Performance of Computer System

Workload Selection and Characteristics

Dr. Lina Xu
lina.xu@ucd.ie

School of Computer Science,
University College Dublin

October 11, 2019

- Different types of workloads
- What workloads are commonly used
- How to select appropriate workload types
  - Right level of details
  - Representativeness
  - Timeliness

Workload Types

- Real workload
  - One observed during normal system operations
  - Non-repeatable
- Synthetic workload
  - Approximation of real workload
  - Can be applied repeatedly in a controlled manner
  - No large data files, no sensitive data
  - Easily modified and ported
  - Easily measured
- Test workload
  - Any workload used in performance studies
  - Real or synthetic

Workload Selection

Workload selection is important

- "Would you tell me, please, which way I ought to go from here?"
- "That depends a good deal on where you want to get to," said the Cat.
- "I don't much care where—"said Alice.
- "Then it doesn't matter which way you go," said the Cat.
- From Alice's Adventures in Wonderland.

Services Exercised

- System Under Test (SUT)
- Component Under Study (CUS)
- Metrics chosen should reflect system level performance
- Workload chosen should reflect SUT, not CUS

Level of Detail

Most frequent request

- valid if one service is requested much more often than others
- examples: add instruction, kernels

Frequency of request types

- example: instruction mix
- context sensitive services - must use a set (e.g. caching)

Time stamped sequence of requests (trace)

- too much detail for analytical modeling
- may require exact reproduction of component behaviour for timing

Level of Detail

Average resource demand

- analytical models use request rate rather than requests
- group similar services in classes; use avg. demand per class

Distribution of resource demands, used if

- variance in resource demands is large
- distribution impacts performance

Representativeness

Arrival rate

- Should be the same or proportional to that of real application

Resource demands

- Total demand should be $=$ or proportional to that of real application

Resource usage profile

- Amount and sequence in which resources are consumed
- Especially important when forming a composite workload

Timeliness

Users change usage pattern based on

- New services available
- Changes in system performance: users optimise demand

Anticipate changes

- Monitor user behaviour on ongoing basis
- Future may be different than past or present

Other Considerations

Load level

- full capacity (best case)
- beyond capacity (worst case)
- real workload (average case)
- for procurement, consider typical case
- for design, consider best $< -- >$ worst cases

Impact of external components

- don't use workload that makes external component bottleneck
- e.g. if studying CPU performance, don't use data so large that system is paging
- otherwise, all alternatives will give equally good performance

Repeatability

- want to be able to repeat results without excessive variance
- highly random resource demands should be avoided

Workload Characteristics

Motivation

- Motivation
  - Since a real-user environment is generally not repeatable, it is necessary to study the real-user environments, observe the key characteristics, and develop a workload model that can be used repeatedly. This process is called workload characterization.
- Different approaches for characterising workload

Workload Characterisation

Why

- Observe key performance characteristics of a workload
- Develop a model that can be used for further study

Workload parameters

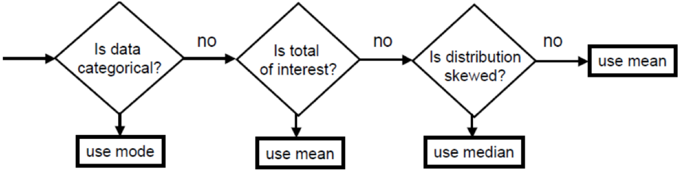
- Measured quantities that depend on workload not system
- Types
  - Service request
  - Resource demands
- Examples
  - Transaction types
  - Instructions,
  - Packet types destinations

Techniques for Workload Characterisation

- Averaging
- Specifying Dispersion
- Single-parameter histograms
- Multiparameter histograms
- Principal components analysis
- Markov models
- Clustering

Averaging

- Arithmetic mean of values $x_1, x_2, ..., x_n$: $\bar{x} = \frac{1}{n}\sum_1^n x_n$
- Caution: arithmetic mean is not always appropriate "index of central tendency"
- Median = 50th percentile value
- Mode = most frequent



Specifying Dispersion

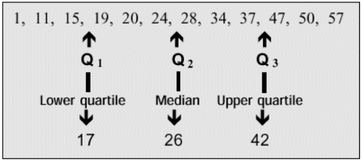
- Averaging is insufficient if there is large variability in data values
- Variability of $\{x_1, x_2, ..., x_n\}$ is commonly specified by variance
  - $s^2 = \frac{1}{n-1} \sum_1^n (x_i - \bar{x})^2$
- Sample standard deviation, s = sqrt(sample variance)
  - often more meaningful: same units as the mean
- Alternatives for summarising variability
  - range: maximum - minimum
  - 10 and 90 percentiles
  - semi-interquartile range (SIQR)
  - mean absolute deviation
- The ratio of the standard deviation to the mean is called the Coefficient Of Variation (C.O.V.). = s/mean

