Increasing Large-Scale Data Center Capacity by Statistical Power Control

Guosai Wang, Shuhao Wang, Bing Luo, Weisong Shi, Yinghang Zhu,
Wenjun Yang, Dianming Hu, Longbo Huang, Xin Jin, Wei Xu

Data Centers

Expensive to build and operate

Building cost (large DCs): $9,000–$13,000/KW*

High power consumption: 10–20 MW

Goal: Fully utilize the capacity of data centers to reduce the TCO.

Our Result:

- +17% servers → +15% throughput
- Power violations effectively avoided.
- No performance disturbance to existing jobs.

[*LA Barroso, etc. The datacenter as a computer: An introduction to the design of warehouse-scale machines. 2013]

Underutilized Capacity in DCs

Observation: Avg power utilization < 72% at DC level

Reason: Conservative power provisioning

- Provision according with rated power
- Running power < Rated power

Underutilized Capacity in DCs

Observation: Avg power utilization < 72% at DC level

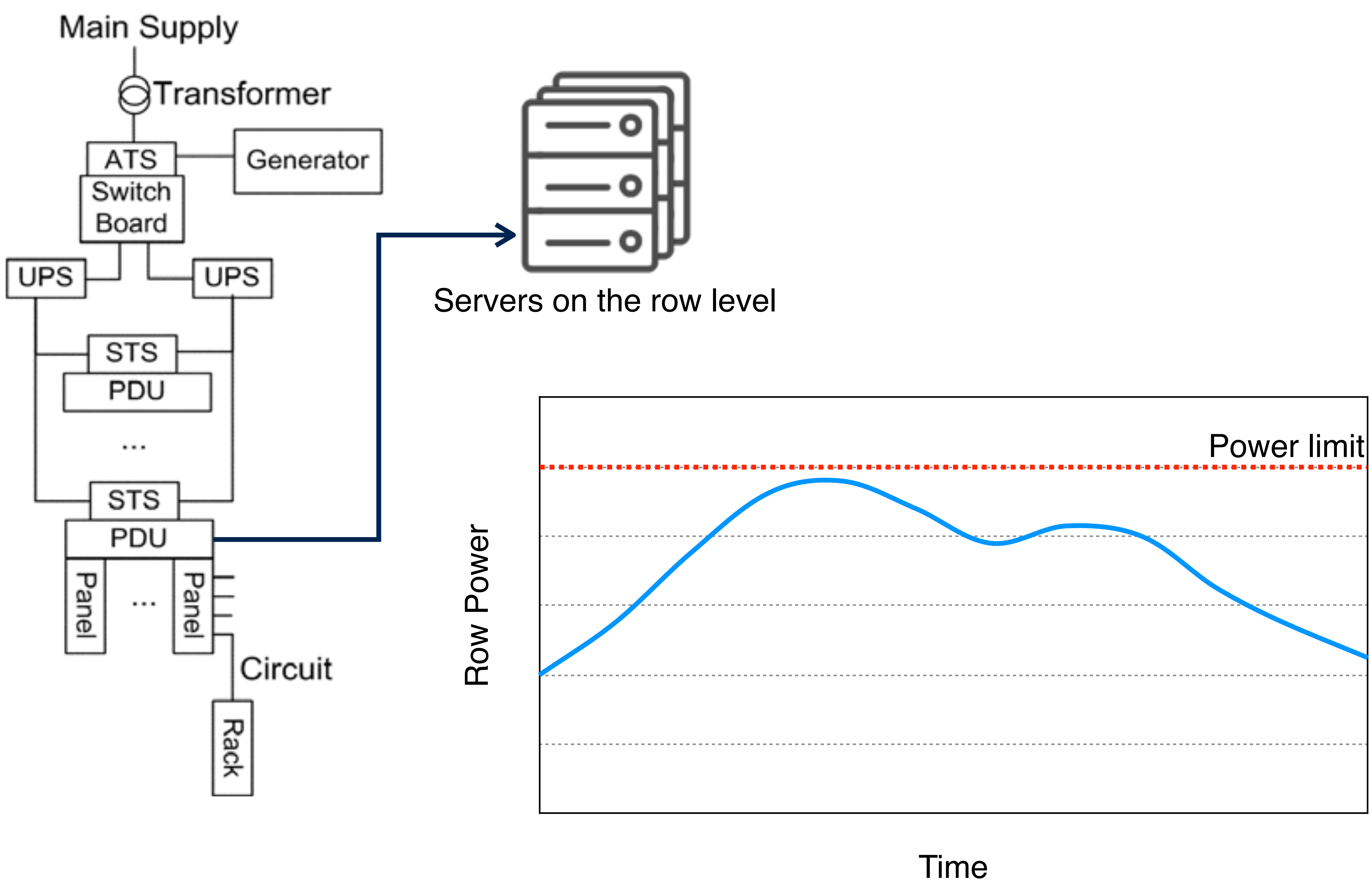
Reason: Conservative power provisioning

- Provision according with rated power
- Running power < Rated power

Over-provisioning of the facility power?

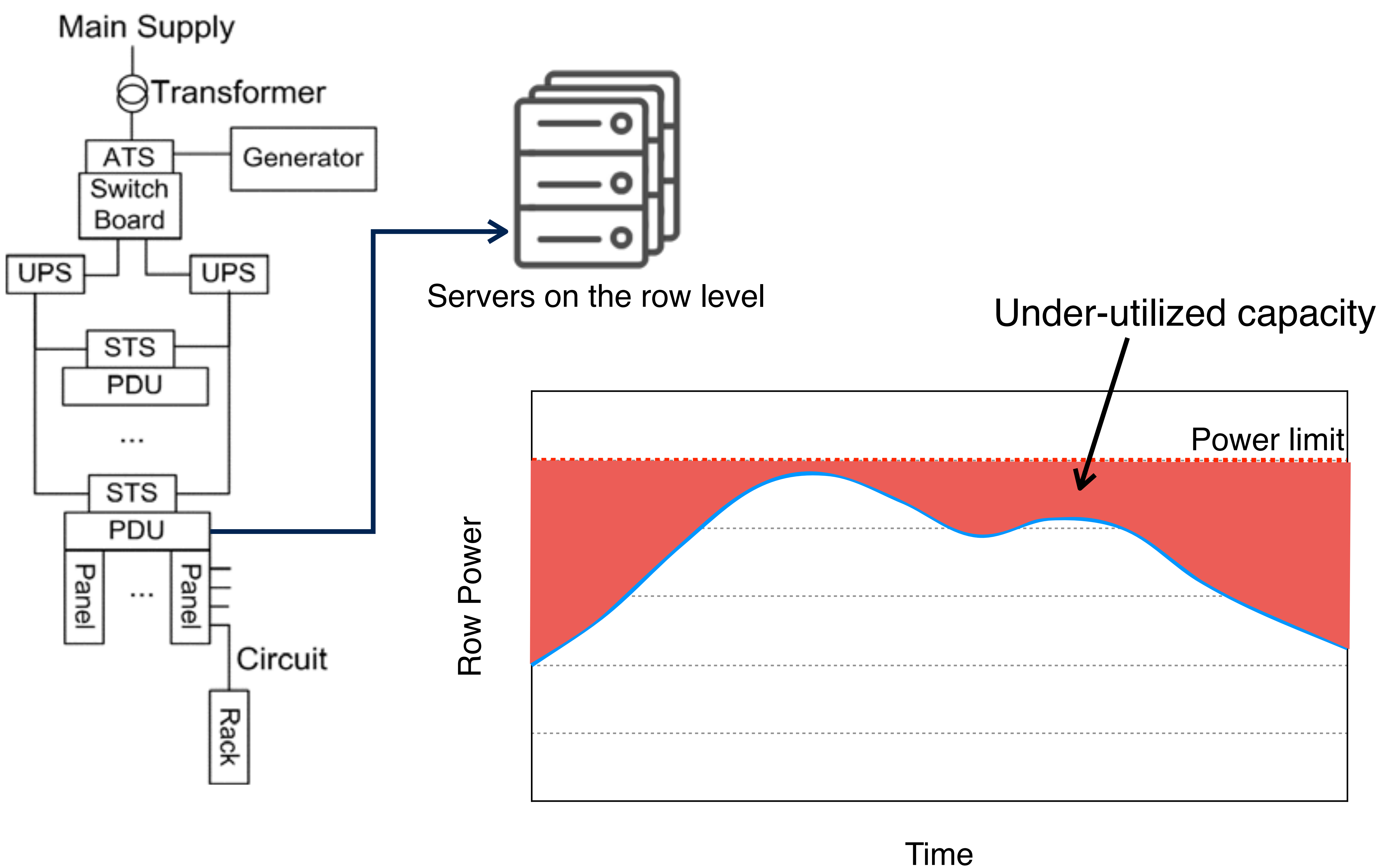
- Increase the number of servers on each rack.

Why People Under-provision?



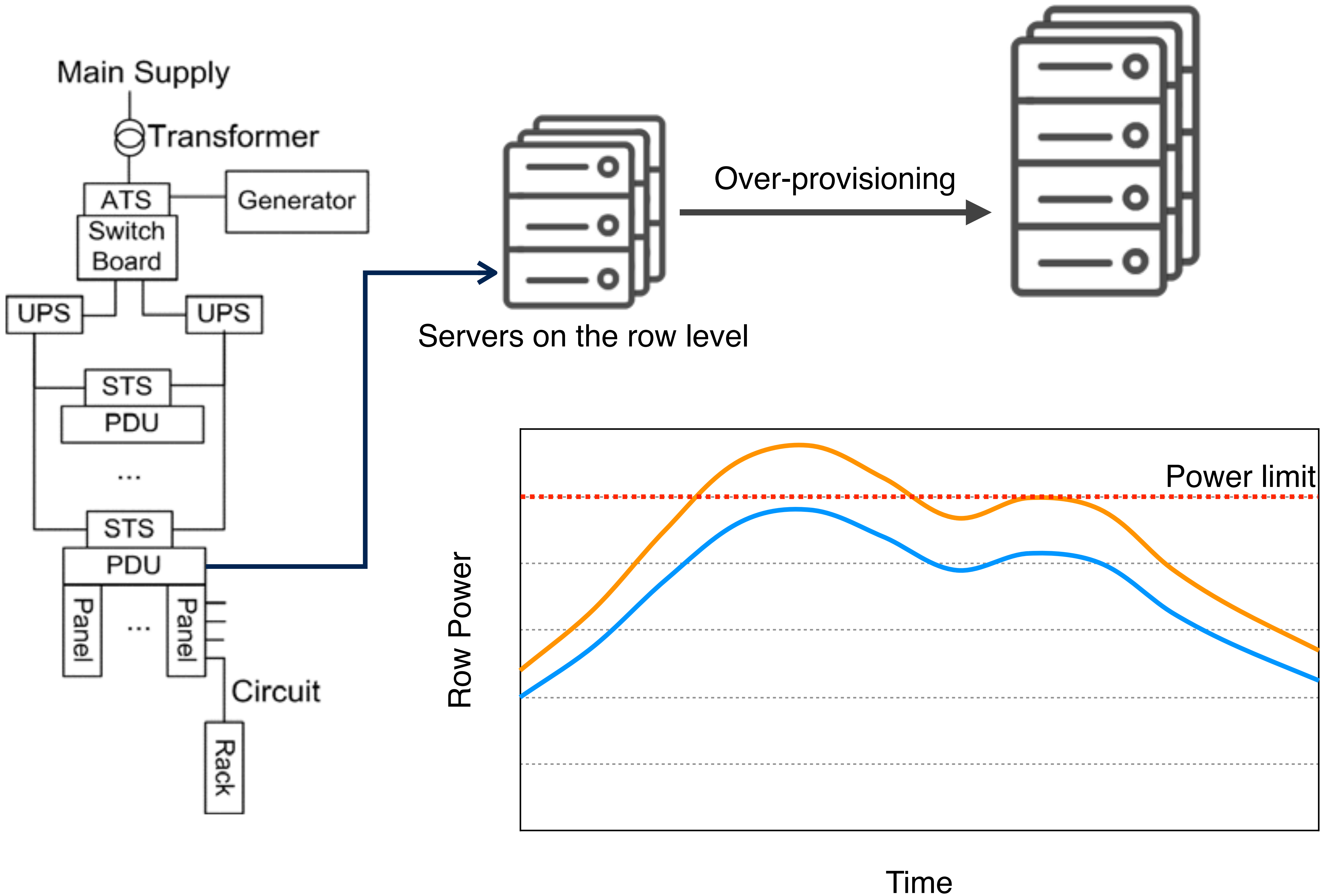
[Fan X, etc. Power provisioning for a warehouse-sized computer. ISCA 2007]

Why People Under-provision?



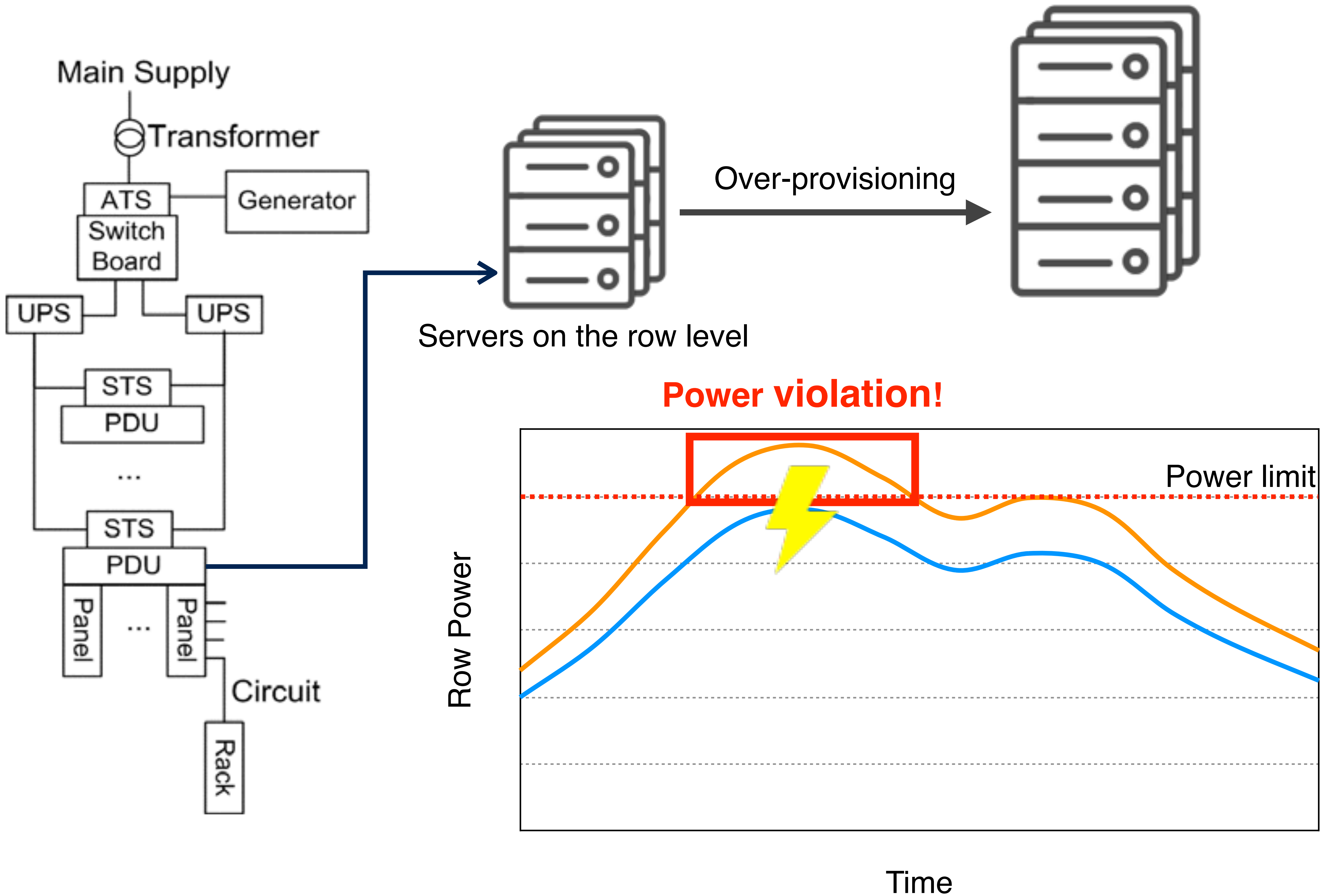
[Fan X, etc. Power provisioning for a warehouse-sized computer. ISCA 2007]

Why People Under-provision?



[Fan X, etc. Power provisioning for a warehouse-sized computer. ISCA 2007]

Why People Under-provision?

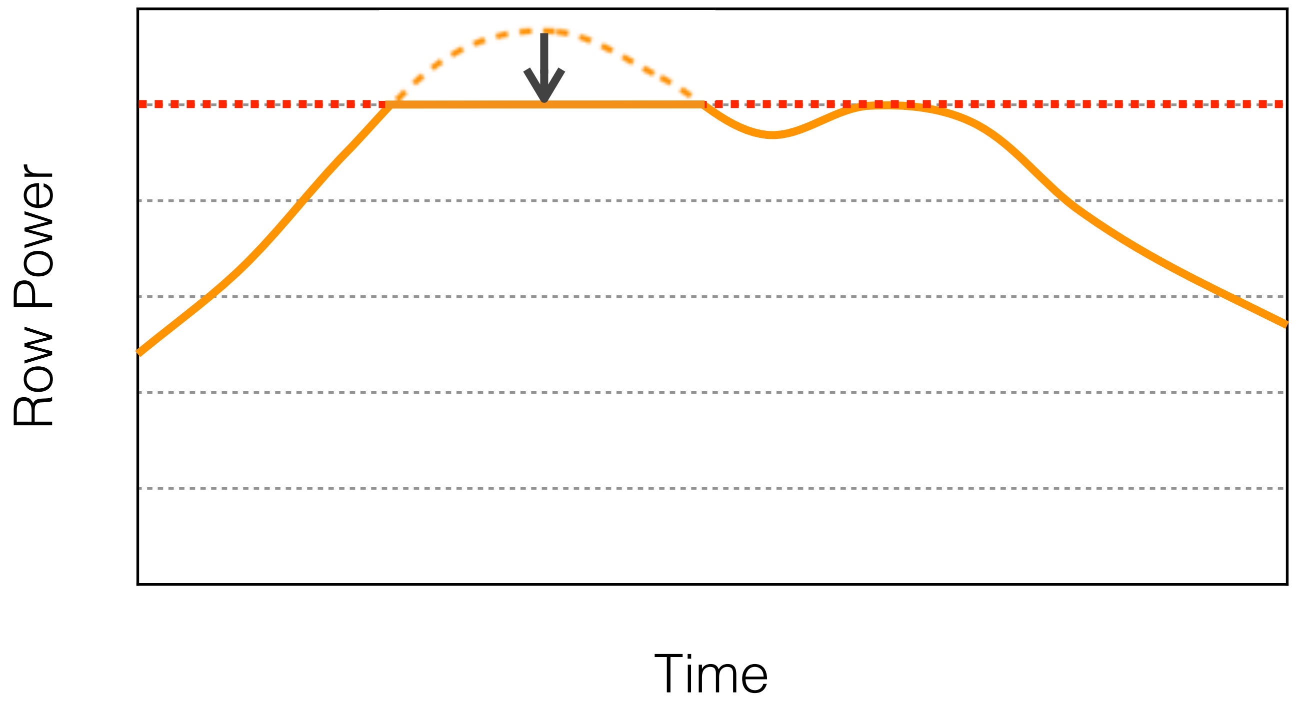


[Fan X, etc. Power provisioning for a warehouse-sized computer. ISCA 2007]

Power Capping Degrades Performance

Traditional approach: Power capping

- Dynamic Voltage and Frequent Scaling (DVFS)
- Power ≈ $C \cdot V^2 \cdot F$



Degrade the performance of running jobs!

- Violate the SLA of the latency-sensitive jobs.

Power Capping Degrades Performance

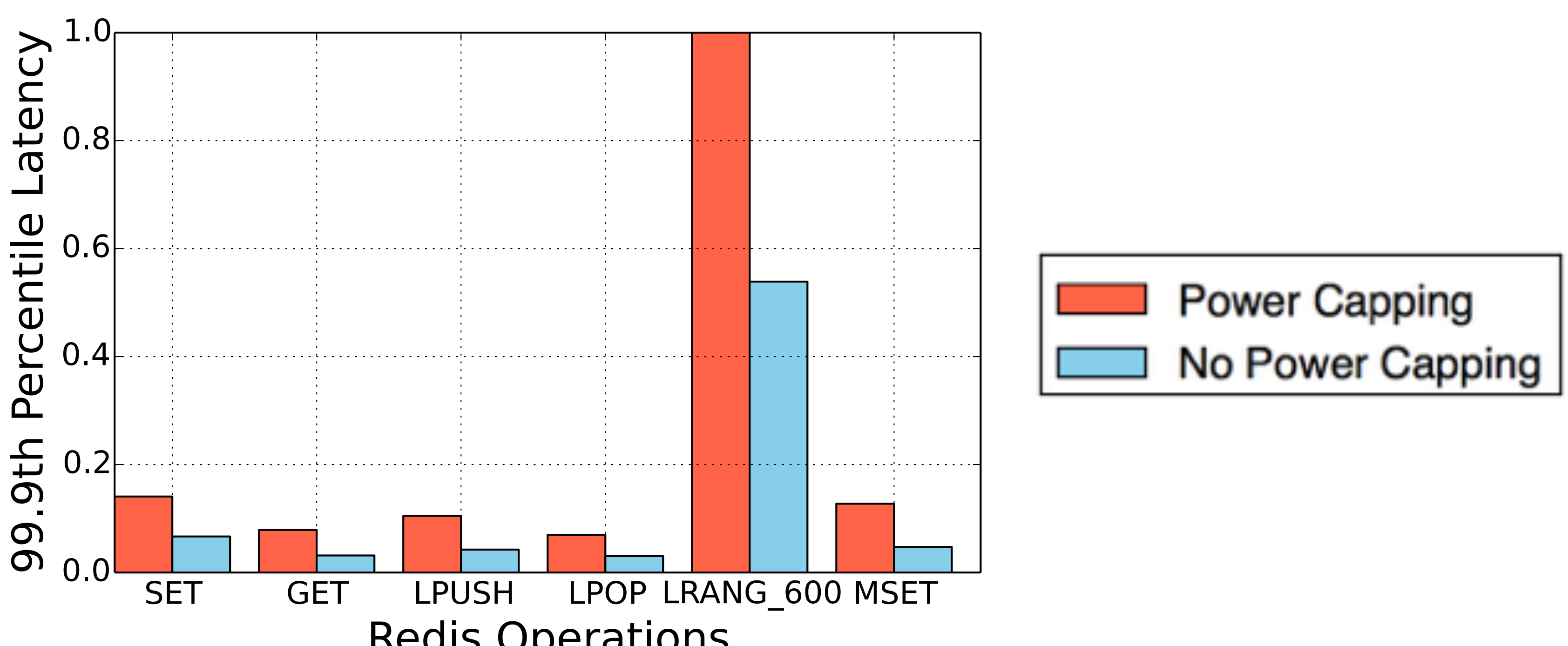
Traditional approach: Power capping

Dynamic Voltage and Frequent Scaling (DVFS)

Power ≈ $C \cdot V^2 \cdot F$

Degrade the performance of running jobs!

Violate the SLA of the latency-sensitive jobs.



