Performance Extrapolation across Servers

Subhasri Duttagupta

www.cmgindia.org

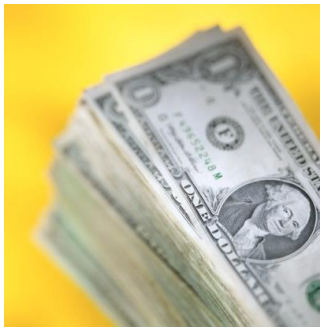
Outline

- **Why** do **performance extrapolation** **across servers?**
- **What** are the techniques for extrapolation?
  - **SPEC-Rates** of servers
  - **Single user service-demand** based technique
- **What information** **is required** for extrapolation?
- **How** the strategies perform for real applications?

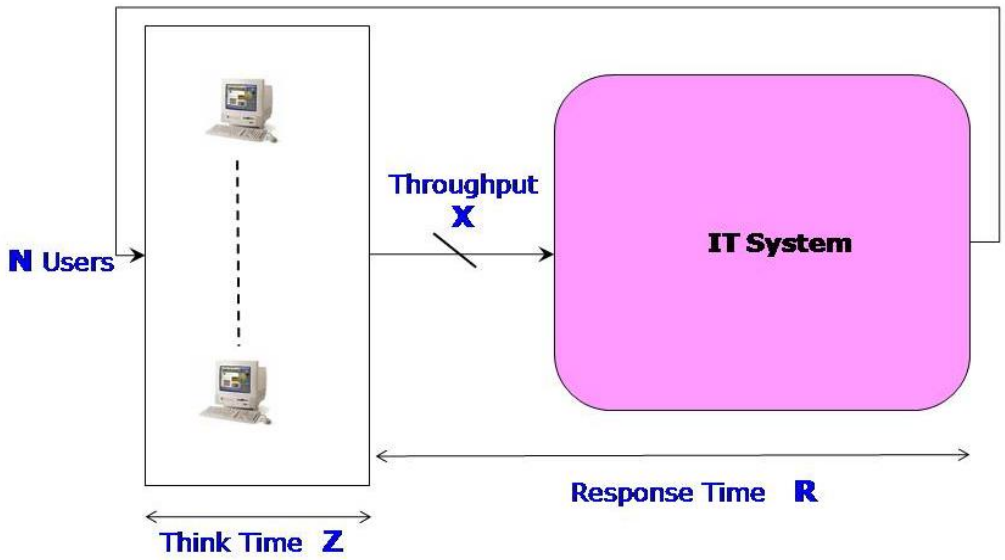
Why use Performance Extrapolation?

- Predict Performance of an application from **Test** **to** **Production** platform
  - **Reduce the effort, cost and time in load testing**
  - **Take into account difference in configuration between test and production servers**

- Predict performance for a large number of users
  - Not enough virtual user licenses



Context of Extrapolation



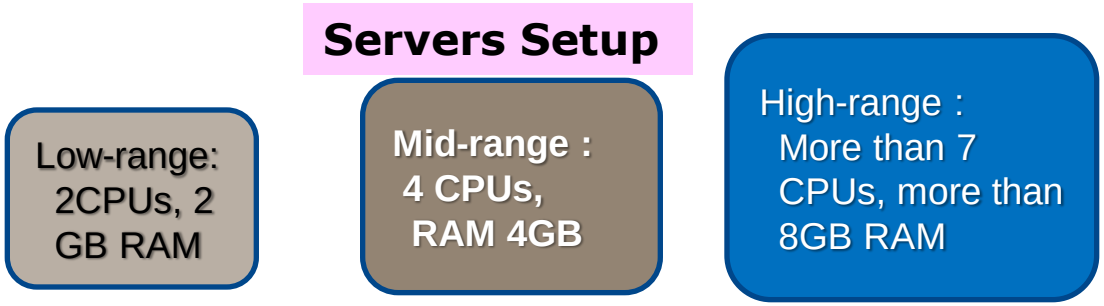
Testing Outcomes

- Maximum Throughput
- **Maximum no of users supported**
- **First bottleneck resource**

Assumptions

- **Application is scalable**
- **No performance optimization by the application**
- Application is deployed on the target environment.

