Simulazione su larga scala di policy di scheduling con MAUI

Mirko Mariotti

FrancescoCantini

Leonello Servoli

Claudio Tanci

Il sito INFN GRID di Perugia

- I benefici della gestione congiunta, hanno spinto i gruppi di ricerca operanti presso il dipartimento di fisica a riunire le rispettive risorse di calcolo.
- Il risultato è un cluster con funzione di Batch System
- Ingresso in INFN GRID nel 2004

Peculiarità della nostra farm

- **Necessità di ripartire equamente tra i gruppi la potenza di calcolo in base alle risorse hardware da questi apportate:**
  - quote computazionali di gruppo
  - quote computazionali di utente
- **Sottomissione job da locale e via GRID (farm ibrida)**
- **Worker Nodes su rete nascosta**
- **Uso di macchine virtuali**

Riepilogo delle risorse hardware apportate dai vari gruppi di ricerca

Esperimento	Totale Macchine	File Server	Spazio Disco	User Interface	Nodi Di Calcolo	Altri Nodi	Totale CPU Core
GRID	22	0	1,5 TB	0	17	5	44
BABAR	3	1	0,75 TB	1	1	0	6
CMS	22	4	> 10 TB	2	13	5	44
GLAST	11	3	> 10 TB	1	8	0	22
NA48	18	3	5,5 TB	2	13	0	36
Teorici	5	0	0 TB	1	4	0	10
Virgo	7	0	0 TB	1	6	0	14
Altri fondi	1	0	0 TB	0	0	1	2
Risorse totali	89	8	> 25 TB	8	62	11	178

Formulazione del problema di ottimizzazione dell'uso delle risorse

1. **Esigenze del livello degli utenti**
   - Usufruire della propria quota computazionale quando serva
   - Usufruire di tutte le risorse di calcolo libere quando serva
2. **Esigenze del livello dei gruppi**
   - Usufruire della propria quota computazionale quando serva
   - Usufruire di tutte le risorse di calcolo libere quando serva
   - Bilanciare i propri utenti, nell'uso della quota computazionale di gruppo
3. **Esigenze dell'amministratore del sistema**
   - Bilanciamento dei gruppi nell'assegnazione delle risorse di calcolo
   - Massimo utilizzo del sistema

Principali code attive e relativi wall-time

Nome coda	*Walltime*	*Nome coda*	*Walltime*	*Nome coda*	*Walltime*
cms-short	02:00:00	na48-short	02:00:00	babar-short	02:00:00
cms-medium	24:00:00	na48-medium	24:00:00	babar-medium	24:00:00
cms-long	52:00:00	na48-long	52:00:00	babar- long	52:00:00
cms-infinite	780:00:00	na48-infinite	780:00:00	babar-infinite	780:00:00
virgo-short	02:00:00	glast-short	02:00:00	theo-short	02:00:00
virgo-medium	24:00:00	glast-medium	24:00:00	theo-medium	24:00:00
virgo-long	52:00:00	glast-long	52:00:00	theo-long	52:00:00
virgo-infinite	780:00:00	glast-infinite	780:00:00	theo-infinite	780:00:00
				test	24:00:00
				CODE LOCALI	
na48	120:00:00				
babar	72:00:00				
virgo	120:00:00	grid	72:00:00	**CODE GRID**	
cms	120:00:00	cert	72:00:00	theophys	72:00:00

Simulazione

La soluzione al problema sta ovviamente in una opportuna configurazione dello scheduler Maui.

Purtroppo raggiungere un tuning accettabile richiede parecchio tempo, numerose prove ed osservazioni.

Un'ulteriore complicazione può inoltre provenire dal cambiamento delle condizioni di utilizzo della farm.

La simulazione del sistema può accelerare enormemente tale indagine

File di configurazione di Maui

- La configurazione di Maui risiede nel file *maui.cfg* sotto forma di ***direttive*** e ***policy di scheduling***. Anche in modalità simulazione la maggior parte delle direttive mantiene lo stesso significato.

```
[mirko]> /home/mirko % cat config-simul-header
# maui.cfg 3.2.6p16
SERVERHOST              test-simulator
ADMIN1                  root
AMCFG[bank]  TYPE=NONE
RMPOLLINTERVAL          00:00:01
SERVERPORT              42559
SERVERMODE              SIMULATION

SIMRESOURCETRACEFILE    resources
SIMWORKLOADTRACEFILE    workload
#SIMRANDOMDELAY         2
#SIMSTOPITERATION       1
SIMAUTOSHUTDOWN         TRUE
#SIMEXITITERATION       100000
#SIMFLAGS               [NONE]
SIMINITIALQUEUEDEPTH     1
SIMJOBSUBMISSIONPOLICY  NORMAL
[mirko]> /home/mirko %
```

- Tramite la direttiva SERVERMODE è possibile dire a Maui di entrare in modalità simulazione.

- Tramite un set di direttive è possibile controllare diversi comportamenti del simulatore.

File Risorse di Maui

- Uno scheduler Maui puo' generare con il comando *schedctl -n* un file che descrive in maniera esaustiva l'intero batch system.

```
[mirko]> /home/mirko % cat resources
COMPUTENODE AVAILABLE 0 test-node01 test-master 256M 256M 1M 1 1 1 0 linux [NONE] [NONE] [batch:1][alpha:1][gamma:1][beta:1] [DEFAULT] [M
COMPUTENODE AVAILABLE 0 test-node02 test-master 256M 256M 1M 1 1 1 0 linux [NONE] [NONE] [batch:1][alpha:1][gamma:1][beta:1] [DEFAULT] [M
COMPUTENODE AVAILABLE 0 test-node03 test-master 256M 256M 1M 1 1 1 0 linux [NONE] [NONE] [batch:1][alpha:1][gamma:1][beta:1] [DEFAULT] [M
COMPUTENODE AVAILABLE 0 test-node04 test-master 256M 256M 1M 1 1 1 0 linux [NONE] [NONE] [batch:1][alpha:1][gamma:1][beta:1] [DEFAULT] [M
```

- Il file è composto da tanti **record** quanti sono i nodi del cluster e ne riassume le caratteristiche come ad esempio nome del nodo, code ammesse, numero di processori ecc.

- Il file di risorse di Maui serve per dire al simulatore come e' fatto il cluster da simulare. Il file si passa a Maui tramite la direttiva SIMRESOURCETRACEFILE

Traccia di Maui

- Uno scheduler Maui riempie, durante il suo normale funzionamento, un **file** detto **di traccia** in cui confluiscono tutte le informazioni di rilievo dei job processati dal sistema.

- La traccia è composta da tanti **record** quanti sono i job la cui esecuzione è stata ultimata ed ogni record è composto da **44 campi** che contengono le informazioni temporali, le credenziali e le statistiche di esecuzione calcolate per il job

- La traccia Maui può essere usata per due scopi:
  - **analisi dello scheduling** prodotto nel sistema reale date le policy impostate o **analisi statistiche** di vario genere
  - **fornire un file di input al simulatore** di Maui tramite la direttiva SIMWORKLOADTRACEFILE

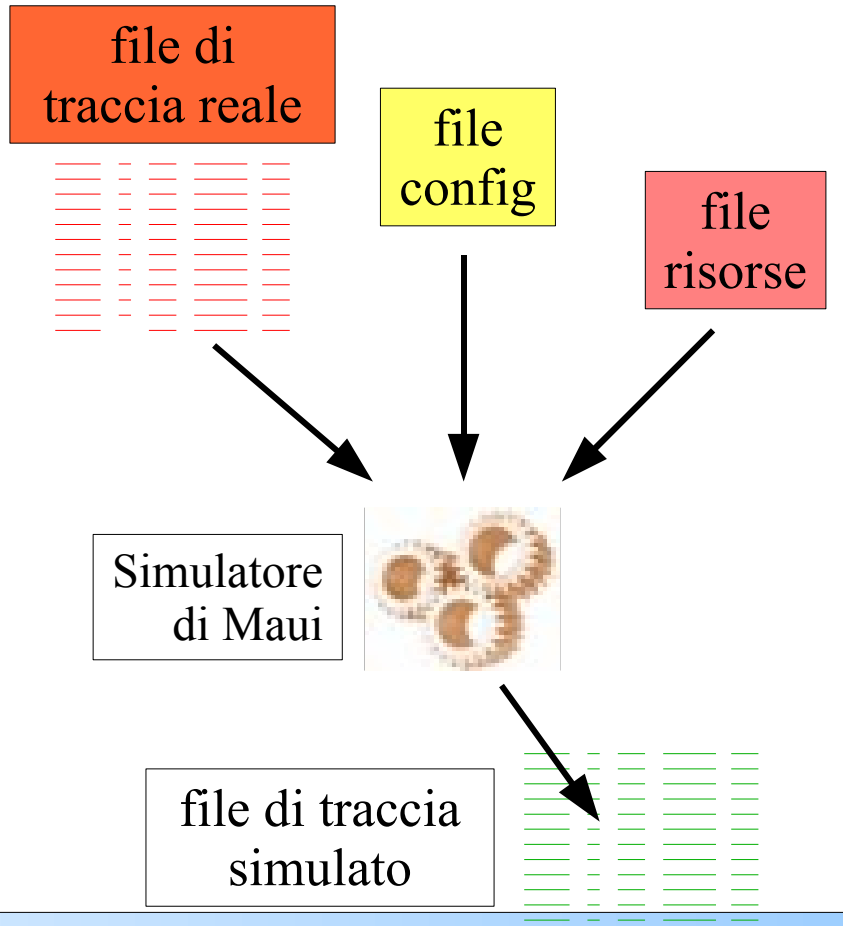
Traccia di Maui

```
-rw-r--r--  1 mirko users    1179 Dec  8 10:23 workload
[mirko]> /home/mirko % cat workload
7690 0 1 rosso_1 rosso 7200 Completed [batch:1] 1220441214 1220441215 1220441215 1220441247 \
[NONE] [NONE] [NONE] >= 0M >= 0M [NONE] 1220359613 1 0 [NONE]:DEFAULT [RESTARTABLE] [NONE] \
[NONE] [NONE] 0 2,21 DEFAULT 1 0M 0M 0M 0 2140000000 sundance4 POST18 [NONE] [NONE] [DEFAULT] [NONE] [NONE]

7691 0 1 blu_1 blu 7200 Completed [batch:1] 1220441214 1220441215 1220441215 1220441247 \
[NONE] [NONE] [NONE] >= 0M >= 0M [NONE] 1220359613 1 0 [NONE]:DEFAULT [RESTARTABLE] [NONE] \
[NONE] [NONE] 0 5,69 DEFAULT 1 0M 0M 0M 0 2140000000 sundance3 POST18 [NONE] [NONE] [DEFAULT] [NONE] [NONE]

7692 0 1 rosso_1 rosso 7200 Completed [batch:1] 1220441224 1220441225 1220441225 1220441277 \
[NONE] [NONE] [NONE] >= 0M >= 0M [NONE] 1220359623 1 0 [NONE]:DEFAULT [RESTARTABLE] [NONE] \
[NONE] [NONE] 0 0,00 DEFAULT 1 0M 0M 0M 0 2140000000 sundance2 POST18 [NONE] [NONE] [DEFAULT] [NONE] [NONE]

7693 0 1 blu_1 blu 7200 Completed [batch:1] 1220441224 1220441225 1220441225 1220441277 \
[NONE] [NONE] [NONE] >= 0M >= 0M [NONE] 1220359623 1 0 [NONE]:DEFAULT [RESTARTABLE] [NONE] \
[NONE] [NONE] 0 6,07 DEFAULT 1 0M 0M 0M 0 2140000000 sundance1 POST18 [NONE] [NONE] [DEFAULT] [NONE] [NONE]
[mirko]> /home/mirko %
```

Simulatore di Maui

E' un **software** che implementa gli stessi algoritmi dello scheduler reale. Esso fa previsioni sull'esito dello scheduling che un certo sistema virtuale effettua, dato un insieme di job utente e di policy.

