GTL : Une interface User-Friendly pour l'analyse de données biologique pour le Centre Jean Perrin

Sébastien Guizard, Matthieu Reichstadt, Yannick Bidet, A Perin

GTL : Une interface User-Friendly pour l'analyse de données biologique pour le Centre Jean Perrin

S. Guizard (1), M. Reichstadt (2), Y. Bidet (3), A. Perin(3)

(1) Laboratoire de Physique Corpusculaire,CNRS/IN2P3, 63000 Clermont Ferrand, France
(2) INRA Clermont-Theix, 63122 SAINT-GENES CHAMPANELLE, France
(3) Centre de Lutte Contre le Cancer, 63000 Clermont-Ferrand, France

Contexte Biologique :

Le laboratoire d'Oncologie Moléculaire du Centre Jean Perrin s'est équipé d'un pyroséquenceur haut débit Roche, le GS-FLX Titanium, capable de délivrer plus d'un million de séquences par run. Ce débit rend impossible l'analyse manuelle des données et nécessite l'utilisation d'un programme de traitement automatisé. La société Roche fournit une suite logicielle assurant le filtrage des données selon leur qualité, ainsi qu'un assemblage sommaire. Il est cependant évident que ce niveau d'analyse est largement insuffisant et que de nouveaux outils informatiques doivent être développés.

Ces outils doivent être d'autant plus flexibles et faciles d'utilisation qu'ils doivent être utilisés à la fois pour des projets de recherche internes et pour des prestations de service à des laboratoires extérieurs.

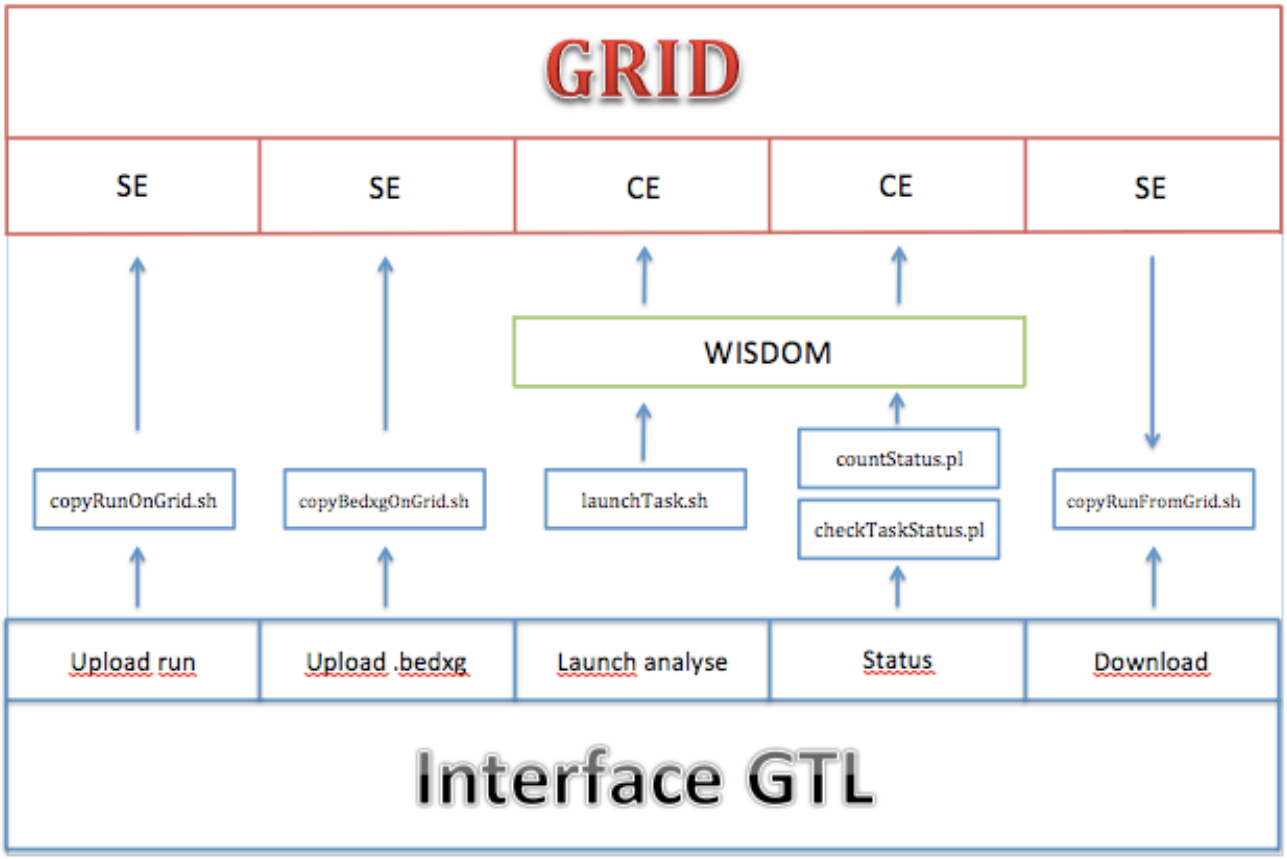
Un des logiciels créé au laboratoire a pour but de détecter et d'annoter les variants dans les lectures pré-assemblées par le logiciel Roche. Cette étape permet un premier tri des séquences sur lesquelles se concentrer ou non. Afin de limiter le temps de calcul, cette détection n'est pas réalisée sur tout le génome de l'espèce séquencée mais seulement sur les régions d'intérêt. L'interface graphique permet d'entrer ces régions au logiciel *via* un fichier texte au format .bed. Une copie de ce fichier traité est stockée au format .bedx afin de ne pas répéter le traitement lors d'une nouvelle analyse sur les mêmes régions-cibles. L'interface permet également de choisir les données à analyser (dossier de sortie du logiciel Roche) et de paramétrer les seuils de détection des variants (couverture minimale et pourcentage de variation). Le langage JAVA a été privilégié pour développer cette application interne afin de s'affranchir des problèmes de portabilité d'un système d'exploitation à un autre. Il permet également une grande souplesse dans la création des interfaces graphiques.

