Conclusion atelier variant

Maria Bernard - INRA de Jouy en Josas

École de bioinformatique AVIESAN-IFB 2017

Bilan du pipeline de détection des SNP

Données de départ

```
├── genome
│   ├── Bos_taurus.UMD3.1.89_6.gtf
│   └── Bos_taurus.UMD3.1.dna.toplevel.6.fa
├── additional_data
│   ├── SRR1205992.g.vcf
│   ├── SRR1205992.g.vcf.idx
│   ├── SRR1262731_mpileup.vcf
│   ├── SRR1262731_mpileup.vcf.gz
│   └── SRR1262731_mpileup.vcf.gz.tbi
└── fastq
    ├── cahier.sh
    ├── cahier_SRR1205992_Ho.sh
    ├── SRR1205992_extract_R1.fq
    ├── SRR1205992_extract_R2.fq
    ├── SRR1262731_extract_R1.fq
    └── SRR1262731_extract_R2.fq
```

Bilan du pipeline de détection des SNP

Indexation du génome :

```
└── genome
    ├── Bos_taurus.UMD3.1.89_6.gtf
    ├── Bos_taurus.UMD3.1.dna.toplevel.6.fa
    ├── Bos_taurus.UMD3.1.dna.toplevel.6.dict
    ├── Bos_taurus.UMD3.1.dna.toplevel.6.fa.fai
    ├── Bos_taurus.UMD3.1.dna.toplevel.6.fa.amb
    ├── Bos_taurus.UMD3.1.dna.toplevel.6.fa.ann
    ├── Bos_taurus.UMD3.1.dna.toplevel.6.fa.bwt
    ├── Bos_taurus.UMD3.1.dna.toplevel.6.fa.pac
    └── Bos_taurus.UMD3.1.dna.toplevel.6.fa.sa
```

bwa index, picard CreateSequenceDictionary, samtools index

Bilan du pipeline de détection des SNP

Partie 1 : qualité et mapping

fastqc
cutadapt
bwa mem
samtools

```
└── part1_qual_align
    ├── FASTQC_rawdata
    │   ├── SRR1262731_extract_R1_fastqc.html
    │   ├── SRR1262731_extract_R1_fastqc.zip
    │   ├── SRR1262731_extract_R2_fastqc.html
    │   └── SRR1262731_extract_R2_fastqc.zip
    ├── SRR1262731_extract_R1.trimmed.fq
    ├── SRR1262731_extract_R2.trimmed.fq
    ├── SRR1262731_extract_trimming_stats.txt
    ├── FASTQC_trimmed
    │   ├── SRR1262731_extract_R1.trimmed_fastqc.html
    │   ├── SRR1262731_extract_R1.trimmed_fastqc.zip
    │   ├── SRR1262731_extract_R2.trimmed_fastqc.html
    │   └── SRR1262731_extract_R2.trimmed_fastqc.zip
    ├── SRR1262731_extract.trimmed.sort.bam
    ├── SRR1262731_extract.trimmed.sort.bam.bai
    └── SRR1262731_extract.trimmed.sort.flagstat
```

Bilan du pipeline de détection des SNP

Partie 2 : filtre sur le mapping et calling

```
└── part2_var_calling
    ├── metrics_md.txt
    ├── SRR1262731_extract.trimmed.sort.md.bam
    ├── SRR1262731_extract.trimmed.sort.md.onTarget.bam
    ├── SRR1262731_extract.trimmed.sort.md.onTarget.bam.bai
    └── SRR1262731_extract.trimmed.sort.md.onTarget.dp.txt
```

picard MarkDuplicates, bedtools, samtools, gatk

Solution 1 sur 1 seul échantillon

```
├── SRR1262731.vcf
└── SRR1262731.vcf.idx
```

OU

Solution 2 sur plusieurs échantillons

```
├── SRR1262731.g.vcf
├── SRR1262731.g.vcf.idx
├── SRR1205992_SRR1262731.g.vcf
├── SRR1205992_SRR1262731.g.vcf.idx
├── SRR1205992_SRR1262731.vcf
└── SRR1205992_SRR1262731.vcf.idx
```

Bilan du pipeline de détection des SNP

Partie 3 : filtre des variants et annotation

```
├── part3_filt_anno
│   ├── Hard_Filtering
│   │   ├── SRR1205992_SRR1262731_SNP.vcf
│   │   ├── SRR1205992_SRR1262731_SNP.vcf.idx
│   │   ├── SRR1205992_SRR1262731_SNP_prefiltered.vcf
│   │   ├── SRR1205992_SRR1262731_SNP_prefiltered.vcf.idx
│   │   ├── SRR1205992_SRR1262731_SNP_filtered.vcf
│   │   └── SRR1205992_SRR1262731_SNP_filtered.vcf.idx
│   ├── Filtre_snpsift
│   │   ├── SRR1205992_SRR1262731_DP20_QUAL30.vcf
│   │   └── SRR1205992_SRR1262731_DP20_QUAL30.HET.vcf
│   └── Comparaison_variants
│       ├── SRR1262731_mpileup_cut.tab
│       ├── SRR1262731_gatk_cut.tab
│       └── common_variants.vcf
```

GATK
SnpSift

Bilan du pipeline de détection des SNP

Partie 3 : filtre des variants et annotation

snpEff
Snpsift

```
└── part3_filt_anno
    └── Annotation_variants
        ├── mon_fichier_snpeff.config
        ├── UMD3.1.db
        ├── UMD3.1
        │   ├── genes.gtf -> ../../../genome/Bos_taurus.UMD3.1.89_6.gtf
        │   ├── sequences.fa ->
../../../genome/Bos_taurus.UMD3.1.dna.toplevel.6.fa
        │   └── snpEffectPredictor.bin
        ├── SRR1205992_SRR1262731.snpEff.vcf
        ├── fichier_snpeff_resulat.csv
        ├── fichier_snpeff_resulat.genes.txt
        ├── fichier_snpeff_resulat.html
        ├── SRR1205992_SRR1262731.snpEff.NO_SYN.NO_INT.vcf
        ├── SRR1205992_SRR1262731.snpEff.NO_SYN.NO_INT.coding.vcf
        └── SRR1205992_SRR1262731.snpEff.NO_SYN.NO_INT.coding.missense.vcf
```

Atelier variant - variant tour de table des données

Les questions qui pourraient moduler le pipeline d'analyse

Disponibilité d'un génome de référence ?

