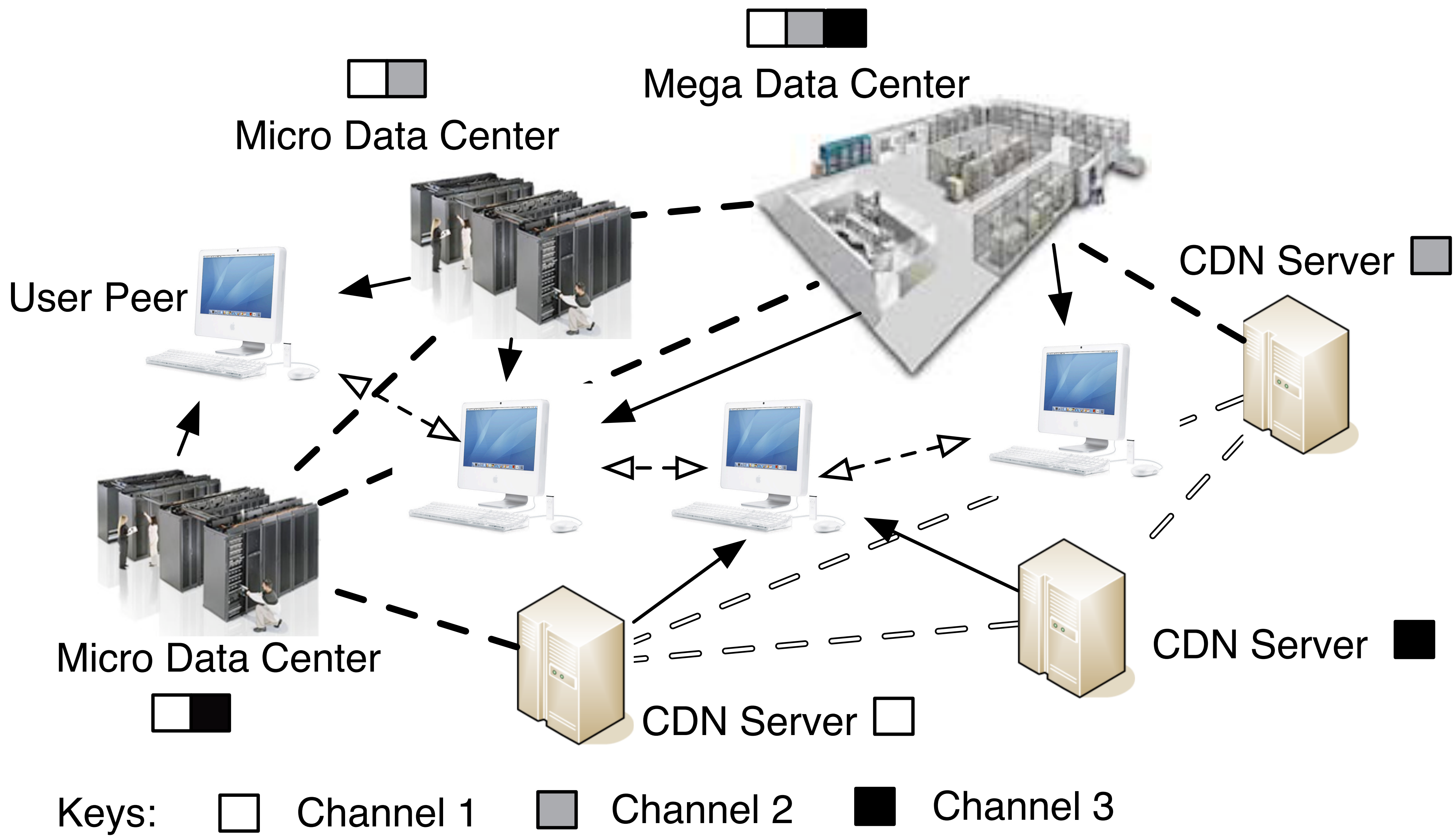
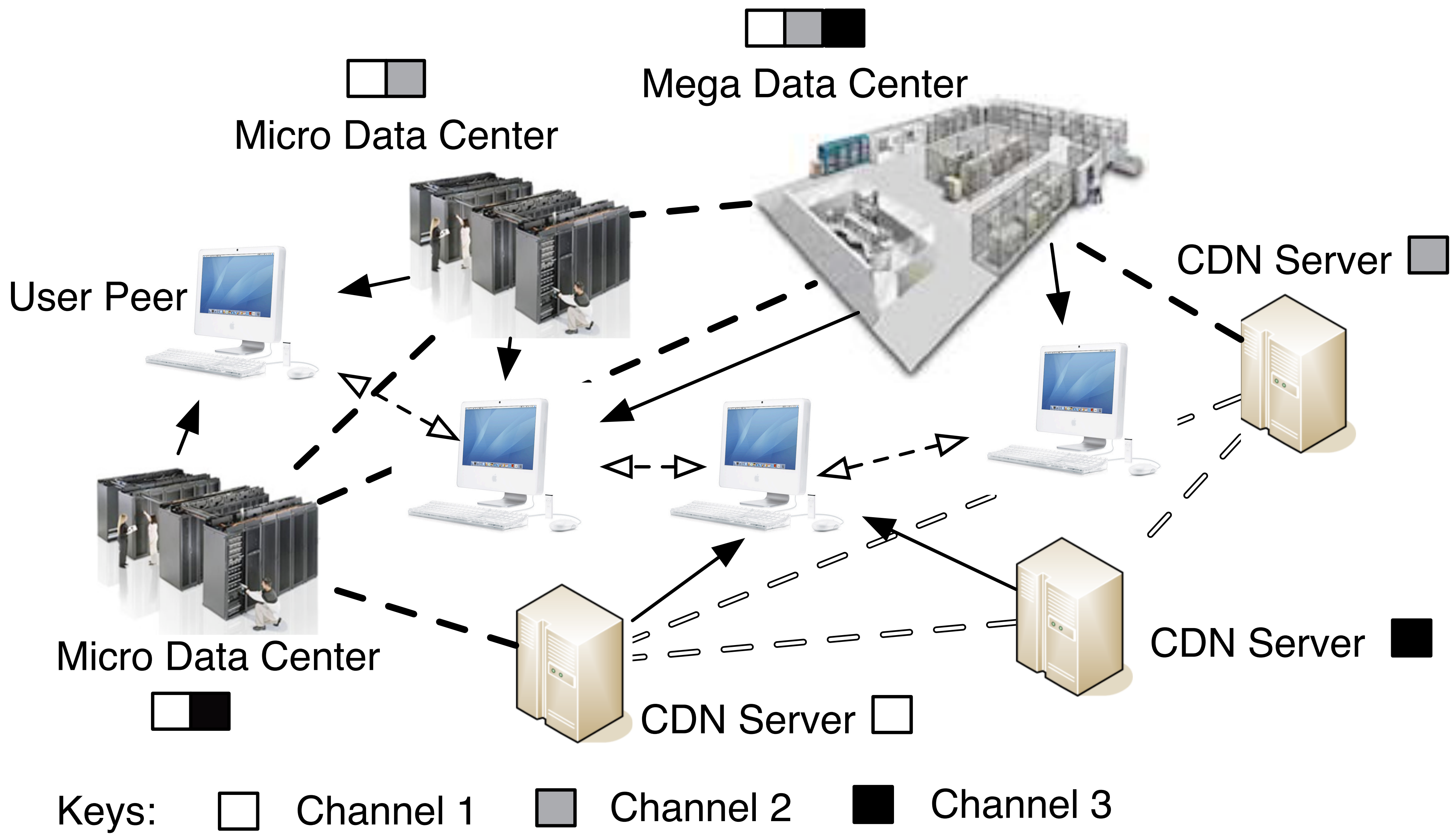
# Self-Diagnostic Peer-Assisted Video Streaming through a Learning Framework

**Di Niu** (Presenter), **Baochun Li**
Department of Electrical and Computer Engineering
University of Toronto

# Large Scale Peer-Assisted Video Streaming



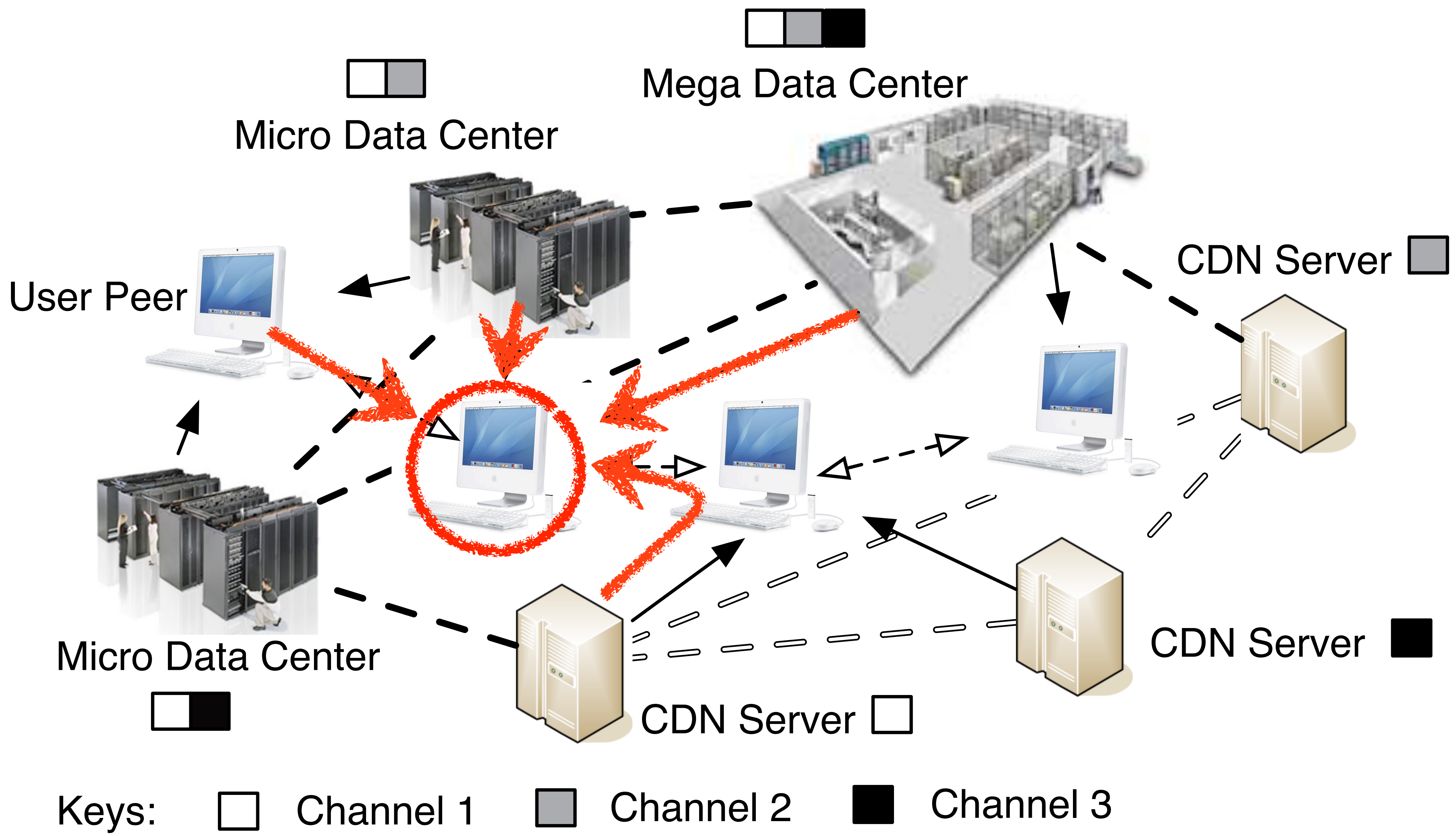
# Large Scale Peer-Assisted Video Streaming




A peer receives media blocks from *data centers*, *CDN servers*, and other *peers*.

**Benefits:** alleviate server bottleneck, robustness, resilience.

# Large Scale Peer-Assisted Video Streaming




A peer receives media blocks from *data centers*, *CDN servers*, and other *peers*.

**Benefits:** alleviate server bottleneck, robustness, resilience.

# Are These Systems Perfect?

# Are These Systems Perfect?

## Not really!

- Quality control is hard due to the large scale, distributed nature of the system, and peer dynamics (hard to predict peer resources)
- Block exchange protocols are only best-effort

# Are These Systems Perfect?

## Not really!

Quality control is hard due to the large scale, distributed nature of the system, and peer dynamics (hard to predict peer resources)

Block exchange protocols are only best-effort

## Inefficient Server Provisioning

Servers are simply responding to requests

Not aware of the performance when allocating server bandwidth

# Are These Systems Perfect?

## Not really!

- Quality control is hard due to the large scale, distributed nature of the system, and peer dynamics (hard to predict peer resources)
- Block exchange protocols are only best-effort

## Inefficient Server Provisioning

- Servers are simply responding to requests
- Not aware of the performance when allocating server bandwidth

## Risks

- Suffer from network-wise performance downgrade in extreme workloads

# Are These Systems Perfect?

## Not really!

Quality control is hard due to the large scale, distributed nature of the system, and peer dynamics (hard to predict peer resources)

Block exchange protocols are only best-effort

## Inefficient Server Provisioning

Servers are simply responding to requests

Not aware of the performance when allocating server bandwidth

## Risks

Suffer from network-wise performance downgrade in extreme workloads

## Load Balancing, Resource Allocation, ...

Which servers and peers to download from...

Which peers to upload to...

# Current Systems Rely on Server Over-Provisioning

# Evidence from a Real-World System

# Evidence from a Real-World System

**UUSee Live Streaming**

A major commercial P2P media streaming service based in China, offering 1000+ live channels to 40+ countries, attracting millions of users at peak times.

**Measurement Period (12 Days)**

2008 Beijing Summer Olympics, Aug 8-19, 2008

**Each UUSee client**

sends "heartbeat" reports to logging servers every 10 minutes using UDP

**A "heartbeat" contains** IP address, the channel it's watching, its buffer availability map, a list of partners and the sending/receiving rates to/from them.

# Existing Problems

## Inefficient server bandwidth usage by channels

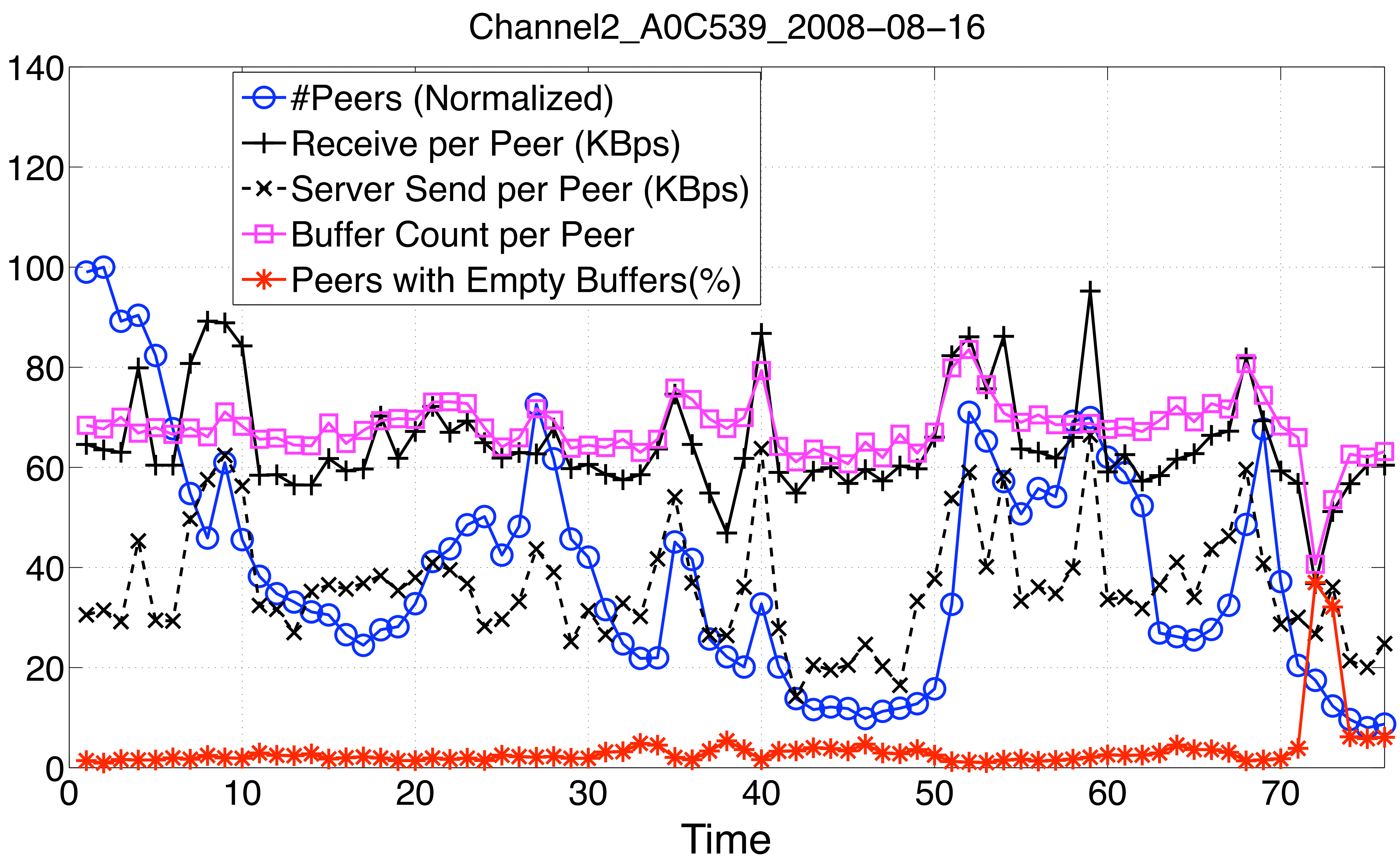
The allocation is sometimes *surplus* and sometimes *deficit*

Time

# Existing Problems

## Inefficient server bandwidth usage by channels

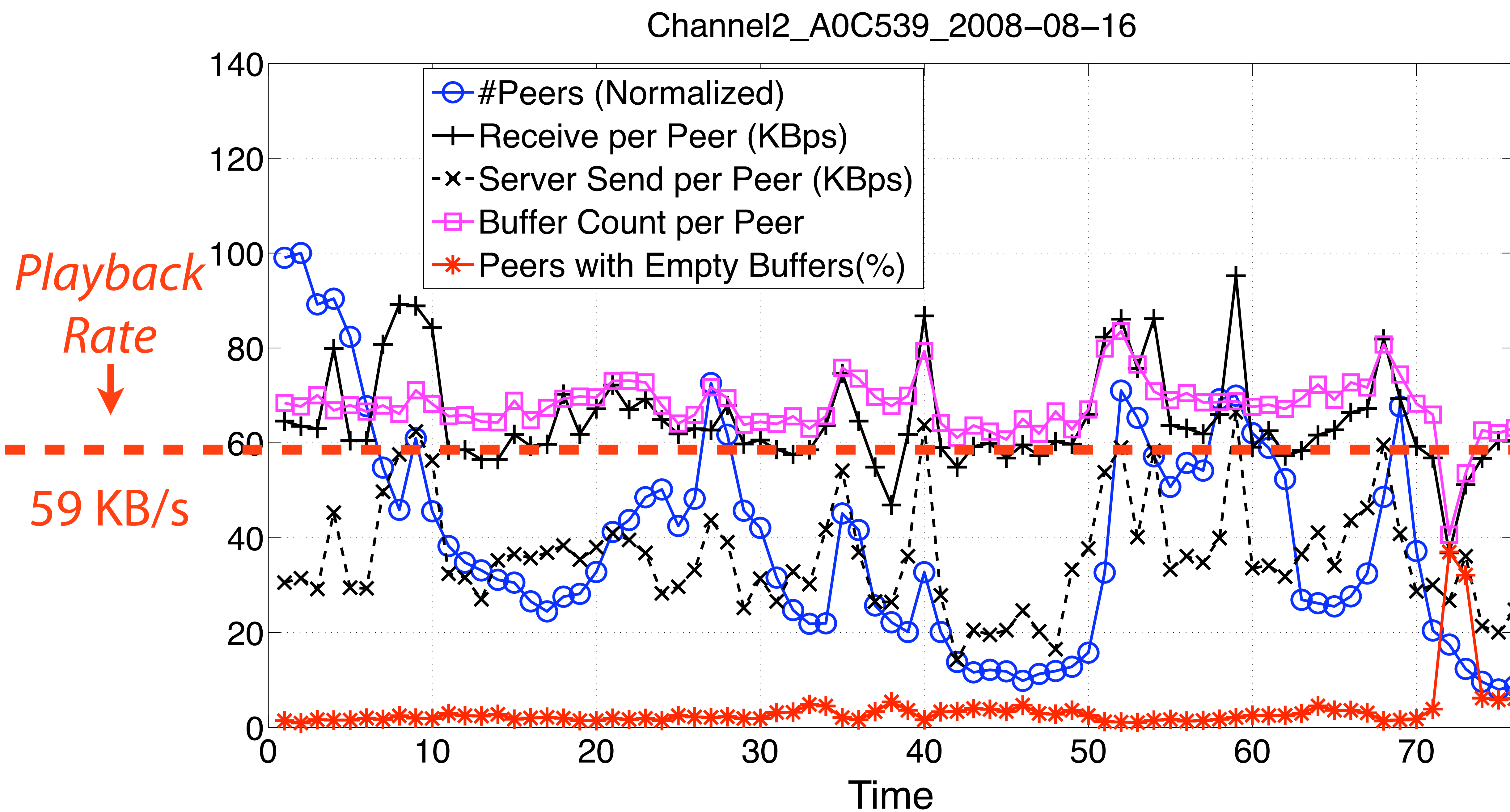
The allocation is sometimes *surplus* and sometimes *deficit*



# Existing Problems

## Inefficient server bandwidth usage by channels

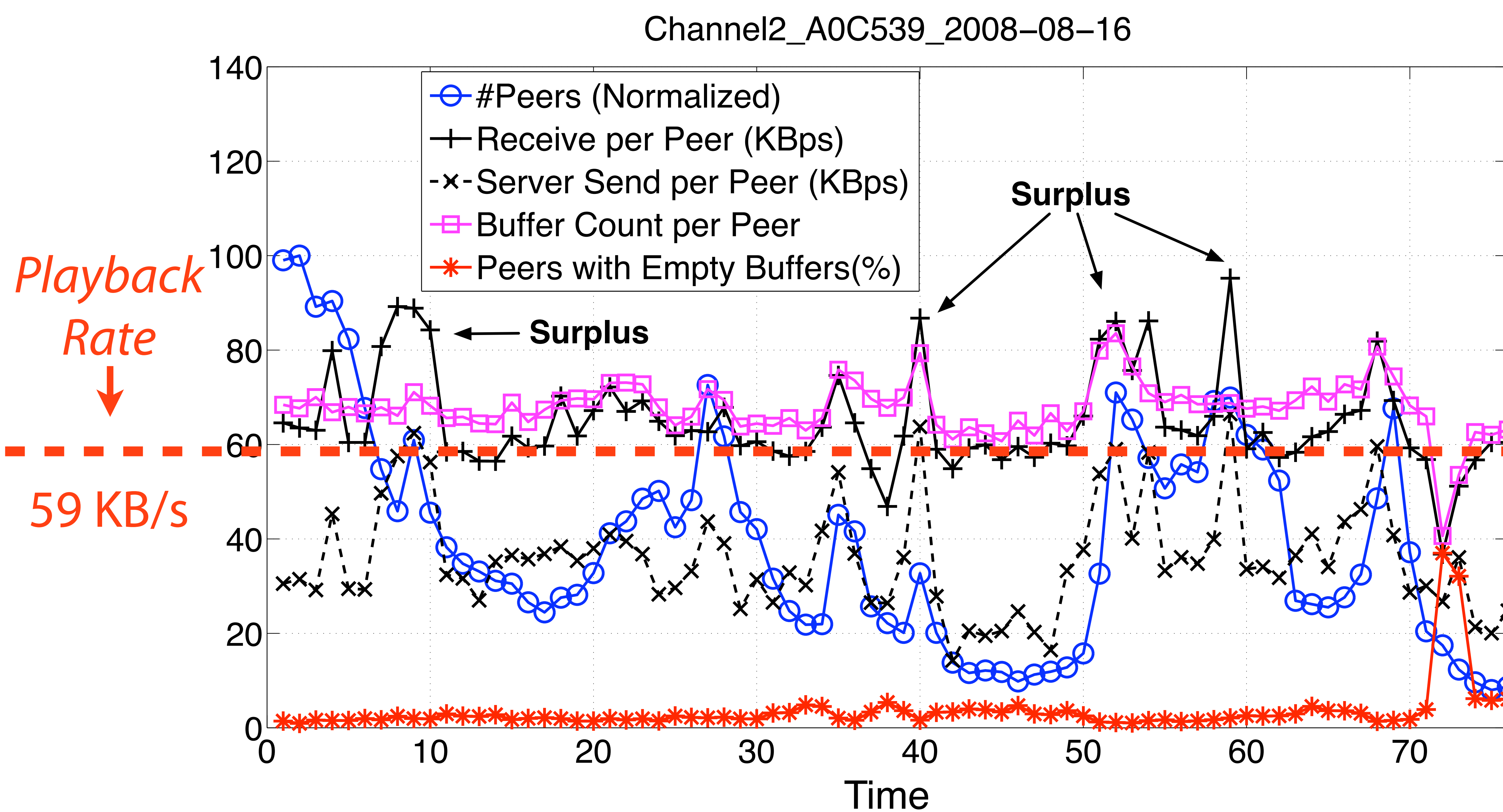
The allocation is sometimes *surplus* and sometimes *deficit*



