Dettagli utilizzo cluster Intel

Maurizio Cremonesi, Claudio Arlandini
CILEA, Segrate

Abstract
Dall'inizio dell'anno 2004 il CILEA ospita e gestisce un cluster di processori Intel a 32 bit denominato *avogadro*. Esso è costituito da 128 nodi biprocessore Intel Xeon 3.06 GHz. L'articolo intende presentare le problematiche relative all'utilizzo della macchina fornendo le istruzioni operative necessarie all'uso più efficace.

Keywords: Software, Supercalcolo, Calcolo Parallelo.

Il cluster *avogadro* [1] è attualmente l'elaboratore più prestante installato al CILEA ed ha fatto misurare una prestazione Linpack di 1,084 TeraFlops che lo ha posizionato fino al novembre 2004 tra i primi 250 computer più potenti del mondo.

L'installazione di questo elaboratore è frutto della collaborazione tra il CILEA, le Università degli Studi di Milano e di Milano Bicocca e il Politecnico di Milano, che hanno contribuito fattivamente all'acquisto della macchina e a cui è riservato primariamente l'utilizzo.

L'architettura di questa macchina, pur non essendo del tutto nuova per il CILEA è tuttavia abbastanza diversa da quella a cui i nostri utenti da qualche anno in qua sono abituati. Il CILEA già una decina d'anni fa aveva ospitato un cluster di workstation, costituito da 8 processori RISC HP 9000, che affiancavano i due processori vettoriali del Convex C3820. In seguito l'architettura cluster era stata abbandonata per finalizzare le risorse all'acquisto di server multiprocessori a memoria condivisa (SMP), più adatti a garantire le risorse di calcolo e di memoria richieste dall'utenza accademica e privata.

Per semplicità si trascura il fatto che l'insieme dei server multiprocessori HP attualmente installati al CILEA costituisce un vero e proprio cluster, anche se non è frequente il suo utilizzo come un'unica entità di elaborazione.

Negli ultimi anni tuttavia il software applicativo è stato sviluppato anche e forse soprattutto in direzione dello sfruttamento dei processori Intel e AMD sia a 32 che a 64 bit, processori decisamente più economici dei calcolatori RISC e vettoriali.

Questo spostamento di interesse degli sviluppatori degli applicativi è motivato sia dal desiderio delle aziende produttrici del software di sfruttare le elevate prestazioni raggiunte dai processori che animano gli ormai diffusissimi Personal Computer sia dal consolidamento degli algoritmi e delle tecniche di parallelizzazione.

Per il profano i cluster sono semplicemente un modo più economico per ottenere risorse di calcolo elevate, ma all'esperto non sfuggono le difficoltà di gestione e di programmazione di un elaboratore multiprocessore a memoria distribuita.

Le caratteristiche hardware e software dei calcolatori SMP a 64 bit facilitano notevolmente il lavoro del programmatore, che può prendersi tutta la memoria che desidera in un unico blocco e non deve preoccuparsi troppo del bilanciamento del carico, in quanto ogni processo nuovo viene assegnato automaticamente al processore più libero.

A livello utente distribuire il carico su un server SMP è facile: basta sottomettere i lavori ed il sistema operativo fa il resto, impegnando a turno tutti i processori, indifferentemente. Questo garantisce una buona distribuzione del carico su tutta la macchina qualunque sia il numero dei processi in esecuzione. Tutt'altra cosa è la distribuzione del carico in un cluster, dove i processori da impegnare devono essere scelti individualmente. Per questo è necessario uno strumento esplicito di equilibramento del carico, unito a meccanismi di controllo.

Perché il cluster *avogadro* possa funzionare come un unico, potente elaboratore è necessario che ogni nodo permetta l'esecuzione di qualun-

que processo provenga da qualsiasi altro nodo del cluster, vale a dire che ci sarà un nodo particolare aperto al mondo che si preoccupa di applicare i controlli di autorizzazione più stringenti per garantire che solo gli individui autorizzati possano accedere al cluster, dopo di che l'utente può impegnare qualunque nodo desidera. Ma qui la fretta, l'abitudine, l'inerzia possono far preferire alcuni nodi ad altri, magari sempre gli stessi per più utenti, saturandone pochi e lasciandone molti del tutto scarichi.