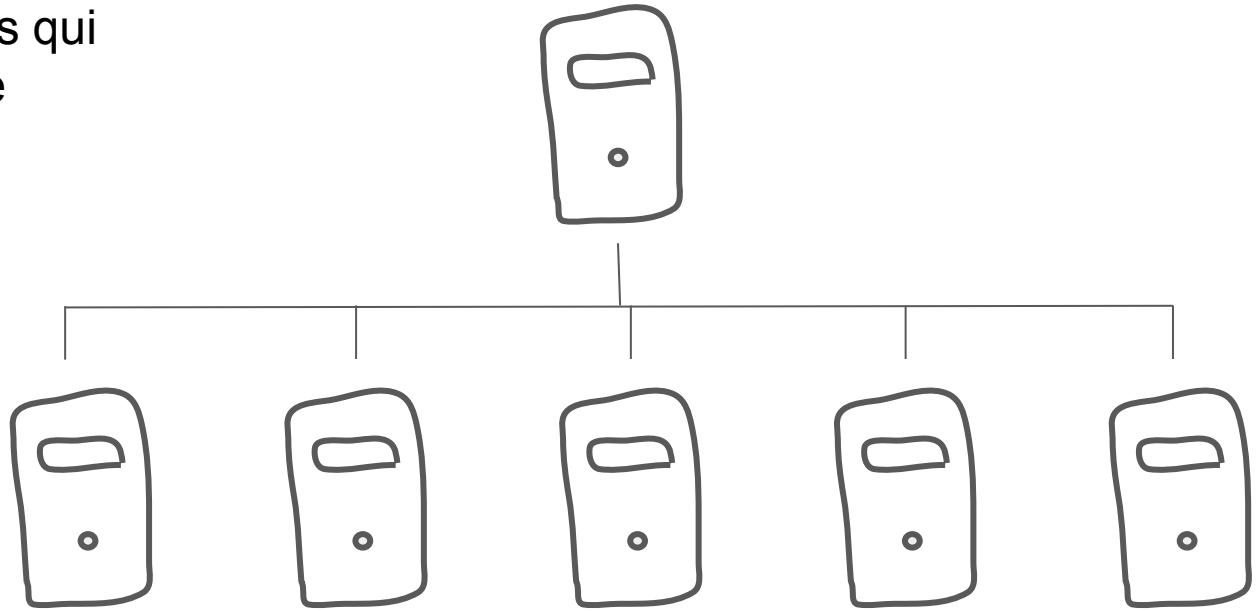
Séquençage Illumina ou autre ?

Séquençage complet ou ciblé ?

Design expérimental : beaucoup d'échantillons sur grand génome complet ? peu d'échantillons sur région ciblée ?

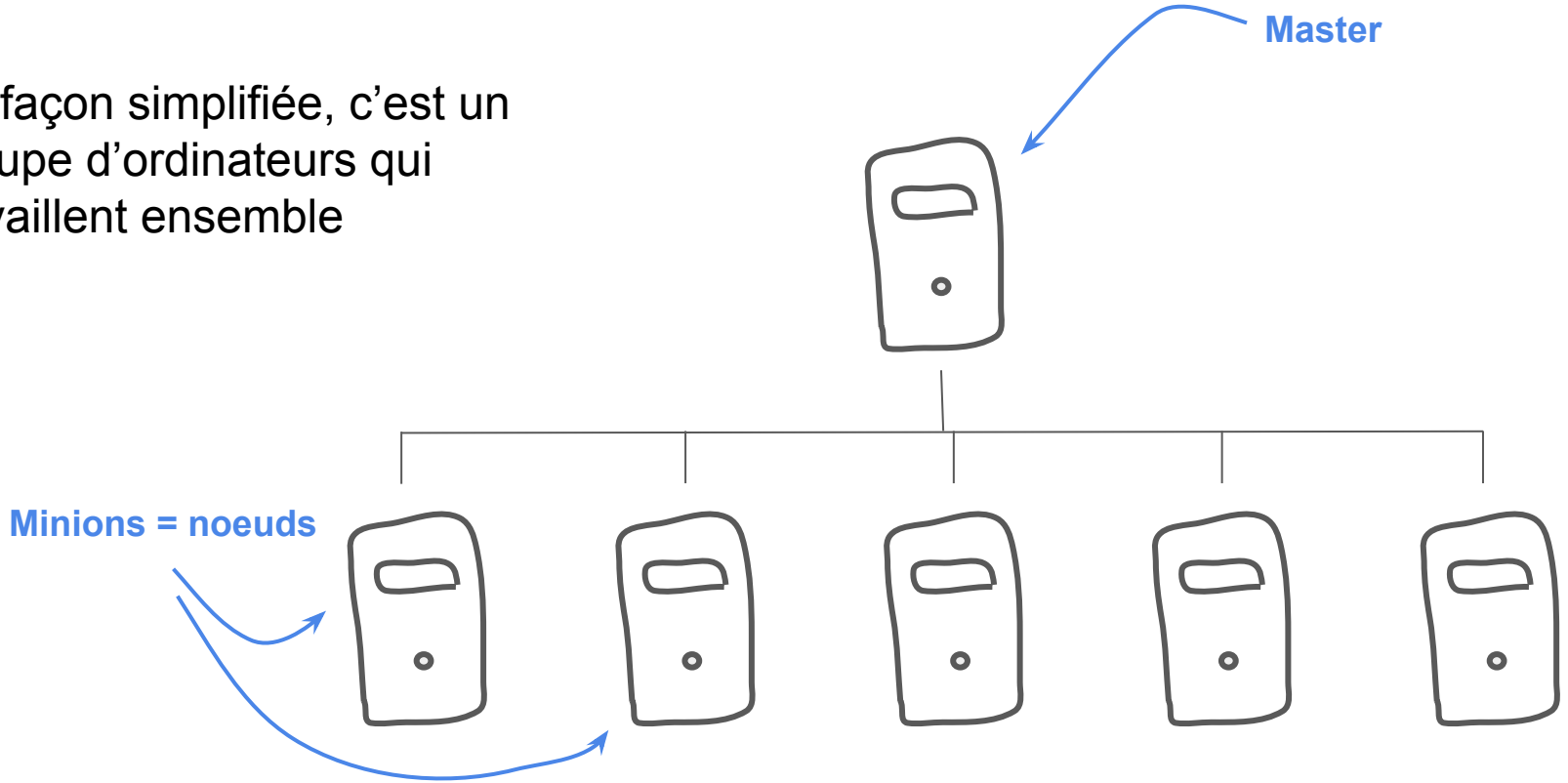
Un cluster de calcul: qu'est ce que c'est?

