Utilisation d'un cluster de calcul openSSI

Centre d'Océanologie de Marseille
UMS 2196 CNRS

JoSy Sept.07 – Lyon

Maurice.Libes@com.univmed.fr

Sommaire

- Présentation du laboratoire
- La modélisation au C.O.M
- Présentation de openSSI
  - Fonctionnalités principales
  - Installation du cluster (noeud maître et esclaves)
  - Administration du cluster (commandes et graphique)
  - Avantages et inconvénients

Présentation du laboratoire

- Centre d'Océanologie de Marseille (C.O.M)
  - Observatoire des Sciences de l'Univers (O.S.U) et UFR de l'Université de la Méditerranée
    - Une unité de services : UMS 2196
    - Au service de 3 Unités de Recherches
    - #250 personnes sur 2 sites
    - 1 équipe de modélisation numérique # 10-15 personnes

Thème de modélisation au COM

- Eco3M (Ecological Mechanistic and Modular Modelling)
- code en F90 de modélisation biogéochimique.

- il permet de modéliser le fonctionnement biogéochimique d'écosystèmes en intégrant les forçages physiques (courants, vents...)
- génère automatiquement les équations du modèle biogéochimique
  - Équations diff de conservation de la matière
  - Etude de la variation de concentrations (phytoplancton..) au cours du temps
- compile les équations du modèle et intègre ces équations
- Plusieurs simulations lancées : variation des constantes du modèle, ou la structure même du modèle, ou changer les forçages (débit des fleuves, vent, conditions limites ..)

Thème de modélisation au COM

- Modèle SYMPHONIE/ECO3M-MED
  - Modèle couplé hydrodynamique océanique -biogéochimie marine ;
  - modèle décrivant à la fois la circulation océanique dans le bassin de Méditerranée nord occidentale
    - Equations de Navier Stokes
  - et la dynamique du plancton marin;
  - il produit des distributions spatiales et temporelles de quantité de plancton, de vitesse et sens des courants
  - 300-800 Mo de RAM
  - Code séquentiel
  - Plusieurs simulations lancées avec des forçages atmosphériques annuels différents

Motivations pour un système de cluster de calcul

- Préoccupations d'ASR : compromis performance/maintenance
  - Équipes se sont constituées progressivement et augmentent depuis 4-5 ans
  - Besoin de centraliser les moyens de calcul
    - Economiser sur l'achat de machines personelles,
    - Eviter l'administration d'une machine par chercheur en *"dual boot"*
  - crédits insuffisants pour l'achat de supercalculateur SMP
  - Codes séquentiels non parallélisés
  - Lancement de plusieurs simulations de modèles avec des paramètres différents
  - => Avoir un *cluster à répartition de charge* pour absorber les multiples modèles lancés par plusieurs chercheurs

openSSI : principales caractéristiques

http://openssi.eu

- Projet openSource sous GPL
- SSI : Single System Image
- Les clusters openSSI accroît la puissance de calcul en mettant en commun les processeurs de plusieurs machines reliées en réseau
- procurent une administration unifiée et centralisée de l'ensemble des machines
- Équilibrage de charge processeur et des connexions TCP
- Migration de processus automatique et préemptive vers les nœuds les moins chargés.
- Failover avec drbd

openSSI : principales caractéristiques

- **Process**
  - Un seul init system
  - Espace de process unique, Pid top, ps uniques pour tous les nœuds
  - Migration de processus avec réouverture des fichiers, socket, ipc
  - Equilibrage de charge entre les noeuds : pendant l'exécution des processus ou dès le départ l'exec time
  - espace IPC, des périphériques et réseau unique et partagé
  - Administration unifiée de l'ensemble des noeuds
- **File system**
  - Un seul file Root system /
  - CFS au dessus de ext3, chaque nœud à la même vision des disques
  - Réouverture des fichiers, devices, ipc quand un processus est migré
- **Swap**
  - Chaque nœud a son propre swap indépendant
- Equilibrage de connexion sur les nœuds (HA-LVS)
- FailOver avec DRBD

openSSI : fonctionnement général

- Un nœud central maître pour l'administration du cluster
  - 2 interfaces réseau une privée, l'autre publique
  - Fournit adresse privée IP aux autres nœuds par serveur dhcp interne
  - Fournit noyau de boot par tftp aux autres noeuds
  - on peut rajouter ou enlever dynamiquement des nœuds
    - $ ssi-addnode, ssi-rmnode
  - Spof : L'arrêt du nœud central , arrête tout (sauf si DRBD)

