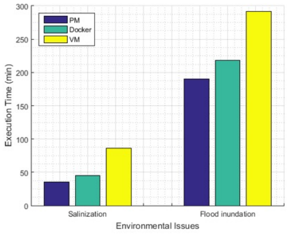
- Today, **<u>High-Performance Computing</u>** (HPC) systems are the key factor to many scientific discoveries.

(nighttime without infrared)

(nighttime with infrared)

WTA 2018

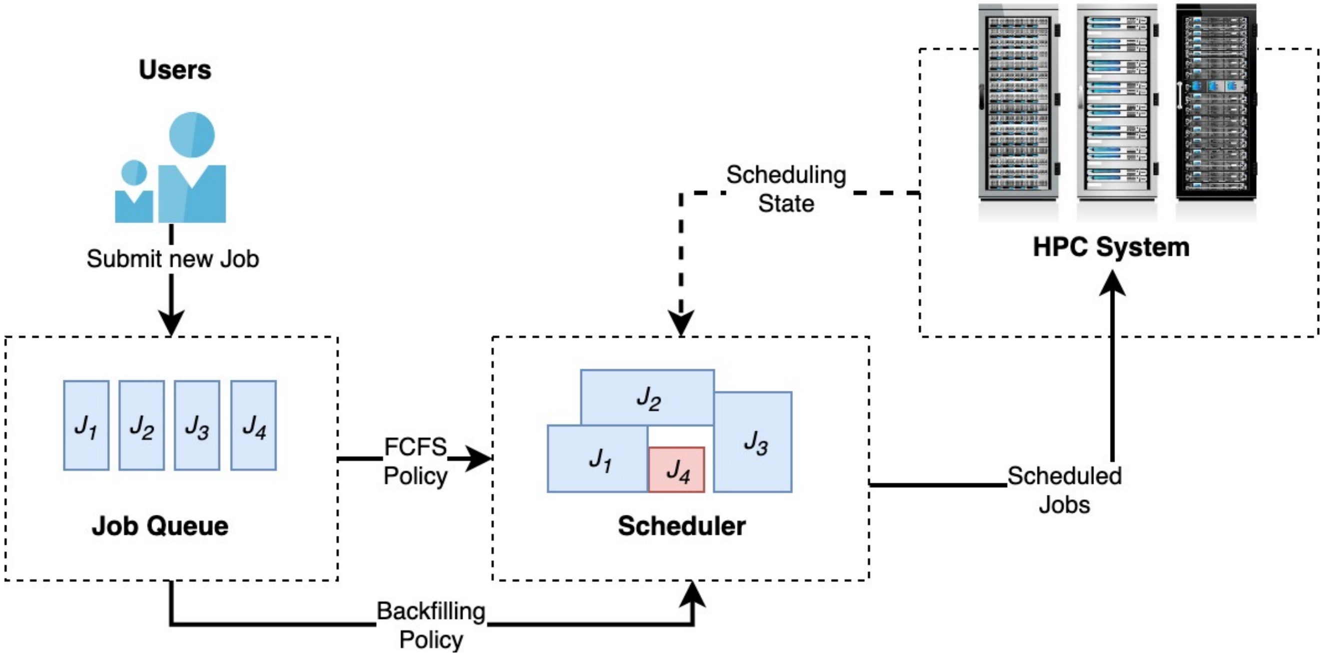




HPC Lab

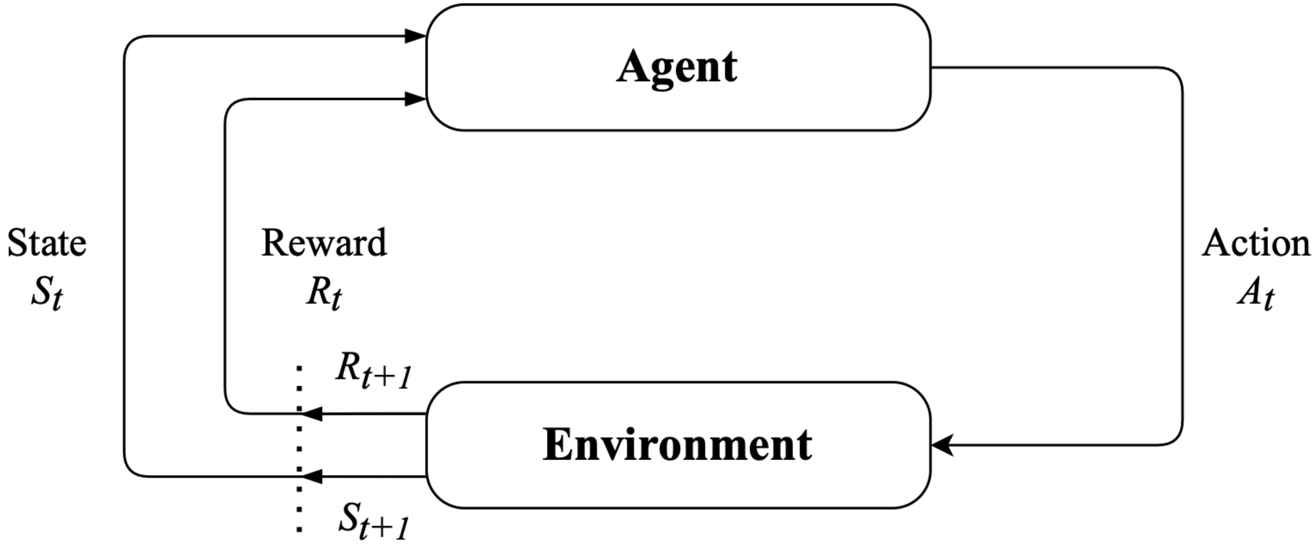
Scheduling on HPC Systems

- **Job Schedulers** help ensure fair access to computing resources while maintaining optimal system utilization.
- HPC systems often use multi-level FCFS queues with Backfilling.



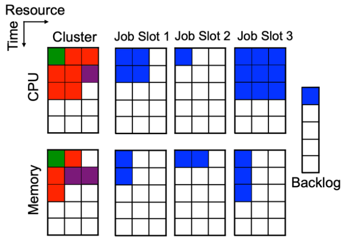
DRL Schedulers

Deep Reinforcement Learning (DRL) is a type of machine learning where an agent learns to make decisions by interacting with an environment and improves by receiving rewards.



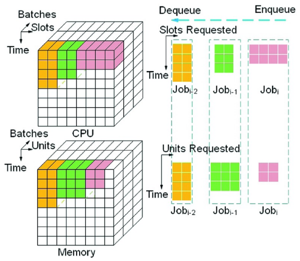
DRL Schedulers

Many **DRL Schedulers** were introduced to deal with the complexity of HPC software, hardware, and multidisciplinary research.