A questo punto il gestore della macchina, preoccupato di sfruttare al meglio le risorse affidategli, non può lasciare alla libera iniziativa dell'utente la scelta dei nodi da utilizzare.

Fortunatamente esistono strumenti software che permettono di controllare costantemente il carico di ogni nodo del cluster e possono indirizzare su quelli più scarichi le nuove richieste di calcolo.

Il CILEA ha scelto di installare sul cluster *avogadro* il sistema OpenPBS. Questi interagisce con l'utente come un sistema batch che, se correttamente utilizzato, garantisce ad ognuno l'uso in esclusiva delle risorse richieste per il tempo necessario.

Infatti, qualora un utente abbia bisogno di un certo numero di processori, il sistema OpenPBS verifica che le risorse richieste siano effettivamente disponibili. In caso contrario la richiesta viene fatta attendere. Quando infine il lavoro può iniziare, tutti gli altri lavori vengono esclusi dall'utilizzo delle risorse già impegnate. In questo modo, dopo una fase di attesa dipendente dal carico della macchina, il lavoro può procedere senza interferenze.

Operativamente, quando l'utente accede al cluster *avogadro*, lavora interattivamente dal proprio direttorio della partizione */home* del nodo *node001*.

Per garantire un livello di sicurezza adeguato si richiede che l'utente si colleghi ad *avogadro* con un protocollo di accesso SSH vers. 2. Per questo sono disponibili diversi programmi, installabili sia in ambiente Windows che Linux [3].

Per il buon funzionamento del cluster, il gestore non tollera che si lancino lavori pesanti sul nodo *node001* di *avogadro*, dedicato ad operazioni interattive leggere quali lo scaricamento o caricamento dei file e l'editing.

L'esecuzione di qualsiasi altro lavoro sul cluster deve quindi avvenire via batch, tramite il sistema OpenPBS.

Per facilitare l'operazione il CILEA ha preparato ove possibile appositi comandi per l'esecuzione degli applicativi, contenuti nella directory `/home/mcae/script/bin`.

In Tabella 1 sono elencati gli applicativi finora installati e le loro principali caratteristiche.

Gli utenti sono invitati a segnalare agli esperti citati in fondo all'articolo i riferimenti degli applicativi non ancora installati ma che vorrebbero utilizzare su *avogadro*, se possono interessare anche altri.

Per quanto riguarda i programmi proprietari, i programmi in Tabella 2 o quelli di uso individuale, converrà che l'utente di *avogadro* si attenga alle modalità descritte in questo articolo.

applicativo	***tipo***	***comando***
ABAQUS 6.4	usabile monoproc.	abaqus
AMBER 7	parallelo	amber7
FLUENT 6.1.22	parallelo	fluent
GAMESS dic2003	parallelo	gamess
GAMESS_NBO dic2003	monoproc.	gamess_nbo
NASTRAN 2001	parallelo	nastran
NAMD 2	parallelo	namd2
OCTOPUS 1.4	parallelo	octopus
RADIOSS 4.1	parallelo	radioss41

Tab. 1 – Applicativi principali installati su avogadro per i quali è disponibile apposito comando di sottomissione

Ad esclusione quindi dei comandi messi a disposizione dal CILEA, l'attivazione di un'esecuzione comincia con la preparazione di uno script di lancio, da sottomettere al sistema batch.