Specifying Dispersion

$$\text{range} = \max - \min \qquad \text{IQR} = Q_3 - Q_1$$

16, 24, 26, 26, 26, 27, 28   23, 25, 28, 28, 32, 33, 35

$$\text{range} = 28 - 16 \qquad \text{IQR} = 33 - 25$$



Specifying Dispersion

- The resource demands of various programs executed on six university sites were measured for 6 months.
- The average demand by each program is shown.
- Notice that the C.O.V. of the measured values are rather high, indicating that combining all programs into one class is not a good idea.
- Programs should be divided into several classes.
- For all editors in the same data. The C.O.V. are now much lower.

TABLE 6.1 Workload Characterization Using Average Values

Data	Average	Coefficient of Variation
CPU time (VAX-11/780)	2.19 seconds	40.23
Number of direct writes	8.20	53.59
Direct-write bytes	10.21 kbytes	82.41
Number of direct reads	22.64	25.65
Direct-read bytes	49.70 kbytes	21.01

TABLE 6.2 Characteristics of an Average Editing Session

Data	Average	Coefficient of Variation
CPU time (VAX-11/780)	2.57 seconds	3.54
Number of direct writes	19.74	4.33
Direct-write bytes	13.46 kbytes	3.87
Number of direct reads	37.77	3.73
Direct-read bytes	36.93 kbytes	3.16

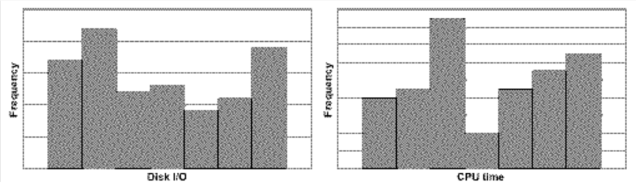
Single-parameter Histograms

SPH: relative frequencies of various values of a parameter

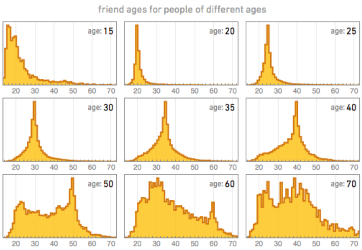
- divide complete range into buckets
- count observations that fall in each

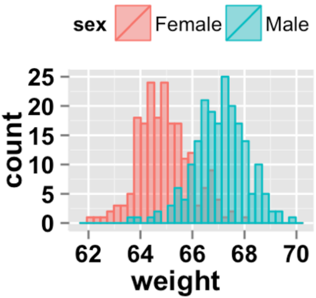
Uses

- simulation: generate test workload matching distribution
- analytical model: validate probability distribution used in model



Single-parameter Histograms Examples





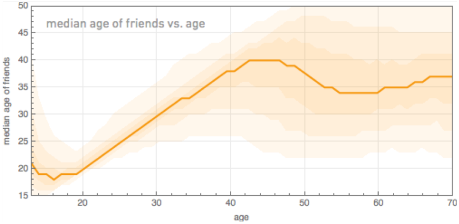
Single-parameter Histograms Examples

Single-parameter Histograms: Disadvantages

- much data: n buckets, m parameters/ component, k components
  - should only be used if variance is high and averages are inappropriate
- SPH ignore correlation among parameters

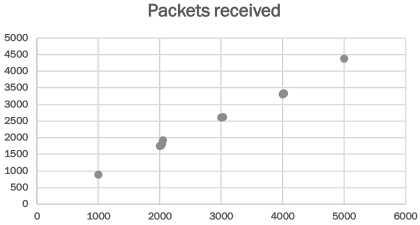
Multiparameter histograms

MPH: use when significant correlation between parameters



Multiparameter histograms

MPH: use when significant correlation between parameters



Multiparameter histograms: Disadvantages

- rendering more than 2-parameter histograms is problematic
- much detail; uncommon to use them

Weighted Sum of Parameters

- Classify workload components by weighted sum of their parameter values
  - $y = \sum_{j=1}^{n} a_j x_j$
- Use y to classify components into categories, e.g. high, low
- Problem: choosing appropriate weights for parameters.
  - Bad choice of weights may group dissimilar components

