Gestion de Clusters de grande taille

Mathieu Hautreux
(CEA/DAM/DIF)
matthieu.hautreux@cea.fr

Plan de la présentation

- Problématique
- Administration des ressources
- Gestion des ressources

Problématique

Problématique

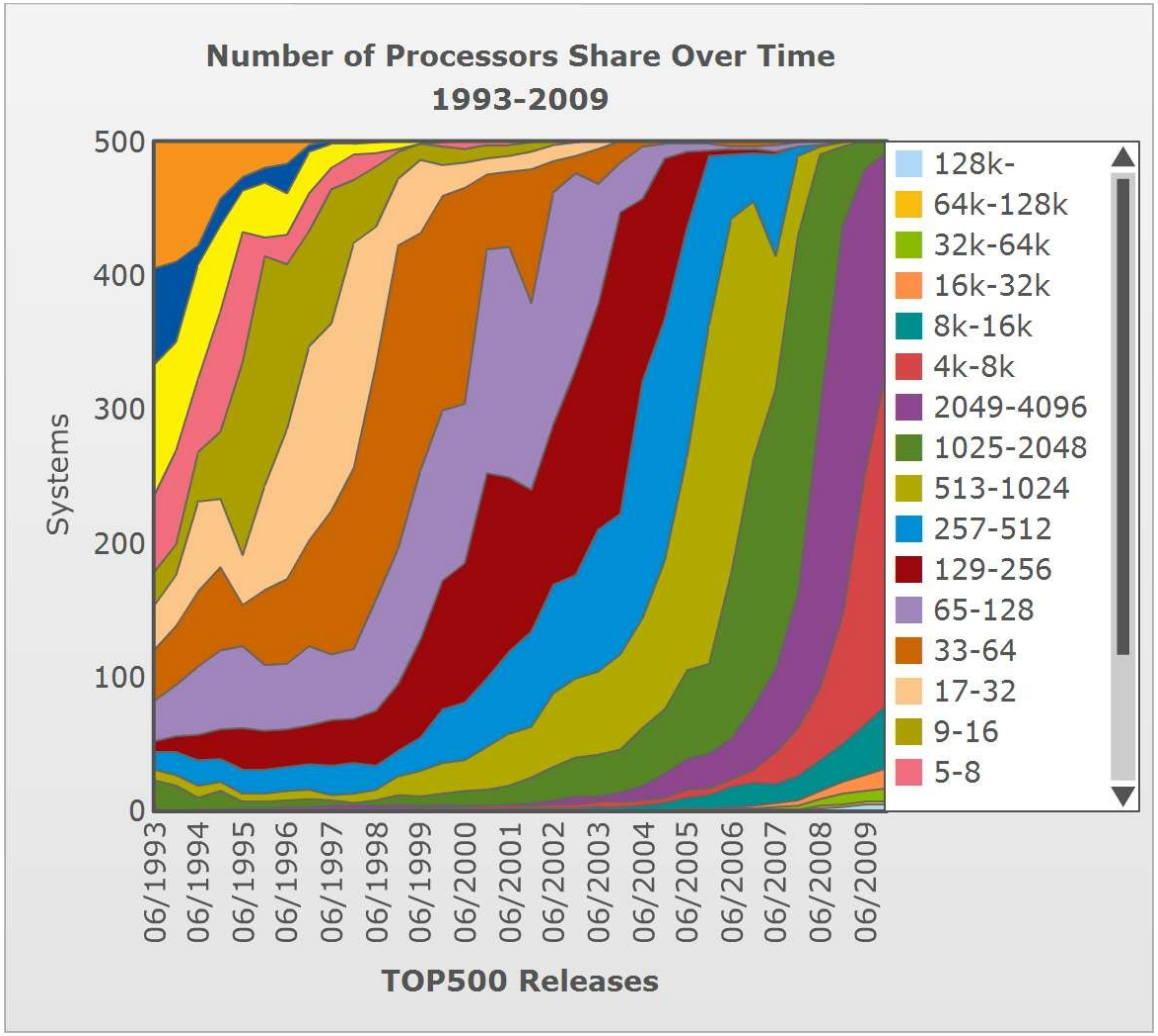
- Codes de calcul
  - De plus en plus complexes
  - De plus en plus exigeants en ressources
- Processeurs génériques
  - Stabilisation des performances des coeurs
  - Augmentation du nombre de coeurs
- Noeuds de calcul
  - Augmentation du nombre de coeurs
  - Augmentation des débits réseaux
  - Densification des composants