De façon simplifiée, c'est un groupe d'ordinateurs qui travaillent ensemble



Un cluster de calcul: qu'est ce que c'est?

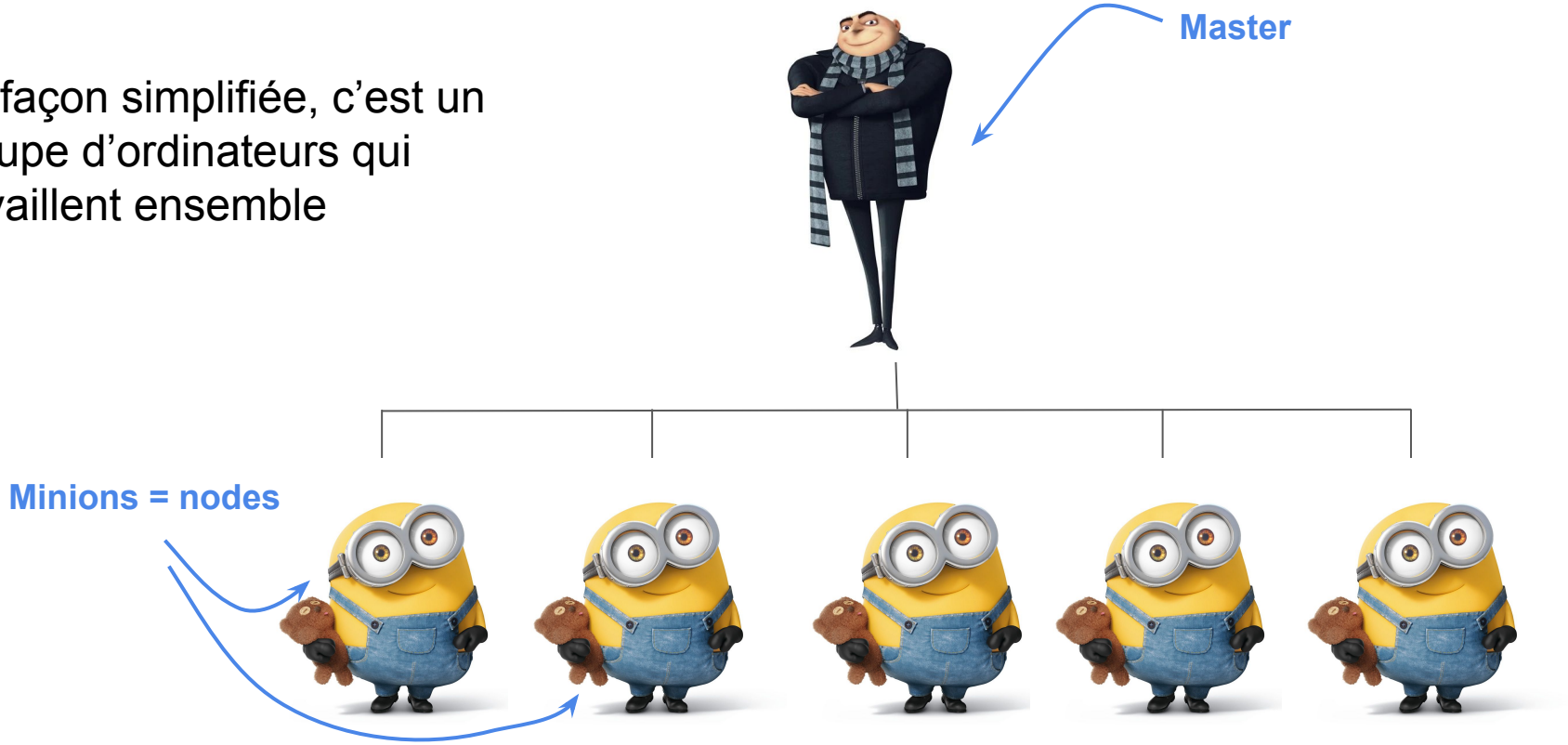
De façon simplifiée, c'est un groupe d'ordinateurs qui travaillent ensemble



Hum…

Un cluster de calcul: qu'est ce que c'est?

De façon simplifiée, c'est un groupe d'ordinateurs qui travaillent ensemble



Un cluster de calcul: qu'est ce que c'est?

Master
- CPUs +
- Memory +

Nodes:
- CPUs ++++
- Memory ++++

Un cluster de calcul: qu'est ce que c'est?

Master:

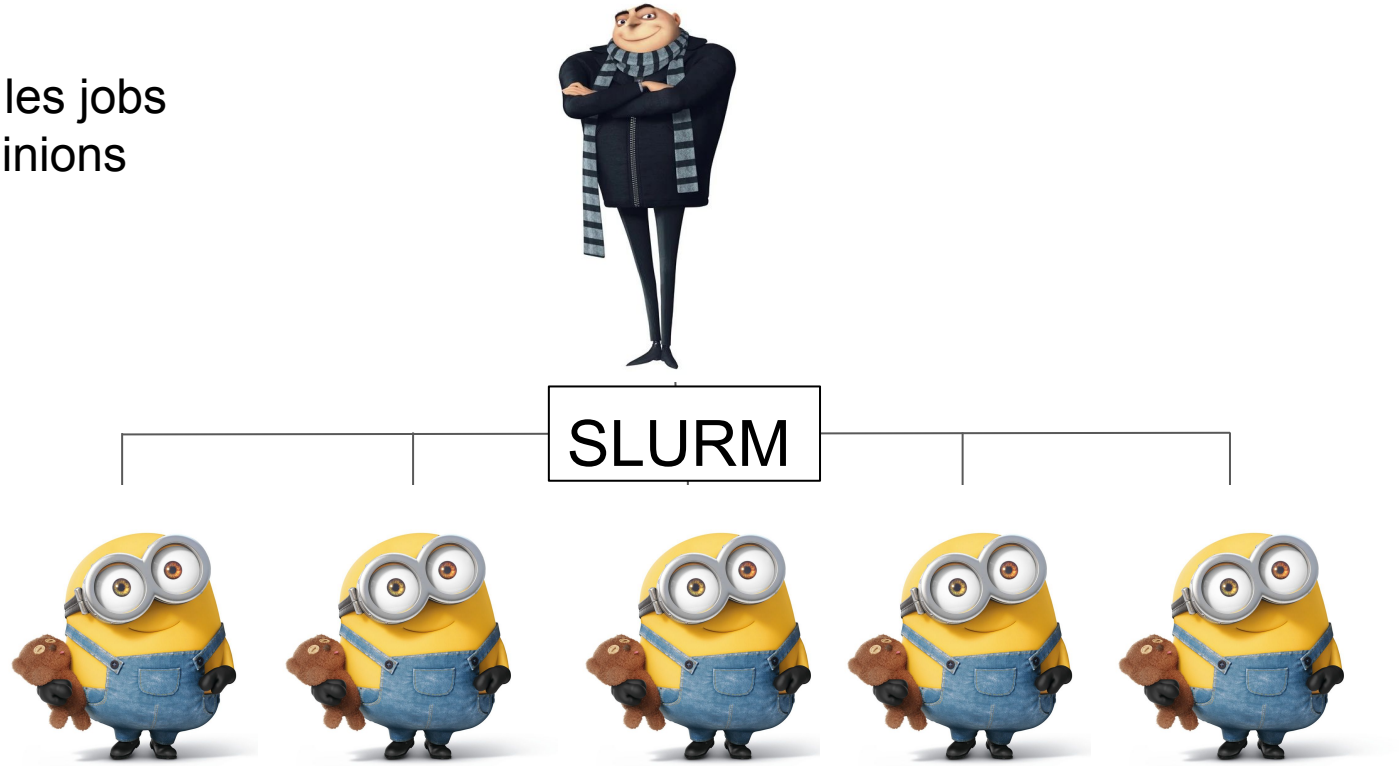
- **Exécute des petites commandes bash. Ex: cd, mv, ls…**
- **Est utilisé pour envoyer des jobs aux Minions**

Noeud:

- **Exécute les jobs du Master**

Un cluster de calcul: comment envoyer des jobs aux Minions?

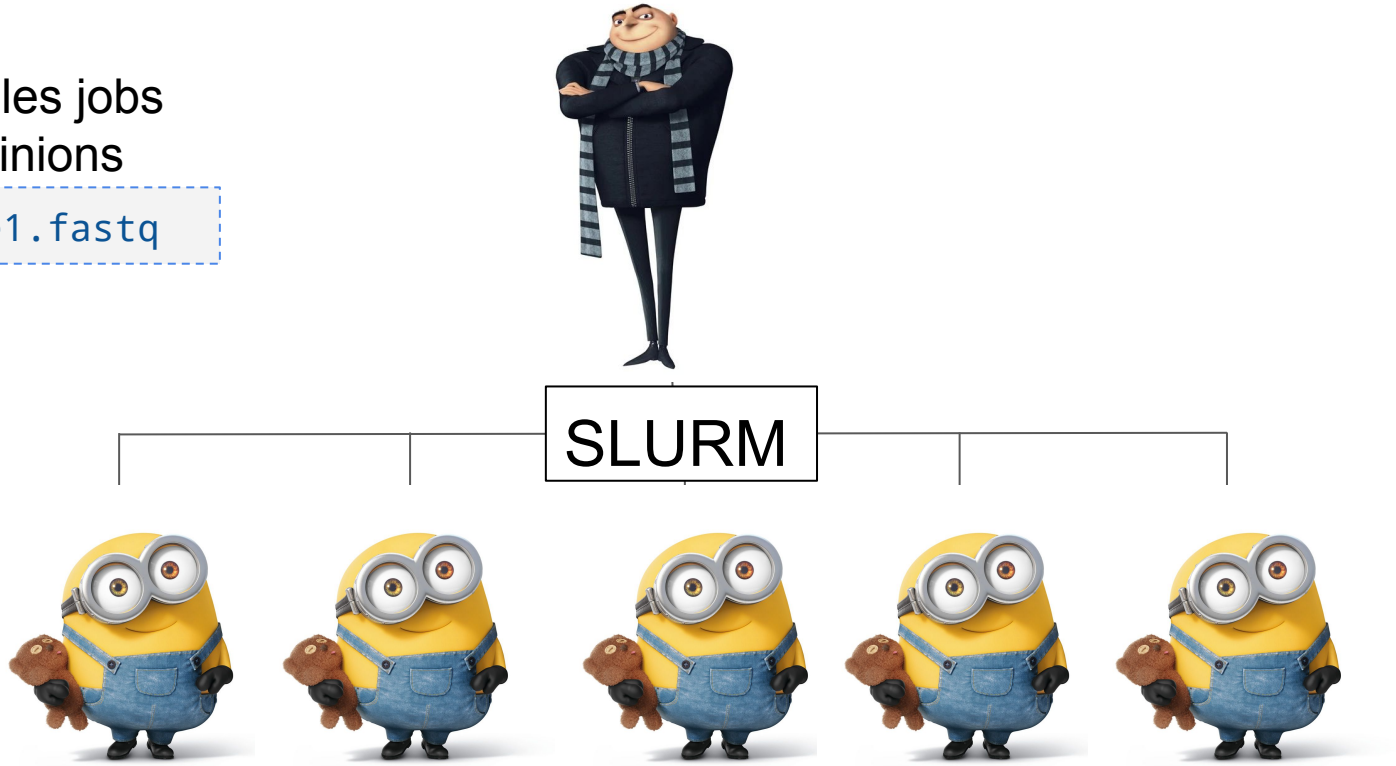
Slurm dispatche les jobs sur les noeuds/Minions



Un cluster de calcul: comment envoyer des jobs aux Minions?