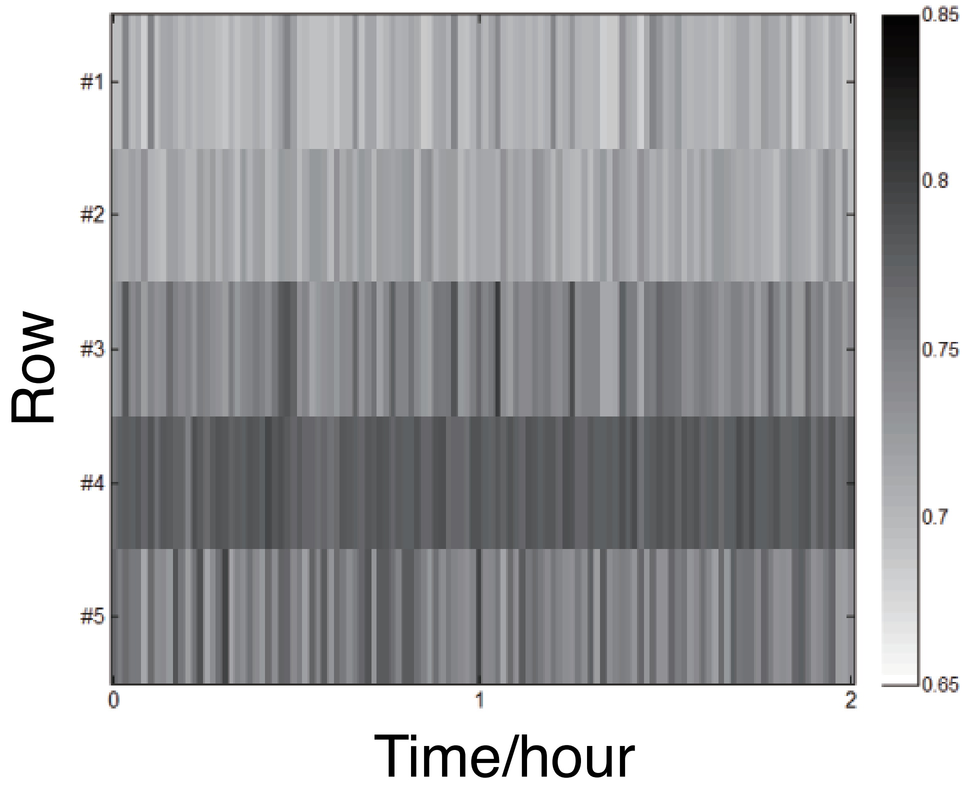
Power Control Method

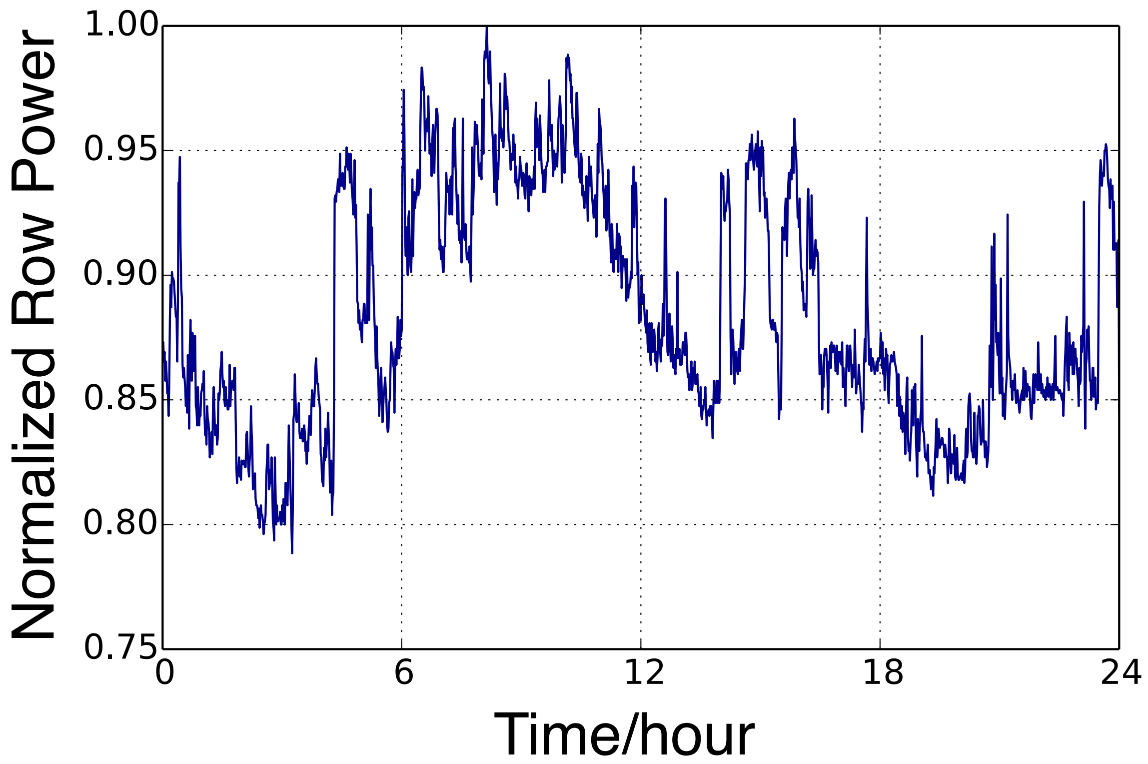
Can we control the power without affecting the performance of existing jobs?

Key Observation

Large variations on power utilization at row level

Temporal (over time) and spatial (across different rows).



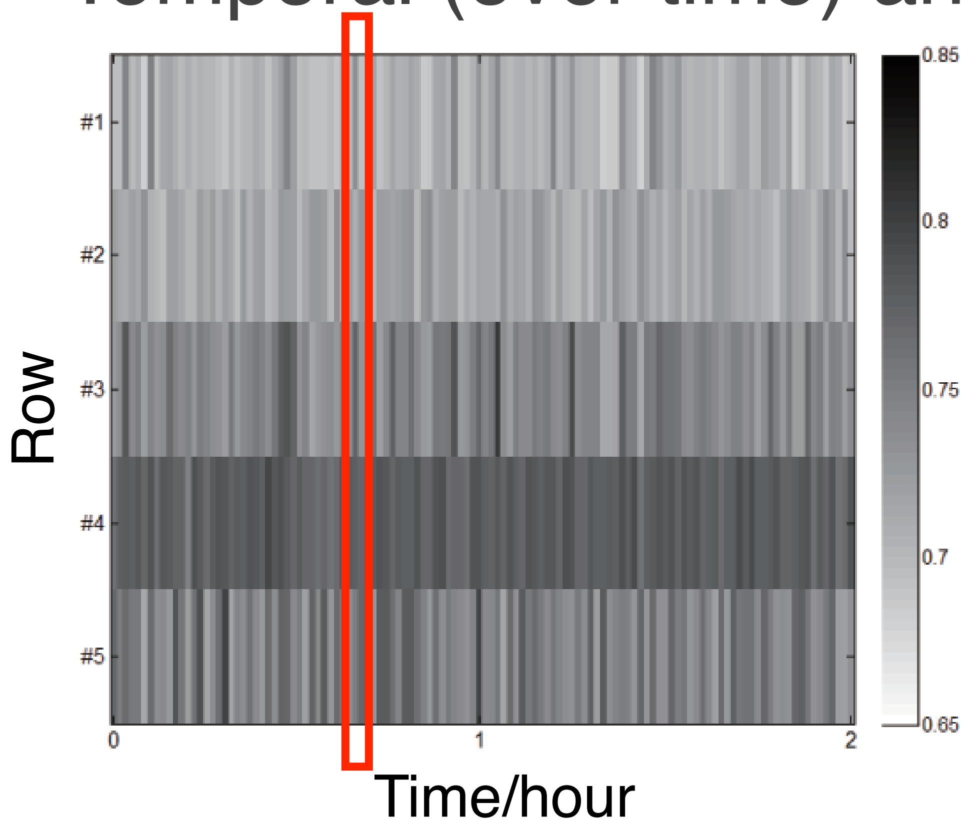


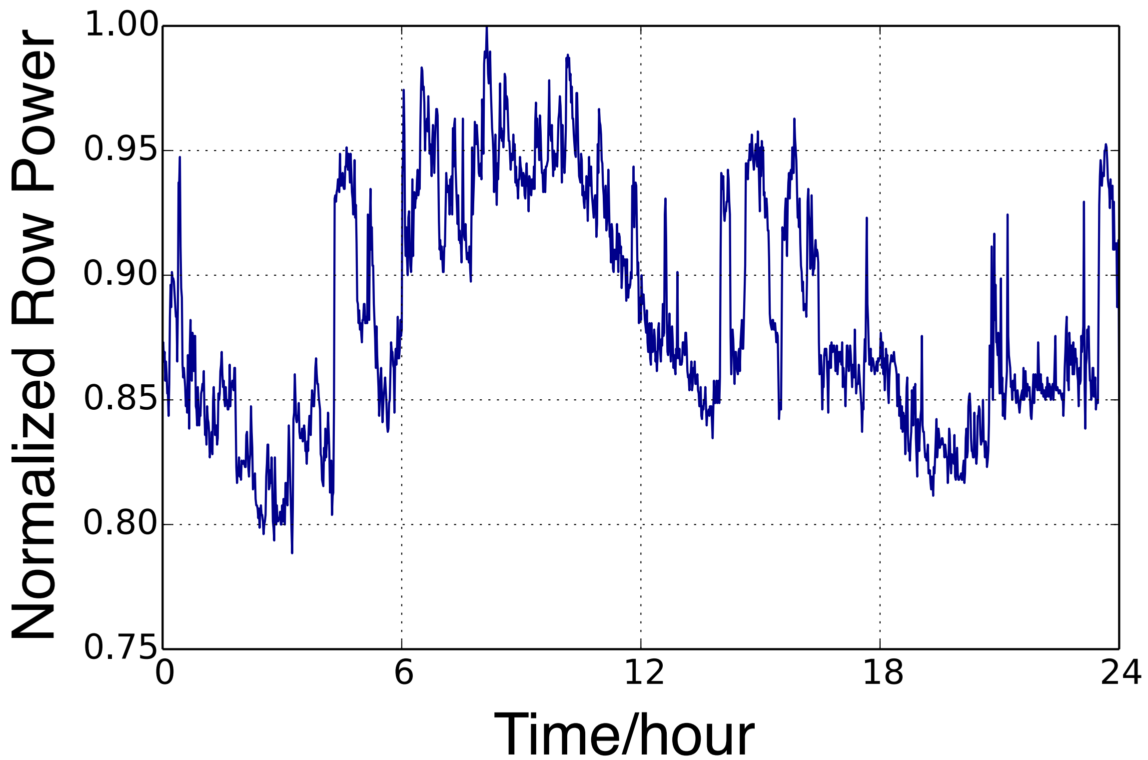
Idea: Dynamically move workload out of the heavily used rows.

Key Observation

Large variations on power utilization at row level

Temporal (over time) and spatial (across different rows).



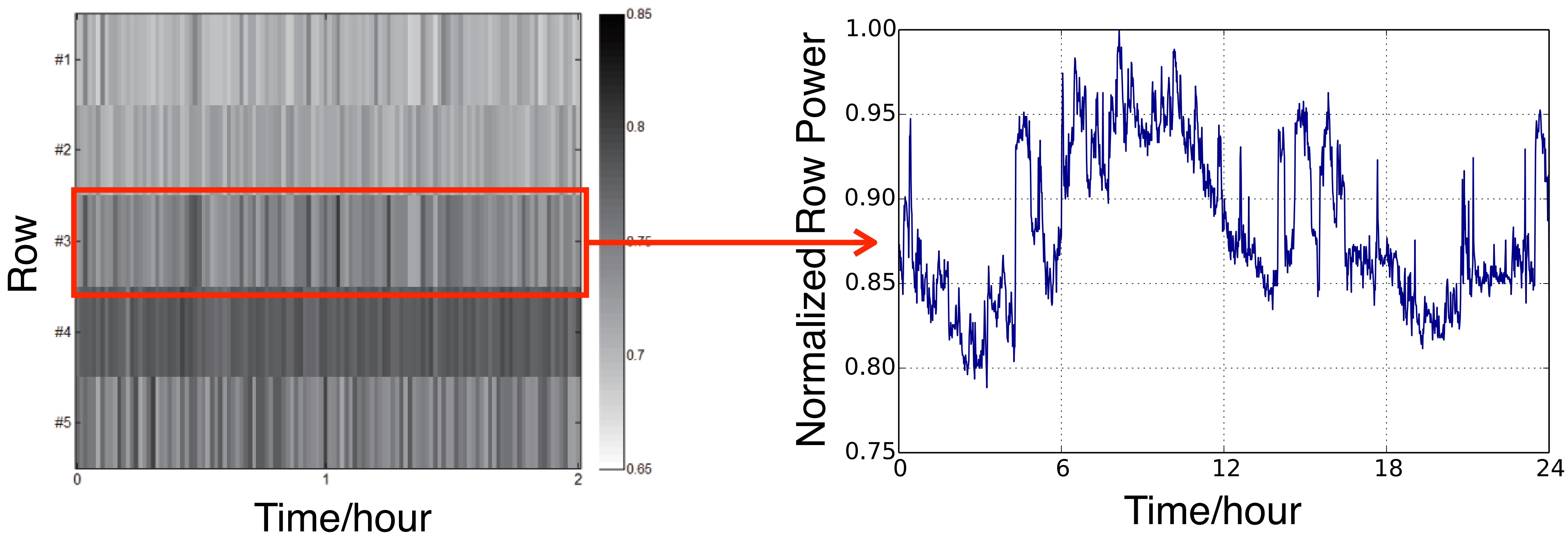


Idea: Dynamically move workload out of the heavily used rows.

Key Observation

Large variations on power utilization at row level

Temporal (over time) and spatial (across different rows).



Idea: Dynamically move workload out of the heavily used rows.

Our Solution: Statistical Power Control

- Minimize interface with the scheduler.

> Two simple APIs: Freeze/unfreeze.
>
> Decoupled with the over-complicated scheduler.

- Statistically influence new job placement.

> Indirect workload balancing.
>
> Running jobs unaffected.
>
> Does not necessarily work perfectly.

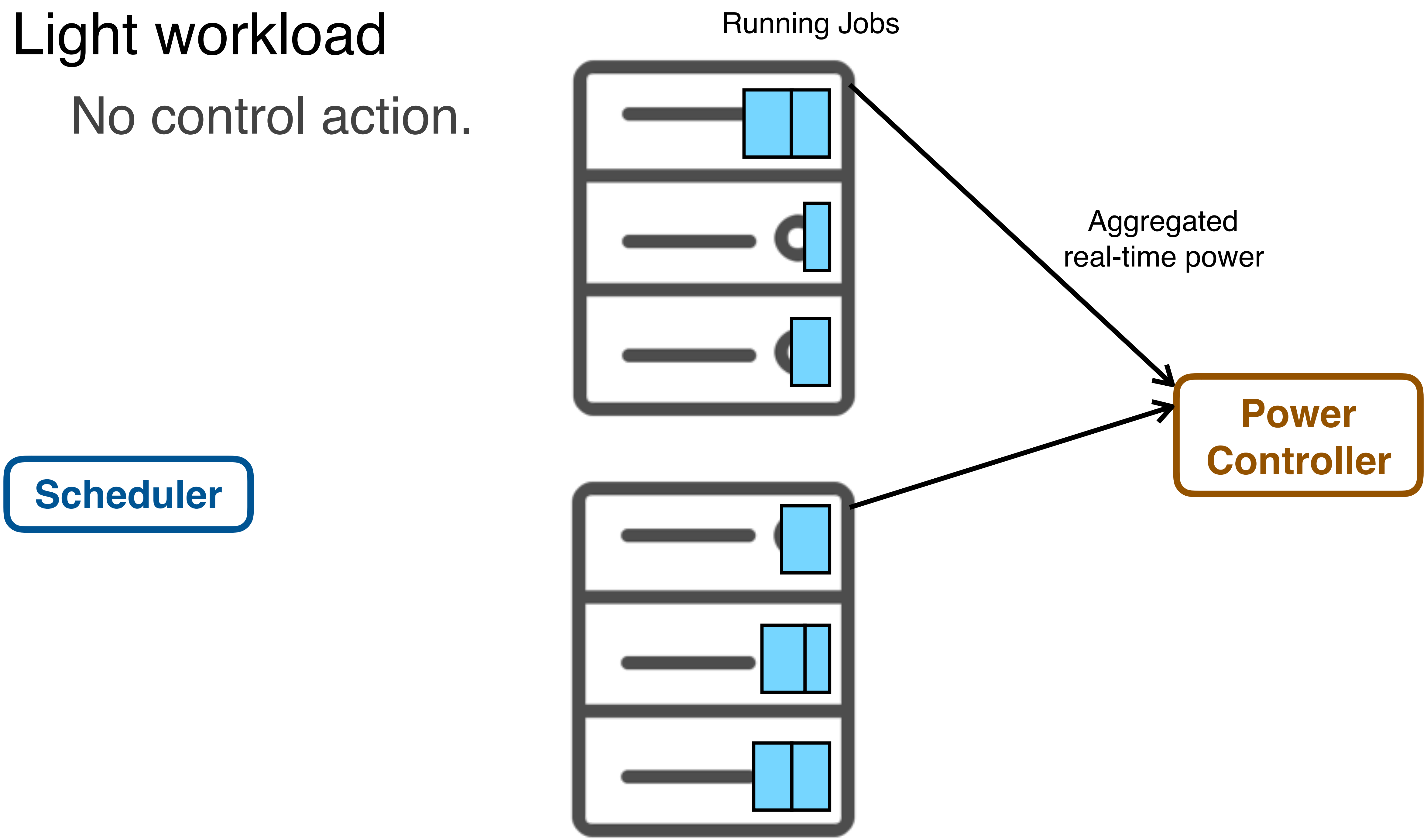
- Dynamic system control

> Tolerate noises.
>
> System identification in a production environment.

Example: Statistical Power Control

Light workload

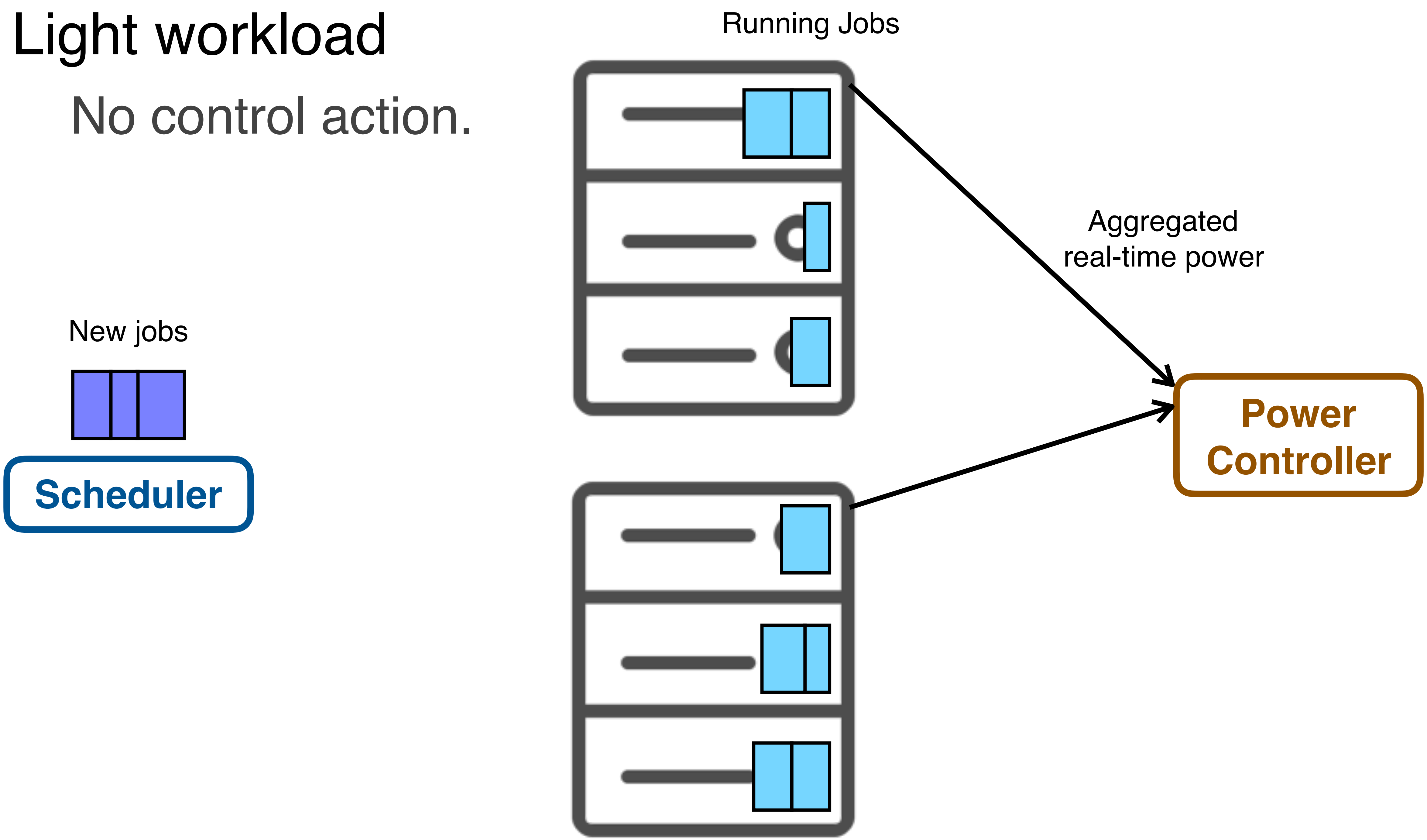
No control action.



Example: Statistical Power Control

Light workload

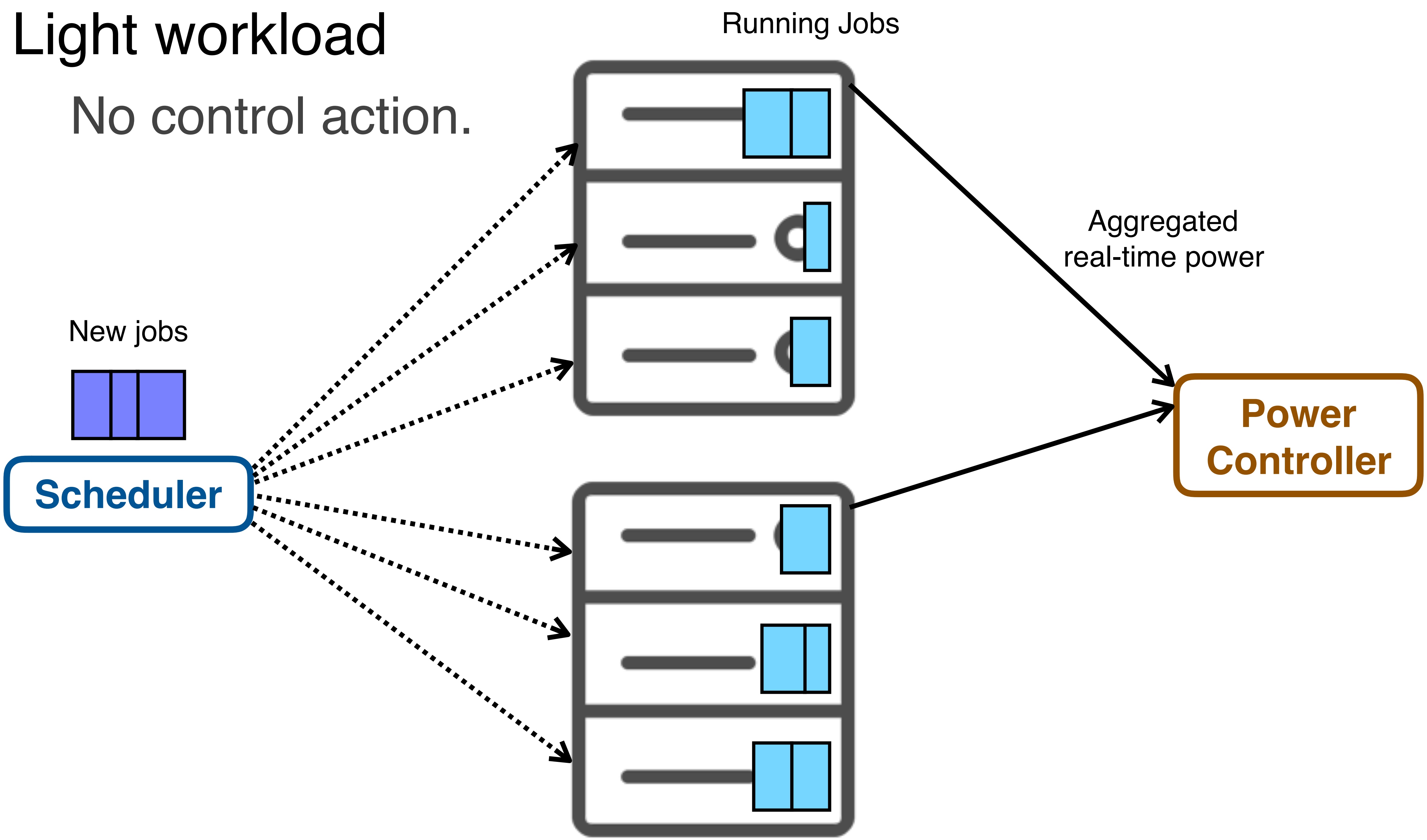
No control action.