Load Testing Setup



Sample Web Applications

- **iBatis JPetstore -** standarized J2EE benchmark
- **Mobile** Usage Reporting Application
- In-house vehicle **insurance** registration and renew policy
- Online **Quizzing** System.
- Rubi's auction site benchmark

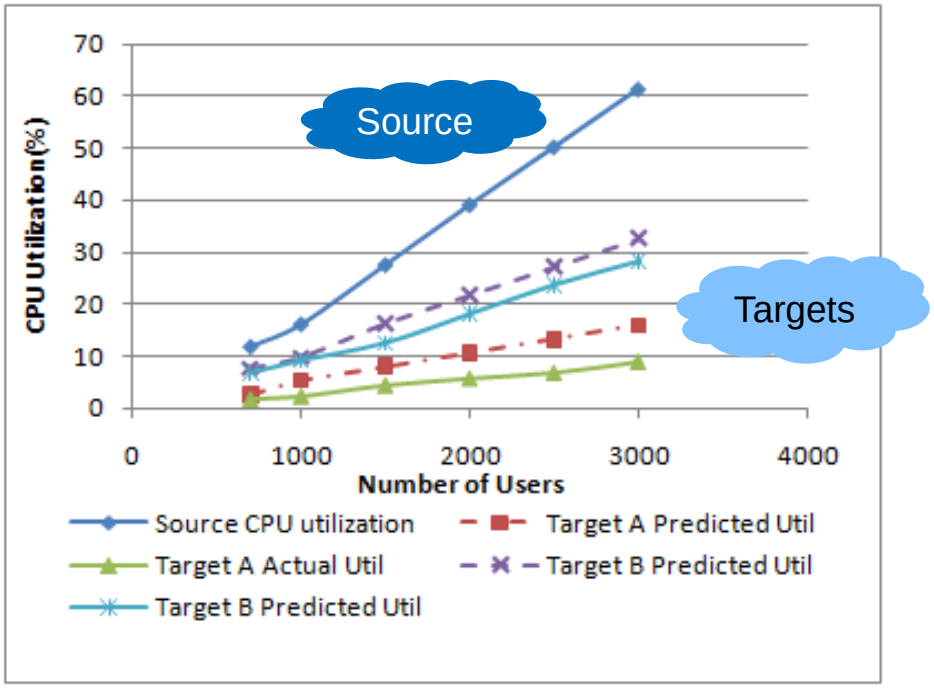
SPEC-rate Based Extrapolation

- **Map service demand** from platform A to Platform B
- **SPEC benchmark** Metrics ➔ Speed and Rate
  - **Speed** - time to complete a single task
  - **Rate** – overall system capacity to complete simultaneous tasks with multiple CPUs.
- **SPECint rate -** Can be used to map CPU performance between two servers

$$\text{service demand on B} = \text{service demand on A} \times \frac{\text{SPEC Int Rate on A}}{\text{SPEC Int Rate on B}}$$

- **Mean Value Analysis** for performance metrics

Results of Extrapolation (SPECrate)



SPECint_base 2006 Ratings
Source: 11.4
Target B : 17.7
Target A : 36.4

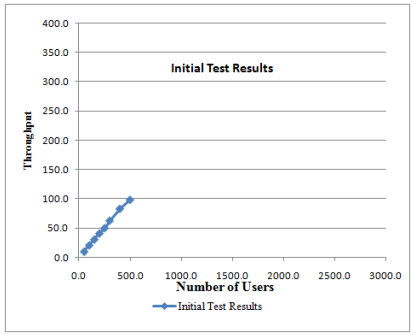
- Target B : Mid-range, Target A: High-range server
- Source: low-range server

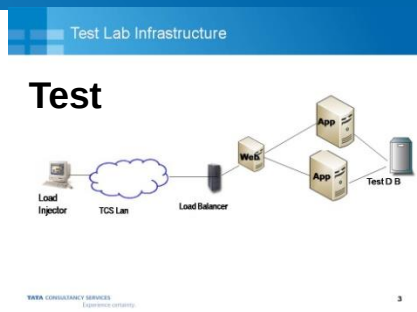
Performance for 2500 users

- *U %*(pred) = 26%, *U%(actual)* = 23.7% on target B
- *U% (*pred) = 11.4%, *U%* (actual) = 6.9% on target A

