Mathematisch-
Naturwissenschaftliche Fakultät

Energy Aware Resource Management for Clusters of Web Servers

Simon Kiertscher
University of Potsdam
Germany

Before we start …

Outline

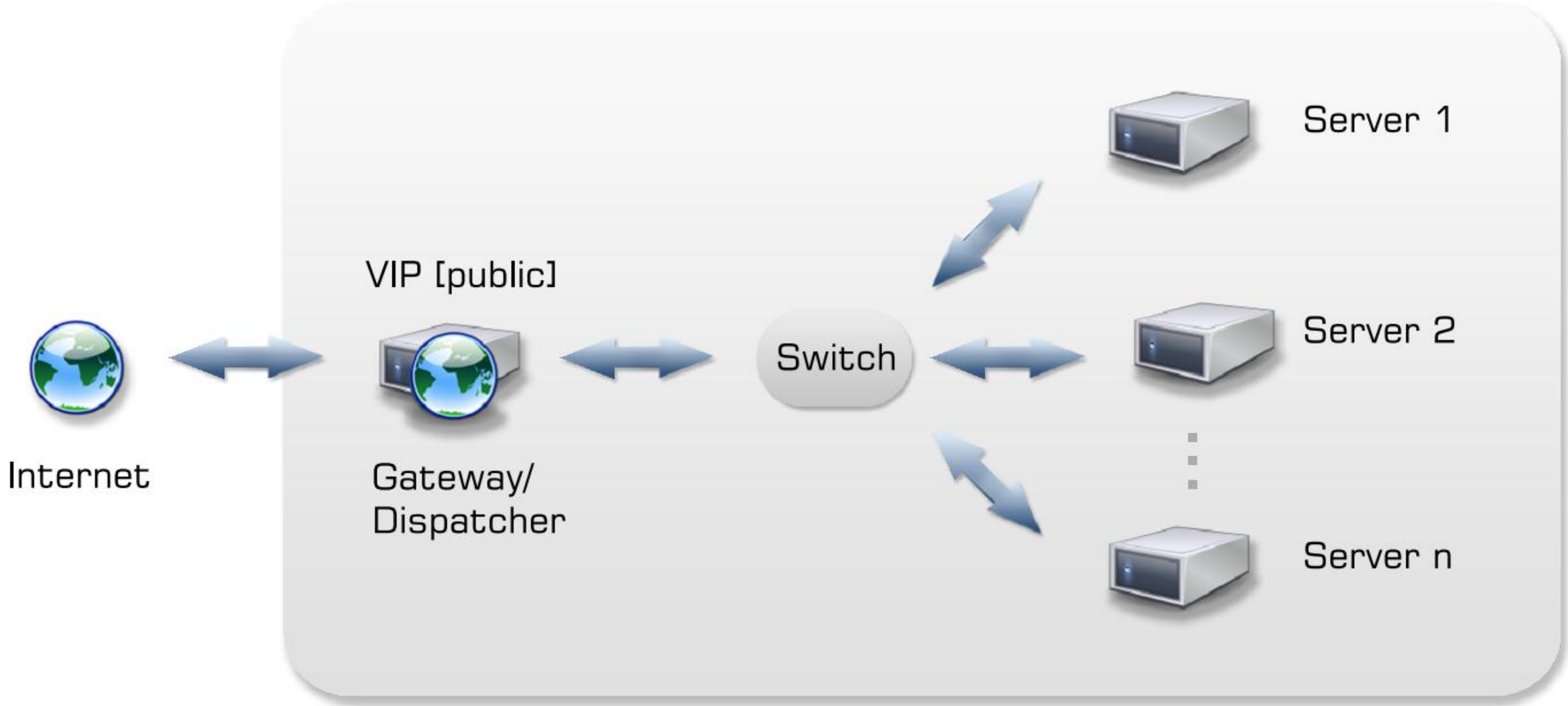
- Cluster Basics
- Motivation
- Energy Saving Daemon (Cherub)
- Load Forecasting
- Evaluation
- Conclusion & Future Work

Mathematisch-Naturwissenschaftliche Fakultät

Cluster Computing Basics

- High-Performance-Computing (HPC)
  - Few computationally intensive jobs which run for a long time (e.g. climate simulations, weather forecasting)
- Web Server / Server-Load-Balancing (SLB)
  - Thousands of small requests
  - Facebook as example:
    - 18.000 new comments per second
    - > 500 million user upload 100 million photos per day

Components of a SLB Cluster



Mathematisch-
Naturwissenschaftliche Fakultät

Outline

- Cluster Basics
- Motivation
- Energy Saving Daemon (Cherub)
- Load Forecasting
- Evaluation
- Conclusion & Future Work

Universität Potsdam
Mathematisch-Naturwissenschaftliche Fakultät

Motivation

- Energy has become a critical resource in cluster designs
- Usage of energy is still permanently rising
- Large scale web servers are mostly company owned
  → very few information available
- datacenterknowledge.com provides a small list of official numbers and estimations

Motivation - Web Server Numbers

Company	Number of Servers	Info
Microsoft	>1 million	according to CEO Steve Ballmer (July, 2013)
Facebook	"hundreds of thousands of servers"	Facebook's Najam Ahmad (June, 2013)
OVH	150,000	company (July, 2013)
Akamai Technologies	127,000	company (July 2013)
SoftLayer	100,000	company (December 2011)
Rackspace	94,122	company press release (March 31, 2013)
Intel	75,000	company (August, 2011)
1&1 Internet	"More than" 70,000	company (Feb. 2010)
eBay	54,011	DSE dashboard (July, 2013)

Source: http://www.datacenterknowledge.com/archives/2009/05/14/whos-got-the-most-web-servers/
Access Date: 2013/08/12

Motivation - Web Server Estimations

Company	Number of Servers	Info
Google	900,000	based on extrapolation on its total energy usage
Amazon	40,000 dedicated to running Amazon Web Services' EC2	estimation by Randy Bias & bought $86 million in servers
Yahoo	100,000	estimation
HP/EDS	380,000 in 180 data centers	company documents

Source: http://www.datacenterknowledge.com/archives/2009/05/14/whos-got-the-most-web-servers/
Access Date: 2013/08/12

Universität Potsdam Mathematisch-Naturwissenschaftliche Fakultät

Motivation - What to do?

- How can we save energy?
- Two main methods:
  1. Switch off unused resources
  2. Virtualization

- Plus some other methods
  - Replace old hardware
  - Effective cooling
  - Build your cluster in arctic regions
  - ...

Motivation - What to do?

- How can we save energy?
- Two main methods:
  1. ***Switch off unused resources***
  2. Virtualization

- Plus some other methods
  - Replace old hardware
  - Effective cooling
  - Build your cluster in arctic regions
  - ...