# Existing Problems

## Inefficient server bandwidth usage by channels

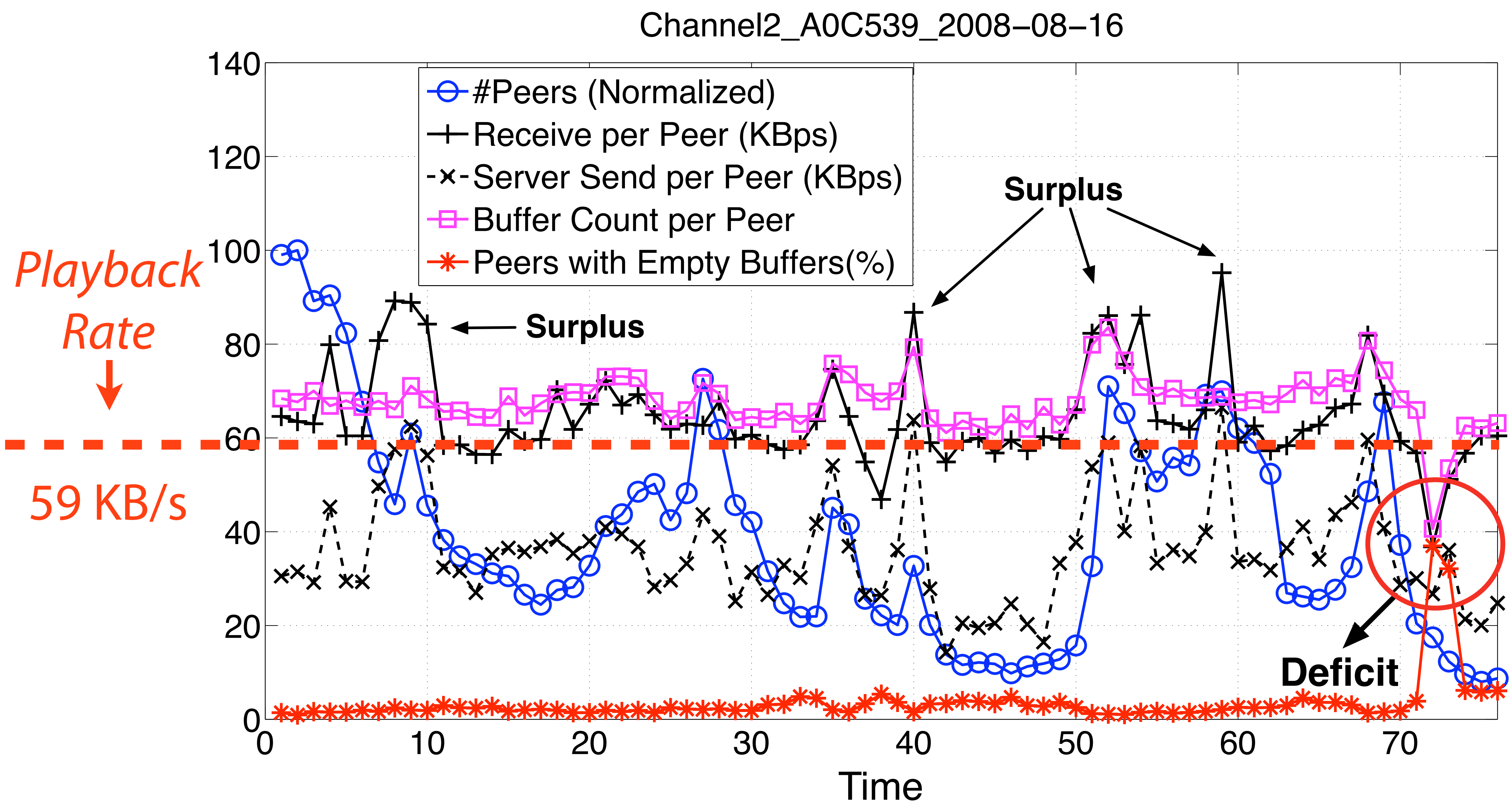
The allocation is sometimes *surplus* and sometimes *deficit*



# Existing Problems

## Inefficient server bandwidth usage by channels

The allocation is sometimes *surplus* and sometimes *deficit*



# Existing Problems

## Performance Anomalies.

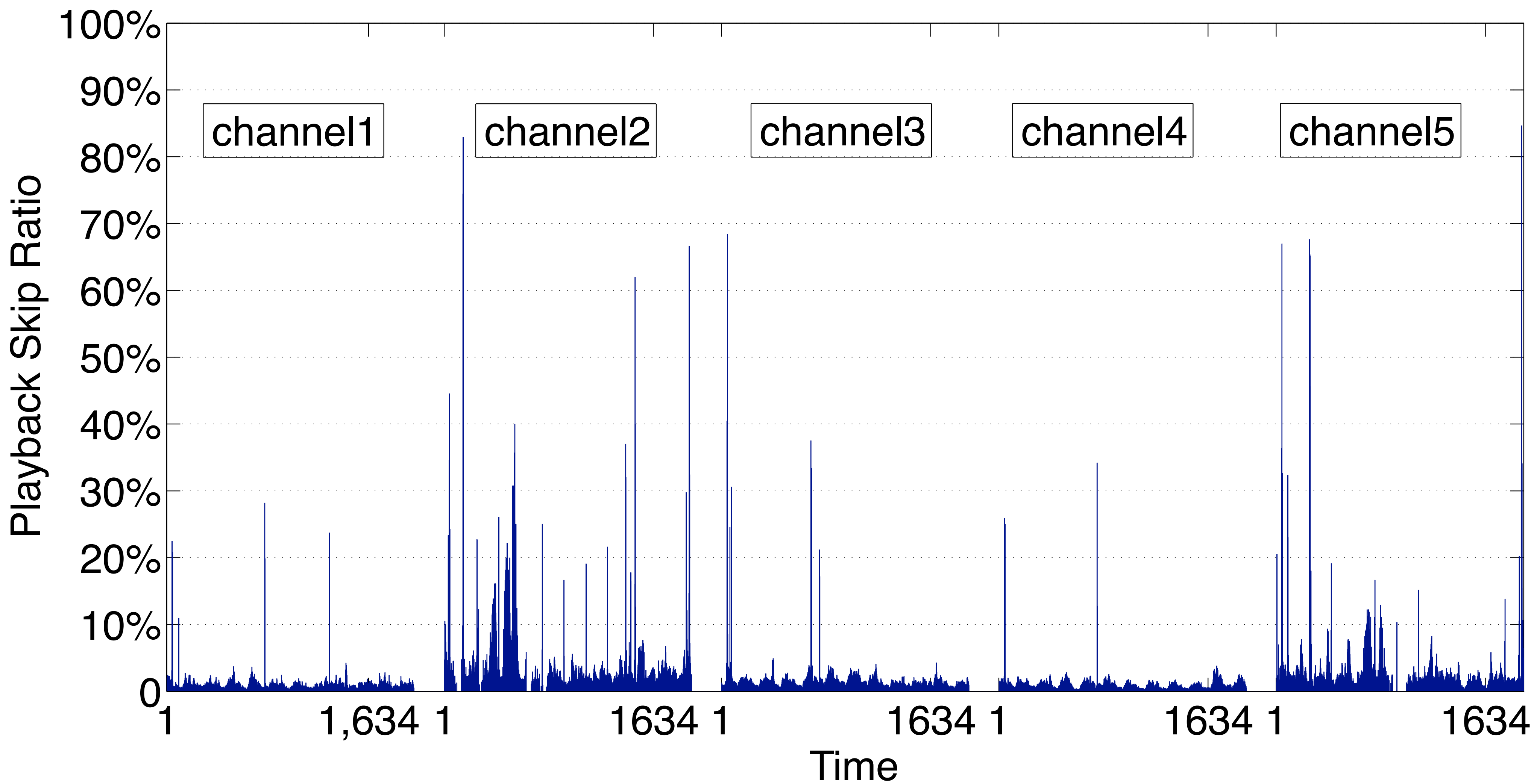



**Figure 4: The playback skip ratio during the period Aug 8-19, 2008 for 5 popular channels. The period includes 1634 timestamps for each channel.**

# Existing Problems

## Performance Anomalies.

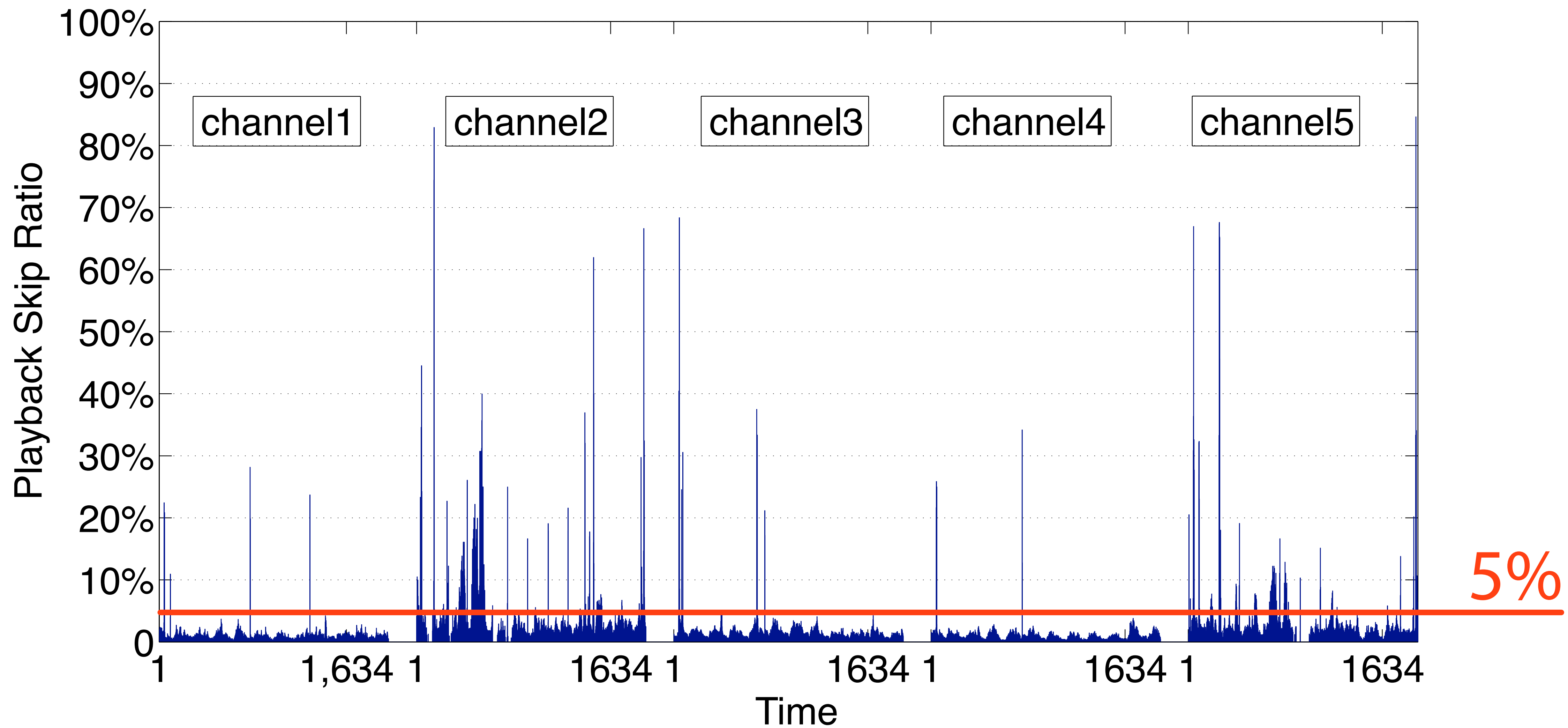



**Figure 4: The playback skip ratio during the period Aug 8-19, 2008 for 5 popular channels. The period includes 1634 timestamps for each channel.**

# Existing Problems

## Performance Anomalies.

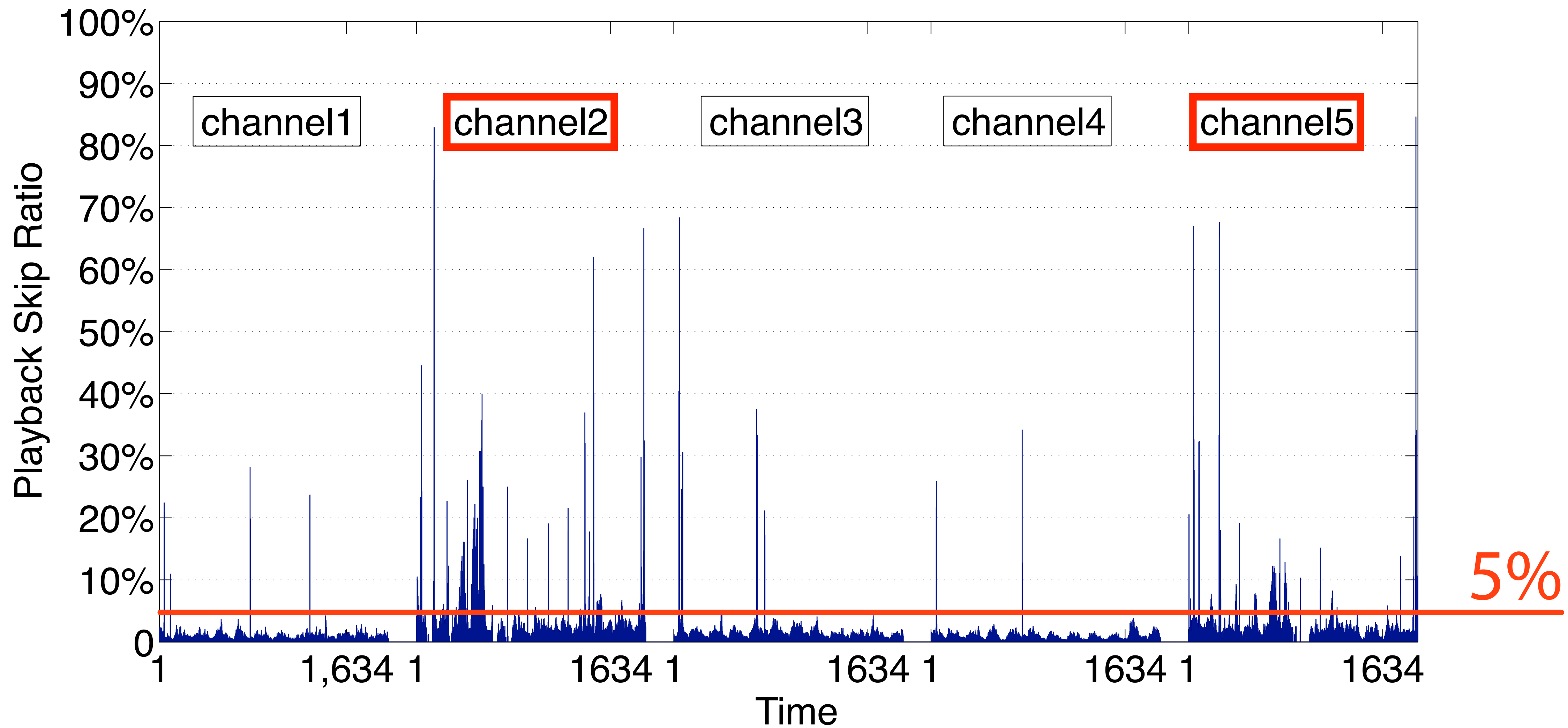
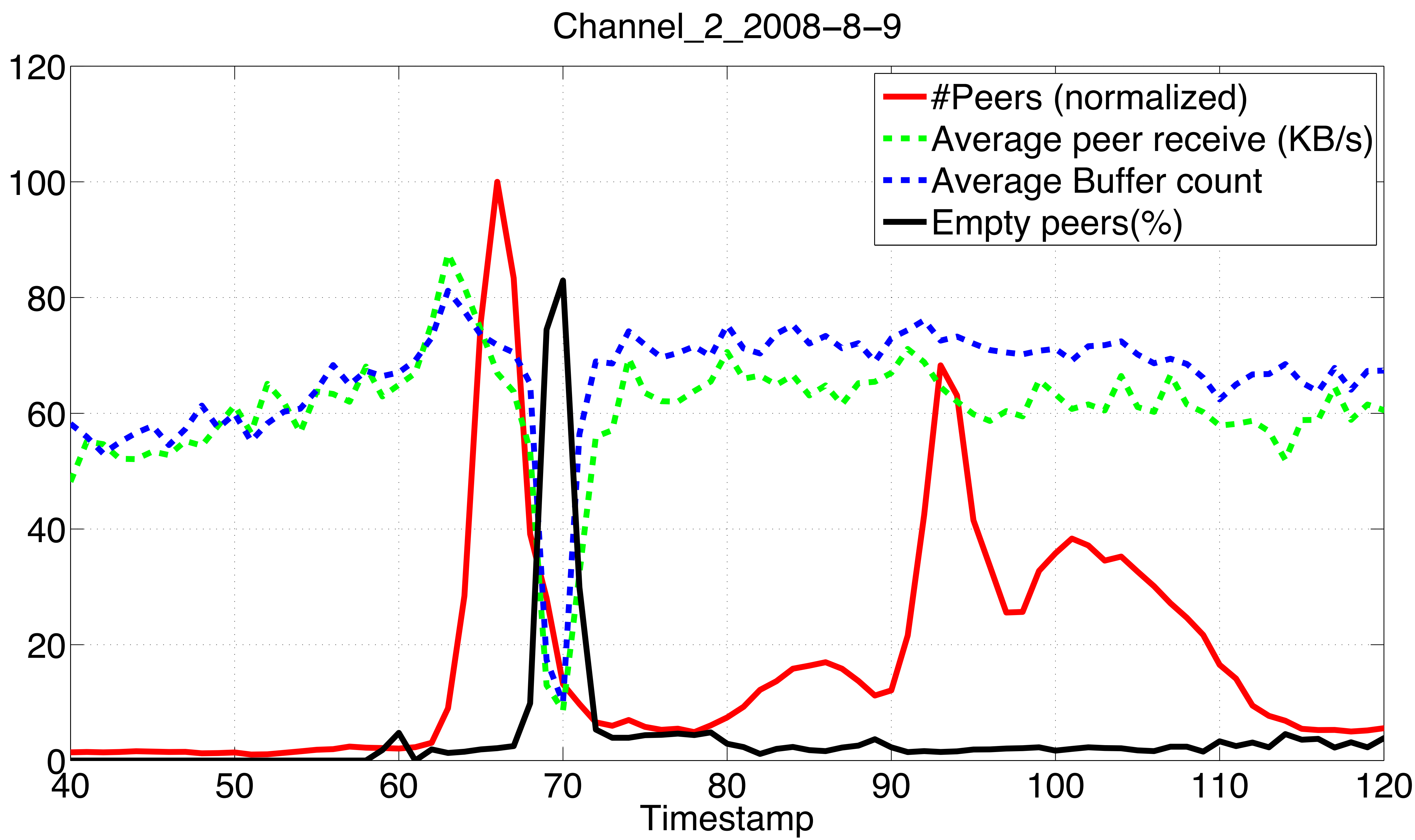



**Figure 4: The playback skip ratio during the period Aug 8-19, 2008 for 5 popular channels. The period includes 1634 timestamps for each channel.**

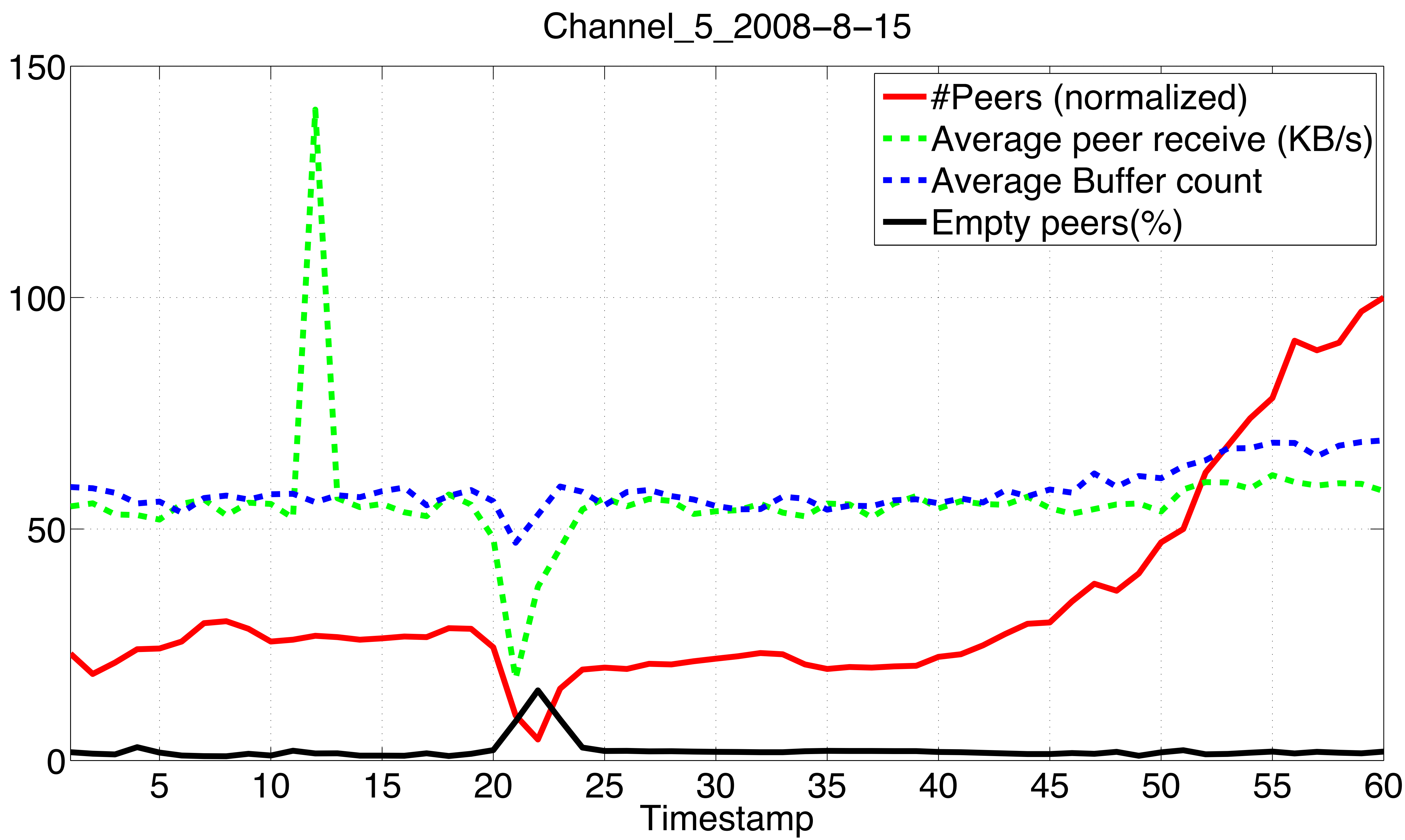
# Existing Problems

## The performance anomaly after a flash crowd.



# Existing Problems

## The performance anomaly after peer departures



# Objectives

# Objectives

## Improve Server Efficiency

- Predict peer upload contribution
- Allocate the right amount of server bandwidth that just achieves the required playback rate.

