

UNIVERSITÀ FEDERICO II DI NAPOLI

SCUOLA POLITECNICA E DELLE SCIENZE DI BASE

CORSO DI IMPIANTI DI ELABORAZIONE - LAUREA MAGISTRALE IN INGEGNERIA INFORMATICA

ANNO ACCADEMICO 2022-2023

ELABORATO FINALE

Impianti di Elaborazione

Realizzazione Homework assegnati durante il corso

Prof.

Ing. Domenico Cotroneo

Assistante

Ing. Pietro Liguori

Ing. Stefano Rosiello

Autori:

Antonio Romano - M63001315

Giuseppe Riccio - M63001314

Umberto Pier Rosario Caturano - M63001260

Indice

1	Introduzione	1
2	Benchmark	2
2.1	Definizione degli obiettivi e overview	2
2.2	Definizione dei sistemi da confrontare	3
2.3	Raccolta dei dati indipendente	4
2.3.1	Le misurazioni	4
2.4	Valutazione della dimensione campionaria	5
2.5	Test statistico	7
2.5.1	Test Visivo	7
2.5.2	Test Parametrico	8
2.5.2.1	Normalità	8
2.5.2.2	Omoschedasticità	11
2.5.2.3	Two-sample t-test (pooled)	15
2.6	Risultati e considerazioni del confronto	17
3	Workload Characterization	18
3.1	Definizione degli obiettivi e overview	18
3.2	Raccolta e comprensione dei dati	19
3.3	Preprocessing dei dati	20
3.3.1	Analisi preliminare dei dati	20
3.3.2	Valutazione delle colonne costanti	21
3.3.3	Valutazione delle colonne correlate	22
3.3.4	Valutazione degli outliers	22
3.3.5	Standardizzazione dei dati	23
3.4	Esecuzione della Principal Component Analysis (PCA)	23
3.5	Esecuzione del Clustering	24
3.6	Analisi di sensitività della devianza persa al variare di PC e Cluster	27
3.6.1	Calcolo devianza del workload reale (pre-PCA)	27
3.6.2	Calcolo devianza post-PCA	27
3.6.3	Calcolo devianza post-CLUSTERING	28
3.6.4	Calcolo della percentuale della perdita di devianza	29
3.7	Report definitivo del Workload Characterization	30
4	Web Server	32
4.1	Definizione degli obiettivi per l'analisi di un Web Server	32
4.1.1	System Under Test	32
4.2	Capacity Test	34
4.2.1	Preparazione	35
4.2.2	Pianificazione	35
4.2.3	Esecuzione	37
4.2.4	Analisi dei dati	37
4.2.4.1	Valutazione della Fairness	40
4.2.5	Report e Conclusioni	40
4.3	Workload Characterization	42
4.3.1	Creazione del workload reale (ipotetico)	42
4.3.1.1	Workload reale di alto livello HL	42
4.3.1.2	Workload reale di basso livello LL	43

4.3.2	Workload Characterization HL	43
4.3.2.1	Preprocessing	44
4.3.2.2	PCA	45
4.3.2.3	Clustering	45
4.3.2.4	Analisi della devianza persa	48
4.3.2.5	Workload sintetico	48
4.3.3	Workload Characterization LL	49
4.3.3.1	Preprocessing	49
4.3.3.2	PCA	50
4.3.3.3	Clustering	51
4.3.3.4	Analisi della devianza persa	52
4.3.3.5	Workload sintetico	52
4.3.4	Applicazione del workload HL_c e creazione del LL'_c	53
4.3.5	Workload Characterization LL'	53
4.3.5.1	Preprocessing	53
4.3.5.2	PCA	54
4.3.5.3	Clustering	55
4.3.5.4	Analisi della devianza persa	56
4.3.5.5	Workload Sintetico LL'c	56
4.3.6	Verifica della bontà della caratterizzazione del workload: Confronto tra LL_c e LL'_c	56
4.4	Design of Experiments	58
4.4.1	Configurazione del piano di DOE	59
4.4.2	Analisi dell'importanza	60
4.4.3	Analisi di significatività statistica dei fattori	62
4.4.3.1	Normalità	62
4.4.3.2	Omoschedasticità	63
4.4.3.3	Anova	63
5	Regression	65
5.1	Definizione degli obiettivi e overview	66
5.2	Dataset Mail Server 1	66
5.2.1	Valutazione delle rette di regressione e R^2	66
5.2.2	Test di Normalità	67
5.2.3	Test di Mann-Kendall	68
5.2.4	Pendenza e intercetta della retta di regressione	68
5.3	Dataset Mail Server 2	69
5.3.1	Valutazione delle rette di regressione e R^2	69
5.3.2	Test di Normalità	70
5.3.3	Test di Mann-Kendall	71
5.3.4	Pendenza e intercetta della retta di regressione	71
5.4	Confronto tra Mail Server 1 e Mail Server 2	72
5.5	Dataset - OS1	72
5.5.1	Valutazione delle rette di regressione e R^2	72
5.5.2	Test di Normalità	73
5.5.3	Test di Mann-Kendall	74
5.5.4	Pendenza e intercetta della retta di regressione	75
5.6	Dataset - OS2	76
5.6.1	Valutazione delle rette di regressione e R^2	76
5.6.2	Test di Normalità	77

5.6.3	Test di Mann-Kendall	78
5.6.4	Pendenza e intercetta della retta di regressione	79
5.7	Dataset - OS3	80
5.7.1	Valutazione delle rette di regressione e R^2	80
5.7.2	Test di Normalità	81
5.7.3	Test di Mann-Kendall	82
5.7.4	Pendenza e intercetta della retta di regressione	83
5.7.5	Confronto tra OS1-OS2-OS3	84
5.8	VmRes1 - Failure Prediction	84
5.8.1	Valutazione della retta di regressione e R^2	84
5.8.2	Test di Normalità	85
5.8.3	Test di Mann-Kendall	85
5.8.4	Pendenza e intercetta della retta di regressione	86
5.8.5	Valutazione Failure Prediction	86
5.9	VmRes2 - Failure Prediction	87
5.9.1	Valutazione delle rette di regressione e R^2	87
5.9.2	Test di Normalità	87
5.9.3	Test di Mann-Kendall	88
5.9.4	Pendenza e intercetta della retta di regressione	88
5.9.5	Valutazione Failure Prediction	89
5.10	VmRes3 - Failure Prediction	89
5.10.1	Valutazione delle rette di regressione e R^2	89
5.10.2	Test di Normalità	90
5.10.3	Test di Mann-Kendall	90
5.10.4	Pendenza e intercetta della retta di regressione	90
5.10.5	Valutazione Failure Prediction	91
6	Reliability	92
6.1	Definizione degli obiettivi e Overview	92
6.2	RBD	93
6.2.1	Teorema dell'Upper Bound	93
6.2.2	Conditioning	94
6.2.3	MTTF del sistema	95
6.3	Confronto tra due sistemi	95
6.3.1	Equilibrare i due sistemi	96
6.4	Skip Ring	97
6.4.1	Valutazione della Reliability dopo un mission time	98
6.5	Confronto tra RBD	98
6.5.1	Confronto 1	99
6.5.1.1	Analisi Visiva	99
6.5.1.2	Analisi analitica e grafica	99
6.5.2	Confronto 2	100
6.5.2.1	Analisi Visiva	100
6.5.2.2	Analisi analitica e grafica	100
6.5.3	Confronto 3	101
6.5.3.1	Analisi Visiva	101
6.5.3.2	Analisi analitica e grafica	101
6.5.4	Confronto 4	101
6.5.4.1	Analisi Visiva	102

6.5.4.2 Analisi analitica e grafica	102
6.6 Reliability di un sistema di controllo di un elicottero	102
6.6.1 Effetto di Coverage	104
7 FFDA	105
7.1 Definizione degli obiettivi e Overview	105
7.2 Analisi log del Calcolatore Mercury	106
7.2.1 Logging collection e Filtering	106
7.2.2 Data Manipulation	107
7.2.2.1 Sensitivity Analysis	108
7.2.3 Data Analysis	109
7.2.3.1 Fitting della reliability per lo studio della natura del fallimento	109
7.2.3.2 Goodness dei Fitting	111
7.3 Analisi log del Calcolatore Blue Gene/L	112
7.3.1 Logging collection e Filtering	112
7.3.2 Data Manipulation	113
7.3.2.1 Sensitivity Analysis	113
7.3.3 Data Analysis	114
7.3.3.1 Fitting della reliability per lo studio della natura del fallimento	114
7.3.3.2 Goodness dei Fitting	116
7.4 Confronto Reliability tra Mercury e Blue Gene	117
7.5 Analisi CWIN tra nodi (Mercury, BG/L) e categorie di errore (Mercury)	118
7.5.1 Coalescence Window per i nodi di Mercury	118
7.5.2 Coalescence Window per i nodi di Blue Gene	120
7.6 Confronto reliability di sistema vs. reliability dei nodi del calcolatore Mercury	121
7.7 Reliability bottlenecks	124
7.7.1 Bottlenecks di Mercury	124
7.7.2 Bottlenecks di BlueGene	124
7.8 Confronto reliability tra nodi simili (Mercury e Blue Gene)	125
7.9 Correlazione tra nodi e tipi di errore nel calcolatore Mercury	127

1 Introduzione

La seguente trattazione è una raccolta degli Homework assegnati durante il corso di Impianti di Elaborazione A.A 2022/2023 presso l'Università degli Studi di Napoli "Federico II". In questo documento, verranno presentati i risultati ottenuti attraverso l'esecuzione degli Homework assegnati durante il corso, accompagnati dalle relative analisi e discussioni. Ogni sezione conterrà una descrizione dettagliata del compito assegnato, dei metodi utilizzati per la risoluzione, dei risultati ottenuti e delle conclusioni tratte esplicitando a monte del capitolo, tramite degli schemi concettuali, il processo con cui è stato realizzato il relativo homework.

Gli Homework realizzati sono:

- **Benchmark**
- **Workload Characterization**
- **Web Server**
- **Regression**
- **Reliability Block Diagram - RBD**
- **FFDA**

Il Benchmark, ad esempio, sarà utilizzato per confrontare le prestazioni di diverse configurazioni di sistemi per l'elaborazione dei dati, mentre la Workload characterization sarà utilizzata per analizzare e comprendere il carico di lavoro su un sistema di elaborazione dati al fine di estrarre uno di dimensioni ridotte ma con la stessa significatività. La valutazione di diversi aspetti del Web Server saranno utilizzate per determinare se il sistema è in grado di gestire il carico previsto, se i fattori in gioco sono significativi ed importanti, o meno. Mentre, la regressione sarà utilizzata per analizzare e prevedere i dati relativi a distribuzioni con particolari trend statistici, invece, la RBD sarà utilizzata per individuare e confrontare la reliability di diverse topologie di sistemi (serie-parallelo o meno). Infine, la FFDA sarà utilizzata per calcolare la reliability e per comprendere meglio il comportamento di alcuni sistemi tramite l'analisi di misurazioni dirette effettuate su di essi e presentati sotto forma di event log.

I file sorgenti usati per svolgere gli homework, gli script e tutto il materiale fornito per le analisi presenti nella seguente relazione sono disponibili, sul repository GitHub creato per il progetto. Si lascia il QR code ed il link al repository:



https://github.com/giuseppericcio/Elaborato_Impianti_di_Elaborazione_2022-23

2 Benchmark

Contenuti

2.1	Definizione degli obiettivi e overview	2
2.2	Definizione dei sistemi da confrontare	3
2.3	Raccolta dei dati indipendente	4
2.3.1	Le misurazioni	4
2.4	Valutazione della dimensione campionaria	5
2.5	Test statistico	7
2.5.1	Test Visivo	7
2.5.2	Test Parametrico	8
2.6	Risultati e considerazioni del confronto	17

2.1 Definizione degli obiettivi e overview

Si effettua il confronto tra due sistemi di elaborazione (computer) differenti utilizzando il **Benchmark N-Body**. Essi è utilizzato per valutare le prestazioni di un sistema di elaborazione poiché richiede una grande quantità di calcoli per modellare il moto di un gran numero di corpi celesti in un sistema gravitazionale. In particolare, il programma utilizza l'algoritmo di integrazione di N-Body, che consente di **calcolare la posizione e la velocità di ogni corpo nel sistema** in base alle forze di gravità esercitate dagli altri corpi. L'algoritmo di integrazione di N-Body utilizza le derivate delle posizioni e delle velocità dei corpi per calcolare l'accelerazione e l'evoluzione del moto dei corpi nel sistema.

Algorithm 1 Pseudocodice dell' Algoritmo di integrazione NBody

```
1: procedure NBODY
2: Per ogni corpo i nel sistema:
3:   imposta l'accelerazione di i a 0
4: Per ogni coppia di corpi i e j nel sistema:
5:   calcola la distanza r tra i e j
6:   calcola la forza F tra i e j utilizzando la legge di gravitazione di Newton
7:   aggiungi la forza F all'accelerazione di i
8:   sottrai la forza F dall'accelerazione di j
9: Per ogni corpo i nel sistema:
10:  aggiorna la velocità di i utilizzando l'accelerazione
11:  aggiorna la posizione di i utilizzando la velocità
```

Dallo pseudocodice si può studiare la complessità computazionale che è pertanto $O(n^2)$ ed è dovuta principalmente al ciclo interno, in cui si calcola **la forza tra ogni coppia di corpi nel sistema**. Poiché ci sono n corpi nel sistema, e per ogni corpo si deve calcolare la forza con tutti gli altri n-1 corpi, il numero totale di coppie di corpi è $n(n-1)/2$. Ciò significa che il ciclo interno esegue $n(n-1)/2$ operazioni, rendendo la complessità computazionale $O(n^2)$.

L'obiettivo è quello di dimostrare che i tempi di esecuzione valutati durante l'esecuzione indipendente di NBody sui due processori diversi risultino statisticamente differenti al fine di diversificare i due sistemi di elaborazione. Il procedimento che si segue è:

- **Raccolta dei dati:** eseguendo il programma N-Body su entrambi i processori con lo stesso set di dati di input per un numero significativo di volte (ad esempio, almeno 30 volte) e registrare i tempi di esecuzione per ogni esecuzione. Si noti che la raccolta dei dati deve essere indipendente e identicamente distribuita.

- **Valutazioni:** calcolando la media e la deviazione standard dei tempi di esecuzione per entrambi i processori verificando la distribuzione normale.
- **Test statistico:** per confrontare i tempi di esecuzione tra i due processori. Se il test statistico indica che c'è una differenza statisticamente significativa tra i tempi di esecuzione dei due processori, si può affermare che i tempi di esecuzione sui due processori sono diversi.

Schema del processo seguito:

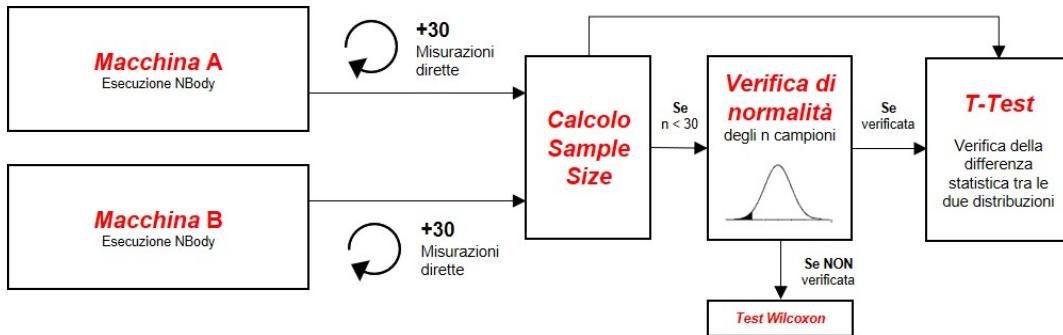


Figura 2.1: Processo applicato per il Benchmarking

2.2 Definizione dei sistemi da confrontare

Per garantire una valutazione più precisa del Benchmark, i SUT (System Under Test) considerati hanno una configurazione Hardware e Software simile tranne per il componente da confrontare: il processore. Pertanto le configurazioni Hardware da confrontare sono le seguenti:

Configurazione macchina A	
CPU	Intel i7-1165G7 2.80 GHz
Core	4
RAM	16 GB 3200 MHz
S.O	Windows 11 Pro
Anno di assemblaggio	2021

Tabella 2.1: Macchina A

Configurazione macchina B	
CPU	Intel i7-7700K 4.20GHz
Core	4
RAM	16 GB 3200 MHz
S.O	Windows 11 Pro
Anno di assemblaggio	2017

Tabella 2.2: Macchina B

La metrica utilizzata per differenziare le due macchine è il **tempo di esecuzione** della simulazione.

2.3 Raccolta dei dati indipendente

La prima fase dell'analisi è la di raccolta dei dati tramite delle operazioni di **misurazione diretta**. La condizione fondamentale da garantire è la indipendenza tra i campioni raccolti utile per applicare il Teorema del Limite Centrale. Esso garantisce infatti che **se si prelevano abbastanza campioni da una distribuzione qualsiasi, la distribuzione della media dei campioni si avvicinerà sempre ad una distribuzione normale**. Pertanto, utilizzando un numero sufficiente di campioni di tempi di esecuzione del programma N-Body, si può garantire che la distribuzione dei tempi di esecuzione sia abbastanza vicina ad una distribuzione normale di media μ e di varianza $\frac{\sigma^2}{\sqrt{n}}$. Si esegue questo passaggio proprio per poter applicare tecniche statistiche standard utili per determinare se esiste una differenza significativa tra i tempi di esecuzione dei due processori. In generale, si prelevano **almeno 30 campioni** per poter applicare il teorema del limite centrale.

2.3.1 Le misurazioni

Per garantire l'indipendenza dei dati, si procede allo spegnimento e alla riaccensione delle macchine per ogni misurazione diretta. Nello specifico si è effettuata tale operazione per trenta volte e ogni volta veniva lanciato il seguente comando:

```
bash ./launch_nbody.sh -r 5 -n ***
```

dove:

- **-r**: indica il numero di ripetizioni per la creazione di un thread
- **-n**: indica il numero di corpi da simulare, nel nostro caso: 10000, 100000, 1000000. (***)

Sono state quindi raccolte 31 misurazioni indipendenti composte di 5 ripetizioni ciascuna. In questo modo sono verificate le ipotesi del teorema del limite centrale, che richiede almeno 30 campioni indipendenti.

Di seguito è riportato un estratto, contenente solo le prime 10 righe, delle osservazioni raccolte dalle due macchine in esame:

Osservazioni Macchina B			Osservazioni Macchina A		
Numero corpi			Numero corpi		
10000	100000	1000000	10000	100000	1000000
2090.4	19185.2	186369.6	2357.4	19589	238638
2231.2	19441.8	186907.2	2744.6	21825.2	200966.6
2079.6	19063.2	196113.2	2238.8	21126.2	194829.4
2402.8	19142.6	186388.6	1881.2	17600.6	189022.2
2083.4	19319	208836.2	1835.8	18357.6	225058
2253	23051	247049.4	2589.6	19084.4	242138.2
2213.6	16473	175375.6	1855.4	17899.4	165417.8
2307.8	19796.8	307517	2683.4	20645.2	192663.2
1992.6	19176.8	192320.8	2613.6	18446	190985.8
2405.8	23623.6	218083.6	2823.8	16636.6	199944

Tabella 2.3: Estratto delle osservazioni del benchmark nbody ottenute dalle due macchine

2.4 Valutazione della dimensione campionaria

Le 31 osservazioni raccolte per ogni numero di corpi costituiscono il precampione, il quale viene utilizzato per stimare la **dimensione campionaria**. Questo valore ci permette di individuare il numero di campioni indipendenti al fine di approssimare media e varianza del campione con quelli della popolazione. Inoltre, si assicura che la differenza $|\bar{x} - \mu|$ sia minore o uguale ad un certo E con un determinato livello di confidenza.

Dato un errore del X% e un intervallo di confidenza del 95%, per calcolare la dimensione campionaria si utilizza la seguente formula:

$$n = \left(\frac{Z_{\frac{\alpha}{2}} * \sigma}{E} \right)^2 \quad (2.1)$$

Essendo che il numero delle osservazioni è maggiore di 30 allora per la valutazione di n si utilizza la deviazione standard campionaria s.

$$n = \left(\frac{Z_{\frac{\alpha}{2}} * s}{E} \right)^2 \quad (2.2)$$

dove

- $Z_{\frac{\alpha}{2}}$ è uguale a 1,96 ottenuto tramite la tabella di riferimento "z-table";
- s è la deviazione standard campionaria;
- E è l'errore massimo commesso nella stima della media della popolazione.

Dunque, innanzitutto, si sono calcolate le statistiche relative alle 31 osservazioni effettuate per ciascuna macchina. I valori ottenuti per la macchina A, sono i seguenti:

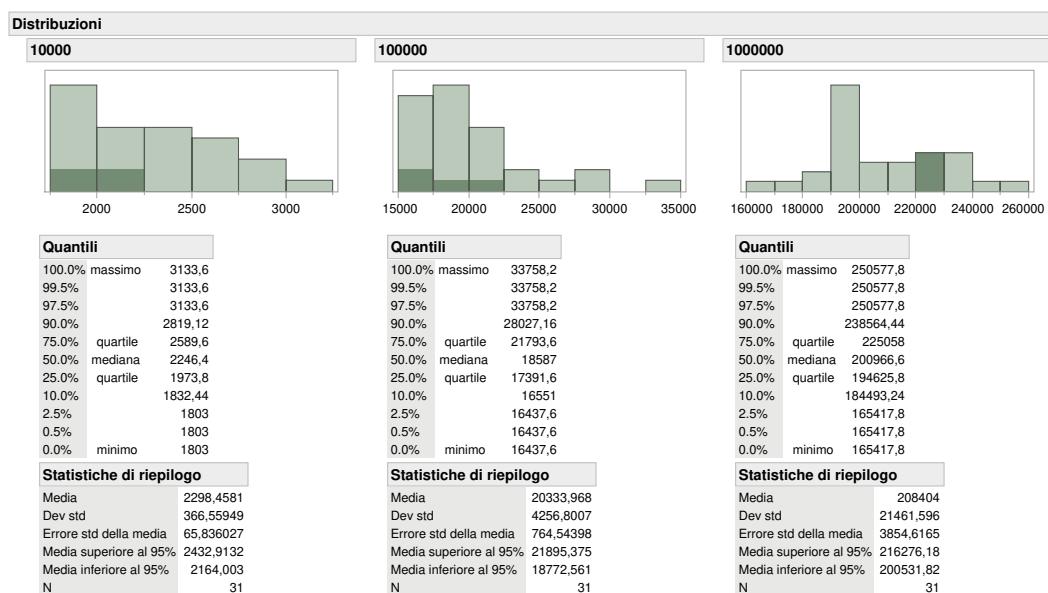


Figura 2.2: Statistiche Macchina A

Si è poi deciso di calcolare le dimensioni campionarie per ogni valore del numero di corpi al variare dell'errore massimo commesso sulla stima (5%, 8%, 10%), di seguito si riporta la tabella con i risultati ottenuti:

Configurazione Macchina A			
Errore	Numero corpi		
	10000	100000	1000000
5%	39	67	16
8%	15	26	6
10%	10	17	4

Tabella 2.4: Valutazione dimensioni campionarie in base all'errore stimato per la macchina A

Allo stesso modo, di quanto appena visto, si è proceduto alla valutazione delle statistiche per la macchina B:

**Figura 2.3:** Statistiche Macchina B

Ed, infine, si sono calcolate anche in questo caso le dimensioni campionarie relative ad ogni numero di corpi valutate nel benchmark, in relazione all'errore:

Configurazione Macchina B			
Errore	Numero corpi		
	10000	100000	1000000
5%	58	30	42
8%	23	12	16
10%	15	8	10

Tabella 2.5: Valutazione dimensioni campionarie in base all'errore stimato per la macchina B

In definitiva, si è scelto un valore di errore massimo commesso nella stima della media della popolazione pari al 10%, in quanto risulta il miglior compromesso tra numero di campioni indipendenti necessari e relativo errore di stima. Inoltre, nel nostro caso possiamo anche scegliere l'8% essendo liberi di effettuare un certo numero di misurazioni, ma nel caso reale in cui la misura risulta essere costosa, avere meno misurazioni è sempre buona norma.

2.5 Test statistico

Per verificare che i campioni delle rispettive macchine provengono dalla stessa o diversa popolazione ci si avvale del test visivo e parametrico.

2.5.1 Test Visivo

Si è provveduto al calcolo degli intervalli di confidenza della media μ delle popolazioni originarie dei tempi di esecuzione considerando anche in questo caso un livello di confidenza del 95%. Il calcolo dell'intervallo di confidenza è:

$$\hat{X} - \frac{\sigma \cdot Z_{\frac{\alpha}{2}}}{\sqrt{n}} \leq \mu \leq \hat{X} + \frac{\sigma \cdot Z_{\frac{\alpha}{2}}}{\sqrt{n}} \quad (2.3)$$

Tuttavia, essendo che il numero dei campioni è ≥ 30 , si considera la deviazione standard campionaria al posto di quella della popolazione, operando l'approssimazione $s \approx \sigma$. Pertanto:

$$\hat{X} - \frac{s \cdot Z_{\frac{\alpha}{2}}}{\sqrt{n}} \leq \mu \leq \hat{X} + \frac{s \cdot Z_{\frac{\alpha}{2}}}{\sqrt{n}} \quad (2.4)$$

Si utilizza JMP per la valutazione degli intervalli di confidenza, in particolare, per la macchina A si hanno i seguenti valori:

Distribuzioni					
10000		100000		1000000	
Intervalli di confidenza					
Parametro	Stima	Cl inferiore	Cl superiore	1-Alfa	
Media	2298,458	2164,003	2432,913	0,950	
Dev std	366,5595	292,9221	489,9703	0,950	
Parametro	Stima	Cl inferiore	Cl superiore	1-Alfa	
Media	20333,97	18772,56	21895,37	0,950	
Dev std	4256,801	3401,66	5689,952	0,950	
Parametro	Stima	Cl inferiore	Cl superiore	1-Alfa	
Media	208404	200531,8	216276,2	0,950	
Dev std	21461,6	17150,22	28687,14	0,950	

Figura 2.4: Intervalli di confidenza Macchina A

Per quanto riguarda la macchina B, si hanno invece i seguenti valori:

Distribuzioni					
10000		100000		1000000	
Intervalli di confidenza					
Parametro	Stima	Cl inferiore	Cl superiore	1-Alfa	
Media	2249,31	2088,804	2409,815	0,950	
Dev std	437,5802	349,6755	584,9018	0,950	
Parametro	Stima	Cl inferiore	Cl superiore	1-Alfa	
Media	20074,55	19253,04	20896,07	0,950	
Dev std	2239,667	1789,745	2993,704	0,950	
Parametro	Stima	Cl inferiore	Cl superiore	1-Alfa	
Media	201634,2	189357,7	213910,8	0,950	
Dev std	33469,13	26745,58	44737,29	0,950	

Figura 2.5: Intervalli di confidenza Macchina B

Come si può notare dalla Figura 2.6, sia le medie che gli intervalli di confidenza delle due macchine in corrispondenza del medesimo numero di corpi si sovrappongono, quindi, **questo test visivo ci suggerisce che i due campioni provengono dalla stessa popolazione d'origine**.

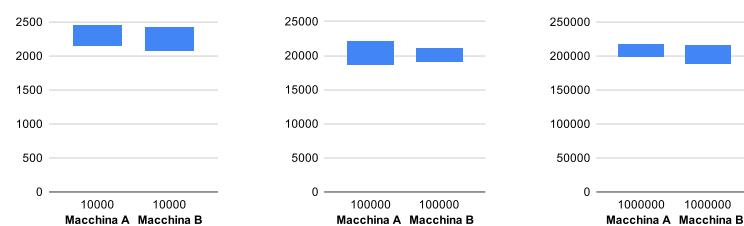


Figura 2.6: Test visivo sovrapposizione degli intervalli di confidenza

2.5.2 Test Parametrico

Per avere un'ulteriore conferma dei risultati ottenuti, si è deciso di eseguire un test di ipotesi sulla differenza delle medie dei due campioni, al fine di determinare se la differenza tra di essi è statisticamente significativa o meno, in altre parole, si vuole capire se i due campioni provengono o meno dalla stessa popolazione d'origine.

Essendo valide le ipotesi del teorema del limite centrale, in quanto sono state raccolte almeno 30 osservazioni, è possibile applicare un test di tipo parametrico. Inoltre, essendo le deviazioni standard delle popolazioni sconosciute, e le dimensioni campionarie sempre inferiori a 30, è necessario applicare un t-test. Dal momento che i sistemi considerati sono indipendenti, non esiste alcuna corrispondenza tra le osservazioni delle coppie di campioni, i quali in alcuni casi presentano addirittura **dimensioni campionarie diverse**. Per questo motivo, si è deciso di effettuare un **unpaired t-test** per ogni coppia di osservazioni relative al medesimo numero di corpi.

In particolare, per la scelta del test d'ipotesi da usare si è fatto uso della seguente Tabella riassuntiva:

Test d'ipotesi	Condizioni necessarie
One-sample z-test	(Normal population or n large) and σ known.
Two-sample z-test	Normal population and independent observations and σ_1 and σ_2 are known
One-sample t-test	(Normal population or n large) and σ unknown
Paired t-test	(Normal population of differences or n large) and σ unknown
Two-sample t-test (pooled)	(Normal pop. or $n_1 + n_2 > 40$) and independent observations and $\sigma_1 = \sigma_2$ unknown
Two-sample t-test (unpooled)	(Normal pop. or $n_1 + n_2 > 40$) and independent observations and $\sigma_1 \neq \sigma_2$ both unknown

Tabella 2.6: Test d'ipotesi

Tenendo conto delle suddette condizioni per il caso in esame, la scelta del test d'ipotesi da usare ricade sul **Two-sample t-test**, al fine di individuare quale degli ultimi due occorre applicare, va verificata innanzitutto la normalità delle popolazioni individuate dalle dimensioni campionarie calcolate nelle Tabelle 2.4 e 2.5 in corrispondenza di $E=10\%$; successivamente si verifica l'omoschedasticità delle suddette popolazioni con un test visivo e parametrico.

2.5.2.1 Normalità

Il primo passo da eseguire è la verifica della normalità delle due popolazioni individuate dalle dimensioni campionarie in corrispondenza di ciascun numero di corpi.

Si inizia dalla verifica della normalità per un numero di corpi pari a 10000, i QQ-Plot ottenuti sono i seguenti:

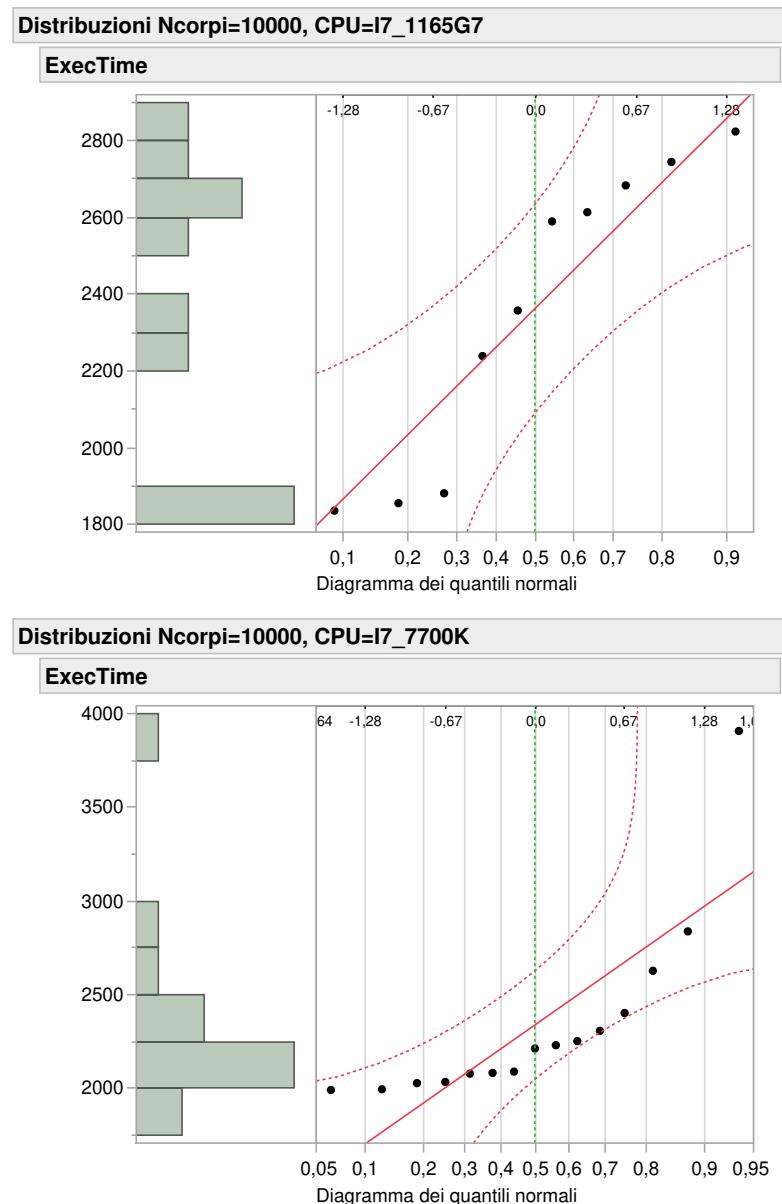


Figura 2.7: Distribuzione per 10000 corpi relative alle due macchine

Si noti come entrambe le popolazioni risultino normali dall'analisi visiva.

Successivamente, si passa ad analizzare la normalità per le popolazioni in esame in corrispondenza di un numero di corpi pari a 100000:

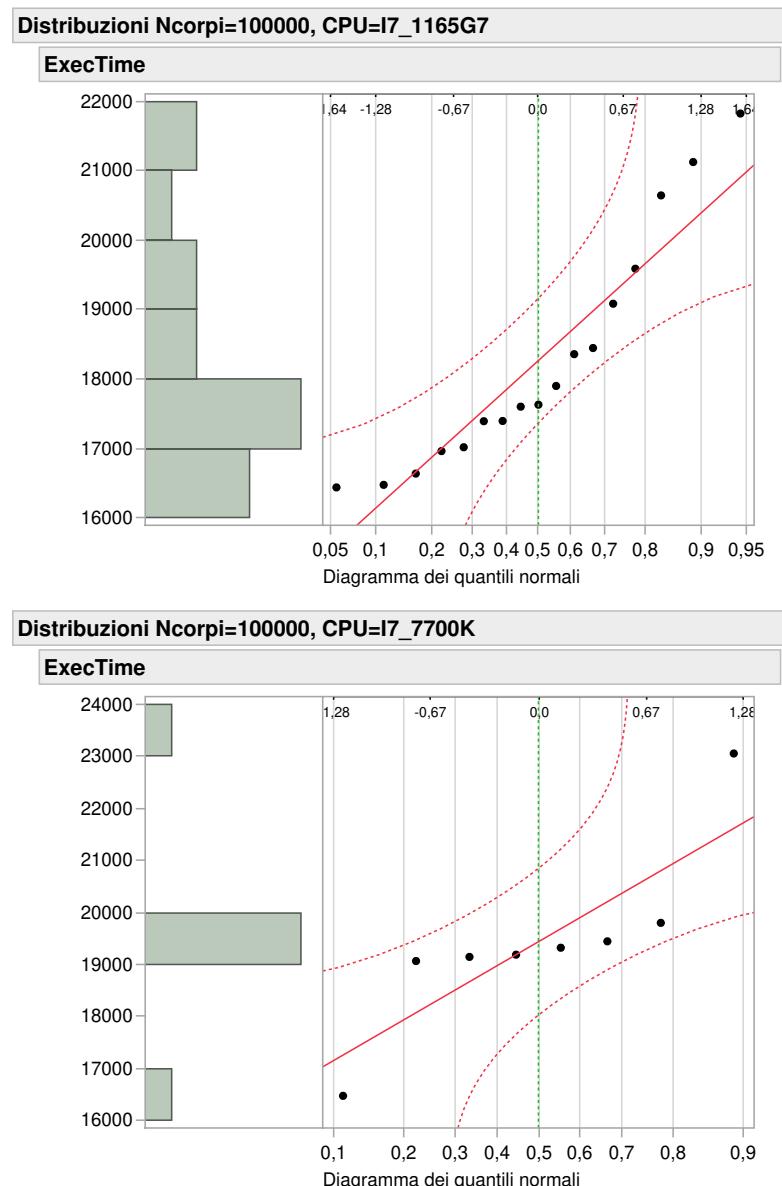


Figura 2.8: Distribuzione per 100000 corpi relative alle due macchine

Anche in questo caso, l'analisi visiva ci suggerisce che le popolazioni sono distribuite normalmente.

Infine, si verifica la normalità delle due popolazioni per un numero di corpi pari ad 1000000:

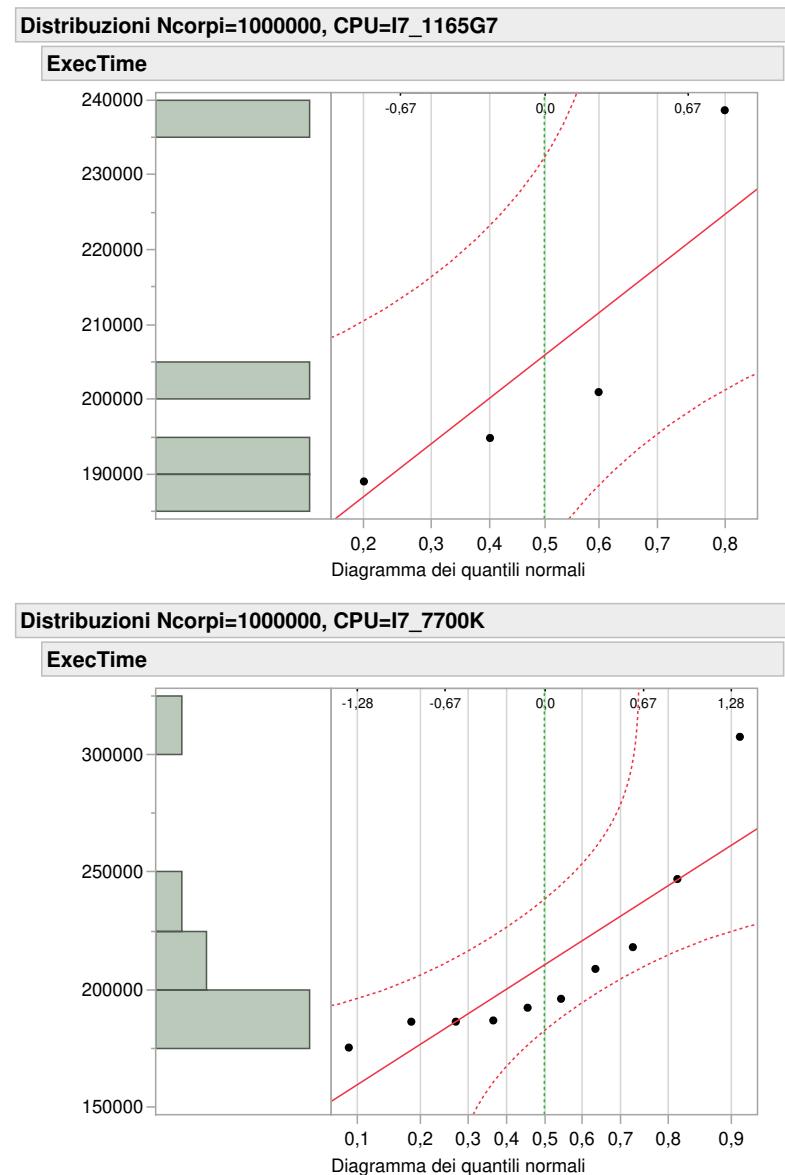


Figura 2.9: Distribuzione per 1000000 corpi relative alle due macchine

Dal **test visivo** si può affermare che entrambi campioni seguono una distribuzione normale.

In definitiva, il test di normalità è verificato per ciascuna coppia di popolazioni si può, quindi, passare alla fase successiva.

2.5.2.2 Omoschedasticità

Per la verifica dell'omoschedasticità, ovvero per determinare se i campioni abbiano la stessa varianza o meno, si è fatto uso del test di **omoschedasticità** fornito dal tool JMP. In particolare, si è usato sia un test visivo che di tipo parametrico per tale compito, formulando come ipotesi nulla (H_0 : le varianze dei due campioni siano uguali), se il p-value ottenuto risulta alto allora l'ipotesi non è rigettata, il che suggerisce che i campioni sono omoschedastici.

Si è proceduto, dunque, al test di omoschedasticità in corrispondenza dei campioni caratterizzato da un numero di corpi pari a 10000, come mostrato di seguito:

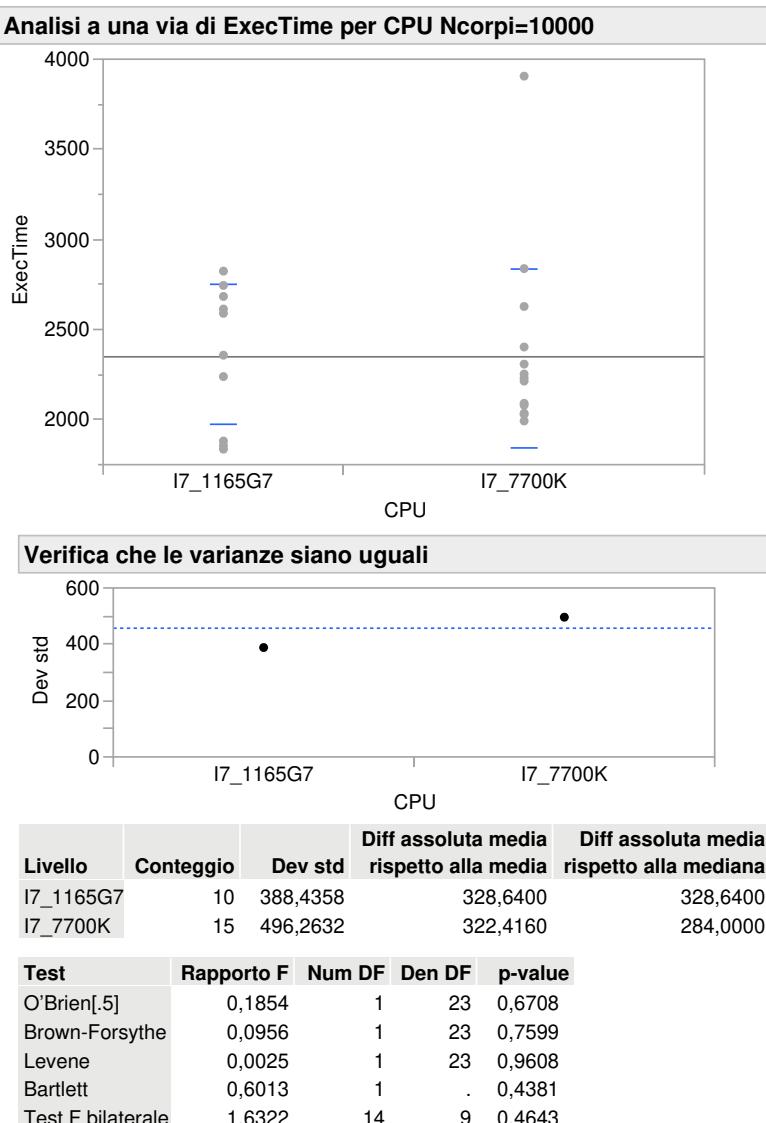


Figura 2.10: Test sulle varianze uguali per numero di corpi 10000

Il test visivo ci suggerisce che i due campioni hanno la stessa varianza. Si nota infatti dalla Figura 2.10 che le deviazioni standard sono vicine, anche i test parametrici confermano tale osservazione. Si nota infatti i test di O'Brien, Levene, Bartlett che hanno tutti un p-value alto. **Si può dunque concludere che i campioni sono omoschedastici.**

Quindi per il campione con numero di corpi 10000 la distribuzione della macchina A e della macchina B risultano normali e omoschedastici.

Si passa, dunque, all'analisi dell'omoschedasticità per campioni con numero di corpi pari a 100000, di seguito i risultati ottenuti:

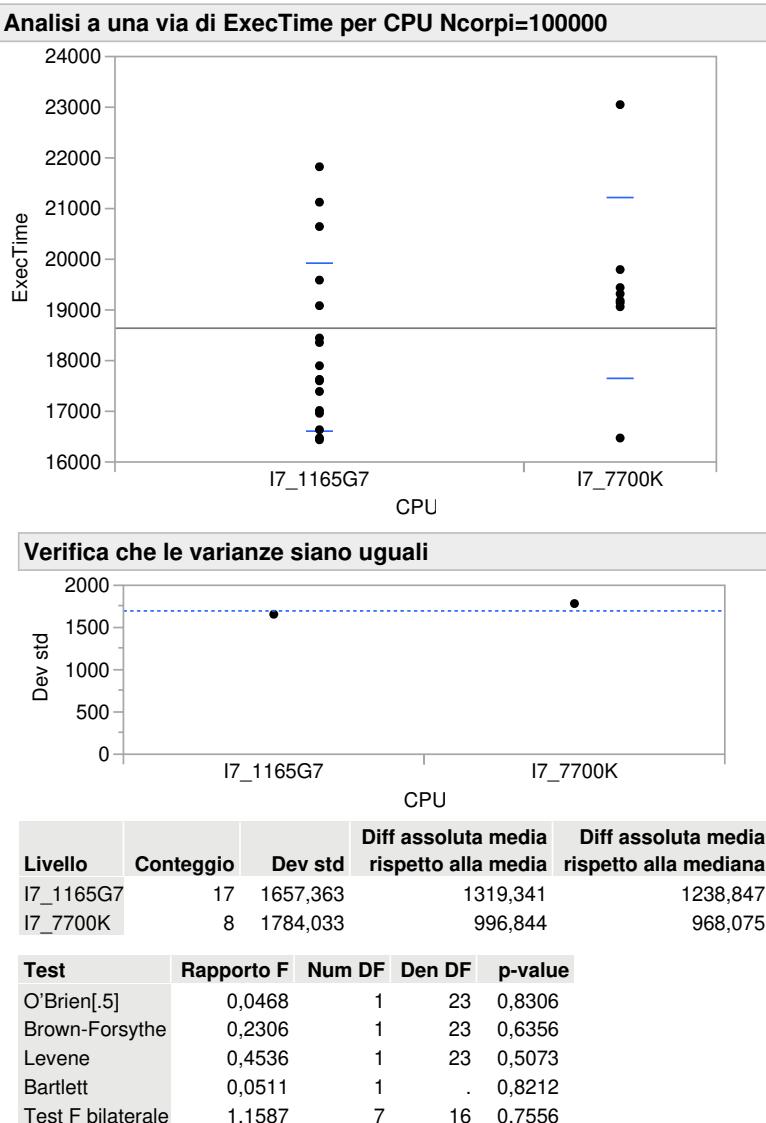


Figura 2.11: Test sulle varianze uguali per numero di corpi 100000

Anche in questo caso, dal test visivo si nota che i campioni hanno la stessa varianza ed i test parametrici lo confermano. Quindi per il campione con numero di corpi 100000 la distribuzione della macchina A e della macchina B risultano normali e omoschedastici.

