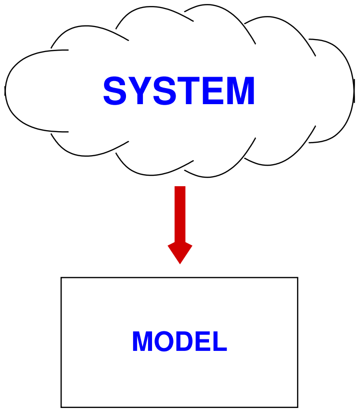
Performance Modelling — Lecture 17: Parameterisation and Workload Characterisation

Jane Hillston
School of Informatics
The University of Edinburgh
Scotland

20th March 2017

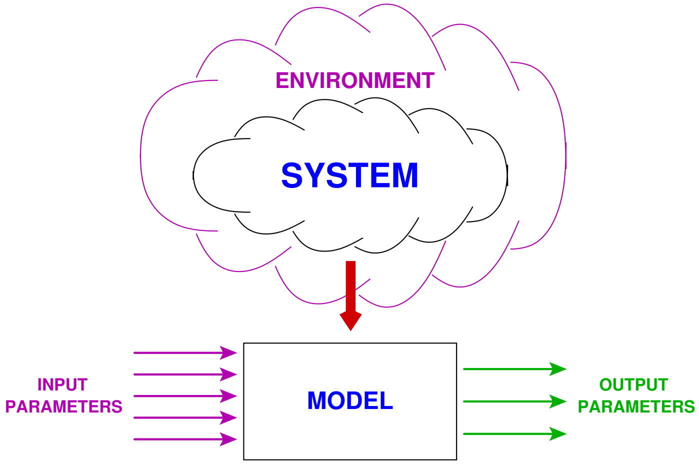
Introduction



Introduction

So far in this course we have been concentrating on constructing a good representation of the system, appropriate for the investigation we wish to carry out. However it is not just the physical system which is represented within the model but also the effect of the environment upon the system.

Introduction



Introduction

Input parameters may actually include information about the system and model control information — but the most important (and hardest to get right) is the information about workload.

Work for the system to do

Therefore it is just as important to have a good representation of the workload of the system as to have a good representation of the system itself. Indeed performance measures are not, in general, absolute for a given system: performance is predicted on the basis of a given workload.

Workload characterisation

Validation of the input parameters should be given equal importance with validation of the assumptions used within the behaviour of the model. Finding suitable and/or realistic values for input parameters is often termed workload characterisation.

Parameterisation failures

Parameterisation failures

There have been well-documented problems arising from badly parameterised models. The Hubble Space Telescope was rigorously simulated prior to its launch, as a cheaper alternative to testing.

Parameterisation failures

- However, once positioned it was found that it could not produce a sharp image because the main mirror was badly flawed.

Parameterisation failures

- However, once positioned it was found that it could not produce a sharp image because the main mirror was badly flawed.
- The error was traced to the erroneous input data used in the simulations.

Parameterisation failures

- However, once positioned it was found that it could not produce a sharp image because the main mirror was badly flawed.
- The error was traced to the erroneous input data used in the simulations.
- Some corrective optics were added to the telescope in space (at a cost much higher than ground-based testing of the mirror), but the telescope never performed as well as planned.

Parameterisation

Parameterisation is the process of assigning values to variables within a model in order to ensure that it is as accurate as feasible or appropriate.

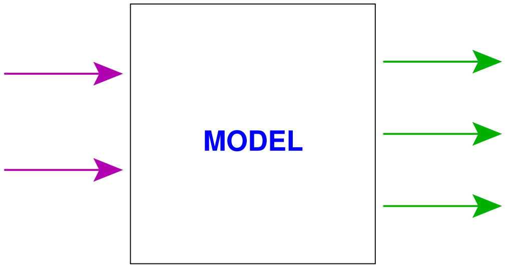
In a simulation model as well as the actual values to be used, there will be some consideration of the appropriate distribution to be used for generating the values.

Availability of data

It is important that we consider the availability of data to assist in parameterising our models from the outset of model design and construction.

Availability of data

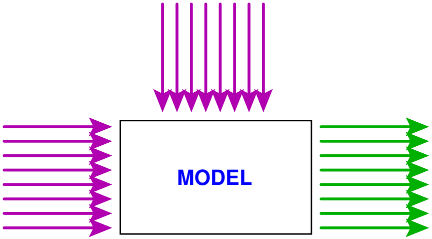
It is important that we consider the availability of data to assist in parameterising our models from the outset of model design and construction.