Problématique

- Cluster
  - Augmentation du nombre de coeurs
  - Augmentation du nombre de noeuds
  - Augmentation de la consommation énergétique
    - ☛ Parallèlement au nombre de noeuds
  - Augmentation de la complexité réseau
    - ☛ Eclatement du nombre de liens
  - Augmentation de la complexité logicielle
    - ☛ Eclatement du nombre d'entités à gérer

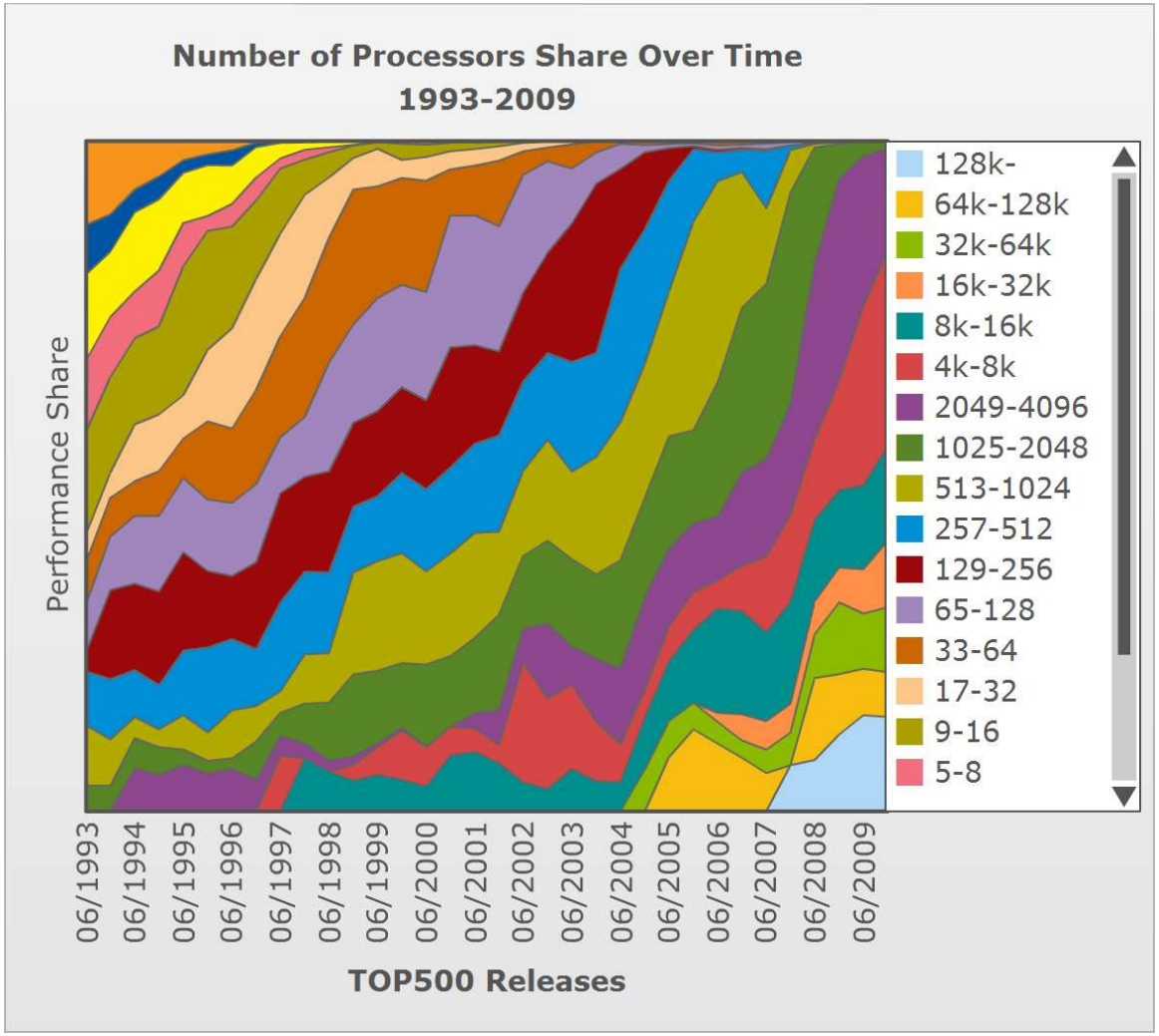
Problématique

- Source : http://www.top500.org



Problématique

- Source : http://www.top500.org



Administration des ressources

Administration des ressources

- Système en production : Tera10
  - Classe “1k nodes”
  - 1 noeud de management (SPOM), redondé
  - Plusieurs dizaines de noeuds IO
  - Plusieurs centaines de noeuds de calcul
- Système cible : Tera100
  - Classe “10k nodes”
  - 1 noeud de management (SPOM), redondé
  - Plusieurs centaines de noeuds IO
  - Plusieurs milliers de noeuds de calcul (~4000)
  - En cours d'installation

Administration des ressources

- Environement logiciel cible
  - Linux RedHat 6
  - Piles logicielles opensource
    - Développements CEA
    - Développements communautaires
    - Intégration/Support Bull (Lustre, Slurm,...)
  - Piles logicielles propriétaire
    - Bull Cluster Management
    - BullMPI
- Besoins
  - Scalabilité
  - Sécurité
  - Tracabilité
  - Simplicité

Administration des ressources

- Principes d'administration
  - Centralisation de la connaissance du système
    - Toutes les informations importantes stockées dans le gestionnaire de configuration
  - Séparation des notions configuration/déploiement
