Politechnika Wrocławska

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

On-line Bayesian Context Change Detection in Web Service Systems

Maciej Zięba, Jakub M. Tomczak

HotTopiCS 2013

maciej.zieba@spwr.wroc.pl

Prague, 20.04.2013

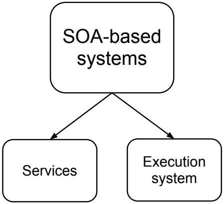
Agenda

1. Problem background
2. Change detection problem statement
3. Approaches for solving the problem
4. Bayesian Model
5. Algorithm for change detection
6. Simulation environment
7. Results and discussion

Problem background

SOA-based systems

- **SOA-based systems**: systems that implement Service Oriented Architecture, i.e., an architectural style whose goal is to achieve loose coupling among interacting software components called *services*.

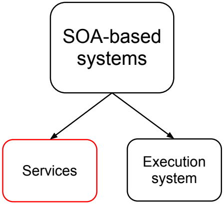


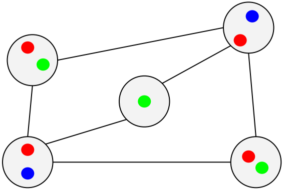
Problem background

SOA-based systems

- **SOA-based systems**: systems that implement Service Oriented Architecture, i.e., an architectural style whose goal is to achieve loose coupling among interacting software components called *services*.

- **Services**: self-describing, stateless, modular applications that are distributed across the Web and which provide functionalities and are described by quality attributes (QoS).



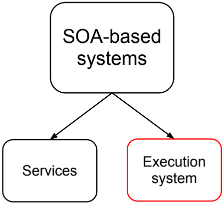


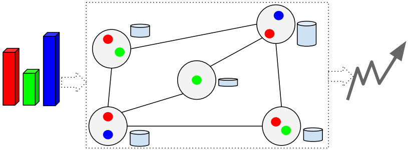
Problem background

SOA-based systems

- **SOA-based systems**: systems that implement Service Oriented Architecture, i.e., an architectural style whose goal is to achieve loose coupling among interacting software components called *services*.

- **Execution system**:
  - network of virtual machines;
  - distributed computational resources;
  - *input*: streams of requests;
  - *output*: system performance, e.g., latency.





Problem background

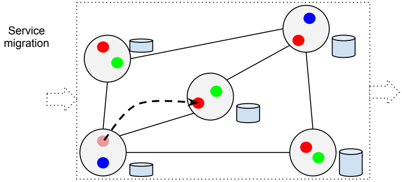
Resource allocation problem

- In order to maintain the performance of an execution system at a satisfactory (or given) level the following decisions are mainly be made:

Problem background

Resource allocation problem

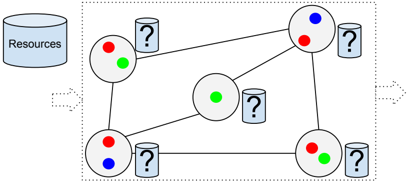
- In order to maintain the performance of an execution system at a satisfactory (or given) level the following decisions are mainly be made:
  - migration of services;



Problem background

Resource allocation problem

- In order to maintain the performance of an execution system at a satisfactory (or given) level the following decisions are mainly be made:
  - migration of services;
  - computational resources allocation.



Problem background

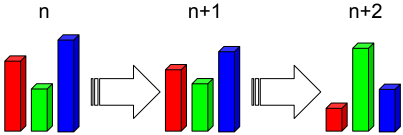
Resource re-allocation

- The execution system evolves in time because of:

Problem background

Resource re-allocation

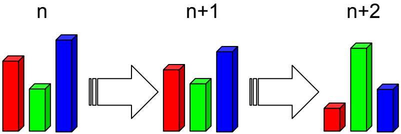
- The execution system evolves in time because of:
  - ▷ non-stationary streams of requests;



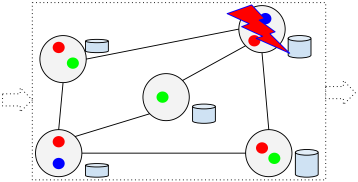
Problem background

Resource re-allocation

- The execution system evolves in time because of:
  - ▷ non-stationary streams of requests;
  - ▷ failures or system's modifications.