Slurm dispatche les jobs sur les noeuds/Minions

```
srun fastqc file1.fastq
```



Un cluster de calcul: comment envoyer des jobs aux Minions?

Slurm dispatche les jobs sur les noeuds/Minions

```
srun fastqc file1.fastq
```

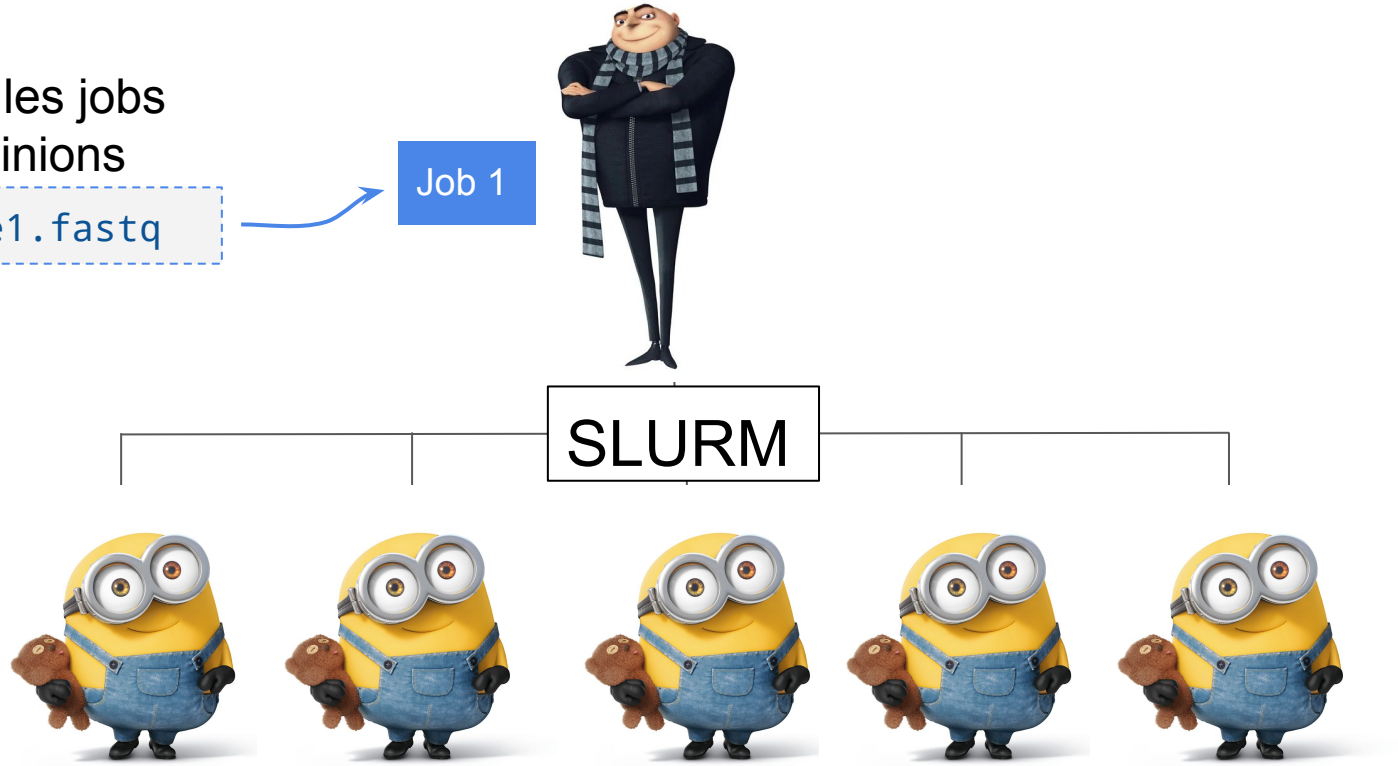
- Ligne de commande à lancer
- Envoi de la commande avec **srun**

SLURM

Un cluster de calcul: comment envoyer des jobs aux Minions?

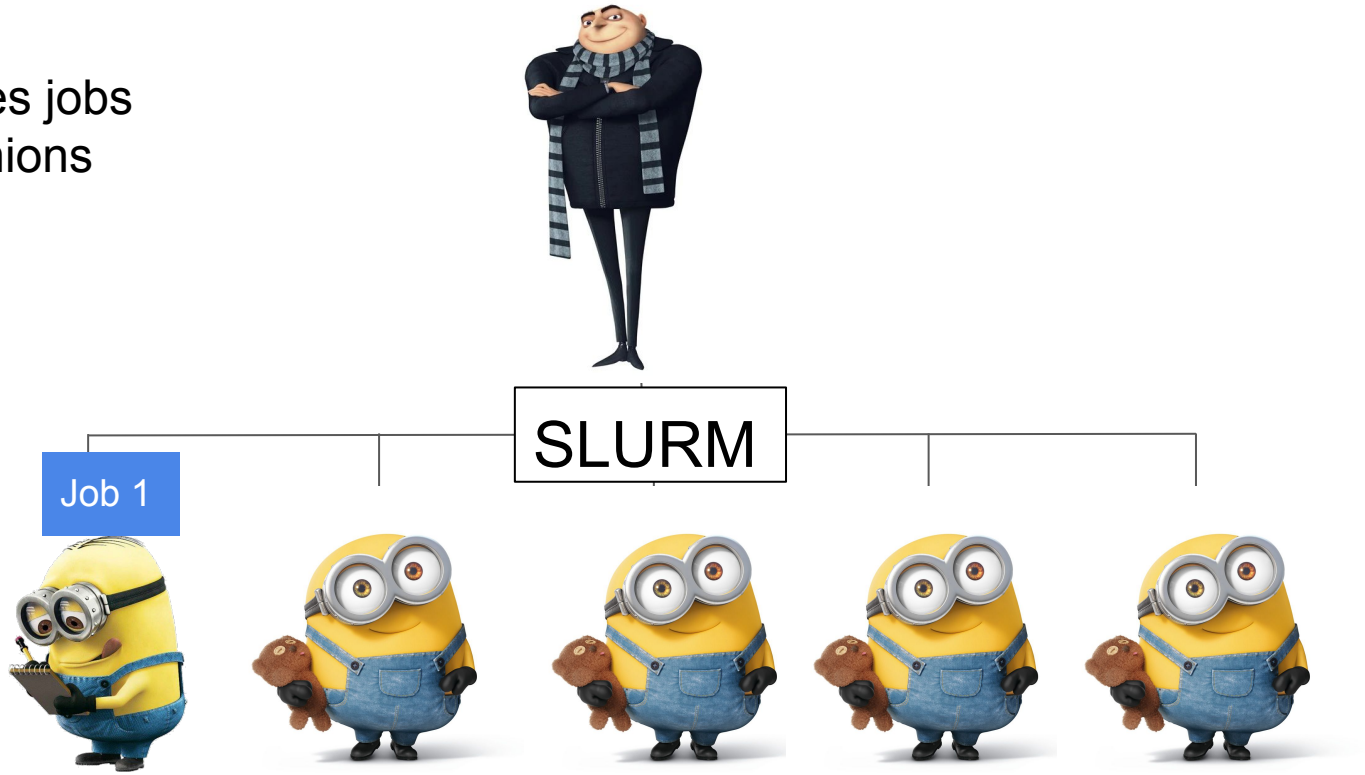
Slurm dispatche les jobs sur les noeuds/Minions

```
srun fastqc file1.fastq
```



