ENEA GRID

CRESCO: Corso di introduzione

Autore: Alessandro Secco
alessandro.secco@nice-italy.com

Lezione 1

- Introduzione
- Architettura
- Connessione
- Lancio di job
- Riferimenti

Introduzione

Introduzione

- L'obiettivo di questo corso è permettere all'utente di ENEA Grid, di utilizzare l'ambiente di Cresco.
- Come sempre, è utile procurarsi la documentazione di LSF utente, client AFS e client Citrix; sono inoltre disponibili le dispense dei corsi precedenti riguardanti ENEA Grid.
- E' consigliato (ma non indispensabile) avere un'infarinatura di shell scripting, conoscenze del sistema Linux.

Architettura

Struttura di cresco

- I server del progetto cresco sono suddivisi in:
  - Server di frontend generici per le sezioni 1 e 2
  - Server di frontend speciali
    - Destinati ad attività particolari
  - Nodi di calcolo sezione 1
  - Nodi di calcolo sezione 2
- A parte i frontend speciali, tutte le altre macchine sono destinate ad utenti di calcolo

Caratteristiche di cresco

- Gli oltre 300 nodi di Cresco sono macchine di ultima generazione con sistema operativo Linux:
  - Architettura multi core fino a 48 core per nodo
    - Standard: 16 core su cresco1 e 8 core su cresco2
  - Memoria fino a 192 GB
    - Standard: 2 GB per core
- In aggiunta, Cresco aggiunge alcune funzionalità allo scopo di migliorare le prestazioni del calcolo parallelo:
  - Infiniband: è la rete di interconnessione tra i nodi, che permette di ottenere alte velocità di scambio dati tra i processi.
  - GPFS: un file system parallelo ad alte prestazioni

Servizi software

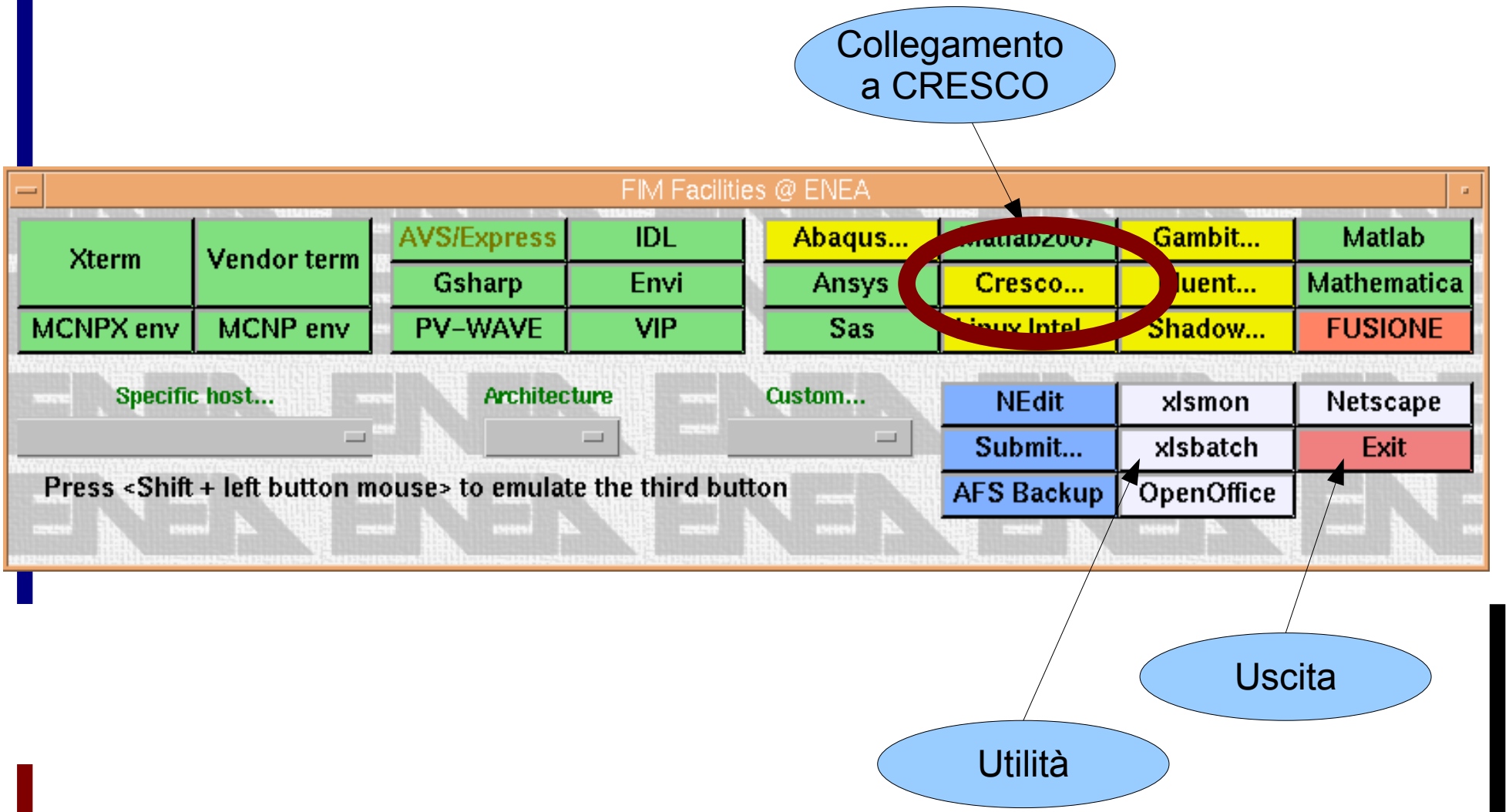
- Cresco è una macchina inserita nel contesto della grid di Enea.
- In particolare fornisce i servizi che gli utenti di ENEA Grid già conoscono:
  - AFS: storage distribuito
  - LSF: lancio di job
  - Citrix: accesso alle applicazioni
  - Applicativi vari
- Viene inoltre aggiunto il supporto a compilatori Portland, Intel e GNU, oltre a MVAPICH (MPICH per Infiniband) e OpenMPI.

