CREDENCE

Augmenting Datacenter Switch Buffer Sharing with ML Predictions

Vamsi Addanki, Maciej Pacut, Stefan Schmid

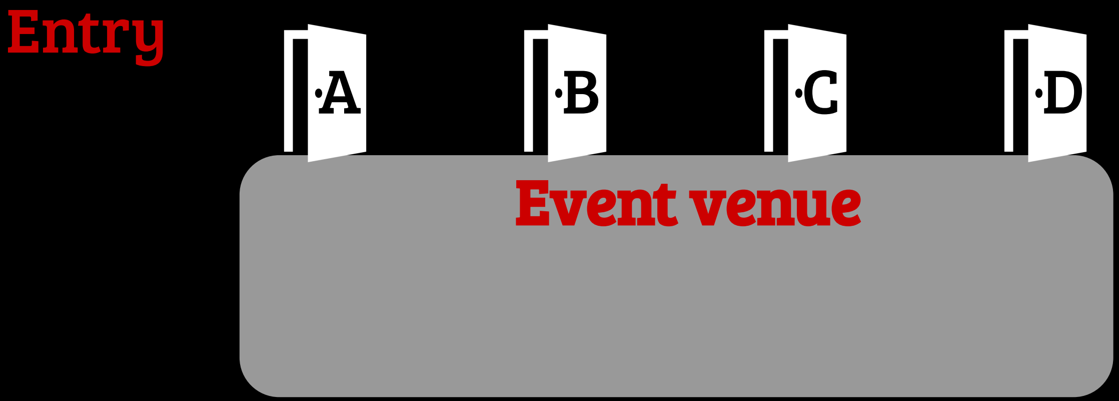
Technische Universität Berlin

European Research Council
Established by the European Commission

Let’s Play a Game

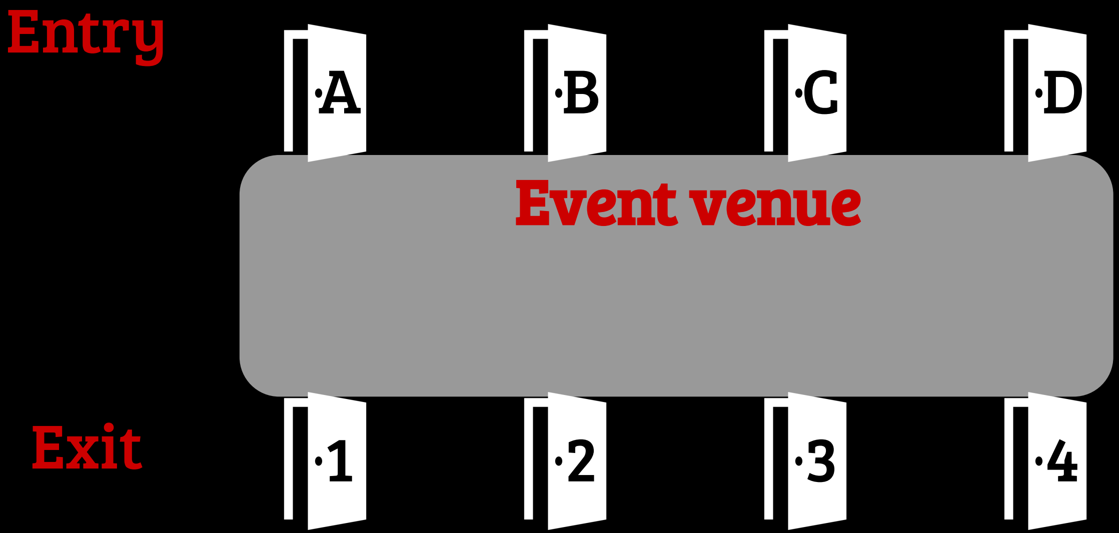
Event venue

Let’s Play a Game



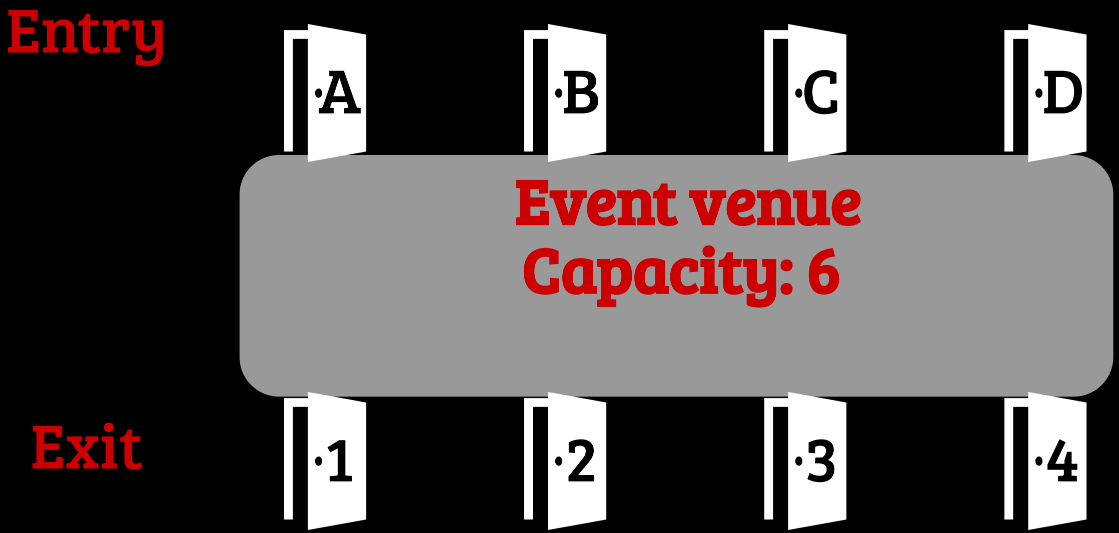
CREDENCE

Let's Play a Game

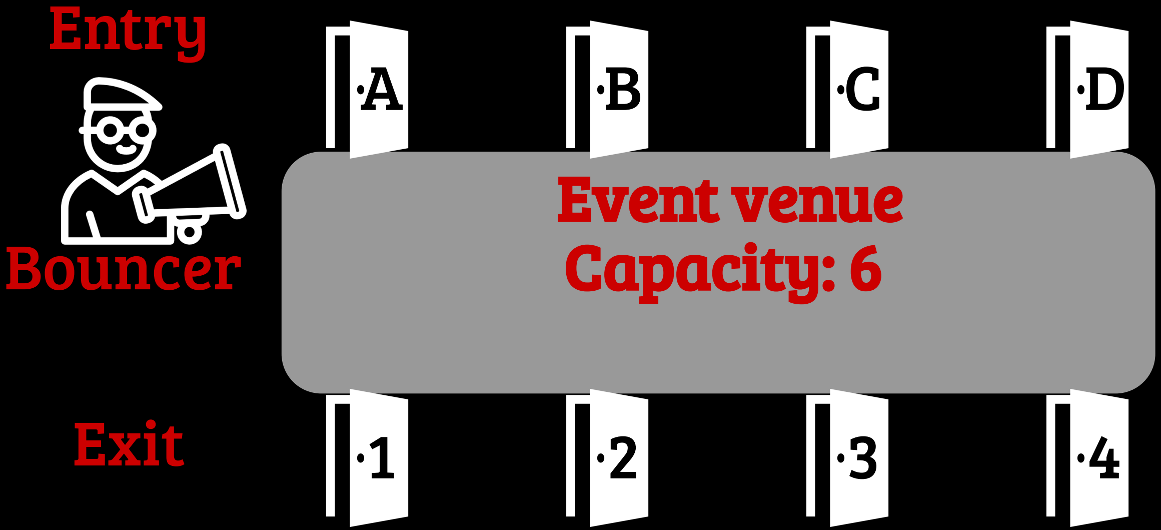


CREDENCE

Let's Play a Game

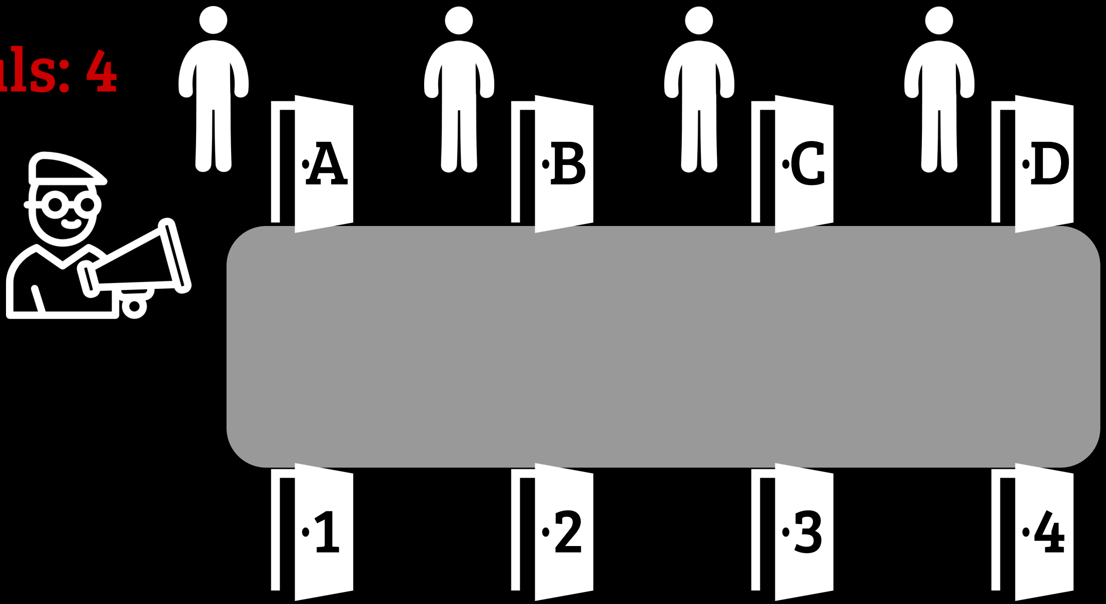


Let's Play a Game



Let's Play a Game

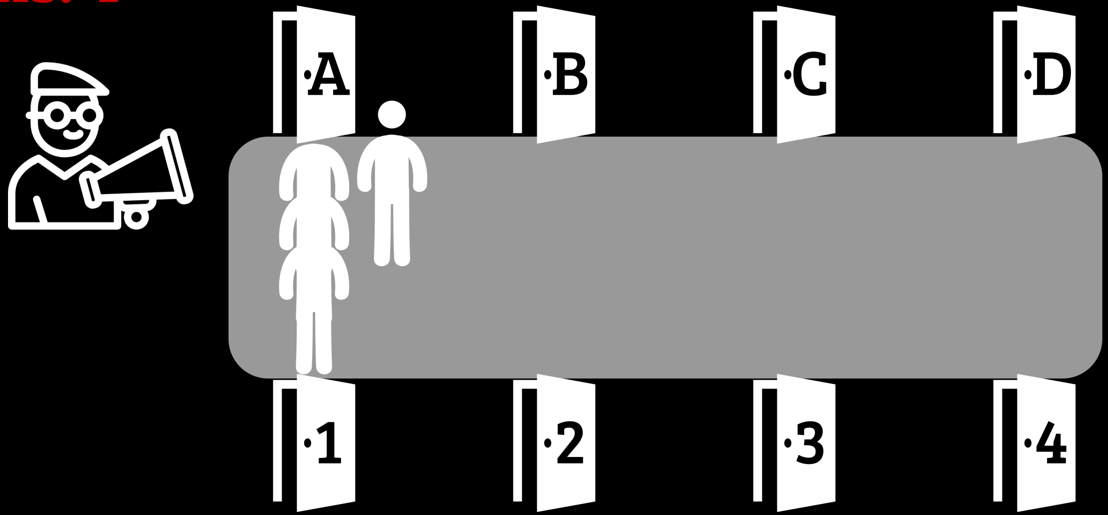
Arrivals: 4



Capacity: 6

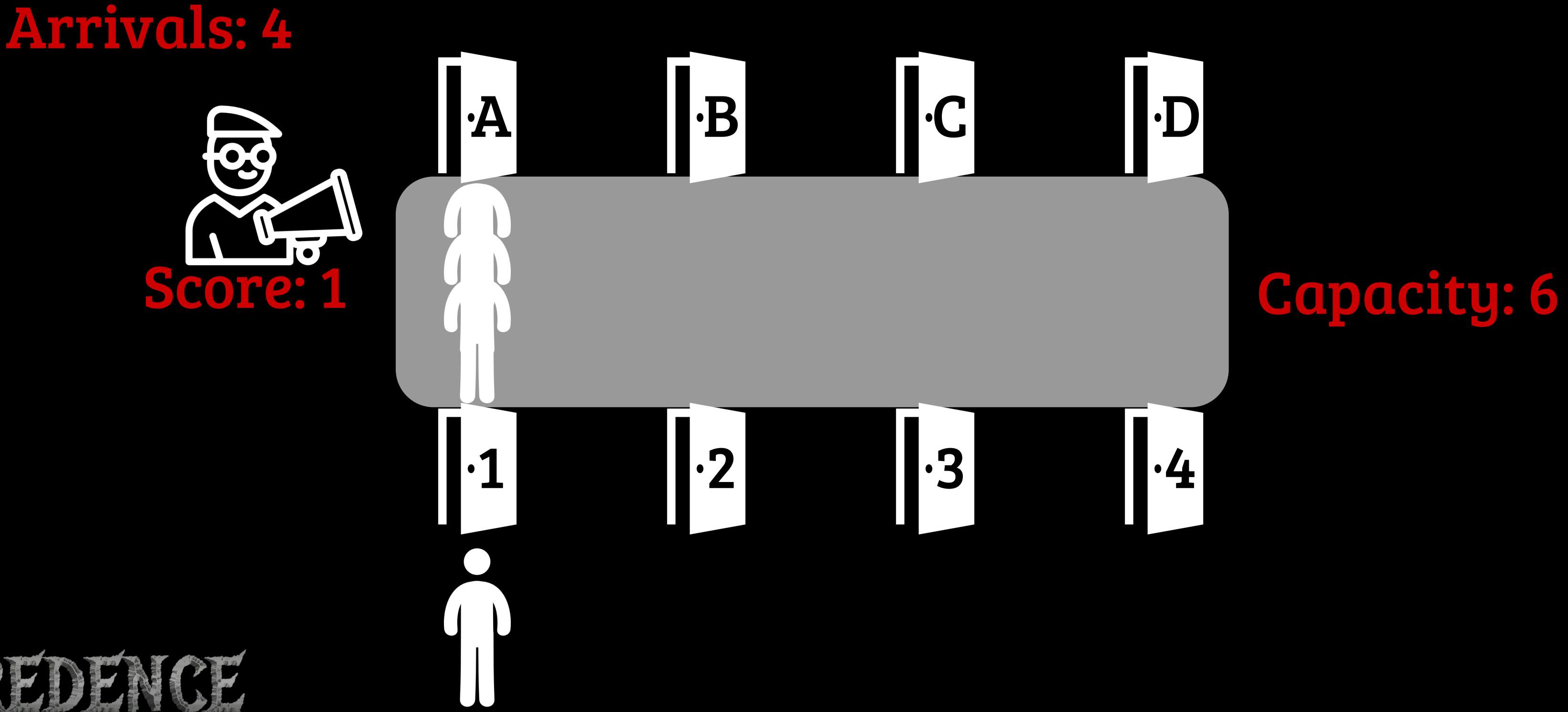
Let’s Play a Game

Arrivals: 4

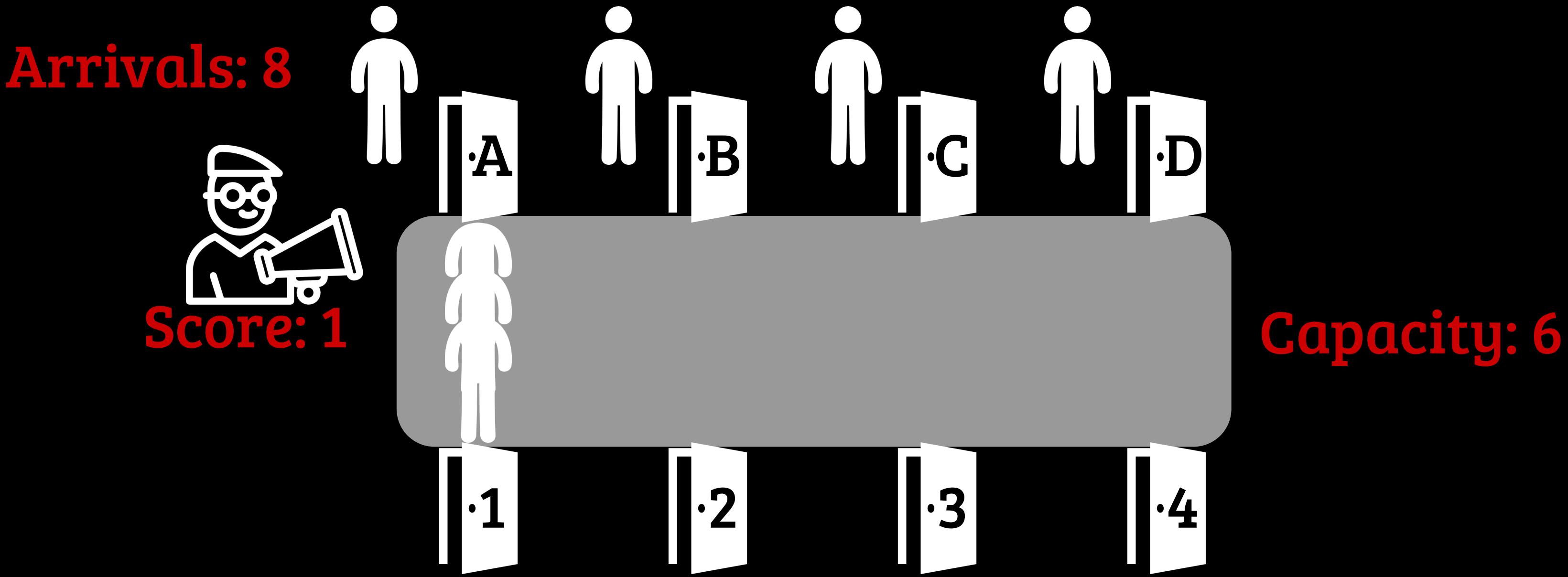


Capacity: 6

Let's Play a Game

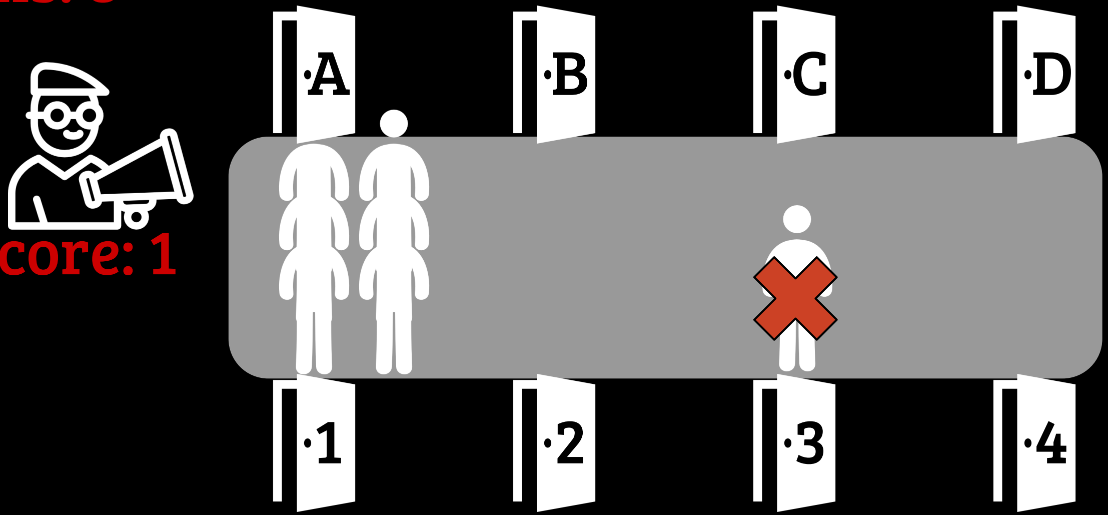


Let's Play a Game



Let's Play a Game

Arrivals: 8

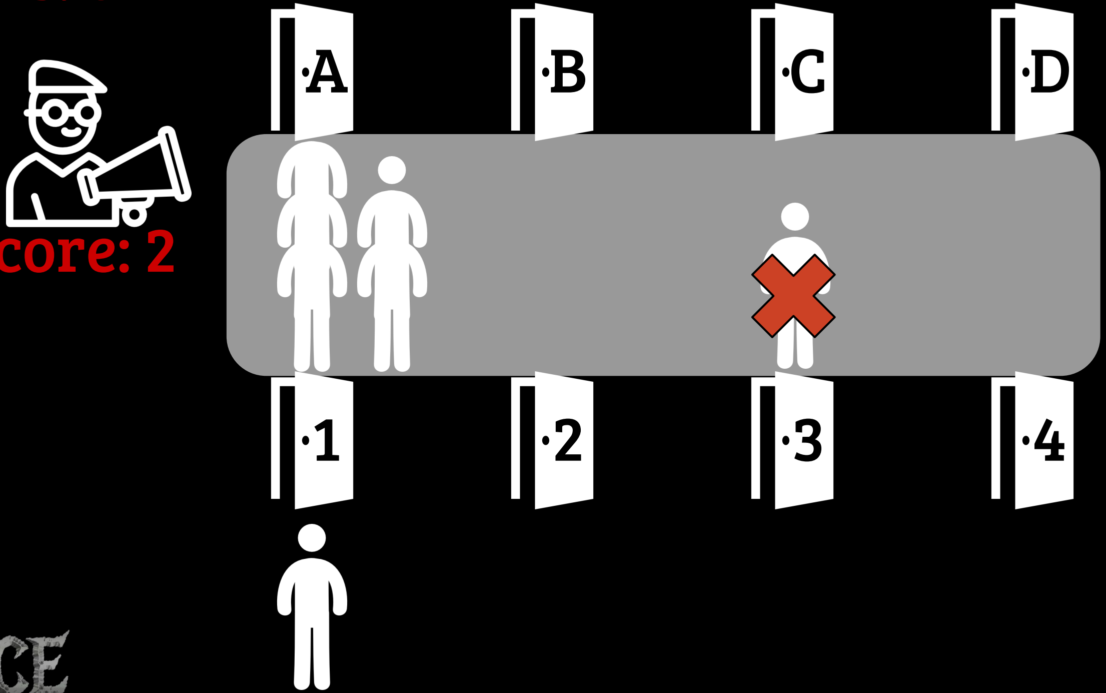


Score: 1

Capacity: 6

Let’s Play a Game

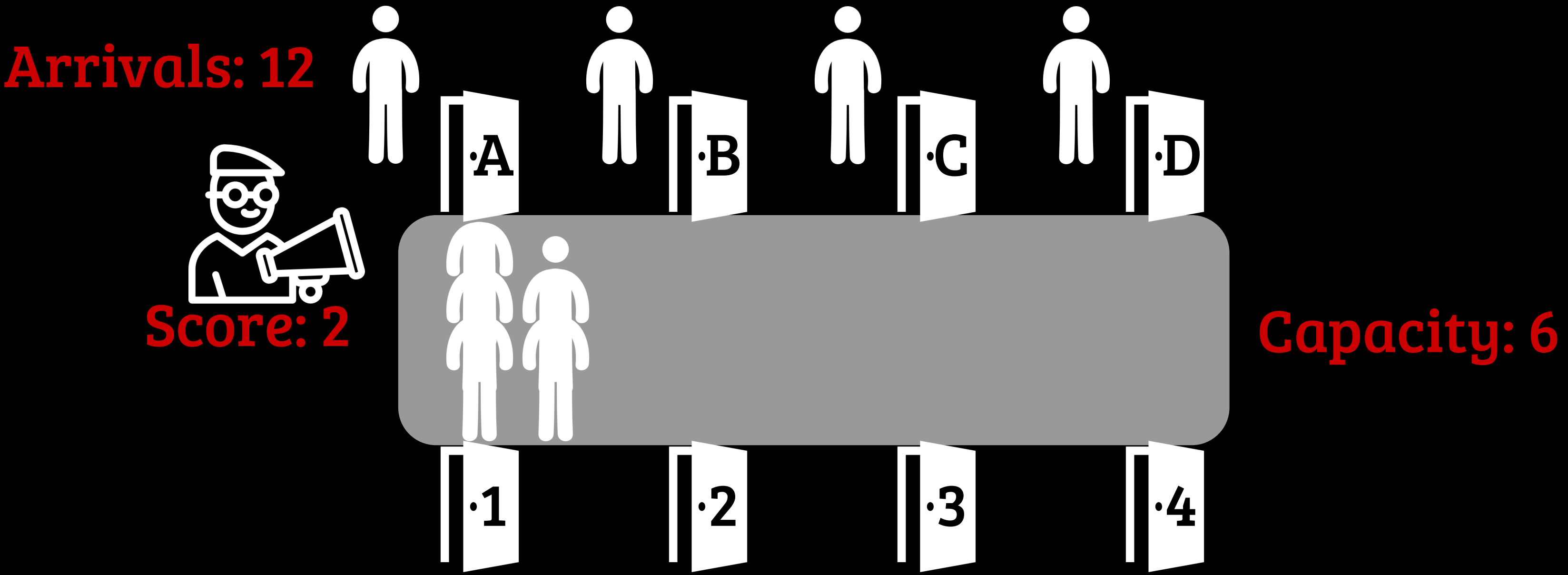
Arrivals: 8



Score: 2