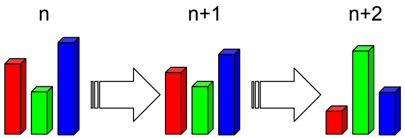
Failure

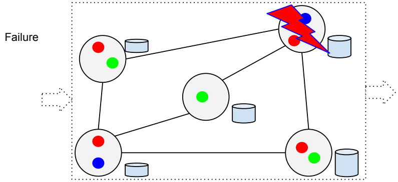


Problem background

Resource re-allocation

- The execution system evolves in time because of:
  - ▷ non-stationary streams of requests;
  - ▷ failures or system's modifications.



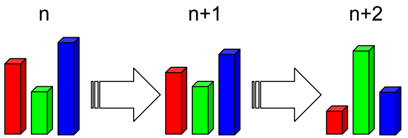


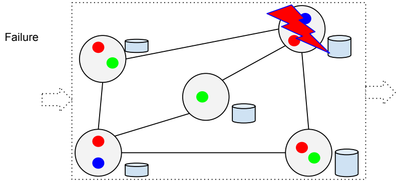
- Hence, there is a need to propose an **adaptive approach** for resource allocation.

Problem background

Resource re-allocation

- The execution system evolves in time because of:
  - ▷ non-stationary streams of requests;
  - ▷ failures or system's modifications.



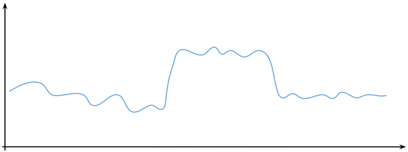


- Hence, there is a need to propose an **adaptive approach** for resource allocation.
- Resource re-allocation: if a *change* in the **input** (or **output**) is reported, then calculate new resource allocation.

Change detection problem

Overview

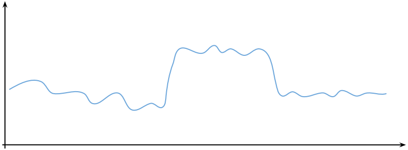
- **Change detection**: identifies changes in the probability distribution of a stochastic process.



Change detection problem

Overview

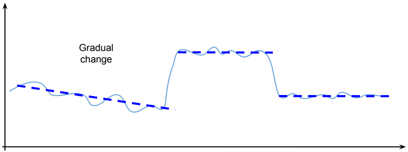
- **Change detection**: identifies changes in the probability distribution of a stochastic process.
- Two kinds of changes:



Change detection problem

Overview

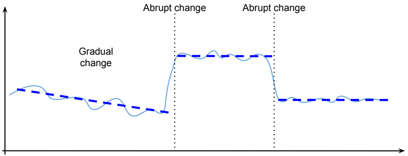
- **Change detection**: identifies changes in the probability distribution of a stochastic process.
- Two kinds of changes:
  - gradual;



Change detection problem

Overview

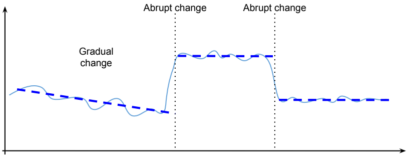
- **Change detection**: identifies changes in the probability distribution of a stochastic process.
- Two kinds of changes:
  - gradual;
  - abrupt.



Change detection problem

Overview

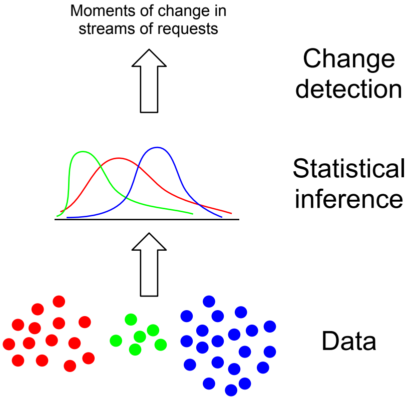
- **Change detection**: identifies changes in the probability distribution of a stochastic process.
- Two kinds of changes:
  - gradual;
  - abrupt.



