CCIN2P3

cnrs

Centre de Calcul
de l'Institut National de Physique Nucléaire
et de Physique des Particules

BBQ – Beautiful Batch Query

Journées Informatiques - Novembre 2022

Guillaume Cochard - CC-IN2P3

Les clusters du CC-IN2P3

- Grille de calcul (WLCG)
  - Une dizaine d'utilisateurs ("grosses" expériences type Atlas ou Alice)
  - Ordonnanceur : HTCondor
  - **765 workers** pour **37 000 CPUs**
  - Jusqu'à **50 000 jobs** « standardisés » par jour

- Ferme locale
  - Plusieurs centaines d'utilisateurs de l'IN2P3
  - Ordonnanceur : Slurm (anciennement UGE/Grid Engine)
  - **423 workers** pour **22500 CPUs** et **80 GPUs**
  - Jusqu'à **120 000 jobs** variés par jour

Situation initiale

- Gros besoin de monitoring
  - Nagios : monitoring d’incident
  - Grafana : suivi dans le temps
  - ?? : chiffres, informations et configuration en instantané ou dans le passé

- Historiquement, plusieurs scripts permettant de suivre les clusters
  - Occupation des clusters
  - Classement des utilisateurs ayant le plus de jobs
  - Répartition des jobs par workers
  - Jobs en attente

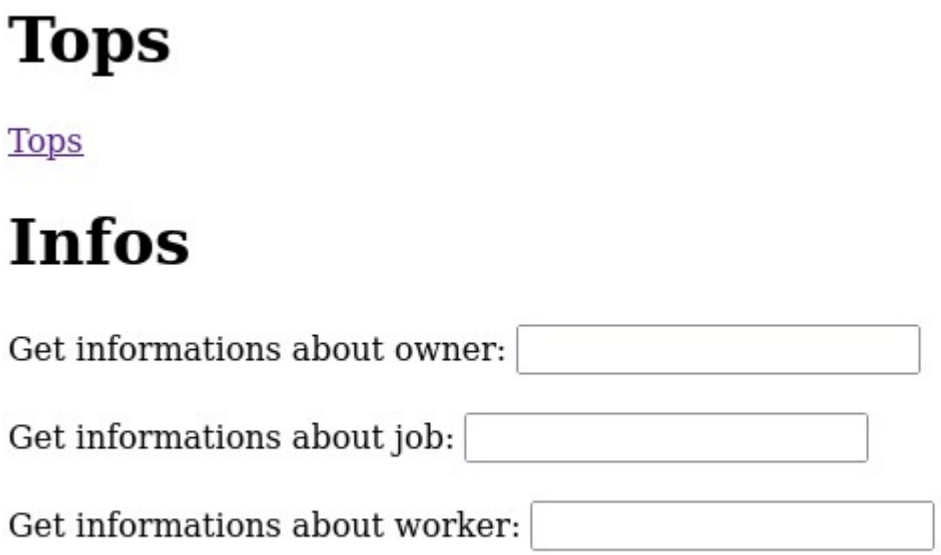
Situation initiale

- Plusieurs sources de données
  - Condor et UGE : parsing de commandes
  - UGE : BDD MySQL mise à jour toutes les 2 minutes
- Utilisés uniquement par les administrateurs des clusters
- Les scripts sont peu pratiques
  - Hétérogénéité
  - Nécessitent de se connecter sur des machines
  - Fonctionnalités limitées (notamment par l'interface)

La première idée

- Pourquoi ne pas faire une interface web ?
  - Corrige les défauts précédents
  - Plus user-friendly
  - Beaucoup de potentiel
- Les questions
  - Comment la déployer ?
  - Compatibilité des sources de données
  - Surcouche technique pas trop importante ?
- La réponse : un POC