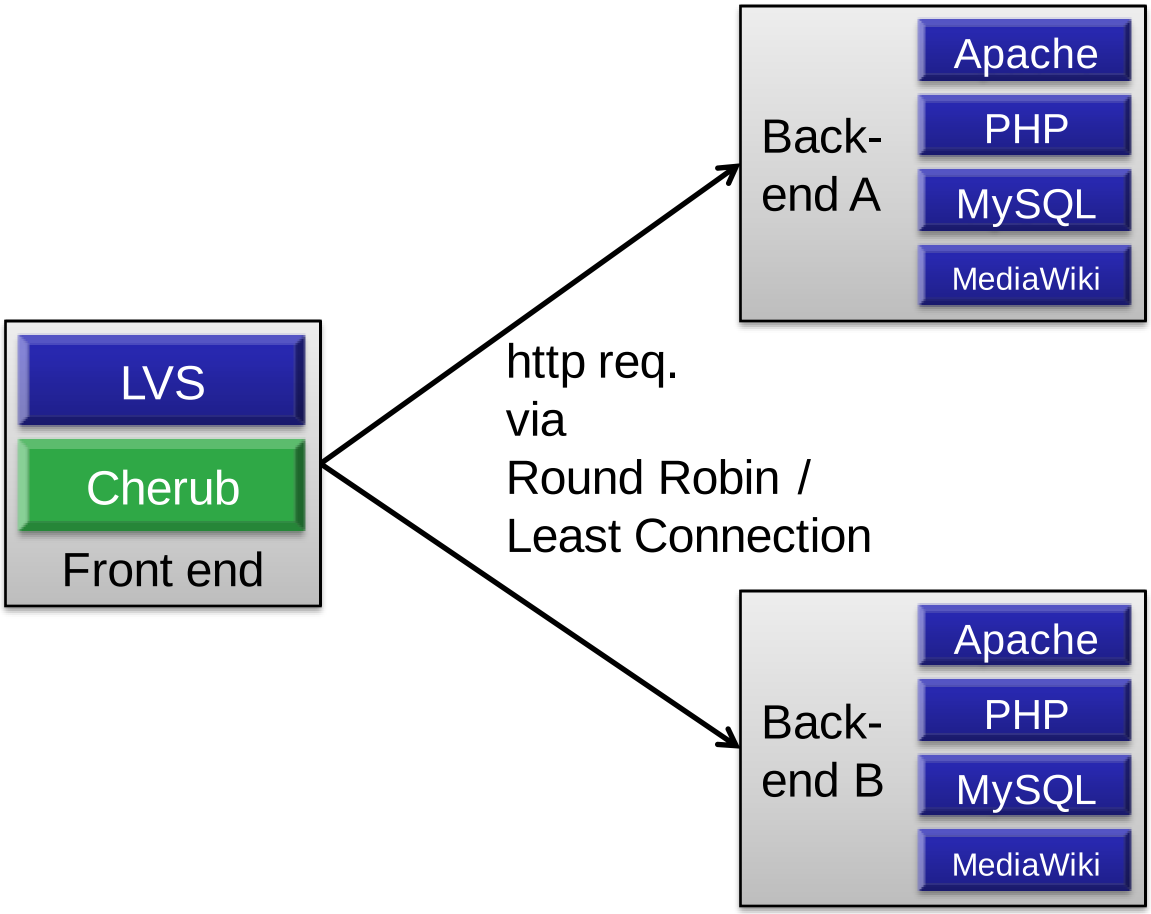
Outline

- Cluster Basics
- Motivation
- Energy Saving Daemon (Cherub)
- Load Forecasting
- Evaluation
- Conclusion & Future Work

Cherub

- Idea born in 2010
- Our institute has a small 28 node cluster
- Homogeneous environment
- interests on saving energy
- Straightforward → software which switches of unused resources and bring them back online if needed

Cherub


LVS
Cherub
Front end
http req.
via
Round Robin /
Least Connection
Back-
end A
Apache
PHP
MySQL
MediaWiki
Back-
end B
Apache
PHP
MySQL
MediaWiki

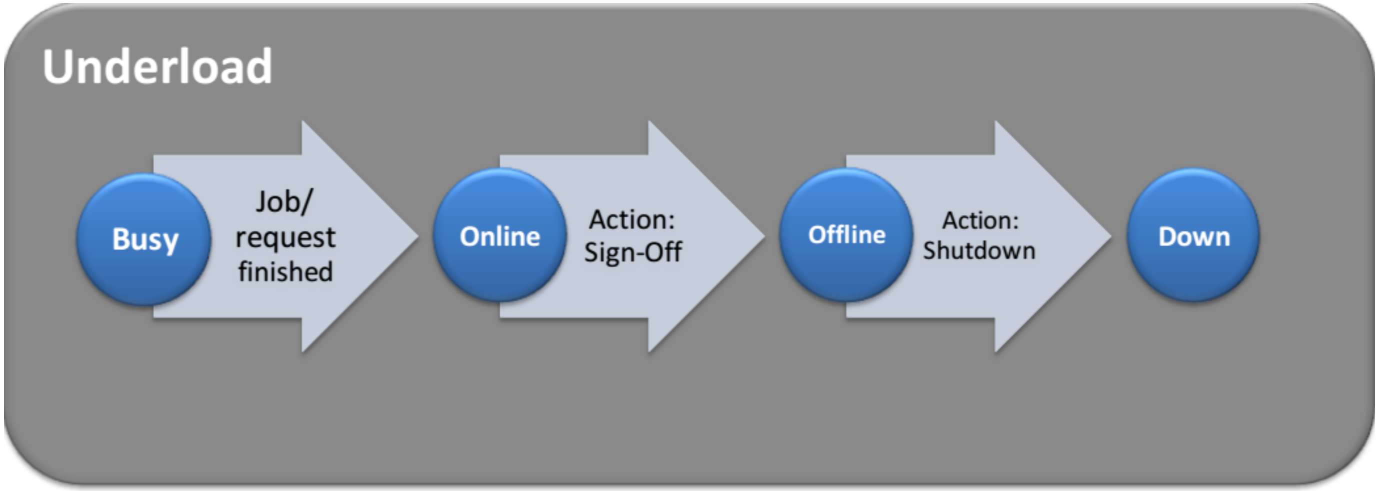
Cherub

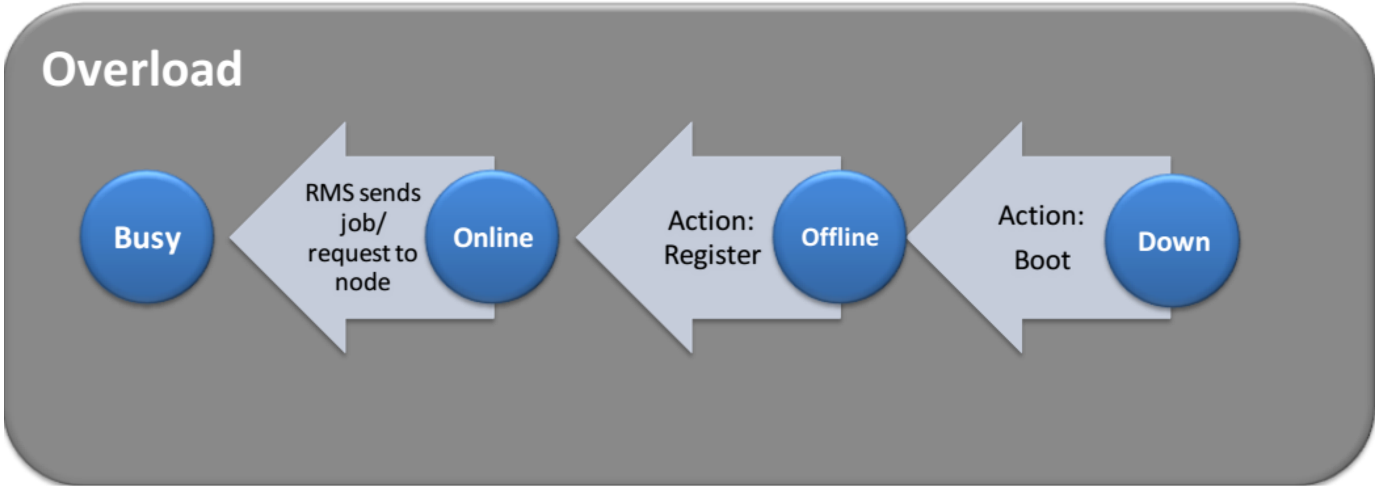
- Daemon on the master node polls the system in fixed time intervals to analyze its state
  - Status of every node
  - Load situation
- Depending on the state and saved attributes, actions are performed for every node
- Online System - we don't need any information about future load
- Decisions are all made at runtime

Cherub - Node States

- Five states needed for an internal representation of an arbitrary cluster
  1. UNKNOWN
  2. BUSY
  3. ONLINE
  4. OFFLINE
  5. DOWN

Cherub - State Transitions





Outline

- Cluster Basics
- Motivation
- Energy Saving Daemon (Cherub)
- Load Forecasting
- Evaluation
- Conclusion & Future Work

Load Forecasting

- Load: number of request / second
- Most systems [1,2,3,4] work with two thresholds
  1. Underload (e.g. 30% system saturation)
  2. Overload (e.g. 60% system saturation)
- Problems related to thresholds:
  1. Workload slightly above overload
  2. Strong increasing workload
- Machine learning can eliminate that problems