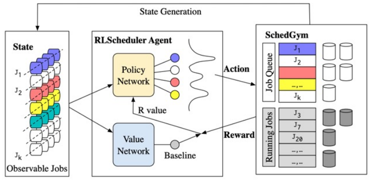
DeepRM
(2016)

Hongzi Mao, Mohammad Alizadeh, Ishai Menache, and Srikanth Kandula. 2016. Resource Management with Deep Reinforcement Learning. In Proceedings of the 15th ACM Workshop on Hot Topics in Networks (HotNets '16). Association for Computing Machinery, New York, NY, USA, 50–56. https://doi.org/10.1145/3005745.3005750



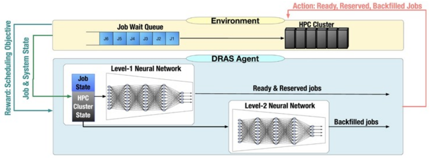
A2cScheduler
(2020)

Liang S, Yang Z, Jin F, Chen Y. Data Centers Job Scheduling with Deep Reinforcement Learning. Advances in Knowledge Discovery and Data Mining. 2020 Apr 17;12085:906–17. doi: 10.1007/978-3-030-47436-2_68. PMCID: PMC7206316.



RLScheduler
(2020)

Di Zhang, Dong Dai, Youbiao He, Forrest Sheng Bao, and Bing Xie. 2020. RLScheduler: an automated HPC batch job scheduler using reinforcement learning. In Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (SC '20). IEEE Press, Article 31, 1–15.