Infine, si è verificata l'omoschedasticità per i campioni in corrispondenza di un numero di corpi pari a 1000000, come mostrato di seguito:

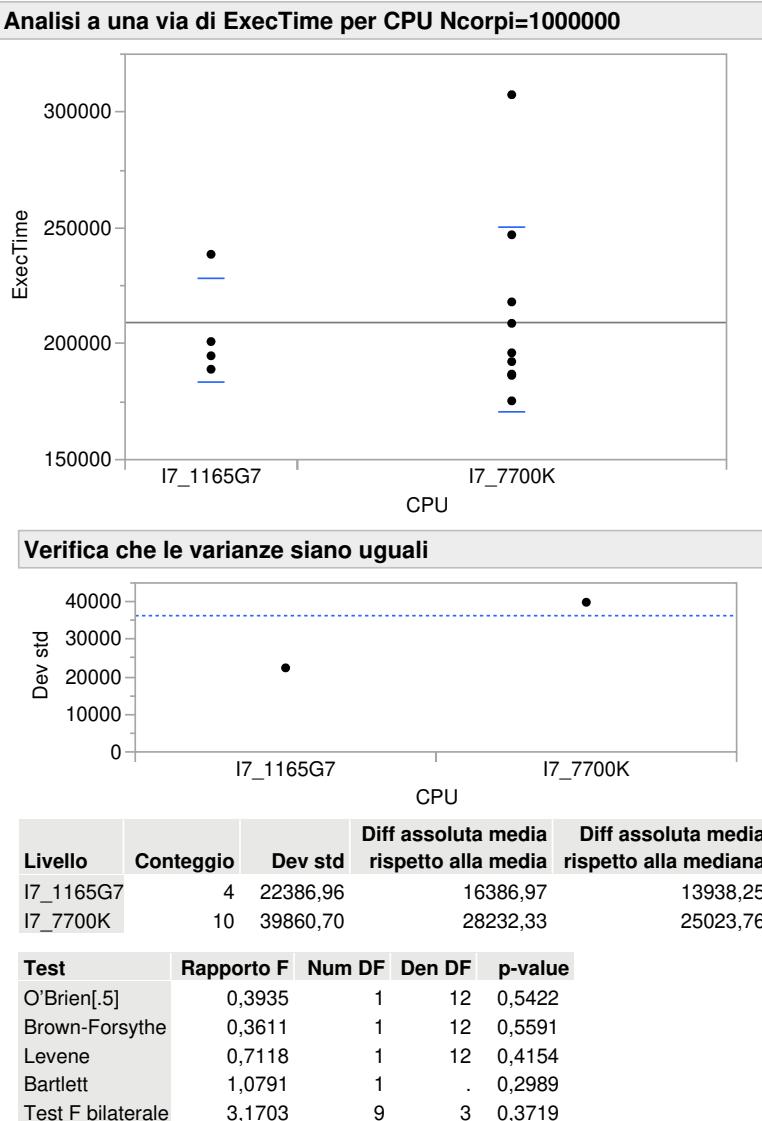


Figura 2.12: Test sulle varianze uguali per numero di corpi 1000000

L'analisi visiva suggerisce che i campioni sono omoschedastici, allo stesso modo anche i test parametrici rigettano l'ipotesi nulla ed affermano che i campioni abbiano la stessa varianza.

Da quest'ultima osservazione possiamo, dunque, confermare che il test d'ipotesi da eseguire sia il **Two-sample t-test (pooled)**.

2.5.2.3 Two-sample t-test (pooled)

Individuato il test d'ipotesi da eseguire, ovvero il **Two-sample t-test (pooled)**, si può passare alla sua esecuzione per ogni coppia di numero di corpi scelta per il benchmark, in cui la statistica t è ottenuta come segue:

$$t = \frac{\bar{X}_1 - \bar{X}_2}{s_p \sqrt{\frac{1}{n_1} + \frac{1}{n_2}}} \quad (2.5)$$

dove:

- \bar{X}_1 e \bar{X}_2 sono le medie campionarie dei due campioni;
- n_1 ed n_2 sono le dimensioni campionarie dei due campioni;
- $s_p = \sqrt{\frac{(n_1-1)s_{X_1}^2 + (n_2-1)s_{X_2}^2}{n_1+n_2-2}}$ è uno stimatore della *pooled standard deviation* dei due campioni;
- s_{X_1} e s_{X_2} sono le deviazioni standard dei due campioni;
- $n_1 + n_2 - 2$ rappresenta il numero di gradi di libertà.

Di fatto il test consiste nel calcolare la statistica t per poi accedere in base al suo valore e al numero di gradi di libertà in una tabella contenente i valori del p-value corrispondente. Individuato il p-value, lo si confronta con il livello di significatività α , pari in questo caso a 0.05. Se p è maggiore ad α allora l'ipotesi nulla non viene rigettata, ovvero si afferma che i due campioni sono estratti dalla stessa popolazione d'origine.

Anche questo test è stato effettuato tramite il tool JMP, in particolare, si è applicato tale test per ogni coppia di numero di corpi scelta per il benchmark.

H_0 : le due distribuzioni provengono dalla stessa popolazione

I risultati del test ottenuti per un numero di corpi pari a 10000, sono i seguenti:

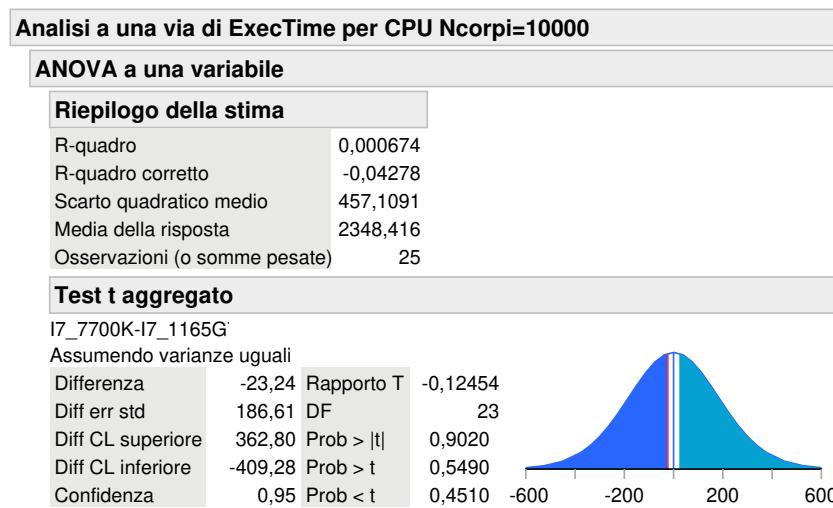


Figura 2.13: Two-sample t-test per 10000 corpi

Dalla Figura 2.13, si può notare come l'ipotesi non venga rigettata confermando quanto detto nel test visivo, ovvero che i due campioni provengono dalla stessa popolazione d'origine.

Si è proceduto ad effettuare lo stesso test visto in precedenza, anche sui campioni relativi ad un numero di corpi pari a 100000, ottenendo quanto segue:

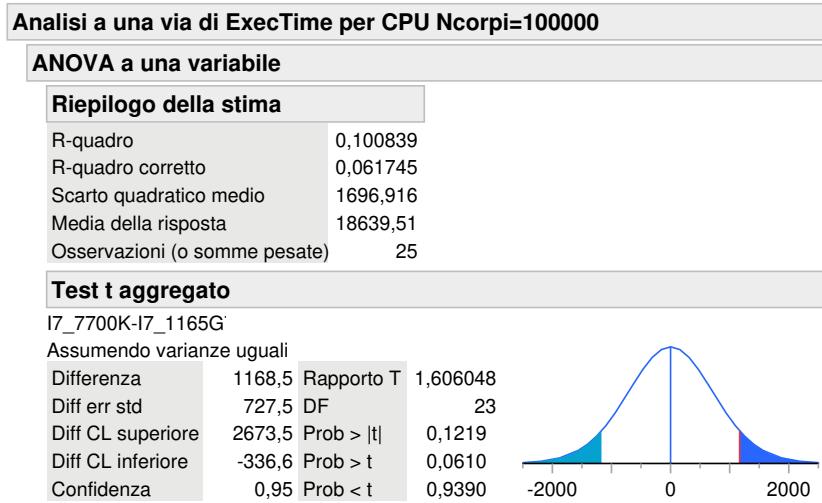


Figura 2.14: Two-sample t-test per 100000 corpi

Anche questo test parametrico, non rigetta l'ipotesi nulla e conferma quanto detto con l'analisi visiva, cioè che i campioni sono estratti dalla stessa popolazione d'origine.

Infine, si è condotto il test anche sui campioni con un numero di corpi uguale a 1000000, con i seguenti risultati:



Figura 2.15: Two-sample t-test per 1000000 corpi

L'ipotesi nulla non viene rigettata, nemmeno in questo caso, affermando che i campioni provengono dalla stessa popolazione d'origine, in accordo con il test visivo.

2.6 Risultati e considerazioni del confronto

Analizzando i dati ottenuti dal confronto delle due macchine, si può affermare, che da un punto di vista prestazionale le due sono praticamente equivalenti.

I motivi dietro tale risultato sono principalmente imputabili a 2 fattori non sottovalutabili in quest'analisi. Il primo fattore è quello **generazionale**, infatti, si sono confrontati due processori I7 di generazioni diverse; quello della macchina A è un processore di 11ma generazione a bassa potenza basato sull'architettura Tiger Lake-U. Tale processore ha una velocità massima di clock di 4,7 GHz ed una cache di 12 MB, il che lo rende più performante rispetto al processore della macchina B che appartiene alla 7ma generazione basato sull'architettura Kaby Lake, con velocità massima di clock di 4,2 GHz ed una cache di 8 MB.

Tuttavia, il secondo fattore è di **tipo architettonico**, alla base proprio di tali processori, infatti la macchina A ha un processore mobile che di conseguenza mira ad avere un consumo di energia inferiore a scapito un po' delle prestazioni. Mentre, la macchina B è un PC fisso che, dunque, monta un processore desktop con un consumo di energia maggiore rispetto al precedente con un conseguente incremento delle prestazioni computazionali della macchina stessa.

Dunque, l'equilibrio che si è riscontrato dall'analisi va cercato nel bilanciarsi di questi due fattori enunciati e non in altri aspetti.

3 Workload Characterization

Contenuti

3.1	Definizione degli obiettivi e overview	18
3.2	Raccolta e comprensione dei dati	19
3.3	Preprocessing dei dati	20
3.3.1	Analisi preliminare dei dati	20
3.3.2	Valutazione delle colonne costanti	21
3.3.3	Valutazione delle colonne correlate	22
3.3.4	Valutazione degli outliers	22
3.3.5	Standardizzazione dei dati	23
3.4	Esecuzione della Principal Component Analysis (PCA)	23
3.5	Esecuzione del Clustering	24
3.6	Analisi di sensitività della devianza persa al variare di PC e Cluster	27
3.6.1	Calcolo devianza del workload reale (pre-PCA)	27
3.6.2	Calcolo devianza post-PCA	27
3.6.3	Calcolo devianza post-CLUSTERING	28
3.6.4	Calcolo della percentuale della perdita di devianza	29
3.7	Report definitivo del Workload Characterization	30

3.1 Definizione degli obiettivi e overview

Sia dato un workload **reale**, caratterizzarne uno **sintetico** rispettando i seguenti requisiti:

- **Valutare** il miglior trade-off tra numero di componenti principale e numero di cluster.
- **Analisi di sensitività**: studiare come varia la varianza persa con le differenti componenti principali e numero di cluster.

Il processo di *Workload Characterization* ha lo scopo di generare un workload sintetico per un determinato caso di test a partire da un workload reale. Le tecniche di analisi dei dati che verranno utilizzati durante il processo di workload characterization sono: La **principal component analysis (PCA)** e **il clustering**. In breve, la PCA è una tecnica di riduzione della dimensionalità che mira a trasformare un insieme di dati multidimensionale in un insieme di dati a **dimensione più bassa mantenendo la maggior parte dell'informazione presente nei dati originali**. Il clustering, d'altra parte, è una tecnica di apprendimento non supervisionato che mira a raggruppare gli elementi dei dati in base alle loro "similitudini", creando così dei cluster.

Le fasi del processo di workload characterization in esame è:

- **1. Raccolta dei dati**: consiste nella raccolta dei dati sull'utilizzo delle risorse del sistema in esame o dell'applicazione.
- **2. Pulizia e preparazione dei dati (Preprocessing dei dati)**: include una possibile eliminazione di dati mancanti o errati, la normalizzazione dei dati, la selezione delle caratteristiche rilevanti, lo studio degli outlier.
- **3. Esecuzione della PCA**: si esegue la PCA per ridurre la dimensionalità dei dati.
- **4. Esecuzione del clustering**: si utilizza il clustering per raggruppare gli elementi dei dati in base alle loro similitudini.

- **5. Analisi dei risultati:** sarà necessario analizzare i risultati ottenuti dalla PCA e dal clustering per comprendere le caratteristiche del workload e identificare eventuali problemi o opportunità di ottimizzazione.
- **6. Reporting:** è importante presentare i risultati dell'analisi in modo da poter prendere decisioni informate su come gestire il workload.

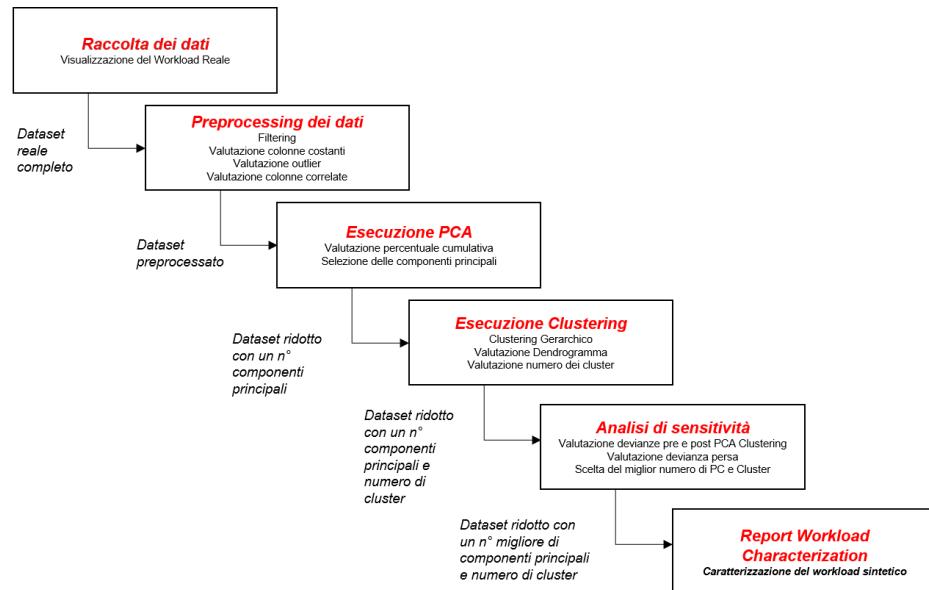


Figura 3.1: Processo applicato per il Workload Characterization

Le valutazioni e i calcoli che verranno fatti nel seguente paragrafo sono stati fatti con **JMP** e **Matlab**.

3.2 Raccolta e comprensione dei dati

Il workload reale è composto da 24 *feature* (colonne del dataset) e da 4995 *osservazioni* circa (righe del dataset) relative all'utilizzo di memoria da parte dei processi eseguiti su una distribuzione Linux e raccolti tramite il filesystem virtuale "proc" gestito dal kernel Linux. Per analizzare da un punto di vista analitico i dati ed effettuare le scelte appropriate, si prova a descrivere le componenti del workload reale in esame.

- **VmPeak:** picco della memoria virtuale;
- **VmSize:** dimensione della memoria virtuale;
- **VmHWM:** picco della porzione di memoria virtuale occupata da un processo ("high water mark");
- **VmRSS:** dimensione della porzione di memoria virtuale occupata dal processo;
- **VmPTE:** dimensione della tabella di paginazione della memoria virtuale;
- **Threads:** numero di thread nel processo;
- **MemFree:** valore di memoria libera, derivante dalla somma di due quantità;
- **Buffers:** archiviazione temporanea per blocchi di dischi grezzi che non dovrebbero diventare grandi (20 MB circa);

- **Cached**: quantità di cache per la lettura di file da disco;
- **Active**: memoria che è stata usata più recentemente e di solito non viene recuperata a meno che non sia assolutamente necessario;
- **Inactive**: memoria che è stata usata meno di recente. È più idonea ad essere recuperata per altri scopi;
- **Dirty**: memoria in attesa di essere riscritta;
- **Writeback**: dati attivamente riscritti in memoria;
- **AnonPages**: Pagine non supportate da file mappate in tabelle di pagina nello spazio utente;
- **Mapped**: file mappati in memoria, come ad esempio librerie;
- **Slab**: memoria usata dal kernel sotto forma di strutture dati cache per uso esclusivo;
- **PageTables**: quantità di memoria dedicata al livello più basso di tabelle di pagina.
- **CommittedAS**: quantità di memoria attualmente allocata sul sistema. La memoria impegnata è una somma di tutta la memoria che è stata allocata dai processi, anche se essi non l'hanno ancora "usata";
- **NumOfAllocFH**: descrizione non disponibile;
- **proc-fd**: descrizione non disponibile;
- **avgThroughput**: throughput medio applicato sul sistema;
- **avgElapsed**: tempo medio di risposta delle richieste;
- **avgLatency**: latenza media misurata;
- **Errors**: numero di errori incorsi.

3.3 Preprocessing dei dati

Il preprocessing dei dati è importante nel workload characterization perché consente di rimuovere i dati non necessari, di trasformare i dati in un formato utilizzabile per gli strumenti di analisi e di correggere eventuali errori o incongruenze nei dati. Ciò permette di ottenere una rappresentazione precisa e completa del workload, facilitando l'identificazione delle caratteristiche chiave e l'ottimizzazione delle prestazioni del sistema. Si elencano i procedimenti di preparazione dei dati **applicati** in questo processo di workload characterization:

3.3.1 Analisi preliminare dei dati

La prima analisi che si prova ad effettuare è verificare se ci sono delle colonne *ridondanti* o *superflue*. Ad impatto, dalla descrizione, le colonne di **VmPeak** e **VmHWM** sono dei valori di picco delle n-esime osservazioni e sono entrambe collegate alla colonna di **VmRSS**, pertanto risulta inutile includerle nella fase successiva del processo, in quanto non esprimono informazioni aggiuntive rispetto a quelle già fornite con la colonna **VmRSS**. Infatti, se si considerano i valori di picco in istanti di tempo specifici tali valori rappresentano niente altro che il valore massimo assunto dalla colonna **VmRSS** tra un campionamento ed il successivo; a riprova di quanto appena detto anche nella matrice di correlazione tali colonne presentano dei valori prossimi all'1, e per questo si è confermata la scelta di eliminare queste colonne dai dati.

Considerazione sui dati duplicati

Durante la fase di analisi dei dati si è notato la presenza di numerose righe duplicate, in particolare, dopo la riga 3000 si hanno circa 2000 righe ridondanti. Approfondendo tale studio, si è riscontrato che le righe univocamente definite sono solo 3030, in questo caso però si è scelto di non eliminare tali dati in quanto nel caso di un workload eseguito su base temporale anche la ripetizione di righe è un'informazione utile ai fini dell'individuazione di una particolare caratterizzazione dei dati. Ad esempio, si è notato come le righe 93, 94, 95 e 96 sono esattamente uguali rispettivamente alle righe 3003, 3004, 3005 e 3006; questo potrebbe indicare che in tali **istanti temporali vengono eseguite delle particolari routine del processore e dunque, eliminarle, significherebbe perdere tale informazione.**

Altra analisi da fare è che osservando le prime righe del dataset, in particolare le prime 85 righe, hanno dei valori che sono differenti dai valori che si presentano nell'intero dataset; questo potrebbe rappresentare uno stato di fase transitoria di avvio del sistema. Se si analizzano i valori di memoria libera (memfree), le prime righe hanno dei valori decisamente più alti rispetto ai valori presenti successivamente. Inoltre, se si analizza il throughput (AvgThroughput) in corrispondenza delle prime righe risulta essere basso rispetto alle successive righe. Pertanto, queste considerazioni risultano essere indice del fatto che il sistema sia in fase transitoria di avvio del sistema. Se si avessero dei limiti a livello di costi per l'analisi sarebbe consigliabile andare ad eliminare tali righe del dataset, ma per tale analisi **si è deciso di tenere in conto anche queste istanze, per avere un workload sintetico che rappresenta meglio il dataset completo**, anche perché il numero delle righe che appartengono alla fase transitoria sono poche rispetto alle dimensioni totali del dataset.

Le dimensioni del dataset dopo questo procedimento sono 22 colonne per 4995 righe

3.3.2 Valutazione delle colonne costanti

Eliminare le colonne costanti è buona norma poiché non forniscono alcuna informazione utile (variabilità nulla), quindi tenerle può distorcere i risultati dell'analisi e rendere più difficile ottenere conclusioni significative. Le colonne costanti da scartare sono: **AnonPages, avgLatency, Errors**. Dalla Figura 3.2 è possibile vedere come le distribuzioni siano costanti:

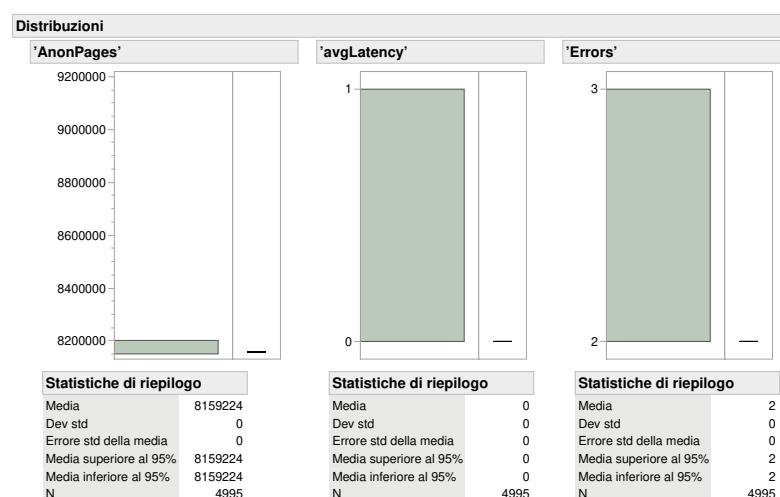


Figura 3.2: Distribuzioni delle colonne costanti scelte

Le dimensioni del dataset dopo questo procedimento sono **19** colonne per **4995** righe

3.3.3 Valutazione delle colonne correlate

Per la valutazione delle colonne correlate si fa uso della matrice di correlazione valutata tramite il tool JMP.

	Multivariato																			
	Correlazioni																			
'VmSize'	1,0000	0,9628	0,9806	0,6143	-0,4892	0,6376	0,3386	-0,0021	0,7456	0,2118	-0,4992	-0,0526	-0,0361	0,9991	0,0672	0,4261	0,5105	0,9666	-0,4250	
'VmRSS'	0,9628	1,0000	0,9416	0,5660	-0,3733	0,5484	0,2250	-0,0039	0,6400	0,1721	-0,3733	-0,0439	-0,0090	0,9488	0,0705	0,2981	0,5180	0,9113	-0,3905	
'VmPTE'	0,9906	0,9416	1,0000	0,6324	-0,4145	0,6179	0,2541	-0,0014	0,6995	0,1958	-0,4145	-0,0491	-0,0087	0,8910	0,0753	0,3525	0,5155	0,9640	-0,4437	
'Threads'	0,6143	0,5660	0,6324	1,0000	-0,2254	0,4707	0,0847	0,0125	0,4577	0,0512	-0,2254	-0,0234	-0,0013	0,5442	0,1034	0,2188	0,3488	0,6373	-0,3569	
'MemFree'	-0,4892	-0,3733	-0,4145	-0,2254	1,0000	-0,7243	-0,9681	-0,0007	-0,9064	-0,9645	1,0000	0,0615	0,0053	-0,3348	-0,0173	-0,9648	-0,1066	-0,4897	0,1354	
'Buffers'	0,6376	0,5484	0,6179	0,4707	-0,7243	1,0000	0,5320	-0,0116	0,8927	0,5554	-0,7243	-0,1290	-0,0339	0,4969	0,1065	0,6364	0,2221	0,6362	-0,2983	
'Cached'	0,3386	0,2250	0,2541	0,0847	-0,9681	0,5320	1,0000	0,0046	0,7825	0,9920	-0,9681	-0,0468	-0,0019	0,1908	-0,0380	0,9797	0,0171	0,3376	-0,0470	
'Active'	-0,0021	-0,0039	-0,0014	0,0125	-0,0007	-0,0116	0,0046	1,0000	-0,0058	0,0042	-0,0007	-0,0018	-0,0005	-0,0040	-0,0063	0,0050	-0,0038	-0,0004	0,0105	
'Inactive'	0,7456	0,6400	0,6995	0,4577	-0,9064	0,8927	0,7825	-0,0056	1,0000	0,7631	1,0000	-0,9645	-0,0821	-0,0243	0,1280	-0,0696	0,9704	0,3069	0,7533	-0,2894
'Dirty'	0,2818	0,1721	0,1958	0,0512	-0,9645	0,5554	0,9920	0,0042	0,7631	1,0000	-0,9645	-0,0821	-0,0243	0,1280	-0,0696	0,9704	-0,0344	0,2756	-0,0248	
'Writeback'	-0,4892	-0,3733	-0,4145	-0,2254	1,0000	-0,7243	-0,9681	-0,0007	-0,9064	-0,9645	1,0000	0,0615	0,0053	-0,3348	-0,0173	-0,9648	-0,1066	-0,4897	0,1354	
'Mapped'	-0,0526	-0,0439	-0,0491	-0,0234	0,0615	-0,1290	-0,0468	-0,0019	-0,0285	-0,0821	1,0000	0,4209	0,1885	0,7718	-0,0252	0,4869	0,0789	0,0806		
'Slab'	-0,0081	-0,0090	-0,0050	0,0052	-0,0007	-0,0006	-0,0001	-0,0001	-0,0001	-0,0001	0,4209	0,1885	0,0166	0,0007	-0,0005	0,0004	0,0004			
'PageTables'	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000		
'Committed_AS'	0,0972	0,0704	0,0753	0,0734	-0,0173	0,1065	-0,0380	-0,0063	0,1456	0,1096	-0,0173	0,7718	0,4407	0,3705	1,0000	0,0287	0,5679	0,2463	0,0207	
'NumOfAllocFH'	0,4261	0,2981	0,3525	0,2188	-0,9848	0,6364	0,9797	0,0050	0,8547	0,9704	-0,9848	-0,0252	0,0048	0,2759	0,0000	0,0000	0,0000	0,0000	-0,1120	
'proc-fd'	0,5105	0,5180	0,5155	0,3488	-0,1066	0,2221	0,0171	-0,0032	0,3069	-0,0344	-0,1066	0,4869	0,1859	0,6597	0,5679	0,1008	0,0000	0,5915	-0,1956	
'avgThroughput'	0,9666	0,9113	0,9640	0,6373	-0,4897	0,6362	0,3376	-0,0004	0,7533	0,2756	-0,4897	0,0789	0,0827	0,9179	0,2463	0,4413	0,5915	0,1000	-0,4100	
'avgElapsed'	-0,4250	-0,3905	-0,4437	-0,3569	0,1354	-0,2983	-0,0470	0,0105	-0,2894	-0,0248	0,1354	0,0806	0,0344	-0,3488	0,0207	-0,1120	-0,1955	-0,4100	1,0000	

Le correlazioni sono stimate per metodo A livello di riga.

Figura 3.3: Matrice di correlazione

Dalla Figura 3.3 si può notare che tra le due componenti **WriteBack** e **MemFree** c'è una perfetta correlazione, infatti, il valore di correlazione tra tali colonne è pari ad 1. Pertanto sarà possibile eliminare una delle due colonne, in particolare, si è deciso di eliminare la colonna **WriteBack**.

Le dimensioni del dataset dopo questo procedimento sono **18** colonne per **4995** righe

3.3.4 Valutazione degli outliers

È possibile valutare gli outliers del dataset attraverso il **box-plot**.

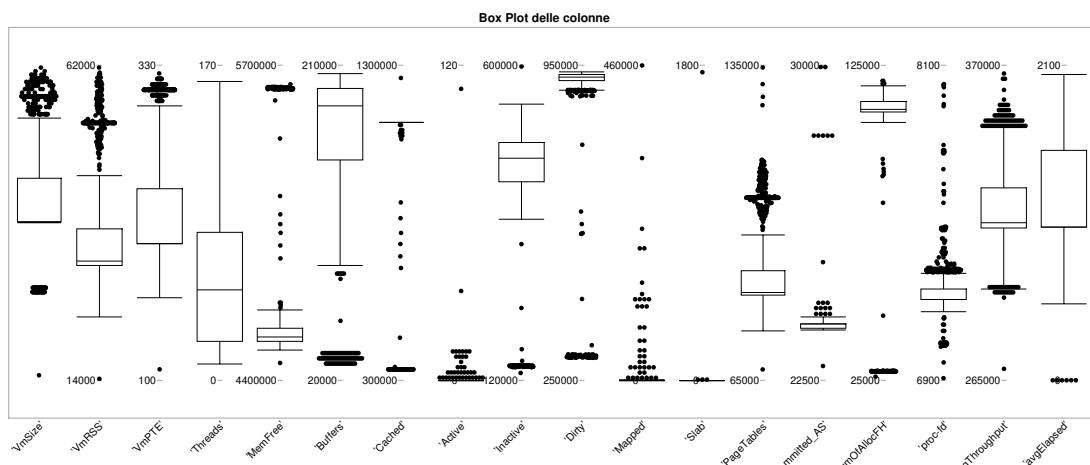


Figura 3.4: Box Plot delle colonne

Dalla Figura 3.4, si può notare come alcune colonne presentino un comportamento quasi costante, in particolare, le colonne in questione sono **Slab** e **Active**; infatti dal box plot di **Slab** possiamo vedere che a meno di alcune righe (4 per la precisione su un totale di 4995 righe) il valore che tale colonna assume è sempre 0, per tale motivo possiamo considerare tale colonna come costante e per questo eliminarla. Allo stesso modo, anche la colonna **Active** presenta solo 39 valori diversi da zero e per questo è stata esclusa dall'analisi in quanto

non fornisce alcuna varianza ai dati, come si nota anche dal valore della sua deviazione standard molto basso.

Le dimensioni del dataset dopo questo procedimento sono **16 colonne per 4995 righe**

3.3.5 Standardizzazione dei dati

Fare la normalizzazione dei dati prima di un'analisi di componenti principali (PCA) è importante perché la PCA è un metodo di riduzione della dimensionalità che si basa sulla varianza dei dati. **Se i valori di una colonna hanno una scala molto diversa rispetto a quelli di un'altra colonna, potrebbero avere un peso maggiore nell'analisi rispetto a quelli che invece dovrebbero avere un peso minore.** Fare la normalizzazione dei dati prima della PCA consente di evitare questo problema, poiché trasforma tutti i valori in modo che abbiano la stessa scala. Ciò significa che ogni colonna avrà lo stesso peso nell'analisi e che i risultati saranno più attendibili. Inoltre, fare la normalizzazione dei dati prima della PCA può anche rendere l'analisi più veloce e facile da interpretare, poiché i valori saranno tutti nello stesso range. È possibile utilizzare la normalizzazione **z-score** per ogni singola osservazione.

$$z = \frac{x - \bar{x}}{\sigma} \quad (3.1)$$

Essendo che, su JMP si utilizza la PCA calcolata sulle correlazioni, tale normalizzazione dei dati viene effettuata automaticamente dal tool.

3.4 Esecuzione della Principal Component Analysis (PCA)

La PCA (Principal Component Analysis, o Analisi delle Componenti Principali) è un metodo di riduzione della dimensionalità utilizzato per analizzare e visualizzare dati multidimensionali. L'obiettivo della PCA è quello di individuare le direzioni di varianza massima nei dati e di proiettare i dati su un nuovo insieme di coordinate ortogonali, chiamate componenti principali. La prima componente principale, descrive la maggior parte della varianza; viceversa, l'ennesima descrive la varianza minore. Tale trasformazione è utile perché riusciamo a capire quante e quali sono le componenti tramite cui riduciamo la dimensionalità del problema tenendo traccia della varianza persa. Con il tool JMP si effettua l'analisi:

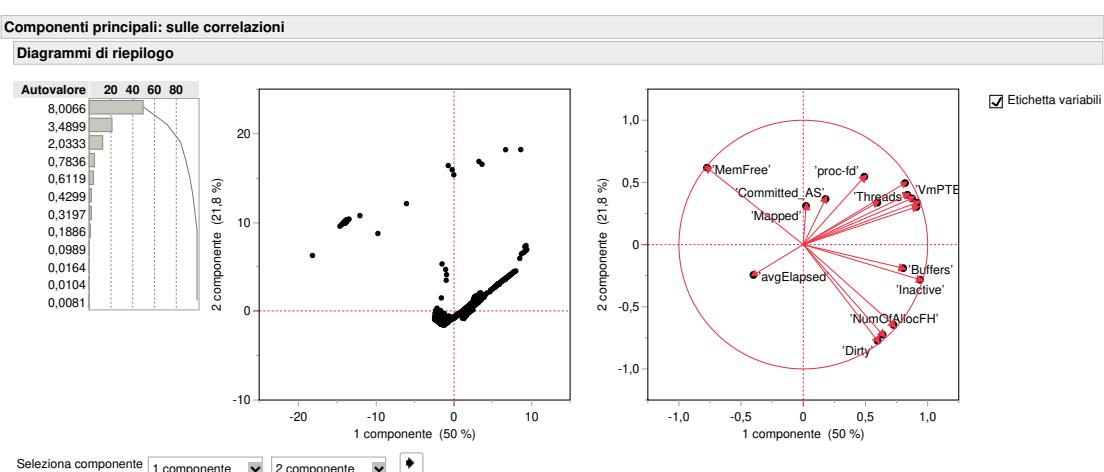


Figura 3.5: Principal Component Analysis

In particolare, si può effettuare la scelta delle componenti principali, andando a visualizzare la varianza spiegata da ogni componente, nonché quella cumulativa. Dunque, dalla Figura 3.6, la scelta **inizialmente** ricade su 6 componenti principali, il quale riesce a conservare addirittura una varianza del $96\% \approx$; tuttavia, in scenari reali è spesso richiesto di rispettare dei budget economici e dunque, anche la scelta di 4 o 5 componenti principali risulta soddisfacente (nei successivi paragrafi si andranno a valutare anche i casi con 4 e 5 componenti principali).

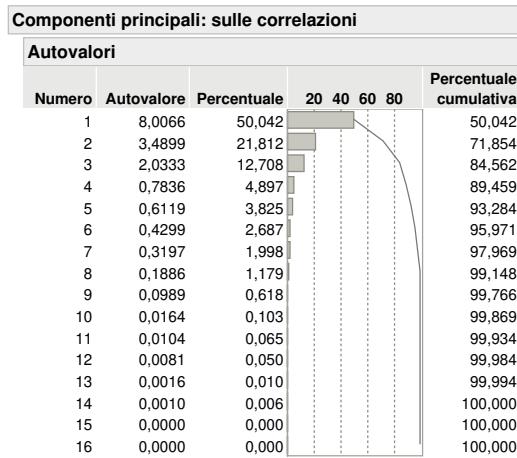


Figura 3.6: Autovalori

3.5 Esecuzione del Clustering

È possibile ridurre il numero delle osservazioni del dataset usando anche la tecnica del clustering. In particolare, nel caso in esame, si utilizza un *clustering gerarchico* utilizzando un approccio *agglomerativo* detto di **metodo di Ward**. Si è scelta questa tecnica perché utilizza la varianza all'interno dei cluster per determinare come unirne due. Tuttavia, è sensibile alle dimensioni dei cluster iniziali e può essere influenzato da outlier o valori anomali, per questo è importante fare un buono studio degli outlier stessi. Lo **scopo** è quello di minimizzare la variabilità interna ad ogni cluster (**devianza intracluster**) e massimizzare quella esterna (**devianza intercluster**). Si valuta il **dendrogramma**, utile per capire come i punti del dataset sono stati raggruppati durante l'analisi di clustering e per decidere il numero ottimale di cluster da utilizzare.

Inoltre, attraverso la cronologia di clusterizzazione è possibile visualizzare come diminuisce la distanza all'aumentare del numero di cluster. Tale tabella è utile per determinare in maniera rapida il numero di cluster, infatti, si sceglie il numero di cluster che **massimizza la distanza tra l'aggiunta e/o la rimozione di un ulteriore cluster da esso**. Questo si può vedere piazzando la distanza e il numero di cluster valutato con JMP e scegliendo opportunamente il numero intorno al "ginocchio" del plot (analisi di sensitività).

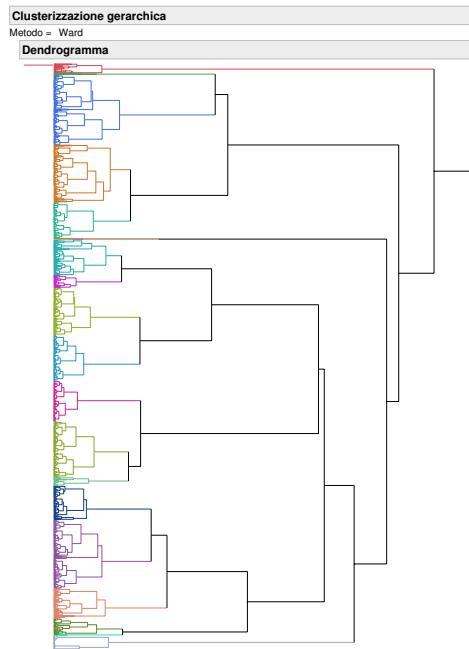


Figura 3.7: Dendrogramma

Cronologia di clusterizzazione			
Numero di cluster	Distanza	Leader	Subordinato
1	70,12488564	1	84
2	63,60625118	84	85
3	61,38306473	85	550
4	55,38363850	550	2885
5	49,87073678	550	1862
6	48,80344921	550	1082
7	35,69217330	1862	2736
8	32,06618708	84	96
9	29,74689664	84	109
10	29,08560103	550	1079
11	20,84754397	1862	1863
12	19,38948349	85	90
13	17,93448521	1862	1931
14	15,98490697	1082	1083
15	15,86486785	1079	1081
16	14,12255197	96	253
17	13,74979357	1083	1091
18	12,65940809	2736	2844
19	12,47555056	550	1926
20	12,09867249	109	111
21	10,38497691	96	102
22	10,03773794	2885	2939
23	9,58406175	550	1929

Figura 3.8: Cronologia di clusterizzazione

Dalla cronologia di clusterizzazione sarà possibile, dunque, costruire il grafico in cui viene posto in ordinata il numero di cluster ed in ascissa la distanza:

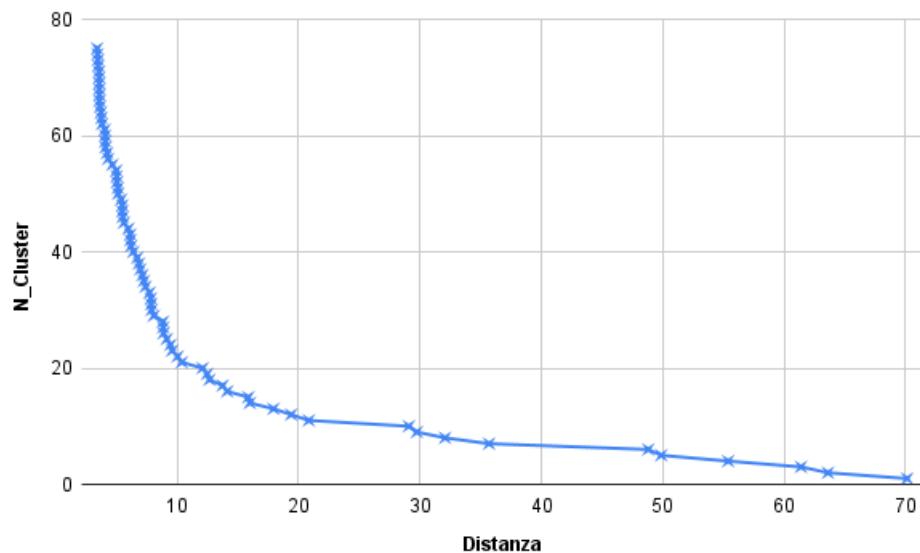


Figura 3.9: Analisi di sensitività per la scelta del numero di cluster

Inizialmente si è scelto, il numero di cluster pari a 10 poi si valuterà anche 15 e 20 per un'analisi e scelta più accurata. Si può andare a verificare la distribuzione delle istanze all'interno dei cluster tramite un istogramma definito sulla base del conteggio delle istanze appartenenti ad un determinato cluster.

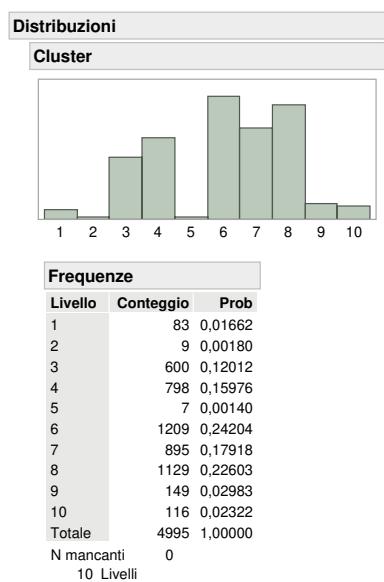


Figura 3.10: Distribuzione delle istanze nei cluster

3.6 Analisi di sensitività della devianza persa al variare di PC e Cluster

Dopo aver fissato il numero delle componenti principali (6) e dei cluster (10), si è deciso di andare ad analizzare la devianza dei dati che si sono analizzati e che si sta cercando di conservare andandoli a confrontare con le ulteriori possibilità di scelta delle ambedue componenti. Tale analisi prende il nome di **analisi di sensitività, in quanto si cerca di valutare gli effetti prodotti da variazioni di "variabili d'ingresso", in questo caso le PC ed il numero di cluster, sulla "variabile di risposta", la devianza persa, nell'ambito di un dato modello.** L'obiettivo è ridurre la devianza **persa** dopo il processo di PCA e Clustering.

Si valutano le devianze ¹ del processo di Workload Characterization:

3.6.1 Calcolo devianza del workload reale (pre-PCA)

Si valuta la devianza **totale** dei dati del workload reale prima di effettuare il processo di PCA:

$$\text{Devianza}_{\text{tot}} = \sum_{i=1}^N \|x_i - \bar{x}\|^2 = 79904 \quad (3.2)$$

dove:

- x_i : rappresenta l'i-esima istanza del dataset;
- \bar{x} : la media delle istanze;
- N: il numero totale delle istanze.

3.6.2 Calcolo devianza post-PCA

A valle dell'operazione di PCA, la devianza dei dati d'origine è cambiata, dal momento che la PCA attua una trasformazione di stato. Infatti, con **6 componenti principali** la devianza è:

$$\text{Devianza}_{\text{post-PCA}} = \sum_{i=1}^N \|x_{\text{PCA}_i} - \bar{x}_{\text{PCA}}\|^2 = 76685 \quad (3.3)$$

dove:

- x_{PCA_i} : rappresenta l'i-esima istanza del dataset ridotto dalla PCA;
- \bar{x}_{PCA} : la media delle istanze del dataset ridotto dalla PCA;
- N: il numero totale delle istanze.

Per verificarne la **correttezza**, il rapporto tra la devianza pre-PCA e la quantità calcolata dopo la PCA deve essere pari alla varianza che si è scelto di preservare a monte. In particolare, nel caso in esame in cui si è scelto di conservare 6 componenti principali è (95,97% di *Percentuale cumulativa*), si ha infatti:

$$\frac{\text{Devianza}_{\text{post-PCA}}}{\text{Devianza}_{\text{tot}}} = 0,9597 \simeq 96\% \quad (3.4)$$

Si valutano le devianze post-PCA con lo **stesso procedimento** anche per i componenti principali 4 e 5 su Matlab. Si mostrano le percentuali valutate che restano simili alle percentuali cumulative a quelle calcolate su JMP 3.1:

¹ La valutazione delle devianze post PCA e Clustering è stata fatta con uno script MatLab presentato durante il corso.

Percentuale devianza Post-PCA		
Numero componenti principali		
4	5	6
89.46%	93.28%	95.97%

Tabella 3.1: Devianza Post-PCA calcolata dallo script Matlab

3.6.3 Calcolo devianza post-CLUSTERING

Anche le operazioni di clustering hanno contribuito alla modifica della devianza dei dati. Per i cluster però vanno considerati due contributi di devianza: quella **intra-cluster** e quella **inter-cluster**, che identificano rispettivamente la devianza tra i dati di uno stesso cluster e tra i dati appartenenti a cluster differenti. La devianza totale dei dati dopo la clusterizzazione è calcolabile attraverso la somma di tali contributi:

$$\text{Devianza}_{\text{intra}} = \sum_{k=1}^K \sum_{i=1}^N \|x_i - \bar{x}_k\| = 8899 \quad (3.5)$$

$$\text{Devianza}_{\text{inter}} = \sum_{k=1}^K N_k \|\bar{x}_k - \bar{x}\| = 67786 \quad (3.6)$$

dove:

- K : il numero dei cluster;
- N_k : il numero di istanze che appartengono al cluster k ;
- x_i rappresenta l' i -esima istanza del dataset;
- \bar{x}_k : istanza centroide del cluster k ;
- \bar{x} : media di tutte le istanze del dataset;

La somma di tali contributi è pari alla devianza preservata dopo l'analisi PCA, infatti andando a scrivere l'equazione si ha che:

$$\text{Devianza}_{\text{post-PCA}} = \text{Devianza}_{\text{intra}} + \text{Devianza}_{\text{inter}} = 8899 + 67786 = 76685 \quad (3.7)$$

Coincide infatti con la 3.3.

Si valutano le devianze post-PCA anche per i componenti principali 4 e 5 con 10,15 e 20 cluster rispettivamente nella Tabella 3.2. I risultati:

Deviance post PCA and Clustering				
Numero componenti principali				
	4	5	6	
Numero di cluster	10	83.83%	83.61%	84.83%
	15	86.08%	88.15%	90.11%
	20	87.04%	89.51%	91.21%

Tabella 3.2: Devianza Post - **PCA and Clustering**

Nota bene: Si è valutato soltanto 10,15 e 20 cluster perchè se si prova a valutare un numero di cluster inferiore a 10 (es. 5) i risultati sono scadenti, mentre 20 è il numero di cluster selezionato da JMP e che risulta essere il

migliore secondo il tool. Si costruisce, inoltre, un plot in funzione del numero di cluster e devianza post processo di PCA e Cluster per ogni componente principale: Dalla Figura 3.11 si nota che fissato il numero di cluster e

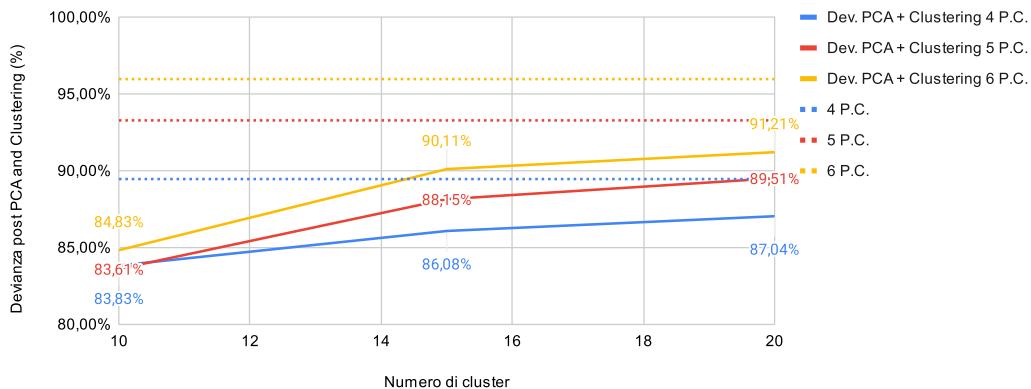


Figura 3.11: Caratteristica Devianza post-PCA Clustering e Numero di Cluster

aumentando il numero di componenti principali la devianza spiegata aumenta e si avvicina sempre più a quella totale esprimibile con la sola operazione di PCA (linea tratteggiata). In maniera analoga, fissato il numero di componenti e aumentando il numero di cluster c'è un aumento della devianza spiegata.

3.6.4 Calcolo della percentuale della perdita di devianza

La valutazione della varianza persa con 6 componenti principali e 10 cluster è possibile calcolarla come:

$$\text{Devianza}_{loss} = (1 - \text{Devianza}_{post-PCA}) + \text{Devianza}_{intra} * \text{Devianza}_{post-PCA} = 0,1517 \quad (3.8)$$

oppure

$$\text{Devianza}_{loss} = (1 - \text{Devianza}_{inter}) * \text{Devianza}_{post-PCA} = 0,1517 \quad (3.9)$$

Pertanto, a valle delle operazioni di clustering e di analisi delle componenti principali, si ottiene un workload che esprime circa il 15% di devianza in meno rispetto al workload reale. Si è analizzato, la devianza persa post-PCA e Clustering anche con 4 e 5 componenti principali e 10,15 e 20 cluster. I risultati:

		Total Lost Deviance		
		Numero componenti principali		
		4	5	6
Numero di cluster	10	16.17%	16.39%	15.17%
	15	13.92%	11.85%	9.89%
	20	12.96%	10.49%	8.79%

Tabella 3.3: Percentuale perdita della devianza

Si valuta anche la Relative Lost Deviance tra la devianza persa post-PCA e devianza persa post-PCA e Clustering. (Es. con 15 Cluster e 5 PC si è perso un ulteriore 5% di devianza dopo il procedimento di Cluster)

Si estende la valutazione plottando la **devianza persa rispetto al numero delle componenti principali**.