Service demand based Extrapolation

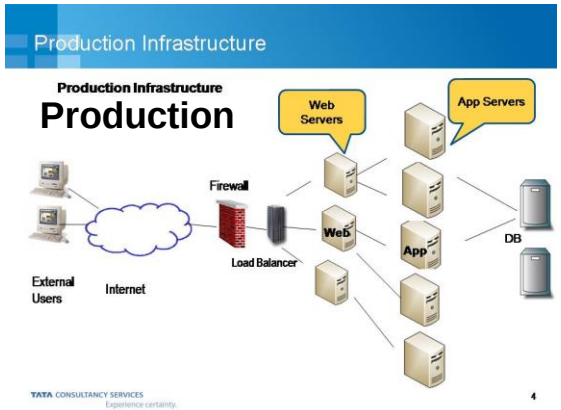
- **Initial Tests on Source**

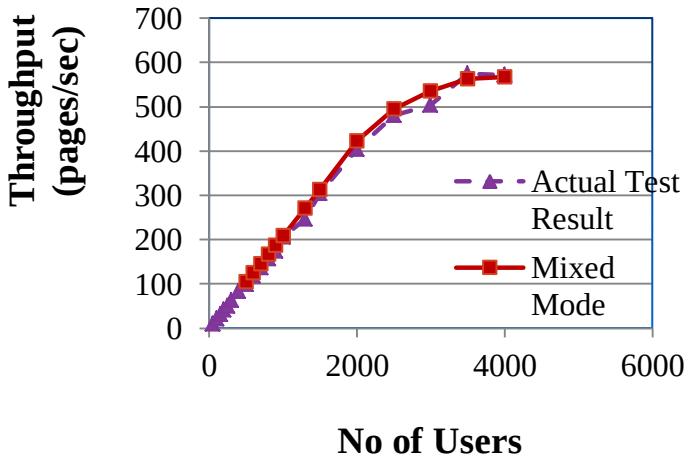




- **Single User Test** on Target
  - Captures basic application characteristics on Target.

- **Load Extrapolation on Target**





Steps Involved

- **Load testing on a Source platform for small number of users.**
  - Obtain throughput values $x_1, x_2 .. x_s$ for $N_1, N_2 \ldots N_s$ users
  - Assume same throughput on the target platform

- **Single User Test** on Target platform with **think time *Z=0***
  - Service demand for all the resources sd_{cpu}, sd_{disk}
  - $z \neq 0$, service demands are high
  - Utilization Law to obtain cpu%, disk%

- **Load Extrapolation on Target**
  - Make use of utilization and throughput values for $N_1, N_2 \ldots N_s$ users
  - Patented load extrapolation technique

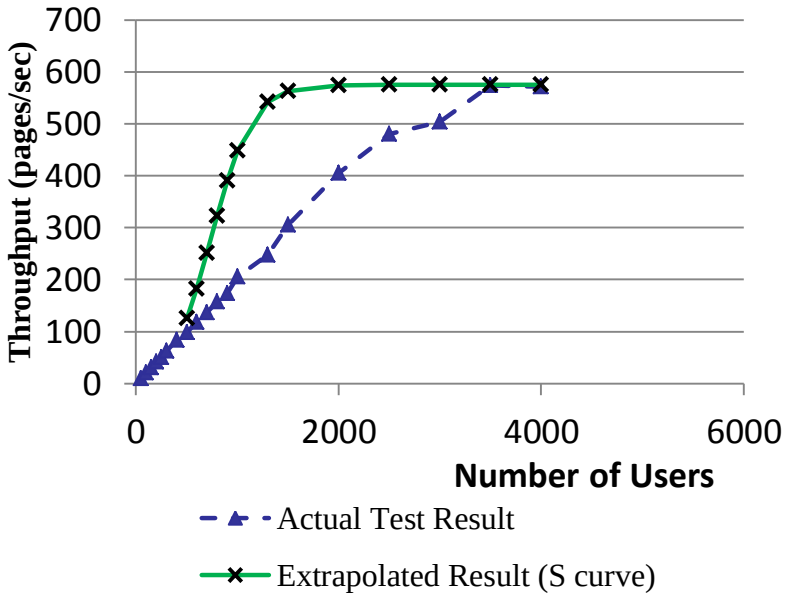
Load Extrapolation using Linear Regression

If no bottleneck, throughput increases linearly with the number of users.

