

Conservatoire National des Arts et Métiers
INSTITUT D'INFORMATIQUE D'ENTREPRISE

Mastère Spécialisé en Bio-Informatique

**Exploration Analyse et Prédiction de Réseaux de
Contacts entre Protéines et d'Arbres Phylogénétiques**

Auteur: Perrine Barjou

Directeur de recherche : M. Thierry ROSE
Unité d'Immunogénétique Cellulaire
Institut PASTEUR
Paris.

Descriptif: A partir d'une protéine requête, il faut pouvoir fournir à l'utilisateur un outil lui permettant d'afficher tous les partenaires de cette protéine sous forme d'un réseau graphique. Pour que les requêtes soient le plus étendues possibles, la conception et la gestion d'une base de données occupa une grosse partie de ce projet.

Conservatoire National des Arts et Métiers
INSTITUT D'INFORMATIQUE D'ENTREPRISE

MEMOIRE

présenté en vue d'obtenir

le Mastère Spécialisé en Bio-Informatique

Perrine BARJOU

**Exploration Analyse et Prédiction de Réseaux de
Contacts entre Protéines et d'Arbres Phylogénétiques**

Directeur du stage : M. Thierry ROSE, chargé de recherches
INSTITUT PASTEUR
Unité d'Immunogénétique Cellulaire
25, rue du Docteur Roux
75 724 PARIS Cedex 15

Soutenu le 29 septembre 2004 devant le jury

Mme C.DUBOIS
M. T. ROSE
M. A. CORNUEJOLS

REMERCIEMENTS

A l'occasion de l'écriture de ce mémoire, je souhaite remercier les personnes qui m'ont accompagnée, aidée et encadrée durant ce stage à l'Institut Pasteur.

Tout d'abord je tiens à remercier Thierry Rose, chargé de recherches au sein l'unité d'Immunogénétique Cellulaire, pour m'avoir proposé de travailler avec lui sur un projet passionnant dans le domaine de la recherche, pour son accueil, sa disponibilité et son écoute.

Je remercie le Professeur Jacques Thèze pour m'avoir accueillie en tant que stagiaire au sein de son laboratoire d'Immunogénétique Cellulaire.

Merci aussi à Virginie Mazard Pasquier et à tous les membres du laboratoire d'Immunogénétique Cellulaire pour leur accueil chaleureux.

Puis merci au personnel du pôle informatique pour l'aide ponctuelle qu'ils m'ont apportée. Enfin je remercie Martin Grana et Vladimir Daric (Biochimie Structurale) pour leur écoute et leur aide.

SOMMAIRE

REMERCIEMENTS.....	3
SOMMAIRE.....	4
INTRODUCTION.....	6
I. PROBLEMATIQUE.....	7
1. POURQUOI LES INTERACTIONS PROTÉINES – PROTÉINES ?.....	7
2. APPLICATION AU SEIN DE L'UNITÉ D'IMMUNOGÉNÉTIQUE CELLULAIRE.....	7
a. Travail de recherche au sein de l'Unité.....	7
b. De la reconnaissance de l'IL2 par son récepteur au développement d'outils informatiques....	8
3. PRÉSENTATION DU PROJET.....	9
a. Décomposition du projet initial.....	9
b. Etat de l'art.....	9
c. Un projet à intégrer.....	10
II LES INTERACTIONS PROTEINES - PROTEINES.....	11
1. TABLEAU RÉCAPITULATIF DES INTERACTIONS ENTRE PROTÉINES.....	11
2. LES INTERACTIONS IDENTIFIÉES EXPÉRIMENTALEMENT.....	11
3. LES INTERACTIONS PRÉDITES.....	13
a. Les types d'interactions structurales.....	14
b. Les interactions de type fonctionnel.....	16
III TRAVAUX REALISES.....	18
1. LA DÉMARCHÉ DE TRAVAIL.....	18
a. Les gestionnaires de cartographie existants.....	18
b. La prise en main des différentes sources de données.....	20
2. REDÉFINITION DES OBJECTIFS PRIORITAIRES.....	23
3. LE DÉVELOPPEMENT.....	23
a. La base de données sous PostgreSQL.....	24
b. Les briques de maintenance et de requêtes de la base de données sous PTOLEMY.....	32
c. Programmes sur les interactions prédites.....	33
4. RÉCAPITULATIF DES OUTILS UTILISÉS DURANT CE STAGE.....	34
IV APPLICATION.....	38
1. INTRODUCTION : L'IL2 ET SES RÉCÉPTEURS.....	38
2. IMPORTANCE MÉDICALE DE L'IL2 ET DU DÉVELOPPEMENT D'AGONISTES ET ANTAGONISTES DE L'IL2R.....	40
3. RECHERCHES EN COURS AU LABORATOIRE D'IMMUNOGÉNÉTIQUE CELLULAIRE (IGC).....	41
4. APPROCHES BIOINFORMATIQUES.....	43
a. Rechercher les séquences orthologues à chaque cytokine, chaînes de récepteur, intermédiaires de signalisation, puis les classer par organismes.....	44
b. Produire des alignements multiples afin de construire les arbres phylogénétiques et permettre la localisation des positions aux interfaces qui ont co-évolué.....	45
c. Produire les réseaux d'interactions propres à chaque cytokine et rechercher avec des méthodes prédictives d'éventuels partenaires des protéines du réseau sélectionné.....	46
d. Produire des arbres phylogénétiques au sein des systèmes cytokines-récepteurs.....	48

<i>e. Rechercher les éventuelles co-évolutions de positions dans les blocs de séquences afin de localiser les points de contacts aux interfaces.....</i>	<i>48</i>
BILAN.....	51
REFERENCES BIBLIOGRAPHIQUES.....	52
GLOSSAIRE.....	54
ANNEXES.....	57

INTRODUCTION

Au cours de ce stage pour valider ma formation de Mastère spécialisé en Bioinformatique, un projet de prédiction et représentation de liens structuraux ou fonctionnels entre des protéines m'a été confié.

Cette expérience s'est réalisée au sein de l'unité de recherche d'Immunogénétique Cellulaire à l'Institut Pasteur. Ce laboratoire est impliqué dans l'étude liée à l'infection par le virus d'immunodéficience humain (VIH). Il y a une part de recherche clinique et une part de recherche fondamentale. Pour cette dernière des approches bioinformatiques sont employées pour assister et rationaliser le choix des procédures expérimentales et aider à leur analyse et leur compréhension.

C'est ainsi que ce document renfermant les résultats de mes travaux débute par l'explication de la problématique des réseaux protéiques, pour poursuivre sur la présentation théorique des différentes interactions protéines -protéines avant de décrire toutes les activités menées durant ce stage, comme l'application à la description du système interleukine 2 – récepteur.

I. PROBLEMATIQUE

1. Pourquoi les interactions protéines – protéines ?

En dix ans les projets de séquençage des génomes ont enrichi rapidement les bases de données de centaines de milliers de nouvelles protéines. Beaucoup d'entre elles n'ont pas de fonctions connues et certaines ne présentent aucune similitude avec une quelconque protéine identifiée et annotée chez un autre organisme. Ce sont des séquences orphelines. De nombreux projets de génomique comparative et fonctionnelle ont récemment été entrepris pour annoter, organiser, structurer et connecter cette avalanche d'informations afin de les rendre utilisables et partageables.

En absence de données expérimentales précises sur la fonction, l'interaction entre deux protéines est une source riche d'informations. En effet l'immense majorité des protéines exercent leur fonction en interagissant avec d'autres protéines. En identifiant les partenaires d'interaction d'une séquence orpheline, il est possible de faire l'hypothèse de sa fonction, si celle de l'un de ces partenaires a été identifiée.

Plusieurs méthodes expérimentales permettent d'identifier et analyser les interactions intermoléculaires à haut débit. Avec ces données il est aujourd'hui possible de faire des prédictions significatives et d'étendre les réseaux construits par projection d'un organisme à l'autre ou par la reconnaissance de domaines d'interaction sur des cibles spécifiques (PRM = Peptide Recognition Module) ou encore la détection de la co-évolution des partenaires.

Les buts pour déterminer la fonction d'une protéine varient selon l'objet de la recherche. Dans l'unité à laquelle je suis rattachée nous nous intéressons à la cytokine interleukine 2 (IL2), voyons pourquoi et dans quoi elle est impliquée.

2. Application au sein de l'Unité d'Immunogénétique Cellulaire

a. Travail de recherche au sein de l'Unité

L'unité est engagée dans l'étude fondamentale de la reconnaissance de l'interleukine 2 (IL2) par son récepteur et du mécanisme de transduction induit par la fixation de la cytokine. Un projet de développement d'agoniste est en cours afin de substituer l'injection d'IL2 dans les thérapies du cancer du rein, des mélanomes et de l'infection par VIH. L'IL2 est administrée pour stimuler la prolifération des lymphocytes impliqués dans la destruction des cellules tumorales ou infectées. L'IL2 est toxique et responsable de fuites vasculaires provoquant, par exemple, de graves œdèmes pulmonaires.

Dans ce contexte, des approches bioinformatiques sont entreprises pour guider et rationaliser cette recherche fondamentale. Cette recherche se découpe en trois niveaux :

D'abord sont recherchés les points communs entre les séquences et les structures des récepteurs membranaires qui sont les cibles des drogues.

Puis sont construits des modèles moléculaires des complexes possibles formés par des récepteurs avec leurs drogues.

Enfin sont conçues de nouvelles drogues soit par filtrage de bibliothèques de molécules, soit par construction itérative des molécules à partir de fragments. Ceci permet alors de produire des hypothèses à vérifier expérimentalement si nécessaire. L'objectif est d'interpréter le mécanisme d'activation du récepteur de l'interleukine 2 et d'identifier les activateurs synthétiques ou recombinants de ce récepteur qui puissent être de bons candidats thérapeutiques.

b. De la reconnaissance de l'IL2 par son récepteur au développement d'outils informatiques

L'analyse comparative des séquences orthologues permet dans de nombreux travaux d'extraire des motifs conservés qui peuvent être corrélés à des propriétés structurales ou fonctionnelles. Les mêmes analyses réalisées simultanément sur des protéines qui interagissent permettent de reconnaître des domaines canoniques et leurs cibles fonctionnelles. De telles analyses permettent également de localiser certains points de contact par la recherche de motifs complémentaires et des mutations compensatoires d'un groupe de protéines à l'autre le long de leur interface. Aucun outil informatique ne permet actuellement de rechercher puis d'analyser simplement et efficacement ces corrélations en absence de données structurales sur le complexe.

L'objet du stage consiste à fournir un tel outil. Cela se traduit par la conception d'un gestionnaire graphique de processus bioinformatiques (ou pipeline) d'exploration, d'analyse et de prédiction d'interactions entre protéines.

Nous souhaitons ensuite, avec des opérateurs logiques et statistiques, pouvoir analyser des données de séquences, de structures et d'arbres phylogénétiques. De telles analyses peuvent aider à identifier les protéines susceptibles d'interagir et de localiser sur les séquences, puis sur les structures si elles sont disponibles, les résidus formant leurs interfaces.

Les outils développés seront utilisés pour identifier les résidus impliqués dans les contacts entre l'interleukine 2 (IL2) et ses récepteurs, et mettre en évidence ceux qui sont responsables de la spécificité de l'interaction par rapport aux autres cytokines de la même famille des hématopoïétines. Il semble très probable que le mécanisme de transduction du signal par les récepteurs de cette famille serait identique (Rose et al. J.Biol.Chem. 2003). Ils seraient dirigés par les domaines externes qui s'associent spontanément en absence de cytokine puis réarrangeraient leur interface lors de la fixation de la cytokine spécifique, provoquant le rapprochement des domaines transmembranaires et cytoplasmiques d'une cinquantaine d'angströms (ceci est développé plus en détail dans la partie IV 1). Des résidus conservés à la surface des chaînes du récepteur semblent critiques dans le basculement puis le maintien d'un état éteint ou allumé du récepteur ; ils encodent le mécanisme. Ils devraient pouvoir être identifiés avec les outils développés.

La désignation des résidus importants dans l'affinité, la spécificité et le mécanisme de reconnaissance de l'IL2 par ses récepteurs permettent de rationaliser les approches par mutagenèse dirigée et marquage chimique, qui sont menées au laboratoire. Ceci doit nous aider à comprendre les bases moléculaires du phénomène et d'en utiliser les connaissances dans le développement d'agonistes du récepteur, associés à des index thérapeutiques supérieurs à celui de l'IL2 ou des molécules mimétiques qui ont été développées et étudiées jusqu'à présent au laboratoire d'Immunogénétique Cellulaire.

3. Présentation du projet

Voyons ici les grands axes du sujet de stage, puis ce qui était déjà développé dans le domaine et enfin où s'intégrera ce projet.

a. Décomposition du projet initial

Deux aspects composent ce projet de bioinformatique d'analyse de réseaux protéiques :

- le problème de visualisation graphique des réseaux protéiques : Quelles solutions déjà existantes allons-nous retenir ou allons-nous redévelopper un outil ?
- le problème de la source de données pour la conception de ces réseaux : il faut mettre à disposition de l'utilisateur une source de données homogènes, sur laquelle il sera facile d'effectuer tous types de requêtes (par exemple selon un organisme pour une protéine requête ou, selon une méthode d'interactions pour trouver ses partenaires spécifiques)

b. Etat de l'art

Chacun des points soulevés ici est développé dans les deux parties suivantes (II et III) de ce mémoire.

En ce qui concerne les séquences protéiques, nous avons importé nos données à partir :

- du NCBI (National Center for Biotechnology Information : <http://www.ncbi.nlm.nih.gov/>) qui fournit la banque non redondante de séquences, la NRprot (communément appelée la nr)
- du SIB (Swiss Institute of Bioinformatics : <http://www.isb-sib.ch/>) et dont le serveur d'accès aux données des séquences est ExPASy Proteomics Server (**Expert Protein Analysis System** sur www.expasy.org). Le SIB fournit trois catégories de banques :
 - la Swissprot banque de séquences de protéines
 - la Trembl banque de séquences ORF (Open Reading Frame) (ARN messenger), ce ne sont pas des protéines identifiées.
 - la New_Trembl banque de séquences en attente de rentrer dans la Trembl

Les interactions entre protéines qui peuvent être établies grâce à diverses méthodes sont exposées dans la seconde partie de ce rapport. Mais retenons dès à présent que celles dites expérimentales sont présentes dans de nombreuses banques de données. Et ces banques sont hétérogènes entre elles. Le lien <http://www.hgmp.mrc.ac.uk/GenomeWeb/prot-interaction.html> donne une liste des sites de ces différentes banques.

Les interactions prédites par des méthodes informatiques, lorsqu'elles sont implémentées, sont souvent commerciales, le lien <http://www.proteinpathways.com/techProNexus.html> en est un exemple.

En parcourant ces différents sites de banques j'ai observé de nombreux problèmes d'identificateurs et de déficiences de liens entre ces sources variées.

Les gestionnaires graphiques de cartographie de réseaux protéiques sont nombreux, ils sont parfois spécifiques à une seule banque de données expérimentales. Citons par exemple **Cytoscape**

(<http://www.cytoscape.org/>). Les références de ces gestionnaires sont données en troisième partie de ce document paragraphe 1 a (III 1 a).

c. Un projet à intégrer

Ce projet, relatif à l'analyse de réseaux d'interactions entre protéines, s'insère dans la continuité de deux autres projets de bioinformatique. Ensemble, les trois projets ont pour objectif commun de fournir à l'utilisateur une plate-forme de conception, d'exécution et de contrôle de processus bioinformatiques permettant de manipuler des séquences, des structures, des arbres phylogénétiques et des réseaux d'interactions de protéines.

- Franck Valentin (Pôle Informatique) était en charge du reconditionnement et de l'adaptation de la plate-forme Ptolemy et de son environnement graphique Vergil. Il s'est aussi occupé de son interfaçage avec PISE (<http://www.pasteur.fr/recherche/unites/sis/Pise/>) afin d'intégrer un parc d'une centaine de logiciels d'analyse de séquences.
- Vladimir Daric (Biochimie Structurale) avait pour mission de fournir pour cette plate-forme graphique une interface avec des packs de logiciels d'analyse de structures (CCP4), de prédiction de structures (Modeller, Robetta) et de criblage virtuel (Gold, Autodock, Dock, Flexx). Sa tâche consistait aussi en un développement d'outils de cartographies structurales (conservation, contacts, empreintes fonctionnelles).
- Mon projet consistait en la création d'une base de données de séquences, d'interactions moléculaires, et d'architecture de réseaux d'interactions moléculaires. Je devais aussi me soucier du développement ou du choix (par veille technologique) d'outils de visualisation, d'analyse et de prédiction de réseaux d'interactions.

Ces trois projets sont appliqués à l'analyse du mécanisme de reconnaissance de l'IL2 par son récepteur. Ainsi, le concept des outils développés, de leur intégration, de leur articulation est pensé en terme d'utilisation finale. La généralisation ne sera entreprise qu'a posteriori.

II LES INTERACTIONS PROTEINES - PROTEINES

Nous présentons dans cette partie la théorie sur les interactions qu'il peut exister entre deux protéines. Lorsqu'il y a une interaction entre deux protéines, nous disons qu'elles sont partenaires. Elles se divisent en deux grands groupes :

- celles dites de type structural (qui a un rapport avec la structure de la protéine) et,
- celles de type fonctionnel (quand deux protéines ont besoin de la présence de l'une et l'autre pour exercer une fonction).

Et toutes ces interactions sont déterminées :

- soit expérimentalement
- soit par prédiction en implémentant des algorithmes

1. Tableau récapitulatif des interactions entre protéines

Liens Expérimentaux	Les Databases Expérimentales : DIP MINT IntAct HPRD BIND, etc. une liste de ces banques est trouvée à l'adresse : www.hgmp.mrc.ac.uk/GenomeWeb/prot-interaction.html
Liens Prédicatifs	<i>Listes de liens STRUCTURAUX</i> Projections de paires par BLAST Fusion de gènes (Rosetta) Compensation évolutive d'interface Domaine Pairs/ Shuffling <i>Listes de liens FONCTIONNELS</i> Profils Phylogénétiques Gene Clustering Voies Métaboliques Les liens bibliographiques

2. Les Interactions identifiées expérimentalement

Les interactions structurales

Il est possible d'identifier des associations physiques entre protéines ou interactions moléculaires structurales par plusieurs méthodes expérimentales, dont voici une liste non détaillée et non exhaustive :

- Co-purification
- Compétition de fixation avec des molécules marquées
- Co-immunoprécipitation

- Expression sous forme de double hybride
- Atténuation ou transfert de fluorescence entre chaque porteur de sonde
- Co-cristallisation
- Retard en chromatographie
- Dialyse à l'équilibre
- Centrifugation analytique

Les interactions fonctionnelles

De même nous pouvons identifier les liens fonctionnels entre deux molécules lorsqu'elles sont impliquées dans des fonctions communes, ou présentent un ligand commun, sans qu'il y ait systématiquement association physique :

- par co-expression des ARN messenger
- ou par suppression ou extinction de l'expression des gènes.

Le nombre des interactions observées croît exponentiellement. En effet la méthodologie permettant la mise en évidence d'interactions protéiques est désormais suffisamment fiable pour être applicable à l'échelle génomique. Les données sont alors produites en très grand nombre. Ainsi de quelques milliers en 2000, à 15 000 en 2002, 45 000 en 2003, ce sont plus de 100 000 paires qui sont connues en juin 2004. Elles s'accroissent à raison de 15 interactions structurales et une centaine d'interactions fonctionnelles par heure.