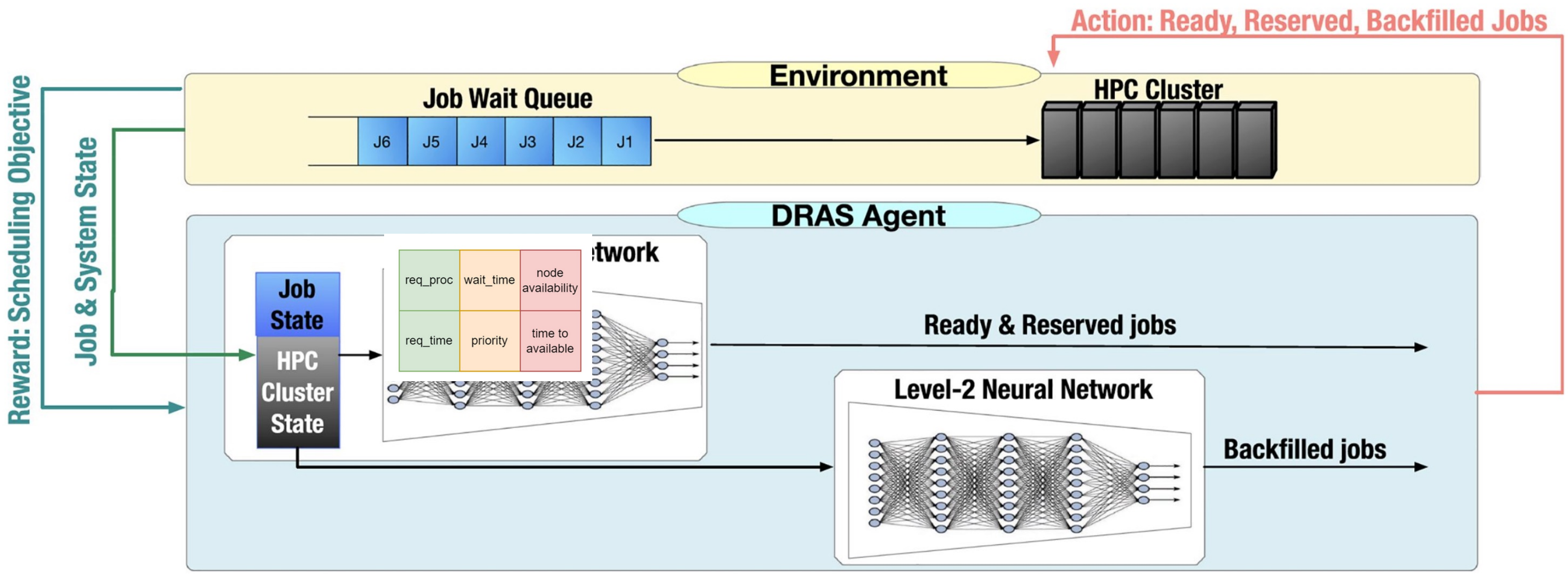


DRAS
(2022)

Y. Fan et al., "DRAS: Deep Reinforcement Learning for Cluster Scheduling in High Performance Computing," in IEEE Transactions on Parallel and Distributed Systems, vol. 33, no. 12, pp. 4903-4917, 1 Dec. 2022, doi: 10.1109/TPDS.2022.3205325.