# Objectives

## Improve Server Efficiency

- Predict peer upload contribution
- Allocate the right amount of server bandwidth that just achieves the required playback rate.

## Handling Performance Anomalies

- Guarantee good quality even under severe peer churn
  - Flash crowd joins
  - Large-scale simultaneous peer departures
- Forecasting performance issues
- Guiding server/peer actions to prevent performance issues

# A Monitor & Learn Framework

## Enhance system “healthiness” by monitoring

- Collect statistics periodically
- Mining the operating rules and internal dependencies of the system
- Prediction is made possible
- Peer and server decisions should be based on online observations and learned rules.

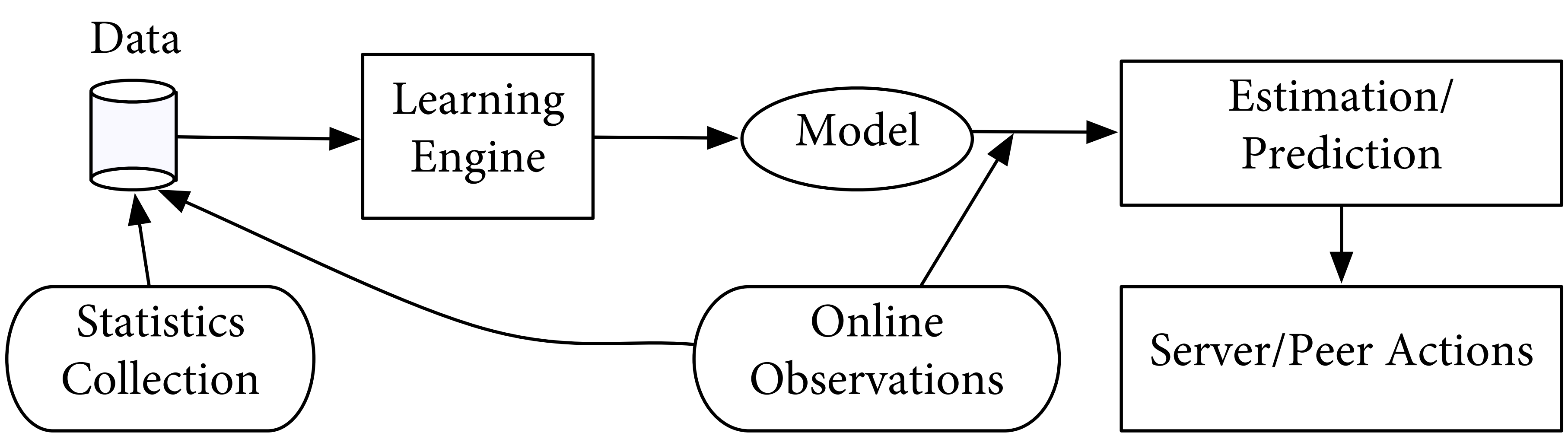



**Figure 1: The concept of self-diagnostic peer-assisted streaming.**

# A Monitor & Learn Framework

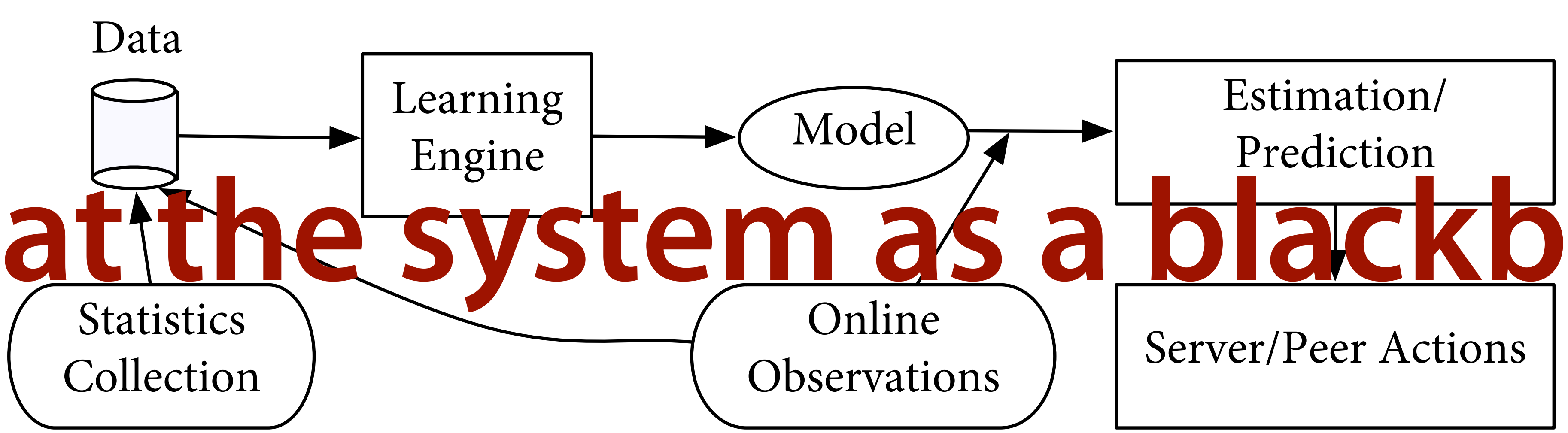
## Enhance system "healthiness" by monitoring

Collect statistics periodically

Mining the operating rules and internal dependencies of the system

Prediction is made possible

Peer and server decisions should be based on online observations and learned rules.




## Treat the system as a blackbox!

**Figure 1: The concept of self-diagnostic peer-assisted streaming.**

# Intelligent Server Provisioning

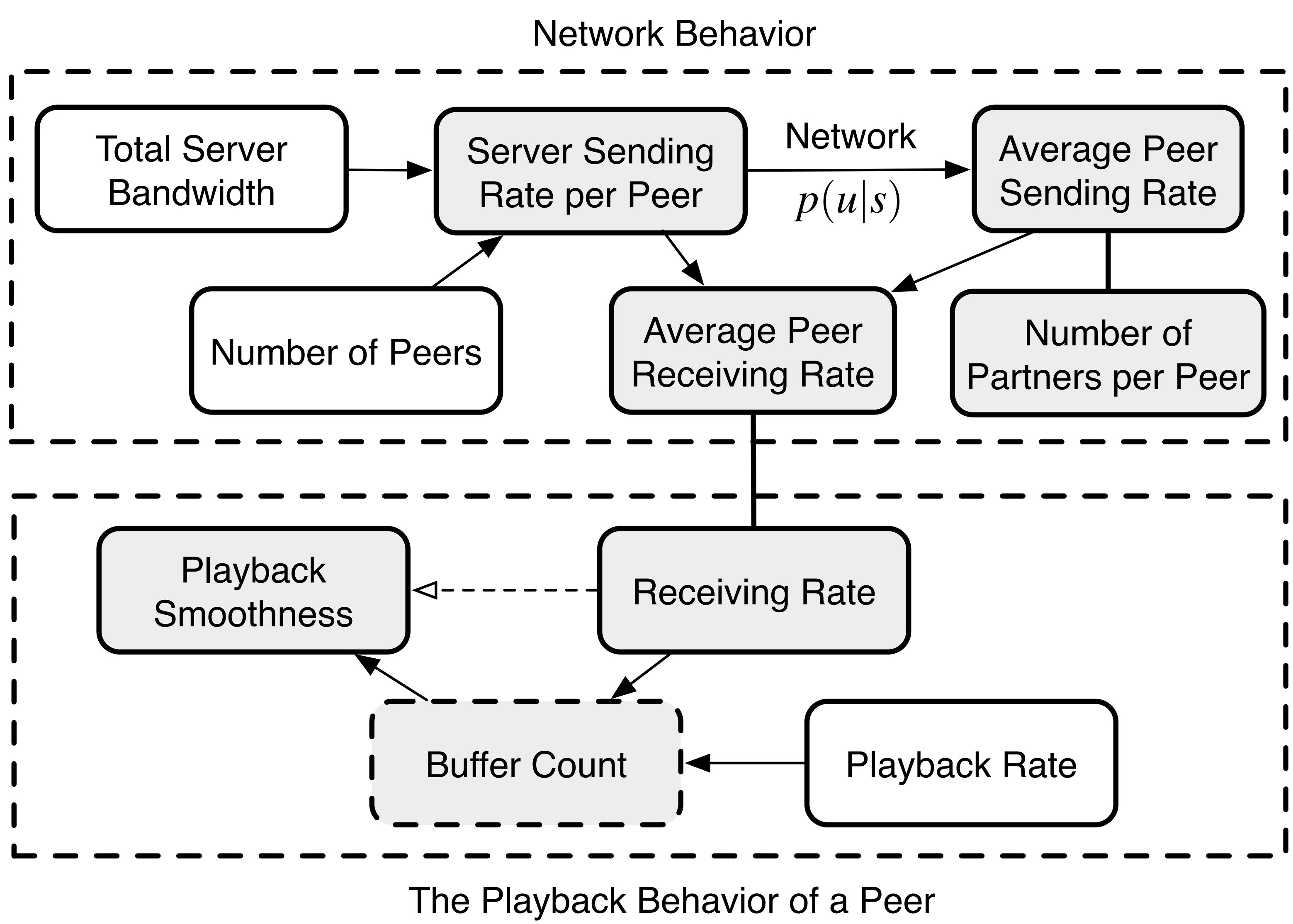
# Know the Return when Allocating Bandwidth

## For each channel, learn the relations (average quantities)

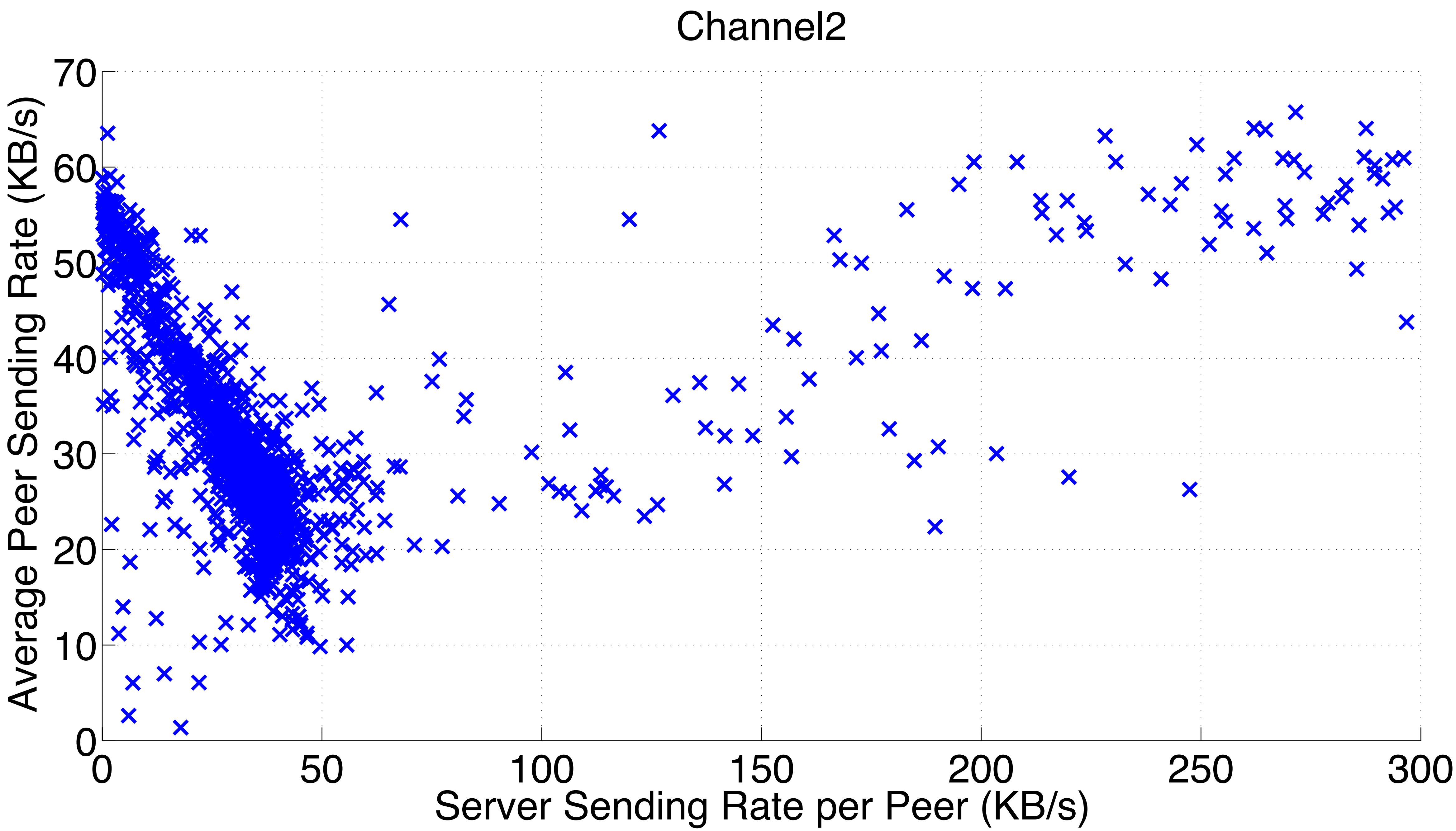
$$\textit{server send} \longrightarrow \textit{peer send}$$

$$\textit{peer receive} = \textit{server send} + \textit{peer send}$$

$$\textit{server send} \longrightarrow \textit{peer receive}$$

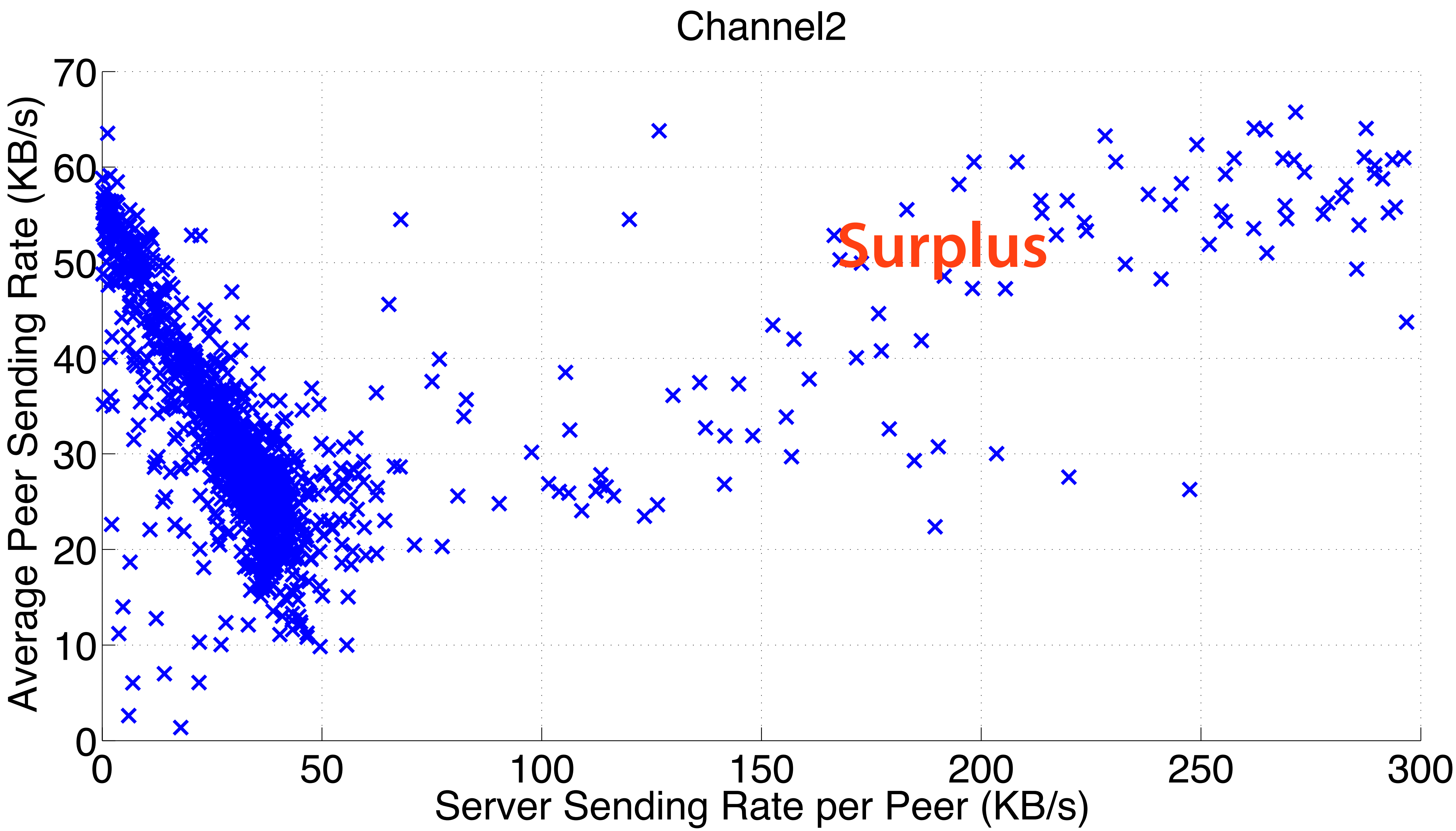


# What's the *Server Send - Peer Send* Relation?



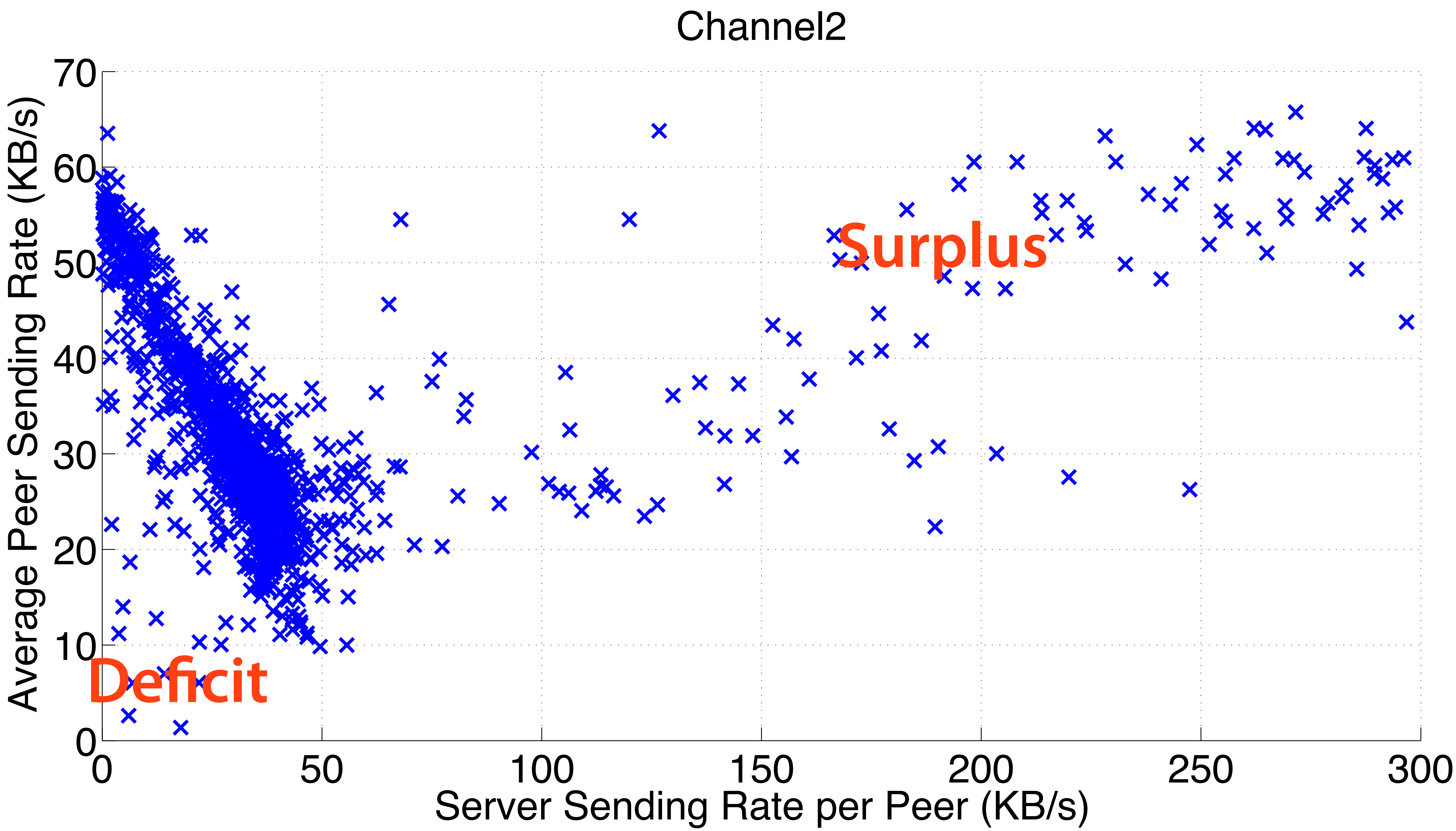

Collected data {(*server send*, *peer send*)} can be classified into three states. Only the normal state is desired.

# What's the *Server Send* - *Peer Send* Relation?



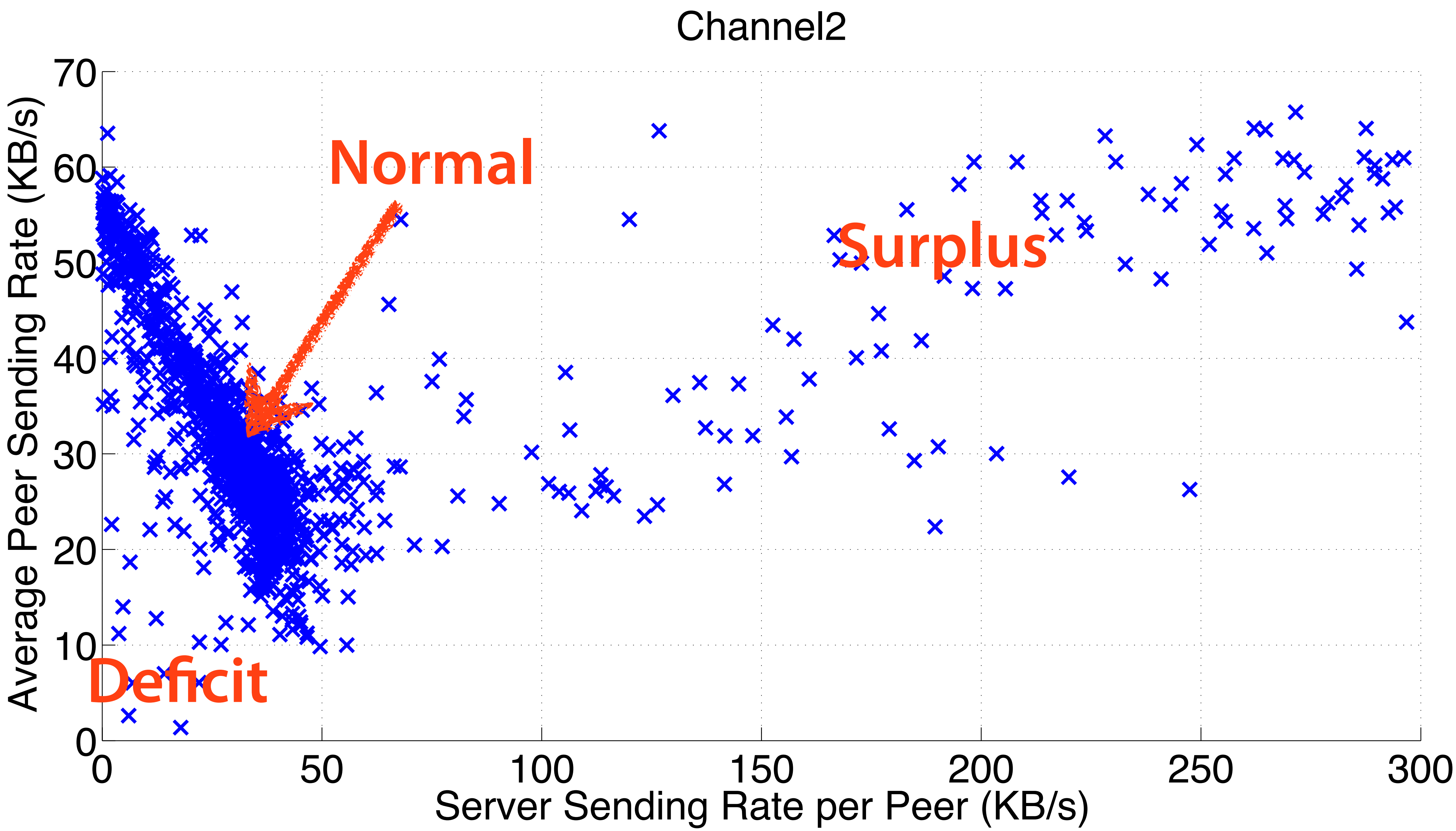

Collected data {(*server send*, *peer send*)} can be classified into three states. Only the normal state is desired.

