B. TECH. PROJECT REPORT

ON

WHAT-IF ANALYSIS ENGINE

MAKING ENTERPRISE IT SYSTEMS AGILE

BY

SIDDHARTH SHANKAR PRASAD

DISCIPLINE OF COMPUTER SCIENCE AND ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY INDORE

NOVEMBER 2017

A PROJECT REPORT ON

What-If Analysis Engine

Making Enterprise IT Systems Agile

Submitted in partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY
in
COMPUTER SCIENCE AND ENGINEERING

Submitted by:
Siddharth Shankar Prasad (Roll No. 140001032)

Guided by:
Dr. Gourinath Banda, Associate Professor, CSE, IIT Indore
Dr. Maitreya Natu, Scientist, Tata Research Development and Design Center
Dr. Vaishali Sadaphal, Scientist, Tata Research Development and Design Center

INDIAN INSTITUTE OF TECHNOLOGY INDORE
November 2017

Declaration

I hereby declare that the project entitled **"What-If analysis engine: Making Enterprise IT Systems Agile"** submitted in partial fulfillment for the award of the degree of Bachelor of Technology in 'Computer Science and Engineering' completed under the supervision of **Dr. Gourinath Banda, Associate Professor, Computer Science and Engineering, IIT Indore and Dr. Maitreya Natu, Scientist, Tata Research Development and Design Center** is an authentic work.

Further, I declare that I have not submitted this work for the award of any other degree elsewhere.

Siddharth Shankar Prasad

Certificate by BTP guide(s)

It is certified that the above statement made by the students is correct to the best of my knowledge.

Dr. Maitreya Natu

Scientist

Tata Research Development and Design Center

Preface

This report on “What-If analysis engine: Making Enterprise IT Systems Agile" is prepared under the guidance of Dr. Maitreya Natu, Dr. Vaishali Sadaphal and Dr. Gourinath Banda based on the work I did at Tata Research Development and Design Center, Pune.

Through this report we propose to develop a change-impact analysis engine, that can enable users to assess the impact of various hypothetical scenarios of business and infrastructure changes.

We have tried to the best of our abilities and knowledge to explain the content in a lucid manner.

Acknowledgements

I would like to thank my BTP supervisors Dr. Maitreya Natu, Dr. Vaishali Sadaphal and Dr. Gourinath Banda for their very important role as mentors, motivators, and co-authors. I am lucky to have had the benefit of their wisdom and their enthusiasm to develop ideas into working solutions.

Without their support this report would not have been possible.

Siddharth Shankar Prasad

B.Tech. IV Year

Discipline of Computer Science and Engineering

IIT Indore

Abstract

Enterprise systems often go through many changes in business logic and infrastructure. Enterprise architects therefore frequently need to plan to accommodate growth and changes. They need to be able to determine what effect these changes would have on their system.

Most of the planning processes these days is manual and intuition-driven which is slow and sub-optimal. Through this project, we propose to bring a paradigm-shift from manual, intuition-driven approach to an automated, analytics-driven approach.

We propose to develop a change-impact analysis engine, that can enable users to assess the impact of various hypothetical scenarios of business and infrastructure changes.

Contents

1. Introduction……………………………………………………...…1

1.1. Problem Statement………………………………………………...2

1.2. Scope……………………………………………………………..…2

1.3. What-If Analysis…………………………………………………...3

1.4. Literature Review…………………………………………………..3

1.5. Contribution………………………………………………………..4

2. Design Rationale……………………………………………….…..5

2.1. Data Sources……………………………………………………..…6

2.2. Offline……………………………………………………………....6

2.2.1. Constructing Dependency Model………………………….…6

2.2.2. Deriving Behavior Model……………………………………7

2.3. Online……………………………………………………………....8

2.3.1. Inserting Change Scenario………………………………....8

2.3.2. Impact Propagation……………………………………...9

3. Proposed Approach……………………………………………….11

3.1. Building Behavior Models…………………………………………12

3.2. Performing Analysis of What-If Scenarios....………………...13

3.2.1. Change in Workload……………………………………...13

3.2.2. Change in Workload Composition……………………..13

3.2.3. Vertical Scaling……………………………………………..14

3.2.4. Horizontal Scaling…………………………………..14

4. Experimental Results…………………………………………..17

4.1. Workload Change…..……………………………………18

4.2. Workload Composition Change..…..…………………………...19

4.3. Vertical Scale……………………………………………...21

4.4. Horizontal Scale…………………………………………...22

4.5. Combination………………………………………………23

5. Conclusion and Future Work…………………………………….27

6. References…………………………………………………….29

7. Appendix……………………………………………………..31

Chapter 1

Introduction

Enterprises go through growth and continuous improvements. Hence, often go through many changes. Therefore, enterprise architects frequently need to plan to accommodate growth and changes. Planning involves determining what effect a business or infrastructure change would have on their system. Changes could include increase or decrease in workload, changes in business logic, infrastructure changes etc.

Further, after determining the effect, as part of the planning process, architects need to recommend changes to mitigate any negative impact that may occur, and again, need to ensure that the recommended change achieves the desired results.

Most of the planning processes today is manual and intuition-driven which is slow and sub-optimal. We propose to bring a paradigm-shift from manual, intuition-driven approach to an automated, analytics-driven approach.

To illustrate how a What-If analysis engine could be useful, consider Bank-A. This bank has planned a merger with Bank-B in a few months. This merger will no doubt lead to an increase in the workload on their systems. The analysts of Bank-A predict that there will be an increase which is approximately three times the current workload on the system and notify the enterprise architects to prepare the systems for the same.