Ce programme a déjà été grandement utilisé par le laboratoire sur leurs machines dédiées au calcul et intégrant huit processeurs double cœur. Cependant, et malgré l'utilisation de la technologie multi-thread proposée par JAVA, les temps d'exécution du programme sont trop longs pour une utilisation régulière du logiciel. En effet, l'analyse d'un run typique de GS-FLX prend 4 à 5 jours. Lorsque le nombre de régions à annoter est particulièrement important ce temps peut s'allonger à plus d'une semaine. Ces temps de calculs nécessaires à l'obtention de données utilisables par les biologistes se posent en goulet d'étranglement à l'utilisation du séquenceur lui-même puisqu'un run de séquence est réalisé en 14h.

Au vu des besoins en calcul de l'équipe de recherche du Laboratoire d'Oncologie Moléculaire, mais également de la plateforme de Service « GINA » (Génotypage Intensif en

Auvergne) - http://www.cjp.fr/divers/gina_2008.pdf -, il a été mis en place une interface web facilitant l'utilisation de la grille pour les utilisateurs non initiés. En effet, celle-ci fait totalement abstraction des couches logicielles inhérentes à la grille rendant l'accès aux ressources simple et intuitif.

Il a donc fallu développer les scripts nécessaires aussi bien coté client que du coté serveur.



L'analyse

L'analyse se déroule en 2 étapes :

Coté client :

L'interface client doit tout d'abord permettre de contrôler l'accès des utilisateurs à l'interface pour éviter qu'elle ne soit exposée à une utilisation abusive des ressources. De ce fait, un système de login a été instauré. Celui-ci est en cours d'amélioration, en prenant en compte le système d'authentification de la grille. Ainsi l'objectif est de permettre uniquement aux utilisateurs possédant un certificat d'utiliser le programme.
Dans le cadre de notre travail, cela n'a volontairement pas été mis en place afin de proposer un service fonctionnel le plus rapidement possible.

Si l'utilisateur ne s'est pas enregistré sur la base de données, il ne pourra accéder uniquement qu'aux pages de login de l'application, au formulaire de création de compte et à la page de contact des administrateurs de l'application.