Un cluster de calcul: comment envoyer des jobs aux Minions?

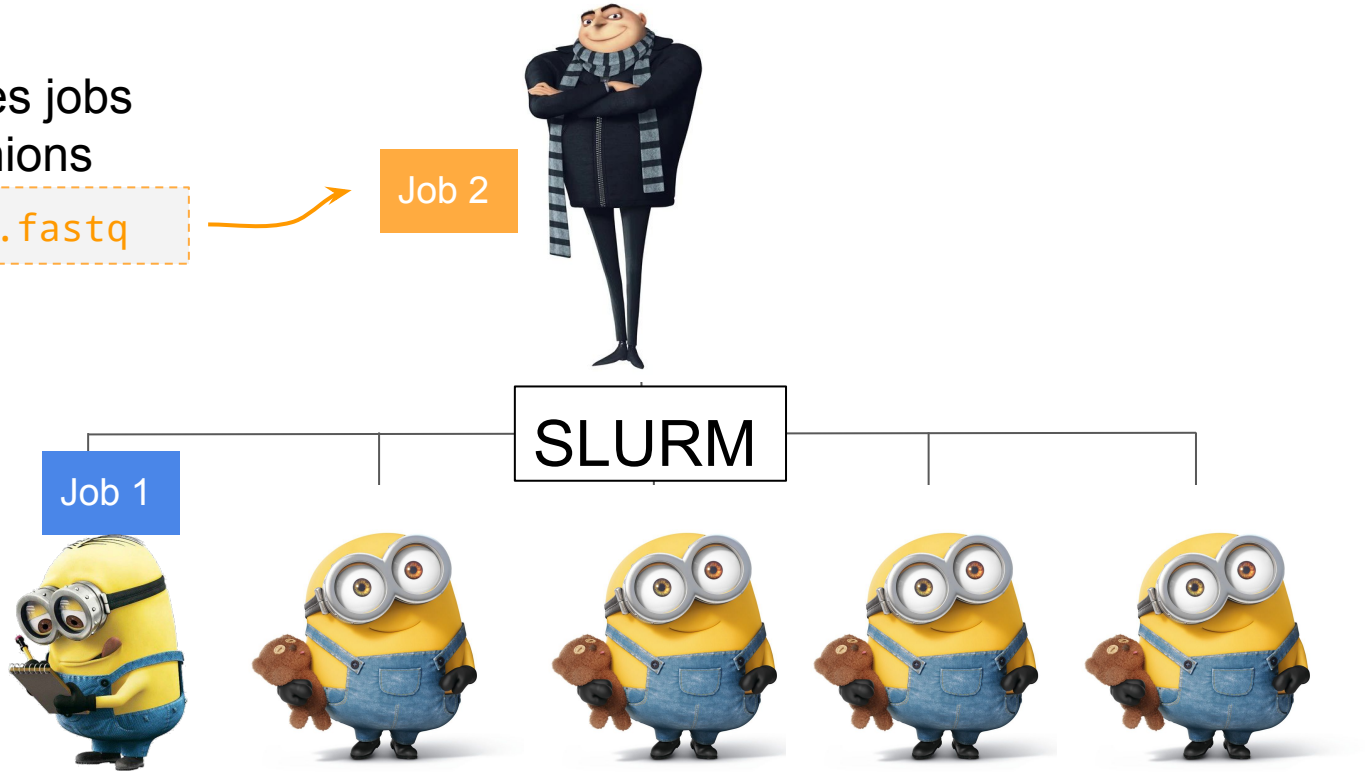
Slurm dispatche les jobs sur les noeuds/Minions



Un cluster de calcul: comment envoyer des jobs aux Minions?

