LIMBO: Modeling of Load Intensity Profiles

SPEC RG, DevOps Performance WG

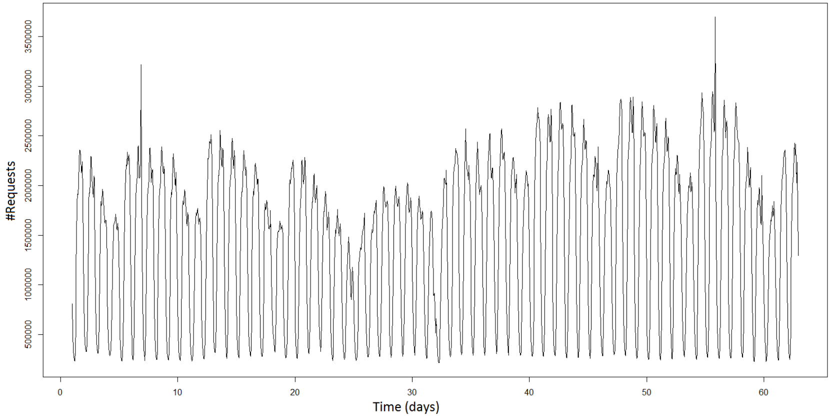
Joakim v. Kistowski, Nikolas Herbst

Chair for Software-Engineering, Uni Würzburg

8.8.2014

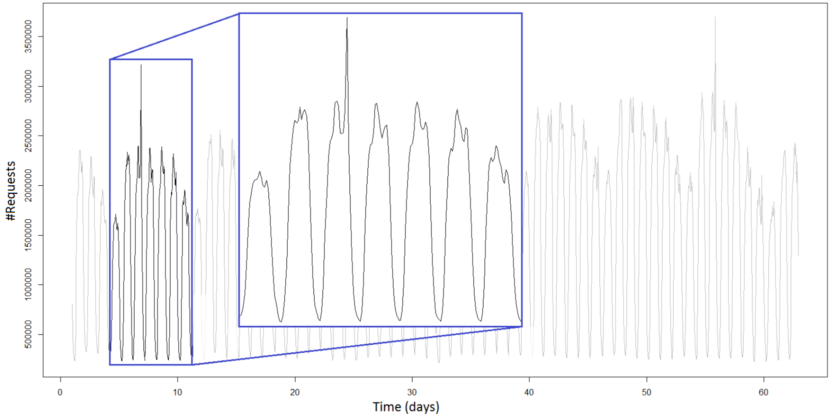
Motivation

- Page Requests for the German Wikipedia



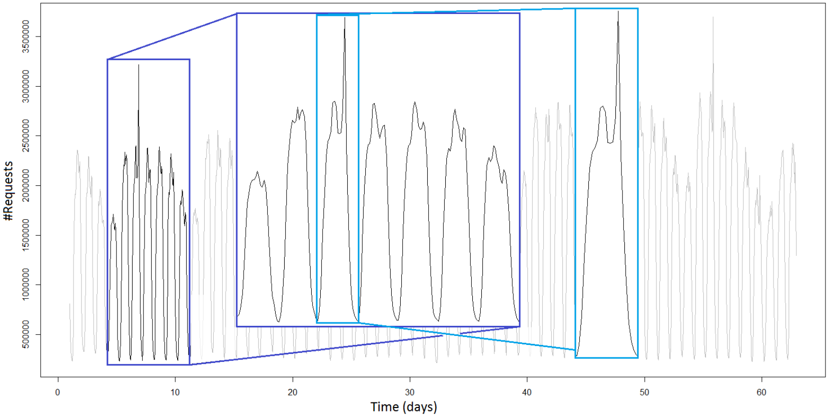
Motivation

- Page Requests for the German Wikipedia

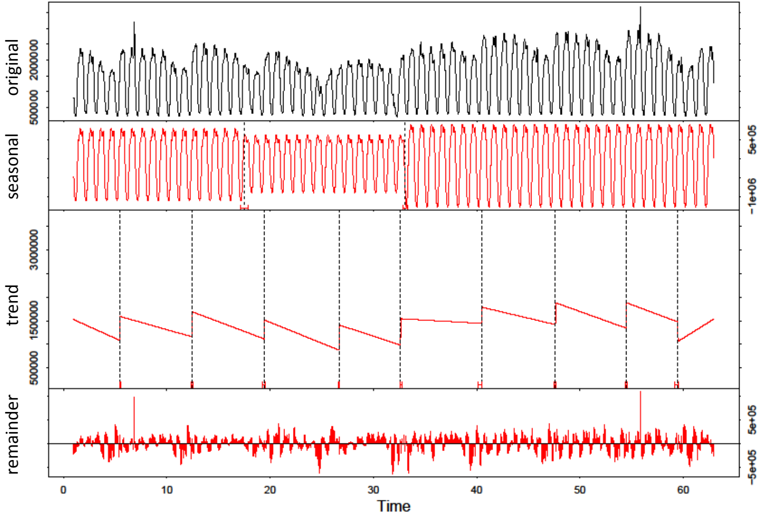


Motivation

- Page Requests for the German Wikipedia



Motivation



Additive decomposition into seasonal part, trend, and remainder. Created using *BFAST* [1].

Outline

- **Problem:** No means to effectively capture, reproduce, and modify varying load intensity of real-world cloud systems
- **Idea:** Support load intensity profile description by creating a meta-model and tooling
