Gestion de la qualité sur Grid'5000

Arnaud Fontaine

14/06/2007

Contexte et objectifs du stage

Antenne de Montbonnot du LIG

- 60 personnes dont 20 permanents
- Axes de recherche sur les systèmes informatiques distribués

Objectifs du stage

Développement d'outils permettant d'améliorer la qualité de l'infrastructure de Grid'5000

Plan

Grilles de calcul

Utilité

- Puissance de calcul importante (physique, mathématiques, etc.)

Cluster ou grappe

- Homogène
- Localisé

Grille de calcul ou grille de grappes

- Hétérogène
- Distribué

Avantages

- Machines de série (moins coûteux qu'un super-calculateur)
- Décentralisé
- Extensible

Grid'5000

Objectifs

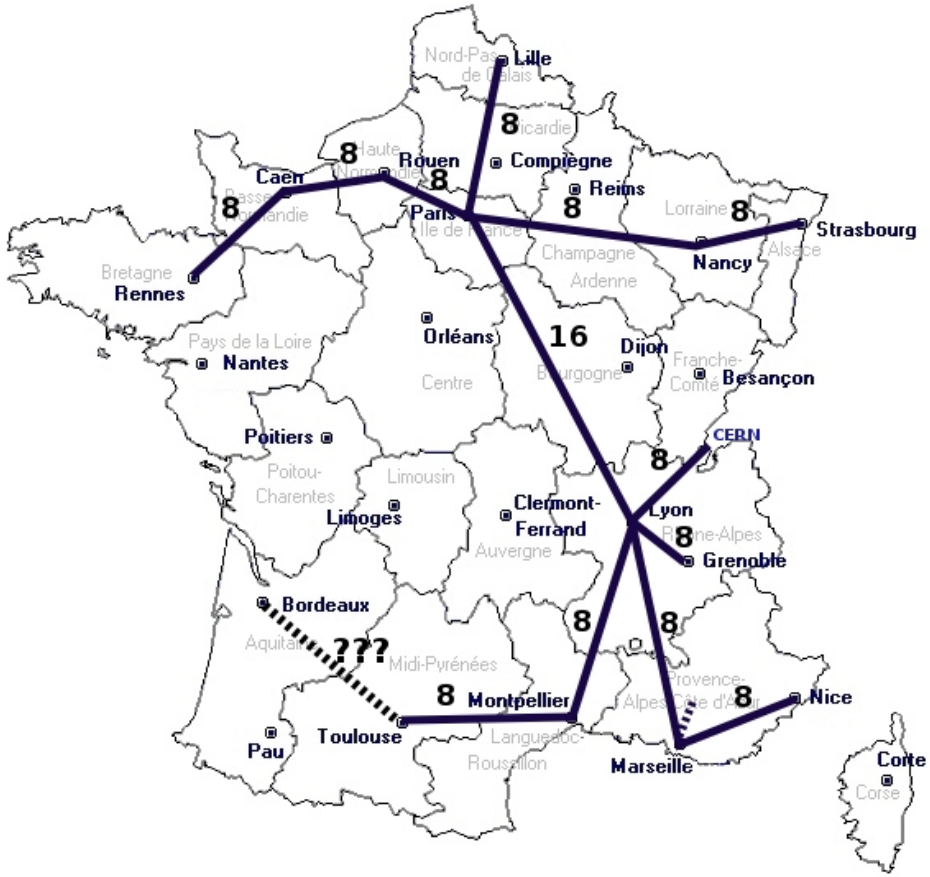
Grille de calcul expérimentale utilisée à des fins de recherche en informatique