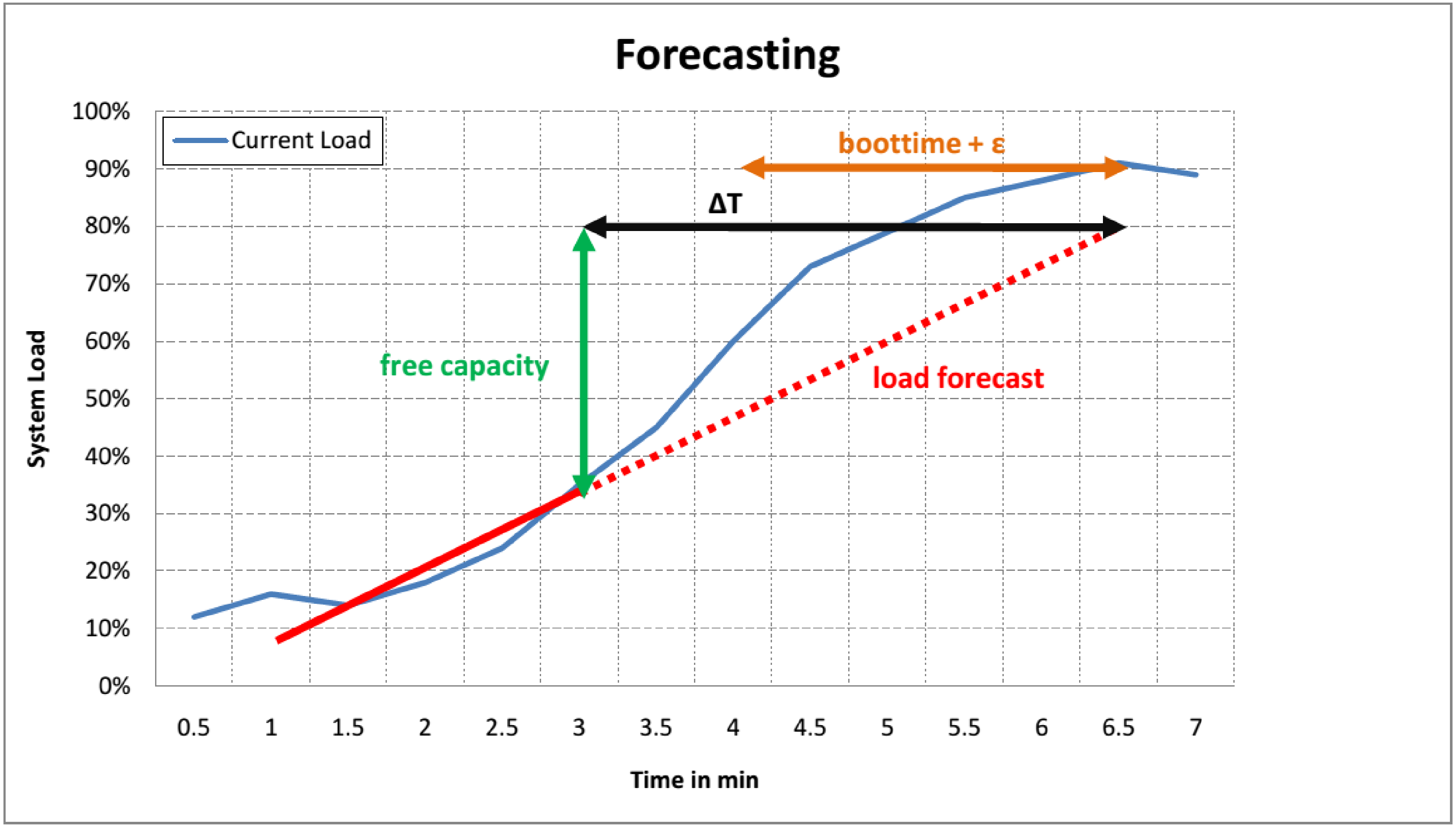
Load Forecasting

Our Propose:

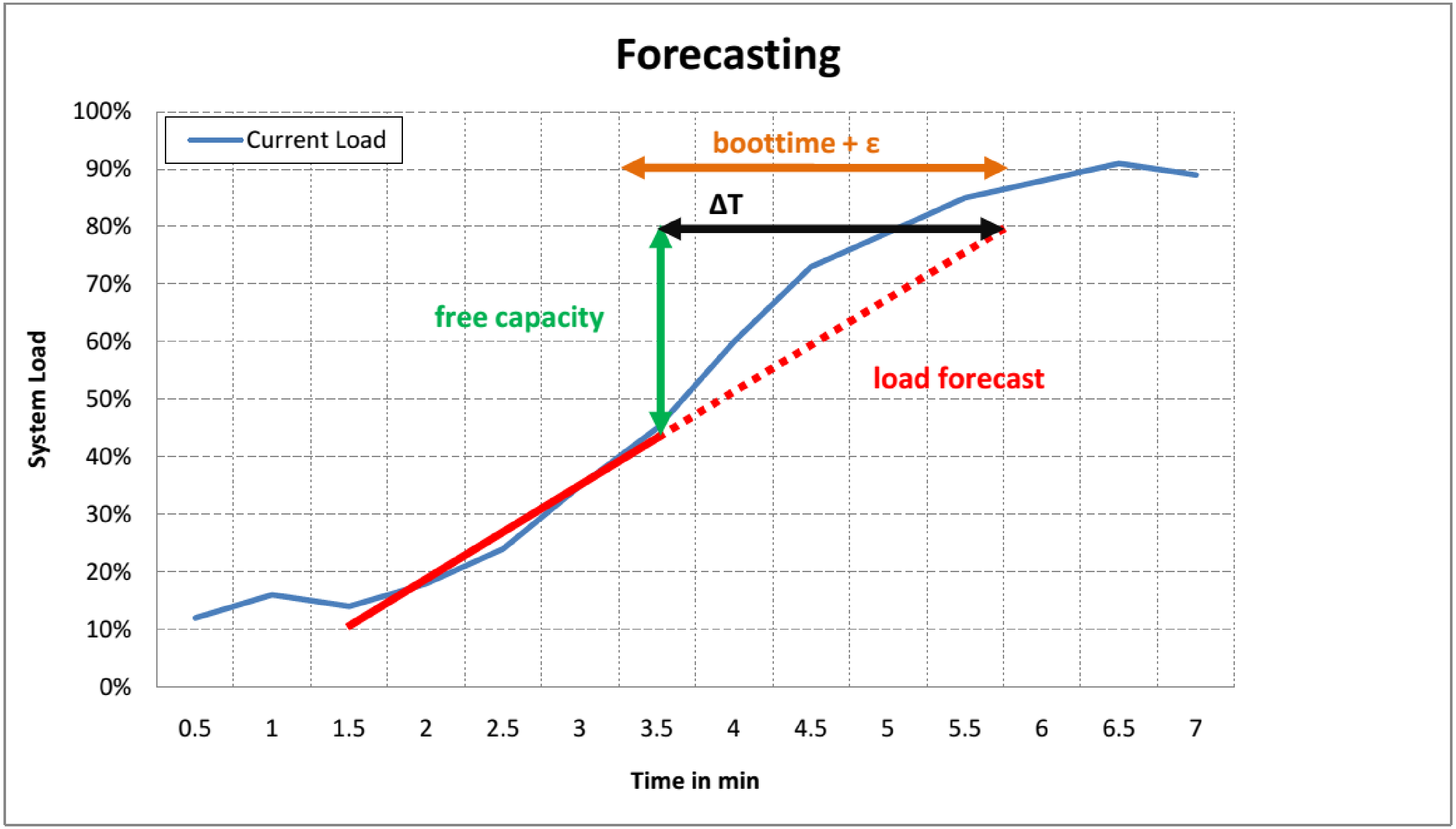
- Use Linear Regression to forecast future system load
  → Nodes can be booted in advance
  → Mitigates boot time related problems

- Decision for a boot command:
  (1) free capacity = overload - current load
  (2) ΔT = free capacity / slope
  (3) ΔT < boottime + ε → Boot new machine