Change detection problem

Overview

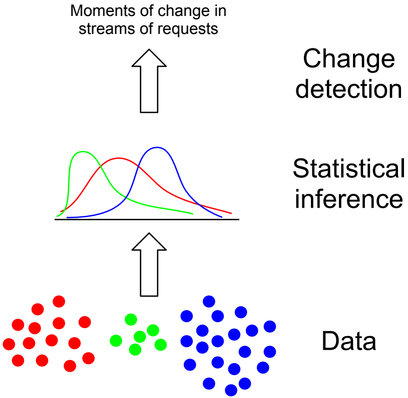
- **Change detection**: identifies changes in the probability distribution of a stochastic process.
- Two kinds of changes:
  - gradual;
  - abrupt.
- Statistical change detection:
  - **frequentist approach**: distribution estimation and comparison using dissimilarity measures;
  - **Bayesian approach**: all quantities are random variables.



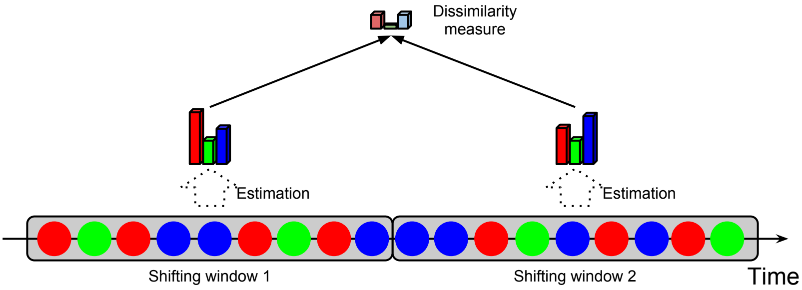
Change detection problem

Overview

- **Change detection**: identifies changes in the probability distribution of a stochastic process.
- Two kinds of changes:
  - gradual;
  - abrupt.
- Statistical change detection:
  - **frequentist approach**: distribution estimation and comparison using dissimilarity measures;
  - **Bayesian approach**: all quantities are random variables.



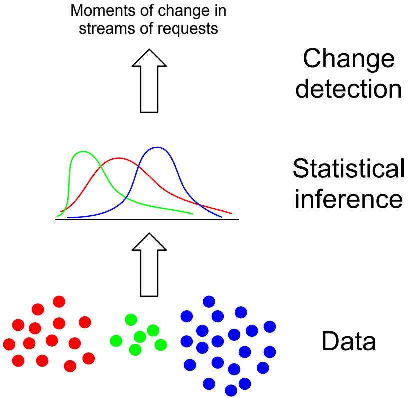
Frequentist approach



Change detection problem

Overview

- **Change detection**: tries to identify changes in the probability distribution of a stochastic process.
- Two kinds of changes:
  - gradual;
  - abrupt.
- Statistical change detection:
  - **frequentist approach**: distribution estimation and comparison using dissimilarity measures;
  - **Bayesian approach**: all quantities are random variables.



Bayesian approach

Likelihood functions for models

Assume that $\mathcal{D}_n^L = \{\mathbf{x}_{n-L+1}, \ldots, \mathbf{x}_n\}$ are examples within shifting window of size L.

Bayesian approach

Likelihood functions for models

Assume that $\mathcal{D}_n^L = \{\mathbf{x}_{n-L+1}, \ldots, \mathbf{x}_n\}$ are examples within shifting window of size L.

1. If there is no context change in $\mathcal{D}_n^L$, then we say that data are generated from a model $\mathcal{M}_0$ and its likelihood function is as follows

$$p(\mathcal{D}_n^L | \mathcal{M}_0, \boldsymbol{\theta}_0) = p(\mathcal{D}_n^L | \boldsymbol{\theta}_0) \tag{1}$$

where $\boldsymbol{\theta}_0$ – parameters of $\mathcal{M}_0$.

Bayesian approach

Likelihood functions for models

Assume that $\mathcal{D}_n^L = \{\mathbf{x}_{n-L+1}, \ldots, \mathbf{x}_n\}$ are examples within shifting window of size L.

1. If there is no context change in $\mathcal{D}_n^L$, then we say that data are generated from a model $\mathcal{M}_0$ and its likelihood function is as follows

$$p(\mathcal{D}_n^L | \mathcal{M}_0, \boldsymbol{\theta}_0) = p(\mathcal{D}_n^L | \boldsymbol{\theta}_0) \tag{1}$$

   where $\boldsymbol{\theta}_0$ – parameters of $\mathcal{M}_0$.

