Audit de la chaîne de calcul de l'IPSL sur les centres de calcul du TGCC et de l'IDRIS

G. Pierre, Arnaud N Caubel, Sébastien Denvil, Marie-alice Foujols, Olivier Marti

Audit de la chaîne de calcul de l'IPSL sur les centres de calcul du TGCC et de l'IDRIS

Guillaume PIERRE

Société INCKA

Arnaud CAUBEL, Sébastien DENVIL,
Marie-Alice FOUJOLS, Olivier MARTI

IPSL

Prestation accord cadre CEA/DSM – 2012

Table des matières

1 Contexte

L'IPSL (Institut Pierre Simon Laplace) est une entité qui regroupe 6 laboratoires et fédère des études multidisciplinaires (scientifiques ou techniques) dans le domaine de l'environnement global. Un des grands projets scientifiques porté par l'IPSL est le pôle de modélisation du climat, qui organise le développement d'un outil de simulation du climat appelé « Modèle Système Terre ». Cet outil se compose d'un modèle de climat et dispose d'un environnement de calcul associé permettant l'installation et l'exécution du modèle du climat sur plusieurs centres de calcul. Le TGCC et l'IDRIS sont les centres de calcul les plus utilisés par l'IPSL. C'est donc sur ces 2 centres et en particulier sur les machines Curie (TGCC) et Vargas (IDRIS) que l'évaluation de la « chaîne de calcul » est réalisée et dont on rend compte dans le présent document.

2 Description de la chaîne de calcul

Cet outil repose sur un enchaînement de jobs effectuant des tâches de simulation ou de post-traitement de résultats de simulation, et profite des ressources de centres de calcul de haute performance. Pour prendre en compte des contraintes différentes imposées par chaque centre de calcul, les équipes de l'IPSL ont adapté le déroulement de la chaîne. Certaines de ces différences sont exposées plus loin dans cette partie descriptive.

2.1 La couche « libIGCM »

La chaîne de calcul de l'IPSL repose sur la couche logicielle « libIGCM » : constituée d'un ensemble de scripts shell, elle pilote de manière cohérente tout le déroulement de la chaîne à l'exécution : elle lance et surveille l'exécution de tous les outils élémentaires qui réalisent des traitements spécifiques. Plus précisément, son rôle est de fournir des données initiales à l'outil de simulation (qui constitue l'intelligence de la chaîne), d'assurer l'enchaînement de ses tâches de calcul, et de coordonner des étapes de post-traitements sur les sorties de cet outil. L'outil de simulation (le modèle « Système Terre ») et les outils de post-traitements utilisés ne sont pas considérés comme faisant partie de la couche « libIGCM ».
Ces outils sont les suivants:

- Exécutables issus de sources fortran pour les calculs de climatologie en parallèle ;
- L'outil « rebuild » (issu de sources Fortran) pour la recombinaison cohérente de sorties provenant des différents processus de calcul ;
- L'outil « ncrcat » (issu du package « nco ») pour la concaténation cohérente de fichiers de type « netCDF » ;
- L'outil « ncra » (issu du package « nco ») pour la création de statistiques saisonnières à partir des résultats de calcul (au format « netCDF ») ;
- L'outil « cdo » peut notamment servir à la génération de séries temporelles pour certaines variables issues des calculs ;

Par la suite, lorsqu'on parlera de job, il s'agira du lancement (en mode « batch ») d'un des scripts de la couche « libIGCM ».

2.2 Enchaînement des jobs de calcul

Une simulation débute par le lancement d'un job appelé « job de calcul » c'est-à-dire un calcul de climatologie sur les premiers mois de la simulation. A ce stade, la chaîne a récupéré ce qu'il lui faut (conditions initiales et conditions limites) pour commencer ce calcul. Juste avant de se terminer, ce job lancera un nouveau job de calcul qui poursuivra la simulation sur les mois suivants, et ainsi de suite jusqu'à la fin de la simulation. Chaque job de calcul se base

sur les résultats du précédent pour poursuivre les calculs. Ce fonctionnement en jobs chaînés est imposé par les centres de calcul qui allouent des ressources de calcul pour un temps limité. Cet enchaînement de jobs de calcul dépendants constitue l'ossature de la chaîne.
Chaque job lance des calculs en parallèle sur plusieurs cœurs de calcul, classiquement sur 32 processeurs.
Ces jobs produisent un rapport détaillé des traitements réalisés au cours de leur exécution.
Les jobs de calcul sont exécutés sur la machine Curie nœuds fins au TGCC et sur Vargas à l'IDRIS.

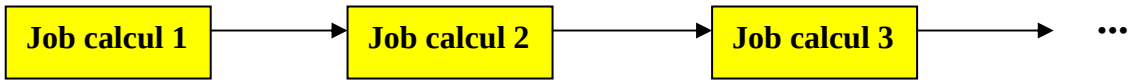


Figure 1 : enchaînement des jobs de calcul

2.3 Les jobs de post-traitements

Il existe d'autres jobs appelés « jobs de post-traitements » dont le rôle est de traiter les sorties des jobs de calcul.
A une fréquence que l'utilisateur fixe avant le lancement de la chaîne (par exemple tous les 2 ans de simulations), le job de calcul courant appelle l'exécution d'un job dit « job de rebuild » chargé de récupérer les résultats de calculs produits par les différents processus afin de les recombiner pour former un résultat de simulation sur le domaine global. A deux autres fréquences fixées elles aussi par l'utilisateur, nécessairement plus grande que la précédente, un job de « rebuild » commande l'exécution de deux nouveaux jobs : « TS » (pour « time series ») et « SE » (pour « seasonal »). Ces deux jobs produisent respectivement :

- Des séries temporelles pour des variables spécifiques du calcul ;
- Des moyennes saisonnières (valeurs moyennes des mois de janvier, février,…au cours d'une période de la simulation).

A leur tour, mais cette fois avec la même fréquence, les jobs « TS » (respectivement « SE ») lanceront un job de « monitoring » (respectivement d'« atlas »).
Les jobs de « monitoring » publient des courbes d'évolution de certaines variables de la simulation, et les jobs d'« atlas » publient des cartes de comparaison à des données observées.
Tout comme les jobs de calcul, tous ces jobs donnent lieu à l'écriture d'un rapport détaillé de leurs activités.
Les jobs de post-traitements sont exécutés sur la machine Curie nœuds larges au TGCC et sur Ulam à l'IDRIS.

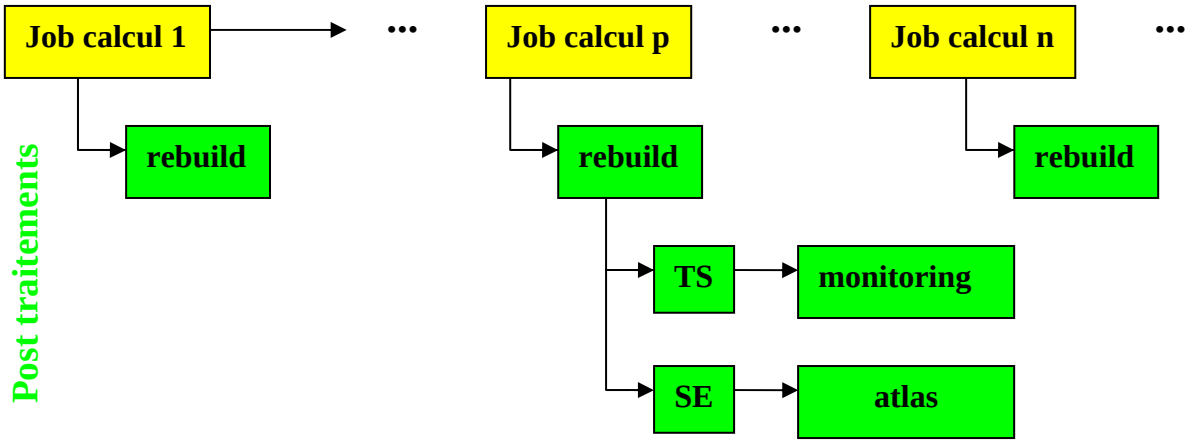


Figure 2 : les jobs de post traitements

2.4 Différences de fonctionnement à l'IDRIS et au TGCC

2.4.1 Transfert de données

Au TGCC, tous les calculs et post-traitements sont effectués par la même machine Curie (nœuds larges et nœuds fins). Les données d'entrées de simulations, les résultats de calculs et de leurs post-traitements, et leur stockage final se trouvent sur des espaces différents mais visibles depuis la machine.
A l'IDRIS, les données d'entrée se situent sur la machine « Gaya », les résultats de calculs sont produits sur la machine « Vargas », les post-traitements se font sur « Ulam » et le stockage final sur « Gaya ». Tous ces transferts se font via les outils mfget/mfput de l'IDRIS.

2.4.2 Concaténation et archivage

Les contraintes actuelles au TGCC imposent de stocker un nombre de fichiers limité sur CCCSTORE (100 000 inodes). Une demande est en cours pour augmenter ces quotas pour certains logins associés à de la production importante. Le nombre de fichiers produits par les simulations a donc dû être réduit. La chaîne de calcul s'est adaptée et opère la concaténation et l'assemblage d'ensembles de fichiers avant stockage. Ce type de contrainte n'existe pas à l'IDRIS. Cependant, il est prévu que la chaîne de calcul utilise également les fonctionnalités de concaténation et d'archivage sur la nouvelle machine Ada.
Cela nécessitera donc un espace de travail temporaire de type « SCRATCH » sur Ada qui n'était actuellement pas disponible sur la machine Vargas.
Au TGCC, ces réductions sont faites par des jobs, dits jobs de « pack », soit en archivant un ensemble de fichiers avec la commande « tar », soit en concaténant un autre ensemble de fichiers grâce à l'outil « ncrcat ».
Les fichiers de « restart » (pour redémarrage de la chaîne) et les fichiers « debug » (rapports d'exécution au format texte provenant du calcul) sont archivés avec « tar ».
Les fichiers de sorties du calcul « output » (fichiers au format « netCDF ») sont concaténés avec « ncrcat ».
Tout comme les autres jobs évoqués plus haut, ces jobs ont une fréquence de lancement paramétrable et produisent des rapports d'exécution.

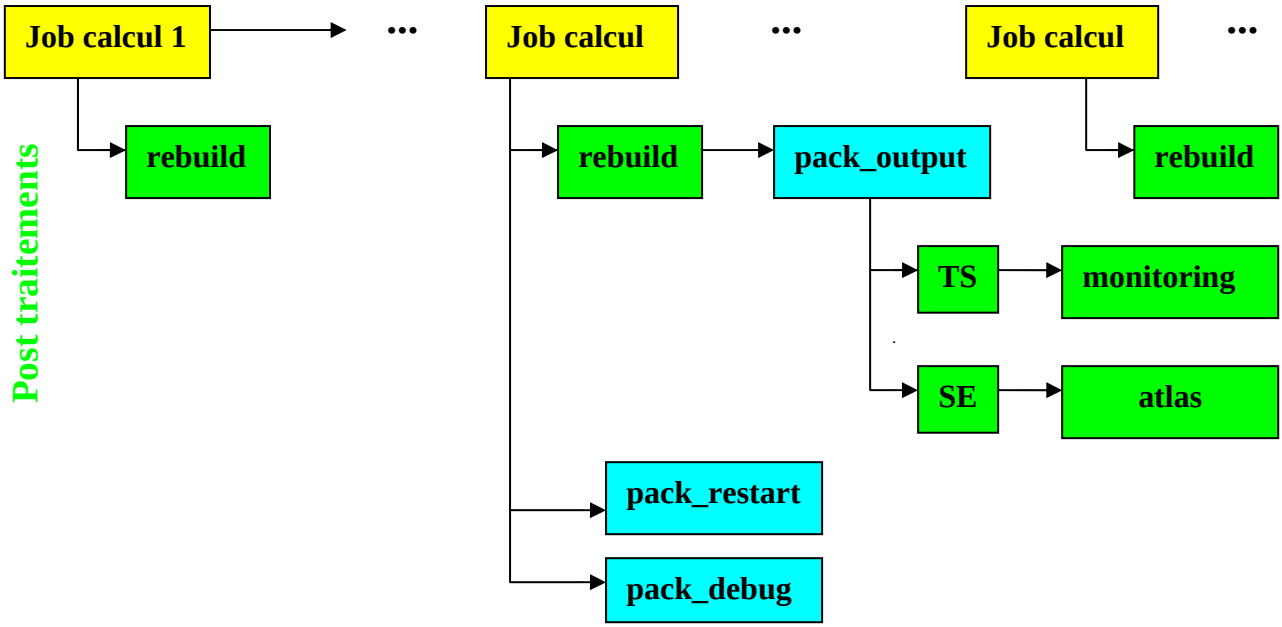


Figure 3 : les jobs de pack au TGCC (en bleu)

3 Introduction de l'évaluation

Les études menées sur la chaîne de calcul, dont les résultats et conclusions sont donnés dans ce document, s'articulent autour de 3 préoccupations :

- Evaluation des performances de la chaîne ;
- Fiabilité des ressources allouées par les centres de calcul ;
- Améliorations à apporter à la chaîne.

L'évaluation des performances de la chaîne s'est faite sur la quantification des caractéristiques suivantes :

- Le temps d'attente des jobs de calcul et de post-traitement (de la soumission à l'exécution) et le temps d'exécution de ces jobs (1er niveau d'analyse) ;
- Pour les jobs de post-traitement (2ème niveau d'analyse):
  - Le temps passé à l'exécution pour chaque type de traitement des résultats de calcul.
  - La part de chacune des tâches qui composent un job de post-traitement : les post-traitements pouvant être rangés en 3 familles (tâches d'importation, de traitement réel et d'exportation), on s'est intéressé à la proportion en temps pris par l'exécution de chaque famille, à l'échelle d'une simulation et à l'échelle plus restreinte d'un job.