It would be a waste of effort to develop a very detailed model if there is not data available from which to construct a similarly detailed workload characterisation.

Approximate models

Conversely, there is no need to invest a lot of effort in sophisticated statistical analysis of workload data to parameterise a crude model which is only intended to give rough estimates.



Which parameters to include?

- The first step is to choose which parameters to include in the model. This will depend on the objectives of the study, and judgement of what is likely to affect performance.

Which parameters to include?

- The first step is to choose which parameters to include in the model. This will depend on the objectives of the study, and judgement of what is likely to affect performance.
- The workload may have many different characteristics, such as inter-arrival time of jobs, type of job, resource required per job, or size of job. However not all of these will necessarily have an impact on the performance of a system.

Which parameters to include?

- The first step is to choose which parameters to include in the model. This will depend on the objectives of the study, and judgement of what is likely to affect performance.
- The workload may have many different characteristics, such as inter-arrival time of jobs, type of job, resource required per job, or size of job. However not all of these will necessarily have an impact on the performance of a system.
- For example, when modelling a router in a communication network, if the packet size does not affect the packet forwarding time then this characteristic of the workload (packets) does not need to be represented.

Homogeneous workloads

In the models we have considered during the course we have usually chosen the parameters implicitly at the same time as choosing the appropriate level of abstraction, and we have generally assumed a homogeneous workload, i.e. that all jobs or customers within the system behave identically.

Assigning values to parameters

Assigning values to parameters has also been simple within our models as they have not been based on real systems. However in practice a considerable part of the effort of a modelling study might be devoted to choosing appropriate values for input parameters.

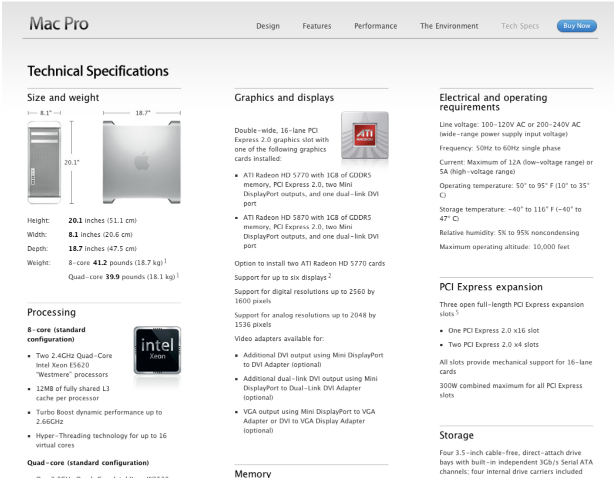
Static parameters

Most models will also contain parameters which represent aspects of the system behaviour.

Many of these will be static parameters which contain information about the system configuration.

For existing systems these are often available from published materials (manuals, specification documents, webpages etc.).

Static parameters



Technical Specifications

Size and weight

Height:	**20.1** inches (51.1 cm)
Width:	**8.1** inches (20.6 cm)
Depth:	**18.7** inches (47.5 cm)
Weight:	8-core **41.2** pounds (18.7 kg)[1]
	Quad-core **39.9** pounds (18.1 kg)[1]

Processing

8-core (standard configuration)

- Two 2.4GHz Quad-Core Intel Xeon E5620 "Westmere" processors
- 12MB of fully shared L3 cache per processor
- Turbo Boost dynamic performance up to 2.66GHz
- Hyper-Threading technology for up to 16 virtual cores

Quad-core (standard configuration)

- [illegible]

Graphics and displays

Double-wide, 16-lane PCI Express 2.0 graphics slot with one of the following graphics cards installed:

- ATI Radeon HD 5770 with 1GB of GDDR5 memory, PCI Express 2.0, two Mini DisplayPort outputs, and one dual-link DVI port
- ATI Radeon HD 5870 with 1GB of GDDR5 memory, PCI Express 2.0, two Mini DisplayPort outputs, and one dual-link DVI port

Option to install two ATI Radeon HD 5770 cards

Support for up to six displays[2]

Support for digital resolutions up to 2560 by 1600 pixels

Support for analog resolutions up to 2048 by 1536 pixels

Video adapters available for:

- Additional DVI output using Mini DisplayPort to DVI Adapter (optional)
- Additional dual-link DVI output using Mini DisplayPort to Dual-Link DVI Adapter (optional)
- VGA output using Mini DisplayPort to VGA Adapter or DVI to VGA Display Adapter (optional)