The average consumption of 17 types of food in grams per person per week for every country in the UK

	England	N Ireland	Scotland	Wales
Alcoholic drinks	375	135	458	475
Beverages	57	47	53	73
Carcase meat	245	267	242	227
Cereals	1472	1494	1462	1582
Cheese	105	66	103	103
Confectionery	54	41	62	64
Fats and oils	193	209	184	235
Fish	147	93	122	160
Fresh fruit	1102	674	957	1137
Fresh potatoes	720	1033	566	874
Fresh Veg	253	143	171	265
Other meat	685	586	750	803
Other Veg	488	355	418	570
Processed potatoes	198	187	220	203
Processed Veg	360	334	337	365
Soft drinks	1374	1506	1572	1256
Sugars	156	139	147	175

Principal components analysis

Problem

- Find weights so that weighted sums provide maximum discrimination among components
- $y_i = \sum_{j=1}^{n} a_{ij} x_j$

For each component i,

- y_i is a linear combination of parameter values x_j
- a_{ij} is loading of x_j on y_i

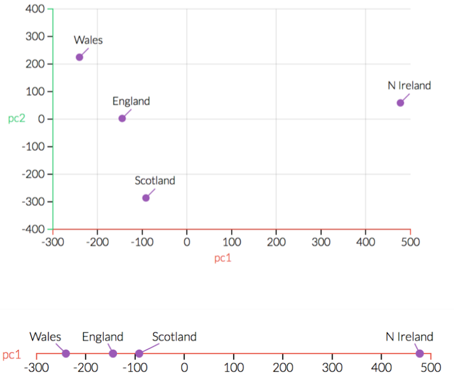
Choose weights so that y's form an orthogonal set, namely

- $< y_i, y_j >= \sum_1^k a_{ik} a_{kj} = 0$

Properties: y's form an ordered set such that

- y_1 explains highest percentage of variance in resource demands
- Successive y_i explain increasingly lower percentages

Principal components analysis

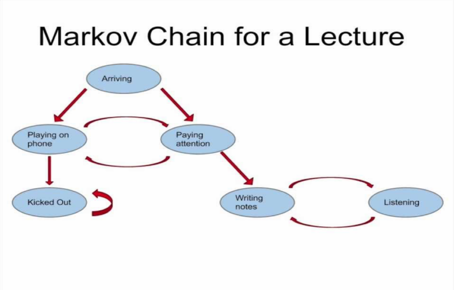


Markov models

One of the Queueing Models

- This section is about being able to describe the behavior of queues. Queues are certainly a prevalent object in computer systems, and our goal here is to write the equations that describe them. The language we'll use here is mathematical, but nothing really more complicated than algebra.
- Sometimes, not only the relative frequency but order of service requests is important!

Markov models



First Order Markov Models

- Probabilistic models that can generate sequences of states (here, representing service requests)
- Next state depends only upon current state
- Completely characterised by a transition probability matrix
- Transition probability matrix properties
  - a_{ij} = P(system will enter state j — system is in state i)
  - $0 \leq a_{ij} \leq 1$
  - $\sum a_{ij} = 1$

First Order Markov Models Example I

Modeling packets on network

- small packets = 87.5%; large packets = 12.5% (1 in 8 is large)
- large packet always followed by a small packet

Pairwise percentages

	next packet	
current packet	small	large
small	75%	12.5%
large	12.5%	0%

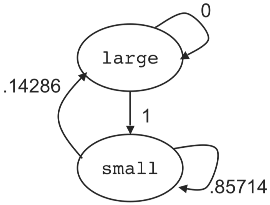
First Order Markov Models Example I

Pairwise percentages

	next packet	
current packet	small	large
small	75%	12.5%
large	12.5%	0%

Transition Matrix

	next packet	
current packet	small	large
small	75/87.5 = .85714	12.5/87.5 = .14286
large	1	0



First Order Markov Models Example II

Modeling packets on network

- small packets = 87.5%; large packets = 12.5% (1 in 8 is large)
- large packet followed by a small packet half the time

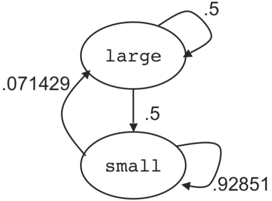
First Order Markov Models Example II

Pairwise percentages

	next packet	
current packet	**small**	**large**
small	**81.25%**	**6.25%**
large	**6.25%**	**6.25%**