Ces données sont organisées dans des banques de données accessibles par le web. Chacune de ces banques a une spécificité particulière, comme celle de s'intéresser uniquement à certains organismes (exemple de l'humain traité par HPID = **H**uman **P**rotein **I**nteraction **D**atabase, <http://wilab.inha.ac.kr/hpid/>), ou encore à un seul type de méthode. L'organisation de ces banques n'est pas standardisée. Les données sont fréquemment redondantes, les maintenances peuvent être automatisées à partir de la littérature ou implémentées manuellement et modérées par des experts.

Le lien suivant fournit une liste des banques de données d'interactions entre deux protéines expérimentalement identifiées :

www.hgmp.mrc.ac.uk/GenomeWeb/prot-interaction.html

Le tableau, qui suit, récapitule le nombre des interactions structurales documentées (septembre 2004) pour certaines banques (les plus grosses en volume de données).

<u>Nom de la banque</u>	<u>url de la banque</u>	<u>Nombre d'Interactions</u>
BIND Biomolecular Interaction Network Database	www.blueprint.org/bind/bind.php	104 430
DIP Database of Interacting Protein	http://dip.doe-mbi.ucla.edu/	44 444
MINT Molecular INTeraction database	http://mint.bio.uniroma2.it/mint/	42 662
GRID General Repository for Interaction Datasets	http://biodata.mshri.on.ca/grid/servlet/Index	Yeast (levure) 25 915 Fly (Mouche) 26 596 Ver (Worm) 4 453
IntAct La banque de l'EBI	http://www.ebi.ac.uk/intact/index.html	36 865
InterDom Database of Interacting Domains	http://interdom.lit.org.sg/	30 037

HPID Human Protein Interaction Database	http://www.hpid.org	18 161
HPRD Human Protein Reference Database	http://www.hprd.org/	

Chacune de ces banques de données expérimentales demande une étude particulière pour comprendre la manière dont sont stockées les informations qu'elle renferme.

Pour pallier ce manque de standardisation des processus de normalisation sont en cours de développement notamment avec le langage PSI-MI. Un exemple de ce format est donné en *annexe 1*, il est tiré de la banque de données HPRD (<http://www.hprd.org/>). Le but de PSI-MI est de proposer un modèle standard commun pour la représentation, la documentation et l'échange des données concernant les interactions protéines-protéines (Henning Hermjakob et al. NATURE BIOTECHNOLOGY vol.22 n°2 (2004)). Plus d'informations sur ce modèle de données sont trouvées sur le site <http://psidev.sourceforge.net/>. Ce standard est développé par les membres de PSI (Proteomics Standards Initiative), un groupe de travail de HUPO (Human Proteome Organization).

Mais ces standards ne sont pas encore fiables et appliqués identiquement. Compte-tenu de ce nombre croissant de données hétérogènes chaque jour, il semble intéressant de pouvoir les stocker localement de manière homogène, et de pouvoir les actualiser régulièrement grâce à des opérations de maintenance simples. Nous verrons par la suite (dans la description des travaux réalisés) que l'idée de construire une réelle base de données, regroupant toutes ces informations produites, apparaît comme un besoin.

NB : Il faut donc aller sur chacun des sites ci-dessus et trouver les liens correspondant aux pages de téléchargement des données. Ces dernières seront alors sous la forme de fichiers plats ou PSI-MI ou encore XML. Les divisions de chacun de ces fichiers sont arbitraires et fonction de la banque dont ils proviennent.

Certaines incohérences sont mises continuellement en relief lors de l'étude de ces fichiers. Par exemple il est dit sur le site de MINT (cf. tableau ci-dessus) que le nombre d'interactions présentes dans la banque MINT, est de plus de 40000. Et le fichier plat, une fois téléchargé, n'en contient qu'environ 18000. Ainsi l'annonce précède souvent de plusieurs mois la mise à disposition réelle des listes d'interactions.

3. Les Interactions Prédites

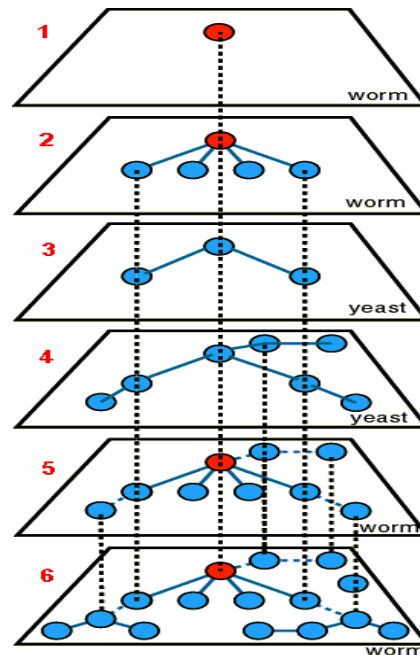
Les interactions dites prédictives sont celles trouvées au moyen d'implémentations d'algorithmes. Elles sont donc purement hypothétiques mais permettent d'apporter d'autres sources d'informations aux chercheurs, en absence de données expérimentales. Sont retrouvées parmi ces interactions prédites :

- celles de type structural
- celles de type fonctionnel.

a. Les types d'interactions structurales

- **Les interactions par extrapolation d'un organisme à l'autre** s'obtiennent en faisant des projections de paires par BLAST (**B**asic **L**ocal **A**lignment **S**earch **T**ool). Ces projections permettent d'identifier les paires homologues (c'est-à-dire provenant d'un ancêtre commun).

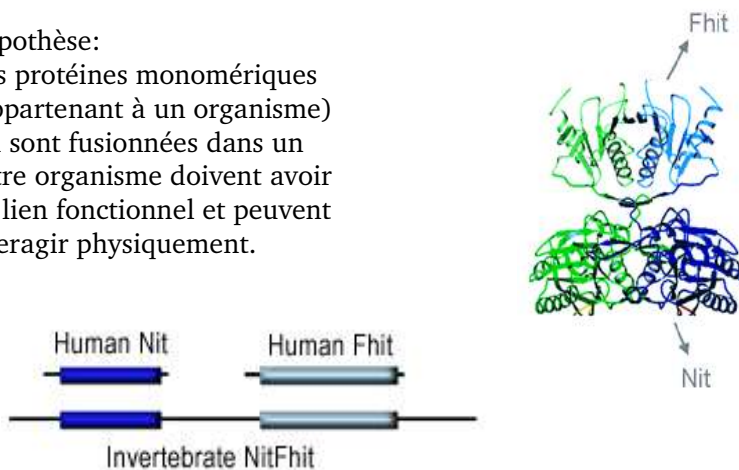
Depuis une protéine *Query* (celle repérée par un rond rouge sur le niveau 1), tous les partenaires dans le même organisme (ici le ver, worm) de la *query* sont sélectionnés (niveau 2 sur la fig.). Puis on projette les séquences de ses partenaires dans un autre organisme (fig.: niveau 3 dans l'organisme de la levure). Cette projection est opérée avec BLAST (sur la fig.: les pointillés noirs). Puis nous recherchons les interactions dans lesquelles ces protéines (de l'organisme de la levure ici) sont impliquées (fig.: niveau 4). Et enfin on projette à nouveau alors ces nouvelles protéines dans l'organisme de la *query* (ici le ver) pour compléter le réseau d'interactions. **Les interactions dessinées au niveau 6 par des pointillés bleus sont alors celles prédites, en fin d'itérations par la méthode de projection par BLAST.**



- **L'algorithme de ROSETTA Stone** permet d'identifier les domaines d'une protéine, fusionnés ou séparés durant l'évolution. Grâce à l'implémentation d'un tel algorithme, il est possible d'obtenir des listes de paires de protéines liées par la structure dans un organisme donné. (David Eisenberg, et al. NATURE Vol 405 15 JUNE 2000)

Hypothèse:

Des protéines monomériques (appartenant à un organisme) qui sont fusionnées dans un autre organisme doivent avoir un lien fonctionnel et peuvent interagir physiquement.



Dans l' exemple ci-dessus:

la méthode de Rosetta Stone a prédit que les protéines humaines Nit et Fhit interagissent car elles ont été fusionnées chez les invertébrés, et elles forment un hétérocomplexe chez les mammifères.

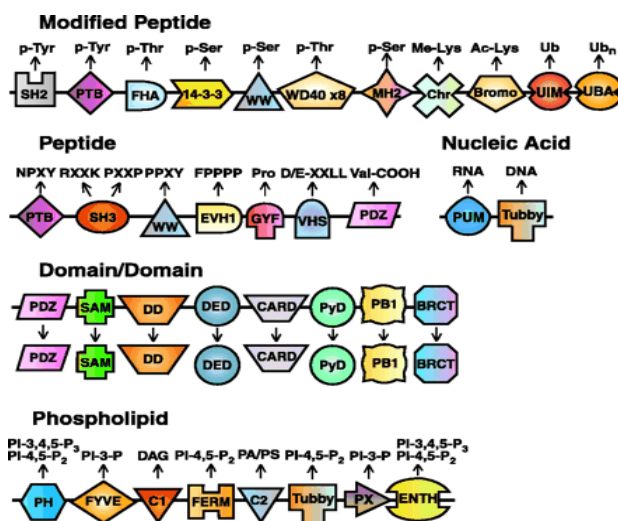
- **La compensation évolutive d'interfaces**

Il est fréquent d'observer à l'interface de deux protéines, au sein d'un complexe, le maintien d'un pont salin acide-base au cours de l'évolution. Certaines de ces paires d'acides aminés ont été inversées chez quelques organismes. La persistance du pont salin après cette double mutation révèle la contrainte fonctionnelle du système pour "assurer" l'association structurale des deux molécules. A partir des alignements multiples pour les deux protéines, nous recherchons les positions auxquelles surviennent des mutations compensatoires. La présence de ces doubles mutations nous permet de suggérer l'interaction des deux partenaires et de localiser sur leurs séquences quelques points de contacts.

Le programme encodant cette méthode a été développé au cours de ce projet et est détaillé dans le chapitre IV Application.

- **La reconnaissance de domaines protéiques** (PRM = Peptide Recognition Module) par des méthodes de filtrage (programme HMMER) et distribués sous forme de profils dans les bases Pfam, ProDom, SMART permet de les associer à leur **cible, ou motif peptidique**, détecté par les programmes Prosite, PRINTS .

Reiss et Schwikowski ont proposé récemment une méthode, élégante et efficace, de reconnaissance des PRM et de leur cible : le programme netmosa <http://sf.net/projects/netmotsa> (Predicting protein peptide interactions via a network-based motif sampler David J.Reiss and Benno Schwikowski Bioinformatics 20(Suppl. 1) Août 2004)

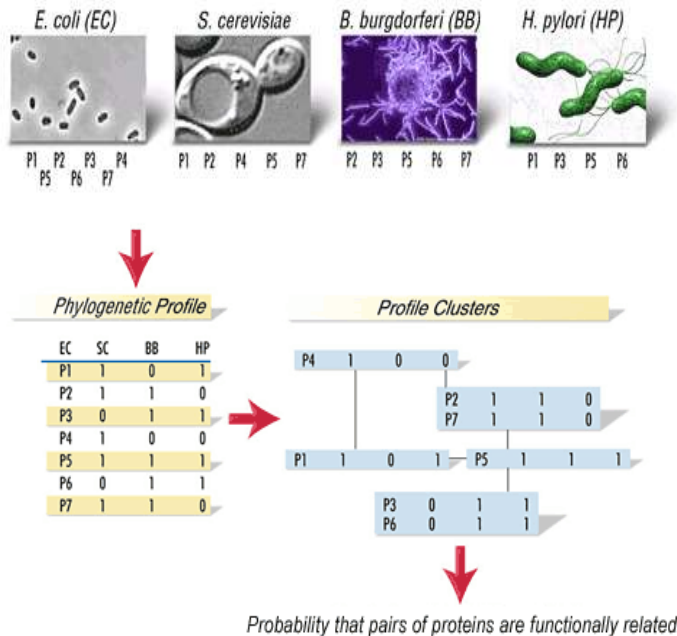


sur la figure ci-dessus : les PRM sont les objets de couleurs et les cibles sont les lettres (suite d' acides aminés) indiquées au-dessus.

Suite à cette présentation des interactions de type structural, voyons à présent les interactions prédictives de type fonctionnel. Grâce aux différentes hypothèses présentées ci-dessous, il est possible de produire des listes de paires de protéines qui sont liées par la fonction qu'elles ont au sein d'une cellule.

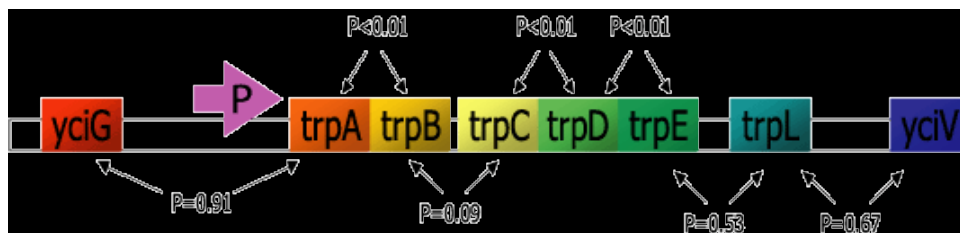
b. Les interactions de type fonctionnel

- Au moyen de **profils phylogénétiques** établis pour toutes les protéines d'un organisme donné, nous pouvons identifier les groupes fonctionnels de protéines encodées pour des fonctions physiologiques spécifiques. (David Eisenberg, et al. NATURE Vol 405 15 JUNE 2000)



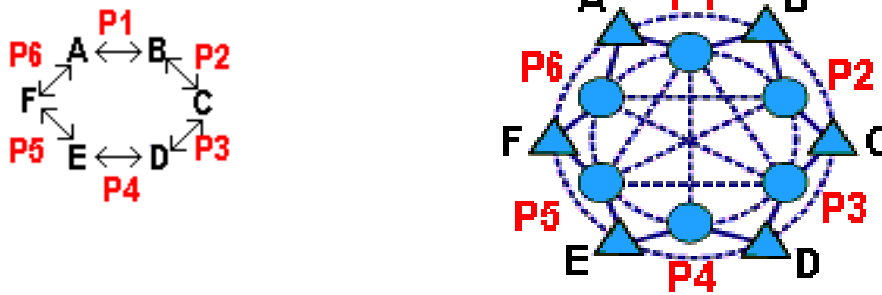
Dans l'exemple ci-dessus, les protéines P2 et P7 ont le même profil phylogénétique : elles sont toutes les deux présentes dans les mêmes organismes (SC et BB) et absentes dans le même organisme (HP). La paire P2-P7 sera alors stockée comme une interaction. De même pour P3 et P6.

- Par le **Gene Clustering**, nous pouvons identifier les groupes de gènes corrégulés. (David Eisenberg, et al. NATURE Vol 405 15 JUNE 2000)



Dans la figure ci-dessus la méthode du Gene Clustering est appliquée à l'opéron Tryptophan. La valeur seuil de P-value < 0.1 a été utilisée pour identifier des paires de gènes qui vont probablement avoir les mêmes fonctions cellulaires et cela sans tenir compte du fait ou ils se trouvent les uns par rapport aux autres sur la séquence de l'opéron.

- Identifier les **acteurs des voies métaboliques** permet également de produire des listes de liens fonctionnels entre des protéines.



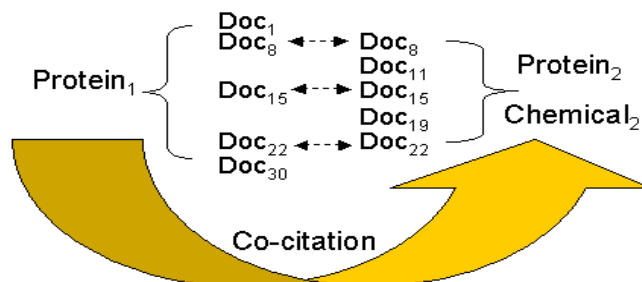
Listes de liens structuraux :

{P1,A},{P1,B},{P2,B},{P2,C},
 {P3,C},{P3,D},{P4,D},{P4,E},
 {P5,E},{P5,F}, {P6,F},{P6,A}

Listes de liens fonctionnels:{P1,P2,P3,P4,P5,P6,A,B,C,D,E,F}

(Réf.:<http://www.genome.jp/kegg/regulation.html>)

- Enfin une dernière méthode prédictive est d'établir des liens bibliographiques entre deux partenaires:



Dans la figure ci-dessus la **Protein1** est citée dans les documents 8, 15 et 22, tout comme l'est la **Protein2**. On suppose alors que les deux protéines sont des partenaires fonctionnels.

Pubgene est un bon exemple. C'est un outil qui permet de faire ce genre de prédictions au moyen de liens bibliographiques : <http://www.pubgene.org/>

Ayant présenté toutes ces méthodes permettant de rechercher deux partenaires, je vais dans la partie qui suit m'attarder sur l'objet de ma tâche durant ces six mois dans l'Unité d'Immunogénétique Cellulaire.

III TRAVAUX REALISES

Rappelons ici l'objectif final du développement informatique: fournir à l'utilisateur final (un chercheur) un outil dans lequel il puisse entrer une protéine requête. A l'exécution de cette requête l'outil doit être capable de lui fournir une liste de partenaires de cette protéine. Cette liste doit pouvoir être visualisée dans un gestionnaire graphique et fournir ainsi à l'utilisateur une cartographie de son réseau protéique.

Il est intéressant de pouvoir imposer des filtres lors de la requête par exemple sur la méthode avec laquelle nous voulons trouver les partenaires de notre protéine requête (toutes ces méthodes sont décrites dans le II de ce document).

Enfin en ce qui concerne la compensation évolutive d'interface (cf. II-2-a), il serait utile de permettre à l'utilisateur une représentation 3D où nous pourrions superposer des couches de réseaux protéiques. Chaque couche représenterait un réseau d'interactions pour un organisme.

Dans cette partie je vais expliquer quelle a été ma démarche de travail au cours de ce stage. Je poursuis ensuite par le compte rendu du développement que j'ai réalisé avant de faire un récapitulatif des outils que j'ai utilisés.

1. La Démarche de travail

L'approche du sujet des interactions protéines protéines et d'un développement d'outil à cet effet s'est découpée en trois grands points :

1. la compréhension du sujet, son intérêt et quel but il y a derrière pour la recherche avec la lecture d'articles (toutes les références bibliographiques sont répertoriées dans la partie bibliographie de ce compte-rendu).
2. L'exploration des différents logiciels ou applications informatiques permettant la visualisation de réseaux protéiques.
3. La prise en main des différentes sources de données avec les banques de séquences et les banques d'interactions expérimentales

Je ne détaille pas davantage ici le premier point ci-dessus, les informations à ce sujet sont trouvées dans les parties précédentes.