Memory

Electrical and operating requirements

Line voltage: 100-120V AC or 200-240V AC (wide-range power supply input voltage)

Frequency: 50Hz to 60Hz single phase

Current: Maximum of 12A (low-voltage range) or 5A (high-voltage range)

Operating temperature: 50° to 95° F (10° to 35° C)

Storage temperature: -40° to 116° F (-40° to 47° C)

Relative humidity: 5% to 95% noncondensing

Maximum operating altitude: 10,000 feet

PCI Express expansion

Three open full-length PCI Express expansion slots[5]

- One PCI Express 2.0 x16 slot
- Two PCI Express 2.0 x4 slots

All slots provide mechanical support for 16-lane cards

300W combined maximum for all PCI Express slots

Storage

Four 3.5-inch cable-free, direct-attach drive bays with built-in independent 3Gb/s Serial ATA channels; four internal drive carriers included

Dynamic parameters

In contrast dynamic parameters usually characterise information about the working system which must be extracted or predicted from records produced during system operation.

Dynamic parameters

In contrast dynamic parameters usually characterise information about the working system which must be extracted or predicted from records produced during system operation.

Moreover, the actual parameter used in a model may not directly match what is measured, but may need to be derived from static and dynamic measures.

Example

For example, in a PEPA model of an application program, access to memory is represented as a single activity, with time delay t_{avg}.

Example

For example, in a PEPA model of an application program, access to memory is represented as a single activity, with time delay t_{avg}.

Direct measurement of this time on a real machine, however, could be hard to achieve.

Example

For example, in a PEPA model of an application program, access to memory is represented as a single activity, with time delay t_{avg}.

Direct measurement of this time on a real machine, however, could be hard to achieve.

But we can take advantage of the expression

$$t_{avg} = ht_c + (1 - h)t_m$$

t_c is the time to access cache (from manufacturers spec)

t_m is the time to access main memory (from manufacturers spec)

h is the hit ratio must be derived from measurements.

Sources of Information

System Description

hardware configuration; operating system (and version); resource allocation and scheduling strategies; tuning parameter values.

Sources of Information

System Description

hardware configuration; operating system (and version); resource allocation and scheduling strategies; tuning parameter values.

Accounting Monitor

CPU usage, by workload component; logical I/O operation count, customer completions, by workload component

Sources of Information

System Description

hardware configuration; operating system (and version); resource allocation and scheduling strategies; tuning parameter values.

Accounting Monitor

CPU usage, by workload component; logical I/O operation count, customer completions, by workload component

Software Monitor

By device — measured busy time; physical I/O operation count; average queue length.

By workload component — throughput, average response time.

Sources of Information

System Description

hardware configuration; operating system (and version); resource allocation and scheduling strategies; tuning parameter values.

Accounting Monitor

CPU usage, by workload component; logical I/O operation count, customer completions, by workload component

Software Monitor

By device — measured busy time; physical I/O operation count; average queue length.

By workload component — throughput, average response time.

Hardware Monitor

observed busy time, by device.

Measurement and Monitoring

Most approaches to measurement gathering are based on some notion of event.

An event is a pre-defined change in system state—dependent on the metric being measured—which triggers the recording of data.

Example events might be memory references, disk accesses or network communication operations.

Metrics

The measures or metrics that a performance analyst might want to record can be classified as follows:

Event-count metrics: simply need to record how often certain event(s) occur within a run.

Metrics

The measures or metrics that a performance analyst might want to record can be classified as follows:

Event-count metrics: simply need to record how often certain event(s) occur within a run.

Secondary-event metrics: predetermined events indicate that the parameter we are interested in has changed, so these events are used to trigger data collection.

Metrics

The measures or metrics that a performance analyst might want to record can be classified as follows:

Event-count metrics: simply need to record how often certain event(s) occur within a run.

Secondary-event metrics: predetermined events indicate that the parameter we are interested in has changed, so these events are used to trigger data collection.

Profile: a complete picture of the overall behaviour is required, and so a whole set of events may be used to initiate the recording of data.

Role of logging

Dynamic information about the operation of a system under a workload can sometimes be extracted from the information recorded for general purposes such as accounting and logging.

Role of logging

Dynamic information about the operation of a system under a workload can sometimes be extracted from the information recorded for general purposes such as accounting and logging.