- **Benefits:** Enable more precise communication and creation of realistic load scenarios for benchmarking
- **Actions:** Creation of meta-models, processes, and tools for load intensity extraction and description

Related Work

User Behavior Models (e.g. using Markov Chains)

- van Hoorn et al. (2008): probabilistic, intensity-varying workloads
- Roy et al. (2013): workload volatility of a streaming system

Workload Models

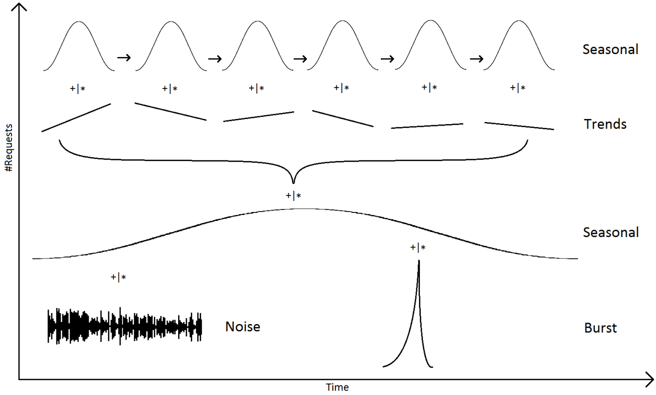
- Barford et al. (1998): file popularity and distribution (web)
- Casale et al. (2012): bursts
- Beich et al. (2010): data popularity and user classes (cloud)

Statistical Models

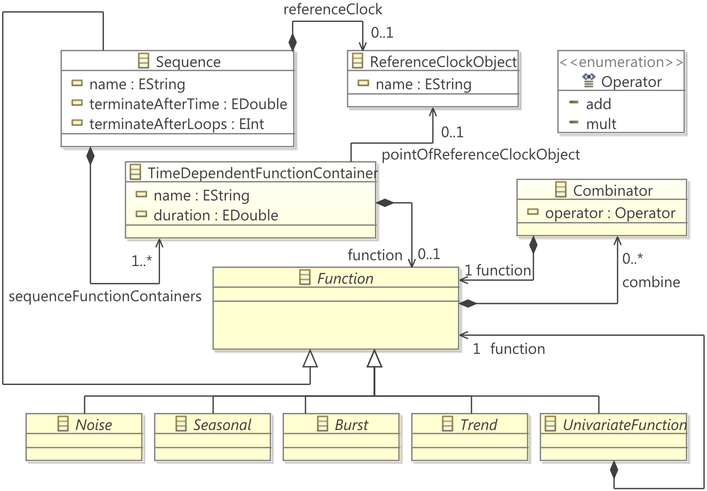
- Feitelson (2002): workload representativity through statistical characteristics