openSSI : Installation du noeud maître

- Sur fedora ou debian ou RH9
- Installation uniquement sur le noeud maître
- Lancer le script "install"
  - *it will ask you a few questions about how you want to configure your cluster and your first node*
    - Numéro de noeud, boot PXE, cvip.conf, adresse virtuelle du cluster
    - etc...
- Puis rajouter des noeuds au cluster avec
  - ssi-addnode

openSSI : ajout de noeuds au cluster

- **les autres noeuds du cluster sont comme des clients légers (PC sans disques) qui s'intégrent au cluster lors du boot en téléchargeant leur système par *« tftp »* sur le director node.**
- **Un nouveau nœud rejoint le cluster par un boot reseau (PXE ou Etherboot)**
- **Aucune installation sur le nœud à rajouter**
- **Le nœud maitre fournit une adresse privée par son serveur dhcp et le noyau de boot SSI par tftp**
- $ ssi-addnode
  - Demande l'adresse MAC du nœud
  - Le numéro de noeud
  - Configure le fichier dhcp.conf du master
  - Les nœuds du cluster sont dans le fichier */etc/clustertab* (construit automatiquement
    - 1 192.168.0.1 00:0D:56:01:98:D4 P 1 /dev/sda1
    - 2 192.168.0.2 00:14:22:35:80:B4 P 0
    - 3 192.168.0.3 00:14:22:35:81:24 P 0

openSSI : ajout de noeuds au cluster

- SSI => même noyau => gestion d'un matériel unique
- Problème de configuration matérielle avec des générations différentes de PC notamment sur les cartes réseau
- Openssi permet de prendre en compte les variétés de matériels
- **/etc/modprobe.conf** contient la liste des modules réseau à charger. Si on rajoute un noeud qui nécessite un nouveau driver réseau, il suffit de le rajouter
  - alias eth1 e1000
  - alias eth0 sk98lin
  - alias eth-extra tg3
- Reconstruire la ramdisk pour inclure le nouveau module. Il doit etre disponible dès le boot réseau
  - *# mkinitrd --cfs -f <initrd-image> <kernel-version>*
- Mettre à jour l'image du noyau de boot réseau :
  - *# ssi-ksync*
- génére un nouveau kernel dans le répertoire /tftpboot pour les noeuds esclaves qui démarrent via PXE ou Etherboot

openSSI : HA-LVS

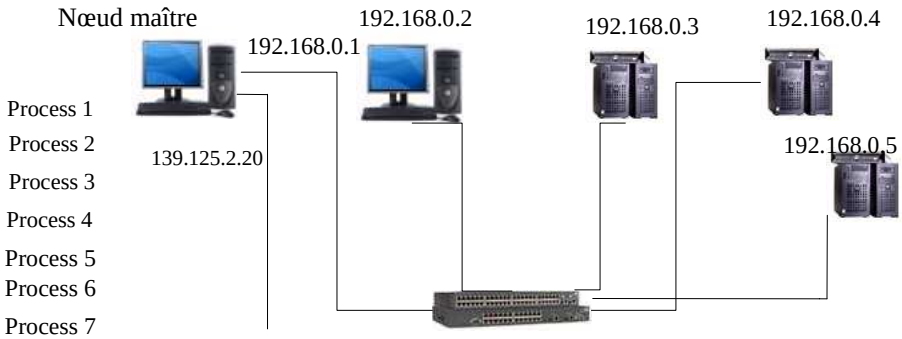
- High Avaibility – Linux Virtual Server
- Le kernel openSSI supporte LVS (load-balancing TCP et UDP) : étalement de charge entre les nœuds pour certains ports TCP et UDP
- connexions réseau entrantes peuvent être « load balancées » par HA-LVS
  - On se connecte sur une adresse virtuelle
  - Ha-lvs etablit la connexion vers un noeud du cluster
- Utilise un fichier de conf *cvip.conf* qui décrit la morphologie du cluster
- Exemple: les processus utilisant les ports de 1 et 80 seront répartis sur les différents noeuds du cluster.
- */usr/sbin/setport_weight --start-port=1 --end-port=80 --weight=1*