Pour les critères ci-dessus, les résultats obtenus sont commentés et quelques comparaisons de ces résultats entre les 2 centres sont effectuées.

En tant qu'utilisateur de la chaîne, l'auteur de ce document a rencontré quelques difficultés liées aux moyens techniques mis à disposition par les centres de calcul. Elles rejoignent parfois celles rencontrées par les utilisateurs réguliers. La fiabilité de la chaîne sera discutée de ce point de vue.

Le premier usage de ce document est interne à l'IPSL ; ce document pourra en effet être utile pour décider d'améliorations à apporter à la chaîne afin de la rendre plus performante. Il

pourra, en outre, servir de base pour des échanges entre les développeurs ou utilisateurs de la chaîne et les centres de calcul.

4 Evaluation des performances

4.1Analyse des temps (ou analyse de 1er niveau)

L'analyse de 1er niveau est basée sur les temps d'attente avant exécution et les temps d'exécution des différents jobs de la chaîne de calcul de l'IPSL. Nous avons extrait des temps à partir de deux simulations de type « historique », l'une ayant été lancée au TGCC sur Curie, l'autre à l'IDRIS sur Vargas (calcul) et Ulam (post-traitement). Il s'agit ici de comparer les résultats obtenus à partir de ces deux simulations.
Ces deux simulations sont similaires : les configurations de la chaîne, les durées simulées (156 ans), le niveau de sorties activé, et les fréquences de lancement de jobs sont identiques. Elles diffèrent néanmoins par les conditions initiales prises en compte au démarrage. Ces différences n'auront cependant aucune incidence sur notre objet d'étude.

4.1.1 Les simulations analysées

Pour chacune des deux simulations, on donne les renseignements suivants :

- L'emplacement du répertoire de soumission, qui contient notamment le script de lancement de la simulation (« Job_xxx ») et le fichier « run.card » (rapport listant entre autre les mois de simulations).
- L'emplacement où l'on trouve les rapports d'exécution des jobs (de calcul et de post-traitement).
- Les dates de début et de fin de la simulation.

4.1.1.1 Au TGCC

Les répertoires suivants se trouvent sur « Curie ».
Répertoire de soumission :
`/ccc/work/cont003/dsm/p86maf/CURIE/IPSLCM5A/T22022012/modipsl/config/IPSLCM5A/v3.historicalC74`
Répertoire des rapports d'exécution des jobs :
`/ccc/scratch/cont003/dsm/p86maf/IGCM_OUT/IPSLCM5A/PROD/historical/v3.historicalC74/Out`
Dates :

- Début : 17/04/2012
- Fin : 15/05/2012

4.1.1.2 A l'IDRIS

Répertoire de soumission (sur « Vargas ») :
`/workgpfs/rech/tgw/rtgw001/CMIP5/IPSLCM5A_20110513/modipsl/config/IPSLCM5A/v3.historicalV52`
Répertoire des rapports d'exécution des jobs :
`/workdir/rech/tgw/rtgw001/IGCM_OUT/IPSLCM5A/v3.historivalV52`
Dates :

- Début : 14/05/2011
- Fin : 13/06/2011

4.1.2 Données extraites

Types	nb files	Attente				Execution			
		min	max	moy	sum	min	max	moy	sum
Script_Output_v3.historicalC74	89	00:00:00	13:25:20	00:47:57	71:07:37	04:02:15	06:16:23	04:48:30	427:56:55
create_ts.2D	16	00:00:00	00:00:34	00:00:14	00:03:38	00:11:14	01:08:44	00:34:49	09:17:05
create_ts.3D	16	00:00:00	00:00:30	00:00:16	00:04:11	00:03:25	00:28:01	00:15:59	04:15:41
create_ts.Chunck2D.ATM.Post_1D_histday	16	00:00:00	00:00:34	00:00:17	00:04:30	00:23:39	02:10:31	01:39:29	26:31:38
create_ts.Chunck2D.ATM.Post_1D_histdayCOSP	2	00:00:29	00:00:33	00:00:31	00:01:02	00:23:01	00:26:45	00:24:53	00:49:46
create_ts.Chunck2D.ATM.Post_3H_histhf3h	16	00:00:00	00:00:33	00:00:16	00:04:12	01:11:36	01:52:29	01:31:59	24:31:50
create_ts.Chunck2D.ATM.Post_3H_histhf3hm	16	00:00:00	00:00:32	00:00:17	00:04:36	00:23:27	00:56:20	00:35:26	09:26:55
create_ts.Chunck2D.ATM.Post_HF_histhf	15	00:00:00	00:00:33	00:00:15	00:03:51	01:55:42	02:47:56	02:20:35	35:08:44
create_ts.Chunck2D.ICE.Post_1D_icemod	16	00:00:00	00:00:31	00:00:16	00:04:17	00:00:42	00:13:52	00:07:59	02:07:44
create_ts.Chunck2D.OCE.Post_1D_grid_T	16	00:00:00	00:00:31	00:00:16	00:04:23	00:15:01	00:36:23	00:25:44	06:51:46
create_ts.Chunck2D.SRF.Post_HF_sechiba_out_2	16	00:00:00	00:00:32	00:00:17	00:04:29	00:03:42	00:10:47	00:06:44	01:47:47
create_ts.Chunck3D.ATM.Post_1D_histday	16	00:00:00	00:01:42	00:00:21	00:05:29	01:00:35	02:00:18	01:29:06	23:45:34
create_ts.Chunck3D.ATM.Post_1D_histdayCOSP	2	00:00:22	00:00:27	00:00:25	00:00:49	00:39:45	00:55:37	00:47:41	01:35:22
create_ts.Chunck3D.ATM.Post_1D_histdayNMC	16	00:00:00	00:02:55	00:00:27	00:07:04	00:05:24	00:12:12	00:10:21	02:45:31
create_ts.Chunck3D.ATM.Post_1M_histmth	16	00:00:00	00:00:30	00:00:15	00:04:05	00:10:18	01:03:45	00:34:49	09:16:59
create_ts.Chunck3D.ATM.Post_1M_histmthCOSP	16	00:00:00	00:00:31	00:00:17	00:04:28	00:00:50	00:07:22	00:04:12	01:07:10
create_ts.Chunck3D.ATM.Post_1M_histmthNMC	16	00:00:00	00:01:06	00:00:19	00:04:58	00:02:02	00:19:27	00:09:22	02:29:56
create_ts.Chunck3D.ATM.Post_HF_histhf	16	00:00:00	00:05:08	00:00:34	00:09:08	01:24:05	01:46:59	01:35:52	25:33:59
create_ts.Chunck3D.ATM.Post_HF_histhfNMC	16	00:00:00	00:05:07	00:00:40	00:10:42	00:07:55	00:17:49	00:13:11	03:30:50
create_ts.Chunck3D.MBG.Post_1M_dbio_T	16	00:00:00	00:22:16	00:02:17	00:36:33	00:04:53	00:40:03	00:20:12	05:23:15
create_ts.Chunck3D.MBG.Post_1M_diad_T	16	00:00:00	00:21:44	00:02:15	00:36:07	00:05:32	00:44:49	00:22:59	06:07:39
create_ts.Chunck3D.MBG.Post_1M_ptrc_T	16	00:00:00	00:20:38	00:02:12	00:35:06	00:09:52	01:08:49	00:35:40	09:30:42
create_ts.Chunck3D.OCE.Post_1D_grid_T	16	00:00:00	00:12:28	00:01:24	00:22:22	00:12:33	00:27:02	00:17:18	04:36:51
create_ts.Chunck3D.OCE.Post_1D_grid_U	16	00:00:00	00:13:33	00:01:28	00:23:21	00:03:38	00:08:37	00:06:33	01:44:46
create_ts.Chunck3D.OCE.Post_1D_grid_V	16	00:00:00	00:13:33	00:01:33	00:24:44	00:03:58	00:08:34	00:06:42	01:47:07
create_ts.Chunck3D.OCE.Post_1M_grid_T	16	00:00:00	00:06:56	00:00:57	00:15:06	00:02:01	00:16:58	00:09:41	02:34:55
create_ts.Chunck3D.OCE.Post_1M_grid_U	16	00:00:00	00:09:11	00:01:11	00:18:51	00:01:03	00:10:53	00:05:56	01:34:54
create_ts.Chunck3D.OCE.Post_1M_grid_V	16	00:00:00	00:11:23	00:01:19	00:20:59	00:01:09	00:10:46	00:05:46	01:32:11
create_ts.Chunck3D.OCE.Post_1M_grid_W	16	00:00:00	00:11:55	00:01:22	00:21:53	00:03:38	00:22:06	00:13:13	03:31:35
create_ts.Chunck3D.SRF.Post_HF_sechiba_out_2	16	00:00:00	00:05:44	00:00:44	00:11:46	00:03:12	00:06:52	00:05:20	01:25:19
create_ts.total	435	00:00:00	00:22:16	00:00:49	05:52:40	00:00:42	02:47:56	00:31:49	230:43:31
atlas_LDMZ	16	00:00:00	00:00:32	00:00:17	00:04:25	00:05:53	00:26:07	00:08:32	02:16:28
atlas_ORCA_LIM	16	00:00:00	00:00:32	00:00:16	00:04:20	00:07:28	00:33:37	00:10:32	02:48:31
atlas_ORCHIDEE	16	00:00:00	00:00:33	00:00:17	00:04:33	00:03:03	00:19:15	00:04:38	01:14:03
atlas_PISCES	16	00:00:00	00:00:31	00:00:16	00:04:13	00:03:34	00:15:57	00:05:05	01:21:19
create_se	16	00:00:00	00:22:16	00:02:17	00:36:27	00:10:03	00:28:36	00:18:06	04:49:37
monitoring	15	00:00:00	00:03:33	00:00:28	00:07:07	00:02:08	00:17:18	00:08:27	02:06:50
pack_debug	152	00:00:00	11:39:11	00:22:01	55:45:57	00:00:05	00:05:55	00:00:14	00:35:39
pack_output	152	00:00:00	09:06:19	00:17:13	43:36:21	00:07:13	01:57:36	00:10:16	26:01:28
pack_restart	152	00:00:00	11:39:10	00:22:00	55:43:12	00:01:31	00:06:57	00:02:17	05:47:22
rebuild_fromWorkdir	152	00:00:00	11:39:09	00:22:00	55:43:27	01:07:59	02:47:53	01:18:59	200:05:49

Figure 4 : Temps d'attente et temps d'exécution par job sur Curie (TGCC)

Types	nb files	Attente				Execution			
		min	max	moy	sum	min	max	moy	sum
ATL_LMDZ	14	00:00:00	04:30:00	00:33:39	07:51:00	00:08:00	00:26:00	00:14:30	03:23:00
ATL_OPA	14	00:00:00	04:36:00	00:38:43	09:02:00	00:14:00	00:29:00	00:17:26	04:04:00
ATL_ORCH	14	00:00:00	04:18:00	00:31:04	07:15:00	00:05:00	00:12:00	00:06:43	01:34:00
ATL_PIS	14	00:00:00	04:37:00	00:47:39	11:07:00	00:08:00	00:15:00	00:10:04	02:21:00
historivalV52	53	00:00:00	10:30:00	00:34:03	30:05:00	00:24:00	19:15:00	12:58:36	687:46:00
MONITORING	28	00:02:00	37:58:00	09:28:36	265:21:00	00:04:00	00:37:00	00:15:39	07:18:00
REBUILDARCH	155	00:00:00	43:55:00	02:31:13	390:39:00	00:59:00	09:44:00	03:35:10	555:51:00
SE	14	03:37:00	32:47:00	14:42:47	205:59:00	00:40:00	01:24:00	00:48:51	11:24:00
TS	405	00:00:00	32:08:00	05:03:04	2045:45:00	00:02:00	20:08:00	03:06:24	1258:14:00

Figure 5 : Temps d'attente et temps d'exécution par job à l'IDRIS

On retrouve dans ces tableaux tous les jobs évoqués dans la partie 2.
Les noms des jobs diffèrent d'un centre de calcul à l'autre :
Les jobs de calcul commencent par « Script_Output_v3 » sur Curie et sont notés « historicalV52 » à l'IDRIS. Les jobs de rebuild sont notés « rebuild_fromWorkdir » sur Curie tandis qu'ils s'appellent « REBUILDARCH » à l'IDRIS.

Les jobs « TS » et « SE » sont notés « create.ts » et « create.se » sur Curie, et « TS », « SE » à l'IDRIS. D'ailleurs, on constate une grande variété de jobs « TS » sur Curie ; ils sont simplement différenciés sur Curie et pas à l'IDRIS. Pour qu'une comparaison soit plus aisée avec l'IDRIS, il existe un sous total pour les jobs « TS » sur Curie.
Les jobs de « pack » sont présents seulement sur Curie, conformément à ce qui a été dit dans la partie 2.4.2.
Les jobs d'« atlas » sont notés « atlas_xxx » sur Curie et « ATL_XXX » à l'IDRIS.

Lorsqu'un job est soumis, il est placé dans une file d'attente (ou queue) avant exécution effective. Les tableaux précédents rendent compte de cet état de fait en séparant, pour tous les jobs, le temps d'attente et le temps d'exécution dans deux blocs de colonnes différents.
On voit que chaque type de job s'est exécuté plusieurs fois et on trouve ce nombre d'occurrences dans la colonne « nb files » des tableaux précédents.
Les indicateurs qui nous intéressent plus particulièrement ici sont les moyennes de temps d'attente et de temps d'exécution pour un type de job.