Load Forecasting



Load Forecasting

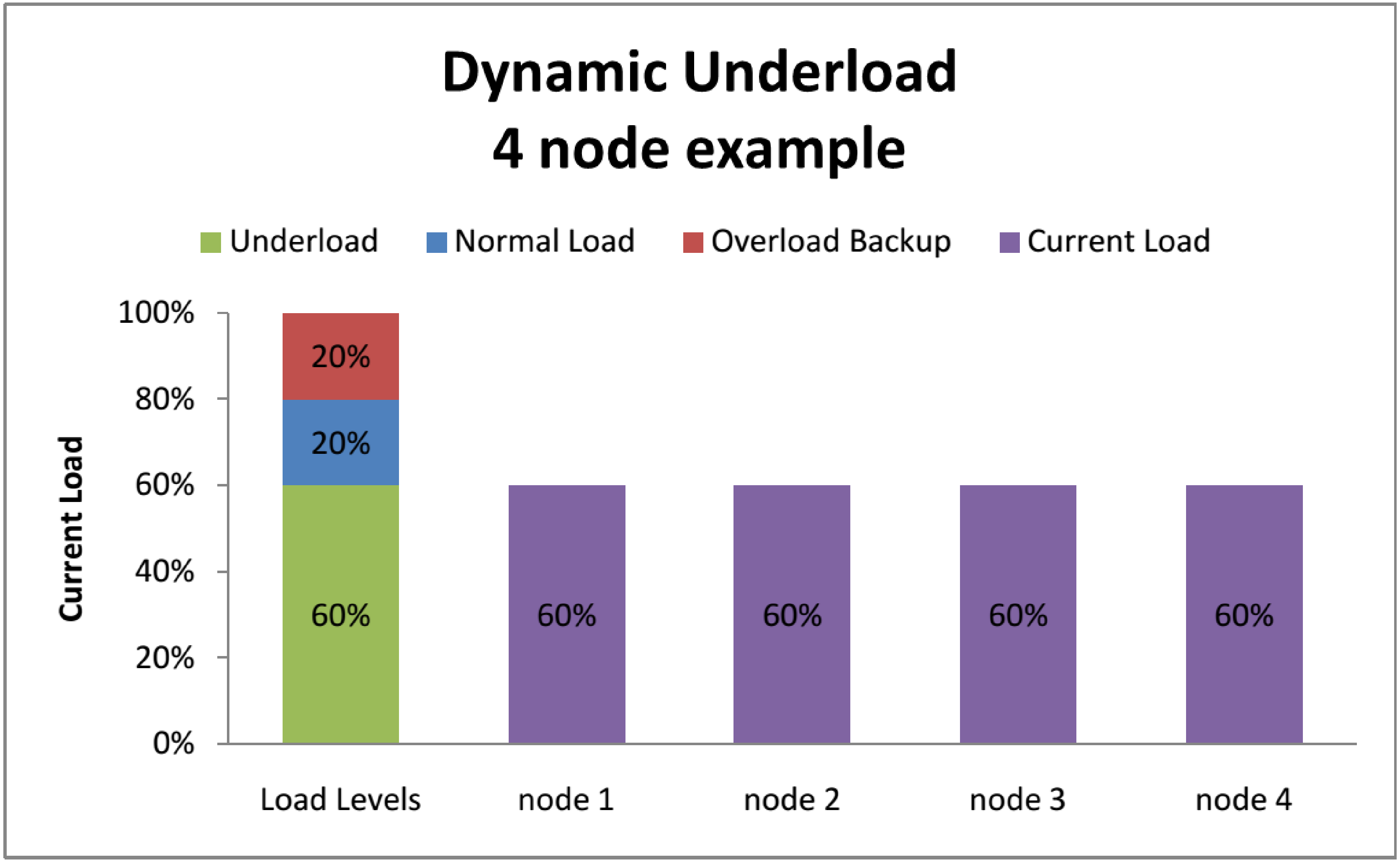


Load Forecasting

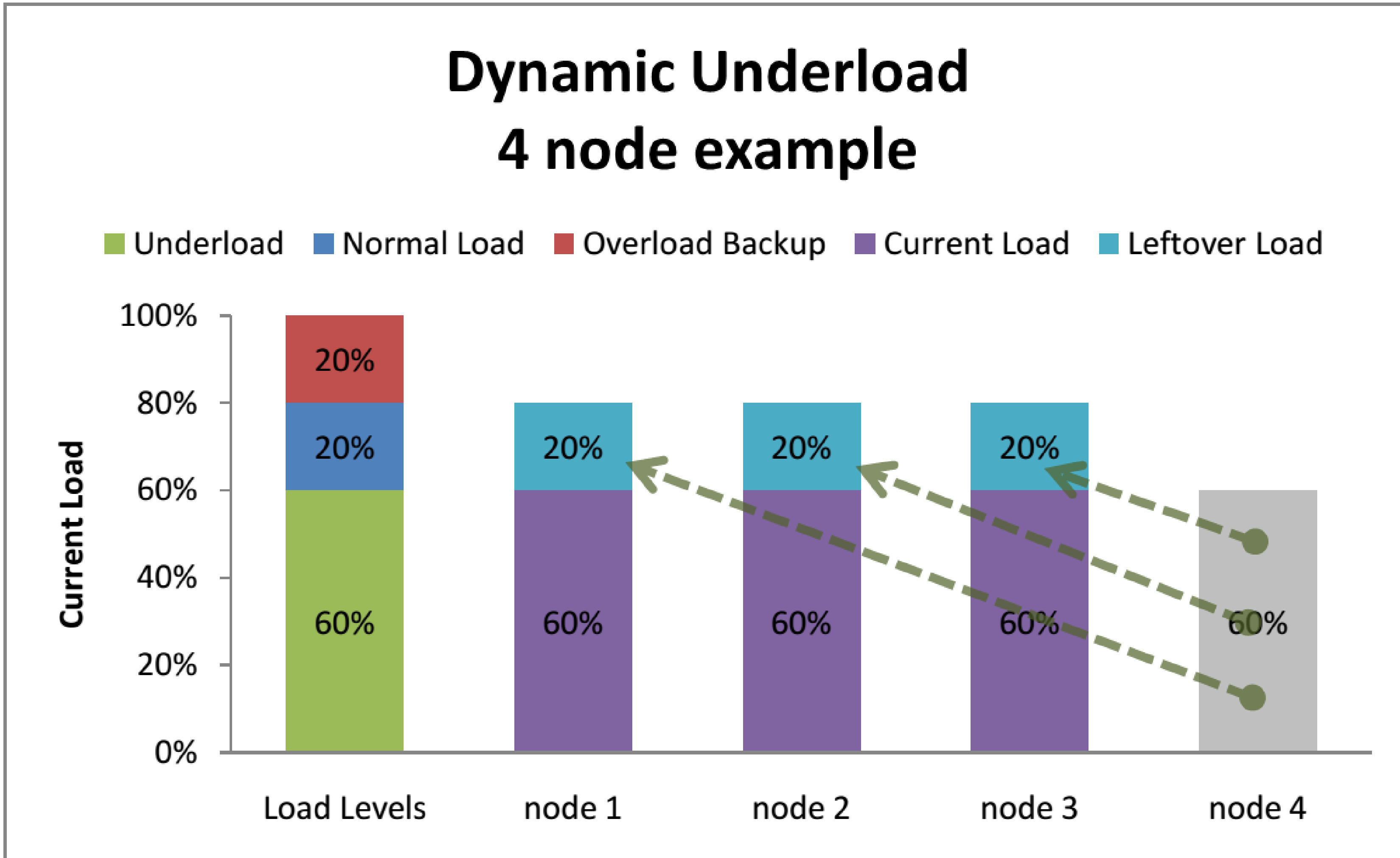
- Simplify thresholds, only one configurable overload threshold
- Derive a dynamic underload threshold