- Linear regression is an obvious choice

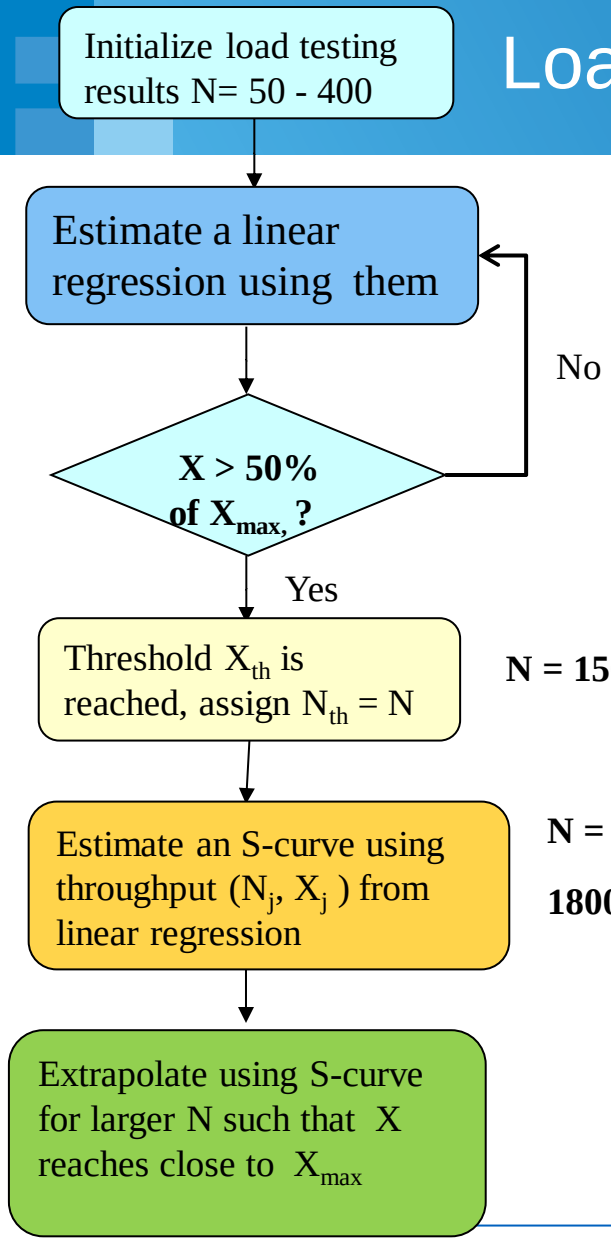
Load Extrapolation using S-curve

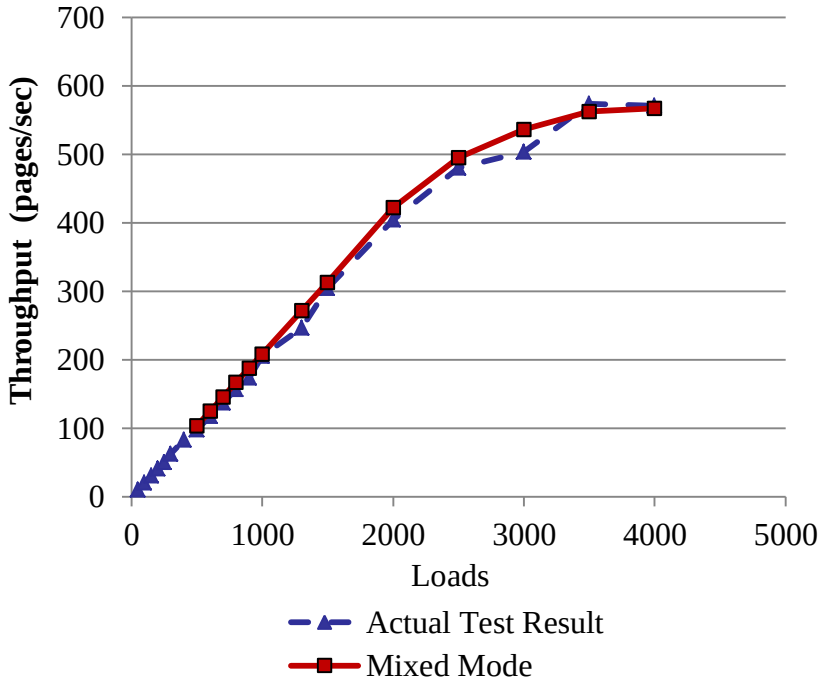
- S-curves represents rapid growth followed by saturation
- S-curve saturates fast