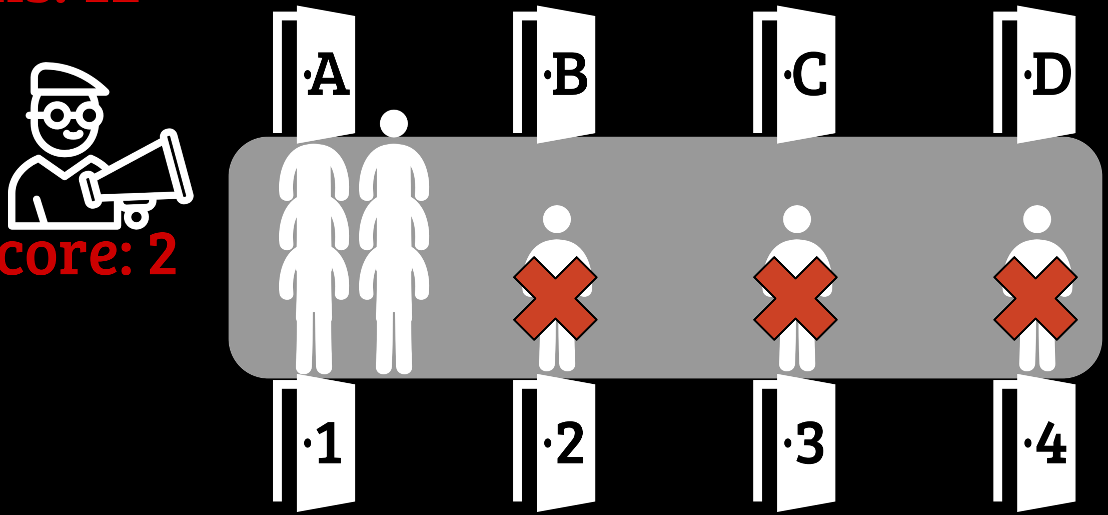
Capacity: 6

Let’s Play a Game



Let's Play a Game

Arrivals: 12

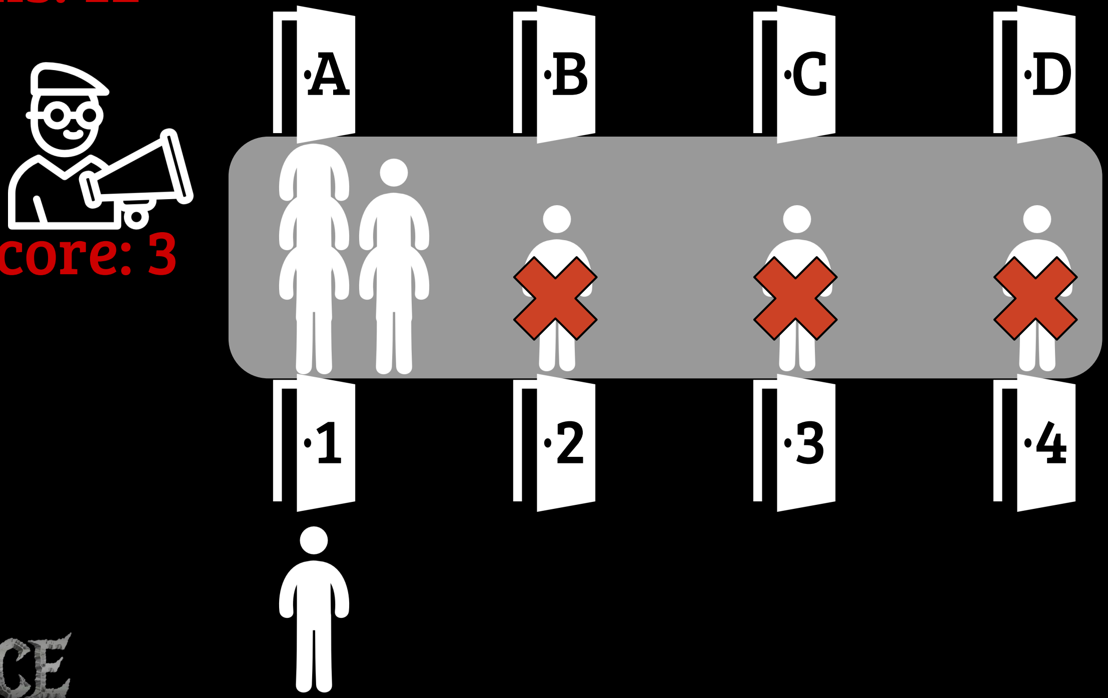


Score: 2

Capacity: 6

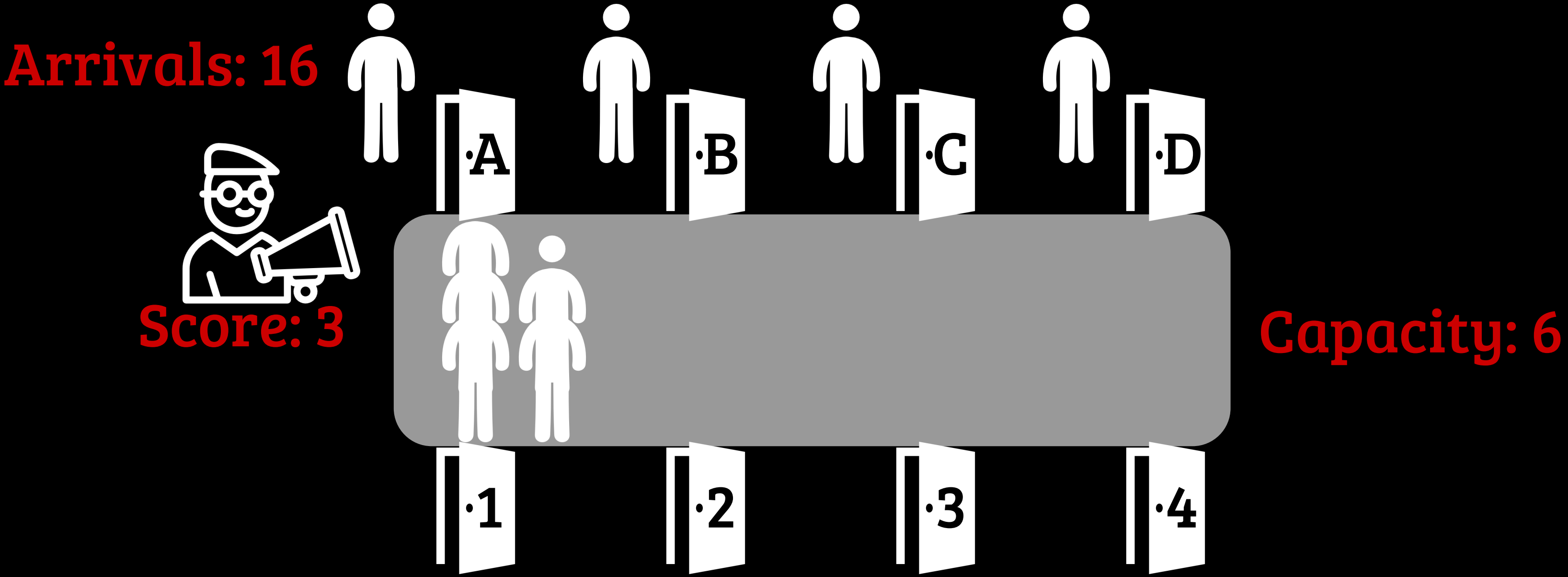
Let's Play a Game

Arrivals: 12

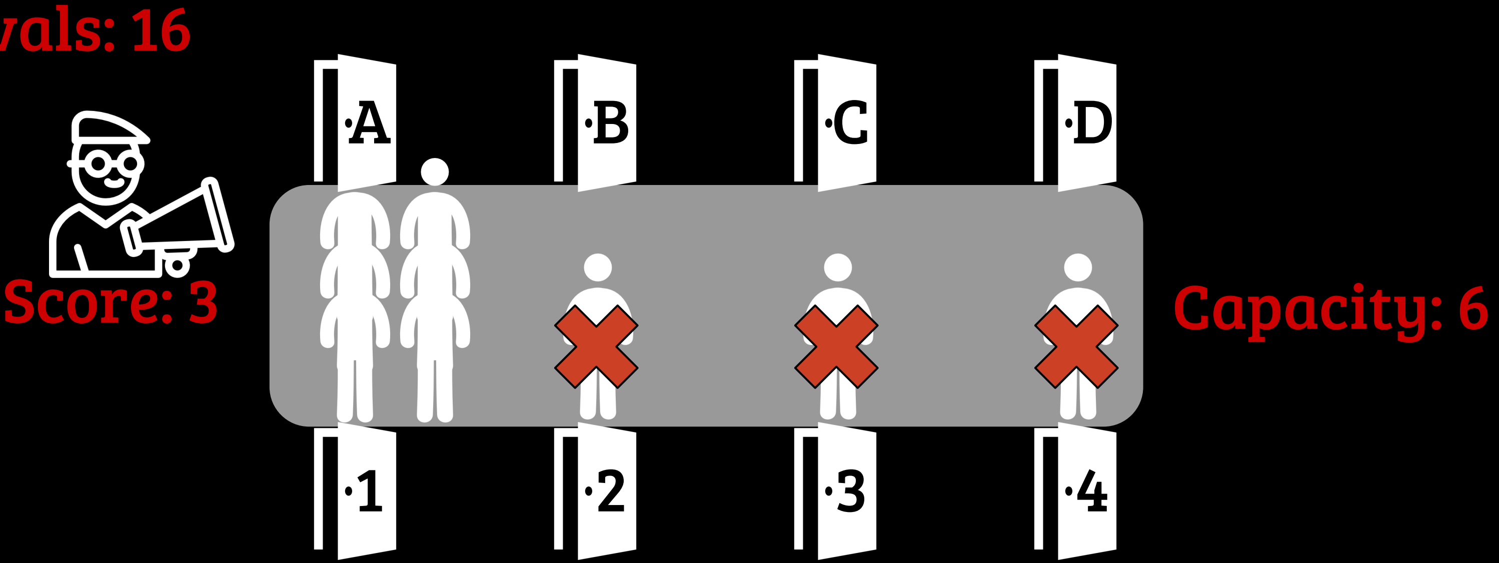


Capacity: 6

Let’s Play a Game



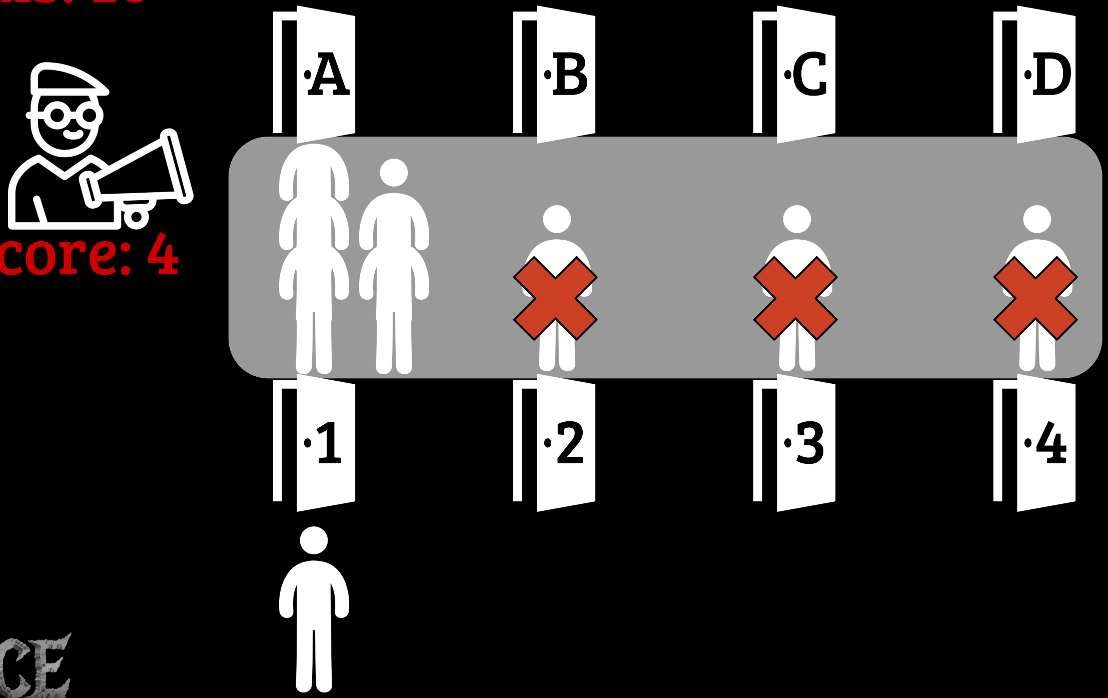
Let's Play a Game



CREDENCE

Let's Play a Game

Arrivals: 16

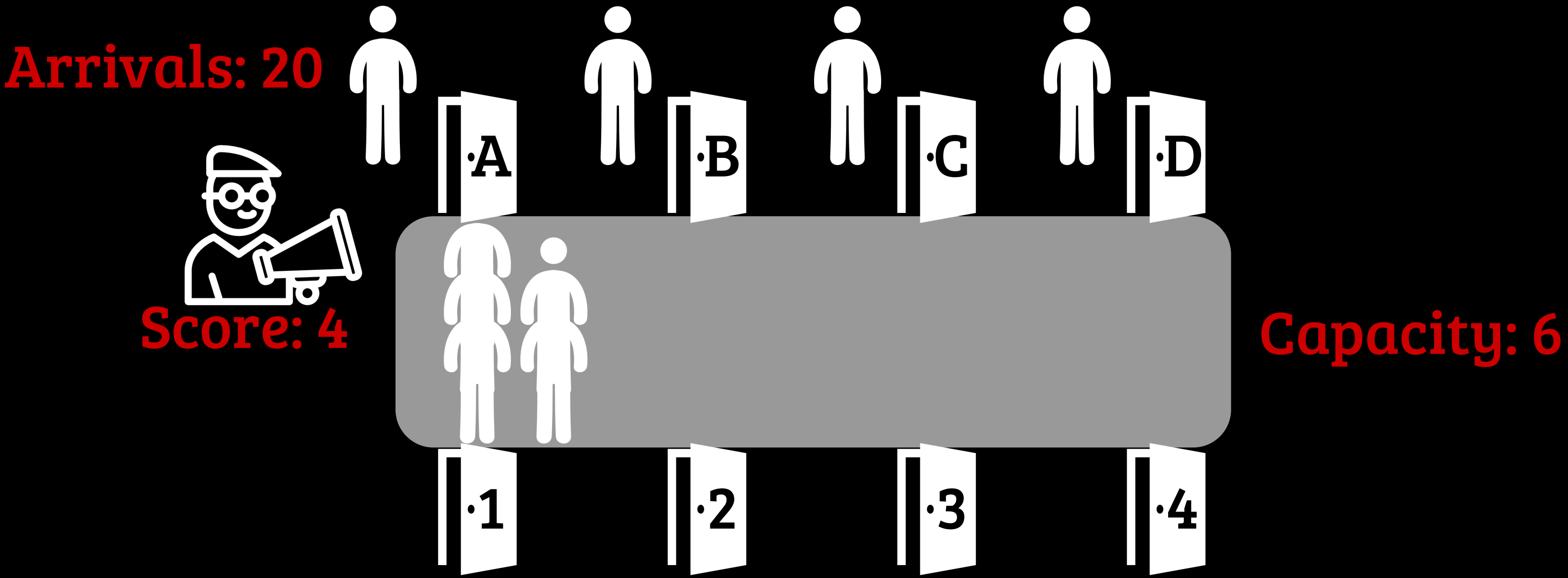


Score: 4

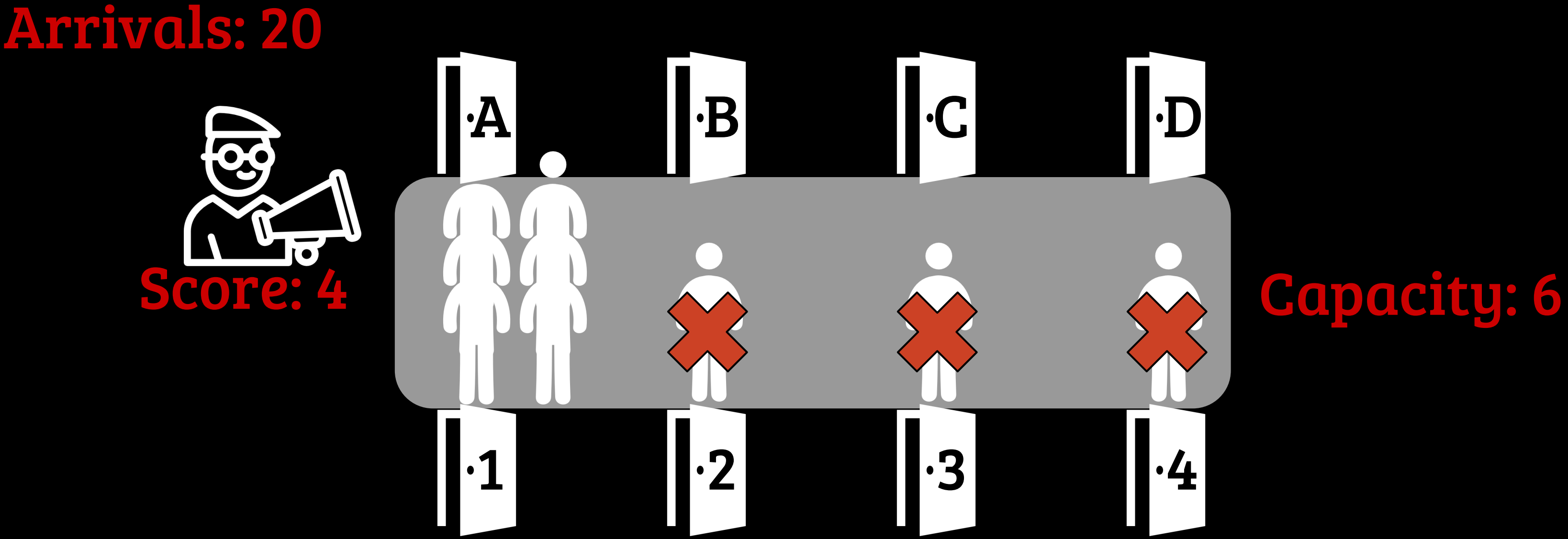
Capacity: 6

CREDENCE

Let's Play a Game

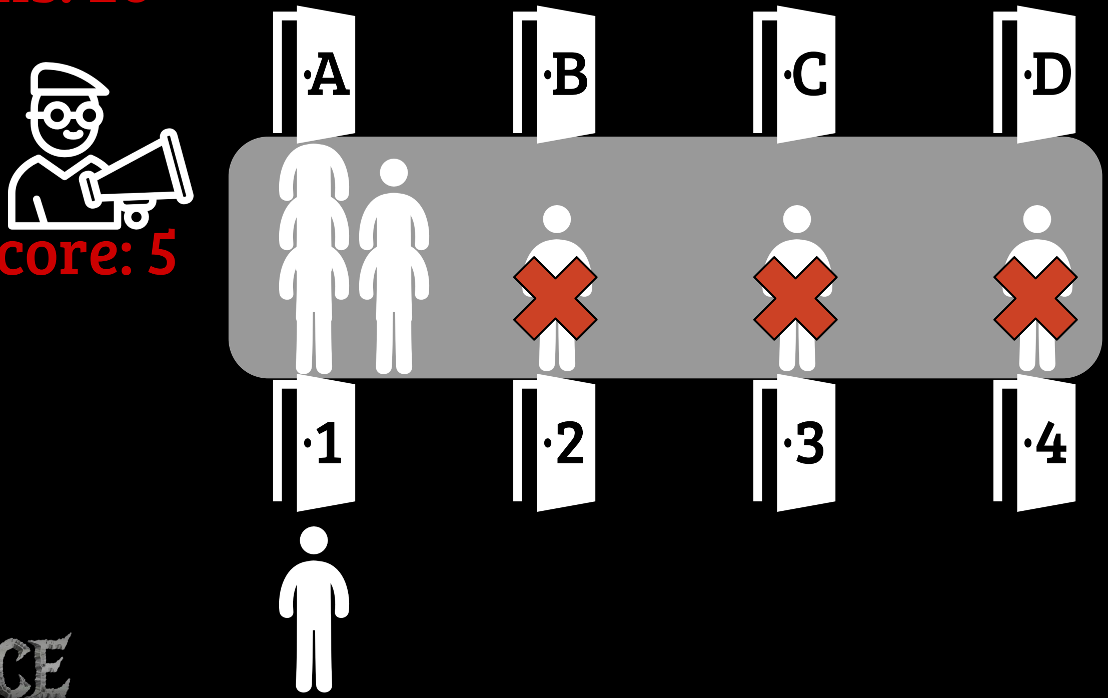


Let's Play a Game



Let's Play a Game

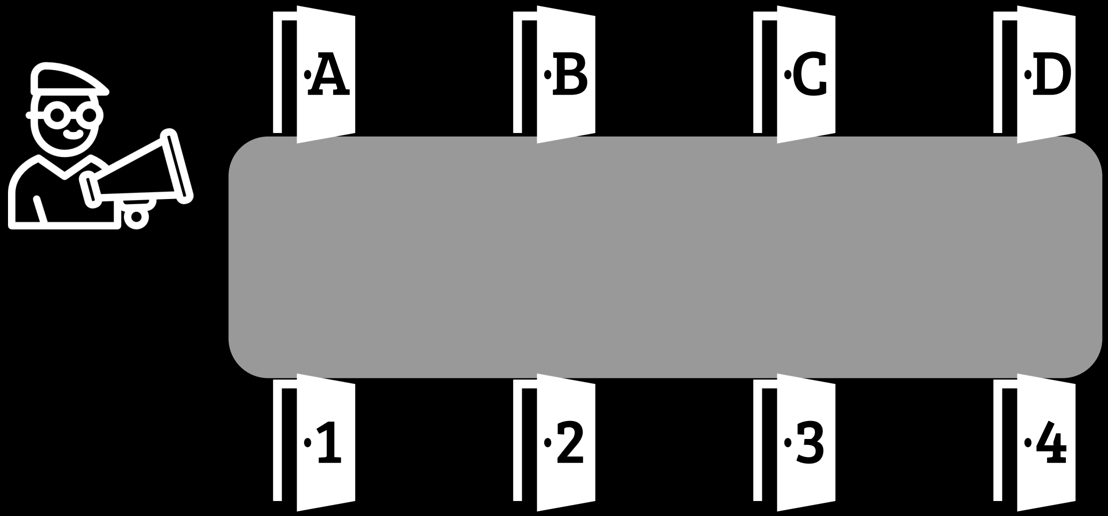
Arrivals: 20



Score: 5

Capacity: 6

Let's Play a Game



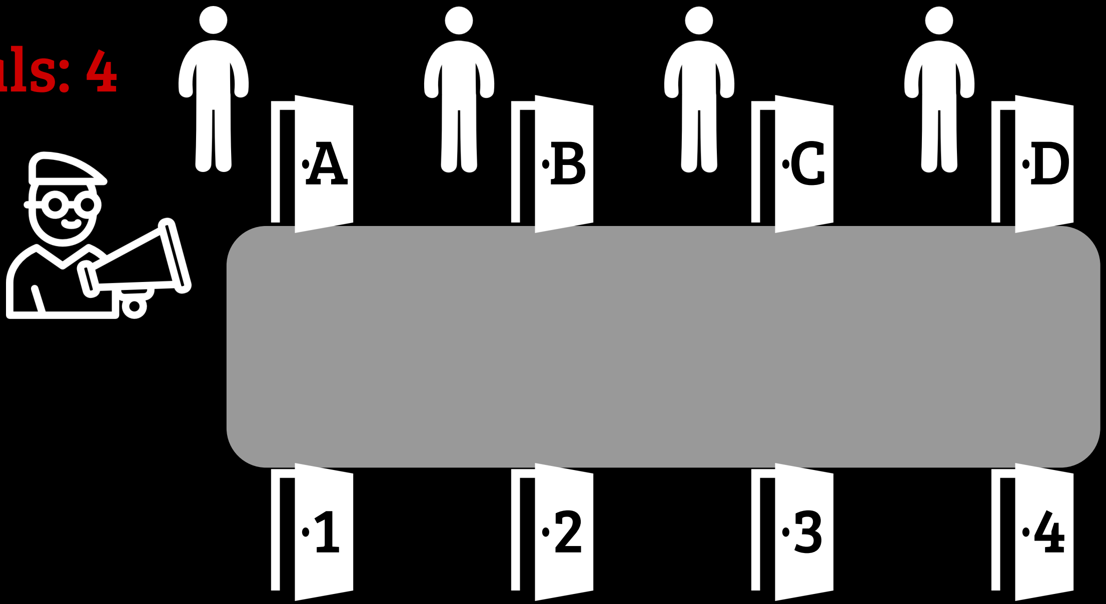
Previous Game

Arrivals: 20
Score: 5

Capacity: 6

Let's Play a Game

Arrivals: 4



Previous Game

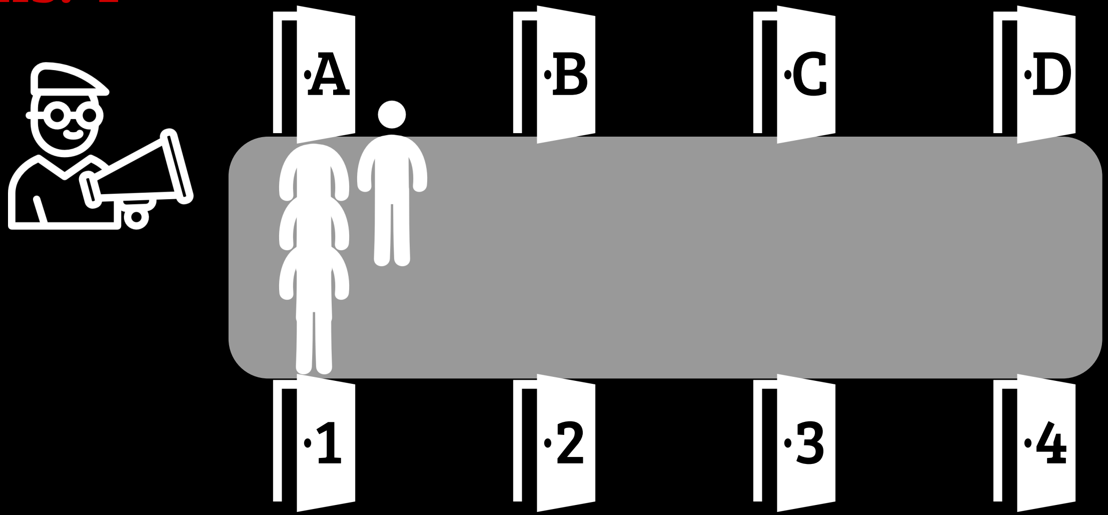
Arrivals: 20

Score: 5

Capacity: 6

Let’s Play a Game

Arrivals: 4



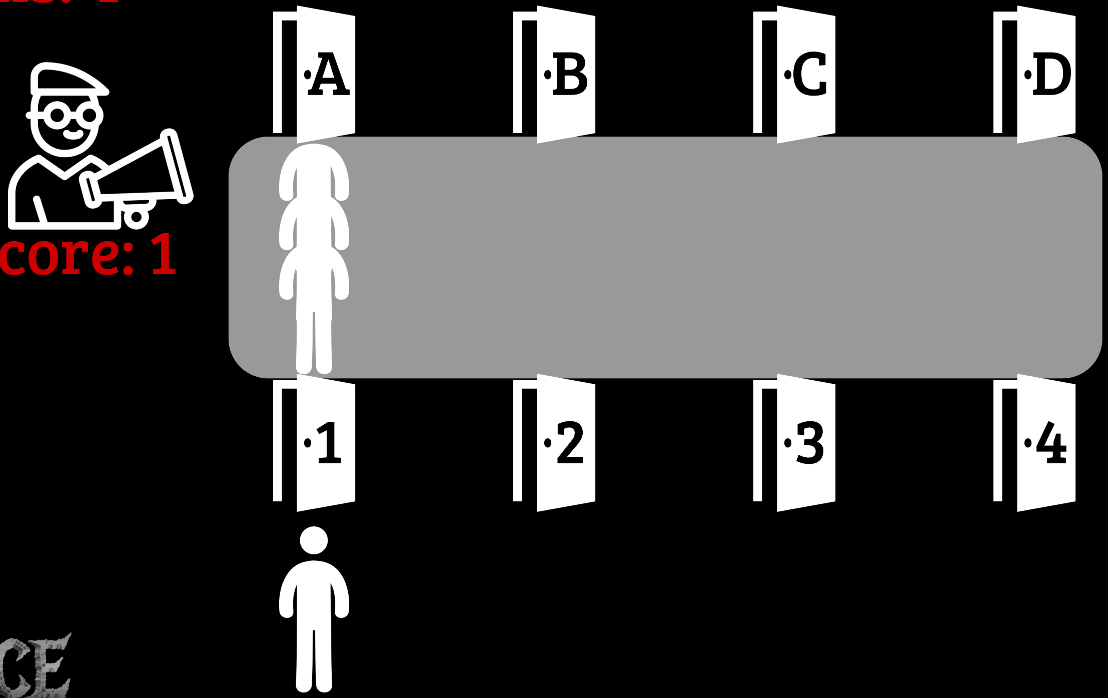
Previous Game