Now that the system architects have a working What-If analysis engine, all they have to do is enter this scenario of increase in workload into the engine, to see its impact. The engine notifies them that the system would have a five-fold degradation in performance and the compute resource would saturate and therefore would need more compute power to handle it. The architect then comes up with a few ways to solve this problem. He looks at the impact of adding more CPUs and also the impact if upgrading the already existing CPUs. Based on these results and other factors such as available budget, time to procure and the respective improvements in performance, he makes a decision. We can see that the what-if engine saved Bank-A a lot of time and money, and made the life of the architects much easier.

We propose to develop such a change-impact analysis engine, that can enable users to assess the impact of various hypothetical scenarios of business and infrastructure changes.

1.1 Problem Statement

Develop an engine to perform change-impact analysis of various hypothetical scenarios of business and infrastructure changes.

1.2 Scope

Change-impact analysis a.k.a, What-If analysis is a large space with many open issues. We will limit our work to multi-tier transactional systems with application layers of web, application, and database servers and infrastructure layers of compute, storage and network components. We will be handling change scenarios of change in workload, change in workload composition, horizontal scaling and vertical scaling (Horizontal and vertical scaling are explained later).

1.3 What-If Analysis

What-If analysis deals with the analysis of impact on the system as a result of the changes made to the system. Some questions that we would like to answer by performing what-if analysis are:

- What is the impact on the system if there is an increase in workload?
- What is the impact on the system if there is a change in the composition of workload?
- What is the impact of adding new CPUs to the system?
- What is the impact of using faster or slower CPUs?

The impact can be :

- On resource utilizations such as CPU utilization.
- On performance metrics such as response time.

1.4 Literature Review

Before performing any What-If analysis, we need to model the systems to understand relationships between various metrics in consideration. [5] describes a robust performance modeling architecture, which leverages the redundancy of high level system specifications described through models and low level system implementation to localize many types of system-model inconsistencies

Work has been done in the past on What-If analysis for distributed systems. Authors in WISE [4] answer questions related to deployment and configuration for CDNs. Authors in [3] present a self predicting clustered storage system. [6] predicts the impact of workload change in complex cloud applications. [7] presents Predico, a workload-based what-if analysis system that uses commonly available monitoring information in large scale systems to ask a variety of workload-based What-If queries about the system.

We noticed that in the literature we surveyed, What-If analysis that was performed was only predicting one output variable, for example request response time. Most systems were made specifically for the particular use-case that was defined. Also, the types of models that were to be used were pre-defined, most of the time.

1.5 Contribution

Our aim is to develop a generic methodology or engine to perform What-If analysis. In that, we predict more variables and provide more insight to the system designers than existing work in the area.

A large amount of the focus of this work is on identifying impact relationships, obtaining mathematical relationships between various metrics and deriving hidden relations. Modeling our engine to capture behavioral relationships using regression techniques.

These are further used to perform What-If analysis to assess impact of various change scenarios.

We then performed experimental evaluation on synthetic and real data.

Finally, a working prototype to demonstrate end-to-end execution of the change-impact analysis has been developed.

Chapter 2

Design Rationale

In this chapter, we explain the design rationale for performing What-If analysis. We perform what-if analysis on 3-tier systems with application layers, database servers and infrastructure layers of compute, storage and network components. The change scenarios that will be handled are (i) change in workload, (ii) change in workload composition, (iii) horizontal scaling and (iv) vertical scaling.

We split our method broadly into two parts – *online* and *offline.* The offline part consists of building the dependency graph and the behavior models from the given input data. The online part consists of taking the input from the user about the change scenario and then processing its impact on the system. The overall steps of our What-If analysis engine are explained below.

Offline:

1. Construct dependency model by modeling the system as a directed graph such that every node represents a component and every edge describes the relationship between the components.

2. Derive behavior models by finding relationships between node metrics.

Online:

3. Insert change, in other words, define a What-If scenario as input.
4. Compute impact of the What-If scenario that has been input.

2.1 Data Sources

The solution requires the following data:

- System components and relationships: Information about the system, its topology and the relationships between the components.
- System Trace: Domain knowledge that specifies which metric depends on which other metrics.
- Metric run history: Historical timeseries data of all the metrics.
- What-If scenario: User input that specifies the change whose impact needs to be computed.

2.2 Offline

2.2.1 Constructing Dependency Model

A graph is constructed with the components as nodes and the edges describing the relationship between the nodes. Each node is further attributed with static attributes such as Operating System and Processor and dynamic attributes such as CPU utilization and response time. The large scale and heterogeneity of the nodes and edges is a challenge here. The number of nodes in a system is usually very large. Relationships are heterogeneous – containment, communication, precedence, etc. and heterogenous nodes – databases, many types of applications, CPUs, load balancers, etc.

2.2.2 Deriving Behavior Model

This step involves deriving dependent metrics and deriving equations that best capture the relationships. [1] uses linear models with curve fitting using Ordinary Least Squares and Least Absolute Residuals.

Since the number of nodes is large, looking at all possible combinations to compute relationships leads to a combinatorial explosion. Hence, this would require intelligent ways of feature selection.

Once the groups of dependent metrics have been identified, computing a model between them is not straightforward. The relationship could be linear, non-linear viz. quadratic, exponential, logarithmic, etc. Many times, one relationship does not suffice. In such cases multiple models are required. Hence, a method that can fit any and all types is required. Eyeballing the data is not an option due to large scale. Further, the aim is to automate the process and obtain best relations.

Another requirement here is deriving the recent steady state. Just building the models on all of the given past data does not always work, as there may have been changes in the behavior due to changes made to the system in the past. Using the data before the change is not relevant to the current state. Thus, figuring out a steady state using which we build models is an area that needs work. [8] surveys methods to find change points in timeseries.

2.3 Online

2.3.1 Inserting Change Scenario

A change scenario is defined by the user. There are four types of what-if scenarios we are dealing with: (i) change in workload amount, (ii) change in workload composition, (iii) horizontal scale and (iv) vertical scale.