4.1.3 Analyse du temps d'attente

L'analyse des temps d'attente permet de distinguer les jobs de calcul des jobs de post-traitement.
En effet, le temps d'attente moyen pour un job de post-traitement est toujours nettement plus grand à l'IDRIS qu'au TGCC. L'écart le plus frappant concerne les jobs « SE » pour lesquels le temps d'attente moyen sur Curie est d'un peu plus de 2 minutes alors que celui à l'IDRIS est d'un peu moins de 15 heures.
En revanche, la comparaison des temps d'attente moyen pour les jobs de calcul entre les deux centres est plus défavorable au TGCC : il est d'environ 48 minutes sur Curie et de 34 minutes à l'IDRIS.
Mais que ce soit au TGCC ou bien à l'IDRIS, on remarque dans les deux cas de grandes différences entre les jobs de calcul et de post-traitement au sein d'un même centre.
On peut donc se poser la question suivante : pourquoi les jobs de calcul se distinguent-ils de cette manière des jobs de post-traitement en termes de temps d'attente ?
Sur Curie, les jobs de calcul sont soumis, non pas sur les nœuds « larges » comme les post-traitements, mais sur les nœuds dits « fins » ou « standards » qui sont davantage dédiés à la haute performance. Ainsi les jobs de calcul sont soumis dans une queue différente de celle des post-traitements sur Curie.
On peut donc imaginer que, au TGCC, si les calculs étaient soumis sur nœuds larges, ils attendraient moins longtemps avant d'être exécutés. Cela dit, leur temps d'exécution serait plus grand.
A l'IDRIS, les jobs de calcul sont soumis sur la machine Vargas tandis que les jobs post-traitements sont lancés sur la machine Ulam.

4.1.4 Analyse du temps d'exécution

L'analyse des temps d'exécution moyens est beaucoup plus simple que pour les temps d'attente.
Aussi bien pour les jobs de calcul que de post-traitement, ils sont exécutés plus rapidement sur Curie qu'à l'IDRIS.

4.1.4.1 Les jobs de calcul

Pour les jobs de calcul, il y a quasiment un facteur 3 entre les temps obtenus sur Curie et Vargas (4h48 sur Curie et 13h sur Vargas).

Cela dit, il faut tempérer ces résultats : en effet, le nombre d'itérations de calcul n'est pas identique d'un job de calcul à l'autre.
Le nombre d'itérations de calcul par job sur Curie a tendance à être plus faible qu'à l'IDRIS.

- Sur Curie certains jobs ont totalisé 18 itérations de calcul et d'autres, 24.
- Sur Vargas (IDRIS), certains jobs ont totalisé 18 itérations de calcul et d'autres, 48.

Ceci explique pourquoi le nombre de jobs de calcul au TGCC (89) est plus important qu'à l'IDRIS (53) pour une durée physique de simulation très similaire (environ 150 ans).
Pour comparer les performances en rapidité d'exécution sur les 2 centres, on ne peut donc pas comparer directement les temps d'exécution moyens des jobs de calcul.
En revanche, la comparaison sera possible si l'on connait le temps moyen d'exécution par itération sur les 2 centres.
La simulation sur Curie compte 1830 itérations et à l'IDRIS, 1867 itérations. En divisant le temps total d'exécution des jobs de calcul par le nombre d'itérations sur chaque centre, on obtient :

- Sur Curie, 1 itération prend en moyenne 14 minutes.
- Sur Vargas (IDRIS), 1 itération prend en moyenne 22 minutes.

Ainsi, la différence de rapidité entre les 2 centres n'est plus d'un facteur 3 mais d'un facteur 1,6.

4.1.4.2 Les jobs de post-traitement

Les différences sont similaires pour les jobs de « rebuild » (1h18 contre 3h35).
Cette différence est encore plus marquée pour les « TS » (18 minutes contre plus de 3 heures).

4.1.5 Conclusion / résumé

Cette analyse montre que les nœuds fins de Curie (dédiés au calcul) sont plus rapides que les nœuds sur Vargas. De la même manière, les nœuds larges de Curie (utilisés pour les post-traitements) semblent plus rapides que ceux sur Ulam.

On peut résumer cette étude par les graphiques qui suivent. Sur ces histogrammes, les colonnes en vert représentent des temps sur Curie (TGCC) et en marron des temps à l'IDRIS. Dans chaque colonne, il y a 3 niveaux : le temps minimum, le temps moyen, le temps maximum. Pour retrouver les résultats des analyses précédentes, il faut donc considérer le deuxième niveau de chaque colonne (temps moyen).

4.1.5.1 Temps d'attente

On donne sur la Figure 6 une vue d'ensemble des temps d'attente des jobs de la chaîne. On y voit la prédominance extrêmement nette des temps d'attente à l'IDRIS.
Les couleurs les plus foncées, représentant le temps d'attente minimum est rarement présent car il est nul pour la majorité des jobs.

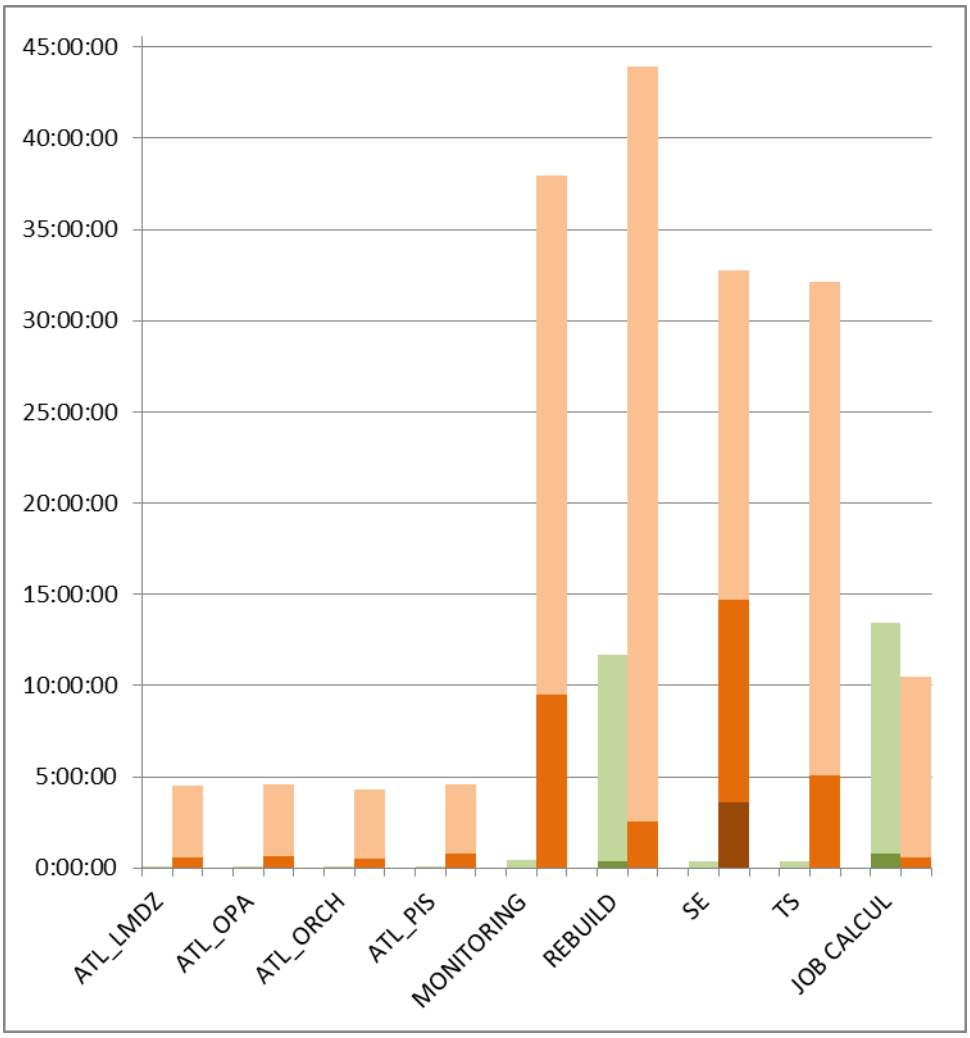


Figure 6 : temps d'attente (vue d'ensemble)

4.1.5.2 Temps d'exécution

Pour les jobs de calcul, on a vu plus haut qu'il était plus pertinent de considérer le temps moyen que prend 1 itération de calcul. La durée moyenne d'une itération de calcul sur les 2 centres figure sur les histogrammes qui suivent.

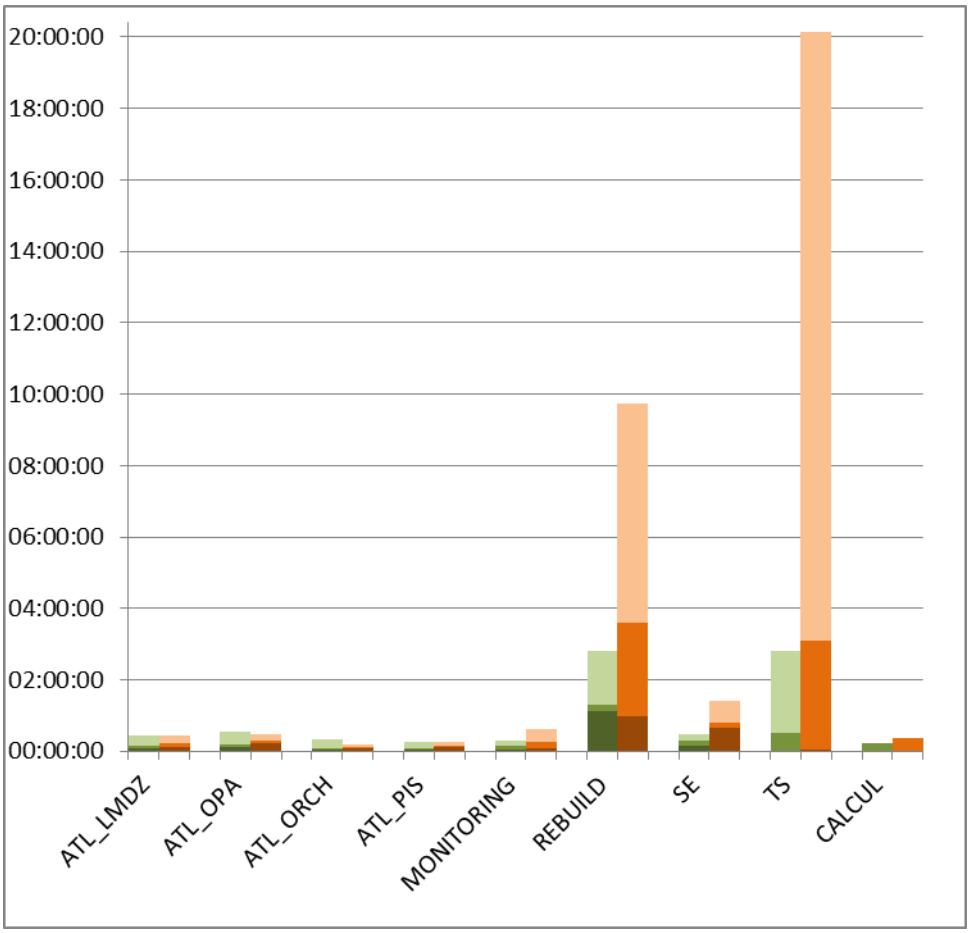


Figure 7 : temps d'exécution par job de post-traitement (vue d'ensemble)

Les colonnes relatives aux jobs de « rebuild » et de « TS » sont hautes et ne permettent pas de voir les détails des autres colonnes. Ils sont présentés sur cet autre histogramme :

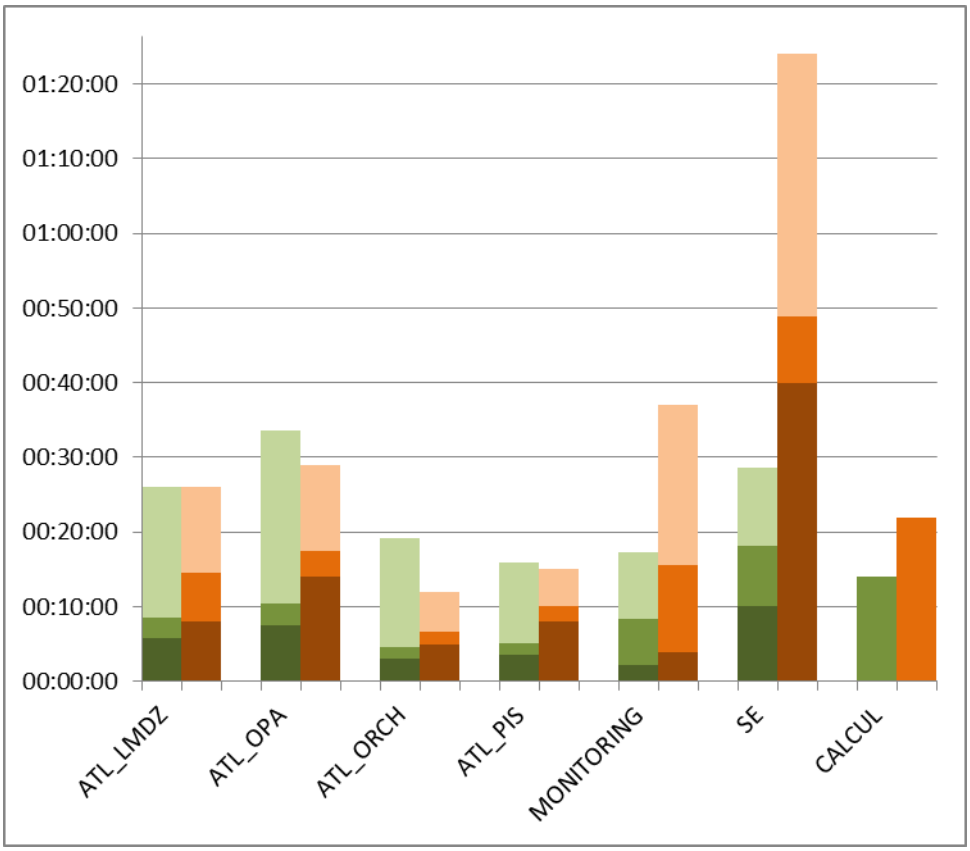


Figure 8 : temps d'exécution (détail de certains jobs)

4.2 Analyse fine des post-traitements (ou analyse de 2ème niveau).

4.2.1 Préliminaires

De manière générale, un job de post-traitement peut se diviser en 3 types de tâche :

- Importer les données localement avant traitement ;
- Traiter les données ;
- Exporter les données une fois traitées.