- **Riceve in input:**
  - file di traccia Maui reale
  - file di configurazione contenente le policy di scheduling
  - file con la descrizione delle risorse del Batch System virtuale
- **Restituisce**
  - file di traccia Maui simulato



Quali sono i nostri obbiettivi...

- ***Simulare una particolare configurazione dello scheduler*** senza doverla “provare sul campo”, processo che sarebbe più lento e piu' invasivo, utilizzando lo stesso workload reale.

- Valutare la risposta di un certo set di policy a dei ***workload di tipo diverso***.

- ***Variare piu' configurazioni*** mantenendo lo stesso workload alla ricerca di quelle ottimali.

... e cosa ci serve per raggiungerli ?

- Occorre che il simulatore **funzioni**.
- Occorre che il simulatore sia **validato**.
- Occorre un'infrastruttura di stoccaggio e recupero dei file di traccia (**repository**).
- Occorre dei **tool di analisi** che ci permettano di estrarre rapidamente informazioni dai file di traccia.
- Occorre trovare delle **metriche** che ci permettano di valutare quantitativamente la bontà delle nostre configurazioni.

Debugging del simulatore

Sottomettendo al simulatore i già descritti *file di traccia*, il *file delle risorse* ed il *file di configurazione* ci si è accorti chequesto **non funziona**, in quanto i risultati ottenuti sono totalmente inconsisteti.

In realtà dopo un attento studio si è capito che **le anomalie sono eliminabili** e dovute soprattutto alla scarsa documentazione disponibile.

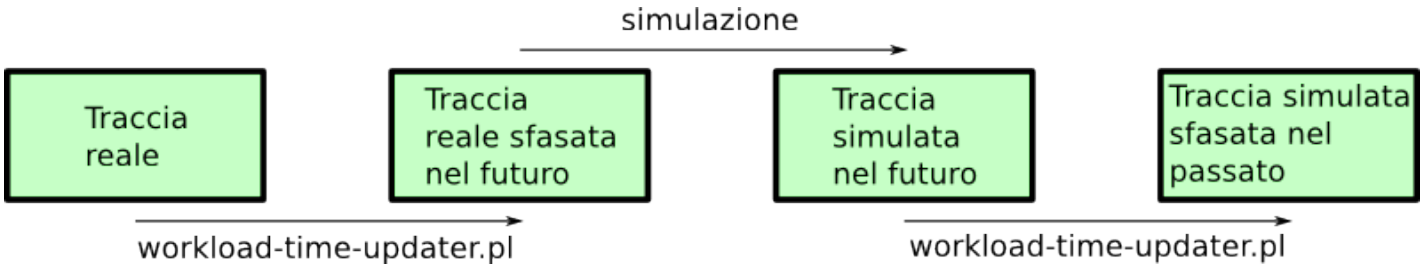
Le principali anomalie individuate e trattate sono state:

- ***Sfasamento temporale.***
- ***Riordino del file di traccia.***
- ***Direttive inesistenti***

Debugging del simulatore - sfasamento temporale -

- Il simulatore utilizza solamente alcune informazioni presenti nel file di traccia, in particolare non gestisce correttamente il **tempi assoluti** dei job della traccia.
- Tutte le simulazioni avvengono invece a partire dal tempo reale e ciò crea problemi qualora si vogliano fare confronti tra la reale e la simulata.

Abbiamo ovviato al problema scalando adeguatamente i tempi assoluti della traccia, prima e dopo la simulazione.



Debugging del simulatore - riordino dei file di traccia -

- Nel generare il file di traccia, Maui scrive una riga ogni volta che un job finisce.
- Non necessariamente quindi l'ordine del file segue quello della sottomissione dei jobs.
- Ci si aspetterebbe che in modalita' simulazione Maui tenesse conto di cio' ed utilizzasse il jobID per identificare la corretta sequenza.
- Non è cosi ! Il file è elaborato secondo l'ordine di scrittura e ciò crea condizioni di inconsistenza con la realtà.

Occorre riordinare il file di workload prima di utilizzarlo come input per il simulatore.

Debugging del simulatore
- direttive inesistenti -

Abbiamo riscontrato che alcune direttive relative alla simulazione, sebbene documentate nella guida per l'amminstratore di Maui, non sono in realtà supportate.

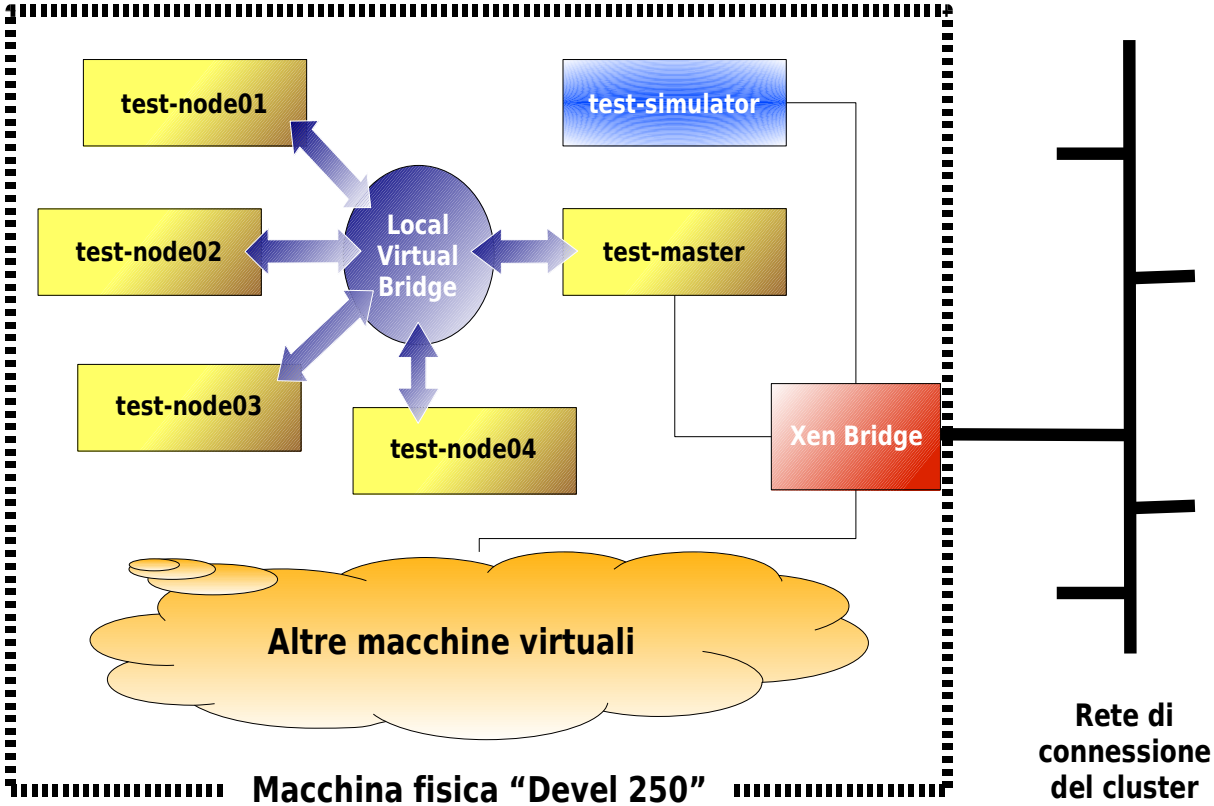
Ad esempio compiendo dei test sulla direttiva "SIMRANDOMDELAY" si è scoperto che questa non nemmeno implemetata nel codice sorgente del simulatore.

Validazione del simulatore

- La possibilità di conoscere in anticipo l'esito di un esperimento o l'evoluzione di un sistema comporta benefici economici enormi. Un simulatore è uno strumento prezioso.
- La simulazione è comunque un'approssimazione del mondo reale e quindi introduce un fattore di errore.
- Tale errore può essere tollerabile ai fini di utilizzo del simulatore.
- Prima di essere impiegato il simulatore deve essere quindi ***validato*** con una serie di prove per stabilire se l'errore introdotto è sistematicamente al disotto della soglia di tolleranza stabilita.

Infrastruttura di validazione

Batch System di test basato su nodi virtuali (VM)



Tools software

1. Tool per la correzione dei bug del simulatore
2. Tool di analisi dei dati
3. Tool di visualizzazione della traccia Maui
4. Tools di automazione e standardizzazione di operazioni ripetitive

Tecniche di validazione

- **Validazione Black Box:**

  1. **generazione** di un file di traccia reale dal batch system di test;
  2. **simulazione** della traccia con le stesse policy del sistema reale;
  3. **confronto** della traccia simulata con quella reale.

- **Validazione per Conferma Di Previsione:**

  1. **generazione** di un file di traccia reale dal batch system di test;
  2. **previsione** dello scheduling ottenibile date nuove policy;
  3. **simulazione** della traccia con le nuove policy;
  4. **confronto** della traccia simulata con la previsione effettuata.

Test di validazione condotti

Verifiche di base

- **Rispetto dei tempi** di accodamento e delle durate di esecuzione dei job
- Correttezza logica dell'**ordine di allocazione** dei nodi di calcolo
- Correttezza logica dell'**ordine di scheduling** prodotto
- Corretta attuazione di politiche di scheduling elementari (es. FIFO)
- **Indipendenza** dalle caratteristiche e dalle credenziali dei job

Verifiche del meccanismo di priorità

- Rispetto dei differenti valori di priorità attribuiti ad **utenti**, **gruppi** e **code** di sottomissione dei job