The workload can be of different types. It is possible that one type of request workload requires more resources and time than the others. Therefore, if a larger number of this type of request may start arriving. The overall workload amount may remain the same. However, there would still be a change in the system. This type of change is a change in the workload composition.

Horizontal scaling is done by adding more machines into our pool of resources, whereas vertical scaling is done by adding more power (CPU, RAM, etc.) to an existing machine.

Most of the time, the different types of requests are logged by the system. However, the monitoring of the resource underneath is only one that captures aggregated utilization. In such cases, we need to derive the resource requirement of each type of request to compute impact of change in workload composition.

If the types of workloads along with separate data for each type of request, we can go about creating separate models for each workload type. If the information of types of requests is not given, we require methods such as Independent Component Analysis blind source separation to infer the workload composition from the given data, as seen in [2].

2.3.2 Impact Propagation

Once the user has provided the input What-If scenario, the next step is to apply the models created offline while traversing the graph and evaluate the impact of the what-if scenario on the entire system.

Here, one needs to deal with the order of change propagation, cyclic dependencies and distribution of workload across components.

Chapter 3

Proposed Approach

In this chapter, we explain the proposed approach to build the behavior models and perform What-If analysis. We perform What-If analysis on business and infrastructure changes, specifically the following four scenarios: (i) change in workload, (ii) change in workload composition, (iii) vertical scaling and (iv) horizontal scaling.

Here are some brief definitions:

Business Changes: Changes that are specific to the business. Such as a change in workload.

Infrastructure Changes: Changes to the infrastructure. Such as adding/removing CPUs.

Horizontal Scaling: Scaling by adding more components into your pool of resources.

Vertical Scaling: Scaling by adding more power/resources (Eg. CPU, RAM, etc.) to an existing machine.

3.1 Building Behavior Models

This part of the engine deals with creating models between the various metrics of the system. The models formed here basically tell us how a particular metric varies with respect to another. The coefficients in these equations tell us how much a metric changes with every unit change in the corresponding metric it depends on. These models would then be used to make predictions based on the input scenario. The algorithm is as follows.

1. Traverse the given topology and for every edge connecting component1 to component2:
   1.1. Infer from trace if any metrics of component2 (M) depends on any metric of component1 (N)
   1.2. If yes, add the metric N to the list of metrics M depends on.
2. For every metric, fit a regression model with the metrics it depends on. Store the coefficients.

Algorithm to build behavior models

Here is an example to illustrate this.

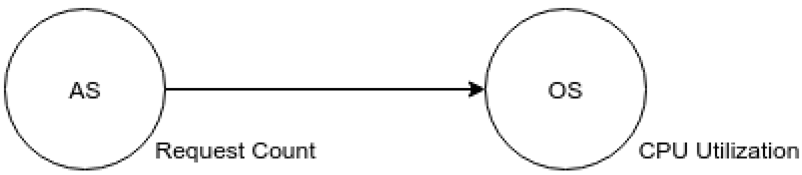

Fig 3.1: Example Topology

Consider the topology in Figure 3.1 – an application server hosted on an OS. The trace has an entry which says CPU's utilization depends on the request count of the application hosted on it. For simplicity, assume there are no other metrics in the entire system. The only edge in the topology will be detected in step 1. From the trace we get

AS request count as N and OS CPU utilization as M (Steps 1.1 and 1.2). We then get this equation from step 2:

*OS CPU utilization = m * AS request count + c.*

Where m and c are the coefficient and constant respectively.

If the composition of the workload is provided, each equation with a workload metric (such as request count) gets split into N equations where N is the number of different types of workloads. Section 3.2.2 talks more about this.

3.2 Performing Analysis of What-If Scenarios

Once the models have been built, we are ready for the What-If scenario. We perform what-if analysis for the following four scenarios namely change in workload, change in workload composition, vertical scaling and horizontal scaling.

3.2.1 Change in workload

The algorithm to compute the effect of change in workload is as follows.

1. Apply the change in the workload metric M to the metric's column in the data.
2. Add M to the evaluation_queue.
3. While the evaluation_queue is not empty:
   - 3.1. Pop metric M from the front of the queue.
   - 3.2. Scan the model list and for every model with M on the RHS, apply the model and change the data of LHS metric and add the LHS metric to the evaluation_queue.

Algorithm to compute effect of change in workload

3.2.2 Change in workload composition

The algorithm to compute the effect of change in workload composition is similar to the previous algorithm. We can model each type of workload as a separate metric of the node and follow the same algorithm as the previous one.

While creating the behavior models, we essentially modeled how much a unit change in one metric would affect another metric. Similarly, over here, we need to figure out how much a unit change in one particular component of the workload would affect the rest of the system. We can see the similarity between the two. Hence, we can model each type of workload as a separate metric of the node and follow the same algorithm as the one to compute a change in workload.

3.2.3 Vertical Scaling

The algorithm to compute the effect of change in workload composition is similar to the previous algorithms. If the static information regarding the component that needs to be vertically scaled is available, such as processor model, RAM, etc, then the input will contain the new value of that variable and what resource metrics would be affected. If not, then we change the resource metrics ourselves based on the scale factor. Once these metrics are changed, we add these metrics to the evaluation_queue of the workload change algorithm and carry out the same algorithm.

3.2.4 Horizontal Scaling

The algorithm to compute the effect of horizontal scaling is as follows.

1. Find all siblings of new component which are of the same type.
2. Find average workload of the siblings and assign that much workload to the new node.

3. Remove workload from siblings equivalent to the amount of workload added to the new component such that the ratio of workloads among the siblings stays the same.

Algorithm to compute effect of horizontal scale

16

Chapter 4

Experimental Results

In this chapter, we show the results of experiments that we obtained by applying the proposed algorithms for the different types of What-If scenarios. We demonstrate the effectiveness of our algorithms through a real world case study. We performed experiments on data of a major video game company in the US.