Arrivals: 20
Score: 5

Capacity: 6

Let's Play a Game

Arrivals: 4



Score: 1

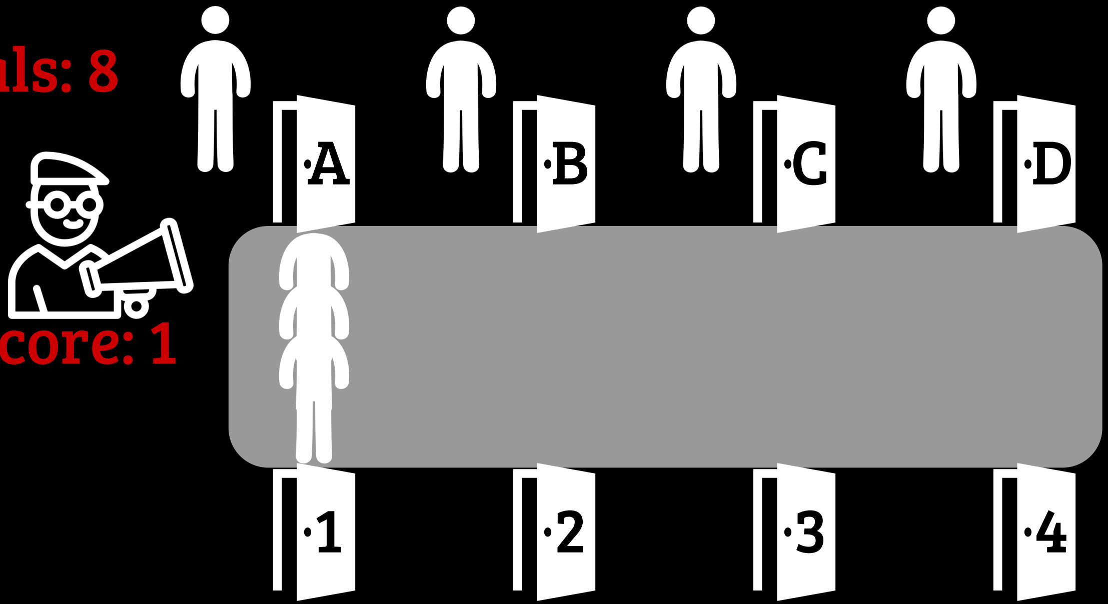
Capacity: 6

Previous Game

Arrivals: 20
Score: 5

Let's Play a Game

Arrivals: 8



Score: 1

Capacity: 6

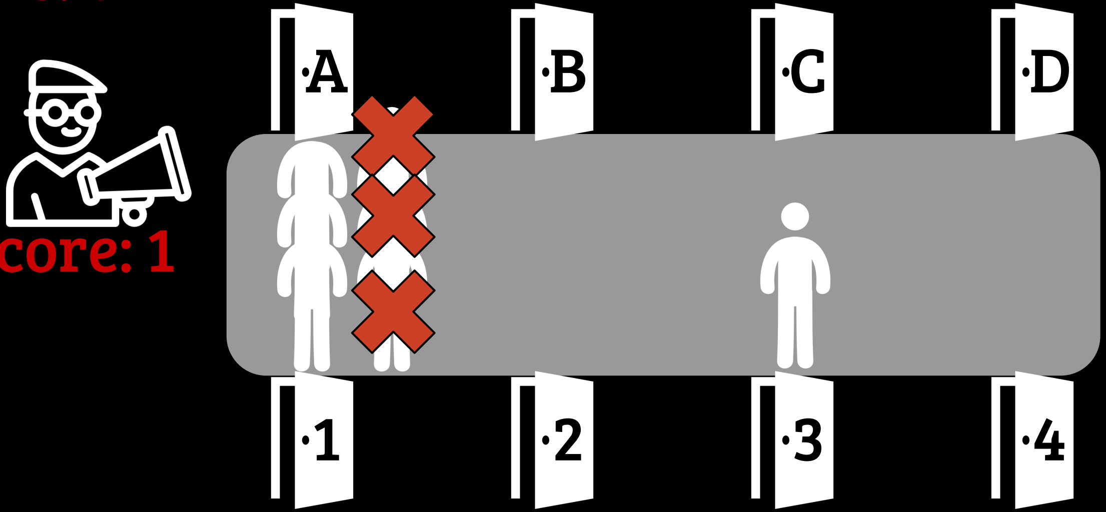
Previous Game

Arrivals: 20

Score: 5

Let's Play a Game

Arrivals: 8



Score: 1

Capacity: 6

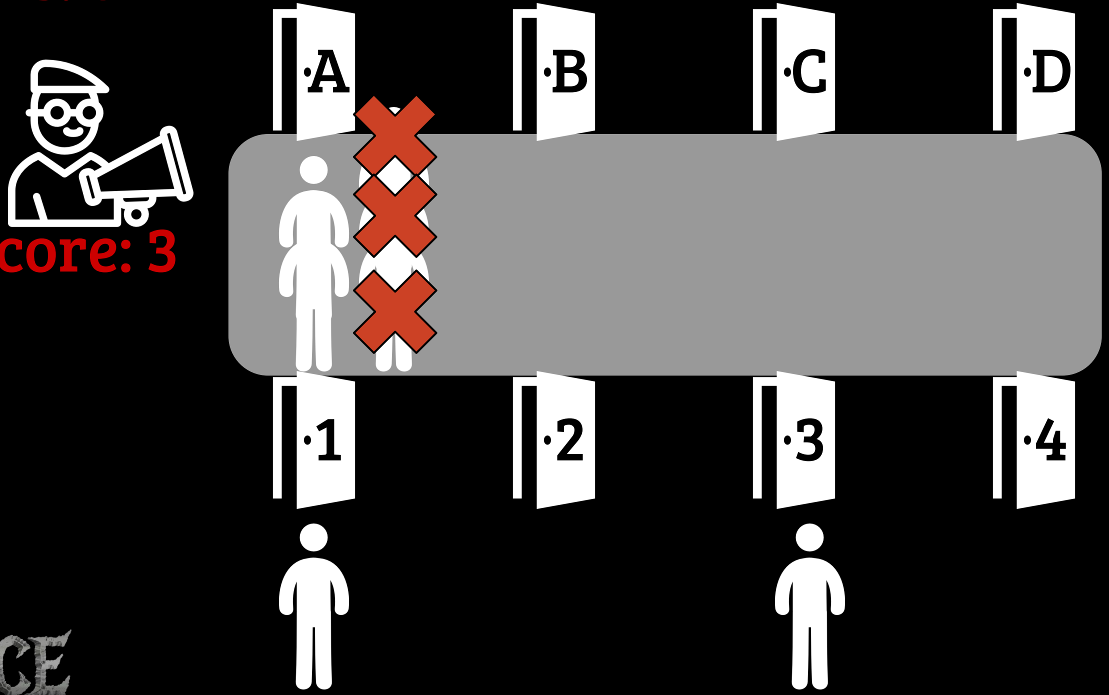
Previous Game

Arrivals: 20

Score: 5

Let’s Play a Game

Arrivals: 8



Score: 3

Capacity: 6

Previous Game

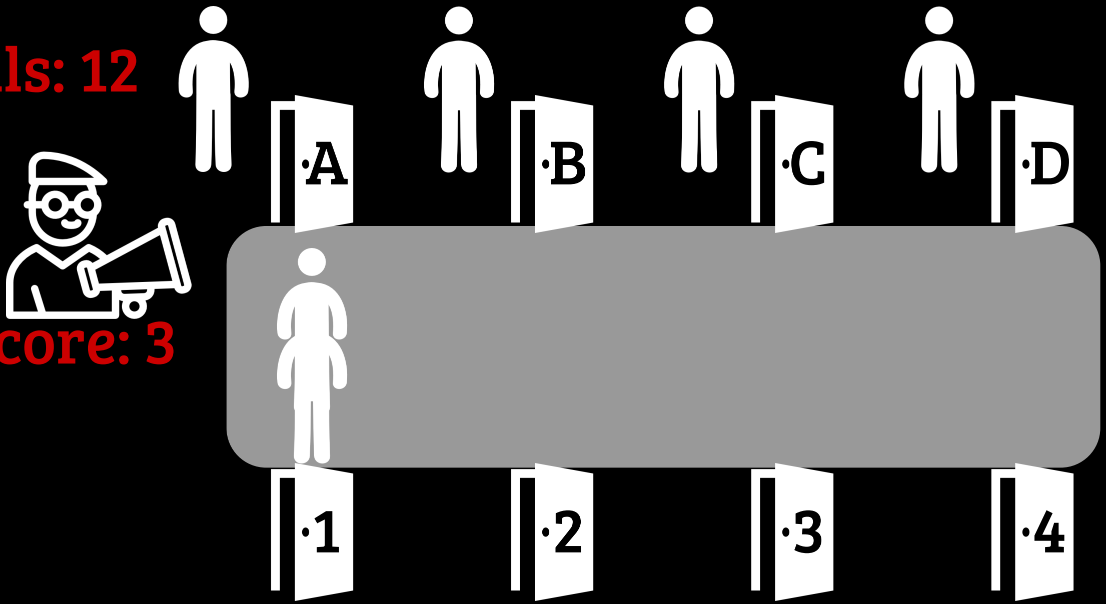
Arrivals: 20

Score: 5

CREDENCE

Let's Play a Game

Arrivals: 12



Score: 3

Capacity: 6

Previous Game

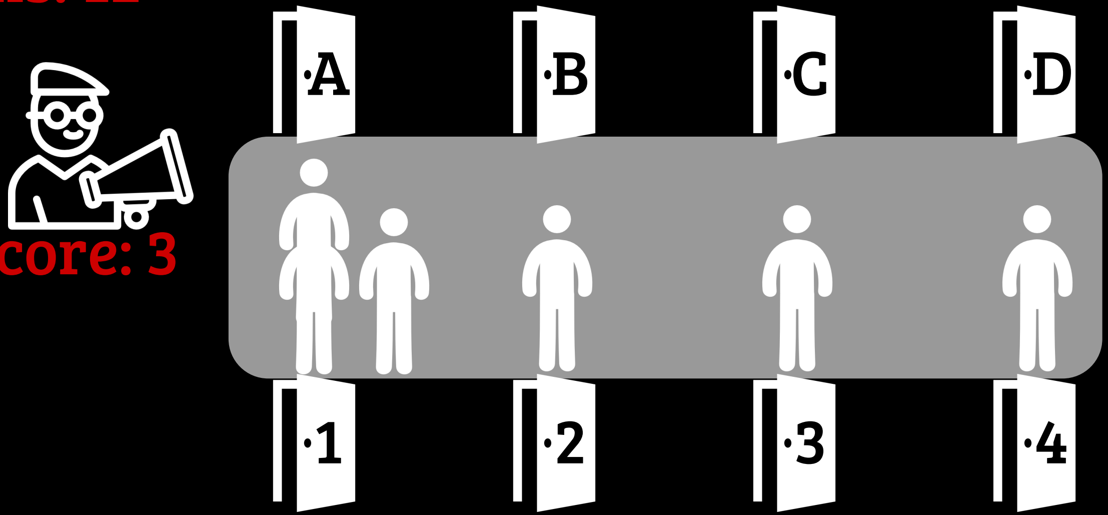
Arrivals: 20

Score: 5

CREDENCE

Let's Play a Game

Arrivals: 12



Score: 3

Capacity: 6

Previous Game

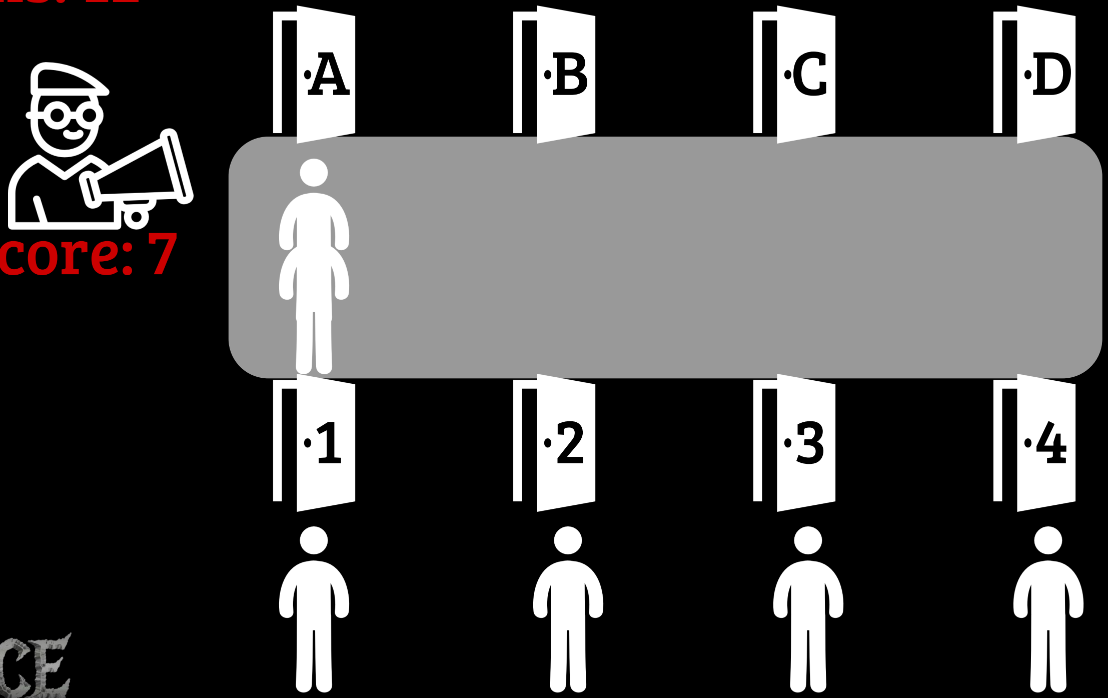
Arrivals: 20

Score: 5

Let's Play a Game

Arrivals: 12

Score: 7



Capacity: 6

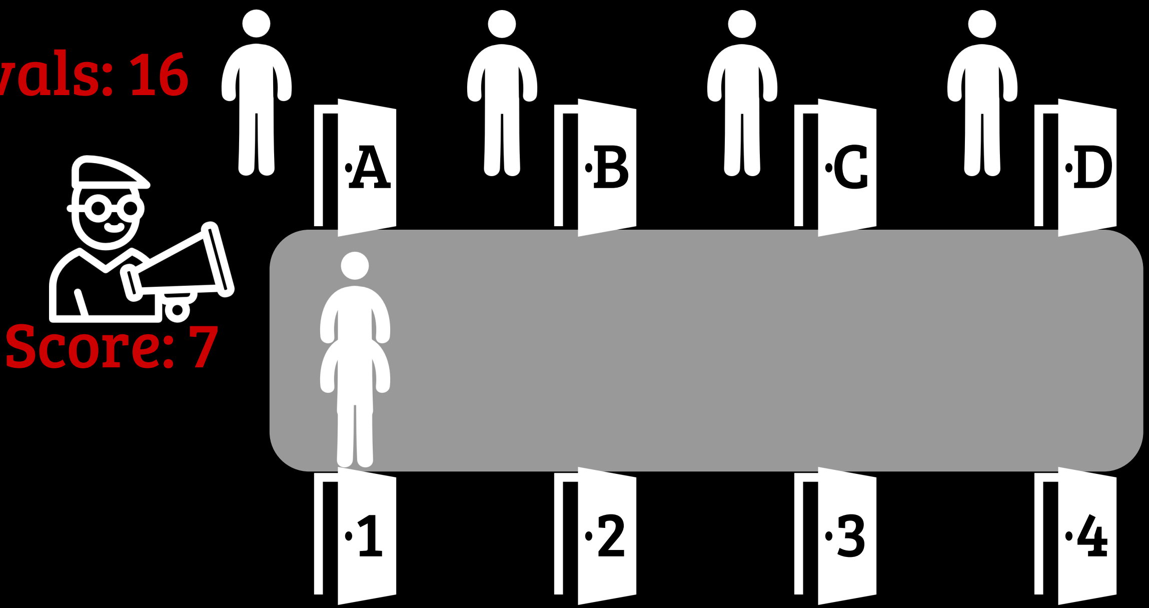
Previous Game

Arrivals: 20

Score: 5

CREDENCE

Let's Play a Game