Les tâches d'importation dans les post-traitements sont matérialisées dans les scripts de « libIGCM » par des fonctions élémentaires « libIGCM » de type « get » : ce sont les « Get_Dir » et « Get ». Sur « Curie » (TGCC), ce sont en réalité des « cp », précédés de « ccc_hsm get» pour éventuellement dé-migrer les éléments à transférer. Sur « Ulam » (IDRIS), « Get_Dir » contient un appel à « rcp » et « Get » un appel à « mfget ».

Les tâches d'exportation ne sont représentées que par la fonction « libIGCM » « Put_Out ». Sur « Curie », cette fonction contient un appel à « cp », et sur « Ulam » un appel à « mfput ».

Les tâches de traitement des données peuvent se ranger en 2 catégories :

- **Les traitements des fichiers de résultats de simulation au format « netCDF** ». Ces traitements sont variés et ont pour but d'extraire des informations ciblées de ces fichiers pour établir des statistiques, de concaténer de manière cohérente un ensemble de ce type de fichiers, etc. Ces tâches sont matérialisées par les fonctions : « **ncap2** », « **ncatted** », « **ncks** », « **ncra** », « **ncrcat** » (qui font partie de l'outil « nco ») et « **nco** ». Les outils « cdo » et « nco » sont des outils de manipulation et d'analyse des

fichiers au format NetCDF. Ils sont installés sur les centres de calcul par les centres de calcul eux-mêmes à la demande de l'IPSL.

- **La recombinaison cohérente de résultats de simulation produits par les différents processus de calcul.** C'est la fonction « **rebuild** ». Cet exécutable, issu de sources fortran, est installé sur les centres de calcul par l'IPSL.

4.2.2 Objectifs

4.2.2.1 Part des importations, des traitements, des exportations

Dans un premier temps, on s'intéresse à la part de chaque tâche, en termes de durée de simulation, aussi bien au plan global que pour le cas particulier de chaque job de post-traitement. Plus précisément, on veut avoir une idée du pourcentage de temps passé à importer localement les données, à traiter ces données et à les exporter, à l'échelle de l'ensemble des post-traitements et à celle de chacun des post-traitements.

Attention : en réalité, les traitements élémentaires effectués par la chaîne dans chacun des jobs sont un peu plus nombreux mais on se restreint à ceux précédemment cités, qui sont les plus significatifs. Ils sont finalement appelés par les jobs suivants : les jobs de « rebuild », les « TS », les « SE » et pour Curie seulement, les jobs de « pack ». Les jobs de « pack » n'étant pas exécutés à l'IDRIS, on les exclut aussi de cette liste lorsqu'on effectue des comparaisons entre les centres.

Pour résumer, on s'intéresse donc à la part de temps passé sur chaque type de tâche pour l'ensemble des jobs retenus, et pour chacun de ces jobs sur chacun des centres.
Une fois les informations pertinentes recueillies pour cette étude à l'IDRIS et au TGCC, nous effectuons des comparaisons entre les centres.

4.2.2.2 Débits

Pour les tâches d'importation et d'exportation, la notion de débit est immédiate : c'est la quantité d'information transférée par seconde.
Concernant les tâches de traitement, on définit un débit comme la quantité d'information traitée par seconde.
Le débit que nous considérons est donc un débit étendu aussi au traitement ; c'est un indicateur de performance pour les 3 types de tâche.
Là encore, nous effectuons à la fois une analyse sur chacun des centres mais aussi des comparaisons entre les centres de calcul sur les résultats obtenus.

4.2.3 Conditions d'étude

Pour extraire les informations de durée et de débit dont nous avons besoin pour mener cette étude, nous avons « instrumenté » la chaîne de calcul, c'est-à-dire rajouté des instructions dans les scripts relatifs aux jobs de la chaîne. Ces instructions inscrivent des données dans les rapports d'exécution des jobs et des scripts ont été développés pour les extraire et les mettre en forme.
Deux simulations identiques et représentatives des simulations standards de l'IPSL ont donc été lancées sur chacun des centres en utilisant la chaîne de calcul instrumentée.

4.2.4 Les simulations analysées

4.2.4.1 Fréquence de lancement des jobs

Les jobs de « rebuild » ont été lancés après chaque année de simulation. Sur les 10 ans de simulations, 10 jobs de « rebuild » ont été lancés. Même chose pour les jobs de « pack » sur « Curie ».
Les autres jobs ont été lancés tous les 10 ans, c'est-à-dire 1 fois.

4.2.4.2 Au TGCC

Répertoire de soumission :
`/ccc/work/cont003/incka/pierreg/GP_EXPERIENCE/modipsl_2/config/IPSLCM5A/EXP00-10ans-instr`
Répertoire des rapports d'exécution des jobs :
`/ccc/scratch/cont003/incka/pierreg/IGCM_OUT/IPSLCM5A/DEVT/pdControl/EXP00-10ans-instr/Out`
Date de fin de simulation : 07/12/2012

4.2.4.3 A l'IDRIS

Répertoire de soumission : (sur « Vargas »)
`/workgpfs/rech/psl/rpsl947/GP_EXPERIENCE/modipsl/config/IPSLCM5A/EXP00-10ans-instr`
Répertoire des rapports d'exécution des jobs : (sur « Ulam »)
`/workdir/rech/psl/rpsl947/IGCM_OUT/IPSLCM5A/DEVT/pdControl/EXP00-10ans-instr/Out`
Date de fin de simulation : 24/11/2012

4.2.5 Performances par type de tâche

Les tableaux qui suivent donnent les performances des opérations élémentaires provenant des jobs de post-traitement exécutés sur chaque centre de calcul.
La 1ère colonne indique le nom de l'action instrumentée.
La colonne « nb de flux » donne le nombre d'appels à une opération au cours de l'exécution de la chaîne tandis que les 3 colonnes suivantes concernent les débits pour l'exécution d'un type d'opération élémentaire : débits « minimum », débits « maximum », et « moyenne » des débits. Ces débits sont exprimés en Mo par seconde.
La 6ème colonne donne le temps total pris pour exécuter l'ensemble des appels à un type d'opération. Ces durées sont exprimées en millisecondes.
La 7ème colonne indique l'espace disque où sont situées les données d'entrée d'une opération ainsi que l'espace disque où est situé le résultat du traitement. On remarque d'ailleurs ici que seules les opérations d'importation et d'exportation peuvent avoir des données d'entrée et de sortie sur un espace différent.

Action	nb de flux	minimum	maximum	moyenne	temps	depuis-vers
+ cdo	731	0.048828	24.002877	3.819191	259690	scratch-->scratch
+ Get_Dir	120	1.485950	30.715222	14.850255	16075079	scratch-->scratch
+ Get	33	0.157934	141.789306	79.400482	1641187	store-->scratch
+ Get	12	0.455728	1.813615	0.882582	891	work-->scratch
+ ncap2	15	5.815618	72.522759	33.651057	155106	scratch-->scratch
+ ncatted	389	0.067348	1008.697509	75.519026	12902	scratch-->scratch
+ ncks	731	0.032869	4.257812	1.505090	71500	scratch-->scratch
+ ncrcat	404	0.054252	161.906567	22.448468	793803	scratch-->scratch
+ Put_Out	1921	0.245659	279.044197	102.024970	1344607	scratch-->scratch
+ Put_Out	793	0.072337	296.996311	80.424637	121644	scratch-->store
+ rebuild	1921	0.000875	91.817505	14.815793	17854634	scratch-->scratch

Figure 9 : Performances des fonctions élémentaires sur Curie

Action	nb de flux	minimum	maximum	moyenne	temps	depuis-vers
+ cdo	731	0.076592	26.367187	6.672892	386078	ulam-->ulam
+ Get_Dir	120	2.853597	49.628897	31.176965	6396297	gaya-->ulam
+ Get	198	0.017657	131.742134	48.361181	1646056	gaya-->ulam
+ ncap2	15	16.673900	107.415718	89.959607	53340	ulam-->ulam
+ ncatted	389	0.260416	3250.000000	227.261937	4526	ulam-->ulam
+ ncks	731	0.048524	11.935763	3.529115	32569	ulam-->ulam
+ ncrcat	404	0.035672	176.227541	38.973110	244328	ulam-->ulam
+ Put_Out	2576	0.007088	154.987135	34.347391	1635210	ulam-->gaya
+ rebuild	1921	0.058506	144.809744	54.733793	2032987	ulam-->ulam

Figure 10 : Performances des fonctions élémentaires à l'IDRIS

Point important :

- Pour éclaircir certains points au cours de cette étude, les informations brutes issues de l'instrumentation de la chaîne (et avant calcul puis conditionnement sous forme de tableaux) ont été examinées. Pour plus de détails sur ces informations, on lira la partie 8.1.2.

4.2.5.1 Importations des données

4.2.5.1.1 L'opération « Get_Dir »

A l'échelle d'une simulation, les transferts par « Get_Dir » prennent nettement moins de temps à l'IDRIS qu'au TGCC. Le rapport des durées est d'environ 2,5. Ceci est confirmé par les valeurs des débits moyens.
Ce type de transfert sur Curie a pourtant pour source et destination le même espace (« SCRATCHDIR ») tandis qu'à l'IDRIS, ces transferts se font à partir de la machine « Gaya » vers la machine « Ulam ».

4.2.5.1.1.1 L'opération « Get »

Sur Curie, on distingue 2 types d'appels à « Get » : depuis le « STOREDIR » vers le « SCRATCHDIR » et depuis de « WORKDIR » vers le « SCRATCHDIR ».
On regarde plus spécifiquement le premier type d'appel qui porte sur des transferts de données de résultats de calcul de taille plus importante. Le second type d'appel transfère des informations de configuration de la chaîne.
Le nombre d'appels à la fonction « Get » est plus important à l'IDRIS qu'au TGCC. Le mode d'appel à cette fonction est un peu différent sur chaque centre : « Get » est appelé plus souvent à l'IDRIS car certains types de fichiers sont transférés un à un tandis qu'ils le sont par paquet au TGCC. La quantité de données transférée est néanmoins la même.
En comparant la somme des durées de transfert par « Get » au TGCC et à l'IDRIS, on constate qu'elles sont très similaires.
La moyenne de débit sur Curie est cependant sensiblement supérieure. Il n'y pas de contradiction cependant avec la grande similarité des temps de transfert.
En effet, le débit de transfert a tendance à être plus faible pour les petits fichiers, sans doute pour des raisons de temps de latence avant une copie d'un emplacement vers un autre. Or, à l'IDRIS, les données à transférer par « Get » sont plus morcelées : il y a donc plus de petits fichiers dont le débit de transfert compte autant que les gros dans la moyenne du débit mais dont le temps de transfert est faible. Ce sont les débits de transfert de ces petits fichiers qui font baisser la valeur du débit moyen à l'IDRIS.

4.2.5.2 Traitements des données

4.2.5.2.1 L'opération « cdo »

Cette opération prend globalement moins de temps au TGCC qu'à l'IDRIS. Pourtant, les moyennes de débit semblent indiquer le contraire.
Cette opération traite des fichiers de tailles assez inégales. Les plus gros de ces fichiers sont traitées nettement plus rapidement au TGCC et les plus petits nettement moins vite.
Ainsi la contradiction apparente s'explique à nouveau par le fait que les moyennes classiques donnent trop d'importance aux petits fichiers.

4.2.5.2.2 L'opération « ncap2 »

Ici les choses sont plus simples : aussi bien en termes de débit qu'en termes de durée globale, ce traitement se fait environ 3 fois plus vite à l'IDRIS.

4.2.5.2.3 L'opération « ncatted »

Tous les indicateurs montrent que cette opération s'effectue nettement plus vite à l'IDRIS, avec un facteur d'environ 3,5.

4.2.5.2.4 L'opération « ncks »

Tous les indicateurs montrent que cette opération s'effectue nettement plus vite à l'IDRIS, avec un facteur d'un peu plus de 2.

4.2.5.2.5 L'opération « ncrcat »

Là encore, ce type de traitement est plus rapide à l'IDRIS.

4.2.5.2.6 L'opération « rebuild »

A nouveau, on peut dire que cette opération est nettement plus rapide à l'IDRIS.

4.2.5.3 Exportation des données

On examine ici les performances de la fonction « Put_Out ».
Sur Curie, on distingue 2 types d'appels à « Put_Out » : depuis le « SCRATCHDIR » vers lui-même et depuis le « SCRATCHDIR » vers le « STOREDIR ». Les appels à « Put_Out » du premier type sont faits depuis des jobs de « rebuild ». Les appels du second type sont faits depuis des jobs « SE », « TS ».
A l'IDRIS, il n'y a qu'un type d'appel, de « Ulam » vers « Gaya » ; il regroupe les appels à partir des jobs « SE », « TS » et « rebuild ».
En examinant à la fois les moyennes des débits de transfert et les durées totales pour l'exécution de cette fonction, on constate que le transfert se fait un peu plus rapidement au TGCC qu'à l'IDRIS.