Dopo un'attenta analisi, per assicurare il miglior trade-off tra devianza persa e riduzione della dimensione del workload **si è scelto di caratterizzare il workload con 5 componenti principali e 15 cluster in quanto si ha solo**

Relative Lost Deviance				
Numero di cluster	Numero componenti principali			
	4	5	6	
10	5.63%	9.67%	11.14%	
15	3.38%	5.13%	5.86%	
20	2.42%	3.77%	4.76%	

Tabella 3.4: Relative Lost Deviance

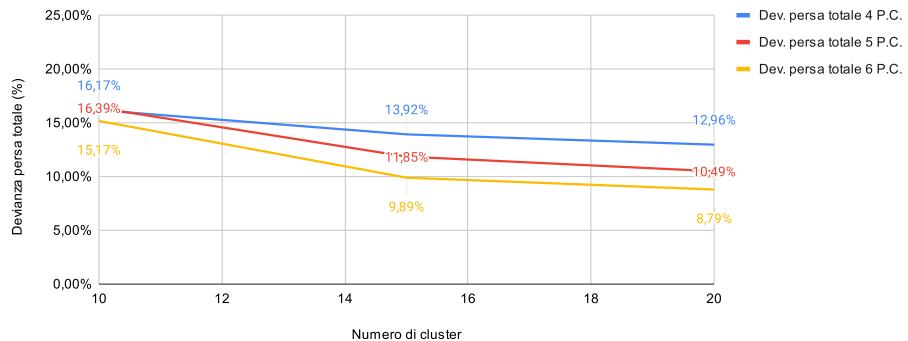


Figura 3.12: Caratteristica Devianza persa e Numero di Cluster

l'11,85% di devianza persa. Si è scelto questo trade-off anche perché la dimensionalità del workload sintetico rispetto a 6 componenti principali e 10 cluster non cambia di molto (75 osservazioni totali vs 60 osservazioni totali con 6 p.c. e 10 cluster) oltre ad una perdita di devianza maggiore (15,17% vs 11,85%). **Pertanto da una devianza totale del 93,28% si passa all' 88,15% di devianza.** Inoltre si è scelto questo trade-off in quanto nella tabella 3.4 la percentuale di devianza persa dopo il procedimento di cluster è minore rispetto al trade off 15 cluster e 6 componenti principali.

3.7 Report definitivo del Workload Characterization

A valle di tutte le operazioni di analisi del dataset e le scelte fatte, è ora necessario identificare e caratterizzare il workload sintetico. Sempre con il tool JMP è possibile caratterizzare l'istanza di ciascun cluster o scegliendolo randomicamente oppure attraverso il calcolo della media (centroide virtuale). Nel caso in esame, però, si hanno dati di tipo continuo pertanto è **preferibile caratterizzare i cluster con i centroidi reali poiché essi rappresentano esattamente i dati all'interno del cluster. Di seguito i centroidi virtuali valutati da JMP:** Tab. 3.5.

A partire dai centroidi virtuali si è quindi proceduto a calcolare la distanza euclidea di tutte le istanze, per poi identificare l'istanza che risultasse più vicina. Il calcolo eseguito è tramite Excel attraverso la function:

$$=RADQ(SOMMA.Q.DIFF(Col.PrincipalComponent;Col.CentroidVirtual)).$$

Questo procedimento ha prodotto il workload sintetico definitivo avente 15 righe e 24 colonne: Fig 3.13

Cluster	Conteggio	Principale1	Principale2	Principale3	Principale4	Principale5
1	83	-13.8866	10.01395	-2.68563	-0.11896	0.670666
2	6	0.011054	15.53034	23.30521	-0.75409	1.266307
3	2	7.67356	18.18836	45.13913	-4.83004	-0.04664
4	8	-1.08275	4.401703	10.53499	-0.62458	-2.29316
5	525	-1.95504	-0.74012	0.263401	0.861629	-0.51529
6	114	-1.24905	-1.16138	0.695816	1.89493	0.732239
7	236	1.720795	0.227113	-0.03611	0.959125	1.057456
8	527	-1.26054	-1.09162	0.432695	0.631491	0.041727
9	527	-0.84991	-1.05241	0.413929	0.44516	0.650424
10	851	-1.39481	-0.73322	0.001185	-0.56812	-0.79926
11	442	-0.66271	-0.90153	0.152585	-0.8109	0.043537
12	393	2.158978	0.616202	-0.62648	-1.58257	-0.06899
13	1016	1.960654	0.460764	-0.28585	-0.31198	0.5125
14	138	6.558175	3.584931	-1.36027	1.096998	0.146506
15	127	5.945601	3.297304	-0.7153	1.54607	-2.5588

Tabella 3.5: Tabella dei centroidi virtuali (calcolati attraverso la media di ciascun cluster)

Cluster	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
VmPeak ^t	152272	153484	170344	170344	172388	172392	185964	172392	172392	172392	172392	189652	189652	215352	21535
VmSize ^t	150216	151436	170340	170340	170340	170340	181104	170340	170340	170340	170340	183532	183564	206136	20613
VmHWM ^t	25228	26612	31144	31208	32696	32868	41280	32868	33228	32696	33228	41320	41968	61784	61784
VmRSS ^t	25208	26472	31140	31140	31460	32092	38956	32004	32316	31516	32316	36816	37280	54968	53204
VmPTE ^t	160	160	200	200	200	200	232	200	200	200	200	244	248	304	312
Threads ^t	34	14	43	13	23	27	96	23	47	28	44	97	94	161	9
MemFree	5604888	5008324	4471580	4709868	4638948	4596512	4561764	4596416	458339	4620552	4581928	4560260	4558912	4534204	45393
Buffers ^t	33144	83852	94884	97944	128876	169816	193268	169600	181464	146336	182292	196336	197200	204140	20462
Cached ^t	334652	767660	1259568	1075684	1117176	1117504	1117952	1117492	111774	1117292	1117764	1118044	1118076	1118392	11184
Active ^t	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Inactive ^t	141436	380596	597900	376648	411924	438216	478104	437876	452956	420872	454308	482032	484516	521076	52110
Dirty ^t	303848	597076	891332	882056	917932	933536	924412	933560	930924	926616	930408	921492	920376	908736	90754
Writeback	5604888	5008324	4471580	4709868	4638948	4596512	4561764	4596416	458339	4620552	4581928	4560260	4558912	4534204	45393
AnonPage	8159224	8159224	8159224	8159224	8159224	8159224	8159224	8159224	8159224	8159224	8159224	8159224	8159224	8159224	8159224
Mapped ^t	36	108968	459632	119972	36	152	20	16	152	148	156	44	36	156	136
Slab ^t	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
PageTable	77472	126120	134504	85176	83804	84464	91288	84208	84660	83860	84660	89012	89620	107304	10554
Committe	23772	28324	29956	24272	23740	23844	23848	23844	23844	23740	23844	23848	23848	23848	23848
NumOfAll	27832	92116	116216	109568	109452	110516	113904	110516	110768	109876	110892	114004	114136	112884	11020
proc-fd ^t	7196	7984	8024	7704	7208	7204	7240	7208	7208	7208	7204	7248	7256	7312	7320
avgThroughput	296468	344944	365692	317992	315704	319300	328840	315200	314880	316264	315168	329396	328236	346996	34986
avgElapsed	1530	2040	1530	1530	1530	2040	1530	1530	1530	1020	1020	510	1020	1020	1020
avgLatency	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Errors ^t	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2

Figura 3.13: Workload Characterization definitivo - Colonne x Righe

4 Web Server

Contenuti

4.1	Definizione degli obiettivi per l'analisi di un Web Server	32
4.1.1	System Under Test	32
4.2	Capacity Test	34
4.2.1	Preparazione	35
4.2.2	Pianificazione	35
4.2.3	Esecuzione	37
4.2.4	Analisi dei dati	37
4.2.5	Report e Conclusioni	40
4.3	Workload Characterization	42
4.3.1	Creazione del workload reale (ipotetico)	42
4.3.2	Workload Characterization HL	43
4.3.3	Workload Characterization LL	49
4.3.4	Applicazione del workload HL_c e creazione del LL'_c	53
4.3.5	Workload Characterization LL'	53
4.3.6	Verifica della bontà della caratterizzazione del workload: Confronto tra LL_c e LL'_c	56
4.4	Design of Experiments	58
4.4.1	Configurazione del piano di DOE	59
4.4.2	Analisi dell'importanza	60
4.4.3	Analisi di significatività statistica dei fattori	62

4.1 Definizione degli obiettivi per l'analisi di un Web Server

Si effettua **l'analisi prestazionale** di un Web Server attraverso le seguenti fasi:

● **Capacity Test:**

- Valutazione Throughput e Response Time
- Valutazione Knee Capacity e Usable Capacity

● **Workload Characterization:**

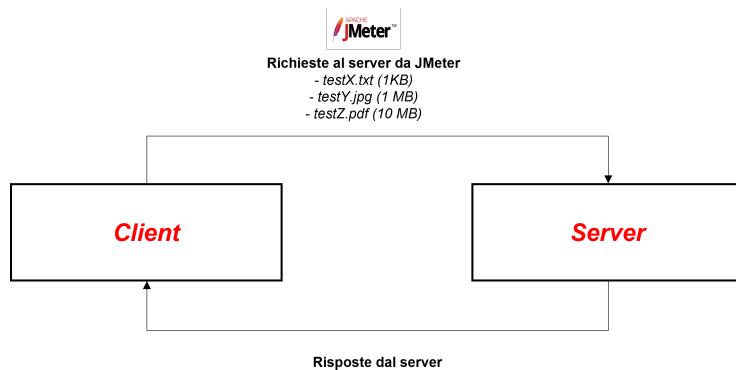
- Valutazione del Workload Characterization ad alto livello
- Valutazione del Workload Characterization a basso livello
- Validazione del Workload Characterization a basso livello con test d'ipotesi

● **Design of Experiment:**

- Studio dell'influenza dei vari parametri sui tempi di risposta

4.1.1 System Under Test

Per svolgere l'esperimento si è scelto di utilizzare la **stessa** macchina fisica per svolgere sia il ruolo di Client che di Server. Il Server è installato sulla macchina virtuale (S.O: Ubuntu virtualizzato tramite VirtualBox - Server: Apache2). Dopo aver installato il server da terminale tramite il comando “`sudo apt-get install apache2`”, è stata aggiunta un'ulteriore scheda di rete “Host-only”, per consentire allo stesso server di essere raggiungibile dall'host (Client).

**Figura 4.1: Risorse sul Server**

Configurazione del Server

Il SUT è un server Linux eseguito su macchina virtuale (Oracle VirtualBox), in particolare si è installato il **Web Server Apache 2.4** (Component Under Study (CUS)).

Configurazione Server (VM)	
CPU	Intel I7 1165G7 2.80GHz
Core	2
RAM	512 MB
S.O	Ubuntu 22.04 64-bit (Virtual Machine)
Memoria Video	16 MB
Scheda Grafica	VMSVGA
Scheda di Rete	Intel PRO/1000 MT Desktop (Scheda "Host-Only")

Tabella 4.1: Configurazione lato - **Server**

Il Server, ovviamente, ha le risorse necessarie per il funzionamento corretto di un servizio web alle quali un generico Client vi accede tramite richieste HTTP. Si è voluto organizzare le risorse in dimensioni differenti per simulare le ipotetiche richieste del Client al Server. In particolare, suddividendoli in 4 file per ogni peso: 100 KB, 1 MB, 10 MB come indicato nella seguente Tabella e nella Figura 4.1.

Risorse sul server			
N. di Risorse	Risorsa	Tipologia	Dimensione
4	testX.txt	Testo	100 KB
4	testY.pdf	Documento	1 MB
4	testZ.jpg	Multimedia	10 MB

Tabella 4.2: Risorse considerate - File fittizi

Si mostra come sono distribuiti i file nella directory del Server (Figura 4.2):

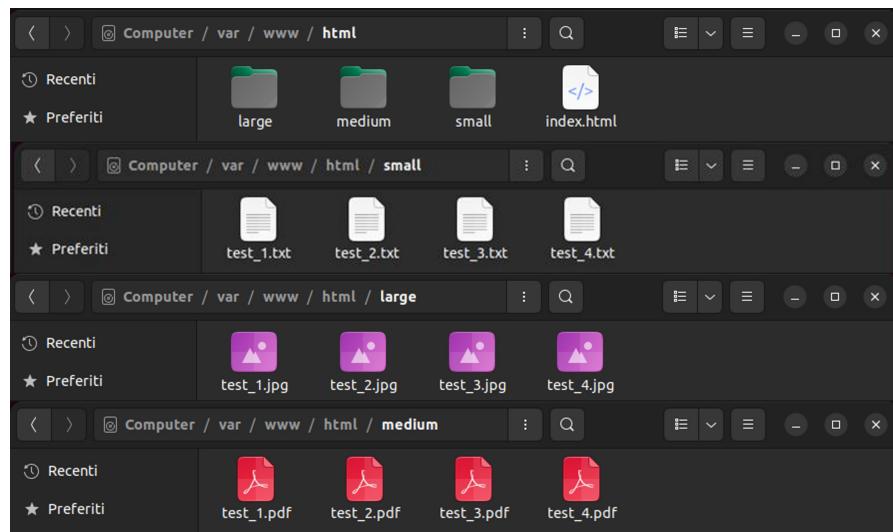


Figura 4.2: Risorse sul Server

Le richieste HTTP sono state generate dal Client utilizzando l'applicativo *Apache JMeter 5.5*.

Configurazione Client	
CPU	Intel I7 1165G7 2.80 GHz
Core	2
RAM	16 GB
S.O	Windows 11 Pro
Memoria Video	2 GB
Scheda Grafica	NVidia GeForce MX450
Scheda di Rete	Intel(R) Wi-Fi 6 AX201 160MHz

Tabella 4.3: Configurazione lato - Client

4.2 Capacity Test

Un capacity test è una forma di test di performance che mira a **determinare la capacità massima di un sistema (web server) di gestire un carico di lavoro elevato senza compromettere le prestazioni**. Il test consiste nell'invio di un gran numero di richieste al server in un breve periodo di tempo, al fine di determinare il punto in cui il server inizia a mostrare segni di sovraccarico e le prestazioni iniziano a diminuire. Le fasi tipiche di un capacity test che sono state applicate sono:

- **Preparazione:** in questa fase vengono identificati gli obiettivi del test, vengono scelti i tool e le metodologie per la generazione del carico, vengono identificati i parametri da monitorare e vengono configurati i sistemi per il test.
- **Pianificazione:** in questa fase vengono create le configurazioni di test e vengono pianificati i tempi di test.
- **Esecuzione:** in questa fase viene generato il carico sul server utilizzando il tool e le metodologie scelte in precedenza e vengono monitorati i parametri delle prestazioni durante il test.

- **Analisi dei dati:** in questa fase vengono analizzati i dati raccolti durante il test per determinare i punti di Knee Capacity e Usable Capacity.
- **Report e conclusioni:** in questa fase vengono redatti un report e vengono formulate le conclusioni del test, compresi i problemi riscontrati e le raccomandazioni per migliorare le prestazioni del server.

4.2.1 Preparazione

Le metriche che vengono utilizzate per caratterizzare le performance del SUT in esame sono:

- **Response time:** intervallo di tempo che intercorre tra la richiesta di una risorsa e il completamento della risposta del sistema;
- **Throughput:** il numero di richieste per unità di tempo che il server è in grado di soddisfare.

Date queste due metriche di performance si andranno a ricercare due punti notevoli per il funzionamento del web server, che sono:

- **Knee Capacity:** punto in cui si ha il miglior trade-off tra le due metriche e corrisponde al punto in cui il rapporto (e quindi la potenza) tra throughput ed response time è massimo; Il punto di knee capacity rappresenta il punto in cui il sistema inizia a mostrare segni di sovraccarico e le prestazioni iniziano a diminuire. È il punto in cui il sistema raggiunge la sua capacità massima e oltre il quale le prestazioni continueranno a peggiorare. Si noterà come il tempo di risposta aumenti in modo significativo una volta superato il punto di knee capacity
- **Usable Capacity:** punto in cui si ha il throughput massimo, senza superare i vincoli sul tempo di risposta. Dopo tale punto il throughput aumenta al crescere del request rate mentre il tempo di risposta peggiora. Il punto di usable capacity è, quindi, il punto oltre il quale è consigliabile aggiungere risorse hardware al sistema per migliorare le prestazioni.

4.2.2 Pianificazione

Su lato Client, per effettuare il test, utilizzando il tool JMeter, si costruisce un **Test-Plan**.

Settaggio Thread Group

Per simulare il numero di utenti (utenti virtuali) che effettuano le richieste alle risorse presenti sul server è il numero di **thread group**. Nella sezione Thread Group si setta infatti:

- **N. Threads:** pari a 50;
- **Il ramp-up period**, ovvero il tempo di attivazione dell'ultimo thread: lo si setta pari a 25 (Quindi un utente verrà attivato ogni 0.5 secondi);
- **Il loop count**, ovvero numero di volte che ogni utente deve essere attivato, viene messo un valore vuoto per consentire così il funzionamento per il giusto quantitativo di tempo;
- **Duration:** 300 secondi (5 minuti).

Sono definiti differenti Thread Group con differenti **Constant Throughput Time (CTT)** che indicano il numero di richieste al minuto che ogni thread all'interno del thread group può effettuare. Nel dettaglio, si sono settati dei CTT come: **(250, 750, 1250, 1625, 2000, 2500, 3000, 4000)**. Ogni Thread Group avrà il suo CTT ed eseguito in sequenza.

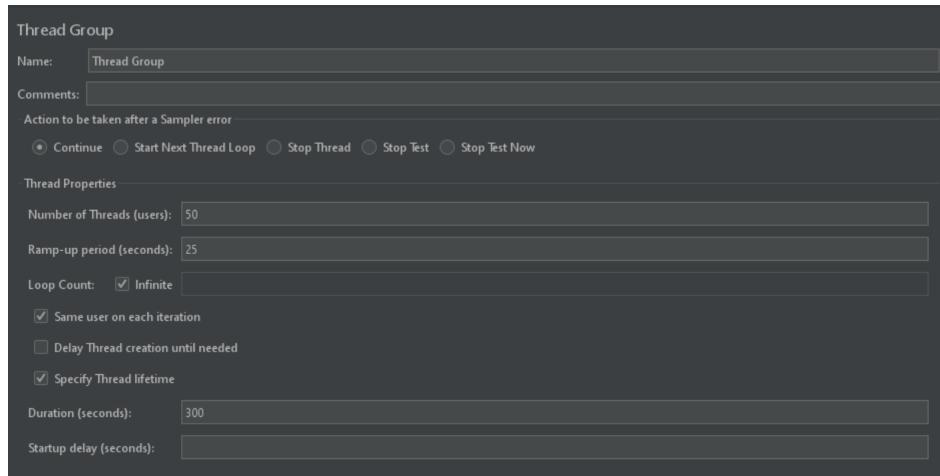


Figura 4.3: Configurazione Test Plan - Thread Group

Settaggio Sampler

Si setta il campionatore per determinare la "direzione" al client verso il server.

- **L'ip del server**, l'indirizzo IP del server creato con virtualbox (192.168.xxx.xxx);
- **Il path**, che identifica il nome della risorsa
- **Set GET**, perché si fanno delle richieste;
- **Set port number**: numero 80.

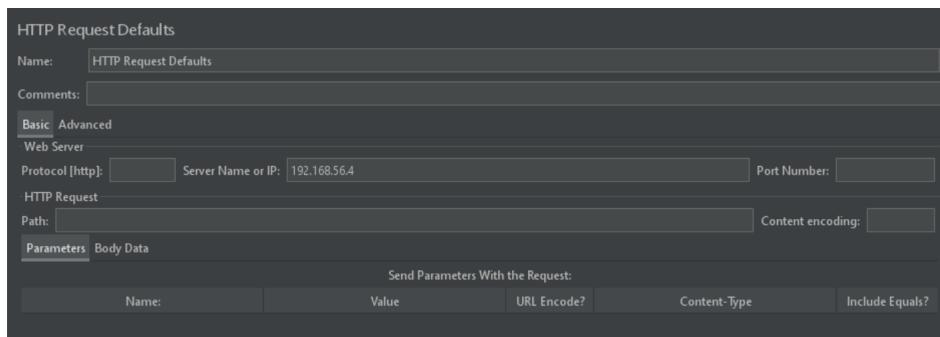


Figura 4.4: Configurazioni HTTP Requests

Per rendere il piano di test quanto più realistico possibile è stato definito come **Logic Controller** un *Random Controller* che in **maniera casuale** andrà a gestire le diverse **HttpRequest** configurate. Infine, si va a definire un

Listener che permetterà di monitorare l'output del test plan. Si è selezionato tra i listener disponibili il **Simple Data Writer**, che consente di salvare tutte le informazioni di ogni singola richiesta all'interno di un file di tipo .csv.

4.2.3 Esecuzione

Il test plan viene lanciato **3 volte** e tra un esperimento e un altro viene resettato il server al fine di evitare fenomeni di accumulo e caching che possano falsare i risultati. Ogni esperimento dura 5 minuti ciascuno.

Durante l'esecuzione dei test plan vengono monitorati i parametri delle prestazioni di **Throughput** come:

$$\text{Throughput} = \frac{\text{Numerodirichieste}}{\text{Tempodiesecuzione del test}} \quad (4.1)$$

e **Response Time** come la **media** dei valori della colonna "Elapsed".

CTT[req/min]	Throughput [req/s]		
	Oss. 1	Oss. 2	Oss. 3
250	4.331805	4.335588	4.331516
750	12.66667	12.66595	12.66684
1250	20.99695	20.99751	19.86265
1625	27.20424	27.1983	27.07126
2000	33.47678	33.48989	33.4878
2500	41.69236	41.80542	41.80014
3000	34.58151	43.98597	48.42274
4000	30.79361	36.58785	36.0014

CTT[req/min]	Response Time[ms]		
	Oss. 1	Oss. 2	Oss. 3
250	31.82679	32.64385	34.16166
750	26.51263	26.02869	47.54857
1250	29.06176	27.7044	356.7244
1625	34.51415	33.96691	50.02598
2000	33.65727	31.00547	33.72862
2500	48.23627	38.52755	35.27903
3000	789.8064	309.9848	118.609
4000	1238.24	958.9021	981.1816

Tabella 4.4: Valori di Throughput e Response Time ottenuti dalle osservazioni

Dalle tabella CTT/Throughput si può notare che il Throughput ottenuto ha un valore che tende a quello **ideale** del CTT finchè esso non satura (la riga 4000).

4.2.4 Analisi dei dati

Nell'analisi si è **calcolata la mediana per ogni carico nelle rispettive metriche di performance, questa scelta è stata dettata dagli elevati valori di COV riscontrati per alcuni valori di CTT nelle osservazioni del response time**. In particolare, si è calcolato il COV (coefficiente di variazione) come il rapporto s/\bar{x} , in cui s è la deviazione standard relativa all'osservazione e \bar{x} è la media delle osservazioni. Dunque, si riportano i valori mediani delle osservazioni come mostrato nelle seguenti tabelle:

CTT[req/min]	Throughput[req/s]			Median Throughput[req/s]
	Oss. 1	Oss. 2	Oss. 3	
250	4.331805	4.335588	4.331516	4.331805
750	12.66667	12.66595	12.66684	12.66667
1250	20.99695	20.99751	19.86265	20.99695
1625	27.20424	27.1983	27.07126	27.1983
2000	33.47678	33.48989	33.4878	33.4878
2500	41.69236	41.80542	41.80014	41.80014
3000	34.58151	43.98597	48.42274	43.98597
4000	30.79361	36.58785	36.0014	36.0014

Tabella 4.5: Throughput mediano per ogni carico

CTT[req/min]	Response Time[ms]			Median Response Time[ms]
	Oss. 1	Oss. 2	Oss. 3	
250	31.82679	32.64385	34.16166	32.64385
750	26.51263	26.02869	47.54857	26.51263
1250	29.06176	27.7044	356.7244	29.06176
1625	34.51415	33.96691	50.02598	34.51415
2000	33.65727	31.00547	33.72862	33.65727
2500	48.23627	38.52755	35.27903	38.52755
3000	789.8064	309.9848	118.609	309.9848
4000	1238.24	958.9021	981.1816	981.1816

Tabella 4.6: Response time mediano per ogni carico

Pertanto, per la valutazione dei punti di Knee Capacity e Usable Capacity si necessita del calcolo della potenza che per definizione è:

$$Power = \frac{Throughput}{ResponseTime} \quad (4.2)$$

Dalla 4.2 valutiamo dunque la seguente Tabella:

CTT[req/min]	Power[(req/s)/(ms)]			Median Power [(req/s)/(ms)]
	Oss. 1	Oss. 2	Oss. 3	
250	0.1361056204537121	0.1328148487387364	0.12679465810502183	0.1328148487387364
750	0.4777598450248051	0.4866149621821152	0.2663979169089628	0.4777598450248051
1250	0.7224940953335242	0.7579124615584527	0.055680659915609915	0.7224940953335242
1625	0.7882054171984534	0.8007292980138612	0.5411440215663941	0.7882054171984534
2000	0.9946374141455917	1.0801284418523571	0.9928600695788917	0.9946374141455917
2500	0.864336317878642	1.0850785995995074	1.1848438009775213	1.0850785995995074
3000	0.04378479333669618	0.14189718334576407	0.4082551914272947	0.14189718334576407
4000	0.02486885418012663	0.03815598067831951	0.036691882522053	0.036691882522053

Tabella 4.7: Potenza per ogni carico

Calcolati tutti i valori si plottano i rispettivi risultati:

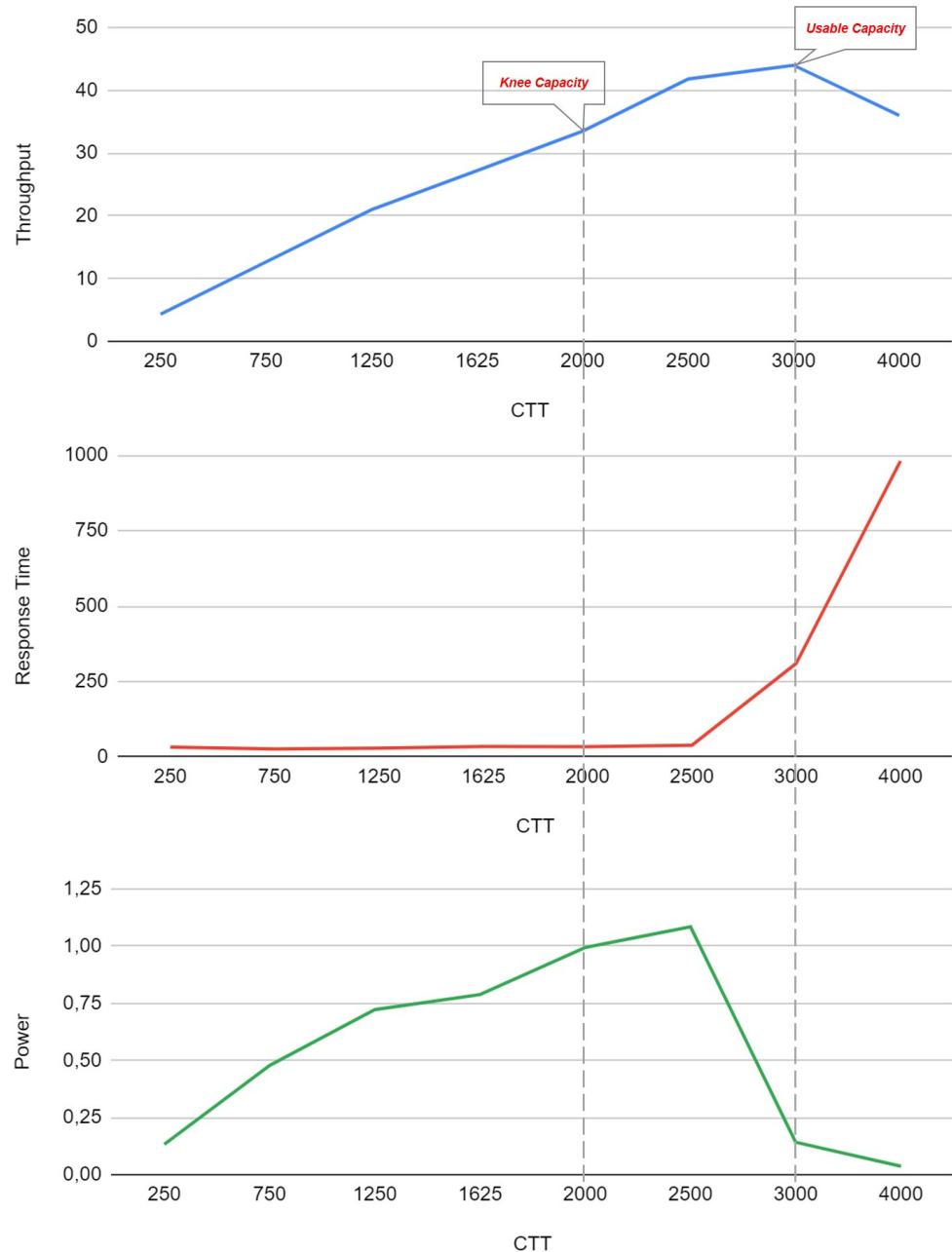


Figura 4.5: Grafici Capacity Test

A partire dalle Figure precedenti è possibile individuare in corrispondenza del punto nel grafico della potenza che assume il valore di picco il valore della Knee Capacity sul grafico del throughput; mentre, in corrispondenza del punto nel grafico della potenza in cui il valore tende a zero è possibile valutare il valore della Usable Capacity nel grafico del throughput.

4.2.4.1 Valutazione della Fairness

Inoltre, si calcola la **fairness** considerando il throughput valutando se il sistema sta fornendo un livello di servizio equo a tutti gli utenti in termini di quantità di richieste gestite.

Ad esempio, se un utente sta ricevendo solo la metà delle richieste rispetto ad un altro utente, questo potrebbe indicare un problema di equità nel sistema. La valutazione è così fatta:

$$fairness = f(x_1, x_2, \dots, x_n) = \frac{\left(\sum_{i=1}^N x_i\right)^2}{n \sum_{i=1}^N x_i^2} \quad (4.3)$$

dove:

$$x_i = \frac{T_i}{O_i} \quad (4.4)$$

Dal quale:

- T_i : è la mediana (o media) del Throughput misurato;
- O_i : è il Throuhgput fornito per la misurazione.

Nella seguente tabella si valuta pertanto la Fairness:

CTT[req/min]	Throughput[req/s]			Median Throughput[req/s]	Normalized Throughput
	Oss. 1	Oss. 2	Oss. 3		
250	4.331805	4.335588	4.331516	4.331805	1.0396332
750	12.666667	12.66595	12.66684	12.666667	1.0133336
1250	20.99695	20.99751	19.86265	20.99695	1.0078536
1625	27.20424	27.1983	27.07126	27.1983	1.004244923076923
2000	33.47678	33.48989	33.4878	33.4878	1.004634
2500	41.69236	41.80542	41.80014	41.80014	1.0032033599999999
3000	34.58151	43.98597	48.42274	43.98597	0.8797194000000002
4000	30.79361	36.58785	36.0014	36.0014	0.540021
Fairness Index					0.9728647635457881

Tabella 4.8: Calcolo dell'indice di fairness

Dalla Tabella il valore di Fairness è maggiore del 75/70% (**97,29%**) allora il sistema, in generale, sta distribuendo il throughput in maniera fair tra i vari thread group. Si nota come solo l'ultimo thread group abbia un valore di fairness basso, il che significa che rispetto al CTT assegnato esso sta ricevendo un throughput decisamente inferiore e risulta per questo svantaggiato rispetto agli altri.

4.2.5 Report e Conclusioni

Dai plot si identificano dunque i punti di Knee Capacity in corrispondenza del carico di **2000 req/min**. Fino a questo punto il sistema funziona al suo meglio, esibendo salti di throughput elevati e response time bassi. Dopodiché il

throughput continua ad aumentare col carico ma in maniera più contenuta e i tempi di risposta iniziano a crescere vertiginosamente: ciò è indice del fatto che si sta andando verso la saturazione del sistema testato. Pertanto, si identifica il punto di Usable Capacity in corrispondenza del valore di **3000 req/min** che come detto è il massimo throughput del sistema. Inoltre si è garantita la stabilità del sistema, ovvero il rapporto dei rispettivi punti è inferiore almeno al 75%. Infatti:

$$\frac{\text{KneeCapacity}}{\text{UsableCapacity}} = \frac{2000}{3000} = 0,66667 = 66,67\% < 75\%. \quad (4.5)$$

Dunque, tanto più è lontano la Usable Capacity rispetto al Knee Capacity, tanto meglio il sistema riesce a gestire il carico. Garantendo che questo rapporto sia inferiore al 75% si può dire che il sistema è in grado di gestire il carico delle richieste con una maggior capacità prima che si verifichino problemi di qualità del servizio o problemi di prestazioni.

4.3 Workload Characterization

Si sperimenta come un workload reale applicato al SUT¹ possa essere sintetizzato al fine di effettuare degli esperimenti con un workload sintetico che è caratterizzato dalla gran parte della variabilità del dataset, ma con una dimensionalità ridotta. Per effettuare questo processo di workload characterization sono state eseguite le seguenti fasi:

- **Creazione del workload reale (ipotetico)** di alto livello (HL^2) e di basso livello (LL^3);
 - **Creazione del workload sintetico** di alto livello (HL_c) e di basso livello (LL_c);
 - **Applicazione del workload di alto livello caratterizzato** (HL_c) e raccolta dei parametri di basso livello LL' in funzione degli HL_c ;
 - **Caratterizzazione del workload LL'** in sintetico (LL_c);
 - **Verifica della bontà della caratterizzazione del workload di alto livello** tramite test di ipotesi (validare $LL_c \approx LL'c$)

Di seguito lo schema delle fasi seguite:

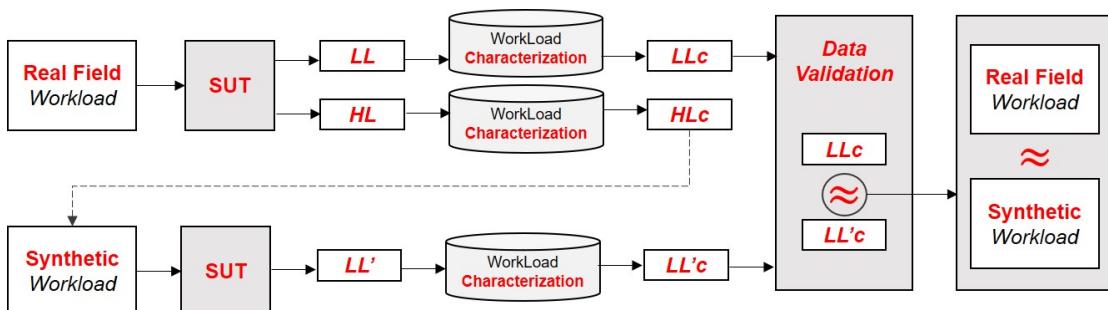


Figura 4.6: Schema del Workload Characterization

4.3.1 Creazione del workload reale (ipotetico)

Come indicato dalla prima fase del processo, si raccolgono i dati per la caratterizzazione del workload reale, anche in questo caso si utilizza il tool Apache JMeter 5.5 per generare delle richieste dal client al server.

4.3.1.1 Workload reale di alto livello HL

Configurazione Test Plan ed esecuzione

A differenza del test plan configurato nel Capacity Test, qui sono stati creati 5 *Thread Group* ognuno dei quali è caratterizzato da:

- 9 richieste HTTP

¹ La configurazione del System Under Test (SUT) è la stessa definita nel paragrafo dedicato al capacity test.

² Parametri raccogibili lato client: throughput e response time, che non forniscono una chiara indicazione su cosa sta accadendo sul lato del servente.

³ parametri del sistema sotto test e consistono in informazioni riguardanti lo stato del server, come l'utilizzo dei processori e della memoria.

- Durata dell'esperimento: 60 secondi e ramp up period di 1s
- CTT pari a 1500

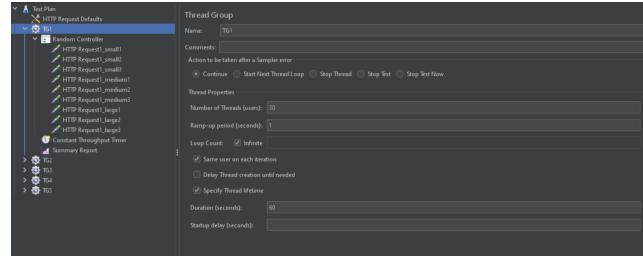


Figura 4.7: Configurazioni HTTP Requests

Di seguito è riportato un estratto del workload di alto livello ottenuto.

timeStamp	elapsed	label	responseCode	responseMessage	threadName	dataType	success	failureMessage	bytes	sentBytes	grpThread	allThreads	URL	filename	latency	encoding	sampleCount	errorCount	hostname	idleTime	connect
1,7E+12	2184	HTTP Req1	200	OK	TG1 1-28	text	true		102714	133	30	30	http://192.168.56.4/	1902	ISO-8859-	1	0	GiuseppeV	0	1627	
1,7E+12	2183	HTTP Req1	200	OK	TG1 1-20	text	true		102714	133	30	30	http://192.168.56.4/	1906	ISO-8859-	1	0	GiuseppeV	0	1626	
1,7E+12	2184	HTTP Req1	200	OK	TG1 1-27	text	true		102714	133	30	30	http://192.168.56.4/	1920	ISO-8859-	1	0	GiuseppeV	0	1627	
1,7E+12	2184	HTTP Req1	200	OK	TG1 1-7	text	true		102714	133	30	30	http://192.168.56.4/	1931	ISO-8859-	1	0	GiuseppeV	0	1627	
1,7E+12	2184	HTTP Req1	200	OK	TG1 1-1	text	true		102714	133	30	30	http://192.168.56.4/	1910	ISO-8859-	1	0	GiuseppeV	0	1627	
1,7E+12	2183	HTTP Req1	200	OK	TG1 1-9	text	true		102714	133	30	30	http://192.168.56.4/	1903	ISO-8859-	1	0	GiuseppeV	0	1626	
1,7E+12	2184	HTTP Req1	200	OK	TG1 1-14	text	true		102714	133	30	30	http://192.168.56.4/	1917	ISO-8859-	1	0	GiuseppeV	0	1627	
1,7E+12	2184	HTTP Req1	200	OK	TG1 1-21	text	true		102714	133	30	30	http://192.168.56.4/	1902	ISO-8859-	1	0	GiuseppeV	0	1627	
1,7E+12	95	HTTP Req1	200	OK	TG1 1-20	text	true		102713	133	30	30	http://192.168.56.4/	63	ISO-8859-	1	0	GiuseppeV	0	0	
1,7E+12	114	HTTP Req1	200	OK	TG1 1-7	text	true		102713	133	30	30	http://192.168.56.4/	62	ISO-8859-	1	0	GiuseppeV	0	0	
1,7E+12	115	HTTP Req1	200	OK	TG1 1-27	text	true		102713	133	30	30	http://192.168.56.4/	63	ISO-8859-	1	0	GiuseppeV	0	0	
1,7E+12	115	HTTP Req1	200	OK	TG1 1-1	text	true		102713	133	30	30	http://192.168.56.4/	61	ISO-8859-	1	0	GiuseppeV	0	0	
1,7E+12	2305	HTTP Req1	200	OK	TG1 1-12	text	true		1048874	134	30	30	http://192.168.56.4/	1901	ISO-8859-	1	0	GiuseppeV	0	1626	
1,7E+12	119	HTTP Req1	200	OK	TG1 1-14	text	true		102713	133	30	30	http://192.168.56.4/	61	ISO-8859-	1	0	GiuseppeV	0	0	
1,7E+12	2366	HTTP Req1	200	OK	TG1 1-6	text	true		1048874	134	30	30	http://192.168.56.4/	1902	ISO-8859-	1	0	GiuseppeV	0	1627	
1,7E+12	2411	HTTP Req1	200	OK	TG1 1-13	text	true		1048874	134	30	30	http://192.168.56.4/	1903	ISO-8859-	1	0	GiuseppeV	0	1626	
1,7E+12	2330	HTTP Req1	200	OK	TG1 1-4	text	true		1048874	134	30	30	http://192.168.56.4/	1798	ISO-8859-	1	0	GiuseppeV	0	1523	
1,7E+12	2541	HTTP Req1	200	OK	TG1 1-11	text	true		1048874	134	30	30	http://192.168.56.4/	1902	ISO-8859-	1	0	GiuseppeV	0	1627	
1,7E+12	140	HTTP Req1	200	OK	TG1 1-4	text	true		102713	133	30	30	http://192.168.56.4/	44	ISO-8859-	1	0	GiuseppeV	0	0	
1,7E+12	2581	HTTP Req1	200	OK	TG1 1-26	text	true		1048874	134	30	30	http://192.168.56.4/	1902	ISO-8859-	1	0	GiuseppeV	0	1627	
1,7E+12	425	HTTP Req1	200	OK	TG1 1-9	text	true		1048873	134	30	30	http://192.168.56.4/	54	ISO-8859-	1	0	GiuseppeV	0	0	
1,7E+12	2623	HTTP Req1	200	OK	TG1 1-5	text	true		1048874	134	30	30	http://192.168.56.4/	1902	ISO-8859-	1	0	GiuseppeV	0	1627	
1,7E+12	60	HTTP Req1	200	OK	TG1 1-26	text	true		102713	133	30	30	http://192.168.56.4/	13	ISO-8859-	1	0	GiuseppeV	0	0	
1,7E+12	335	HTTP Req1	200	OK	TG1 1-12	text	true		1048873	134	30	30	http://192.168.56.4/	24	ISO-8859-	1	0	GiuseppeV	0	0	

Figura 4.8: Workload di alto livello

4.3.1.2 Workload reale di basso livello LL

I dati di basso livello sono stati raccolti mediante l'utilizzo del comando `vmstat` eseguito tramite terminale sulla macchina virtuale Linux che ospita il Web Server **durante l'esecuzione degli esperimenti**. Di seguito si riporta il comando completo:

```
vmstat -n 1 300 > workloadLL.csv
```

Il comando è stato lanciato subito prima di avviare l'esecuzione del Test Plan su JMeter (Applicazione del WL reale al SUT): con tali settaggi la utility ha registrato e riportato lo stato del server in ogni secondo per i successivi 300 secondi (durata del test JMeter). Di seguito è riportato un estratto del workload di basso livello ottenuto.

4.3.2 Workload Characterization HL

A valle della creazione del workload reale di alto livello lo si caratterizza seguendo gli stessi passi applicati nel capitolo dedicato alla workload characterization.

r	b	swpd	free	buff	cache	si	so	bi	bo	in	cs	us	sy	id	wa	st
2	2	541968	6516	3292	137960	1888	3536	16544	4198	1468	2561	6	44	37	13	0
1	0	540952	5340	3720	135844	5792	2136	18464	2168	1664	2967	0	18	48	35	0
1	0	530464	5088	3416	131380	14104	2100	17212	2100	1689	2099	0	16	46	38	0
4	0	533552	9540	2580	115392	8672	3996	22328	3996	3477	5425	1	75	4	20	0
12	0	542544	16656	1812	120144	1820	9128	27560	9128	2573	2409	2	76	19	3	0
4	0	544360	22440	1820	117980	4700	5996	10244	6060	2885	2989	1	72	20	8	0
1	0	548992	23268	2596	124460	616	5108	10891	5176	2998	3198	1	72	24	3	0
1	0	548992	20548	2752	128732	72	0	3150	0	2206	3718	0	46	51	2	0
1	0	548736	20404	2752	128736	48	0	48	0	1463	8386	0	25	74	1	0
0	0	548736	20936	2752	128740	56	0	56	0	989	2300	0	12	88	1	0
2	0	548736	19792	2752	128740	60	0	60	0	1315	5093	1	20	80	0	0
1	0	550280	16000	2748	129592	1296	2168	4568	2256	2269	7941	0	40	54	6	0
0	0	550792	19672	2748	131544	72	312	1556	312	1756	5086	1	28	70	1	0
0	0	550536	21016	2748	131548	24	0	24	0	1339	6347	0	27	72	1	0
4	0	550536	19100	2748	131548	60	0	60	0	1304	6293	0	25	75	1	0
1	0	550536	19272	2748	131552	24	0	24	0	1578	9518	1	27	72	0	0
0	0	550536	20224	2756	131544	16	0	16	48	944	2223	0	13	86	1	0
2	0	550536	17468	2756	131560	20	0	20	4	1416	5933	0	34	65	1	0
2	0	552592	23452	2748	129280	312	2728	564	2740	1852	2106	0	54	44	2	0
1	0	554648	26648	2744	127032	28	2136	336	2148	1920	1601	0	51	49	1	0
0	0	554648	29844	2744	127036	64	0	64	0	1208	4299	0	24	76	1	0
5	0	554136	23640	2752	129316	1176	0	3472	28	1925	9349	0	36	58	6	0
5	0	553112	20512	2752	129536	1596	0	1836	0	1620	6968	0	36	60	4	0

Figura 4.9: Workload di basso livello

4.3.2.1 Preprocessing

Si effettua il preprocessing dei dati per ridurre le dimensioni preliminari:

Eliminazione delle colonne costanti e colonne nominali

Le colonne eliminate:

- timeStamp
- label
- Hostname
- URL
- ResponseCode
- threadName
- dataType
- Success
- Encoding
- SampleCount
- IdleItem
- ResponseMessage
- ErrorCount

Eliminazione delle colonne correlate

Si valuta la matrice delle correlazioni per eliminare altre colonne:

Multivariato							
Correlazioni							
elapsed	elapsed	bytes	sentBytes	grpThreads	allThreads	Latency	Connect
elapsed	1,0000	0,2962	-0,1339	0,0093	0,0093	0,8050	0,7973
bytes	0,2962	1,0000	-0,2929	0,0321	0,0321	-0,0023	-0,0028
sentBytes	-0,1339	-0,2929	1,0000	-0,0182	-0,0182	-0,0306	-0,0292
grpThreads	0,0093	0,0321	-0,0182	1,0000	1,0000	0,0082	0,0097
allThreads	0,0093	0,0321	-0,0182	1,0000	1,0000	0,0082	0,0097
Latency	0,8050	-0,0023	-0,0306	0,0082	0,0082	1,0000	0,9987
Connect	0,7973	-0,0028	-0,0292	0,0097	0,0097	0,9987	1,0000

Le correlazioni sono stimate per metodo A livello di riga.

Figura 4.10: Matrice di correlazione del workload di alto livello

Essendo che le colonne *grpThread* e *AllThread* sono fortemente correlate, rimuoviamo una di esse, in particolare si è deciso di eliminare AllThread.

4.3.2.2 PCA

Si applica poi la PCA, dall'analisi del tool JMP si selezionano 4 componenti principali per ottenere una varianza di circa il 96%.