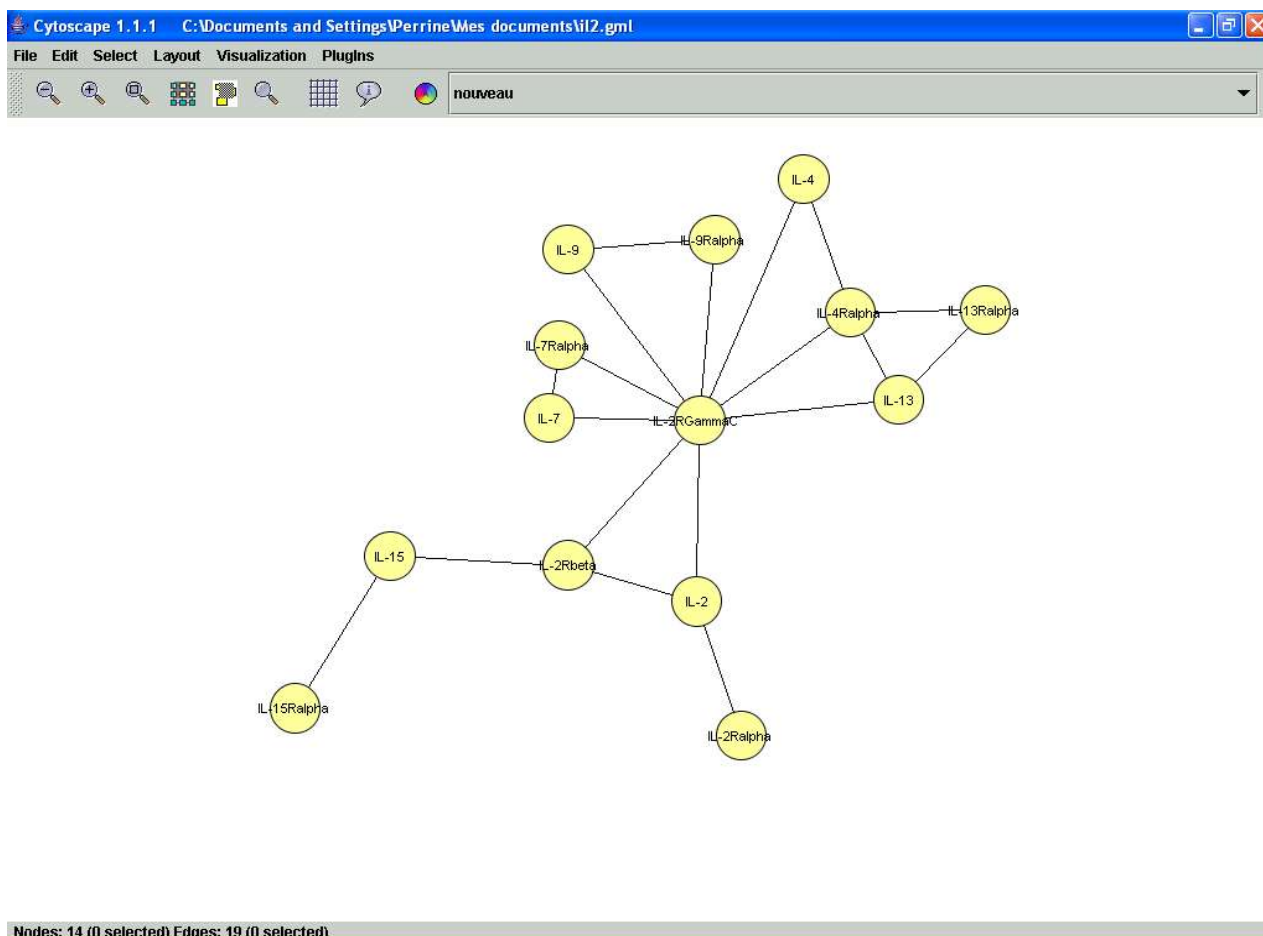
Voyons plutôt quels sont les gestionnaires graphiques que j'ai regardés et lequel j'ai retenu.

a. Les gestionnaires de cartographie existants

Je me suis intéressée à quelques gestionnaires graphiques afin d'explorer les différentes fonctionnalités qu'ils proposent.

J'ai commencé par Cytoscape dont le site web est www.cytoscape.org Cytoscape est un logiciel open source (libre accès) qui permet de visualiser des réseaux d'interactions moléculaires et d'intégrer ces interactions avec des profils d'expressions géniques et d'autres données. L'un des promoteurs de ce programme a pris la direction d'un groupe à Pasteur, ce qui laisse présager des interactions étroites avec le groupe IGC (ImmunoGénétique Cellulaire).

L'outil Cytoscape propose des fonctionnalités intéressantes tant au niveau graphique qu'au niveau des données biologiques qui peuvent être prises en compte.
La capture d'écran ci-dessous montre l'aspect du logiciel:



Nous pouvons observer ci-dessus un réseau d'interactions mettant en jeu différentes cytokines IL2, IL4, IL7, IL9, IL13, IL15, IL21 et leurs différents récepteurs.

Pour stocker un tel type de graphe, sous cytoscape il existe deux types de formats: le .sif (format simple qui prend en compte seulement deux partenaires) et le .gml (format qui permet d'associer à chaque nœud des coordonnées graphiques et d'autres contraintes).

J'ai testé l'outil Osprey, spécifique pour les banques d'interactions expérimentales GRID <http://biodata.mshri.on.ca:80/osprey/servlet/Index>

Osprey est une plate-forme logicielle permettant la visualisation de réseaux complexes d'interactions limitées à celles annotées et mises à jour par GRID. De plus l'emploi du logiciel ne peut pas se faire si les utilisateurs se connectent par un proxy, ce qui est le cas à Pasteur.

J'ai aussi exploré Proviz et l'outil Tulip sur lequel il repose pour faire de la 3D. Mais ce sont des outils portés uniquement sous linux. Etant donné que notre plate-forme de gestion de programmes de bioinformatique doit pouvoir tourner sous n'importe quel système d'exploitation, Proviz ne peut pas être retenu.

<http://www.ebi.ac.uk/intact/doc/html/proviz/proviz-manual/proviz-manual.html>

J'ai survolé quelques autres gestionnaires. Pour la plupart, les fonctionnalités proposées sont souvent très limitées, en voici une liste non exhaustive:

Dynamic signaling Maps	http://www.hippron.com/hippron/index.html
Interviewer	http://interviewer.inha.ac.kr/
PathwayAssist	http://www.ariadnegenomics.com/products/pathway.html
PathDB	http://www.ncgr.org/pathdb/
PIMRider	http://pim.hybrigenics.com/pimriderext/common/

Nous décidons alors de retenir Cytoscape dont le développement apporte continuellement des fonctionnalités intéressantes nouvelles. De plus ils pensent à la 3D.

En ce qui concerne ce problème de 3D, après l'exploration du logiciel PyMol, qui est un système de modélisation moléculaire (<http://pymol.sourceforge.net/> développé par V.DeLano) nous nous apercevons qu'il est possible de transformer un fichier GML en PDB. Le GML correspond au format de fichier d'un graphe 2D que fournit Cytoscape et le PDB est le format de fichier correspondant à une molécule en 3D.

Nous prenons donc le format GML sur lequel nous appliquons un script pour le transformer en format PDB. Ce fichier au format PDB est alors à ouvrir sous PyMol. Et là, nous sommes capables, avec cet outil, de faire des superpositions de couches qui ouvrent des perspectives de visualisation et de manipulation pour la compensation évolutive d'interface, c'est alors un outil intéressant.

Faire des cartes de réseaux protéiques suppose interroger des sources de données qui renferment des informations sur les protéines et leurs séquences et des informations sur les liens existants entre deux partenaires. J'explore alors ces différentes données et regarde quelle est l'architecture de leur stockage.

b. La prise en main des différentes sources de données

Les deux grands types de sources de données pour notre sujet ici sont les banques de séquences et les banques des interactions expérimentales. J'essaie alors de comprendre comment elles marchent, de quelle manière sont stockées les données qu'elles renferment et par quel moyen je peux les récupérer.

– les banques de séquences protéiques

Deux grands organismes stockent et distribuent des séquences : l'organisme américain NCBI (National Center of Biotechnology Information) et le SIB (Swiss Institute of Bioinformatics).

- Le NCBI distribue une banque de données non redondante, la nr

Il est possible de télécharger toute cette banque sous forme d'un fichier FASTA à partir de <ftp://ftp.ncbi.nlm.nih.gov/blast/db/FASTA/> (ou il faut prendre le fichier nr.gz). Un exemple d'une fiche FASTA est donné en *annexe 2*.

L'identificateur officiel d'une séquence stockée dans la NR est le **gi**.

Au 16 Septembre 2004 cette banque contient environ 2 000 000 de séquences normalement bien différenciées. En effet comme son nom l'indique elle est non redondante c'est-à-dire que deux séquences exactement identiques sont fusionnées en une seule et unique entrée dans la nr. Comme il est indiqué dans le "*ReadMe*" de cette banque, pour pouvoir fusionner en une seule entrée, deux séquences doivent avoir exactement la même longueur et chaque résidu (ou acide aminé) doit à chaque position être le même. Et techniquement les lignes d'identification FASTA (c'est-à-dire la ligne à chevron) de chacune de ces deux séquences identiques sont regroupées derrière une seule et même ligne à chevron dans la nr. Les identifications différentes sont alors séparées entre elles par le caractère `control-A`. Dans l'exemple qui suit les deux entrées `gi | 1469284` et `gi | 1477453` sont une seule et même séquence, elles ne correspondent alors qu'à une seule et même entrée dans la nr:

```
>gi|3023276|sp|Q57293|AFUC_ACTPL   Ferric transport ATP-binding protein afuC
^Agi|1469284|gb|AAB05030.1|   afuC gene product ^Agi|1477453|gb|AAB17216.1|
afuC [Actinobacillus pleuropneumoniae]
MNNDFLVLKNIKTSFGKATVIDNLDLVIKRGTMVTLTLPSPGCGKTTVLRLVAGLENPTSGQIFIDGEDVT
KSSIQNRDICIIVFQSYALFPHMSIGDNVGYGLRMQGVSNNEERKQRVKEALELVDLAGFADRFVDQISGGQ
QQRVALARALVLKPKVLILDEPLSNLDANLRRSMREKIRELQQLGITSLYVTHDQTEAFVAVSDEVIVMN
KGTIMQKARQKIFIYDRILYSLRNFMGESTICDGNLNLQGTVSI GDYRFPPLHNAADF SVADGACLVGVRPE
AIRLTATGETSQRCQIKSAVYMGNHWEIVANWNGKDVLIANPDPQFDPDATKAFIHFTEQGIFLLNKE
```

Cette banque pose quelques problèmes quant à l'exploitation immédiate de ses données:

- Si les séquences peuvent être fusionnées comme expliqué ci-dessus alors pour une seule entrée protéique je peux avoir de un à plusieurs **gi**. L'identification d'une séquence dans la nr n'est donc pas unique.
- Puisque cette fusion est permise, les séquences identiques provenant d'organismes différents consisteront en une seule et même entrée. Or dans tout notre projet l'organisme est un élément bien caractéristique de notre séquence étudiée.
- Enfin le caractère non-redondant de la nr n'est pas fiable à 100%, il arrive que deux séquences exactement identiques puissent être stockées en tant que deux entrées, il y a néanmoins des déchets.

• Les banques provenant du SIB

Sur le serveur ExPASy su SIB nous pouvons faire des requêtes sur les banques de données Swissprot Trembl et NewTrembl.

La Swissprot est une banque de protéines clairement identifiées. Elle renferme actuellement un peu plus de 150 000 séquences.

La Trembl est une banque de séquences d'acides aminés, résultats de la traduction d'ORF (Open Reading Frame) (mRNA). Ce sont des séquences de résidus auxquelles ne sont pas associées une protéine connue.

La NewTrembl contient les nouvelles séquences en attente d'entrer dans la Trembl.

A elles trois, ces banques fournissent environ 1 500 000 de séquences différentes, repérées par un identificateur spécifique, le **sp**. Les données de ces différentes banques sont téléchargeables à partir de l'adresse <http://www.expasy.org/sprot/download.html> ou bien:

sur ftp://ftp.expasy.org/databases/swiss-prot/release_compressed pour la Swissprot

et sur ftp://ftp.expasy.org/databases/trembl/release_compressed pour toutes les données de Trembl.

Par ces serveurs nous pouvons alors récupérer toutes les informations de ces banques de données. Elles sont sous la forme de fichiers plats dont un exemple est donné en *annexe 3* pour le cas de l'IL2.

N.B.: notons que sur ces serveurs les fichiers spécifiés **updates** concernent des entrées déjà existantes dans la Swissprot et modifient quelques données sur ces séquences. Ces changements ne sont pas clairement indiqués. Nous nous n'y attardons pas.

• Autres sources de données possibles pour les séquences protéiques

Signalons aussi que les chercheurs peuvent utiliser leurs propres sources de données dans lesquelles aucun **sp** ni **gi** ne peut être trouvé. Si par exemple ces chercheurs peuvent nous fournir des informations intéressantes sur les matrices de profils phylogénétiques (cf. II.3.b) par exemple, elles ne seront pas exploitables directement. Il faut trouver au préalable à quelles entrées de la **NR** ou de la **Swissprot** les séquences concernées correspondent.

Conclusion sur les banques de séquences:

- Ces différentes banques de données présentent donc parfois au sein d'elles même des incohérences qu'il serait alors intéressant de pouvoir corriger.
- De plus les données venant du SIB ne font jamais référence au **gi** (identificateur nr). Celles du NCBI font référence au **sp** (identificateur Swissprot) mais dans un fichier FASTA difficilement exploitable.
- Enfin les identificateurs de ces banques ne sont pas uniques, par exemple un même **gi** peut être associé à deux séquences différentes et une séquence peut être référencée par plusieurs **gi**.

=> **Il n'existe donc pas d'identificateur fiable et unique auquel nous pouvons associé une seule et unique séquence provenant d'un seul et même organisme.**

La solution, qui commence à apparaître ici à la découverte de ces données, serait le montage d'une base de données en local qui permettrait d'homogénéiser le stockage de toutes ses séquences en créant un identificateur local unique `idprotein` auquel correspondrait une seule et unique séquence d'un seul et unique organisme.

– les banques de données expérimentales

Les banques de données expérimentales ont été présentées dans le II-2 de ce document. Il en existe donc de multiples. Et chacune d'entre elles stocke ses données d'une façon qui lui est propre. Chaque banque permet à l'utilisateur de télécharger ses informations en local. Mais leur déchiffrement est parfois compliqué, les parseurs de tels fichiers n'en sont que plus difficiles à écrire. Il y a un manque de normalisation, c'est pourquoi des efforts sont menés par HUPO (Human Protein Organisation) afin de définir un standard pour ces banques de données grâce au format de fichier PSI-MI. (Henning Hermjakob et al. NATURE BIOTECHNOLOGY VOL.22 N°22 February 2004). C'est un format de fichier qui ressemble à du xml avec des spécificités biologiques concernant les interactions en biologie (cf. *annexe 1* en exemple).

Actuellement ce format de fichier est quasiment imparsable. Il est même parfois préférable de télécharger un fichier texte plat fourni par la banque expérimentale que de récupérer les fichiers "xml".

Par ailleurs ces banques font référence à des séquences protéiques (celles impliquées dans les paires stockées) au moyen d'identificateurs. Suivant la banque les identificateurs sont soient des **gi** ou bien des **sp** ou même des nouveaux identificateurs propres à la banque en question.

Conclusion sur les banques d'interactions:

Nous nous apercevons alors que les banques d'interactions posent aussi des problèmes d'hétérogénéité. Il serait plus pratique de pouvoir stocker toutes ces interactions dans un seul et même endroit avec un identificateur unique repérant chaque interaction. Ceci renforce l'idée qu'une base de données construite en local peut pallier ce genre d'obstacle.

Toutes ces banques, aussi bien de séquences que d'interactions représentent une mine d'informations. Mais cette mine est difficilement exploitable dans sa globalité. Or nous voulons permettre la construction de nombreux réseaux protéiques avec le maximum de données. En conséquence il serait intéressant d'avoir une base de données homogènes renfermant toutes les séquences et les interactions aussi bien expérimentales que prédites.

Dans cette base chaque séquence serait unique associée à un seul organisme et repérée par un seul et unique identificateur, qui serait alors repris dans les interactions dans lesquelles la séquence est impliquée.

Cette source de données serait construite suivant des normes et en respectant des règles pour la globalité des données, il serait finalement facile de faire tout type de requête dessus.

2. Redéfinition des objectifs prioritaires

Pour la gestion graphique nous retenons alors le logiciel CYTOSCAPE qui fournit une représentation 2D d'un réseau de partenaires protéiques. Et pour le besoin de représentation tridimensionnelle nous nous appuyons sur le gestionnaire 3D de visualisation moléculaire PyMOL.

Le problème de l'hétérogénéité des données nous conduit alors à placer la conception d'une Base de Données comme objectif primaire, c'est-à-dire avant même l'implémentation de tous les algorithmes des méthodes prédictives déterminant l'interaction entre deux partenaires.

Dans le paragraphe suivant, j'explique alors la conception de la base de données que j'ai conçue sous PostgreSQL, son remplissage et sa maintenance. Ce projet doit pouvoir s'intégrer complètement dans la plate-forme graphique de gestion de programmes bioinformatiques, développée par Franck Valentin. En conséquence je présente aussi les outils qui permettent de maintenir et faire des requêtes sur cette base au sein de cette plate-forme.

3. Le développement

Il a donc été développé une base de données relationnelle sous PostgreSQL en local afin de pouvoir stocker les séquences protéiques de tous les organismes séquencés de façon homogène ainsi que les paires de partenaires et les listes de graphes.

Les opérations de maintenance de cette base devant être facilement réalisable par l'utilisateur ont mené au développement d'outils à intégrer sous PTOLEMY, la plate-forme de gestion graphique de processus bioinformatiques, objet du projet de Franck Valentin.

Les algorithmes de prédiction de paires de protéines n'ont pas été développés pour la plupart par manque de temps, où parce qu'entre avril et août ces algorithmes ont été implémentés et rendus

publiques par d'autres groupes, mais celui de ROSETTA STONE est présenté avec les résultats déjà acquis à ce niveau sur différents organismes.

a. La base de données sous PostgreSQL

1. Architecture de la base de données

• Tout d'abord quelles données souhaitons stocker?

- les séquences protéiques et le maximum d'informations les concernant.
- les listes de paires de protéines avec leur provenance (Banque de données expérimentales ou paires prédites) et la méthode par laquelle cette paire est déterminée (double hybride, Rosetta Stone, etc..)
- les listes de graphes: pouvoir avoir un identificateur qui pointe sur une liste d'identificateurs protéiques, tous étant partenaires entre eux (directement ou indirectement) avec la même méthode.

A propos des séquences , nos deux principales banques, mises à disposition par Internet, sont celle du **NCBI**, la **nr** et celles de **Expasy**, la **Swissprot** et **TREMBL**.

Les données provenant de la **nr** sont sous forme FASTA c'est-à-dire que chaque séquence possède une ligne d'identificateur derrière un chevron (>) puis les lignes suivantes sont la suite des acides aminés (ou résidus) constituant la séquence de la protéine. (Cf. Annexe 2).

Les informations fournies par la Swissprot sont téléchargeables sous forme de fichier plat et renseignent beaucoup plus de choses à propos d'une séquence protéique. Un exemple d'une fiche protéique pour l'IL2 provenant de la Swissprot est donnée en *annexe 3*. Toutes les protéines provenant de la Swissprot sont renseignées sous ce même format à la suite des unes des autres dans un fichier *sprot.dat*.

Parmi tous ces renseignements, nous décidons ceux qui seront utiles pour l'application future (dans le cas d'éventuelles requêtes), le tableau suivant récapitule les champs du fichier plat, *sprot.dat*, dont il faut stocker l'information dans notre base.