$$underload = overload - \frac{overload}{\#nodes}$$

Load Forecasting



Load Forecasting



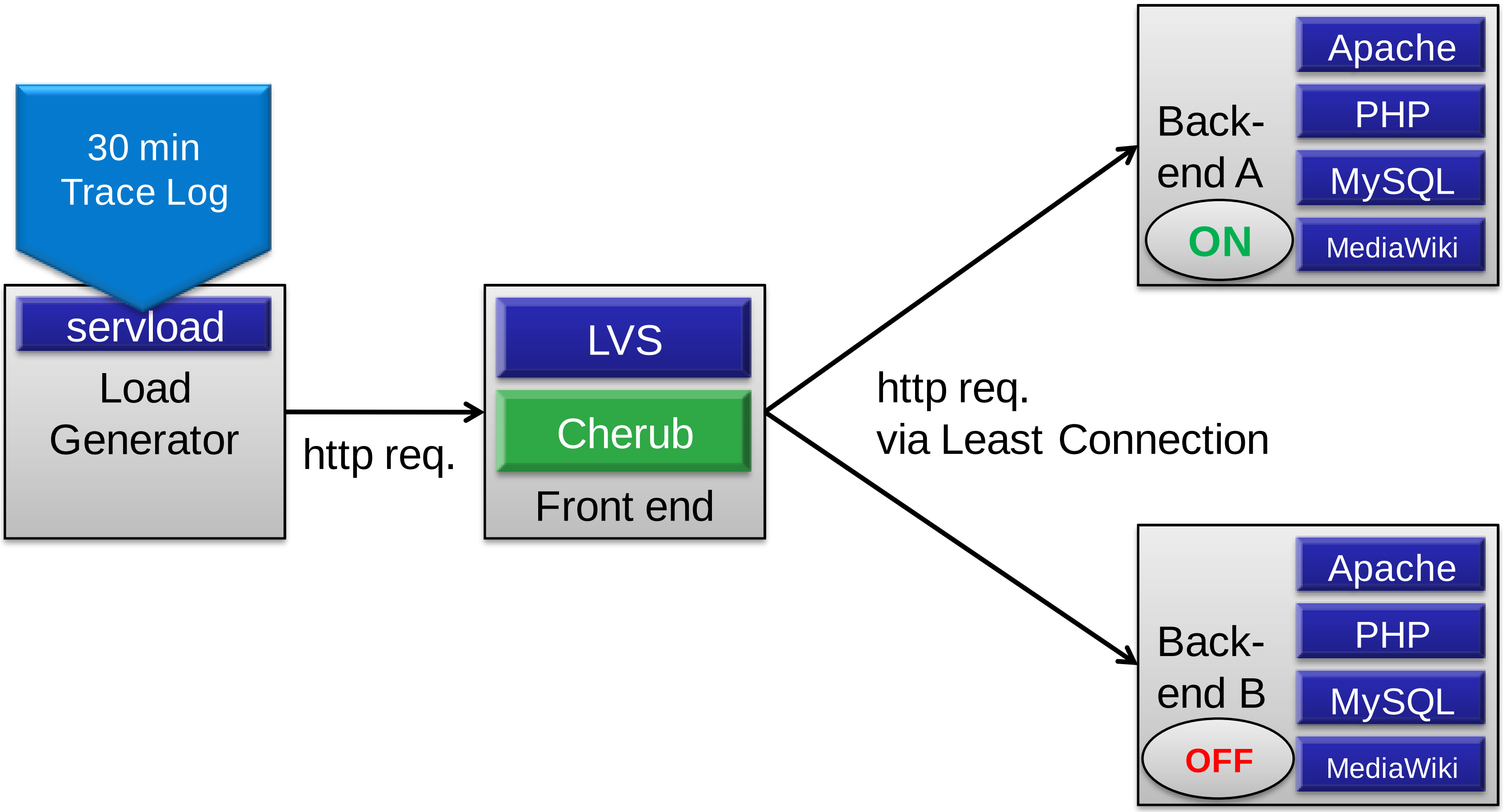
Outline

- Cluster Basics
- Motivation
- Energy Saving Daemon (Cherub)
- Load Forecasting
- Evaluation
- Conclusion & Future Work

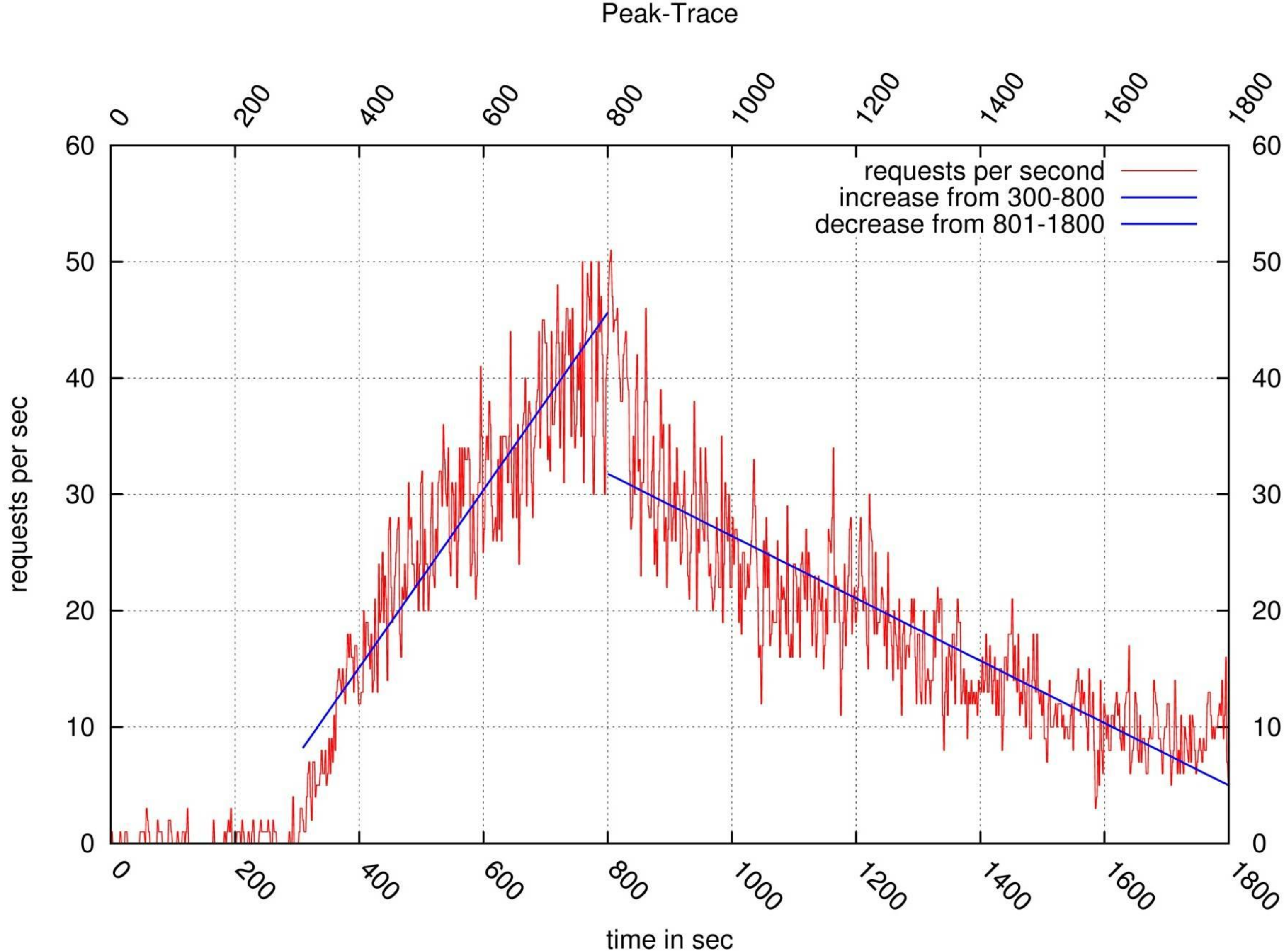
Evaluation Aims / Metrics / Methods

- Peak Trace is the most challenging situation
- Evaluation method: measurement
- Questions now:
  - Does load detection work fast enough?
  - How many lost requests?
  - How do different runtime solutions perform?
- Metrics:
  - Service Level Agreement (SLA) violations (request needs longer then 5 sec)
  - First Response Time (FRT)
  - Downtime