DRL Schedulers

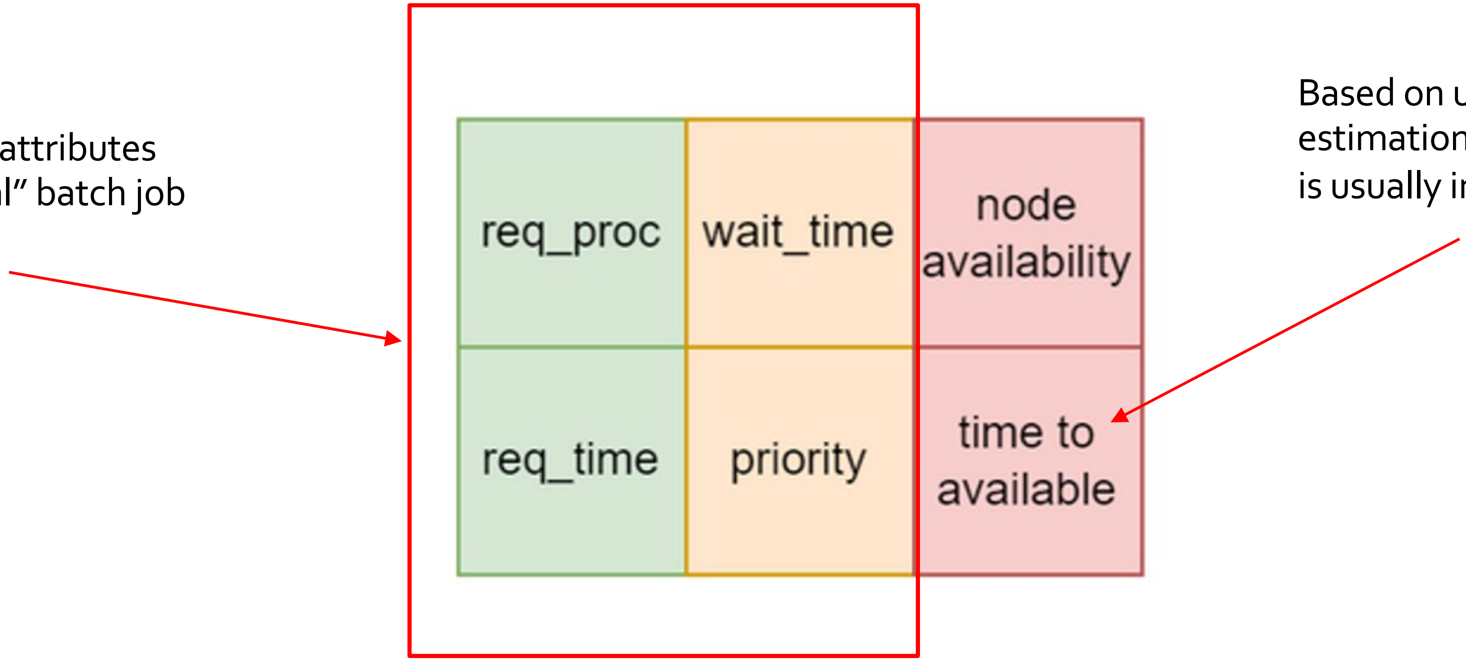
- **DRAS** (**Deep Reinforcement Agent for Scheduling**) is the latest HPC scheduling model based on DRL and a hierarchical neural network.
- DRAS performs resource reservation and backfilling and efficiently learns and adapts its policy in response to workload changes.



Y. Fan, et al.,"DRAS: Deep Reinforcement Learning for Cluster Scheduling in High Performance Computing" in IEEE Transactions on Parallel & Distributed Systems, vol. 33, no. 12, pp. 4903-4917, 2022. doi: 10.1109/TPDS.2022.3205325

DRL Schedulers

However, the HPC state representation of DRAS is simple and does not reflect the behavior in real HPC systems.



