Workload characterization:

estrarre dei parametri che mi permettono di descrivere il workload

pratico: un numero ragionevolmente piccolo di parametri

parametri ad alto livello => lato client (throughput, response time, ..), ma non sappiamo cosa sta accadendo lato server

parametri di basso livello => parametri collezionati lato server, sono a livello applicativo (uso dei processori, della memoria, ..) che mi aiutano a capire come il server sta rispondendo

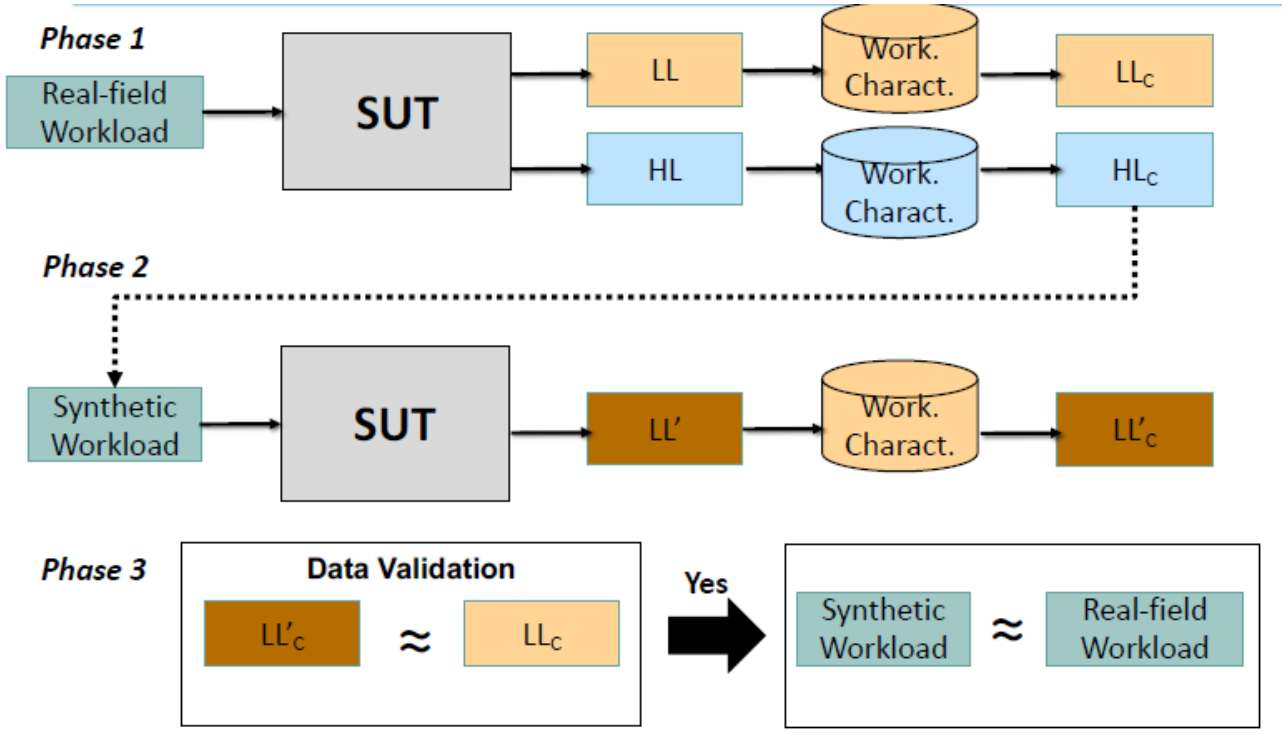
dobbiamo cercare di associare i parametri di alto livello a quelli di basso livello, se vario il client cosa succede al server?

esercizio: generare un random workload che sia quanto più possibile reale

-collezionare i dati di basso livello attraverso vm-stat per esempio (vmstat -n, me lo fa per 300 sec se n=300)

-creare un workload sintetico

-validare il workload sintetico, devo verificare che le richieste del workload sintetico approssimino bene le richieste fatte nel workload reale