Valeur première colonne de ligne du fichier sprot.dat	Information correspondant au champ
ID	<i>id standard de la protéine</i>
AC	<i>le sp (identificateur théoriquement unique de la séquence protéique dans la Swissprot)</i>
DE	<i>le nom de la protéine suivi de tous ses synonymes qui sont chacun entre parenthèses ()</i>
GN	<i>le ou les noms des gènes associés à la protéine</i>
OS	<i>le premier nom de taxon (l'organisme de la protéine)</i>
OC	<i>la liste des noms de taxon secondaire (même organisme)</i>
OX	<i>l'identificateur du taxon</i>
DR EMBL;	<i>la liste de tous les id EMBL</i>
DR PIR;	<i>la liste de tous les id PIR</i>
DR PDB;	<i>la liste de tous les id PDB</i>
DR GO;	<i>la liste de tous les id GO</i>
DR InterPro;	<i>la liste de tous les id InterPro</i>
DR Pfam;	<i>la liste de tous les id Pfam</i>
DR PRINTS;	<i>la liste de tous les id PRINTS</i>
DR ProDom;	<i>la liste de tous les id ProDom</i>

Valeur première colonne de ligne du fichier sprot.dat	Information correspondant au champ
DR SMART;	la liste de tous les id SMART
DR PROSITE;	la liste de tous les id PROSITE
FT SIGNAL	la liste de tous les intervalles SIGNAL
FT CARBOHYD	la liste de tous les intervalles CARBOHYD
FT HELIX	la liste de tous les intervalles HELIX
FT STRAND	la liste de tous les intervalles STRAND
SQ SEQUENCE	on y obtient la longueur de la chaîne protéique et à partir de la ligne d'en dessous on a la suite des résidus

N.B.: Toutes les données venant de TREMBL sont téléchargeables sous le même format que celles de la Swissprot. Elles sont découpées en plusieurs fichiers (17) dont le descriptif est donné ci-dessous:

<i>arc.dat</i> (Archaea):	4245 entrées
<i>arp.dat</i> (Complete Archaeal proteomes):	33050 entrées
<i>fun.dat</i> (Fungi):	41959 entrées
<i>hum.dat</i> (Human):	49176 entrées
<i>inv.dat</i> (Invertebrates):	147306 entrées
<i>mam.dat</i> (Other Mammals):	18352 entrées
<i>mhc.dat</i> (MHC proteins):	10528 entrées
<i>org.dat</i> (Organelles):	112691 entrées
<i>phg.dat</i> (Bacteriophages):	13750 entrées
<i>pln.dat</i> (Plants):	116371 entrées
<i>pro.dat</i> (Prokaryotes):	169966 entrées
<i>prp.dat</i> (Complete Prokaryotic Proteomes):	330392 entrées
<i>rod.dat</i> (Rodents):	47097 entrées
<i>unc.dat</i> (Unclassified):	963 entrées
<i>vrl.dat</i> (Viruses):	124972 entrées
<i>vrt.dat</i> (Other Vertebrates):	30294 entrées
<i>vrv.dat</i> (Retroviruses):	116571 entrées

data de la release 44 (datant du 5 juillet 2004).

A propos des listes de partenaires, il faut aussi récupérer dans les différentes sources les informations que nous souhaitons retenir quant à la détermination d'une paire. Pour chacune des banques expérimentales auxquelles nous nous intéressons, il faut alors parser les fichiers plats (ou autres formats) et retirer les paires. Ceci signifie pouvoir sortir les caractéristiques suivantes:

- les id des deux partenaires,
- la méthode par laquelle la paire a été déterminée
- la ou les références PubMed associée(s) à cette paire
- la qualité lorsqu'elle est présente
- lequel des deux partenaires est considéré comme la proie et lequel comme le prédateur.

Enfin les graphes sont définis comme étant une liste d'identificateurs de séquences protéiques et une méthode. Ces identificateurs sont tous partenaires entre eux avec la méthode considérée du graphe. Et ils le sont avec une profondeur plus ou moins grande. En effet ils ne sont pas nécessairement partenaires directement: entre deux éléments d'un graphe il peut y avoir plusieurs intermédiaires.

- Les différentes tables de ma base de données:

D'après le descriptif ci-dessus, trois tables principales vont permettre de construire une base de données PostgreSQL **proteinDB**:

- la table **proteins** dans laquelle je vais alors stocker le maximum d'informations récupérables à propos d'une séquence protéique. Le tableau ci-dessous récapitule les différents champs de la table **proteins** avec l'exemple de l'IL2 en tant qu'enregistrement.

Noms des champs de la table PROTEINS	Commentaires sur les champs de la table PROTEINS	Exemple d'enregistrement avec l'IL-2
Idprotein	Id local créé, ce sera ma clef primaire	1 (par exemple, c'est arbitraire)
idstandard	Identificateur trouvé dans le fichier plat de la protéine	IL2_HUMAN
sp	Identificateur d'une séquence protéique dans la banque Swissprot.	P60568
premiernom_prot	Nom de la protéine	Interleukin-2 precursor
listenom_prot	Liste des noms secondaires de la protéine	{IL-2, T-cell growth factor, TCGF, Aldesleukin}
premiernom_gene	Le premier nom de gène associé à la protéine	{IL2.}
listenom_gene	Liste des noms des gènes associés à la protéine	{}
premiertaxon	l'organisme d'où vient la protéine	Homo sapiens (Human)
listetaxon	Liste des noms des taxons de la protéine	{Eukaryota, Metazoa, Chordata, Craniata, Vertebrata, Euteleostomi, Mammalia, Eutheria, Primates, Catarrhini, Hominidae, Homo.}
idtaxon	Identificateur du taxon	9606
listeembl	Liste des identificateurs embl de la protéine	{J00264, X01586, V00564, X00695, K02056, M13879, K03174, S77834, S82692, AF359939, M22005, M33199, A14844}
listepir	Liste des identificateurs pir	{A01849}
listepdb	Liste des identificateurs pdb de la protéine	{1M48, 1M49, 1M4A, 3INK, 1IRL, 1ILM, 1ILN, 1M47, 1M4B, 1NBP}
listego	Liste des identificateurs go de la protéine	{GO:0005615, GO:0008189, GO:0005134, GO:0019209, GO:0019735, GO:0007155, GO:0007267, GO:0006955, GO:0030101, GO:0030307, GO:0008284, GO:0030217}
listeinterpro	Liste des identificateurs interpro de la protéine	{IPR000779}
listepfam	Liste des identificateurs pfam de la protéine	{PF00715}
listeprints	Liste des identificateurs prints de la protéine	{PR00265}
listeprodom	Liste des identificateurs prodom de la protéine	{PD003649}
listesmart	Liste des identificateurs smart de la protéine	{SM00189}
listeprosite	Liste des identificateurs prosite de la protéine	{PS00424}
listesignal	La liste des intervalles SIGNAL sur la séquence de la protéine	{1-20}
listeglyc	La liste des intervalles CARBOHYD sur la séquence de la protéine	{23-23}
listehelix	La liste des intervalles HELIX sur la séquence de la protéine	{27-49, 53-59, 73-76, 77-80, 83-93, 102-117, 134-149}
listestrand	La liste des intervalles STRAND sur la séquence de la protéine	{64-64, 67-67, 127-127, 132-132}
longueur	Longueur de chaîne des résidus	153

Noms des champs de la table PROTEINS	Commentaires sur les champs de la table PROTEINS	Exemple d'enregistrement avec l'IL-2
sequenceaa	La suite des acides aminés constituant la protéine	MYRMQLLSICIALSLALVTNSAPTSSSTKKTQLQLEH LLLDLQMLNGINNYKNPKLTRMLTFKFYMPKKATE LKHLCLEELKPLEEVLNLAQSKNFHLRPRDLISNI NVIVLELKGSETTFMCEYADETATIVEFLNRWITFCQ SIISTLT
gi	Le premier gi associé à la protéine	45593463
listegi	La liste des gi associé à la protéine	{47682793,42542685,42542582,5729676,5032248,69702,69701,13447388,1836111,33781,33809,33784,386819,177013,223637}

- La table **interactions** va stocker les paires de partenaires. Chacun des partenaires de l'interaction est repéré par son `idprotein`, local et unique (cf. Tableau précédent)

Noms des champs de la table INTERACTIONS	Commentaires sur les champs de la table INTERACTIONS
idinteraction	Un identificateur unique pour repérer l'interaction
base	Provenance de l'interaction : nom de la base quand l'interaction est expérimentalement déterminée "prédite" quand l'interaction est déterminée par prédiction
methode	Méthode par laquelle est déterminée l'interaction
idproteina	Idprotein de la protéine partenaire A impliqués dans l'interaction
idproteina	Idprotein de la protéine partenaire B impliquée dans l'interaction
qualité	Qualité de la méthode (champ rarement renseigné actuellement)
proie	Indique lequel des deux partenaires A ou B est la proie
predateur	Indique qui de A ou B est le prédateur

- On pensera à construire par la suite une table de graphes (la table **graphs**) dont les données se basent sur celles contenues dans les deux tables ci-dessus.

2. Les règles définies au fur à mesure de l'avancement et de la découverte des incohérences des fichiers bruts de données.

Règles sur les séquences

La règle essentielle à suivre pour le remplissage de la table **proteins**, celle qui contient toutes les informations sur les séquences protéiques est la suivante :

1 séquence = 1 organisme, c'est-à-dire que chaque enregistrement de la table **proteins** doit être associé à un et un seul organisme.

Règles sur les interactions

- une interaction correspond à une et une seule méthode:
soit "*P1 interagit avec P2*", résultat trouvé avec par exemple l'algorithme de Rosetta Stone **et** les profils phylogénétiques. Ceci est stocké comme deux interactions différentes (2 enregistrements distincts dans la table **interactions**)

- une interaction correspond à une et une seule source:
soit "*P3 interagit avec P4*", résultat listé dans DIP **et** BIND **et** MINT par exemple, ceci sera alors stocké comme 3 interactions différentes (3 enregistrements différents dans la table **interactions**)

- pour la maintenance des interactions stockées dans la base:

lorsqu'une opération de maintenance est lancée sur les interactions il faudra vider complètement la table des interactions et la remplir avec les nouveaux fichiers fraîchement téléchargés des banques de données expérimentales.

Règles sur les graphes

un groupe (ou graphe) considère les interactions données par une seule méthode. Les enregistrements des différents graphes seront à remettre à jour au même moment que la mise à jour des interactions, car ces dernières sont vidées pour être à nouveau chargées.

Règles sur le remplissage de la base

Avec toutes ces séquences provenant de banques de données diverses je fais le choix de remplir la table **proteins** comme suit:

- Commencer par les séquences stockées dans les fichiers plats de la Swissprot et TREMBL. Ecrire un parseur pour ces fichiers afin de ne ressortir que les données qui m'intéressent et qui viendront remplir les différents champs de ma table **proteins**.

Pour suivre la règle 1 séquence = 1 organisme:

si je rencontre plusieurs taxons différents (soit des organismes distincts) pour un même **sp**
alors je duplique l'enregistrement autant de fois qu'il y a d'organismes.

fsi.

- Compléter ensuite le remplissage de la table par les séquences stockées dans la nr (Non Redundant database du NCBI). Dans ce cas les séquences sont associées à des fiches FASTA (cf. Annexe 2). Pour chacune de ces fiches:

si un **sp** (identificateur swissprot) est présent dans la fiche,

alors si l'organisme correspondant de la fiche FASTA est le même que celui du **sp** déjà stocké dans la table **proteins**

alors mettre à jour les champs **gi** et **listegi** de la table **proteins** (cf. tableau de la table **proteins** ci-dessus).

sinon rajouter un enregistrement dans la table **proteins**.

fsi.

sinon insérer un nouvel enregistrement dans la table **proteins** en renseignant dans la mesure du possible les champs **gi**, **listegi**, **sequenceaa** et **premiertaxon**.

si l'organisme (**premiertaxon**) n'est récupérable pas dans la fiche FASTA

alors forcer le champ **premiertaxon** à "unknown".

fsi.

fsi.

Nous avons remarqué plus haut que pour un même enregistrement FASTA plusieurs organismes peuvent lui être associés (deux séquences pouvant exactement être identiques et appartenir à deux organismes distincts). Dans ce cas la démarche retenue est :

– je crée une liste de **gi** correspondant à la fiche FASTA {**gi**1, **gi**2, ..., **gi**N}

– je crée une liste d'organismes correspondants {**orga**1, **orga**2, ..., **orga**N}.

si il existe une chaîne de caractères entre crochets ([]) à la suite d'un **gi** et avant le **gi** suivant

alors cette chaîne est l'organisme correspondant à ce **gi**

sinon forcer l'organisme à "unknown"

fsi.

– Ayant cette liste d'organismes pour une fiche FASTA je la traite en supposant que les "unknown" de la liste proviennent du même organisme que le précédent trouvé dans la liste. Pour le cas du

premier élément de la liste, ce sera le même que le deuxième ou le plus proche différent de "unknown".

- J'obtiens donc une liste d'organismes dans laquelle je compare combien d'éléments différents se trouvent. Dans la table **proteins**, j'ajoute (ou met à jour dans le cas d'un sp existant) autant d'enregistrements qu'il y a d'organismes différents dans cette liste {orga1, orga2, ..., orgaN}.

Remarques:

- Quand il y a un **sp** présent dans une fiche FASTA le premier **gi** qui lui est associé est celui se trouvant dans le même bloc que le **sp**, tous les autres constituent une liste de **gi** stockés sous forme de liste dans la table **proteins**. Il faudra au préalable avoir vérifié que tous les **gi** de cette liste appartiennent au même organisme que celui du **sp** considéré.
- Quand il n'y a pas de **sp** dans une fiche FASTA tirée de la NR le premier **gi** est le premier rencontré dans la fiche. Tous les autres constituent une liste associée à la séquence nouvellement ajoutée. Comme précédemment l'organisme devra alors être vérifié au cours de la construction d'une telle liste.
- Les entrées provenant de la TREMBL ne sont pas repérables par leur **sp** dans la NR mais par un de leur identificateur soit **emb** soit **gb** soit **dbj**. Sachant cela les règles à suivre sont les mêmes que pour le traitement d'un **sp**. Nous nous assurons qu'il n'y a pas de **sp** dans la fiche FASTA au préalable.
- Si nous sommes confrontés à des sources de données autres que celles de la **NR** et celles de la **Swissprot** (comme signalé plus haut), il faudra alors explorer ces données et réfléchir à la méthode à employer pour faire des tables de correspondance entre ces séquences et celles déjà stockées.

3. Les programmes sources

les scripts SQL pour la base de données

Avant de construire les différentes tables concevant ma base de données, j'installe PostgreSQL, prends en main ce SGBDR et crée une base de données **proteinDB**. Dans cette base de données j'utilise les commandes psql et le langage SQL pour créer mes différentes tables.

Par exemple pour la table **proteins** dont le fichier de création, `tableProteins.sql`, est donné en annexe 4, j'utilise la commande :

```
proteinDB=# \i ~/tableProteins.sql
```

les codes java pour parser les fichiers téléchargés

Dans cette partie je décris les différents parseurs que j'ai écrits pour remplir ma base de données. Je décris l'architecture de chacun des programmes et dis quelles sont les principales api de JAVA dont je me suis servi.

Les principaux parseurs que j'ai écrits sont :

- Le Parseur du fichier plat Swissprot/Trembl
- Le Parseur du fichier FASTA de la nr
- Le Parseur du fichier plat de la banque de données MINT
- Le Parseur du fichier xml de la banque de données expérimentale DIP

Le parseur du fichier plat Swissprit/Trembl a pour but de remplir la table **proteins**. Je crée deux classes pour cela:

```
public class SwPrTremblParser
```

```
public class TestSwPrTremblParser
```

TestSwPrTremblParser est la classe qui contient un programme main et fait appel à la classe **SwPrTremblParser**, celle qui contient les méthodes nécessaires pour parser les fichiers plats de Swissprot et Trembl.

Pour parser et tester le flux de caractères du fichier passé en entrée, c'est-à-dire celui de la Swissprot ou de Trembl, dont le type de format est en *annexe 3*, j'utilise l'api JAVA StreamTokenizer. Grâce à l'objet StreamTokenizer que je crée au moment de mon parsing, je peux lui signaler quels caractères sont considérés comme éléments d'un mot et ceux qui ne le sont pas. Je décide quels sont mes séparateurs de mots.

Quand j'ai récupéré toutes les informations qui m'intéressent à propos de l'entrée Swissprot ou Trembl, je lance une requête SQL du type Insert to pour insérer un nouvel enregistrement dans la table **proteins**. Cette requête SQL est réalisable au milieu du code JAVA après avoir installé le driver JDBC (Java DataBase Connectivity) et lancé le postmaster de PostgreSQL (cf. Paragraphe suivant). Pour pouvoir faire une requête SQL dans un programme JAVA à ce niveau il faut aussi faire un `import java.sql.*;`. Ce parsing et ces requêtes d'insertion suivent les règles précédemment définies.

Le parsing des fichiers bruts reste compliqué. Parfois un détail mal écrit, dans le fichier source brut, entraîne une cascade de problèmes sur les entrées suivantes. Je fais le choix alors d'écrire un script de pré-édition de tout ce fichier. Ce script awk (cf. *Annexe 5*) me fournit en sortie un fichier texte plat plus facilement parsable avec mon programme JAVA.

Le parseur du fichier FASTA de la nr a pour objectif de mettre à jour les champs **gi** et **listegi** pour les enregistrements déjà renseignés dans la table **proteins** et d'en ajouter de nouveaux lorsque la séquence rencontrée dans le fichier FASTA n'existe pas dans la table.

Le parseur de la **nr** s'articule sur les mêmes principes que ceux du parseur de la Swissprot et Trembl:

-les deux classes créées sont :

```
public class NRFastaParser
```

```
public class TestNRFastaParser
```

La classe **TestNRFastaParser** contient la méthode main permettant l'exécution du programme en faisant appel à la classe **NRFastaParser** qui lance les méthodes de parsing et les requêtes SQL.

-j'utilise encore l'API StreamTokenizer du langage JAVA.

-et je communique avec ma base de données **proteinDB** via JDBC pour faire des requêtes SQL de mise à jour de la table **proteins**.

La différence se situe au niveau du traitement des données récupérées dans les différentes fiches, et pour cela j'utilise aussi l'API Hashtable de JAVA. Avec cela je gère toutes les listes créées par les différents identificateurs de la ligne de définition de la fiche FASTA.

Comme l'*annexe 2* sur le cas de l'IL2 le montre bien, il peut y avoir plusieurs **gi** pour une même séquence. De plus derrière chaque **gi** il y a des informations qui sont écrites différemment suivant leurs sources originelles. J'appelle bloc, un **gi** et la suite des informations qui le suivent jusqu'au prochain **gi**, ou à la fin de la ligne chevron. La syntaxe générale d'un bloc est :

```
gi |XXXXXX|databasename|YYYYYY|nom et "info dépendant de la base"
```

Le databasename peut être égal à :

```
sp ou emb ou gb ou dbj ou ref ou pir ou prf ou pdb ou pat ou bbs ou gnl ou lcl
```

Généralement ce ne sera que quand je rencontrerais un **emb** ou **gb** ou **dbj** ou **ref** que je pourrais parser un organisme. Dans le cas où je rencontrerais un **sp** je pourrais alors interroger ma table **proteins** et ressortir le ou les organismes associés au **sp**.

Je procède alors de la façon suivante:

pour chaque fiche FASTA (ligne de définition derrière le chevron plus la séquence d'acides aminés):

- construire une liste de gi
- récupérer la séquence
- construire une liste de embl remplie par les id **YYYYYYY** lorsque le **databasename** est **emb** ou **gb** ou **dbj** . Si le **databasename** est différent de **emb** ou **gb** ou **dbj** alors j'ajoute un élément du type "" (chaîne de caractères vide), ce qui permet d'avoir une taille de liste de embl identique à celle des gi.
- Construire exactement comme le point ci-dessus une liste de sp en prenant comme test sur le **databasename** le **sp**.
- Construire enfin une liste d'organisme, sa taille sera identique à celle des listes de gi, embl et sp construites précédemment.