2. If there is one context change in $\mathcal{D}_n^L$ at $t < n$, then we say that data are generated from a model $\mathcal{M}_1$ and its likelihood function is as follows

$$p(\mathcal{D}_n^L | \mathcal{M}_1, \boldsymbol{\theta}_1, t) = p(\mathcal{D}_t^{L-n+t} | \boldsymbol{\theta}_1^1)\ p(\mathcal{D}_n^{n-t} | \boldsymbol{\theta}_1^2) \tag{2}$$

   where $\boldsymbol{\theta}_1 = (\boldsymbol{\theta}_1^1\ \boldsymbol{\theta}_1^2)^{\mathrm{T}}$ – parameters of $\mathcal{M}_1$, $\boldsymbol{\theta}_1^1$ are parameters for partition before context change, and $\boldsymbol{\theta}_1^2$ – parameters after context change.

Bayesian approach

Model evidence

In order to select one model which is *more probable* to generate observed data we need to calculate *model evidences*. The model evidence of $\mathcal{M}_0$ can be calculated as follows

Bayesian approach

Model evidence

In order to select one model which is *more probable* to generate observed data we need to calculate *model evidences*. The model evidence of $\mathcal{M}_0$ can be calculated as follows

$$p(\mathcal{D}_n^L|\mathcal{M}_0) = \int p(\mathcal{D}_n^L|\mathcal{M}_0, \boldsymbol{\theta}_0)\ p(\boldsymbol{\theta}_0|\mathcal{M}_0)\ \mathrm{d}\boldsymbol{\theta}_0, \tag{3}$$

where $p(\boldsymbol{\theta}_0|\mathcal{M}_0)$ – *a priori* probability distribution of parameters.

Bayesian approach

Model evidence

In order to select one model which is *more probable* to generate observed data we need to calculate *model evidences*. The model evidence of $\mathcal{M}_0$ can be calculated as follows

$$p(\mathcal{D}_n^L|\mathcal{M}_0) = \int p(\mathcal{D}_n^L|\mathcal{M}_0, \boldsymbol{\theta}_0)\ p(\boldsymbol{\theta}_0|\mathcal{M}_0)\ \mathrm{d}\boldsymbol{\theta}_0, \tag{3}$$

where $p(\boldsymbol{\theta}_0|\mathcal{M}_0)$ – *a priori* probability distribution of parameters. Next, the model evidence of $\mathcal{M}_1$ is the following (using the independence of $\boldsymbol{\theta}_1^1, \boldsymbol{\theta}_1^2, t$)

$$\begin{aligned} p(\mathcal{D}_n^L|\mathcal{M}_1) = \iint p(\mathcal{D}_n^L|\mathcal{M}_1, \boldsymbol{\theta}_1, t)\ p(\boldsymbol{\theta}_1^1|\mathcal{M}_1)\times \\ \times\ p(\boldsymbol{\theta}_1^2|\mathcal{M}_1)\ p(t|\mathcal{M}_1)\ \mathrm{d}\boldsymbol{\theta}_1\ \mathrm{d}t, \end{aligned} \tag{4}$$

where $p(\boldsymbol{\theta}_1^1|\mathcal{M}_1)$, $p(\boldsymbol{\theta}_1^2|\mathcal{M}_1)$, $p(t|\mathcal{M}_1)$ – *a priori* probability distributions of parameters.

Bayesian approach

Model evidence approximation

In order to calculate model evidences of $\mathcal{M}_0$ and $\mathcal{M}_1$ we make the following assumptions:

Bayesian approach

Model evidence approximation

In order to calculate model evidences of $\mathcal{M}_0$ and $\mathcal{M}_1$ we make the following assumptions:

- the *a priori* probability distributions of $\boldsymbol{\theta}_0$ and $\boldsymbol{\theta}_1$ are taken to be non-informative;

Bayesian approach

Model evidence approximation

In order to calculate model evidences of $\mathcal{M}_0$ and $\mathcal{M}_1$ we make the following assumptions:

- the *a priori* probability distributions of $\boldsymbol{\theta}_0$ and $\boldsymbol{\theta}_1$ are taken to be non-informative;
- the context change occurs in the middle of the shifting window, i.e., $n - \lceil \frac{1}{2} L \rceil$, hence the *a priori* probability distribution of t is a Dirac delta function in the point $n - \lceil \frac{1}{2} L \rceil$.