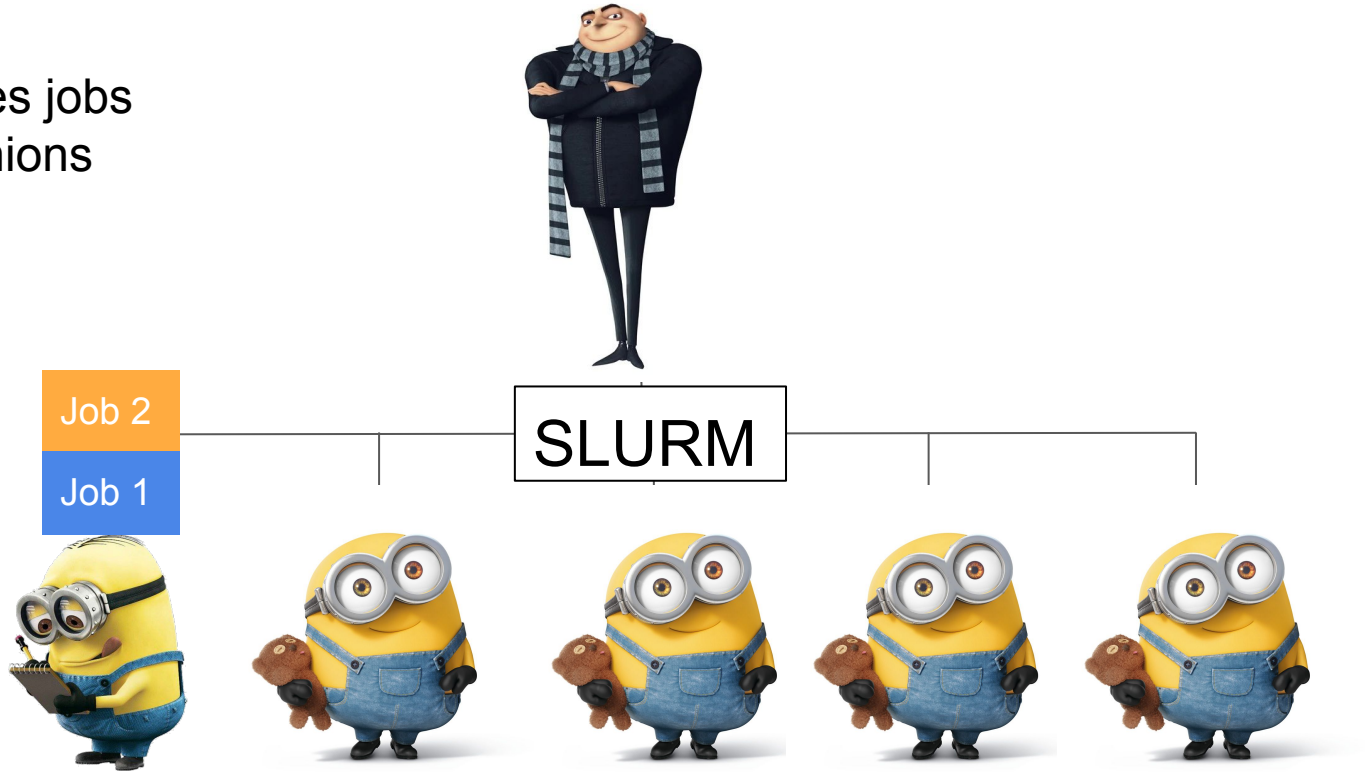
Slurm dispatche les jobs sur les noeuds/Minions

```
srun fastqc file2.fastq
```



Un cluster de calcul: comment envoyer des jobs aux Minions?

Slurm dispatche les jobs sur les noeuds/Minions



Un cluster de calcul: comment envoyer des jobs aux Minions?

Slurm dispatche les jobs sur les noeuds/Minions

```
sbatch  cahier.sh
```

Job 2

Job 1

SLURM

Un cluster de calcul: comment envoyer des jobs aux Minions?

Slurm dispatche les jobs sur les noeuds/Minions

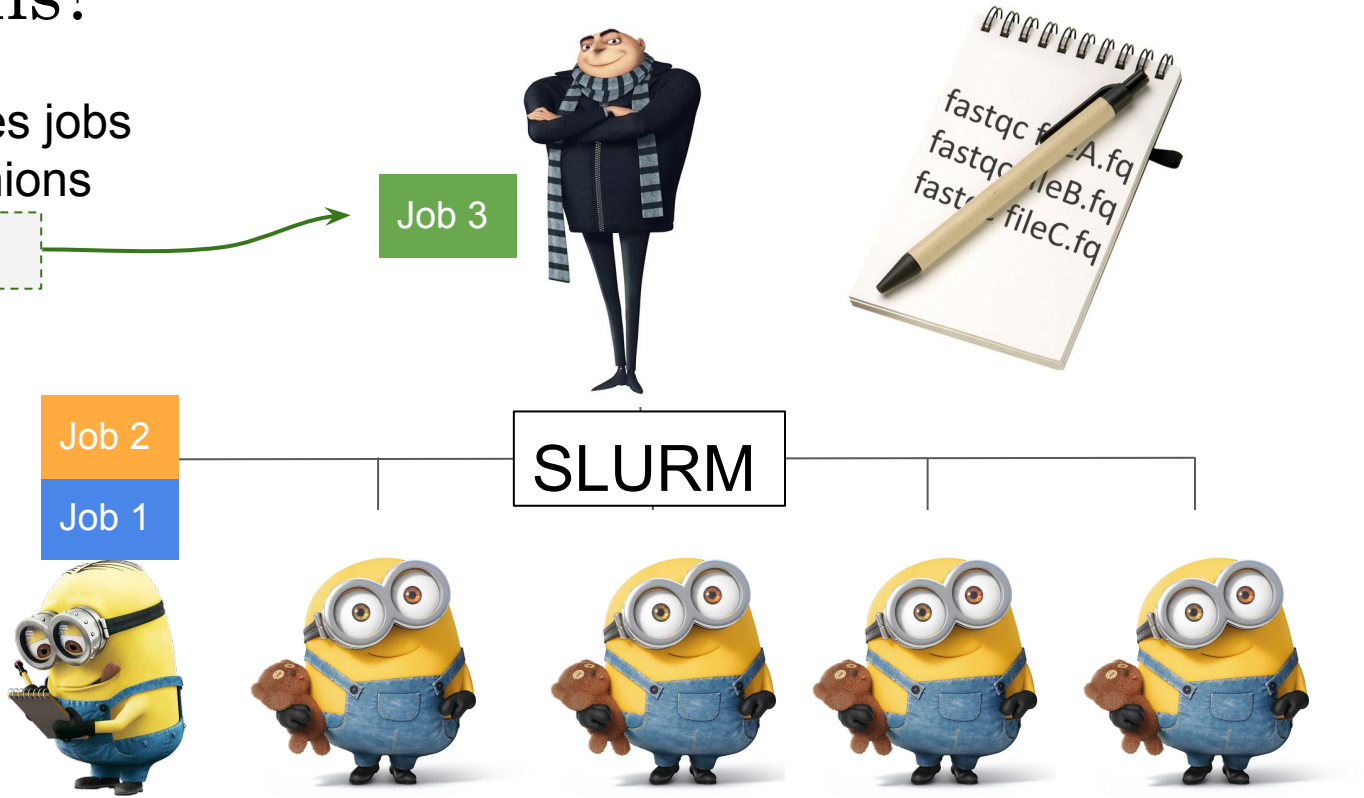
```
sbatch cahier.sh
```

- Script à soumettre
- Envoi du script with **sbatch**

Un cluster de calcul: comment envoyer des jobs aux Minions?