4.2.5.4 Conclusion

Les transferts dans le sens de l'importation se font sur une durée plus courte à l'IDRIS.
Les temps d'exportations sont comparables sur les 2 centres.
Les opérations de traitement sont nettement plus rapides à l'IDRIS si l'on excepte « cdo », qui est plus rapide au TGCC.

4.2.6 Part de chaque type de tâche

4.2.6.1 Sur l'ensemble des post-traitements

Les tableaux suivants montrent, pour chaque centre, la part de durée prise par chacune des tâches d'importation, de traitement et d'exportation.

Type action	Temps cumule	pourcentage	Actions correspondantes
importation	17717157	46.22	/Get/Get_Dir
traitement	19147635	49.95	/cdo/ncap2/ncatted/ncks/ncrcat/rebuild
exportation	1466251	3.82	/Put_Out

Figure 11 : Pourcentage des différentes tâches sur Curie

Type action	Temps cumule	pourcentage	Actions correspondantes
importation	8042353	64.69	/Get/Get_Dir
traitement	2753828	22.15	/cdo/ncap2/ncatted/ncks/ncrcat/rebuild
exportation	1635210	13.15	/Put_Out

Figure 12 : Pourcentage des différentes tâches à l'IDRIS

Au TGCC, les durées d'importation et de traitement sont les plus grandes, l'exportation n'occupant qu'une moindre place.
A l'IDRIS, les durées d'importation et de traitement sont nettement plus courtes qu'au TGCC (c'est particulièrement frappant concernant les traitements proprement dits), la durée d'exportation étant assez proche de celle au TGCC.
Dans ces conditions, l'aspect des colonnes de « pourcentage » est très différent d'un centre à l'autre :

- Au TGCC, la moitié du temps d'exécution des jobs de post-traitements est passée à exécuter les traitements tandis qu'à l'IDRIS, seul 1/5ème de ce temps est dédié aux traitements.
- A l'IDRIS, la part des importations est largement dominante, alors qu'au TGCC, ce sont les importations et les traitements qui dominent à part à peu près égale.

4.2.6.2 Par post-traitement

Les tableaux qui suivent montrent, pour chaque type de jobs, à la fois les durées d'exécution sur les tâches d'importation, de traitement proprement dit et d'exportation, ainsi que leur part relative en pourcentage.
On notera d'abord qu'au TGCC, les jobs de « TS » se divisent en 2 catégories : ceux qui traitent des données pour le 2D et ceux pour le 3D. Ces 2 types de traitements ont bien lieu à l'IDRIS mais ne sont pas distingués ici.
On remarque que les profils des tableaux sont assez différents d'un job à l'autre. Par exemple, au TGCC, le pourcentage du temps d'exécution des tâches d'importation est très différent selon qu'on considère les jobs « SE » ou les jobs « TS » pour le 3D. On peut dire la même chose à l'IDRIS.
On peut se demander comment, à partir de ces tableaux, on en arrive aux tableaux plus généraux de la Figure 11 et Figure 12. On remarque que les tableaux relatifs aux jobs de « rebuild » pour les 2 centres ont un profil très proche des tableaux généraux des Figure 11 et Figure 12.En effet, ces jobs (qui recombinent les résultats de simulation provenant de plusieurs

processus) sont les plus nombreux (car lancés à la plus grande fréquence) et prennent une part importante du temps d'exécution consacré aux post-traitements. On le constate d'ailleurs dans les colonnes « temps cumulé » des tableaux par job : le job « rebuild » est dominant en termes de durée d'exécution.

Ainsi le profil des tableaux généraux de la partie 4.2.6.1 est largement influencé par celui du tableau relatif au job de « rebuild ». Ceci se vérifie sur les 2 centres de calcul.

Type de job : create_se

Type action	Temps cumule	pourcentage	Actions correspondantes
importation	860238	77.61	/Get
traitement	227933	20.56	/ncap2/ncrcat
exportation	20169	1.81	/Put_Out

Type de job : create_ts__2D

Type action	Temps cumule	pourcentage	Actions correspondantes
importation	655963	62.45	/Get
traitement	366191	34.86	/cdo/ncatted/ncks/ncrcat
exportation	28200	2.68	/Put_Out

Type de job : create_ts__3D

Type action	Temps cumule	pourcentage	Actions correspondantes
importation	125877	14.01	/Get
traitement	698877	77.82	/cdo/ncatted/ncks/ncrcat
exportation	73275	8.15	/Put_Out

Type de job : rebuild_fromWorkdir

Type action	Temps cumule	pourcentage	Actions correspondantes
importation	16075079	45.57	/Get_Dir
traitement	17854634	50.61	/rebuild
exportation	1344607	3.81	/Put_Out

Figure 13 : Part des différentes tâches par job sur Curie

Type de job : REBUILDARCH

Type action	Temps cumule	pourcentage	Actions correspondantes
importation	6396297	65.65	/Get_Dir
traitement	2032987	20.86	/rebuild
exportation	1312345	13.47	/Put_Out

Type de job : SE

Type action	Temps cumule	pourcentage	Actions correspondantes
importation	752187	83.61	/Get
traitement	92580	10.29	/ncap2/ncrcat
exportation	54852	6.09	/Put_Out

Type de job : TS

Type action	Temps cumule	pourcentage	Actions correspondantes
importation	893869	49.93	/Get
traitement	628261	35.09	/cdo/ncatted/ncks/ncrcat
exportation	268013	14.97	/Put_Out

Figure 14 : Part des différentes tâches par job à l'IDRIS

4.2.6.3 Jobs de pack au TGCC

Comme expliqué précédemment, les jobs de « pack » ne sont pas lancés au cours d'une simulation s'exécutant à l'IDRIS. Ils n'ont donc pas été pris en compte dans les analyses précédentes. Néanmoins, on peut montrer la répartition des différentes tâches pour ces types de jobs au sein d'une simulation exécutée au TGCC.

Type de job : pack_debug

Type action	Temps cumule	pourcentage	Actions correspondantes
importation	55892	10.06	/Get
traitement	422375	76.05	/tar
exportation	77119	13.88	/Put_Out

Type de job : pack_output

Type action	Temps cumule	pourcentage	Actions correspondantes
importation	38398	0.52	/Get
traitement	5066718	69.49	/ncrcat
exportation	2185193	29.97	/Put_Out

Type de job : pack_restart

Type action	Temps cumule	pourcentage	Actions correspondantes
importation	57322	0.99	/Get
traitement	2754057	47.72	/tar
exportation	2959085	51.27	/Put_Out

Figure 15 : les jobs de pack au TGCC

Les tâches d'importation sont quasi inexistantes. En effet, il s'agit ici de l'importation de fichiers de configuration ou d'avancement de la simulation, qui sont petits et n'entrent pas directement en jeu dans la fonction de réduction du nombre de fichiers effectuée par ces jobs.
On remarque la domination nette de la part des traitements de « tar » ou de concaténation dans ces jobs.
On note aussi que les durées relatives à ces jobs ne sont pas négligeables en comparaison de celles qu'on trouve dans les tableaux des autres jobs du TGCC. Cela tient au fait que, dans nos simulations, les jobs de pack ont été lancés avec la même fréquence que les jobs de « rebuild » (après chaque année de simulation), contrairement aux autres jobs (tous les 10 ans). D'ailleurs, seuls les jobs de « rebuild » ont une durée cumulée supérieure.
Les jobs de « pack output », qui sont les plus longs en termes de durée d'exécution, dure environ 700 secondes, alors qu'un job de « SE » dure à peu près 1000 secondes sur « Curie ».

5 Fiabilité

L'évaluation de la fiabilité est basée d'une part sur l'étude des interruptions ou perturbations dont peuvent être sujettes les simulations au cours de leur exécution, et d'autre part sur le recensement des problèmes rencontrés par l'auteur de ce document pendant la réalisation de cet audit.

5.1 Les interruptions de simulations

5.1.1 Interruptions courantes

On revient ici sur les 2 simulations (une sur chacun des centres) qui ont servi pour l'étude des temps d'attente et des temps d'exécution, rapportée dans la partie 4.1 de ce document.
La liste des causes d'interruption donnée ci-dessous n'est pas exhaustive.

5.1.1.1 TGCC

La simulation exécutée sur Curie s'est arrêtée et a du être relancée une quinzaine de fois. La figure suivante illustre le déroulement de la simulation.

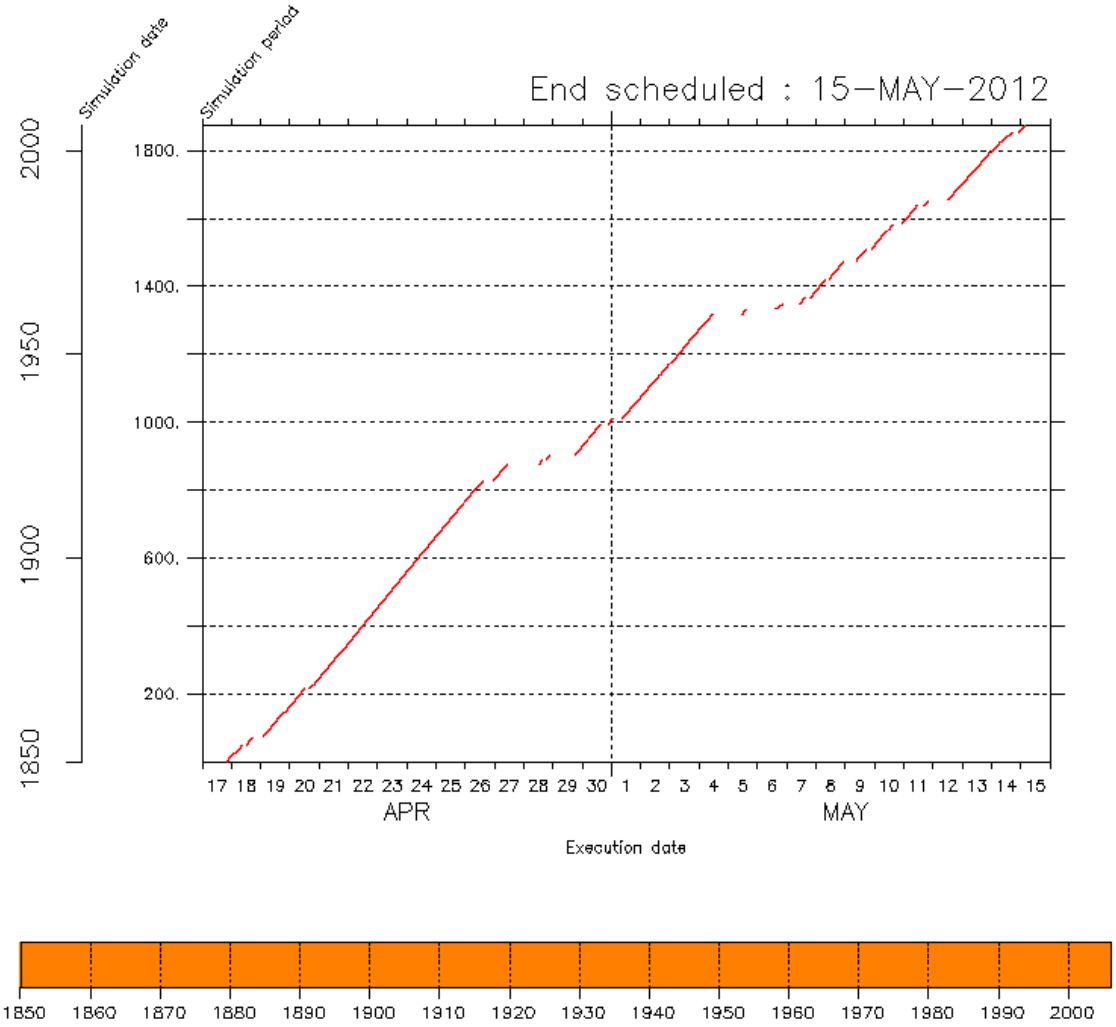


Figure 16 : progression de la simulation du 4.1 sur Curie

Les messages relatifs aux erreurs ayant entraîné ces interruptions sont présents dans les rapports d'exécution des jobs de calcul et sont de 3 types :

Message d'erreur 1 :

```
/tmp/p86maf_run_file_RYo8dr_5526_350274: line 19: 5767: Killed
srun: error: curie4617: task 8: Exited with exit code 9
srun: First task exited 600s ago
srun: tasks 0-7,9-31: running
srun: task 8: exited abnormally
srun: Terminating job step 350274.0
slurmd[curie3430]: *** STEP 350274.0 KILLED AT 2012-04-20T19:19:23 WITH SIGNAL 9 ***
slurmd[curie6106]: *** STEP 350274.0 KILLED AT 2012-04-20T19:19:23 WITH SIGNAL 9 ***
slurmd[curie4617]: *** STEP 350274.0 KILLED AT 2012-04-20T19:19:23 WITH SIGNAL 9 ***
slurmd[curie6462]: *** STEP 350274.0 KILLED AT 2012-04-20T19:19:23 WITH SIGNAL 9 ***
srun: Job step aborted: Waiting up to 2 seconds for job step to finish.
slurmd[curie4617]: *** STEP 350274.0 KILLED AT 2012-04-20T19:19:23 WITH SIGNAL 9 ***
slurmd[curie6462]: *** STEP 350274.0 KILLED AT 2012-04-20T19:19:23 WITH SIGNAL 9 ***
slurmd[curie3430]: *** STEP 350274.0 KILLED AT 2012-04-20T19:19:23 WITH SIGNAL 9 ***
slurmd[curie6106]: *** STEP 350274.0 KILLED AT 2012-04-20T19:19:23 WITH SIGNAL 9 ***
```

Ce message, connu des utilisateurs de la chaîne de calcul comme un problème provenant des machines de calcul, est référencé dans la page web suivante : http://forge.ipsl.jussieu.fr/igcmg/wiki/Modipsl_curie#Erreursfr%C3%A9quentessurcurielorsdelexecutiondessimulations

Ce souci provenait d'un bug de la commande « ccc_mprun » lorsqu'elle était utilisée avec l'option "-f" pour faire du couplage de codes en mode MPMD comme c'est le cas pour le modèle IPSL : la raison du bug était que les exécutables n'étaient pas broadcastés sur les nœuds de calcul. Il est à noter que ce souci a été pris en considération par le TGCC et que, suite à une demande de l'IPSL, le ccc_mprun a été modifié par le TGCC le 19 juillet 2012.