Caractéristiques générales

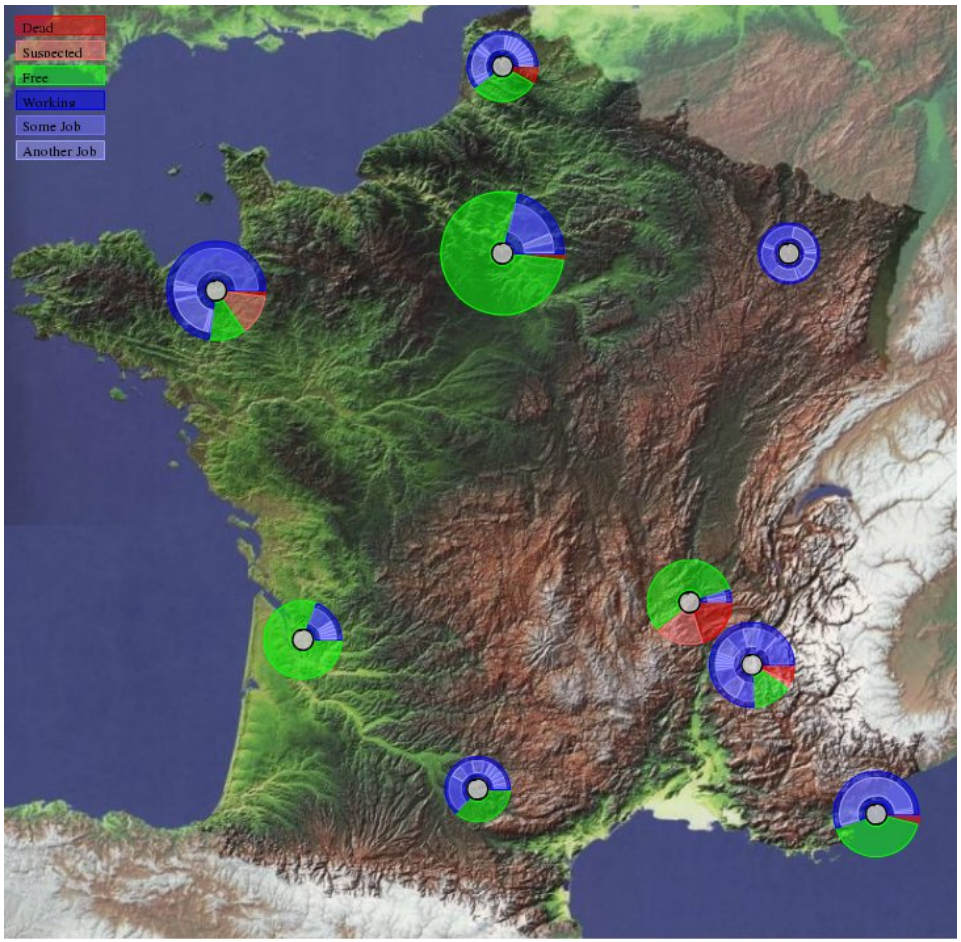
- 9 sites
- 1500 noeuds
- 3500 coeurs

Caractéristiques réseaux de Grid'5000

- Connexion réseau 10Gb/s dédiée grâce à RENATER
- Un frontal par cluster afin d'accéder ensuite aux noeuds



Répartition des noeuds et leur état sur la France



Caractéristiques logicielles de Grid'5000

Caractéristiques logicielles

- Système d'exploitation GNU/Linux
- Outils standards (NFS, SSH, etc.)
- OAR
- KADEPLOY

OAR

Utilité

Nécessité d'ordonnancer les *jobs* (multi-utilisateur)