# What's the *Server Send* - *Peer Send* Relation?




Collected data {(*server send*, *peer send*)} can be classified into three states. Only the normal state is desired.

# What's the *Server Send - Peer Send* Relation?




Collected data {(*server send*, *peer send*)} can be classified into three states. Only the normal state is desired.

# Iterative States Classification & Learning

Learn *server send - peer send* relation in the normal state, while classifying the states.

A novel iterative clustering-learning algorithm similar to *“K-means clustering”*.

# Iterative States Classification & Learning

Learn *server send - peer send* relation in the normal state, while classifying the states.

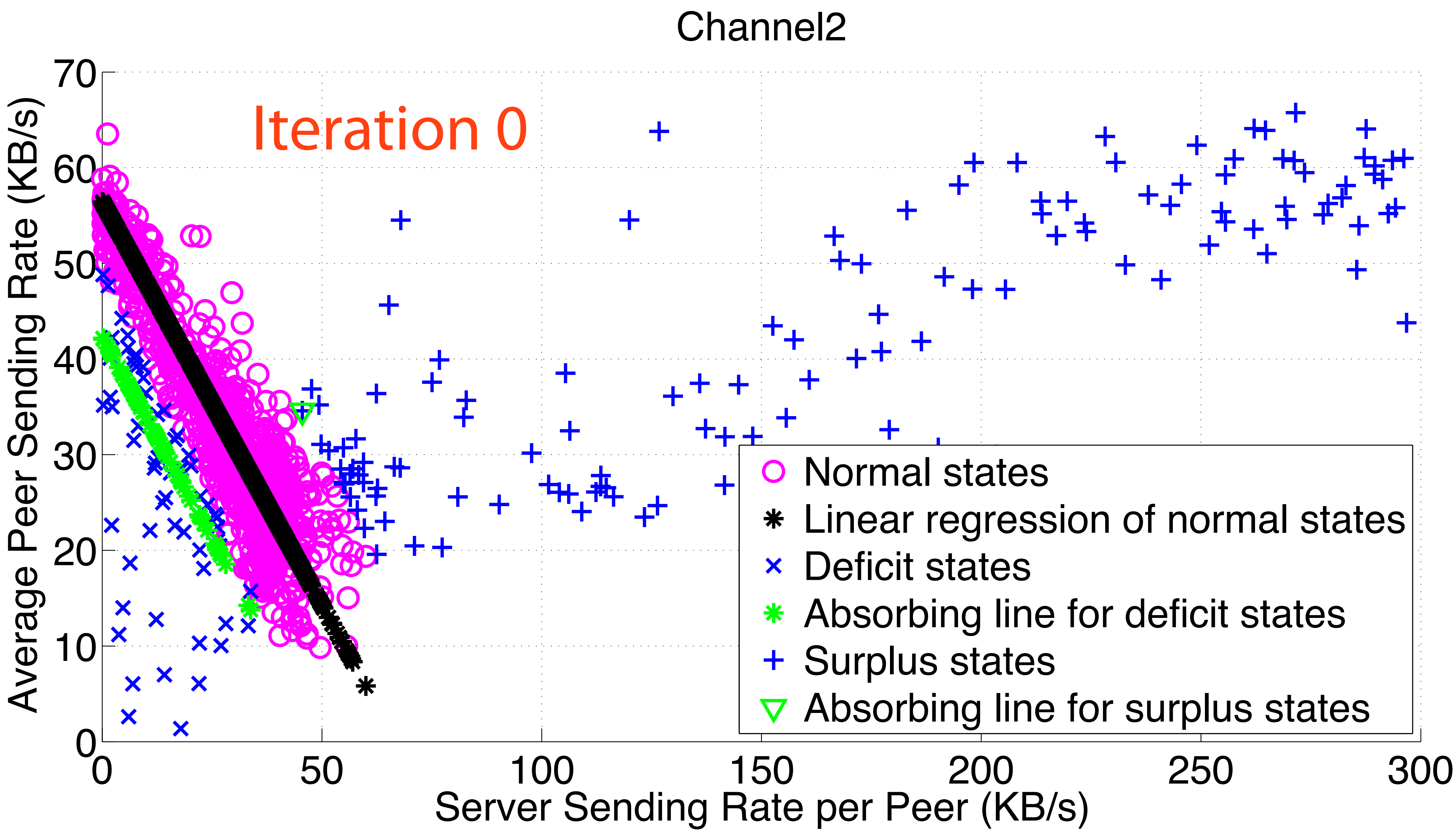
A novel iterative clustering-learning algorithm similar to *"K-means clustering"*.

Iteration 0

# Iterative States Classification & Learning

Learn *server send - peer send* relation in the normal state, while classifying the states.

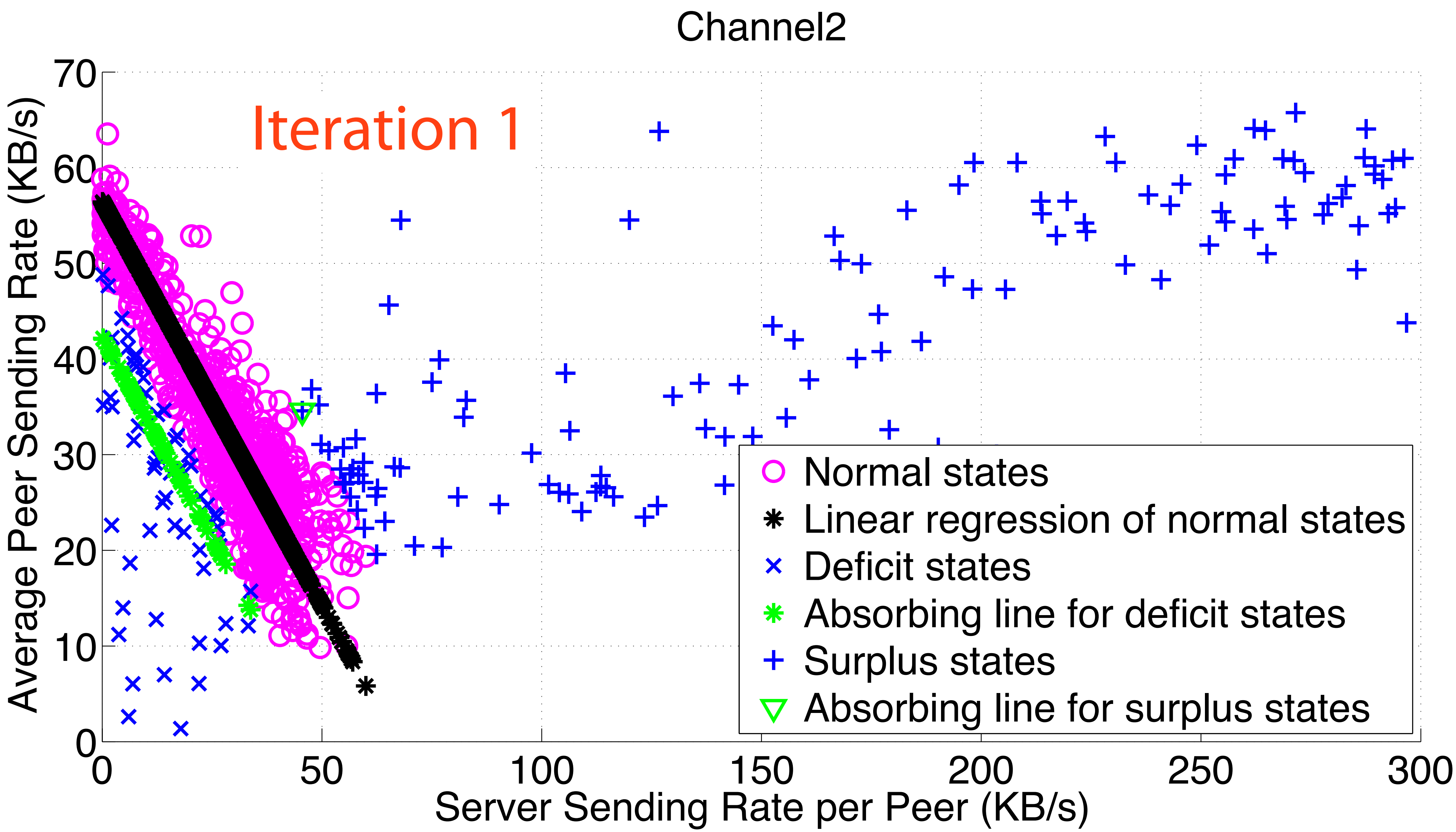
A novel iterative clustering-learning algorithm similar to *"K-means clustering"*.



# Iterative States Classification & Learning

Learn *server send - peer send* relation in the normal state, while classifying the states.

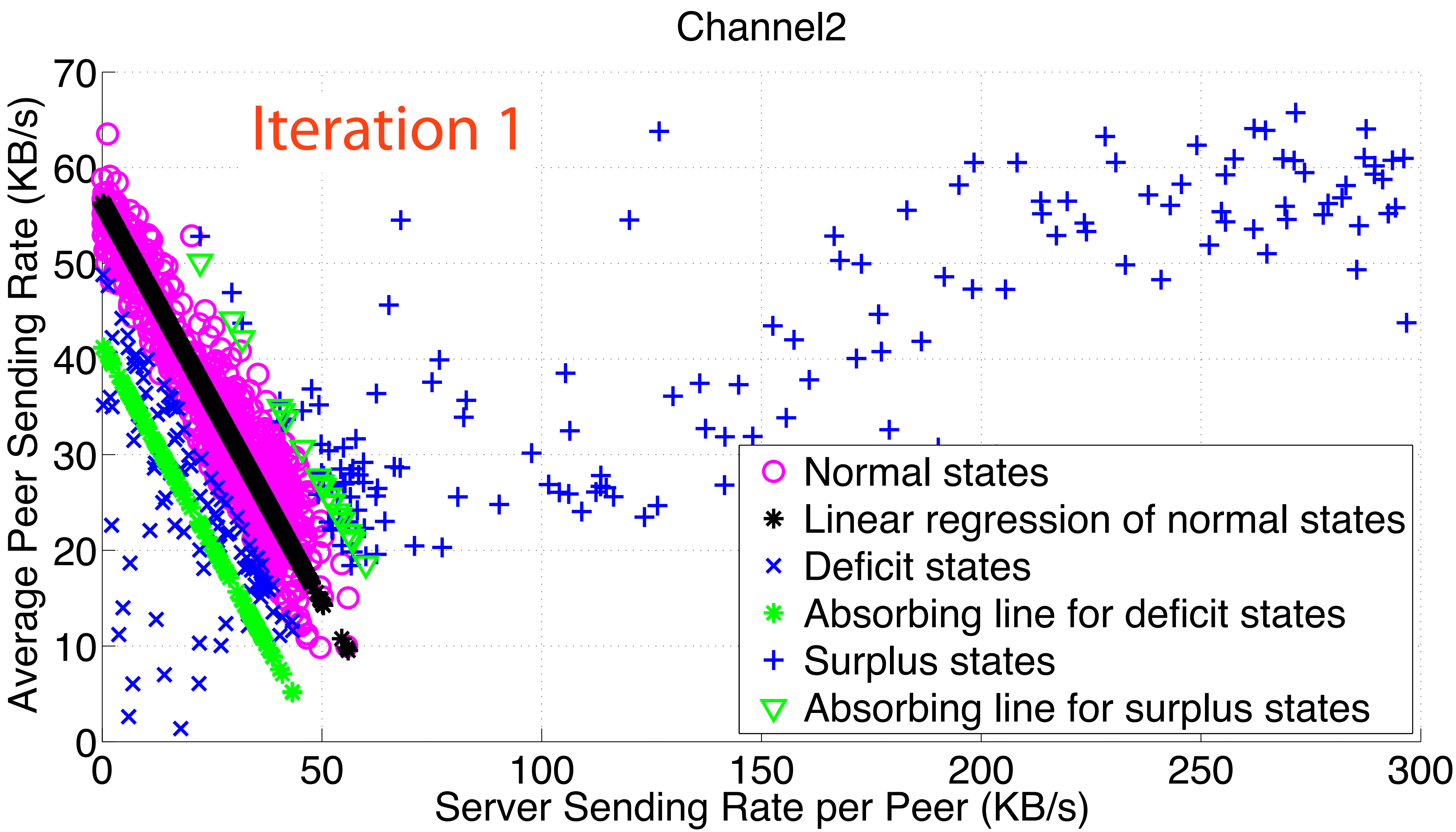
A novel iterative clustering-learning algorithm similar to *"K-means clustering"*.



# Iterative States Classification & Learning

Learn *server send - peer send* relation in the normal state, while classifying the states.

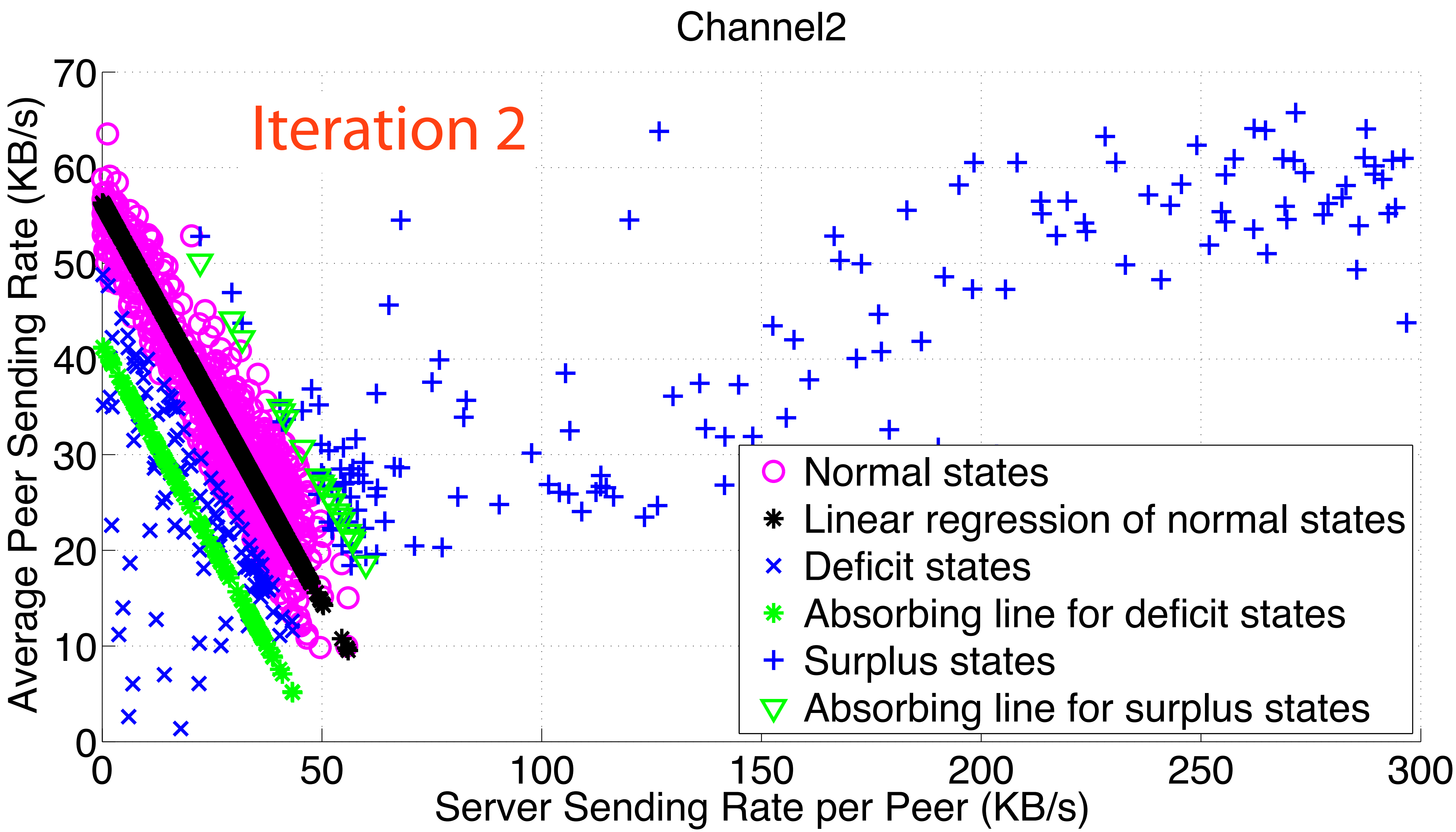
A novel iterative clustering-learning algorithm similar to *"K-means clustering"*.



# Iterative States Classification & Learning

Learn *server send - peer send* relation in the normal state, while classifying the states.

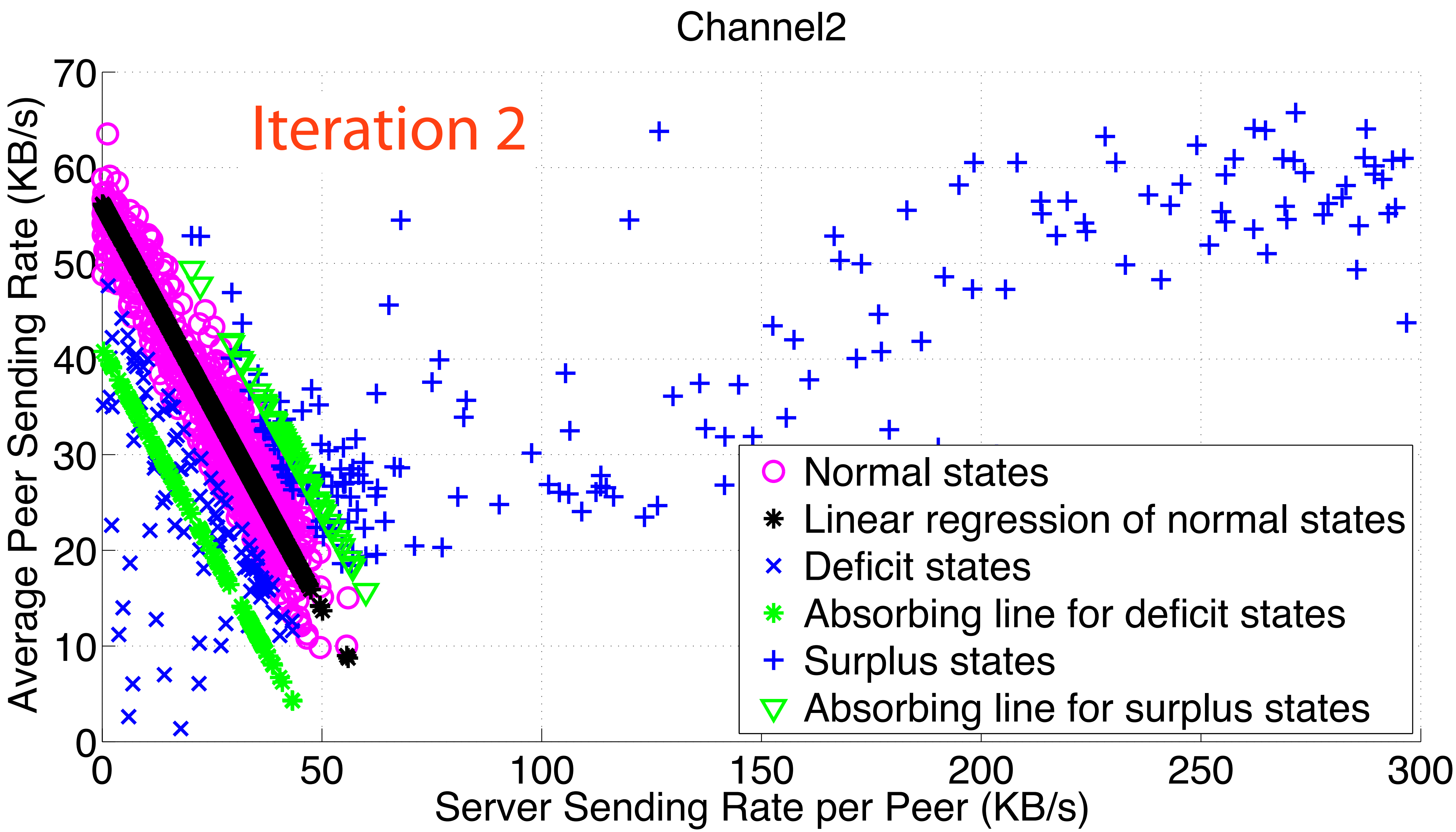
A novel iterative clustering-learning algorithm similar to *"K-means clustering"*.



# Iterative States Classification & Learning

Learn *server send - peer send* relation in the normal state, while classifying the states.

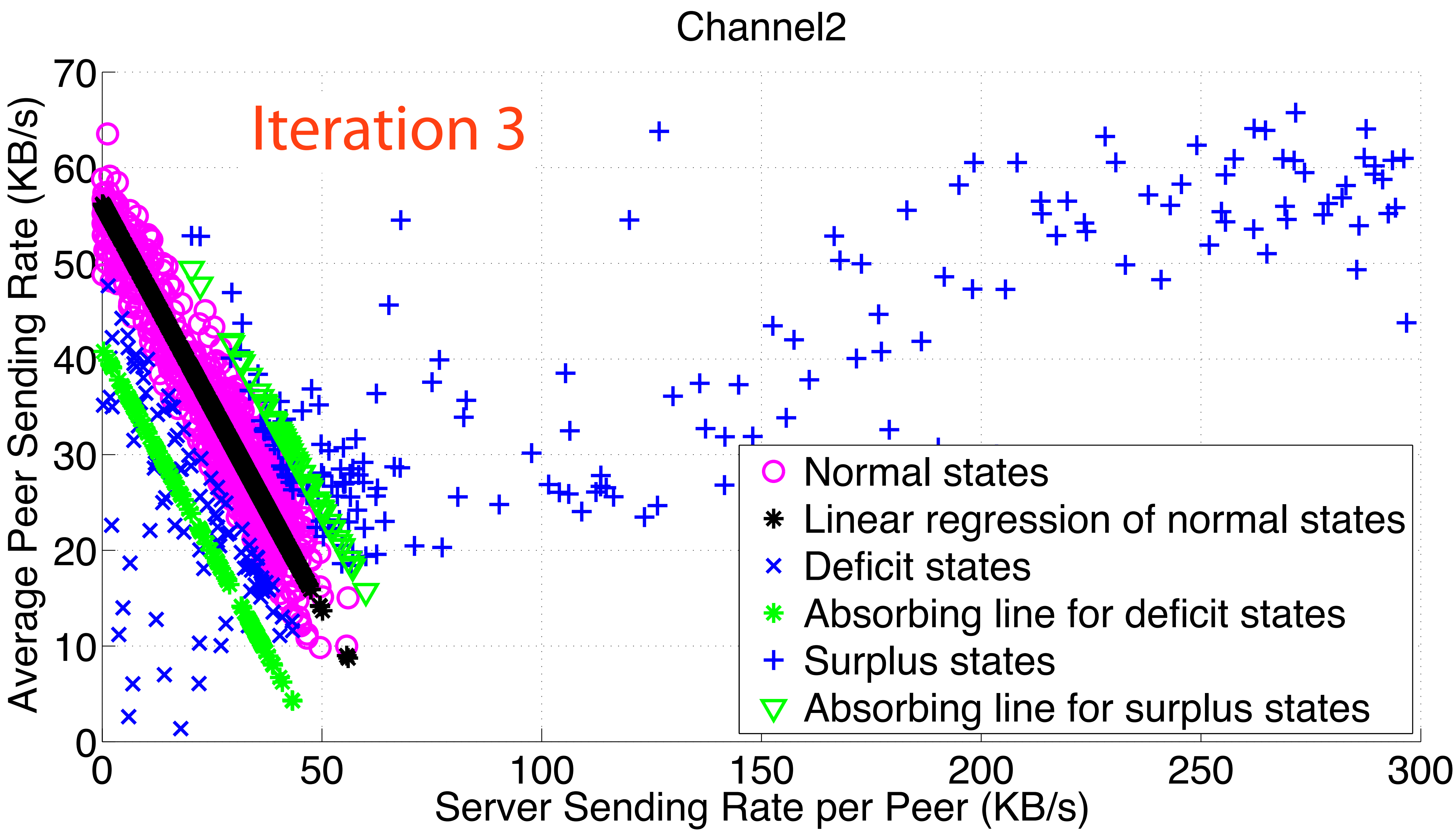
A novel iterative clustering-learning algorithm similar to *"K-means clustering"*.