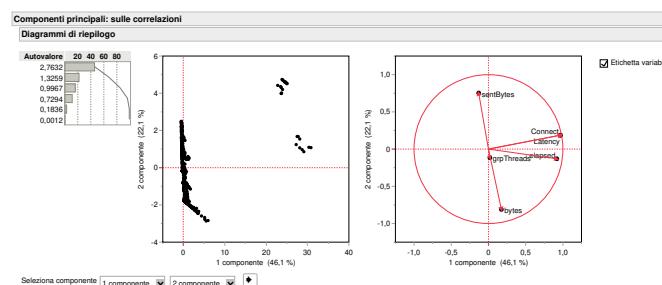


Figura 4.11: Loading Plot della PCA

Componenti principali: sulle correlazioni					
Autovalori					
Numero	Autovalore	Percentuale	20	40	Percentuale cumulativa
1	2,7632	46,053			46,053
2	1,3259	22,098			68,151
3	0,9967	16,612			84,763
4	0,7294	12,157			96,920
5	0,1836	3,060			99,980
6	0,0012	0,020			100,000

Figura 4.12: Autovalori della PCA del workload di alto livello

4.3.2.3 Clustering

Si applica il Clustering Gerarchico attraverso il **metodo di Ward**, in particolare a partire dal dendrogramma e dalla cronologia di clusterizzazione mostrata nelle Figure seguenti, applicando gli stessi passaggi del capitolo

precedente, è possibile determinare in maniera rapida il numero di cluster, infatti, si sceglie il numero di cluster che **massimizza la la distanza tra l'aggiunta e/o la rimozione di un ulteriore cluster da esso.**

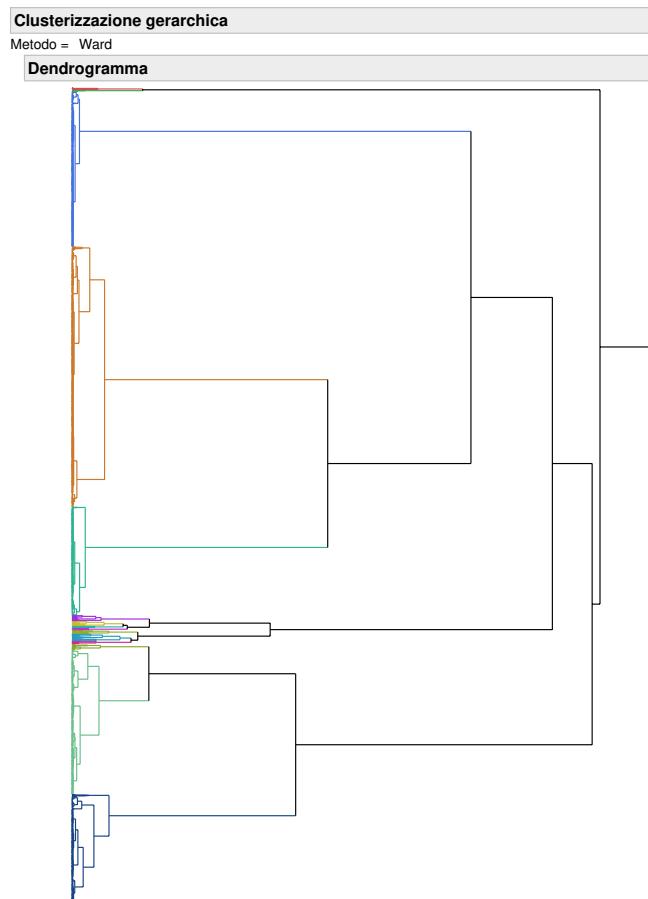


Figura 4.13: Dendrogramma del workload di alto livello per 4 componenti

Cronologia di clusterizzazione				
Numero di cluster	Distanza	Leader	Subordinato	
1	85,55489263	1	9	
2	84,28156850	9	53	
3	77,83384390	9	1390	
4	64,63885385	9	21	
5	41,42505598	21	4501	
6	36,23292400	53	1393	
7	32,10056032	1390	1406	
8	12,53086739	1390	1399	
9	12,43747839	53	105	
10	11,43658058	1	42	
11	10,64359878	1406	1407	
12	9,55120785	1407	1409	
13	8,94368867	1399	4485	
14	8,26104474	1399	1400	
15	7,76263931	1407	1413	
16	5,98039545	1393	4505	
17	5,94217789	1406	1415	
18	5,26398927	21	4506	
19	5,19600732	1399	1404	
20	4,90306505	1407	4476	
21	4,71364925	1390	1394	
22	4,57660835	53	62	
23	4,36185215	105	113	
24	4,18221414	1	13	
25	3,93221773	1413	4464	

Figura 4.14: Cronologia di clusterizzazione del workload di alto livello per 4 componenti

Al fine di minimizzare la variabilità interna ad ogni cluster (**devianza intraccluster**) e massimizzare quella esterna (**devianza intercluster**), si può plottare la distanza e il numero di cluster valutato con JMP e scegliere opportunamente il numero di cluster intorno al "ginocchio" del plot (tale tecnica prende il nome di **analisi di sensitività**).

Dalla cronologia di clusterizzazione sarà possibile, dunque, costruire tale grafico in cui viene posto in ordinata il numero di cluster ed in ascissa la distanza, come mostrato di seguito:

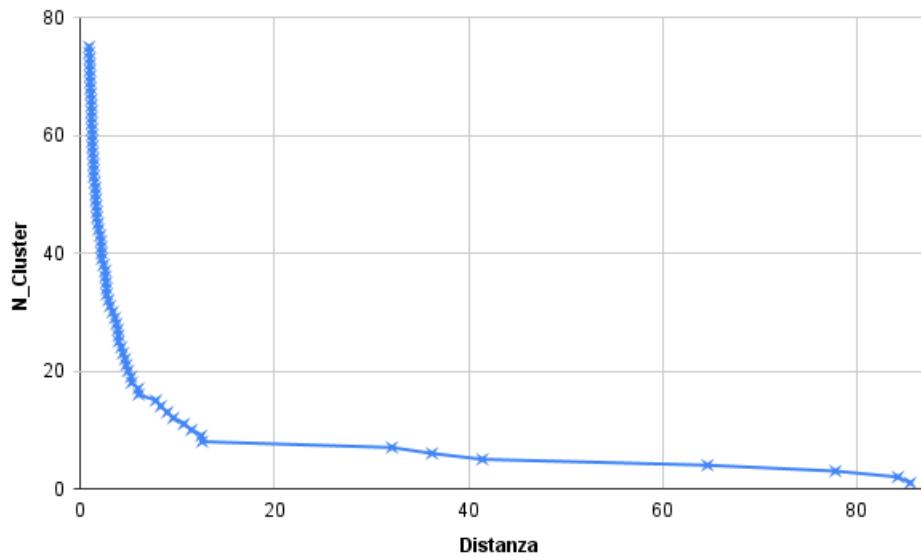


Figura 4.15: Analisi di sensitività per la scelta del numero di cluster

Inizialmente si è scelto, il numero di cluster pari a 15 poi si valuterà la devianza persa anche per un numero pari a 20 per un'analisi e scelta più accurata.

Si noti che, la scelta del numero di cluster è influenzata anche dalla devianza persa a valle di tale operazione, dunque, la scelta potrebbe non corrispondere esattamente con il ginocchio di tale grafico ma con un valore prossimo ad esso. Tale situazione, tuttavia, non risulta particolarmente svantaggiosa dato che la distanza massima di tali punti adiacenti al ginocchio risulta comunque minore di quella iniziale.

4.3.2.4 Analisi della devianza persa

Si valuta la devianza persa utilizzando lo script Matlab "deviance.m" ed i risultati ottenuti sono i seguenti:

Cluster HL	Devianza Persa HL	
	3	4
15	16.37%	4.57%
20	16.05%	4.16%

Tabella 4.9: Analisi devianza persa nel workload HL

Dalla precedente Tabella, **si è dunque, scelto come valore di PC e cluster rispettivamente 4 e 15 in quanto il trade-off tra dimensionalità e devianza persa risulta ottimale, avendo solo il 4,57% di devianza persa.**

4.3.2.5 Workload sintetico

A partire da questa caratterizzazione è stato realizzato un workload sintetico tramite l'**analisi di contingenza** della label (HTTP Request) rispetto ai cluster; **in particolare per ciascun cluster, si è scelto l'HTTP Request con conteggio maggiore.**

Dapprima si è effettuata un'analisi visiva grazie al diagramma a mosaico in Figura 4.16, per individuare le richieste più frequenti ad un primo impatto.

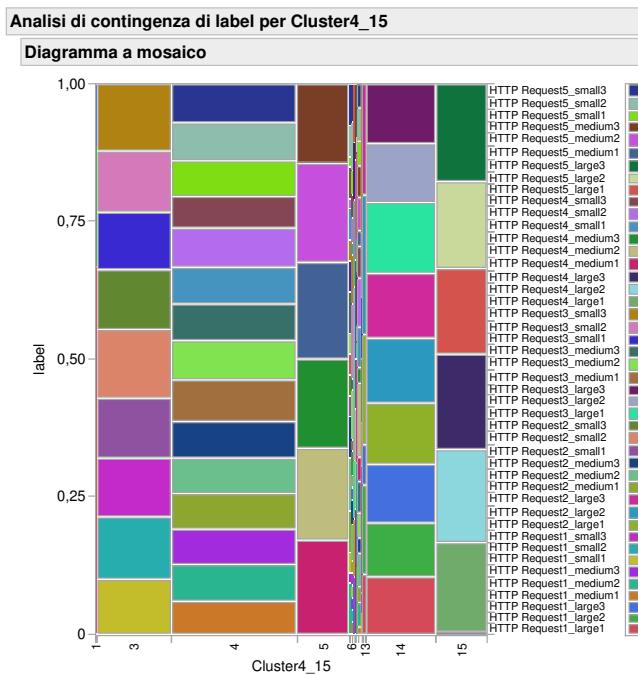


Figura 4.16: Analisi di contingenza del workload di alto livello per 4 componenti e 15 cluster

Successivamente, si è fatto uso della Tabella di contingenza, di cui è mostrato un estratto in Figura 4.17, per la più precisa individuazione delle HTTP Request più frequenti per ogni cluster.

Tabella di contingenza									
Conteggio	HTTP Request1.large1	HTTP Request1.large2	HTTP Request1.large3	HTTP Request1.medium1	HTTP Request1.medium2	HTTP Request1.medium3	HTTP Request1.small1	HTTP Request1.small2	
Cluster									
1	0	0	0	3	4	6	2	0	0
2	6	2	1	0	0	0	0	0	0
3	0	0	0	0	0	0	146	0	162
4	0	0	0	147	159	159	0	0	0
5	0	0	0	0	0	0	0	0	0
6	0	0	0	0	5	1	2	0	0
7	0	0	0	1	1	1	1	0	0
8	0	0	2	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0
10	0	4	3	0	0	0	0	0	0
11	0	0	0	1	3	0	0	0	0
12	0	0	0	0	0	0	1	0	0
13	6	9	4	0	0	0	0	0	0
14	140	131	144	0	0	0	0	0	0
15	0	0	1	0	0	0	0	0	0
Totali	152	146	155	152	172	167	152	0	162

Figura 4.17: Estratto della tabella di contingenza per il workload di alto livello per 4 componenti e 15 cluster

Pertanto, a partire dall'individuazione delle richieste più frequenti, il Workload Sintetico HL risulta così composto:

Cluster	Richiesta più frequente
1	HTTP Request1.medium3
2	HTTP Request1.large1
3	HTTP Request2.small2
4	HTTP Request3.medium1
5	HTTP Request5.medium2
6	HTTP Request1.medium2
7	HTTP Request3.small2
8	HTTP Request3.large2
9	HTTP Request4.medium1
10	HTTP Request2.large3
11	HTTP Request4.medium2
12	HTTP Request3.small3
13	HTTP Request2.large2
14	HTTP Request3.large1
15	HTTP Request5.large3

Tabella 4.10: Workload Sintetico HL

4.3.3 Workload Characterization LL

A valle del raccoglimento dei dati LL, anche in questo caso seguendo il processo di workload characterization del capitolo precedentemente.

4.3.3.1 Preprocessing

Si effettua il preprocessing dei dati per ridurre le dimensioni preliminarmente:

Eliminazione delle colonne costanti e colonne nominali Le colonne eliminate:

- st
- b

Eliminazione delle colonne correlate Si valuta la matrice delle correlazioni per eliminare altre colonne:

Multivariato																		
Correlazioni																		
r	1,0000	-0,1574	-0,1801	-0,0149	-0,1064	0,1461	0,3653	0,2921	0,3726	0,2793	0,1155	0,2174	0,3174	-0,3135	0,1710			
swpd	-0,1574	1,0000	0,3577	0,3394	0,7070	-0,2665	-0,1774	-0,0420	-0,1769	-0,2218	-0,6195	0,0335	-0,1057	0,1396	-0,1979			
free	-0,1801	0,3577	1,0000	-0,0237	-0,3601	-0,2279	-0,2187	-0,1782	-0,2172	-0,1762	-0,3045	-0,0542	-0,0790	0,1388	-0,2540			
buff	-0,0149	0,3394	-0,0237	1,0000	0,3323	0,0097	-0,1319	0,0550	-0,1219	0,0474	-0,2816	0,0552	0,0662	-0,0969	0,1523			
cache	-0,1064	0,7070	-0,3601	0,3323	1,0000	-0,0780	-0,1377	-0,0367	-0,1380	-0,1714	-0,3770	-0,0097	-0,1369	0,1249	-0,0509			
si	0,1461	-0,2665	-0,2279	0,0097	-0,0780	1,0000	0,2664	0,3108	0,2656	0,4708	0,2163	0,1495	0,2866	-0,4801	0,8429			
so	0,3653	-0,1774	-0,1801	-0,1319	-0,1377	0,2664	1,0000	0,6002	0,9982	0,6080	0,1344	0,5617	0,6233	-0,6438	0,3991			
bi	0,2921	-0,0420	-0,1782	0,0550	-0,0367	0,3108	0,6002	1,0000	0,6079	0,7585	0,1578	0,5096	0,6867	-0,7538	0,6146			
bo	0,3726	-0,1769	-0,2172	-0,1219	-0,1380	0,2656	0,9982	0,6079	1,0000	0,6146	0,1344	0,5777	0,6313	-0,6526	0,4054			
in	0,2793	-0,2218	-0,1762	0,0474	-0,1714	0,4708	0,6080	0,7585	0,6146	1,0000	0,4035	0,4846	0,8887	-0,9301	0,6683			
cs	0,1155	-0,6191	-0,3045	-0,2816	-0,3770	0,2165	0,1344	0,1578	0,1344	0,4035	1,0000	0,0934	0,2651	-0,2889	0,2508			
us	0,2174	0,0335	-0,0542	0,0552	-0,0097	0,1495	0,5617	0,5096	0,5777	0,4846	0,0934	1,0000	0,4852	-0,5479	0,3820			
sy	0,3174	-0,1057	-0,0790	0,0662	-0,1369	0,2866	0,6233	0,6867	0,6313	0,8887	0,2651	0,4852	1,0000	-0,9644	0,4768			
id	-0,3135	0,1396	0,1388	-0,0969	0,1249	-0,6438	-0,7538	-0,6526	-0,9301	-0,2889	-0,5479	-0,9644	1,0000	-0,6883				
wa	0,1710	-0,1979	-0,2540	0,1523	-0,0509	0,8429	0,3991	0,6146	0,4054	0,6683	0,2508	0,3820	0,4768	-0,6883	1,0000			

Le correlazioni sono stimate per metodo A livello di riga.

Figura 4.18: Matrice di correlazione del workload di basso livello

NON c'è nessuna forte correlazione tra le colonne.

4.3.3.2 PCA

Si applica la PCA, dall'analisi del tool JMP si selezionano 6,7,8 componenti principali per ottenere le rispettive varianze, in particolare, in prima battuta ci si soffermerà su 8 componenti principali per l'analisi:

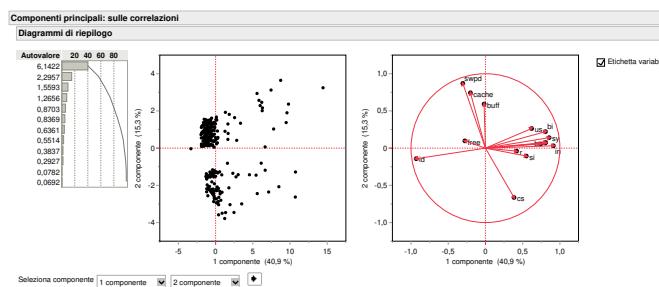


Figura 4.19: Loading Plot della PCA

Componenti principali: sulle correlazioni

Autovalori

Numero	Autovalore	Percentuale	20	40	60	80	Percentuale cumulativa
1	6,1422	40,948					40,948
2	2,2957	15,305					56,253
3	1,5593	10,396					66,648
4	1,2656	8,437					75,085
5	0,8703	5,802					80,887
6	0,8369	5,580					86,467
7	0,6361	4,241					90,707
8	0,5514	3,676					94,383
9	0,3837	2,558					96,941
10	0,2927	1,951					98,893
11	0,0782	0,522					99,414
12	0,0692	0,462					99,876
13	0,0168	0,112					99,988
14	0,0015	0,010					99,998
15	0,0003	0,002					100,000

Figura 4.20: Autovalori della PCA del workload di basso livello

4.3.3.3 Clustering

Si applica il Clustering Gerarchico attraverso il metodo di Ward e si scelgono 10, 15 e 22 cluster da analizzare.

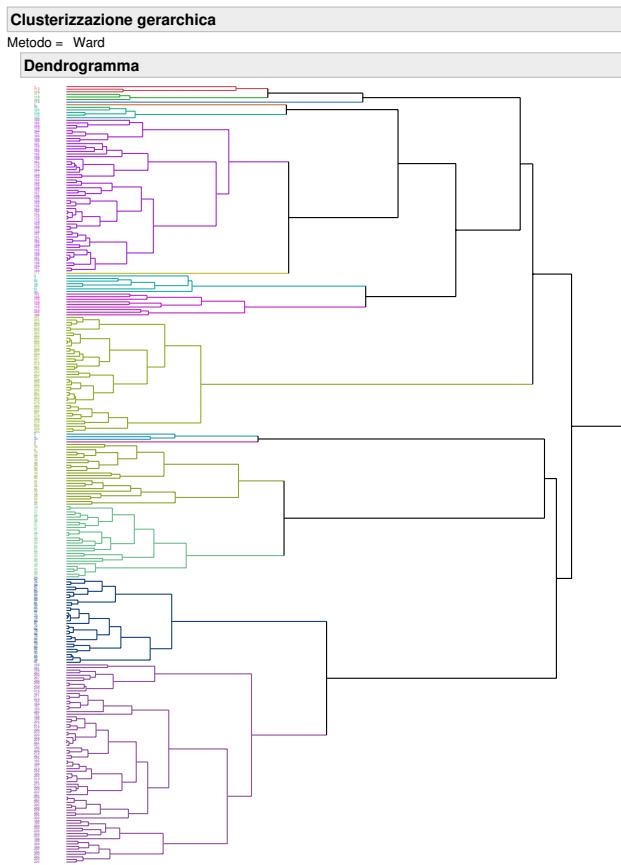


Figura 4.21: Dendrogramma del workload di basso livello per 8 componenti

Cronologia di clusterizzazione			
Numero di cluster	Distanza	Leader	Subordinato
2	15,06730242	2	62
3	14,70083316	2	8
4	14,34027810	1	241
5	13,95388184	1	5
6	11,97374247	5	6
7	10,19765284	5	100
8	9,20370324	6	101
9	9,11470486	1	114
10	8,00043032	62	156
11	6,83471189	100	111
12	6,76448165	5	98
13	6,69244406	8	10
14	6,19369816	1	117
15	5,89360714	2	3
16	5,69227537	156	157
17	5,48155514	101	112
18	5,29257295	8	11
19	5,21708463	1	113
20	4,99363829	100	121
21	4,94677950	157	188
22	4,60599667	121	122
23	4,15417848	101	116
24	4,13338533	241	245
25	3,86448450	6	99
26	3,74436364	6	7
27	3,68512069	10	13
28	3,34739847	11	22
29	3,33782960	2	4

Figura 4.22: Cronologia di clusterizzazione del workload di basso livello per 8 componenti

Dalla cronologia di clusterizzazione sarà possibile, dunque, costruire il grafico in cui viene posto in ordinata il numero di cluster ed in ascissa la distanza:

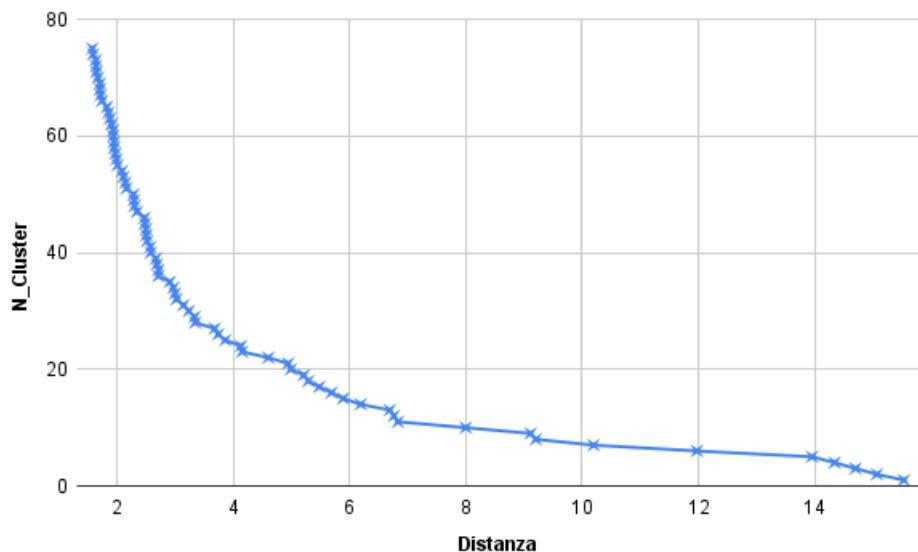


Figura 4.23: Analisi di sensitività per la scelta del numero di cluster

Inizialmente si è scelto, il numero di cluster pari a 10 poi si valuterà la devianza persa anche per un numero pari a 15 e 22 per un'analisi e scelta più accurata.

4.3.3.4 Analisi della devianza persa

Si valuta la devianza persa utilizzando lo script Matlab "deviance.m" e i risultati sono:

Cluster LL	Devianza Persa LL		
	6	7	8
10	31.06%	32.92%	33.51%
15	27.42%	25.30%	23.36%
22	24.27%	20.72%	18.84%

Tabella 4.11: Analisi devianza persa nel workload LL

Come si può notare dalla tabella, la scelta 10 può risultare infelice in quanto presenta un'anomalia, ovvero, all'aumentare del numero delle componenti principali si è notato un incremento della devianza persa. Dall'analisi si è pertanto scelto la combinazione: 8 componenti principali e 22 cluster con una perdita del 18,84% di varianza. Lo abbiamo potuto fare perché non abbiamo limiti di costi.

4.3.3.5 Workload sintetico

Per costruire il workload sintetico, questa volta, ci si avvale del calcolo del centroide reale per ogni cluster (come fatto nel capitolo del Workload Characterization 3 tramite il calcolo della distanza euclidea tra ogni istanza del cluster ed il relativo centroide virtuale). Usando questo criterio di costruzione del workload sintetico, a differenza di una scelta randomica, si garantisce che l'istanza rappresentativa del cluster è anche quella che ha una distanza media da tutte le altre pressoché uguale e quindi, essa può ben rappresentarle tutte.

Il Workload Sintetico HL_c ottenuto è il seguente:

r	b	swpd	free	buff	cache	si	so	bi	bo	in	cs	us	sy	id	wa	st
2	2	541968	6516	3292	137960	1888	3536	16544	4198	1468	2561	6	44	37	13	0
4	0	580640	36596	2436	123356	0	6336	42736	6336	2938	1795	4	77	13	6	0
1	0	598648	55644	4492	150688	1904	640	13232	652	2692	2536	5	64	19	12	0
5	1	595544	14024	2012	147496	16	15068	45544	15120	2880	1874	7	62	18	13	0
12	0	542544	16656	1812	120144	1820	9128	27560	9128	2573	2409	2	76	19	3	0
5	0	590720	29664	4628	154496	0	0	0	0	1172	606	0	28	72	0	0
2	0	565472	14184	4768	143896	0	0	0	0	1289	717	1	29	70	0	0
0	0	590968	32044	4348	160164	0	0	0	0	1128	658	0	25	75	0	0
4	0	571640	13260	7664	141852	196	5780	26393	5952	2766	2303	2	74	10	15	0
3	0	559808	33812	3280	116064	12	4128	33388	4132	2725	3855	0	64	32	3	0
2	0	592504	29776	4292	163840	3068	0	39504	0	3303	3483	1	66	18	15	0
4	0	594320	35532	3804	157204	180	3840	32124	4236	2674	1813	1	64	31	5	0
1	0	594576	17432	3036	172256	0	0	0	4	1099	567	0	25	74	1	0
1	0	553888	21908	2988	127444	6576	0	6676	0	2824	6068	0	40	32	28	0
1	0	530464	5088	3416	131380	14104	2100	17212	2100	1689	2099	0	16	46	38	0
1	0	556208	26256	2988	124100	12	0	12	0	1483	5928	0	33	67	0	0
3	0	556208	27876	2988	124088	8	0	56	28	1391	4975	1	26	72	1	0
0	0	555952	29384	2988	124108	4	0	4	4	1108	3667	0	21	79	0	0
1	0	561608	41184	3076	117072	0	0	0	4	1119	607	0	26	74	0	0
3	0	594064	49712	3488	140416	0	0	0	0	1186	607	0	24	76	0	0
0	0	594064	53792	3472	140228	0	0	0	8	1186	652	0	27	73	0	0
1	0	594064	44012	3520	147012	0	0	0	0	924	586	1	28	71	0	0

Tabella 4.12: Workload sintetico LL

4.3.4 Applicazione del workload HL_c e creazione del LL'_c

Pertanto, a partire dalla caratterizzazione di alto livello, viene generato un workload sintetico composto da solo 22 richieste, considerando che all'inizio si era partiti con 45 richieste HTTP. Il carico poi è sottoposto al SUT sempre tramite il tool Apache JMeter 5.5. Le impostazioni del test plan nel tool rimangono le stesse di quelle descritte all'inizio, con l'unica differenza che ora in ogni thread group ci saranno meno richieste.

Si raccolgono dunque i dati di basso livello quando al SUT viene applicato il workload sintetico prescelto, analogamente a quanto descritto precedentemente si valuta il LL'.

4.3.5 Workload Characterization LL'

Si procede alla workload characterization di LL' con lo stesso procedimento applicato al LL:

4.3.5.1 Preprocessing

Si effettua il preprocessing dei dati per ridurre le dimensioni preliminarmente:

Eliminazione delle colonne costanti e colonne nominali Le colonne eliminate:

- st

● b

Eliminazione delle colonne correlate Si valuta la matrice delle correlazioni per eliminare altre colonne:

		Multivariato																	
		Correlazioni																	
		r	swpd	free	buff	cache	si	so	bi	bo	in	os	us	sy	id	wa			
r	1,0000	-0,1347	-0,0035	-0,2377	-0,0641	0,0871	0,1575	0,2063	0,1606	0,2687	0,2614	0,2074	0,3964	-0,3849	0,1359				
swpd	-0,1347	1,0000	-0,2661	0,0185	0,5131	-0,1331	-0,4939	-0,0143	-0,4969	-0,2824	-0,3052	-0,1832	-0,2832	0,2713	-0,0643				
free	-0,0035	-0,2661	1,0000	-0,6142	-0,9414	0,1856	0,2784	-0,0501	0,2739	0,2325	0,3517	0,0275	0,1772	-0,1861	0,1558				
buff	-0,2377	0,0185	-0,6142	1,0000	0,6205	-0,2320	-0,2283	-0,2668	-0,2244	-0,4932	-0,5557	-0,2125	-0,6475	0,6447	-0,3339				
cache	-0,0641	0,5131	-0,9414	0,6205	1,0000	-0,2311	-0,4378	-0,0698	-0,4349	-0,3521	-0,4558	-0,1482	-0,3428	0,3507	-0,2245				
si	0,0871	-0,1331	0,1856	-0,2320	-0,2311	1,0000	0,0705	0,2719	0,0753	0,5612	0,7029	0,1582	0,3329	-0,4496	0,8479				
so	0,1575	-0,4939	0,2784	-0,2283	-0,4378	0,0705	1,0000	0,3135	0,9985	0,3416	0,3138	0,4143	0,4037	-0,3992	0,1315				
bi	0,2063	-0,0143	0,2739	-0,2244	-0,4349	0,0753	0,9985	0,3163	0,4976	0,4837	0,3911	0,4772	-0,5392	0,5737					
bo	0,1606	-0,4969	0,2739	-0,2244	-0,4349	0,0753	0,3163	1,0000	0,3448	0,3206	0,4240	0,4095	-0,4060	0,1392					
in	0,2687	-0,2824	0,2325	-0,4932	-0,3521	0,5612	0,3416	0,4976	0,3448	1,0000	0,8378	0,2986	0,8151	-0,8509	0,6358				
cs	0,2614	-0,3052	0,3517	-0,5557	-0,4558	0,7029	0,3138	0,4837	0,3200	0,8378	1,0000	0,3313	0,7314	-0,7975	0,7560				
us	0,2074	-0,1832	0,0275	-0,2125	-0,1482	0,1582	0,4143	0,3911	0,4240	0,2986	0,3313	1,0000	0,3887	-0,4400	0,3419				
sy	0,3964	-0,2832	0,1772	-0,6475	-0,3428	0,3329	0,4037	0,4772	0,4095	0,8151	0,7314	0,3887	1,0000	-0,9865	0,4416				
id	-0,3849	0,2713	-0,1861	0,6447	0,3507	-0,4496	-0,3992	-0,5392	-0,4060	-0,8509	-0,7975	-0,4400	-0,9865	1,0000	-0,5791				
wa	0,1359	-0,0643	0,1558	-0,3339	-0,2245	0,8479	0,1315	0,5737	0,1392	0,6358	0,7560	0,3419	0,4416	-0,5791	1,0000				

Le correlazioni sono stimate per metodo A livello di riga.

Figura 4.24: Matrice di correlazione del workload di basso livello sintetico

NON c'è nessuna forte correlazione tra le colonne.

4.3.5.2 PCA

In questo caso si applica la stessa analisi dei componenti principali che è stata fatta al LL, scegliendo quindi 8 componenti principali.

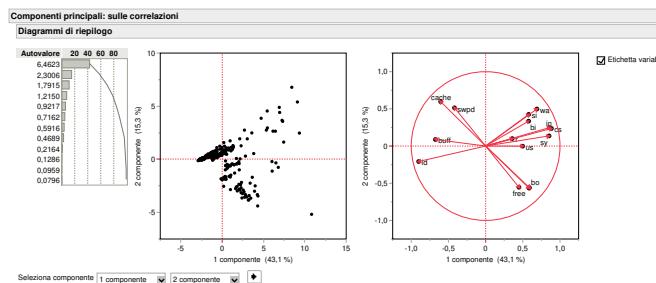


Figura 4.25: Loading Plot della PCA

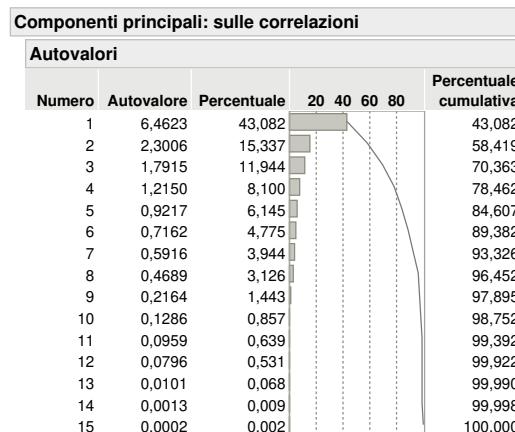


Figura 4.26: Autovalori della PCA del workload di basso livello sintetico

4.3.5.3 Clustering

Anche la scelta del numero dei cluster è la medesima fatta con LL, e dunque, 22.

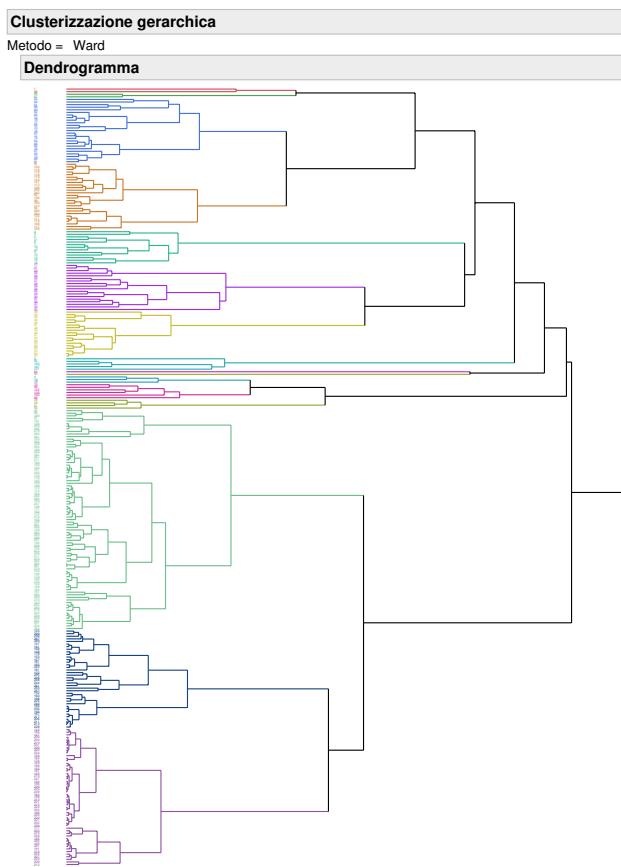


Figura 4.27: Dendrogramma del workload di basso livello sintetico per 8 componenti

Cronologia di clusterizzazione			
Numero di cluster	Distanza	Leader	Subordinato
2	15.44150047	1	4
3	14.76221952	1	84
4	13.83668956	1	2
5	12.61687229	1	3
6	12.46175268	84	86
7	12.29738796	3	17
8	10.76848975	1	22
9	9.21567106	17	38
10	9.17598104	95	124
11	8.09277416	124	142
12	7.99068256	4	39
13	7.09018346	1	85
14	6.79237811	22	91
15	5.67713896	4	75
16	5.24346073	1	88
17	5.08317801	95	123
18	4.92595010	17	24
19	4.88677789	2	92
20	4.73538013	24	35
21	4.49081519	92	121
22	4.10867337	22	63
23	4.04898664	91	94
24	3.74022849	124	152
25	3.49579113	75	89
26	3.49321105	22	62
27	3.44899717	3	6
28	3.23017171	38	41
29	3.15766957	6	13

Figura 4.28: Cronologia di clusterizzazione del workload di basso livello sintetico per 8 componenti

4.3.5.4 Analisi della devianza persa

Si valuta la devianza persa utilizzando lo script Matlab "deviance.m" ed il risultato ottenuto afferma che viene persa una devianza del 13,56%.

4.3.5.5 Workload Sintetico LL'c

Il Workload sintetico LL'c è:

r	b	swpd	free	buff	cache	si	so	bi	bo	in	cs	us	sy	id	wa	st
4	0	504200	15196	4308	130964	1560	2722	12944	3142	1183	1932	5	50	35	10	0
3	0	561592	53360	1436	139752	1732	288	12388	288	2306	2348	5	75	7	13	0
3	1	562104	14000	1440	138100	228	236	31784	280	2659	2349	2	81	5	13	0
1	0	548480	69276	2248	125976	0	0	0	0	1733	1065	1	42	57	0	0
1	0	555704	21268	2132	164560	0	192	0	196	1276	1122	0	47	53	0	0
1	0	510120	57892	3160	107008	0	1260	0	1260	1588	1452	1	36	64	0	0
0	0	529160	91804	2612	98800	36	2320	40	2320	1869	1631	0	49	51	0	0
0	1	546640	119104	2364	83896	16	2072	192	2072	1715	1180	1	40	59	0	0
6	1	555376	128340	2020	84480	52	3736	96	3780	1623	1356	0	47	53	1	0
1	0	556160	122668	1904	92860	44	0	44	0	1218	653	0	29	71	1	0
13	0	505224	22204	3480	126360	1148	1728	3732	1728	2209	2228	1	69	24	6	0
14	0	556728	40000	2140	157848	0	0	0	0	1539	1335	1	61	39	0	0
12	0	555704	46580	2592	151356	32	0	22928	68	2186	1557	0	60	37	3	0
1	2	551816	19192	1800	144144	8	3400	74048	3404	2586	2266	1	81	9	10	0
3	0	561840	14504	1196	144716	32	9460	21792	9464	2014	1565	5	85	5	5	0
2	0	524024	66360	2696	103296	3508	2060	4408	2060	2276	2109	1	67	26	6	0
1	0	556728	30788	2140	157776	2596	0	6280	304	2159	2368	1	59	28	13	0
0	0	550528	107916	1912	92948	4652	0	4652	28	2551	2925	1	43	34	22	0
4	0	555976	40392	3160	152476	0	0	0	28	1541	817	0	35	64	1	0
0	0	555456	31948	3556	164032	0	0	0	28	1364	736	0	29	71	0	0
0	0	555968	30352	3472	163744	0	0	0	28	1156	703	1	22	77	0	0
0	0	555448	30060	3524	164104	0	0	0	28	849	531	0	14	85	1	0

Tabella 4.13: Workload sintetico LL'c

Si noti che viene eseguita la stessa analisi in modo tale che si possano confrontare due dataset con lo stesso numero di colonne nella verifica della bontà della caratterizzazione del workload LL'c e LLc.

4.3.6 Verifica della bontà della caratterizzazione del workload: Confronto tra LL_c e LL'_c

Come ultimo step del seguente homework, **si va a verificare se i due workload di basso livello ottenuti tramite la caratterizzazione siano statisticamente differenti o meno**, al fine di capirne la bontà. In particolare, nella Figura 4.6, viene mostrato come l'obiettivo di questo procedimento sia quello di affermare che il workload sintetico LL'_c ottenuto tramite la caratterizzazione approssimi correttamente il workload reale LL_c e che pertanto le richieste fatte dai due workload siano pressoché equiparabili. Al fine di effettuare la data validation dei due workload, si segue il seguente pseudocodice:

Algorithm 2 Procedimento per la validazione di LL'c e LLc

```

1: procedure VERIFICA_BONTA'
2: Per ogni componente principale si controlla la normalità dei dati di  $LL'_c$  e  $LL_c$  (Controllo visivo è preferito):
3:   if I dati non sono distribuiti normalmente then
4:     Applicare un test non parametrico (Ranksumtest)
5:   else Verificare se tra i dati c'è la stessa varianza (omoschedasticità)
6:     if La varianza è uguale then
7:       Applicare two sample t-test
8:     else Applicare two sample t-test con varianza non uguale

```

In particolare, come è possibile notare dalla Figura 4.29, le componenti principali di entrambi i workload non seguono la distribuzione normale e per questo motivo possiamo utilizzare un test non parametrico per verificare la statistica differenza tra i due workload. Inoltre, per una maggiore sicurezza si è deciso di effettuare anche il test di Kolmogorov-Smirnov (su MATLAB) per verificarne la normalità, anche in questo caso l'ipotesi nulla di normalità è stata rigettata in quanto si è ottenuto un valore pari ad 1.

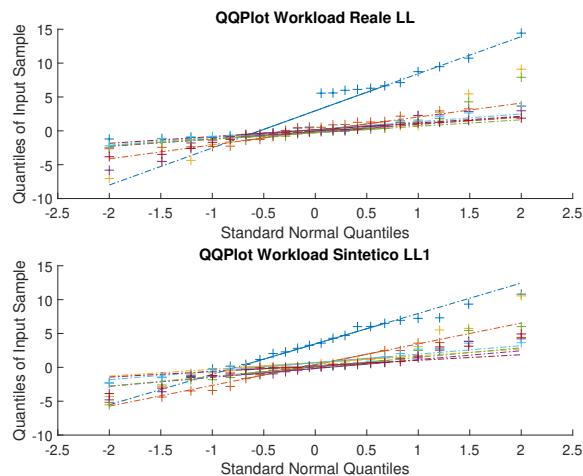


Figura 4.29: QQPlot delle componenti principali dei workload

Successivamente, si è deciso di verificare anche l'omoschedasticità (per questioni di completezza) delle componenti per avere maggiori informazioni sui dati, dai box plot di Figura 4.30 possiamo notare come le varianze non siano uguali in quanto le dimensioni dei box non sono simili tra le componenti in questione.

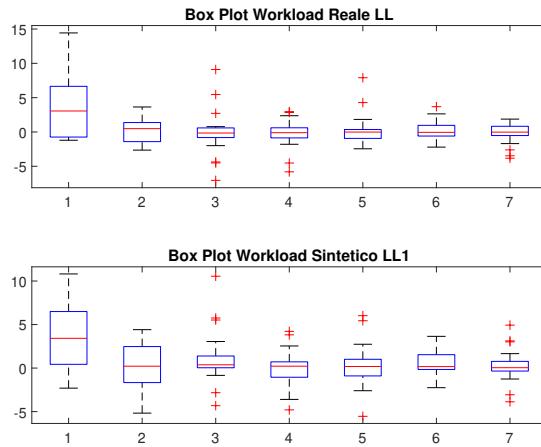


Figura 4.30: Box Plot delle componenti principali dei workload

Determinato, dunque, l'esigenza di eseguire un test non parametrico, si è proceduto all'esecuzione del test di Wilcoxon, tale test assume come ipotesi nulla la seguente:

- H_0 : i due campioni indipendenti provengono da distribuzioni di dati che hanno la stessa mediana

Tale test, effettuato per ogni componente principale, ha restituito un valore pari a 0 indicando che l'ipotesi nulla non può essere rigettata, e affermando che le componenti dei due workload non sono statisticamente differenti. Tale conclusione può essere dedotta anche guardando i p-value in Figura 4.31, sempre ottenuti con tale test.

Workspace	
Name	Value
h_wilc	[0,0,0,0,0,0]
i	7
N	8
norm_wl_real	1
norm_wl_synth	1
p_value_norm_real	4.7885e-04
p_value_norm_sy...	3.4023e-08
p_wilc	[0.8235,0.7336,0.0933,0.8419,0.7872,0.4047,0.5186]
real	22x7 double
synthetic	22x7 double

Figura 4.31: Risultati del test di Wilcoxon

In definitiva, si può dedurre che il workload sintetico LL'_c generato a partire dalla caratterizzazione del workload reale HL_c di alto livello produce effetti statisticamente equivalenti a quello reale sul sistema in analisi.

4.4 Design of Experiments

Il Design of Experiment (DOE) è un metodo statistico utilizzato per identificare i fattori che influiscono su una determinata risposta (output del sistema) e di determinare come i fattori in input interagiscono tra loro. Una corretta analisi degli esperimenti aiuta anche a separare gli effetti dei vari fattori che potrebbero influenzare le prestazioni, determinando se un fattore è significativo o/e importante oppure se le differenze osservate dipendono semplicemente da variazioni casuali dovute ad errori di misura e/o parametri non controllabili. Inoltre, il design of experiments è fondamentale anche per ridurre il numero di esperimenti, quindi il costo, necessario al testing del sistema.

4.4.1 Configurazione del piano di DOE

Si caratterizza la configurazione del DOE (**full factorial design**); i fattori che influenzano il sistema sono descritti nel seguente schema:

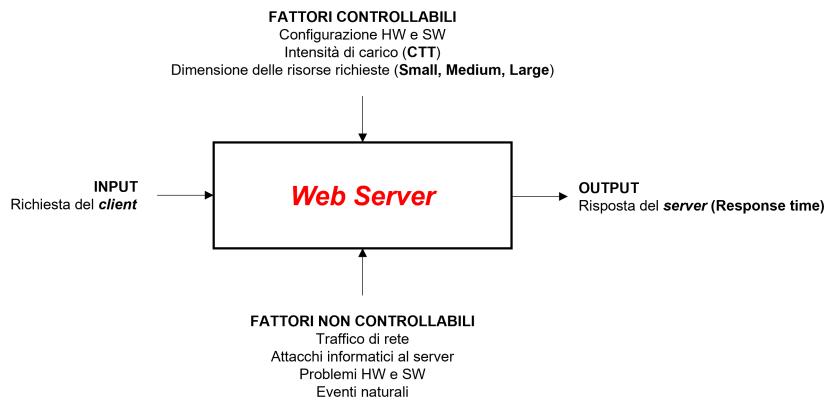


Figura 4.32: Schema dei Fattori

Ogni fattore presenta i rispettivi **livelli**

- **Intensità:** intensità di carico
 - Livello 1: 25% (750 CTT) dell'usable capacity (3000 CTT)
 - Livello 2: 50% (1500 CTT) dell'usable capacity (3000 CTT)
 - Livello 3: 75% (2250 CTT) dell'usable capacity (3000 CTT)
- **DimensioneRisorsa:**
 - Livello 1: Small (10 KB)
 - Livello 2: Medium (1 MB)
 - Livello 3: Large (10 MB)

Per ogni combinazione sono state eseguite 5 ripetizioni per un **totale di $3^2 * 5 = 45$ esperimenti eseguiti tramite JMeter, della durata di un minuto ciascuno (45 minuti totali)**. La **risposta di interesse** è il **Response Time** (calcolato come la media dell'Elapsed Time). L'obiettivo è dunque determinare l'influenza di tali fattori sulla risposta di interesse. In particolare si vuole verificare l'importanza e la significatività dei fattori:

- **Importanza:** un fattore è importante se esprime una buona percentuale di variazione rispetto alla variazione totale del sistema.
- **Significativo:** un fattore è significativo se il suo contributo alla spiegazione di un fattore è maggiore rispetto al contributo dato dall'errore alla spiegazione del medesimo fattore. Un fattore è significativo, quindi, se i suoi effetti sul sistema non sono dovuti alla casualità.

NOTA: La differenza tra i due è che l'importanza non è un concetto statistico mentre la significatività non è un parametro soggettivo. Il **piano fattoriale** può essere costruito su JMP, come in Figura 4.33.

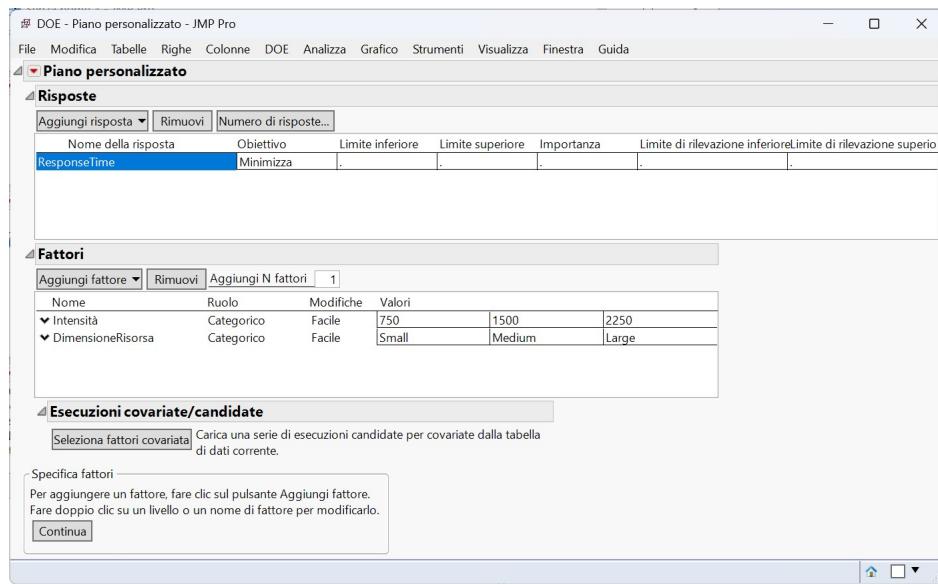


Figura 4.33: Creazione piano DOE

In seguito, si è riempito il piano con i risultati ottenuti dagli esperimenti come mostrato nel seguente estratto:

	Intensità	DimensioneRisorsa	ResponseTime
1	750	Large	443,4417989
2	750	Large	616,4878398
3	750	Large	980,839233
4	750	Large	1672,744337
5	750	Large	235,2506527
6	750	Medium	20,81007752
7	750	Medium	19,32684825
8	750	Medium	23,66364812
9	750	Medium	20,8072445
10	750	Medium	24,0931436
11	750	Small	14,83203125
12	750	Small	7,769728331
13	750	Small	13,21345408
14	750	Small	8,103492885
15	750	Small	8,125485123
16	1500	Large	1941,870298
17	1500	Large	1874,513174
18	1500	Large	2073,484765
19	1500	Large	1982,386544
20	1500	Large	1958,575916

Figura 4.34: Estratto del Piano DOE

4.4.2 Analisi dell'importanza

Per verificare l'importanza di un fattore all'interno del piano, è necessario andare a calcolare alcuni parametri:

- $SS_{\text{Intensità}}$ (Sum of square factor Intensità): variazione spiegata dall'intensità;
- $SS_{\text{DimensioneRisorsa}}$ (Sum of square factor DimensioneRisorsa): variazione spiegata dalla dimensione della risorsa;
- $SS_{\text{IntensitàDimensioneRisorsa}}$ (Sum of Square Factor IntensitàDimensioneRisorsa): variazione spiegata dalla interazione tra dimensione della risorsa e intensità;
- SS_{Errore} : variazione spiegata dall'errore.