Message d'erreur 2 :

```
probleme IO sur scratch :
lmdz.x: posixio.c:248: px_pgin: Assertion `*posp == ((off_t)(-1)) || *posp == lseek(nciop->fd, 0, 1)' failed.
forrtl: error (76): Abort trap signal
```

Ce message, dont la première ligne est explicite, est un problème provenant des machines de calcul.

Message d'erreur 3 :

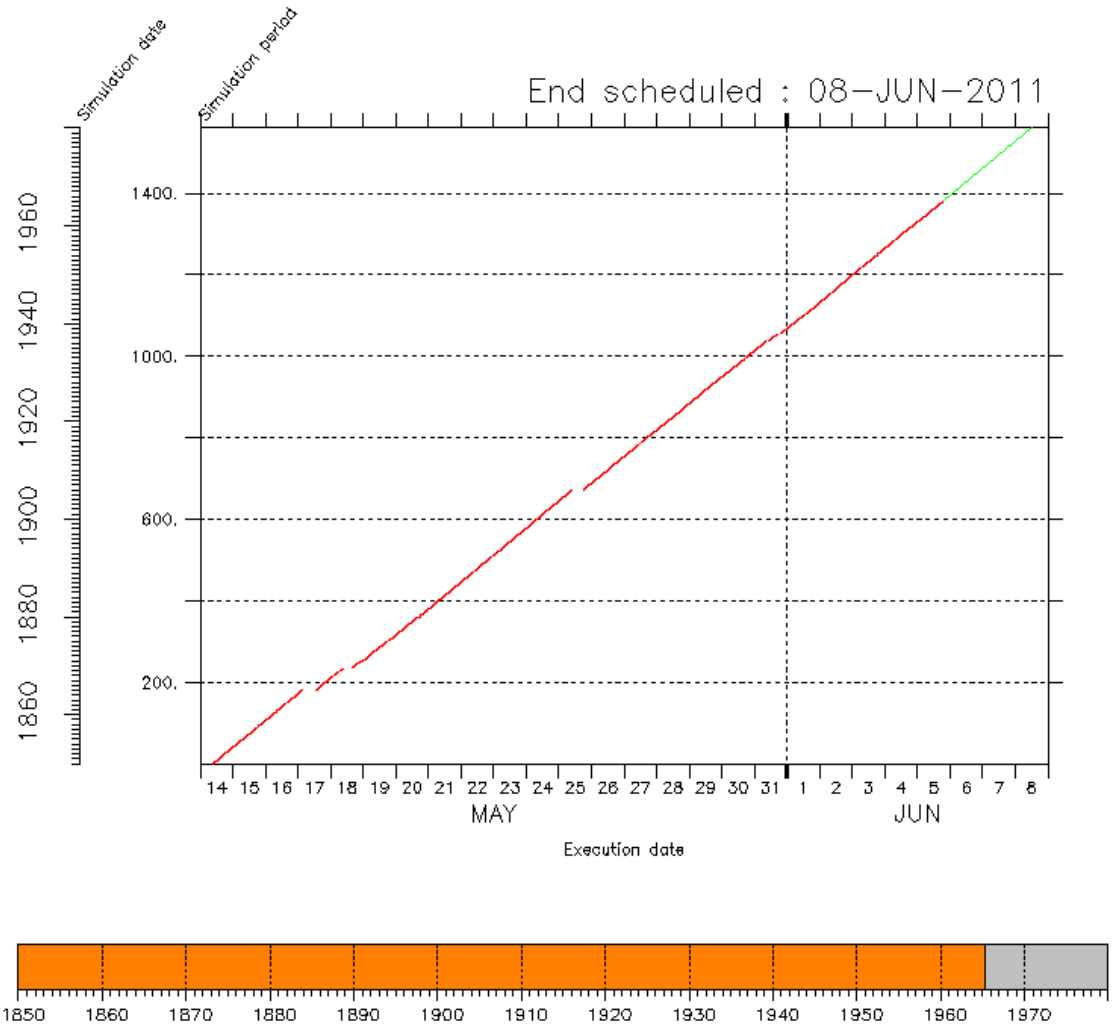
```
IGCM_sys_Mv : SISUTESW_out.1987-10-01T00:00:00.nc /ccc/scratch/cont003/dsm/p86maf/RUN_DIR/375309/IPSLCM5A/v3.hist
IGCM_sys_Mv : SIICECOV_out.1987-10-01T00:00:00.nc /ccc/scratch/cont003/dsm/p86maf/RUN_DIR/375309/IPSLCM5A/v3.hist
IGCM_sys_Mv : SIICTEMW_out.1987-10-01T00:00:00.nc /ccc/scratch/cont003/dsm/p86maf/RUN_DIR/375309/IPSLCM5A/v3.hist
IGCM_sys_Mv : SIICEALW_out.1987-10-01T00:00:00.nc /ccc/scratch/cont003/dsm/p86maf/RUN_DIR/375309/IPSLCM5A/v3.hist
IGCM_sys_Mv : COTAUXXU_out.1987-10-01T00:00:00.nc /ccc/scratch/cont003/dsm/p86maf/RUN_DIR/375309/IPSLCM5A/v3.hist
IGCM_sys_Mv : CURRENTX_out.1987-10-01T00:00:00.nc /ccc/scratch/cont003/dsm/p86maf/RUN_DIR/375309/IPSLCM5A/v3.hist
IGCM_sys_Mv : CURRENTY_out.1987-10-01T00:00:00.nc /ccc/scratch/cont003/dsm/p86maf/RUN_DIR/375309/IPSLCM5A/v3.hist
IGCM_sys_Mv : CURRENTZ_out.1987-10-01T00:00:00.nc /ccc/scratch/cont003/dsm/p86maf/RUN_DIR/375309/IPSLCM5A/v3.hist
IGCM_sys_Mv : COTAUXXV_out.1987-10-01T00:00:00.nc /ccc/scratch/cont003/dsm/p86maf/RUN_DIR/375309/IPSLCM5A/v3.hist
IGCM_sys_Mv : error in mv.
IGCM_debug_Exit :  IGCM_sys_Mv
```

On voit qu'un ensemble de fichiers est déplacé d'un emplacement à un autre, mais que l'un de ces déplacements (le dernier) rencontre un problème. Il s'agit là encore d'un problème machine.

Les erreurs 2 et 3 correspondent à des bugs /scratch connus ; ils ont été corrigés sur les nœuds « xlarge » à la fin octobre 2012 et corrigés sur l'ensemble de la configuration de « Curie » le 20 novembre 2012.

5.1.1.2 IDRIS

La simulation exécutée sur les machines de l'IDRIS s'est arrêtée et a dû être relancée 3 fois. La figure suivante illustre le déroulement de la simulation.



Les messages d'erreur rencontrés sont reproduits ici :
Message d'erreur 1 :

```
connect to address ::ffff:130.84.196.29: Connection refused
...
No POST-TREATMENT avaible because ulam is down
```

Message d'erreur 2 :

```
connect to address ::ffff:130.84.196.29: Connection timed out
...
ulam.idris.fr: Connection timed out
```

Le premier message est du à une indisponibilité de la machine « Ulam » sur laquelle les post-traitements sont effectués, et le second à un problème de connexion entre machine de l'IDRIS.

5.1.1.3 Conclusion

Les interruptions entraînent inévitablement un allongement des temps de simulation et un coût supplémentaire en temps humain, qui correspond au temps passé par l'utilisateur à identifier la cause de l'échec, puis à relancer la simulation.

Avant les corrections apportées, ce problème d'interruption était particulièrement présent sur Curie : la simulation s'est interrompu 15 fois (coût humain) et la somme des durées pendant lesquelles la simulation n'a pas tourné représente environ 25% de la durée totale de la simulation. Comme expliqué précédemment, le travail réalisé par le TGCC a permis de corriger beaucoup de problèmes qui engendraient des arrêts de la simulation : la fiabilité de Curie s'est donc bien améliorée diminuant ainsi le temps humain nécessaire.

5.1.2 Autres interruptions

Des interruptions ayant d'autres causes peuvent survenir. L'auteur de ce document les ayant rencontrées au cours de l'audit, Elles sont indiquées ici.

- Si le mot de passe du compte utilisateur expire, comme c'est arrivé sur la machine « Vargas » de l'IDRIS, la simulation est arrêtée.
- Après les travaux de maintenance des machines du TGCC, les simulations interrompues par ces travaux ne sont pas relancées. Ceci était dû à une option « norerun » des jobs de post-traitements sur curie, ajoutée volontairement, qui a été enlevé début décembre 2012.

5.2 Perturbations

5.2.1 Inégalités dans les durées d'un même job

Il s'agit ici d'analyser la reproductibilité des performances obtenues sur un job, ou autrement dit comment expliquer l'écart entre les temps minimum et moyen d'exécution de certains jobs (voir les Figure 4 et Figure 5) ?

5.2.1.1 Les jobs de calcul

A l'IDRIS, par exemple, le temps minimum d'un job de calcul est de 24 minutes et le temps moyen d'un tel job est de 13 heures. Après vérification, ce job de calcul de 24 minutes ne comporte qu'une seule itération. A noter que ce job d'une seule itération était une astuce nécessaire pour démarrer d'un restart NEMO « mono-processus » alors que le modèle NEMO tournait sur 5 processus : c'est corrigé depuis dans l'environnement « libIGCM ». Nous ne pouvons donc pas tirer de conclusion sur cette disparité dans les temps d'exécution des jobs de calcul à l'IDRIS.
En revanche, si on examine les mêmes données pour les jobs de calcul au TGCC, la différence est moins nette.

5.2.1.2 Les jobs de post-traitement

Les figures suivantes représentent le temps d'exécution des jobs de « rebuild » au cours de 10 ans de simulations. La fréquence de « rebuild » sur ces 10 ans est de 1 an ce qui permet donc de comparer 10 jobs de « rebuild » consécutifs à la fois sur le TGCC (Figure 17) et l'IDRIS (Figure 18). On s'aperçoit que sur les nœuds larges de Curie il y a une grande disparité dans le temps d'exécution de ces jobs-là, les durées du job variant de 6h30 à 30 min. Cette hétérogénéité est beaucoup moins nette voire même quasiment nulle sur « Ulam » à l'IDRIS.

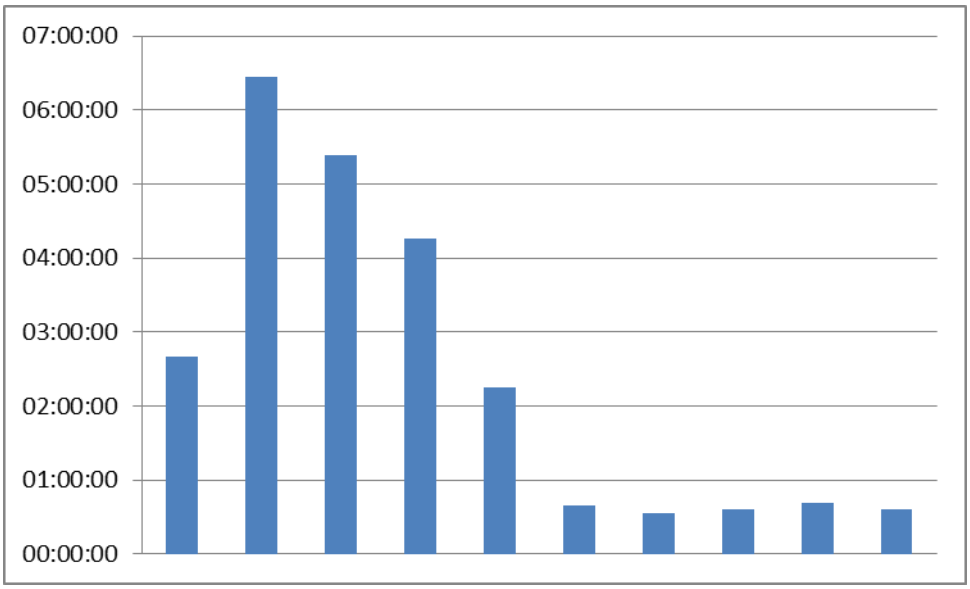


Figure 17 : durées des jobs de « rebuild » sur Curie sur 10 ans de simulation (1 rebuild par an)