Example: Statistical Power Control

Light workload

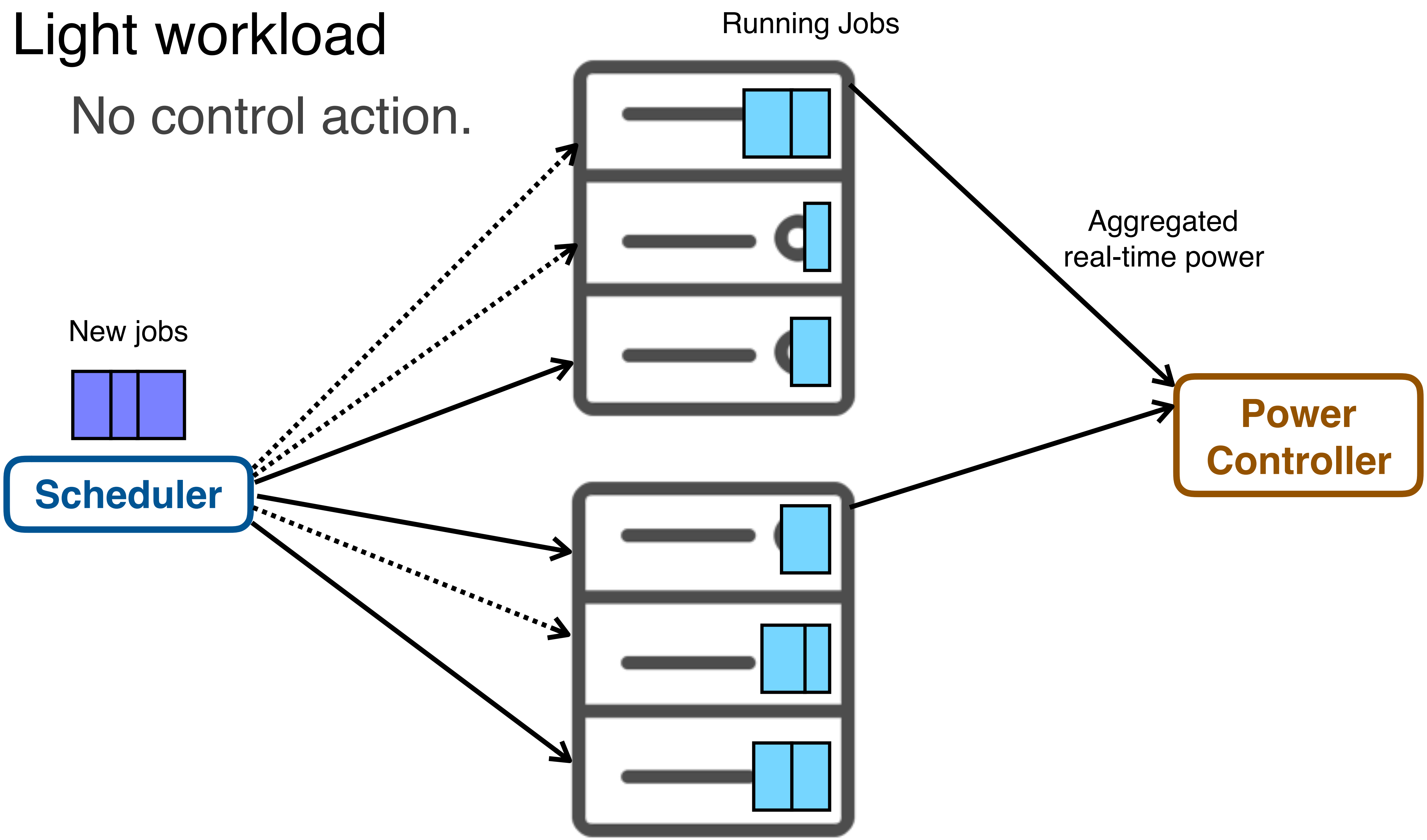
No control action.



Example: Statistical Power Control

Light workload

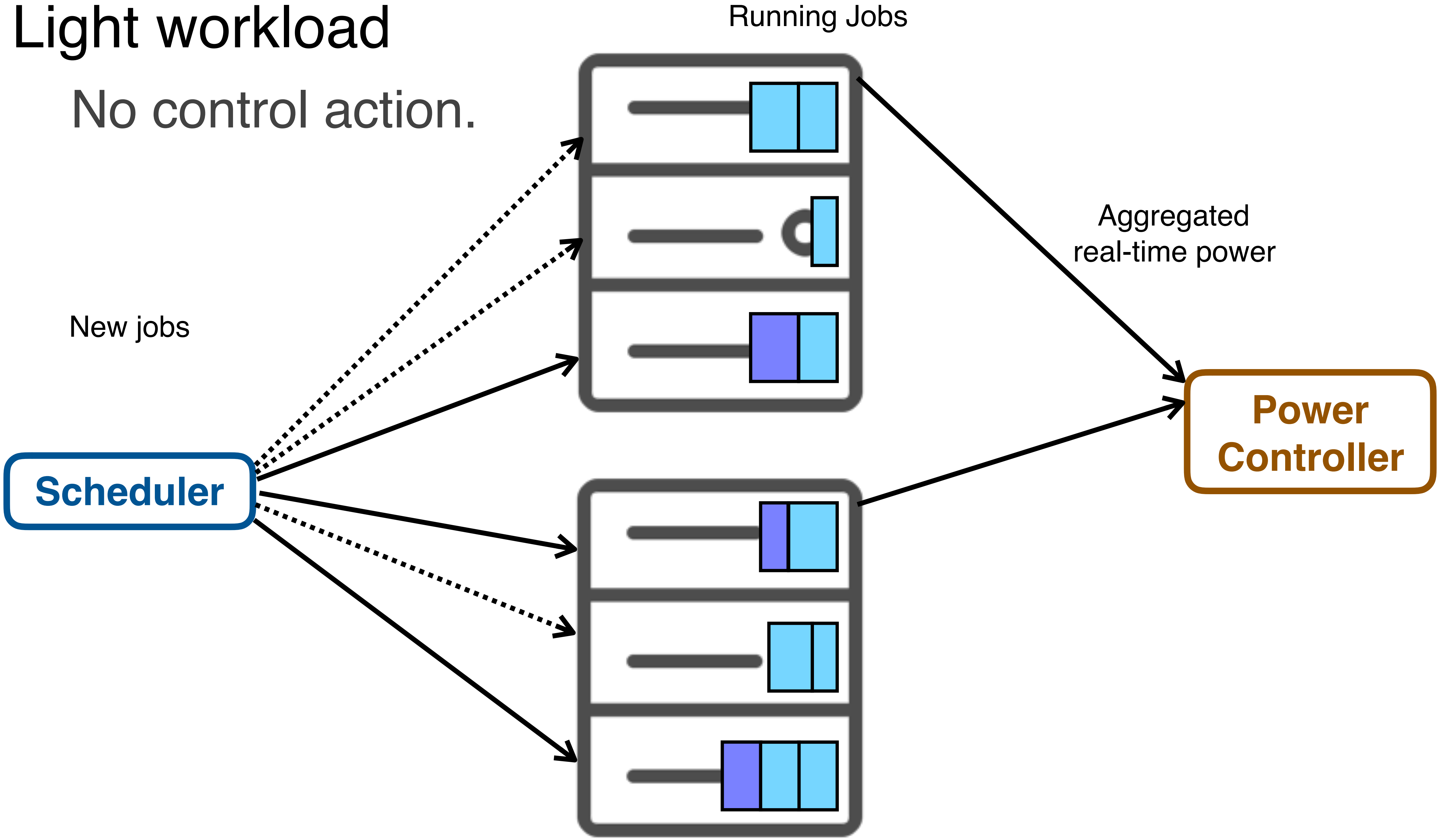
No control action.



Example: Statistical Power Control

Light workload

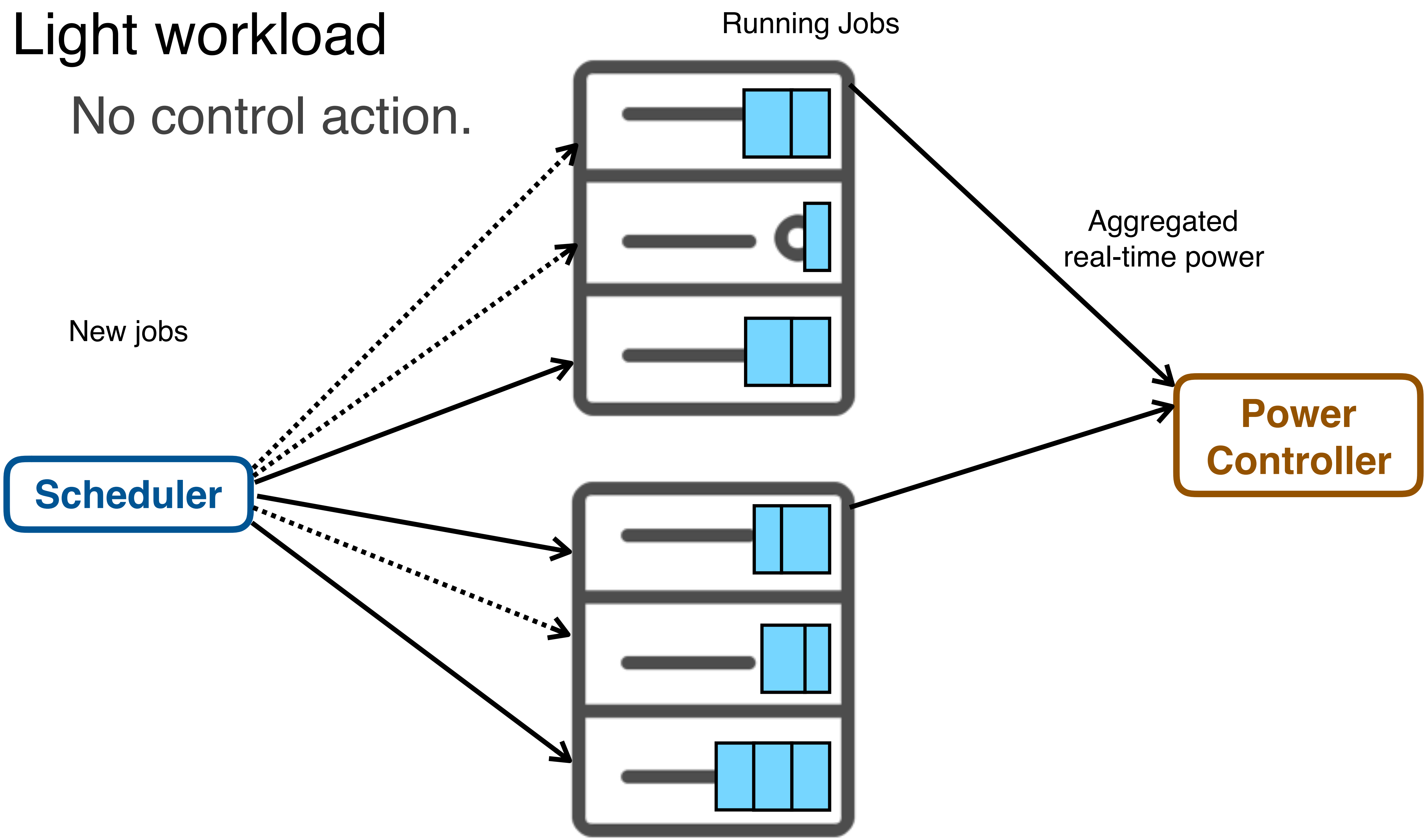
No control action.



Example: Statistical Power Control

Light workload

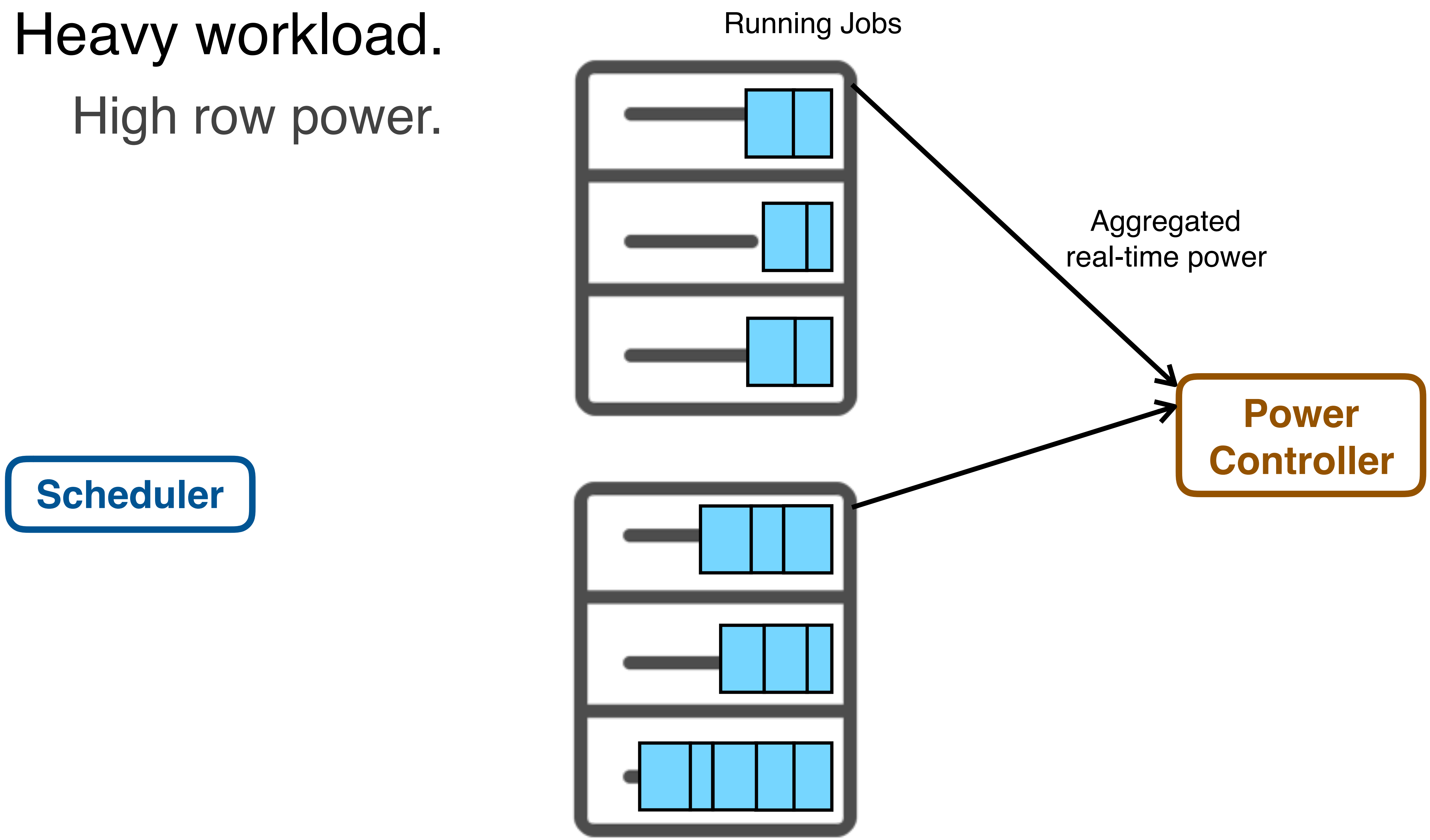
No control action.



Example: Statistical Power Control

Heavy workload.

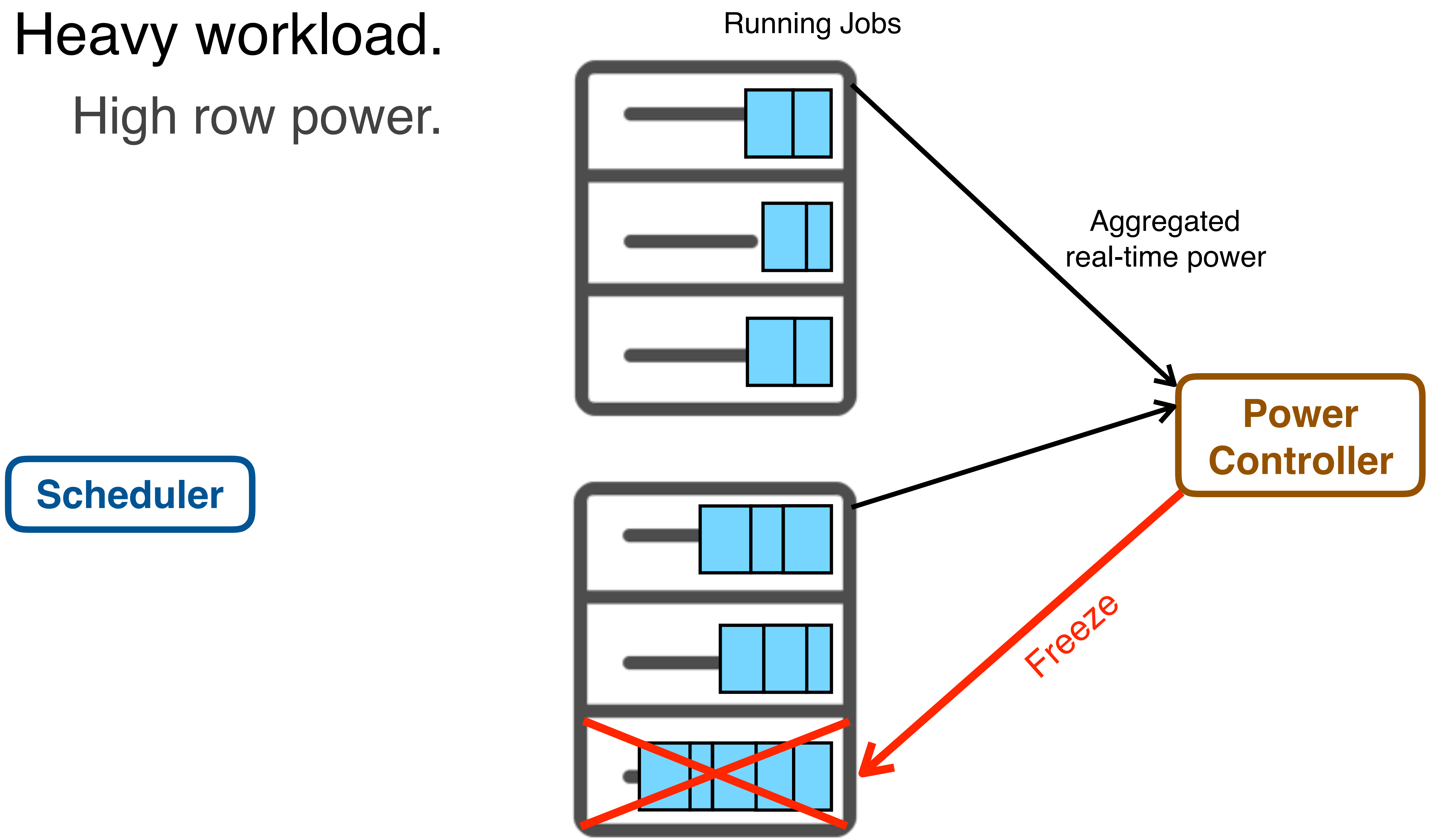
High row power.



Example: Statistical Power Control

Heavy workload.

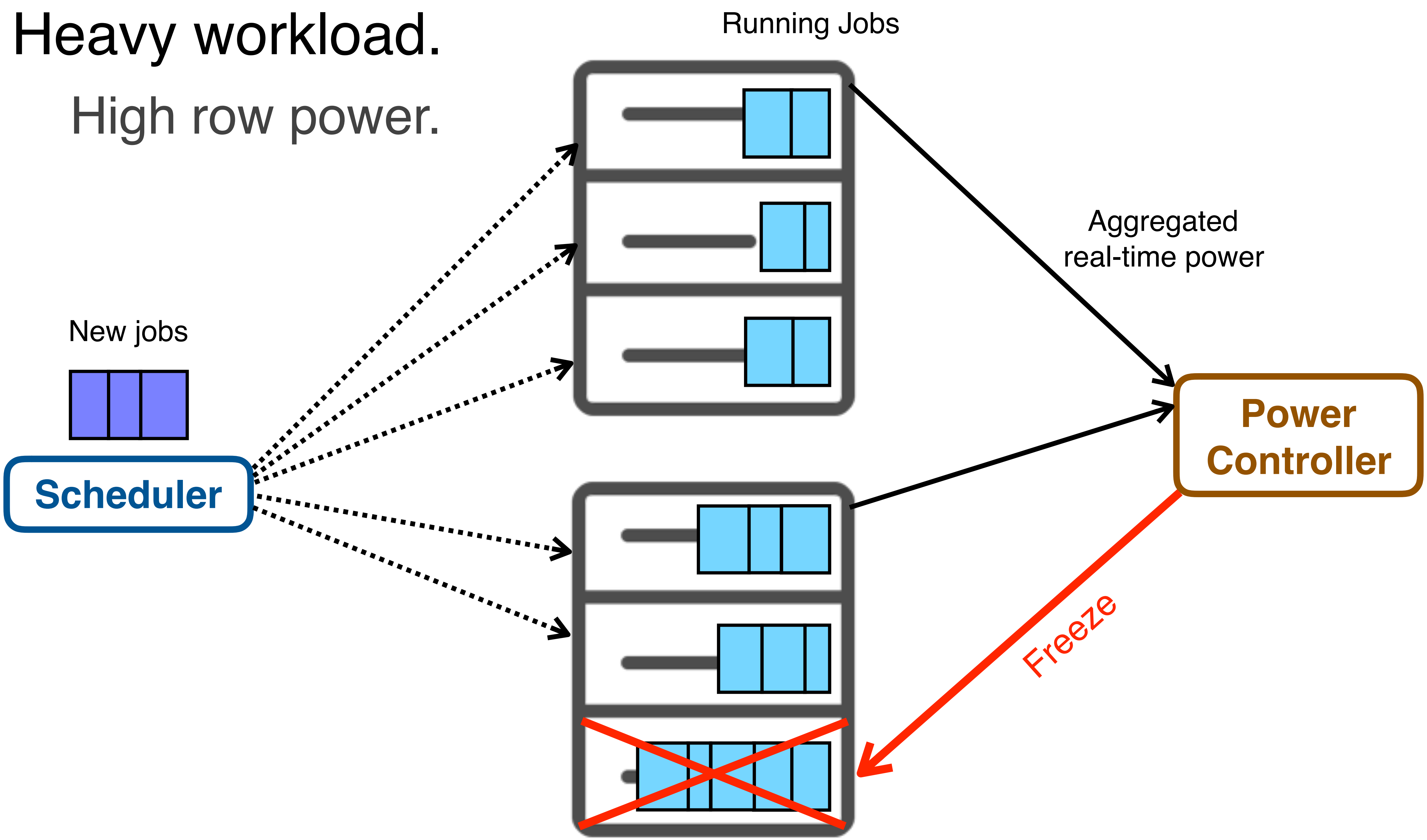
High row power.