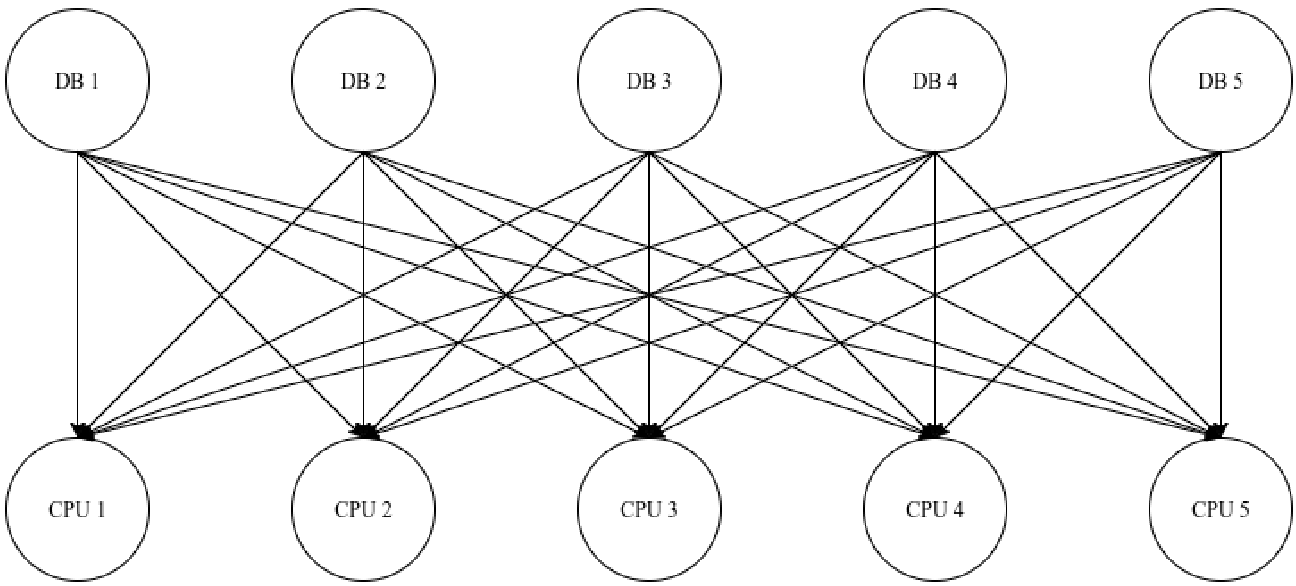

Fig 4.1: Topology of the given system

The topology of the data is given in Figure 4.1. There are 5 databases running on 5 servers. All databases are connected to all servers.

The metrics are usercalls for databases and utilization and interconnect for CPUs.

There are 3 types of workloads (We call them A, B and C) which fed to all the databases.

We now take a look at different scenarios.

4.1 Workload Change

Scenario

The overall workload on database 1 is doubled.

Impact

The impact of this change is shown in the graphs of Figure 4.2.

We can see that CPU1 and CPU 2 have a high probability of getting saturated and CPU5 has a slight probability of getting saturated. We can also infer from this output that CPUs 1 and 2 take a higher share of the workload from DB1 as compared to the other CPUs. Thus, giving us some insight on how the workload from the DBs is distributed among the CPUs.

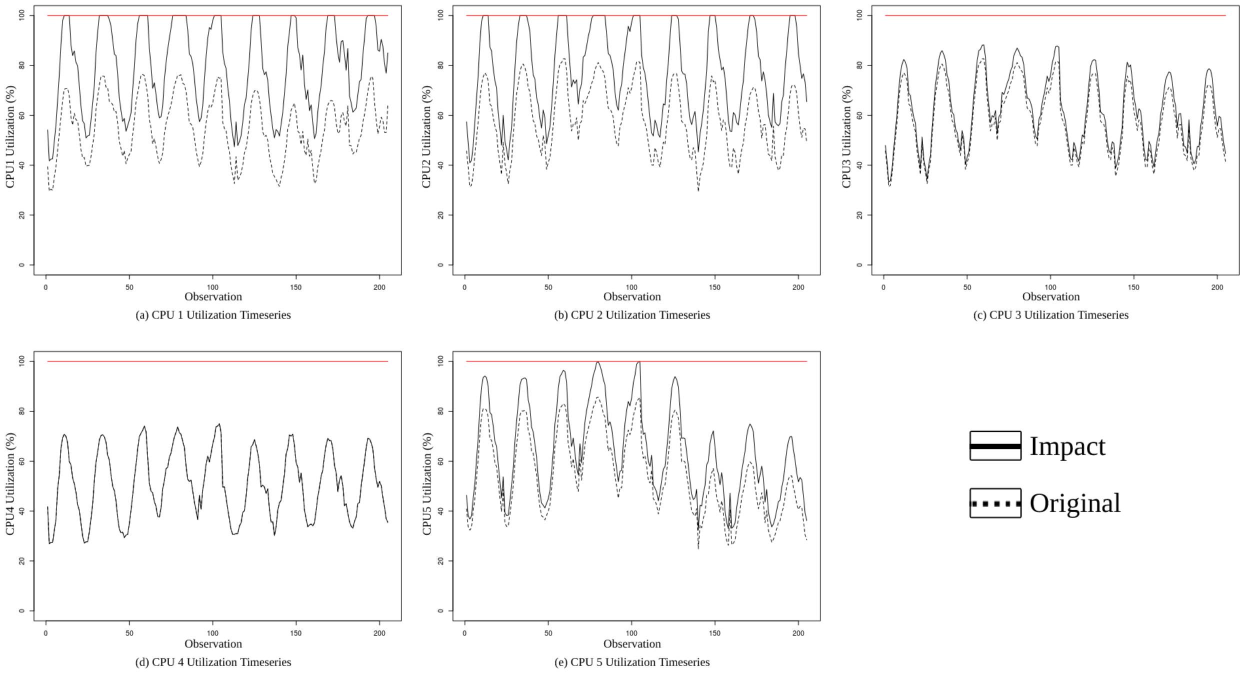

Fig 4.2: Impact of Workload Change on CPU Utilization

4.2 Workload Composition Change

Scenario

The graph in Figure 4.3 shows the workload composition of DB1.