However, in general the information required for performance analysis is more sophisticated and requires specific monitoring and measurement of the system.

Software performance monitors

Software performance monitors analyse the behaviour of the system during operation and write records describing the resource usage and the performance status of the system.

For example, at specified intervals, queue lengths or device state indicators may be sampled and the results written to the record.

Software performance monitors

Alternatively, certain events which are considered to be significant (such as swapping a page) may be documented in the record.

In general the volume and form of the data recorded by software monitoring means that it is not possible to use it directly to parameterise the model. Instead it must be processed and analysed to produce suitable summaries.

Most software monitors will include a reporting component which will undertake at least the initial stages of this task.

Hardware monitors

Hardware monitors are also available for some systems, and have the advantage of being "external" to the system under observation: they do not perturb system operation.

Hardware monitors

Hardware monitors are also available for some systems, and have the advantage of being “external” to the system under observation: they do not perturb system operation.

However, in general the type of information which they can record is less detailed and so favour a homogeneous view of workload.

Specialised monitors

Some commonly occurring subsystems, such as databases, have given rise to specialised application software monitors. This is necessary because the subsystem has a certain amount of autonomy from the host operating system which makes access to its internal behaviour difficult.

Collecting measurement data

There are four basic strategies for collecting measurement data. Each has advantages and disadvantages, and in particular the analyst should be aware of the extent to which the chosen strategy perturbs the system under observation.

1. Event-driven
2. Tracing
3. Sampling
4. Indirect

Event-driven

Event-driven

This strategy is best-suited to event-count metrics and secondary-event metrics for which the event frequency is low.

Each event of interest is recorded.

Monitoring overhead is only incurred for the events of interest, but perturbation of the systems can be high if the frequency is high.

An additional mechanism must also be provided to dump the contents of the counter when the program finishes execution.

Event-driven

Event-driven: example

If the desired metric is the number of page faults that occur in the execution of a program, the performance analyst can modify the page-fault-handling routine in the operating system to increment a counter whenever the routine is entered.

Tracing

Tracing

This is similar to event-driven since the event of interest is used to trigger data collection, but in addition to the event details of the system state are also recorded.

It has the same disadvantages as event-driven.

In the previous example, for each page fault we would record the address which caused the fault. This increases the time overhead of the strategy and can, additionally, lead to storage problems when the event of interest is frequent.

Sampling

Sampling: definition

This is a general statistical measurement technique whereby a subset of the members of a population being examined is selected at random.

It assumes that since the subset is chosen at random, its characteristics will approximate those of the total population.

Sampling

Sampling as a measurement strategy

Record data at time intervals which are previously fixed and independent of the running of the system. Large volumes of data are recorded, especially if the event of interest is not frequent. However perturbation is more uniform.

Note that each run of a sampling-based experiment is likely to produce a different result since the samples occur asynchronously with respect to the system's execution. Nevertheless, while the exact behaviour may differ, the statistical behaviour should remain approximately the same.

Indirect

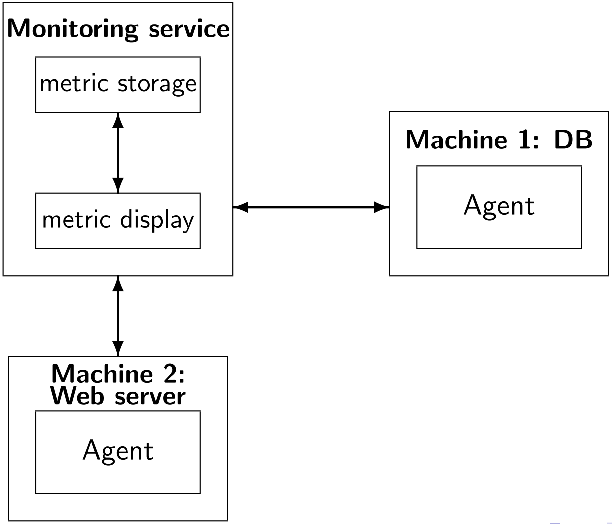
Indirect

An indirect strategy must be used for measures which are not directly measurable.

The general approach is to define an alternative metric from which the metric of choice can be deduced or derived.

In these cases the appropriate metric and measurement strategy will often need to be defined on an ad hoc basis and will depend on the ingenuity and creativity of the analyst.

Metric collection systems



Data Deluge

One of the advantages of using a metric collection system, such as RRDTool, is that it has built in mechanisms for coping with the huge amounts of data that will be collected from an operational system.