Lorsque l'utilisateur est logué, il peut accéder aux différentes parties de l'utilitaire :

- **Upload .bedgx** : page permettant d'envoyer sur le serveur le fichier de description de la puce sur le serveur
- **Uploadrun** : page permettant d'envoyer le fichier de run sur le serveur

- **Launch analyse** : page se rapprochant le plus à l'interface originelle du programme AgsA_Shotgun. Elle propose de sélectionner les fichiers de run et .bedxg à utiliser pour l'analyse ainsi que de paramétrer le nombre minimum de reads ainsi que le pourcentage de mutations. Une fois paramétré, il suffit de cliquer sur le bouton « Launch ! » pour que l'analyse soit lancée sur grille.
- **Status** : page permettant de visualiser l'avancement de l'analyse en affichant le statut des tâches sur la grille de calcul
- **Download** : page permettant à l'utilisateur de télécharger les résultats de l'analyse

Cette application se reprochant le plus possible du programme d'origine ne déstabilisera pas les utilisateurs en ne montrant pas toute la complexité de la grille et de l'usage de WISDOM.

Coté serveur :

Pour pouvoir exploiter au maximum les capacités de la grille, il à été décidé de faire appel à l'environnement de production WISDOM. Cette couche logicielle composée de quatre éléments (AMGA, Jobmanager, Taskmanager, Datamanger) fait appel au « pull model » offrant une optimisation de l'utilisation des jobs sur la grille.

Le serveur doit se charger de façon automatique du pré-traitement des données afin de les adapter et de les rendre disponibles sur la grille. En effet, pour pouvoir paralléliser l'exécution du programme et il est nécessaire de diviser le travail en tâches plus petites. Dans le cas du programme AgsA_Shotgun, le fichier divisible est le .bedxg.

Pour ce faire, un cron a pour mission de détecter la présence du nouveau fichier .bedxg, de le découper en paquet de 500 lignes, puis de les stocker sur des Storage Element (SE), sélectionnés sur les sites français. Ainsi, le fichier de run pourra être analysé contre tous les fragments de fichiers dans le même temps.

Le fichier de run doit lui aussi être copié sur la grille. C'est pour cela qu'un deuxième programme démon surveille le téléchargement d'un nouveau et le copie sur un SE français sur la grille.

Il va de soi que ces fichiers ne sont visibles sur l'interface uniquement que quand le stockage sur grille est complet.

Lorsqu'une analyse a été demandée par un utilisateur de l'application, un troisième cron va lancer les tâches sur la grille avec les paramètres souhaités et stocker les identifiants des tâches dans un fichier. Par la suite deux autres scripts mettront régulièrement leur statut à jour et comptabilisera le nombre de tâches « Waiting », « Running » et « Done ».

Pour finir un dernier script vérifie en permanence la présence de résultats sur la grille, les rapatrie au fur et à mesure sur le serveur, et, quand l'analyse est terminée, fusionne tous les fichiers et les préparent pour pouvoir être téléchargés par l'utilisateur.

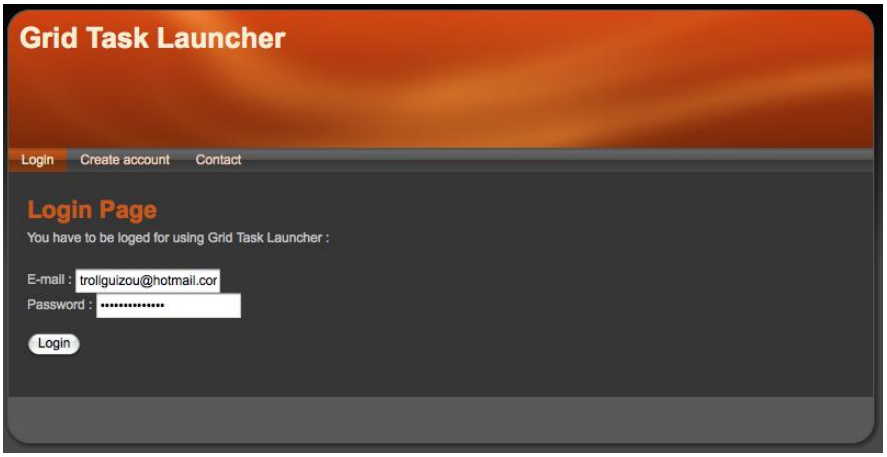
L'utilisation de la Grille est donc fortement liée à la seconde étape. En effet, il était primordial de pouvoir compter sur un grand nombre de nœuds de calcul afin de pouvoir effectuer toutes les étapes sensibles dans un laps de temps très restreint :