Workload A on DB1 is doubled.

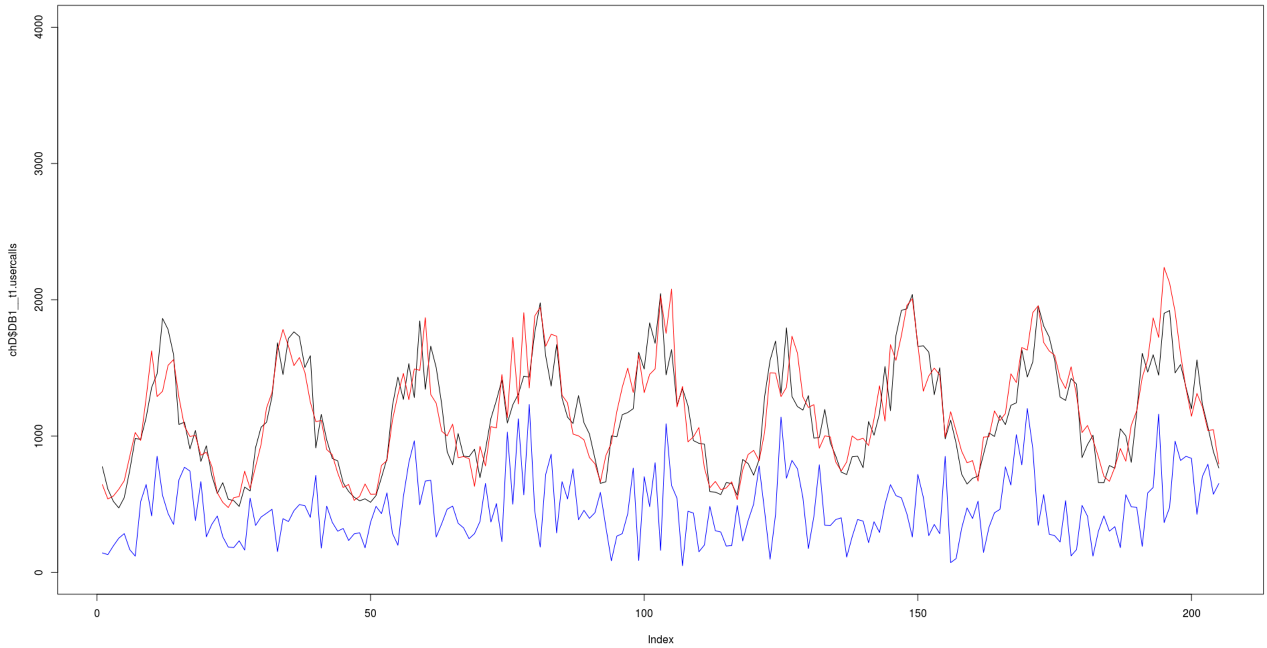

Fig 4.3: Composition of Workloads on DB1: workload A (Black), B (Red) and C (Blue)

Impact

The impact of the change is shown in the graphs of Figure 4.4.

From this we can infer that CPU2 has a high chance of getting saturated and CPU1 has a slight chance. We can also infer from the graphs that 'Workload A' on DB1 relies more on CPU2.

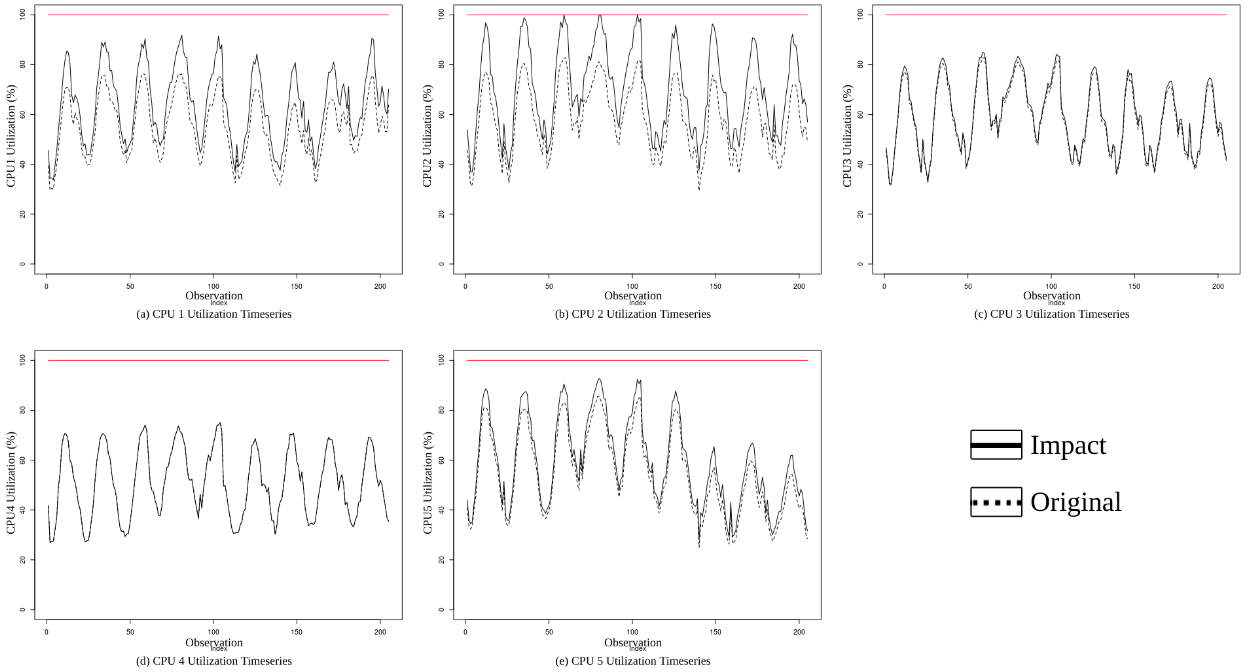

Fig 4.4: Impact of 'Workload A' Change on CPU Utilization

4.3 Vertical Scale

Scenario

CPU2 is vertically scaled to have twice the performance.