These strategies are typically based on the assumption that the detailed information is only required for the recent past.

Data is selectively discarded as time progresses until at some user-defined age, say one year, the data is completely deleted.

Monitoring network devices

Network devices and embedded systems are generally closed, meaning that it is not possible for the user to install software within them.

This means that it is not possible to use a metric collection system, based on agents, for these systems.

Instead it is common to use the Simple Network Management Protocol (SNMP) to collect data about the operation of these devices.

Workload characterisation

Workload characterisation is the process of selecting the workload or workloads on which to base the performance study. Difficult questions arise even in considering an existing computing environment:

Workload characterisation

Workload characterisation is the process of selecting the workload or workloads on which to base the performance study. Difficult questions arise even in considering an existing computing environment:

What constitutes a typical workload?

Workload characterisation

Workload characterisation is the process of selecting the workload or workloads on which to base the performance study. Difficult questions arise even in considering an existing computing environment:

What constitutes a typical workload?

How should a measurement interval, or time window, be selected?

Workload characterisation

Workload characterisation is the process of selecting the workload or workloads on which to base the performance study. Difficult questions arise even in considering an existing computing environment:

What constitutes a typical workload?

How should a measurement interval, or time window, be selected?

Should data from several measurement intervals be averaged?

Problems in workload characterisation

It may not be possible to measure an environment directly, for example when an existing workload is being moved to a new system, or a new workload is being introduced to an existing system.

Problems in workload characterisation

It may not be possible to measure an environment directly, for example when an existing workload is being moved to a new system, or a new workload is being introduced to an existing system.

Moreover users who will often change their usage patterns dependent on the service available.

Workload components

It can be useful to distinguish different workload components.

These are the significantly different tasks the system may be required to undertake.

The choice of workload components can have significant impact on the results of a performance study. For example, if the packets in two networks are generally a mixture of two sizes—short and long—the workload to compare the networks should consist of short and long packet sizes. Using the wrong workload components will lead to inaccurate conclusions.

Workload components example

The following data was collected from 6 universities over a 6 month period: it considers all applications.

	Average	Coefficient of Variation
CPU time	2.19	40.23
No. of direct writes	8.20	53.59
Direct write kbytes	10.21	82.41
No. of direct reads	22.64	25.65
Direct read kbytes	49.70	21.01

Workload components example

The following data was collected from 6 universities over a 6 month period: it considers editing only.

	Average	Coefficient of Variation
CPU time	2.57	3.54
No. of direct writes	19.74	4.33
Direct write kbytes	13.46	3.87
No. of direct reads	37.77	3.73
Direct read kbytes	36.93	3.16

Workload components

For most performance studies, based on the objective of the studies we can abstract details of the workload components down to a minimal set.

For example, if we wished to study the impact of scientific computation on communication we might consider a model with just three workload components: scientific computation, communication and other.

Workload components

If we were developing a queueing network model each of these workload components would be represented a distinct class of customers.

For each class we would then need to know the service demand of customers of that class at each of the resources in the system, as well as the number of such customers (for a closed class) or the arrival rate (for an open class).

Workload components

If we were developing a GSPN model each class of customer would be represented in distinct cycles within the model (only closed classes can be represented in a GSPN) competing for the resources via synchronisations (cf. the reader-writer model).

Workload components

If we were developing a GSPN model each class of customer would be represented in distinct cycles within the model (only closed classes can be represented in a GSPN) competing for the resources via synchronisations (cf. the reader-writer model).

Similarly in a PEPA model, each class of customer is likely to be represented by a distinct component within the model.

Devolved workload

Often the resources we are interested in within our systems are quite low level, e.g. CPU and disks, whereas our workload components are expressed in terms of user applications. We must therefore derive the devolved workload.

Devolved workload

Often the resources we are interested in within our systems are quite low level, e.g. CPU and disks, whereas our workload components are expressed in terms of user applications. We must therefore derive the devolved workload.

This means viewing the system as a series of layers each built one on top of another. A layer offers services to the layer above and generates workload to the layer below.

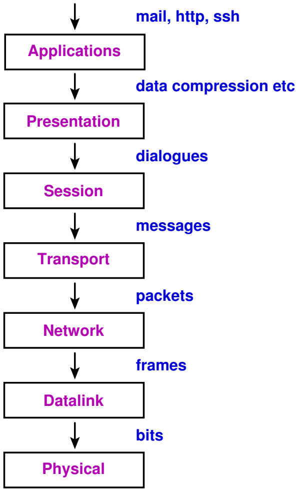
Devolved workload

Often the resources we are interested in within our systems are quite low level, e.g. CPU and disks, whereas our workload components are expressed in terms of user applications. We must therefore derive the devolved workload.