Setup

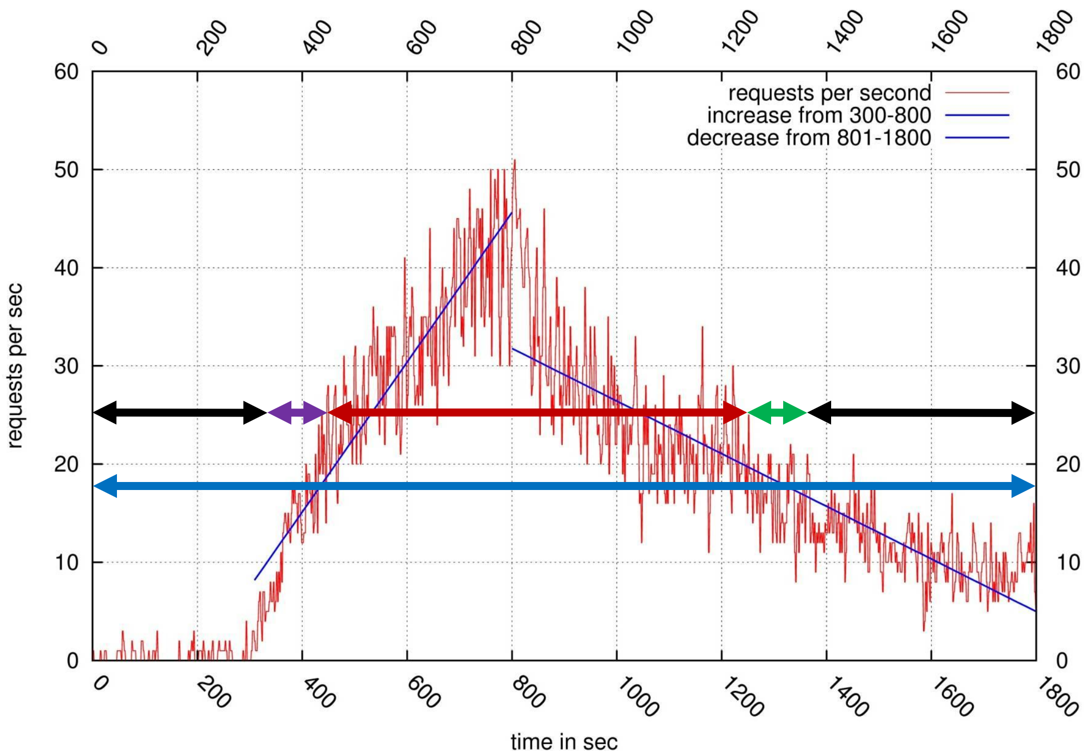

30 min
Trace Log
servload
Load
Generator
http req.
LVS
Cherub
Front end
http req.
via Least Connection
Back-
end A
ON
Apache
PHP
MySQL
MediaWiki
Back-
end B
OFF
Apache
PHP
MySQL
MediaWiki

The Trace - derived from Wikipedia



Additional Metrics

- Optimum Saving:
  Maximum downtime without losing requests
- For two nodes:
  $T_{maxdown} = T_{duration} - (T_{last} - T_{first}) - T_{boot} - T_{delay}$



Experiments performed

1. Reference measurement without Cherub
2. Basic thresholds only, no dynamic threshold, no forecasting
3. Dynamic thresholds, no forecasting
4. Linear Regression #1
5. Linear Regression #2 (mean load)

Reference Measurement

- Both machines ON
- No Cherub
- 3 runs, each 30 min

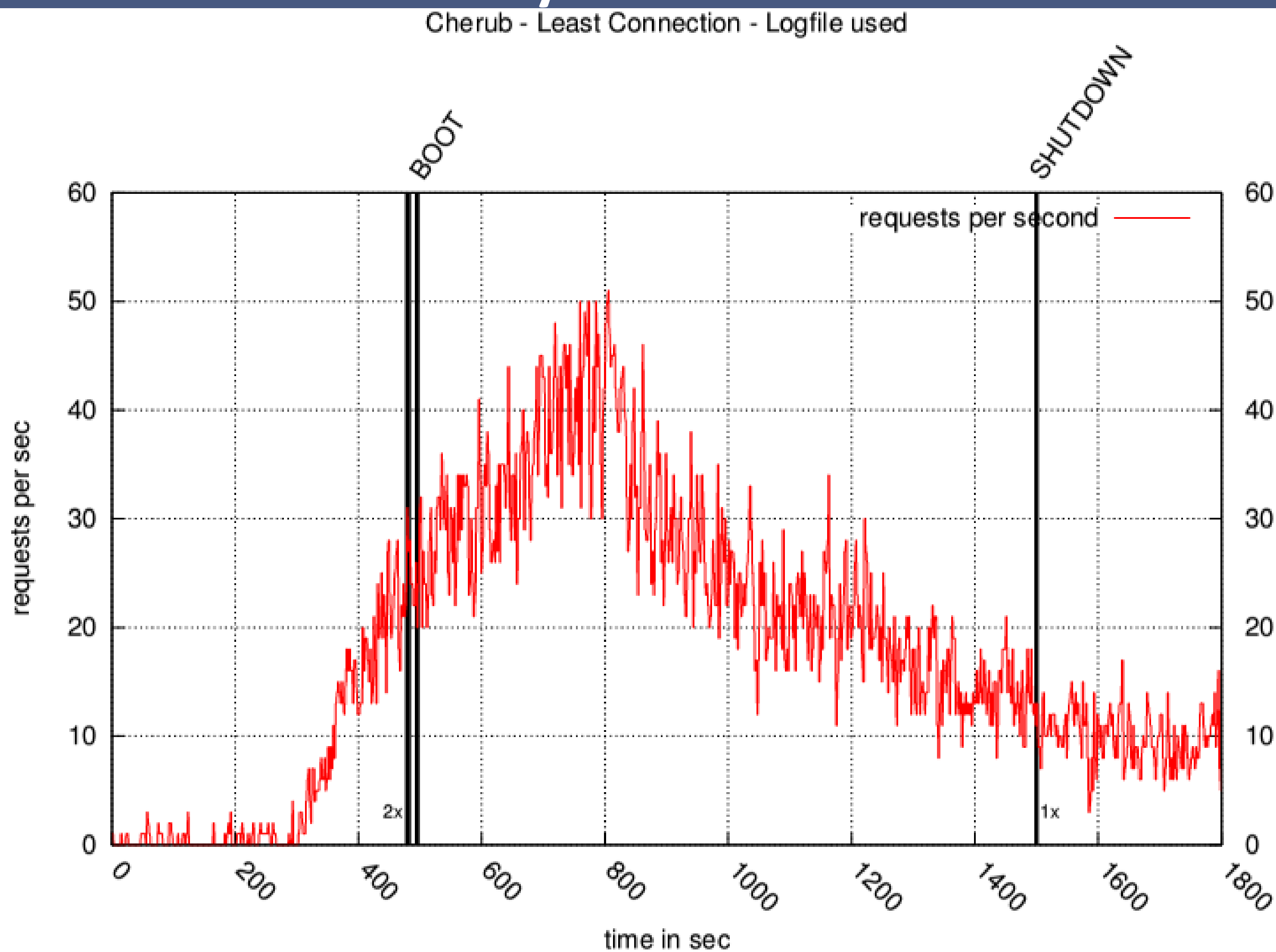
Metric	Avg.
SLA in %	99.63
First Response Time in msec	15.07
Downtime in min	0
Deviation from optimum in %	100

Basic thresholds only

- Overload by 60% saturation
- Underload by 20% saturation
- No dynamic threshold
- No forecast

Metric	Avg.	Ref. / Opt.
SLA in %	98.93	99.63
First Response Time in msec	23.60	15.07
Downtime in min	9.34	0 / 14
Deviation from optimum in %	33.29	100