Tutti i lavori dovrebbero essere sottomessi da un direttorio dedicato contenente solo i file necessari all'esecuzione. In questo direttorio si può generare lo script che realizza l'esecuzione, guidata dal sistema OpenPBS.

applicativo	***tipo***
GROMACS 3.2.1	parallelo
AUTODOCK 3.0.5	monoproc
TINKER 4.1	monoproc
FLUKA 2003	monoproc
CPMD 3.9.1	parallelo
OCTOPUS 1.4	parallelo

Tab. 2 – Applicativi installati su avogadro senza un comando di sottomissione pre-definito

È importante ricordare che lo script attivato via batch comincia l'esecuzione dal direttorio $HOME del nodo designato, perciò occorre aver

cura di posizionarvisi sempre con un opportuno comando *cd*.

Il sistema batch comunica con gli applicativi mediante file e variabili d'ambiente. La Tabella 3 elenca queste variabili d'ambiente ed il loro significato. Queste informazioni sono comunque accessibili con il comando `man qsub`.

nome	*modo*	*significato*
PBS_O_HOST	*output*	nodo di esecuzione
PBS_O_QUEUE	*output*	coda batch di sottomissione
PBS_O_WORKDIR	*output*	il direttorio da cui è stato sottomesso il lavoro
PBS_ENVIRONMENT	*output*	vale *PBS_BATCH* per un lavoro batch, *PBS_INTERACTIVE* per un lavoro interattivo (sotto il controllo di OpenPBS)
PBS_JOBID	*output*	identificatore del lavoro
PBS_JOBNAME	*input*	nome mnemonico per riconoscere il lavoro in esecuzione
PBS_NODEFILE	*output*	nome del file contenente l'elenco dei nodi riservati al lavoro in esecuzione
PBS_QUEUE	*output*	nome della coda batch usata in esecuzione

Tab. 3 – Variabili d'ambiente proprie di OpenPBS

Quando l'esecuzione necessita di più nodi è necessario consultare il contenuto del file di nome $PBS_NODEFILE per conoscere i nodi che OpenPBS ha riservato al lavoro. Questo file tra l'altro può essere usato direttamente con MPI, come dall'esempio sotto illustrato.

Venendo quindi ad alcuni esempi pratici, ecco come potrebbe essere uno script che lancia un'applicativo monoprocessore.

```
#!/bin/csh
#PBS -N MioLav
#PBS -q max16
#PBS -l nodes=1:ppn=1
cd $PBS_O_WORKDIR
echo "Lavoro in esecuzione su $PBS_O_HOST"
echo "  dal direttorio $PBS_O_WORKDIR"
set path=( $HOME/bin \$path)
echo "L'esecuzione inizia `date`"
miocalcolo < input > output
wait
echo "L'esecuzione termina `date`"
exit
```

Le prime righe permettono di assegnare i valori più opportuni ad alcuni parametri di OpenPBS; in particolare, il batch avrà nome mnemonico "MioLav", sarà sottomesso alla coda max16 ed utilizzerà 1 processore.

Quest'altro invece è un esempio di script di lancio di un'applicazione multi-processore MPI:

```
#!/bin/csh
#PBS -N MioPar
#PBS -r n
#PBS -q max16
#PBS -j oe
#PBS -l nodes=4:ppn=2
umask 077
cd $PBS_O_WORKDIR
echo "Lavoro in esecuzione da $PBS_O_WORKDIR"
echo "  sui nodi:"
cat $PBS_NODEFILE
set path=( /opt/nmyri/bin $path)
echo "L'esecuzione inizia `date`"
mpirun -np 8 -machinefile $PBS_NODEFILE \
   $HOME/bin/calcpar < input > output
wait
echo "L'esecuzione termina `date`"
exit
```

In questo esempio l'applicazione calcpar utilizza 8 processori, impegnando 4 nodi del cluster *(ppn= processors per node)*. Il valore "oe" del parametro "-j" permette di unire *stdout* e *stderr* in un unico file. Si definisce il PATH in modo da utilizzare la versione di MPI implementata per la rete di interconnessione Myrinet, che garantisce le prestazioni più elevate su *avogadro*.