- Copie du fichier de configuration se plusieurs Go sur les SE de la Grille et découpage de ce fichier en milliers de sous-fichiers
- Lancement des analyses en utilisant ces fichiers de config, ce qui donne au final des runs de 200 000 lignes

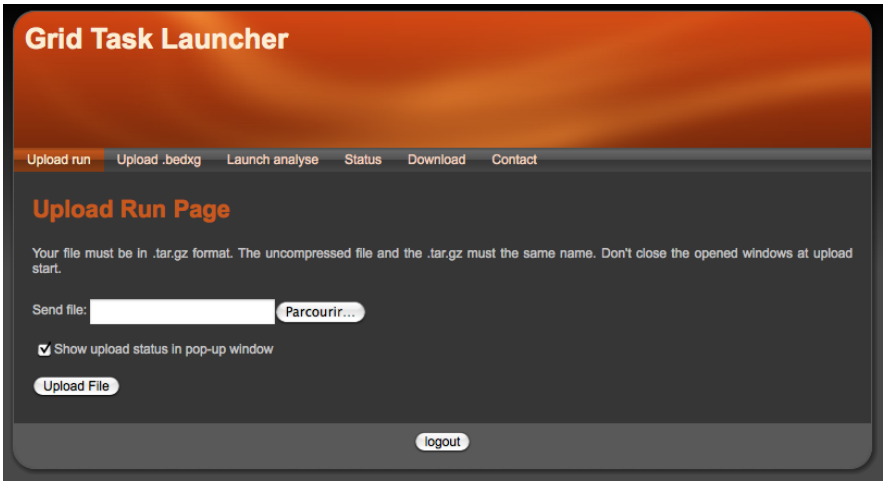
Le point crucial qui ne pouvait être débloqué que grâce à la grille est le fait que ces runs peuvent être lancés à tout moment, ce qui fait qu'il est nécessaire d'avoir disponible une grande quantité de CPU de libre à tout instant. Le principe du pull model de WISDOM s'est donc imposé à nous, pour éviter les attentes inhérentes à la soumission des jobs en push model.

De plus, le principe de fonctionnement de WISDOM permet aux différents calculs (runs) effectués de se terminer en erreur, la tâche sera automatiquement remise en file d'attente afin

d'être relancée sur un autre CE.