$$X = X_{\max} /[1 + a \times \exp(-bN)]$$

Load Extrapolation using Mixed Mode





High Accuracy

Predicted 536 pages/sec

Actual 504 pages/sec } ***N*= 2500 users**

What contributes to accuracy of Extrapolation?

- **Target and source may** differ
  - CPU configurations, # of cores, memory size and storage

- **Single user test** on the *target*
  - Captures impact of target architecture on the performance
  - Captures basic application characteristics

- **Mixed Mode strategy** on the target
    - Does not use MVA algorithm
    - Accurate prediction on throughput at high users
    - Load dependent Service Demand (yet to incorporate)

Case Study – RUBiS Auction Site

- **Load testing on low-range server for small number of users.**
  - $N_1 = 200$, $N_2 = 400$, $N_s = 600$ users
  - $x_1 = 42$, $x_2 = 84$, $x_s = 127$ pages/sec

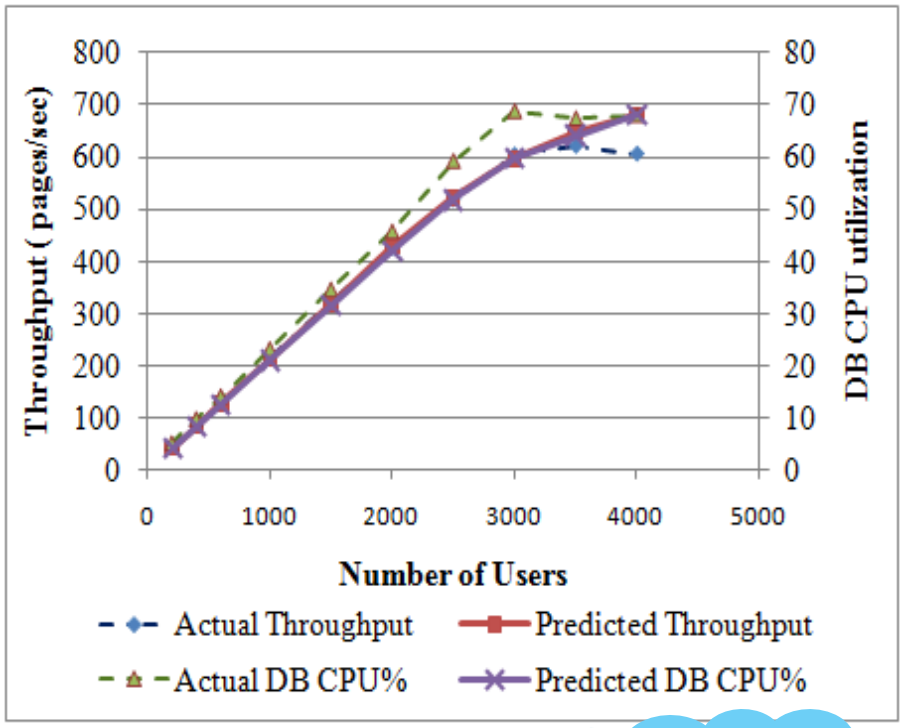
Same throughput on target

- **Single User Test** on Target platform with **think time *Z=0***
  - sd_{cpu},= 0.7 ms on **App Server**
  - sd_{cpu},= 1.08 ms sd_{disk} = 1.37 ms on **DB server**

- **Load Extrapolation on Target using PerfExt**
  - $N_1 = 200$, $N_2 = 400$ $N_s = 600$ users
  - Utilization law to obtain utilization of resources
  - DB server *CPU = 4.1%,9.5%, 13.5% , Disk% = 5.7%, 11.5%, 17.4%*