Capacity: 6

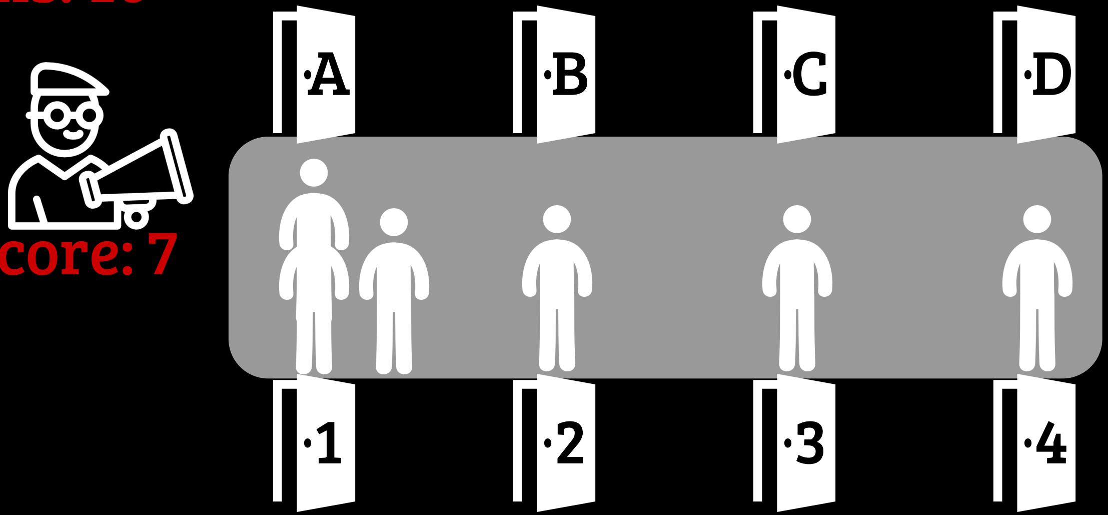
Previous Game

Arrivals: 20

Score: 5

Let’s Play a Game

Arrivals: 16



Score: 7

Capacity: 6

Previous Game

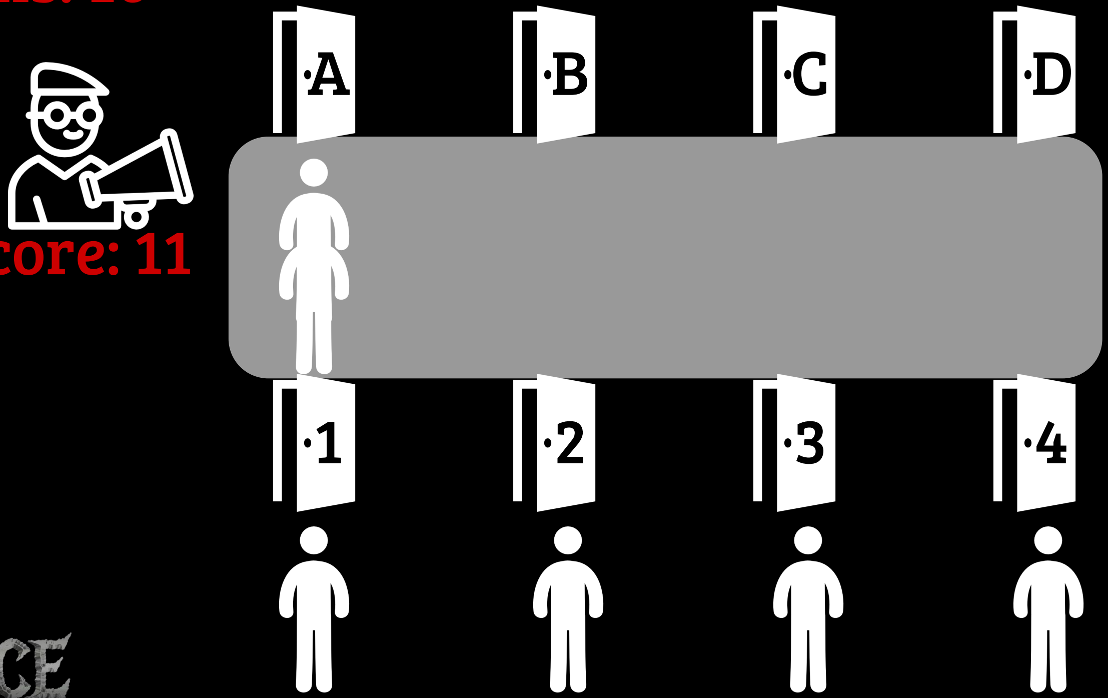
Arrivals: 20

Score: 5

Let's Play a Game

Arrivals: 16

Score: 11



Capacity: 6

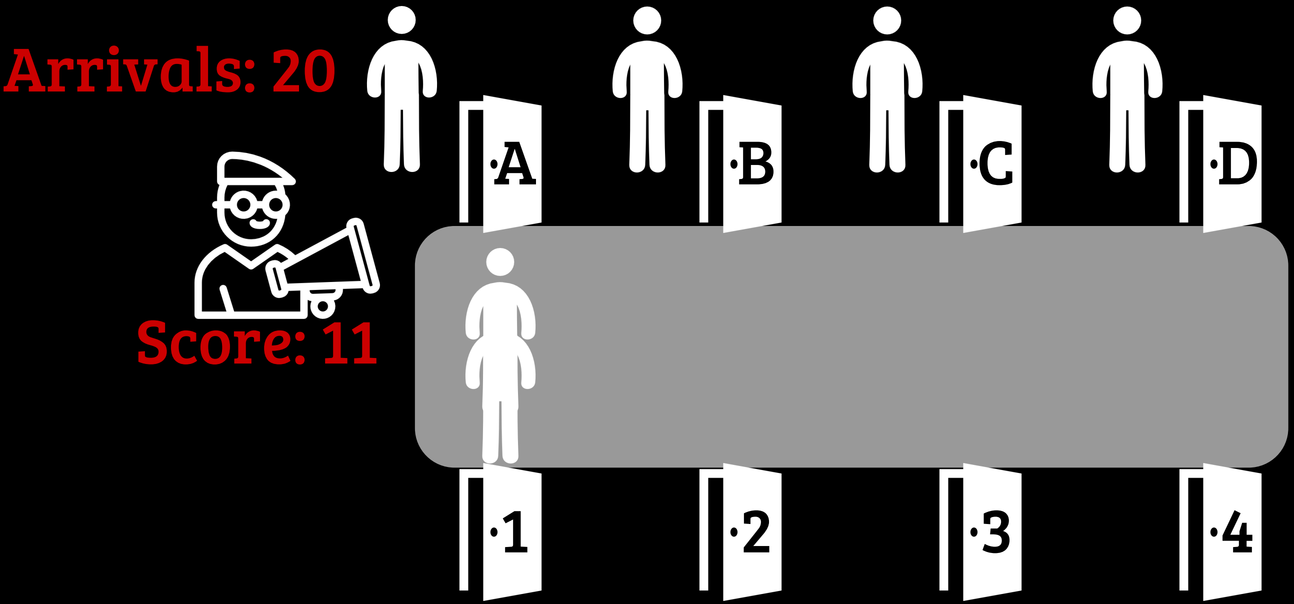
Previous Game

Arrivals: 20

Score: 5

CREDENCE

Let's Play a Game



Capacity: 6

Previous Game

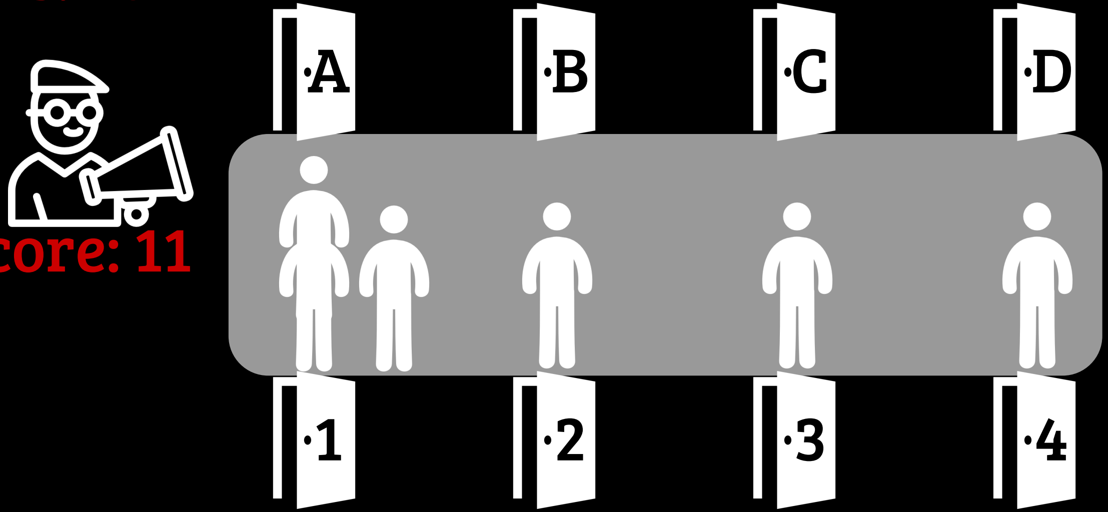
Arrivals: 20

Score: 5

CREDENCE

Let's Play a Game

Arrivals: 20



Score: 11

Capacity: 6

Previous Game

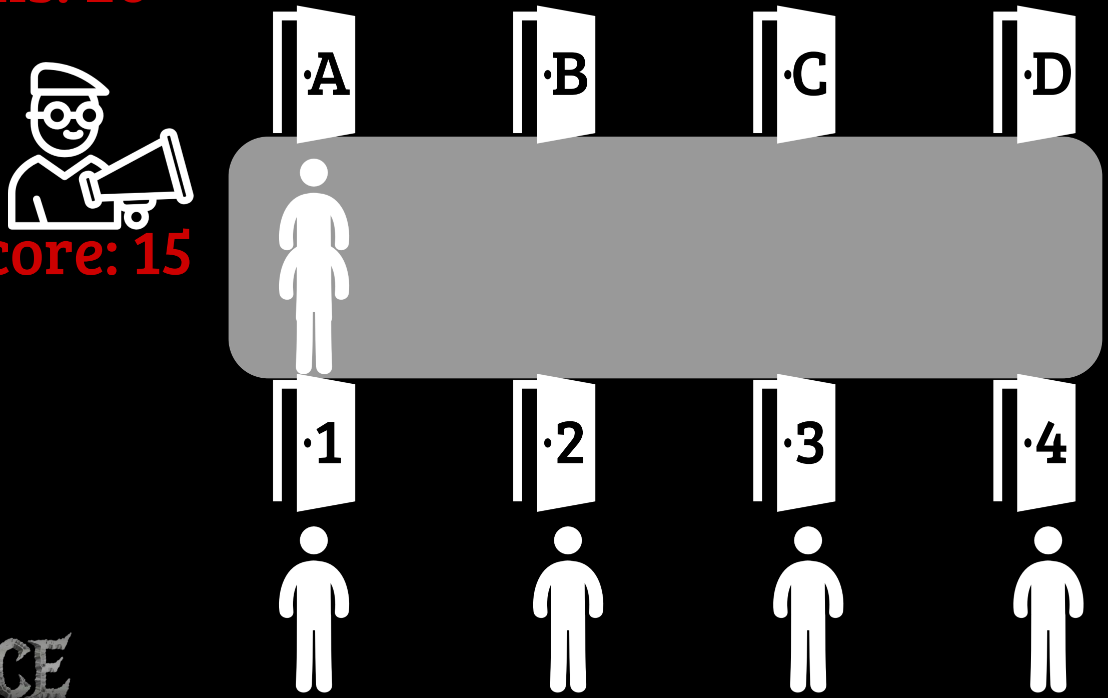
Arrivals: 20

Score: 5

CREDENCE

Let's Play a Game

Arrivals: 20



Score: 15

Capacity: 6

Previous Game

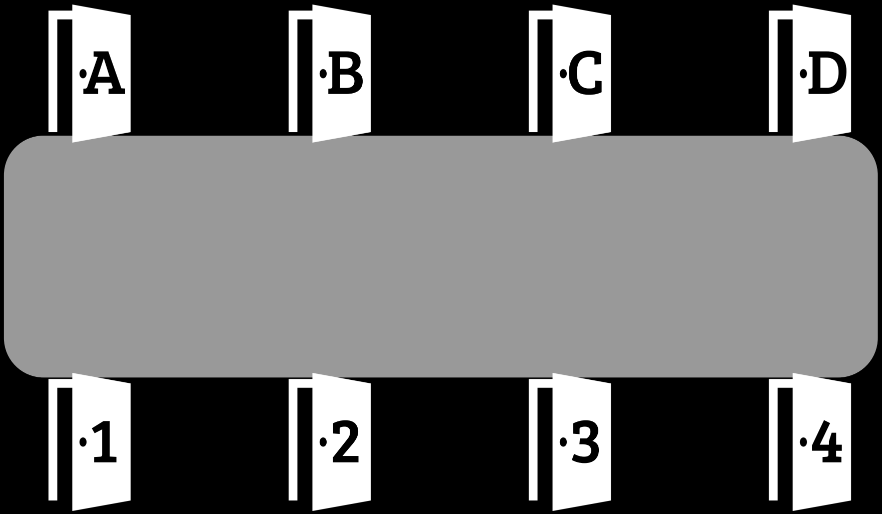
Arrivals: 20

Score: 5

Let's Play a Game

New Game

Arrivals: 20
Score: 15



Previous Game

Arrivals: 20
Score: 5

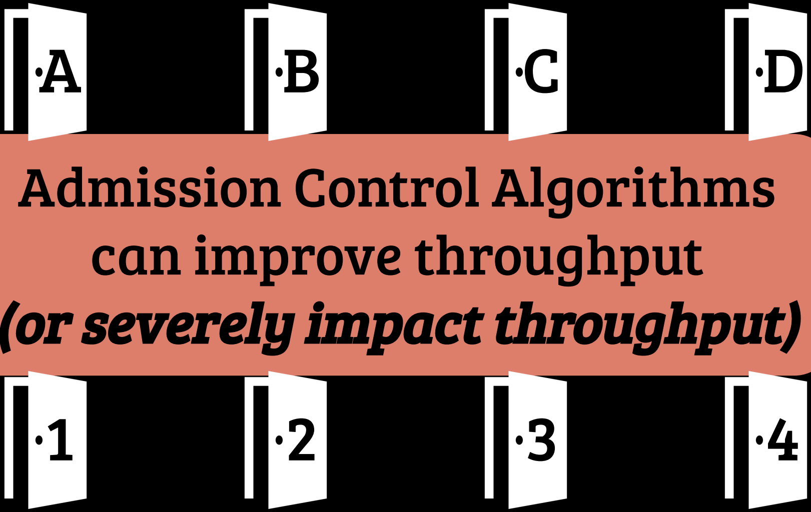
Let's Play a Game

New Game

Arrivals: 20
Score: 15

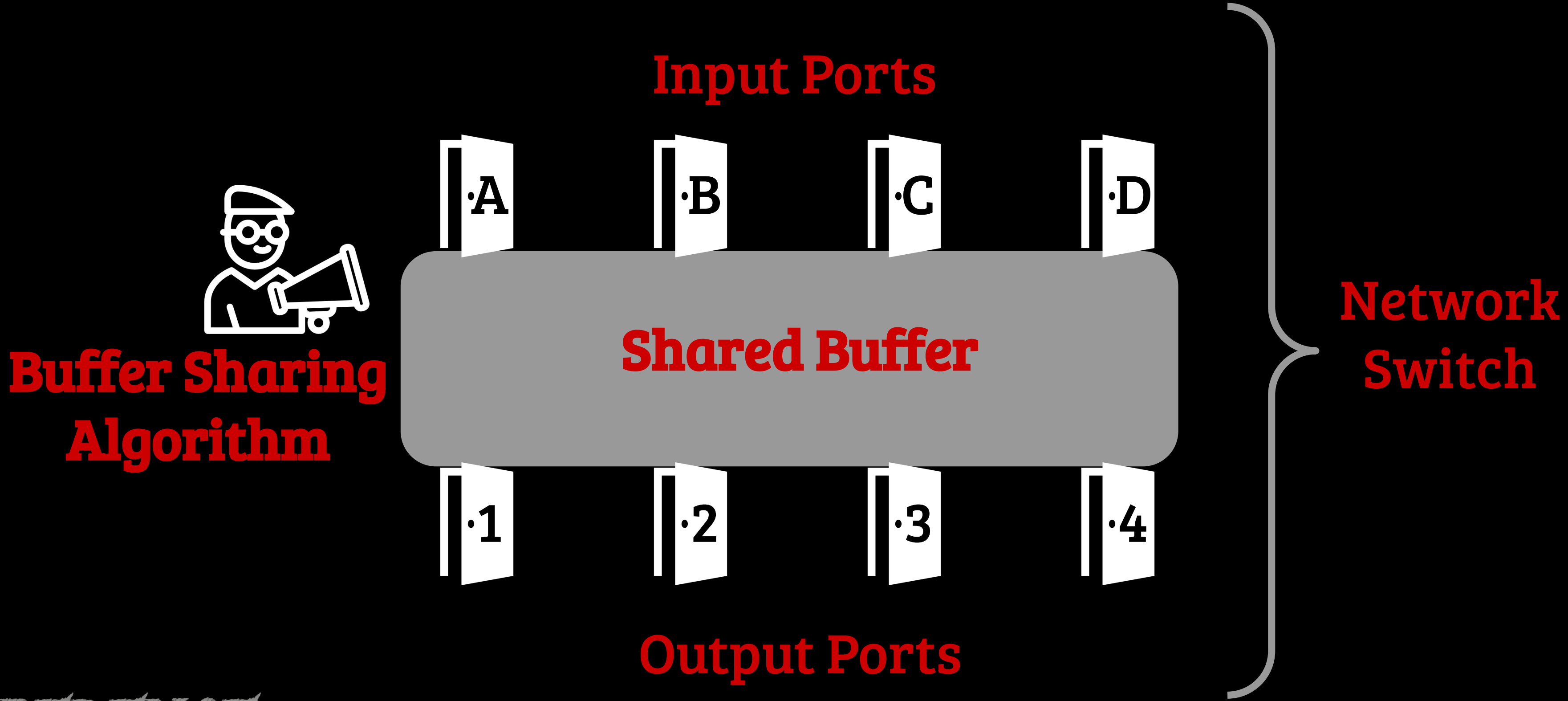
Previous Game