# Iterative States Classification & Learning

Learn *server send - peer send* relation in the normal state, while classifying the states.

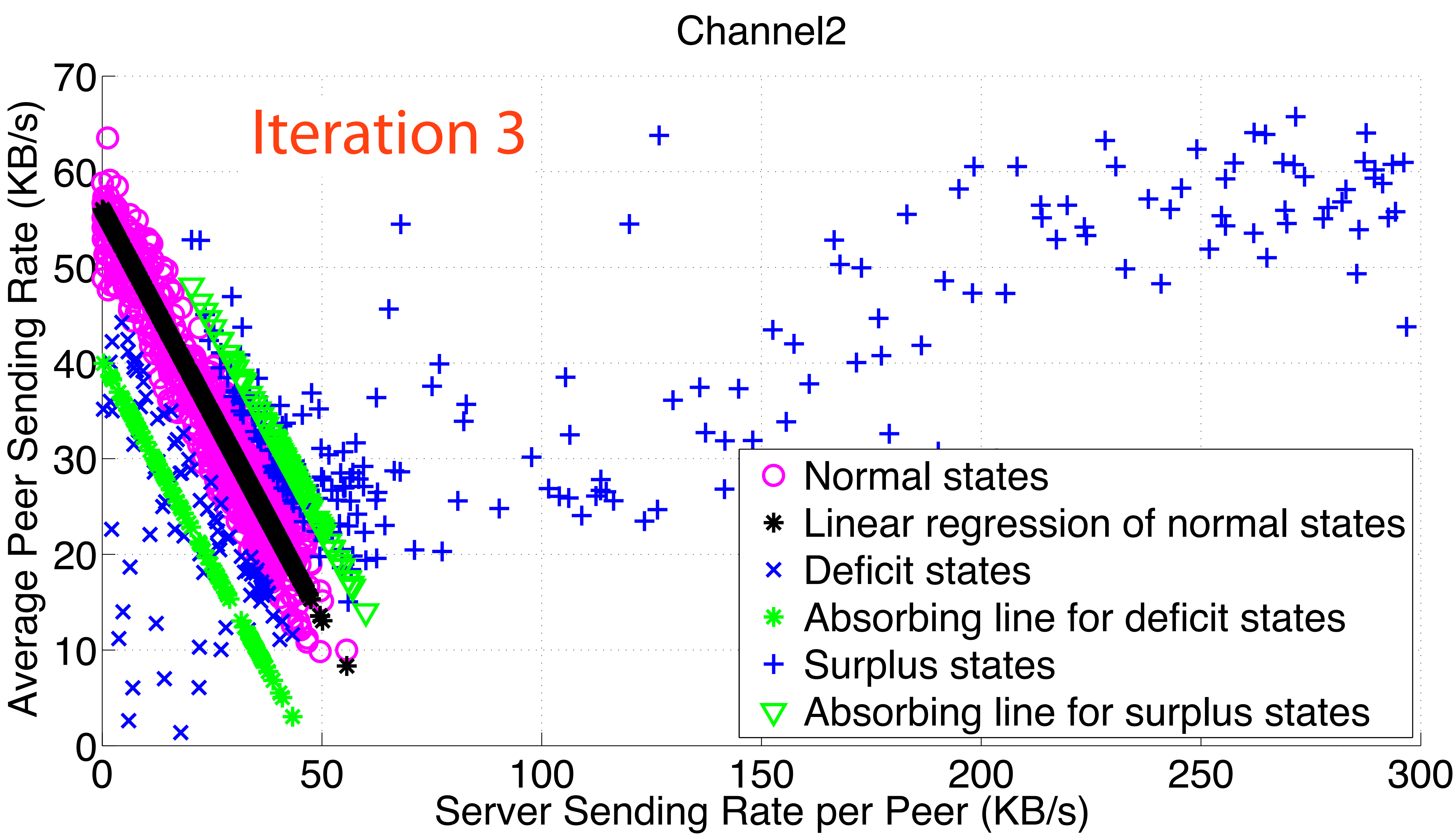
A novel iterative clustering-learning algorithm similar to *"K-means clustering"*.



# Iterative States Classification & Learning

Learn *server send - peer send* relation in the normal state, while classifying the states.

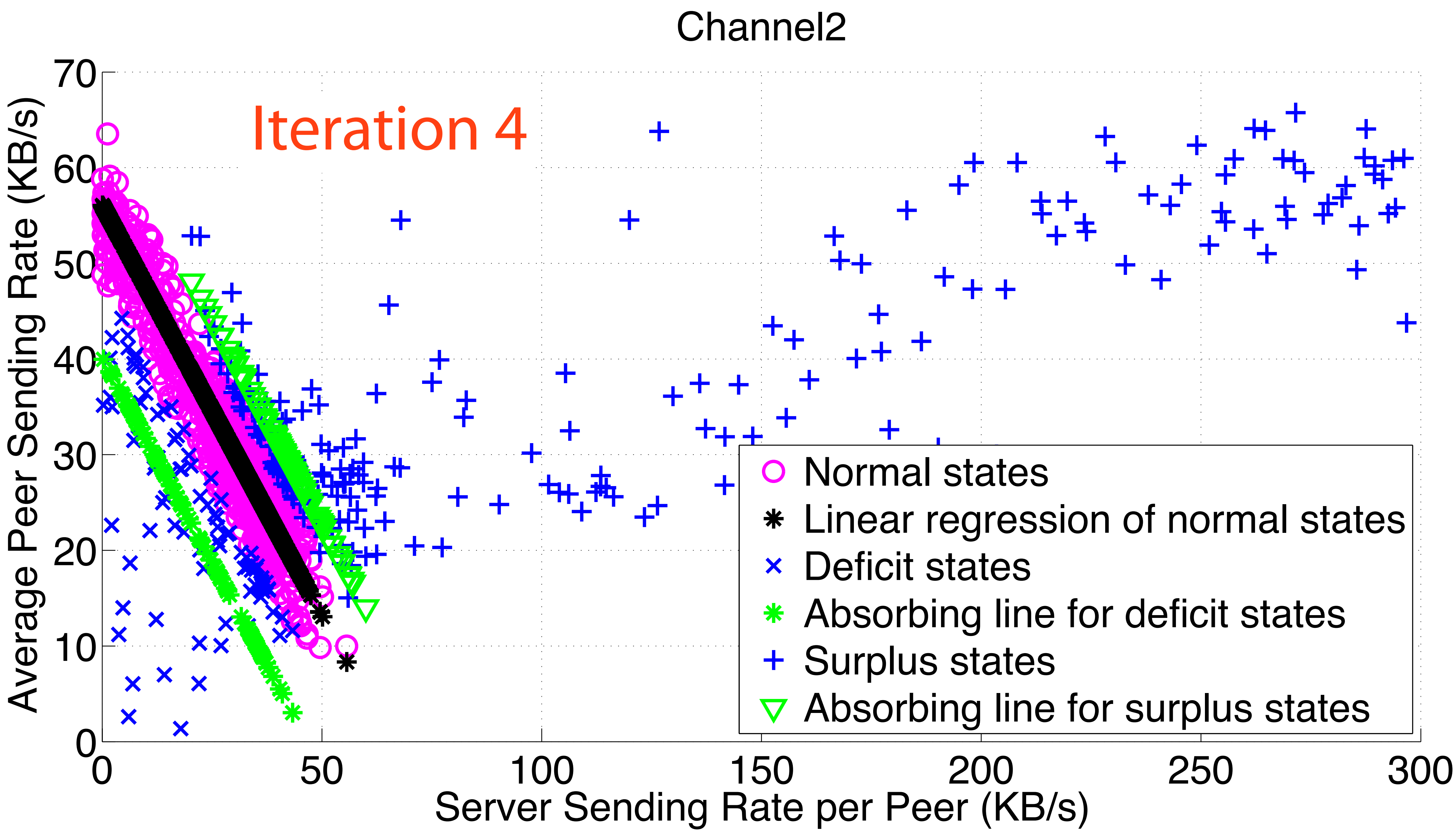
A novel iterative clustering-learning algorithm similar to *"K-means clustering"*.



# Iterative States Classification & Learning

Learn *server send - peer send* relation in the normal state, while classifying the states.

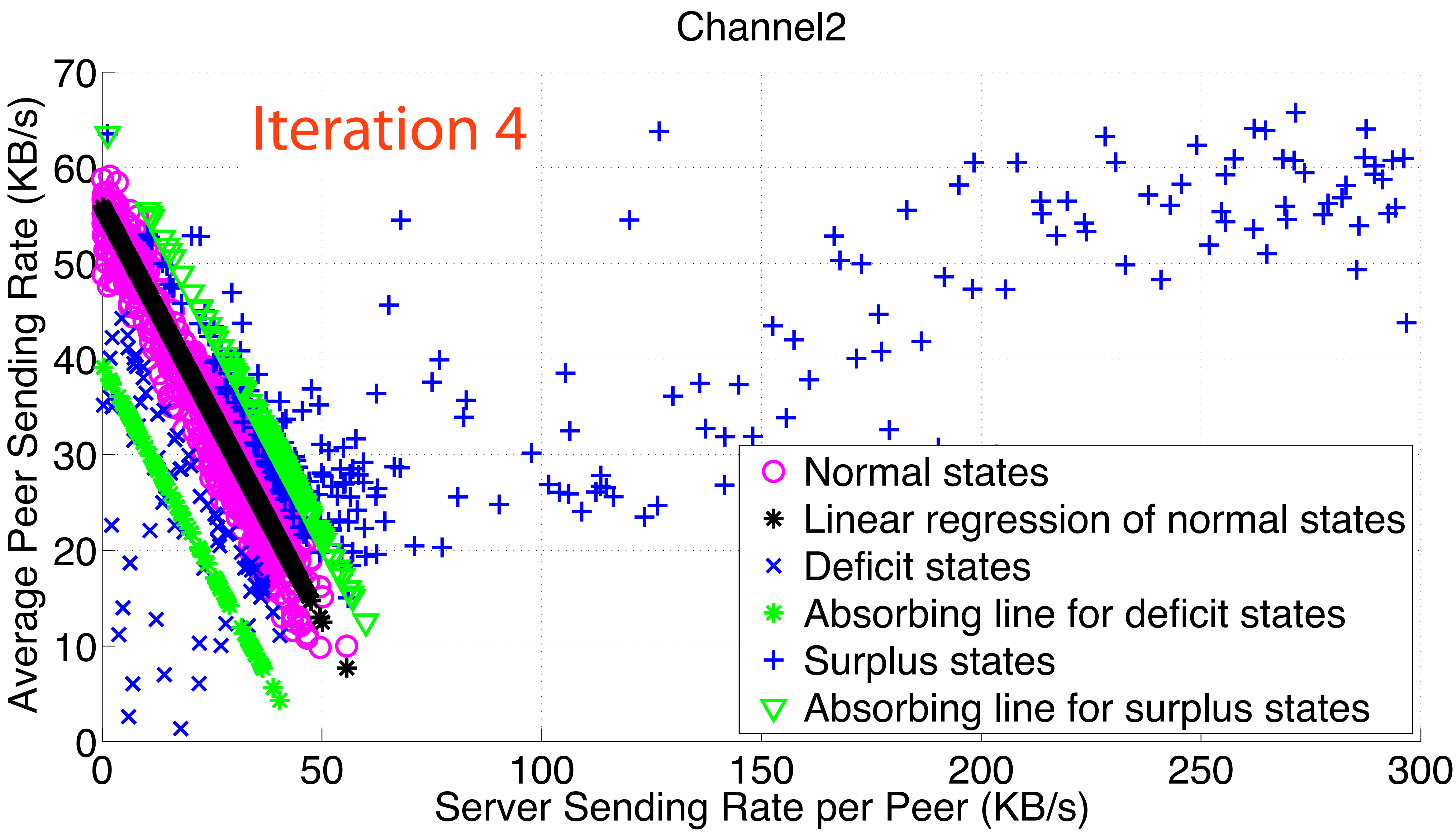
A novel iterative clustering-learning algorithm similar to *"K-means clustering"*.



# Iterative States Classification & Learning

Learn *server send - peer send* relation in the normal state, while classifying the states.

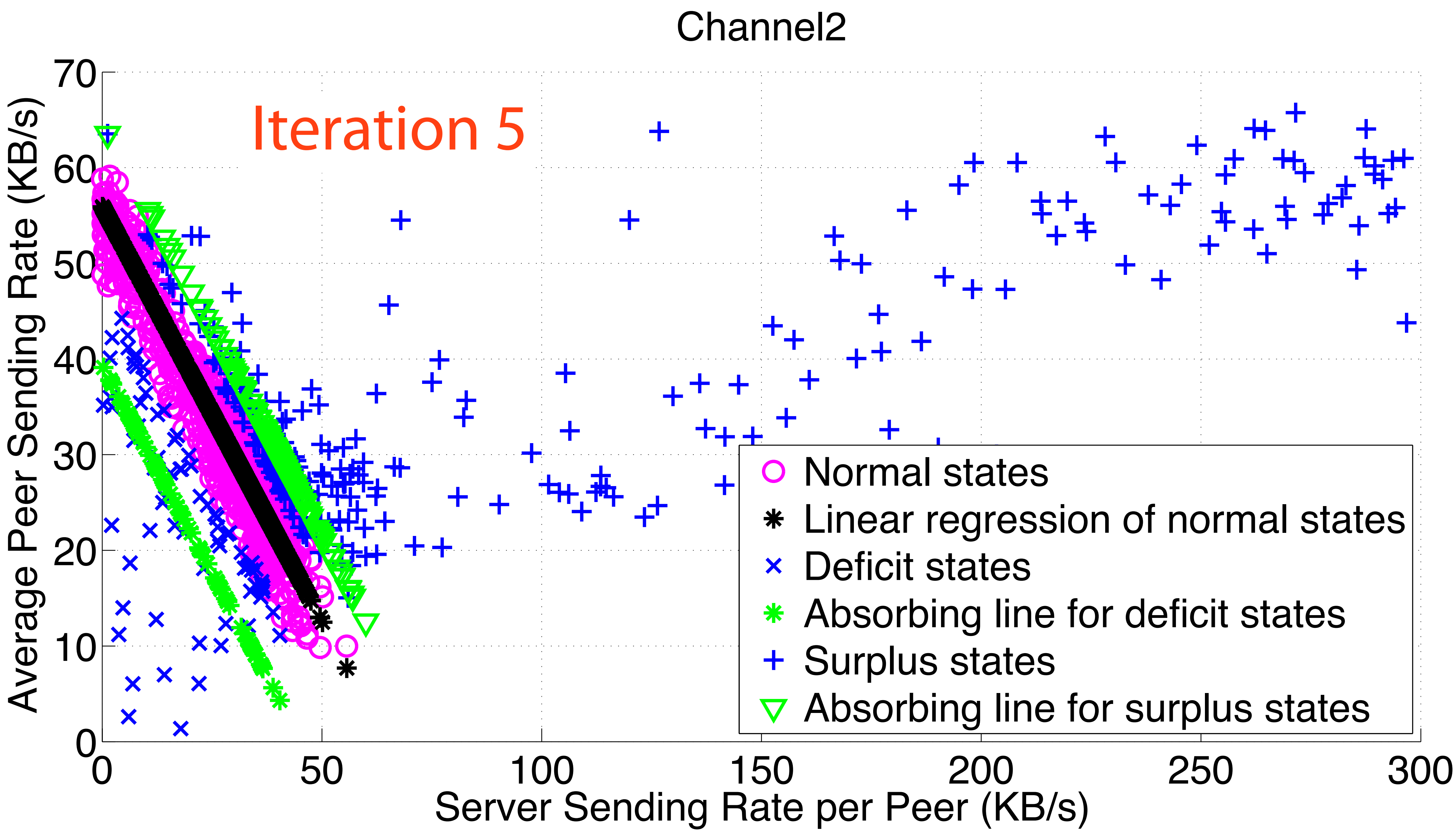
A novel iterative clustering-learning algorithm similar to *"K-means clustering"*.



# Iterative States Classification & Learning

Learn *server send - peer send* relation in the normal state, while classifying the states.

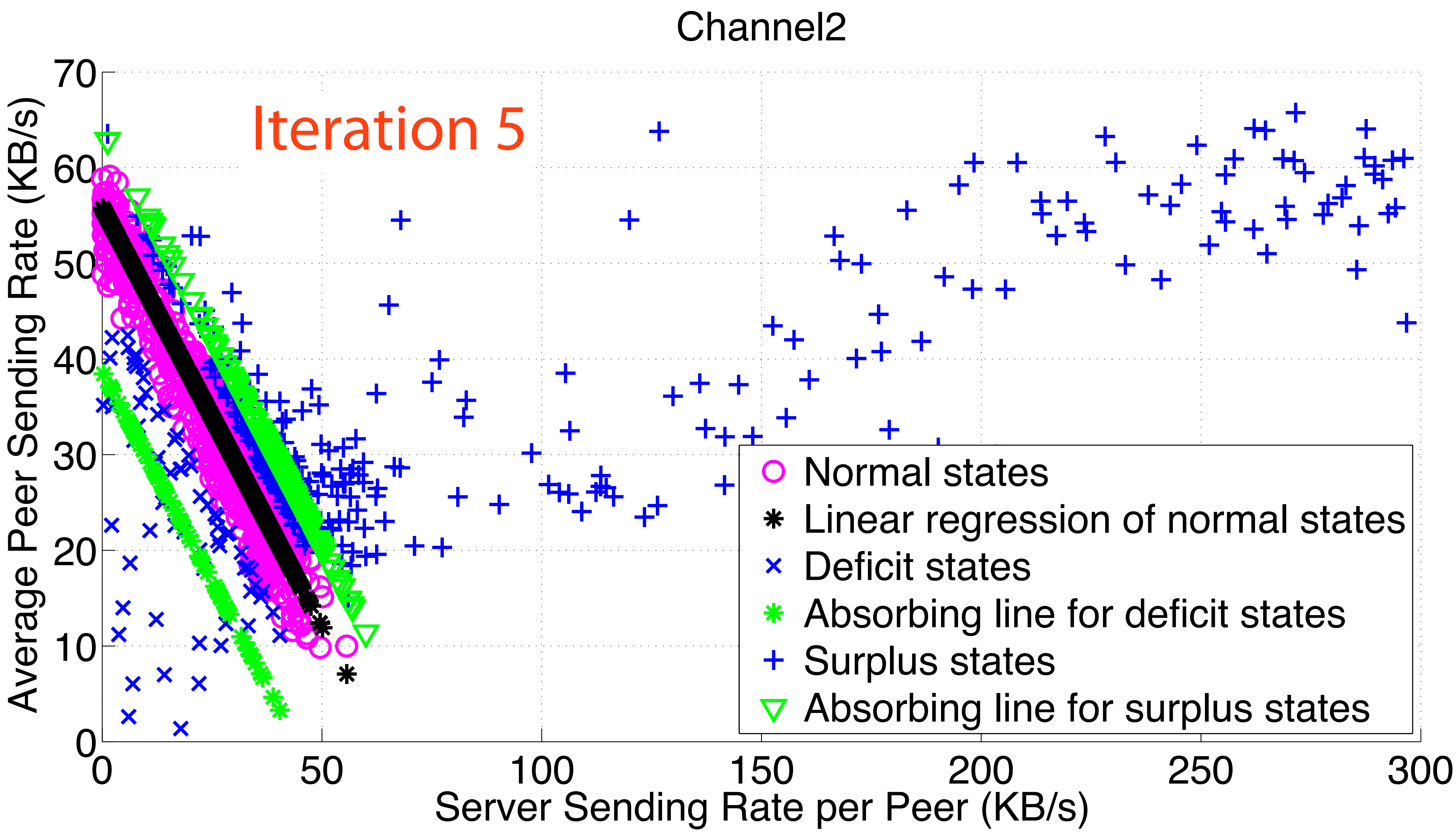
A novel iterative clustering-learning algorithm similar to *"K-means clustering"*.



# Iterative States Classification & Learning

Learn *server send - peer send* relation in the normal state, while classifying the states.

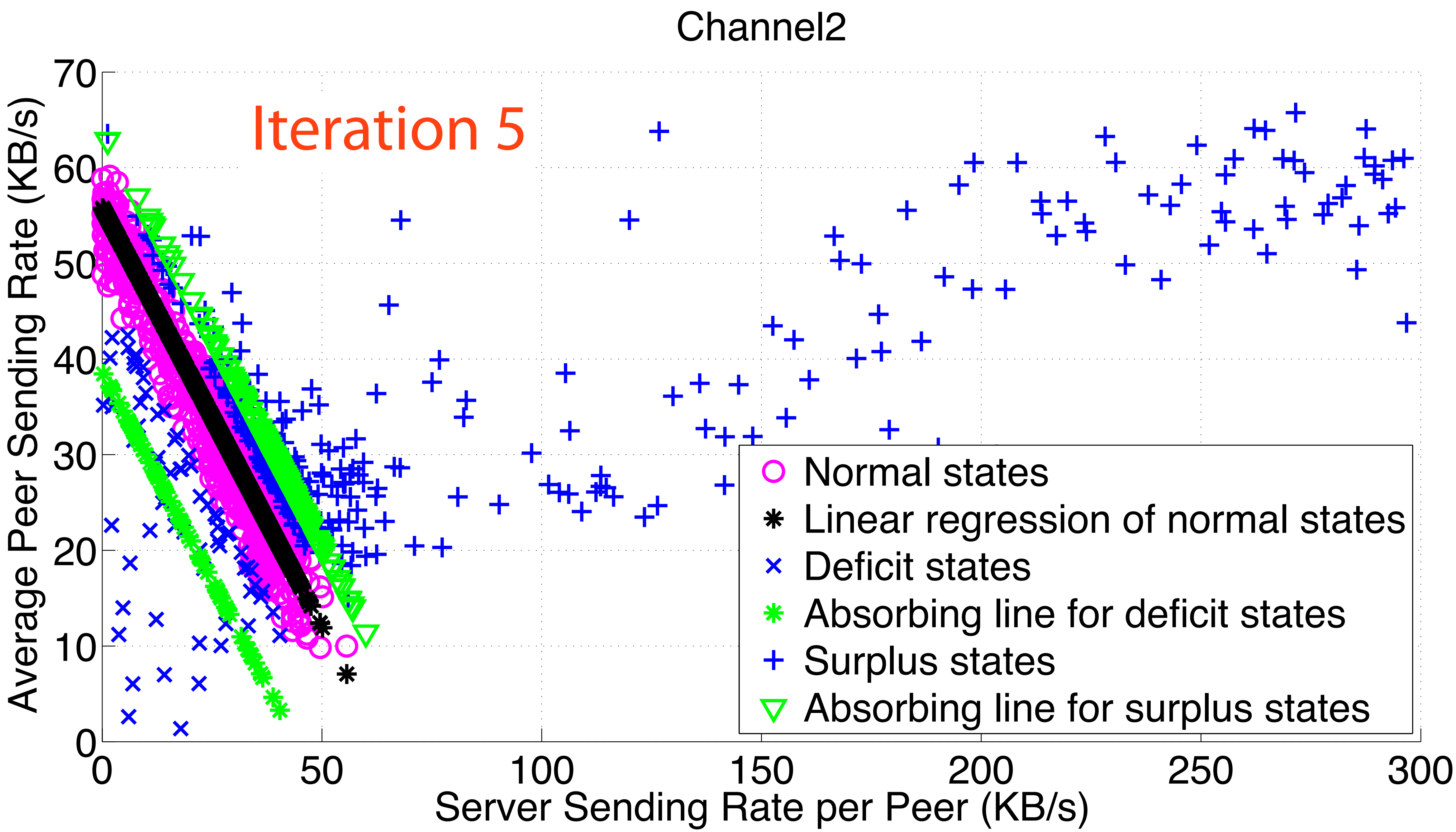
A novel iterative clustering-learning algorithm similar to *"K-means clustering"*.