req_proc	wait_time	node availability
req_time	priority	time to available

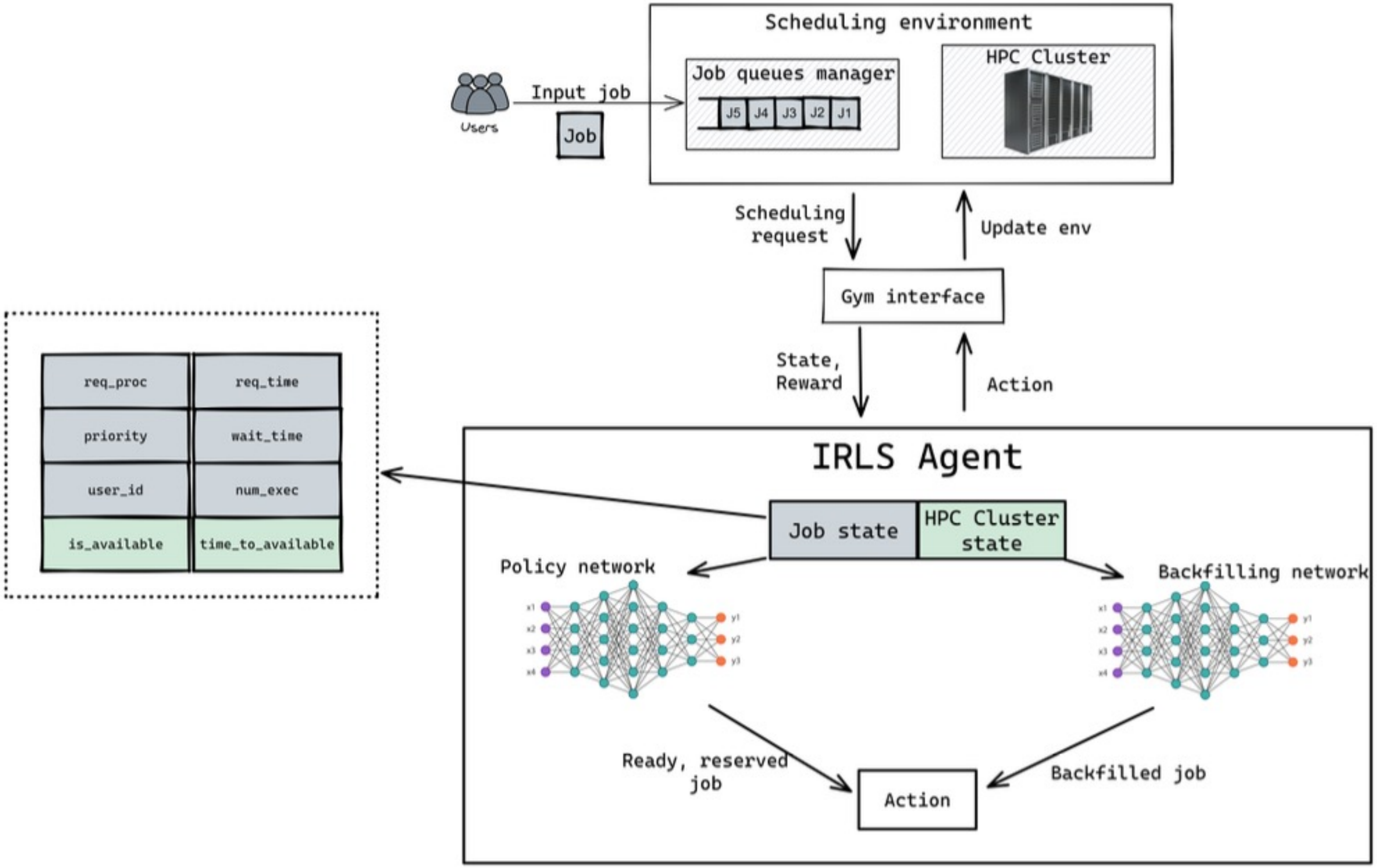
Two few attributes for a "real" batch job

Based on user estimations which is usually incorrect

Research Goals

- Improve the HPC state representation of the DRL model with applicable attributes.
- Assist the Scheduling Agent with more accurate information from the production environment.
- Prove the possibility of the prosed solution with date from our SuperNode-XP HPC System in HCMUT, VNU-HCMC.