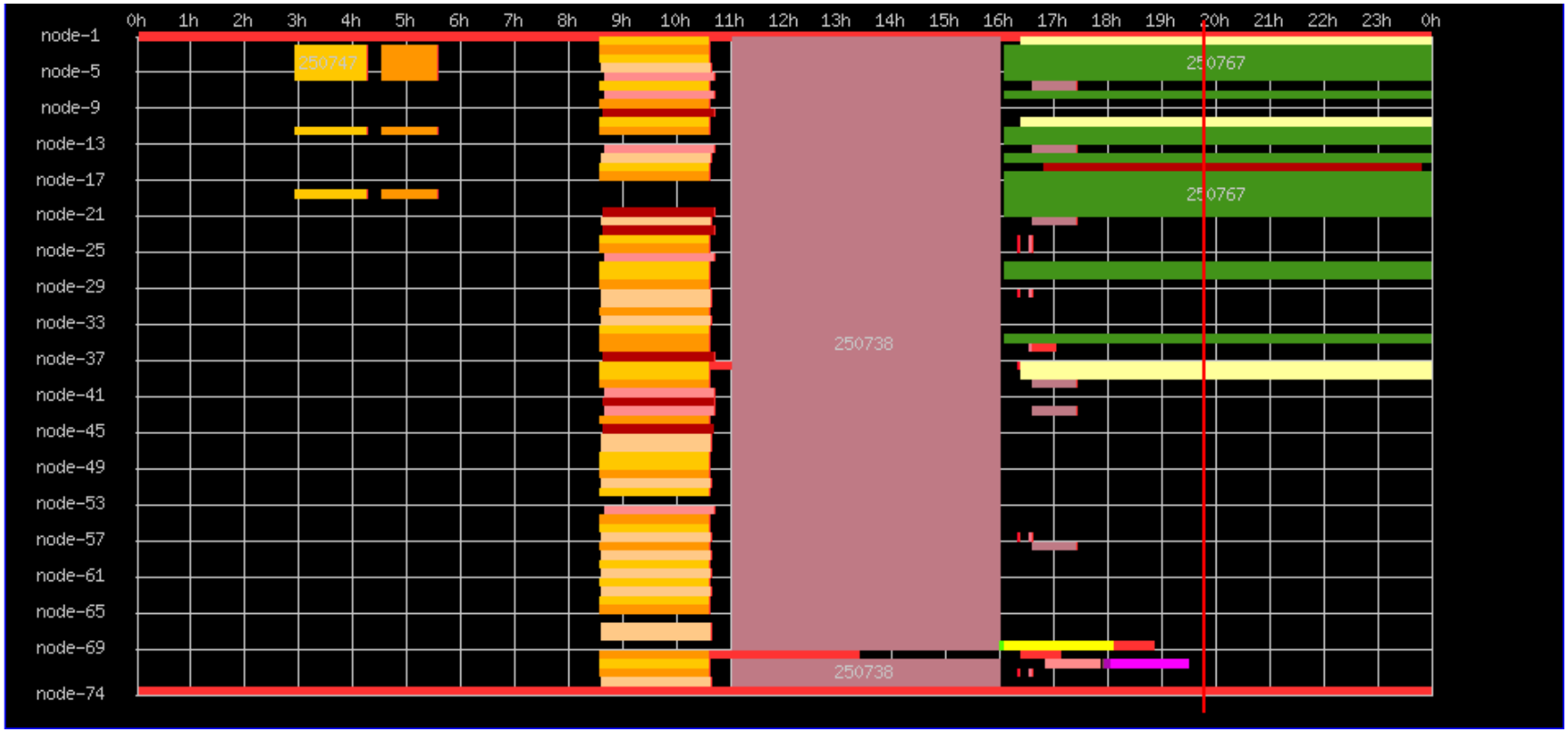
Description

- Outil permettant de réserver un ensemble de noeuds
- Utilisation d'une base de données
- Outils de visualisation (DRAWGANTT, MONIKA)

Cas d'utilisation : mode interactif

1. Choix du cluster
2. Connexion à distance sur le frontal via SSH
3. Utilisation de `oarsub` pour réserver les noeuds
   - Durée
   - Nombre de noeuds
   - Exemple : `oarsub -l nodes=308,walltime=5 -I`

Diagramme de Gantt d'utilisation de la plate-forme



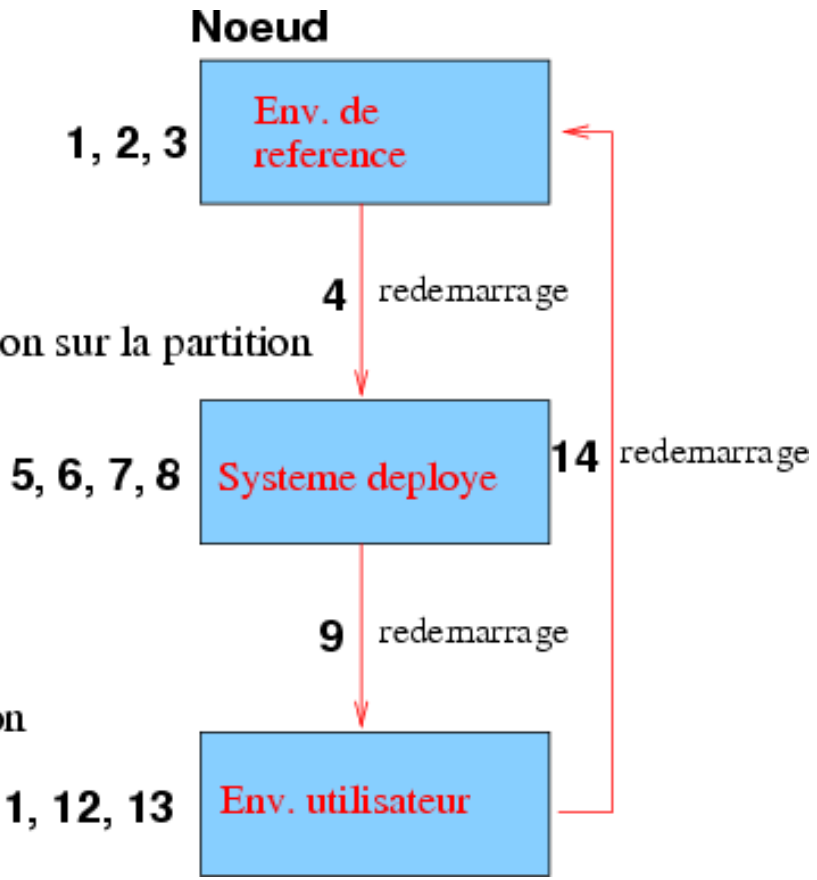
KADEPLOY

Description

- Déploiement d'environnements personnalisés
- Déploiement rapide même sur un grand nombre de noeuds
- Installation sur une partition spécifique