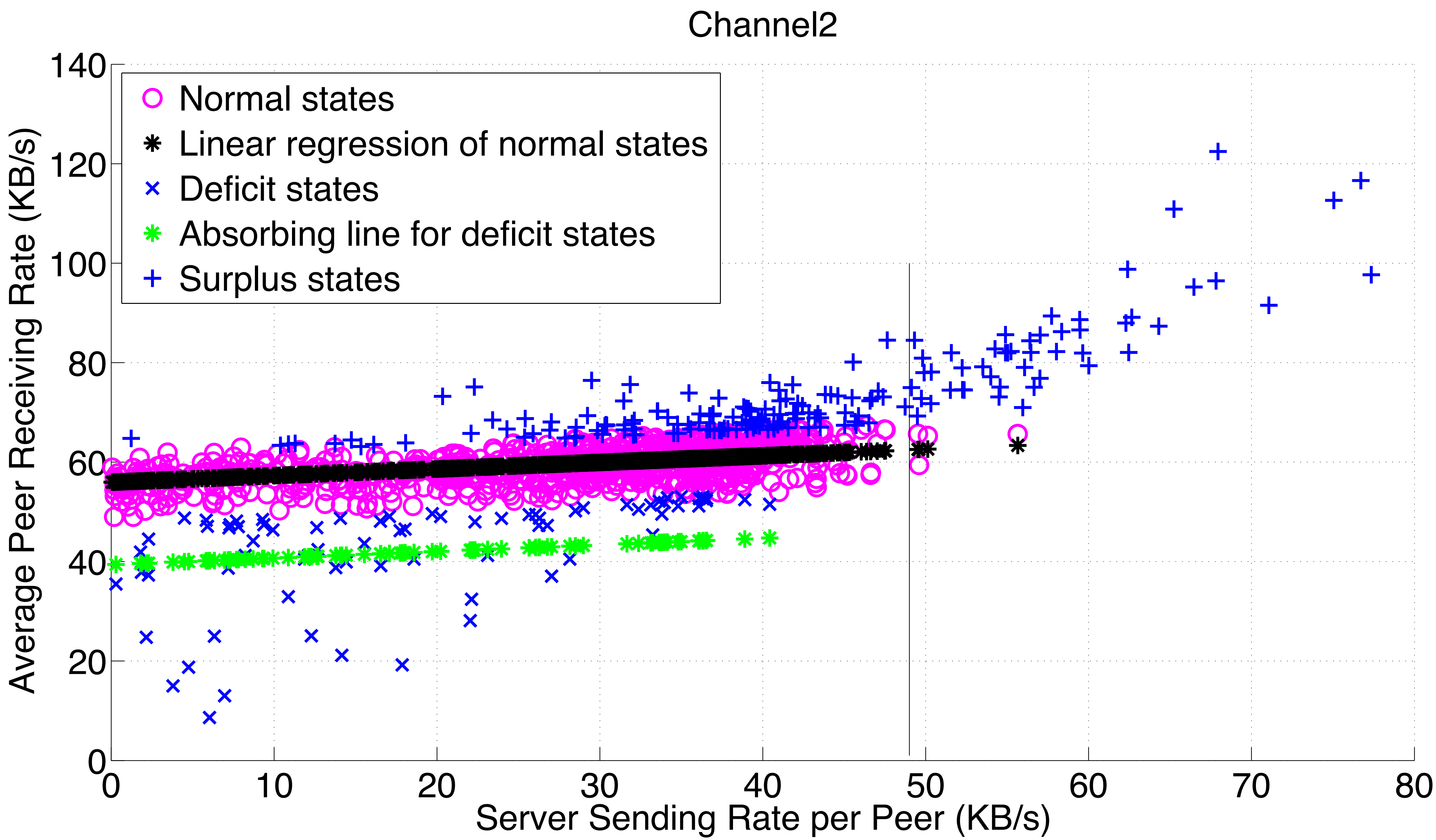
# Iterative States Classification & Learning

The more the server sends, the less the peers send.

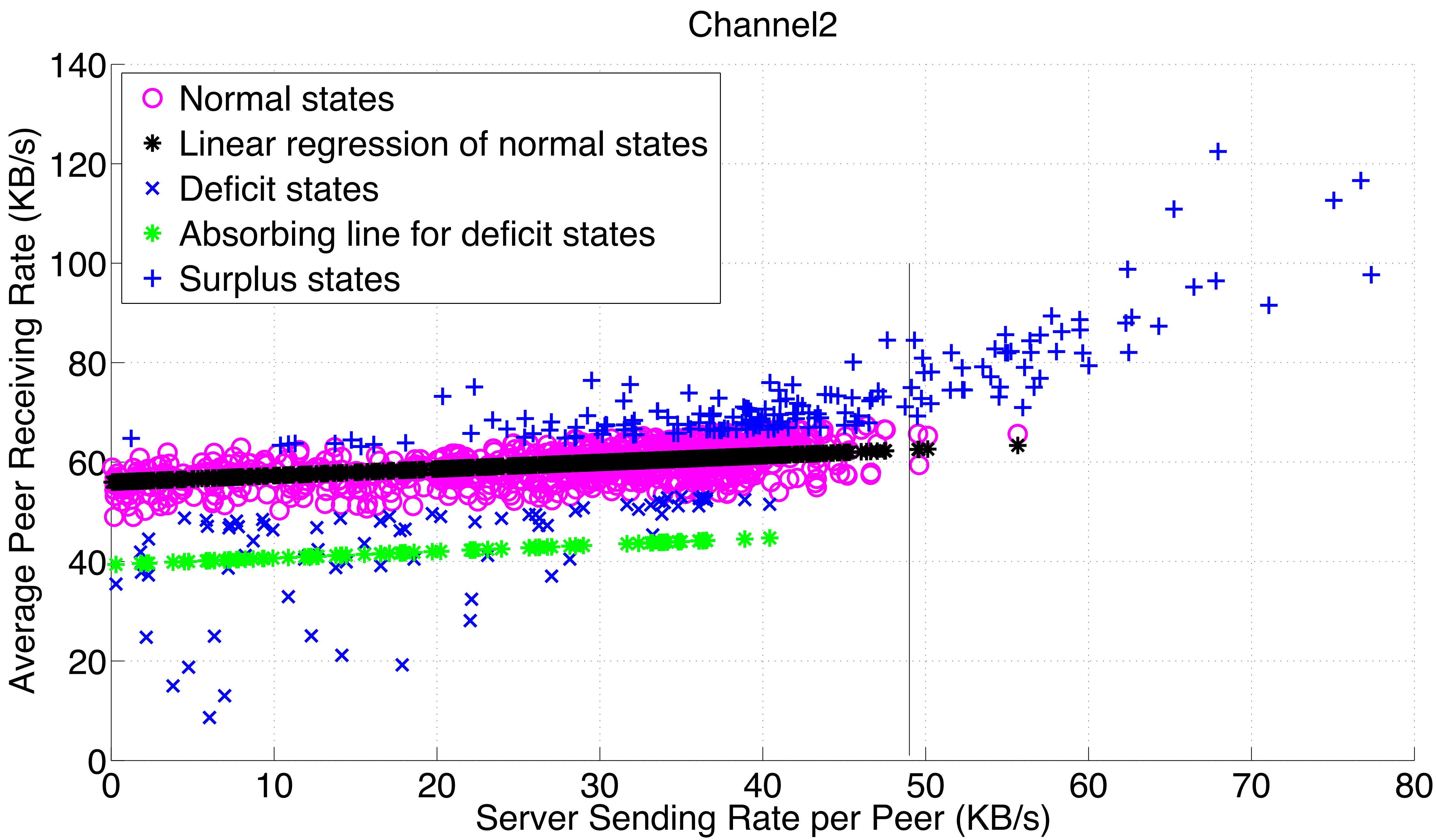
The sum of the two is around  playback rate, but increases slightly.



# Server Rate Limiting: Soft Decision



# Server Rate Limiting: Soft Decision

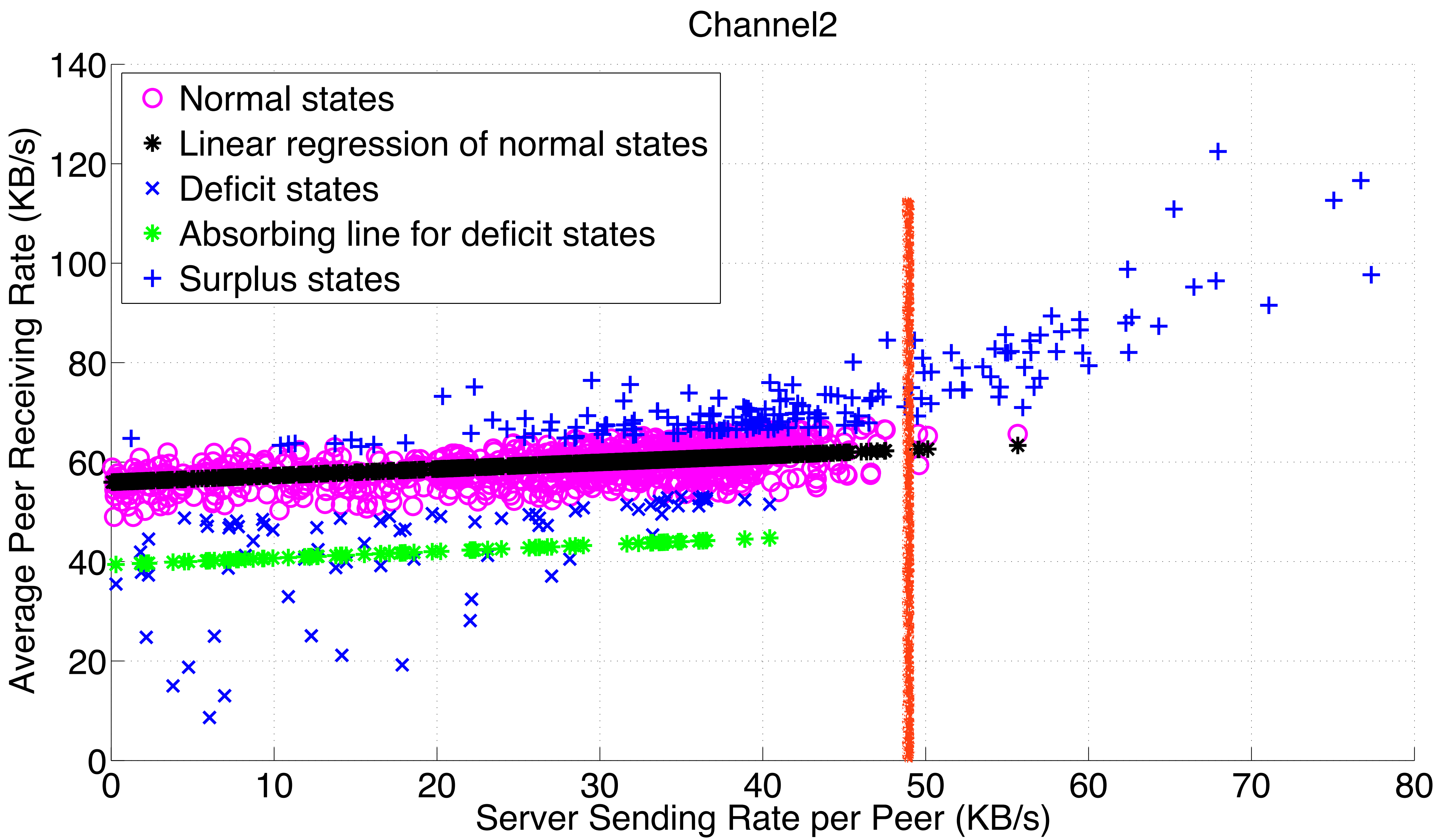



*peer receive = server send + peer send*

Set a cap on server sending rate to avoid surplus states

Increase server send until the system is more likely to operate in surplus states.

# Server Rate Limiting: Soft Decision

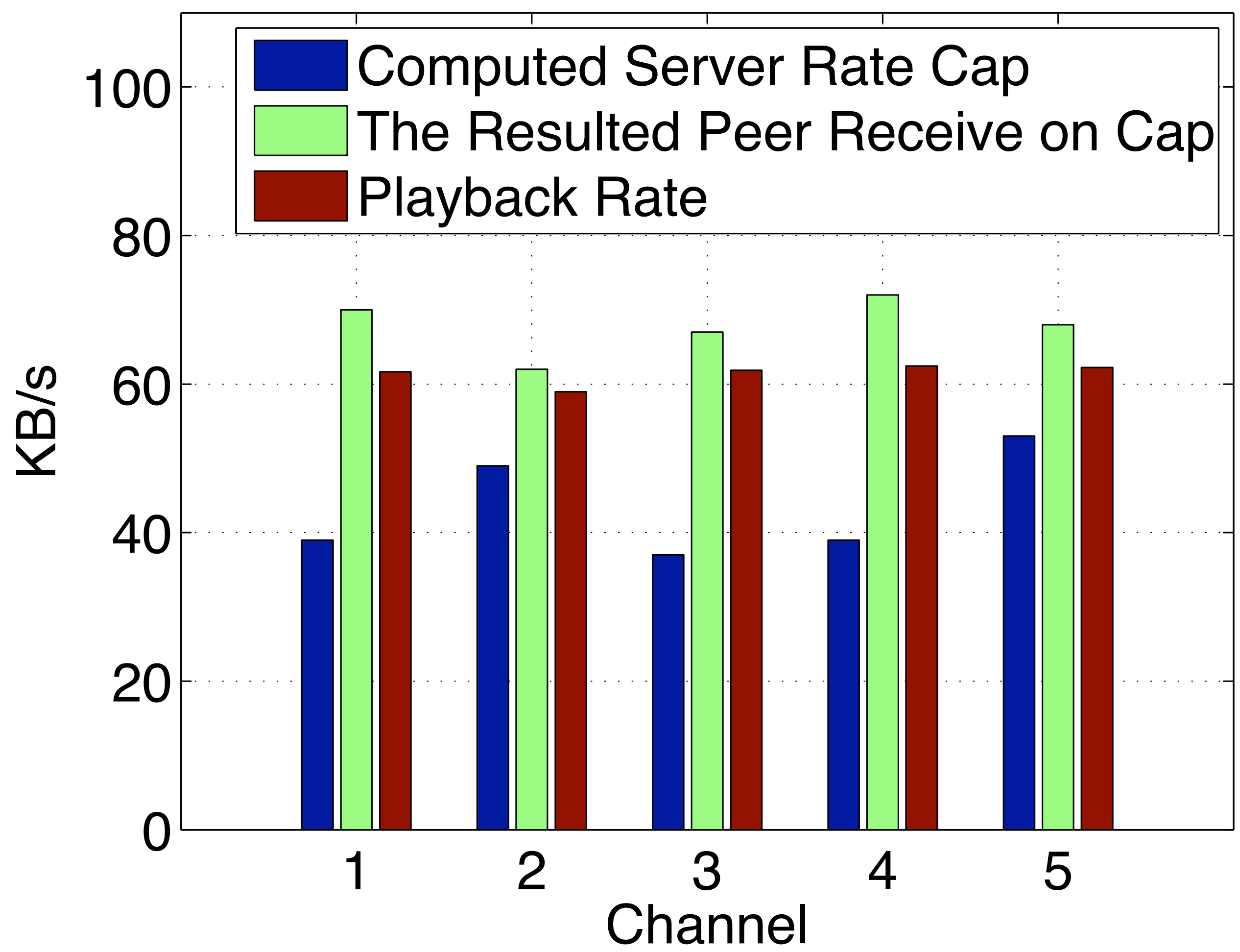


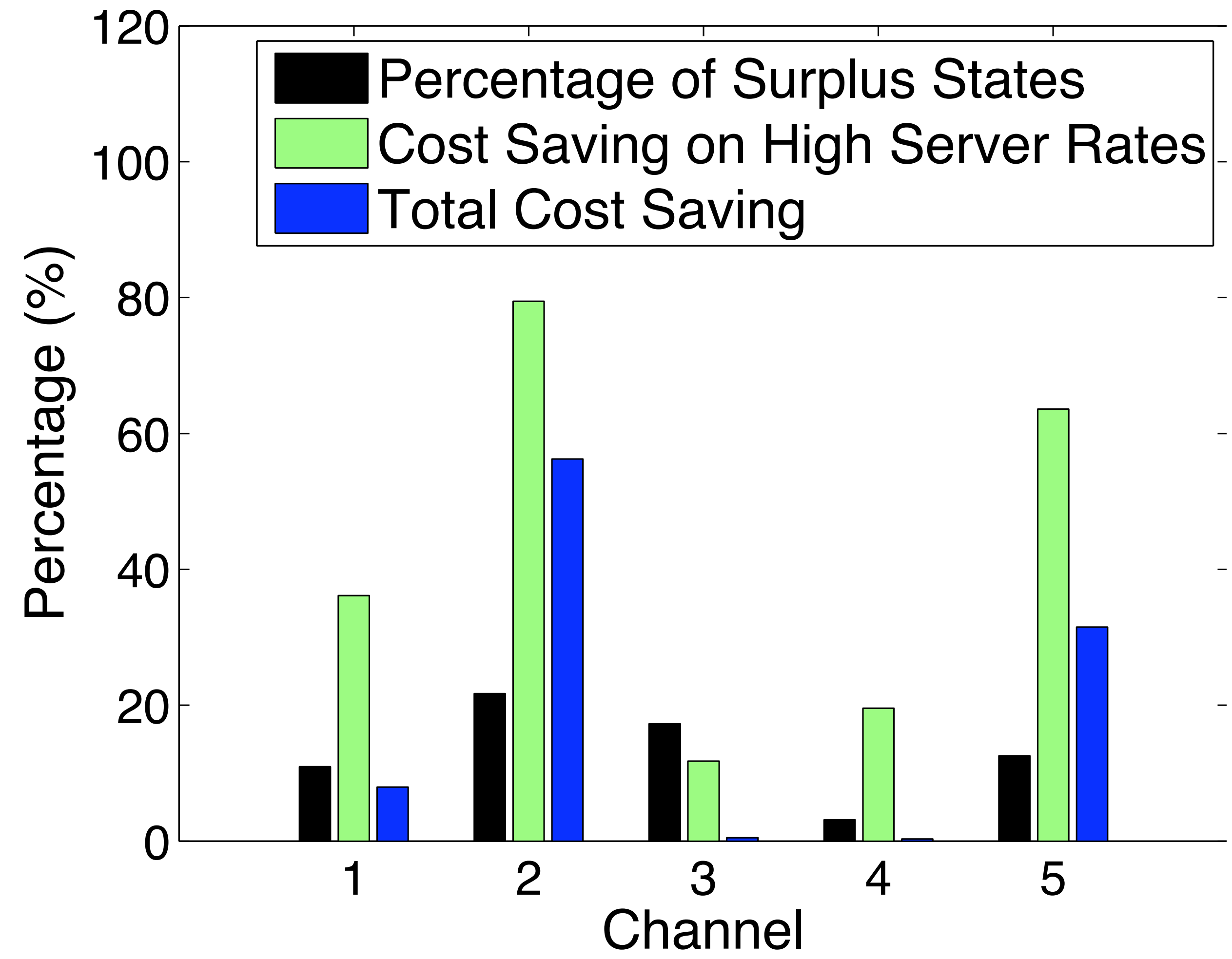

*peer receive = server send + peer send*

Set a cap on server sending rate to avoid surplus states

Increase server send until the system is more likely to operate in surplus states.

# Server Rate Limiting




Total cost saving close to 60% in some channels

# Optimized Server Allocation

# Optimized Server Allocation

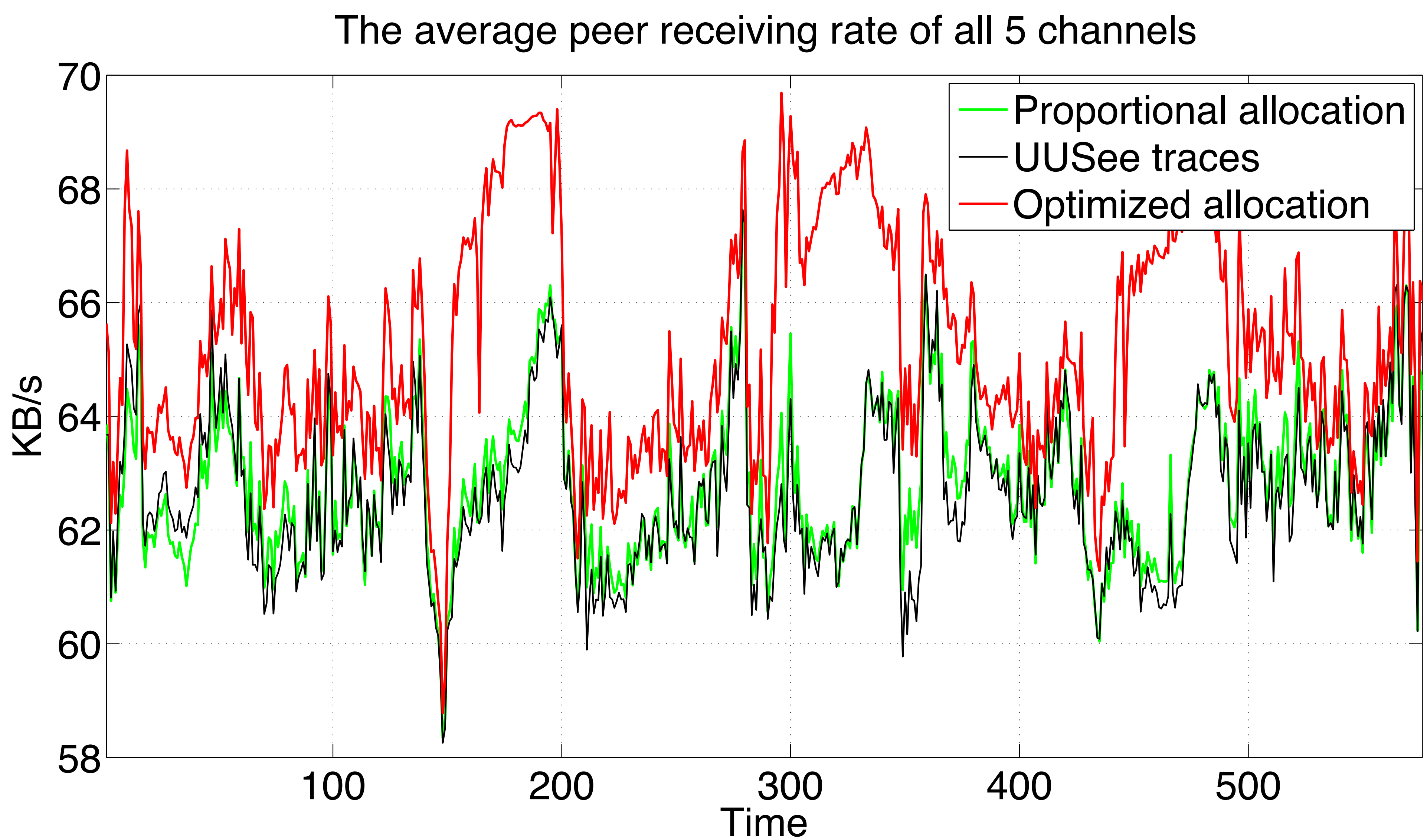
**Allocate server bandwidth *U* across *C* channels according to the *server send-peer receive* relation in each channel**

**Goal: to maximize the total receiving rate of all the peers**

# Optimized Server Allocation

**Allocate server bandwidth *U* across *C* channels according to the *server send-peer receive* relation in each channel**