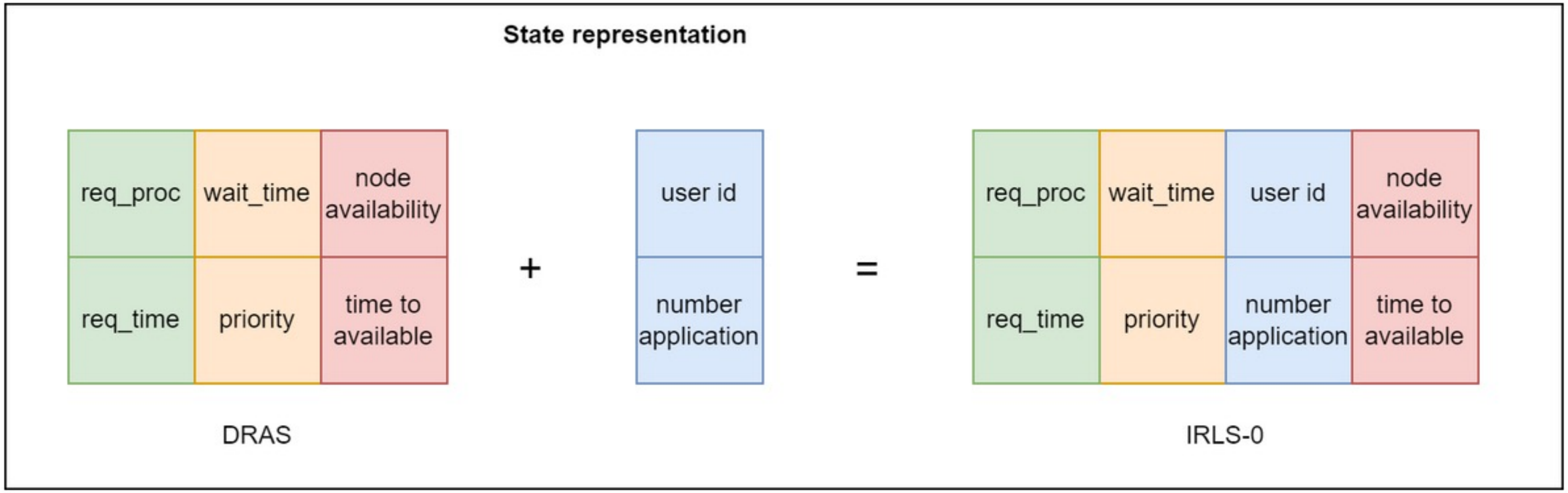
IRLS Overview



req_proc	req_time
priority	wait_time
user_id	num_exec
is_available	time_to_available

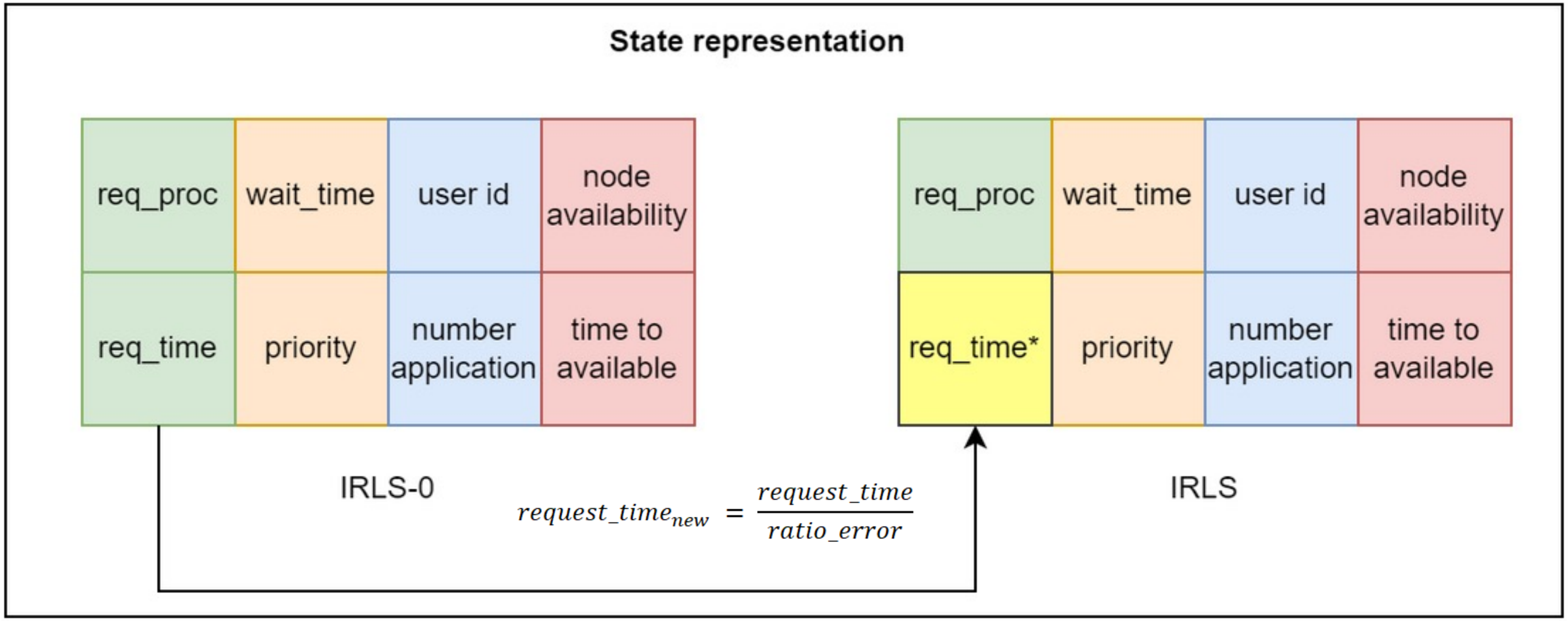
IRLS Environment Representation

Based on DRAS, we improve the state by adding two important factors: **user ID** and **application ID,** to help the agent learn from similar submitted jobs.