Descartes Load Intensity Model (DLIM)

- Describes arrival rate variations over time
- Provides structure for piece-wise mathematical functions
- Independent of work/request type

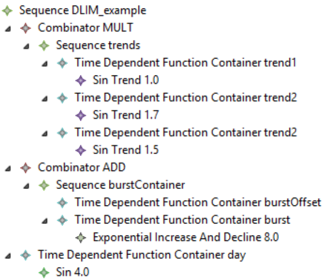


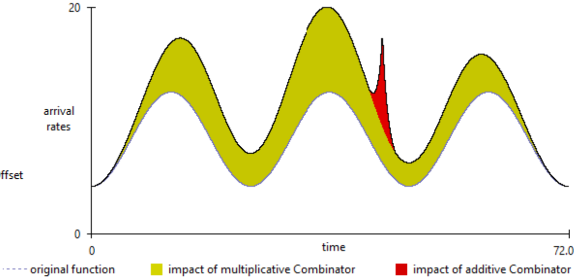
Descartes Load Intensity Model (DLIM)



DLIM Example Instance

- Created using LIMBO eclipse plugin [1]
- Contains *Seasonal* part, *Trends*, and *Burst*





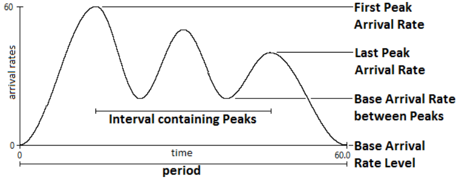
[1] LIMBO: `http://go.uni-wuerzburg.de/limbo`

high-level DLIM

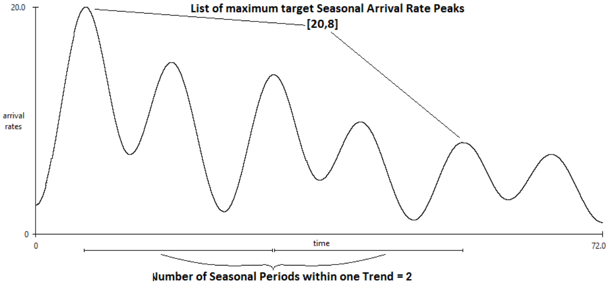
- Benefits of DLIM:
  - Powerful and expressive
  - Easy derivation of arrival rates or request time-stamps
- Drawbacks of DLIM:
  - Instances can become complex
  - Large trees may be unintuitive
- Solution: **high-level DLIM**
  - Fewer parameters for load intensity profile description
  - Strictly structured into single *Seasonal*, *Trend*, recurring *Burst*, and *Noise* parts

hl-DLIM Seasonal and Trend parts

- hl-DLIM *Seasonal* part:

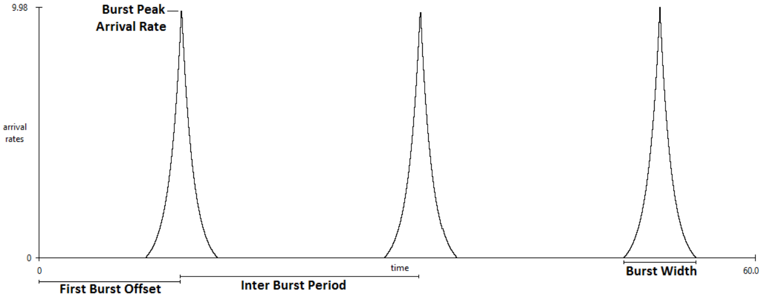


- hl-DLIM *Trend* part:



hl-DLIM Burst and Noise parts

- hl-DLIM *Burst* part:



- hl-DLIM *Noise* part:

Uniform Distribution

- Minimum Noise Rate
- Maximum Noise Rate

hl-DLIM Example Instance

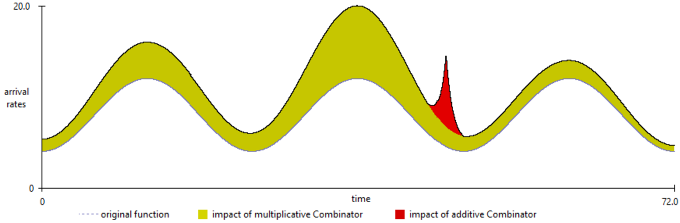
- *Seasonal* part:

  Period: 24
  Peaks per Seasonal: 1
  Base Arrival Rate: 4
  First Peak Arrival Rate: 12

- *Burst* part:

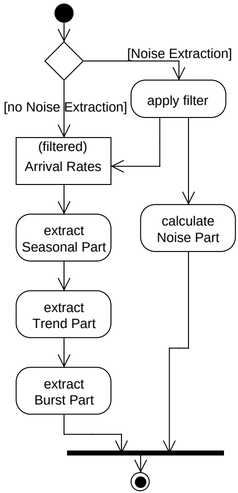
  First Burst Offset: 46
  Burst Peak Arrival Rate: 8
  Burst Width: 4

- *Trend* part:
  - Number of Seasonal Periods within one Trend: 1
  - Trend List: 16, 20, 14



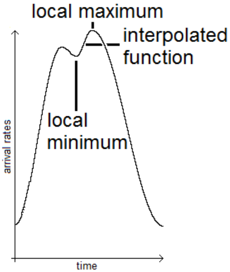
Automated Model Instance Extraction I

- Automated process for extracting DLIM or hl-DLIM instances from existing arrival rate traces
- Structured into *Seasonal*, *Trend*, *Burst*, and *Noise* part extraction
- Noise reduction and extraction is optional and separate

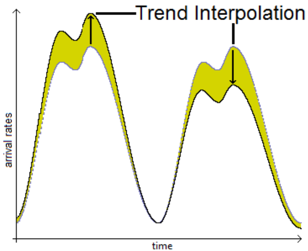


Automated Model Instance Extraction II

- **Seasonal Part**:
  - Extracts median local min/max within *Seasonal* iterations
  - Interpolates using DLIM *Functions*



- **Trend Part**:
  - Adds at each maximum *Seasonal* Peak to trend-list



Automated Model Instance Extraction III

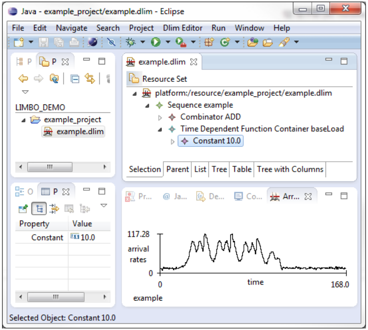
- **Burst Part**:
  - Bursts are detected at strong positive deviations from predicted *Seasonal* and *Trend* behavior
  - Peak is set to match arrival rate in trace
- **Noise Part**:
  - Before Extraction: High frequencies are reduced using a gaussian filter
  - After Extraction: Reduced noise (normal) destribution is added to model instance

Automated Model Instance Extraction IV

- **Simple Model Instance Extraction Process (S-MIEP)**:
  - Uses a single Trend-List to describe one overlying *Trend Part*
  - Extracts a DLIM instance
- **Periodic Model Instance Extraction Process (P-MIEP)**:
  - Uses a multiple recurring Trend-Lists to describe repeating trends
  - Extracts a DLIM instance
- **high-level Model Instance Extraction Process (hl-MIEP)**:
  - Modified version of the Simple Model Instance Extraction Process
  - Extracts an hl-DLIM instance