Sample Application: RUBiS system

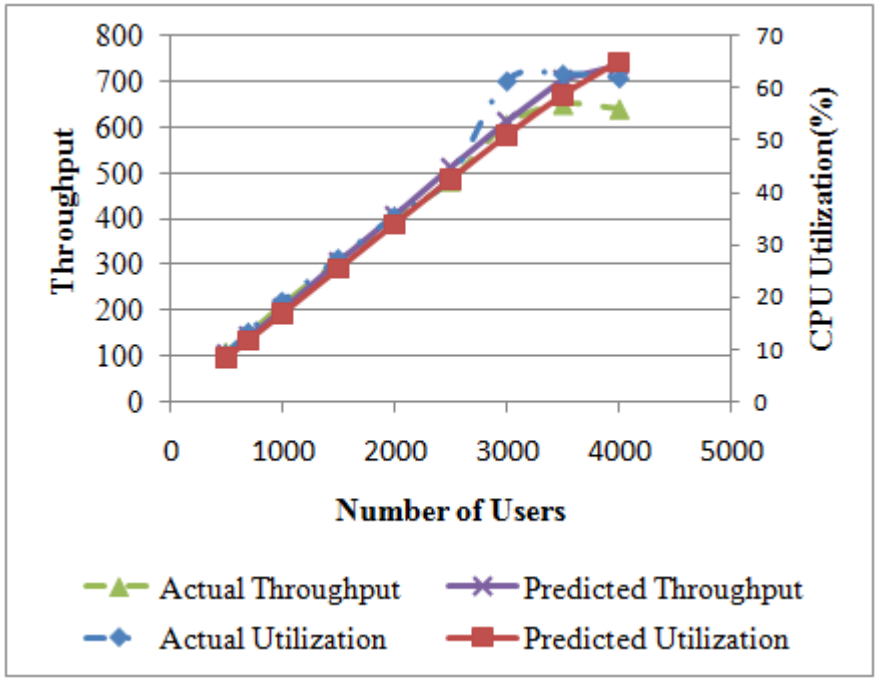
DB Disk is the bottleneck



High IO wait% at $N > 3000$

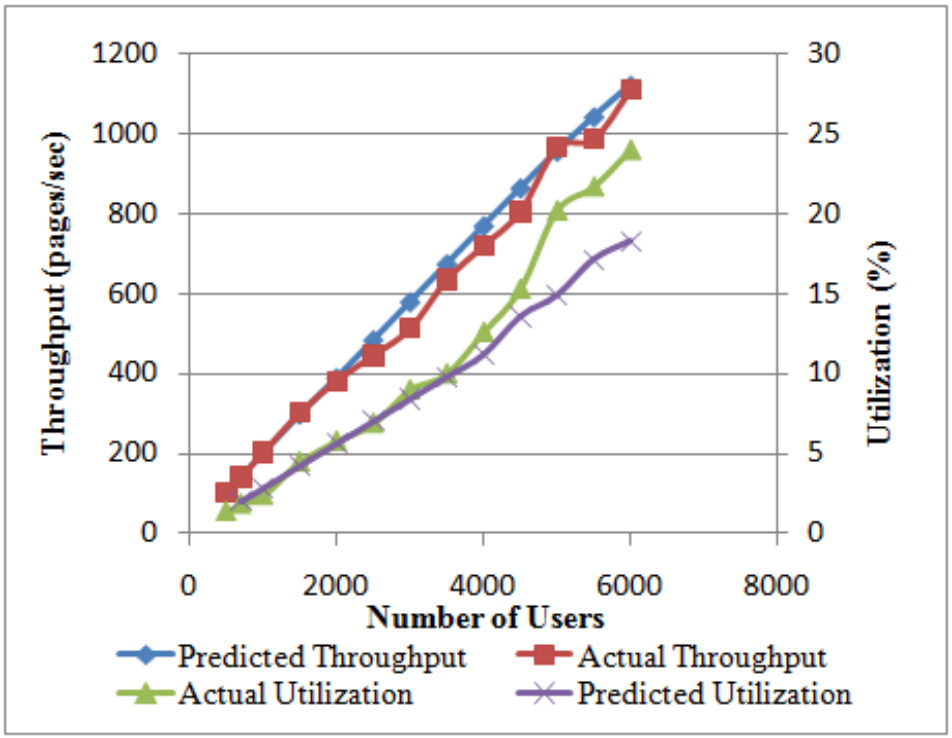
- Browsing transactions and bidding mix with 15% read write transactions
- Target mid-range Server Initial test (N = 200, 400 and 600 users)
- DB server $D_{CPU} = 1.08$ ms $D_{Disk} = 1.5$ ms
- Actual and predicated throughput match