Si le **databasename** est **emb** ou **gb** ou **dbj** ou **ref**

alors chercher le nom de l'organisme entre les crochets

sinon si le **databasename** est **sp**

alors faire une requête dans la base de données pour obtenir l'organisme (on prend arbitrairement le premier trouvé si la requête sort plusieurs enregistrements)

sinon forcer l'organisme à "unknow"

fsi.

Ajouter l'organisme ainsi déterminé à la liste d'organismes.

Mes listes étant ainsi créées, j'utilise une *Hashtable*, structure de données qui permet d'associer une clef à une valeur et ce sous forme de tableau, c'est-à-dire que je peux avoir plusieurs enregistrements dans une *Hashtable*. Chacun est composé d'une clef et d'une valeur.

Dans mon cas précis le type de clef sera un organisme et le type de valeur une liste d'index. Dans la liste d'organismes je regarde combien d'éléments différents il existe, ce qui correspond au nombre d'enregistrements de la *Hashtable*. La structure de la *hashtable* se résume par le tableau:

<i>Clef de la Hashtable</i>	<i>Valeurs associées à la clef</i>
<i>Nom d'un organisme</i>	<i>Liste d'index (correspondant à des entiers)</i>

La liste des index est une liste d'entiers dont les valeurs varient entre 0 et un maximum. Ce maximum est égal à la taille de liste de gi (cf. ci-dessus), qui est aussi la taille de liste de embl ou encore de liste de sp. La taille de la liste d'index est inférieure ou égale à la taille de la liste de gi.

La liste des index dont la clef est l'organisme **human** par exemple est l'ensemble des index des gi dans la liste de gi qui appartiennent à l'organisme **human**.

J'ajoute ou mets à jour autant d'enregistrements dans ma base table **proteins** qu'il y a d'entrées dans ma *Hashtable*.

Le Parseur du fichier plat de la banque de données MINT a pour objectif d'alimenter ma table **interactions**.

Le fichier de données de MINT est un fichier texte plat dans lequel chaque ligne représente une interaction. Nous parons alors facilement ce fichier au moyen de deux classes JAVA:

```
public class ParserMINT
public class TestParserMINT
```

La classe **TestParserMINT** permet d'exécuter le programme en faisant appel à la classe **ParserMINT** dans sa méthode principale `main`.

Comme dans les parseurs précédents j'utilise une connexion JDBC pour faire des requêtes de mise à jour de ma base au sein de mon programme JAVA.

Le Parseur du fichier xml de la banque de données expérimentale DIP, comme celui écrit pour MINT, doit permettre un remplissage de la table **interactions**.

Dans le fichier fourni par DIP, sont trouvés deux principaux blocs:

-un qui liste toutes les protéines (`proteinInteractor`) impliquées dans les interactions de la banque.

-le second décrit les interactions de la banque.

Le premier bloc associe le gi et parfois d'autres identificateurs (`sp`, `pir`, etc.) à un numéro du type "G_23". Et c'est ce numéro qui est repris dans la description des interactions dans laquelle la protéine est impliquée.

Il est donc nécessaire de parser les deux blocs, le premier pour construire une table de correspondance entre des gi et des numéros "DIP" et le second pour ressortir les interactions. En utilisant la table de correspondance préalablement construite, nous pourrons établir les liens entre des **gi**.

Pour parser des fichiers écrits en xml, JAVA fournit des API : DOM et SAX.

J'utilise SAX dans mon programme de passage qui est la classe :

```
public class ParserDIP extends DefaultHandler
```

J'y fais les "import" nécessaires pour pouvoir utiliser SAX :

```
import org.xml.sax.*;
```

```
import org.xml.sax.helpers.DefaultHandler;
```

```
import javax.xml.parsers.SAXParserFactory;
```

```
import javax.xml.parsers.SAXParser;
```

Ma classe hérite donc de la classe `DefaultHandler`, classe JAVA qui fournit des méthodes de passage pour des fichiers au format xml. En implémentant ces méthodes dans ma classe `ParserDIP` j'adapte le parsing au fichier DIP que j'ai en entrée.

Par exemple la méthode :

```
public void startElement(String namespaceURI, String sName, String qName, Attributes attrs) throws SAXException
```

est réimplémentée pour orienter le comportement du parser en fonction des valeurs des balises xml qu'il rencontre dans le fichier DIP.

Remarque:

Pour une même paire de protéines je peux avoir entre mes balises `<interaction>` et `</interaction>` de mon fichier téléchargé de DIP en xml, une liste d'interactions différentes (ou même ayant le même nom mais avec des références PubMed différentes) : dans ce cas je duplique l'enregistrement dans ma table **interactions** autant de fois qu'il y a d'interactions différentes dans ma liste d'interactions entre les balises `<interaction>` et `</interaction>`.

[b. Les briques de maintenance et de requêtes de la base de données sous PTOLEMY](#)

L'objectif de ces briques est de permettre à l'utilisateur final de pouvoir lancer des opérations de maintenance de la base de données au sein de la plate-forme de gestion graphique que Franck VALENTIN a développée, c'est-à-dire sous PTOLEMY.

Je modifie alors mes classes JAVA répertoriées ci-dessus pour qu'elles puissent s'insérer dans le code de PTOLEMY et qu'elles présentent les mêmes méthodes que tous les objets créés dans PTOLEMY.

Pour pouvoir modifier ce qu'exécute l'acteur PTOLEMY indépendamment de lui-même, nous faisons le choix de créer deux classes distinctes. L'une d'entre elles sera l'acteur et l'autre sera son action (ce pourquoi l'acteur est conçu). C'est à l'intérieur de l'acteur que l'on appelle son action.

La classe de l'action doit comporter au moins trois méthodes :

-le constructeur.

-une méthode `initialize()` qui sert à initialiser l'objet : dans mon pipeline quand je dépose l'objet sur mon interface graphique, l'objet est alors créé puis initialisé.

-et enfin une méthode `fire()` qui permet à l'objet de réaliser une action, cette dernière sera exécutée quand j'appuierai sur le bouton RUN dans mon pipeline graphique, mon objet étant créé.

Cette démarche est illustrée en **annexe 6**, avec l'acteur `MintParser.java` et son action `OrglMINParser.java`.

Au moyen de cet acteur, en lui mettant en entrée le fichier plat MINT (cf. Paragraphe précédent), j'exécute une mise à jour de ma table **interactions** (de la base **proteinDB**) au moyen du pipeline graphique développé sous PTOLEMY.

Il est ensuite facile de développer tous les autres acteurs dont nous avons besoin sur le même principe.

c. Programmes sur les interactions prédites

L'algorithme de ROSETTA Stone (méthode de prédiction d'interaction structurale) a été implémenté (Eisenberg et al. NATURE Vol 405 15 JUNE 2000). Il a été exécuté sur différentes banques de données. Les résultats sont présentés dans le tableau ci-après.

<i>COLI</i>	<i>YEAST</i>	<i>FLY</i>	<i>C.Elegans</i>	<i>E value</i>	<i>Tps Exécution ROSETTA</i>	<i>Nb. Paires total</i>	<i>Nb Paires / Organismes</i>
*	*			<10-4	2h12	4 305	Coli 3 422 Yeast 883
*	*			<10-10	3h15	1 925	Coli 1 764 Yeast 161
*		*		<10-4	7h57	8 721	Coli 3 568 Fly 5 153
*		*		<10-10	18h26	5 184	Coli 2 882 Fly 2 302
*			*	<10-4	7h32	6 806	Coli 3 638 C.Elegans 3 168
*			*	<10-10	5h26	4 478	Coli 3 022 C.Elegans 1 456
	*	*		<10-10	8h48	43 555	Yeast 7 560 Fly 35 995
	*		*	<10-10	11h52	34 177	Yeast 8 194 C.Elegans 25 983

NB: plus la e-value est faible, plus le fait que l'alignement donné par BLAST soit le résultat du hasard est faible.

Pour opérer tout ce développement j'ai utilisé, et me suis intéressée à différents outils dont je vais alors faire un bref récapitulatif dans le paragraphe qui suit.

4. Récapitulatif des outils utilisés durant ce stage

Pour développer la base de données en local, nous avons choisi le SGBDR (Système de Gestion de Bases de Données Relationnelle) **PostgreSQL**.

Afin de parser tous nos fichiers de données, nous avons utilisé le langage de programmation **JAVA**.

Nous avons été confrontés à des parsing de fichier xml ce qui nous a conduit à utiliser **SAX** (Simple API for XML) en JAVA.

Pour pouvoir établir une connection avec la base de données implémentée sous PostgreSQL et un programme écrit en JAVA, nous avons utilisé **JDBC** : Java DataBase Connectivity.

Enfin pour l'exploration des fichiers ou autres tâches nous avons écrit des scripts, en utilisant souvent la commande **AWK**.

- PostgreSQL

Le SGBDR PostgreSQL a été choisi car il est Open Source et fournit des fonctionnalités intéressantes.

En effet, par exemple pour les types de données, il est possible de stocker dans un champ une liste dont la taille dépend de chacun des enregistrements insérés dans la table. Dans la description précédente de la base proteinDB, nous avons vu que nous voulons stocker des listes de noms de gènes (table **proteins**). Ainsi dans la création de notre table **proteins**, au moment de la définition du champ `liste_nomgene` il faut déclarer son type de la façon suivante :

```
listenom_gene text[], -- Liste de tous les noms de gènes
```

Le champ `listenom_gene` est une liste d'éléments texte (`text []`), sa taille peut être différente pour chaque création d'un nouvel enregistrement. Les valeurs de cette liste sont entre accolades (`{}`)

Par ailleurs il est possible de modifier les tables en ajoutant et retirant des colonnes qu'elles soient vides ou remplies. Je peux ainsi créer les colonnes **gi** et **listegi** dans ma table **proteins** après avoir inséré tous les enregistrements provenant de la Swissprot et Trembl. Je réalise cela au moyen des commandes suivantes:

```
ALTER TABLE proteins ADD COLUMN gi text;
--ajout de la colonne gi de type text.
ALTER TABLE proteins ADD COLUMN listegi text[];
--ajout de la colonne listegi de type text[].
```

Enfin, il est possible de créer des index afin d'améliorer les temps d'exécution de requête SQL. Les index sont des moyens connus pour améliorer les performances d'une base de données. Un index permet au serveur de la base de données de trouver et récupérer les enregistrements demandés plus rapidement. Leur utilisation doit toutefois se faire avec précaution. Plus d'informations à ce sujet sont trouvées dans la documentation de PostgreSQL en ligne à l'adresse : <http://www.postgresql.com/docs/7.4/interactive/indexes.html>

J'ai, au cours de ce stage, pris en main PostgreSQL pour construire la base **proteinDB**. Le site de PostgreSQL est www.postgresql.com .
Toute la documentation est en ligne sur <http://www.postgresql.com/docs/7.4/interactive/index.html> .

J'ai installé PostgreSQL sur ma machine au moyen d'un "configure, make et make install" classique sous LINUX.

Pour utiliser PostgreSQL, j'ai créé un utilisateur postgres qui peut se connecter au SGBD. J'ai créé un répertoire de données spécifique sur ma machine où doivent être stockées toutes les données générées par la création de bases sous PostgreSQL. Je l'ai signalé alors au moyen de la commande:

```
/usr/local/pgsql/bin/initdb -D /mnt/hda4/DataproteinDB
```

(/mnt/hda4/DataproteinDB est le chemin de mon répertoire où sont stockées les données générées par les utilisateurs PostgreSQL).

En tant qu'utilisateur postgres j'ai demandé la connection au serveur de PostgreSQL en lançant le postmaster comme suit:

```
postgres@igc2:~$ nohup /usr/local/pgsql/bin/postmaster -i -D /mnt/hda4/DATApoteinDB </dev/null >>server.log 2>&1 </dev/null &
```

(le postmaster est à lancer à chaque utilisation de PostgreSQL par le client postgres)

Tout cela étant exécuté je peux alors créer toutes les bases de données que je souhaite au moyen de la commande `createdb`. Puis je peux créer des tables sous cette base nouvellement créée en lançant la commande `psql`. L'exemple ci-dessous montre la création de la base **mydb**, et la création de la table **weather** de la base **mydb**.

```
postgres@igc2:~$ /usr/local/pgsql/bin/createdb mydb
CREATE DATABASE
postgres@igc2:~$ /usr/local/pgsql/bin/psql mydb
Welcome to psql 7.4.2, the PostgreSQL interactive terminal.
Type: \copyright for distribution terms
      \h for help with SQL commands
      \? for help on internal slash commands
      \g or terminate with semicolon to execute query
      \q to quit

mydb=# CREATE TABLE weather (
mydb(#          city varchar(80),
mydb(#          temp_lo int,      -- low temperature
mydb(#          temp_hi int,      -- high temperature
mydb(#          prcp real,        -- precipitation
mydb(#          date date
mydb(# );
CREATE TABLE
```

- JAVA et environnement de développement sous eclipse

Le langage JAVA multi-plateforme a été choisi pour le parsing des fichiers séquences et interactions.

JAVA possède des **api** (SAX et DOM) permettant de parser des fichiers de format xml. Et il existe aussi dans la technologie JAVA une **api** SQL qui, au moyen d'un driver JDBC, permet de communiquer avec une base de données SQL.

Java est open source et le site officiel du langage est <http://java.sun.com/>. Le descriptif des API de la version 1.4.2 est trouvé à l'adresse <http://java.sun.com/j2se/1.4.2/docs/api/>.

Enfin, pour programmer en JAVA j'utilise l'environnement de développement Eclipse développé par IBM (<http://www.eclipse.org/>)

- les API JAVA pour le parsing des fichiers .XML

Pour parser ce fichier xml et en retirer les informations dont j'ai besoin pour remplir ma table **interactions**: je découvre les API: DOM et SAX en java qui font partie de JAXP (Java API for Xml Processing).

Les liens ci-après me permettent de prendre en main ses api.

<http://java.sun.com/xml/jaxp/dist/1.1/docs/tutorial/>

http://java.sun.com/xml/jaxp/dist/1.1/docs/tutorial/sax/2a_echo.html

- JDBC (Java DataBase Connectivity)

Grâce à cette fonctionnalité JAVA, je suis capable de faire des requêtes SQL au milieu d'un programme codé en JAVA. Les sites répertoriés ci-dessous sont ceux que j'ai parcourus pour me documenter sur cette technologie:

<http://java.sun.com/docs/books/tutorial/jdbc/basics/index.html>

<http://java.sun.com/developer/Books/JDBCTutorial/>

http://www.improve-technologies.com/pages/Java/JDBC/Articles_et_Tutoriaux/

<http://www.infini-fr.com/Sciences/Informatique/Langages/Imperatifs/Java/Jdbc/>

Je peux donc avec JDBC interroger une base de données SQL et la mettre à jour en lançant des requêtes du type **update**. Concrètement je fais en parallèle deux tâches au sein du même programme: je parse des fichiers et je mets à jour une base de données. Cela évite de parser un fichier et d'en écrire un autre en sortie dont il faut faire ensuite la copie dans la base de données.

Pour utiliser JDBC il faut :

1. Installer le driver: pour cela deux possibilités s'offrent à nous:

-soit faire `apt-get install libpgjava -- Java database (JDBC) driver for PostgreSQL`

-soit dans le répertoire **src** descendu du site de PostgreSQL (www.postgresql.com) il y a un dossier `jdbc`, avec un `readme`)

2. Avant l'écriture de tout programme, changer la variable d'environnement `CLASSPATH`: il faut que le chemin d'accès des classes qu'utilise `jdbc` y soit précisé. Dans mon cas le **.jar** est dans `/usr/local/pgsql/share/java/postgresql-jdbc3.jar`, donc le `CLASSPATH` devient:

```
$CLASSPATH=$CLASSPATH:/usr/local/pgsql/share/java/postgresql-jdbc3.jar
```

```
$export CLASSPATH
```

3. Toujours avant l'écriture de tout programme java utilisant une connexion JDBC, il faut s'assurer que le SGBD (soit PostgreSQL pour nous) accepte les connexions TCP/IP. Pour cela en appelant le `postmaster` lors du démarrage du serveur, il faut ajouter l'option `-i`. Ou bien il est aussi possible de faire une modification du script de démarrage du serveur (fichier `postgresql.conf`) en forçant la variable `tcpip_socket` à `true` (`tcpip_socket=true`).

4. Dans mon programme JAVA je peux alors charger et initialiser le driver JDBC:

```
* pour utiliser JDBC, importer le package java.sql: import java.sql.*;
```

```
* charger le driver comme suit : Class.forName("org.postgresql.Driver");
```

```
* se connecter à la base de données proteinDB de la manière suivante:
```

```
Connection db = DriverManager.getConnection(url, username, password);
```

```
avec url = jdbc:postgresql:proteinDB
```

```
username : celui de connection à proteinDB (ma base de données sous PostgreSQL)
```

```
password : celui de connection à proteinDB
```

Pour tester cela le programme de test `JDBCtest1`, donné en *annexe 7*, a été écrit et donne à l'exécution ceci:

```
perrine@igc2:~/tmp/workspace/SQLdataBaseCONNECTION$ date; java JDBCtest1; date
```

```
Mon Aug 2 15:40:00 EDT 2004
```

```
57245
```

```
IL2_HUMAN
```

```
0
```

```
reussie!!!
```

```
Mon Aug 2 15:41:28 EDT 2004
```

```
perrine@igc2:~/tmp/workspace/SQLdataBaseCONNECTION$
```

Ce programme lance deux requêtes différentes: un `select` et un `update`.

N.B:

- Ne pas oublier de lancer le `postmaster` avant toute exécution de programme java qui utilise une connexion JDBC
- Sous Eclipse: il est possible de programmer en se connectant à une base de données via JDBC
Pour cela il faut ajouter le **.jar** externe (ici c'est `/usr/local/pgsql/share/java/postgresql-jdbc3.7-4.2.jar`) dans les propriétés du projet Eclipse:
 - clic droit nom du projet
 - properties => fenêtre qui s'ouvre.
 - dans la fenêtre à gauche: Java Build Path
 - onglet Libraries
 - bouton Add External JARs

- Script AWK

Lors de la découverte des fichiers de NR par exemple ou SWISSPROT j'ai utilisé des scripts AWK. Cette commande est très puissante et constitue quasiment un langage à elle-seule. Son écriture ressemble alors à celle du langage C. Les liens qui m'ont permis de me documenter à propos de AWK sont les suivants:

<http://www.vectorsite.net/tsawk.html>

<http://balno.free.fr/awk.html>

<http://www.shellunix.com/>

Les outils bioinformatiques:

J'ai aussi grâce à ce stage découvert et utilisé des outils bioinformatiques comme BLAST, Clustal, Cytoscape, ou encore PyMol. Le lecteur trouvera des renseignements sur ces outils dans le glossaire.

IV APPLICATION

Le gestionnaire graphique de processus bioinformatiques d'analyse de séquences, de structures, d'arbres phylogénétiques et de réseaux d'interactions de protéines a été conçu autour d'une application commandée par l'utilisateur final, l'Unité d'Immunogénétique Cellulaire. L'application est focalisée sur la détection des protéines interagissant directement ou indirectement avec le récepteur des cytokines de la famille des hématopoïétines, la localisation des interfaces sur les séquences et sur les structures, et l'identification des résidus encodant la spécificité de la reconnaissance de l'interleukine-2 humaine par ses récepteurs au cours de l'évolution par comparaison aux autres systèmes cytokine – récepteur.