    - Outils de configuration pour configurer
    - Outils de déploiement pour déployer
  - Cohérence des états
    - Installation + configuration = Déploiement + configuration
    - Déploiement pour optimisation “installation + configuration”
  - Utilisation d'outils éprouvés
    - Largement utilisés et/ou communautairement supportés

Administration des ressources

- Méthodologie d'administration
  - Séparation des noeuds par rôle
    - IO, Compute, Login, ...
  - Définition de noeuds de référence
    - Typiquement 1 par noeud
  - Automatisation de l'installation/configuration des noeuds de référence
    - Installation réseau + post-configuration
  - Réplication des noeuds de référence
    - Prise d'images des noeuds de référence
    - Déploiement des images sur les noeuds du cluster en fonction des rôles de chacun
  - Post-configuration des noeuds répliqués
    - Application du delta résiduel (spécifique à chaque noeud)

- Outils impliqués
  - Boot réseau
    - Le SPOM décide du comportement au démarrage des noeuds
    - **DHCP/PXE** (ethernet) - **DHCP/GPXE** (infiniband)
  - Installation des noeuds références
    - Serveur **HTTPD** hébergeant les piles logicielles sur le SPOM
    - Installation réseau via **kickstart**
    - Mise à jour réseau via **Yum**
  - Configurations des noeuds (références et répliqués)
    - Utilisation de l'outil **Puppet**
    - Base de connaissance sur le SPOM
    - Serveur dédié sur le SPOM
    - En mode local sur le SPOM (autoconfiguration)
    - En mode distant sur les autres noeuds
    - Sécurité fournie par une couche **SSL** (encapsulée par **Puppet**)