Example: Statistical Power Control

Heavy workload.

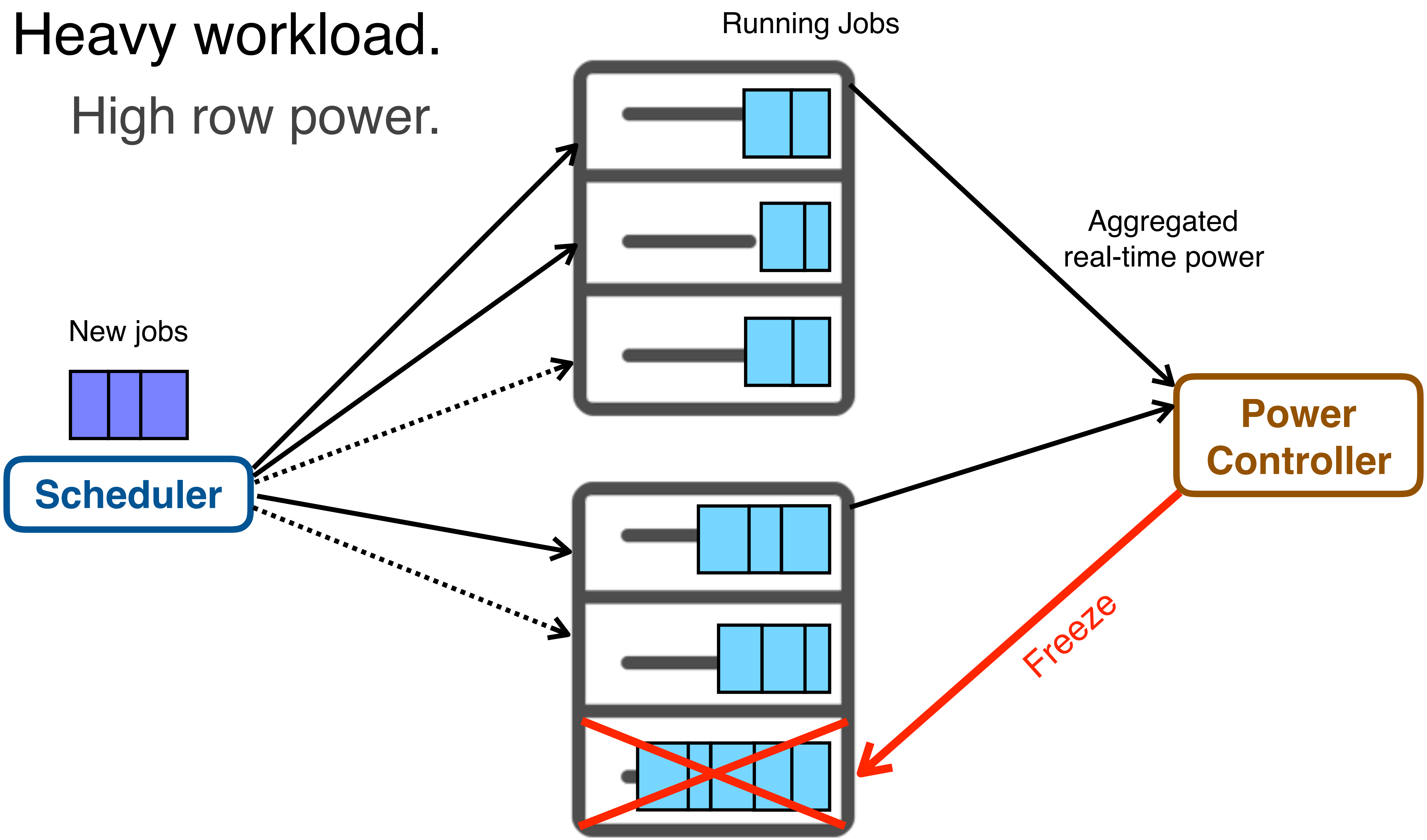
High row power.



Example: Statistical Power Control

Heavy workload.

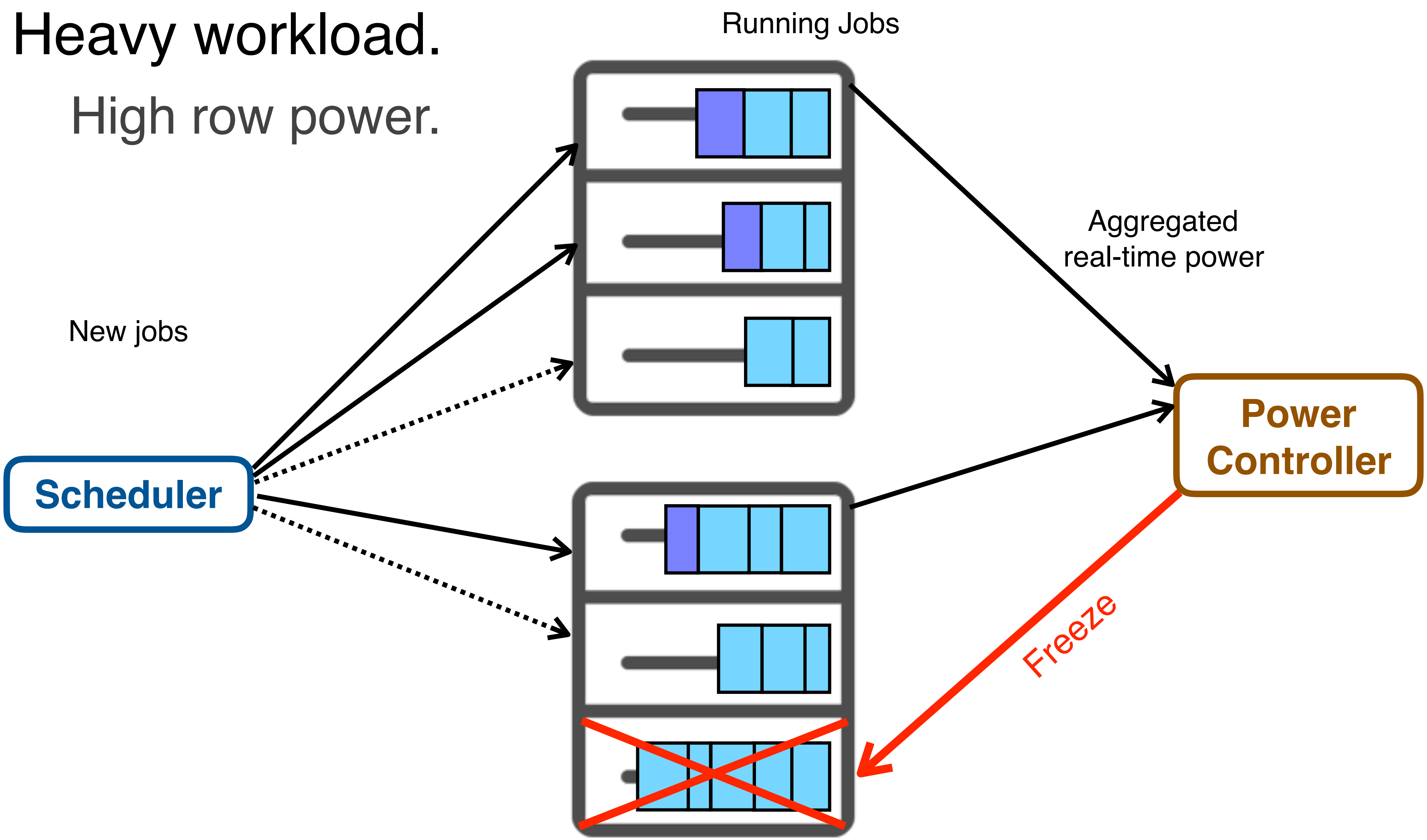
High row power.



Example: Statistical Power Control

Heavy workload.

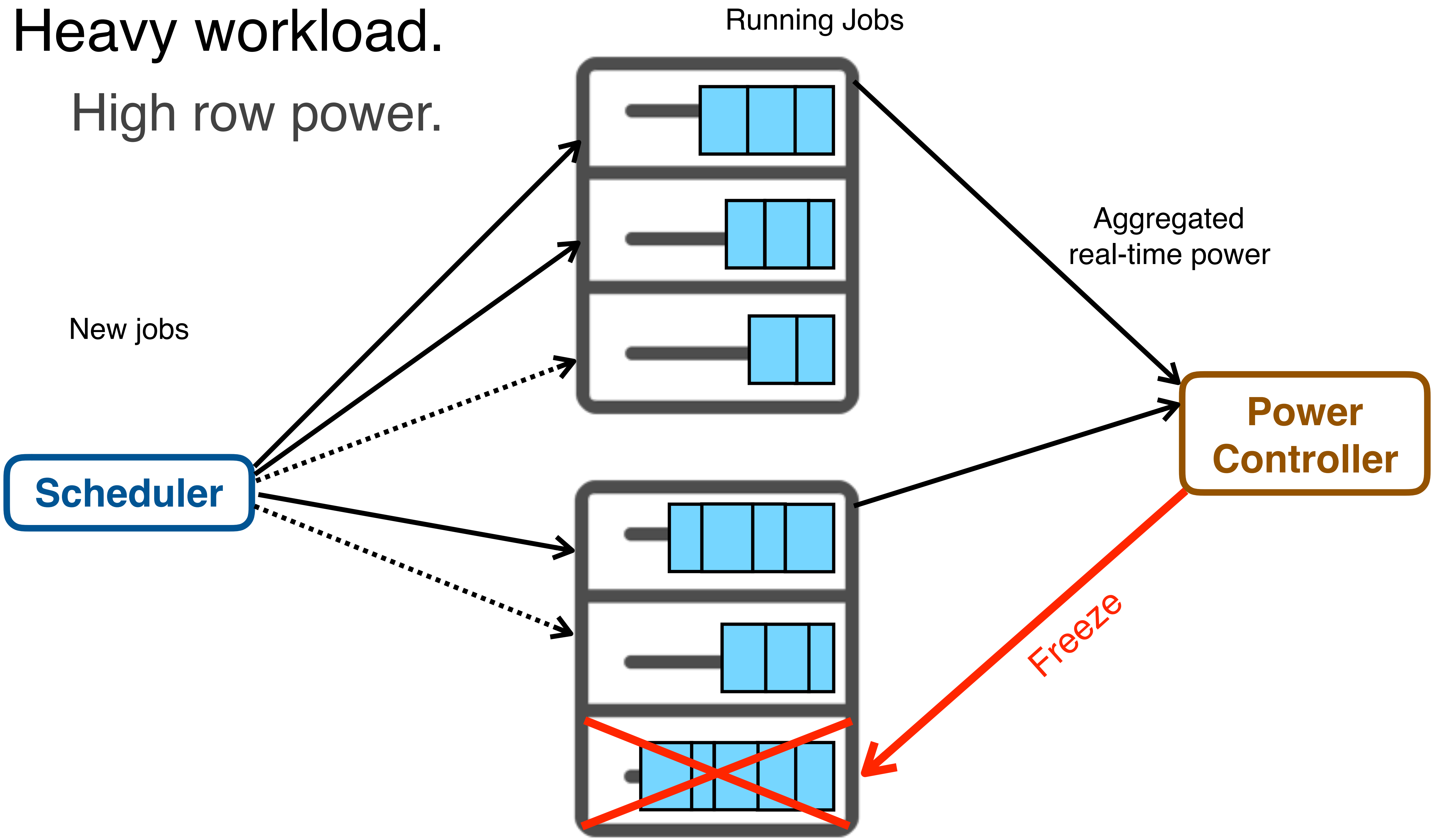
High row power.



Example: Statistical Power Control

Heavy workload.

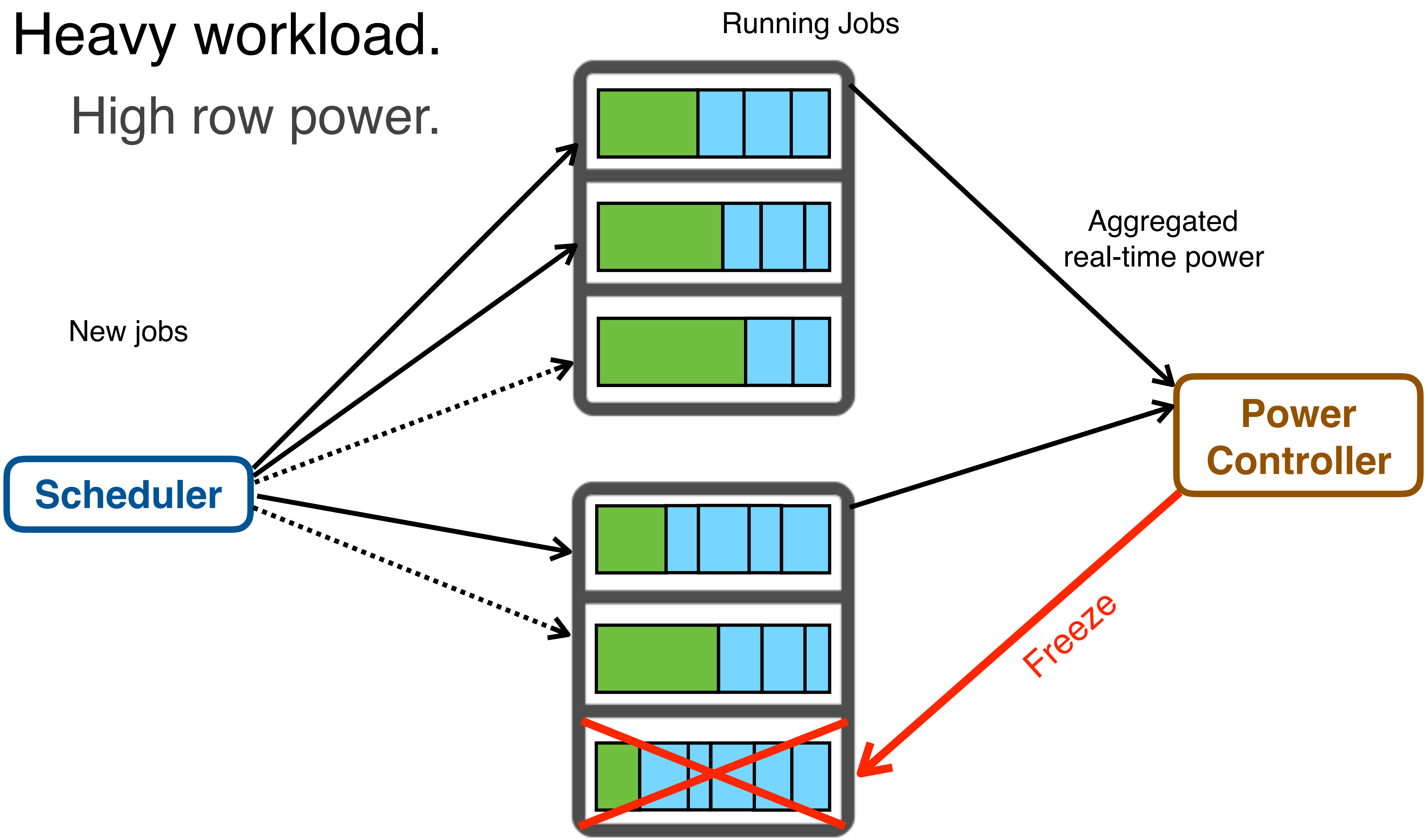
High row power.



Example: Statistical Power Control

Heavy workload.

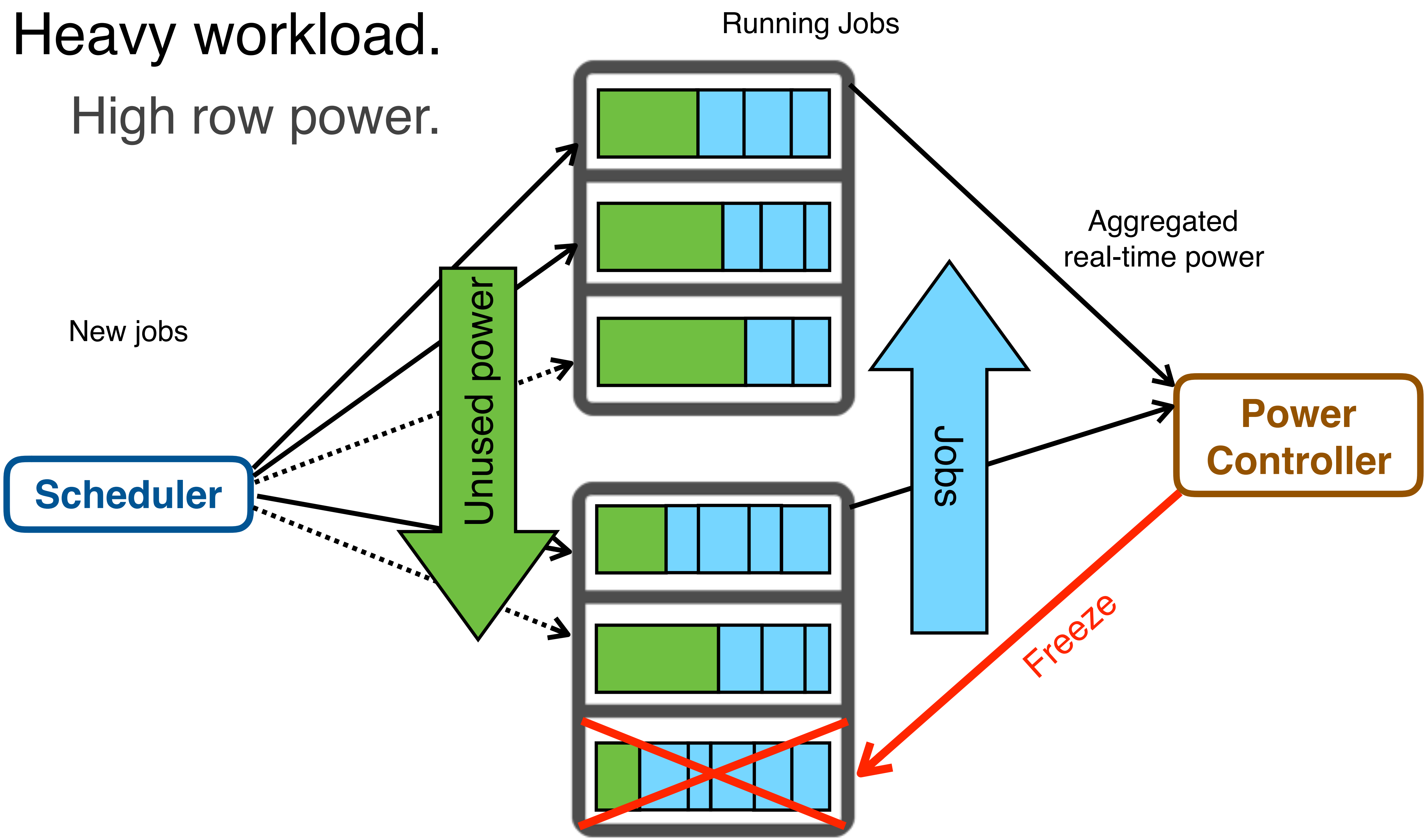
High row power.



Example: Statistical Power Control

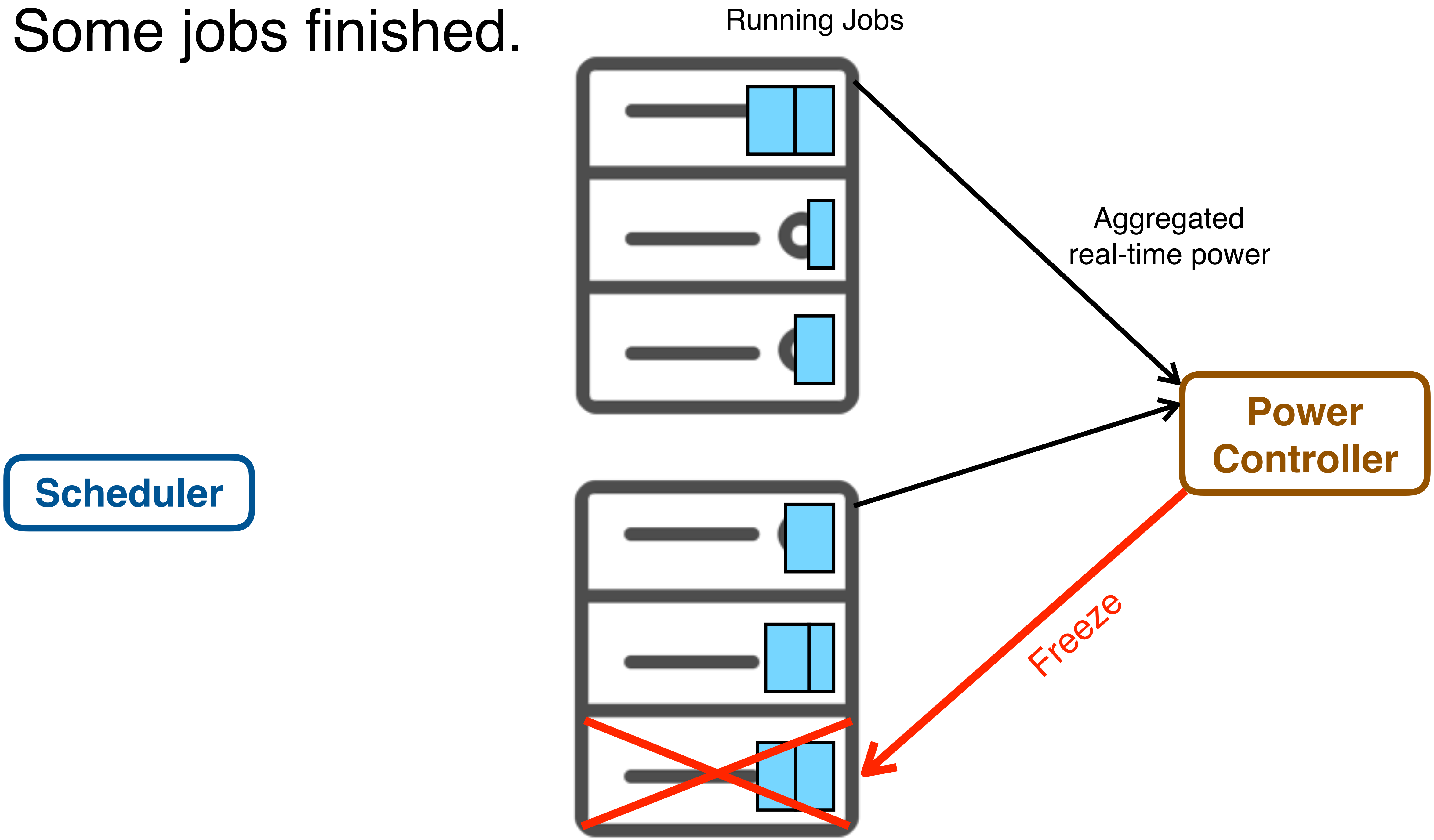
Heavy workload.

High row power.