1. Introduction : l'IL2 et ses récepteurs

L'interleukine-2 (IL2) est une cytokine de 133 résidus sécrétée naturellement par les lymphocytes CD4 lors d'une stimulation par les cellules présentant les antigènes (monocytes, cellules dendritiques, lymphocytes B) au cours d'une infection ou en présence de cellules tumorales. L'IL2 est reconnue par plusieurs types de récepteurs (IL2R) à la surface des lymphocytes (*Fig.1*) et stimule leur prolifération.

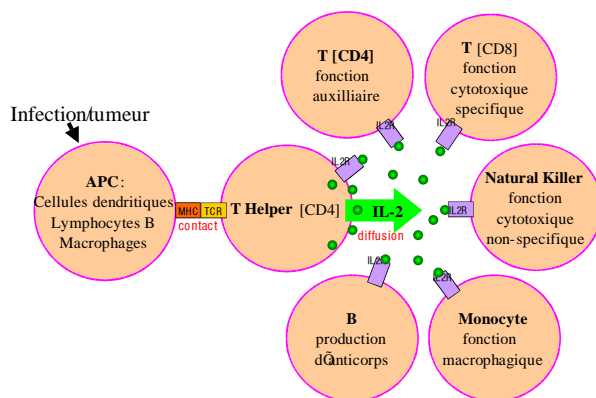


Fig.1: Quand et par quelles cellules est produite et sécrétée l'IL2?

Les récepteurs de l'IL2 comprennent jusqu'à trois chaînes membranaires α , β et γ dont le contrôle de l'expression et de la maturation varie d'un type de lymphocyte à l'autre. La proportion relative des chaînes exprimées dicte la composition oligomériques des récepteurs. Ils se distinguent entre eux par leurs affinités pour l'IL2: $\alpha\beta\gamma$ (10pM), $\beta\gamma$ (1nM) (*Fig.2*)

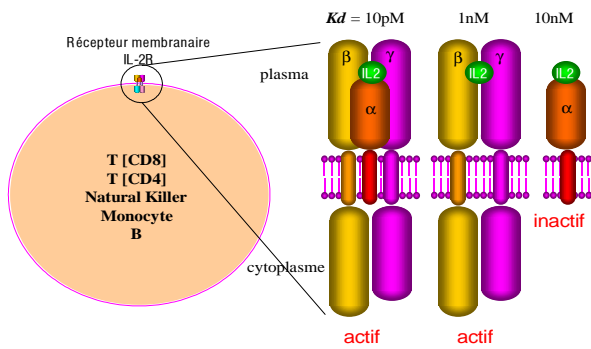


Fig.2: Quels sont les récepteurs actifs de l'IL2?

Aucune structure des chaînes du récepteur n'a été résolue, pas même celles de ses domaines intra ou extra-cytoplasmiques. Seules les structures cristallographiques et par RMN de l'IL2 ont été mises à jour. Les bases moléculaires de la reconnaissance de l'IL2 par ses récepteurs et le mécanisme de modification conformationnelle à l'origine de l'initiation de la transduction du signal ne sont pas documentées expérimentalement. En particulier les hypothèses de pré-assemblage du récepteur avant la fixation de la cytokine ou le rôle de l'assemblage induit par la cytokine sur la transduction du signal ne sont ni démontrées ni invalidées.

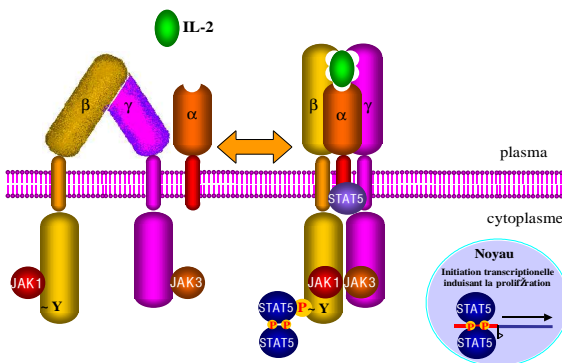


Fig.3: Comment agit l'IL2 par la voie Jak-Stat?

Comme le décrit la figure.3, le domaine cytoplasmique de la chaîne IL2R β fixe JaK1 (tyrosine kinase Janus 1) et IL2R γ fixe JaK3. Le rapprochement des domaines cytoplasmiques met en contact JaK1 et JaK3 qui forment un complexe actif qui catalyse la phosphorylation d'une tyrosine de l'extrémité C-terminale de la chaîne IL2R β . Cette tyrosine modifiée est reconnue par le complexe de facteurs de transcription Stat5a-Stat5b qui s'active par auto-phosphorylation. Ces facteurs induisent l'expression de plusieurs gènes, celui de l'IL2R α par exemple. L'activation du récepteur IL2R induit aussi le blocage des mécanismes apoptotiques via Bcl2, la modification du cytosquelette par la voie Rho et la prolifération par l'activation de plusieurs gènes par la voie Ras/Raf/MapK (Fig.4).

Comment agit l'IL-2 ?

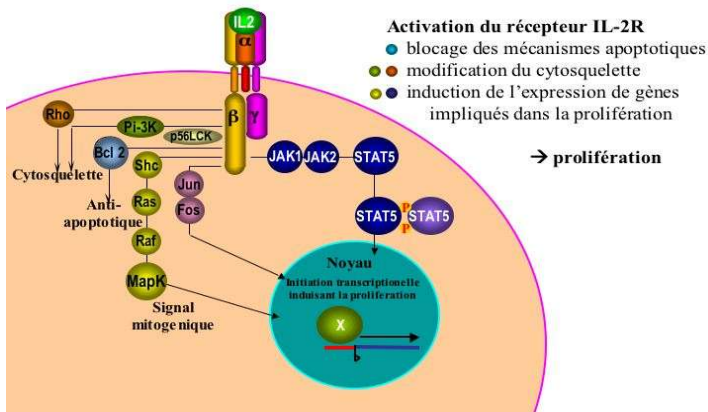


Fig.4: Quels sont les rôles de l'IL2?

2. Importance médicale de l'IL2 et du développement d'agonistes et antagonistes de l'IL2R

L'injection d'IL2 a été testée depuis 1980 chez des patients pour provoquer la prolifération de lymphocytes et réduire le développement de tumeurs (Fig.5). La thérapie a été agréée depuis une dizaine d'années pour le traitement de carcinomes rénaux et de mélanomes. Plus récemment, l'IL2 est utilisée pour reconstituer le taux de CD4 chez certains patients infectés par le virus VIH. Malheureusement, l'IL2 induit aussi des effets secondaires indésirables. Le syndrome de fuite vasculaire (VLS) responsable entre autres d'œdèmes pulmonaires en est un exemple. La gravité des effets secondaires oblige à réduire les doses injectées ce qui diminue l'efficacité des traitements. Aucun agoniste du récepteur ni aucune IL2 modifiée n'a permis jusqu'à présent d'améliorer significativement l'index thérapeutique de l'IL2.

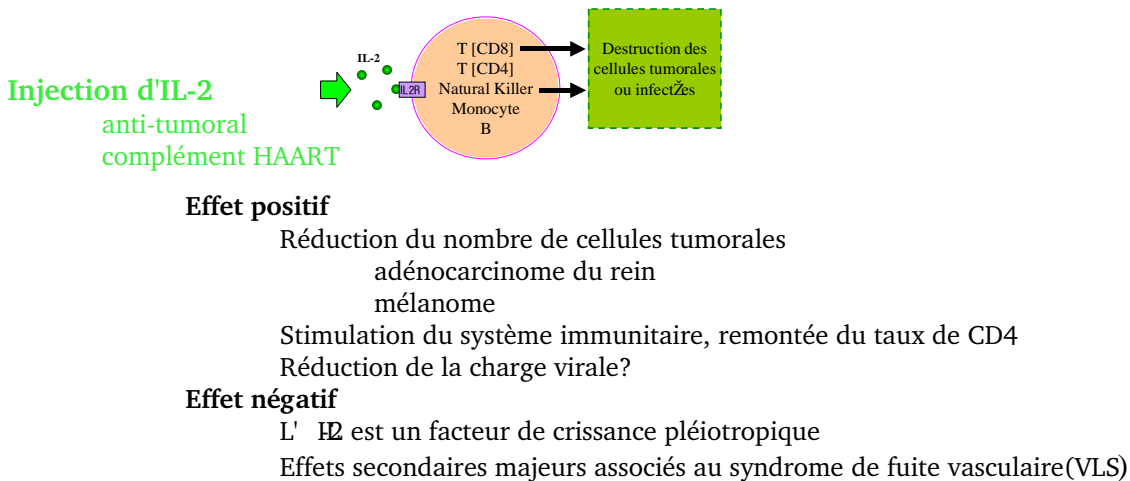


Fig.5: Quels usages thérapeutiques pour l'IL2?

La recherche d'inhibiteurs du récepteur de l'IL2 offrirait des espoirs de thérapies contre certaines maladies auto-immunes ou en complément de la cyclosporine prescrite dans le cas de greffes d'organes. En effet, la cyclosporine est toxique pour le rein et le seul moyen actuellement d'en réduire les doses, est l'injection d'anticorps anti-IL2R α , inhibiteur du récepteur IL2R.

3. Recherches en cours au laboratoire d'Immunogénétique

Cellulaire (IGC)

L'Unité IGC est entre autre engagée dans deux projets conjoints l'un est l'identification du mécanisme de transduction induit par l'IL2 sur les lymphocytes «Natural killers» et l'autre de sélectionner ou concevoir des molécules susceptibles de stimuler cette transduction à la place de l'IL2, toxique chez les patients en traitement.

Ce groupe a été le premier à démontrer la présence de récepteurs pré-assemblés de l'IL2 spécifiques des NK (Eckenberg et al, 2000). Ceux-ci sont aussi stimulables spécifiquement par des peptides mimant la structure de l'IL2 entière (Rose et al., 2003). En absence de données cristallographiques Thierry Rose a reconstruit par modélisation moléculaire comparative les structures tridimensionnelles du dimère de chaînes IL2R $\beta\beta$ libres et associées au tétramère de peptide p1-30₄ (Rose et al., 2003) sur la base du complexe de l'érythropoïétine et de son récepteur résolue par cristallographie. Ces simulations suggèrent un mécanisme de transduction de signal à travers la membrane par la modification des interactions des deux chaînes pré-assemblées $\beta\beta$ (Fig.6). Sous l'effet de la fixation du peptide, les extrémités C-terminales des deux domaines extracytoplasmiques distantes de plus de 80Å, se rapprochent à moins de 30 Å. Par extension de ces domaines, les hélices transmembranaires uniques sont elles aussi rapprochées de 80Å à 30Å. Ce mouvement de plus de 50 Å serait responsable du rapprochement des domaines cytoplasmiques (Fig.6). Chaque domaine cytoplasmique fixe une tyrosine kinase (JaK); le rapprochement des deux kinases permettrait de phosphoryler une tyrosine de l'IL2R β reconnue comme site de fixation du complexe des facteurs de transcription Stat5a-Stat5b.

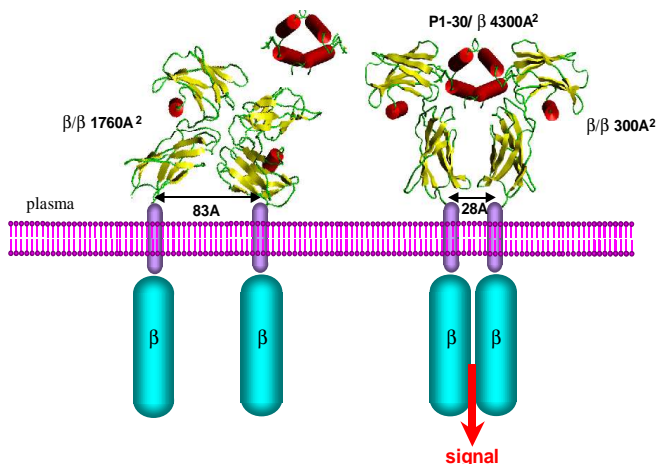


Fig.6: Modèles de la transition de conformation entre le récepteur libre et lié au tétramère de p1-30?

La symétrie du tétramère p1-30₄ serait responsable de l'association préférentielle au récepteur symétrique $\beta\beta$ plutôt que le récepteur asymétrique $\beta\gamma$ (Fig.7). En effet, le peptide p1-30 a la même séquence que l'hélice A de l'IL2 et l'interaction de A avec le récepteur IL2R $\alpha\beta\gamma$, se ferait par l'intermédiaire de la chaîne β .

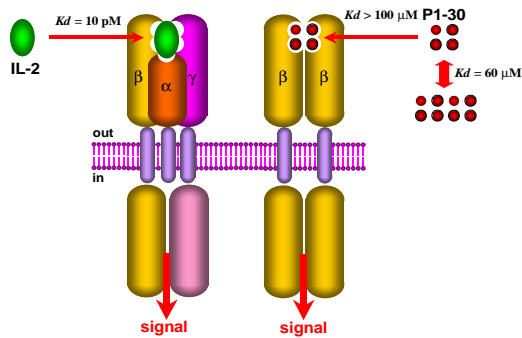


Fig.7: Spécificité de p1-30 par rapport à l'IL2?

Ce mécanisme semble être généralisable aux IL2 de tous les organismes (vertébrés postérieurs aux poissons osseux) et leurs mimétiques sur leurs récepteurs naturels $\alpha\beta\gamma$, $\beta\beta\gamma$ et $\beta\gamma$ mais aussi pour tous les systèmes cytokine-récepteur de la famille des hématopoiétines. Le rapprochement des domaines internes juxtapose deux kinases Jak; celles-ci peuvent différer en fonction des chaînes mais beaucoup utilisent les mêmes qui activeront des facteurs de transcription STAT proches voire identiques comme l'indique le tableau ci-dessous. Les chaînes de signalisations sont très intriquées et constituent de véritables réseaux. Pourtant les réponses physiologiques sont distinctes en raison des contextes d'activation et des régulations d'expression des différents protagonistes.

Tableau : Groupes des cytokines et de leurs récepteurs constitués d'une chaîne IL2Rg.

Cytokines	Récepteurs
IL2	IL2R α , IL2R β , IL2R γ
IL4	IL4R α , IL2R γ
IL7	IL7R α , IL2R γ
IL9	IL9R α , IL2R γ
IL13	IL13R α , IL4R α , IL2R γ
IL15	IL15R α , IL2R β , IL2R γ
IL21	IL21R α , IL2Rg

Fig.8: Partage de chaînes entre les récepteurs du groupe de l'IL2

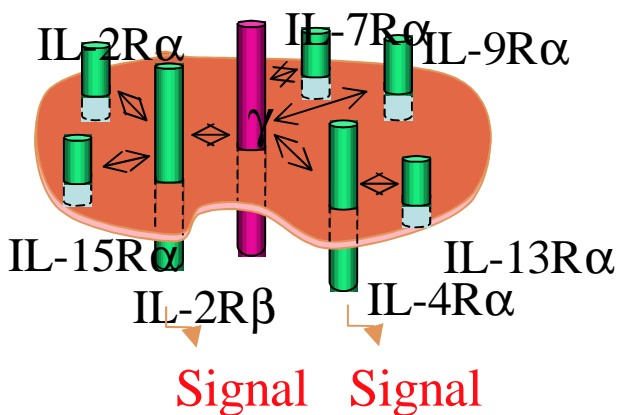


Tableau : Jak et STAT activés par les cytokines de la famille des hématopoïétines (extrait de Itoh et Arai, 1999, p173, The cytokine network and Immune functions, editor J.Thèze, Oxford Press)

Cytokines	Jak	STAT
IL2, IL7, IL9, IL15, IL21	Jak1, Jak3	STAT3, 5
IL4	Jak1, Jak3	STAT6
IL13	Jak1	STAT6
IL3,IL5	Jak1,Jak2	STAT5
IL6,IL11,LIF,CNTF,OSM	Jak1,Jak2/Tyk2	STAT3/1
IL12	Jak2, Tyk2	STAT3,4
G-CSF	Jak1, Jak2	STAT1,3
EPO	Jak2	STAT5
TPO	Jak2	STAT1,3,5
GH	Jak2	STAT5
PRL	Jak1,Jak2	STAT5
Leptin	?	STAT3,5,6
IFNa/b	Jak1,Tyk2	STAT1,2,3
IFNg	Jak1,Jak2	STAT1
IL10	Jak1,Tyk2	STAT1,3,5
EGF	Jak1	STAT1,3
PDGF	Jak1,Jak2/Tyk2	STAT1,3
M-CSF	Jak1,Tyk2	STAT1,3,5

La structure d'aucun des domaines internes des récepteurs est connue, ni même celle d'aucune des Jak ou Tyk. Nous disposons actuellement de données structurales sur la partie externe du récepteur d'érythropoïétine (EPO), de l'hormone de croissance et de l'IL4, et sur un fragment d'un des facteurs de transcription STAT.

Les connaissances au niveau moléculaire sont encore plus confuses sur les autres voies de signalisation (Rho, MapK, Pi3K) en raison du nombre d'intermédiaires, de la faible documentation de leurs fonctions biochimiques et de leurs structures.

4. Approches bioinformatiques

Les approches bioinformatiques que j'ai mises en application sur le projet cytokine-récepteur ont pour objet d'assembler les outils dont nous avons besoins pour documenter les interactions à partir de bases de données, visualiser les réseaux connus et proposer d'éventuelles interactions par des méthodes prédictives.

Dans le chapitre précédent j'ai décrit la constitution des bases de données de séquences, d'interactions et de graphes qui sont la clé de voûte de ce projet. L'utilisation de ces données lors de la phase d'application peut être décrite ainsi :

- Rechercher les séquences orthologues à chaque cytokine, chaînes de récepteur, intermédiaires de signalisation, puis les classer par organismes
- Produire des alignements multiples afin de construire les arbres phylogénétiques et permettre la localisation des positions aux interfaces qui ont co-évolué.
- Produire les réseaux d'interactions propres à chaque cytokine et rechercher avec des méthodes prédictives d'éventuels partenaires des protéines du réseau sélectionné.
- Produire des arbres phylogénétiques au sein des systèmes cytokines récepteurs.
- Rechercher les éventuelles co-évolutions de positions dans les blocs de séquences afin de localiser les points de contacts aux interfaces.

- Figurer les réseaux selon des critères de filtrage et permettre leur documentation interactive

L'annexe 8 est un workflow représentatif de toutes ses étapes sous PTOLEMY

a. Rechercher les séquences orthologues à chaque cytokine, chaînes de récepteur, intermédiaires de signalisation, puis les classer par organismes

Les paramètres de recherche des séquences (matrice de substitution, espérance e, création et extension de lacune) dans la database de séquences NRprot (NCBI) par la méthode Blast (NCBI) ont été optimisés par un workflow de recherche itérative mise au point par Franck Valentin. Le niveau de détection de séquence n'est pas amélioré dans les cas présents par l'utilisation de la méthode itérative PSI-Blast ou par l'utilisation de profils HMM.