This means viewing the system as a series of layers each built one on top of another. A layer offers services to the layer above and generates workload to the layer below.

The devolved workload is then the amount of workload requested from the lower layer in order to satisfy one unit of workload from the layer above.

Devolved workload example



For example, this technique might be used to derive the typical usage of a communication gateway involved in one `http` connection.

Seasonal behaviour

Most systems display varying behaviour over time, possibly at different timescales.

Seasonal behaviour

Most systems display varying behaviour over time, possibly at different timescales.

Examples include:

- Web pages offering advice on filling tax returns will exhibit varying patterns of use over a year,
- Most human-centric systems, such as bank ATMs, will have a workload which varies throughout the day.

Seasonal behaviour

Most systems display varying behaviour over time, possibly at different timescales.

Examples include:

- Web pages offering advice on filling tax returns will exhibit varying patterns of use over a year,
- Most human-centric systems, such as bank ATMs, will have a workload which varies throughout the day.

Generally the time window which exhibits peak use is chosen for performance studies on the understanding that performance will only improve during the rest of the cycle.

Behaviour under future workload

When the performance study is focussed on the behaviour of a system under a future workload, we cannot use measurement of the current workload directly although it may be used as the basis for prediction of the future workload.

Behaviour under future workload

When the performance study is focussed on the behaviour of a system under a future workload, we cannot use measurement of the current workload directly although it may be used as the basis for prediction of the future workload.

There are two major ways in which this prediction can be carried out.

1. Trend Analysis
2. Key Value Indicators

Trend Analysis

Trend Analysis

If historical workload measurements exist, the current workload measurements may be considered in relation to these and examined for trends.

Moving averages and exponential smoothing are both techniques for extrapolating future values from a series.

In this way the expected future values of the workload parameters are predicted.

Key Value Indicators

Key Value Indicators

Users or system managers often find it easier to predict the future in terms of their business than in terms of the service demands or load intensity of the resulting computer system workload.

The key value indicator (KVI) is a quantifiable business variable such as the number of customers.

The current workload is then related to the current value of the KVI and the users' predicted future computing needs in terms of KVI are then related back to give future workload estimates.

Monitoring, parameterisation, characteristics

Even when the parameters for a model have been chosen and monitoring software or hardware have been used to measure the system much work is generally needed to extract appropriate values from the substantial volume of data.

Monitoring, parameterisation, characteristics

Even when the parameters for a model have been chosen and monitoring software or hardware have been used to measure the system much work is generally needed to extract appropriate values from the substantial volume of data.

For Markovian models, where we only require the mean duration, the analysis will generally be less sophisticated than what is required to find input values for a simulation models which use arbitrary distributions with multiple parameters.

Parameterisation techniques

Commonly used techniques include the following:

- Averaging
- Finding variability
- Single parameter histograms
- Multi-parameter histograms
- Cluster analysis

Cluster analysis

Most large computer systems have workloads consisting of several identifiable components or classes.

There are two key goals when the observations of the system are analysed to identify these classes.

Cluster analysis

Most large computer systems have workloads consisting of several identifiable components or classes.

There are two key goals when the observations of the system are analysed to identify these classes.

Clearly, customers or jobs within a class should exhibit similar patterns of behaviour within the system.

Cluster analysis

Most large computer systems have workloads consisting of several identifiable components or classes.

There are two key goals when the observations of the system are analysed to identify these classes.

Clearly, customers or jobs within a class should exhibit similar patterns of behaviour within the system.

But also, if the objectives of the performance study require separate performance predictions for different workload components, they must be represented in different classes.

Cluster Analysis

The data about a system generated by software monitoring may contain several thousand “user” profiles.

The objective of cluster analysis is to reduce this data to a few key characteristics, identifying the different types of users in the system.

Users who share characteristics are gathered together into clusters; eventually each cluster will be represented by one class in the model.

Cluster Analysis

The steps of cluster analysis are as follows:

1. Sample the workload data.
2. Select important workload parameters.
3. Transform parameters, if necessary.
4. Remove outliers.
5. Select a distance measure.
6. Perform clustering.
7. Interpret results.
8. Change parameters, or number of clusters, and repeat steps 3 to 6.
9. Select representative components from each cluster.