IRLS Environment Representation

- We also “**calibrate**” the user-provided **req_time** value based on their estimation history



State representation

req_proc	wait_time	user id	node availability
req_time	priority	number application	time to available

IRLS-0

req_proc	wait_time	user id	node availability
req_time*	priority	number application	time to available

IRLS

$$request\_time_{new} = \frac{request\_time}{ratio\_error}$$

where $$ratio\_error = \sum_{past} \frac{request\_time}{runtime}$$

Agent Scheduling Reward

- The reward function to be **maximized**:
  - Avoid the problem of job starvation
  - Focus on large-scale jobs
  - Using as many nodes of the system as possible to improve utilization.

Average wait time of finished jobs

Average number of requested nodes

Number of allocated nodes

weights based on system policy

$$w_1 * \frac{\bar{t}}{t_{max}} + w_2 * \frac{\bar{n}}{N} + w_3 * \frac{N_{used}}{N}$$

Max recorded wait time of finished jobs

Total number of nodes in the system

Evaluation

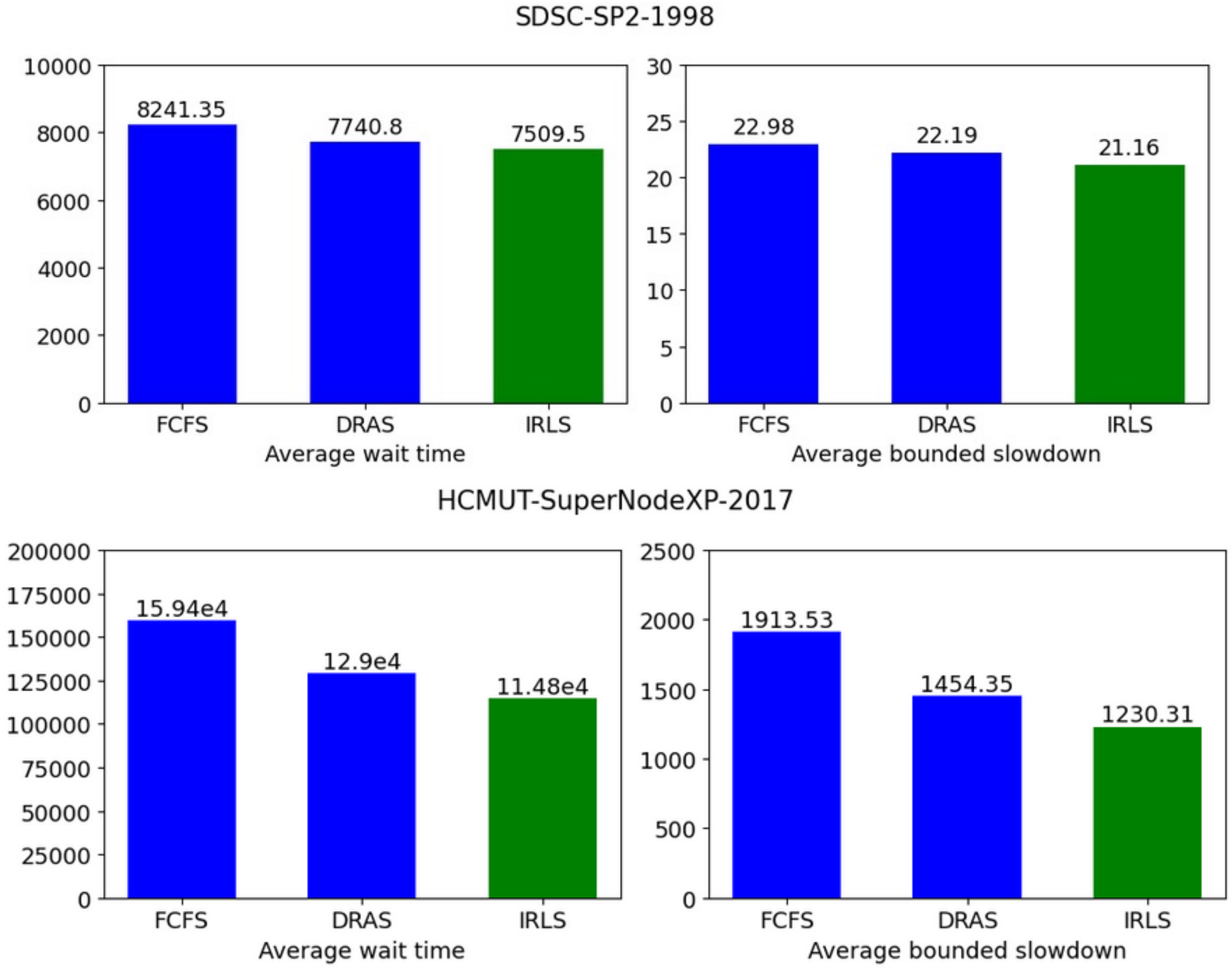
- Experiments uses CQSim[1] on Google Colab
- Use two datasets:.