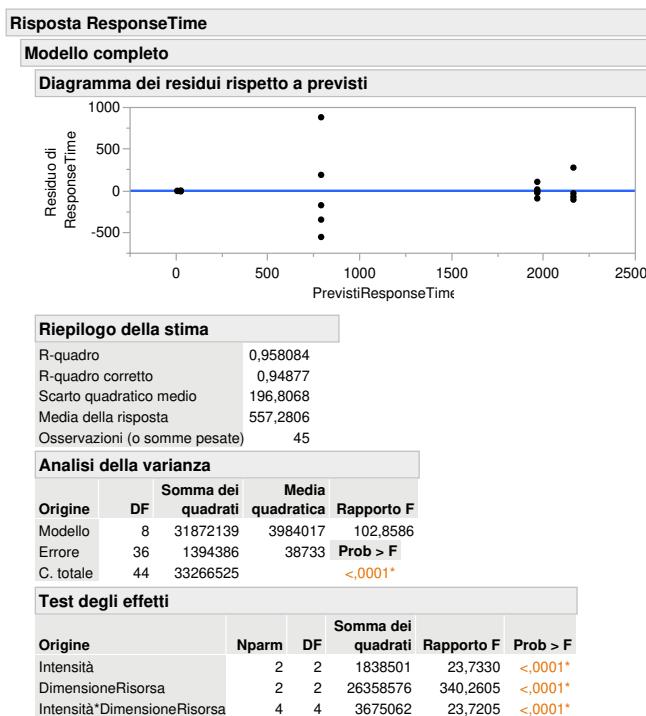


Figura 4.35: Stima Modello

Tramite il tool JMP è possibile valutare le variazioni:

Dalla Figura 4.35 sono state ottenute le Sum of Square dei fattori, dell'interazione dei fattori, dell'errore e quella totale per valutare la percentuale di variazione spiegata da ognuno.

Pertanto:

$$\frac{SS_{\text{Intensità}}}{SS_{\text{Totale}}} = \frac{1838501}{33266525} = 0,0552657 \simeq 5,53\% \quad (4.6)$$

$$\frac{SS_{\text{DimensioneRisorsa}}}{SS_{\text{Totale}}} = \frac{26358576}{33266525} = 0,7923453 \simeq 79,23\% \quad (4.7)$$

$$\frac{SS_{\text{Intensità*DimensioneRisorsa}}}{SS_{\text{Totale}}} = \frac{3675062}{33266525} = 0,1104732 \simeq 11,05\% \quad (4.8)$$

$$\frac{SS_{\text{Errore}}}{SS_{\text{Totale}}} = \frac{1394386}{33266525} = 0,0419155 \simeq 4,19\% \quad (4.9)$$

Dalla valutazione si ottiene dunque che **il fattore con l'importanza maggiore è la Dimensione della Risorsa che spiega il 79,23% della varianza totale (complessiva)**, il fattore d'intensità spiega solo il 5,53%, mentre il fattore d'interazione spiega circa l'11,05%. I termini di errore spiegano all'incirca il 4% della varianza complessiva, dunque, della varianza totale sui dati solo il 4% è dato da un errore sperimentale di misura.

4.4.3 Analisi di significatività statistica dei fattori

Si valuta dunque la percentuale di variabilità spiegata da un fattore rispetto a quella spiegata dall'errore. Per studiare la significatività dei fattori considerati si applica un metodo detto **ANOVA (ANalysis Of VAriance)**. Il metodo preciso da applicare dipende da alcune caratteristiche della distribuzione, in particolare la **normalità** e la **omoschedasticità**.

4.4.3.1 Normalità

Si verifica se gli **errori** sono approssimabili ad una distribuzione normale. In particolare si calcolano i **residui** e si valuta il QQPlot dei residui per testare visivamente la normalità. Dalla Figura 4.36 il QQPlot ci evidenzia la **non normalità** dei dati.

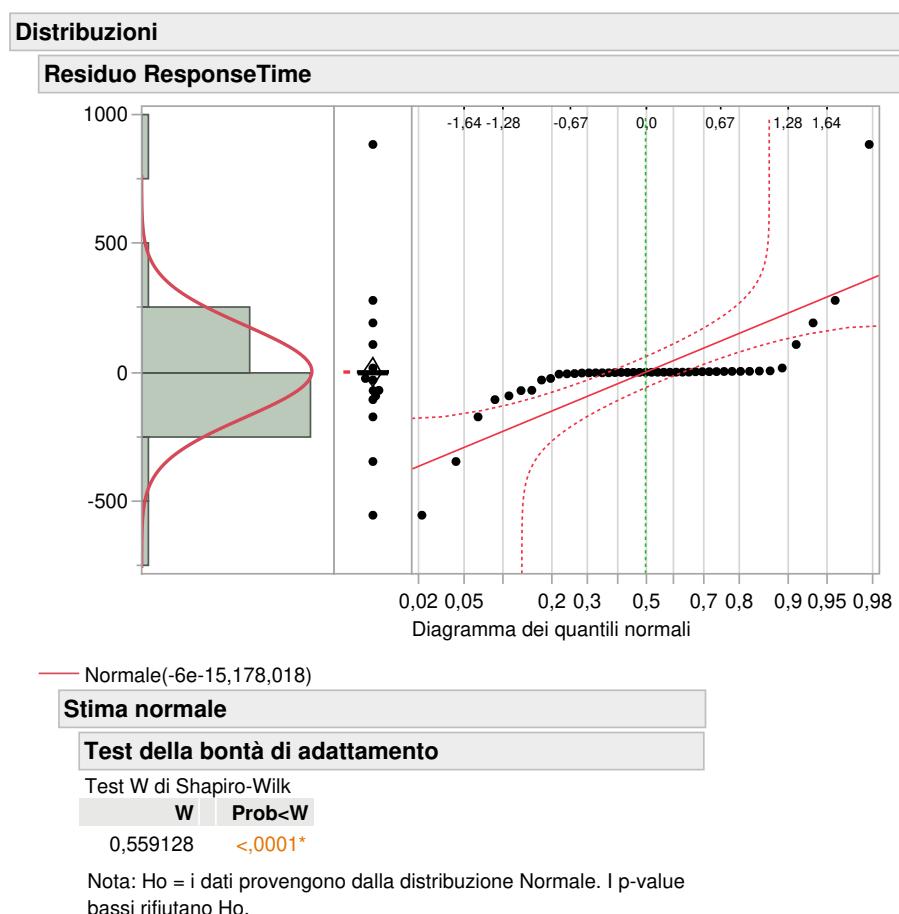


Figura 4.36: Distribuzione dei residui di ResponseTime

È possibile verificare il test visivo con un test non parametrico di Shapiro-Wilk ($N < 2000$). Sempre dalla Figura 4.36 è possibile verificarlo in quanto sia data l'ipotesi: H_0 : i dati provengono dalla distribuzione normale, il p-value calcolato è molto basso rigettando l'ipotesi H_0 . Confermando che i dati in esame **non sono distribuiti normalmente**.

4.4.3.2 Omoschedasticità

Nonostante sia superfluo, si è deciso di verificare ugualmente l'omoschedasticità. Si verifica se la deviazione standard degli errori è costante. Anche in questo caso ci si avvale del test visivo:

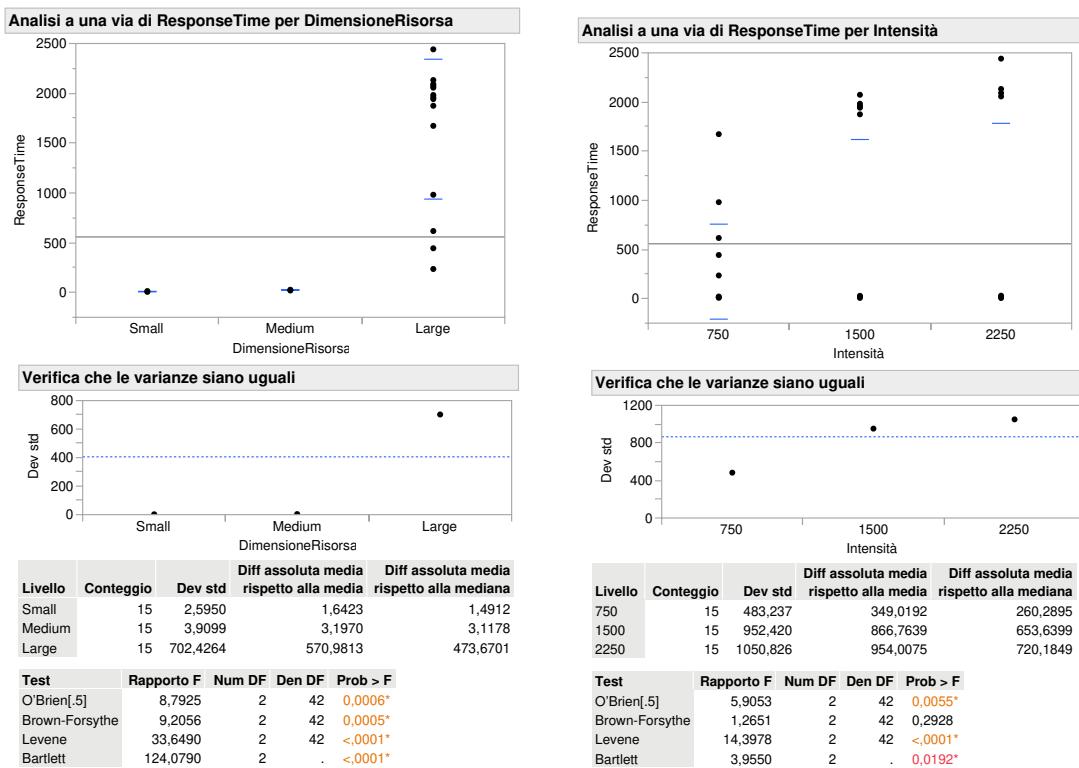


Figura 4.37: Valutazione dell'omoschedasticità

Per verificare l'omoschedasticità dei residui è possibile effettuare un test visivo, tracciando i punti a seconda dei livelli dei fattori. Da tale grafico, si può notare infatti come la dispersione dei punti nello scatter plot non è costante, per cui è possibile rifiutare l'ipotesi di omoschedasticità dei residui sia nel caso della DimensioneRisorsa che nel caso del fattore di Intensità. Inoltre, anche guardando ai valori delle deviazioni standard dell'intensità delle risorse e della dimensione della risorsa si nota che esse sono differenti, pertanto, si può dedurre che non ci sia omoschedasticità.

Per confermare il test visivo ci si avvale anche di un test non parametrico, in particolare, del test di O'Brien, il quale conferma anche esso la **non omoschedasticità (Eterschedasticità)** in quanto il p-value è molto basso, e quindi, l'ipotesi nulla di varianze uguali è rigettata.

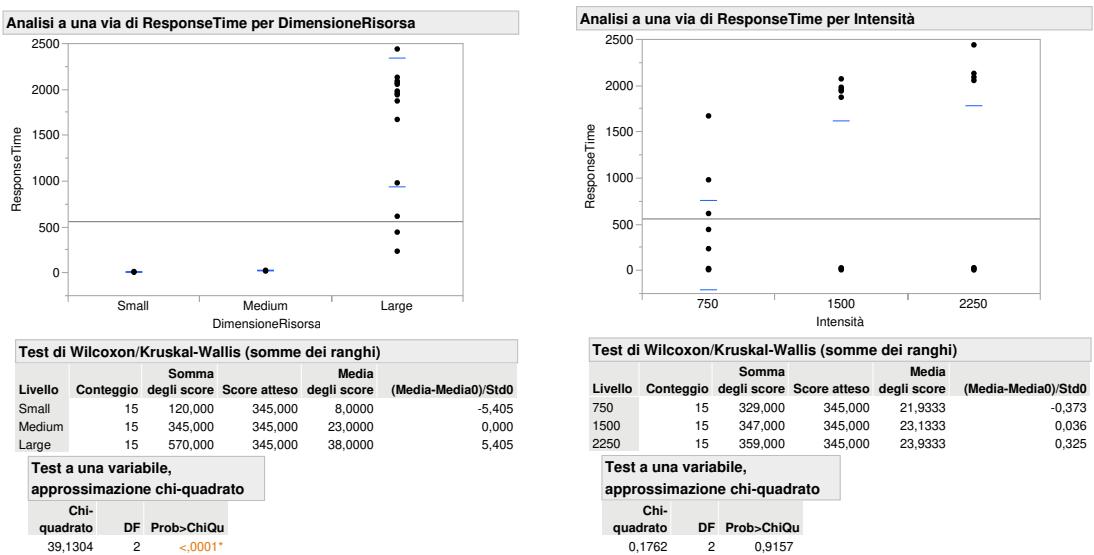
4.4.3.3 Anova

Dalle analisi della normalità e dell'omoschedasticità è possibile valutare il test d'applicare nella seguente tabella:

Normalità	Omoschedasticità	Test Anova
Si	Si	F-test
No	Si	Kruskal-Wallis test
Si	No	Welch's test
No	No	Kruskal-Wallis test o Friedman test

Tabella 4.14: Combinazioni Normalità e Omoschedasticità per la scelta del test d'applicare

Essendo che non c'è normalità, non c'è omoschedasticità si opta per un test di **Kruskal-Wallis**, detto anche test della somma dei ranghi, per **verificare quale fattore sia statisticamente significativo**.

**Figura 4.38:** Test di Kruskal-Wallis

Si nota dalla Figura 4.38 che l'unico fattore significativo a livello statistico è la **DimensioneRisorsa**. Questo perchè il p-value della DimensioneRisorsa è molto basso **rigettando l'ipotesi della non significatività statistica**. Mentre l'**Intensità** delle risorse risulta non statisticamente significativo in quanto il **p-value indica che l'ipotesi H_0 non è stata rigettata**.

NOTA: Il test di Kruskal-Wallis utilizza la distribuzione Chi-square per la valutazione del p-value, in quanto i dati non sono distribuiti normalmente.

In conclusione: si può dire che il fattore DimensioneRisorsa è sia importante e sia significativo.

5 Regression

Contenuti

5.1	Definizione degli obiettivi e overview	66
5.2	Dataset Mail Server 1	66
5.2.1	Valutazione delle rette di regressione e R^2	66
5.2.2	Test di Normalità	67
5.2.3	Test di Mann-Kendall	68
5.2.4	Pendenza e intercetta della retta di regressione	68
5.3	Dataset Mail Server 2	69
5.3.1	Valutazione delle rette di regressione e R^2	69
5.3.2	Test di Normalità	70
5.3.3	Test di Mann-Kendall	71
5.3.4	Pendenza e intercetta della retta di regressione	71
5.4	Confronto tra Mail Server 1 e Mail Server 2	72
5.5	Dataset - OS1	72
5.5.1	Valutazione delle rette di regressione e R^2	72
5.5.2	Test di Normalità	73
5.5.3	Test di Mann-Kendall	74
5.5.4	Pendenza e intercetta della retta di regressione	75
5.6	Dataset - OS2	76
5.6.1	Valutazione delle rette di regressione e R^2	76
5.6.2	Test di Normalità	77
5.6.3	Test di Mann-Kendall	78
5.6.4	Pendenza e intercetta della retta di regressione	79
5.7	Dataset - OS3	80
5.7.1	Valutazione delle rette di regressione e R^2	80
5.7.2	Test di Normalità	81
5.7.3	Test di Mann-Kendall	82
5.7.4	Pendenza e intercetta della retta di regressione	83
5.7.5	Confronto tra OS1-OS2-OS3	84
5.8	VmRes1 - Failure Prediction	84
5.8.1	Valutazione della retta di regressione e R^2	84
5.8.2	Test di Normalità	85
5.8.3	Test di Mann-Kendall	85
5.8.4	Pendenza e intercetta della retta di regressione	86
5.8.5	Valutazione Failure Prediction	86
5.9	VmRes2 - Failure Prediction	87
5.9.1	Valutazione delle rette di regressione e R^2	87
5.9.2	Test di Normalità	87
5.9.3	Test di Mann-Kendall	88
5.9.4	Pendenza e intercetta della retta di regressione	88
5.9.5	Valutazione Failure Prediction	89
5.10	VmRes3 - Failure Prediction	89
5.10.1	Valutazione delle rette di regressione e R^2	89
5.10.2	Test di Normalità	90
5.10.3	Test di Mann-Kendall	90
5.10.4	Pendenza e intercetta della retta di regressione	90
5.10.5	Valutazione Failure Prediction	91

5.1 Definizione degli obiettivi e overview

Un modello regressivo è un tipo di modello statistico che cerca di stabilire una relazione tra una variabile dipendente (risposta o stimata) e una o più variabili indipendenti (di predizione). La relazione viene stabilita attraverso una funzione lineare che descrive il modello.

L'obiettivo è **lo studio del trend ovvero l'analisi di una serie temporale al fine di individuare una tendenza sottostante**. La tendenza può essere positiva, negativa o nulla e viene utilizzata per prevedere i futuri valori della serie temporale.

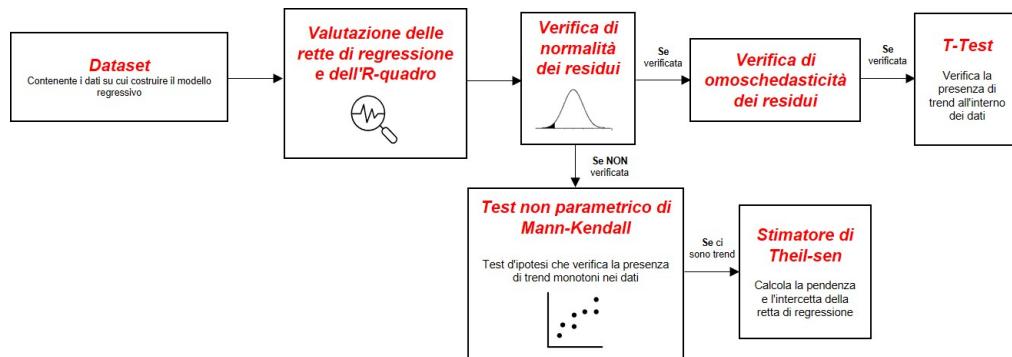


Figura 5.1: Schema di analisi del modello regressivo

5.2 Dataset Mail Server 1

Il dataset è così formato:

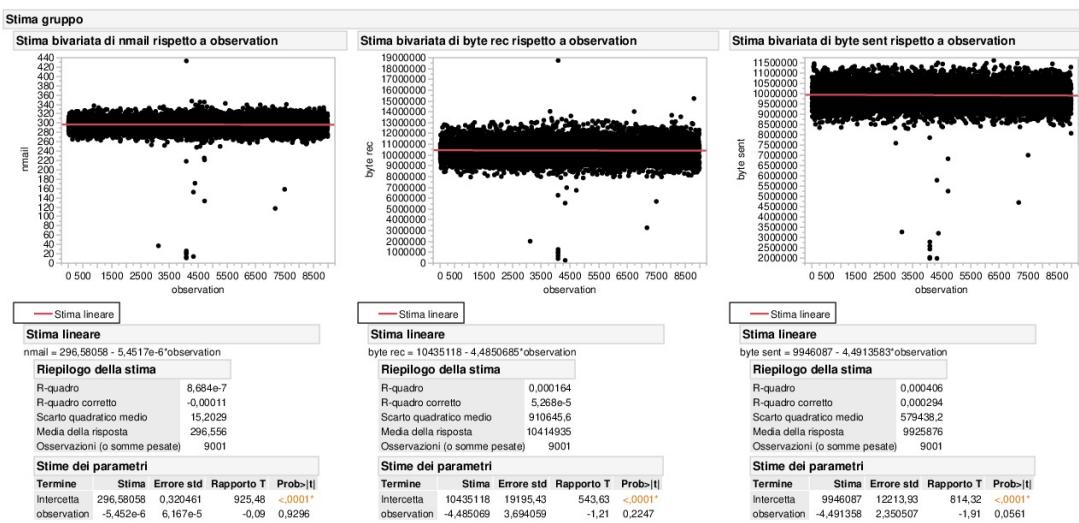
- Variabili di risposta:
 - Nbyte
 - Byte_rec
 - Byte_sent
- Variabili di predizione
 - Observation

5.2.1 Valutazione delle rette di regressione e R^2

Viene applicato il modello regressivo lineare la cui risposta è una funzione lineare del predittore. Si valuta su JMP **la stima lineare delle variabili** come mostrato in Figura 5.2.

Nella stima viene indicato l'R-squared, ovvero il coefficiente di determinazione, che misura il legame tra la variabilità dei dati e la correttezza del modello statistico utilizzato, ovvero indica quanto bene le previsioni del modello si adattano ai dati effettivi. Il valore di R-squared varia tra 0 e 1, dove un valore di 1 indica che il modello spiega perfettamente la varianza dei dati, mentre un valore di 0 indica che il modello non spiega affatto la varianza dei dati. In tutti e 3 i casi, **l'R-squared assume un valore molto basso a dimostrazione del fatto che il modello regressivo lineare usato ha uno scarso adattamento ai dati, e dunque, le prestazioni predittive sono insufficienti**.

Se si vuole utilizzare il modello regressivo per fare trend detection, il coefficiente R-squared può essere basso. Tuttavia, occorre verificare la significatività dei parametri della retta regressiva, tramite un opportuno test statistico.

Figura 5.2: Stima lineare e valutazione R^2

5.2.2 Test di Normalità

Per la scelta del test da fare, di tipo parametrico o meno, occorre valutare la distribuzione dei **residui** delle rispettive variabili. In altre parole, nei metodi di inferenza statistica l'ipotesi di normalità dei **residui** implica che le differenze tra i valori previsti dal modello e i valori osservati nei dati seguano una distribuzione normale. In caso contrario, si potrebbe dover considerare di utilizzare un test statistico non parametrico, come quello di Mann-Kendall.

In particolare, si applica il test visivo tramite dei QQplot.

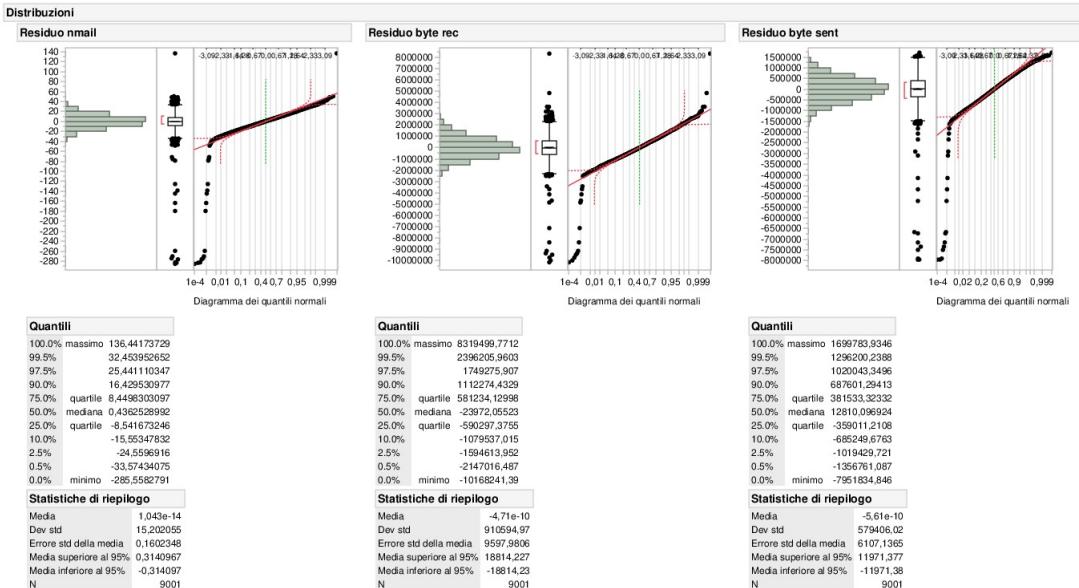


Figura 5.3: Valutazione Normalità

In tutti e 3 i casi le distribuzioni non sono normali, a questo punto dovrebbe essere verificata l'omoschedasticità ma essendo stata rifiutata l'assunzione di normalità degli errori è risultato superfluo effettuare tale verifica. Si valuta dunque il test di Mann-Kendall (test non parametrico).

5.2.3 Test di Mann-Kendall

Il test non parametrico di Mann-Kendall calcola il coefficiente di correlazione τ e un p-value. I valori di τ vanno da -1 a 1. Se il valore è negativo indica che le variabili sono inversamente correlate, ovvero che quando una variabile aumenta, l'altra diminuisce. I valori positivi indicano invece che quando una variabile aumenta, aumenta anche l'altra (diretta proporzionalità). **L'ipotesi nulla:**

H_0 : non è presente un trend monotono nella distribuzione.

Se il p-value è inferiore o uguale a 0,05 significa che il risultato ottenuto è statisticamente significativo per l'analisi. Pertanto l'ipotesi è rigettata.

I risultati:

Multivariato											
Non parametrico: τ di Kendall											
Variable	Variable by	τ di Kendall	Prob> τ	>.8	>.6	>.4	>.2	<.2	<.4	<.6	<.8
observation	byte rec	-0.0090	0.2002								

Figura 5.4: Kendall - Byterec

Multivariato											
Non parametrico: τ di Kendall											
Variable	Variable by	τ di Kendall	Prob> τ	>.8	>.6	>.4	>.2	<.2	<.4	<.6	<.8
observation	byte sent	-0.0142	0.0435*								

Figura 5.5: Kendall - Bytesent

Multivariato											
Non parametrico: τ di Kendall											
Variable	Variable by	τ di Kendall	Prob> τ	>.8	>.6	>.4	>.2	<.2	<.4	<.6	<.8
observation	nmail	-0.0026	0.7164								

Figura 5.6: Kendall - Nmail

Si indicano i valori in tabella e la rispettiva scelta:

Test NON parametrico - Kendall - Dataset 1 - Mail Server			
	τ	p-value	H
Observation - NMail	-0.0026	0.7164	0
Observation - byte.sent	-0.0142	0.0435	1
Observation - byte.rec	-0.009	0.2002	0

Tabella 5.1: Test NON parametrico - Kendall - Dataset 1 - Mail Server

Dalla tabella l'unica variabile che presenta un trend è bytes_sent, mentre le altre due variabili non presentano trend significativi.

5.2.4 Pendenza e intercetta della retta di regressione

Si valuta, dunque, la pendenza e l'intercetta della retta di regressione utilizzando la **procedura di Theil-Sen**. Questa procedura è una **regressione lineare robusta progettata per fornire risultati più precisi e affidabili in presenza di valori outlier nei dati**.

Per il calcolo dei valori si è utilizzato uno script **Python**¹, facendo uso della funzione *theilslopes* fornita dalla libreria SciPy:

¹ Lo script è presente nel repository GitHub dell'elaborato

Regressione Lineare Robusta - Theil e Sen	
Observation (x) - byte.sent (y)	
Intercetta	9959247.527968435
Pendenza	-4.514783992985356
CI inferiore	-8.891050583657588
CI superiore	-0.13067767792081614

Tabella 5.2: Intercetta e Pendenza di Theil e Sen - Dataset 1

Dalla Tabella, si può notare che l'intervallo di confidenza al 95% relativo alla pendenza della retta di regressione non include lo zero, dunque, possiamo affermare che la pendenza, relativa al **modello della popolazione**, ha una probabilità del 95% che sia significativamente diversa da zero.

L'**intercetta** del **modello della popolazione** sarà valutata a valle del calcolo della pendenza. Lo stesso procedimento verrà applicato a tutte le valutazioni successive.

5.3 Dataset Mail Server 2

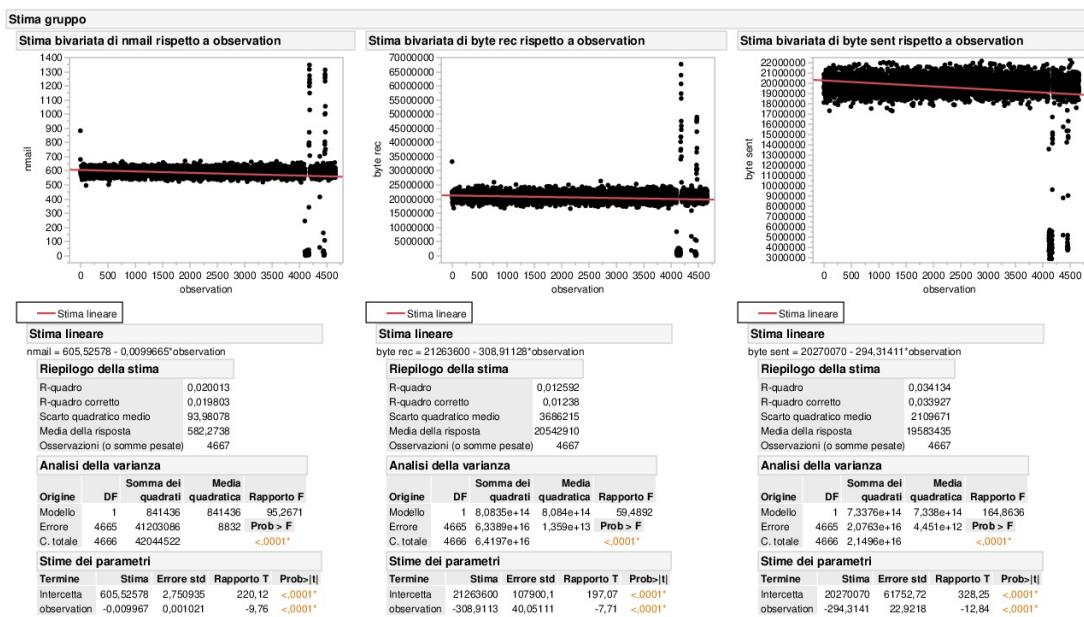
Il dataset è così formato:

- Variabili di risposta:
 - Nbyte
 - Byte_rec
 - Byte_sent
- Variabili di predizione
 - Observation

5.3.1 Valutazione delle rette di regressione e R^2

Viene applicato il modello regressivo lineare la cui risposta è una funzione lineare del predittore. Si valuta su JMP la stima lineare delle variabili, come si nota dalla Figura 5.7.

Nella stima viene indicato anche l'R-quadro. **Anche in questo caso tutti e 3 assumono valore molto basso a dimostrazione del fatto che il modello lineare non riesce ad adattarsi correttamente ai dati, incidendo negativamente sulla sua capacità predittiva.**

Figura 5.7: Stima lineare e valutazione R^2

5.3.2 Test di Normalità

Anche in questo caso per la scelta del tipo di test da applicare è stata valutata la distribuzione dei residui delle rispettive variabili.

In particolare, si applica il test visivo tramite dei QQplot.

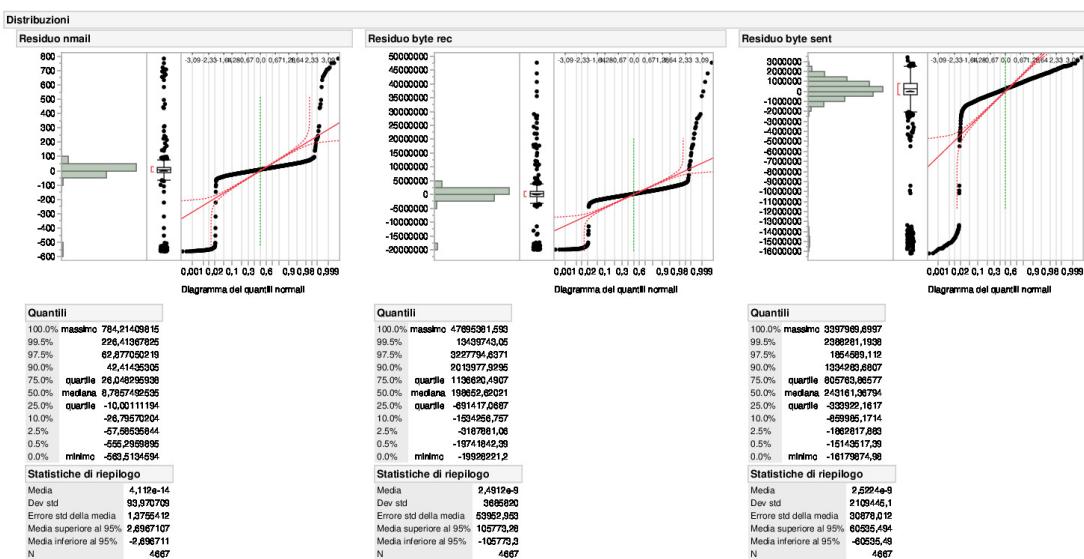


Figura 5.8: Valutazione Normalità

In tutti e 3 i casi le distribuzioni non sono normali, quindi, si evita di verificare l'omoschedasticità. Si valuta dunque il test di Mann-Kendall (test non parametrico).

5.3.3 Test di Mann-Kendall

Si ripete lo stesso procedimento fatto nel primo paragrafo, ottenendo i seguenti risultati:

Multivariato												
Non parametrico: τ di Kendall												
Variable	Variable by	τ di Kendall	Prob> t	-8	-6	-4	-2	0	.2	.4	.6	.8
observation	byte rec	-0.0316	0.0012*									

Figura 5.9: Kendall - Byterec

Multivariato												
Non parametrico: τ di Kendall												
Variable	Variable by	τ di Kendall	Prob> t	-8	-6	-4	-2	0	.2	.4	.6	.8
observation	byte sent	-0.0392	<.0001*									

Figura 5.10: Kendall - Bytesent

Multivariato												
Non parametrico: τ di Kendall												
Variable	Variable by	τ di Kendall	Prob> t	-8	-6	-4	-2	0	.2	.4	.6	.8
observation	nmail	-0.0242	0.0138*									

Figura 5.11: Kendall - Nmail

Si indicano i valori in tabella e la rispettiva scelta:

Test NON parametrico - Kendall - Dataset 2 - Mail Server			
	τ	p-value	H
Observation - NMail	-0.0242	0.0138	1
Observation - byte.sent	-0.0392	0.0001	1
Observation - byte.rec	-0.0316	0.0012	1

Tabella 5.3: Test NON parametrico - Kendall - Dataset 2 - Mail Server

Dalla tabella tutti e 3 i test indicano che è presente un trend significativo nei dati.

5.3.4 Pendenza e intercetta della retta di regressione

Si ripete lo stesso procedimento del paragrafo 1:

Regressione Lineare Robusta - Theil e Sen			
	Observation (x) - NMail (y)	Observation - byte.sent	Observation - byte.rec
Intercetta	592.3936678614098	19903064.10051782	20890131.991364423
Pendenza	-0.0005973715651135006	-34.06305208650625	-48.29360967184802
CI inferiore	-0.0011248593925759281	-50.672672672672675	-77.54773269689737
CI superiore	0.0	-17.374087591240876	-18.993146773272414

Tabella 5.4: Intercetta e Pendenza di Theil e Sen - Dataset 2

Dalla Tabella, si può notare che gli intervalli di confidenza al 95% relativi alle pendenze delle rette di regressione non includono lo zero per byte.sent e byte_rec, dunque, possiamo affermare che tali pendenze, relative al **modello della popolazione**, hanno una probabilità del 95% che siano significativamente diverse da zero. Mentre, per la pendenza della retta di regressione relativa alla variabile NMail si nota che l'intervallo di confidenza al 95% contiene

lo zero e per questo motivo si può affermare che essa ha una probabilità del 95% che sia significativamente uguale a zero.

5.4 Confronto tra Mail Server 1 e Mail Server 2

Come si evince dall'analisi condotta sui due dataset, si è notato che il secondo dataset presenta un numero maggiore di trend nei dati rispetto al primo, ciò potrebbe indicare che ci sono cambiamenti significativi nei dati del secondo dataset rispetto al primo. In generale, è importante valutare i dati e le tendenze all'interno del contesto d'interesse e fare eventuali ulteriori analisi per capire la causa del maggior trend e le implicazioni per l'attività in esame. Nel nostro caso, si vuole valutare un Mail Server in relazione al trend nei dati, per fare ciò possono essere usate diverse metriche:

- Volume delle email: **un trend crescente nel numero delle email (NMail)** potrebbe indicare un aumento dell'utilizzo del server, e quindi la necessità di una maggiore capacità di elaborazione e di archiviazione, quindi effettuare tali valutazioni sul secondo dataset risulterebbe più significativo;
- Tasso di errore: **un trend crescente nel tasso di errore** (aumento della differenza tra **byte-sent** e **byte-rec**) potrebbe indicare problemi con la consegna delle email, dovuto a problemi tecnici, o a indirizzi email non validi; e quindi, anche per questo caso le valutazioni sul secondo dataset sono da preferire;
- Tasso di bounce: **un trend crescente nel tasso di bounce** (diminuzione dei **byte-rec** rispetto ai **byte-sent**) potrebbe indicare problemi con la consegna delle email, dovuto a problemi con i server di destinazione o indirizzi email non validi; per questo il secondo dataset risulta più significativo.

In definitiva, grazie all'analisi sul trend dei dati effettuate sui due dataset, si è in grado di determinare quale dataset risulta più significativo per le predizioni; in particolare, in relazione ad alcune delle metriche che si possono considerare per valutare un Mail Server, e l'importanza di ciascuna dipenderà dalle esigenze specifiche del contesto operativo.

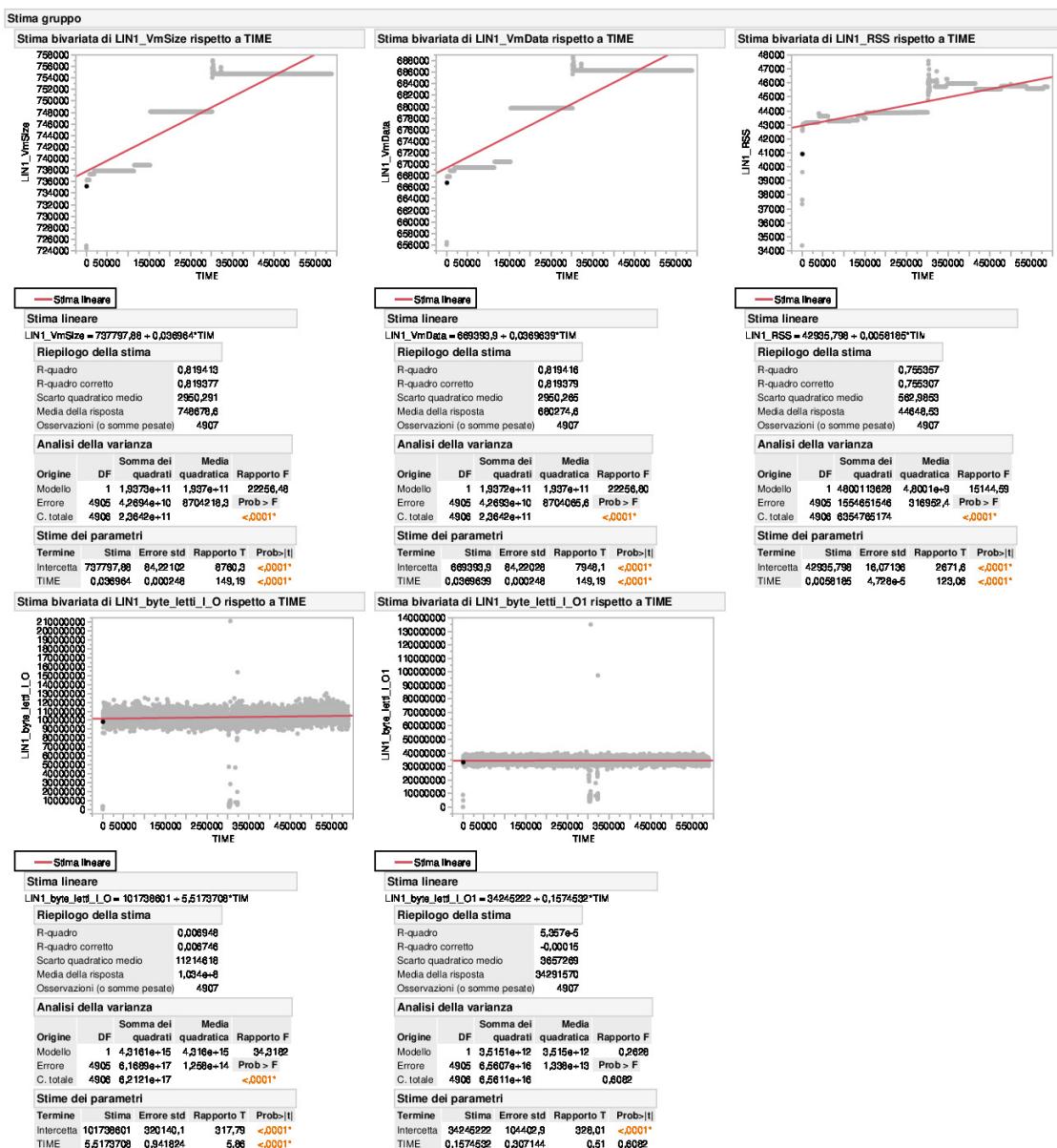
5.5 Dataset - OS1

Il dataset è così formato:

- Variabili di risposta:
 - VmSize
 - VmData
 - RSS
 - Byte_letti_IO
 - Byte_letti_IO1
- Variabili di predizione
 - Time

5.5.1 Valutazione delle rette di regressione e R^2

Viene applicato il modello regressivo lineare la cui risposta è una funzione lineare del preditore. Si valuta su JMP la stima lineare delle variabili, come mostrato in Figura 5.12.

Figura 5.12: Stima lineare e valutazione R^2

Nella stima viene indicato l'R-quadro. In questo caso R-quadro assume valore basso per le variabili ByteLetti IO e ByteLetti IO1 e alto per le altre 3 variabili.

5.5.2 Test di Normalità

Per la scelta del test d'ipotesi da eseguire per valutare la presenza di trend nei dati, è necessario valutare la distribuzione dei residui delle rispettive variabili.

In particolare, si applica il test visivo tramite dei QQplot.

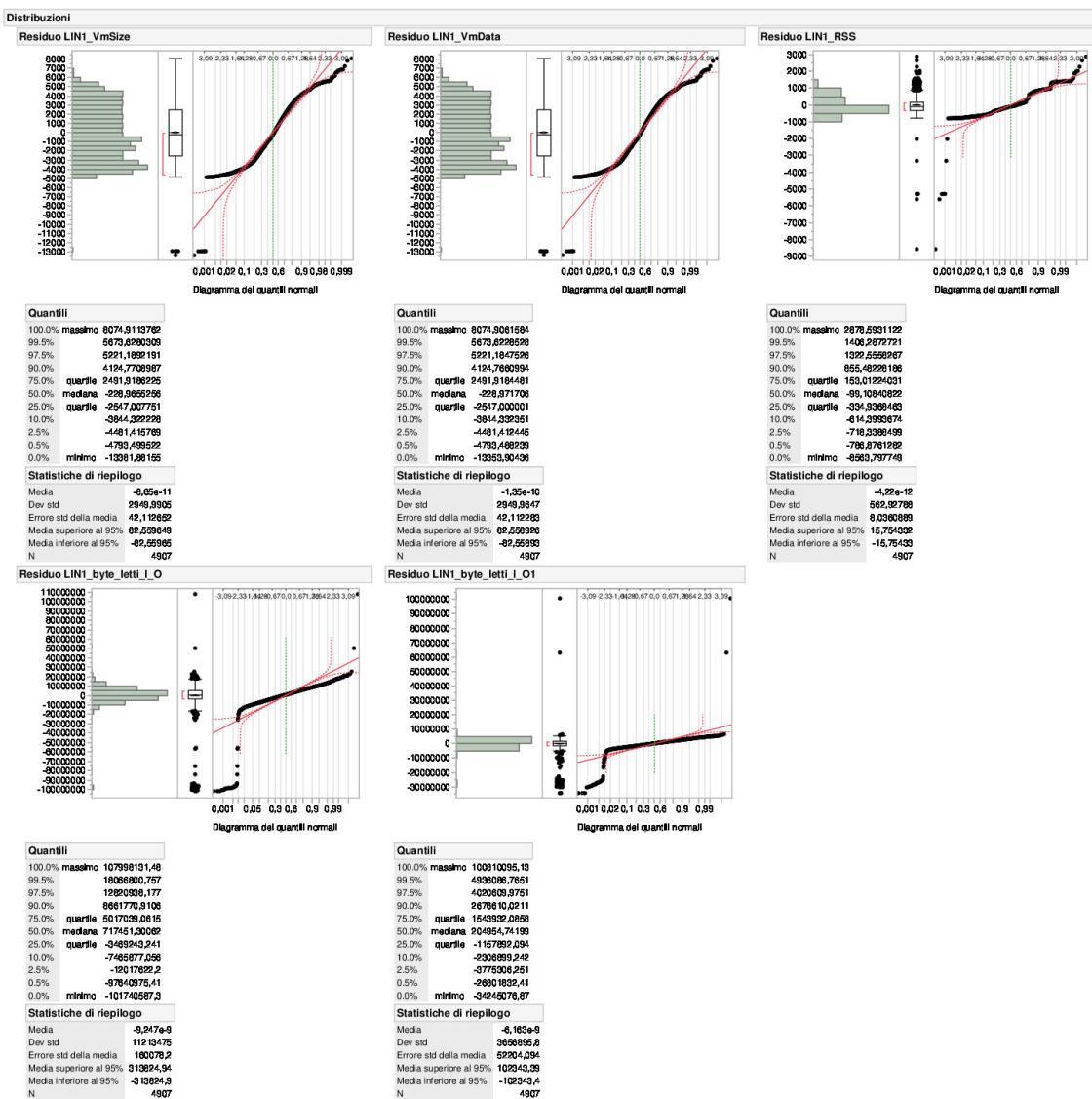


Figura 5.13: Valutazione Normalità

In tutti e 5 i casi le distribuzioni non sono normali, pertanto si omette la verifica dell'omoschedasticità. Si valuta dunque il test di Mann-Kendall (test non parametrico).

5.5.3 Test di Mann-Kendall

Si ripete lo stesso procedimento fatto nel primo paragrafo ottenendo i seguenti risultati:

Multivariato						
Non parametrico: τ di Kendall						
Variable	Variable by	τ di Kendall	Prob> τ	-8	-6	-4
TIME	LIN1_VmData	-0,8076	<0,0001*			

Figura 5.14: Kendall - VmData



Figura 5.15: Kendall - VmSize



Figura 5.16: Kendall - Rss



Figura 5.17: Kendall - Byte-letti-I-O

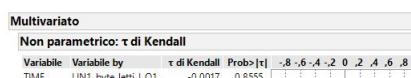


Figura 5.18: Kendall - Byte-letti-I-O1

Si indicano i valori in tabella e la rispettiva scelta:

Test NON parametrico - Kendall - Dataset 1 - VM			
	τ	p-value	H
Time - VmSize	0.8076	0.0001	1
Time - VmData	0.8076	0.0001	1
Time - RSS	0.6827	0.0001	1
Time - Byte.letti.I.O	0.0823	0.0001	1
Time - Byte.letti.I.O1	-0.0017	0.8555	0

Tabella 5.5: Test NON parametrico - Kendall - Dataset 1 - VM

Dalla Tabella 5.5, 4 test su 5 indicano che hanno un trend significativo nei dati.

5.5.4 Pendenza e intercetta della retta di regressione

Si ripete lo stesso procedimento del paragrafo 1:

Regessione Lineare Robusta - Theil e Sen				
	Time (x) - VmSize(y)	Time (x) - VmData(y)	Time (x) - RSS(y)	Time (x) - Byte.letti.I.O (y)
Intercetta	738854.3947841726	670450.3947841726	42428.80701754386	102763229.503125
Pendenza	0.03157224220623501	0.03157224220623501	0.004970760233918129	4.737578125
CI inferiore	0.030715952172645087	0.030715952172645087	0.004876649454962708	3.6739155483258776
CI superiore	0.03232097662647015	0.03232097662647015	0.005055788005578801	5.80459886547812

Tabella 5.6: Intercetta e Pendenza di Theil e Sen - Dataset 1 - VM

Dalla Tabella, si può notare che gli intervalli di confidenza al 95% relativi alle pendenze delle rette di regressione non includono lo zero per nessuna variabile, dunque, possiamo affermare che le pendenze, relative al **modello della popolazione**, hanno una probabilità del 95% che siano significativamente diverse da zero.