Version 0

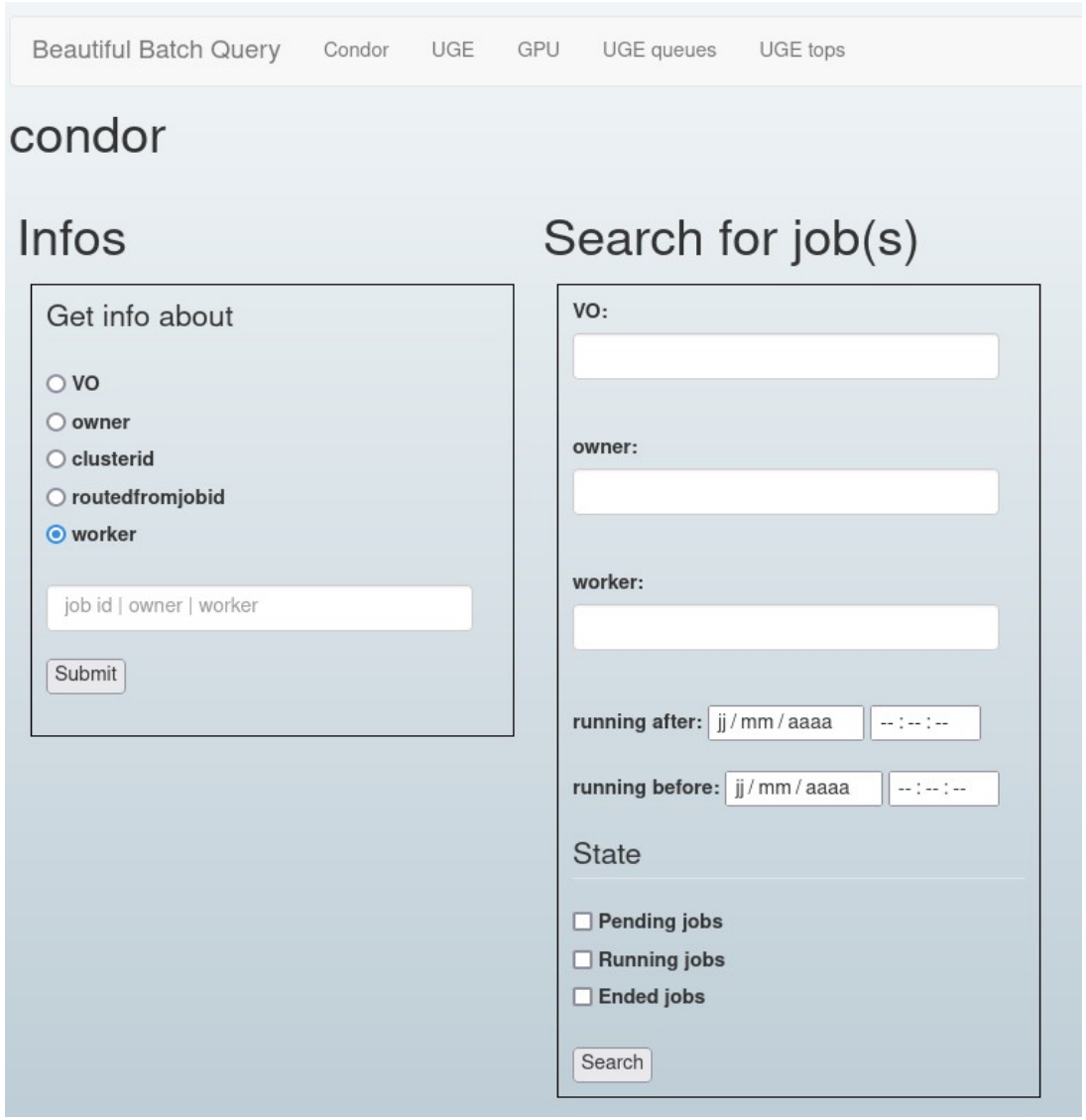


- Développé en Python
  - Flask : simple, puissant, efficace
  - Templating avec Jinja2
  - Requête SQL directes

- Très simple, mais efficace
  - Première conclusion : c’est une bonne idée
  - Première remarque : on pourrait le rendre plus beau