Name	***SDSC-SP2-1998***	***HCMUT-SuperNodeXP***
P (proc)	128	288
N (job)	59715	15886
$\bar{i}$ (sec)	1055	6976
$\bar{r}$ (sec)	17394	139933508
$\bar{p}$ (proc)	14	48

[1] X. Yang, Z. Zhou, S. Wallace, Z. Lan, W. Tang, S. Coghlan, and M. Papka, "Integrating Dynamic Pricing of Electricity into Energy Aware Scheduling for HPC Systems", Proc. of SC'13, 2013.

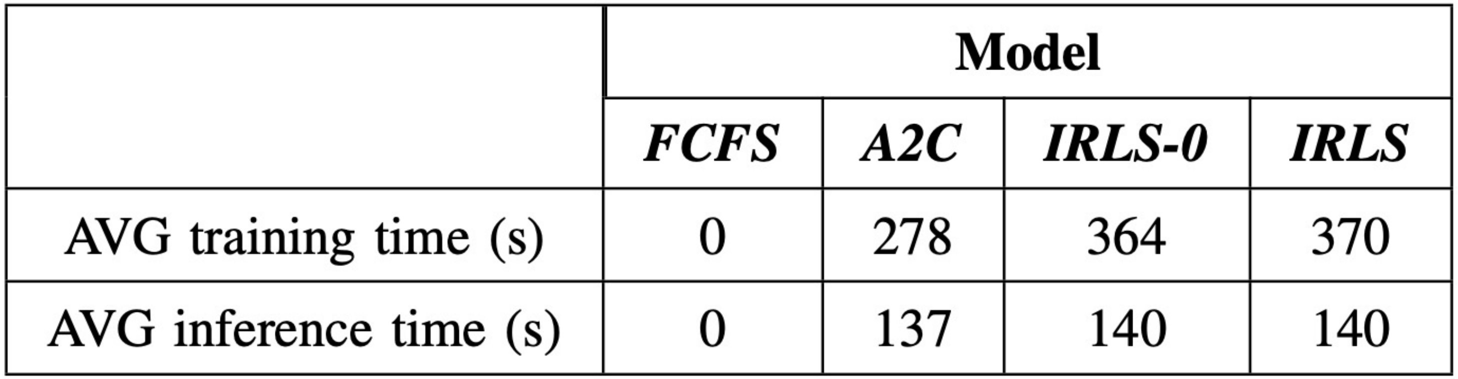
Evaluation

- IRLS has better scheduling performance



Evaluation

- IRLS still maintains an acceptable inference time



	Model			
	FCFS	***A2C***	***IRLS-0***	***IRLS***
AVG training time (s)	0	278	364	370
AVG inference time (s)	0	137	140	140

Conclusion and future work

- The proposed improvement effectively increases the model's performance without affecting the training and inference time, especially on our collected SuperNode-XP dataset.
- This is only the very, very first step:
    - Investigate the state representation for each particular HPC system with more constraints.
    - Extend experiments with more latest datasets from other HPC systems.

Thank you!

thanhhoang@hcmut.edu.vn

Thanh Hoang Le Hai,
Luan Le Dinh, Dat Ngo Tien, Dat Bui Huu Tien and Nam Thoai

High Performance Computing Laboratory, Faculty of Computer Science & Engineering
Advanced Institute of Interdisciplinary Science and Technology
Ho Chi Minh City University of Technology (HCMUT)
Vietnam National University Ho Chi Minh City