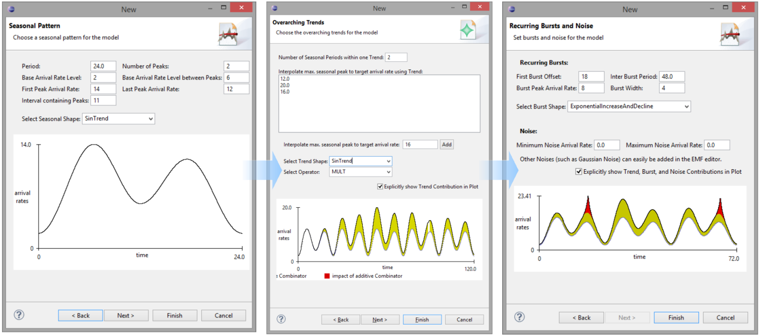
LIMBO

- EMF-based modeling platform
- Uses DLIM for load intensity description
- New Model Creation Wizard based on hl-DLIM
- Allows arrival rate and request time-stamp generation
- Visualizes and compares arrival rate profiles
- Provides automated model instance extraction



LIMBO - Model Creation

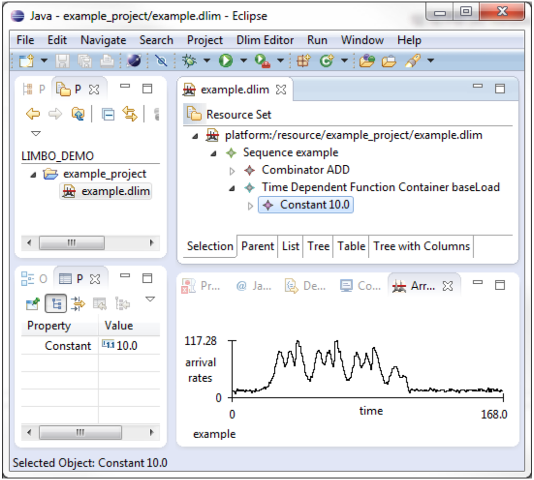
- Use hl-DLIM based wizard



- Alternatively: Extract model instance

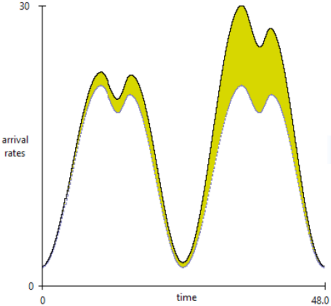
LIMBO - Model Refinement

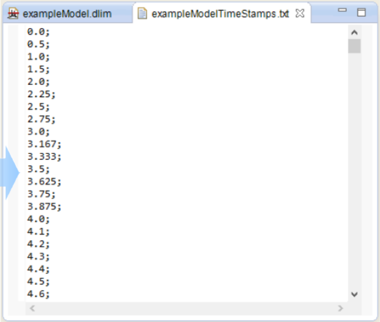
- EMF-Editor for customization of DLIM instances