KADEPLOY : étapes d'un déploiement

1) Soumission
2) Attribution / Ouverture de session
3) Control des permissions de deploiement
4) Demarrage sur l'environnement minimal
5) Preinstallation
6) Propagation de l'environnement + decompression sur la partition
7) Postinstallation
8) Redemarrage
9) Demarrage sur le nouvel environnement
10) Travail sur l'environnement
11) Indication de fin de session
12) Retrait des droits de deploiement / Fin de session
13) Redemarrage
14) Demarrage sur l'environnement de reference



KADEPLOY

Problèmes

- Fiabilité
- Efficacité

Nécessité de développer un outil permettant de détecter les problèmes de configuration et les bogues de KADEPLOY

Plan

Présentation I

Objectifs

- Tests de KADEPLOY
  - problèmes de configuration
  - tests de non-régression
- Affichage de statistiques sur les temps et échecs de déploiements
- Lancement sur l'ensemble des clusters du projet Grid'5000

Présentation II

Déroulement

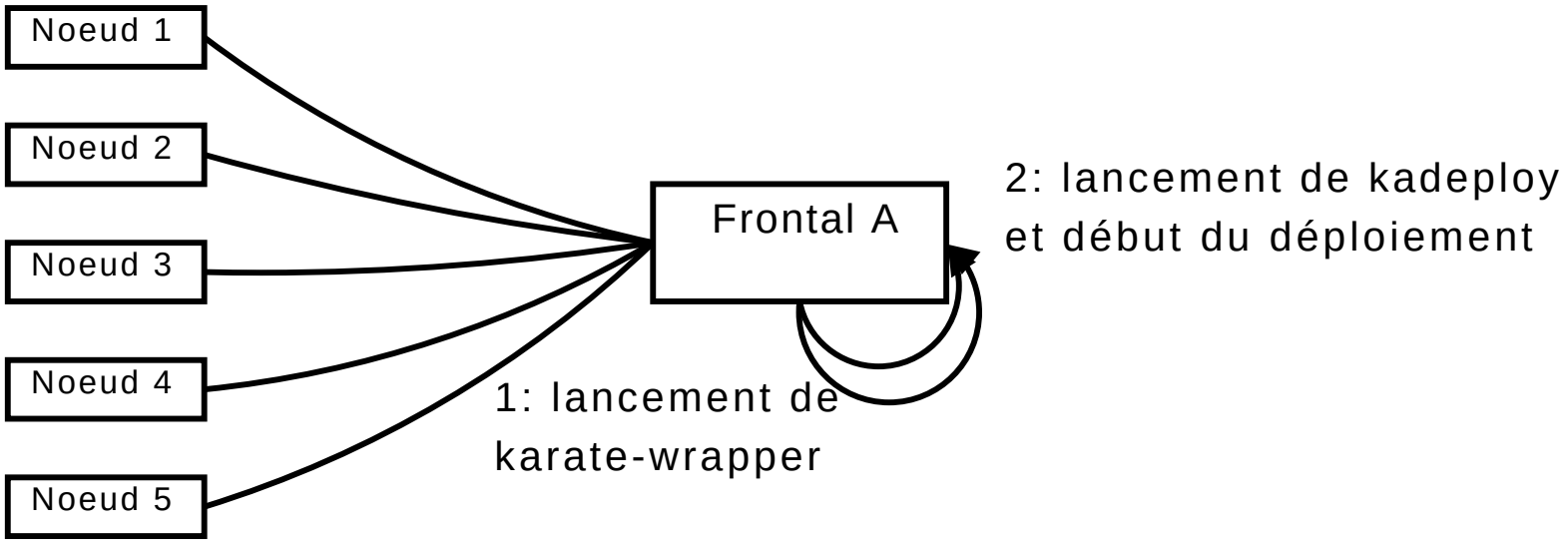
Pour un test donné :

1. 1 déploiement
2. 2 déploiements en parallèle
3. 4 déploiements en parallèle
4. 8 déploiements en parallèle