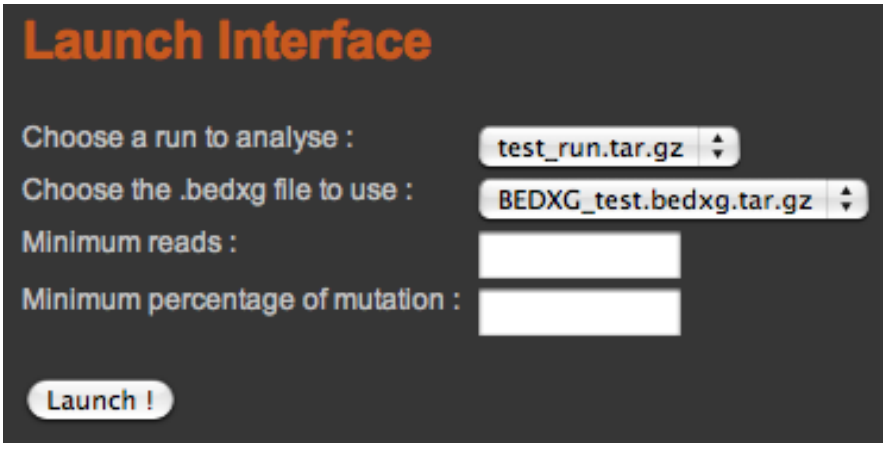


Page d'accueil

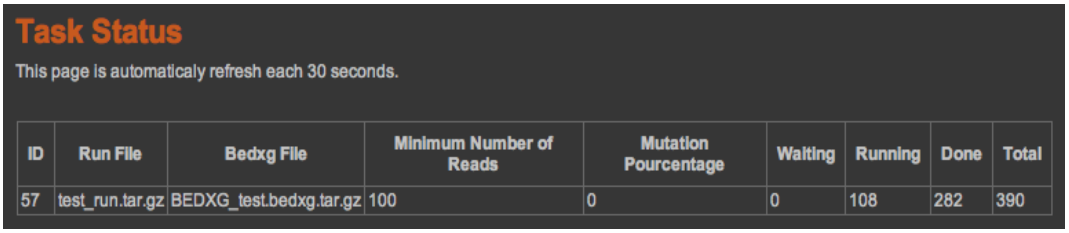


Accès aux fonctions de l'interface une fois connecté

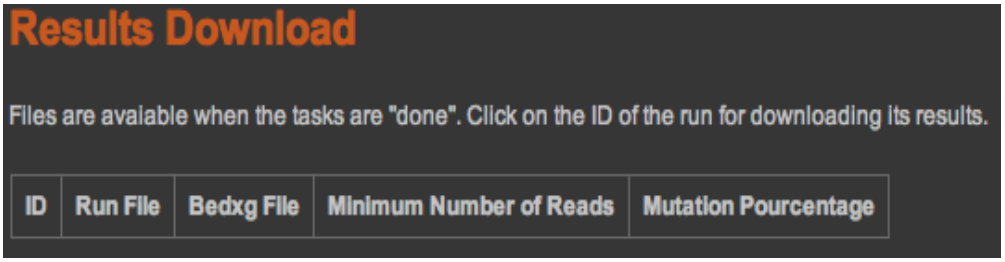
Page d'envoi des fichiers



Lancement du service



Page de suivi du statut des tâches



Page de téléchargement des résultats

Résultats

L'un des problèmes majeurs lié à l'usage d'internet dans ce genre d'application se situe dans la difficulté à transférer et traiter des fichiers de tailles importantes. Dans le cadre de notre application, les fichiers peuvent faire plusieurs Go, ce qui pénalise énormément la partie envoi de fichier sur le serveur. Celui-ci est donc impossible via une méthode classique (temps de téléchargement très important, blocage par les restrictions du serveur). Pour outrepasser ce problème il a été fait usage de Xupload permettant de télécharger plusieurs giga-octets de données en très peu de temps : *4 à 5 min pour 4Go*

Les précédentes exécutions du programme AgsA_Shotgun sur le serveur de calcul du Centre Jean Perrin ont montré que le programme peut s'exécuter pendant plusieurs semaines pour finaliser l'analyse.

Lors des phases de tests de l'application nous avons utilisé un jeu de donnéesdont les résultats étaient déjà connus. L'analyse complète de ce jeu de données sur les machines dédiées au calcul de l'équipe de recherche demande au programme AgsA_Shotgun 3 semaines d'exécution. Ce temps est entre autres expliqué par de nombreux crashs du programme inhérent à la grande quantité de calculs effectuée.

Lorsque la même analyse est lancée en utilisant l'application développée, sous condition que l'environnement de production WISDOM aitmis en œuvre un nombre suffisant de jobs (agents) sur la grille, la totalité de l'analyse est exécutée en l'espace de quelques heures. Ce temps comprend uniquement l'analyse et exclut les phases de téléchargement des fichiers. En effet, la copie du fichier .bedxg sur la grille est très longue car ce dernier est divisé en plusieurs centaines de fragments qui sont copiés en triple exemplaire sur trois différents Storage Element pour être disponibles sur la grille en permanence même si l'un d'entre eux vient a tomber en panne.

Conclusion

Toute la complexité de ce programme tient dans le fait qu'il a fallu mettre en place une application de type web mais permettant de faire des calculs que seule la grille permet de mettre en place. La relation web/grille a été le point crucial du développement, et l'utilisation de l'environnement WISDOM a permis d'obtenir des résultats très encourageants. Les temps de traitement sont ainsi passés de 4-5 jours à 2 heures lorsque la grille était totalement utilisable.

La prochaine étape dans la mise en place de cette application et l'amélioration de la stabilité de l'utilisation de la grille, ainsi qu'une meilleure répartition des charges entre CE. Une réécriture du programme dans un langage facilement interprétable par tous les CE est également envisagée.

Enfin une amélioration avec cryptage des données est envisagé, afin d'améliorer encore la confidentialité et la sécurité des données. Ainsi ce programme pourra être utilisé par l'intégralité des centres anti-cancéreux français, dans le cadre de leur recherche.