LIMBO - Time-Stamp Generation

- Generate request time-stamps / arrival rates for benchmarking



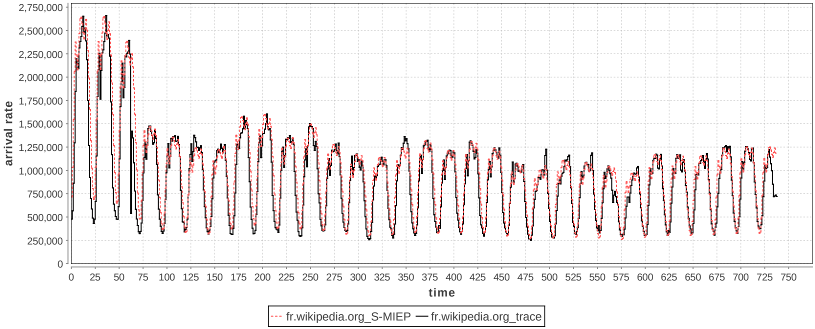


Evaluation

- **Usability Evaluation**
  - Usability evaluated using a questionnaire
  - Users are computer scientists from five different organizations
  - Mean Usability (1 = easy, 4 = difficult): **1.44**
  - Mean Feature Usefulness (1 = useful, 4 = not useful): **1.2**
- **Model Extraction Accuracy Evaluation**
  - 9 real world web server traces
  - Metric: median arrival rate deviation
  - S-MIEP and hl-MIEP applied to all traces
  - P-MIEP to traces longer than one month

S-MIEP I

- Most accurate of the extraction processes
- Does not require noise reduction
- Median deviation across all traces: **12.4%**



French Wikipedia S-MIEP result, median deviation: 7.6%

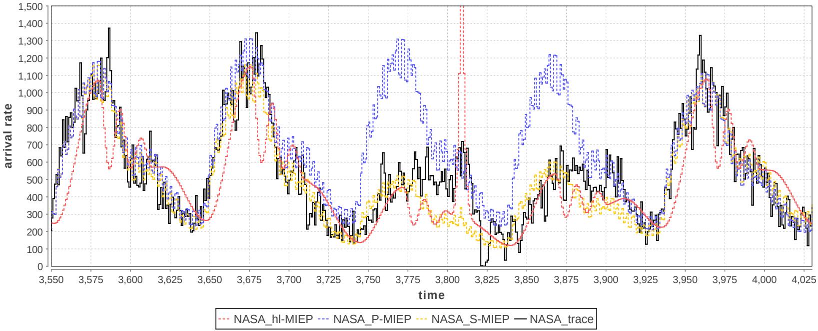
S-MIEP II

Trace	S-MIEP relative median error (%)	BFAST relative median error (%)
ClarkNet-HTTP	12.409	12.243
NASA-HTTP	18.812	-
Saskatchewan-HTTP	26.492	-
WorldCup98	12.979	-
IBM Transactions	74.368	-
de.wikipedia.org	8.538	11.223
fr.wikipedia.org	7.6	8.511
ru.wikipedia.org	9.912	5.809
wikipedia.org	4.855	2.302

- S-MIEP performs on average **8354** times faster than BFAST

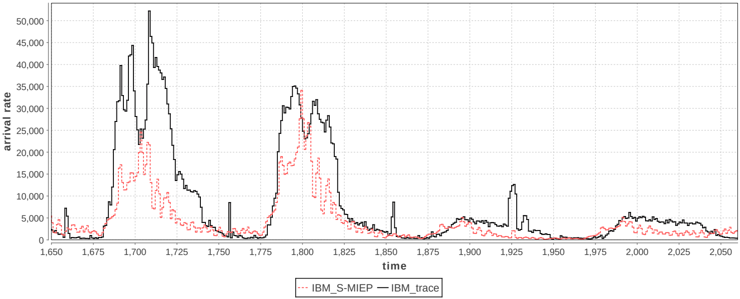
P-MIEP and hl-MIEP

- Both processes are less accurate than S-MIEP
- hl-MIEP: "recurring bursts" - limitation may lead to phantom bursts
- P-MIEP: ignores small deviations from recurring periodic patterns



Observations

- **S-MIEP** is the most accurate of the extraction processes
- **P-MIEP** works well for regular load intensity profiles
- **hl-MIEP** heavily relies on noise reduction
- Challenge: Seasonal pattern drift in long traces
  - Extraction uses one seasonal pattern for approximation



Summary I

- Two Meta-Models for load intensity variation description
  - **DLIM:** Powerful and expressive
  - **hl-DLIM:** Abstract and concise

  J. G. von Kistowski, N. R. Herbst, and S. Kounev, "Modeling Variations in Load Intensity over Time", in *Proceedings of the 3rd International Workshop on Large-Scale Testing (LT 2014)*. ACM, March 2014.