$ ssh comcluster -l root
[root@comclust2 root]# hostname

openSSI File System : CFS

- Problématique d'un File System de cluster : *comment rendre accessibles les fichiers aux processus qui après migration, s'exécutent sur d'autres noeuds?*
- Quelle que soit la machine où il a été migré, un processus est assuré d'y trouver un montage du système de fichiers qu'il avait sur le noeud d'origine.
- openSSI procure un root filesystem unique et une visibilité automatique du file systme sur chaque noeud.
- CFS est automatiquement superposé au montage des filesystem ext2 or ext3
- Chaque noeud du cluster voit les mêmes points de montage.
- Les noeuds qui se joignent au cluster ont également la même vision de l'espace de fichiers

openSSI : configuration



- On privilégiera un ensemble de PC homogènes et un réseau Gigab/s

openSSI : un “benchmark” simple

Lancement de 7 processus de compression de données **.wav en *.mp3* avec *“bladeenc”*

- ***En bloquant la migration*** ***(loadlevel -n 2 off)*** : conditions d'exécution classiques de 7 processus concurrentiels sur un seul noeud.. durée 5 minutes environ.
- ***En autorisant la migration*** (*bash-ll*) : répartition automatique des 7 processus sur les meilleurs noeuds du cluster. Meilleures ressources mémoire et CPU. L'encodage *mp3* se termine en 2 minutes environ.
- Overhead “surcharge du temps de migration” dépend de la taille des processus

openSSI : migration de processus et répartition de charge

- La granularité des échanges est le «processus»
- openSSI travaille au niveau du processus et ne *nécessite pas de modification des applications*, (pas de librairies de parallélisation MPI)
- L' équilibrage de charge se fait de manière transparente par le noyau Linux modifié
  - recherche automatique d'un noeud à la performance optimale pour l'exécution des processus.
  - capacité à répondre *dynamiquement* et *préemptivement* aux variations de ressources des différents noeuds et donc à des conditions d'exécution *irrégulières et non prédictibles*

openSSI : migration de processus et répartition de charge

- Les mêmes algorithmes de décision sont calculés sur chaque nœud
- Chaque nœud est autonome et prend ses décisions indépendamment sur la base d'informations prises sur une vue partielle du cluster (nœuds tirés au hasard)
- Un cluster de 4 nœuds fonctionnera comme avec 400
- L'algorithme de décision de répartition de charge est issu de openMosix
- Un algorithme de « migration des processus » : migration se fait sur la base d'informations retournées par les autres nœuds
- Un algorithme de «dissémination d'information probabiliste»
- Un ensemble d'algorithme de « partage des ressources » :
  - cherche à réduire les différences de charge entre nœuds en migrant les processus vers les nœuds moins chargés
  - mettre le plus de processus en mémoire sans activer le swap
  - Calcul d'un « coût » caractéristique d'une machine (meilleur CPU+RAM…)
  - Envoi des processus vers la machine ayant le meilleur coût

openSSI : process load balancing contrôler la migration

- Plusieurs manières d'administrer l'équilibrage de charge et la migration de processus
  - Lister explicitement le nom des programmes autorisés à migrer
    - /etc/sysconfig/loadlevellist
      - /bin/bash-ll
      - /usr/bin/bladeenc
  - Lancer bash-ll dès l'entrée en session
    - Toutes les commandes tapées dans le shell seront éligibles à la migration

openSSI : process load balancing maîtriser la migration

- Autoriser l'equilibrage de charge par noeuds
  - loadlevel -a on
  - loadlevel -n <node #> [on| off]
- Autoriser l'equilibrage de charge par pid
  - $ loadlevel –p <numerodeprocess>
- Forcer la migration sur un autre noeud
  - $ migrate <numero_noeud> <pid>

openSSI : quelques commandes d'administration

- ***$ cluster –v*** *: voir l'état des nœuds sur le cluster*
- ***$ cluster –V*** *: détail des informations sur chaque nœud (UP, CPU…)*
- ***$ localview*** *cmd : éxécute la commande cmd en local sur un noeud*

- ***$ onnode N°nœud localview cmd*** *: exécute commande cmd sur nœud*
- ***$ onall <commande>***
  - ***$ onall ip addr :*** *détails sur la configuration du cluster.*