Impact

The impact is shown in the graphs of Figure 4.5.

We can see that since CPU2 has been vertically scaled to have twice the performance, the load it bears has been reduced significantly.

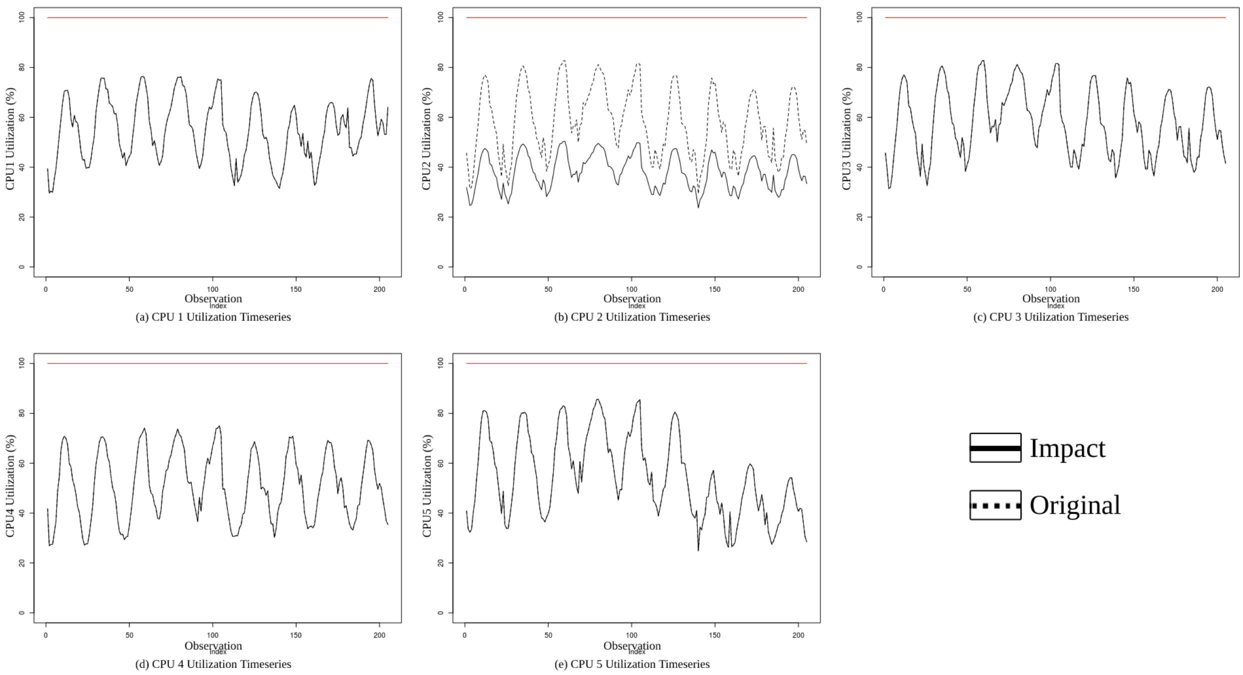

Fig 4.5: Impact of vertical scale on CPU utilizations

4.4 Horizontal Scale

Scenario

Horizontal scaling is done by adding one CPU and then two CPUs.

Impact

The impact is shown in the graphs of figure 4.6.

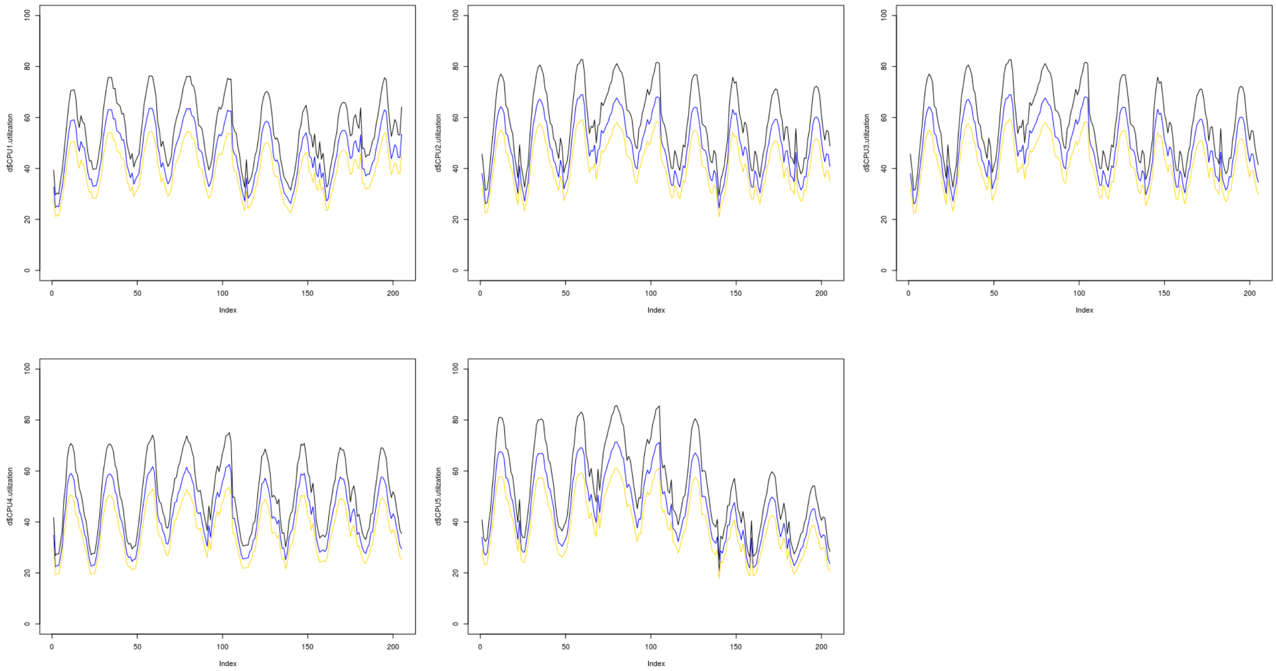

Fig 4.6: Impact of adding one CPU (blue) and two new CPUs (yellow)