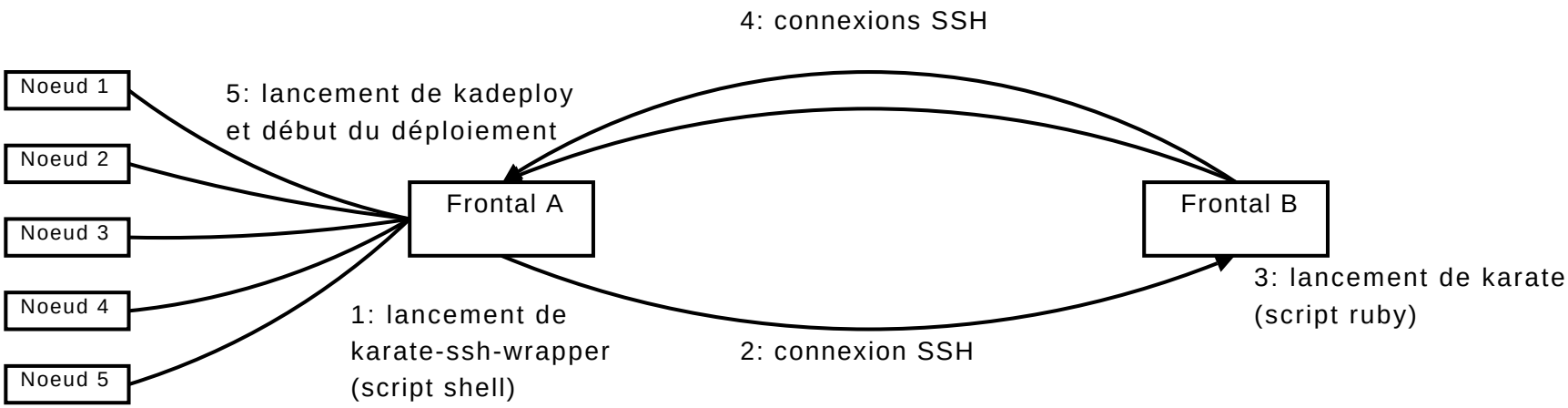
Implémentation

- Script écrit en RUBY
- Rapport au format texte/YAML
- Sauvegarde de la sortie de KADEPLOY grâce à la modification d'un module externe (`cmdctrl`)

Fonctionnement normal



Wrapper SSH


4: connexions SSH
Noeud 1
Noeud 2
Noeud 3
Noeud 4
Noeud 5
5: lancement de kadeploy
et début du déploiement
Frontal A
Frontal B
3: lancement de karate
(script ruby)
1: lancement de
karate-ssh-wrapper
(script shell)
2: connexion SSH