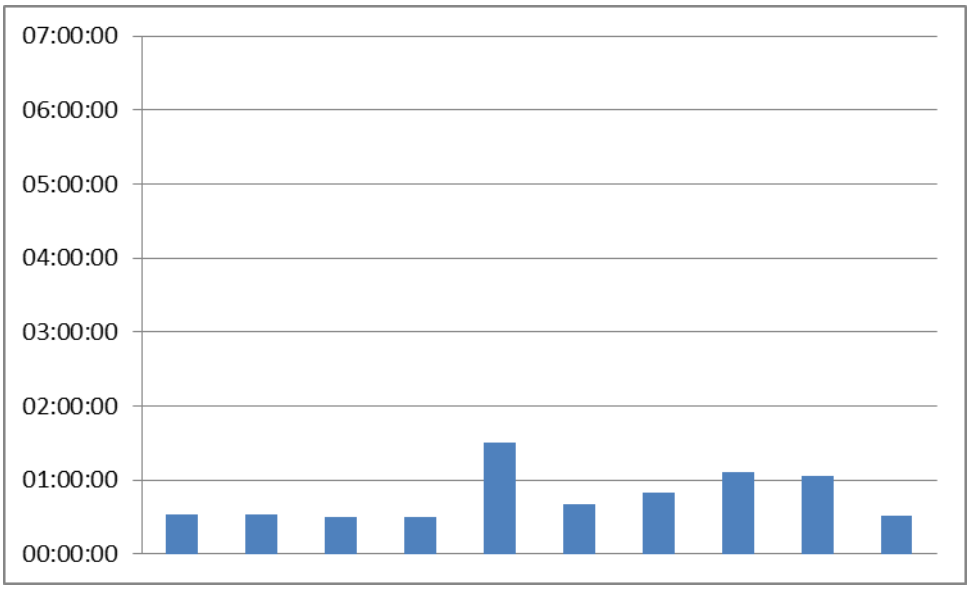


Figure 18 : durées des jobs de « rebuild » sur Ulam sur 10 ans de simulation (1 rebuild par an)

5.2.2 Débits

A la manière de l'analyse précédente sur la variation de durée d'un même job au cours d'une simulation, on peut se pencher sur les différents débits obtenus et s'étonner d'une part de la variation entre les débits moyens et maximum et d'autre part de l'écart entre le débit maximum atteint et le débit théorique du « filesystem » utilisé. Par exemple, on note sur le tableau 9 un débit moyen pour la commande « ncrcat » (essentiellement des écritures) de 22 Mo/s (« scratchdir » vers « scratchdir ») pour un débit maximum de 161Mo/s. Ce débit moyen pourrait être défini comme un débit « réel » assez éloigné du débit maximum, lui-même n'atteignant pas le débit théorique (400Mo/s sur le scratchdir Curie). Il est fort probable que ces disparités des débits jouent un rôle important dans les inégalités observées sur le temps des différents jobs. Ce même constat peut être réalisé à la vue des débits obtenus à l'IDRIS (tableau 10) : 38Mo/s de débit moyen et 176Mo/s pour le débit maximum obtenu.

5.3 Autres problèmes rencontrés

5.3.1 La commande « du »

Comme indiqué dans la partie 8.1.2, l'extraction des données pour l'étude de 2ème niveau (voir partie 4.2) a pu être réalisée grâce à l'insertion de nouvelles instructions dans les scripts des jobs de post-traitements. Certaines de ces instructions comportent un appel à l'utilitaire « du » qui retourne la taille d'un fichier ou répertoire qu'on lui fournit.
Au TGCC, cette commande ne donne pas un résultat correct si elle est exécutée sur un fichier qui vient d'être créé. Pour que son résultat soit correct, il faut attendre quelques secondes avant de lui faire évaluer la taille du fichier. La raison est très certainement liée à des effets de cache qui apparaissent ou des latences réseau.
Ainsi a-t-il fallu imaginer une autre solution pour récupérer la taille des fichiers sur « Curie », basée sur la commande « ls -lrtk ».

6 Pistes d'amélioration - Recommandations

6.1 Amélioration sur la chaîne

6.1.1 Performances

Tous les jobs de post-traitement comportent, entre autre, des tâches de traitements et des tâches d'exportation du résultat des traitements.
D'une manière générale, une des préoccupations pour améliorer les performances de la chaîne sera de trouver comment réduire la durée associée à chaque type de tâche (importations, traitements et exportations).
Au TGCC, éliminer les tâches d'exportation en créant directement les produits des traitements à leur emplacement final (sur l'espace « CCCSTOREDIR ») entraînerait à la fois un gain de temps et un gain en fiabilité. C'est d'ailleurs ce qui est recommandé par le TGCC.
Concernant les tâches de traitement, les options de compilation (OpenMP,…) ainsi que les paramètres d'exécution (taille des blocks, paramètres des filesystems,…) réduiraient vraisemblablement leur durée d'exécution.

6.1.2 Mesure de performance

Les travaux réalisés au cours de cet audit permettent l'extraction d'informations sur les performances de la chaîne ainsi que leur conditionnement sous forme de statistiques. Ils peuvent aisément être intégrés dans une version de production de la chaîne (à ce sujet, voir partie 8.1.2). On pourrait ainsi prévoir l'insertion dans la chaîne de nouveaux jobs faisant ce travail d'extraction et de conditionnement pour une analyse de 1er niveau.
Concernant les informations ayant permis l'étude de 2ème niveau, l'instrumentation de la chaîne pourrait être conservée, moyennant quelques améliorations. On pourrait par exemple publier ces statistiques en ligne et les rendre consultables pendant le déroulement de la simulation sous forme de monitoring.

6.1.3 Améliorations de la fiabilité de la chaîne

6.1.3.1 Réduction du nombre de tâches de la chaîne

La fiabilité de la chaîne dépend de l'enchaînement réussi ou non des différentes tâches qui la composent. La réduction du nombre de ces tâches ne peut qu'impliquer une amélioration de la

fiabilité de la chaîne. Une des étapes de cette chaîne est l'étape de « rebuid » qui permet de recombiner les fichiers sorties par les processus de calcul d'un modèle en un seul fichier. L'utilisation à venir de la librairie d'IO « XIOS », développée à l'IPSL, permettra la réalisation d'IO parallèles et il sera donc possible d'obtenir en sortie du modèle un fichier déjà recombiné de façon performante. L'étape de « rebuild » sera donc supprimée des étapes de post-traitement.

6.1.3.2 Suivi et contrôle en ligne de la chaîne

Afin de diminuer le temps humain nécessaire à la réalisation d'une simulation complète (calcul + post-traitements), plusieurs aspects de la chaîne sont à améliorer, l'environnement actuel libIGCM permet le lancement et l'enchaînement des nombreuses tâches de la chaîne de calcul mais ne permet pas un suivi et un contrôle précis des différentes étapes de la chaine. Le développement de ces fonctionnalités de contrôle dans libIGCM ou bien l'utilisation d'un outil de type « superviseur » serait la solution. Ces nouvelles fonctionnalités permettraient idéalement plusieurs choses :

- Monitorer l'ensemble des tâches (visualisation de l'avancement).
- Détecter les erreurs de la chaine (chacun des jobs) et pour les rendre consultables facilement pour l'utilisateur.
- Analyse, correction et relance éventuelle des tâches.

6.1.4 Réduction du nombre d'inodes stockés

La réduction du nombre d'inodes produit par la chaîne et stocké sur le système de stockage est une contrainte imposée par le TGCC. Cette réduction apparaît être également un confort supplémentaire à la fois au sein même de la chaîne (moins de fichiers à copier d'un filesytem à l'autre) et pour l'utilisation a postériori des fichiers stockés (moins de fichiers à démigrer), C'est pourquoi comme nous l'avons vu, l'étape de concaténation et d'archivage de fichiers a été également introduite dans la chaîne sur la nouvelle machine de l'IDRIS Ada. Dans le but de réduire encore plus ce nombre de fichiers stockés, plusieurs pistes sont à envisager :

- l'utilisation de la librairie d'IO « XIOS » développée à l'IPSL. Son intégration dans les modèles de l'IPSL permettrait de gérer plus facilement l'activation (ou la désactivation) de certaines sorties dans les modèles de l'IPSL et donc de n'avoir en sortie des modèles que certaines variables pour les simulations non destinées à une utilisation par une large communauté. Il devrait également permettre de tourner des runs annuels (plutôt que mensuel) et d'avoir donc au final moins de fichiers produits.
- l'utilisation de l'outil de pack « pack_ipsl » développé dans le cadre de cette même prestation de support à la migration des données pourraient réduire le nombre d'inodes déjà existant sur le sytème de stockage. On pourrait ainsi envisager de « packer » les résultats d'une simulation déjà stockés de la même manière que le TGCC « packe » les simulations présentes sur le DMFDIR avant migration vers le STOREDIR. Pour réaliser cette opération le TGCC utilise en production l'outil « pack_ipsl ».

6.2Amélioration sur les scripts d'extraction

Dans l'optique d'intégrer à la chaîne les scripts développés au cours de l'audit, des idées d'amélioration sont proposées ici. Elles portent surtout sur les travaux ayant débouchés sur l'analyse de 2ème niveau concernant les post-traitements.

- Pour compléter les informations déjà présentes dans les tableaux (produits par les scripts), on pourrait y faire figurer la taille des données traitées ;
- La mesure de la taille des données traitées se fait par un « ls » au TGCC et par un « du » à l'IDRIS. En effet, une mesure de la taille d'un fichier au TGCC avec « du » est fausse si elle est faite juste après la création du fichier. Pour une mesure correcte, il faut attendre plusieurs secondes. On a donc préféré « ls -lrtk ». A l'IDRIS, l'outil « du » ne pose pas de problème. Cependant, les mesures de taille avec ces 2 outils produisent des résultats un peu différents, surtout s'agissant des fichiers de moins d'1 Mo. La mesure des tailles demande donc à être harmonisée sur les centres.
- Une moyenne de débit de transfert n'est pas un indicateur satisfaisant. Il donne trop d'influence aux petits fichiers. On pourrait réfléchir à se doter d'une moyenne pondérée.

6.3 Conseils sur les relations IPSL/centres et centres/IPSL

Les interactions entres les centres de calcul et les utilisateurs/développeurs de l'IPSL sont essentielles. Une très bonne connaissance de la façon de travailler de l'IPSL (éléments techniques de la chaine de calcul) par les centres de calcul permettrait aux centres :

- d'alerter la communauté des utilisateurs IPSL (mail, site internet) sur des problèmes ponctuels qui impacteraient spécifiquement la chaîne de calcul
- d'alerter les développeurs de la chaîne de calcul IPSL sur des modifications apportées aux calculateurs (versions de software, variables d'environnement, …) et susceptibles d'impacter la chaîne.
- de travailler en amont avec les développeurs de la chaîne de calcul afin de déterminer des choix techniques optimum en regard des machines utilisées.
- de choisir les futures configurations de calculateurs ou de système de stockage de concert avec les besoins et contraintes de la communauté climat IPSL.

Les interactions entre l'IPSL et les centres se font déjà de plusieurs façons : telcos ou réunions régulières, participation à des projets communs (projets ANR), participation aux comités utilisateurs. La possibilité d'avoir un interlocuteur privilégié au sein de chaque centre améliorerait certainement la collaboration entre les centres de calcul et l'IPSL.

7 Retour des centres sur le présent document

L'auteur de cet audit a rendu visite aux centres de calcul pour recueillir les impressions des personnes concernées par ce document. Ces visites ont donné lieu à des discussions autour des points suivants :

- Perception par les centres de leurs interactions avec l'IPSL.
- La chaîne de calcul de l'IPSL
- Le présent document

Cette partie est un compte rendu de ces discussions.

7.1 Remarques du TGCC

La réunion s'est tenue le 5 juillet 2013 au TGCC en présence de Bruno Frogé assisté de l'équipe d'administration et de support du TGCC.

7.1.1 Interactions entre le TGCC et l'IPSL

Les spécificités de la chaîne de calcul de l'IPSL n'ont pas toujours été en accord avec les contraintes techniques du TGCC : c'était le cas notamment du nombre d'inodes (alors trop important et désormais correct) produit par la chaîne de calcul de l'IPSL.

Aujourd'hui, un dialogue plus fructueux semble avoir été établi, preuve en est ce présent rapport imaginé conjointement par le TGCC et l'IPSL pour tenter de réduire encore les obstacles à une collaboration fluide.
Il comporte une description de la chaîne de calcul de l'IPSL : l'objectif est de contribuer à hausser le niveau de connaissance des centres de calcul concernant cette chaîne.
Le TGCC trouve cette description appropriée et est demandeur de ce type d'information.

7.1.2 La chaîne de calcul de l'IPSL

Pour le TGCC, l'IPSL est une grosse communauté (comparée aux autres utilisateurs de ses calculateurs) et sa chaîne de calcul est complexe :

- Par le nombre de jobs important qu'elle génère
- Par le nombre de fichiers qu'elle produit
- Par l'hétérogénéité de son modèle couplé

Toute simplification de cette chaîne est encouragée par le TGCC.
Par exemple, le TGCC approuve les modifications à venir sur la chaîne concernant la librairie « XIOS » développée par l'IPSL (voir 6.1.3 et 6.1.4). Cet outil permettra de réduire le nombre de jobs lancés et le nombre de fichiers produits.
D'une manière générale, le TGCC recommande à l'IPSL :

- De prendre en compte les contraintes du système de fichier « Lustre » ;
- De minimiser les transferts de fichiers autant que possible ;
- De grouper les jobs autant que faire se peut pour en limiter le nombre et réduire le temps passé en queue ;
- De veiller à ce que le temps de calcul soit consommé régulièrement tout au long de l'année.
- Dans la mesure du possible, de lancer les jobs de post-traitements sur les nœuds standard pour éviter les situations d'engorgement. Cela dit, le TGCC indique que faire réaliser les post-traitements par des jobs différents des jobs de calcul est conforme aux bonnes pratiques.

7.1.3 Remarques sur l'audit

- Temps d'attente des jobs avant exécution (voir 4.1) : le TGCC pointe le caractère fluctuant des temps d'attente. L'examen d'une seule simulation ne saurait donner des durées représentatives de ce paramètre.
- Aspects techniques : il aurait été intéressant donner plus de précisions sur les caractéristiques techniques des machines et des systèmes de fichiers utilisés pour expliquer un peu mieux les performances de la chaîne de calcul sur les 2 centres.
  Par exemple, au TGCC, le système « Lustre » gère plus efficacement les gros fichiers que les petits.
- La commande « ls » dans l'instrumentation de la chaîne de calcul (voir 4.2 et 5.3.1) : Cette commande est en effet utile pour récupérer des volumes de fichiers. Il faut cependant lui préférer la commande « stat », plus adaptée. En effet, « ls » extrait des informations inutiles, ce qui la rend plus lente.