Work. charact. ⇔ PCA & Clustering

HLc è il mio workload sintetico perchè sto caratterizzando il client, lo applico al server (il mio SUT) e ottengo LL'

effettuare la stessa workload characterization sia nella fase 1 che nella fase 2 per quanto riguarda i low level parameters, così posso confrontare due dataset con lo stesso numero di colonne

non posso confrontare dei carichi di lavoro in base a come i client li percepiscono, devo far riferimento al sistema a cui viene applicato il workload (quindi, il server)

Generare un workload reale:

- creare uno o più thread group
- ogni thread group può avere un numero diverso di thread
- settare una durata di 5 min del test
- settare un ramp-up period
- settare un CTT per settare il request rate, ognuno diverso per ogni thread group
- usare un random controller per cercare di emulare un workload reale
- settare un nome differente alle richieste che vengono fatte dai vari thread group

più thread group ➔ esegui thread groups in modo consecutivo, per rendere indipendenti le esecuzioni dei vari thread group (si fa nelle opzioni del test plan)

tante richieste diverse, alcune simili tra loro, in modo tale che PCA & Clustering mettano insieme le richieste simili e dividano le richieste diverse

Test di ipotesi

- a un campione: zero mean test, testare se la media è significativamente diversa da zero
- osservazioni accoppiate: i dati che ho collezionato dai campioni vengono raccolti nelle stesse condizioni, due vettori di dimensioni n ordinati nella stessa maniera, deve esserci una corrispondenza tra gli elementi corrispondenti dei due vettori

T-test a un campione

h=ttest(x,m) in matlab, m è la media e x sono i dati, assume che io non conosca la varianza e che i dati provengano da una distribuzione normale

- Ipotesi nulla H0: la media è m
- Ipotesi alternativa: la media non è m

quando applico questo test, il risultato è 1 se viene accettata l'ipotesi alternativa altrimenti è 0

esempio: [h,p]=ttest(protein, 20) voglio capire se la media è diversa da 20

h=1 ➔ la media non è 20

p=0.0046 ➔ p-value, valore più piccolo per rigettare l'ipotesi nulla, in questo caso il valore è bassissimo quindi posso assumere che l'ipotesi nulla è rigettata

dovrà essere minore di 0.05 se l'intervallo di confidenza è al 95%, dipende dall'intervallo di confidenza scelto (in MATLAB è di default al 95%)

i dati devono essere indipendenti e random, innanzitutto, per accettare che il t-test sia un metodo appropriato

Paired t-test:

richiede che la differenza di ogni coppia sia distribuita normalmente

h=ttest(x,y)

Two sample t-test:

misurazioni non accoppiate

richiede che i dati da entrambi i campioni siano distribuiti normalmente e abbiano la stessa varianza

h=ttest2(x,y)

Two sample t-test senza assumere varianze uguali:

h=ttest2(x,y,'Vartype','unequal')

Box plot: grafico che riassume valore minimo, massimo, media, primo e terzo quartile, più lunga è la box più è alta la varianza

Test parametrici: assumono che i dati della popolazione sono distribuiti normalmente, mi dà più sicurezza

Central Limit Theorem

Replacement: quando ho estratto un sample potrei estrarlo di nuovo in maniera random

Visual test: per capire se i dati vengono da una distribuzione normale

qqplot(x) ➔ disegna un q-q plot

H0 non rigettata ➔ Test parametrico

H0 rigettata ➔ Test non parametrico

se i dati non sono distribuiti secondo una normale, applichiamo il Wilcoxon rank sum test

[p, h]=ranksum(x,y) ➔ testa l'ipotesi nulla che i dati in x e y sono sample da distribuzioni continue con la stessa mediana (assume che i due sample siano indipendenti, ma possono avere lunghezze differenti)

il primo valore di uscita è il p-value

il secondo è l'output del test

esercizio (data validation): per ogni componente principale:

- check della normalità dei dati (LLc e LL'c), il visual test è preferito
    - se non sono normalmente distribuiti, allora applico un test non-parametrico (rank sum test)
    - altrimenti controlla le varianze uguali (box plot o uso vartest2 come ulteriore sicurezza)
        - se sono uguali, two sample t-test
        - altrimenti, two sample t-test con varianze diverse

vogliamo che il p-value sia alto, più non ci sono differenze statistiche e più ho approssimato bene

cerchiamo di prediligere più cluster rispetto a più componenti principali, per quanto riguarda l'esercizio