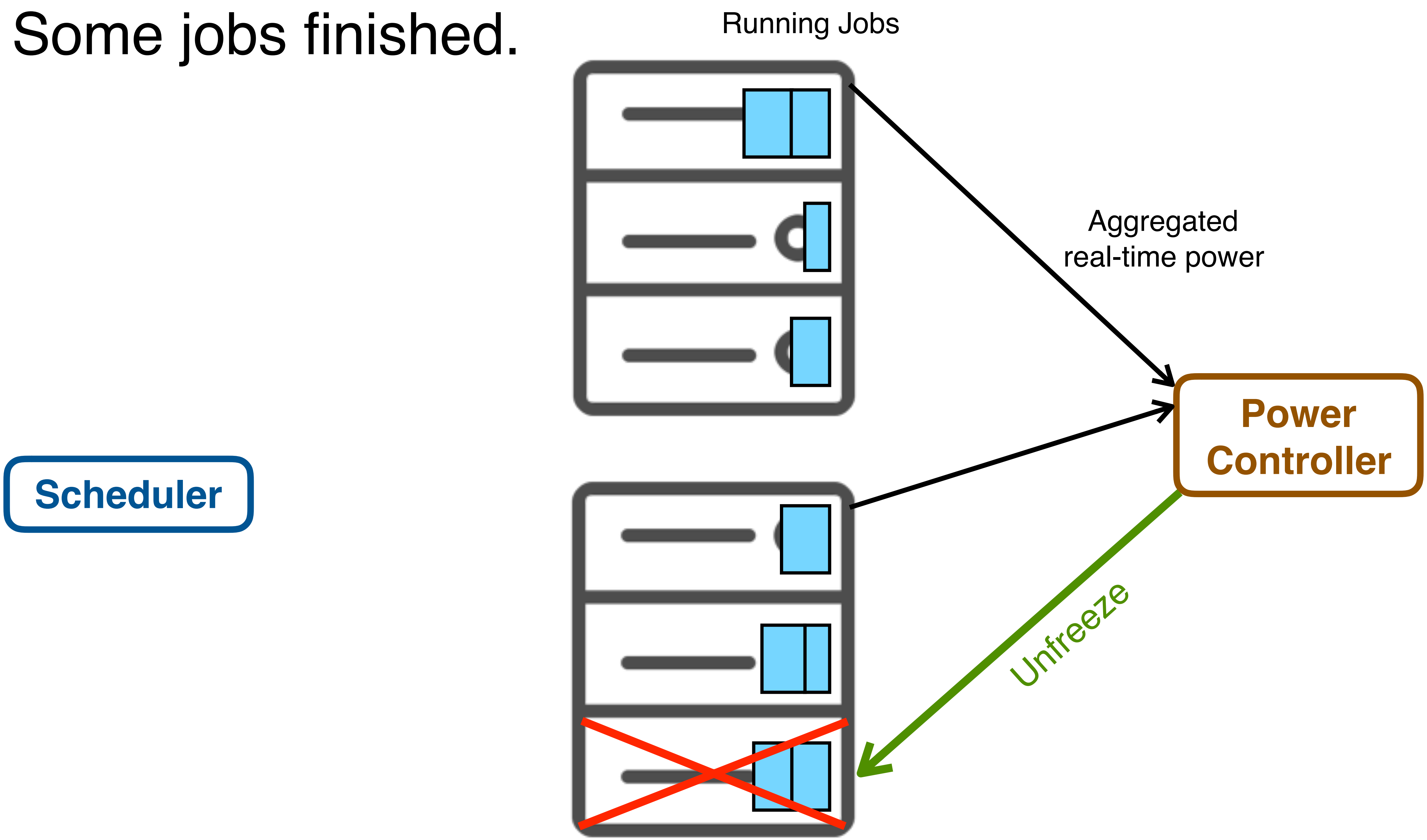
Example: Statistical Power Control

Some jobs finished.



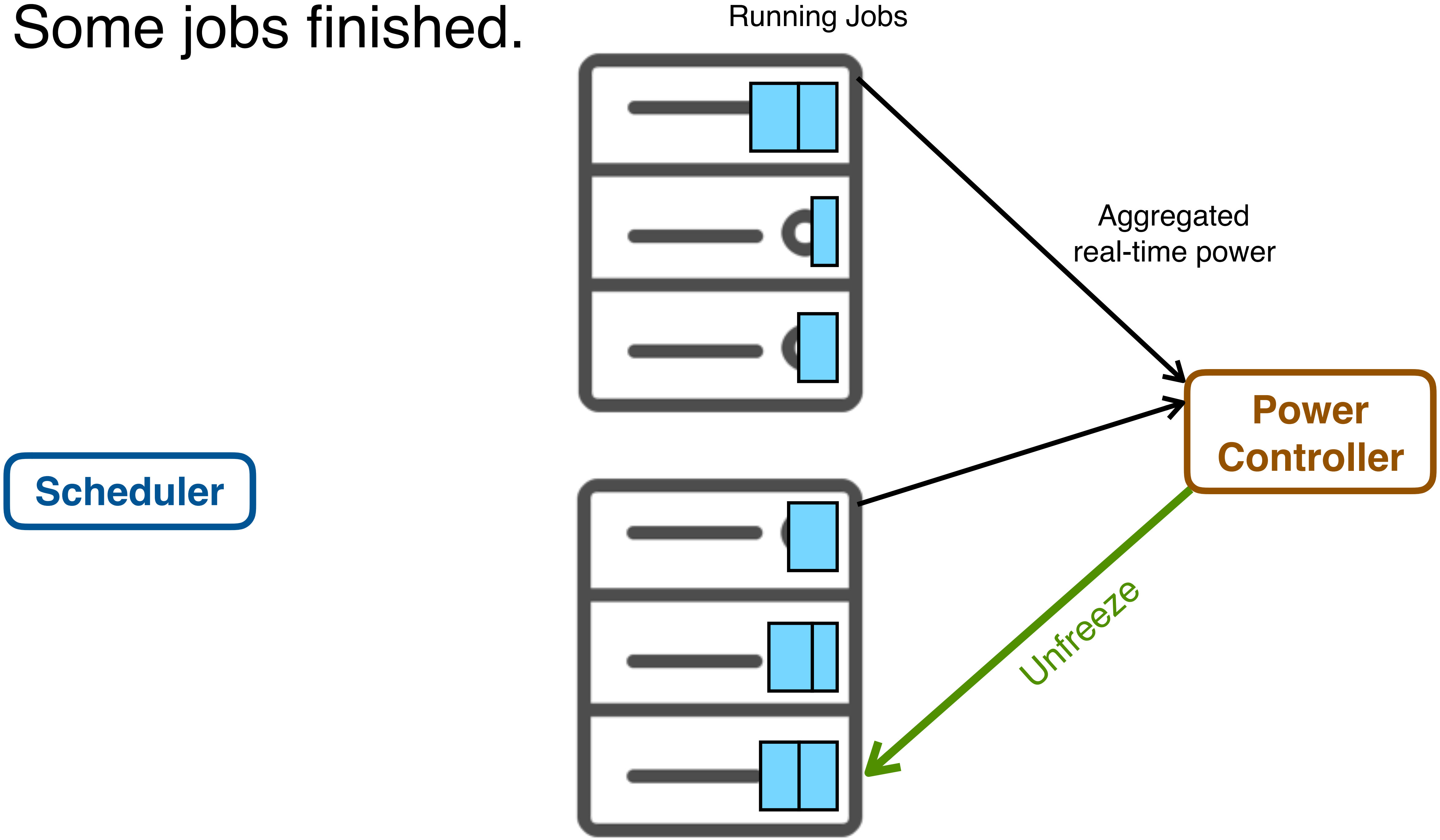
Example: Statistical Power Control

Some jobs finished.



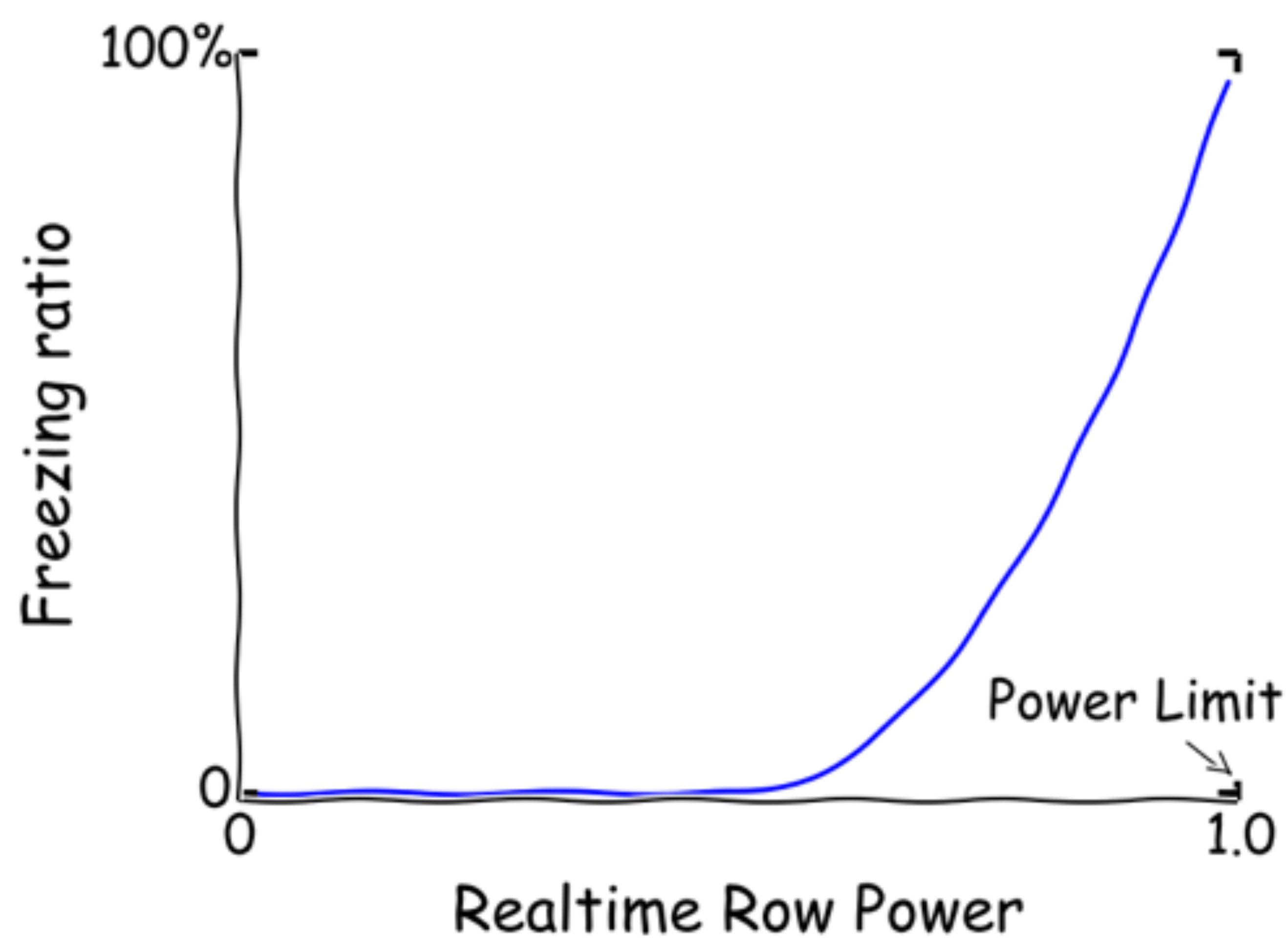
Example: Statistical Power Control

Some jobs finished.



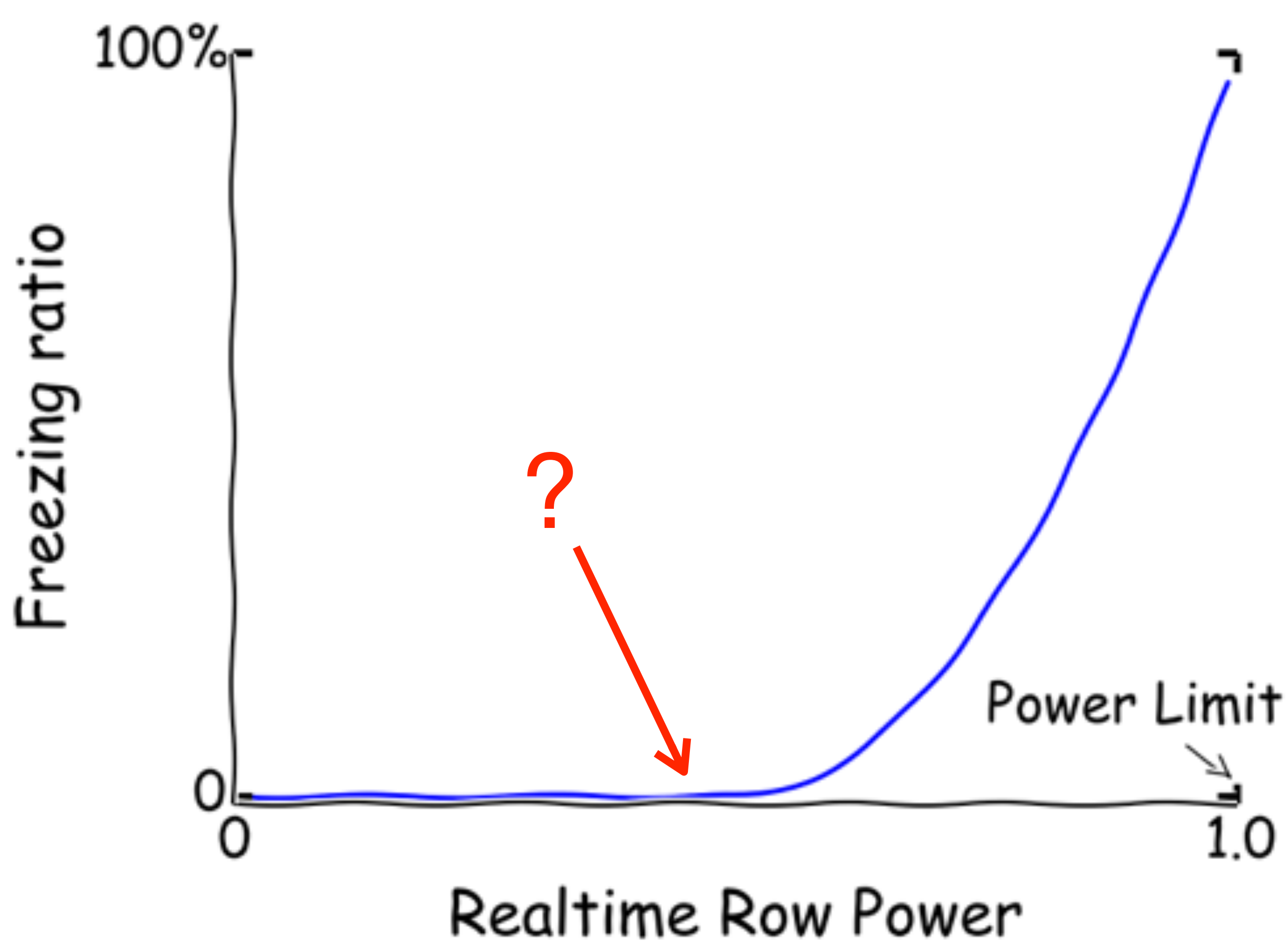
Power Control Model Blueprint

- Dynamic control at each minute.
- No control needed when the power is low.
- Freeze more/fewer servers when power is high/low.



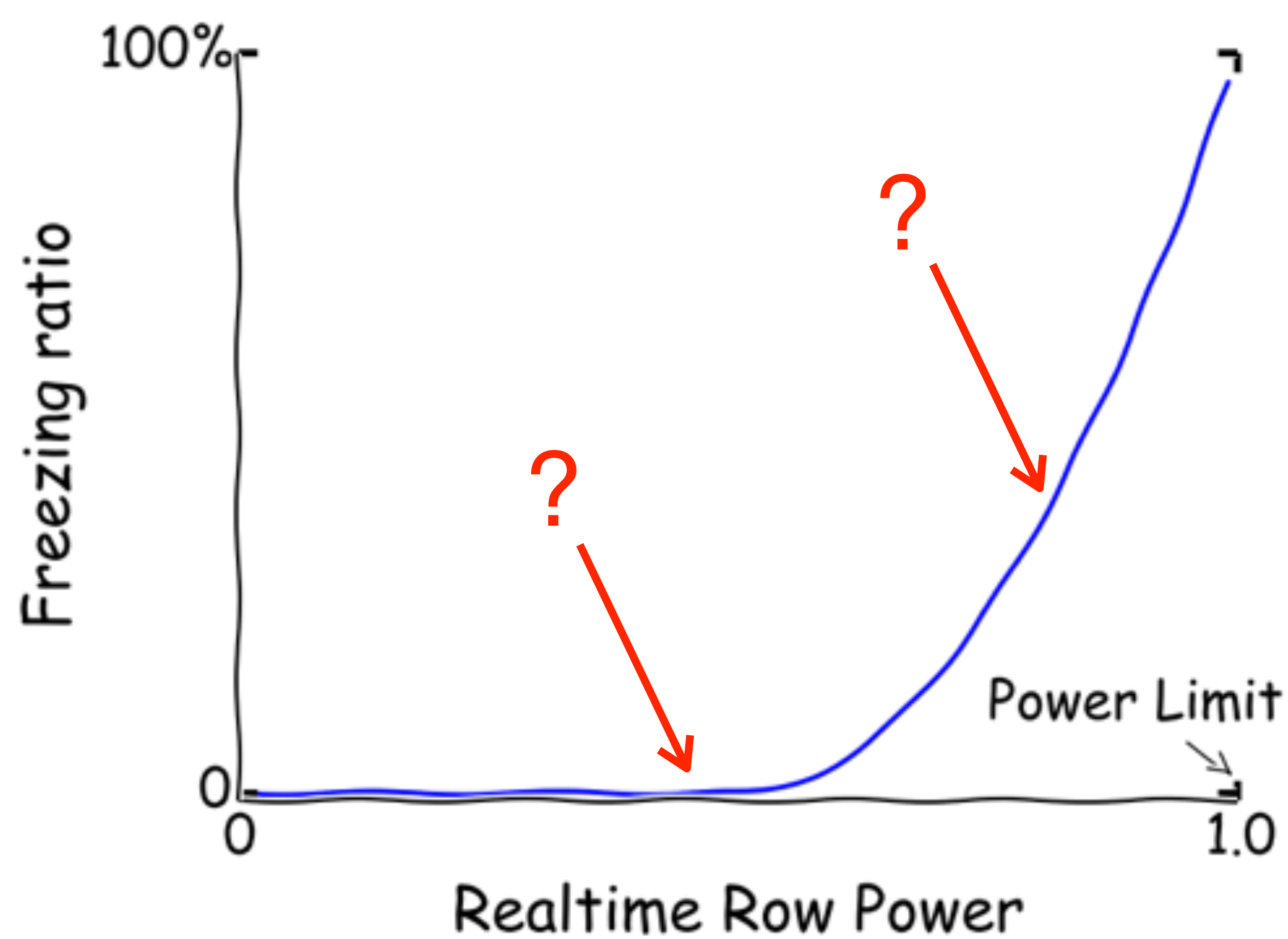
Power Control Model Blueprint

- Dynamic control at each minute.
- No control needed when the power is low.
- Freeze more/fewer servers when power is high/low.



Power Control Model Blueprint

- Dynamic control at each minute.
- No control needed when the power is low.
- Freeze more/fewer servers when power is high/low.



Effect of Freezing Servers

Two effects jointly impact on the row-level power.

- Existing jobs will finish
- Statistically fewer jobs scheduled to the row

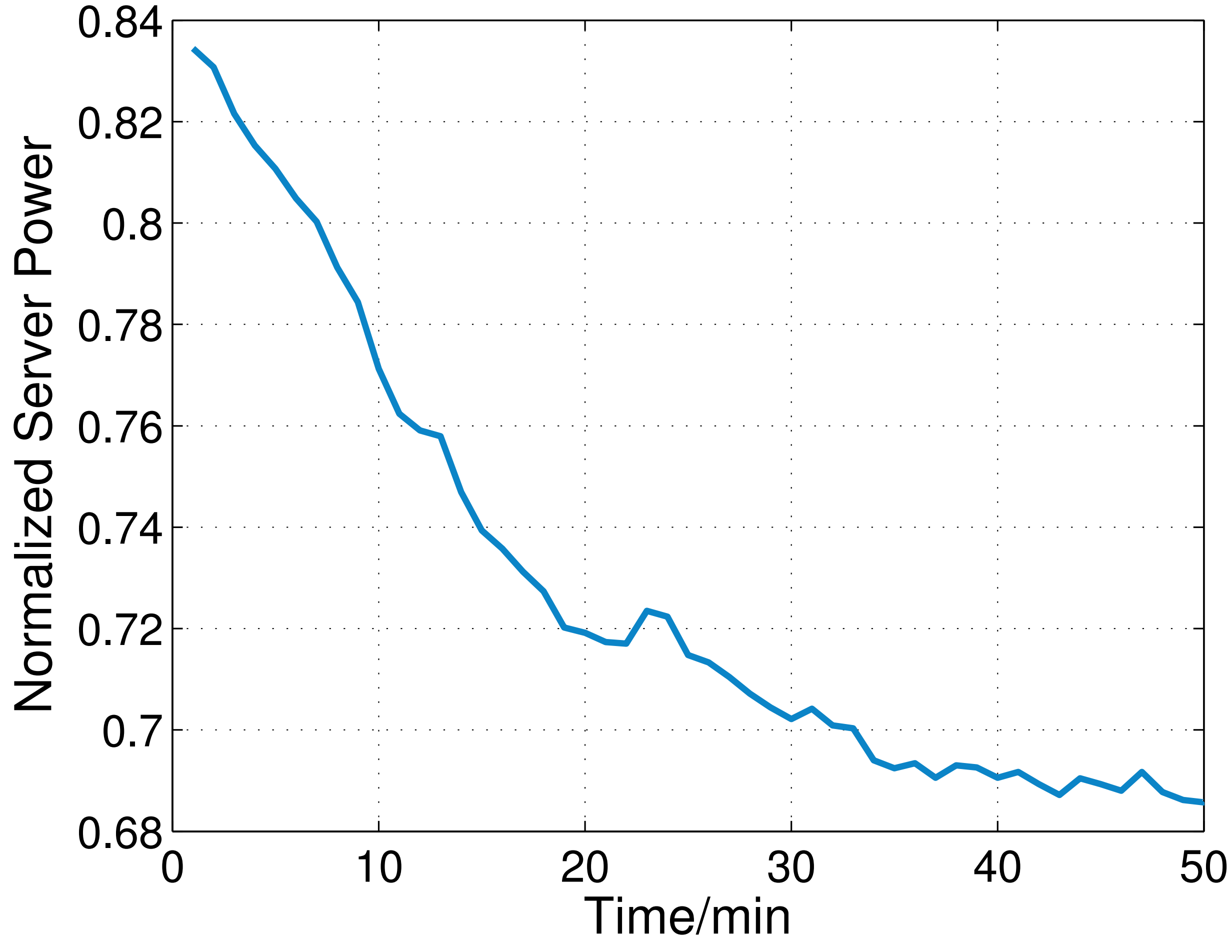


Fig: Average normalized power of about 80 servers after they are frozen.

Effect of Freezing Servers

Two effects jointly impact on the row-level power.

- Existing jobs will finish
- Statistically fewer jobs scheduled to the row

How to **quantify** these effects?