- ***$ ps ef --shownode :*** *Savoir sur quel noeud tourne un daemon.*
- ***$ where_pid <num_pid> :*** *sur quel nœud tourne tel process*
- ***$ fastnode*** *: renvoie le numéro du noeud le plus rapide*
- ***$ fast cmd*** *: exécute la commande cmd sur le noeud le plus rapide.*
- ***$ /sbin/clusternode_shutdown : pour rebooter un noeud***

openSSI : quelques commandes d'administration

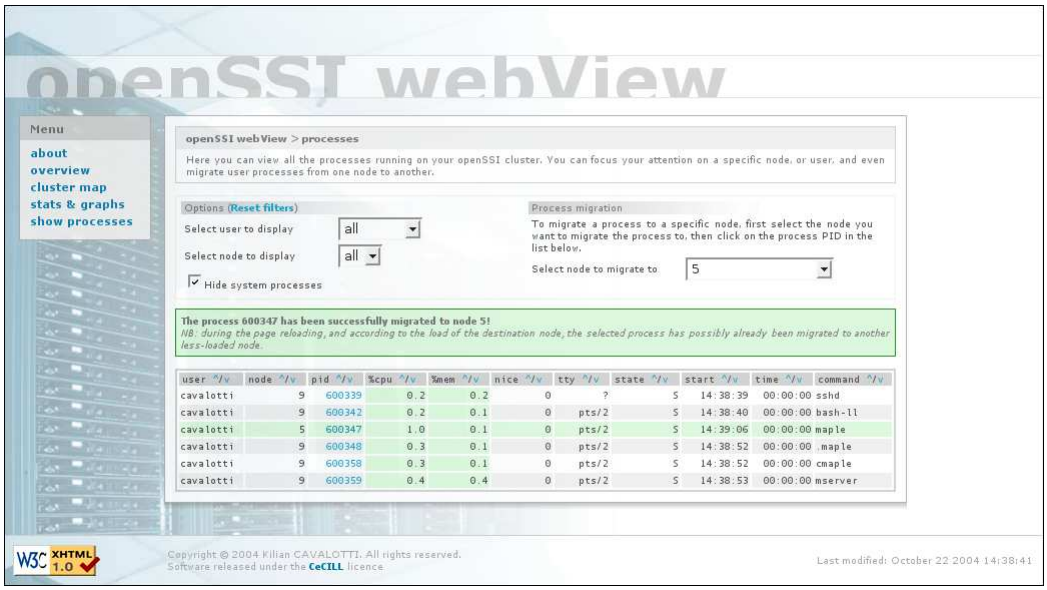
- *$ cluster –v*
  - *1: UP*
  - *2: UP*
- *$ loads*

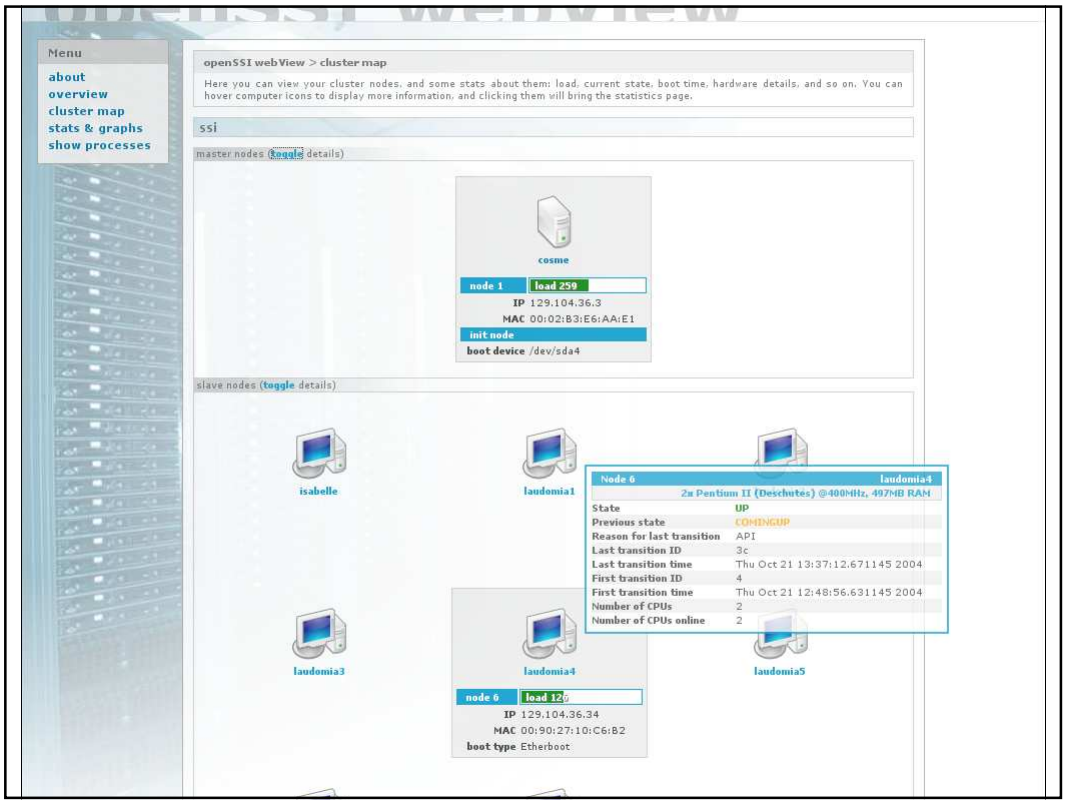
  1: 26
  2: 8
  3: 9
  4: 21
- *$ onnode 2 date*
- *$ onall date*
- *$ onnode 3 localview top*