Administration des ressources

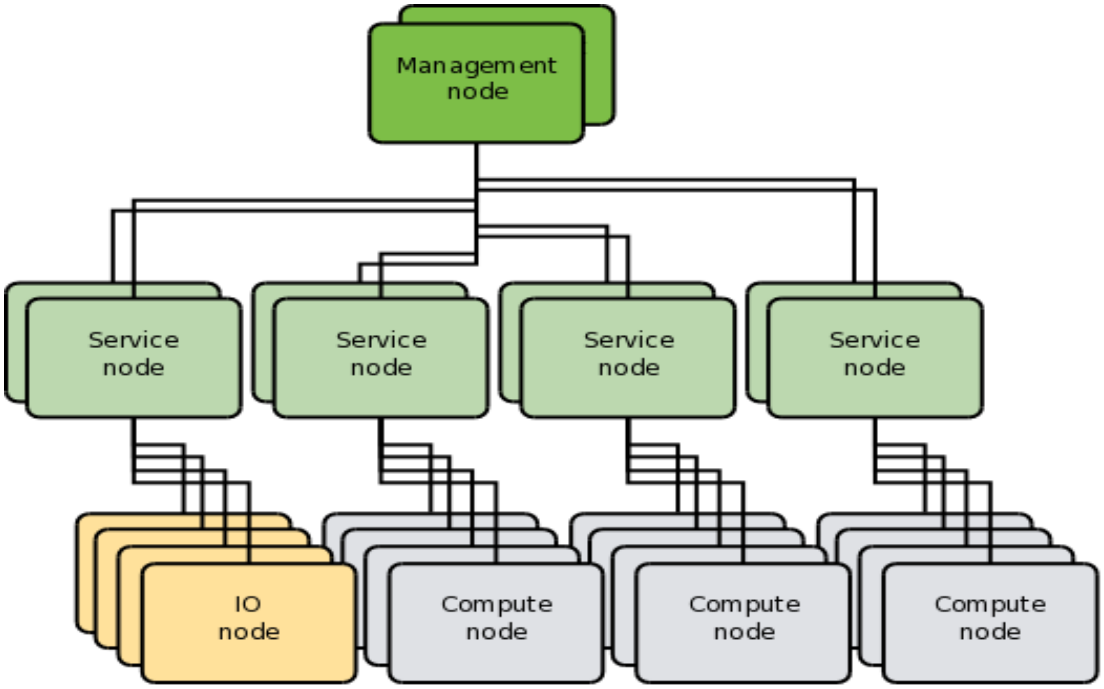
- Outils impliqués
  - Déploiement – **Systemimager**
    - Mode “Pull”, lecture de données publiques hébergée par le SPOM
    - **Rsync/Torrent** comme moteur de téléchargement
    - Transferts d'images multiples en parallèles
  - Déploiement – **KSIS**
    - Propriétaire **BULL**
    - Mode “Push”, transmission des données depuis le SPOM
    - Chaînes de transmission des données via **SSH**
    - Transferts d'images multiples en parallèles
  - Gestion à distance
    - **Ipmitools** pour le “power management” (on, off, cycle, diag,...)
    - **Conman** pour l'accès aux consoles séries
    - **Clustershell** pour l'exécution de commandes en parallèle via **SSH** (développement opensource CEA)

Administration des ressources

- Retour d'expérience “1k nodes”
  - Expérience Tera10
    - Principes semblables, outils différents
  - Expérimentation dans la phase R&D de Tera100
    - Clusters de R&D Tera+, Prototypes PRACE
    - Activités R&D CEA/Bull (Prototype Tera100)
    - 2008-2009
  - Définition d'une solution adéquate
    - Architecture
    - Sélection d'outils adaptés (HTTPD, Puppet,...)
    - Développement d'outils adaptés (KSIS, Clustershell, ...)
- Perspectives “10k nodes”
  - Exploiter l'expérience “1K” pour garantir le fonctionnement sur “10k”

Administration des ressources

- Design d'une solution “10k nodes”
  - Cluster de clusters
  - 1 noeud de management (SPOM) redondé
  - 1 cluster de noeuds de services/administration
  - n clusters de noeuds d' IO et de calcul