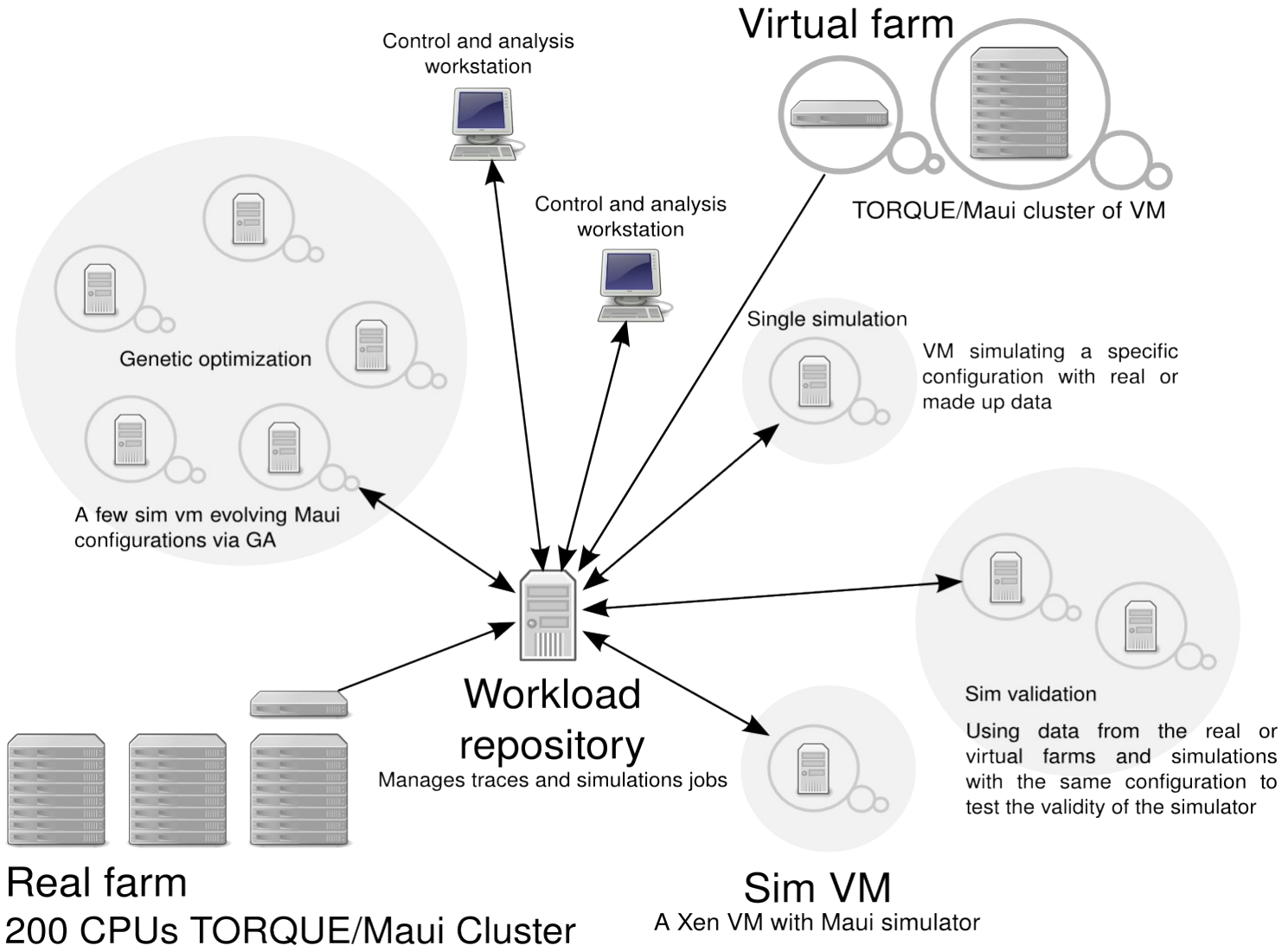
Verifica di policy di *fairness*

- Corretta attuazione della ***politica di restrizione delle allocazioni di tipo "hard and soft limits"*** impostata per **utenti**, **gruppi** e **code**

Test di validazione condotti

Tutti test di validazione sono stati superati

Infrastruttura complessiva



Accesso al repository dei workload: "sync-traces.sh"

Si tratta di un tool per l'accesso al workload repository, con le seguenti caratteristiche:

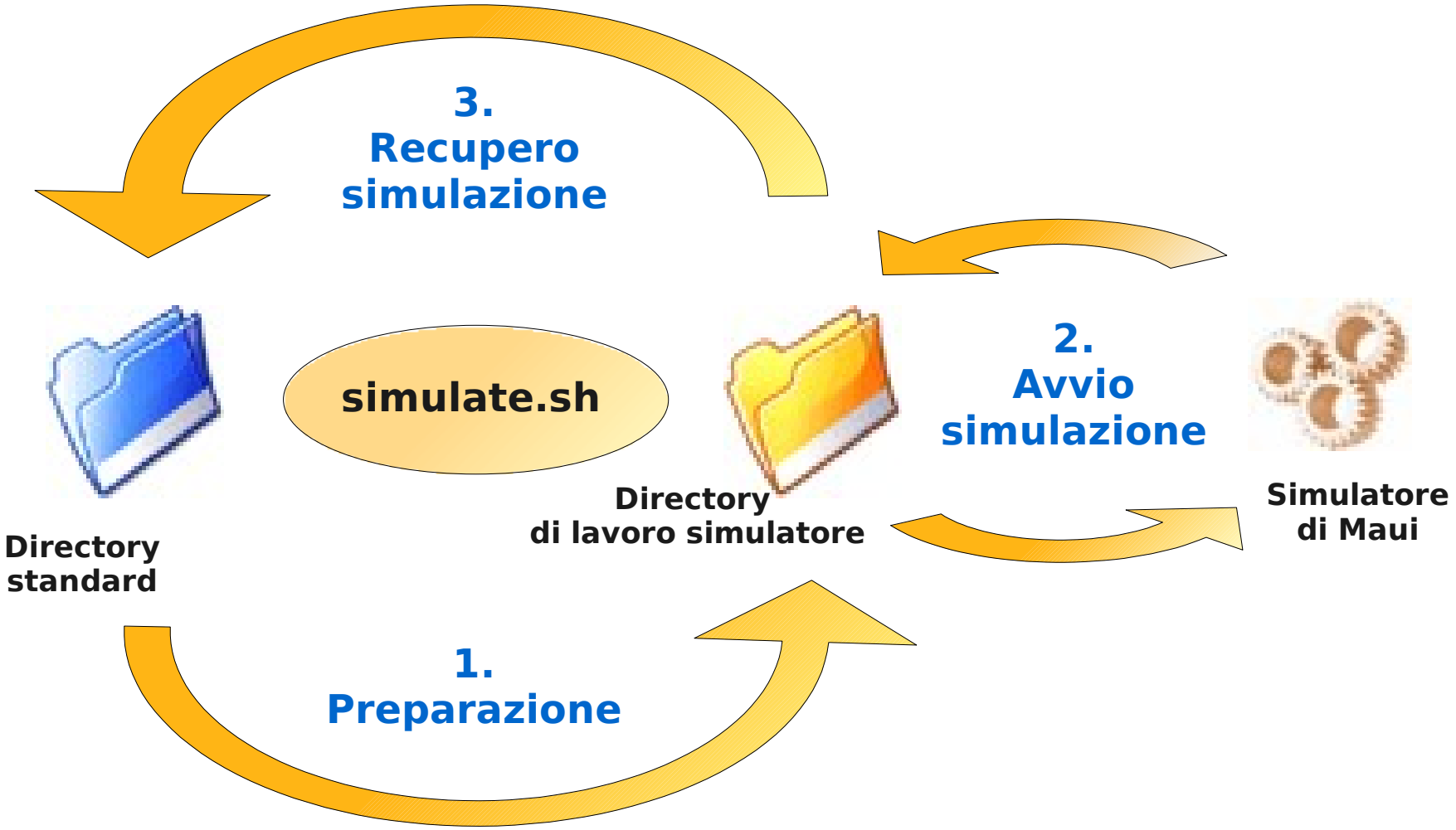
- effettua la ***sincronizzazione*** tra tracce locali e remote
- gestisce le "***feature***" associate a tracce e ***"Runs"*** come ad esempio i meccanismi degli accessi al repository in mutua esclusione o per l'archiviazione di metadati

Wrapper di simulazione: "simulate.sh"

- Per automatizzare tutti i passi descritti fino a qui abbiamo realizzato uno script *simulate.sh* che si occupa di tutti gli aspetti.

- *simulate.sh* esegue questa sequenza di azioni:
  - prende in input una ***directory standard*** (eventualmente anche sul repository di tracce) che deve contenere ***tutti i file necessari alla simulazione***.
  - Inizializza l'ambiente di simulazione e verifica che ci siano tutti i files necessari.
  - Fa le modifiche necessarie ai files (sfasamento e riordino)
  - Esegue la simulazione.
  - Archivia i risultati nella directory vista in precedenza.

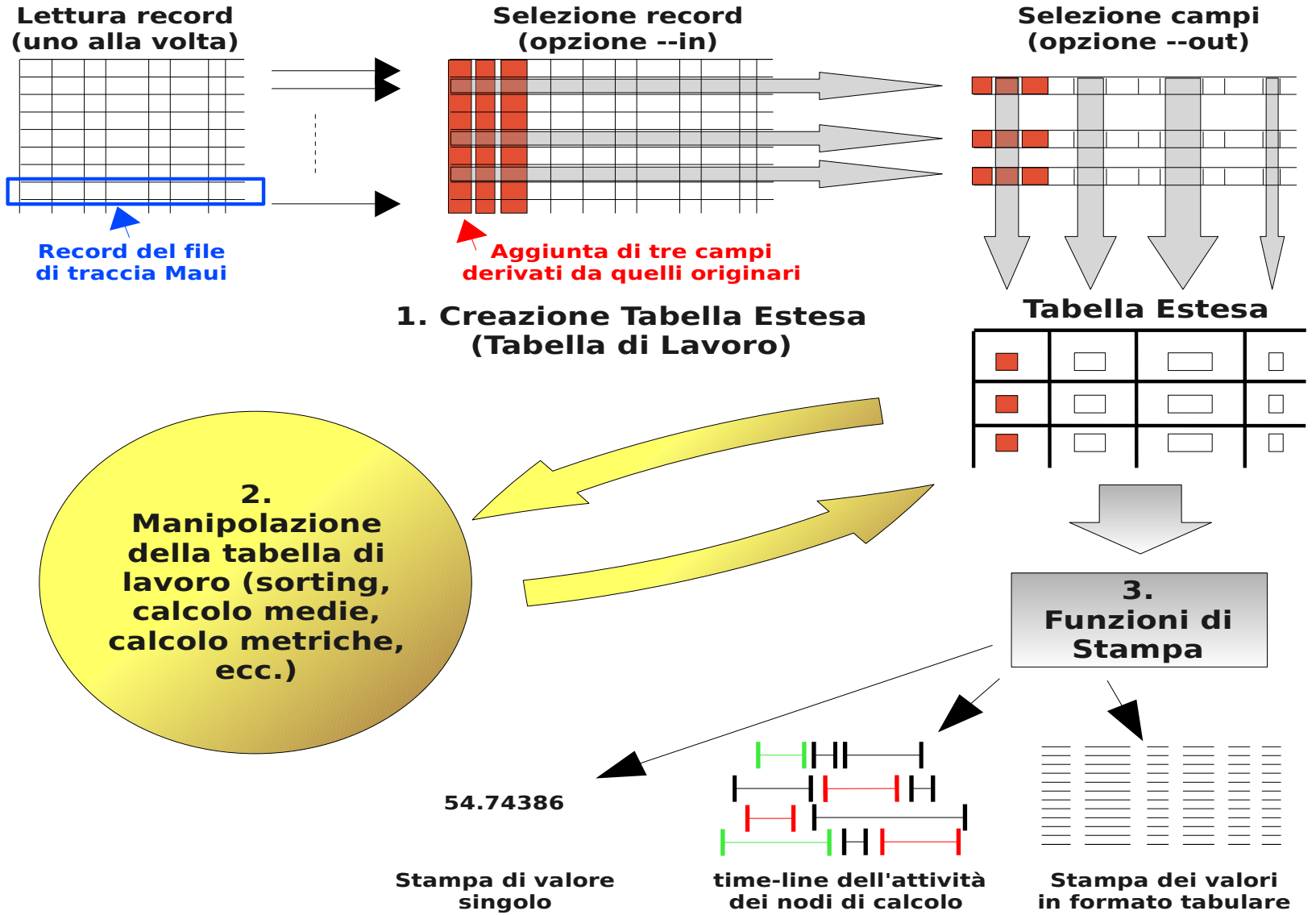
Wrapper di simulazione: "simulate.sh"



Tool di analisi: "workload-analyzer.pl" (1)