- Données manquantes : sur les Figure 9 et Figure 10 de 4.2.5, il manque la taille des fichiers traités. Cette information est intéressante car, pour les transferts notamment, il existe un temps de latence qui rend les opérations plus lentes pour les petits fichiers.
- Durée des jobs de « rebuild » : au vu de la Figure 17, le TGCC s'étonne de la variété des durées des jobs de « rebuild ». Il propose une analyse plus détaillée afin de déterminer si le « scratchdir » est en cause. Une solution pourra être imaginée après analyse.

7.2 Remarques de l'IDRIS

La réunion s'est tenue le 5 juillet 2013 à l'IDRIS en présence de Pascal Voury.

7.2.1 Interactions entre l'IDRIS et l'IPSL

Du point de vue de l'IDRIS, les interactions entre l'IPSL et l'IDRIS sont satisfaisantes.

Pour l'IDRIS, le processus de collaboration est constructif : l'IPSL y met de la volonté et des moyens humains (en temps). La communauté de l'IPSL est structurée dans le sens où elle présente des interlocuteurs privilégiés dans ses relations avec l'IDRIS, où elle a fédéré les besoins (utilisation commune d'une chaîne de calcul) et les logiciels.
L'IPSL a fait le choix du parallélisme pour ses modèles (tous les consommateurs de temps de calcul à l'IDRIS n'ont pas fait ce choix et utilise les machines en mode séquentiel), qui est une utilisation avancée des ressources en calcul.
Les discussions sur les modifications de la chaîne sont amorcées avec l'IDRIS suffisamment en avance pour laisser le centre s'organiser (l'IDRIS cite l'exemple de l'augmentation de la résolution); il peut s'agir d'une période de 3 à 4 ans. L'IDRIS trouve cet aspect très positif.

7.2.2 La chaîne de calcul de l'IPSL

Pour l'IDRIS, les logiciels nécessaires à l'exécution de la chaîne de calcul ou à l'exploitation de ses résultats sont en nombre trop important au sens où quelques-uns sont redondants ou semblent inutiles.
Par exemple, les outils « CVS » et « SVN » (tous les deux des outils de gestion de configuration) doivent ils coexister sur ses machines de l'IDRIS ? « Firefox » n'est-il pas trop lourd s'il s'agit simplement de lire des fichiers au format « html » ? Comment s'explique la présence de « Latex » ?
Des outils non nécessaires entraînent un coût non justifié en maintenabilité d'anciennes versions sur les machines de l'IDRIS. Par exemple, sur le temps passé par l'IDRIS pour parvenir à ce que la chaîne de calcul s'exécute utilisant « Ada » (qui remplace « Ulam »), la moitié a concerné l'installation des outils (les problèmes restant, de transfert entre « Ada » et « Gaya », ont d'ailleurs été évoqués lors de l'entretien).
Pour ces raisons, l'IDRIS recommande à l'IPSL de faire une revue des logiciels de la chaîne et d'établir un classement par priorité.
Une simulation avec la chaîne de calcul réclame régulièrement l'exécution de 600 jobs. D'après l'IDRIS, c'est beaucoup, mais la chaîne est bien adaptée à ses machines. Sur ce centre, les jobs de post-traitement sont exécutés sur des nœuds dédiés au post-traitement et dont l'utilisation n'est pas décomptée des demandes d'heures annuelles. Ce dispositif est nouveau et on manque de recul pour évaluer son impact sur les performances de la chaîne.

7.2.3 Remarques sur l'audit

L'IDRIS fait remarquer que la description de la chaîne de calcul est intéressante.
Ces informations auraient par exemple été utiles pour dimensionner la machine « Ulam » il y a quelques années.

8 Annexes

8.1 Fonctionnement des outils d'extraction d'informations

Nous décrivons ici les méthodes employées pour extraire les informations sur la base desquelles nous avons mené ces études.

8.1.1 Temps d'attente et d'exécution des jobs de la chaîne (analyse de 1er niveau)

Ces informations sont extraites en 2 temps.
Ces 2 étapes sont différentes dans la forme selon qu'elles servent à extraire des informations depuis une simulation ayant tourné à l'IDRIS ou au TGCC.

Dans un premier temps, on veut se doter d'un fichier « bilan » qui indique le nom de chaque job de la chaîne (de calcul et de post-traitement) ainsi que ses temps d'attente et d'exécution ou des informations pour y parvenir, comme sa date de soumission, sa date de début et de fin d'exécution.
Ce fichier peut contenir plusieurs milliers de lignes si, comme c'est le cas pour cette partie de l'étude, on s'appuie sur des simulations couvrant 150 ans et commandant certains types de post-traitement tous les ans.

Dans un second temps, on dresse les tableaux que l'on peut voir sur les Figure 4 et Figure 5, à partir de ce fichier de bilan. Les données brutes de ce fichier ont été traitées de manière à calculer des moyennes ou des sommes de durée par type de job.

8.1.1.1 TGCC

8.1.1.1.1 Fichier bilan

Au TGCC, un rapport d'exécution de job (établi par la chaîne calcul) est systématiquement agrémenté d'une « banderole » finale contenant les dates de soumission, de début et de fin d'exécution.
Ainsi, le script chargé de construire le fichier de bilan boucle sur les rapports d'exécution des jobs, récupère ces dates et calcule les temps d'attente et d'exécution au format « hh:mm:ss ».

En pratique, ce script se lance de la façon suivante :

```
./bilan.sh -dir ${repRapports} -run ${runCard} -y 1850-2000 -o
${outputStep_1_File}
```

Commentaires sur les options du script :

- « -dir » demande le chemin vers le répertoire contenant les rapports d'exécution des jobs,
- « -run » demande le chemin du fichier « run.card » (avancement de la simulation),
- « -y » la période de simulation à laquelle on veut se restreindre,
- « -o » le nom du fichier bilan produit.

Toutes ces options sont à renseigner obligatoirement.

8.1.1.1.2 Données finales

Il suffit ensuite d'exploiter le fichier de bilan pour calculer pour chaque job, des durées minimum, maximum, moyennes et sommes d'attente et d'exécution.
Le script chargé de ces traitements s'utilise comme suit :

```
./stat-niv1-curie.sh -f output_step_1.txt -y 1850-2000 –o
resultats_step_2.txt
```

Commentaires sur les options du script :

- « -f » demande le chemin du fichier de bilan,
- On peut encore restreindre l'établissement des statistiques à une période donnée via l'option « -y ».
- « -o » demande le nom du fichier de statistiques

8.1.1.2 IDRIS

8.1.1.2.1 Fichier bilan

Les calculateurs de l'IDRIS n'adjoignent pas d'informations de date ou de temps concernant l'attente et l'exécution dans les sorties des jobs. Il faut procéder autrement.
Il existe des traces d'exécution de jobs à l'IDRIS et pour y accéder, on se sert de la commande « jar » à partir d'une des machines de l'IDRIS :
Sur « Vargas » (machine où s'exécutent les jobs de calcul de la chaîne), on peut par exemple taper :

```
jar –v –e 13/05/2011-16/06/2011 > bilan-job-vargas.txt
```

pour obtenir les dates de soumission, de début et de fin d'exécution de tous les jobs exécutés sur « Vargas » entre le 13 mai et le 16 juin 2011 et les recueillir dans un fichier. Ce fichier contiendra donc les informations souhaitées concernant les jobs de calcul.
Les dates appliquées à la commande « jar » ont été choisies de manière à ne récupérer que des informations concernant une simulation ciblée. Par chance, entre ces dates, seule la simulation qui nous intéresse a tourné. Mais en imaginant que le même utilisateur ait lancé d'autres simulations, ou bien des jobs issus d'un tout autre calcul, les informations recherchées se seraient trouvées parmi d'autres informations indésirables.

La même commande tapée à partir d'« Ulam » permet d'obtenir les mêmes informations sur les jobs de post-traitement.
Il suffit ensuite de concaténer manuellement les bilans issus de « Vargas » et d'« Ulam » pour obtenir un bilan de l'ensemble des jobs d'une simulation.

8.1.1.2.2 Données finales

Le format du fichier de bilan produit par la commande « jar » n'est pas celui du fichier de bilan provenant de l'analyse des rapports d'exécution des jobs au TGCC.
Il faut donc traiter ce fichier spécifiquement afin d'obtenir des statistiques comparables à celles obtenues sur le TGCC.

8.1.2 Analyse des post traitements (analyse de 2ème niveau)

On indique ici comment ont été extraites les informations permettant l'analyse des post-traitements.
On a procédé en 3 temps :

- Instrumentation de la chaîne ;
- Ecriture et exécution d'un script permettant la récupération des informations « brutes » provenant de l'instrumentation dans un fichier de bilan ;
- Ecriture et exécution d'un script permettant le réarrangement des données recueillies à l'étape précédente dans le fichier de bilan sous forme de tableaux.

Ces étapes sont les mêmes sur les 2 centres de calcul, et les scripts sont très similaires.

8.1.2.1 Instrumentation de la chaîne de calcul

La chaîne de calcul a été instrumentée dans le but de récupérer, pour chaque opération élémentaires (« Get », « Get_Dir », « Put_Out », « cdo », « ncap2 », …), des informations concernant la taille (et si possible le nom) des données traitées, le temps pris pour les traiter, le job à partir duquel est lancé l'opération et les espaces des données d'entrée et de sortie de l'opération.
Cette « instrumentation » a été réalisée sur les machines « Curie » (TGCC) et « Vargas » (IDRIS).
Comme déjà dit dans 4.2.2, les jobs concernés par l'instrumentation sont les jobs suivants :

- rebuild
- TS
- SE

Et sur « Curie » uniquement :

- pack_debug
- pack_output
- pack_restart

Chacun de ces jobs est matérialisé par un script dans la couche « libIGCM ». Par exemple, le script correspondant au job « SE » se nomme « create_se.job ».
Par ailleurs, chaque job produit un rapport d'exécution.
Instrumenter la chaîne de calcul signifie donc ici rajouter des instructions en langage « ksh » dans les scripts des jobs précédents permettant de récupérer les informations pertinentes pour l'étude 4.2.
Le résultat de cette instrumentation se trouve dans les rapports d'exécution des jobs : après chaque trace d'exécution d'une opération élémentaire (« Get », « Get_Dir », « Put_Out », « cdo », « ncap2 », …), on imprime une ligne respectant le format suivant :

==> act :<action réalisée>|sz :<taille en mo>|ms:<durée en millisecondes>|
fx(mo):<flux en mo/s>|nm:<nom du fichier traité>|
dirSource:<emplacement d'origine du fichier traité>|
dirOut :<emplacement du résultat du traitement>

Dans le format indiqué ci-dessus, deux informations différentes sont séparées par un caractère « | ».
Si, pour une opération élémentaire (par exemple « Get »), plusieurs fichiers sont traités, le nom du fichier traité est indiqué comme inconnu en affectant à l'information « dirSource » le mot-clé « doNotKnow »).

Toutes ces informations, présentes dans différents rapports d'exécution de job, sont d'abord agrégées et réarrangées dans un seul « fichier de bilan », puis ce fichier de bilan est lu et les informations qui s'y trouvent sont exploitées pour obtenir les tableaux des Figure 9 à Figure 14.

Ces deux étapes sont réalisées par 2 scripts différents. Ils sont identiques selon qu'ils sont exécutés au TGCC ou à l'IDRIS.

8.1.2.2 Fichier bilan

Un script a été développé pour lire les traces de l'instrumentation dans les rapports d'exécution des jobs de post-traitement. Il réunit et classe toutes ces informations dans un fichier « bilan ». Les informations sont classées par type d'opération élémentaire : pour chaque type (par exemple « cdo »), un tableau recense toutes les opérations de ce type effectuées au cours la simulation et donne pour chaque opération sa taille, sa durée, …

La construction du fichier bilan est réalisée par un script « bilan_Post_niv2.sh » qui s'utilise comme suit :

```
./bilan_Post_niv2.sh -dir ${repRapports} -o output_step_1.txt
```

Commentaires sur les options du script :

- « -dir » demande le chemin vers le répertoire contenant les rapports d'exécution des jobs,
- « -o » le nom du fichier bilan produit.

Toutes ces options sont à renseigner obligatoirement.

8.1.2.3 Obtention des tableaux finaux

Un autre script permet ensuite d'obtenir les tableaux des Figure 9 à Figure 14. Ce script s'utilise comme suit :

```
./stat-niv2.sh -f output_step_1.txt –o resultats_analyse.txt
```

Commentaires sur les options du script :

- « -f » demande le chemin du fichier de bilan,
- « -o » demande le nom du fichier de statistiques contenant les tableaux.

8.2 Outils nécessaires pour faire tourner la chaîne de calcul sur une machine

- ksh
- svn
- fortran
- c++
- python
- rebuild (outil fortran) installé
- netcdf/4.1.3 (avec (dépendance hdf5) + netcdf4 + dapclient)
- nco/4.0.8 (avec (dépendance hdf5) + netcdf4 + dapclient + SMP + Optimization)
- hdf5 (clarifier les différentes versions, hdf, phdf)
- cdat/5.2.0
- R
- udunits/2.1.5
- ferret/6.7.2
- netpbm
- imagemagick

- tetex-latex
- cdo/1.5.2 (avec dapclient : compiler avec "CFLAGS=".... -DHAVE_LIBNC_DAP")
- RSYNC
- ncl/6.0.0
- VTK
- Subversion
- Paraview
- gnuplot
- Firefox
- ghostscript
- Pouvoir envoyer des mail depuis les jobs de calcul.