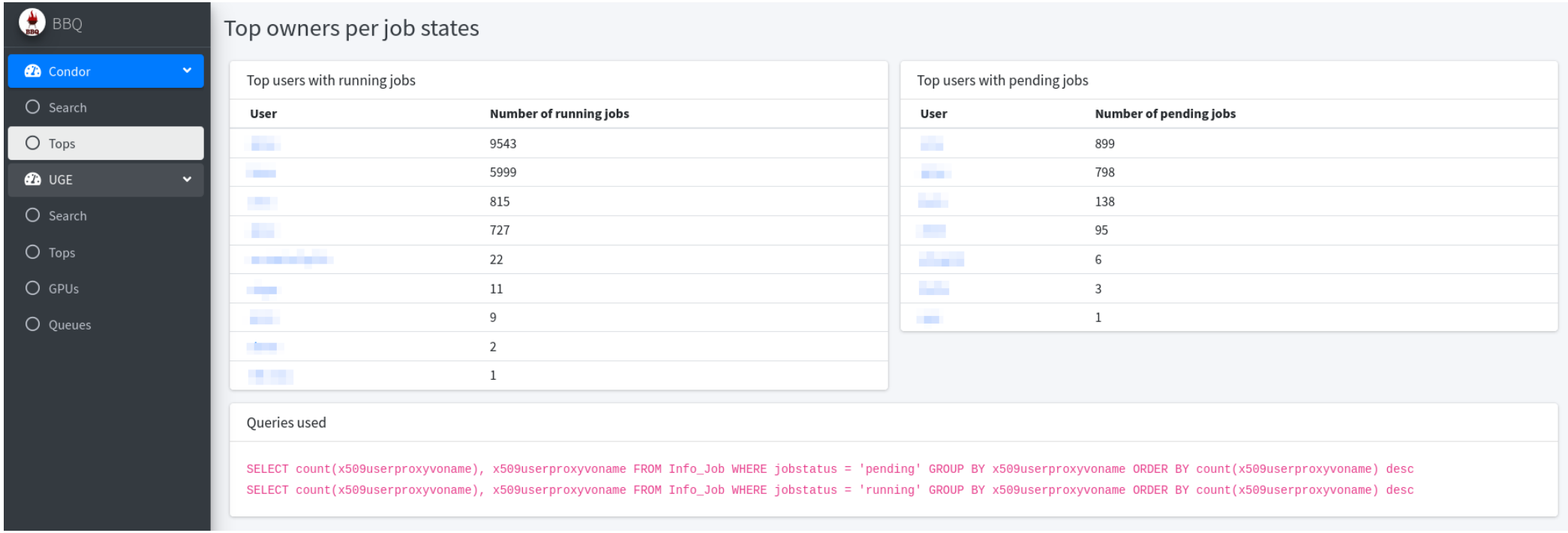
- Validation du POC

Version 1 : ajout de condor



- “Amélioration” de l’interface
- Ajouts de fonctionnalités pour UGE
- Création d’une BDD pour Condor
  - Travail en amont
  - Séparation entre récupération/parsing et affichage
  - Nécessaire pour l’historique
- Choix de rendre BBQ le plus agnostique possible entre les clusters en interne

Version 1 : mise à jour de l'interface



- Utilisation de AdminLTE / Bootstrap
  - Pourquoi réinventer la roue ?

Fin de la première étape

- Développement à temps partiel pendant 4 mois

- Beaucoup de changements depuis le début (interface et données)
- Plusieurs nouvelles idées de fonctionnalités

- Mais la partie technique a peu changé : requêtes SQL directes
  - Bricolage pour construire les requêtes
  - Pas adapté aux évolutions futures

- Arrivée de Slurm en remplacement de UGE/Grid Engine

Version 2

- Cluster UGE → Slurm
  - Fin de la compatibilité entre BBQ Condor et BBQ Slurm

- Passage à SQLAlchemy
  - Les requêtes ne sont plus faites “à la main”
  - Beaucoup plus simple à maintenir
  - Très puissant

- Mais traitements locaux pour limiter les requêtes SQL
  - Code complexe à base de multiples Dictionnaires imbriqués
  - Équivalent de count() et de group by
  - Moins performant que du SQL

Pandas : la solution ?

- Passage à Pandas pour les traitements locaux
  - Les données sont traitées dans des matrices
  - Corrige une partie de la complexité (plus de dictionnaires)
  - Solution pythonesque

- Mais Pandas n’était pas la solution
  - Problème de performances une fois de nouvelles fonctionnalités ajoutées
  - C’est finalement assez complexe aussi