- E' uno strumento per ***l'estarzione***, la ***manipolazione*** e ***l'analisi*** dei contenuti informativi racchiusi nella traccia di Maui.
- Sfrutta la naturale vocazione di ***"Perl"*** alla manipolazione di stringhe e file di testo (la traccia).
- E' in grado di eseguire operazioni di:
  - ***selezione di record*** della traccia, sulla base di filtri definiti dall'utente
  - ***selezione di campi*** specifici dei record
  - ***sorting*** (diversi tipologie)
  - calcolo di ***valori medi*** e di ***"media aggregata"*** sui campi richiesti
  - calcolo di diverse ***metriche*** per l'analisi di parametri inerenti lo scheduling dei job e le performance di sistema

Tool di analisi: "workload-analyzer.pl" (2)



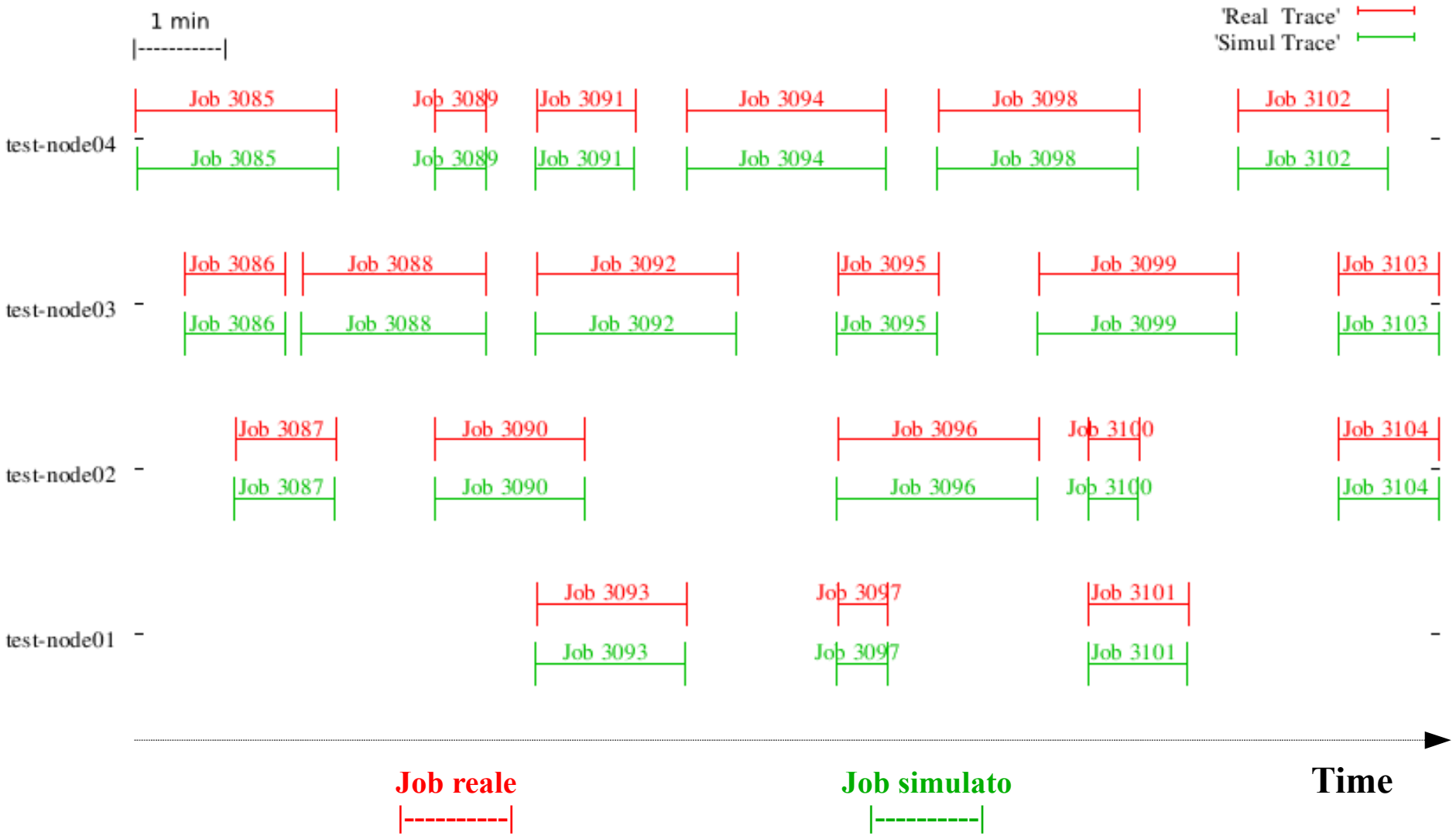
Tool di visualizzazione: "trace-timeliner.pl" (1)

- Serve a visualizzare le ***durate di esecuzione dei job*** presenti nella traccia di Maui, ***ripartite per nodo di calcolo esecutore***.
- E' in grado di graficare simultaneamente un numero arbitrario di tracce, al fine di un rapido ***confronto visivo*** tra le stesse.
- Le timeline dei job possono essere accompagnate da ***qualsiasi informazione*** presente nella traccia Maui o nella "traccia estesa".
- E' possibile ***impostare filtri*** per ricavare le timeline di esecuzione dei soli job che soddisfino determinate condizioni .

Tool di visualizzazione: "trace-timeliner.pl" (2)

- E' scritto in ***"Perl"*** e si avvale:
  - di ***"workload'analyzer.pl"*** per estrarre dalle tracce Maui i dati di interesse
  - del tool *"**GNUplot**"* per la rappresentazione grafica dei dati elaborati.
- Gli utilizzi prevalenti di questo strumento grafico riguardano:
  - l'avere ***percezione visiva*** dello scheduling attuato da Maui
  - il confronto deterministico ***traccia reale – traccia simulata***
  - la ***verifica visiva*** dell'attività dei WN in produzione.

Esempio dell'operazione di confronto traccia reale/simulata



Criteri di misurazione

La valutazione quantitativa della bontà delle policy di scheduling testate, è effettuata per mezzo di metriche di valutazione, allo scopo definite.

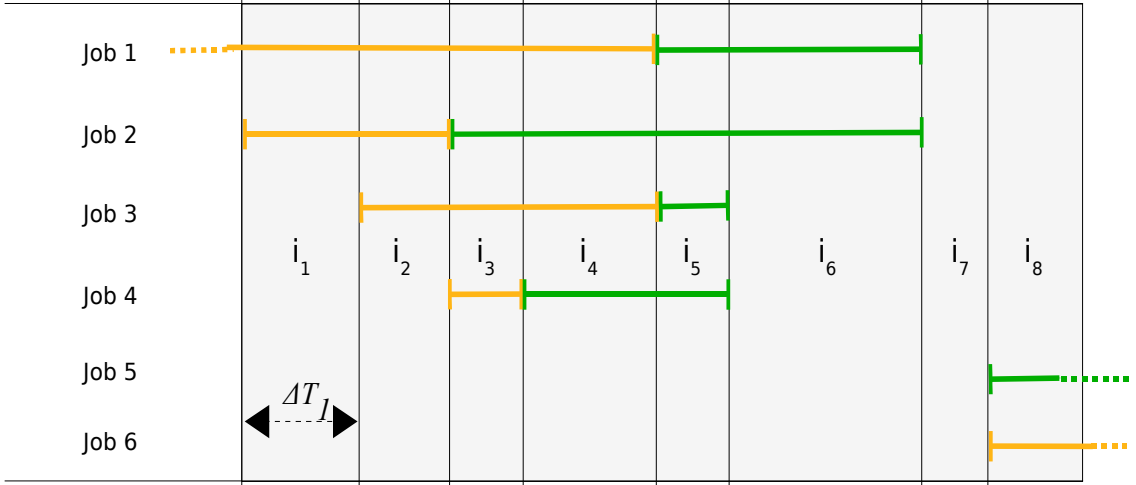
Tali metriche sono implementate in

"workload-analyzer.pl"

Metrica di equità

Ris (Risorse spettanti di diritto) = 3 Nodi di calcolo

Q (Queued Job)	**2**	**3**	**3**	**2**	**0**	**0**	**0**	**1**
R (Running Job)	**0**	**0**	**1**	**2**	**4**	**2**	**0**	**1**
Eq_{ε} (Equità elementare)	**0**	**0**	⅓	⅔	**1**	**1**	**1**	⅓



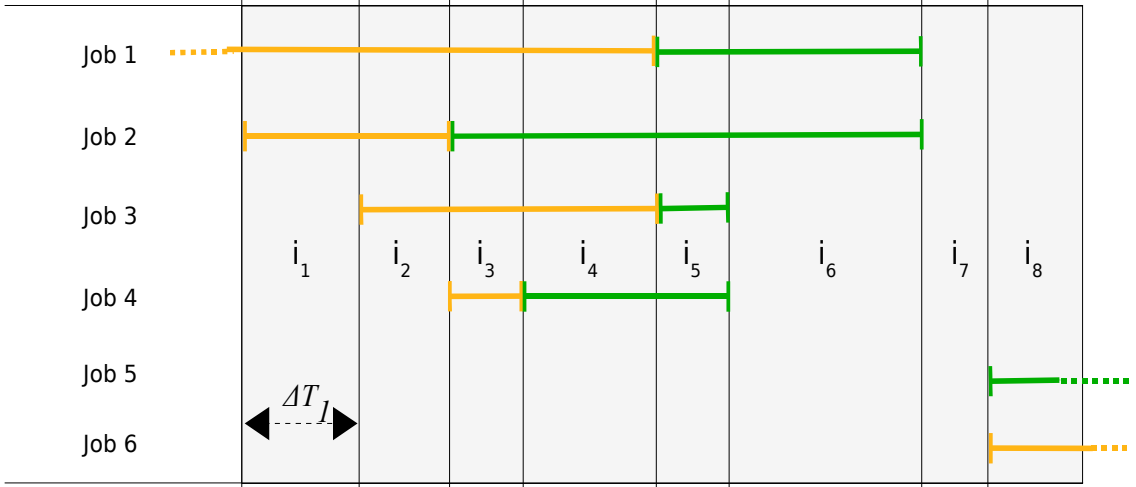
Δt_i : Intervallo di sovrapposizione i-esimo

I job appartengono ad un solo utente/gruppo

Metrica di equità

Ris (Risorse spettanti di diritto) = 3 Nodi di calcolo

Q (Queued Job)	2	3	3	2	0	0	0	1
R (Running Job)	0	0	1	2	4	2	0	1
Eq_ε (Equità elementare)	0	0	⅓	⅔	1	1	1	⅓