Bayesian approach

Model evidence approximation

In order to calculate model evidences of $\mathcal{M}_0$ and $\mathcal{M}_1$ we make the following assumptions:

- the *a priori* probability distributions of $\boldsymbol{\theta}_0$ and $\boldsymbol{\theta}_1$ are taken to be non-informative;
- the context change occurs in the middle of the shifting window, i.e., $n - \lceil \frac{1}{2}L \rceil$, hence the *a priori* probability distribution of t is a Dirac delta function in the point $n - \lceil \frac{1}{2}L \rceil$.

For such assumptions we can approximate the model evidence by the Bayesian Information Criterion (BIC)

$$\ln p(\mathcal{D}_n^L|\mathcal{M}) \approx \ln p(\mathcal{D}_n^L|\hat{\boldsymbol{\theta}}) - \frac{K}{2}\ln L, \tag{5}$$

where $\hat{\boldsymbol{\theta}}$ is the maximum likelihood estimator of $\boldsymbol{\theta}$.

Bayesian approach

Bayes factor

To compare both models, we calculate the Bayes factor (assuming equal probabilities over models):

$$B_{10} = \frac{p(\mathcal{D}_n^L|\mathcal{M}_1)}{p(\mathcal{D}_n^L|\mathcal{M}_0)}. \tag{6}$$

B_{10}	$\ln(B_{10})$	Evidence in favor of $\mathcal{M}_1$
$1-3$	$0-1.1$	Weak
$3-10$	$1.1-2.3$	Substantial
$10-100$	$2.3-4.6$	Strong
>100	>4.6	Decisive

Algorithm description

Algorithm 1: Change detection using approximated Bayes factor

Input : $\mathcal{D}$, L, $\mathcal{M}_0$, $\mathcal{M}_1$

Output: Moments of context change $\tau_1, \ldots, \tau_M$

$n \longleftarrow 1$, $m \longleftarrow 0$, $\tau_0 \longleftarrow 0$;

while $n < \text{card}\{\mathcal{D}\}$ **do**

  Calculate $\ln p(\mathcal{D}_n^L|\mathcal{M}_0)$ and $\ln p(\mathcal{D}_n^L|\mathcal{M}_1)$;

  Calculate $\ln B_{10}$;

  if $\ln B_{10} > \sigma$ **then**

    if $\big((n - \lceil L/2 \rceil) - \tau_m\big) \geqslant \lceil L/2 \rceil$ **then**

      $m := m + 1$;

      $\tau_m \longleftarrow n - \lceil L/2 \rceil$;

    end

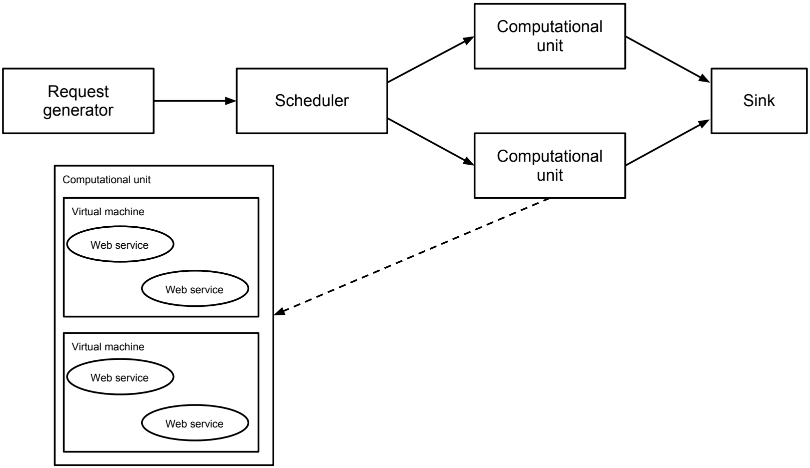
  end

  $n := n + 1$;

end

Simulator

Structure



Simulator

Details

- Streams of requests are generated with **Poisson process**.

Simulator

Details

- Streams of requests are generated with **Poisson process**.
- Computational nodes are represented by web servers with **processors** as computational resources.