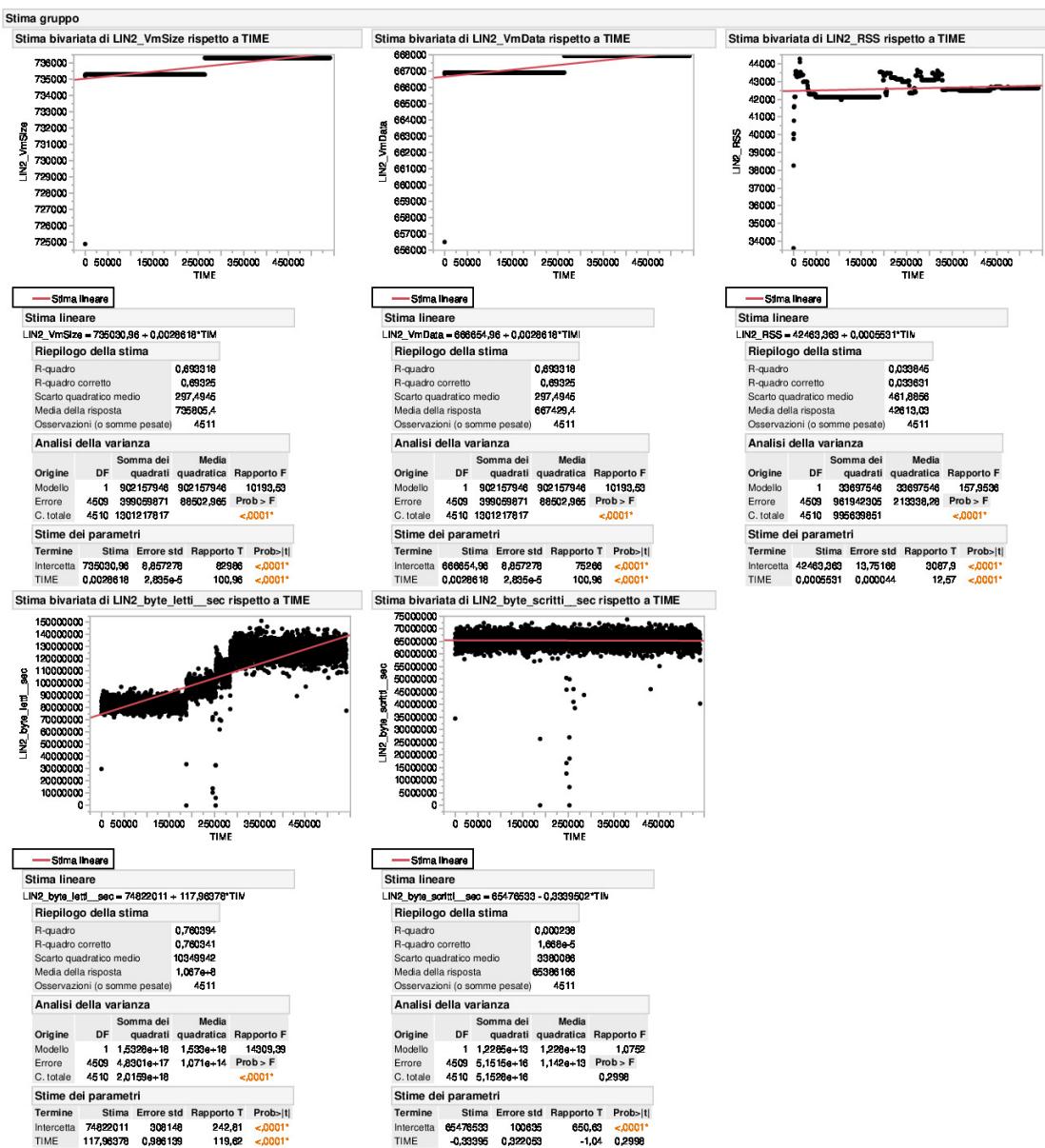
5.6 Dataset - OS2

Il dataset è così formato:

- Variabili di risposta:
 - VmSize
 - VmData
 - RSS
 - Byte_letti_sec
 - Byte_scritti_sec
- Variabili di predizione
 - Time

5.6.1 Valutazione delle rette di regressione e R^2

Viene applicato il modello regressivo lineare la cui risposta è una funzione lineare del predittore. Si valuta su JMP la stima lineare delle variabili, come si evince dalla Figura 5.19.

Figura 5.19: Stima lineare e valutazione R^2

Nella stima viene indicato l'R-squared. In questo caso R-squared assume valore basso per le variabili `Byte_scritti_sec` e `RSS` e alto per le altre 3 variabili.

5.6.2 Test di Normalità

Per la scelta del test da effettuare per controllare la presenza di trend nei dati è stata valutata la distribuzione dei residui delle rispettive variabili.

In particolare, si applica il test visivo tramite dei QQplot.

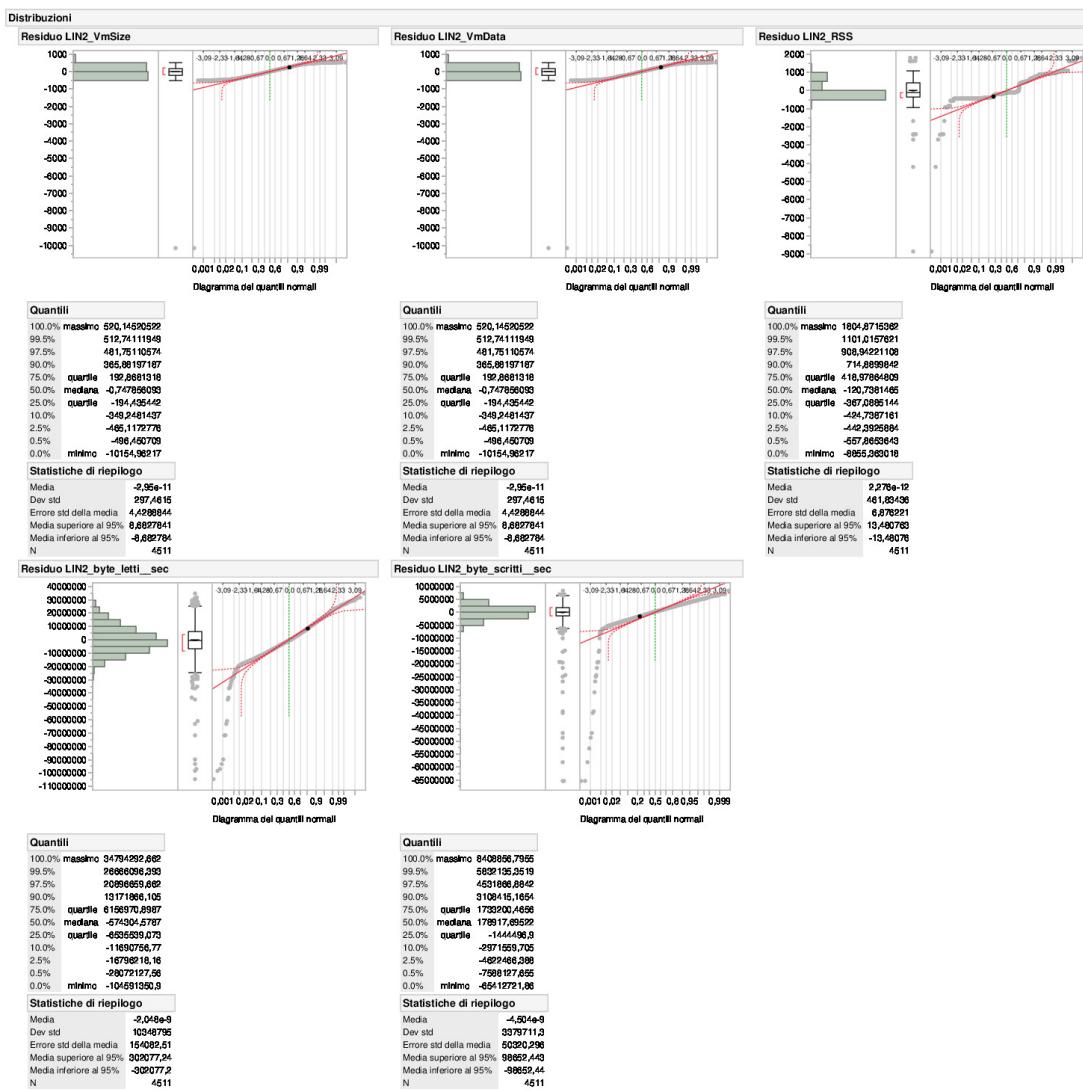


Figura 5.20: Valutazione Normalità

In tutti e 5 i casi le distribuzioni non sono normali, pertanto si omette la verifica dell'omoschedasticità. Si valuta dunque il test di Mann-Kendall (test non parametrico).

5.6.3 Test di Mann-Kendall

Si ripete lo stesso procedimento fatto nel primo paragrafo ottenendo i seguenti risultati:

Multivariato						
Non parametrico: τ di Kendall						
Variable	Variable by	τ di Kendall	Prob> τ	-8	-6	-4
TIME	LIN2_VmData	0,7065	<0001*	0	2	4

Figura 5.21: Kendall - VmData



Figura 5.22: Kendall - VmSize



Figura 5.23: Kendall - Rss



Figura 5.24: Kendall - Byte-letti-sec



Figura 5.25: Kendall - Byte-scritti-sec

Si indicano i valori in tabella e la rispettiva scelta:

Test NON parametrico - Kendall - Dataset 2 - VM2			
	τ	p-value	H
Time - VmSize	0.7065	0.0001	1
Time - VmData	0.7065	0.0001	1
Time - RSS	0.1788	0.0001	1
Time - Byte.letti.sec	0.6259	0.0001	1
Time - Byte.scritti.sec	-0.021	0.0342	1

Tabella 5.7: Test NON parametrico - Kendall - Dataset 2 - VM

Dalla Tabella 5.7, 5 test su 5 indicano che hanno un trend significativo nei dati.

5.6.4 Pendenza e intercetta della retta di regressione

Si ripete lo stesso procedimento del paragrafo 1:

Regressione Lineare Robusta - Theil e Sen					
	Time(x) - VmSize(y)	Time - VmData	Time - RSS	Time - Byte.letti.sec	Time - Byte.scritti.sec
Intercetta	735790.3761488455	667414.3761488455	42384.85453267744	77733384.23784722	65693615.57629428
Pendenza	0.0019128745423298214	0.0019128745423298214	0.0006324666198172874	114.55350983796296	-0.48861262488646684
CI inferiore	0.0	0.0	0.0005555555555555556	112.73359327217125	-0.9390005359056806
CI superiore	0.002085369827305311	0.002085369827305311	0.0007445442875481386	116.37659764826176	-0.036319163292847505

Tabella 5.8: Intercetta e Pendenza di Theil e Sen - Dataset 2 - VM

Dalla Tabella, si può notare che gli intervalli di confidenza al 95% relativi alle pendenze delle rette di regressione non includono lo zero per byte_letti_sec e byte_scritti_sec e RSS, dunque, possiamo affermare che tali pendenze, relative al **modello della popolazione**, hanno una probabilità del 95% che siano significativamente diverse da zero. Mentre, per le pendenze delle rette di regressione relative alle variabili VmSize e VmData si nota che gli intervalli di confidenza al 95% contengono lo zero e per questo motivo si può affermare che esse hanno una probabilità del 95% che siano significativamente uguali a zero.

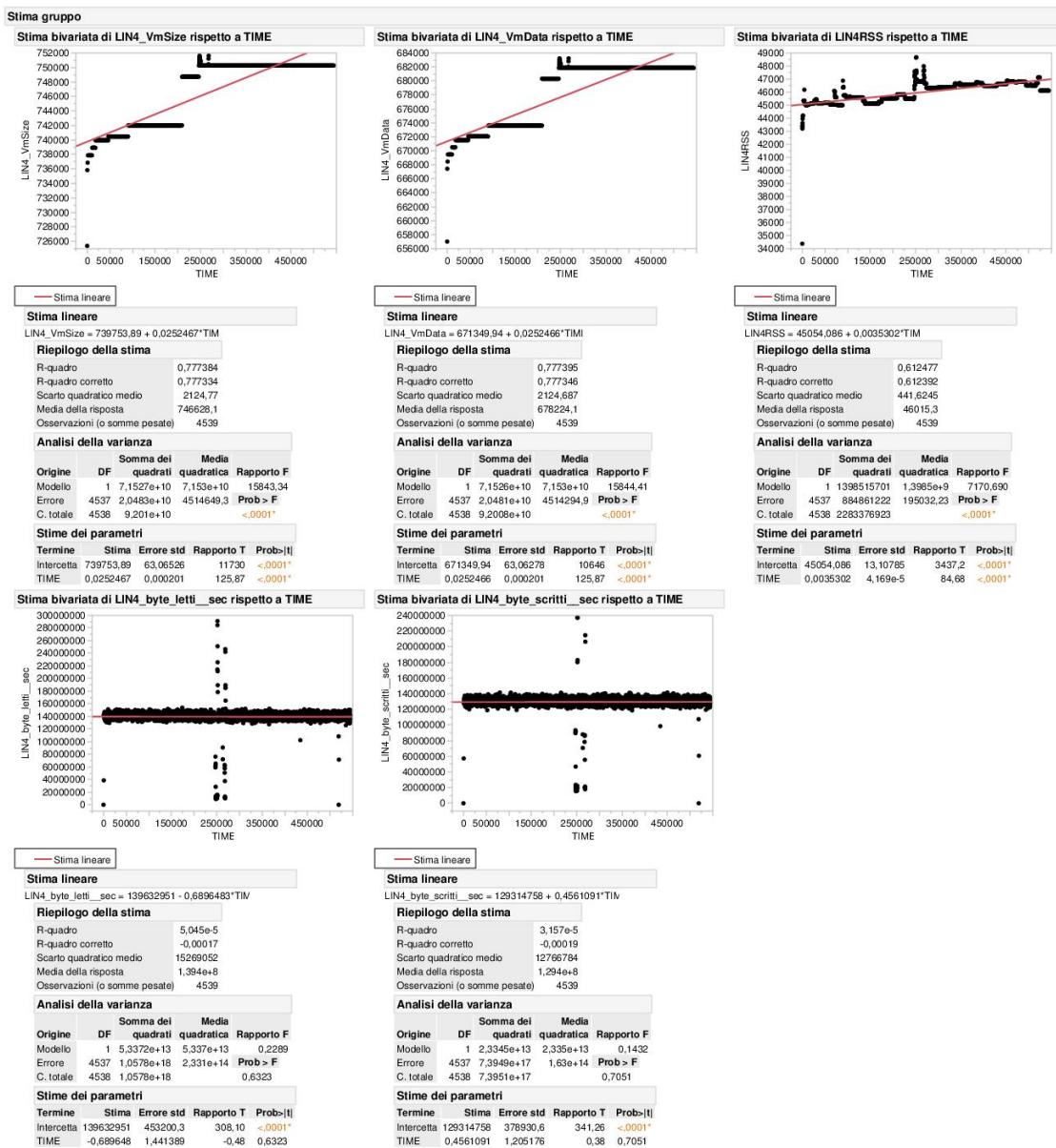
5.7 Dataset - OS3

Il dataset è così formato:

- Variabili di risposta:
 - VmSize
 - VmData
 - RSS
 - Byte_letti_sec
 - Byte_scritti_sec
- Variabili di predizione
 - Time

5.7.1 Valutazione delle rette di regressione e R^2

Viene applicato il modello regressivo lineare la cui risposta è una funzione lineare del preditore. Si valuta su JMP la stima lineare delle variabili, come mostrato in Figura 5.26.

Figura 5.26: Stima lineare e valutazione R^2

Nella stima viene indicato l'R-quadro. In questo caso R-quadro assume valore basso per le variabili **Byte_letti_sec** e **Byte_scritti_sec** e alto per le altre 3 variabili.

5.7.2 Test di Normalità

Per la scelta del tipo di test d'ipotesi da eseguire è stata valutata la distribuzione dei residui delle rispettive variabili. In particolare, si applica il test visivo tramite dei QQplot.

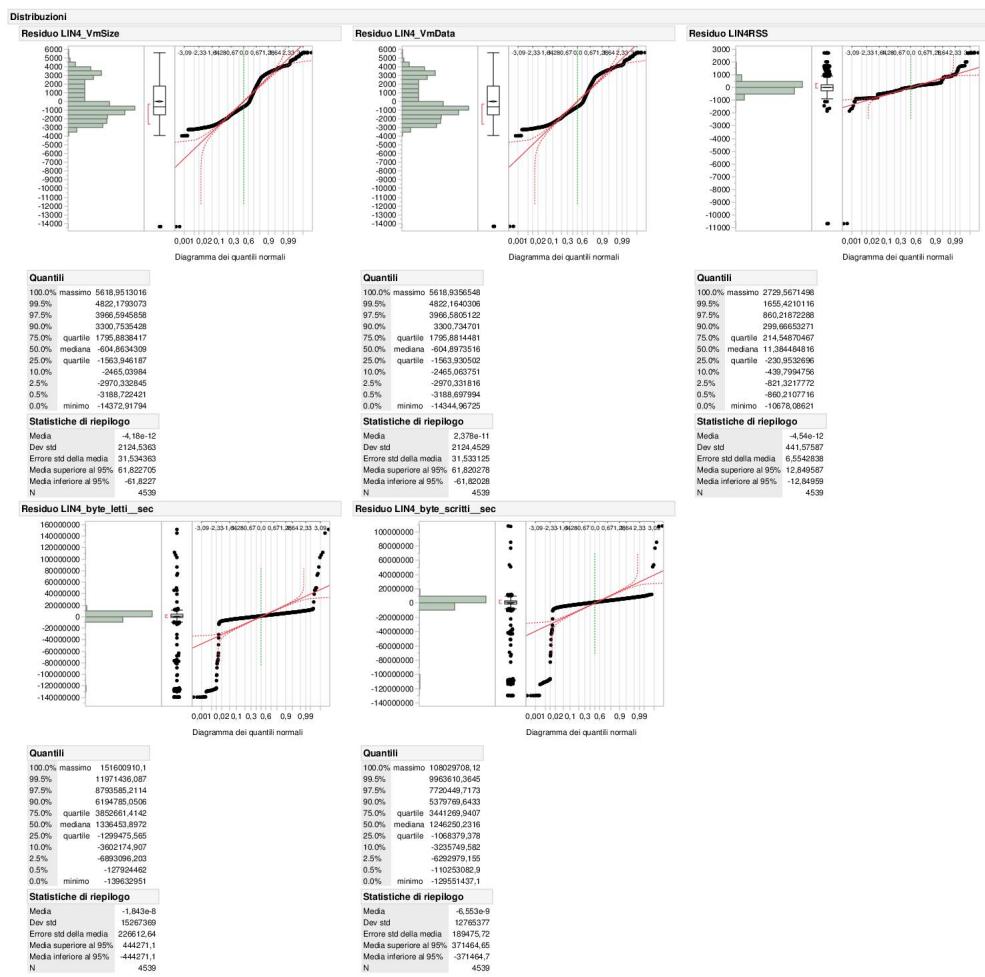


Figura 5.27: Valutazione Normalità

In tutti e 5 i casi le distribuzioni non sono normali, pertanto si omette di verificare l'omoschedasticità dei residui. Si valuta dunque il test di Mann-Kendall (test non parametrico).

5.7.3 Test di Mann-Kendall

Si ripete lo stesso procedimento fatto nel primo paragrafo ottenendo i seguenti risultati:

Multivariato						
Non parametrico: τ di Kendall						
Variable	Variable by	τ di Kendall	Prob> τ	<.8	>-.6	>-.4
TIME	LIN4_VmData	0,7832	<.0001*	0	.2	.4

Figura 5.28: Kendall - VmData

Multivariato						
Non parametrico: τ di Kendall						
Variable	Variable by	τ di Kendall	Prob> τ	<.8	>-.6	>-.4
TIME	LIN4_VmSize	0,7832	<.0001*	0	.2	.4

Figura 5.29: Kendall - VmSize



Figura 5.30: Kendall - Rss



Figura 5.31: Kendall - Byte-letti-sec



Figura 5.32: Kendall - Byte-scritti-sec

Si indicano i valori in tabella e la rispettiva scelta:

Test NON parametrico - Kendall - Dataset 3 - VM4			
	τ	p-value	H
Time - VmSize	0.7832	0.0001	1
Time - VmData	0.7832	0.0001	1
Time - RSS	0.5938	0.0001	1
Time - Byte.letti.sec	-0.0506	0.0001	1
Time - Byte.scritti.sec	-0.0171	0.0841	0

Tabella 5.9: Test NON parametrico - Kendall - Dataset 3 - VM

Dalla Tabella 5.9, si nota che 4 test su 5 affermano la presenza di trend significativi nei dati.

5.7.4 Pendenza e intercetta della retta di regressione

Si ripete lo stesso procedimento del paragrafo 1:

Regressione Lineare Robusta - Theil e Sen				
	Time(x) - VmSize(y)	Time - VmData	Time - RSS	Time - Byte.letti.sec
Intercetta	nan	nan	nan	nan
Pendenza	0.021659324522760644	0.021659324522760644	0.0034579439252336447	-1.891048815359477
CI inferiore	0.02127364220024293	0.02127364220024293	0.0033883947479881405	-2.618303689064559
CI superiore	0.0220266666666666667	0.0220266666666666667	0.0035223947783029484	-1.167004048582996

Tabella 5.10: Intercetta e Pendenza di Theil e Sen - Dataset 3 - VM

Dalla Tabella, si può notare che gli intervalli di confidenza al 95% relativi alle pendenze delle rette di regressione non includono lo zero per nessuna variabile, dunque, possiamo affermare che le pendenze, relative al **modello della popolazione**, hanno una probabilità del 95% che siano significativamente diverse da zero.

5.7.5 Confronto tra OS1-OS2-OS3

Dall'analisi del trend dei dati raccolti nei tre dataset relativamente all'occupazione di memoria di una macchina virtuale (VM), si è riscontrato come il dataset con un numero maggiore di trend nei dati sia il secondo in quanto tutte le variabili di risposta presentano un trend significativo. Avere un maggior trend nei dati può essere utile perché può indicare una tendenza o una direzione in cui i dati si stanno muovendo. Ciò può essere utile per prevedere i futuri cambiamenti e per prendere decisioni informate, nel caso di una VM, può essere utile per decidere di allocare più o meno memoria alla macchina stessa.

In particolare, tramite dei dati con una tendenza significativa per una macchina virtuale (VM), ci sono diverse metriche che si possono considerare:

- Utilizzo della memoria: un trend crescente nell'utilizzo della memoria (aumento di **VM_Size** e **VM_Data**) potrebbe indicare una maggiore attività della VM, e quindi la necessità di una maggiore quantità di memoria; quindi, si possono effettuare le valutazioni indifferentemente su uno dei tre dataset in quanto in tutti queste variabili hanno un trend;
- Utilizzo dello spazio di archiviazione: un trend crescente nell'utilizzo dello spazio di archiviazione (aumento di **VM_RSS**) potrebbe indicare una maggiore attività della VM, e quindi la necessità di una maggiore quantità di spazio di archiviazione; anche in questo caso le osservazioni al fine di prendere decisioni possono essere fatte su uno dei tre dataset essendo tutti significativi;
- Tasso di errore: un trend crescente nel tasso di errore (riduzione di **Byte_letti_sec** rispetto a **Byte_scritti_sec**) potrebbe indicare problemi con la VM, come ad esempio problemi di compatibilità con il sistema operativo o problemi di configurazione; in quest'ultimo caso per effettuare delle valutazioni per prevedere andamenti futuri, è necessario prendere in considerazione il secondo dataset in quanto le variabili di risposta interessate per questa metrifica presentano un trend significativo solo in quell'insieme di dati.

Queste osservazioni rappresentano solo alcune delle metriche che si possono considerare per valutare una VM, mirate a sfruttare i dati che sono messi a disposizione e le tendenze in essi presenti al fine di correggere eventuali situazioni critiche.

5.8 VmRes1 - Failure Prediction

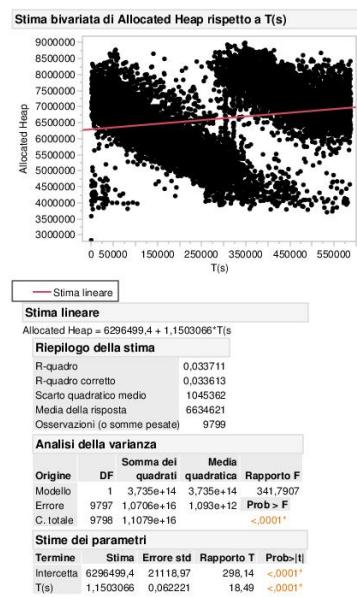
Il dataset è così formato:

- Variabili di risposta:
 - allocatedheap
- Variabili di predizione
 - Time

5.8.1 Valutazione della retta di regressione e R^2

Viene applicato il modello regressivo lineare la cui risposta è una funzione lineare del preditore. Si valuta su JMP la stima lineare delle variabili, come si evince dalla Figura 5.33.

Nella stima viene indicato l'R-quadro. **In questo caso R-quadro assume valore basso per la variabile AllocatedHeap.**

Figura 5.33: Stima lineare e valutazione R^2

5.8.2 Test di Normalità

Per la scelta del test da fare, per individuare delle tendenze significative nei dati, è stata valutata la distribuzione dei residui delle rispettive variabili.

In particolare, si applica il test visivo tramite dei QQplot.

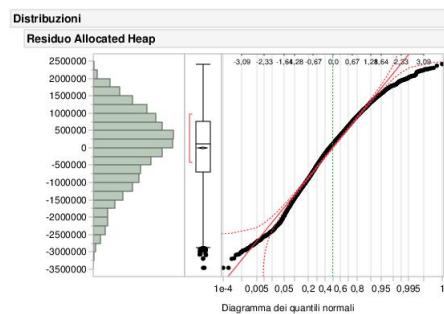


Figura 5.34: Valutazione Normalità

In questo caso la distribuzione non è normale, pertanto non si procede alla verifica dell'omoschedasticità.

Si valuta, dunque, il test di Mann-Kendall (test non parametrico).

5.8.3 Test di Mann-Kendall

Si ripete lo stesso procedimento fatto nel primo paragrafo ottenendo il seguente risultato:

Multivariato												
Non parametrico: τ di Kendall												
Variable	Variable by	τ di Kendall	Prob > τ	-8	-6	-4	-2	0	.2	.4	.6	.8
T(s)	Allocated Heap	0,0700	<.0001*									

Figura 5.35: Kendall - AllocatedHeap

Dalla Figura 5.35, si può notare il valore di p-value è molto basso e dunque, l'ipotesi H_0 è pari ad 1; ciò denota la presenza di un trend significativo nella rispettiva variabile.

5.8.4 Pendenza e intercetta della retta di regressione

Si ripete lo stesso procedimento del paragrafo 1:

Regressione Lineare Robusta - Theil e Sen - Predizione 1	
Ts (x) - Heap (y)	
Intercetta	6562275.461669506
Pendenza	0.6911224682945296
CI inferiore	0.563777264562484
CI superiore	0.8178240089963452

Tabella 5.11: Intercetta e Pendenza di Theil e Sen - Predizione 1

Dalla Tabella, si può notare che l'intervallo di confidenza al 95% relativo alla pendenza della retta di regressione non include lo zero, dunque, possiamo affermare che la pendenza, relativa al **modello della popolazione**, ha una probabilità del 95% che sia significativamente diversa da zero.

5.8.5 Valutazione Failure Prediction

La prediction della Failure è:

$$\text{allocatedheap} = m + bT(s) \quad (5.1)$$

Pertanto, dati i valori in Tabella 5.11, abbiamo che:

$$T(s)^2 = \frac{1GB - m}{b} = \frac{1GB - 6562275.46}{(0.69 \pm 0.12)} = 49 \pm 8 \text{ anni} \quad (5.2)$$

In cui, l'intervallo di confidenza al 95% è stato calcolato come segue:

$$CI = \frac{CI_{superiore} - CI_{inferiore}}{2} \simeq 0.12 \quad (5.3)$$

²1GB = 1,073,741,824 bytes

5.9 VmRes2 - Failure Prediction

Il dataset è così formato:

- Variabili di risposta:
 - allocatedheap
- Variabili di predizione
 - Time

5.9.1 Valutazione delle rette di regressione e R^2

Viene applicato il modello regressivo lineare la cui risposta è una funzione lineare del predittore. Si valuta su JMP la stima lineare delle variabili, come mostrato in Figura 5.36.

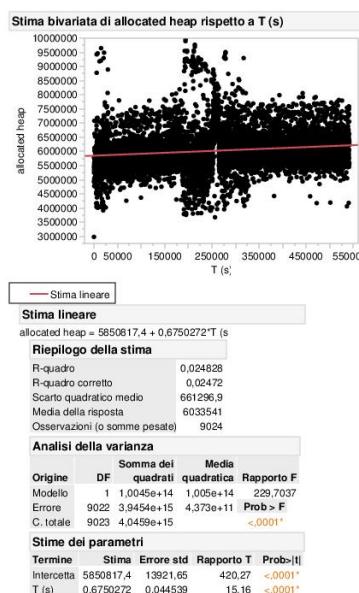


Figura 5.36: Stima lineare e valutazione R^2

Nella stima viene indicato l'R-quadro. **In questo caso R-quadro assume valore basso per la variabile AllocatedHeap.**

5.9.2 Test di Normalità

Per la scelta del test (parametrico o non) da applicare è stata valutata la distribuzione dei residui delle rispettive variabili.

In particolare, si applica il test visivo tramite dei QQplot.

In questo caso la distribuzione non è normale, pertanto non si effettua la verifica dell'omoschedasticità. Si valuta, dunque, il test di Mann-Kendall (test non parametrico).

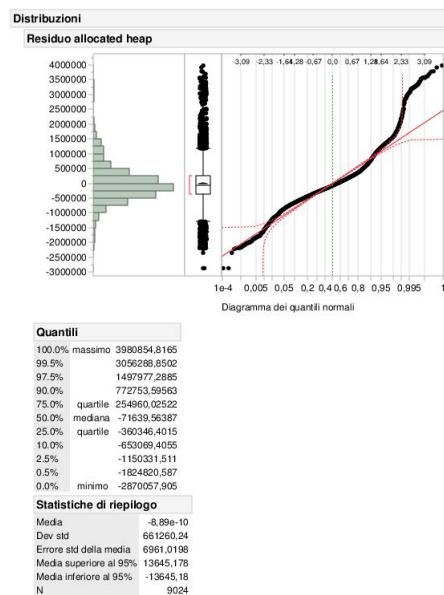


Figura 5.37: Valutazione Normalità

5.9.3 Test di Mann-Kendall

Si ripete lo stesso procedimento fatto nel primo paragrafo ottenendo il seguente risultato:

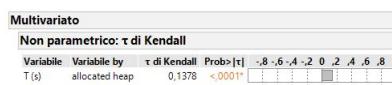


Figura 5.38: Kendall - AllocatedHeap

Dalla Figura 5.38, si può notare il valore di p-value è molto basso e dunque, l'ipotesi H è pari ad 1; ciò denota la presenza di un trend significativo nella rispettiva variabile.

5.9.4 Pendenza e intercetta della retta di regressione

Si ripete lo stesso procedimento del paragrafo 1:

Regressione Lineare Robusta - Theil e Sein - Predizione 2	
Ts (x) - Heap (y)	
Intercetta	5780180.563324536
Pendenza	0.6709499304572171
CI inferiore	0.6048638415610893
CI superiore	0.7366538131962297

Tabella 5.12: Intercetta e Pendenza di Theil e Sen - Predizione 2

Dalla Tabella, si può notare che l'intervallo di confidenza al 95% relativo alla pendenza della retta di regressione non include lo zero, dunque, possiamo affermare che la pendenza, relativa al **modello della popolazione**, ha una probabilità del 95% che sia significativamente diversa da zero.

5.9.5 Valutazione Failure Prediction

La prediction della Failure è:

$$\text{allocatedheap} = m + bT(s) \quad (5.4)$$

Pertanto, dati i valori in Tabella 5.12, abbiamo che:

$$T(s) = \frac{1GB - m}{b} = \frac{1GB - 5780180.56}{(0.67 \pm 0.06)} = 50 \pm 5 \text{ anni} \quad (5.5)$$

In cui, l'intervallo di confidenza al 95% è stato calcolato come segue:

$$CI = \frac{CI_{\text{superiore}} - CI_{\text{inferiore}}}{2} \simeq 0.06 \quad (5.6)$$

5.10 VmRes3 - Failure Prediction

Il dataset è così formato:

- Variabili di risposta:
 - allocatedheap
- Variabili di predizione
 - Time

5.10.1 Valutazione delle rette di regressione e R^2

Viene applicato il modello regressivo lineare la cui risposta è una funzione lineare del preditore. Si valuta su JMP la stima lineare delle variabili, come mostrato in Figura 5.39.

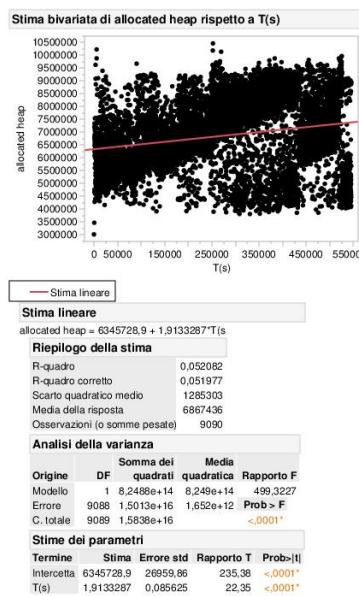


Figura 5.39: Stima lineare e valutazione R^2

Nella stima viene indicato l'R-quadro. **In questo caso R-quadro assume valore basso per la variabile AllocatedHeap.**

5.10.2 Test di Normalità

Per la scelta del test da eseguire è stata valutata la distribuzione dei residui delle rispettive variabili.

In particolare, si applica il test visivo tramite dei QQplot.

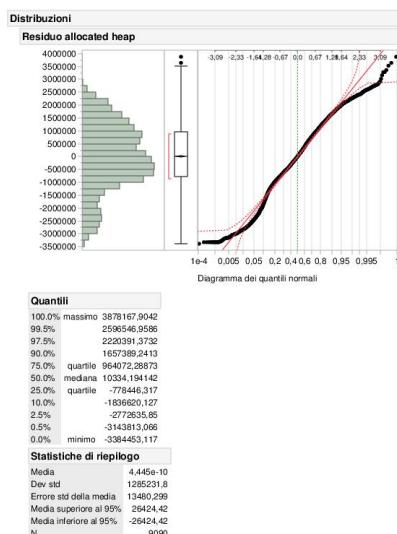


Figura 5.40: Valutazione Normalità

In questo caso la distribuzione non è normale, pertanto si omette la verifica dell'omoschedasticità. Si valuta, dunque, il test di Mann-Kendall (test non parametrico).

5.10.3 Test di Mann-Kendall

Si ripete lo stesso procedimento fatto nel primo paragrafo ottenendo il seguente risultato:

Multivariato											
Non parametrico: τ di Kendall											
Variable	Variabile by	τ di Kendall	Prob> τ	>8	>6	>4	>2	<2	<4	<6	<8
T(s)	allocated heap	0,2031	<.0001*								

Figura 5.41: Kendall - AllocatedHeap

Dalla Figura 5.41, si può notare il valore di p-value è molto basso e dunque, l'ipotesi H è pari ad 1; ciò denota la presenza di un trend significativo nella rispettiva variabile.

5.10.4 Pendenza e intercetta della retta di regressione

Si ripete lo stesso procedimento del paragrafo 1:

Regressione Lineare Robusta - Theil e Sen - Predizione 3	
Ts (x) - Heap (y)	
Intercepta	6041585.556431387
Pendenza	2.9030932760062105
CI inferiore	2.714759535655058
CI superiore	3.0923529411764705

Tabella 5.13: Intercetta e Pendenza di Theil e Sen - Predizione 3

Dalla Tabella, si può notare che l'intervallo di confidenza al 95% relativo alla pendenza della retta di regressione non include lo zero, dunque, possiamo affermare che la pendenza, relativa al **modello della popolazione**, ha una probabilità del 95% che sia significativamente diversa da zero.

5.10.5 Valutazione Failure Prediction

La prediction della Failure è:

$$\text{allocatedheap} = m + bT(s) \quad (5.7)$$

Pertanto, dati i valori in Tabella 5.13, abbiamo che:

$$T(s) = \frac{1GB - m}{b} = \frac{1GB - 6041585.56}{(2.9 \pm 0.19)} = 12 \pm 1 \text{ anni} \quad (5.8)$$

In cui, l'intervallo di confidenza al 95% è stato calcolato come segue:

$$CI = \frac{CI_{superiore} - CI_{inferiore}}{2} \simeq 0.19 \quad (5.9)$$

6 Reliability

Contenuti

6.1	Definizione degli obiettivi e Overview	92
6.2	RBD	93
6.2.1	Teorema dell'Upper Bound	93
6.2.2	Conditioning	94
6.2.3	MTTF del sistema	95
6.3	Confronto tra due sistemi	95
6.3.1	Equilibrare i due sistemi	96
6.4	Skip Ring	97
6.4.1	Valutazione della Reliability dopo un mission time	98
6.5	Confronto tra RBD	98
6.5.1	Confronto 1	99
6.5.2	Confronto 2	100
6.5.3	Confronto 3	101
6.5.4	Confronto 4	101
6.6	Reliability di un sistema di controllo di un elicottero	102
6.6.1	Effetto di Coverage	104

6.1 Definizione degli obiettivi e Overview

Si svolgono alcune analisi di Reliability di alcuni sistemi di esempio posti tramite l'utilizzo di tecniche model-based analitiche. In breve, la Reliability di un sistema misura la probabilità che il sistema funzioni senza problemi per un determinato periodo di tempo. La Reliability di un sistema può essere influenzata da molti fattori, tra cui la qualità dei componenti, la progettazione del sistema, la manutenzione e le condizioni ambientali. Si elencano alcune formule utilizzate durante l'analisi:

$$MTTF = \int_0^{+\infty} R_{sys}(t) dt \quad (6.1)$$

Se $R_{sys} = e^{-\lambda t}$ allora:

$$MTTF = \int_0^{+\infty} e^{-\lambda t} dt = \frac{1}{\lambda} \quad (6.2)$$

Se in un sistema i sottosistemi sono posti in:

- Serie:

$$R_{series}(t) = \prod_{i=1}^n R_i(t) \quad (6.3)$$

- Parallelo:

$$R_{parallel}(t) = 1 - \prod_{i=1}^n (1 - R_i(t)) \quad (6.4)$$

Se in un sistema non si riesce ad individuare la serie e il parallelo si applica il **Teorema dell'Upper Bound**:

$$R_{sys} \leq UPP.BOUND \quad (6.5)$$

dove:

$$UPP.BOUND = 1 - \prod_{i=1}^n (1 - R_{successpath_i}(t)) \quad (6.6)$$

Oppure il **Condizionamento** ove si sceglie il sottosistema su cui si vuole condizionare (disabilitazione o meno del sottosistema). Si usa il **Teorema della Probabilità Totale**:

$$P(A) = \sum_i P(A|B_i)P(B_i) \quad (6.7)$$

ovvero:

$$R_{sys} = R_m P(\text{system works}|\text{m works}) + (1 - R_m) P(\text{system works}|\text{m not works}) \quad (6.8)$$

6.2 RBD

Sia dato il seguente schema:

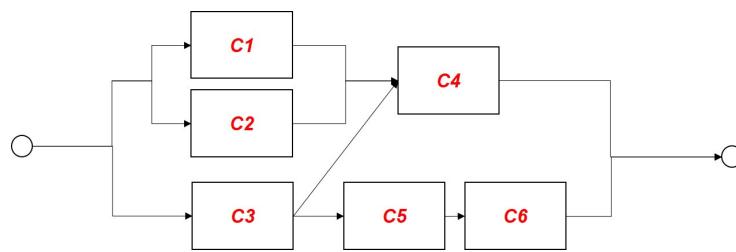


Figura 6.1: Schema Iniziale

Il sistema in figura è composto da sei sottosistemi dove:

$$R_1(t) = R_2(t) = R_3(t) = R_4(t) = R_5(t) = R_6(t) = R(t) = e^{-\lambda t} \quad (6.9)$$

Dalla Figura 6.1 si può notare tempestivamente come C1 e C2 siano in **parallelo** e C5 e C6 siano in **serie** dunque si può semplificare lo schema come segue:

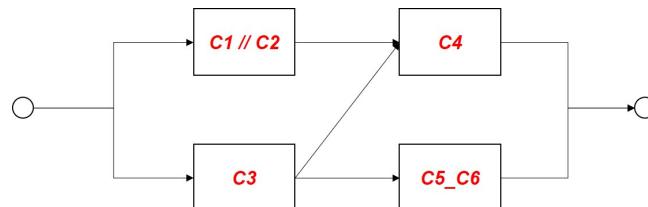


Figura 6.2: Schema con semplificazioni serie e paralleli

dove:

$$R_{56} = R \cdot R = R^2 \quad (6.10)$$

$$R_{12} = 1 - (1 - R)^2 \quad (6.11)$$

Dalla Figura 6.2 si nota un sistema in forma **non serie parallelo**, pertanto si utilizza il **Teorema dell'Upper Bound (1)** e successivamente la tecnica del **Conditioning (2)**.

6.2.1 Teorema dell'Upper Bound

Per applicare il teorema si necessita di enumerare i **success path** del sistema iniziale 6.1 e posizionare ognuno di essi in parallelo. Si applicano le formule 6.5 e 6.6. Si individuano pertanto i success path:

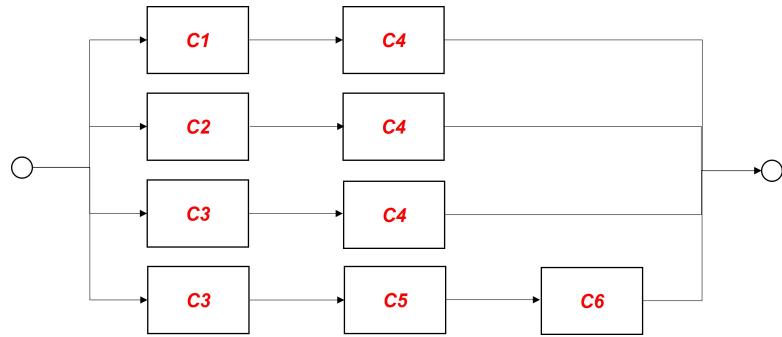


Figura 6.3: Schema dei Success Path

dove:

$$C1 - C4 = R^2$$

$$C2 - C4 = R^2$$

$$C3 - C4 = R^2$$

$$C3 - C4 - C5 = R^3$$

Pertanto la R_{sys} avrà come Upper Bound il seguente valore:

$$\begin{aligned} R_{sys} &\leq 1 - (1 - R^2)^3(1 - R^3) \\ &\leq 3R^2 + R^3 + R^9 - 3R^5 - 3R^7 \end{aligned}$$

6.2.2 Conditioning

Si procede a condizionare il sistema rispetto al **sottosistema C4** per calcolare la reliability del sistema garantendo di essere minore del upper bound valutato nel paragrafo precedente. Il condizionamento permette di analizzare il sistema complessivo in corrispondenza di funzionamento e fallimento del sottosistema selezionato. In particolare avrà le seguenti 2 condizioni:

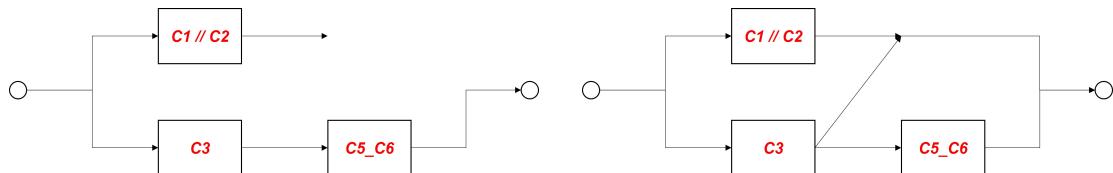


Figura 6.4: Fallimento di C4 (sinistra) e funzionamento di C4 (destra)

Si applica dunque la Probabilità Totale per valutare R_{sys} :

$$R_{sys} = R \cdot P(\text{system works} | \mathbf{C4 works}) + (1 - R) \cdot P(\text{system works} | \mathbf{C4 not works}) \quad (6.12)$$

Quando **C4 works** (Figura 6.4 (destra)) si ha che C1//C2 e C3 sono in parallelo, mentre quando **C4 not works** (Figura 6.4 (sinistra)) si ha che C3 e C5_C6 sono in serie. Pertanto si può scrivere:

$$\begin{aligned}
 R_{sys} &= R \cdot [1 - [1 - (1 - R)^2](1 - R)] + (1 - R) \cdot [R^3] \\
 &= R \cdot [1 - (1 - R)^3] + (1 - R) \cdot [R^3] \\
 &= 3R^2 - 2R^3
 \end{aligned}$$

Ma sapendo che $R = e^{-\lambda t}$ allora la **Reliability complessiva del sistema** è:

$$R_{sys} = 3e^{-2\lambda t} - 2e^{-3\lambda t} \quad (6.13)$$

ed è proprio minore dell' upper bound valutato nel paragrafo precedente.

6.2.3 MTTF del sistema

A valle di questi calcoli sarà possibile calcolare il MTTF come:

$$MTTF = \int_0^{+\infty} R_{sys}(t) dt \quad (6.14)$$

Se $R_{sys} = 3e^{-2\lambda t} - 2e^{-3\lambda t}$ allora:

$$MTTF = \int_0^{+\infty} 3e^{-2\lambda t} - 2e^{-3\lambda t} dt = \frac{3}{2\lambda} - \frac{2}{3\lambda} = \frac{5}{6\lambda} \quad (6.15)$$

6.3 Confronto tra due sistemi

Si confrontano le reliability di due sistemi composti dagli stessi sottosistemi, ma con una **topologia differente**.

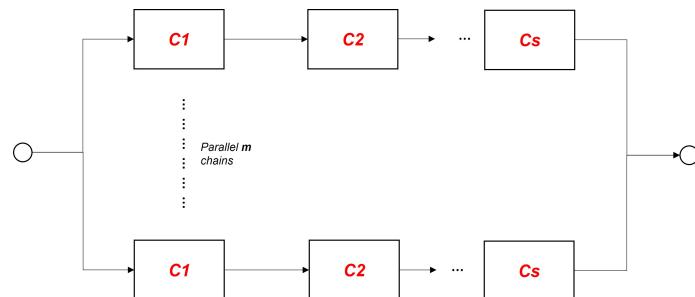


Figura 6.5: sys₁ Parallels of chain

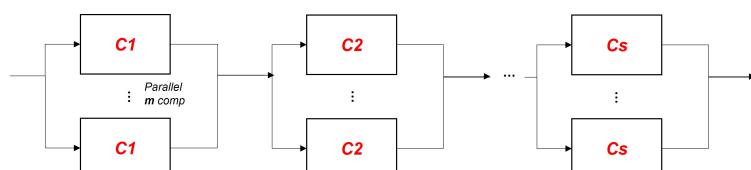


Figura 6.6: sys₂ Series of parallels

Da una prima analisi visiva è possibile definire che il sys₂ (series of parallels) è più reliable rispetto al sys₁ in quanto presenta più successpath. Si può notare infatti come il primo sistema 6.5 sia composto dal parallelo

di m catene composte dalla serie di s sottosistemi dalle stesse proprietà. Pertanto la Reliability di ciascuna catena può essere calcolata come:

$$R_{chain_i} = R^s \quad (6.16)$$

Dunque la R_{sys_1} sarà:

$$R_{sys_1} = 1 - (1 - R^s)^m \quad (6.17)$$

Il **secondo sistema** invece 6.6 è composto dalla serie di s paralleli di m sottosistemi pertanto:

$$R_{parallel_i} = 1 - (1 - R^m)^s \quad (6.18)$$

Dunque la R_{sys_2} sarà:

$$R_{sys_2} = (1 - (1 - R^m))^s \quad (6.19)$$

Nel caso in esame si ha $m = 5$ e $s = 3$ con $MTTF = 1400h$ ($\lambda = \frac{1}{1400h}$) con $R = e^{-\lambda t}$ allora:

$$R_{sys_1} = 1 - (1 - R^3)^5 \quad (6.20)$$

$$R_{sys_2} = (1 - (1 - R^5))^3 \quad (6.21)$$

Per il confronto dei sistemi si usa MATLAB:

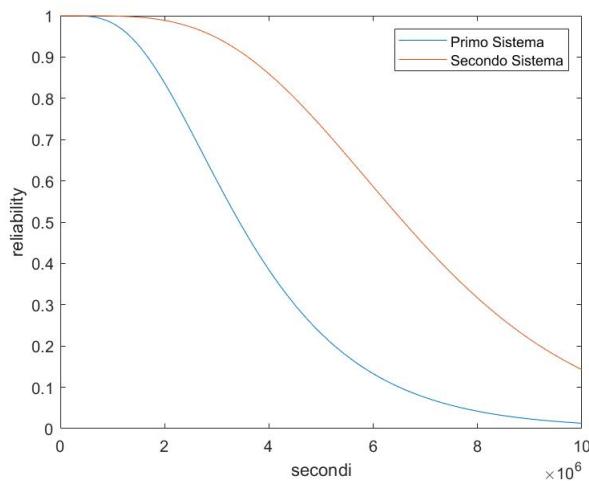


Figura 6.7: Confronto tra i 2 sistemi

In accordo con la deduzione fatta in precedenza, il plot determina che $R_{sys_2} \gg R_{sys_1}$.