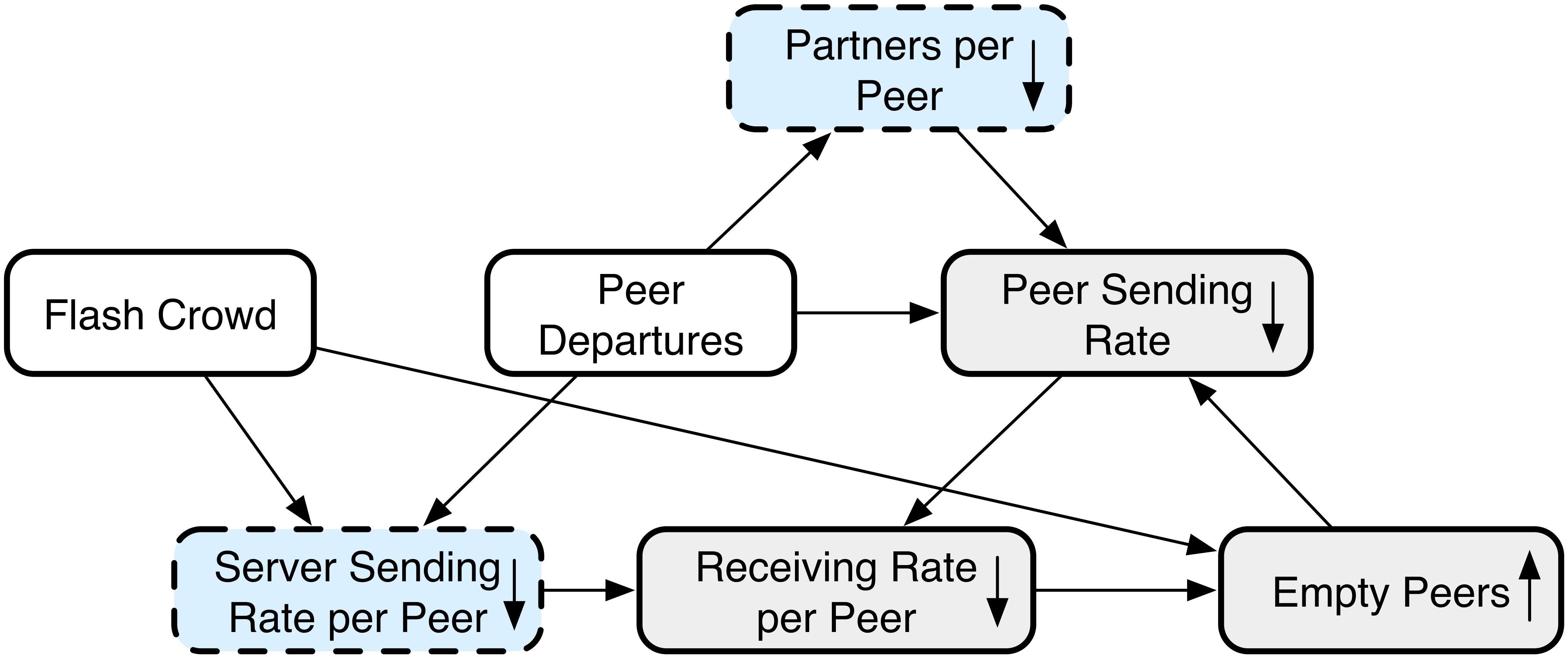
**Goal: to maximize the total receiving rate of all the peers**



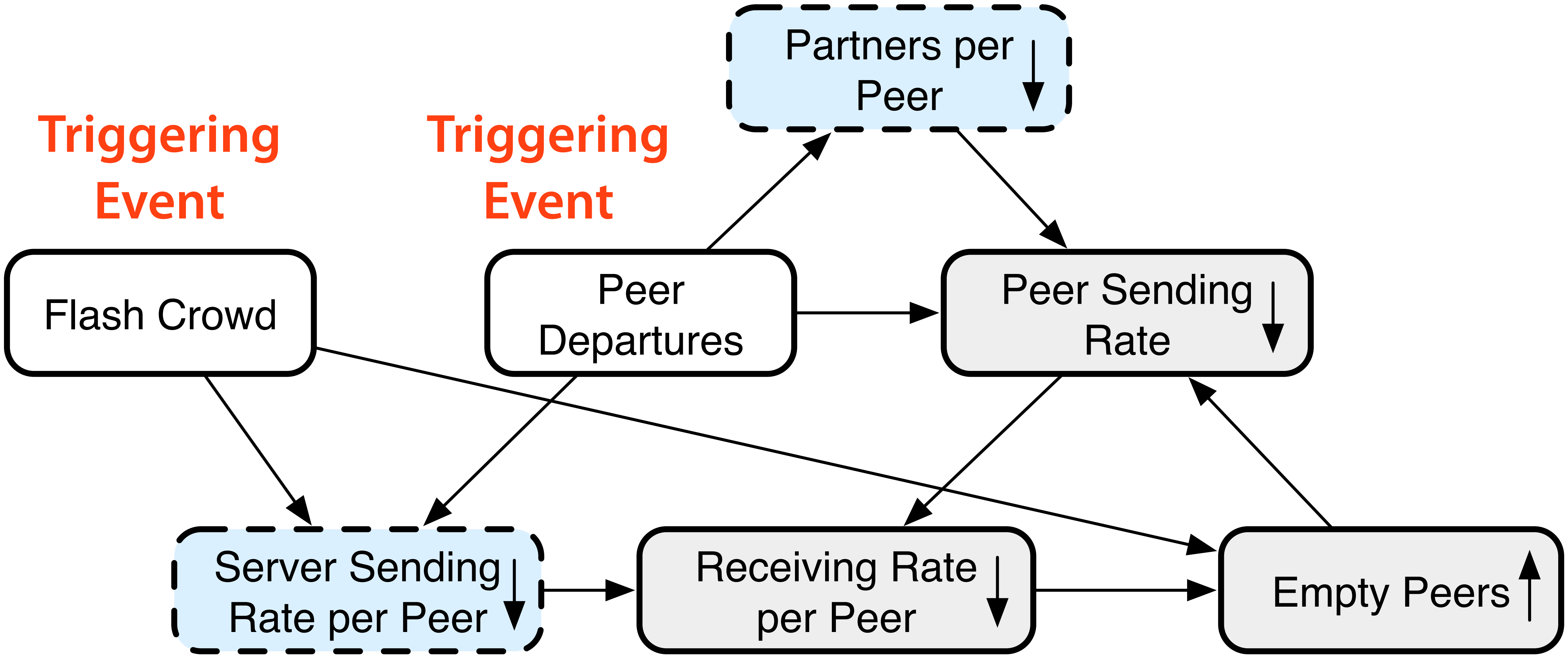

10% Improvement for only 5 channels

# Performance Anomalies Can be Predicted

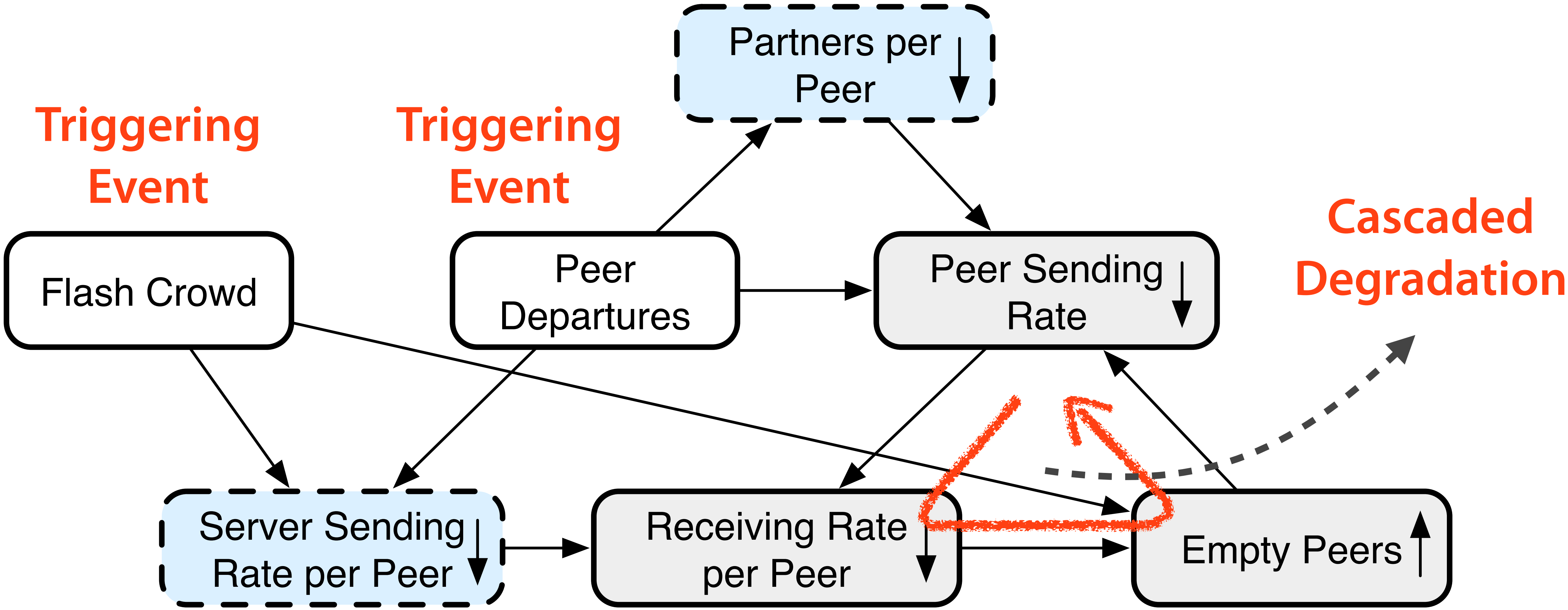
# Root Causes of Performance Anomaly



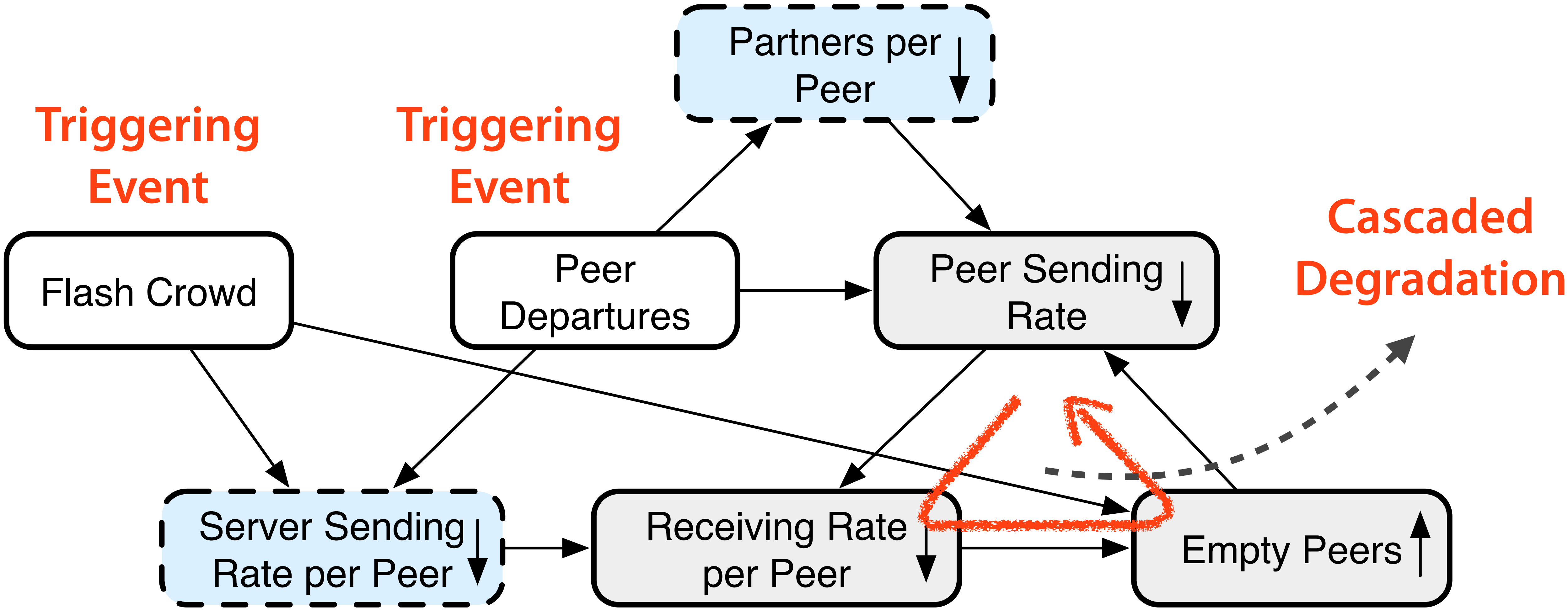
# Root Causes of Performance Anomaly



# Root Causes of Performance Anomaly

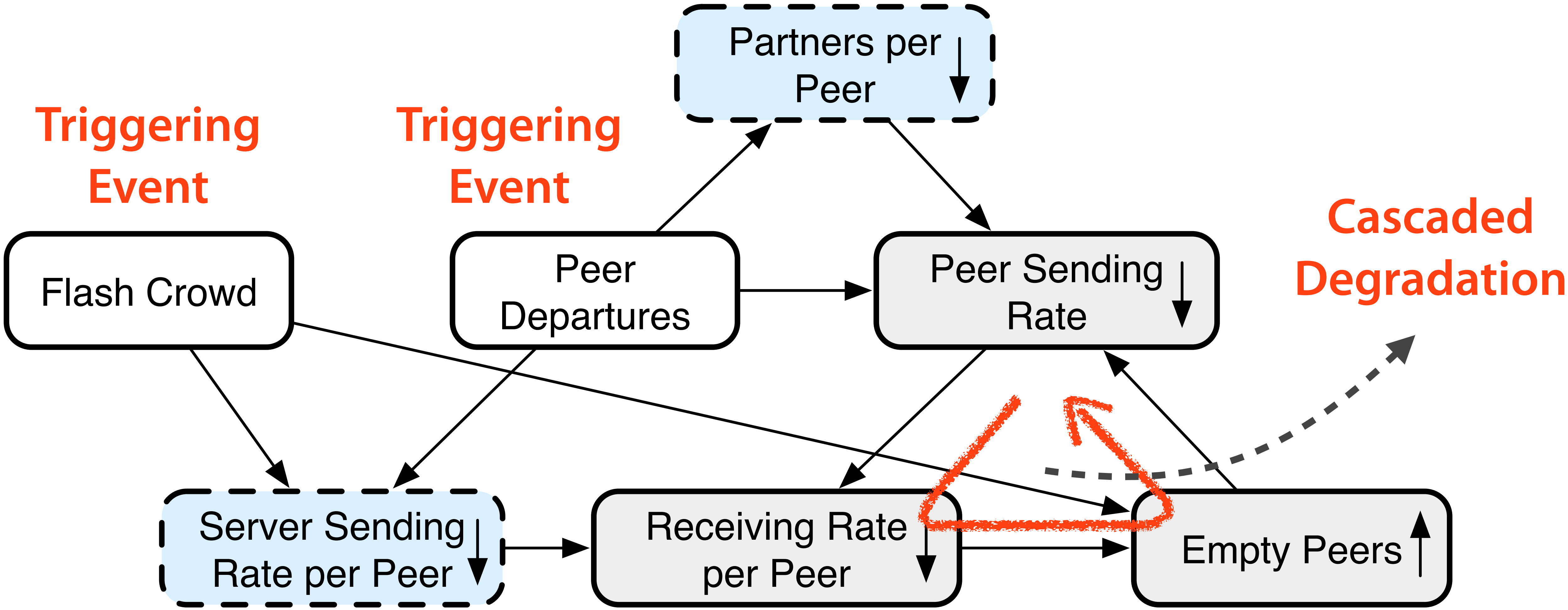


# Root Causes of Performance Anomaly




Non-linear relationships in nature!

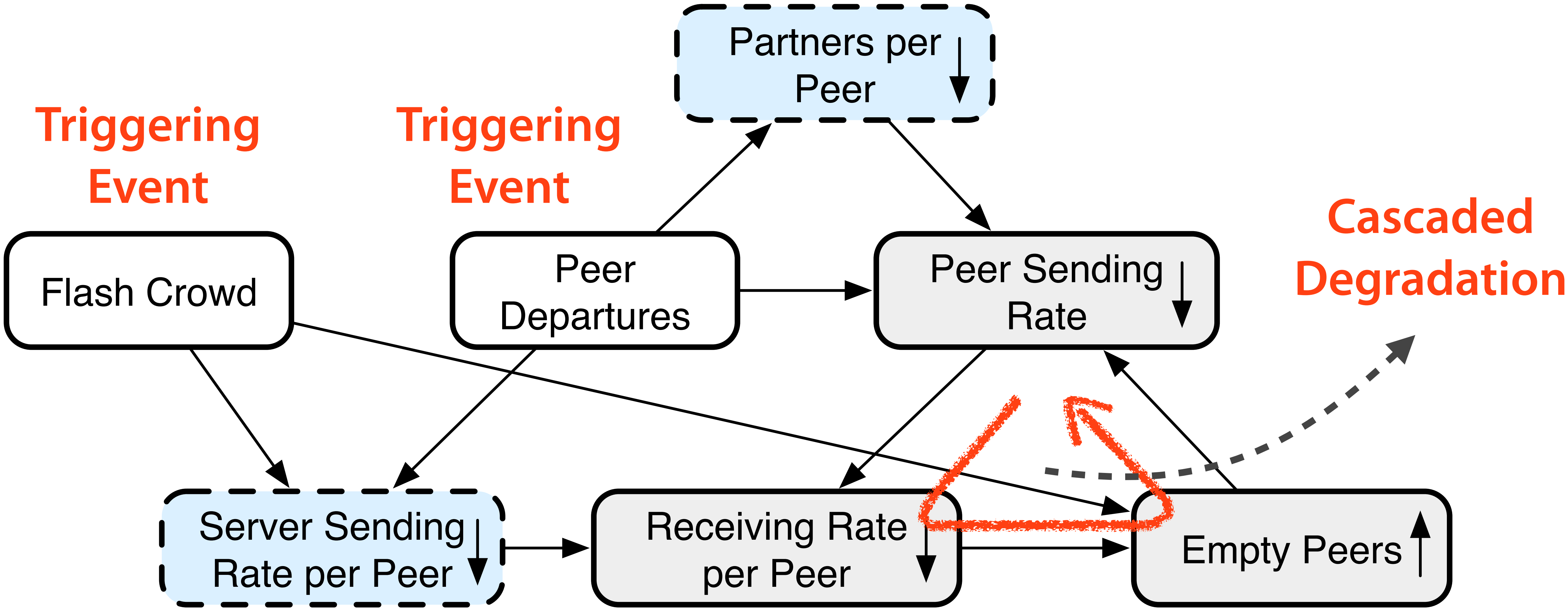
# Root Causes of Performance Anomaly




Non-linear relationships in nature!

Flash crowds, peer departures do not necessarily lead to degradation

# Root Causes of Performance Anomaly

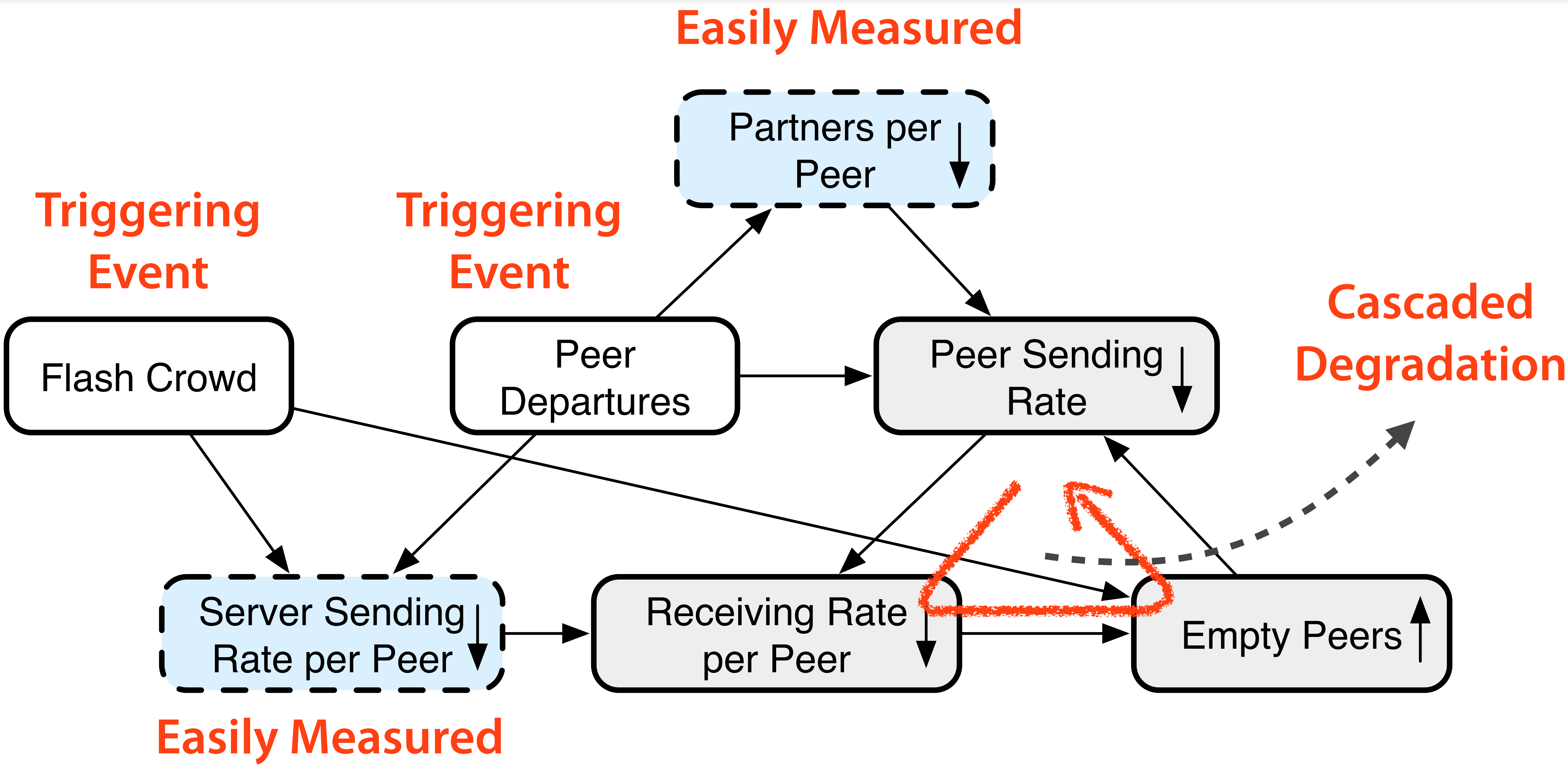



Non-linear relationships in nature!

Flash crowds, peer departures do not necessarily lead to degradation

Monitor *Server Send*, *#Partners* to forecast degradation

# Root Causes of Performance Anomaly



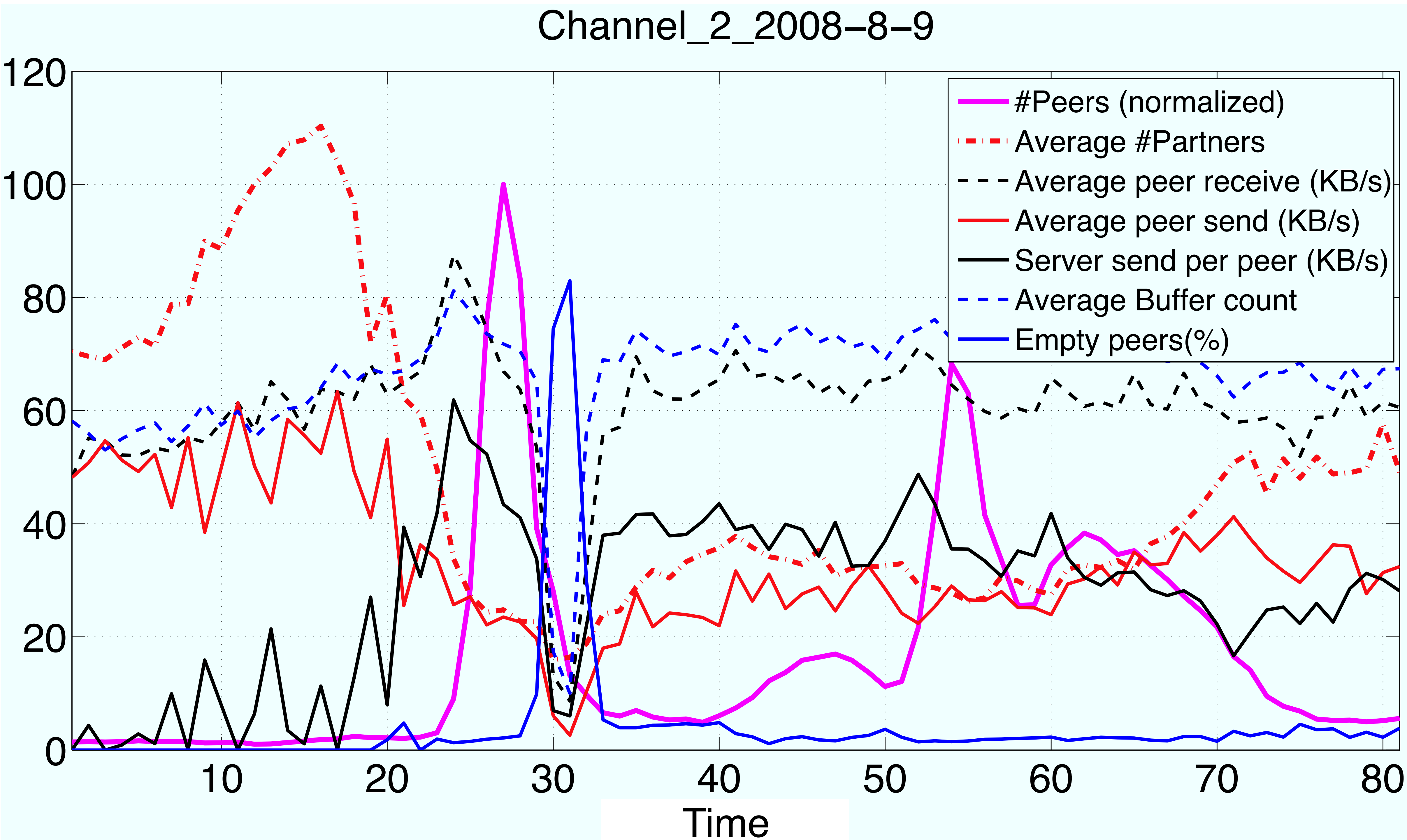

Non-linear relationships in nature!