Le tableau ci-dessous résume notre recherche des orthologues des cytokines de la famille des hématopoïétines (homologue à l'EPO) du groupe de l'IL2 (récepteur à chaîne IL2Rg), des chaînes des récepteurs, des protéines Janus kinases et des facteurs nucléaires de transcription STAT. Les différentes colonnes représentent de gauche à droite :

- le nom de la protéine
- la longueur de la séquence humaine avant maturation
- le nombre d'organismes où cette protéine a été identifiée
- la présence d'un signal de sécrétion détecté par la méthode SignalP <http://www.cbs.dtu.dk/services/SignalP/>
- le nombre de segments transmembranaires détectés par la méthode TMHMM (Trans Membran Hidden Markov Model) <http://www.cbs.dtu.dk/services/TMHMM/>
- les domaines d'interaction peptidiques reconnus (Pfam, Prosite, Smart)
- l'identification expérimentale d'une activité kinase
- nombre de tyrosines phosphorylées après activation
- nombre de ponts disulfures
- nombre de sites de glycosylation
- existence d'une structure cristallographique (Protein Database à RCSB, Rutgers University)
- existence d'un modèle prédictif calculé par l'unité IGC.

Protéines	Long	Nombre	signal	TM	PRM	Kin	pY	S-S	Glyc	x3D	model
IL2	153	34	1-20	non	non	non	non	1	1	oui	oui
IL4	153	26	1-24	non	non	non	non	3	1	oui	oui
IL7	177	7	1-25	non	non	non	non	3	3	non	oui
IL9	144	4	1-18	non	non	non	non	non	4	non	oui
IL13	132	11	1-20	non	non	non	non	2	4	non	oui
IL15	162	12	1-29	non	non	non	non	2	1	non	oui
IL21	162	7	1-22	non	non	non	non	1	1	non	oui
IL2Ra	272	8	1-21	1	non	non	non	5	6	non	partiel
IL2Rb	551	1	1-25	1	non	non	1	2	4	non	partiel
IL2Rg	379	6	1-22	1	non	non	non	2	5	non	partiel
IL4Ra	825	10	1-25	1	non	non	4	2	6	non	partiel
IL7Ra	459	3	1-20	1	non	non	2	non	6	non	partiel
IL9Ra	522	4	1-40	1	non	non	1	non	2	non	partiel
IL13Ra	427	8	1-36	1	non	non	1	non	oui	non	partiel
IL15Ra	267	3	1-37	1	non	non	1	non	oui	non	partiel
Jak1	1142	4	non	non	SH2,PK,FERM	oui	1	non	non	non	non
Jak2	1132	3	non	non	SH2,PK,FERM	oui	1	non	non	non	partiel
Jak3	1124	3	non	non	SH2,PK,FERM	oui	1	non	non	non	non
Tyk2	1187	2	non	non	SH2,PK,FERM	oui	1	non	non	non	non

STAT1	750	2	non	non	SH2	oui	2	non	non	non	non
STAT2	851	3	non	non	SH2	oui	2	non	non	non	non
STAT3	770	4	non	non	SH2	oui	2	non	non	non	non
STAT4	748	2	non	non	SH2	oui	2	non	non	non	non
STAT5a	794	6	non	non	SH2	oui	2	non	non	non	non
STAT5b	787	4	non	non	SH2	oui	2	non	non	non	non
STAT6	847	2	non	non	SH2	oui	1	non	non	non	non

b. Produire des alignements multiples afin de construire les arbres phylogénétiques et permettre la localisation des positions aux interfaces qui ont co-évolué

Les fichiers résultats de recherche de séquences par Blast ont été filtrés par le programme blast2list écrit par le groupe IGC et qui permet de sortir, filtrer et classer les séquences par score ou organismes. Les listes ont été utilisées pour produire des fichiers où figurent les séquences complètes au format FASTA avec le programme fastacmd du package Blast (NCBI). Ce fichier est alors utilisé en entrée par le programme ClustalW (Thomson et al. 1987) pour produire un fichier d'alignement multiple. Les paramètres d'alignement ont aussi été testés par Franck Valentin (matrices de substitution, espérance e, création et extension de lacune) en vue d'optimiser le taux d'identité par paire et le taux de similitude sur la totalité du multi-alignement calculé par le programme cluscore écrit par le groupe IGC

Les fichiers Clustal (.aln) ont été utilisés pour évaluer le niveau de conservation de chaque résidu pour chaque sous-famille au moyen du programme Cosa développé par le groupe IGC. Ce programme permet aussi d'afficher l'entropie de Shannon (Oliveira et al. PROTEINS: Structure, Function, and Genetics 52:544-552 (2003)) à chaque position qui donne une estimation de la diversité de la nature des résidus à chaque position. Vladimir Daric a intégré Cosa dans le gestionnaire de workflow pour produire des images de modèles moléculaires tridimensionnels (Fig. 9) affichant le niveau de conservation des résidus ou leur diversité à partir d'un fichier au format Clustal contenant parmi toutes les séquences au moins la séquence d'une protéine de structure connue.

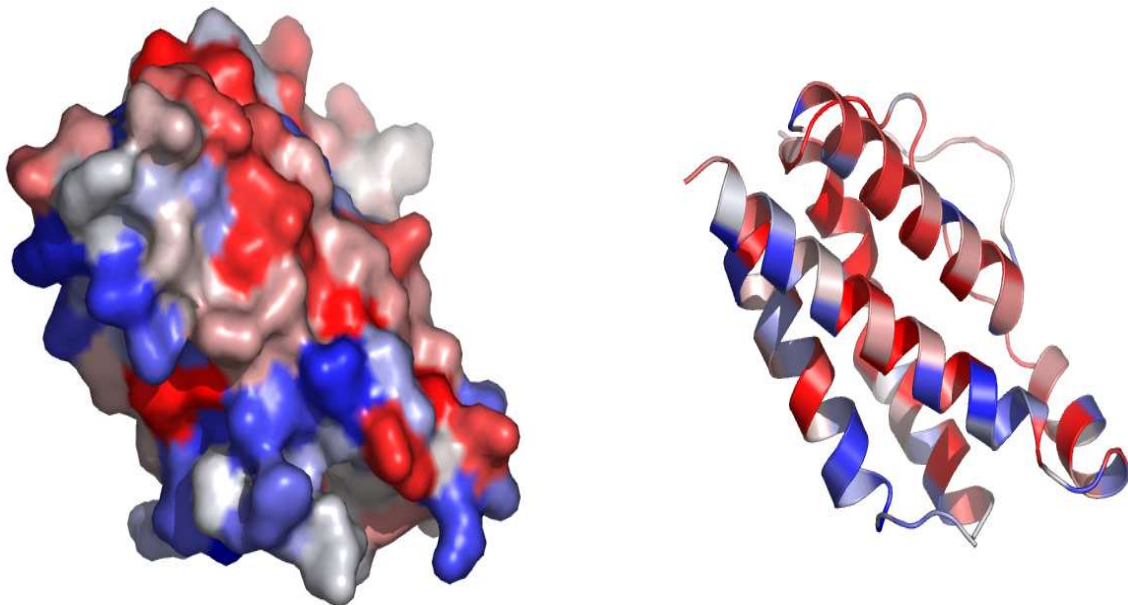


Fig.9 Conservation projetée sur la structure cristalline de l'IL2 humaine. La surface de l'IL2 accessible au solvant est figurée à gauche et le schéma du squelette peptidique à droite. Les zones rouges sont très conservées. Les zones en bleu sont peu conservées.

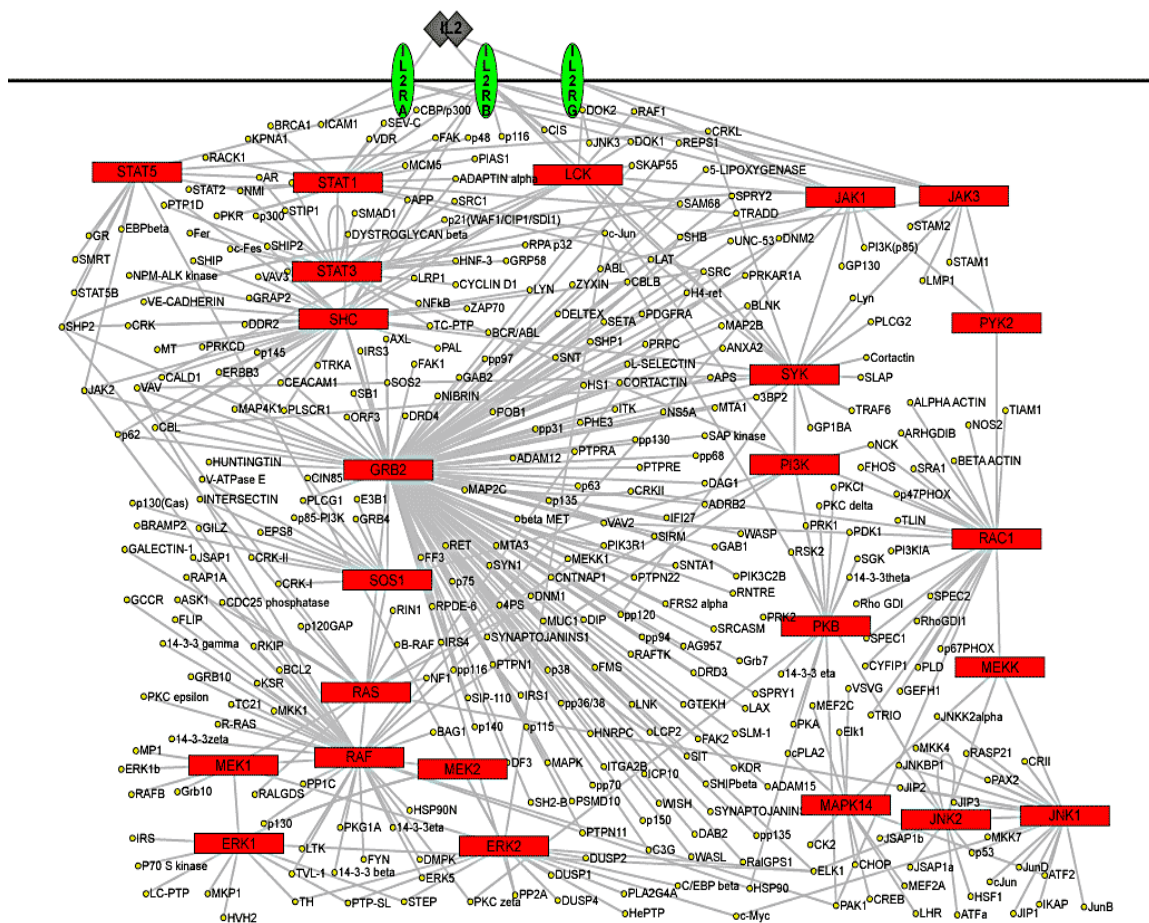
Le niveau de conservation est hétérogène à la surface de l'IL2. L'interface avec les chaînes IL2Ra et IL2Rg du récepteur sont conservées (surfaces rouges) d'un organisme à l'autre mais pas à l'interface avec IL2Rb.

Ceci suggère que la chaîne IL2Rb confère la spécificité pour la reconnaissance de l'IL2 par son propre récepteur. Ceci est cohérent avec le fait que l'IL2 de souris ne se fixe pas sur le récepteur humain et qu'il faut produire une souris transgénétique dotée du gène humain encodant pour l'IL2Rb pour observer un effet de l'IL2 humain chez la souris. L'observation est identique pour l'IL4R α , IL7R α , IL9R α et IL21R α qui encoderaient d'après notre approche bioinformatique, la spécificité pour les cytokines IL4, IL7, IL9 et IL21 respectivement.

c. Produire les réseaux d'interactions propres à chaque cytokine et rechercher avec des méthodes prédictives d'éventuels partenaires des protéines du réseau sélectionné

Comme nous l'avons décrit dans la construction de la base de données, nous avons collecté les listes d'interactions expérimentales à partir de différents sites qui se distinguent par les méthodes de récupération des informations à partir de leurs sources bibliographiques. La liste d'interactions distribuée par HPRD se distingue parmi les autres par le fait qu'elle est focalisée sur un seul organisme, l'homme, et que les sources bibliographiques sont validés ou rejetées à la main par un expert pour chaque interaction détectée par une méthode expérimentale qui ne met pas en jeu de double hybride. Nous avons construit à partir de cette base de données le squelette de notre réseau chez l'homme des protéines qui interagisse avec l'IL2 ou l'un de ses partenaires (Fig.10)

Fig.10: Réseau d'interaction des protéines associées à l'IL2 humaine directement ou indirectement à partir de la base HPRD



Nous avons reconstitué par extrapolation ce réseau chez les organismes les plus documentés en termes de séquences de systèmes cytokine-récepteur : la souris, le rat, le chien, le chat, le bœuf, le porc et le babouin. Nous avons recherché les séquences orthologues de chaque séquence humaine du réseau chez chacun des organismes sélectionnés. Chaque réseau constitue une couche où les orthologues sont rigoureusement superposés. Enfin nous avons complété chaque couche avec les données provenant des autres bases de données DIP, MINT et Intact. Les réseaux par organismes sont visualisés avec Cytoscape (Fig.11 a et b). Les réseaux en couches multiples sont visualisés et manipulés avec PyMol en utilisant le programme multi_gml2pdb qui traduit une suite de fichiers GML en un fichier au format PDB.

Fig.11 : Réseau d'interactions des cytokines du groupe de l'IL2 avec leurs récepteurs et la propagation sur, au plus, deux voisins. A gauche une représentation selon l'algorithme hiérarchique de Sugiyama et à droite une distribution radiale parmi les nombreuses offertes par les plug-ins de layout de Cytoscape.

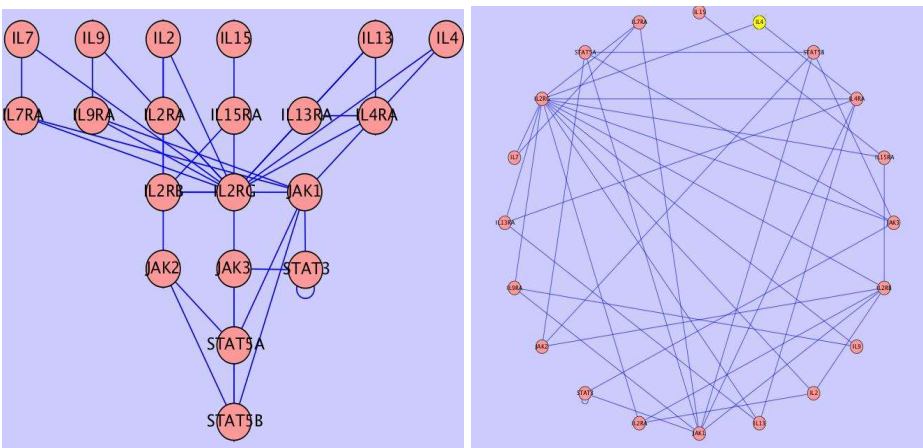
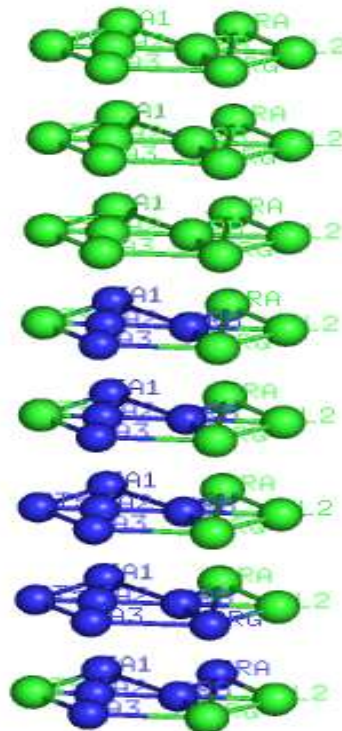


Figure 12. Réseau d'interactions multicouche des protéines associées à l'IL2 directement ou indirectement par l'intermédiaire d'au plus 3 protéines.

Dans l'ordre de haut en bas : humain souris rat cochon vache chien chat mouton



les nœuds de ces réseaux sont en 2D

STAT5

Jak1 IL2-Ra

Jak2 IL2-Rb

Jak3 IL2-Rg

IL2

d. Produire des arbres phylogénétiques au sein des systèmes cytokines-récepteurs

ClustalW produit des arbres phylogénétiques par une méthode ‘neighbor-joining’. Sans être excellente cette méthode s’est avérée suffisante dans notre étude afin de produire les cladogrammes des différentes séquences indiquées dans le tableau du paragraphe 4.2.1. En effet le nombre de séquences est réduit (vertébrés postérieurs aux poissons osseux), la diversité est faible et le niveau d’identité réparti sur toute la longueur des chaînes. Nous avons exploré ces arbres afin de rechercher dans leur superposition en multicouche, une couche par organisme, d’éventuelle corrélation entre l’évolution de la cytokine et de son récepteur. Nous avons utilisé PyMol pour visualiser ces arbres superposés, de la même manière que nous l’avons fait pour les réseaux.

L’utilisation d’opérateurs sur les branches de l’arbre permet de visualiser sur un alignement multiple ou sur un modèle moléculaire le niveau de conservation dans les branches considérées.

e. Rechercher les éventuelles co-évolutions de positions dans les blocs de séquences afin de localiser les points de contacts aux interfaces.

L’un des objectifs de ce travail est de mettre au point une méthode de prédiction d’interaction entre deux protéines et de localiser sur la séquence certains des résidus impliqués dans l’éventuelle interface.

Le programme evoluswap écrit dans le cadre de ce projet permet de retrouver à partir de 2 alignements multiples, toutes les positions où ont eu lieu des échanges de paires ioniques ou de résidus aromatiques d’un organisme à l’autre.