6.3.1 Equilibrare i due sistemi

Per garantire, per uno specifico mission time (1400h), la stessa reliability tra ambedue i sistemi, **si aumenta la Reliability del primo sistema uguagliandola con quella del secondo sistema aumentando il numero di catene in parallelo** (aumentando quindi il valore di m). Calcolando quindi la $R_{sys_2}(t)$ con t (mission time) = 1400 h e $R_{sys_1}(t)$ con il medesimo t si avrà:

$$R_{sys_2}(1400) = 0,726754 \approx 73\% \quad (6.22)$$

$$R_{sys_1}(1400) = 0,225351 \approx 23\% \quad (6.23)$$

Allora per calcolare il nuovo valore:

$$\begin{aligned}
 R_{sys_1} &= 1 - (1 - R^3)^m = 0,73 \\
 &= (1 - R^3)^m = 1 - 0,73 = 0,27 \\
 &= (1 - e^{-3\lambda t})^m = 0,27 \\
 &= (1 - e^{-\frac{3}{1400} \cdot 1400})^m = 0,27 \\
 &= (1 - e^{-3})^m = 0,27 \\
 &= (0,95)^m = 0,27 \\
 &= m \log(0,95) = \log(0,27)
 \end{aligned}$$

Allora m sarà:

$$m = \frac{\log(0,27)}{\log(0,95)} = 25,526 \approx 26 \quad (6.24)$$

Sarà dunque necessario avere **26 catene nel primo sistema per garantire, per questo specifico mission time, la stessa reliability del secondo, tuttavia, la complessità spaziale nella realizzazione del sistema risulterà maggiore.**

6.4 Skip Ring

Si valuta la Reliability della seguente configurazione:

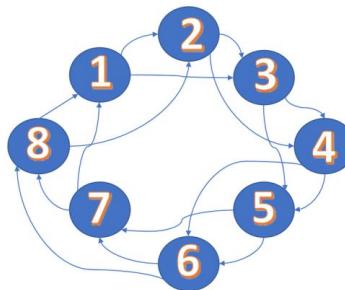


Figura 6.8: Skip Ring

Questa configurazione è costruita in modo tale da garantire che il fallimento di un nodo non limiti la possibilità degli altri nodi di comunicare, infatti, ogni nodo è connesso con i due nodi seguenti. Si prova a valutare le combinazioni per cui il sistema non funziona (il sistema va in fallimento) calcolando la Reliability del sistema con la seguente equazione (M-out-of-N-Systems) dove M è il numero di sottosistemi fuori uso e N è il numero totale di sottosistemi in una rete.

$$R_{MN} = \sum_{i=0}^{N-M} \binom{N}{i} R_m^{N-i} (1 - R_m)^i \quad (6.25)$$

Sapendo che R_m è la reliability di ogni sottosistema e $\lambda = 0,005$ il Failure Rate si valutano i casi:

- **i = 0:** Tutti i nodi funzionano ed il sistema funziona sempre. La Reliability sarà:

$$R_{i=0} = \binom{8}{0} R_m^8 = R_m^8 \quad (6.26)$$

- **i = 1:** fallisce un solo nodo ma la Skip Ring network continua a funzionare. La Reliability sarà:

$$R_{i=1} = R_m^8 + \binom{8}{1} R_m^7(1 - R_m) = R_m^8 + 8R_m^7(1 - R_m) \quad (6.27)$$

- **i = 2:** falliscono due nodi, il sistema non funziona se i nodi che falliscono sono adiacenti, quindi si hanno 8 combinazioni di fallimento. La Reliability sarà:

$$\begin{aligned} R_{i=2} &= R_m^8 + 8R_m^7(1 - R_m) + [\binom{8}{2} - 8]R_m^6(1 - R_m)^2 \\ &= R_m^8 + 8R_m^7(1 - R_m) + 20R_m^6(1 - R_m)^2 \end{aligned}$$

- **i = 3:** falliscono tre nodi, il sistema fallisce sia se i nodi sono tutti e 3 adiacenti (quindi 8 combinazioni di fallimento) sia se 2 nodi sono adiacenti e l'altro no. In questo ultimo caso abbiamo 4 nodi non adiacenti per ogni coppia e le coppie sono 8 (quindi 8x4 combinazioni). La Reliability sarà:

$$\begin{aligned} R_{i=3} &= R_m^8 + 8R_m^7(1 - R_m) + 20R_m^6(1 - R_m)^2 + [\binom{8}{3} - 32 - 8]R_m^5(1 - R_m)^3 \\ &= R_m^8 + 8R_m^7(1 - R_m) + 20R_m^6(1 - R_m)^2 + 16R_m^5(1 - R_m)^3 \end{aligned}$$

- **i = 4:** se falliscono 4 nodi si ha che solo il 50% dei nodi iniziali funzionano correttamente, il sistema in queste condizioni continua ad operare solo se i nodi funzionanti sono alternati nella rete, di queste combinazioni ce ne sono solo 2 (1-3-5-7 e 2-4-6-8). La Reliability sarà:

$$R_{i=4} = R_m^8 + 8R_m^7(1 - R_m) + 20R_m^6(1 - R_m)^2 + 16R_m^5(1 - R_m)^3 + 2R_m^4(1 - R_m)^4 \quad (6.28)$$

- **i ≥ 5:** se falliscono più di 5 nodi nel sistema non esistono condizioni in cui la rete lavorerebbe, si avrà sempre almeno una coppia di sottosistemi adiacenti in down facendo fallire il sistema.

Pertanto la Reliability del sistema sarà:

$$R_{sys}(t) = R_m^8 + 8R_m^7(1 - R_m) + 20R_m^6(1 - R_m)^2 + 16R_m^5(1 - R_m)^3 + 2R_m^4(1 - R_m)^4 \quad (6.29)$$

Dove $R_m = e^{-\lambda t} = e^{-0,005t}$.

6.4.1 Valutazione della Reliability dopo un mission time

Per la valutazione della Reliability della rete dopo 48h (mission time definito) si ha che:

$$R_{sys}(48) = 0,728880 \approx 73\% \quad (6.30)$$

6.5 Confronto tra RBD

Siano 4 coppie di sistemi composti da 3 sottosistemi (A,B,C) caratterizzati dai seguenti MTTF:

- $MTTF_A = 900h$
- $MTTF_B = 7000h$

- $MTTF_C = 1000h$

Anche in questo caso la $R = R_A = R_B = R_C = e^{-\lambda t}$.

6.5.1 Confronto 1

La prima coppia di sistemi da confrontare è:

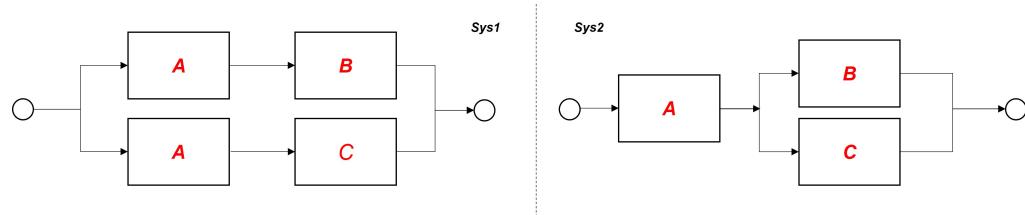


Figura 6.9: Confronto 1

6.5.1.1 Analisi Visiva

Ad impatto visivo si può intuire che il sistema più Reliable è il primo in quanto presenta la replicazione del sottosistema A, e dunque, aumentano i possibili success path. Se A fallisce, il secondo sistema fallisce mentre il primo non comporta automaticamente il fallimento del sistema, visto che il componente è replicato.

6.5.1.2 Analisi analitica e grafica

Si verifica quanto detto nell'analisi visiva attraverso la valutazione analitica e grafica:

$$R_{sys1} = 1 - (1 - R_A R_B)(1 - R_A R_C) \quad (6.31)$$

$$R_{sys2} = R_A \cdot (1 - (1 - R_B)(1 - R_C)) \quad (6.32)$$

Dal grafico seguente:

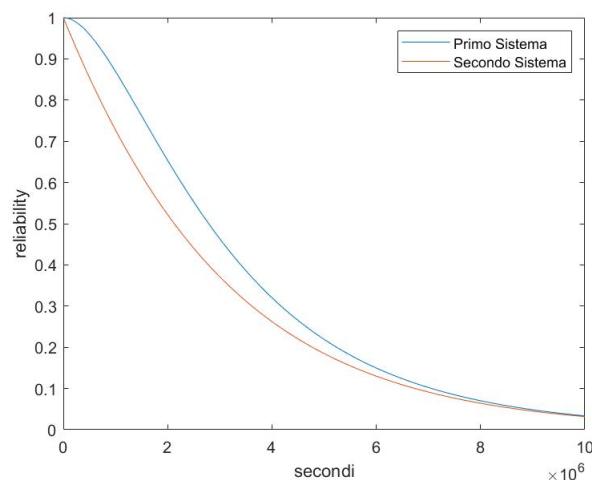


Figura 6.10: Confronto tra i 2 sistemi

In accordo con quanto detto in precedenza R_{sys_1} è meglio di R_{sys_2} .

6.5.2 Confronto 2

La coppia di sistemi da confrontare è:

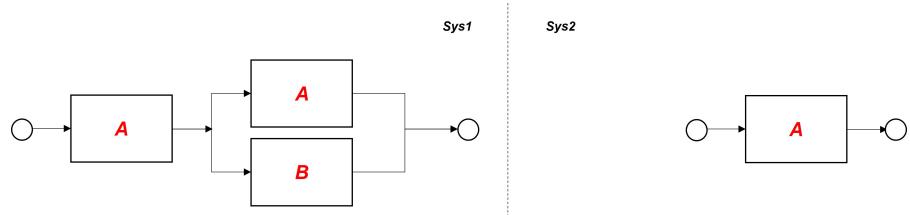


Figura 6.11: Confronto 2

6.5.2.1 Analisi Visiva

In questo caso, ad impatto visivo si può intuire che se A fallisce il sistema 2 fallisce. Mentre se B fallisce entrambi i sistemi continuano a funzionare, dunque, la reliability dei sistemi sarà molto simile. Tuttavia, si può notare che nel primo sistema è presente una serie in più, il che dalla teoria ci dice che la reliability di tale sistema è limitata superiormente dalla reliability dell'elemento meno affidabile della serie stessa.

6.5.2.2 Analisi analitica e grafica

Si verifica quanto detto nell'analisi visiva attraverso la valutazione analitica e grafica:

$$R_{sys1} = R_A \cdot (1 - (1 - R_A)(1 - R_B)) \quad (6.33)$$

$$R_{sys2} = R_A \quad (6.34)$$

Dal grafico seguente:

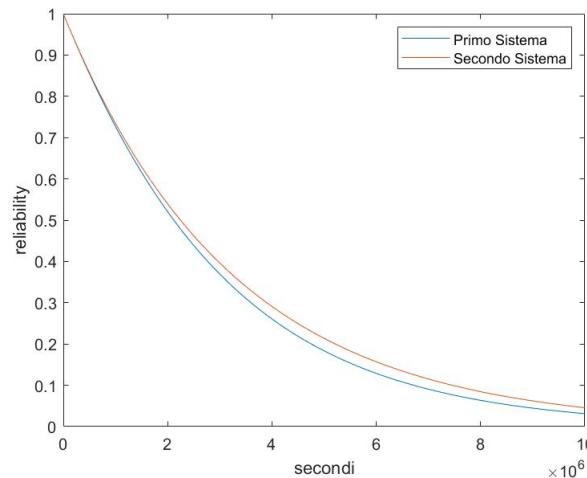


Figura 6.12: Confronto tra i 2 sistemi

In accordo con quanto detto in precedenza R_{sys_2} è meglio di R_{sys_1} . **Conferma anche del fatto che la reliability della serie di più elementi è infatti minore o uguale della reliability dell'elemento meno affidabile nella serie.**

6.5.3 Confronto 3

La coppia di sistemi da confrontare è:

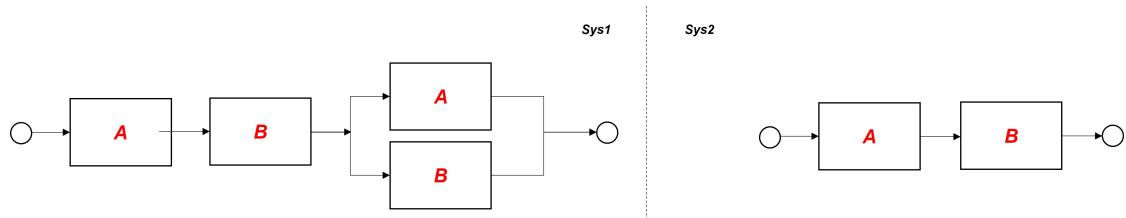


Figura 6.13: Confronto 3

6.5.3.1 Analisi Visiva

In questo caso, ad impatto visivo sono simili ai sistemi precedenti ma con un sottosistema in più, pertanto rimane lo stesso ragionamento di prima.

6.5.3.2 Analisi analitica e grafica

Si verifica quanto detto nell'analisi visiva attraverso la valutazione analitica e grafica:

$$R_{sys1} = R_A R_B \cdot (1 - (1 - R_A)(1 - R_B)) \quad (6.35)$$

$$R_{sys2} = R_A R_B \quad (6.36)$$

Dal grafico seguente:

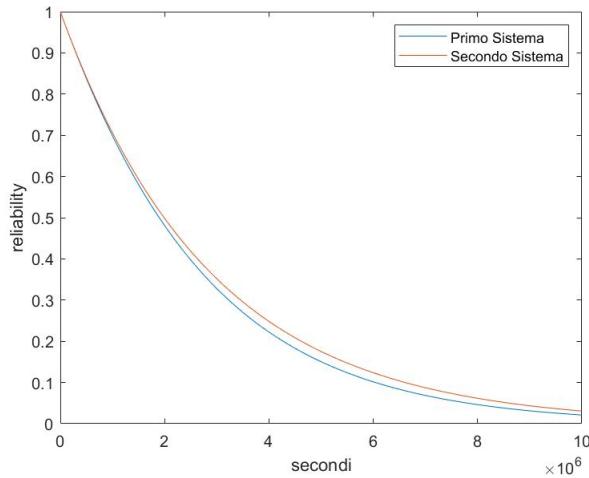


Figura 6.14: Confronto tra i 2 sistemi

In accordo con quanto detto in precedenza R_{sys2} è meglio di R_{sys1} .

6.5.4 Confronto 4

La coppia di sistemi da confrontare è:

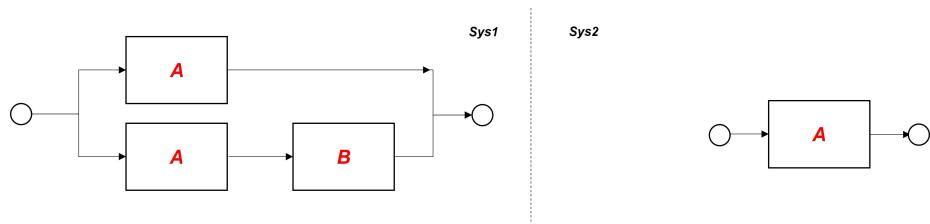


Figura 6.15: Confronto 4

6.5.4.1 Analisi Visiva

In questo caso, ad impatto visivo si può intuire che se A fallisce il sistema 2 fallisce. Mentre se B fallisce entrambi i sistemi continuano a funzionare. Il primo sistema è più reliable del secondo sistema.

6.5.4.2 Analisi analitica e grafica

Si verifica quanto detto nell'analisi visiva attraverso la valutazione analitica e grafica:

$$R_{sys1} = 1 - (1 - R_A)(1 - R_B R_A) \quad (6.37)$$

$$R_{sys2} = R_A \quad (6.38)$$

Dal grafico seguente:

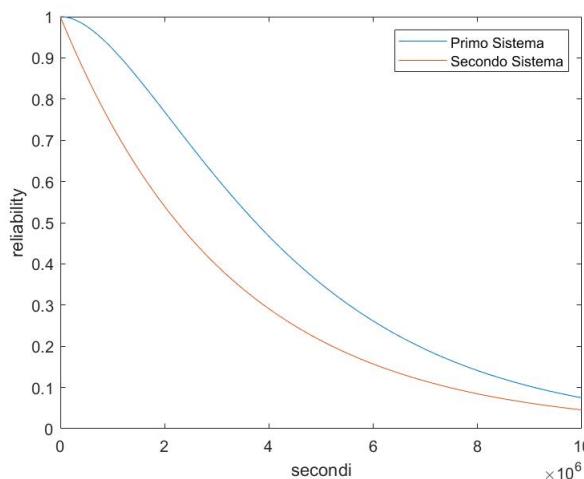


Figura 6.16: Confronto tra i 2 sistemi

In accordo con quanto detto in precedenza R_{sys_1} è meglio di R_{sys_2} .

6.6 Reliability di un sistema di controllo di un elicottero

Si modella e analizza, tramite RBD e FT, una centralina di controllo di un elicottero. In particolare, si caratterizza attraverso lo schema RBD un sistema seguendo le seguenti proprietà:

- Il sistema presenta due processori ridondanti, ed è in grado di funzionare finché uno dei due funziona

- Il sistema presenta due terminali di controllo ridondanti
- Entrambi i bus utilizzati, bus A e bus B, sono ridondanti
- Il sistema di navigazione è basato sull' Internal Navigation System, ma, per le emergenze, è sostituibile da un Doppler e da uno dei tre moduli AHRS presenti all'interno del sistema.

Il modello del sistema sarà dunque:

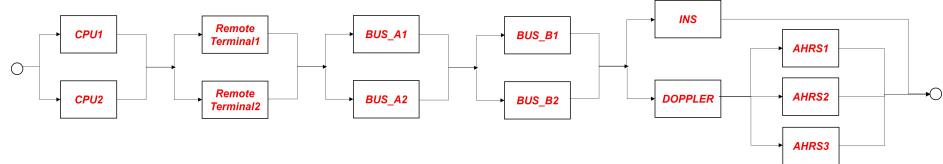


Figura 6.17: Modello RBD

A partire da questo si può calcolare la reliability complessiva del sistema:

$$\begin{aligned}
 R_{sys} &= (1 - (1 - R_{CPU1})(1 - R_{CPU2}))(1 - (1 - R_{RT1})(1 - R_{RT2}))(1 - (1 - R_{BUSA1})(1 - R_{BUSA2})) \\
 &\quad (1 - (1 - R_{BUSB1})(1 - R_{BUSB2}))(1 - (1 - R_{INS})(1 - R_D(1 - (1 - R_{AHRS})^3))) \\
 &= (1 - (1 - R_{CPU})^2)(1 - (1 - R_{RT})^2)(1 - (1 - R_{BUSA})^2)(1 - (1 - R_{BUSB})^2) \\
 &\quad (1 - (1 - R_{INS})(1 - R_D(1 - (1 - R_{AHRS})^3)))
 \end{aligned}$$

dove $MTTF_{CPU} = 10000$, $MTTF_{RT} = 4500$, $MTTF_{BUSA} = 60000$, $MTTF_{BUSB} = 60000$, $MTTF_{INS} = 2000$, $MTTF_{DOPPLER} = 500$, $MTTF_{AHRS} = 2000$.

$$R_{sys}(1h) \simeq 99,99\% \text{ (per la precisione } 0,999998941322750)$$

Si modella il sistema con l'utilizzo dei fault trees, ed individuare i minimal cutsets che provocano il failure del sistema. Dal modello RDB di sopra si realizza il Fault Tree model sostituendo tutte le porte parallele la AND e a tutte le serie delle porte la OR. Pertanto il Fault Tree risultante sarà:

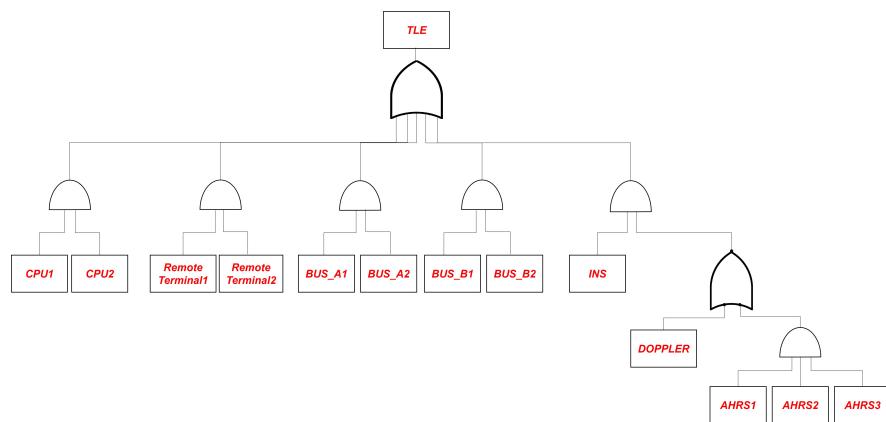


Figura 6.18: Fault Tree

Si calcolano i **minimal cutsets** il cui malfunzionamento causa il fallimento del sistema. Si indicano i seguenti minimal cutsets nella seguente Figura:

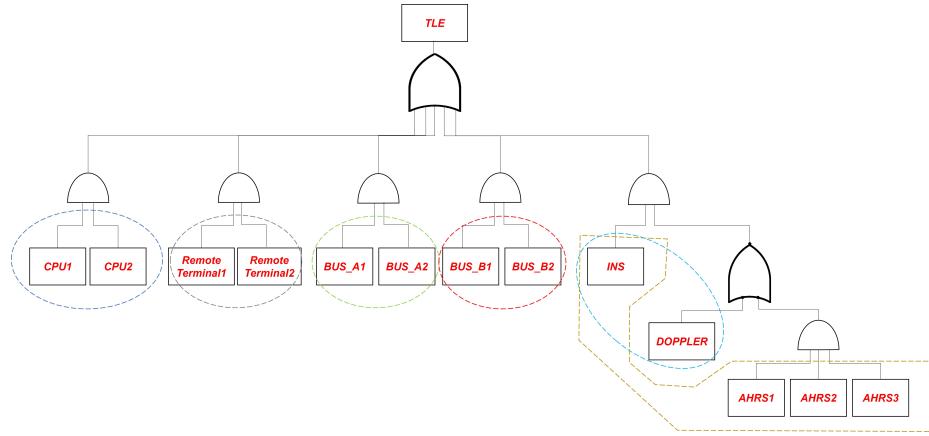


Figura 6.19: Fault Tree e Minimal Cuts

6.6.1 Effetto di Coverage

Si considera il fattore di coverage in serie ad uno dei due processori replicati. Pertanto la Reliability del sistema risultante sarà:

$$R_{sys} = (1 - (1 - R_{CPU})(1 - R_{CPU}C))(1 - (1 - R_{RT})^2)(1 - (1 - R_{BA})^2)(1 - (1 - R_{BB})^2) \\ (1 - (1 - R_{INS})(1 - R_D(1 - (1 - R_{AHRS})^3))) = 0,99999 \quad (6.39)$$

Si rinomina il seguente prodotto:

$$R_{sys,parz} = (1 - (1 - R_{RT})^2)(1 - (1 - R_{BA})^2)(1 - (1 - R_{BB})^2)(1 - (1 - R_{INS})(1 - R_D(1 - (1 - R_{AHRS})^3)))$$

Dunque, si riscrive l'equazione 6.39 per ricavare il fattore di coverage C come segue:

$$(1 - (1 - R_{CPU})(1 - R_{CPU}C))R_{sys,parz} = 0,99999 \\ (1 - (1 - R_{CPU})(1 - R_{CPU}C)) = \frac{0,99999}{R_{sys,parz}} \\ -(1 - R_{CPU}C)) = \frac{0,99999}{(1 - R_{CPU})R_{sys,parz}} - \frac{1}{(1 - R_{CPU})} \\ C = \frac{1}{R_{CPU}} + \frac{0,99999}{R_{CPU}(1 - R_{CPU})R_{sys,parz}} - \frac{1}{R_{CPU}(1 - R_{CPU})}$$

$$C = 0,910573265762650$$

L'effetto della coverage sul modello:

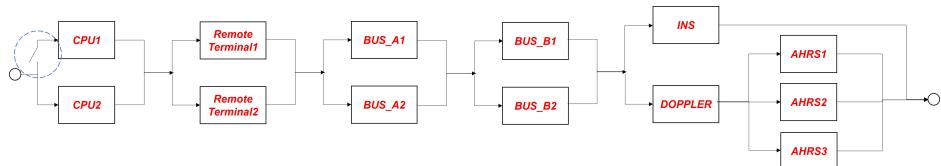


Figura 6.20: Effetto della Coverage

Contenuti

7.1	Definizione degli obiettivi e Overview	105
7.2	Analisi log del Calcolatore Mercury	106
7.2.1	Logging collection e Filtering	106
7.2.2	Data Manipulation	107
7.2.3	Data Analysis	109
7.3	Analisi log del Calcolatore Blue Gene/L	112
7.3.1	Logging collection e Filtering	112
7.3.2	Data Manipulation	113
7.3.3	Data Analysis	114
7.4	Confronto Reliability tra Mercury e Blue Gene	117
7.5	Analisi CWIN tra nodi (Mercury, BG/L) e categorie di errore (Mercury)	118
7.5.1	Coalescence Window per i nodi di Mercury	118
7.5.2	Coalescence Window per i nodi di Blue Gene	120
7.6	Confronto reliability di sistema vs. reliability dei nodi del calcolatore Mercury	121
7.7	Reliability bottlenecks	124
7.7.1	Bottlenecks di Mercury	124
7.7.2	Bottlenecks di BlueGene	124
7.8	Confronto reliability tra nodi simili (Mercury e Blue Gene)	125
7.9	Correlazione tra nodi e tipi di errore nel calcolatore Mercury	127

7.1 Definizione degli obiettivi e Overview

La **Field Failure Data Analysis** è una tecnica di misurazione diretta a misurare attributi di dependability di un sistema. Esso è un processo utilizzato per analizzare i dati sui fallimenti che si verificano durante il funzionamento di un sistema, al fine di identificare le cause alla base dei fallimenti e trovare soluzioni per prevenirli in futuro. Ciò può includere la raccolta di dati sui fallimenti, l'analisi statistica dei dati raccolti e la creazione di report che descrivono le cause alla base dei fallimenti e le azioni consigliate per prevenirli. L'analisi dei dati sui fallimenti in campo è un importante strumento per migliorare la qualità e l'affidabilità dei prodotti e dei sistemi. A partire da tale tecnica è possibile ottenere una serie di informazioni come:

- Stima del tempo medio di reliability del sistema (MTTF)
- Dati per la realizzazione del fitting delle curve di reliability empirica

A partire dai fallimenti e dagli errori del sistema, la FFDA è composta dalle seguenti fasi:

- **Logging Collection:** raccolta e memorizzazione dei dati relativi al sistema mentre è in funzione. Il Logger deve essere opportunamente configurato al fine di effettuare il monitoraggio ad hoc. Il file che si ottiene contiene dati grezzi;
- **Filtering:** si effettua un filtraggio dei dati raccolti per concentrare l'analisi solo sui dati significativi, tale processo è realizzato sulla base del log di interesse mediante approcci **White Listing o Black Listing**. Il file che si ottiene è filtrato e pronto ad essere manipolato;
- **Data Manipulation:** i dati sono aggregati per mezzo di un processo di **coalescenza**. Tale operazione permette di raggruppare failure relativi allo stesso guasto all'interno di uno stesso elemento detto **tupla** per ridurre in maniera ulteriore e notevole la dimensione del dataset iniziale;

- **Analysis:** i dati sono sottoposti ad analisi, per differenti scopi, che possono essere quello di trovare la curva di reliability del sistema, fino all'identificazione della distribuzione del Time to Failure.



Figura 7.1: Schema concettuale del procedimento di FFDA

7.2 Analisi log del Calcolatore Mercury

Il Calcolatore Mercury possiede un'architettura *three-layer* con l'aggiunta di un nodo di management chiamato (**tg-master**). Dal seguente schema è possibile notarlo:

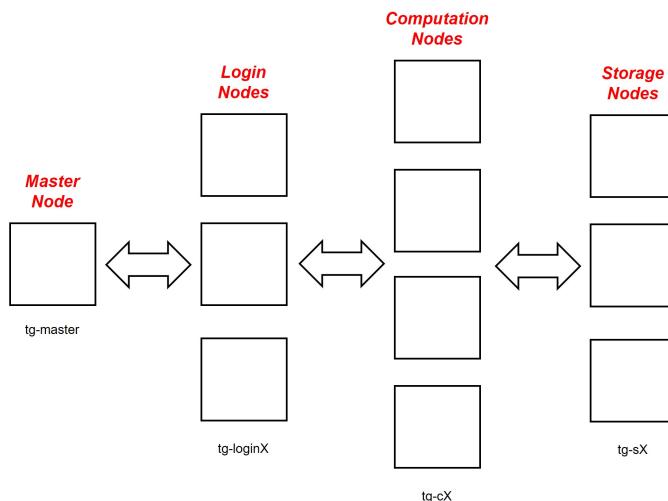


Figura 7.2: Schema del supercalcolatore Mercury

Oltre al nodo **Master** su cui sono in esecuzione i servizi di infrastruttura è possibile individuare:

- **tg-loginX:** nodi nel quale vengono eseguiti i servizi per garantire un accesso corretto alle risorse di sistema;
- **tg-cX:** nodi nel quale avvengono l'esecuzioni di processi cpu-intensive;
- **tg-sX:** nodi nel quale avviene la memorizzazione dei dati.

7.2.1 Logging collection e Filtering

I dati relativi all'analisi sono raccolti in un file di log con **80854** entry che riguardano **solo errori fatali del sistema**. I log sono stati già de-parametrizzati e formattate nel seguente modo:

- **timestamp**
- **nodo origine di failure**

- **categoria di failure**
- **messaggio di failure**

Gli esempi di error entries sono:

- **DEV**: Platform PCI Component Error Info Section
- **MEM**: Errori di Memoria come Physical Address, Address Mask, Node, Card, Module, Bank, Device, Row e Column
- **NET**: Connection down
- **I-O**: errori di input e output
- **PRO**: begin hardware error

Esempio di entry log:

```
1167667229 tg-c238 DEV Component Info: Vendor Id = ** , Device Id = **, Class Code = **, SegBus  
DevFunc= *****
```

7.2.2 Data Manipulation

Nella fase di manipulation si identificano, a partire dal log del sistema, i fallimenti del sistema. Per identificare i fallimenti di un sistema si utilizza la tecnica della **Coalescenza** in grado di rilevare la correlazione tra le entry del file di log. Essa ci permette di ridurre l'insieme delle informazioni raccolte durante la fase di logging collection. Tramite operazioni di aggregazione o di cancellazione, consente l'eliminazione di dati ridondanti o equivalenti tra di loro; ciò è necessario in quanto un guasto genera in un sistema più fallimenti e dal momento che gli effetti di un guasto si propagano in un sistema essi vengono rilevati più volte da componenti differenti del sistema. Per cui si vanno a raggruppare tutti i fallimenti relativi al medesimo guasto in un unico evento riducendone la lista dei log. In particolare si utilizza la tecnica della **coalescenza temporale** valutata sulla base della seguente condizione:

$$\text{IF } t(X_{i+1}) - t(X_i) < W \text{ THEN ADD } X_{i+1} \text{ to the tuple} \quad (7.1)$$

dove:

- X_i è la i-esima entry nel log;
- $t(X_i)$ è il timestamp della X_i entry;
- W è la Coalescence Window (CWIN).

Dalla 7.1 allora due o più entry sono raggruppate in una tupla ed associata al singolo failure se la distanza temporale tra le due entry è minore di un certo W . Pertanto serve scegliere la W tale che:

- se si considera una finestra di coalescenza troppo grande si incorre in **collisioni**¹ tra le entry di una stessa tupla.
- se si considera una finestra di coalescenza troppo piccola allora si incorre in **troncamimenti**²

Si cerca dunque di preferire una Coalescence Window piccola al fine di effettuare una stima pessimistica della curva di reliability. **Si accettano quindi più volentieri troncamimenti tra le tuple piuttosto che collisioni.**

¹Una collisione si ha quando **failure** differenti sono contenute in una stessa tupla

²I troncamimenti sono **eventi** relativi allo stesso failure che sono inseriti in tuple differenti

7.2.2.1 Sensitivity Analysis

Per la scelta del valore di W si effettua un'analisi di sensitività applicando l'algoritmo di Tupling 7.1. Si plottano i risultati in un grafico³ (caratteristica **Dimensioni della finestra di coalescenza - numero di tuple**). Si sceglie come valore ottimo della finestra di coalescenza, il valore di W immediatamente successivo al valore corrispondente al ginocchio della curva. Nel caso dell'analisi del sistema Mercury, tale fase ha prodotto il seguente andamento:

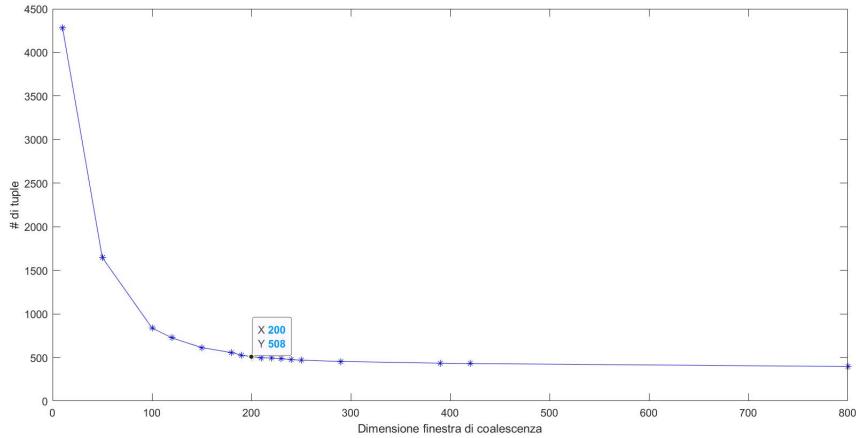


Figura 7.3: Analisi di sensitività - Mercury

Si è scelto un **CWIN pari a 200s** (≈ 3 minuti e 20 secondi) che presenta un **numero di tuple pari a 508**.

A valle della scelta del W migliore, si esegue uno script Python (UStupling.with_CWIN.py) grazie al quale si ottiene un file diverso per ogni gruppo di entry ottenendo ulteriori informazioni aggiuntive come:

- **length**: differenza tra i timestamp della prima e dell'ultima entry della tupla, **indica la durata della tupla** (in secondi);

$$\text{length}_i = \text{timestamp}_{N,i} - \text{timestamp}_{1,i} \quad (7.2)$$

dove N è l'ultima entry della tupla ed i è la tupla i-esima.

- **starting points**: timestamp della prima entry della tupla con annessa durata della stessa;
- **interarrival**: tempo (in secondi) che intercorre tra la prima entry della tupla corrente e l'ultima entry della tupla precedente (TTF). Grazie ad essi sarà possibile realizzare uno studio sulla reliability del sistema.

$$\text{interarrivo}_s = \text{timestamp}_{1,i} - \text{timestamp}_{N,i-1} \quad (7.3)$$

dove N è l'ultima entry della tupla ed i è la tupla i-esima e s il numero dell'interarrivo s-esimo (con $2 \leq i \leq s$).

Per **confermare la bontà del tupling** eseguito sul sistema Mercury, è utile valutare il numero di possibili collisioni e troncamenti ottenuti a valle di tale raggruppamento con la CWIN scelta, tramite uno script Python⁴. In particolare, si è ottenuto il seguente valore di troncamenti:

Troncamenti : 234

Percentuale di troncamenti: 46.06299212598425%

³ Per il plot si è utilizzato uno script MATLAB (SensitivityTuple.m) rilasciato durante il corso

⁴ Analisi_Troncamenti_Collisioni.ipynb

Dall'analisi del numero di troncamenti si può notare un valore elevato, ma da un'osservazione più approfondita si nota che i troncamenti stessi, **riguardano solo alcuni nodi chiave come c242, c401 e s176** ad esempio, che **concorrono anche alla maggior parte delle entry nel file di log, il che permette di accettare il gran numero di troncamenti ottenuti dato che circoscritti a pochi nodi.** Per quanto riguarda, invece, il numero di collisioni si ha la seguente situazione:

Collisioni : 47
Percentuale di collisioni: 9.251968503937007%

Come è possibile dedurre dalla scarsa percentuale di collisioni, **il valore scelto per la finestra di coalescenza è da ritenersi buona.** Tuttavia avere un **numero di troncamenti maggiore implicherà ad avere una reliability più bassa.**

7.2.3 Data Analysis

In questa fase si effettua uno studio della reliability del sistema a partire dai risultati ottenuti in precedenza. Grazie al calcolo degli interarrivi sarà possibile estrarre dai dati la distribuzione di probabilità empirica del time to failure (TTF) del sistema (Unreliability) utilizzando uno script MATLAB. Sarà, dunque, facile calcolare la Reliability del sistema come $R = 1 - TTF$.

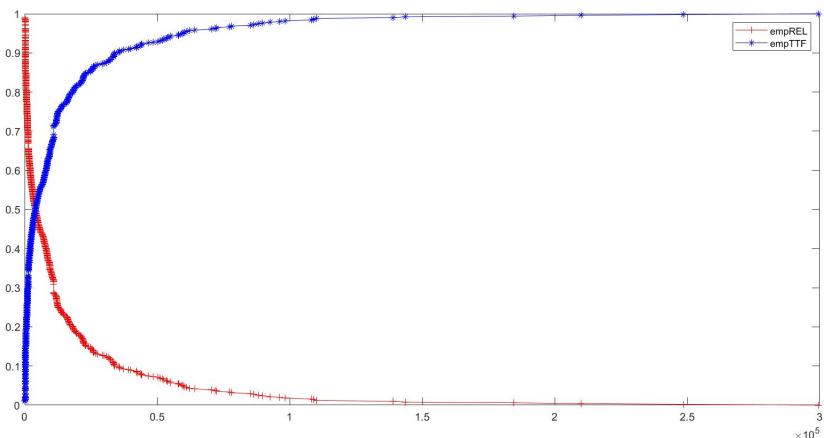


Figura 7.4: Confronto tra Reliability empirica e Unreliability empirica - Mercury

7.2.3.1 Fitting della reliability per lo studio della natura del fallimento

Il passo successivo è quello di cercare una distribuzione teorica nota che sia la più rappresentativa possibile della Reliability empirica valutata nella fase precedente. Sulla base di **quale distribuzione caratterizza meglio la Reliability empirica si può dedurre il tipo di fallimento presente nei log ricavando così informazioni sulla tipologia di failure che si verificano nel sistema.** Si effettuerà infatti il confronto della distribuzione della "empRef" con tre distribuzioni notevoli attraverso il **tool MATLAB: "cftool"**. Le tre distribuzioni notevoli sono:

- **Esponenziale:** ci da informazioni relativi a guasti di natura isolata (**memoryless**), tipicamente di moduli hardware indipendenti.

$$f(x, a, b) = \begin{cases} ae^{bx} & : x \geq 0 \\ 0 & : x < 0 \end{cases} \quad (7.4)$$

Si effettua il fitting della funzioni di reliability con una curva esponenziale:

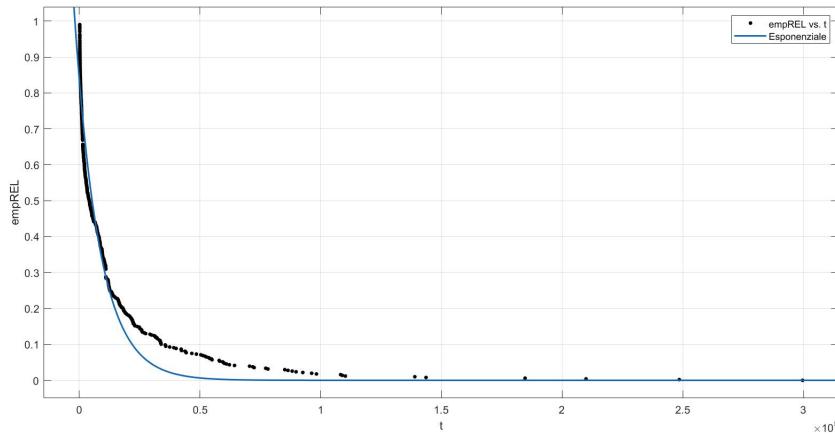


Figura 7.5: Confronto tra Reliability empirica e Distribuzione Esponenziale - Mercury

- **Iperesponenziale:** ci da informazioni di fallimenti causati da più moduli hardware distinti ed indipendenti.

$$f(x, a, b, c, d) = \begin{cases} ae^{bx} + ce^{dx} & : x \geq 0 \\ 0 & : x < 0 \end{cases} \quad (7.5)$$

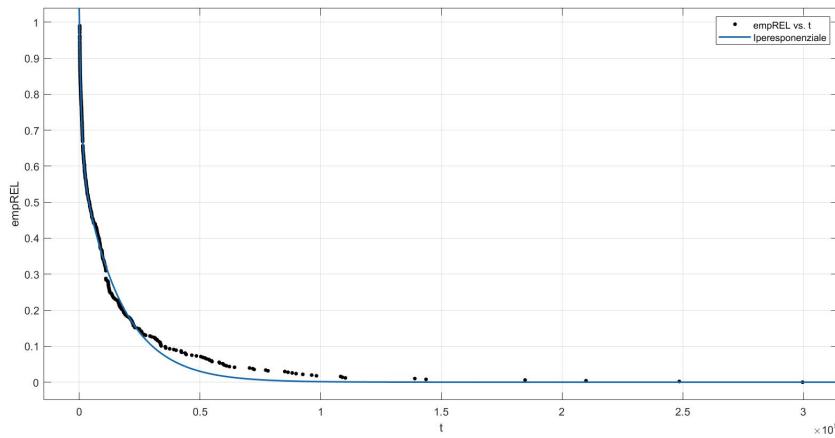


Figura 7.6: Confronto tra Reliability empirica e Distribuzione IperEsponenziale - Mercury

- **Weibull:** ci da informazioni relative tipicamente ad errori di accumulazione e failure di sistemi che sono in esecuzione per molto tempo causando degradazione e/o usura dei componenti. Essa è una distribuzione con **memoria**.

$$f(x, a, b) = \begin{cases} e^{-bx^a} & : x \geq 0 \\ 0 & : x < 0 \end{cases} \quad (7.6)$$

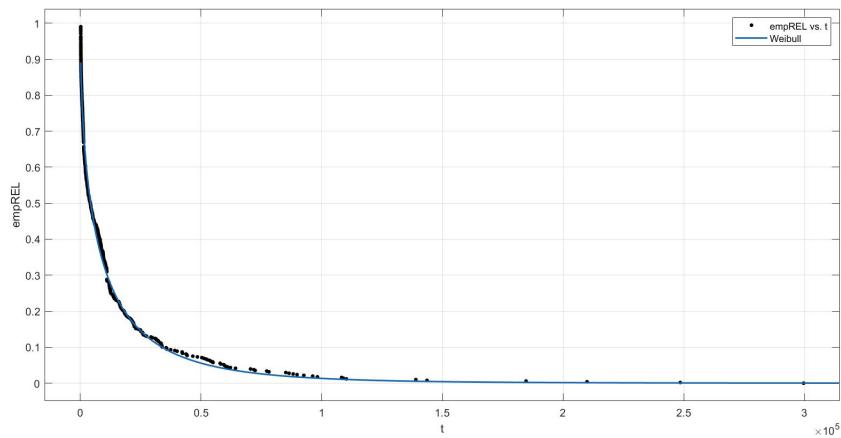


Figura 7.7: Confronto tra Reliability empirica e Distribuzione IperEsponenziale - Mercury

7.2.3.2 Goodness dei Fitting

Per determinare la bontà dei fitting valutati nel paragrafo precedente si calcola il valore dell' R^2 di ogni distribuzione e verificare la corretta approssimazione dei modelli attraverso il test non parametrico di Kolmogorov-Smirnov. In particolare, l'**ipotesi nulla** è:

$$H_0 = \text{le distribuzioni provengono dalla stessa distribuzione.}$$

Pertanto, si effettua tale test tramide il comando matlab seguente:

```
[h,p,k] = ktest2(empRel,Xfittedmodel(t))
```

dove:

- H: Uguale a 1 se rigetta l'ipotesi nulla al 5% di livello di significatività, 0 altrimenti. Di conseguenza se è 0, con un livello di confidenza del 95% le distribuzioni provengono dalla stessa distribuzione;
- P: Quanto più alto è il valore del p-value, maggiore è la probabilità che l'ipotesi nulla **non** venga rigettata;
- K: Valore del test così definito $K = [\max(\text{EmpRel}, \text{fittedRel})]$.

Si riportano, dunque, i risultati:

Distribuzione teorica migliore della Reliability empirica (Mercury)				
Distribuzione	p-value	k	R^2	H
Esponenziale	3.3449e-05	0.1524	0.9534	1
Ipersponenziale	0.6666	0.0472	0.9961	0
Weibull	0.0724	0.0837	0.9918	0

Tabella 7.1: Risultati ottenuti

Dalla Tabella si può notare che per quanto riguarda:

- R^2 : sono tutti abbastanza alti, indicano infatti una buona approssimazione, soprattutto per la distribuzione Iperesponenziale e di Weibull.

- Il Test di Kolmogorov-Smirnov rigetta l'ipotesi della distribuzione dell'esponenziale, non per l'iperesponenziale e weibull che pertanto possono essere considerati buoni modelli per il fitting. Tra questi ultimi due, però, l'iperesponenziale mostra un p-value superiore a quello della Weibull e di conseguenza viene eletto come **modello migliore la distribuzione iperesponenziale dato che rigetta l'ipotesi con una probabilità maggiore.**

In definitiva: avendo evidenziato un tipo di curva empirica simile alla distribuzione iperesponenziale, è facile intuire che probabilmente la reliability del sistema Mercury **dipende principalmente da fallimenti che sono riconducibili all'hardware**. Tale teoria viene confermata dal fatto che la maggioranza delle entry presenti nel file di log è appartenente alla categoria DEV, che rappresentano errori legati a schede PCI e alle periferiche in generale, ed a quella MEM (memoria).

7.3 Analisi log del Calcolatore Blue Gene/L

Il Calcolatore Blue Gene possiede un'architettura composta da: Rack(RX), Nodi(NX), Compute Card(JX), I/O Card(J18) e Compute Chip(U01,U11). Lo schema del calcolatore è il seguente:

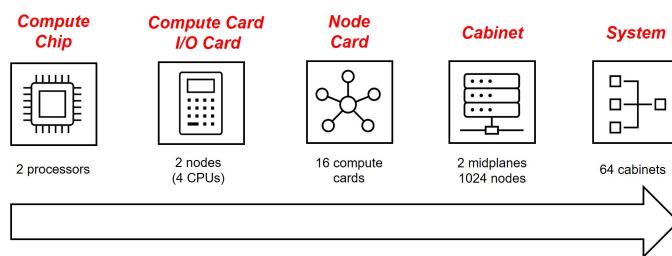


Figura 7.8: Schema del calcolatore Blue Gene/L

Esso è un calcolatore a parallelismo massivo e di dimensioni maggiori del sistema Mercury.

7.3.1 Logging collection e Filtering

I dati relativi all'analisi sono raccolti in un file di log con **125624** entry che riguardano solo errori fatali del sistema. I log sono stati già filtrati e formattate nel seguente modo:

- timestamp**
- nodo origine di failure**
- card di origine di failure**
- messaggio di failure**

Gli esempi di error entries sono:

- R72-M1-NCJ09-U11 **Parity Error**
- R35-M0-N3J12-U01 **Machine Check Interrupt**
- R46-M1-N8J04-U11 **Error receiving packet on tree network**
- ... altri errori

Esempio di entry log:

1128621351 R04-M0-N0 J18-U01 Lustremount FAILED : bglio66 : block-id: location

7.3.2 Data Manipulation

Si applica lo stesso procedimento di manipulation anche per il calcolatore Blue Gene.

7.3.2.1 Sensitivity Analysis

Anche in questo caso si plottano i risultati in un grafico⁵ (caratteristica **Dimensioni della finestra di coalescenza - numero di tuple**). Si sceglie come valore ottimo della finestra di coalescenza, il valore di W immediatamente successivo al valore corrispondente al ginocchio della curva. Nel caso dell'analisi del sistema Mercury, tale fase ha prodotto il seguente andamento.