Server Extrapolation on Sample Application: JPetstore



- Predict on Mid-range Server
- Initial test (*N* = 500, 700)
- Application service demand on target:
  CPU= 0 .83 ms
  Disk = 0.94 ms
- Scales up-to 4000 users
- *U%* (pred) = 62.4%, *U%* (actual) = 58%
- *X* (pred) = 704, *X(actual)* = 650 pages/sec

Sample Application: Telecom Reporting

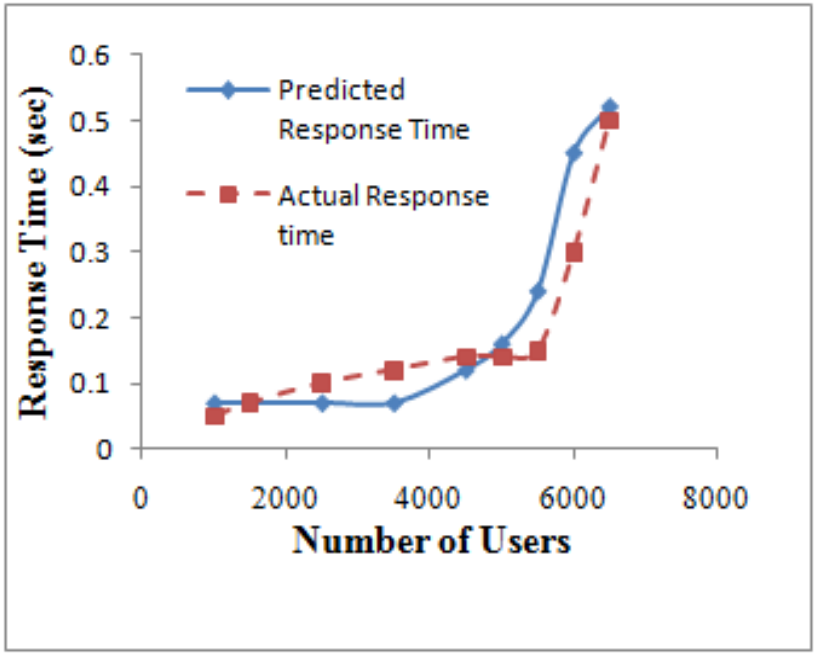


CPU% differs at *N* > *4000* users

- Target is a High-range Server
- Initial test (*N* = 400, 500)
- Application service demand on target: *CPU= 0.14 ms*
  Network = 0.65 ms
- Scales up-to 6000 users
- *U%* (pred) = 17.1%
- *U%* (actual) = 21%
- *X* (pred) = 1041 pages/sec
- *X(actual)* = 986 pages/sec