- Modeling Platform: **LIMBO**
  - Enables creation of custom load intensity variations for open workload based benchmarking
  - Provides automated load intensity profile extraction

  J. G. von Kistowski, N. R. Herbst, and S. Kounev, "LIMBO: A Tool For Modeling Variable Load Intensities (Demonstration Paper)", in *Proceedings of the 5th ACM/SPEC International Conference on Performance Engineering (ICPE 2014)*. ACM, March 2014.

Summary II

- LIMBO Usability (1 = easy, 4 = difficult): 1.44
- Automated model instance extraction:
  - **S-MIEP**: most accurate, median deviation: 12.4%
  - **P-MIEP**: good for regular profiles, median deviation: 37.6%
  - **hl-MIEP**: relies on noise reduction, median deviation: 15.6%
- LIMBO is open-source[2] and already being used in different contexts.

[2] LIMBO: `http://go.uni-wuerzburg.de/limbo`

Next Steps

- Our work on LIMBO:
  - Extraction of multiple and overlaying seasonal patterns
  - Change detection
  - Advanced calibration and noise reduction
- Ideas for integration/extension
  - Extend JMeter to use LIMBO timestamps
    → TimestampTimer by Andreas Weber (KIT)[3]
    → Combination with Markov4JMeter (AvH ?)
  - Extend PCM to use DLIM instances / LIMBO timestamps
    → Sebastian Lehrig (Uni Paderborn)
  - Using DLIM models for improving anomaly detection accuracy

[3] LIMBO: https://github.com/andreaswe/JMeterTimestampTimer

References

J. Verbesselt, R. Hyndman, G. Newnham, and D. Culvenor, "Detecting trend and seasonal changes in satellite image time series," *Remote Sensing of Environment*, vol. 114, no. 1, pp. 106 – 115, 2010. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S003442570900265X

A. van Hoorn, M. Rohr, and W. Hasselbring, "Generating probabilistic and intensity-varying workload for web-based software systems," in *Proceedings of the SPEC international workshop on Performance Evaluation: Metrics, Models and Benchmarks*, ser. SIPEW '08. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 124–143. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-69814-2_9

S. Roy, T. Begin, and P. Goncalves, "A complete framework for modelling and generating workload volatility of a vod system," in *Wireless Communications and Mobile Computing Conference (IWCMC), 2013 9th International*, 2013, pp. 1168–1174.

P. Barford and M. Crovella, "Generating representative web workloads for network and server performance evaluation," in *Proceedings of the 1998 ACM SIGMETRICS joint international conference on Measurement and modeling of computer systems*, ser. SIGMETRICS '98/PERFORMANCE '98. New York, NY, USA: ACM, 1998, pp. 151–160. [Online]. Available: http://doi.acm.org/10.1145/277851.277897

G. Casale, A. Kalbasi, D. Krishnamurthy, and J. Rolia, "Burn: Enabling workload burstiness in customized service benchmarks," *IEEE Transactions on Software Engineering*, vol. 38, no. 4, pp. 778–793, 2012.

A. Beitch, B. Liu, T. Yung, R. Griffith, A. Fox, and D. A. Patterson, "Rain: A workload generation toolkit for cloud computing applications," EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2010-14, Feb 2010. [Online]. Available: http://www.eecs.berkeley.edu/Pubs/TechRpts/2010/EECS-2010-14.html

D. Feitelson, "Workload modeling for performance evaluation," in *Performance Evaluation of Complex Systems: Techniques and Tools*, ser. Lecture Notes in Computer Science, M. Calzarossa and S. Tucci, Eds. Springer Berlin Heidelberg, 2002, vol. 2459, pp. 114–141. [Online]. Available: http://dx.doi.org/10.1007/3-540-45798-4_6