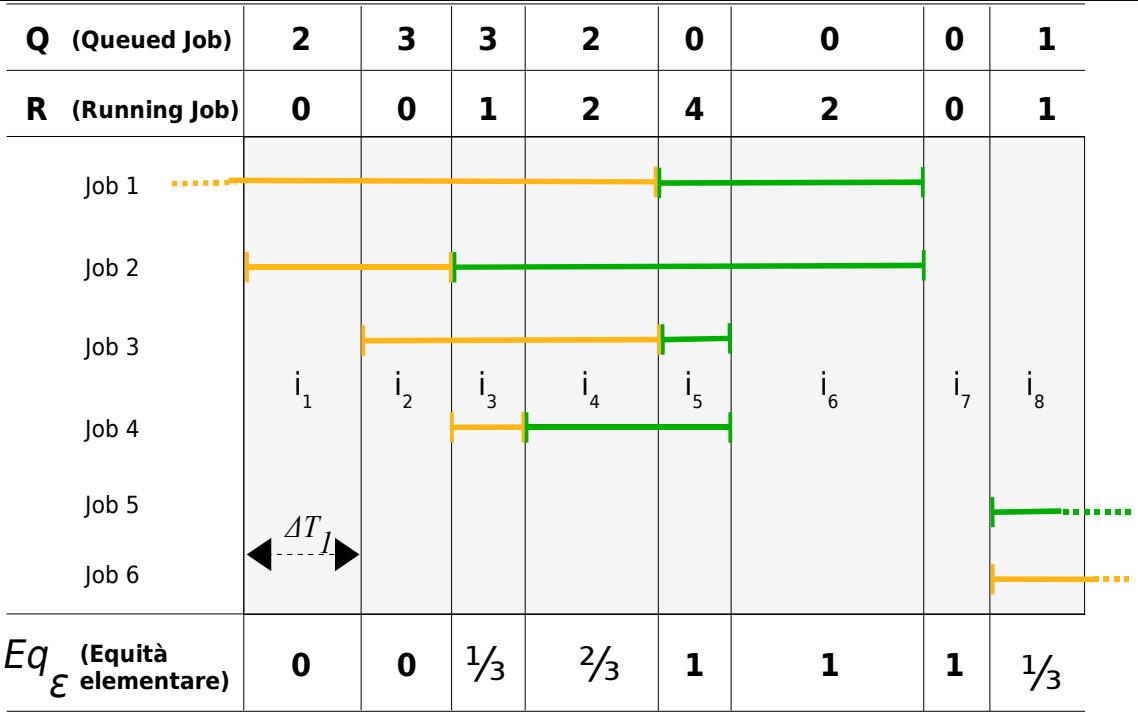
Equità Elementare

Frazione di risorse utilizzate,
in un dato intervallo temporale,
da un utente/gruppo rispetto
a quelle minime garantite

Δt_i : Intervallo di sovrapposizione i-esimo

Metrica di equità

Ris (Risorse spettanti di diritto) = 3 Nodi di calcolo



Equità Elementare

Frazione di risorse utilizzate, in un dato intervallo temporale, da un gruppo / utente rispetto a quelle minime garantite

Δt_i : Intervallo di sovrapposizione i-esimo

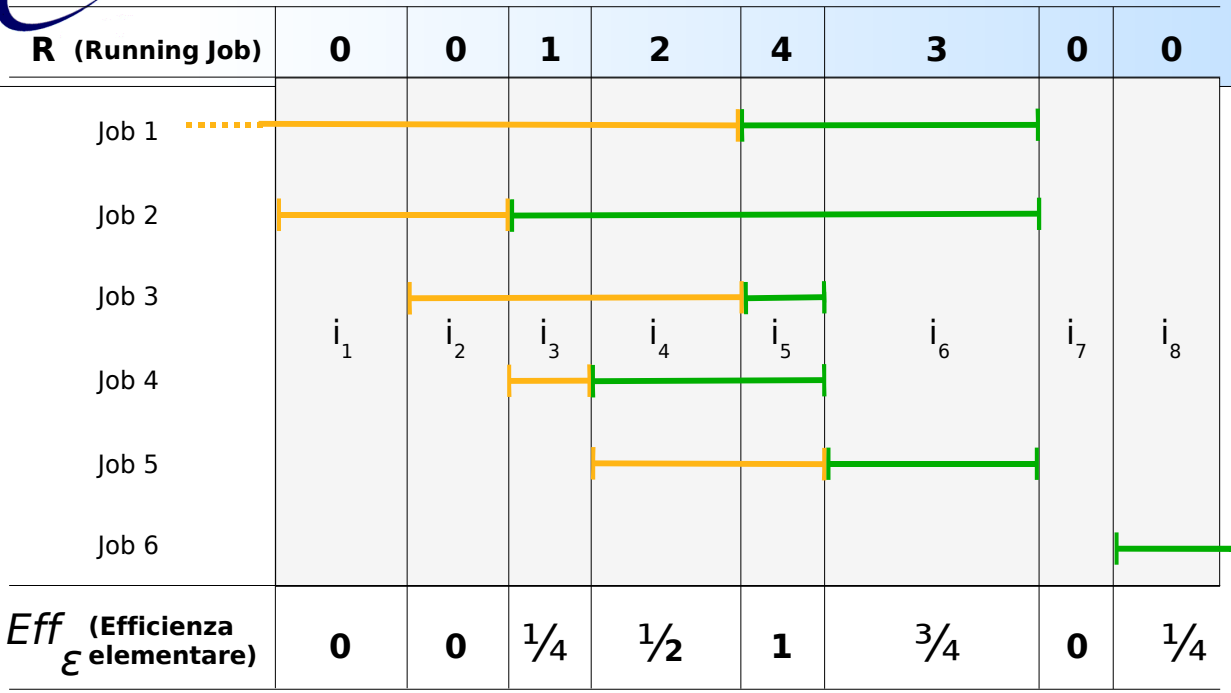
Equità Elementare

$$Eq_\varepsilon = 1 - \frac{min(Q, Ris - R)}{Ris}$$

Equità globale

$$Eq = \frac{\sum_{j=1}^{n} Eq_{\varepsilon j} * \Delta T_j}{\sum_{j=1}^{n} \Delta T_j}$$

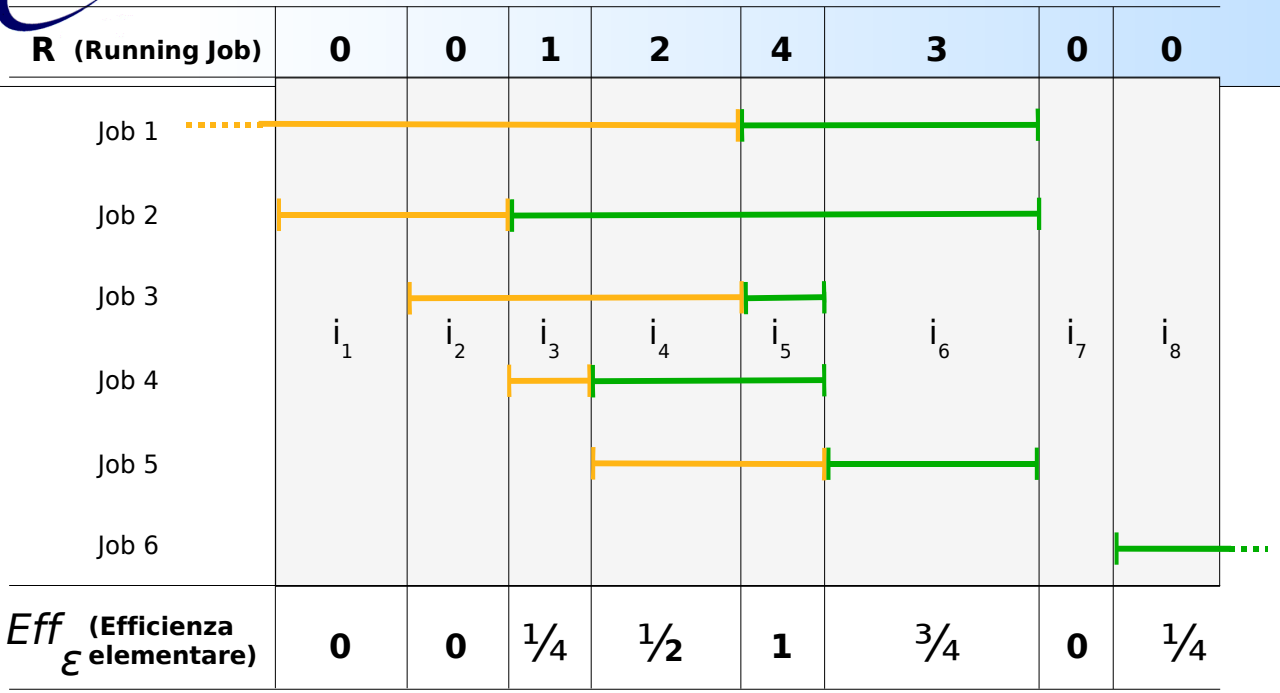
Metrica di efficienza



n : numero dei job considerati

RisSys : Risorse del sistema = 4 Nodi di calcolo

Metrica di efficienza



Efficienza Elementare

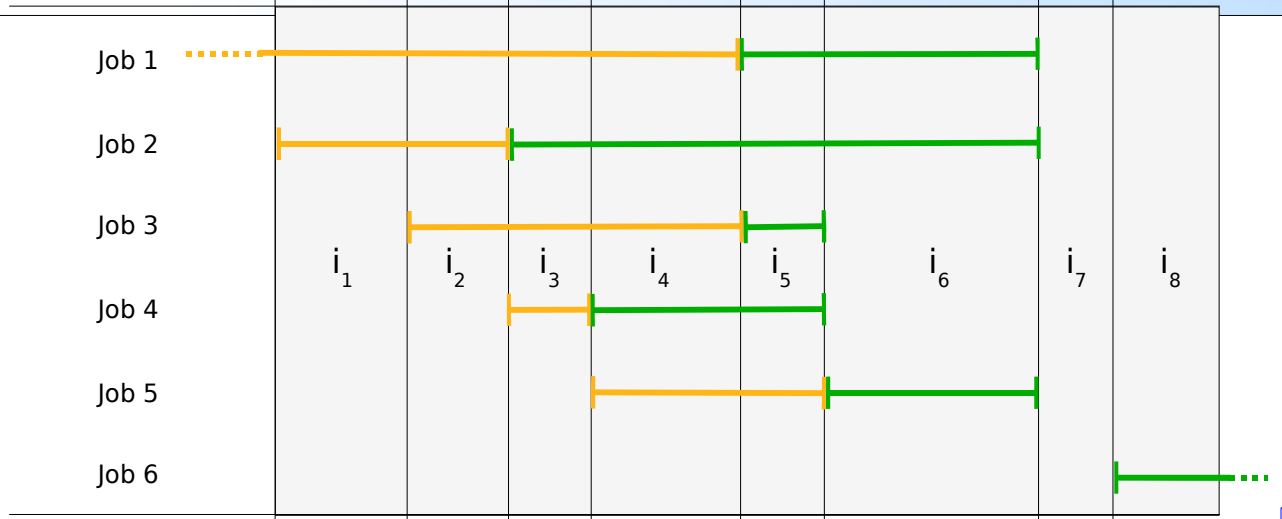
Misura della capacità del sistema di essere utilizzato efficientemente.
E' il rapporto tra il numero di job in esecuzione e le risorse disponibili

n : numero dei job considerati

RisSys : Risorse del sistema = 4 Nodi di calcolo

Metrica di efficienza

R (Running Job)	0	0	1	2	4	3	0	0
Eff ε (Efficienza elementare)	0	0	¼	½	1	¾	0	¼



Efficienza Elementare

Misura della capacità del sistema di essere utilizzato efficientemente.
E' il rapporto tra il numero di job in esecuzione e le risorse disponibili

n : numero dei job considerati

m : numero intervalli coincidenti l'attesa in coda di un dato job

RisSys : Risorse del sistema = 4 Nodi di calcolo

ΔQ_i : tempo in coda del job i-esimo

Efficienza Elementare del sistema in Δt_i

$$Eff_{\varepsilon i} = \frac{R}{RisSys}$$

Efficienza del sistema per l'attesa di un job

$$EffJob = 1 \qquad \Delta Qi=0$$

$$EffJob = \frac{\sum_{i=1}^{m} Eff_{\varepsilon i} * \Delta T_i}{\Delta Q_i} \qquad \Delta Qi>0$$

Efficienza globale del Sistema

$$Eff = \frac{\sum_{1=1}^{n} EffJob_i * \Delta Q_i}{\sum_{i=1}^{n} \Delta Q_i}$$

Conclusioni (parte 1)

- Siamo in grado di ***simulare in modo affidabile*** qualsiasi configurazione dello scheduler, tutte le operazioni richieste sono espletate dallo script *simulate.sh*.
- Siamo in grado di ***analizzare rapidamente*** il contenuto di un file di traccia alla ricerca di informazioni, lo script *workload-analyzer.pl* e' stato creato a questo scopo.
- Abbiamo definito un ***set di metriche*** che ci danno la bonta' delle nostre configurazioni.
- Abbiamo un ***tool di visualizzazione***.
- Siamo in grado di eseguire tutte queste operazioni tramite un ***repository di tracce centrale***.