Transition Matrix

	next packet	
current packet	**small**	**large**
small	**81.25/87.5 = .928571**	**6.25/87.5 = .071429**
large	**6.25/12.5 = .5**	**6.25/12.5 = .5**

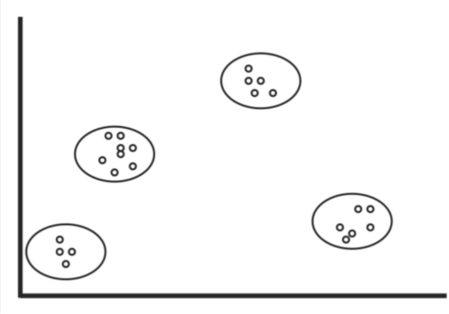


First Order Markov Models Propertie

- Finite: The model consists of a finite number of states
- Memory-less: The next state depends only upon the present state, not past states
- Absorbing state: (enter, but never leave) any state with a 1 on the main diagonal in the transition probability matrix
- Time independent: Transition matrix probabilities do not vary over time

Clustering

- Workload often consists of large number of components
- Want to classify components into small number of clusters whose members are similar



How to Cluster

- Take a sample, that is, a subset of workload components.
- Select workload parameters
- Transform parameters, if necessary.
- Remove outliers.
- Scale observations (e.g. normalise to 0 mean, unit variance)
- Select distance metric, e.g. Euclidean, Manhattan distance
- Select clustering algorithm
- Perform clustering

How to Cluster

Select workload parameters

- Select those that (1) impact performance, and (2) vary significantly

Transform parameters, if necessary.

- If the distribution of a parameter is highly skewed, one should consider the possibility of replacing the parameter by a transformation or function of the parameter.

How to Cluster

Remove outliers

- Outliers affect normalisation and thus can affect clustering
- Remove if they do not consume significant fraction of system resources

Scale observations (e.g. normalise to 0 mean, unit variance)

- Normalize to Zero Mean and Unit Variance:
  - $x'_{ik} = \frac{x_{ik} - \bar{x}_k}{s_k}$
- Weights
  - $x'_{ik} = W_k x_{ik}$

How to Cluster

Scale observations (e.g. normalise to 0 mean, unit variance)

- Range Normalisation: The range is changed from $[x_min, k, x_max, k]$ to $[0, 1]$.
- $x'_{ik} = \frac{x_{ik} - x_{min,k}}{x_{max,k} - x_{min,k}}$

How to Cluster

Select distance metric

- Euclidean
- Manhattan distance
- Weighted Euclidean Distance

How to Cluster

Select clustering algorithm

- Nearest neighbour: min distance between objects in two clusters
- Furthest neighbour: max distance between objects in two clusters

How to Cluster

Perform clustering

- Start with n clusters, using minimum spanning tree to merge
- Represent with dendrogram

Minimum spanning tree

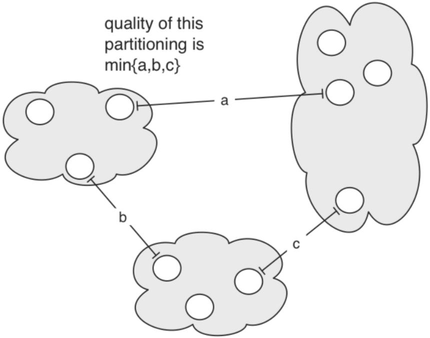
Minimum Spanning Tree Problem

- Undirected graph G with vertices for each of n objects
- Weights $d(u, v)$ on the edges giving the distance u and v,
- Find the subgraph T that connects all vertices and minimises

T will be a tree!

Minimum spanning tree – clustering

- Agglomerative approach
- Divide the n items up into k groups so that the minimum distance between items in different groups is maximized



Minimum spanning tree – clustering

1. Start with $k = n$ clusters.
2. Find the centroid of the $i_t h$ cluster, $i = 1, 2, ..., k$. The centroid has parameter values equal to the average of all points in the cluster.
3. Compute the inter-cluster distance matrix. Its $(i, j)_{th}$ element is the distance between the centroids of cluster i and j.
4. Find the smallest nonzero element of the distance matrix. Let the distance between clusters l and m, be the smallest. Merge clusters l and m. Also merge any other cluster pairs that have the same distance.
5. Repeat steps 2 to 4 until all components are part of one cluster.

Clustering Example