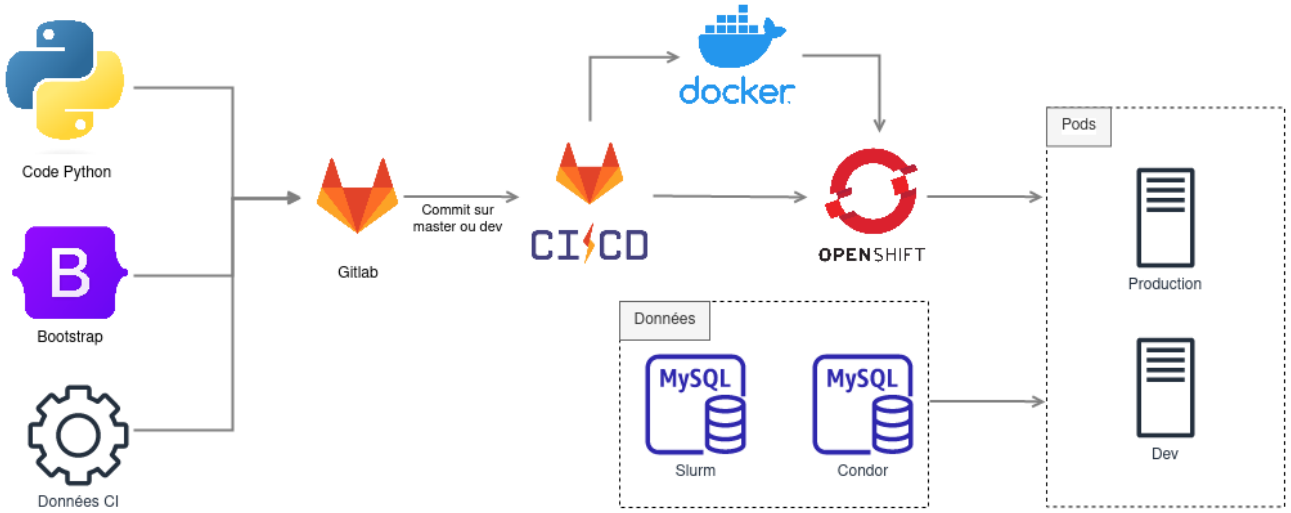
La vraie solution : SQL

- Python ne fait pas tout
- SQL est extrêmement puissant et performant

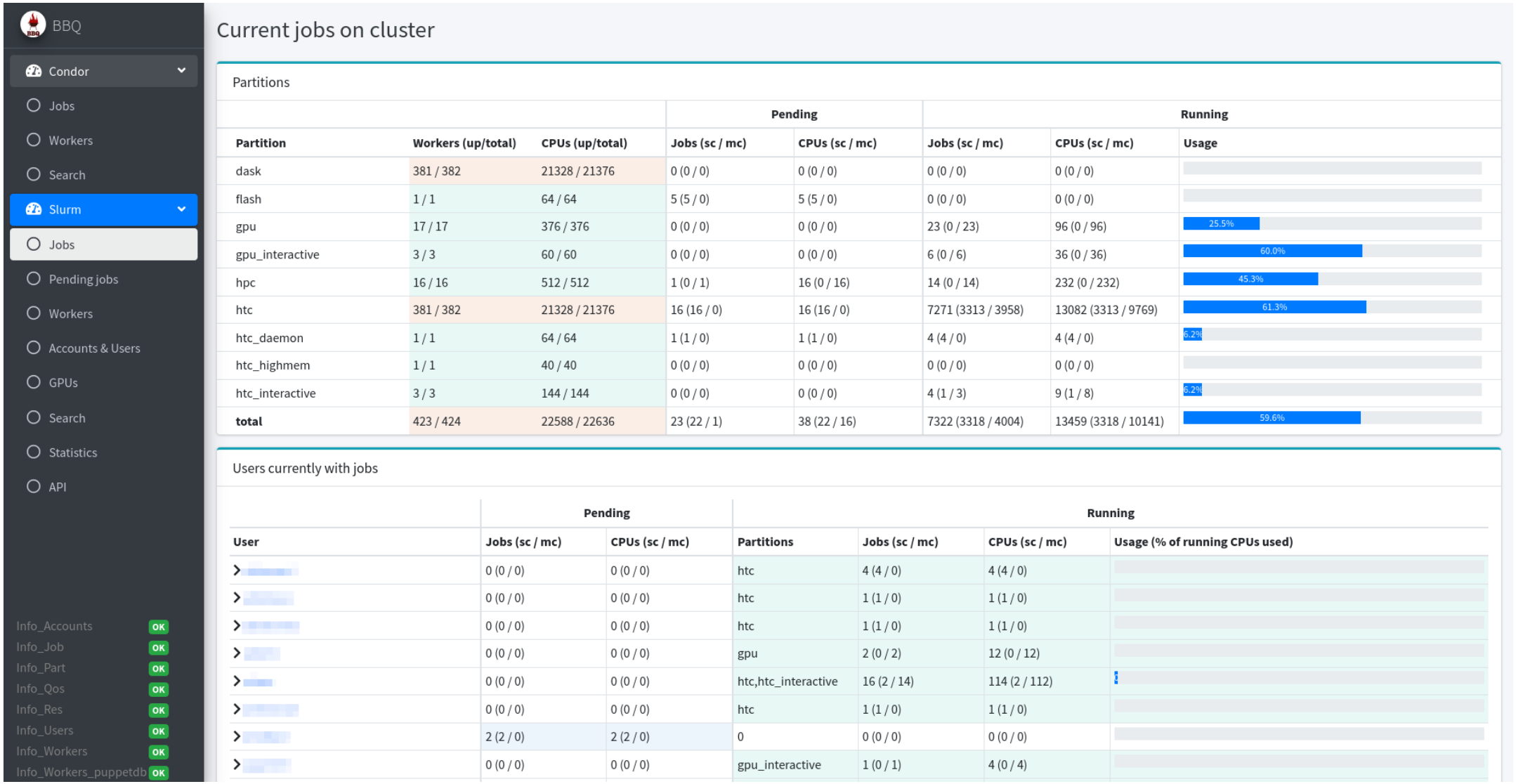
- Tous les calculs complexes sont maintenant fait en SQL via SQLAlchemy
  - De multiples requêtes sont quand même plus rapides
  - Multi-threading pour les pages les plus lentes
  - Optimisation des BDD
- Le chargement des pages est passé de plus de 20 secondes à moins de 5 secondes