Simulator

Details

- Streams of requests are generated with **Poisson process**.
- Computational nodes are represented by web servers with **processors** as computational resources.
- **Two virtual machines** are situated on each of servers.

Simulator

Details

- Streams of requests are generated with **Poisson process**.
- Computational nodes are represented by web servers with **processors** as computational resources.
- **Two virtual machines** are situated on each of servers.
- Each of two web servers in the model uses 8 **processors**, which are assigned to virtual machines in following way:

Simulator

Details

- Streams of requests are generated with **Poisson process**.
- Computational nodes are represented by web servers with **processors** as computational resources.
- **Two virtual machines** are situated on each of servers.
- Each of two web servers in the model uses 8 **processors**, which are assigned to virtual machines in following way:
  - **6** and **2 processors** are respectively used by **first** and **second virtual machine** (first server).

Simulator

Details

- Streams of requests are generated with **Poisson process**.
- Computational nodes are represented by web servers with **processors** as computational resources.
- **Two virtual machines** are situated on each of servers.
- Each of two web servers in the model uses 8 **processors**, which are assigned to virtual machines in following way:
  - **6** and **2 processors** are respectively used by **first** and **second virtual machine** (first server).
  - **4** and **4 processors** are respectively used by **first** and **second virtual machine** (second server).

Simulator
Details

- Streams of requests are generated with **Poisson process**.
- Computational nodes are represented by web servers with **processors** as computational resources.
- **Two virtual machines** are situated on each of servers.
- Each of two web servers in the model uses 8 **processors**, which are assigned to virtual machines in following way:
  - **6** and **2 processors** are respectively used by **first** and **second virtual machine** (first server).
  - **4** and **4 processors** are respectively used by **first** and **second virtual machine** (second server).
- Processing delays for web servers are equal **0.0004 seconds** and for virtual machines are equal **0.0008 seconds**

According to the technical report: ***Lite Technologies, Web server performance comparison: Litespeed 2.0 vs.***.

Simulator

Modelling Web services

- Performance of real data processing services implemented in **PlaTel** was modelled: ***Naive Bayes***, ***Logistic Regression***, ***J48*** and ***Multilayer Perceptron***.

Simulator

Modelling Web services

- Performance of real data processing services implemented in **PlaTel** was modelled: ***Naive Bayes***, ***Logistic Regression***, ***J48*** and ***Multilayer Perceptron***.
- Processing time for each of selected services was modelled with **triangular distribution**.

Simulator

Modelling Web services

- Performance of real data processing services implemented in **PlaTel** was modelled: ***Naive Bayes***, ***Logistic Regression***, ***J48*** and ***Multilayer Perceptron***.
- Processing time for each of selected services was modelled with **triangular distribution**.
- The values parameters for distributions (minimum, maximum and average value) for each of services were estimated using **soapUI tool**.

Simulator

Modelling Web services

- Performance of real data processing services implemented in **PlaTel** was modelled: ***Naive Bayes***, ***Logistic Regression***, ***J48*** and ***Multilayer Perceptron***.
- Processing time for each of selected services was modelled with **triangular distribution**.
- The values parameters for distributions (minimum, maximum and average value) for each of services were estimated using **soapUI tool**.
- Following resource allocation of web services were proposed:

Simulator

Modelling Web services

- Performance of real data processing services implemented in **PlaTel** was modelled: ***Naive Bayes***, ***Logistic Regression***, ***J48*** and ***Multilayer Perceptron***.
- Processing time for each of selected services was modelled with **triangular distribution**.
- The values parameters for distributions (minimum, maximum and average value) for each of services were estimated using **soapUI tool**.
- Following resource allocation of web services were proposed:
  - ***Multilayer Perceptron*** - total number of **10 processors**.

Simulator

Modelling Web services

- Performance of real data processing services implemented in **PlaTel** was modelled: ***Naive Bayes***, ***Logistic Regression***, ***J48*** and ***Multilayer Perceptron***.
- Processing time for each of selected services was modelled with **triangular distribution**.
- The values parameters for distributions (minimum, maximum and average value) for each of services were estimated using **soapUI tool**.
- Following resource allocation of web services were proposed:
  - ***Multilayer Perceptron*** - total number of **10 processors**.
  - ***Logistic Regression*** total number of **10 processors**.