Arrivals: 20
Score: 5



Admission Control Algorithms can improve throughput
(or severely impact throughput)

Buffer Sharing



CREDENCE

Buffer Sharing: An Emerging Critical Problem

- Bursty traffic requires buffers to avoid packet losses
- Stringent performance requirements
- But buffer sizes are unable to scale with capacity increase

Buffer Sharing: An Emerging Critical Problem

- Bursty traffic requires buffers to avoid packet losses
- Stringent performance requirements
- But buffer sizes are unable to scale with capacity increase

Buffer Sharing algorithm can severely impact end-to-end performance e.g., FCTs

Buffer Sharing (An Online Perspective)

- Goal: Maximize the number of transmitted packets
  - Throughput maximization

Buffer Sharing (An Online Perspective)

- Goal: Maximize the number of transmitted packets
  - Throughput maximization
- Online algorithm (ALG) takes spontaneous decisions upon every packet arrival

Buffer Sharing (An Online Perspective)

- Goal: Maximize the number of transmitted packets
  - Throughput maximization
- Online algorithm (ALG) takes spontaneous decisions upon every packet arrival
- Offline optimal algorithm (OPT) has prior knowledge of the entire arrival sequence and performs optimally

Buffer Sharing (An Online Perspective)

- ALG is C competitive if OPT transmits no more than C times that of ALG
  - $OPT \leq C \cdot ALG$

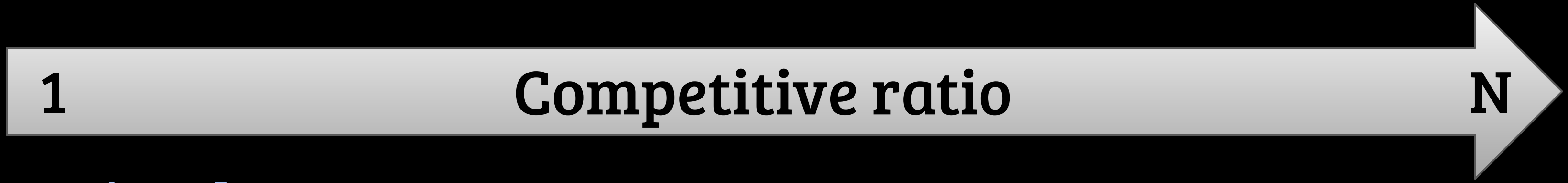
Buffer Sharing (An Online Perspective)

- ALG is C competitive if OPT transmits no more than C times that of ALG
  - $OPT \leq \boxed{C}\ ALG$

Competitive Ratio

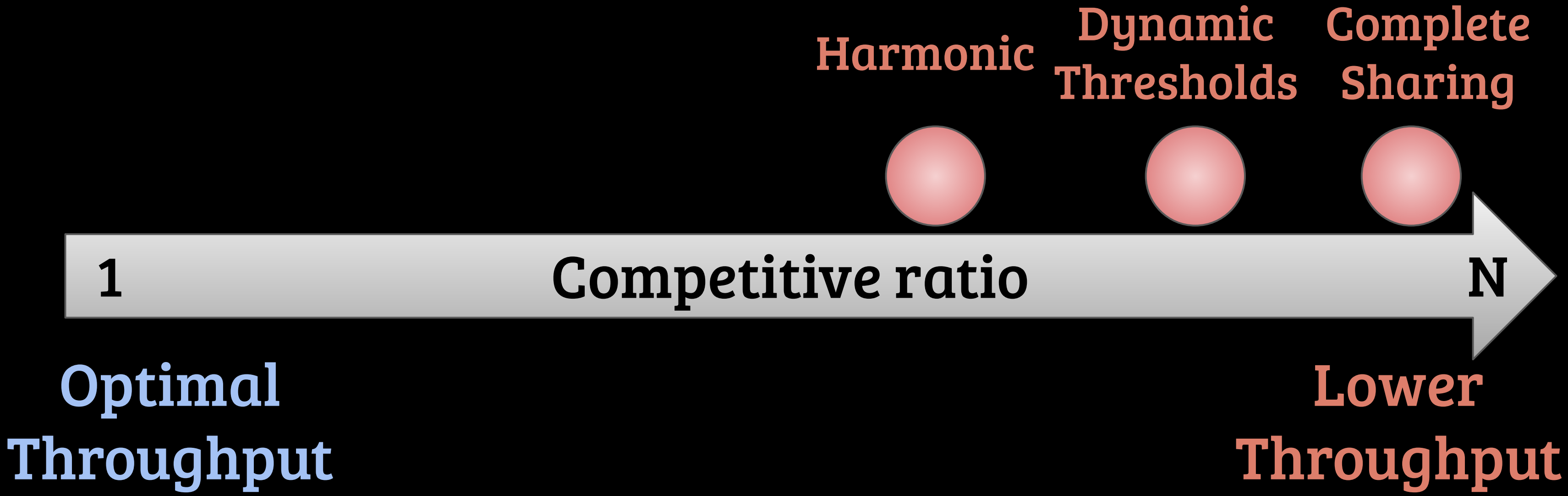
Online Buffer Sharing Algorithms

- Drop-tail: Drop on arrival or accept
  - All commodity switches support drop-tail buffers
- Push-out: Accept all packets and push a packet out when the buffer is full
  - *Not supported in hardware*



Optimal Throughput

Lower Throughput



Harmonic

Dynamic Thresholds

Complete Sharing

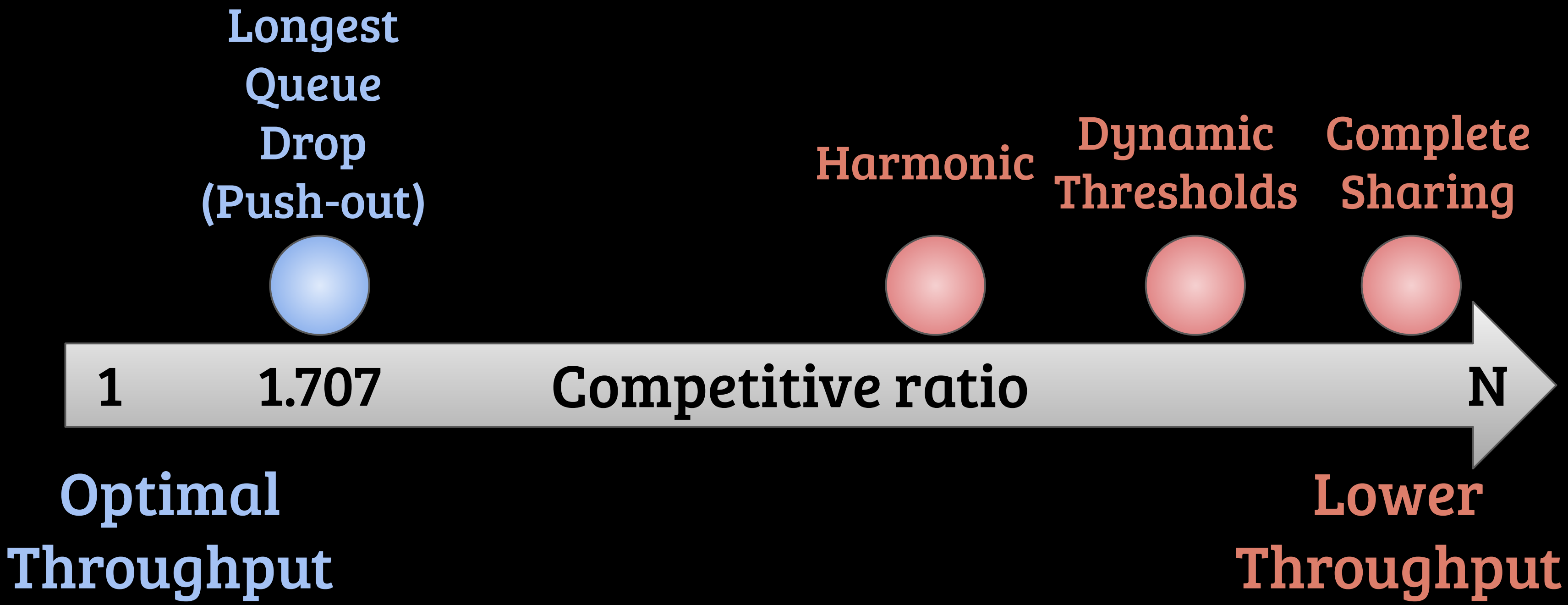
1

Competitive ratio

N

Optimal Throughput

Lower Throughput



Longest Queue Drop (Push-out)