Exemple d'utilisation de KARATE

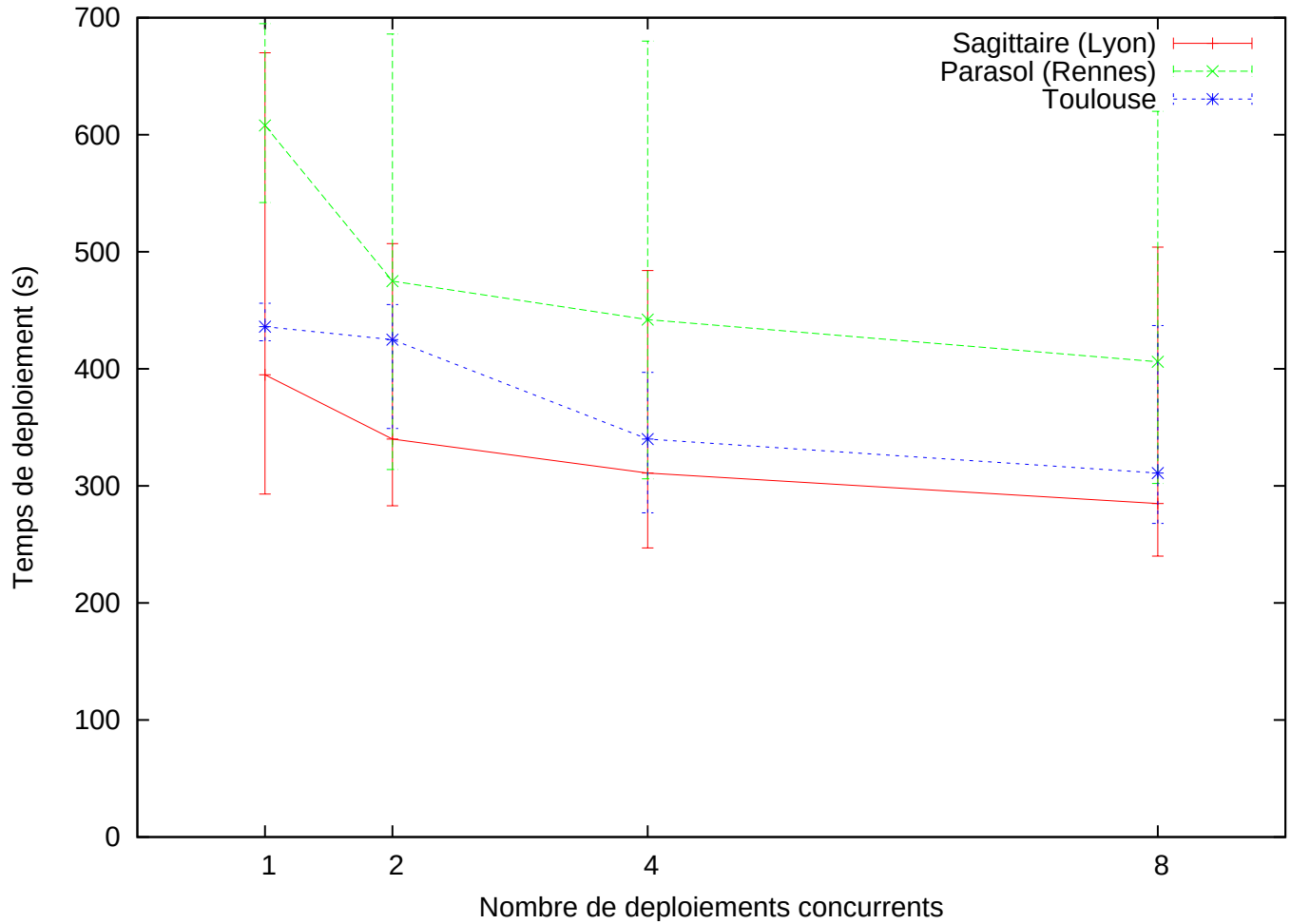
```
$ oarsub -q deploy -l nodes=308,walltime=5 \
  "./bin/karate-wrapper -e sid-x64-base-1.0 -p sda3 -n 6 -o orsay-gdx-308-6"
[...]
* Deployments duration per number of concurrents deployments (seconds)
conc. depl.    min       avg       max       stddev
1 (308n * 1)   521       553       701       66.21
2 (154n * 2)   408       459       701       96.78
4 (77n * 4)    337       374       413       8.77
8 (38n * 8)    37        333       511       58.70

* Failures per number of concurrents deployments
conc. depl.    min       avg       max       stddev    nodes
1 (308n * 1)   1         5.17      10        3.34      1.68%
2 (154n * 2)   0         3.67      10        6.45      2.38%
4 (77n * 4)    0         1.71      6         3.53      2.22%
8 (38n * 8)    0         0.94      5         3.79      2.47%
```