Simulator

Modelling Web services

- Performance of real data processing services implemented in **PlaTel** was modelled: ***Naive Bayes***, ***Logistic Regression***, ***J48*** and ***Multilayer Perceptron***.
- Processing time for each of selected services was modelled with **triangular distribution**.
- The values parameters for distributions (minimum, maximum and average value) for each of services were estimated using **soapUI tool**.
- Following resource allocation of web services were proposed:
  - ***Multilayer Perceptron*** - total number of **10 processors**.
  - ***Logistic Regression*** total number of **10 processors**.
  - ***J48*** total number of **6 processors**.

Simulator

Modelling Web services

- Performance of real data processing services implemented in **PlaTel** was modelled: ***Naive Bayes***, ***Logistic Regression***, ***J48*** and ***Multilayer Perceptron***.
- Processing time for each of selected services was modelled with **triangular distribution**.
- The values parameters for distributions (minimum, maximum and average value) for each of services were estimated using **soapUI tool**.
- Following resource allocation of web services were proposed:
  - ***Multilayer Perceptron*** - total number of **10 processors**.
  - ***Logistic Regression*** total number of **10 processors**.
  - ***J48*** total number of **6 processors**.
  - ***Naive Bayes*** total number of **4 processors**.

Experiment

Description

- The aim of the experiment is to compare the performance of frequentist approaches with Bayesian approach for change detection problem.

Experiment

Description

- The aim of the experiment is to compare the performance of frequentist approaches with Bayesian approach for change detection problem.
- Following dissimilarity measures were considered:

Experiment

Description

- The aim of the experiment is to compare the performance of frequentist approaches with Bayesian approach for change detection problem.
- Following dissimilarity measures were considered:
  - **Bhattacharyya**

Experiment

Description

- The aim of the experiment is to compare the performance of frequentist approaches with Bayesian approach for change detection problem.
- Following dissimilarity measures were considered:
  - **Bhattacharyya**
  - **Kullback-Leibler**

Experiment

Description

- The aim of the experiment is to compare the performance of frequentist approaches with Bayesian approach for change detection problem.
- Following dissimilarity measures were considered:
  - **Bhattacharyya**
  - **Kullback-Leibler**
  - **Lin-Wong**

Experiment

Description

- The aim of the experiment is to compare the performance of frequentist approaches with Bayesian approach for change detection problem.
- Following dissimilarity measures were considered:
  - **Bhattacharyya**
  - **Kullback-Leibler**
  - **Lin-Wong**
  - **modified Lin-Wong**

Experiment

Description

- The aim of the experiment is to compare the performance of frequentist approaches with Bayesian approach for change detection problem.
- Following dissimilarity measures were considered:
  - **Bhattacharyya**
  - **Kullback-Leibler**
  - **Lin-Wong**
  - **modified Lin-Wong**
- **Average latency** in request responses was considered as a quality rate for entire system.

Experiment
Description

- The aim of the experiment is to compare the performance of frequentist approaches with Bayesian approach for change detection problem.
- Following dissimilarity measures were considered:
  - **Bhattacharyya**
  - **Kullback-Leibler**
  - **Lin-Wong**
  - **modified Lin-Wong**
- **Average latency** in request responses was considered as a quality rate for entire system.
- The simulation model was implemented in discrete events simulation environment **Arena**.

Experiment

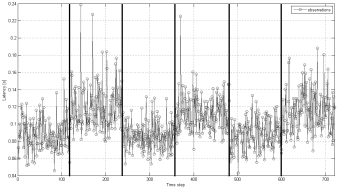
Description

- The aim of the experiment is to compare the performance of frequentist approaches with Bayesian approach for change detection problem.
- Following dissimilarity measures were considered:
  - **Bhattacharyya**
  - **Kullback-Leibler**
  - **Lin-Wong**
  - **modified Lin-Wong**
- **Average latency** in request responses was considered as a quality rate for entire system.
- The simulation model was implemented in discrete events simulation environment **Arena**.
- Algorithms for change detection were implemented in **Matlab**.