Basic thresholds only



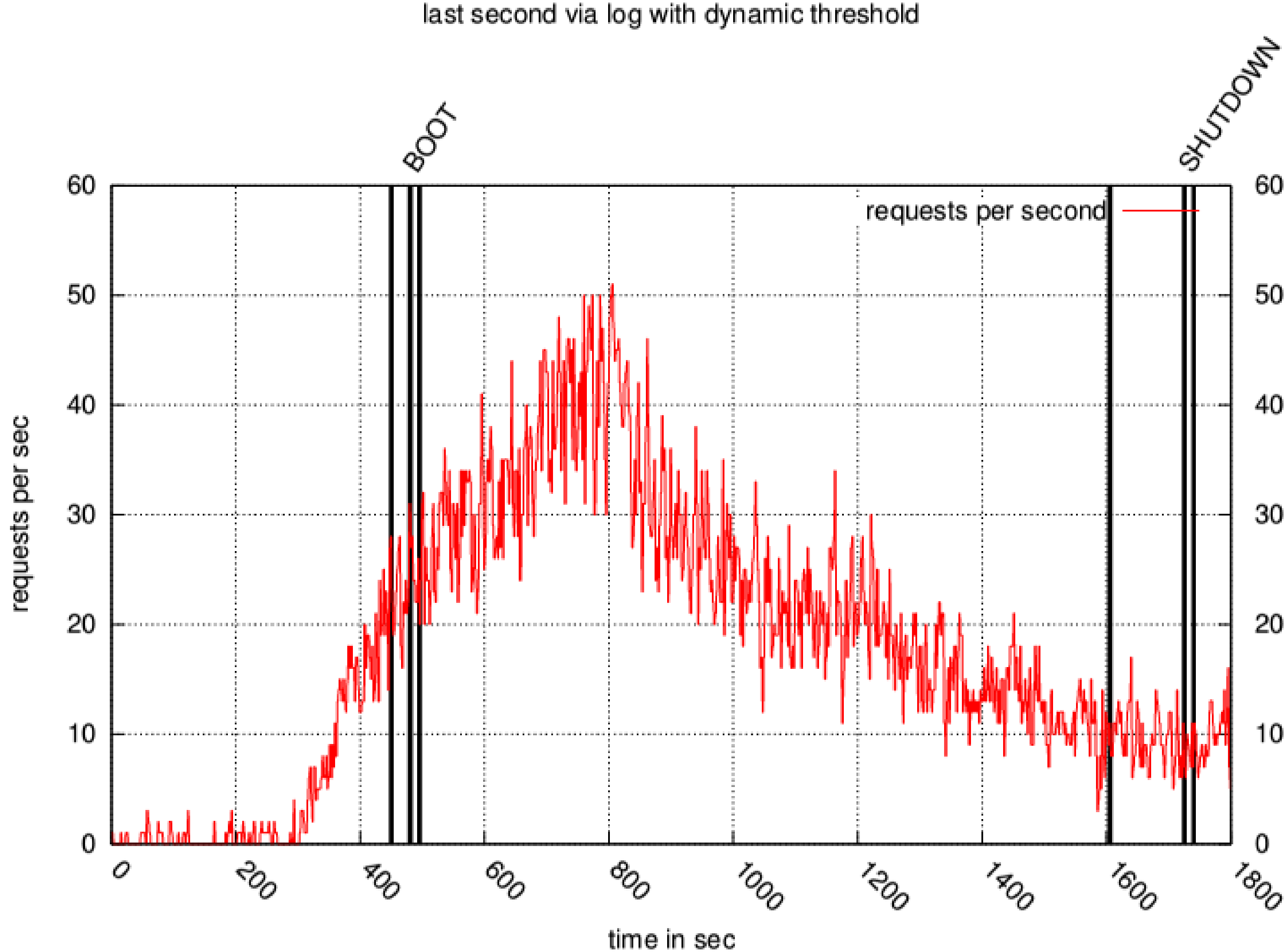
Dynamic thresholds

- Overload by 60% saturation
- Underload (dynamic) by 30% saturation
- No forecasting

Metric	Avg.	Ref. /Opt.
SLA in %	98.82	99.63
First Response Time in msec	34.29	15.07
Downtime in min	9.63	0 / 14
Deviation from optimum in %	31.21	100

Universität Potsdam Mathematisch-Naturwissenschaftliche Fakultät

Dynamic thresholds

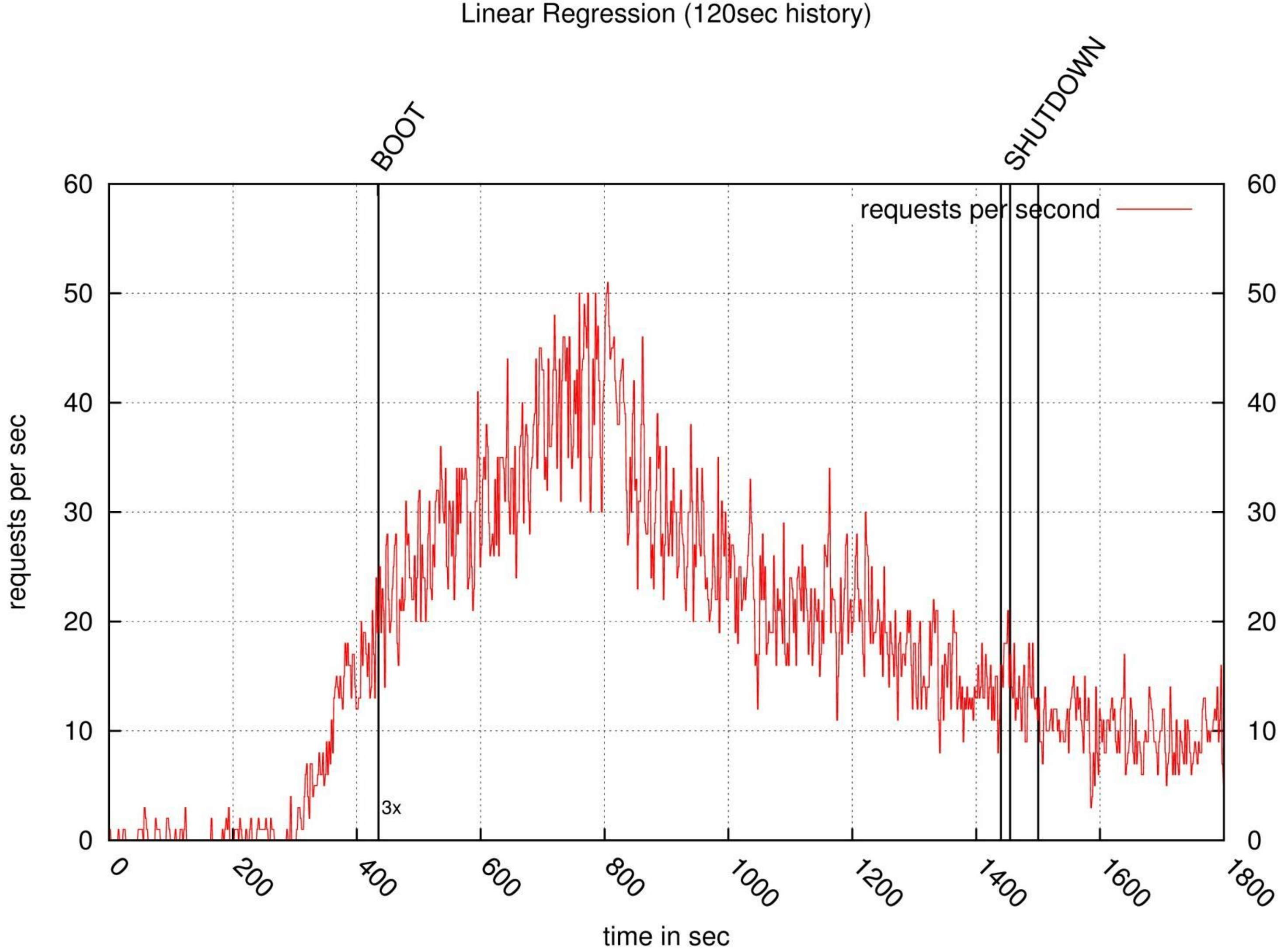


Linear Regression #1

- Overload by 80% saturation
- Underload (dynamic) by 40% saturation
- Load forecasting with linear regression
- 120 seconds history