Clustering example

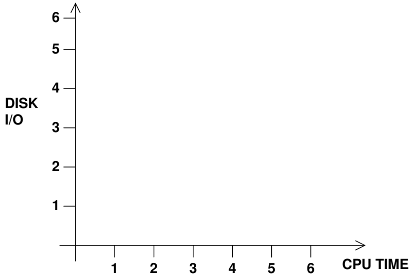
Consider a workload with 5 components and 2 parameters:

Program	A	B	C	D	E
CPU time	2	3	1	4	5
Disk I/O	4	5	6	3	2

Clustering example

Consider a workload with 5 components and 2 parameters:

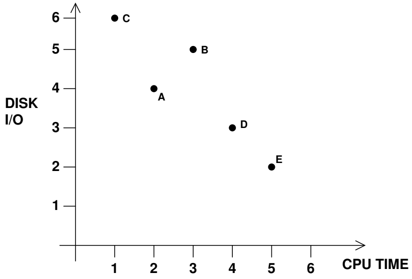
Program	A	B	C	D	E
CPU time	2	3	1	4	5
Disk I/O	4	5	6	3	2



Clustering example

Consider a workload with 5 components and 2 parameters:

Program	A	B	C	D	E
CPU time	2	3	1	4	5
Disk I/O	4	5	6	3	2

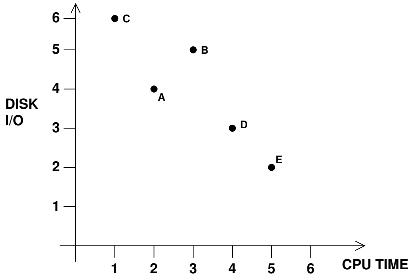


Clustering example

Consider a workload with 5 components and 2 parameters:

Program	A	B	C	D	E
CPU time	2	3	1	4	5
Disk I/O	4	5	6	3	2

$$\begin{array}{cccccc} & A & B & C & D & E \\ A & 0 & \sqrt{2} & \sqrt{5} & \sqrt{5} & \sqrt{13} \\ B & & 0 & \sqrt{5} & \sqrt{5} & \sqrt{13} \\ C & & & 0 & \sqrt{18} & \sqrt{32} \\ D & & & & 0 & \sqrt{2} \\ E & & & & & 0 \end{array}$$

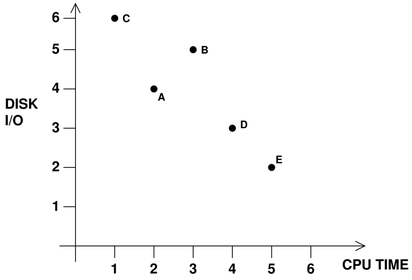


Clustering example

Consider a workload with 5 components and 2 parameters:

Program	A	B	C	D	E
CPU time	2	3	1	4	5
Disk I/O	4	5	6	3	2

$$\begin{array}{c|ccccc} & A & B & C & D & E \\ A & 0 & \sqrt{2} & \sqrt{5} & \sqrt{5} & \sqrt{13} \\ B & & 0 & \sqrt{5} & \sqrt{5} & \sqrt{13} \\ C & & & 0 & \sqrt{18} & \sqrt{32} \\ D & & & & 0 & \sqrt{2} \\ E & & & & & 0 \end{array}$$

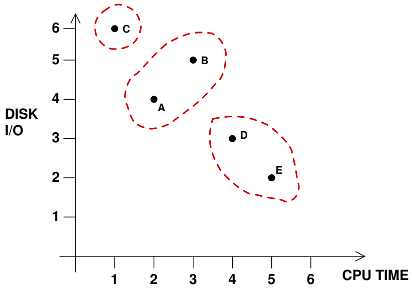


Clustering example

Consider a workload with 5 components and 2 parameters:

Program	A	B	C	D	E
CPU time	2	3	1	4	5
Disk I/O	4	5	6	3	2

$$\begin{array}{cccccc} & A & B & C & D & E \\ A & 0 & \sqrt{2} & \sqrt{5} & \sqrt{5} & \sqrt{13} \\ B & & 0 & \sqrt{5} & \sqrt{5} & \sqrt{13} \\ C & & & 0 & \sqrt{18} & \sqrt{32} \\ D & & & & 0 & \sqrt{2} \\ E & & & & & 0 \end{array}$$