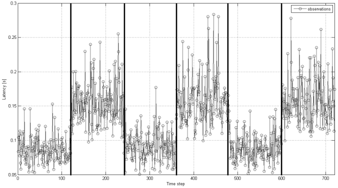
Experiment

Considered scenarios (1)

1. *Slight context change*. The context is changed periodically (5 times per simulation) and change is gained by increasing the intensity parameters of Poisson process three times.



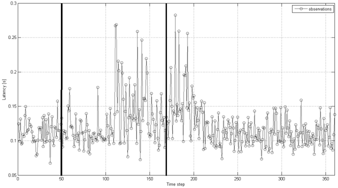
2. *Significant context change*. The context is changed periodically (5 times per simulation) and change is gained by increasing the intensity parameters of Poisson process six times.



Experiment

Considered scenarios (2)

3. *Processors failure (anomaly)*.
Anomaly is gained by failure of 4 processors on first virtual machine.



Experiment

Results for slight context change simulation

Measure	Correctly detected (max. 5)	Incorrectly detected
Bhattacharyya ($L = 25$, $\sigma = 0.2$)	3.2	0.2
Kullback-Leibler ($L = 25$, $\sigma = 1$)	3.8	0.8
Lin-Wong ($L = 25$, $\sigma = 0.15$)	2.8	0.7
mod. Lin-Wong ($L = 25$, $\sigma = 0.02$)	2.9	0.9
Bayesian approach ($L = 25$)	3	0.2

Experiment

Results for significant context change simulation

Measure	Correctly detected (max. 5)	Incorrectly detected
Bhattacharyya ($L = 25$, $\sigma = 0.2$)	4.6	0.1
Kullback-Leibler ($L = 25$, $\sigma = 1$)	4.8	0.2
Lin-Wong ($L = 25$, $\sigma = 0.15$)	4.6	0.3
mod. Lin-Wong ($L = 25$, $\sigma = 0.02$)	4.6	0.2
Bayesian approach ($L = 25$)	5	0

Experiment

Results for processors failure simulation

Measure	Correctly detected (max. 2)	Incorrectly detected
Bhattacharyya ($L = 25$, $\sigma = 0.2$)	1	0.3
Kullback-Leibler ($L = 25$, $\sigma = 1$)	0.7	0.3
Lin-Wong ($L = 25$, $\sigma = 0.15$)	1.1	0.1
mod. Lin-Wong ($L = 25$, $\sigma = 0.02$)	1	0.1
Bayesian approach ($L = 25$)	1.1	0.1

Discussion

- Most of changes were successfully detected using frequentist and Bayesian approaches.

Discussion

- Most of changes were successfully detected using frequentist and Bayesian approaches.
- The lowest number of detected changes were gained for *Slight context change* and *Processors failure* simulation scenarios.

Discussion

- Most of changes were successfully detected using frequentist and Bayesian approaches.
- The lowest number of detected changes were gained for *Slight context change* and *Processors failure* simulation scenarios.
- Bayesian approach performed slightly better for *Significant context change* and *Processors failure (anomaly)* scenarios.

Discussion

- Most of changes were successfully detected using frequentist and Bayesian approaches.
- The lowest number of detected changes were gained for *Slight context change* and *Processors failure* simulation scenarios.
- Bayesian approach performed slightly better for *Significant context change* and *Processors failure (anomaly)* scenarios.
- The number of incorrectly detected changes using Bayesian model was the lowest for all considered scenarios.

Discussion

- Most of changes were successfully detected using frequentist and Bayesian approaches.
- The lowest number of detected changes were gained for *Slight context change* and *Processors failure* simulation scenarios.
- Bayesian approach performed slightly better for *Significant context change* and *Processors failure (anomaly)* scenarios.
- The number of incorrectly detected changes using Bayesian model was the lowest for all considered scenarios.
- The best results for slight context changes were gained using Bhattacharyya measure.

Discussion

- Most of changes were successfully detected using frequentist and Bayesian approaches.
- The lowest number of detected changes were gained for *Slight context change* and *Processors failure* simulation scenarios.
- Bayesian approach performed slightly better for *Significant context change* and *Processors failure (anomaly)* scenarios.
- The number of incorrectly detected changes using Bayesian model was the lowest for all considered scenarios.
- The best results for slight context changes were gained using Bhattacharyya measure.
- Bayesian approach, in comparison to the frequentist approach, does not demand defining additional parameters beside shifting window's size.