Farm Virtuali

- Grazie all'utilizzo di ***tecnologie di virtualizzazione*** (Xen) e delle metodologie tecniche sviluppate in Sezione, siamo in grado di creare vere e proprie farm virtuali.

- Tali infrastrutture rappresentano un ambiente semplificato e controllato, all'interno del quale eseguire ***job di test particolarmente semplici***, idonei alla nostra indagine.

- I job in questione sono dei semplici ***"sleep"*** di durata opportuna, la cui sottomissione alla farm virtuale è stata automatizzata per mezzo del tool software dedicato:

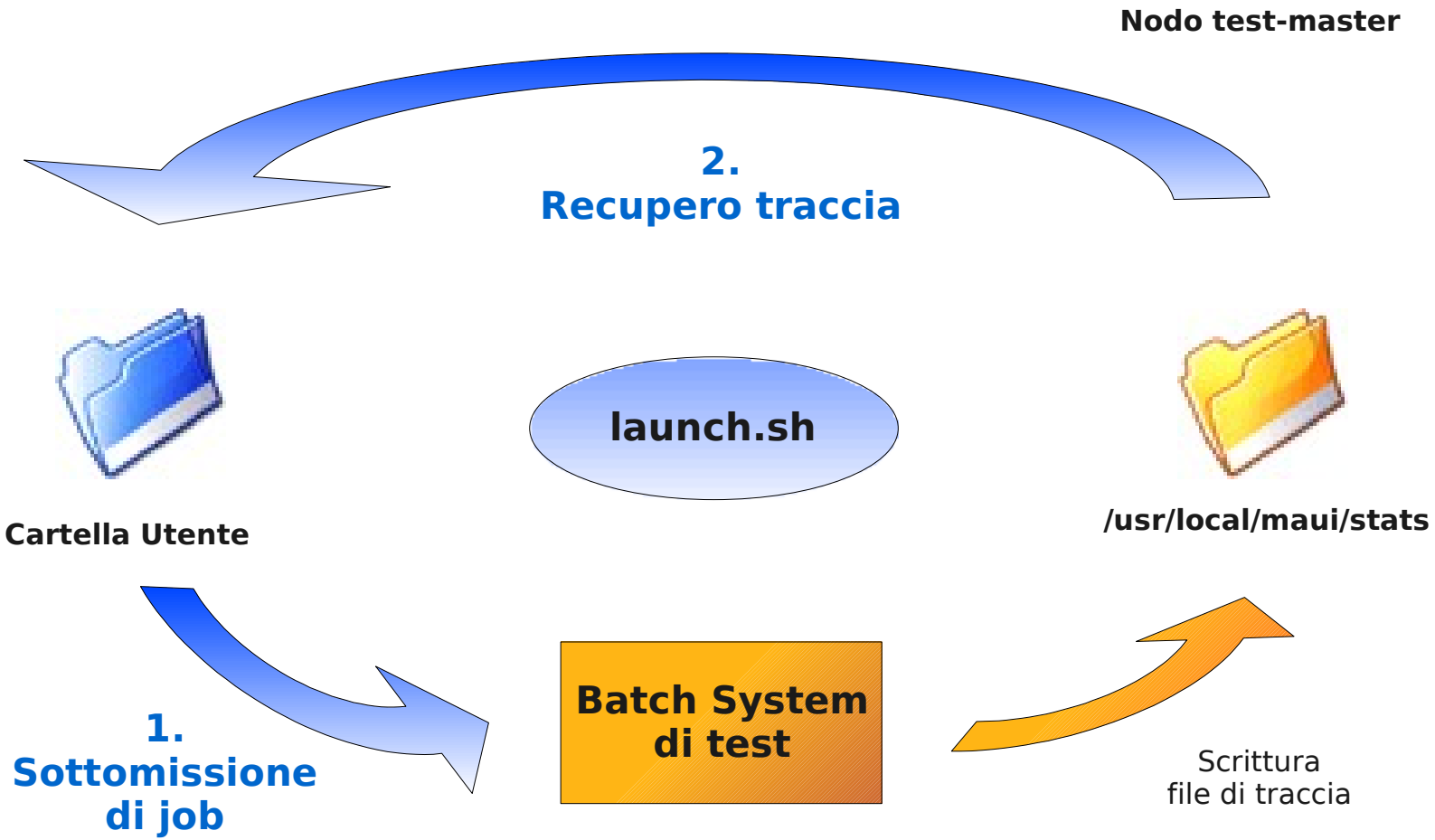
"launch.sh"

Wrapper di lancio dei job: "launch.sh" (1)

- Analogamente a "simulate.sh", questo wrapper opera a partire da una directory standard

- Le azioni eseguite sono le seguenti:
  - Legge un file contenente una sequenza di lancio (utenti, code, durata sleep job, ecc.)
  - Riconfigura utenti e gruppi su tutte le macchine
  - Riconfigura il batch system
  - Sottomette i job
  - Recupera il file di traccia prodotto e lo colloca nella directory standard

Wrapper di lancio dei job: "launch.sh" (2)

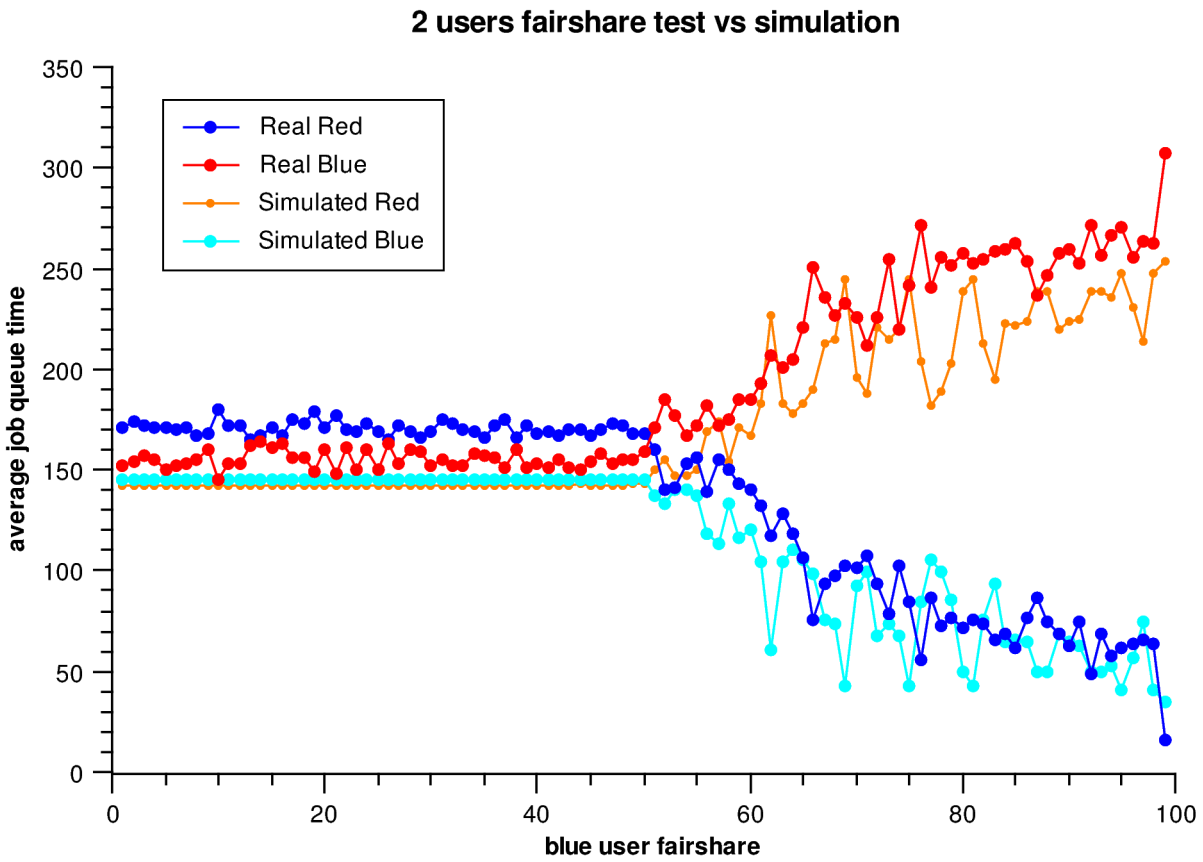


Conclusioni (parte 2)

- Possiamo eseguire dei lanci di job in maniera automatizzata e controllata.
- Possiamo verificare direttamente le differenze tra una simulazione ed un lancio reale.
- Tramite le metriche di valutazione, siamo in grado di valutare quantitativamente queste differenze.
- Utilizzando la metrica di efficienza, possiamo stimare l'overhead di una farm reale rispetto alla stessa simulata.

Overhead reale - simulato

Il confronto di ***efficienza*** farm reale / farm simulata, può dare una misura del corretto funzionamento della farm reale.



Modifica automatica delle configurazioni

Il passo successivo e' stato quello di realizzare un sistema per automatizzare le sequenze di piu' simulazioni o lanci reali.

In maniera simile a quanto fatto per *simulate.sh* e *launch.sh* abbiamo realizzato lo script *run.sh* che interagendo con una directory (eventualmente sul repository) prepara delle sottodirectory per *simulate.sh* e *launch.sh*

Il file principale che determina il comportamento di un run (sequenza) e' *run.conf* mentre la configurazione dello scheduler e' passata tramite un template chiamato *config-simul-template*

config-simul-template

Si tratta di un file di configurazione dello scheduler, modificato per supportare variabili.

- Le variabili sono specificate con un formalismo del tipo: <<[variabile]>>

- La risoluzione delle variabili viene fatta da *run.sh* utilizzando le impostazioni presenti in *run.conf*

```
RMPOLLINTERVAL          00:00:01

LOGFILE                 maui.log
LOGFILEMAXSIZE          10000000
LOGLEVEL                0

QUEUETIMEWEIGHT         1

BACKFILLPOLICY          FIRSTFIT
RESERVATIONPOLICY       CURRENTHIGHEST

NODEALLOCATIONPOLICY    MINRESOURCE

CREDWEIGHT 1
USERWEIGHT 1

USERCFG[Group_01_User_01] PRIORITY=50
USERCFG[Group_01_User_02] PRIORITY=<<prio1>>
USERCFG[Group_01_User_03] PRIORITY=<<prio2>>
USERCFG[Group_01_User_04] PRIORITY=<<prio3>>
USERCFG[Group_01_User_05] PRIORITY=<<prio4>>
USERCFG[Group_01_User_06] PRIORITY=<<prio5>>
USERCFG[Group_01_User_07] PRIORITY=<<prio6>>
USERCFG[Group_01_User_08] PRIORITY=<<prio7>>
```

run.conf

Ogni riga del file *run.conf* contiene una coppia chiave-valore, le chiavi piu' significative sono:

```
RUNTYPE=genetic
RUNFEATURES=parallel
MAXPROC=5
MAXWORLDS=5
GENERATIONS=20
POPULATION=100
MUTATIONRATE=10
CROSSOVERRATE=70
SHIFTRATE=5
GENOTYPE=prio1:prio2:prio3:prio4:prio5:prio6:prio7
FITNESS=fitness.pl
RUNPREFIX=run-
VARINTERVAL_prio1=1..100
VARINTERVAL_prio2=1..100
VARINTERVAL_prio3=1..100
VARINTERVAL_prio4=1..100
VARINTERVAL_prio5=1..100
VARINTERVAL_prio6=1..100
VARINTERVAL_prio7=1..100
```

- **RUNTYPE:** determina il tipo di run da eseguire.
- **GENOTYPE:** e' la lista delle variabili da risolvere
- **VARENUM:** specifica una lista di possibilita' per una data variabile
- **VARINTERVAL:** specifica un intervallo per una variabile

Sequenza enumerativa

Un primo tipo di sequenza (run) e' quella ***enumerativa***, in cui tutte le possibili permutazioni delle configurazioni vengono lanciate/simulate.