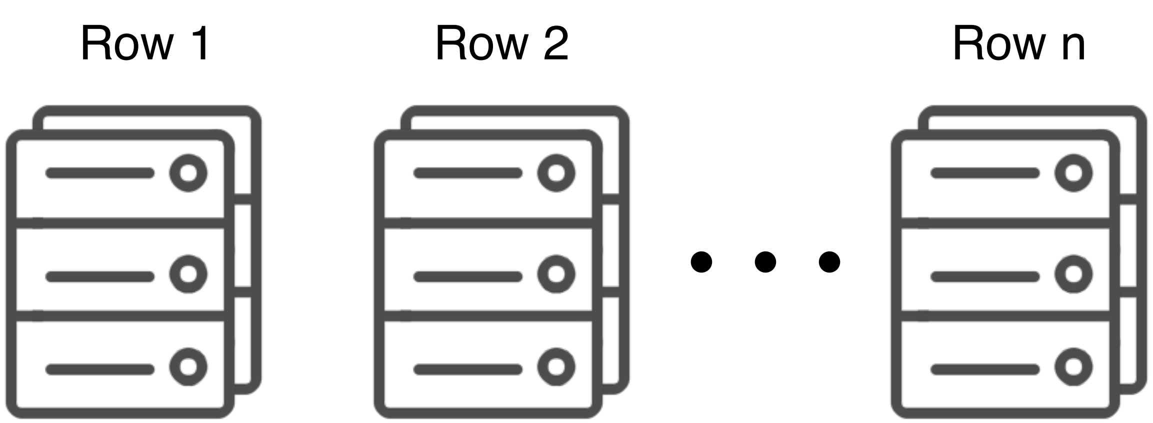
System identification in a production environment?

Designed a controlled experiment.

Controlled Experiment Design

Controlled experiment in production environment.

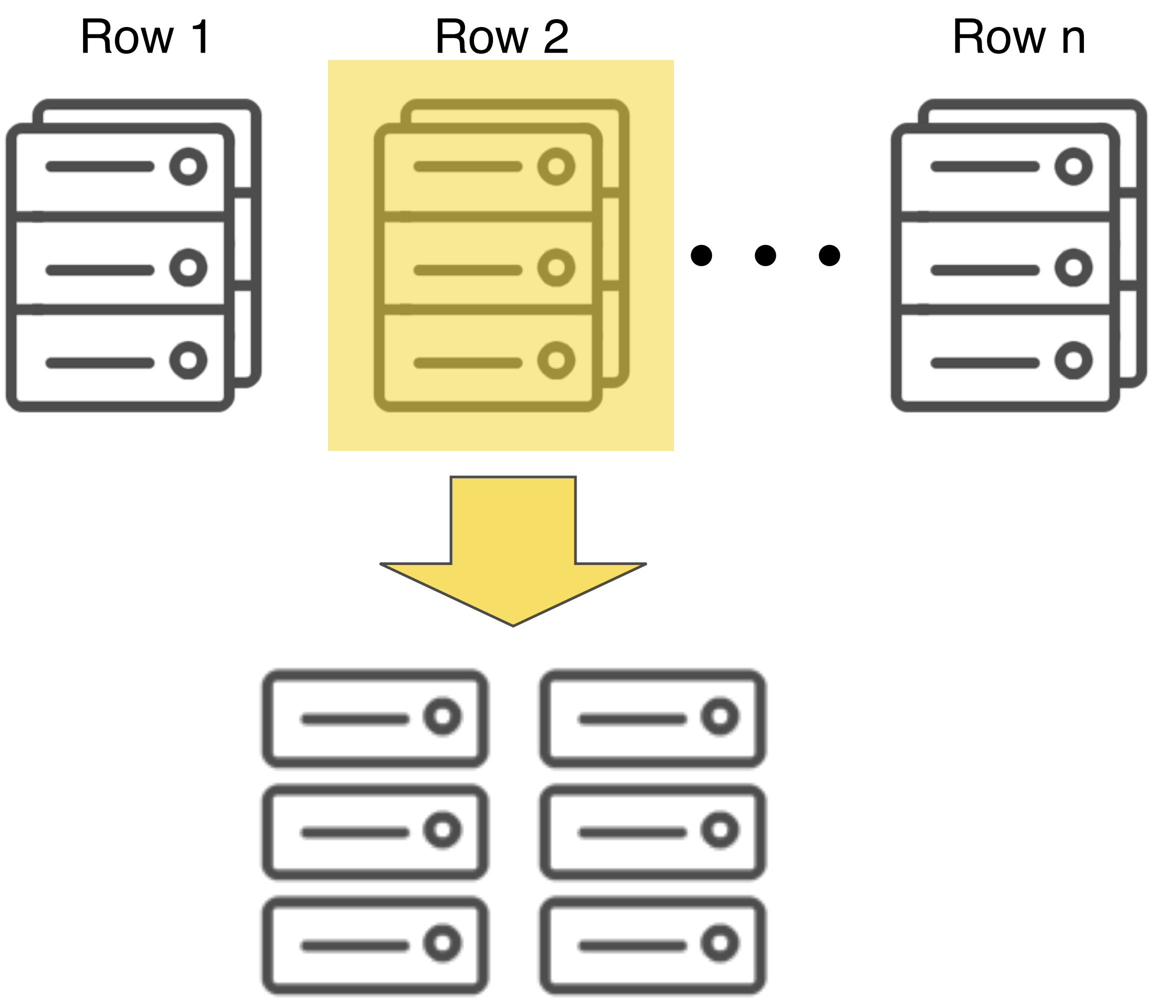
Idea: A/B testing



Controlled Experiment Design

Controlled experiment in production environment.

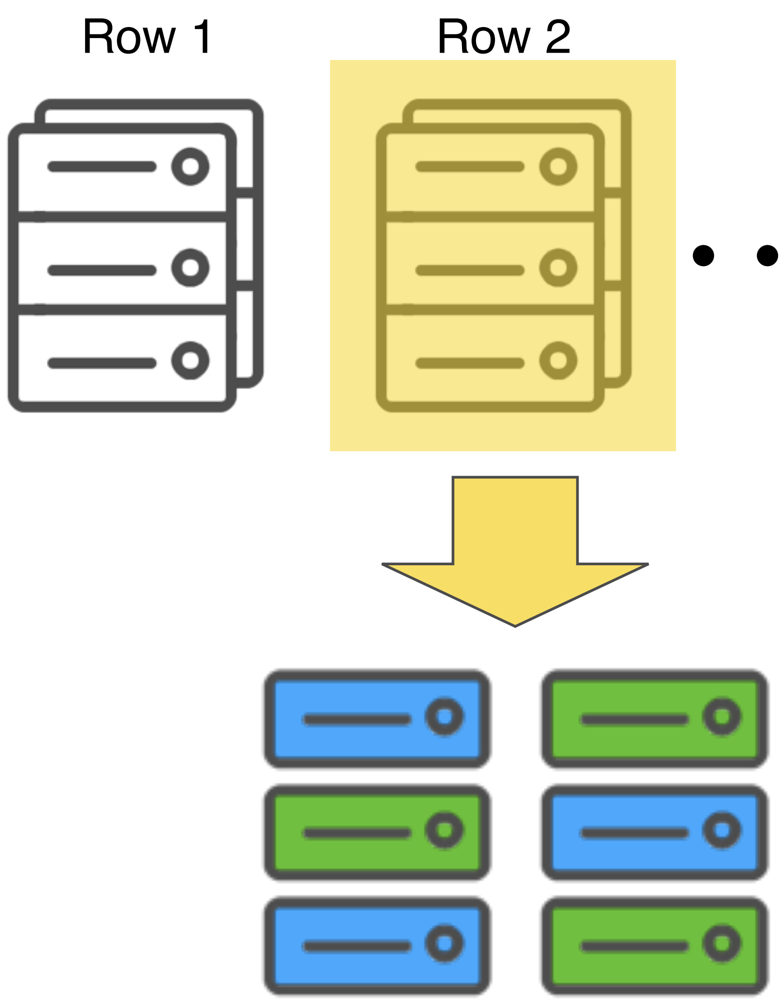
Idea: A/B testing



Controlled Experiment Design

Controlled experiment in production environment.

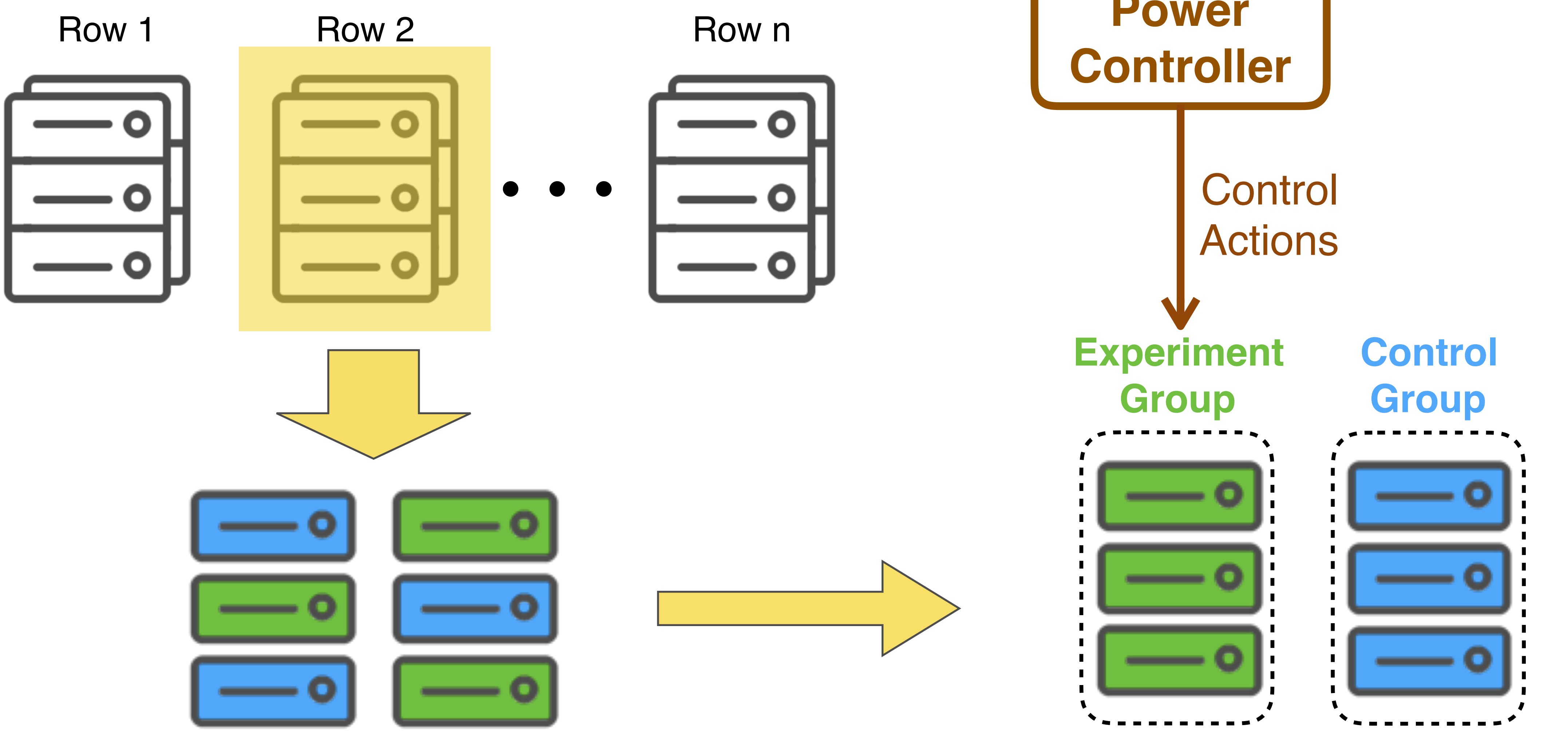
Idea: A/B testing



Controlled Experiment Design

Controlled experiment in production environment.

Idea: A/B testing



Correlation coefficient of the group power is 0.946

Dynamic Control Model

How many servers do we need to freeze in a row?

- Freeze too few: Risk of Power violations!
- Freeze too many: Reduce the throughput!

Optimization problem:

- Maximize: TPW (Throughput per Provisioned Watt)
- s.t. No power violation

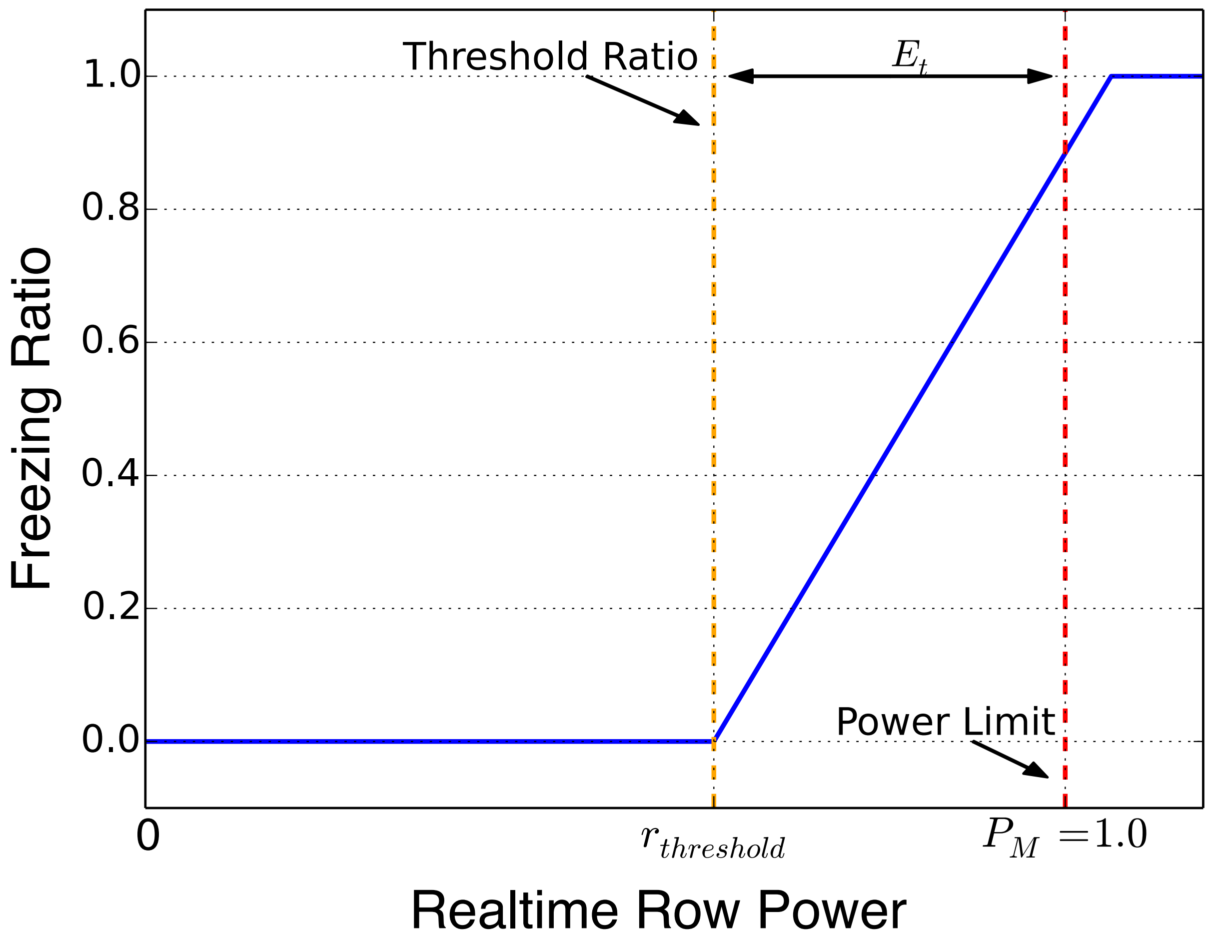
Key idea:

- Use simple system model and tolerate inaccuracy with dynamic control.

Dynamic Control Model

Use heuristics to derive a simple control model.

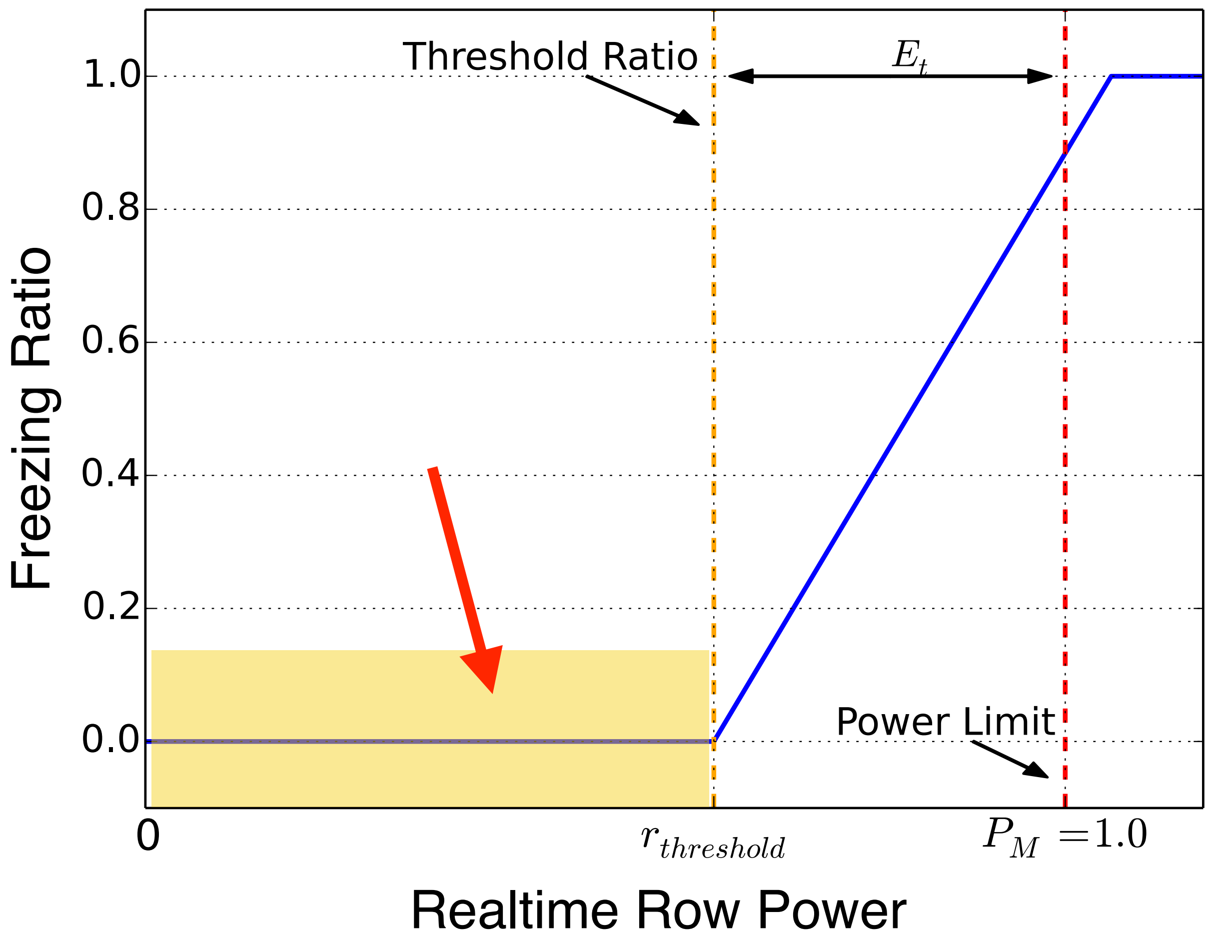
- Take control actions at each minute.
- Details in the paper.



Dynamic Control Model

Use heuristics to derive a simple control model.

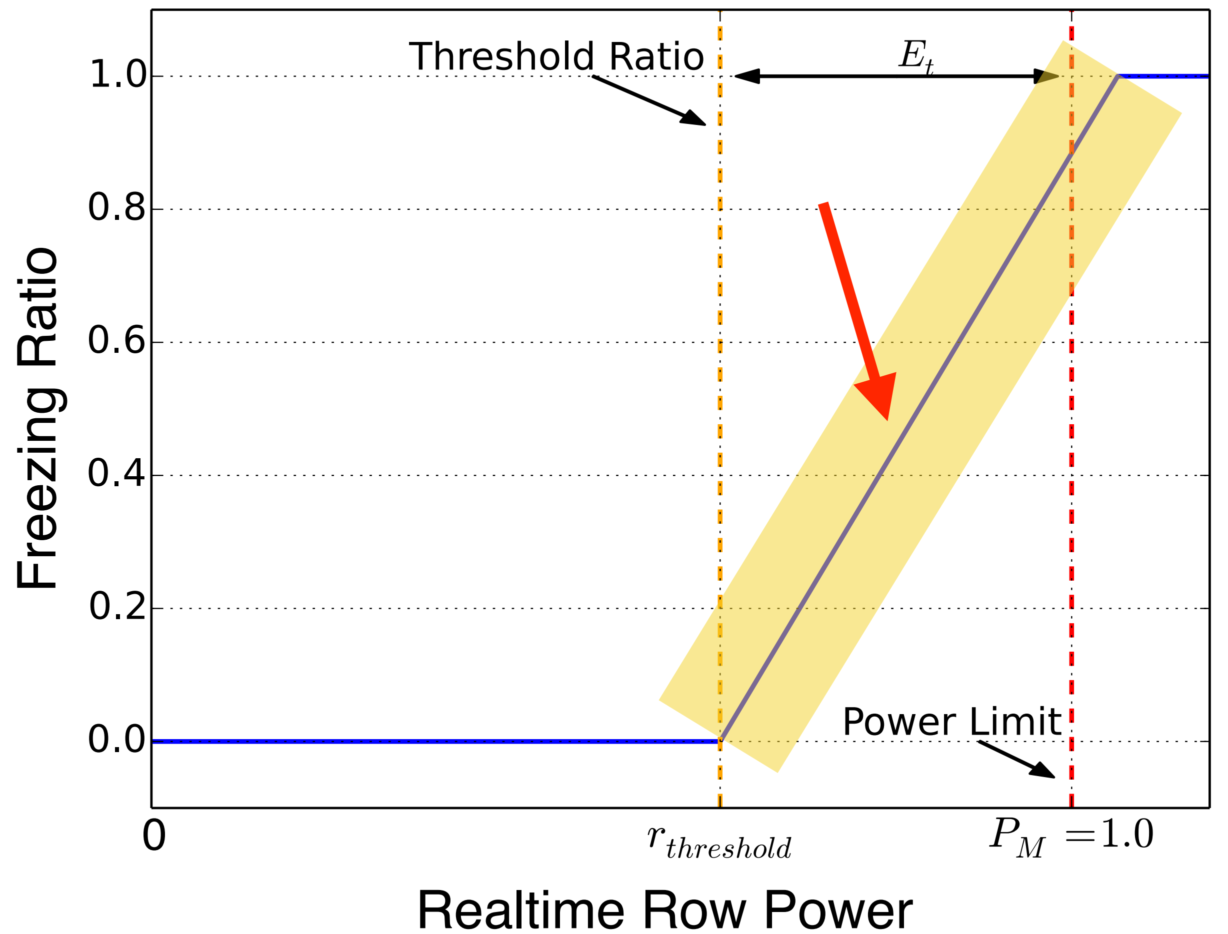
- Take control actions at each minute.
- Details in the paper.



Dynamic Control Model

Use heuristics to derive a simple control model.