Flash crowds, peer departures do not necessarily lead to degradation

Monitor *Server Send*, *#Partners* to forecast degradation

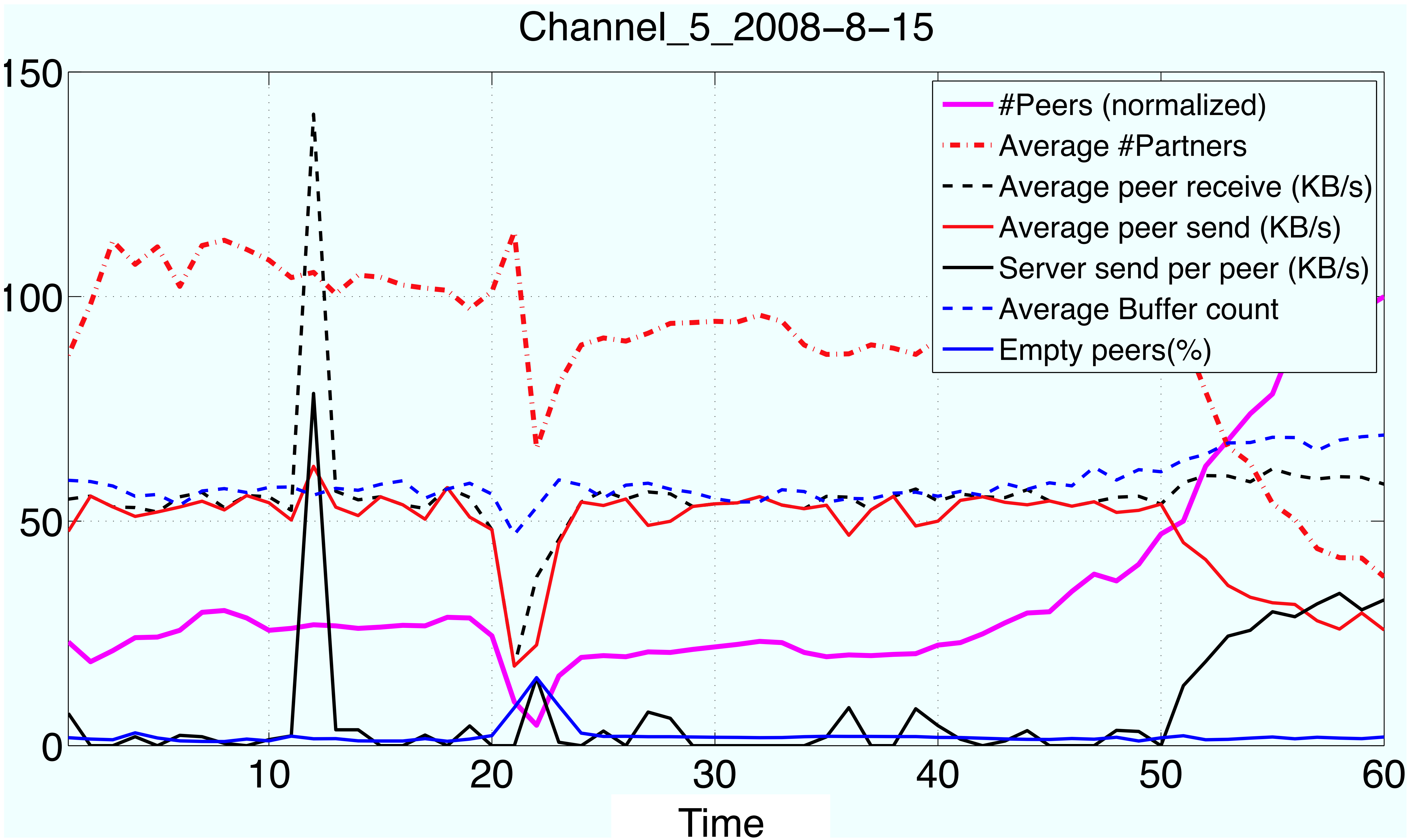
# Example 1

## The performance anomaly after a flash crowd.



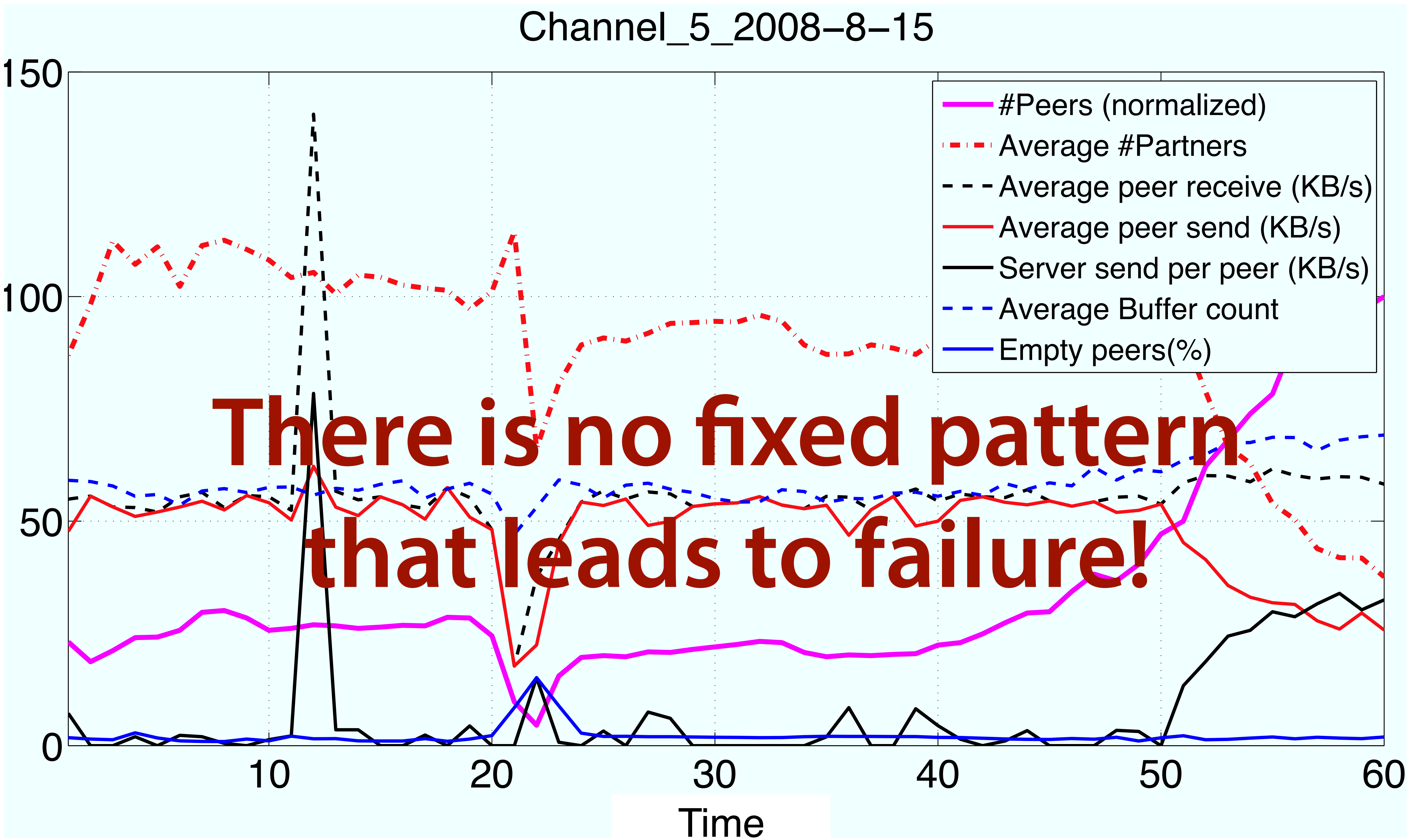
# Example 2

## The performance anomaly after peer departures



# Example 2

## The performance anomaly after peer departures




**There is no fixed pattern that leads to failure!**

# Prediction using Neural Networks

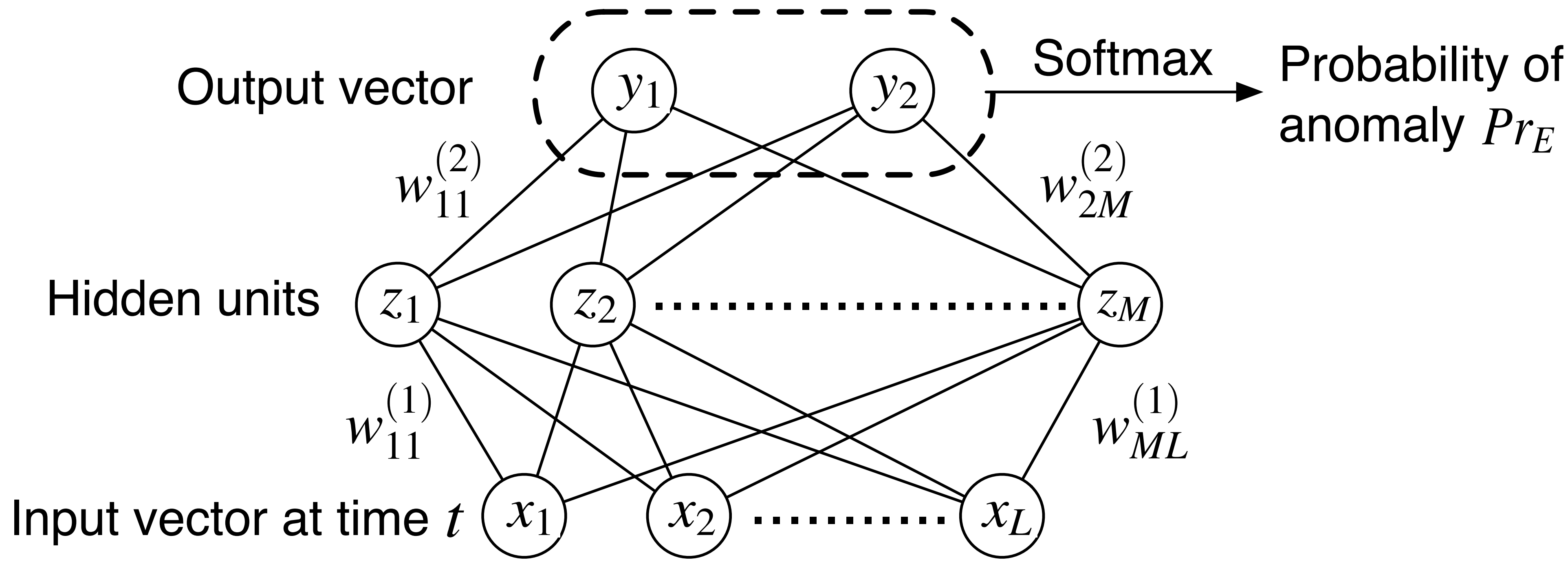
**Input:** *server send* per peer at previous $L_1$ timestamps

*number of partners* per peer at previous $L_2$ timestamps

$$\vec{\boldsymbol{x}}_t = (x_1, \ldots, x_L) = (\bar{s}_{t-L_1+1}, \ldots, \bar{s}_t, \overline{P}_{t-L_2+1}, \ldots, \overline{P}_t)/100,$$

**Output:** probability of performance anomaly at time $t$

$$Pr_E(t) = \Pr(\vec{\boldsymbol{T}}_t = (0,1)|\vec{\boldsymbol{x}}_t) = \frac{e^{y_2}}{e^{y_1} + e^{y_2}}.$$



# Prediction Details

# Prediction Details

## Training

For each channel, training set is one day with performance issues, and test set is the rest of the 11 days

The weights of the neural network are trained 1000 iterations using *error-backpropagation*

Fast: takes 40 seconds on a MacBook with 2.26GHz Intel Core2 Duo processor

Can be easily retrained as recent data arrive

# Prediction Details

## Training

For each channel, training set is one day with performance issues, and test set is the rest of the 11 days

The weights of the neural network are trained 1000 iterations using *error-backpropagation*

Fast: takes 40 seconds on a MacBook with 2.26GHz Intel Core2 Duo processor

Can be easily retrained as recent data arrive

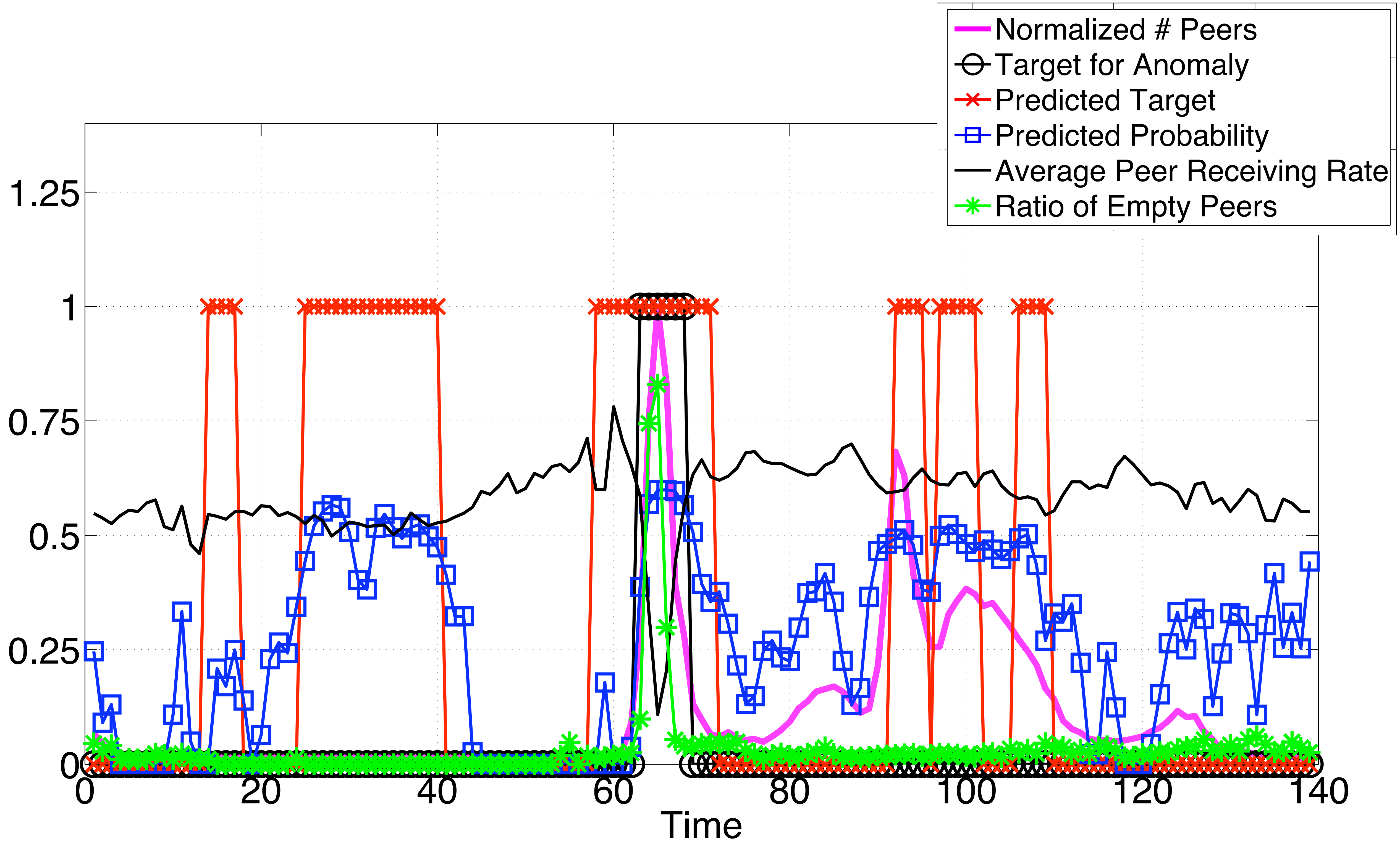
## Remove False Positives

Anomaly happens only around severe peer dynamics

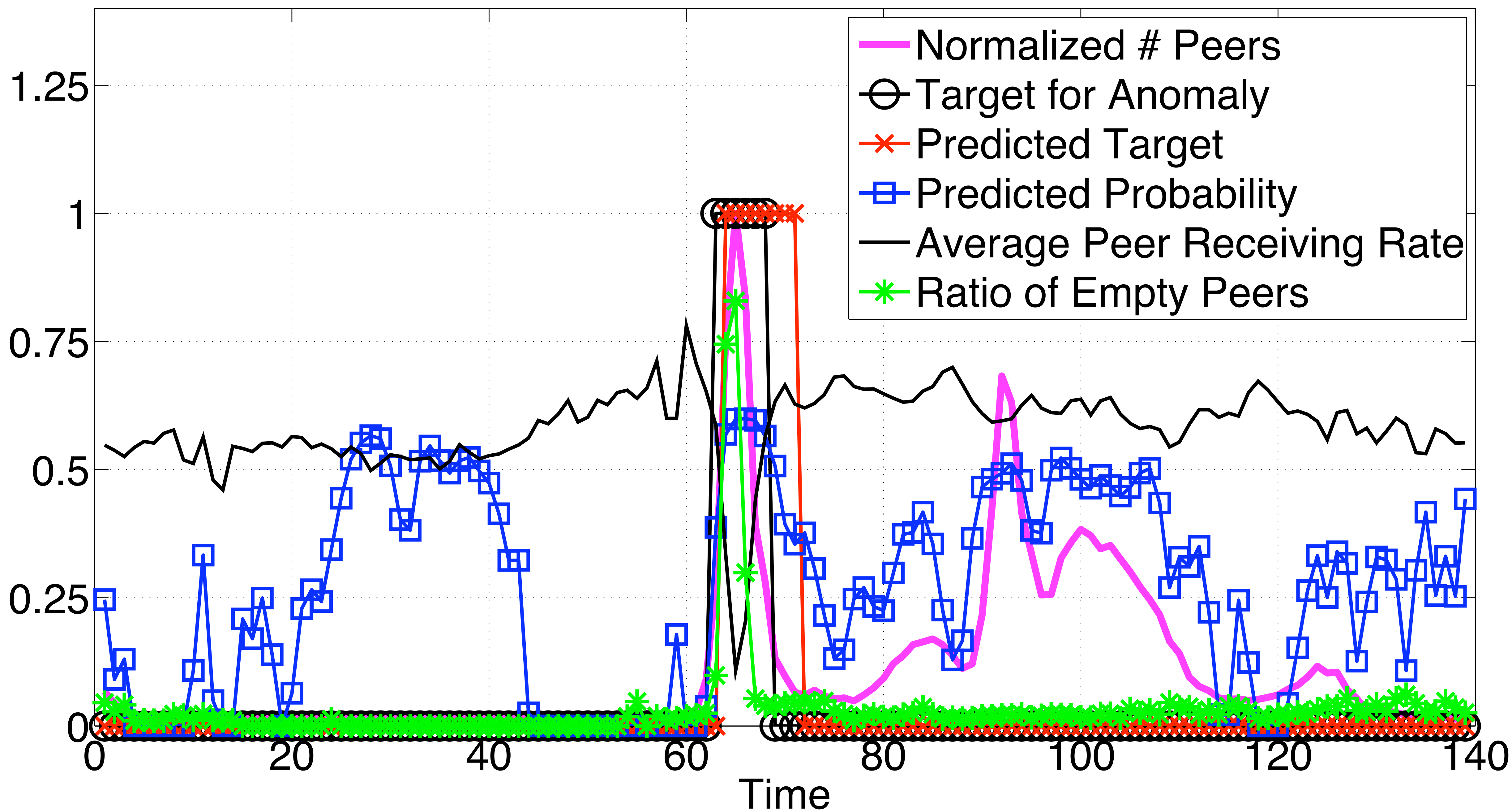
Use the change of online peer number to remove false positives

See Equation (8)

# Performance (Neural Network)



# Performance (Neural Network + FP Removal)



# Prediction Performance in 5 channels

**Table 4: Prediction Errors over 12 days in 5 channels.**

| | NN Prediction | | | | | NN with FP removal | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Channels | 1 | 2 | 3 | 4 | 5 | 1 | 2 | 3 | 4 | 5 |
| # Anomalies | 4 | 16 | 5 | 2 | 7 | 4 | 16 | 5 | 2 | 7 |
| # FN | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 |
| # Alarms | 12 | 41 | 14 | 6 | 28 | 4 | 16 | 4 | 2 | 10 |
| # FP | 8 | 28 | 10 | 4 | 20 | 0 | 3 | 0 | 0 | 3 |

False Negatives are low in both schemes

False Positives are removed by checking the change in peer number

# Prediction Performance in 5 channels

**Table 4: Prediction Errors over 12 days in 5 channels.**

| | NN Prediction | | | | | NN with FP removal | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Channels | 1 | 2 | 3 | 4 | 5 | 1 | 2 | 3 | 4 | 5 |
| # Anomalies | 4 | 16 | 5 | 2 | 7 | 4 | 16 | 5 | 2 | 7 |
| # FN | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 |
| # Alarms | 12 | 41 | 14 | 6 | 28 | 4 | 16 | 4 | 2 | 10 |
| # FP | 8 | 28 | 10 | 4 | 20 | 0 | 3 | 0 | 0 | 3 |

Server Send,
#Partners

False Negatives are low in both schemes

False Positives are removed by checking the change in peer number

# Prediction Performance in 5 channels

**Table 4: Prediction Errors over 12 days in 5 channels.**

| | NN Prediction | | | | | NN with FP removal | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Channels | 1 | 2 | 3 | 4 | 5 | 1 | 2 | 3 | 4 | 5 |
| # Anomalies | 4 | 16 | 5 | 2 | 7 | 4 | 16 | 5 | 2 | 7 |
| # FN | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 |
| # Alarms | 12 | 41 | 14 | 6 | 28 | 4 | 16 | 4 | 2 | 10 |
| # FP | 8 | 28 | 10 | 4 | 20 | 0 | 3 | 0 | 0 | 3 |

Server Send, #Partners

Server Send, #Partners, Change in # Peers

False Negatives are low in both schemes

False Positives are removed by checking the change in peer number

# Conclusions

# Conclusions

## Identified problems of real-world systems

- Inefficient resource usage
- Not robust to flash crowds or peer departures

# Conclusions

## Identified problems of real-world systems

- Inefficient resource usage
- Not robust to flash crowds or peer departures

## Propose a novel angle

- **Treat the complex system of peer-assisted streaming almost as a black box, monitor it, and learn the rules!**

# Conclusions

## Identified problems of real-world systems

- Inefficient resource usage
- Not robust to flash crowds or peer departures

## Propose a novel angle

- **Treat the complex system of peer-assisted streaming almost as a black box, monitor it, and learn the rules!**

## Large-scale media streaming systems needs to be

- **Adaptive to Demands, Intelligent and Self-Diagnostic**

# Conclusions

## Identified problems of real-world systems

Inefficient resource usage

Not robust to flash crowds or peer departures

## Propose a novel angle

**Treat the complex system of peer-assisted streaming almost as a black box, monitor it, and learn the rules!**

## Large-scale media streaming systems needs to be

**Adaptive to Demands, Intelligent and Self-Diagnostic**

## Questions/Comments?