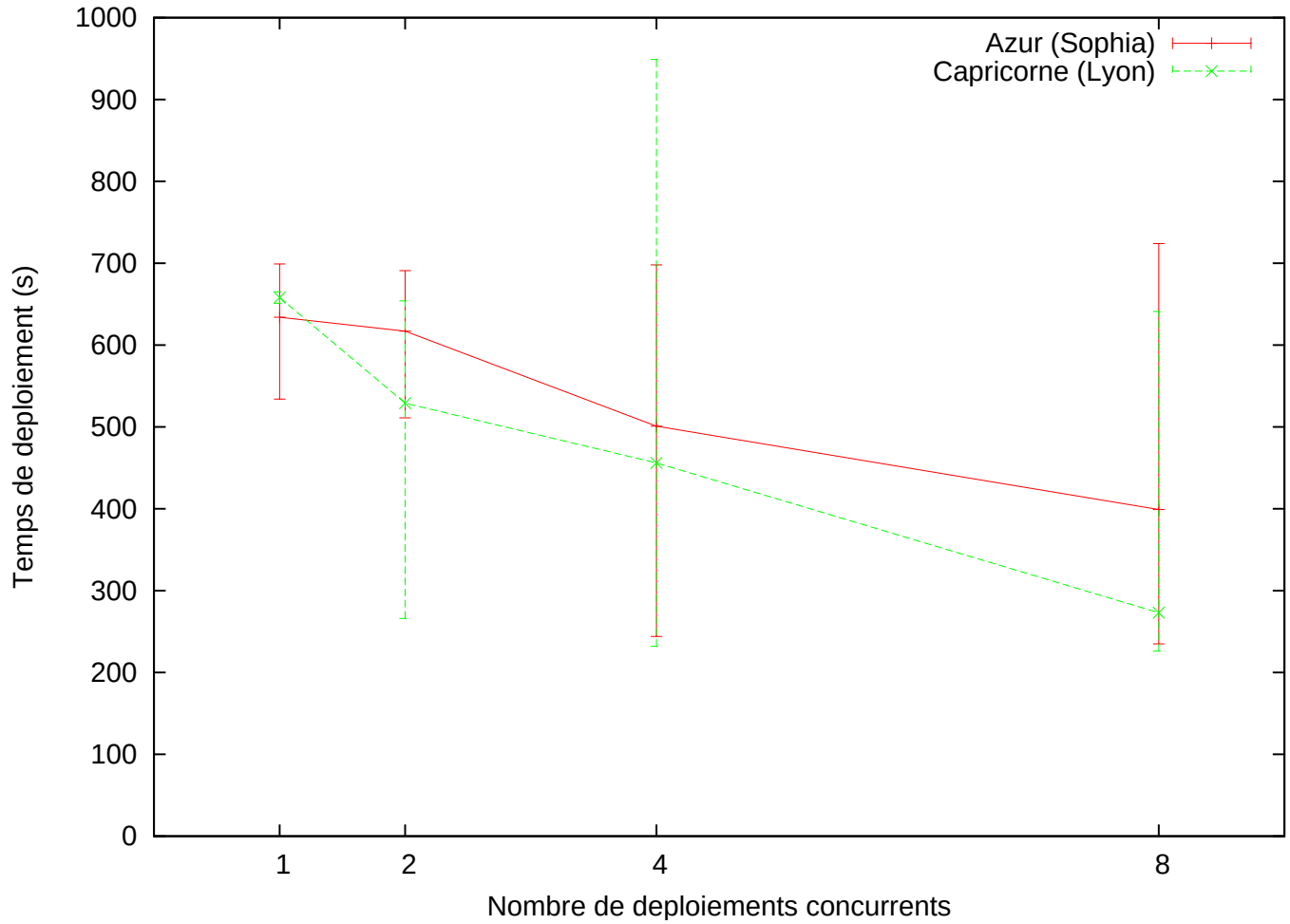
Arguments de la commande

- Arguments `-m`, `-e`, `-p` comme KADEPLOY
- `-n` précise le nombre de tests
- `-o` précise le répertoire où seront écrit les *logs*