Administration des ressources

- Cluster de services/administration
  - Fourniture des services du SPOM
    - Chaque noeud renseigne 1 ou n clusters sous-jacents
    - En mode “cascade” (i.e. **LDAP**)
    - En mode “proxy” (i.e. **Squid**)
    - En mode “réplication” (i.e. **Puppet)**
  - Stations de traitement du SPOM
    - Chaque noeud sert de passerelle à 1 ou n clusters
    - Parallélisation des exécutions distribuées
    - Utilisation transparente par le SPOM (**Clustershell** v2.0, en cours de développement)
  - Extensible
    - Fonction du nombre de clusters
    - Fonction du niveau de redondance souhaité
  - Transparent
    - Redirections transparentes vers le bon interlocuteur (via utilisation d'alias et zoning **DNS**)
    - Les noeuds clients n'ont pas conscience de ce cluster

Gestion des ressources

Gestion des ressources

- Spécificités CEA
  - Utilisations interactives
    - Exécution interactive d'applications parallèles
    - Soumission batch
    - Monitoring des applications en attente/exécution
    - Gestion des applications (arrêt, modifications des paramètres,...)
  - Utilisations planifiées
    - Soumissions automatisées de jobs batch
    - Metascheduler multi-clusters CEA, **openKMS**
    - Gestion des quotas utilisateurs
    - Exécution d'étude (train de jobs jusqu' à convergence vers le résultat)

Gestion des ressources

- Besoins
  - Performance
  - Scalabilité
  - Fiabilité / Tolérance aux pannes
  - Optimisations des exécutions
  - Exécutions Batch
- Thématiques
  - Lanceur de tâche(s)
  - Exécution Batch
  - Gestion des files d'attentes
  - Gestion des limites utilisateurs

Gestion des ressources

- Tera10, principe
  - Quadrics **RMS**
    - ☛ Lancement des tâches parallèles (prun remplaçant mpirun)
    - ☛ Intégration fine avec la couche MPI **MPIBULL**
  - Platform **LSF**
    - ☛ Exécution de scripts Batch
    - ☛ Gestion des files d'attentes et des limites
  - CEA **BRIDGE**
    - ☛ Gestion de la jonction LSF/RMS
    - ☛ Masque les outils sous-jacents aux utilisateurs et au metascheduler
- Tera10, retour sur expérience
  - Complexité de l'ordonnancement
    - ☛ Jobs batch **LSF** encapsule les exécutions **RMS**
    - ☛ 2 outils, 2 schedulers en cascade, déterminisme difficile
  - Importance du lanceur de tâches
    - ☛ Performances de **RMS** nettement au dessus de ses concurrents

Gestion des ressources

- Tera100, nouveaux besoins
  - Augmentation du nombre de noeuds
    - ☛ Importance d'un lanceur parallèle performant
  - Augmentation du nombre de coeurs
    - ☛ Importance d'un ordonancement déterministe et fiable
  - Augmentation de la complexité réseau
    - ☛ Importance d'un ordonancement adapté
- Tera100, solution recherchée
  - 1 lanceur optimisé et performant
  - 1 système de Batch évolué
  - 1 ordonanceur efficace
  - 1 outil unique
  - 1 système de comptabilité adapté

- Slurm
  - Origines et spécificités
    - Déloppement opensource (GPL) démarré au **LLNL**
    - Initialement inspiré de Quadrics **RMS**
    - Initialement dédié au rôle de lanceur de tâches
    - Principalement destiné aux clusters LINUX
    - Portable, écrit en C, configuré via GNU Autoconf
  - Activité
    - 1 à 2 versions majeurs par an
    - 1 à 2 versions mineurs (bugfix) par mois
    - Version de développement mise à jour mensuellement
    - Très bonne réactivité de la mailing list
    - ~1/3 des clusters du Top500 l'utiliserait

Gestion des ressources

- Slurm
  - Scalabilité
    - Jusqu'à 65 536 noeuds de calcul
    - Conçu initialement pour faire face à cette problématique
    - Gestion arborescente des communications (paramétrable)
    - Serveurs multithread
  - Fiabilité
    - Redondance des composants centraux
    - Monitoring des ressources avec éviction des noeuds suspects
    - Relance automatique des job en cas de crash (paramétrable)
  - Fonctionalités
    - Exécutions intéractive et Batch
    - Gestion de queues d'exécution
    - Partage avancés des ressources (fairShare, QOS, ...)
    - Ordonancement avancé (linéaire, backfill, gang scheduling)
    - Sélection des ressources (noeuds, coeurs, mémoire)