Harmonic

Dynamic Thresholds

Complete Sharing

1

1.707

Competitive ratio

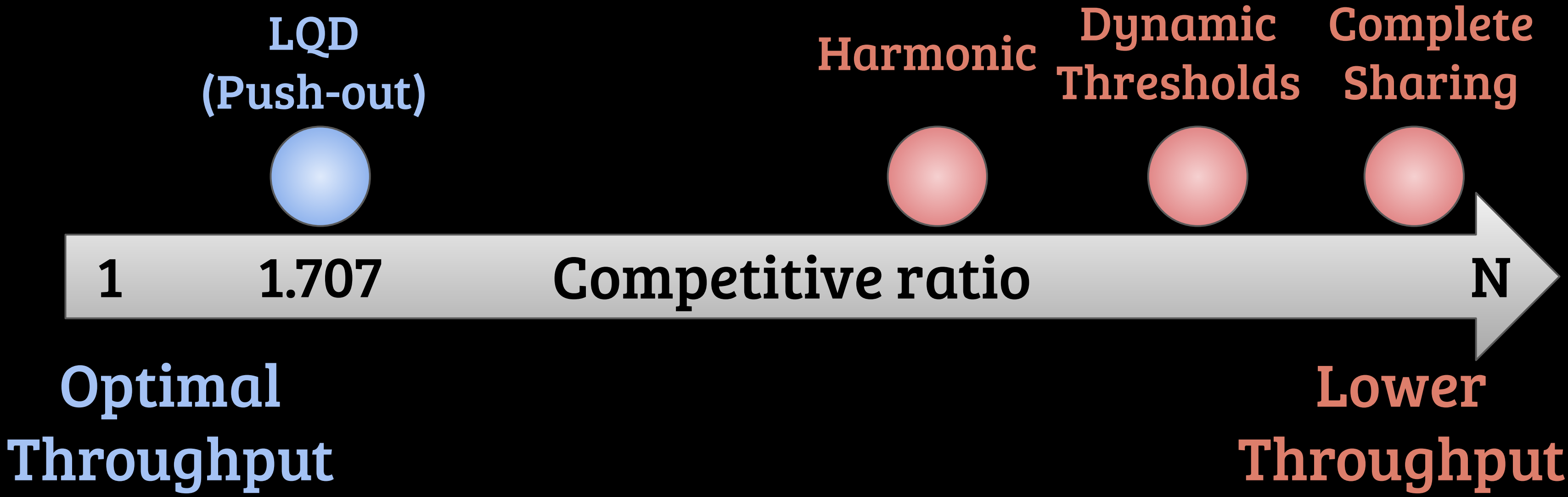
N

Optimal Throughput

Lower Throughput

Not supported in hardware!

All commodity switches support drop-tail



LQD (Push-out)

Harmonic

Dynamic Thresholds

Complete Sharing

1

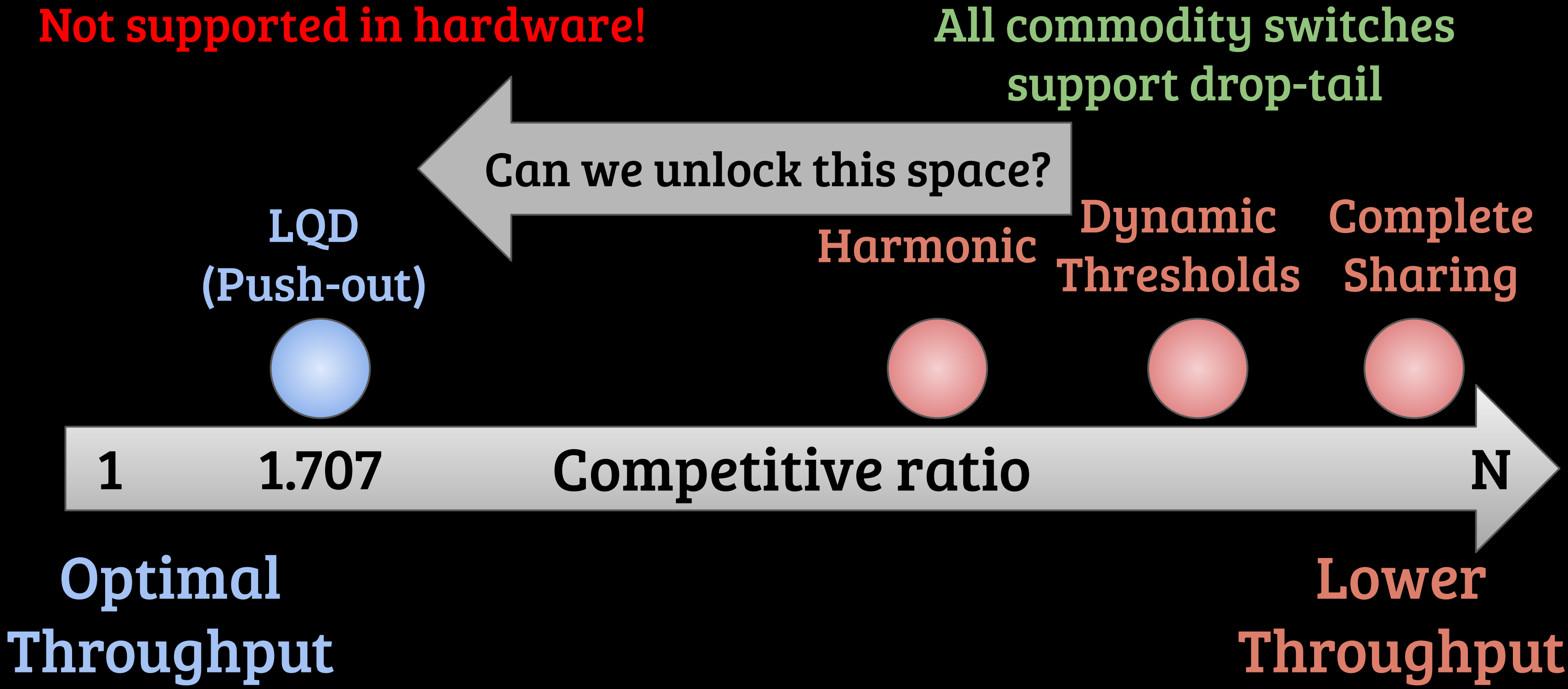
1.707

Competitive ratio

N

Optimal Throughput

Lower Throughput



Not supported in hardware!

All commodity switches support drop-tail

Can we unlock this space?

LQD (Push-out)

Harmonic

Dynamic Thresholds

Complete Sharing

1

1.707

Competitive ratio

N

Optimal Throughput

Lower Throughput

CREDENCE

Limitations of Traditional Drop-tail Buffer Sharing

- Proactive unnecessary drops → throughput loss

CREDENCE

Limitations of Traditional Drop-tail Buffer Sharing

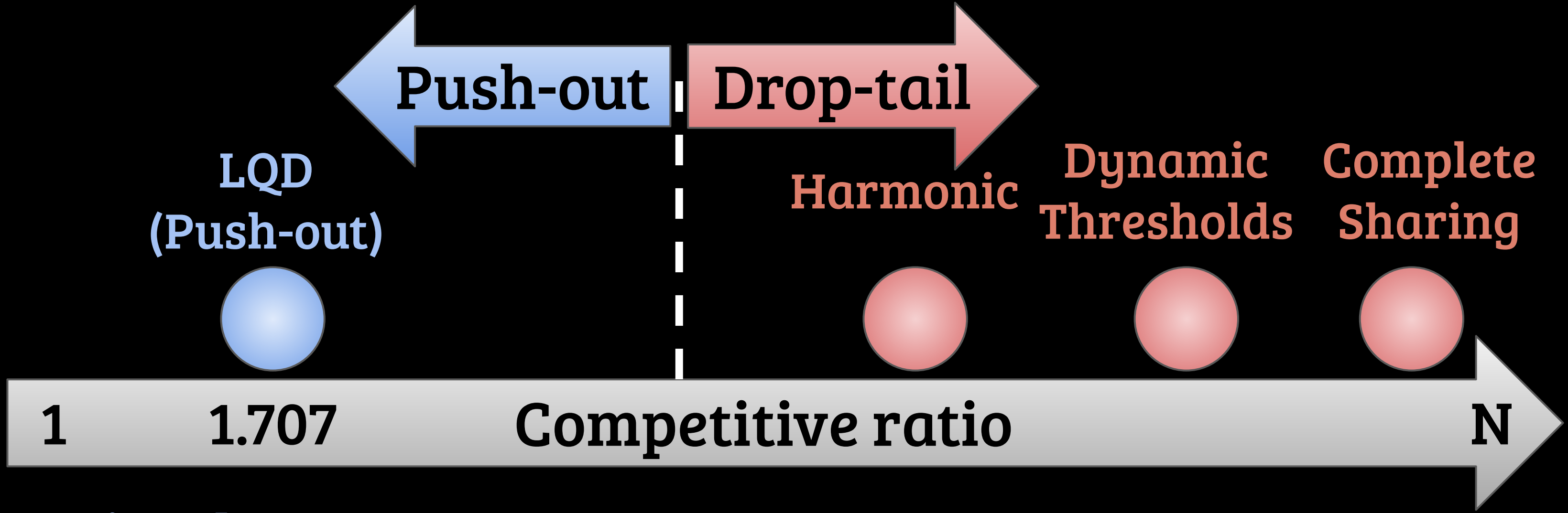
- Proactive unnecessary drops ⟶throughput loss
    - Overprovisioning for the *unknown* future arrivals
    - Packet drops are unnecessary if the future is known

Limitations of Traditional Drop-tail Buffer Sharing

- Proactive unnecessary drops ⟶throughput loss
  - Overprovisioning for the *unknown* future arrivals
  - Packet drops are unnecessary if the future is known
- Reactive avoidable drops ⟶throughput loss

Limitations of Traditional Drop-tail Buffer Sharing

- Proactive unnecessary drops ⟶ throughput loss
  - Overprovisioning for the *unknown* future arrivals
  - Packet drops are unnecessary if the future is known
- Reactive avoidable drops ⟶ throughput loss
  - Underprovisioning for the unknown future arrivals
  - Packet drops are avoidable if the future is known



Push-out

Drop-tail

LQD (Push-out)

Harmonic

Dynamic Thresholds

Complete Sharing

1

1.707

Competitive ratio

N

Optimal Throughput

Lower Throughput

Predictions: A Hope for Competitive Buffer Sharing

- Predict the actions of a push-out algorithm (LQD)
- Augment drop-tail algorithms with predictions
  - Peek into the future

Predictions: A Hope for Competitive Buffer Sharing

- Predict the actions of a push-out algorithm (LQD)
- Augment drop-tail algorithms with predictions
  - Peek into the future

Can predictions improve drop-tail's competitive ratio?

CREDENCE

Naive Approach

- Upon a packet arrival
  - Predict LQD’s action
  - If prediction is to accept, then accept
  - If prediction is to drop, then drop

Challenge: Imperfect Predictions

True Positive Ground Truth: Drop Prediction: Drop	**False Negative** Ground Truth: Drop Prediction: Accept
False Positive Ground Truth: Accept Prediction: Drop	**True Negative** Ground Truth: Accept Prediction: Accept

Challenge: Imperfect Predictions

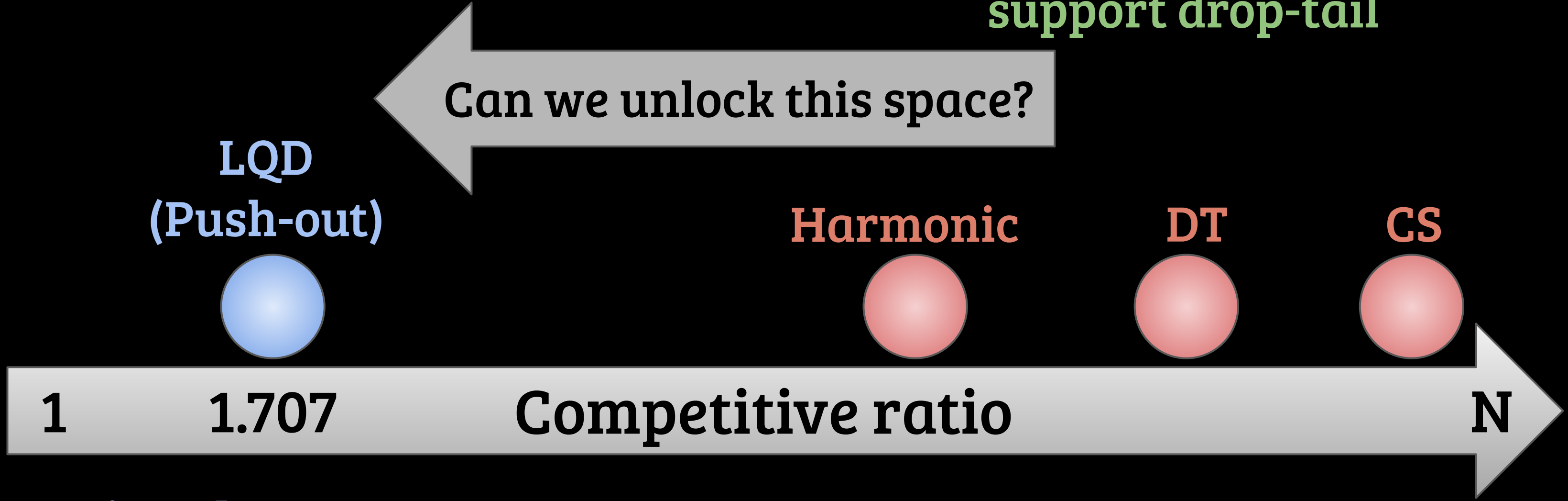
- Excessive false positives can lead to starvation
  - eg., every prediction is “drop”

Challenge: Imperfect Predictions

- Excessive false positives can lead to starvation
  - eg., every prediction is “drop”
- Even a single false negative can hurt throughput forever
  - (discussed in the paper)

Goals

- Consistency (under perfect predictions)
  - Competitive ratio close to push-out
- Robustness (with large prediction error)
  - Competitive ratio close to existing algorithms
- Smoothness
  - Competitive ratio smoothly degrades with prediction error



Not supported in hardware!

All commodity switches support drop-tail

Can we unlock this space?