Le déploiement

- Projet sur Gitlab
- Déploiement via Gitlab-CI pour pousser une image Docker sur Openshift
  - Il suffit de commit pour mettre à jour automatiquement
- Deux instances : prod et dev
  - L'équipe utilise dev
  - Le reste du centre utilise prod



Jobs sur le cluster Slurm



Current jobs on cluster

Partitions

			Pending		Running		
Partition	**Workers (up/total)**	**CPUs (up/total)**	**Jobs (sc / mc)**	**CPUs (sc / mc)**	**Jobs (sc / mc)**	**CPUs (sc / mc)**	**Usage**
dask	381 / 382	21328 / 21376	0 (0 / 0)	0 (0 / 0)	0 (0 / 0)	0 (0 / 0)	
flash	1 / 1	64 / 64	5 (5 / 0)	5 (5 / 0)	0 (0 / 0)	0 (0 / 0)	
gpu	17 / 17	376 / 376	0 (0 / 0)	0 (0 / 0)	23 (0 / 23)	96 (0 / 96)	25.5%
gpu_interactive	3 / 3	60 / 60	0 (0 / 0)	0 (0 / 0)	6 (0 / 6)	36 (0 / 36)	60.0%
hpc	16 / 16	512 / 512	1 (0 / 1)	16 (0 / 16)	14 (0 / 14)	232 (0 / 232)	45.3%
htc	381 / 382	21328 / 21376	16 (16 / 0)	16 (16 / 0)	7271 (3313 / 3958)	13082 (3313 / 9769)	61.3%
htc_daemon	1 / 1	64 / 64	1 (1 / 0)	1 (1 / 0)	4 (4 / 0)	4 (4 / 0)	6.2%
htc_highmem	1 / 1	40 / 40	0 (0 / 0)	0 (0 / 0)	0 (0 / 0)	0 (0 / 0)	
htc_interactive	3 / 3	144 / 144	0 (0 / 0)	0 (0 / 0)	4 (1 / 3)	9 (1 / 8)	6.2%
total	423 / 424	22588 / 22636	23 (22 / 1)	38 (22 / 16)	7322 (3318 / 4004)	13459 (3318 / 10141)	59.6%

Users currently with jobs

	Pending		Running			
User	**Jobs (sc / mc)**	**CPUs (sc / mc)**	**Partitions**	**Jobs (sc / mc)**	**CPUs (sc / mc)**	**Usage (% of running CPUs used)**
[redacted]	0 (0 / 0)	0 (0 / 0)	htc	4 (4 / 0)	4 (4 / 0)	
[redacted]	0 (0 / 0)	0 (0 / 0)	htc	1 (1 / 0)	1 (1 / 0)	
[redacted]	0 (0 / 0)	0 (0 / 0)	htc	1 (1 / 0)	1 (1 / 0)	
[redacted]	0 (0 / 0)	0 (0 / 0)	gpu	2 (0 / 2)	12 (0 / 12)	
[redacted]	0 (0 / 0)	0 (0 / 0)	htc,htc_interactive	16 (2 / 14)	114 (2 / 112)	
[redacted]	0 (0 / 0)	0 (0 / 0)	htc	1 (1 / 0)	1 (1 / 0)	
[redacted]	2 (2 / 0)	2 (2 / 0)	0	0 (0 / 0)	0 (0 / 0)	
[redacted]	0 (0 / 0)	0 (0 / 0)	gpu_interactive	1 (0 / 1)	4 (0 / 4)	

Workers du cluster Slurm

Workers with non nominal status

Worker	Partition	Smurf status	Slurm status	Load	CPUs	Slurm reason
ccwislurm0001	htc_interactive	up	IDLE+DRAIN		0 / 64	Kill task failed [root@2022-11-07T15:56:46]
ccwslurm0320	htc,dask	up	IDLE+DRAIN		0 / 48	fan check (mhamant) [slurmadm@2022-11-07T10:34:04]
ccwslurm0321	htc,dask	up	IDLE+DRAIN		0 / 48	fan check (mhamant) [slurmadm@2022-11-07T10:34:04]
ccwslurm0344	htc,dask	test	DOWN+DRAIN		0 / 48	memory issue (mhamant) [slurmadm@2022-11-07T08:39:33]
ccwslurm0345	htc,dask	up	IDLE+DRAIN		0 / 48	fan check (mhamant) [slurmadm@2022-11-07T10:34:04]

All workers

Show entries

Copy CSV Excel

Search:

Worker	Partition	Smurf status	Slurm status	Load	CPUs	Usage
ccwgislurm0001	gpu_interactive	up	ALLOCATED	0.50	16 / 16	100.0%
ccwgislurm0100	gpu_interactive	up	MIXED	0.51	12 / 24	50.0%
ccwgislurm0103	gpu_interactive	up	MIXED	1.18	8 / 20	40.0%
ccwgslurm0002	gpu	up	IDLE	0.41	0 / 16	
ccwgslurm0100	gpu	up	MIXED	8.90	8 / 24	33.3%
ccwgslurm0101	gpu	up	MIXED	5.82	12 / 24	50.0%
ccwgslurm0102	gpu	up	MIXED	1.40	4 / 24	16.7%
ccwgslurm0103	gpu	up	MIXED	0.97	4 / 24	16.7%
ccwgslurm0104	gpu	up	MIXED	3.94	16 / 24	66.7%

Jobs en attente

Number of pending jobs

	submit<6h		6h<submit<24h		24h<submit<48h		48h<submit<72h		72h<submit		Total	
Reason	**Jobs**	**CPUs**	**Jobs**	**CPUs**	**Jobs**	**CPUs**	**Jobs**	**CPUs**	**Jobs**	**CPUs**	**Jobs**	**CPUs**
AssocGrpCpuLimit	4	4	0	0	0	0	0	0	0	0	**4**	**4**
BeginTime	2	2	10	10	0	0	0	0	0	0	**12**	**12**
None	1	1	0	0	0	0	0	0	0	0	**1**	**1**
Resources	1	16	0	0	0	0	0	0	0	0	**1**	**16**
total	8	23	10	10	0	0	0	0	0	0	**18**	**33**

Jobs

Show entries

Copy CSV Excel

Search:

jobid	user	account	partition	submit	priority	alloccpus	reqmem	reason	share	num_jobs
17806671			flash	2022-11-07 16:01:36	1187	1	3G	BeginTime	415	3
17852901			flash	2022-11-08 09:00:10	1187	1	3G	BeginTime	415	1
17842439			flash	2022-11-07 23:30:49	1186	1	3G	BeginTime	415	1
17777830			htc_daemon	2022-11-07 12:00:03	1186	1	1G	BeginTime	415	1
17860708			hpc	2022-11-08 10:29:52	609	16	48G	Resources	4198	1
17849560			htc	2022-11-08 05:00:10	451	1	3G	BeginTime	9179	1
17849559			htc	2022-11-08 05:00:10	443	1	3G	BeginTime	9179	2
17847933			htc	2022-11-08 03:02:09	188	1	2000M	BeginTime	415	1
17865092			htc	2022-11-08 11:21:44	187	1	3G	BeginTime	415	1
17842821			htc	2022-11-07 23:32:50	187	1	3G	BeginTime	415	1
17870681			htc	2022-11-08 11:41:54	186	1	4G	AssocGrpCpuLimit	415	5

Infos d’un job

mainjobid	1782	**jobid**	1782	**user**	
start	2022-11-07 20:07:04	**state**	completed	**submit**	2022-11-07 20:07:04
end	2022-11-08 00:37:23	**hostname**	ccwslurm0187	**partition**	htc
qos	normal	**exitcode**	0:0	**account**	
alloccpus	2	**allocnodes**	1		

Secondary info

alloctres	billing=2,cpu=2,mem=6G,node=1	**associd**	5104	**avecpu**	None
avediskread	None	**avediskwrite**	None	**averss**	None
avevmsize	None	**constraints**	None	**cputime**	09:00:38
derivedexitcode	0:0	**elapsed**	04:30:19	**eligible**	2022-11-07 20:07:04
group		**jobname**		**maxdiskread**	None
maxdiskreadnode	None	**maxdiskreadtask**	None	**maxdiskwrite**	None
maxdiskwritenode	None	**maxdiskwritetask**	None	**maxpages**	None
maxpagesnode	None	**maxpagestask**	None	**maxrss**	1595056K
maxrssnode	None	**maxrsstask**	None	**maxvmsize**	None
maxvmsizenode	None	**maxvmsizetask**	None	**mincpu**	None
mincpunode	None	**mincputask**	None	**ncpus**	2
nnodes	1	**ntasks**	None	**priority**	324
reason	None	**reqcpus**	2	**reqmem**	6G
reqnodes	1	**reqtres**	billing=2,cpu=2,mem=6G,node=1	**reservation**	None
reserved	00:00:00	**suspended**	00:00:00	**systemcpu**	00:06.373
timelimit	1-00:00:00	**totalcpu**	04:30:11	**usercpu**	04:30:05
workdir		**gpu_type**	None	**gpu_slots**	None
hs06	None	**gpu**	None	**licenses**	None
stderr	None	**stdout**	None	**submithost**	None
cancelled_by	None				

Other info

Infos d’un utilisateur

Status

Account	Parent Account	QOS	Maxjobs	Maxsubmit
[redacted]	out	daemon,flash,normal	∞	∞
[redacted] D	root	daemon,flash,gpu,normal	∞	∞