Gestion des ressources

- Slurm
  - Modularité
    - Sous forme de plugins pour les différents sous-systèmes
    - Priorisation des jobs
    - Ordonancement
    - Sélection des ressources
    - Gestion de l'affinité des tâches (binding cpus)
    - Couche MPI (mvapich, mpich2,...)
  - Flexibilité
    - De quelques stations à plusieurs milliers de noeuds
    - De quelques utilisateurs à une large population utilisatrice
    - D'une queue d'exécution simple à de multiples queues avec préemption
    - D'une tracabilité faible au maintient d'une base de tous les jobs exécutés pour la comptabilité
    - ...

- Slurm
  - Composants usuels
    - Slurmctld : démon central en charge de la supervision du système et de l'exécution des jobs (1 primaire et un secondaire)
    - Slurmd : démon en charge de la gestion des applications sur les noeuds de calcul
    - Munged : démon fournissant la couche de sécurité pour l'authentification des communications
    - Slurmdbd : démon responsable du stockage des informations du cluster pour les modes avancés (Tracabilité, FairShare, QOS,...)
  - Commandes usuelles
    - Exécutions : srun, salloc, sbatch
    - Monitoring : sstat, squeue, sinfo, scontrol, sview (GUI)
    - Administration, gestion : scancel, scontrol
    - Tracabilité, comptabilité : sacct, sreport

Gestion des ressources

- Slurm & Tera100
  - Queues d'exécution intéractive et batch
    - Temps limites d'exécution relativement faible
    - Limitation du nombre de jobs et de coeurs par queue
    - Associées aux noeuds de calcul
  - Queue d'exécution associé au metascheduler
    - Sans temps limites
    - Sans limitation du nombre de jobs et de coeurs
    - Associée aux noeuds de calcul
  - Queue d'exécution dédiée au “stage in/out”
    - Utilisable avant/après un job
    - Facilite les transferts de données de et vers le cluster
    - Limitation à un coeur par job
    - Associée aux noeuds de login

Gestion des ressources

- Slurm & Tera100
  - Allocation exclusive de coeurs de calcul
    - Avec sélection de la quantité de mémoire nécessaire par coeur
    - Choix des nombres de noeuds, de processus et de coeurs par processus possibles
  - Ordonancement type Backfill
    - Améliore l'ordonancement des charges hétérogènes
    - Repose sur une bonne description des durées des jobs
  - Sélection des noeuds selon la topologie réseau
    - Noeuds de calcul répartis en ilôts
    - Interconnection fat-tree en interne d'un ilôt
    - Interconnection “pruned tree” entre ilôts
    - Utilisation du nombre d'ilôts minimal en priorité

- Slurm & Tera100
  - Lanceur et souches MPI
    - **Openmpi** <= 1.4.2, support partiel via **salloc+mpirun**
    - **Openmpi** branche dev, support complet via **srun**
    - **Bullmpi**, support complet via **srun** (basé sur branche dev openmpi)
  - Optimisation des performances applicatives
    - Cloisonnement des tâches via **cpuset**
    - Affinité via **cpuset**
    - Stratégie basée sur les **cgroup** en développement (pour cloisonnement mémoire, suspend/resume atomique, ...)

Gestion des ressources

- Slurm & Tera100
  - Gestion des utilisateurs
    - Arborescence d'utilisateurs et de groupes
    - Calquée sur l'organisation interne au site
    - Limites Slurm asssociées au noeud de l'arbre
    - Via le composant optionnel **slurmdbd**
  - Comptabilité et tracabilité
    - Archivage des jobs exécutés
    - Synchronisation quotidienne du FairShare **openKMS**
    - Rapport quotidien d'activité
    - Via le composant optionnel **slurmdbd**
  - Premiers retours d'expérience
    - En production sur le démonstrateur Tera100 (~400 noeuds)
    - Bon fonctionnement général
    - Intégration complete avec **openmpi** importante pour les performances (gestion de l'affinité)

Questions ?