LQD (Push-out)

Harmonic

DT

CS

1

1.707

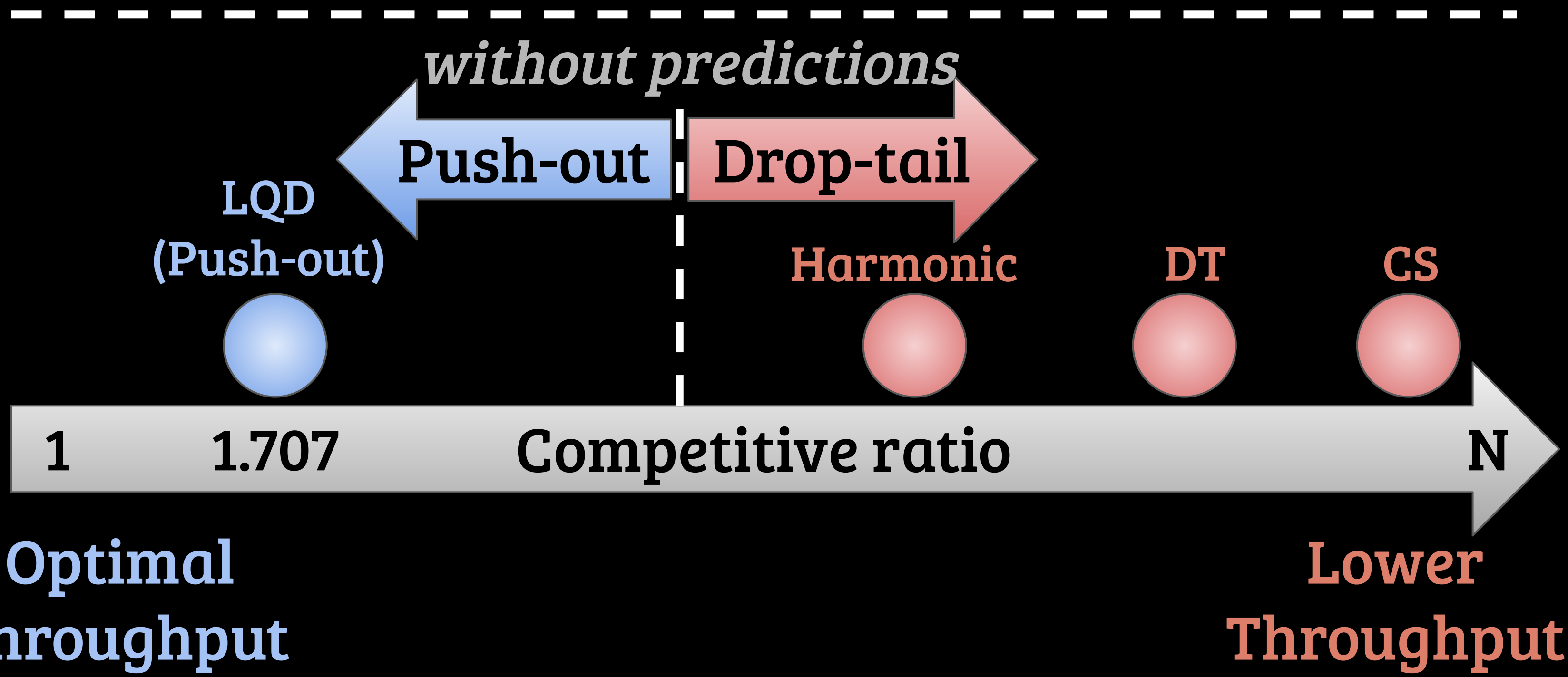
Competitive ratio

N

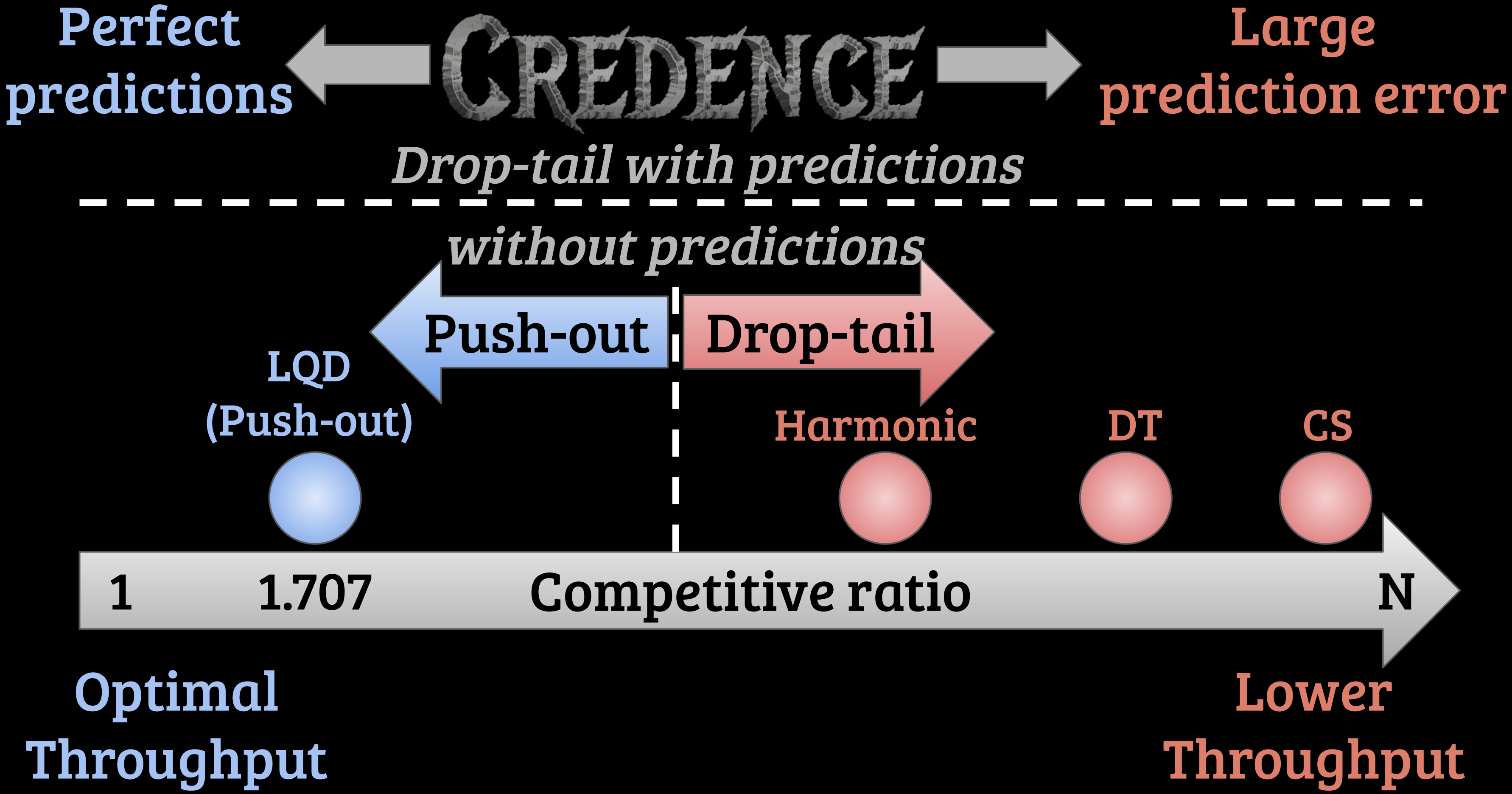
Optimal Throughput

Lower Throughput

CREDENCE


without predictions
Push-out
Drop-tail
LQD
(Push-out)
Harmonic
DT
CS
1
1.707
Competitive ratio
N
Optimal
Throughput
Lower
Throughput

CREDENCE



Perfect predictions

CREDENCE

Large prediction error

Drop-tail with predictions

without predictions

Push-out

Drop-tail

LQD (Push-out)

Harmonic

DT

CS

1

1.707

Competitive ratio

N

Optimal Throughput

Lower Throughput

Credence

- Drop-tail buffer sharing augmented with predictions
- Threshold-based (similar to existing algorithms)

Credence

- Drop-tail buffer sharing augmented with predictions
- Threshold-based (similar to existing algorithms)
- Consistency ✅
  - Close to push-out under perfect predictions

Credence

- Drop-tail buffer sharing augmented with predictions
- Threshold-based (similar to existing algorithms)
- Consistency ✅
  - Close to push-out under perfect predictions
- Robustness ✅
  - Close to existing algorithms even with large prediction error

Credence

- Drop-tail buffer sharing augmented with predictions
- Threshold-based (similar to existing algorithms)
- Consistency ✅
  - Close to push-out under perfect predictions
- Robustness ✅
  - Close to existing algorithms even with large prediction error
- Smoothness ✅
  - Smoothly degrades with prediction error

Credence’s Thresholds

- Per-queue thresholds
    - Thresholds are incremented and decremented based on Longest Queue Drop (Push-out) algorithm

Credence's Thresholds

- Per-queue thresholds
  - Thresholds are incremented and decremented based on Longest Queue Drop (Push-out) algorithm
- A packet is rejected immediately if the queue length is greater than its corresponding threshold

Credence's Thresholds

- Per-queue thresholds
    - Thresholds are incremented and decremented based on Longest Queue Drop (Push-out) algorithm
- A packet is rejected immediately if the queue length is greater than its corresponding threshold
- A **prediction** is obtained *only if* the queue length is lower than its corresponding threshold

Credence’s Thresholds

- Thresholds enable tackling false negative errors
  - Prevents accepting too many packets eg., if all the predictions are “accept”

Credence's Thresholds

- Thresholds enable tackling false negative errors
  - Prevents accepting too many packets eg., if all the predictions are "accept"
- Safe guard criterion to tackle false positive errors
  - Always accept a packet if the longest queue is lower than fair-share of buffer partition
  - Prevents dropping too many packets eg., if all the predictions are "drop"

Further Details in the Paper

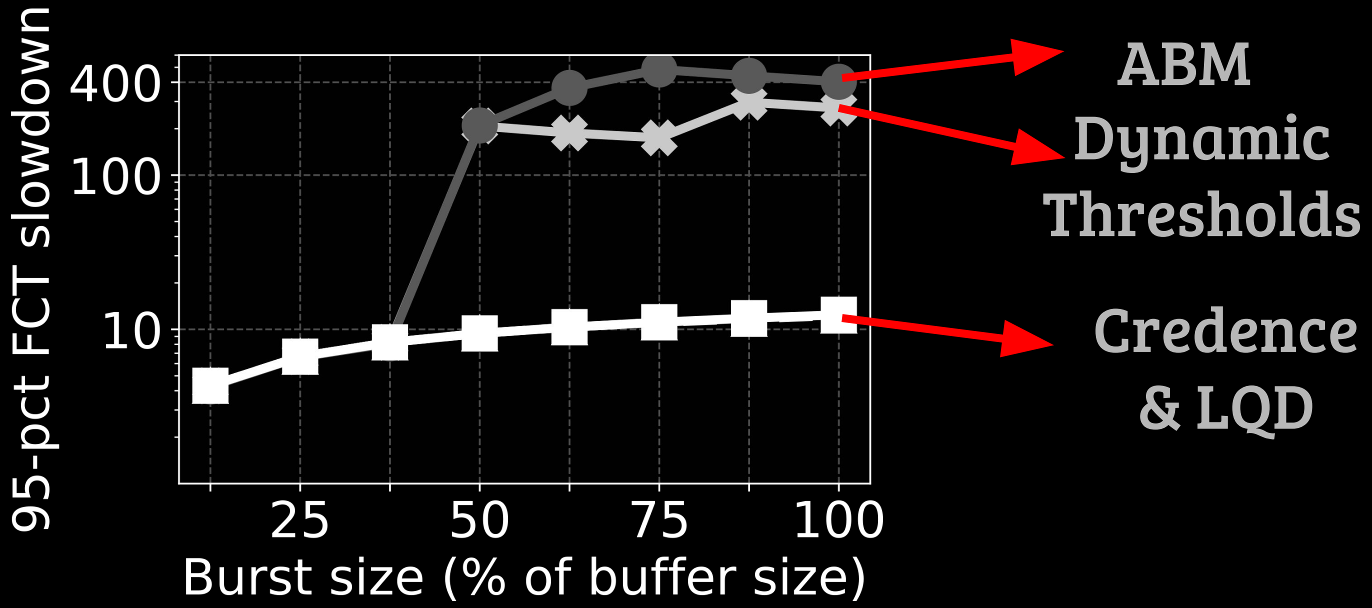
- Competitive analysis
- Theoretical bounds for Credence's performance
- …

Evaluation

- Packet-level simulations using NS3
- 256 servers, 4 spine switches and 16 ToR switches
- 10Gbps NICs
- Shared buffer at the switches
- Random Forest-based prediction oracle for Credence

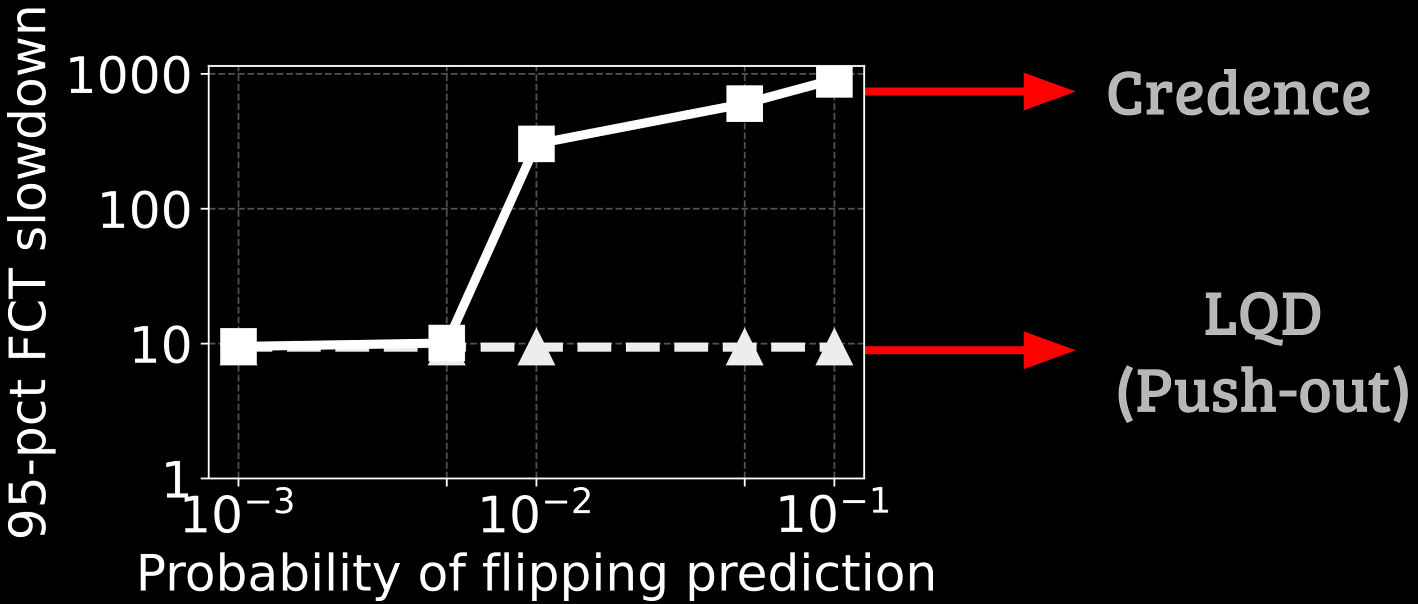
CREDENCE

Credence Performs Close to Push-out



CREDENCE

Credence Degrades with Prediction Error



Credence

LQD
(Push-out)

Open Questions and Future Research Directions

- Practically training a prediction oracle
  - Simulation-based data (may not capture real-world scenarios)
  - Real-world network data (more accurate but complex to obtain)
    - Online reinforcement learning
    - ...
- Understanding push-out operation complexity
- Improving the robustness of Credence
- Considering latency for competitive analysis

Conclusion

- Traditional drop-tail buffer sharing approaches cannot be improved further
- Credence is the first buffer sharing algorithm augmented with predictions
- Credence offers bounded performance guarantees
- Credence can improve the performance of datacenter traffic in terms of flow completion times for short flows and incast flows
- Source code: https://github.com/inet-tub/ns3-datacenter

Vamsi Addanki
vamsi@inet.tu-berlin.de
🐦 @Vamsi_DT

Maciej Pacut
maciej@inet.tu-berlin.de

Stefan Schmid
stefan.schmid@tu-berlin.de
🐦 @schmiste_ch

Thank You