Current jobs

State	Jobs (sc / mc)	CPUs (sc / mc)	GPUs
running	1994 (0 / 1994)	3990 (0 / 3990)	0
pending	0 (0 / 0)	0 (0 / 0)	0

Last 24h

Status	Jobs

Jobs per worker

Worker	Jobs	CPUs
ccwslurm0002	4	8
ccwslurm0003	9	20
ccwslurm0004	9	18
ccwslurm0005	7	14
ccwslurm0006	9	18
ccwslurm0007	3	6

Jobs

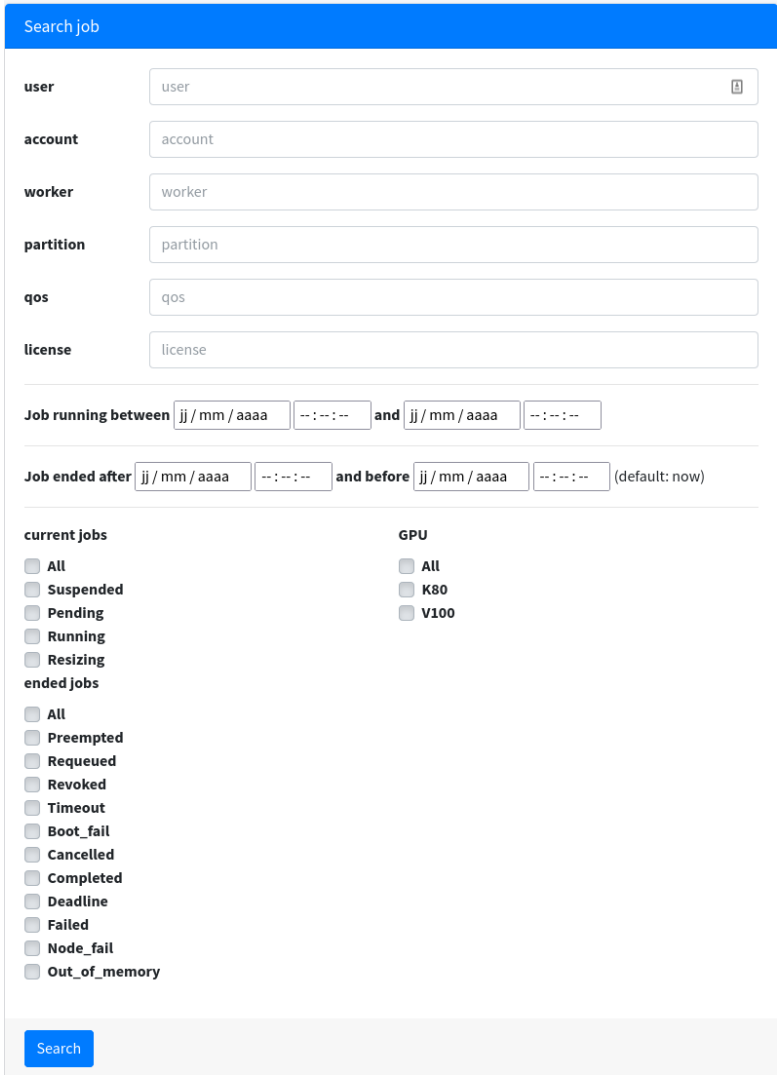
Show 50 entries

Copy CSV Excel

Search:

jobid	state_badge	user	account	partition	qos	worker	submit	start	alloccpus
178[redacted]	running	[redacted]	[redacted]	htc	normal	ccwslurm0003	2022-11-07 19:30:34	2022-11-07 19:30:34	4
178[redacted]	running	[redacted]	[redacted]	htc	normal	ccwslurm0269	2022-11-08 09:07:54	2022-11-08 09:07:59	2
178[redacted]	running	[redacted]	[redacted]	htc	normal	ccwslurm0386	2022-11-08 09:12:03	2022-11-08 09:12:04	2
178[redacted]	running	[redacted]	[redacted]	htc	normal	ccwslurm0230	2022-11-08 09:12:06	2022-11-08 09:12:06	2

Recherche avancée de jobs



Statistiques

- Étude des comportements des jobs et de la ferme Slurm
- Utile pour les choix de configuration et l’optimisation des ressources

Les leçons

- L’apparence ça compte
- Savoir où on va
  - BBQ a commencé comme un script mais est maintenant un produit complet
- Gare à l’effet nouveauté
  - C’est dans les vieux pots qu’on fait (parfois) la meilleure confiture
- Prendre en compte l’environnement
  - La pratique ≠ la théorie

La suite

- Adapter BBQ aux évolutions futures de nos fermes
  - Notamment la prise en compte plus poussée des demandes RAM
- Ajouter des actions directement dans BBQ
  - Nécessite une gestion des identités