We can see that adding new CPUs decreases the load on each of the other CPUs as the newly added CPUs take part of the workload and the workload is distributed across more CPUs.

4.5 Combination

Scenario

Workloads on DB1 and DB3 are doubled and two new CPUs are added.

Impact

The graphs in Figure 4.7 show impact of only the workload changes.

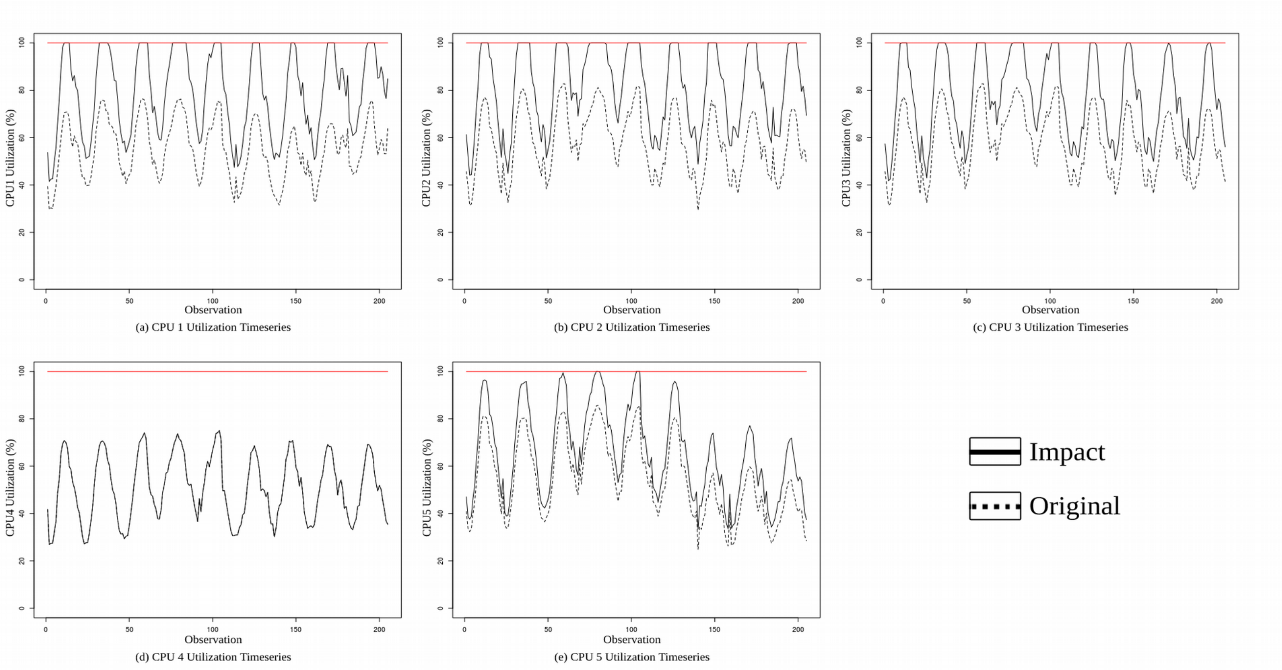

Fig 4.7: Impact of only doubling workloads on DB1 and DB3

We can see that CPU1, CPU2, CPU3 and CPU5 have a high chance of saturating.