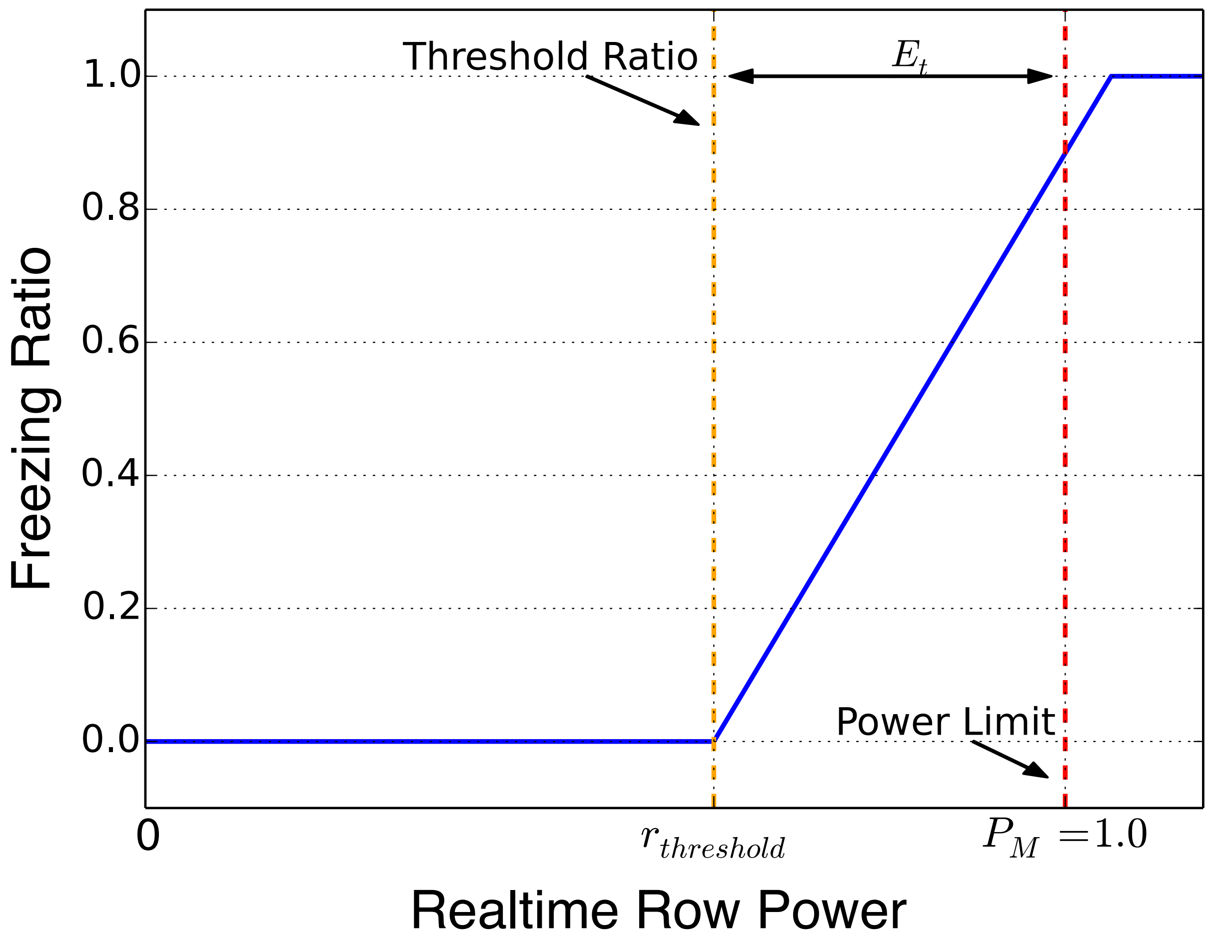
- Take control actions at each minute.
- Details in the paper.



Dynamic Control Model

Use heuristics to derive a simple control model.

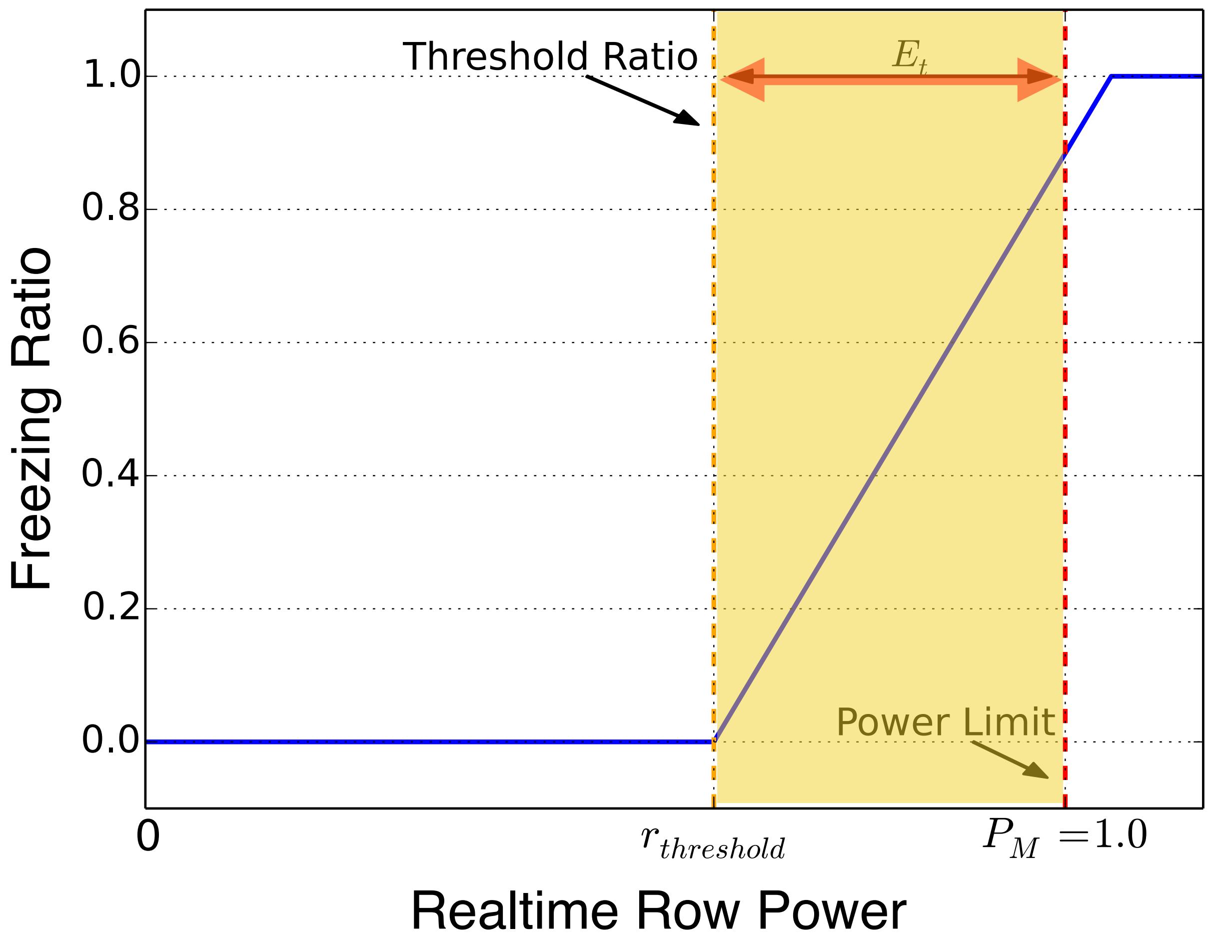
- Take control actions at each minute.
- Details in the paper.



Dynamic Control Model

Use heuristics to derive a simple control model.

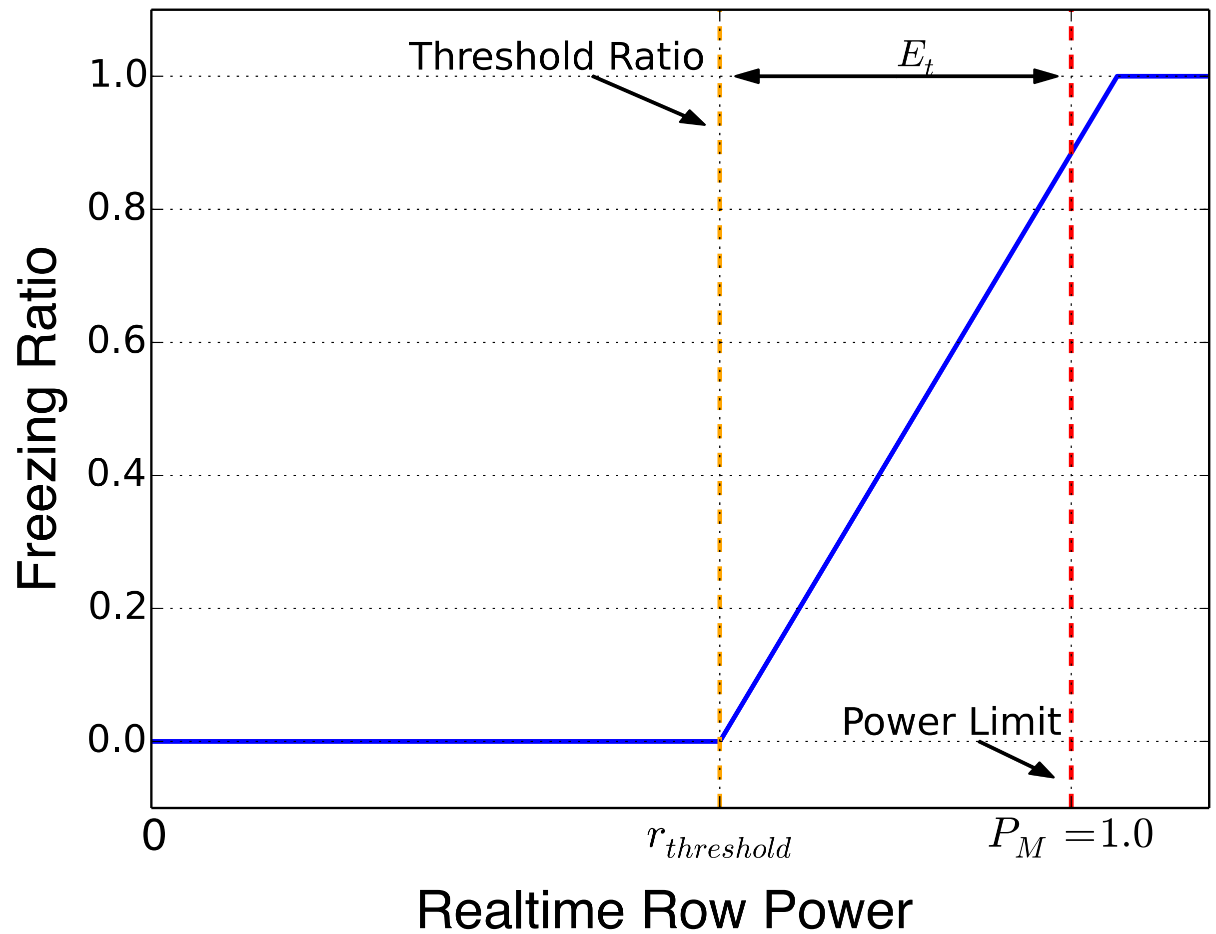
- Take control actions at each minute.
- Details in the paper.



Dynamic Control Model

Use heuristics to derive a simple control model.

- Take control actions at each minute.
- Details in the paper.



How to Emulate Over-provisioning?

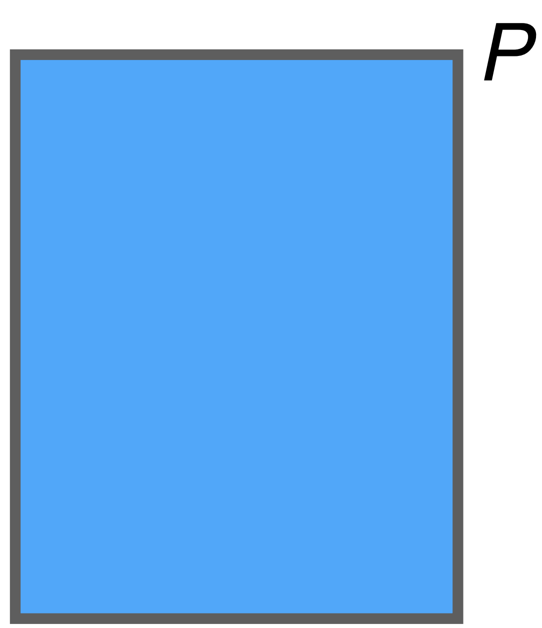
- Safety: Unacceptable to truly trigger power violations in production environment.
- Flexibility: How to test various over-provisioning ratio?

Solution: Emulating power violations by virtually scaling down the power budget of the row.

How to Emulate Over-provisioning?

- Safety: Unacceptable to truly trigger power violations in production environment.
- Flexibility: How to test various over-provisioning ratio?

Solution: Emulating power violations by virtually scaling down the power budget of the row.



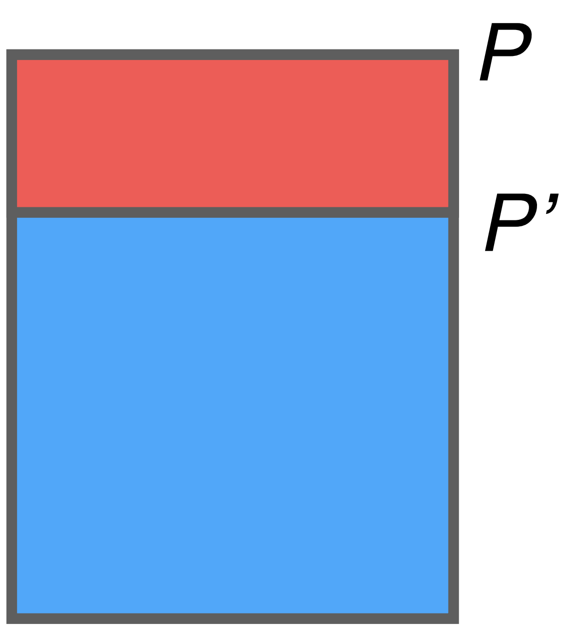
Actual row power budget: *P*

Assumed row power budget: *P'*
Over-provisioning ratio: *(P-P')/P'*

How to Emulate Over-provisioning?

- Safety: Unacceptable to truly trigger power violations in production environment.
- Flexibility: How to test various over-provisioning ratio?

Solution: Emulating power violations by virtually scaling down the power budget of the row.



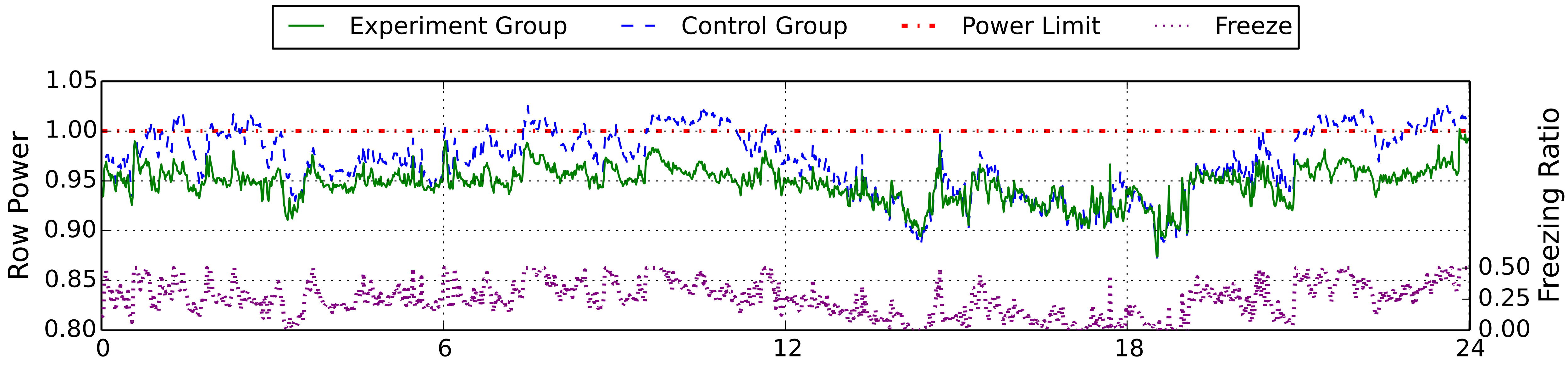
Actual row power budget: *P*

Assumed row power budget: *P’*
Over-provisioning ratio: *(P-P’)/P’*

Effectiveness

Controlled experiments on production environment.

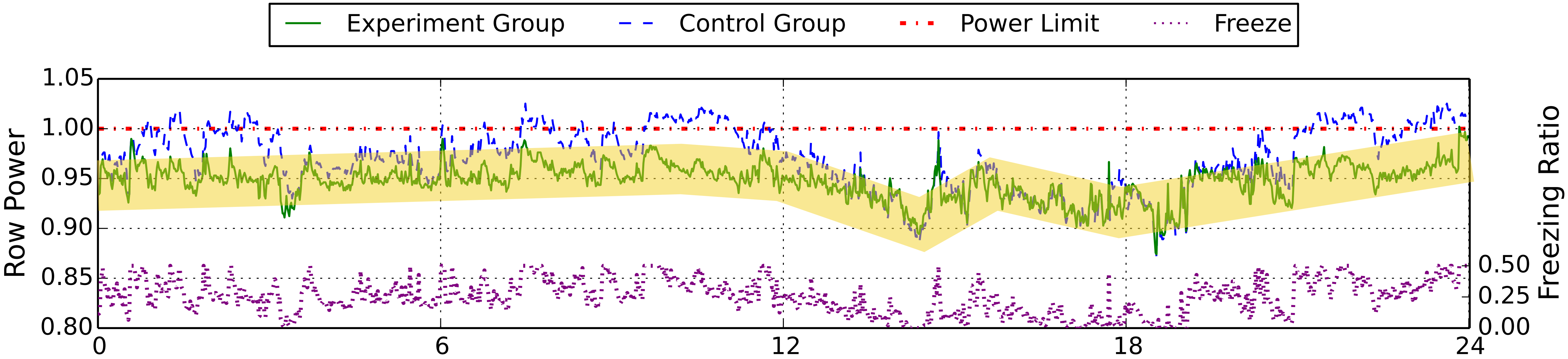
Over-provisioning ratio = 0.25



Effectiveness

Controlled experiments on production environment.

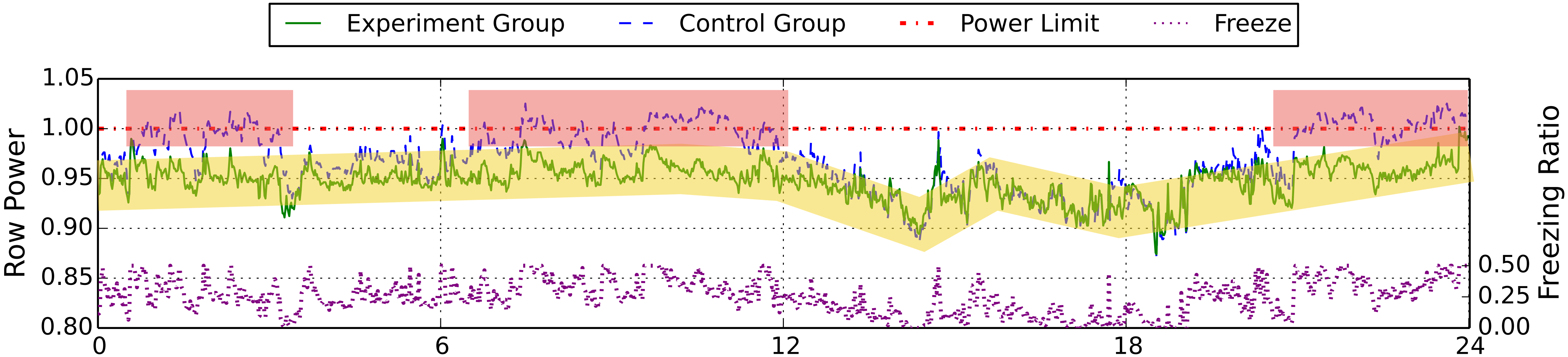
Over-provisioning ratio = 0.25



Effectiveness

Controlled experiments on production environment.

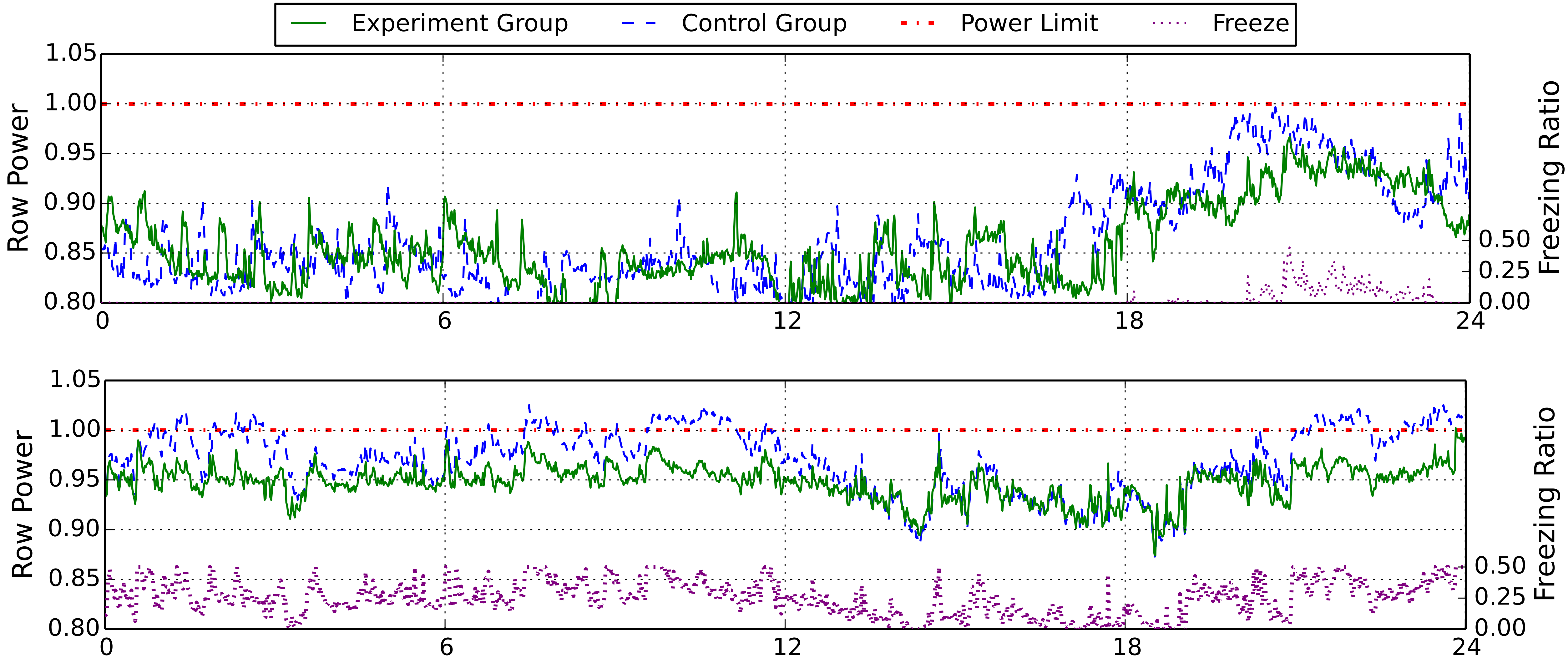
Over-provisioning ratio = 0.25



Effectiveness

Controlled experiments on production environment.