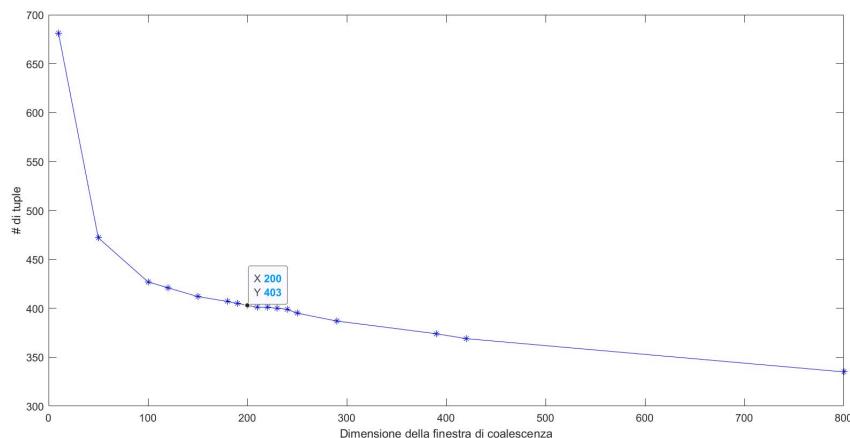


Figura 7.9: Analisi di sensitività - BlueGene

Si è scelto un **CWIN pari ai 200s** (≈ 3 minuti e 20 secondi) che presenta un **numero di tuple pari a 403**.

A valle della scelta del W migliore, si esegue uno script Python (UStupling_with_CWIN.py) grazie al quale si ottiene un file diverso per ogni gruppo di entry ottenendo ulteriori informazioni aggiuntive come:

- **length**: differenza tra i timestamp della prima e dell'ultima entry della tupla, indica la durata della tupla (in secondi);

$$\text{length}_i = \text{timestamp}_{N,i} - \text{timestamp}_{1,i} \quad (7.7)$$

dove N è l'ultima entry della tupla ed i è la tupla i-esima.

- **starting points**: timestamp della prima entry della tupla con annessa durata della stessa;
- **interarrival**: tempo (in secondi) che intercorre tra la prima entry della tupla corrente e l'ultima entry della tupla precedente (TTF). Grazie ad essi sarà possibile realizzare uno studio sulla reliability del sistema.

$$\text{interarrivo}_s = \text{timestamp}_{1,i} - \text{timestamp}_{N,i-1} \quad (7.8)$$

dove N è l'ultima entry della tupla ed i è la tupla i-esima e s il numero dell'interarrivo s-esimo (con $2 \leq i \leq s$).

⁵Per il plot si è utilizzato uno script MATLAB (SensitivityTuple.m) rilasciato durante il corso

Per confermare la bontà del tupling eseguito sul sistema Blue Gene/L, è utile valutare il numero di possibili collisioni e troncamenti ottenuti a valle di tale raggruppamento con la CWIN scelta, tramite lo stesso script Python usato in precedenza. In particolare, si è ottenuto il seguente valore di troncamenti:

Troncamenti : 21
Percentuale di troncamenti: 5.2109181141439205%

Dall'analisi del numero di troncamenti si può notare un valore molto basso, il quale indica che sia stata effettuata una buona scelta della finestra di coalescenza per il sistema in esame, da un ulteriore analisi il nodo con più troncamenti è R36-M1-N5. Per quanto riguarda, invece, il numero di collisioni si ha la seguente situazione:

Collisioni : 267
 Percentuale di collisioni: 66.25310173697271%

Come si può **notare** dal numero di collisioni elevato, si è deciso di effettuare un'analisi più approfondita per ogni nodo al fine di individuare dei comportamenti caratteristici nelle collisioni. Si è scelto, dunque, di considerare una collisione solo se all'interno della stessa tupla sono inseriti fallimenti relativi a nodi di I-O e a nodi computazionali di nodi diversi. Quindi, riconteggiando le collisioni si è ottenuto il seguente valore:

Collisioni : 39
Percentuale di collisioni : 9.67741935483871%

Il nuovo numero di collisioni ottenuto è sicuramente più basso, il che permette di affermare che **la scelta del valore di finestra di coalescenza effettuata per il sistema BG/L è buona.**

7.3.3 Data Analysis

Si fa la stessa Data Analysis anche per il calcolatore Blue Gene. Si valuta la Reliability del sistema:

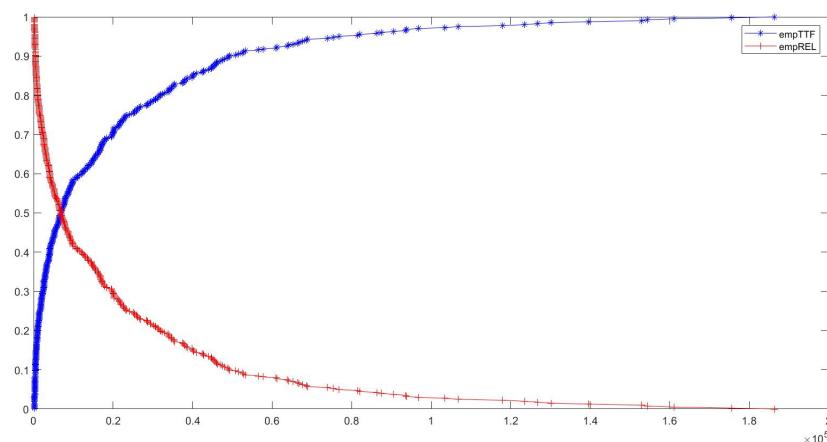


Figura 7.10: Confronto tra Reliability empirica e Unreliability empirica - Blue Gene

7.3.3.1 Fitting della reliability per lo studio della natura del fallimento

Anche in questo caso si effettuerà il confronto della distribuzione della "empRel" con tre distribuzioni notevoli attraverso il **tool MATLAB: "cftool"**. Le tre distribuzioni notevoli sono:

- **Esponenziale:** ci da informazioni relativi a guasti di natura isolata (**memoryless**), tipicamente di moduli hardware indipendenti.

$$f(x, a, b) = \begin{cases} ae^{bx} & : x \geq 0 \\ 0 & : x < 0 \end{cases} \quad (7.9)$$

Si effettua il fitting della funzioni di reliability con una curva esponenziale:

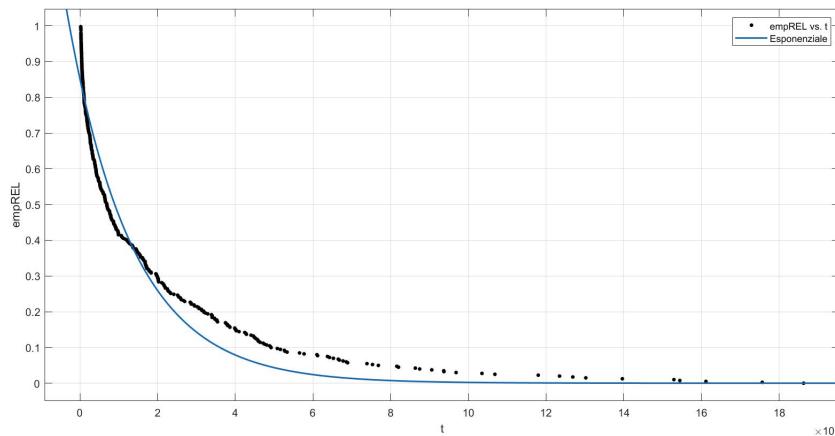


Figura 7.11: Confronto tra Reliability empirica e Distribuzione Esponenziale - Blue Gene

- **Iperesponenziale:** ci da informazioni di fallimenti causati da più moduli hardware distinti ed indipendenti.

$$f(x, a, b, c, d) = \begin{cases} ae^{bx} + ce^{dx} & : x \geq 0 \\ 0 & : x < 0 \end{cases} \quad (7.10)$$

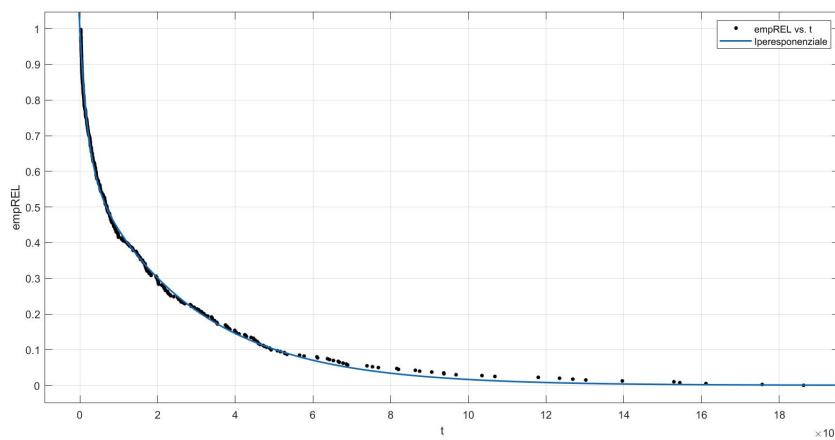


Figura 7.12: Confronto tra Reliability empirica e Distribuzione IperEsponenziale - Blue Gene

- **Weibull:** ci da informazioni relative tipicamente ad errori di accumulazione e failure di sistemi che sono in esecuzione per molto tempo causando degradazione e/o usura dei componenti. Essa è una distribuzione

con memoria.

$$f(x, a, b) = \begin{cases} e^{-bx^a} & : x \geq 0 \\ 0 & : x < 0 \end{cases} \quad (7.11)$$

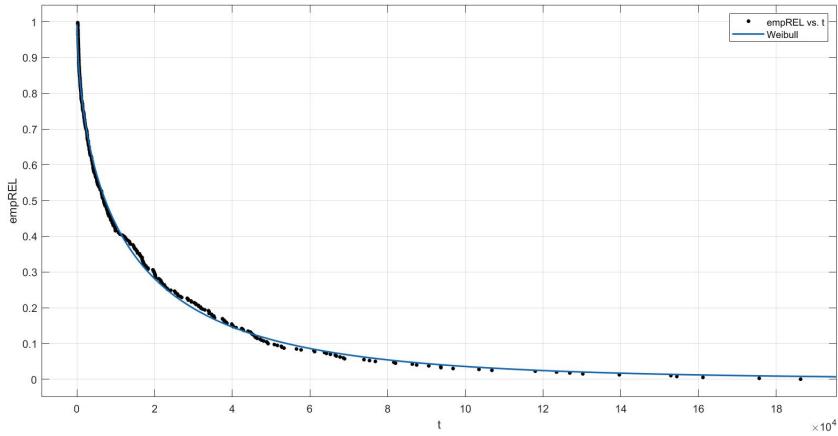


Figura 7.13: Confronto tra Reliability empirica e Distribuzione IperEsponenziale - Blue Gene

7.3.3.2 Goodness dei Fitting

Per determinare la bontà dei fitting valutati nel paragrafo precedente si calcola il valore dell' R^2 di ogni distribuzione e verificare la corretta approssimazione dei modelli attraverso il test non parametrico di **Kolmogorov-Smirnov**. In particolare, l'ipotesi nulla è:

$$H_0 = \text{le distribuzioni provengono dalla stessa distribuzione.}$$

Pertanto, si effettua tale test tramide il comando matlab seguente:

```
[h,p,k] = ktest2(empRel,Xfittedmodel(t))
```

dove:

- H: Uguale a 1 se rigetta l'ipotesi nulla al 5% di livello di significatività, 0 altrimenti. Di conseguenza se è 0, con un livello di confidenza del 95% le distribuzioni provengono dalla stessa distribuzione;
- P: Quanto più alto è il valore del p-value, maggiore è la probabilità che l'ipotesi nulla **non** venga rigettata
- K: valore del test così definito $K = [\max(\text{EmpRel}, \text{fittedRel})]$

Si riportano dunque i risultati:

Dalla Tabella si può notare che per quanto riguarda:

- R^2 : sono tutti abbastanza alti, indicano infatti una buona approssimazione, soprattutto per la distribuzione Iperesponenziale e di Weibull.
- Il Test di Kolmogorov-Smirnov rigetta l'ipotesi della distribuzione esponenziale, non per l'iperesponenziale e weibull che pertanto possono essere considerati buoni modelli per il fitting. Tra questi ultimi due, però, l'iperesponenziale mostra un p-value superiore a quello della Weibull e di conseguenza, anche in questo caso, viene eletto come **modello migliore la distribuzione iperesponenziale**.

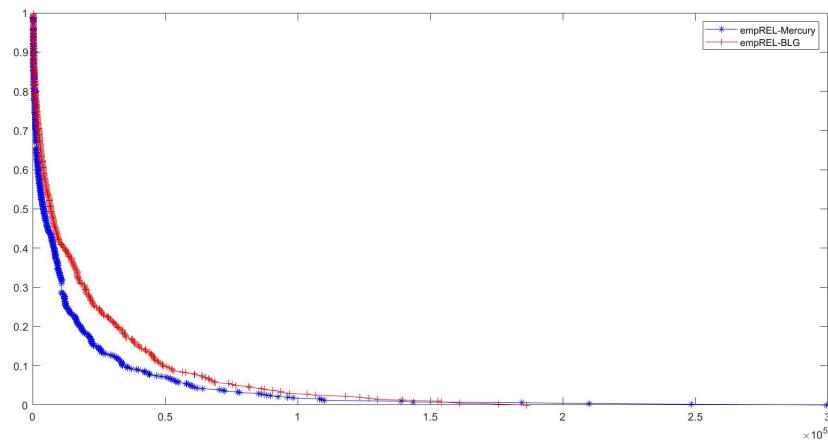
Distribuzione teorica migliore della Reliability empirica (BlueGene)				
Distribuzione	p-value	k	R ²	H
Esponenziale	3.3507e-04	0.1472	0.9563	1
Ipersponenziale	0.8954	0.0406	0.9978	0
Weibull	0.2631	0.0711	0.9956	0

Tabella 7.2: Risultati ottenuti - Blue Gene

In **definitiva**: anche in questo caso, avendo evidenziato un tipo di curva empirica simile alla distribuzione iper-esponenziale, è facile intuire che probabilmente la reliability del sistema Blue Gene (come Mercury) dipende principalmente da **fallimenti che sono riconducibili all'hardware**.

7.4 Confronto Reliability tra Mercury e Blue Gene

Dalle funzioni di reliability di Mercury e di Blue Gene è possibile confrontare l'affidabilità dei due sistemi.

**Figura 7.14:** Confronto tra Reliability di Mercury e Blue Gene

Dalla Figura 7.14 si nota come il sistema Blue Gene risulta più affidabile di Mercury. Si è inoltre confrontato il MTTF dei due sistemi utilizzando il calcolo dell'**integrale**:

$$\int_0^{+\infty} R_{Mercury}(t) dt = 1.2677 \cdot 10^4 s \quad (7.12)$$

dove $R_{Mercury}$ è la reliability empirica del calcolatore Mercury.

$$\int_0^{+\infty} R_{BlueGene}(t) dt = 2.0312 \cdot 10^4 s \quad (7.13)$$

dove $R_{BlueGene}$ è la reliability empirica del calcolatore Blue Gene.

Viene, dunque, confermato anche numericamente che **il calcolatore Blue Gene risulta più affidabile di quello Mercury**. Si noti che il calcolo dell'integrale è stato fatto su MATLAB.

7.5 Analisi CWIN tra nodi (Mercury, BG/L) e categorie di errore (Mercury)

Per rispondere alla prima domanda **"È possibile utilizzare le stesse finestre di coalescenza nodi (Mercury, BG/L) e le categorie di errore (Mercury)"** si utilizza lo script Python rilasciato durante il corso (USlogStatistics.py). Grazie ad esso è possibile individuare le statistiche relative ai nodi più fallimentari (10 nodi più fallimentari Mercury-BlueGene) e agli errori più frequenti. A valle di questa statistica, è possibile filtrare dal log principale le entries relative ad un unico nodo o ad un'unica categoria di errore attraverso lo script USfilter.py.

Al fine di individuare le finestre di coalescenza per i nodi e per le categorie di errore, si ripete la stessa procedura applicata nella fase di data manipulation in modo tale da capire se la stessa CWIN può essere applicata ai singoli nodi dei calcolatori e alle categorie di errore generalizzando quella trovata per il sistema complessivo.

7.5.1 Coalescence Window per i nodi di Mercury

Si è scelto di effettuare l'analisi di sensitività dei nodi a cui sono associati un maggior numero di entry all'interno del log di Mercury. Di seguito sono mostrate le sensitivity analysis effettuate:

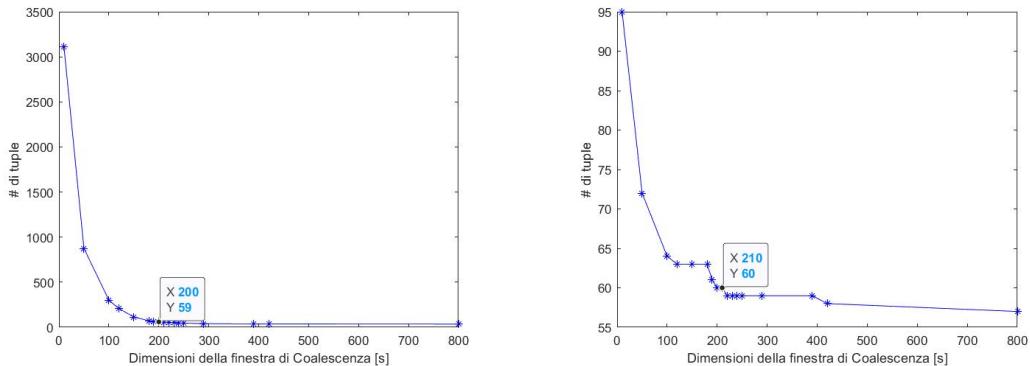


Figura 7.15: Nodi: tg-c401 (sinistra) tg-master(destra)

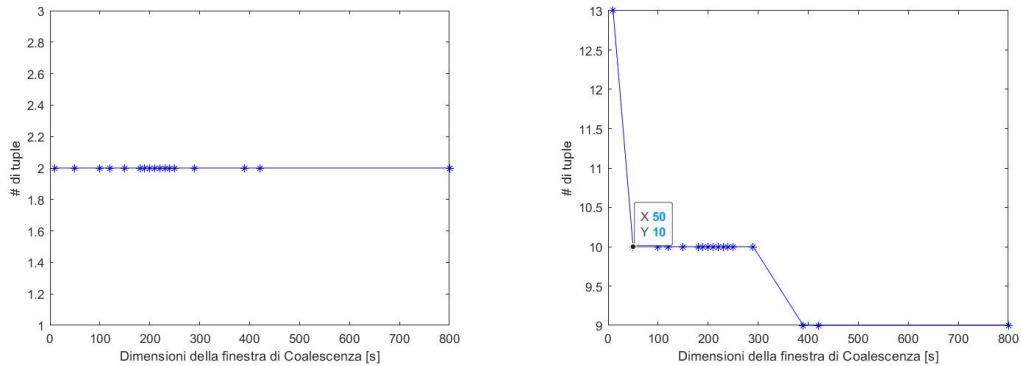


Figura 7.16: Nodi: tg-c572 (sinistra) tg-s044(destra)

Dall'analisi delle Figure precedenti si ottiene che per i nodi tg-c401, tg-master e tg-c238 può essere scelta una CWIN prossima a quella scelta per il sistema completo. Per gli altri nodi, invece, è possibile scegliere una CWIN molto più breve. Di particolare rilievo è, invece, il nodo tg-c572 in cui la scelta della CWIN non ha importanza.

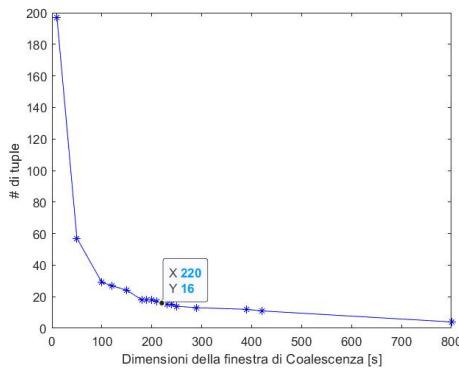


Figura 7.17: Nodo: tg-c238

Generalizzando la scelta si può concludere che per i **nodi computazionali** e per quello master si può essere utilizzata una finestra di coalescenza pari a quella utilizzata per l'intero sistema, dunque, la scelta effettuata in precedenza risulta buona per tali nodi, mentre per i **nodi di tipo storage** tale scelta non è la migliore. Si ripete la stessa analisi anche alle 6 differenti categorie di errore registrate nel log:

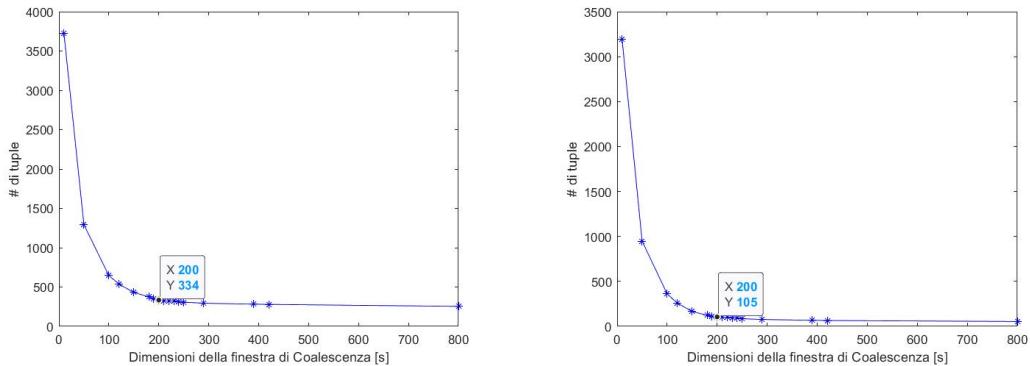


Figura 7.18: Categorie: DEV (sinistra) MEM (destra)

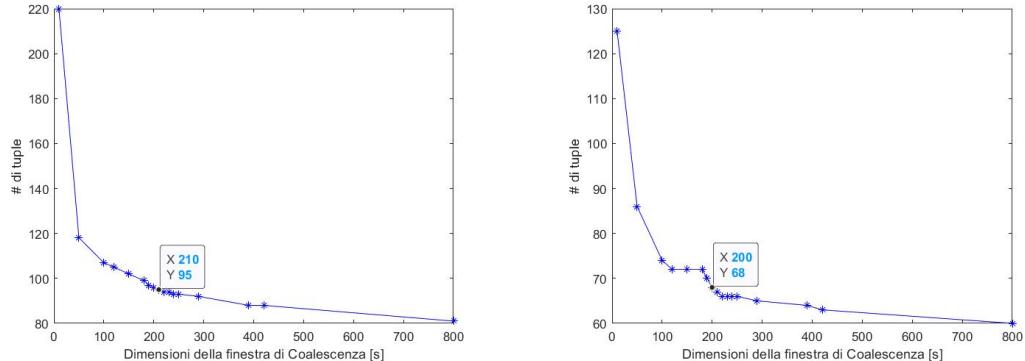
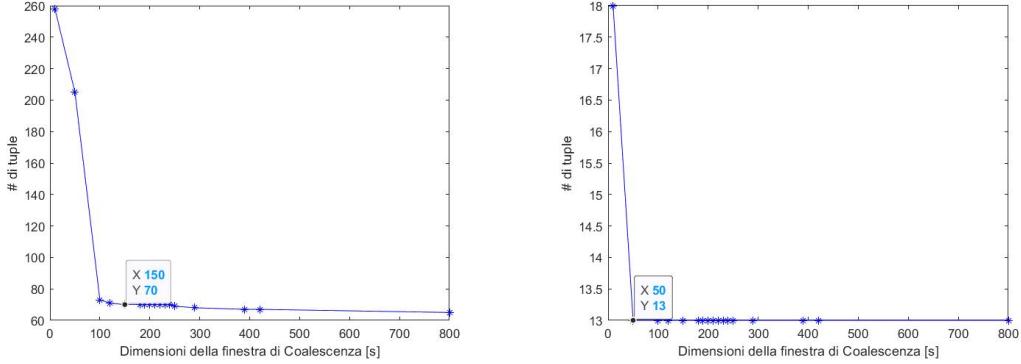


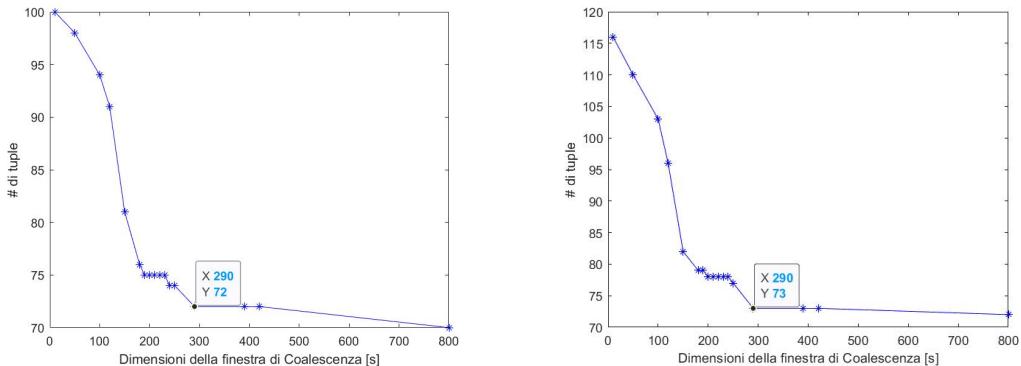
Figura 7.19: Nodi: I-O (sinistra) NET (destra)

**Figura 7.20:** Nodi: PRO (sinistra) OTH (destra)

Per le analisi di sensitività delle categorie di errore il risultato ottenuto sottolinea che per **le classi di errore più frequenti, come DEV, MEM, I-O e NET, può essere scelta la stessa durata della finestra di coalescenza** sia tra di loro che rispetto al sistema complessivo, mentre per **le classi di errore meno frequenti, come OTH e PRO la scelta ricade su una CWIN più breve**.

7.5.2 Coalescence Window per i nodi di Blue Gene

Si ripete lo stesso procedimento anche per il calcolatore Blue Gene valutando le sensitivity analysis per i nodi più presenti nel log:

**Figura 7.21:** Nodi: R71-M0-N4 (sinistra) R12-M0-N0 (destra)

Dall'analisi delle Figure precedenti si può notare che le durate di CWIN dei nodi R71-M0-N4, R12-M0-N0 è maggiore del CWIN del sistema complessivo (infatti, abbiamo un CWIN circa di 290s). Anche per il nodo R63-M0-N0 abbiamo che la CWIN è leggermente più grande della CWIN di sistema, ma inferiore rispetto a quella dei due nodi precedenti (circa 240s).

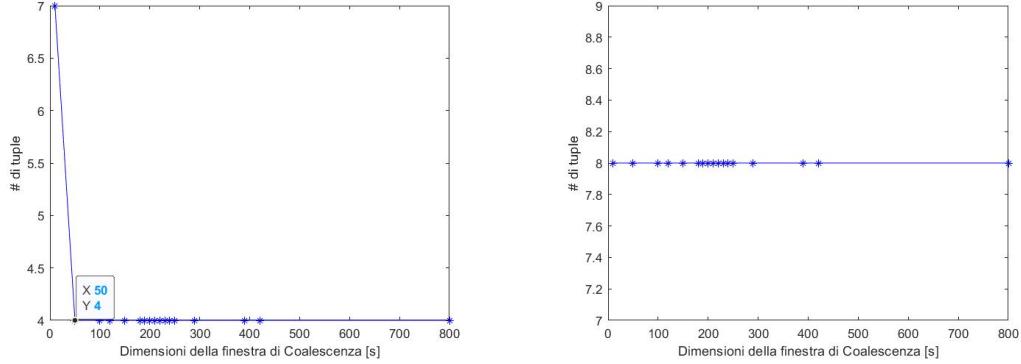


Figura 7.22: Nodi: R63-M0-N2 (sinistra) R03-M1-NF (destra)

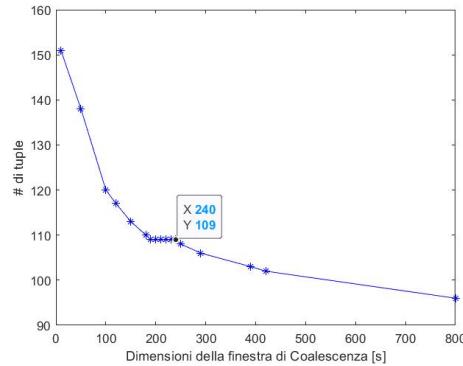


Figura 7.23: Nodo: R63-M0-N0

Mentre, per i nodi R63-M0-N2 e R03-M1-NF abbiamo una finestra di coalescenza notevolmente inferiore rispetto a quella del sistema e nel secondo nodo essa è addirittura irrilevante perché genera solo entry isolate (**memoryless**). In definitiva, si può notare che per i **nodi di Blue Gene che presentano card di IO sia possibile usare una CWIN maggiore di quella scelta per il sistema**, mentre, per i **nodi con sole card computazionali la CWIN scelta risulta più bassa**.

7.6 Confronto reliability di sistema vs. reliability dei nodi del calcolatore Mercury

Si risponde alla seconda domanda **"Cosa si può dire riguardo alla relazione tra l'affidabilità del sistema e l'affidabilità dei nodi?"** calcolando innanzitutto le reliability dei singoli nodi come fatto per il sistema complessivo, ovvero tramite il calcolo degli interarrivi si estrae la CDF del Time to Failure (TTF) del nodo e successivamente da esso si calcola la Reliability come $Rel = 1 - TTF$.

In generale, ci si aspetta che l'affidabilità del sistema sia inferiore all'affidabilità dei suoi nodi più affidabili a causa della possibilità di guasti nei collegamenti tra i nodi. Si cerca, dunque, di capire non solo il tipo di distribuzione ma anche il rapporto di grandezza tra le reliability in gioco.

Come nodi da confrontare per il calcolatore **Mercury** si sono scelti i nodi già visti nel paragrafo precedente dato che sono quelli più frequenti nell'entry log, come già detto. Tuttavia, da quest'analisi è stato escluso il nodo tg-c572 in quanto la sua CWIN non è rilevante; per gli altri nodi si è effettuata l'analisi con la CWIN già individuata.

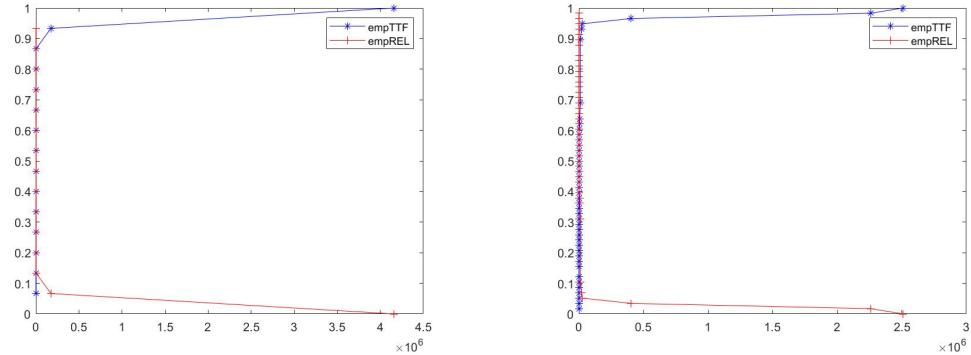


Figura 7.24: Reliability empirica: tg-c238 (sinistra) tg-c401 (destra)

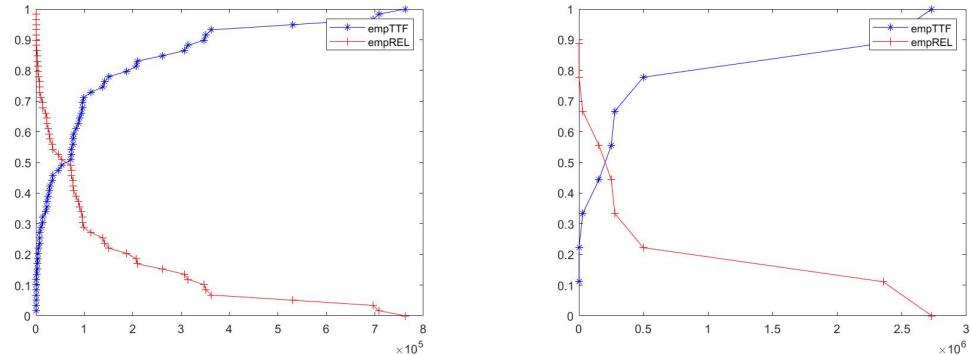


Figura 7.25: Reliability empirica: tg-master (sinistra) tg-s044 (destra)

A partire dalle singole reliability dei nodi, si cerca di individuare una relazione tra la reliability di sistema e quella dei singoli nodi tramite il grafico in Figura 7.26, come già anticipato in precedenza si nota che **l'affidabilità dei singoli nodi è maggiore di quella del sistema; inoltre, si può notare che gli unici nodi che seguono una distribuzione simile a quella del sistema sono i nodi computazionali** e, quindi, sono tali nodi che maggiormente contribuiscono a tale andamento, come già notato dall'analisi della CWIN.

Si effettua la medesima analisi anche per i nodi di **Blue Gene**, i nodi selezionati sono quelli già visti nel precedente paragrafo, ad esclusione del nodo R03-M1-NF per gli stessi motivi che hanno portato all'esclusione del nodo tg-c572 di Mercury. Le reliability dei nodi ottenute sono le seguenti:

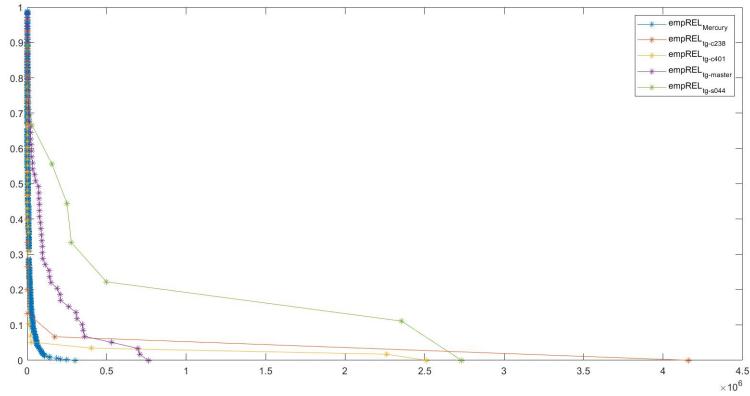


Figura 7.26: Confronto tra Reliability di sistema e Reliability dei nodi per Mercury

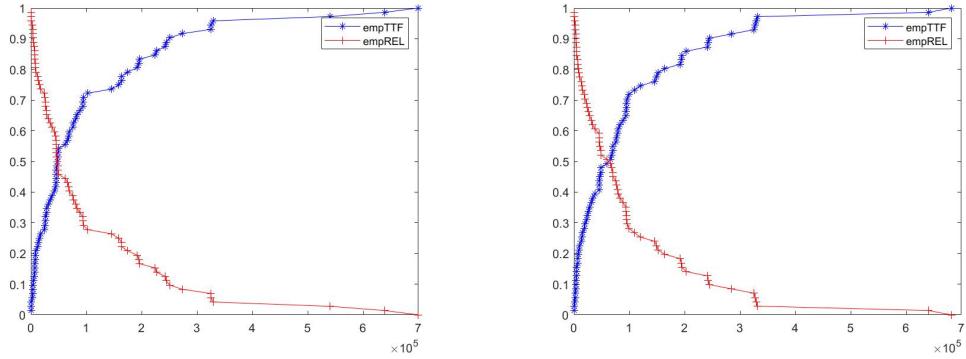


Figura 7.27: Reliability empirica: R12-M0-N0 (sinistra) R71-M0-N4 (destra)

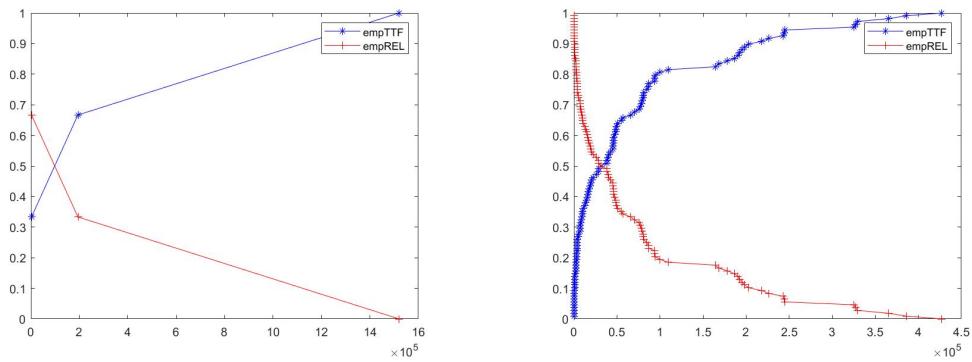


Figura 7.28: Reliability empirica: R63-M0-N2 (sinistra) R63-M0-N0 (destra)

Dunque, dall'analisi della Figura 7.29 si può evincere che anche in questo caso **la reliability del sistema è inferiore a quella dei singoli nodi come ci si aspettava**. Inoltre, **i nodi che hanno un andamento più simile a quello del sistema sono i nodi con maggior presenza di Card di IO** (infatti, R63-M0-N2 non presenta nemmeno una card di IO).

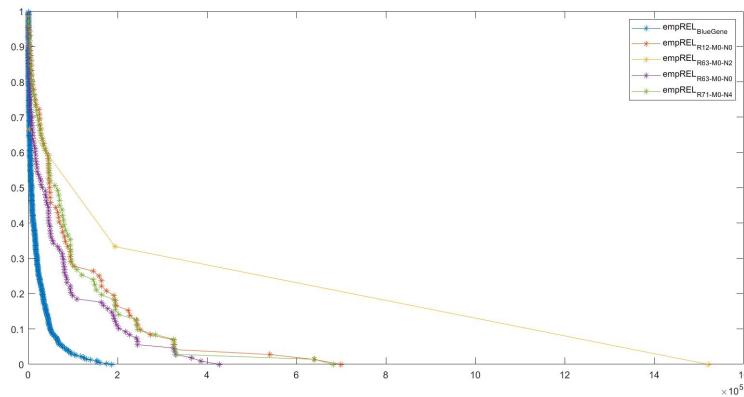


Figura 7.29: Confronto tra Reliability di sistema e Reliability dei nodi per Blue Gene

7.7 Reliability bottlenecks

Si risponde alla terza domanda: **"Esistono dei colli di bottiglia dell'affidabilità (ad es. che contribuiscono maggiormente al numero totale di guasti)"** a partire dalle statistiche calcolate con lo script USlogStatistics.py risalendo ai 10 nodi che causano più failure all'interno dei calcolatori Mercury e Blue Gene.

7.7.1 Bottlenecks di Mercury

10 most node occurring in Mercury LOG		
1	tg-c401	62340
2	tg-master	4098
3	tg-c572	4030
4	tg-s044	3224
5	tg-c238	1273
6	tg-c242	1067
7	tg-c648	643
8	tg-login3	382
9	tg-c117	268
10	tg-c669	267

Tabella 7.3: Bottlenecks di Mercury - nodi

Si nota dalla tabella come la percentuale delle entry del nodo tg-c401 è molto maggiore rispetto agli altri nodi, pertanto questo nodo si comporta come bottleneck per l'affidabilità del sistema. È possibile fare lo stesso ragionamento anche per le categorie di errore:

Si nota anche in questo caso che le categorie DEV e MEM, in particolare DEV, hanno la maggioranza delle entry. Dunque possono essere considerati bottleneck.

7.7.2 Bottlenecks di BlueGene

Si ripete l'analisi per il log di Blue Gene e i risultati sono:

6 most category occurring in Mercury LOG		
1	DEV	57248
2	MEM	12819
3	IO	3702
4	NET	3702
5	PRO	1504
6	ETH	34

Tabella 7.4: Bottlenecks di Mercury - categorie di errori

10 most node occurring in BlueGene LOG		
1	R71-M0-N4	1716
2	R12-M0-N0	1563
3	R63-M0-N2	976
4	R03-M1-NF	960
5	R63-M0-N0	791
6	R36-M1-N0	788
7	R62-M0-N4	515
8	R63-M0-NC	460
9	R63-M0-N8	454
10	R63-M0-N4	452

Tabella 7.5: Bottlenecks di BlueGene

In questo caso non si individua un nodo responsabile di molte più entry all'interno del log. Pertanto non sono presenti bottleneck nel calcolatore Blue Gene.

7.8 Confronto reliability tra nodi simili (Mercury e Blue Gene)

Per rispondere alla quarta domanda **"Nodi funzionali simili (ad esempio, due nodi di calcolo Mercury tg-cX o nodi I/O BG/L Ri:Mx:Nz) presentano parametri di affidabilità simili?"** si considerano per il calcolatore Mercury due nodi computazionali, in particolare, si è deciso di confrontare il nodo tg-c238 ed il nodo tg-c401 che sono i nodi computazionali più frequenti nell'entry log (anche questa volta si è ignorato il nodo tg-c572 perché ritenuto non rilevante ai fini dell'analisi).

Come primo passo dell'analisi, si è messo a confronto la dimensione della finestra di coalescenza dei due nodi, ottenendo il seguente grafico:

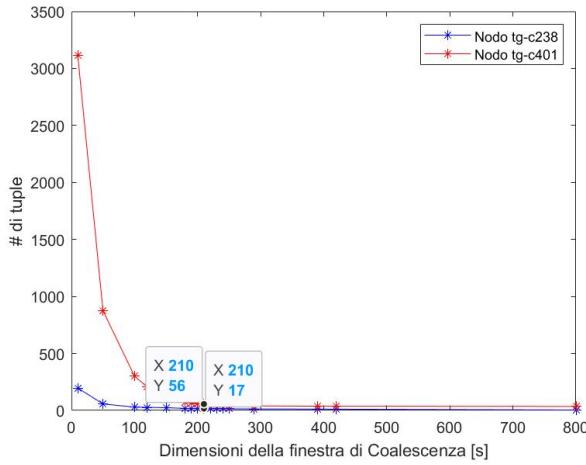


Figura 7.30: Confronto tra CWIN del nodo tg-c238 e del nodo tg-c401

Come si nota dalla Figura 7.30, i due nodi hanno una finestra di coalescenza di dimensioni equiparabili, in particolare, si può scegliere un valore CWIN pari a 210s.

Dunque, a partire dalla CWIN individuata, si è passato al confronto delle reliability empiriche dei due nodi calcolate allo stesso modo dei paragrafi precedenti, ovvero a partire dalla CDF degli interarrivi ed ottenuta come $Rel = 1 - TTF$. Si è poi costruito il seguente grafico:

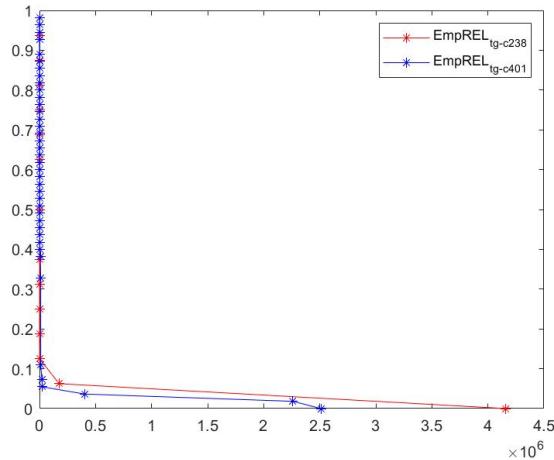


Figura 7.31: Confronto tra Reliability empirica del nodo tg-c238 e del nodo tg-c401

Dalla Figura 7.31 si può notare che le due reliability empiriche sono molto simili, sia come andamento che come ordini di grandezza, e quindi si può concludere che nodi computazionali del calcolatore Mercury presentano parametri di affidabilità pressoché simili.

Allo stesso modo di quanto fatto per Mercury, si scelgono due nodi funzionali simili anche per il calcolatore Blue Gene; in particolare, si è scelto di considerare i nodi R71-M0-N4 e R12-M0-N0 che possono essere considerati nodi di IO. Si è, dunque, considerata la dimensione della finestra di coalescenza di entrambi i nodi come mostrato nel seguente grafico:

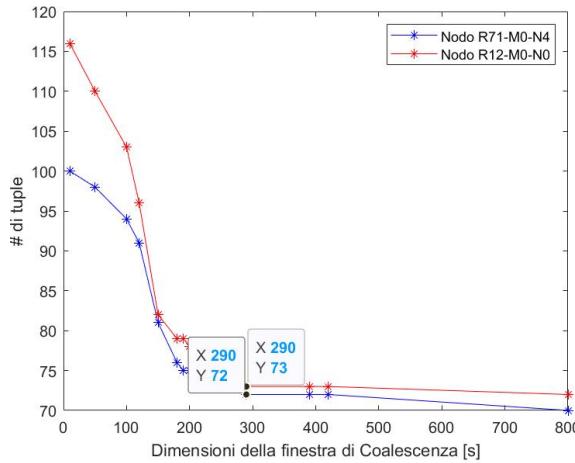


Figura 7.32: Confronto tra CWIN del nodo R71-M0-N4 e del nodo R12-M0-N0

Dalla Figura 7.32, si può notare che le due CWIN sono molto simili e pari a 290s, dunque, si è proceduto al confronto delle reliability empiriche dei nodi R71-M0-N4 e R12-M0-N0 per determinare se i due nodi hanno anche affidabilità simile.

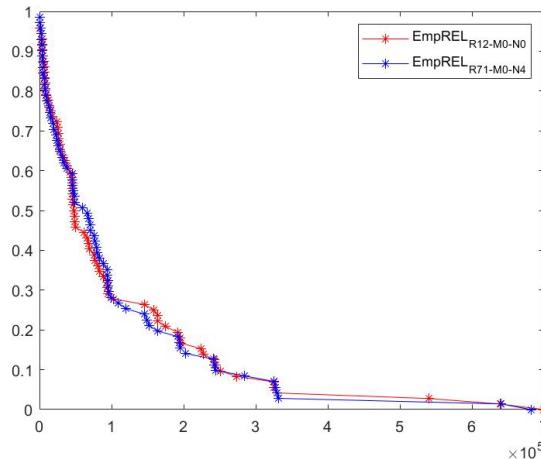


Figura 7.33: Confronto tra Reliability empirica del nodo R71-M0-N4 e del nodo R12-M0-N0

Analizzando la Figura 7.33 si può notare che le due reliability empiriche sono molto simili e, quindi, si può concludere che nodi di IO del calcolatore Blue Gene presentano parametri di affidabilità pressoché simili.

7.9 Correlazione tra nodi e tipi di errore nel calcolatore Mercury

Si risponde alla quinta domanda **"Esiste una relazione tra i tipi di errore tipi di errore e il nodo (MERCURY)"** sapendo che ad ogni entry è associata la tipologia di errore sarà possibile studiare la correlazione tra i nodi e tipi di errore.

Si può definire, infatti, che i nodi di calcolo sono tipicamente soggetti ad errori di tipo DEV o MEM. Mentre nel caso del nodo master esso è soggetto per la maggior parte ad errori di tipo NET. Invece, entrambi i nodi di tipo Login

Correlazione tra nodi e tipi di errore Mercury						
Nodo	DEV	MEM	IO	NET	PRO	OTH
tg-c401	50782	11558	0	0	0	0
tg-master	0	0	452	3639	0	0
tg-c572	3176	845	0	0	0	0
tg-s044	0	0	3208	2	0	14
tg-c238	1071	0	230	3	0	0
tg-c242	918	149	0	0	0	0
tg-c648	643	0	0	0	616	0
tg-login3	0	0	381	1	0	0
tg-c117	263	5	0	0	0	0
tg-c669	257	10	0	0	0	0

% di Correlazione tra nodi e tipi di errore Mercury						
Nodo	DEV	MEM	IO	NET	PRO	OTH
Calcolo	80%	18.2%	0	0	1.8%	0
Login	0	0	99.7%	0.3%	0	0
Master	0	0	12%	88%	0	0
Storage	0	0	99%	0.3%	0	0.9%

Tabella 7.6: Correlazione tra nodi e tipi di errore nel calcolatore Mercury

e Storage sono soggetti per la stragrande maggioranza (99%) ad errori di tipo IO. Quest'ultima osservazione è in accordo al fatto che i nodi sono principalmente interessati a comunicare con altri nodi e/o con l'utente.