Sample Application: Telecom Reporting

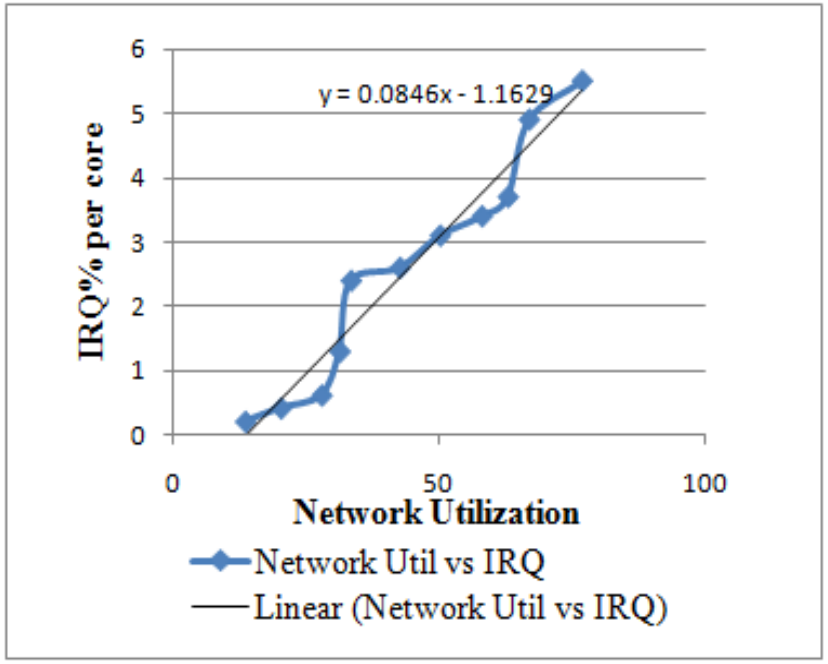
Actual vs. Predicted Response time



- Extrapolated using Little's Law
- *N*= 6000 users, response time increases rapidly
- Actual and predicated response time follow the same trend
- *R*(pred) = 0.45 sec
- *R(actual*) = 0.3 sec

Why Predicted and Actual CPU% differ?

CPU IRQ% increases linearly with Net%



- Model Load Dependent Service Demand
- At high number of users, network utilization is high
- Software interrupt is raised *when a* packet is received
- Linear relationship depends on Application only.

Limitations of the current strategy

- Running **Single user test** on *Production* Environment
- *Alternatives..*
  - obtain resource consumption by running Micro-benchmarks
  - Use a number of micro-benchmarks to characterize application

- **Good for Online Applications**
  - Throughput, Response time are main input values

- **Not tested for Open Systems and Batch Jobs**

- **Not Capacity Planning tool**

Related Work

- **Simulation tool** [ARSH1996]
- Gather trace for *Parallel Programs* [MALO1995] and run on a trace-driven simulator
- **Build analytical model for Specific Applications [GIMA2004, KOU2003]**
- **Architecture simulator [YOUR2007] if target environment not available**
- **Build Performance Model and predict performance**
  - **METASIM [CARR2003] convolves application signature and machine profile**
  - **[MARI2004] Gather architecture neutral characteristics and map to different architecture for Cross-architecture performance prediction**

Conclusions

- Extrapolation strategy from **test** to **production**
- *Load test* *small number* of users on the test, followed by single user test on the production
- **Load Extrapolation** strategy for smaller to larger load
- **High Accuracy for a number of Sample applications**
- **Future Work**: Characterize an application and Predict performance without single user test on the target

References

- [ARSH1996] H. Arsham, "Performance Extrapolation in Discrete-event Systems Simulation," Int. Journal of Systems Science, vol. 27, no. 9, 1996 pp. 863-869.
- [MALO1995] A.D. Malony and K. Shanmugam, "Performance extrapolation of parallel programs," in Proc. of Int. Conf. on Parallel Processing, 1995, pp. 117-120.
- [GIMA2004] R. Gimarc, A Spellmann, and J. Reynolds, "Moving Beyond Test and Guess - Using modeling with load testing to improve web application readiness," in Computer Measurement Group's Conference, 2004, pp 429-444.
- [KOUN2003] S. Kounev and A. Buchmann, "Performance Modeling and evaluation of large-scale J2EE applications," in Computer Measurement Group's Conference, 2003, pp 273-284.
- [DUTT2012] Subhasri Duttagupta and Rupinder Virk, "PerfExt: Performance Extrapolation Tool," in Proc. of Int. Conf on Computational Intelligence, Modeling and Simulation, CIMSIM, 2012.
- [YOUR2007] Matt T. Yourst, "PTLsim: A Cycle Accurate Full System x86-64 Micro architectural Simulator," in Proc. of Int. Symposium on Performance Analysis of Systems and Software, 2007.
- [CARR2003] L. Carrington, A. Snavely, X. Gao, and N. Wolter, "A Performance Prediction Framework for Scientific Applications," in Proc. of Int. Conf. on Computational Science Workshop on Performance Modeling and Analysis, 2003.
- [MARI2004] G. G. Marin and J. Mellor-Crummey, Cross-Architecture Performance Predictions for Scientific Applications using Parameterized Models.: Proc. of ACM SIGMETRICS, 2004, pp 2-13.

Thank You