In entrambi i casi il job verrà sottomesso al sistema di code con il comando:

```
qsub script
```

Qualora fosse necessario per motivi di test o altro l'utilizzo interattivo di un programma di simulazione, questo non dovrà essere lanciato sul nodo di front-end, ma si potrà ottenere l'apertura di una shell su un nodo di calcolo mediante il sistema di code con il comando:

```
qsub -I -q short
```

In questo modo ci si garantisce l'uso esclusivo di una CPU, migliorando le prestazioni per il proprio codice e non togliendo risorse ad altri.

Sono attualmente definite le seguenti code:

- **short**: per lavori fino a 16 processori e cpu time minore di 1 ora (alta priorità)

- **max16**: per lavori fino a 16 processori e cpu time illimitata (bassa priorità)
- **max128**: per lavori da 17 a 128 processori e cpu time illimitata (alta priorità)
- **total**: per lavori richiedenti più di 128 processori (altissima priorità).

Attenzione: l'uso della coda **total** è subordinata all'accettazione da parte del sistemista e previa presentazione di dati di scalabilità che dimostrino l'efficienza del programma ad un numero di processori così elevato.

Per quanto riguarda l'utilizzo dei comandi di OpenPBS rimandiamo alle rispettive man pages. In breve, il comando per monitorare lo stato del proprio job è *qstat*, mentre per uccidere il job in esecuzione è necessario usare *qdel*.

Per quanto riguarda gli ambienti di sviluppo disponibili, sono presenti i compilatori elencati in Tabella 4:

Compilatore	*Vers*	*locazione*
GNU [C e F77]	3.2.2-5	/usr/bin
GNU F95	9/6/04	/usr/bin
Intel C	8.0	/home/mcae/intel_cc_80.066
Intel Fortran	8.0	/home/mcae/intel_fc_80.046
Intel debugger	7.3	/home/mcae/intel_idb_73

Tab. 4 – Compilatori disponibili

Mentre per quanto riguarda le librerie matematiche sono disponibili quelle riportate in Tabella 5:

Libreria	*Vers*	*locazione*
Intel MKL	7.0	/opt/intel/mkl70
PETSC	2.2.0	/home/mcae/petsc-2.2.0
FFTW	2.1.5	/home/mcae/fftw-2.1.5

Tab. 5 – Librerie matematiche disponibili

Per finire le librerie di comunicazione presenti sono elencate in Tabella 6:

Libreria	*Versione*	*locazione*
MPICH-GM	1.2.6..13	/opt/nmyri/
MPICH	1.2.6	/opt/mpich-1.2.6-ch_p4-intel
LAM-GM	7.0.4	/opt/lam-with-gm-7.0.4
LAM	7.0	/opt/lam-7.0

Tab. 6 – Librerie di comunicazione disponibili

Contatti utili

AMBIENTE DI SISTEMA:
- C. Arlandini – arlandini@cilea.it
- M. Pirola – pirola@cilea.it

SOFTWARE APPLICATIVO:
fluidodinamica
- R. Ponzini – ponzini@cilea.it
- P. Ramieri – ramieri@cilea.it

analisi strutturale
- S. Bozzini – bozzini@cilea.it
- M. Cremonesi – cremonesi@cilea.it

chimica
- M. Marchisio – marchisio@cilea.it

SVILUPPO SOFTWARE:
- M. Cremonesi – cremonesi@cilea.it
- P. Dagna – dagna@cilea.it
- P. Ramieri – ramieri@cilea.it

Bibliografia

[1] C. Arlandini, "*AVOGADRO: il CILEA oltre il muro del TeraFlops*", Bollettino del CILEA, n. 91 febbraio 2004, pagg. 4-7;

[2] Cluster Xeon – EXADRON - modalità di utilizzo e caratteristiche hardware:
URL:
http://www.cilea.it/redazione/risorse/hardware/Avogadro/avogadro.htm

[3] Risorse per collegamenti SSH:
URL:
http://www.chiark.greenend.org.uk/~sgtatham/putty/
URL: http://sshwindows.sourceforge.net/
URL: http://www.openssh.com/