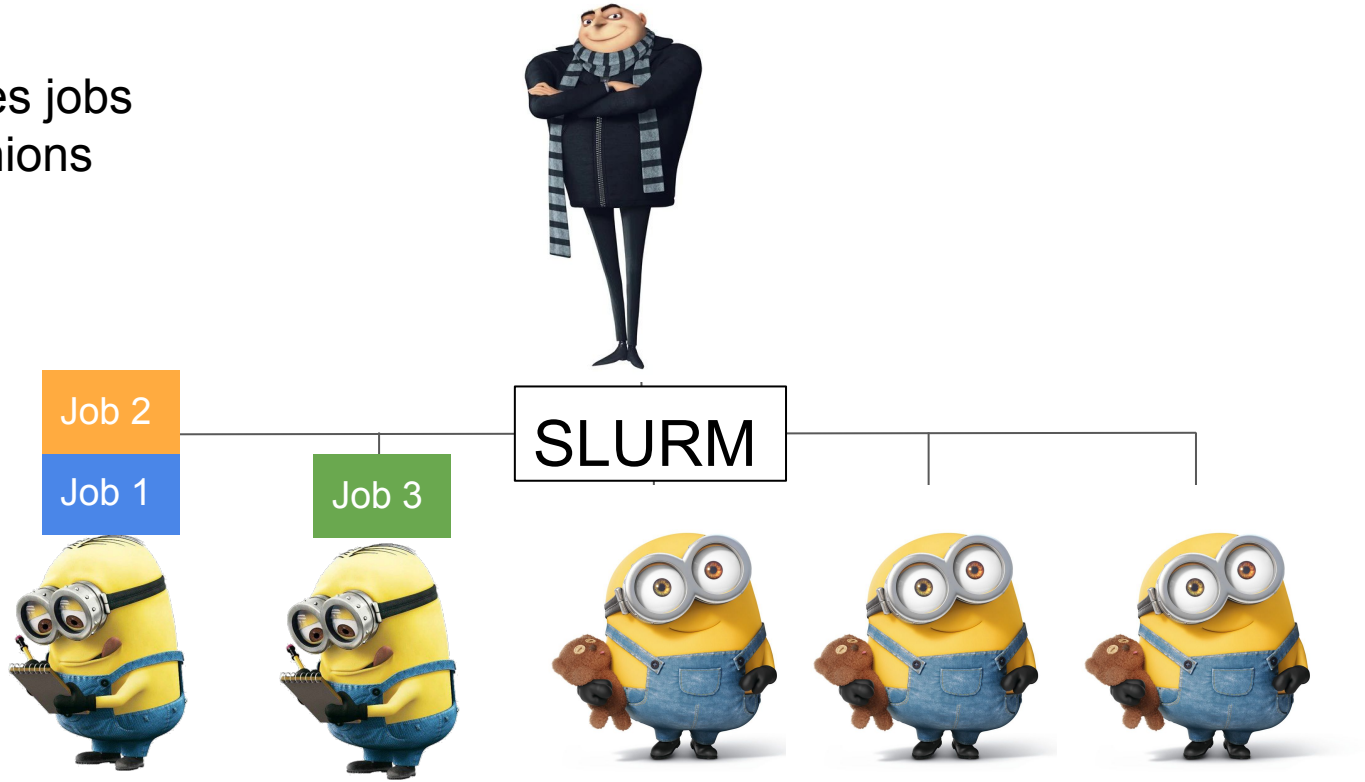
Slurm dispatche les jobs sur les noeuds/Minions

```
sbatch  cahier.sh
```



Un cluster de calcul: comment envoyer des jobs aux Minions?

Slurm dispatche les jobs sur les noeuds/Minions



Un cluster de calcul: comment envoyer des jobs aux Minions?

srun

- Les résultats s'affichent **dans le terminal**
- Vous **ne reprenez la main** qu' à la fin de la commande

⇒ ideal pour **tester un outil**

sbatch

- Les résultats sont stockés **dans un fichier**
- Vous **reprenez la main** dès le lancement de la commande sbatch

⇒ commande **la plus utilisée**

Un cluster de calcul: comment envoyer des jobs aux Minions?

- Certains calculs peuvent demander plus de ressources:
  - En CPU pour paralléliser les calcul

```
bwa mem -t 4 genome.fa reads_R1.fq reads_R2.fq  > aln.sam
```

  - En mémoire si vous travaillez sur un grand génome.

Un cluster de calcul: comment envoyer des jobs aux Minions?

- Ajouter des options aux commandes **srun** ou **sbatch**
  - **-c** pour le nombre de CPU
  - **--mem** pour la mémoire

Exemple avec 4 CPUs et 10G de mémoire pour chaque CPU

```
srun -c 4 --mem=10G 'bwa mem -t 4 genome.fa reads_R1.fq reads_R2.fq  > aln.sam'
```

ou

```
sbatch -c 4 --mem=10G cahier_BWA.txt
```

Comment utiliser un cluster de calcul: résumé

- **Se connecter sur le noeud Master** et rester travailler dessus
- Pour les **commandes de base** (cd, ls, mv, mkdir…), lancer les directement sur **"Master"**
- Pour tout le reste, notamment **les outils bioinformatiques**, envoyer les lignes de commandes avec srun ou sbatch pour que les jobs soient lancés sur un **noeud du cluster**
- **Adapter les ressources** (CPU : `-c`, mémoire : `--mem=`) à votre contexte et à l'infrastructure de calcul utilisée

TIPS

- **Garder toutes les lignes de commandes lancées.** Vous pouvez par exemple créer un fichier texte dans lequel vous les écrivez. *Sinon comment se souvenir comment vous avez généré les fichiers*
- **Donner des noms explicites** à vos fichier. *Comment se rappeler ce que le fichier test.txt file contient.*
- **Donner la bonne extension** à vos fichiers. *Un fichier bed a comme suffixe ".bed", un fichier bam le suffixe ".bam"...*
- **Créer des répertoires!!** *Ne pas garder des tonnes de fichiers dans un seul répertoire. Il y a toujours un moyen de mieux s'organiser*