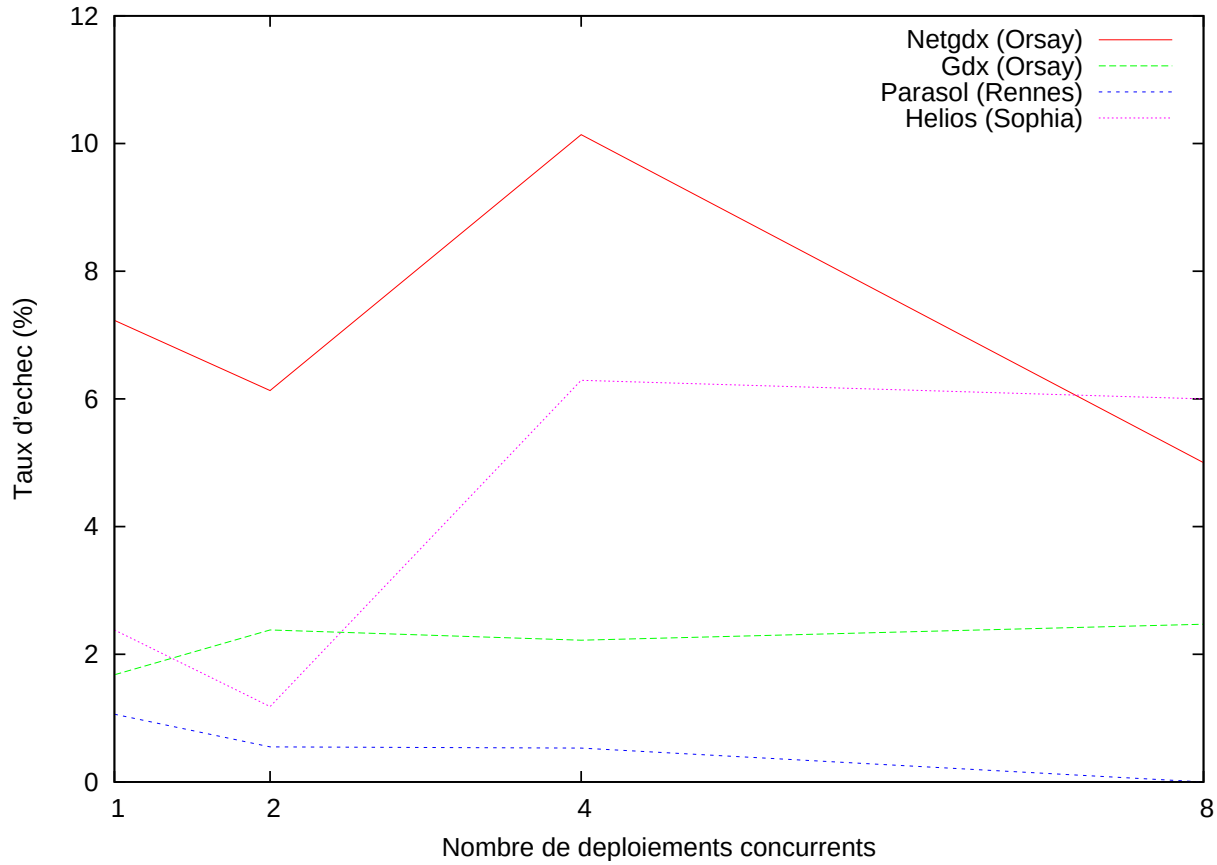
Temps de déploiement : Sun V20Z



Temps de déploiement : IBM e325



Graphique des causes d'échec par nombre de déploiements concurrents



Gdx (Orsay)

Erreurs rencontrées

- `Preinstall failed on node`
- `simple remote execution failed on node`
- `not there on first check`
- `mount of /dev/sda3 failed on node`

Explication possible

Cas limite dû au nombre de noeuds (308) ?

Netgdx (Orsay)

Erreurs rencontrées

- `not there on last check`
- `Preinstall failed on node`

Explication possible

Seconde carte réseau changeant suivant les noeuds ?

Parasol (Rennes)

Erreur rencontrée

```
Not there on last check
```

Explications possibles

- Problème de réglage du *timeout* de KADEPLOY ?
- Problème de changement de l'ordre de *boot* sur deux des noeuds ?

Plan

Présentation générale

Objectifs

- Tests sur les environnements de références (noeuds et frontaux)
  - Vérifier la présence de programmes et leurs versions
  - Permettre l'exécution de scripts de test
- Basé sur un script *shell* pré-existant
  - Plus modulaire
  - Plus extensible

Implémentation

- Script écrit en RUBY
- Rapport au format texte/YAML
- Fichier de spécification de test en YAML

Présentation générale

Objectifs

- Tests sur les environnements de références (noeuds et frontaux)
  - Vérifier la présence de programmes et leurs versions
  - Permettre l'exécution de scripts de test
- Basé sur un script *shell* pré-existant
  - Plus modulaire
  - Plus extensible

Implémentation

- Script écrit en RUBY
- Rapport au format texte/YAML
- Fichier de spécification de test en YAML

Exemple d'utilisation I

Exemple de fichier de spécification : `example.yaml`

```
title: lang_script
description: Vérification des interpréteurs pour les langages de script
version: 0.1
target: node
---
target: frontend
cmd: ruby
# Output: ruby 1.8.6 (2007-03-13 patchlevel 0) [i486-linux]
cmd-version: ruby -v
version-min: 1.8.4
version-max: 1.9
version-re: !ruby/regexp "/^ruby (.*) \(.*/"
---
target: all
cmd: python
cmd-version: python -V
version-min: 2.4.6
version-re: !ruby/regexp "/^Python (.*)/"
---
target: all
directory: examples/scripts
```

Exemple d'utilisation II

Exécution de KARAFON

```
$ ./bin/karafon-wrapper --target=frontend examples/example.yaml
OK        ruby                              found /usr/bin/ruby (1.8.4 <= 1.8.6 < 1.9)
FAILED    python                            Installed version is 2.4.4, \
                                              whereas it requires at least 2.4.6
OK        examples/scripts/example1.sh      status code 0
```

Rapport détaillé

```
##################### Test1 v0.1 (examples/example.yaml) ####################
==================================== ruby ====================================
OK: found /usr/bin/ruby (1.8.4 <= 1.8.6 < 1.9)
OUTPUT: ruby 1.8.6 (2007-03-13 patchlevel 0) [i486-linux]
================================== python ====================================
FAILED: Installed version is 2.4.4, whereas it requires at least 2.4.6
OUTPUT: Python 2.4.4
========================= directory: examples/scripts ========================
---------------------- examples/scripts/example1.sh ----------------------
OK: status code 0
OUTPUT: foo
```

Conclusion I

Problèmes de Grid'5000

- Problèmes au niveau de KADEPLOY
  - Fiabilité
  - Efficacité
- Environnements de référence obsolètes

Objectifs du stage

- Développement d'outils permettant d'améliorer la qualité de l'infrastructure de Grid'5000
  - KADEPLOY
  - Environnements de référence

Conclusion II

Travail effectué

- Script permettant de tester KADEPLOY (`karate`)
  - Rapport des problèmes rencontrés sur certains clusters
- Script permettant de vérifier les environnements de référence (`karafon`)

Objectifs futurs

- Automatisation des tests
- Génération de page web donnant les résultats