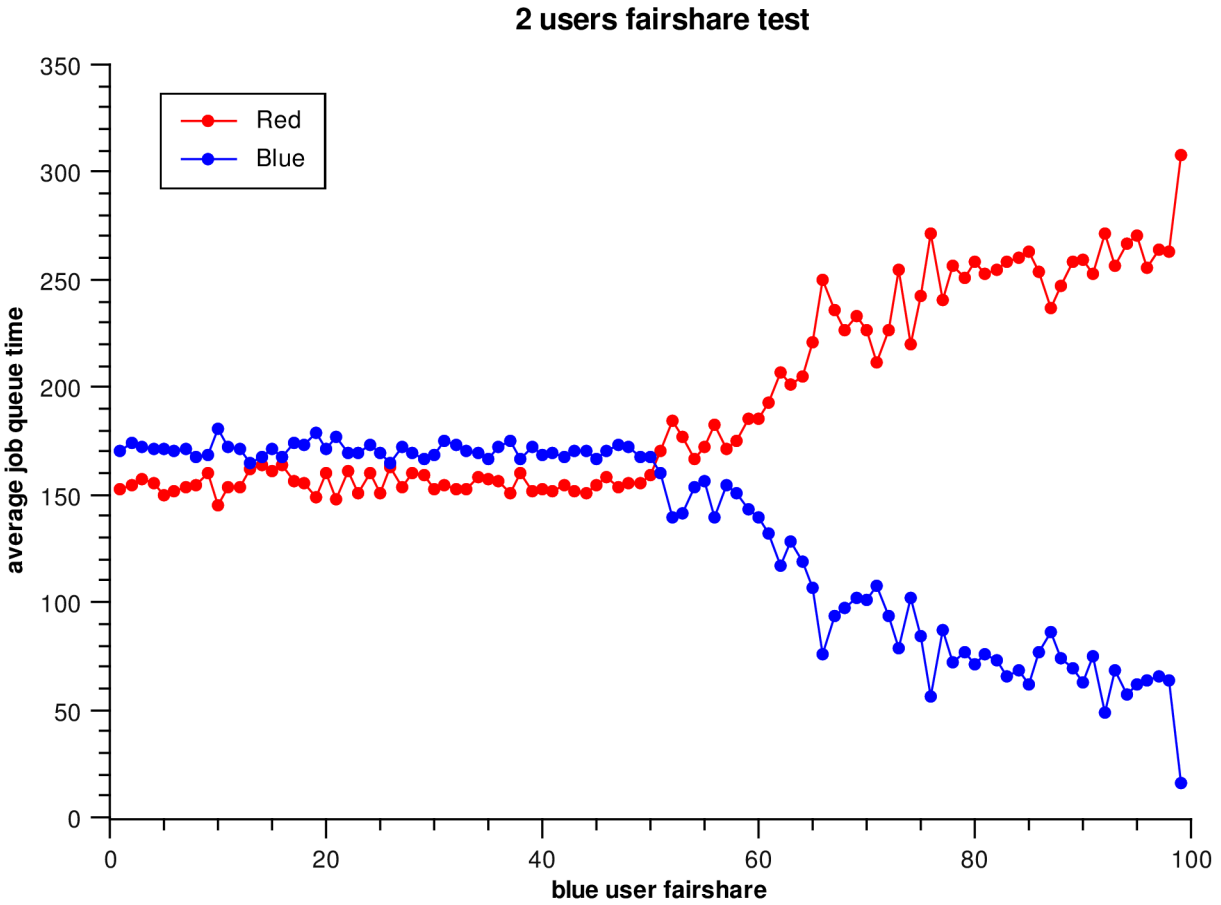
- Tutte le variabili presenti nel template vengono risolte per tutti i possibili valori del range indicato.
- Per ogni permutazione di configurazione viene generata una directory standard
- “launch.sh” e/o “simulate.sh” vengono lanciati su ognuna di queste directory

Esempio di “launch” enumerativo

Un test sul fairshare con le combinazioni di tutti i possibili valori delle variabili scelte:

- 4 nodi
- Due utenti, il primo con FSTARGET=50, il secondo con FSTARGET variabile
- Stesso schedule per i due utenti, job della durata di 30s lanciati ogni 10s

Test enumerativo

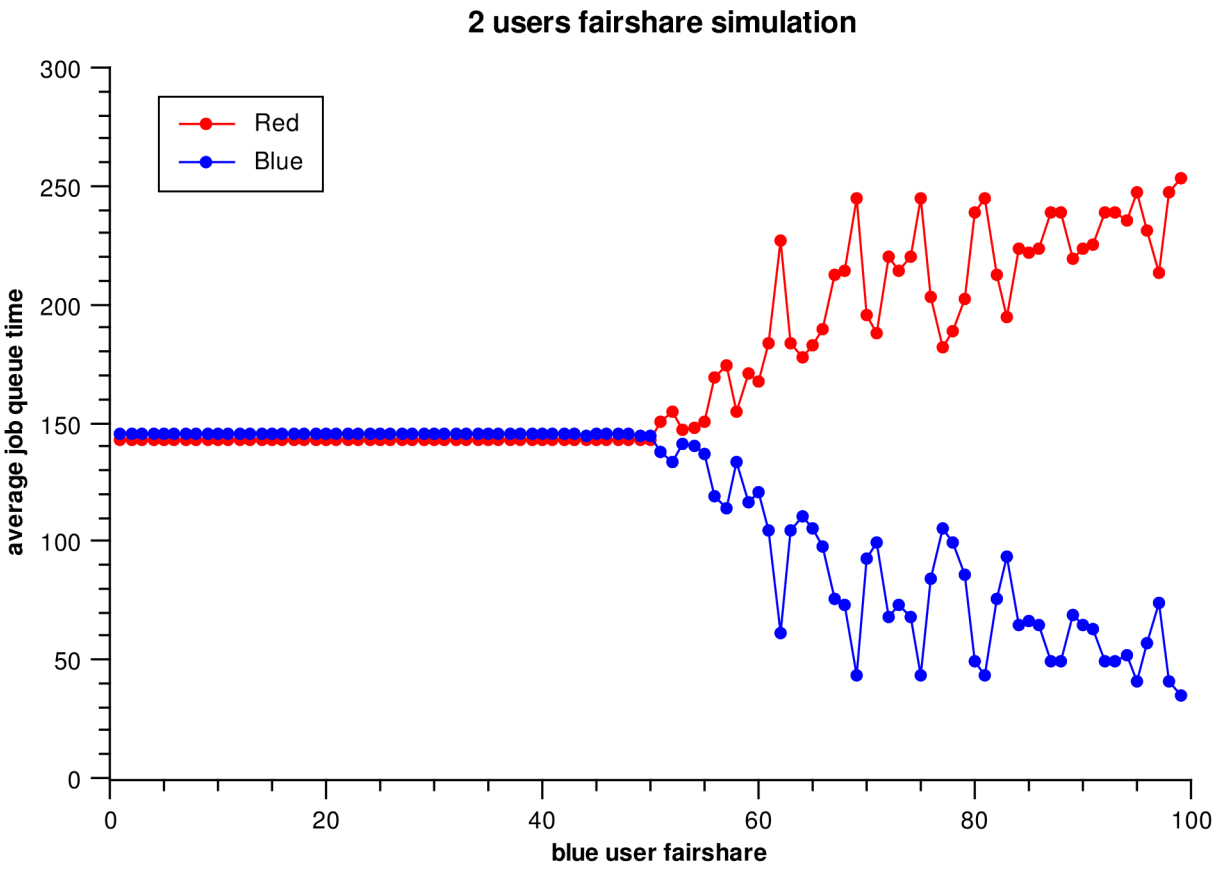


Esempio di simulazione enumerativa

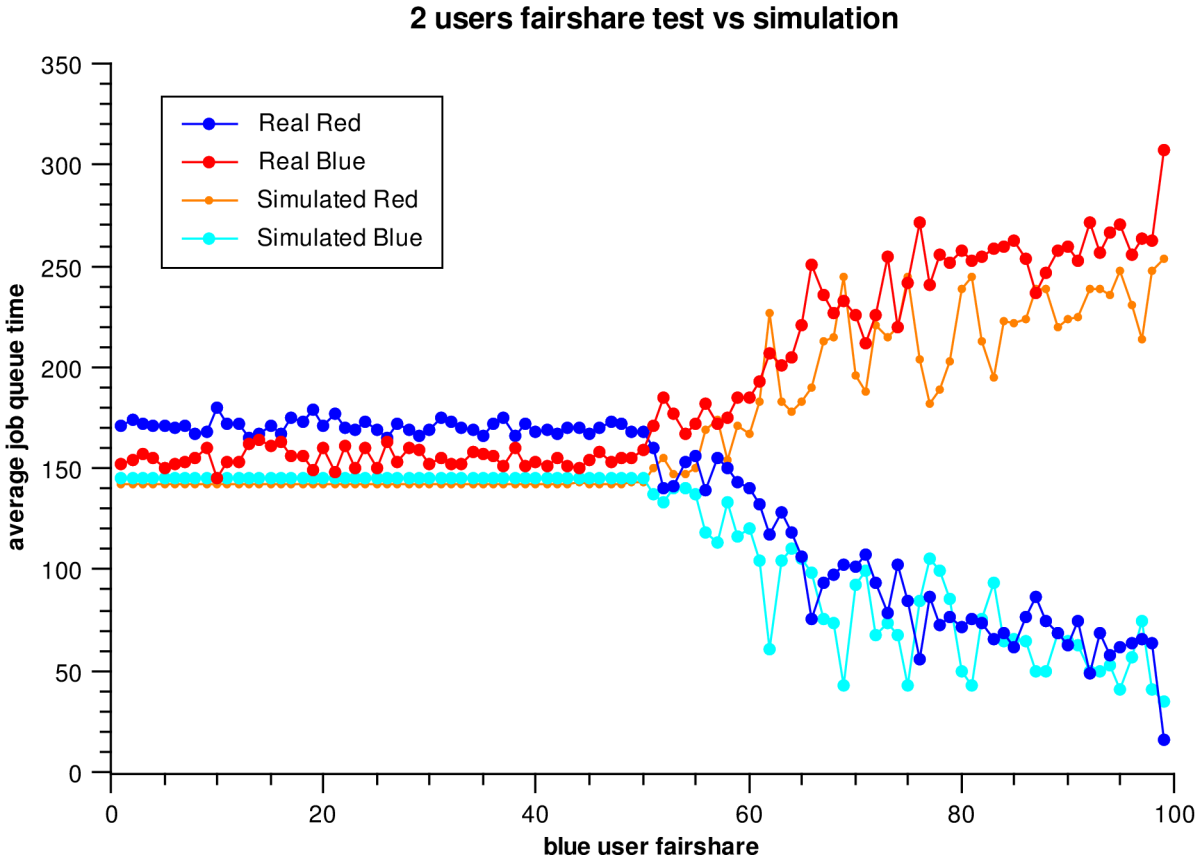
Una simulazione del test precedente, stessi parametri di configurazione

- 4 nodi
- Due utenti, il primo con FSTARGET=50, il secondo con FSTARGET variabile
- Stesso schedule per i due utenti, job della durata di 30s lanciati ogni 10s
- SIMULATO

Simulazione enumerativa



Confronto Simulazione



Sequenza montecarlo

Un perfezionamento della validazione del simulatore di Maui può essere conseguito tramite una sequenza di tipo "Montecarlo":

- Vengono generate delle schedule di lancio casuali.
- Ogni schedule viene lanciata e successivamente simulata.
- Le tracce reali e simulate sono quindi misurate per mezzo di metriche adeguate.
- Il confronto dei risultati quantifica una stima dell'errore introdotto dal simulatore.