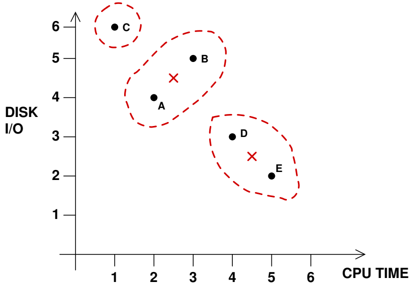


Clustering example

Consider a workload with 5 components and 2 parameters:

Program	A	B	C	D	E
CPU time	2	3	1	4	5
Disk I/O	4	5	6	3	2

$$\begin{array}{c|ccccc} & A & B & C & D & E \\ A & 0 & \sqrt{2} & \sqrt{5} & \sqrt{5} & \sqrt{13} \\ B & & 0 & \sqrt{5} & \sqrt{5} & \sqrt{13} \\ C & & & 0 & \sqrt{18} & \sqrt{32} \\ D & & & & 0 & \sqrt{2} \\ E & & & & & 0 \end{array}$$

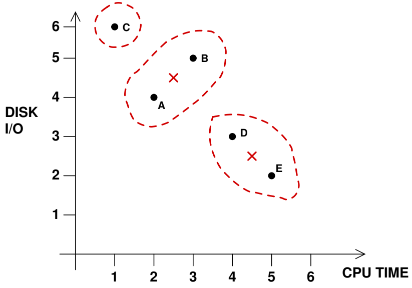


Clustering example

Consider a workload with 5 components and 2 parameters:

Program	A	B	C	D	E
CPU time	2	3	1	4	5
Disk I/O	4	5	6	3	2

$$\begin{array}{cccc} & AB & C & DE \\ AB & 0 & \sqrt{4.5} & \sqrt{10.25} \\ C & & 0 & \sqrt{24.4} \\ DE & & & 0 \end{array}$$

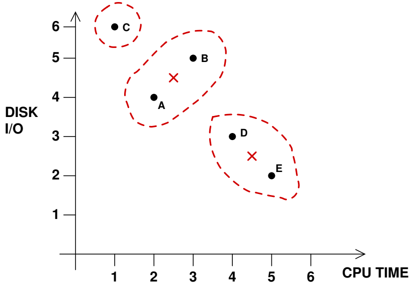


Clustering example

Consider a workload with 5 components and 2 parameters:

Program	A	B	C	D	E
CPU time	2	3	1	4	5
Disk I/O	4	5	6	3	2

$$\begin{array}{cccc} & AB & C & DE \\ AB & 0 & \sqrt{4.5} & \sqrt{10.25} \\ C & & 0 & \sqrt{24.4} \\ DE & & & 0 \end{array}$$

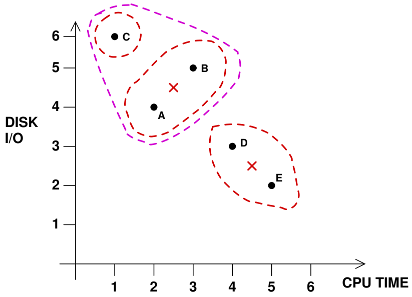


Clustering example

Consider a workload with 5 components and 2 parameters:

Program	A	B	C	D	E
CPU time	2	3	1	4	5
Disk I/O	4	5	6	3	2

$$\begin{array}{cccc} & AB & C & DE \\ AB & 0 & \sqrt{4.5} & \sqrt{10.25} \\ C & & 0 & \sqrt{24.4} \\ DE & & & 0 \end{array}$$

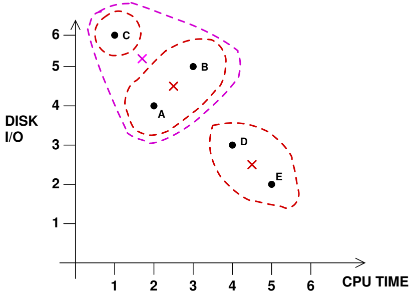


Clustering example

Consider a workload with 5 components and 2 parameters:

Program	A	B	C	D	E
CPU time	2	3	1	4	5
Disk I/O	4	5	6	3	2

$$\begin{array}{cccc} & AB & C & DE \\ AB & 0 & \sqrt{4.5} & \sqrt{10.25} \\ C & & 0 & \sqrt{24.4} \\ DE & & & 0 \end{array}$$


DISK I/O
CPU TIME
C
B
A
D
E