Connessione

Connessione

- L'accesso a cresco è limitato al Client Citrix ed a ssh.
- Si consiglia di leggere il tutorial presente sul sito di Enea grid:
  - **http://www.eneagrid.enea.it**
- Frontend citrix di portici:
  - kleos.portici.enea.it
- Indirizzo di connessione via ssh:
  - cresco-fx.portici.enea.it

Interfaccia ENEA



Interfaccia CRESCO

1a: Scelta nodo

1b: Scelta architettura

2: Scelta compilatore (non obbligatoria)

3: Scelta mpi (non obbligatoria)

4: Apertura terminale

Lancio di job

LSF: riepilogo comandi più usati

- Ecco alcuni comandi essenziali per l'uso di LSF. I comandi hanno help e man pages.
  - lshosts: mostra le risorse statiche per host
  - lsload: risorse dinamiche per host
  - lsinfo: mostra l'elenco dei nomi di risorsa
  - bhosts: mostra il numero di job sui vari host
  - bqueues: informazioni sulle code
  - bsub: lancia un job
  - bjobs: controlla lo stato di un job
  - bkill: termina un job in esecuzione

Preparazione al lancio

- Quali code sono presenti in Cresco ?
  - Lanciare il comando bqueues
- Le code PRIORITARIE per i job paralleli e consigliate per lavorare in Cresco sono (in ordine di priorità):
  - cresco_512h2, cresco_512h8, cresco_512h24 per i job che usano più di 512 processori (2, 8, 24 ore, solo cresco2)
  - cresco_128h2, cresco_128h8, cresco_128h24 per i job che usano più di 128 ed al massimo 512 processori (2, 8, 24 ore)
  - cresco_16h2, cresco_16h8, cresco_16h24 per i job che usano più di 16 ed al massimo 128 processori (2, 8, 24 ore)
  - cresco_h2, cresco_h8, cresco_h24 per i job paralleli fino a 16 processori (2, 8, 24 ore)
- **IMPORTANTE**: I job paralleli DEVONO richiedere un numero di processori multipli di 8.

Preparazione al lancio

- Esistono anche code per i job seriali:
  - cresco_serh48 e cresco_serh72
- Queste 2 code sono meno prioritarie rispetto alle precedenti, e devono essere usate:
  - Specificando la durata del job (minore è la durata, maggiore la probabilità di girare)
  - Specificando la memoria da usare nel caso di job molto grandi
- Infine, esiste una coda interattiva, limitata a sessioni di 12 ore:
  - Interactive
- Si consiglia vivamente di migrare i propri job dalle code standard Enea a quelle apposite per cresco, per migliorare l'efficienza.

Preparazione al lancio

- Quali risorse sono disponibili ?
  - lshosts -R cresco
    - Questo comando visualizza la configurazione dei nodi di calcolo delle due sezioni e le risorse fornite
  - lsload -R cresco
    - Situazione attuale del carico delle macchine
  - bhosts -R cresco portici
    - Situazione dell'occupazione dei singoli slot (=processori) da parte dei job in esecuzione.