- Ajouter des tests fonctionnels
- Continuer à améliorer la qualité du code et à implémenter les bonnes pratiques
- Continuer à adapter les BDD

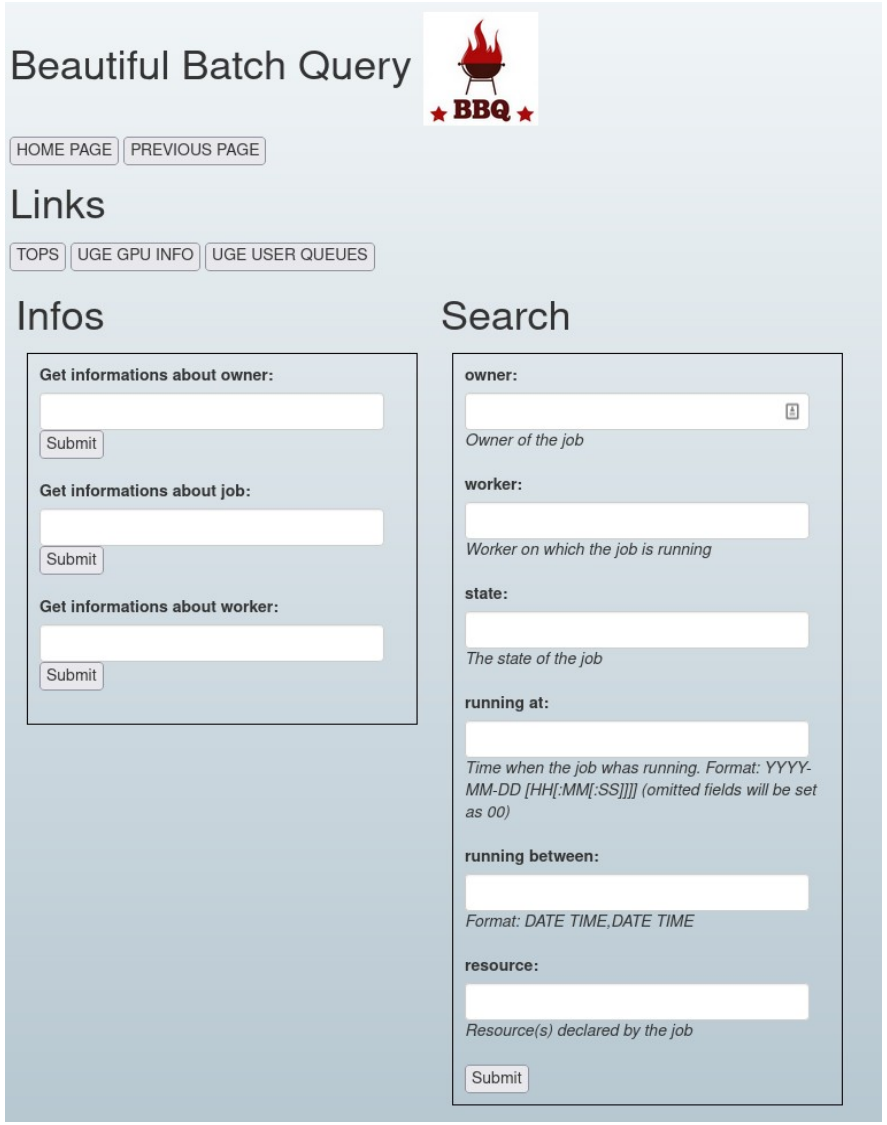
Partager le code : les problèmes

- Sources de données maison
  - BBQ ne communique pas directement avec les ordonnanceurs
- Configuration en dur dans le code
  - BBQ prend en compte des spécificités du CC
- Code “interne”
  - Ajouter des commentaires et des tests

La fin

Merci de votre attention

Version 1



- Ajouts de quelques fonctionnalités
  - Validation du POC
- Mise en place du déploiement automatique
- Toujours que UGE
  - Modification des données en BDD
  - Pas d’accès aux scripts pour Condor
- Trouvaille du nom
  - Promis le jeu de mot n’est pas de moi

Résumé de la stack technologique

- Python
- Données
  - Requêtes : SQLAlchemy
  - Traitements secondaires : Pandas
- Web et affichage
  - Back-end : Flask
  - Front-end : AdminLTE / Bootstrap, un peu de Javascript
  - Templating : Jinja2 + HTML
  - Graphes : Bokeh
- Serveur web : Gunicorn

Quelques chiffres

- 400 commits sur la branche dev
- Code :
  - 29 fichiers Python pour 4300 lignes de code
  - 38 fichiers HTML/Jinja2 pour 4500 lignes
- Refactoring : beaucoup trop de fois
- Développé sous VS Code