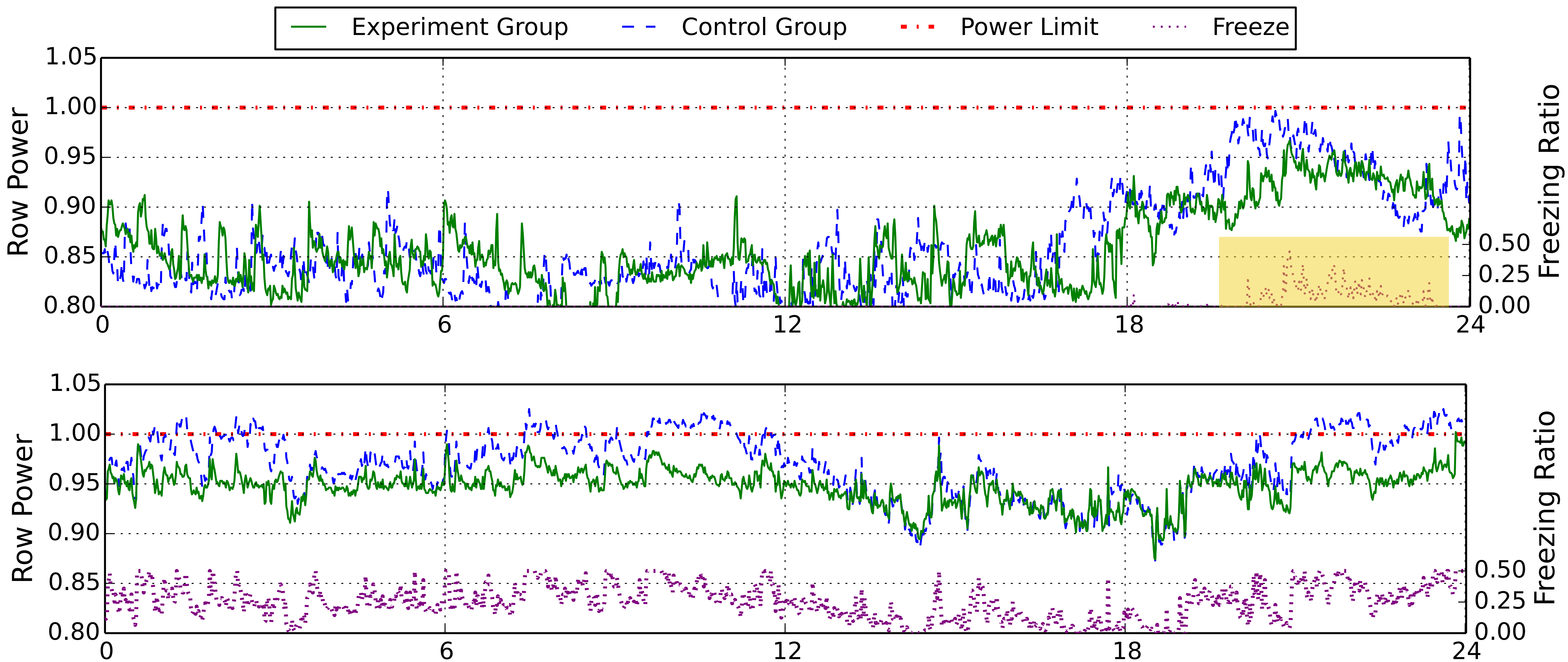
Over-provisioning ratio = 0.25



Effectiveness

Controlled experiments on production environment.

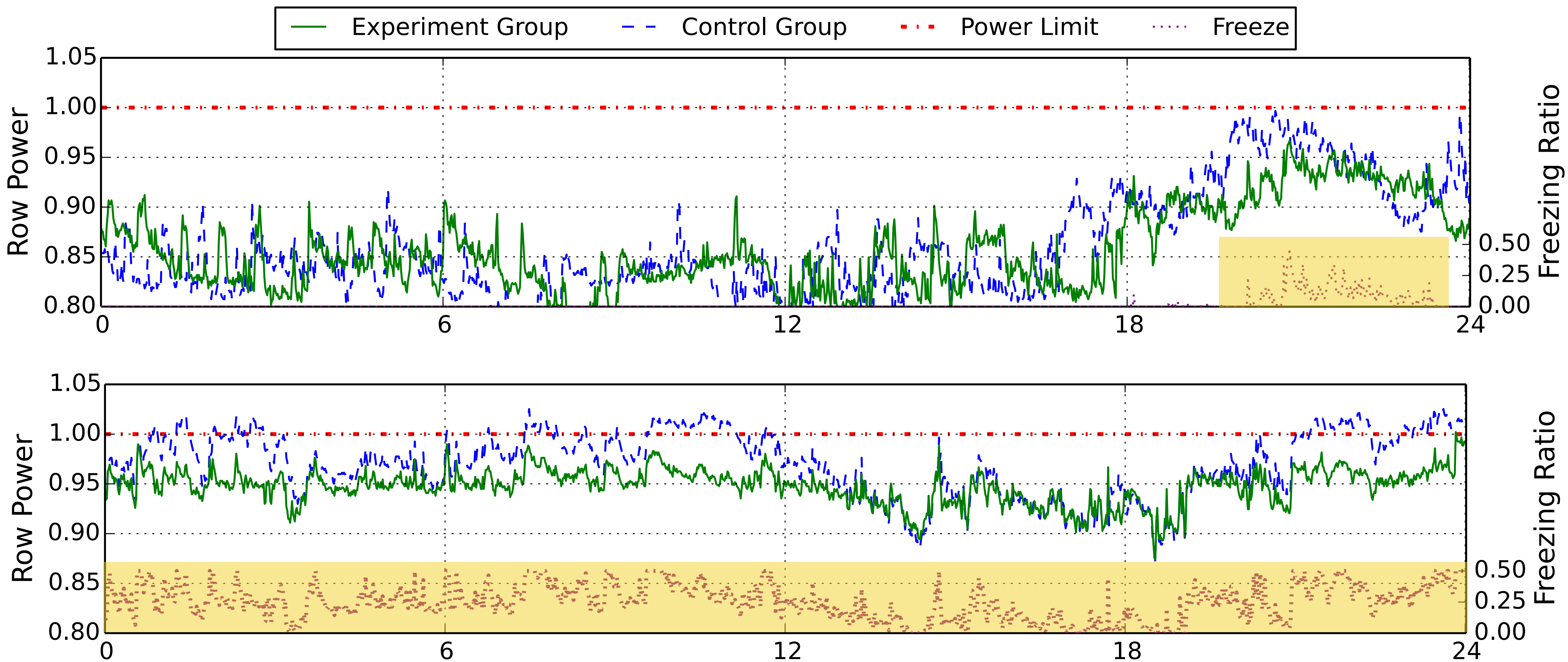
Over-provisioning ratio = 0.25



Effectiveness

Controlled experiments on production environment.

Over-provisioning ratio = 0.25



How to Decide Over-provisioning Ratio?

Throughput per Provisioned Watt (TPW):

$$\text{TPW} = \frac{\text{Throughput during time interval } T}{P \cdot T}$$

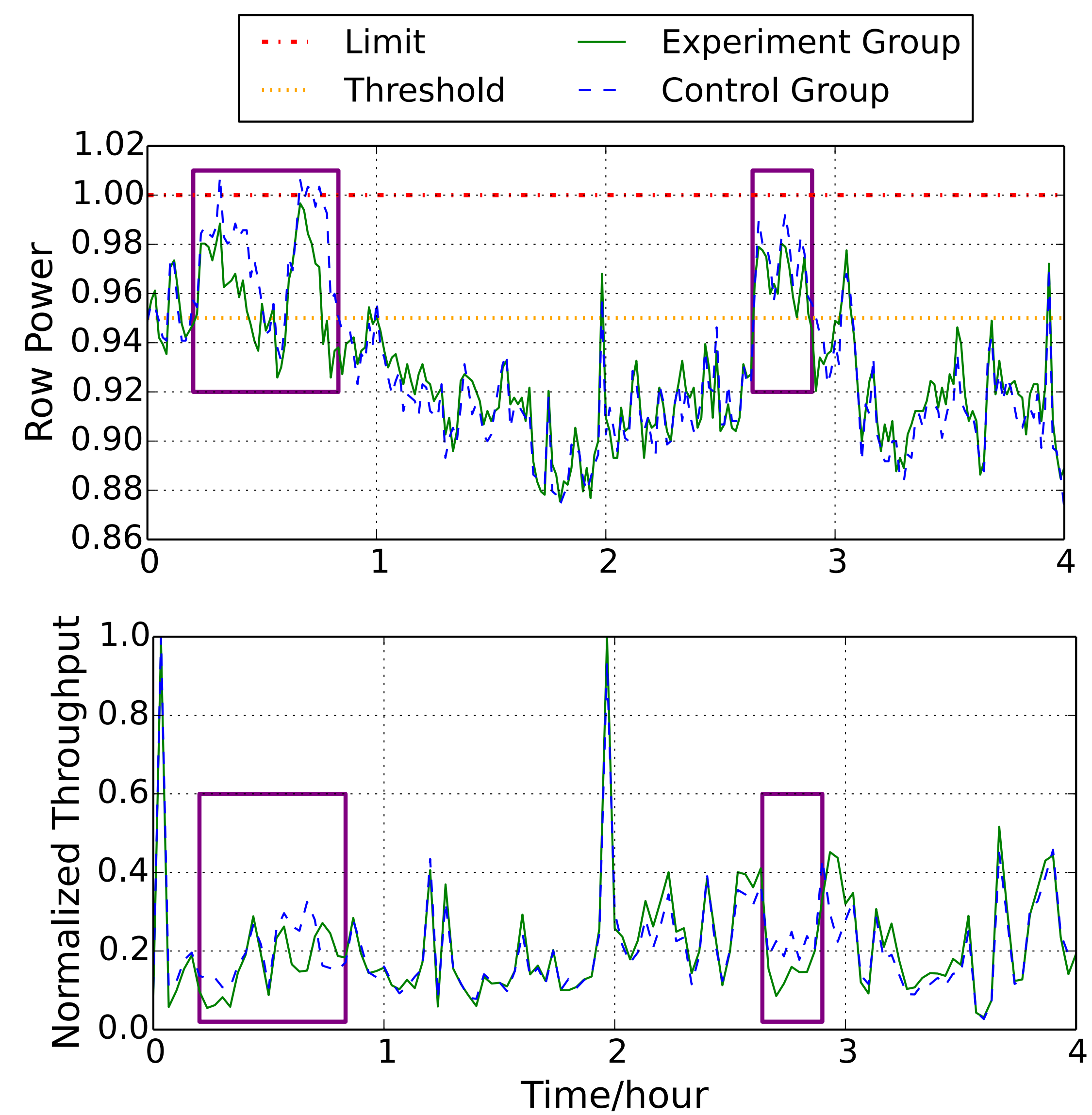
Gain in TPW:

$$G_{TPW} = r_T \cdot (1 + r_O) - 1$$

P Provisioned power

r_T Throughput ratio (≤1)

r_O Over-provisioning ratio (≥1)



How to Decide Over-provisioning Ratio?

Throughput per Provisioned Watt (TPW):

$$\text{TPW} = \frac{\text{Throughput during time interval } T}{P \cdot T}$$

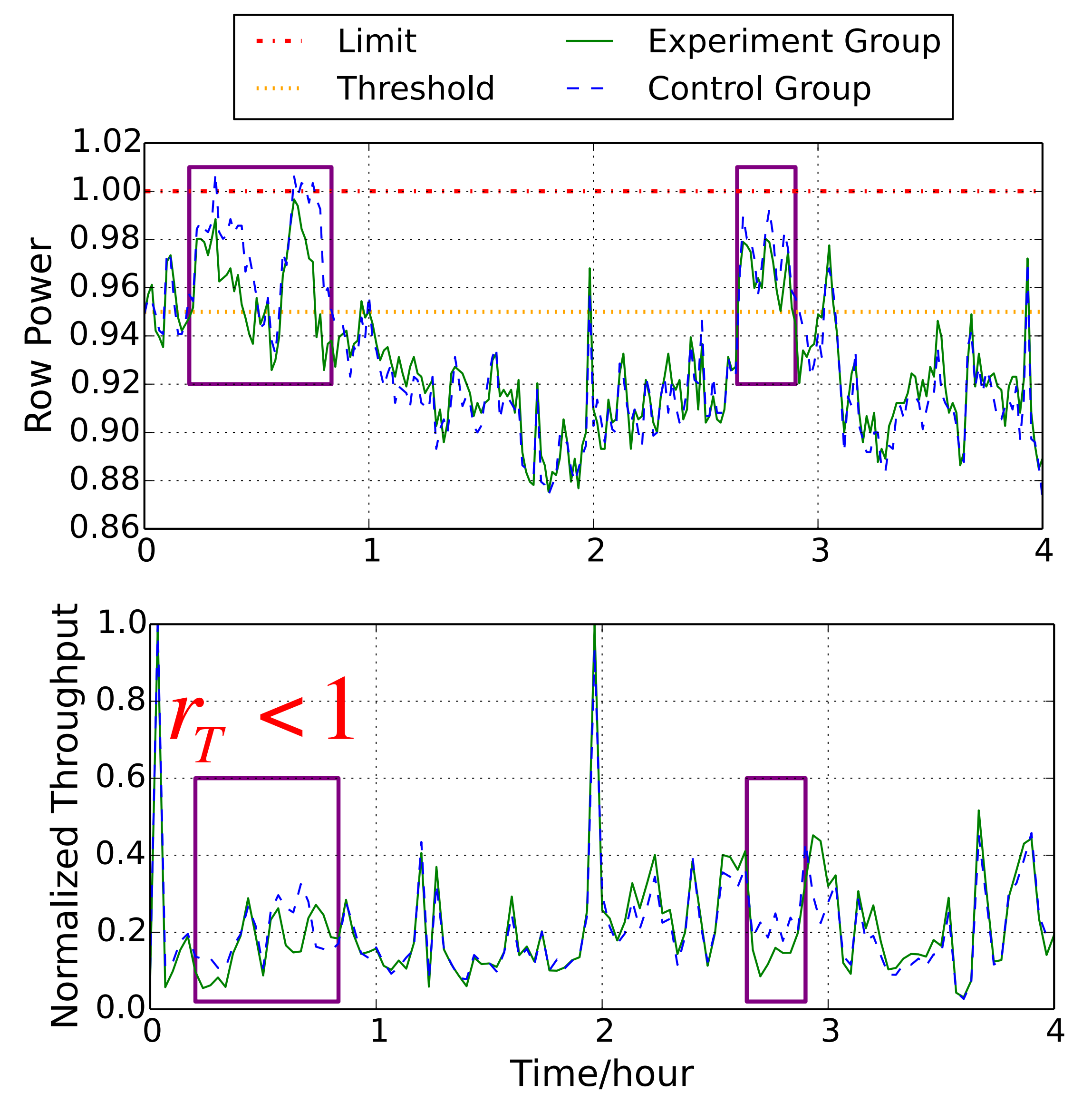
Gain in TPW:

$$G_{TPW} = r_T \cdot (1 + r_O) - 1$$

P Provisioned power

r_T Throughput ratio (≤1)

r_O Over-provisioning ratio (≥1)



How to Decide Over-provisioning Ratio?

Throughput per Provisioned Watt (TPW):

$$\text{TPW} = \frac{\text{Throughput during time interval } T}{P \cdot T}$$

Gain in TPW:

$$G_{TPW} = r_T \cdot (1 + r_O) - 1$$

By emulations we found $G_{TPW} = 0.149$ when $r_O = 0.17$.

P Provisioned power

r_T Throughput ratio (≤1)

r_O Over-provisioning ratio (≥1)

Conclusion

- Admission control to statistically influencing new job placement
- Minimal APIs (freeze/ unfreeze)
- Simple dynamic system control
- Controlled experiment

Avoid performance degradation.

Decouple the power control module and the complicated scheduler.

Tolerate inaccuracy.

Build and evaluate system model in production environment without disturbing it too much.

Conclusion

• Admission control to statistically influencing new job placement	→	Avoid performance degradation.
• Minimal APIs (freeze/ unfreeze)	→	Decouple the power control module and the complicated scheduler.
• Simple dynamic system control	→	Tolerate inaccuracy.
• Controlled experiment	→	Build and evaluate system model in production environment without disturbing it too much.

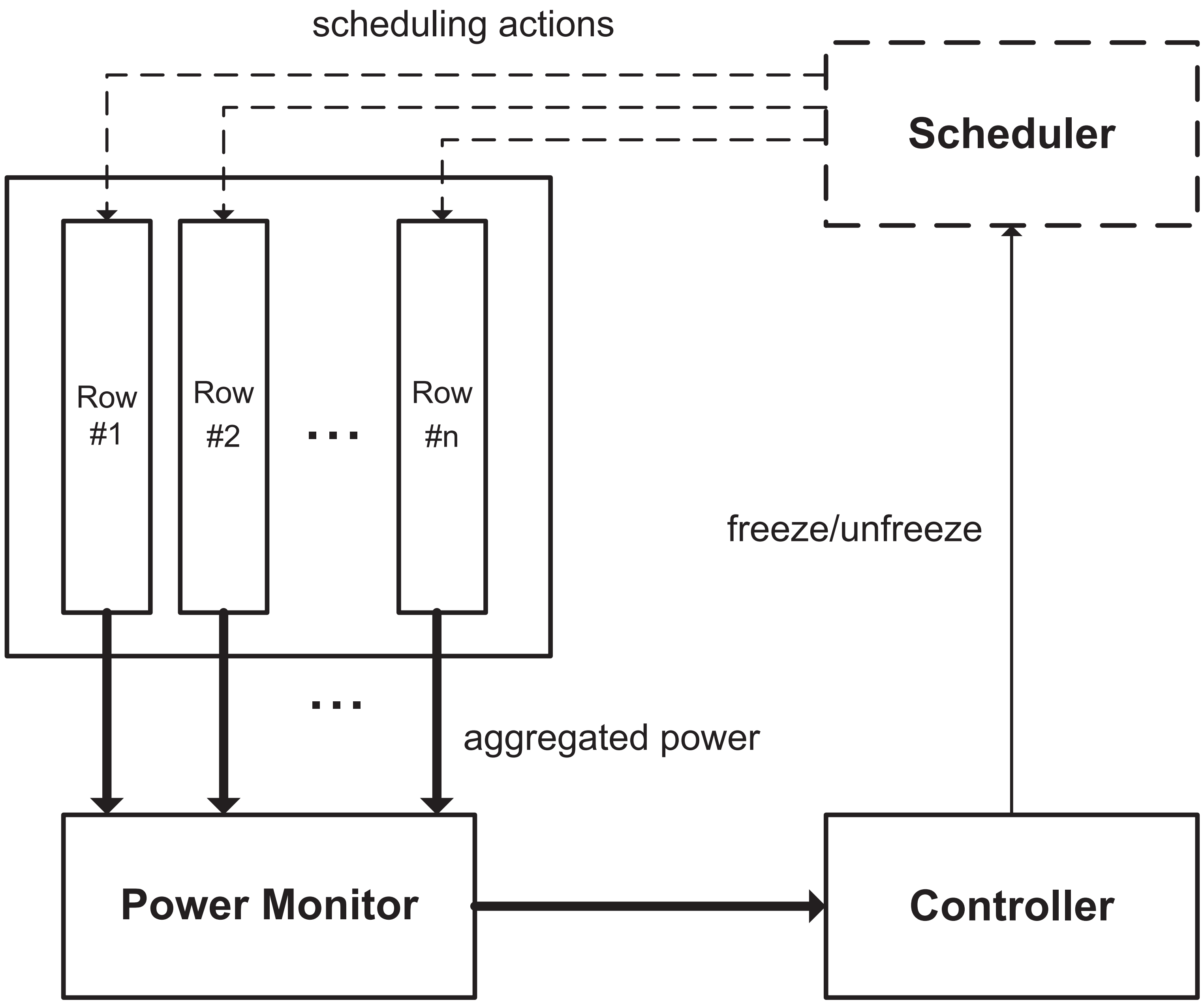
Q&A

Outline:

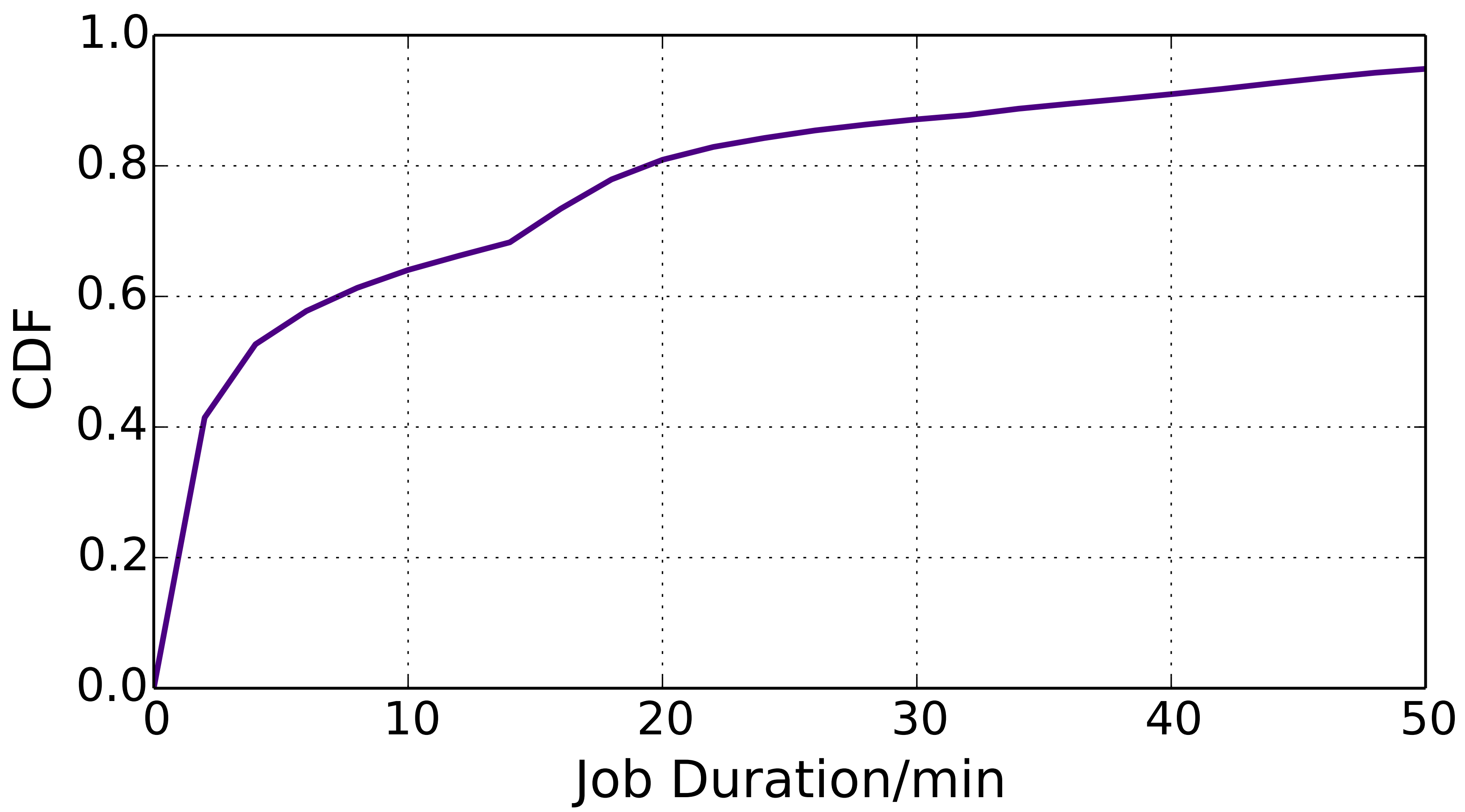
- Power over-provisioning motivation
- Ideas of statistical power control
- Dynamic Control model
- Controlled experiment design
- Effectiveness
- Deciding over-provisioning ratio
- Conclusion

Backup Slides

Ampere Architecture



Job Durations



G_{TPW} under Different r_O

#	r_O	P_{mean}	P_{max}	u_{mean}	r_T	G_{TPW}
1	0.25	0.903	1.028	0.019	0.953	19.70%
2		0.931	1.062	0.134	0.941	17.60%
3		**0.936**	**1.062**	**0.152**	**0.885**	**10.60%**
4		0.927	1.061	0.196	0.835	4.30%
5	0.21	0.786	0.913	0	1.0	20.70%
6		0.835	0.982	0.0016	1.0	20.70%
7		0.894	1.000	0.009	0.979	18.20%
8		**0.903**	**1.036**	**0.11**	**0.88**	**6.20%**
9	0.17	0.836	0.931	0	1.0	17%
10		0.839	0.926	0	1.0	17%
11		**0.908**	**0.992**	**0.07**	**0.984**	**14.90%**
12		0.938	1.004	0.12	0.904	5.50%
13	0.13	0.847	0.969	0	1.0	13%

Fix $r_O, P_{mean} \nearrow \Rightarrow u_{mean} \nearrow \Rightarrow r_T \searrow \Rightarrow G_{TPW} \searrow$

$r_O \nearrow \Rightarrow u_{mean} \nearrow$ $\quad$ $G_{TPW} < r_O$