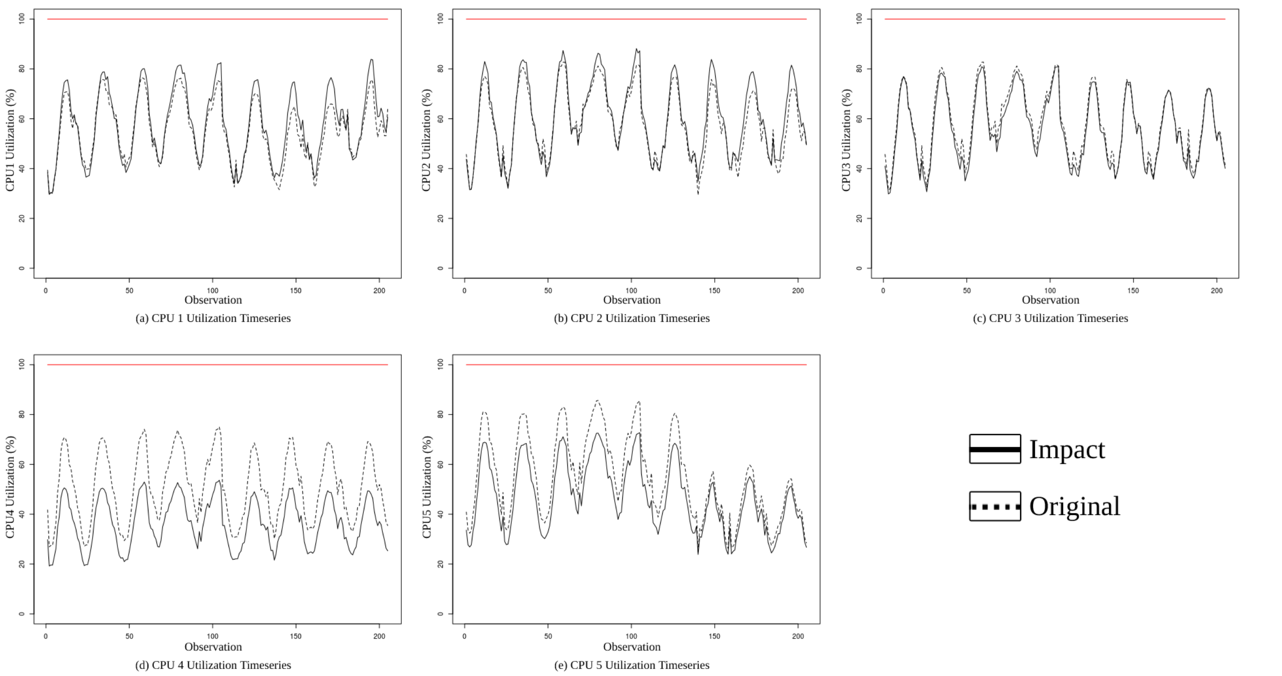

Fig 4.8: Impact of doubling workloads on DB1 and DB3 and adding two new CPUs

Now, the graphs in Figure 4.8 show the impact after adding the two CPUs as well.

Adding the two new CPUs as well has brought the utilizations of most of the CPUs back down almost to the original state.

Chapter 5

Conclusion and Future Work

In this project, we addressed the problem of What-If and analysis on transactional systems.

We developed a basic, generic methodology or engine to perform What-If analysis. In that, we can now predict more variables and provide more insight to the system designers than existing work in the area.

We were able to identify impact relationships, obtain mathematical relationships between various metrics and derive hidden relations which are further used to perform What-If analysis.

We performed What-If analysis for four types of scenarios: (i) Change in workload, (ii) Change in workload composition, (iii) Vertical scale and (iv) Horizontal scale. We performed experiments on synthetic and real world data using our algorithms.

Finally, a working prototype to demonstrate end-to-end execution of the change-impact analysis was developed.

We aim to do further work on the following:-

- Explore Feature Selection Techniques
- Explore Non-linear Regression Models
- Explore Queuing Models
- Support components such as Load Balancers, Fail-overs, Hypervisors, etc.
- Model relationships between virtual and physical components

References

[1] Terence Kelly, Hewlett-Packard Laboratories. Detecting Performance Anomalies in Global Applications. In WORLDS 2005 Proceedings of the 2nd conference on Real, Large Distributed Systems - Volume 2

[2] Abhishek Sharma, Ranjita Bhagwan, Monojit Choudhury, Leana Golubchik, Ramesh Govindan and Geoffrey M. Voelker. Automatic Request Categorization in Internet Services. In Newsletter ACM SIGMETRICS Performance Evaluation Review, Volume 36 Issue 2, September 2008.

[3] Eno Thereska, Michael Abd-El-Malek, Jay J. Wylie, Dushyanth Narayanan, and Gregory R. Ganger. Informed data distribution selection in a self-predicting storage system. In Proceedings of the 3rd International Conference on Autonomic Computing, ICAC 2006, Dublin, Ireland, 13-16 June 2006, pages 187–198, 2006.

[4] Mukarram Tariq, Amgad Zeitoun, Vytautas Valancius, Nick Feamster, and Mostafa Ammar. Answering what-if deployment and configuration questions with wise. SIGCOMM Comput. Commun. Rev., 38(4), August 2008.

[5] Eno Thereska and Gregory R. Ganger. Ironmodel: robust performance models in the wild. In ACM SIGMETRICS Conference on Measurement and Modeling

of Computer Systems (Annapolis, MD, 02,Äì 06 June 2008, pages 253–264. ACM, 2008.

[6] Rahul Singh, Prashant Shenoy, Maitreya Natu, Vaishali Sadaphal, and Harrick Vin. Analytical modeling for what-if analysis in complex cloud computing applications. SIGMETRICS Perform. Eval. Rev., 40(4):53–62, April 2013.

[7] Rahul Singh, Prashant Shenoy, Maitreya Natu, Vaishali Sadaphal, and Harrick Vin. Predico: A system for what-if analysis in complex data center applications. In Fabio Kon and Anne-Marie Kermarrec, editors, Middleware 2011, volume 7049 of Lecture Notes in Computer Science, pages 123–142. Springer Berlin Heidelberg, 2011.

[8] Aminikhanghahi, Samaneh, and Diane J. Cook. “A Survey of Methods for Time Series Change Point Detection.” *Knowledge and information systems* 51.2 (2017): 339–367. *PMC*. Web. 24 Nov. 2017.

Appendix

Before I started work on What-If analysis, I had to choose my topic for the BTP. The other problem spaces I had explored prior to What-If for about 5 weeks are as follows.

Conversational Interface

Deals with text and natural language processing.

Navigating the menus in their product – Ignio, is very time consuming as there are just so many options. Therefore to make this process easier and quicker, Digitate is working on a conversational interface through which a user can communicate with the product to get what he/she wants rather than having to navigate the menus and waste a lot of time.

I worked on transforming English sentences to SQL queries. I was able to build a very basic prototype that handled very basic queries.

Analysis of Timeseries of Graphs

Deals with network graphs (nodes and edges) and timeseries analysis.

We modeled batch jobs and their dependencies as a graph (nodes and edges). These graphs change with time, as the jobs as well as dependencies change with time. This essentially gives us a timeseries of graphs.

I used graph similarity algorithms to find similarity between graphs and built a ‘similarity graph’ where each node is a graph and an edge between two nodes means that their similarity score was above a certain threshold. I then used graph clustering techniques to find clusters of similar graphs and then fit labels on these clusters to try and find day of week/month patterns.

I had 6-months data of batch jobs was able to get some interesting results in a short span of time. It was a difficult decision choosing between this and what-if analysis as the problem statement for my BTP.