Sequenza genetica

Per mezzo di un algoritmo genetico, vengono fatte evolvere popolazioni di configurazioni, selezionate poi sulla base di una funzione di fitness specificata.

- Per ogni configurazione sono generati diversi genotipi, che rappresentano la popolazione iniziale.
- Viene valutata la fitness dei genotipi.
- Sulla base della funzione di fitness, i genotipi sono selezionati, mutati, incrociati, soppressi per generare una nuova popolazione.
- L'iterazione del processo seleziona configurazioni via via più adeguate.

Test genetico

Per testare l'algoritmo genetico e l'infrastruttura, si è deciso di sottomettere al sistema un semplice problema di ottimizzazione, per il quale è nota la configurazione ottimale.

- Ambiente di 4 nodi, 1 processore per nodo
- 8 utenti, tutti con il medesimo schedule
- Priorità 1° utente = 50
- Priorità altri utenti: libere, da evolvere

Funzione di fitness

L'obbiettivo è minimizzare la differenza tra le medie dei tempi in coda dei processi degli utenti

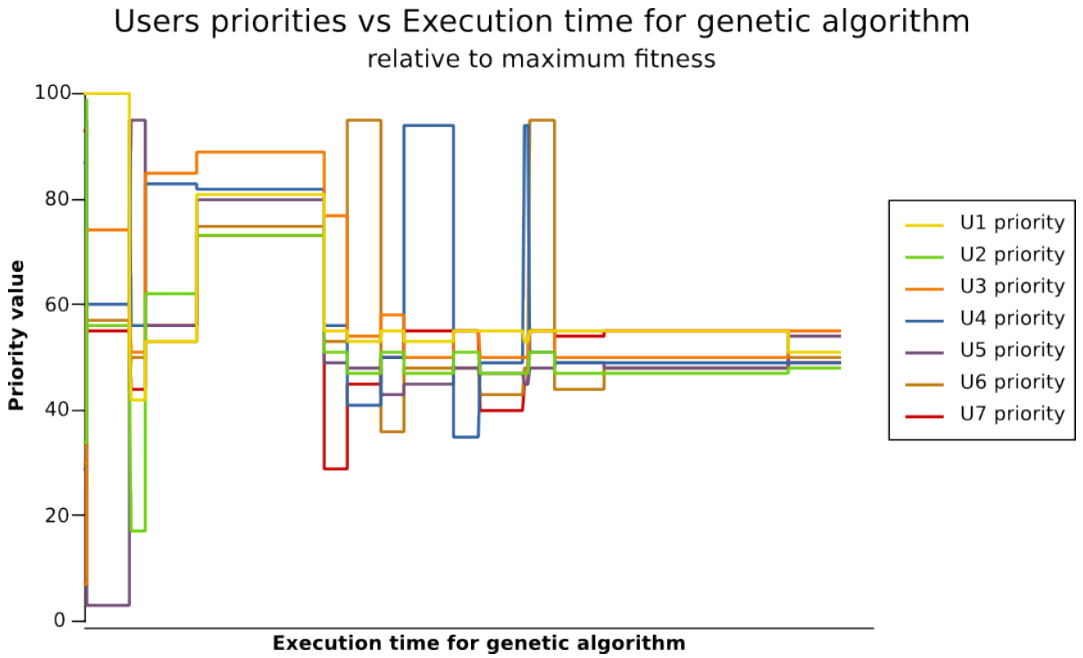
n	number of jobs of user i
Q_{ij}	time in queue for job j of user i
u	number of users
n, m	indexes of users

$$\overline{Q}_i = \frac{\sum_{j=1}^{n} Q_{ij}}{n}$$ average queue time of process for user i

$$Fitness = 1 - \sum_{n=1}^{u} \sum_{m=1}^{u} \frac{\left|\overline{Q}_n - \overline{Q}_m\right|}{2}$$ fitness function

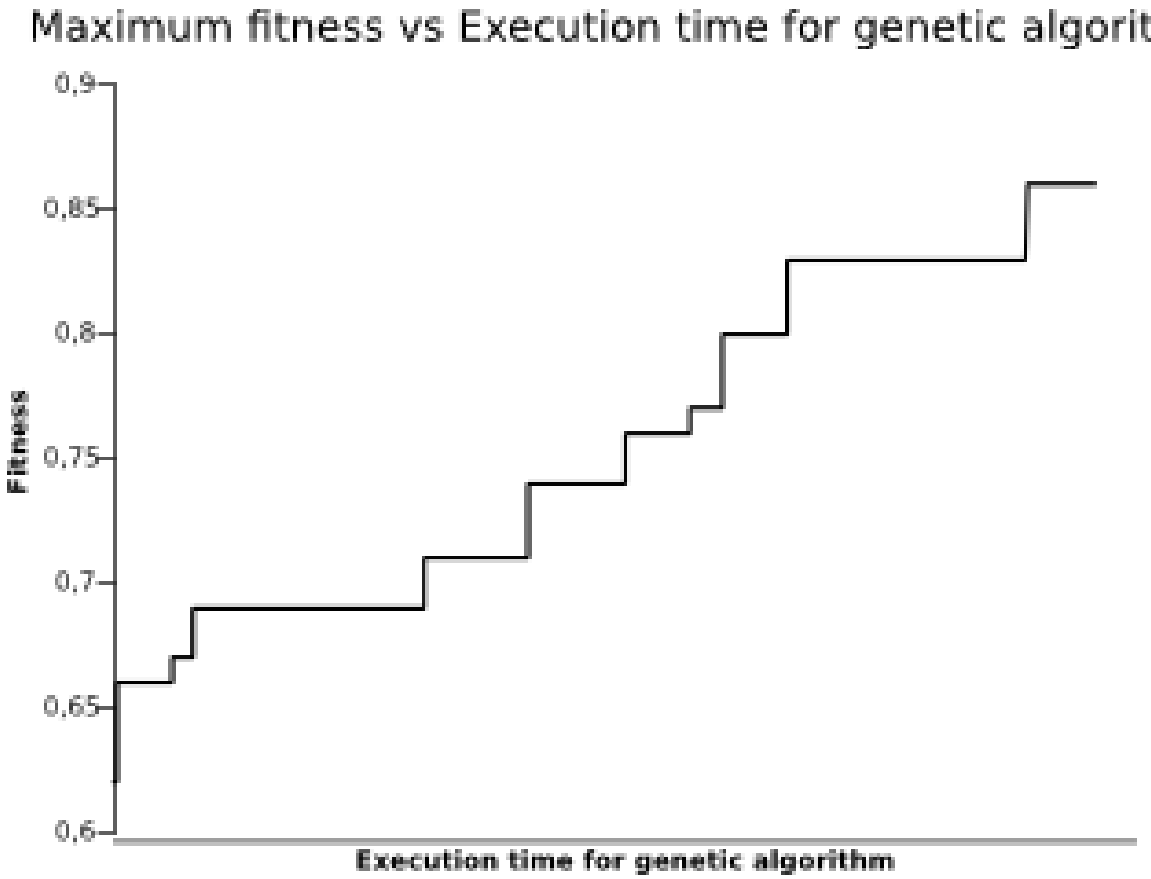
Soluzione prevista: priorità di tutti gli utenti = 50

Dopo 20 generazioni da 100 individui...



Le configurazioni di massima fitness trovate si avvicinano, come ci si attendeva, all'optimum teorico.

Dopo 20 generazioni da 100 individui...



Il valore di fitness delle configurazioni “migliori” si avvicina ad 1 come atteso.

Analisi delle sequenze: "run-analyzer.pl"

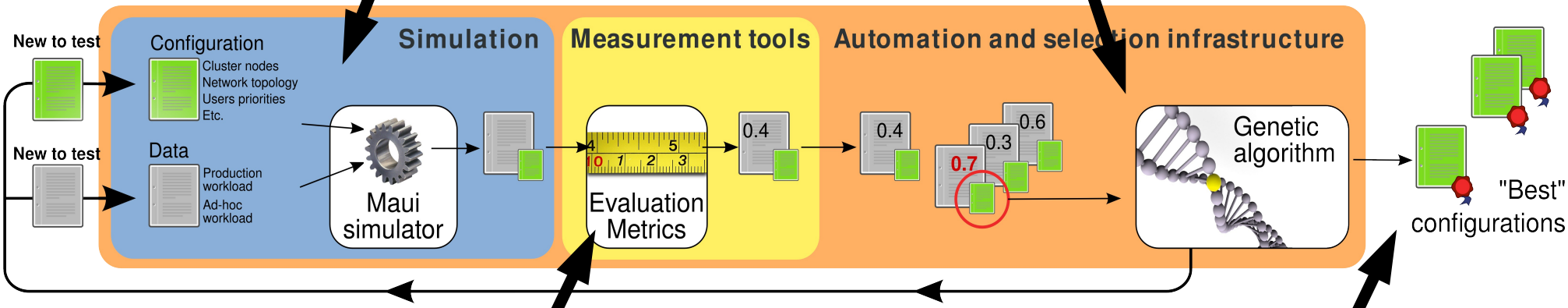
E' questo un tool di analisi per l'estrazione di dati e statistiche dai **"Runs"** di tracce.

- Richiama workload-analyzer per l'estrazione dei dati dalle singole tracce di test o simulazione
- Permette il confronto, la selezione e il calcolo di statistiche aggregate

L'infrastruttura allestita in sintesi...

Modulo di automazione della singola simulazione

Infrastruttura di automazione e modulo di selezione genetica



Modulo di misurazione e dell'ouput di simulazione

Cofigurazioni di Maui selezionate

Conclusioni e sviluppi futuri

- Abbiamo realizzato un'infrastruttura completa per la simulazione su larga scala di policy di scheduling.
- Intendiamo utilizzare tale infrastruttura per individuare specifiche configurazioni ottimali per lo scheduler Maui della nostra farm.
- Vogliamo inoltre indagare sugli aspetti relativi all'analisi delle discrepanze tra farm reale e farm simulata, che sembrano ben prestarsi alla rilevazione di problematiche inrastrutturali.

Links

- http://www.mirkomariotti.it/projects/projects/010-maui/index.html
- http://grid.pg.infn.it
- http://www.clusterresources.com/

Contatti

- francesco.cantini@pg.infn.it
- mirko.mariotti@fisica.unipg.it
- leonello.servoli@pg.infn.it
- claudio@tanci.it