Metric	Avg.	Ref. / Opt.
SLA in %	99.40	99.63
First Response Time in msec	32.99	15.07
Downtime in min	12.87	0 / 14
Deviation from optimum in %	8.07	100

Linear Regression #1

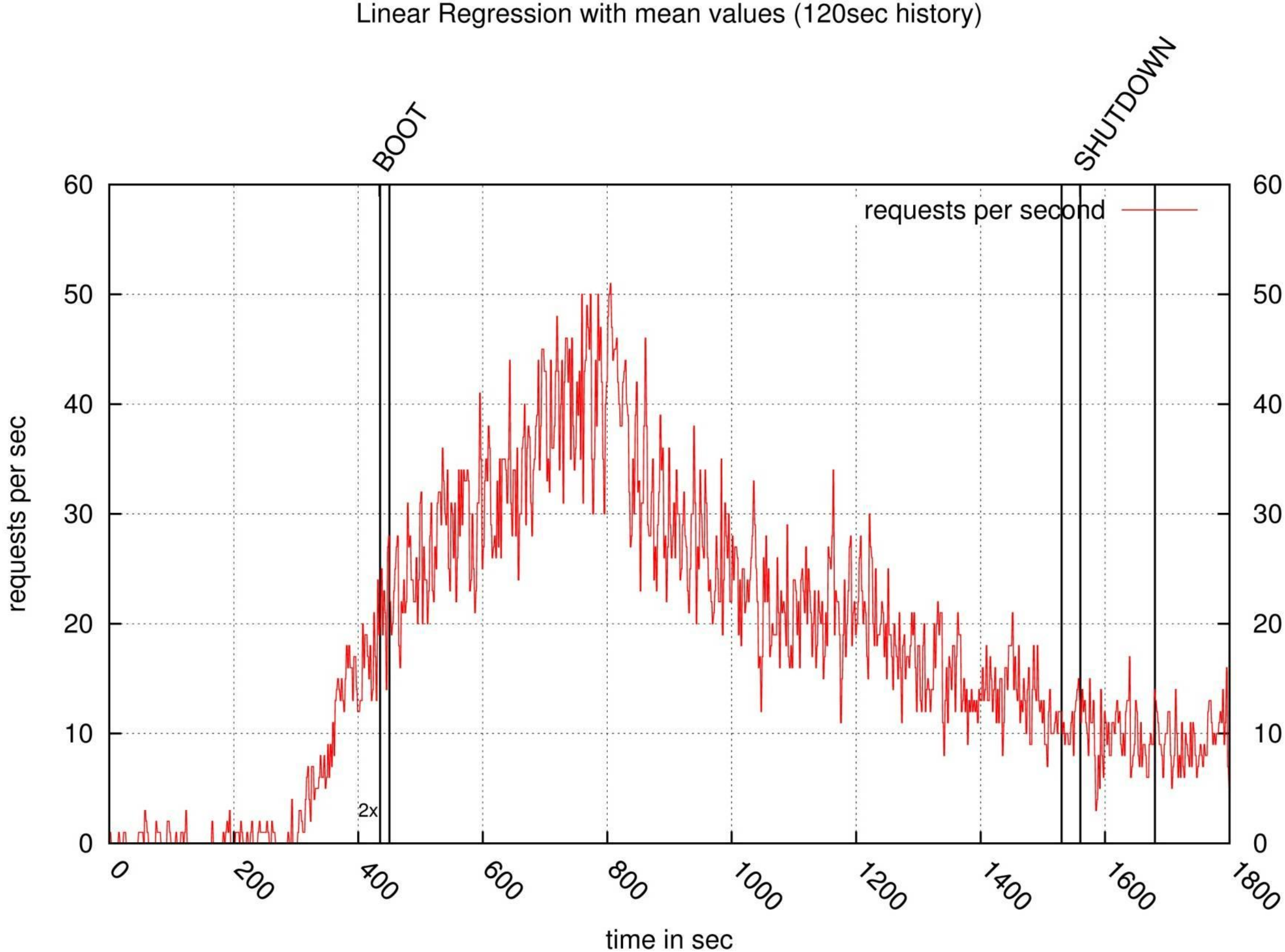


Linear Regression #2 (mean load)

- Overload by 80% saturation
- Underload (dynamic) by 40% saturation
- Load forecasting with linear regression
  - 120 seconds history
  - Use mean load (last 15 sec) as current load base

Metric	Avg.	Ref. / Opt.
SLA in %	99.79	99.63
First Response Time in msec	34.07	15.07
Downtime in min	10.89	0 /14
Deviation from optimum in %	22.21	100

Linear Regression #2 (mean load)



Outline

- Cluster Basics
- Motivation
- Energy Saving Daemon (Cherub)
- Load Forecasting
- Evaluation
- Conclusion & Future Work

Universität Potsdam
Mathematisch-Naturwissenschaftliche Fakultät

Conclusion

- Optimal Maximum Downtime:
  14 minutes (100%)
- With Linear Regression we achieved:
  - 12.87 minutes (92%) (current load)
    while maintaining the SLA at 99.40%
  - 10.89 minutes (78%) (mean load)
    while maintaining the SLA at 99.79%

Conclusion

- Load Forecasting can significantly increase the functionality of on/off algorithms
- Dynamic thresholds making configuration easier and supporting on/off algorithms as well

Future Work

- Prove, that this method scales.
- At the moment:
  Environment Simulator for Cherub, for emulating any number of back end nodes.
- Strategy adaptation for heterogeneous clusters
- What about curve fitting for even better forecasting? Faster peak detection?

Thank you for your attention!
Any Questions?

Contact:

kiertscher@cs.uni-potsdam.de

www.cs.uni-potsdam.de

Literature

[1] E. Pinheiro, R. Bianchini, E. V. Carrera, and T. Heath.
Compilers and operating systems for low power.
chapter Dynamic cluster reconfiguration for power and performance, pages 75–93.
Kluwer Academic Publishers, Norwell, MA, USA, 2003.

[2] K. Rajamani and C. Lefurgy.
On evaluating request-distribution schemes for saving energy in server clusters.
In Proceedings of the 2003 IEEE International Symposium on Performance Analysis of Systems and Software, ISPASS ’03, pages 111–122.
IEEE Computer Society, Washington, DC, USA, 2003.

[3] C. Santana, J. C. B. Leite, and D. Mossé.
Power management by load forecasting in web server clusters.
Cluster Computing, 14(4), 2011.

[4] J. L. Berral, I. n. Goiri, R. Nou, F. Julià, J. Guitart, R. Gavaldà, and J. Torres.
Towards energy-aware scheduling in data centers using machine learning.
In Proceedings of the 1st International Conference on Energy-Efficient Computing and Networking, e-Energy ’10, pages 215–224.
ACM, New York, NY, USA, 2010.

Appendix

- 1 Front-end
  - running Linux Virtual Server (LVS-1.2.1)
  - AMD Opteron with 2x 1,8GHz
  - 4 GB RAM
- 2 Homogenous Back-ends
  - each running a Wikipedia instance from Jan. 2008 with more than 6 mio. english articles
  - Intel(R) Xeon(R) with 2x 1,86GHz
  - 4 GB RAM
- 1 Load generator
  - running http_load (version from 12.03.2006, with seed option) / servload (0.5)
  - AMD Opteron with 2x 1,8GHz
  - 4 GB RAM

Appendix

- Additional Software on the Back-ends

Tool	Version	Release
Apache (httpd)	2.2.3	53.el5.centos.3
PHP	5.1.6	27.el5_5.3
MySQL	14.12	4.el5_6.6
Mediawiki	1.16.5	-

The Trace

- In Numbers:
  - 31806 Requests
  - 3 Parts
    - First, constant very low load
    - Second, strong positive slope
    - Third, weak negative slope

Part of the Trace	Avg. req/sec	Standard deviation in req/seq	Slope in req/sec² (req/min²)
0-5min	0,47	0,77	-
5-13,33min	26,60	12,04	0,076 (4,6)
13,33-30min	18,38	8,95	-0,027 (-1,6)

Full State Transitions