Risorse

- **cresco**
  - Indirizza tutti i nodi di calcolo
- **cresco1, cresco2**
  - Vanno sulle macchine della sezione 1 e 2
- **smp8, smp16**
  - Macchine con 8 e 16 core
- **frontend**
  - Sono le macchine di frontend → non usare !!!
- **cresco_amd, amd_3d, nvidia, fpga, cresco_cell**
  - Risorse speciali → non usare se non sapete cosa sono
- **infiniband**
  - Vale “up” se la rete infiniband sta funzionando
- **maxmem, ncpus**
  - Usare queste risorse per indirizzare la memoria e il numero di core

Lancio di un job parallelo

```
/* hello.c
*
* Simple "Hello World" program in MPI.
* NOTA: Programma di test preso in prestito dal sito del CERN
*/
#include "mpi.h"
#include <stdio.h>
int main(int argc, char *argv[]) {
int numprocs; /* Number of processors */
int procnum; /* Processor number */
/* Initialize MPI */
MPI_Init(&argc, &argv);
/* Find this processor number */
MPI_Comm_rank(MPI_COMM_WORLD, &procnum);
/* Find the number of processors */
MPI_Comm_size(MPI_COMM_WORLD, &numprocs);
printf ("Hello world! from processor %d out of %d\n", procnum, numprocs);
/* Shut down MPI */
MPI_Finalize();
return 0;
}
```

Lancio di un job parallelo: OPENMPI

- Compilazione:
  - mpicc hello.c
- Per lanciare il job, crearsi un semplice script di lancio:

```
#!/bin/sh

PROCS=`cat $LSB_DJOB_HOSTFILE | wc -l`

mpirun -n $PROCS --mca pls_rsh_agent "blaunch.sh" -hostfile $LSB_DJOB_HOSTFILE ./a.out
```

- Lo script deve
  - Contare i processori assegnati da LSF
  - Sostituire “ssh” con “blaunch.sh”

Lancio di un job parallelo: OPENMPI

- Infine, è necessario lanciare lo script da dentro LSF:
  - bsub -n 128 -q cresco_16h2 ./lancio.sh
- Si ricordano le opzioni di bsub più utili:
  - -o <filename>: ridirezione stdout (%J → jobid)
  - -e <filename>: ridirezione stderr (%J → jobid)
  - -W <minutes>: specifica la durata (minore della coda)
  - -R: specifica risorse aggiuntive

Lancio di un job parallelo: MVAPICH

- Al momento il lancio di job con MVAPICH non è ancora rilasciato e va concordato con gli amministratori di sistema.
- In sostanza l'integrazione sarà molto simile a quella per OpenMPI in quanto il comando /usr/bin/rsh verrà sostituito con il blaunch di LSF.

Job Pending

- I job possono rimanere pending per diversi motivi:
  - Mancano CPU
  - Mancano le risorse richieste
  - L'amministratore ha chiuso un certo numero di host per manutenzione
- Capire perché un job è pending, non è semplice all'interno di un cluster.
- Il job rimane pending se il numero di CPU richiesto è minore dell'intersezione tra l'insieme delle cpu che rispondono alla stringa di resource requirement del job (bsub -R) e l'insieme delle CPU selezionate dalla coda (bqueues -l).

Job Pending

- In generale, se un job è in pending per mancanza di risorse, LSF cerca di prevedere il tempo massimo di attesa in coda, all'interno del comando bjobs -l
  - Sun Sep 14 08:58:15: Job will start no sooner than indicated time stamp;
- Questo è comunque solo un'indicazione. Per effetto del gioco delle priorità tra code, il tempo può anche allungarsi.

Strategie per non finire in pending

1. Usare sempre le code corrette: le code di durata minore sono prioritarie rispetto a quelle di durata maggiore.
2. Se la durata del job è minore della durata della coda, specificarlo con bsub -W
3. Se si usano code di durata breve e con meno di 512 processori, si può passare davanti ai job più grandi in attesa, per un tempo limitato, senza penalizzazioni.
4. Non specificare le risorse con bsub -R se non strettamente necessario: le code selezionano già le risorse giuste.

Riferimenti

Riferimenti

- Home dell'utente AFS:
  - ~/ENEAGRID_info.txt
- Enea grid home page:
  - http://www.eneagrid.enea.it
- Documentazione di LSF:
  - /afs/enea.it/software/lsf/docs/lsf/7.0.2/index.html
- Documentazione di AFS:
  - http://www.openafs.org/doc/index.htm
- Documentazione di GPFS:
  - http://www.afs.enea.it/project/eneagrid/pfs/
- Documentazione di CRESCO:
  - http://www.afs.enea.it/project/eneagrid/Resources/CRESCO_documents

Riferimenti

- http://gridticket.enea.it: permette di aprire “ticket” di supporto con gli amministratori. E' il modo consigliato per segnalare problemi inerenti ENEA grid.
- griduser@enea.it: è possibile contattare gli altri utenti per condividere esperienze sull'uso della GRID.
- alessandro.secco@nice-italy.com: problemi relativi a questo corso possono essere indirizzati a me. La collaborazioni di tutti è gradita.