openSSI : administration graphique

- http://openssi-webview.sourceforge.net/
- ***Système de surveillance***
  - *Surveillance de l'état du cluster,*
  - *Détail sur les noeuds, information sur le hardware et le status des noeuds*
  - *Graphes RRDTool pour afficher le taux CPU*
- ***Gestion des processus***
  - *Liste des processus processes list and information accross the cluster,*
  - *Possibilité de migrer les processus*

openSSI : administration graphique




openSSI webView
Menu
about
overview
cluster map
stats & graphs
show processes
openSSI webView > cluster map
Here you can view your cluster nodes, and some stats about them: load, current state, boot time, hardware details, and so on. You can hover computer icons to display more information, and clicking them will bring the statistics page.
ssi
master nodes (toggle details)
cosme
node 1
load 259
IP 129.104.36.3
MAC 00:02:B3:E6:AA:E1
init node
boot device /dev/sda4
slave nodes (toggle details)
isabelle
laudomia1
laudomia3
laudomia4
laudomia5
node 6
load 12
IP 129.104.36.34
MAC 00:90:27:10:C6:B2
boot type Etherboot
Node 6
laudomia4
2x Pentium II (Deschutes) @400MHz, 497MB RAM
State UP
Previous state COMINGUP
Reason for last transition API
Last transition ID 3c
Last transition time Thu Oct 21 13:37:12.671145 2004
First transition ID 4
First transition time Thu Oct 21 12:48:56.631145 2004
Number of CPUs 2
Number of CPUs online 2


openSSI webView > stats
Here you can view some graphical statistics about your openSSI cluster:
Cluster overview gathers links bringing to thematic stats pages, showing data accross the cluster.
Load overview shows openSSI cumulated load for each node on the cluster.
Node overview gives statistics by node.
Cluster overview
CPU usage
Load average
Memory usage
Swap usage
Processes
Network traffic
Load overview
Cumulated openSSI loads
1.5 k
1.0 k
0.5 k
0.0
00:00
02:00
04:00
06:00
08:00
10:00
Load #1: 266
Load #2: 208
Load #3: 127
Load #4: 127
Load #5: 127
Load #6: 127
Load #7: 127
Node overview
Select the node you want to display stats for:
node 1
cpu usage (node 1)
load average (node 1)
CPU usage
Load average

openSSI : conclusions

Tirer bénéfice d'un cluster openSSI revient à écrire des applications qui lancent des processus (« fork and forget »).

- Avantages:
  - Facile à installer et administrer
  - Facilité d'addition des noeuds (si matériel identique)
  - Pas de modification des applications
  - Anneau de calcul où les PC collaborent entre eux.
  - Migration automatique des processus vers les machines les moins chargées
  - Répartition de charge permanente automatique
  - Gain global de temps d'exécution
- Inconvénients
  - plante de temps en temps
  - Gymnastique pour intégrer de nouveaux modules dans le kernel

Conclusions

Avec OpenSSI nous avons

- un moyen de constituer un cluster de calcul, simple pour l'administrateur et efficace pour les besoins de calcul de notre laboratoire.
- un bon compromis entre l'achat de coûteuses machines de calcul parallèle et l'achat de multiples machines individuelles pour chaque chercheur
- un gain dans les temps de calcul de processus concurrentiels

Merci…

- Questions?