A partir des alignements multiples de 8 couples de séquences retrouvées chez 8 organismes de l’IL2 (il2_evo8.aln) et de l’IL2Ra (il2ra_evo8.aln) nous avons trouvé 2 positions où les chaînes acides et basiques ont été échangées chez 2 organismes (bœuf et mouton) parmi les 8 :

```
Search for ion pair swaps with a threshold > 99.000000%
Between the sets il2_evo8.aln and il2ra_evo8.aln
i_set 1 , seq 9 , pos 49 , j_set 2 , seq 9 , pos 45 , fpair 2 , bpair 6, hole 0
il2_evo8.aln      il2ra_evo8.aln
mouse             QQQHL E QLLMD   KALSY K NGTIL
rat               -QQHL E QLLLD   KALSY K NGTIL
human            -QLQL E HLLLD   KAMAY K EGTM
pig              -KKQL E PLLLD   KILGY K VGTTL
bovine           -MKEV K SLLLD   KVFRY E VGTMI
dog              -EQQM E QLLLD   KALTY K TGTVL
cat              -QQQL E QLLLD   KALTY K TGTML
sheep            -MKEV K SLLLD   KVLRY E VGTMI

i_set 1 , seq 9 , pos 49 , j_set 2 , seq 9 , pos134 , fpair 2 , bpair 6, hole 0
il2_evo8.aln      il2ra_evo8.aln
mouse             QQQHL E QLLMD   LTGHC R EPPPW
rat               -QQHL E QLLLD   LAGHC R EPPPW
human            -QLQL E HLLLD   LPGHC R EPPPW
pig              -KKQL E PLLLD   LPGHC K EPPPW
bovine           -MKEV K SLLLD   LPGHC E EPPPW
dog              -EQQM E QLLLD   LVGHC R EPPPW
cat              -QQQL E QLLLD   LPGHC R EPPPW
sheep            -MKEV K SLLLD   LPGHC E EPPPW
```

A partir des alignements multiples de 3 couples de séquences retrouvées chez 3 organismes de l'IL2 (il2_evo3.aln) et de l'IL2Rb (il2rb_evo3.aln) nous avons trouvé 1 position où les chaînes acides et basiques ont été échangées:

```
Search for ion pair swaps with a threshold > 99.000000%
Between the sets il2_evo3.aln and il2rb_evo3.aln
i_set 1 , seq 4 , pos143 , j_set 2 , seq 4 , pos293 , fpair 2 , bpair 1, hole 0
      il2_evo3.aln      il2rb_evo3.aln
mouse_ncbi      -VTVV K LKGS      IPDPS E FFSQL
rat_sp          -VTVV K LKGSE     IPDPS E FFSQL
human          -VIVL E LKGSE     TPDPS K FFSQL
```

A partir des alignements multiples de 6 couples de séquences retrouvées chez 6 organismes de l'IL2 (il2_evo6.aln) et de l'IL2Rg (il2rg_evo6.aln) nous n'avons trouvé aucune position où les chaînes acides et basiques ont été échangées:

```
Search for ion pair swaps with a threshold > 99.000000%
Between the sets il2_evo6.aln and il2rg_evo6.aln
none
```

A partir des alignements multiples de 6 couples de séquences retrouvées chez 6 organismes de l'IL2Ra (il2ra_evo6.aln) et de l'IL2Rg (il2rg_evo6.aln) nous avons trouvé 4 positions où les chaînes acides et basiques ont été échangées:

```
Search for ion pair swaps with a threshold > 99.000000%
Between the sets il2ra_evo6.aln and il2rg_evo6.aln
i_set 1 , seq 7 , pos 94 , j_set 2 , seq 7 , pos116 , fpair 3 , bpair 3, hole 0
      il2ra_evo6.aln      il2rg_evo6.aln
il2a_mouse_     SNSH- D KSRKQ      HYLFS K EITSX
il2a_rat_77     SNSH- D NSREQ      HYLFS K EITSX
il2a_human_     SSAT- R NTKKQ      HYLFS E EITSX
il2a_pig_39     PTSSP R IPVKQ      HYLFS E GITSX
il2a_bovine     STSPA K NQVKQ      HYLFS E GITSX
il2a_dog_66     SVSP- E NRK GK     HYLFS R EVTAX

i_set 1 , seq 7 , pos140 , j_set 2 , seq 7 , pos217 , fpair 2 , bpair 4, hole 0
      il2ra_evo6.aln      il2rg_evo6.aln
il2a_mouse_     EPPPW K HED--      LIVNH E PRFSL
il2a_rat_77     EPPPW R HED--      QIVDH E PRFSL
il2a_human_     EPPPW E NEA--      QSVDY R HKFSL
il2a_pig_39     EPPPW E HES--      QSVDH R QSFSL
il2a_bovine     EPPPW E HEREP      QSVDH R HSFSL
il2a_dog_66     EPPPW E HEN--      QSVDH R NSFSL

i_set 1 , seq 7 , pos140 , j_set 2 , seq 7 , pos265 , fpair 2 , bpair 4, hole 0
      il2ra_evo6.aln      il2rg_evo6.aln
il2a_mouse_     EPPPW K HED--      GSHTV E E----
il2a_rat_77     EPPPW R HED--      GSHTA E E----
il2a_human_     EPPPW E NEA--      GSNTS K E----
il2a_pig_39     EPPPW E HES--      G-NTS K E----
il2a_bovine     EPPPW E HEREP      GSNTS K ENIEN
il2a_dog_66     EPPPW E HEN--      GSNTS K E----

i_set 1 , seq 7 , pos147 , j_set 2 , seq 7 , pos108 , fpair 5 , bpair 1, hole 0
      il2ra_evo6.aln      il2rg_evo6.aln
il2a_mouse_     ED--S K RIYHF      NNTFQ E CSHYL
il2a_rat_77     ED--T K RIYHF      NNTFQ E CSHYL
il2a_human_     EA--T E RIYHF      NDKVQ K CSHYL
il2a_pig_39     ES--L K RYVYHF     DDKVQ E CGHYL
il2a_bovine     EREPL K RYVYHF     DDKLQ E CGHYL
il2a_dog_66     EN--S K RIYHF      DDKVQ E CGHYL
```

A partir des alignements multiples de 3 couples de séquences retrouvées chez 3 organismes de l'IL2Ra (il2ra_evo3.aln) et de l'IL2Rb (il2rb_evo3.aln) nous avons trouvé 2 positions où les chaînes acides et basiques ont été échangées:

```
Search for ion pair swaps with a threshold > 99.000000%
Between the sets il2ra_evo3.aln and il2rb_evo3.aln
```

```

i_set 1 , seq 4 , pos140 , j_set 2 , seq 4 , pos293 , fpair 2 , bpair 1, hole 0
      il2ra_evo3.aln   il2rb_evo3.aln
il2a_mouse_  EPPPW K HED--      IPDPS E FFSQL
il2a_rat_77  EPPPW R HED--      IPDPS E FFSQL
il2a_human_  EPPPW E NEA--      TPDPS K FFSQL

i_set 1 , seq 4 , pos147 , j_set 2 , seq 4 , pos293 , fpair 2 , bpair 1, hole 0
      il2ra_evo3.aln   il2rb_evo3.aln
il2a_mouse_  ED--S K RIYHF      IPDPS E FFSQL
il2a_rat_77  ED--T K RIYHF      IPDPS E FFSQL
il2a_human_  EA--T E RIYHF      TPDPS K FFSQL

```

A partir des alignements multiples de 3 couples de séquences retrouvées chez 3 organismes de l'IL2Rb (il2rb_evo3.aln) et de l'IL2Rg (il2rg_evo3.aln) nous avons trouvé 3 positions où les chaînes acides et basiques ont été échangées:

```

Search for ion pair swaps with a threshold > 99.000000%
Between the sets il2rb_evo3.aln and il2rg_evo3.aln
i_set 1 , seq 4 , pos293 , j_set 2 , seq 4 , pos116 , fpair 1 , bpair 2, hole 0
      il2rb_evo3.aln   il2rg_evo3.aln
il2b_mouse_  IPDPS E FFSQL      HYLFS K EITSX
il2b_rat_69  IPDPS E FFSQL      HYLFS K EITSX
il2b_human_  TPDPS K FFSQL      HYLFS E EITSX

i_set 1 , seq 4 , pos293 , j_set 2 , seq 4 , pos147 , fpair 1 , bpair 2, hole 0
      il2rb_evo3.aln   il2rg_evo3.aln
il2b_mouse_  IPDPS E FFSQL      DPQ-- K PQRRA
il2b_rat_69  IPDPS E FFSQL      DPQ-- K PQRRA
il2b_human_  TPDPS K FFSQL      DPR-- E PRRQA

i_set 1 , seq 4 , pos293 , j_set 2 , seq 4 , pos252 , fpair 1 , bpair 2, hole 0
      il2rb_evo3.aln   il2rg_evo3.aln
il2b_mouse_  IPDPS E FFSQL      SQQWS K WSQPV
il2b_rat_69  IPDPS E FFSQL      TQQWS K WSQPI
il2b_human_  TPDPS K FFSQL      AQHWS E WSHPI

```

Nous généralisons actuellement cette approche sur l'ensemble des paires de protéines du réseau d'interactions de l'IL2 figuré au paragraphe IV 2 c. Nous l'étendrons ensuite aux réseaux des autres cytokines.

Simultanément nous effectuons la recherche de telle compensation sur un ensemble réduit de 48 complexes de protéines humaines dont la structure est connue (résolution <2.5Å) que nous poursuivrons par un test sur un ensemble de 1500 structures connues issues de la PDB, sélectionnées parmi les 30000 complexes. Ces tests ont pour objectif de quantifier le nombre de faux positifs pour lesquelles un échange acide-base a eu lieu dans une zone du complexe hors de l'interface. De même nous quantifierons les faux négatifs pour lesquels l'échange a eu lieu mais à une position décalée, souvent en raison d'un échange sur 3 chaînes.

En conclusion, cette approche d'identification des éventuels contacts aux interfaces des cytokines avec leur récepteur, sont compatibles avec les modèles moléculaires qui ont été construits et publiés par le groupe IGC (Rose et al, J.Biol.Chem,2003). Elle prévoit aussi les positions des contacts des récepteurs avec les Jak qui ne sont pas actuellement documenté et dont nous n'avons aucun modèle moléculaire. Ces positions sont bien sûr des cibles prioritaires dans les études par mutagenèse dirigées menées par le groupe IGC.

BILAN

Rappelons notre objectif : à partir d'une séquence protéique requête, nous souhaitons obtenir une liste de protéines effectivement observées ou prédites comme partenaires puis identifier les résidus (ou acides aminés) de la séquence potentiellement impliqués dans leur reconnaissance. Le but final de ce genre d'approche est de pouvoir soit déterminer et annoter les séquences orphelines découvertes et produites par les projets de séquençage, soit retrouver ses partenaires structuraux et fonctionnalités afin de mieux appréhender le mécanisme de sa fonction.

J'ai donc construit un enchaînement de programmes qui permettent de réaliser les requêtes, visualiser les résultats et maintenir une base de données. Je les ai adaptés et intégrés à la plate-forme de gestion graphique des programmes de bioinformatique, en continuité du travail de F. Valentin et V. Daric.

Le succès particulier de ce projet repose sur une intégration des données hétérogènes publiées. Un effort particulier a été nécessaire pour développer une base de données qui unifie, standardise ces informations provenant de banques de séquences (la Swissprot (Expasy) et la NR (NCBI)) ou d'interactions (DIP, BIND, IntAct..). De plus l'opération de maintenance de ces données est facilement réalisable au moyen de briques créées dans le pipeline d'opérations bioinformatiques sous PTOLEMY.

REFERENCES Bibliographiques

Peer Bork, Lars J Jensen, Christian von Mering, Arun K Ramani, Insuk Lee and Edward M Marcotte
Protein interaction networks from yeast to human
Current Opinion in Structural Biology 2004

David Eisenberg, Edward M. Marcotte, Ioannis Xenarios & Todd O. Yeastes
Protein function in the post-genomic era
NATURE Vol 405 15 JUNE 2000

Shawn M. Gomez, William Stafford Noble and Andrey Rzhetsky
Learning to predict protein-protein interactions from protein sequences
BIOINFORMATICS Vol. 19 no. 15 2003, pages 1875-1881

Higgins, Bleasby, and Fuchs
ClustalW
(1991) *CABIOS*, 8, 189-191

Robert Hoffmann and Alfonso Valencia
Protein interaction: same network, different hubs
TRENDS in Genetics Vol.19 No.12 December 2003

Ivan Iossifov, Michael Krauthammer, Carol Friedman, Vasileios Hatzivassiloglou, Joel S. Bader, Kevin P. White and Andrey Rzhetsky
Probabilistic inference of molecular networks from noisy data sources.
Bioinformatics 20(8) 2004

Brian P. Kelley, Bingbing Yuan, Fran Lewitter, Roded Sharan, Brent R. Stockwell and Trey Ideker
PathBLAST: a tool for alignment of protein interaction network
Nucleic Acids Research, 2004, Vol. 32

Brian P. Kelley, Roded Sharan, Richard M. Karp, Taylor Sittler, David E. Root, Brent R. Stockwell, and Trey Ideker
Conserved pathways within bacteria and yeast as revealed by global protein network alignment
PNAS September 30, 2003 vol.100

Edward M. Marcotte, Matteo Pellegrini, Ho-Leung Ng, Danny W. Rice, Todd O. Yeastes, David Eisenberg
Detecting Protein Function and Protein-Protein Interactions from Genome Sequences
SCIENCE Vol. 285 30 July 1999

L. Oliveira, A.C.M. Paiva, and G. Vriend
Correlated Mutation analyses on very Large Sequence Families
CHEMBIOCHEM 2002, 3, 1010-1017

Tony Pawson and Piers Nash
Assembly of Cell Regulatory Systems through Protein Interaction Domains
SCIENCES Vol.300 18 April 2003

Jeffrey M. Perkel
Validating the Interactome

The Scientist June 21, 2004

David J.Reiss and Benno Schwikowski

Predicting protein peptide interactions via a network-based motif sampler

Bioinformatics 20(Suppl. 1) Aout 2004

Andrey Rzhetsky, Ivan Iossifov, Tomohiro Koike, Michael Krauthammer, Mitzi Morris, Hong Yu,

Pablo Ariel Duboué, Wubin Weng, W. John Wilbur, Vasileios Hatzivassiloglou and Carol Friedman

GeneWays: a system for extracting, analyzing, visualizing, and integrating molecular pathway data

Journal of Biomedical Informatics 37 (2004) 43-53

Alfonso Valencia and Florencio Pazos

Chap. 20: *Prediction of Protein-Protein Interaction from Evolutionary Information*

Book: STRUCTURAL BIOINFORMATICS edited by Philip E. Bourne & Helge Weissig (2003)

GLOSSAIRE

AGONISTE

Drogue ou autre substance chimique qui peut se lier à un récepteur sur une cellule pour produire une réaction physiologique.

BLAST Basic Local Alignment Search Tool

BLAST est un algorithme pour comparer les séquences biologiques, comme les séquences d'acides aminés de différentes protéines ou les séquences d'ADN de différents gènes. La recherche par BLAST permet de trouver, dans une librairie ou base de séquences, les séquences qui ressemblent à celle qu'un chercheur étudie. Par exemple considérons un gène non connu provenant d'un organisme animal non humain. Pour rechercher des informations sur ce gène, le scientifique peut alors opérer un BLAST de sa séquence sur une base de données humaines, dans laquelle toutes les séquences sont déterminées. Ce programme BLAST est accessible à www.ncbi.nlm.nih.gov/BLAST

CLUSTAL

ClustalW est un programme d'alignement multiple de séquences biologiques, maintenu par Dr Gibson (European Molecular Biology Laboratory), Dr Des Higgins (University of County Cork), and Dr Julie Thompson (IGBMC). Le programme ClustalW aligne progressivement des séquences multiples de protéines ou d'ADN. (<http://www.ebi.ac.uk/clustalw/>)

Higgins, Bleasby, and Fuchs (1991) CABIOS, 8, 189-191

ClustalX est une fenêtre d'interface pour le programme ClustalW.

CYTOKINE

Protéine ou peptide de signalisation extracellulaire qui agit comme médiateur local de la communication entre les cellules.

CYTOSCAPE (www.cytoscape.org)

Plate-forme logicielle qui permet de visualiser des réseaux d'interactions de protéines, gènes ou complexes.

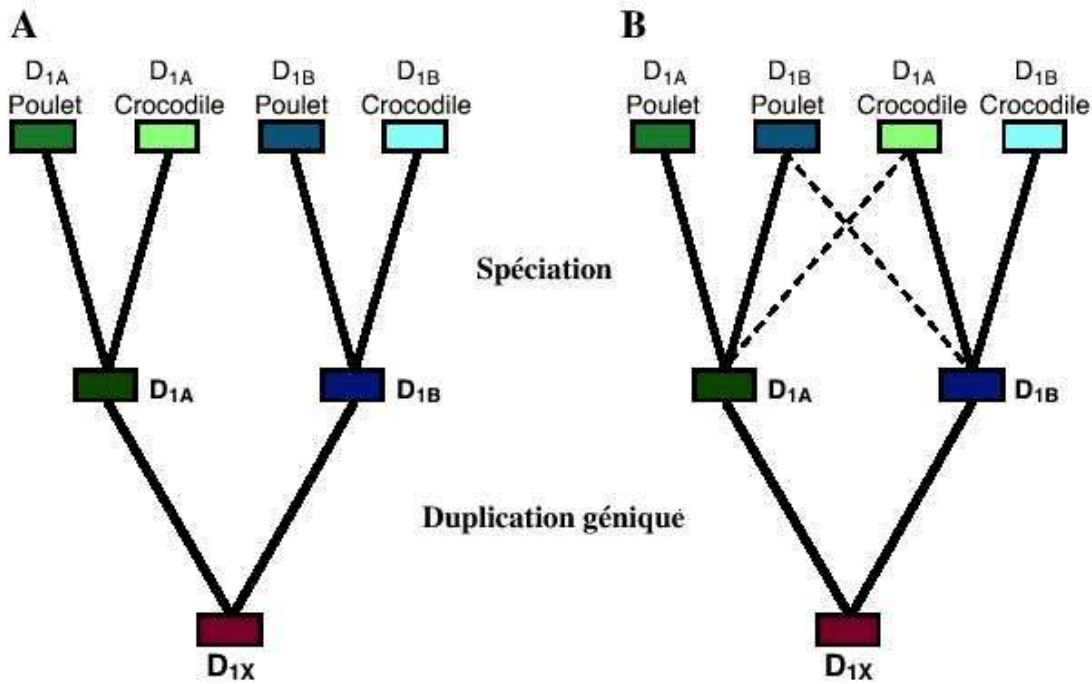
Projet encore en développement en partenariat avec l'ISB (Institute for Systems Biology – Seattle), l'Université de Californie à San Diego, le MSKCC (Memorial Sloan Kettering Cancer Center) et l'Institut Pasteur.

FASTA

Le format FASTA d'une séquence commence avec une ligne de description, suivie par des lignes d'enchaînement d'acides aminés (ceux de la séquence). La ligne de description est distinguée des autres lignes par son premier caractère qui est un chevron (">"). Il est recommandé que les lignes de texte soient longues de moins de 80 caractères.

Gènes Homologues Paralogues et Orthologues : (d'après Vernier et al., 1993).

- A. La duplication génique du gène D1X donne deux paralogues D1A et D1B. Après fixation des caractères, un phénomène de spéciation intervient qui maintient dans les deux espèces (poulet et crocodile) les caractères D1A et D1B. Les gènes D1A de poulet et de crocodile sont donc orthologues.
- B. Souvent, les remaniements géniques rendent difficile l'établissement de relations phylogénétiques. Elles peuvent aboutir à des relations fausses si les séquences de poulet sont plus ressemblantes entre elles qu'entre les paralogues (conversion génique). La relation correcte est indiquée en pointillés.



Quelques types d'homologies moléculaires :

Homologue	Gènes provenant d'un ancêtre commun (exemple, tous les récepteurs D ₁).
Orthologue	Gènes homologues qui ont divergé entre eux après un événement de spéciation (les récepteurs D _{1A} de poulet et de crocodile).
Paralogue	Gènes homologues qui ont divergé entre eux après un phénomène de duplication génique (récepteurs D _{1A} et D _{1B} par exemple)

GENOMIQUE FONCTIONNELLE

La finalité de la génomique fonctionnelle est la compréhension du fonctionnement des gènes et des autres composantes du génome. Puisque le génome s'apparente au plan d'un organisme, la connaissance des gènes, de leur régulation et de leurs interactions avec leur environnement est utile pour prédire le fonctionnement de cet organisme.

GML Graph Modelling Language

C'est un format de fichier portable pour définir des graphes. Les principales caractéristiques du GML sont sa portabilité, sa syntaxe simple, son extensibilité et sa flexibilité. Un fichier GML est constitué de listes hiérarchiques de valeurs clefs.

HEMATOPOIETINES

Cellules sanguines.

HMMER

HMMER est une implémentation gratuitement distribuée de HMM : Hidden Markov Models pour les séquences de protéines. Le programme HMMER est maintenu et développé par le Dr Sean Eddy à l'Université de Washington à Saint-Louis. (www.hmmerr.wustl.edu).

INTERLEUKINE

Peptide ou protéine sécrété(e) qui assure principalement des interactions locales entre leucocytes (globules blancs).

LYMPHOCYTE

Globule blanc qui produit une réponse immunitaire quand il est activé par une molécule étrangère (un antigène). Les lymphocytes T se développent dans le thymus et sont responsables de l'immunité à médiation cellulaire. Les lymphocytes B se développent dans la moelle osseuse chez les mammifères et sont responsables de la production d'anticorps circulants.

NCBI National Center of Biotechnology Information :

Un des deux organismes officiels fournissant les données des séquences protéiques. Chacune de ces séquences est théoriquement repérée par un identificateur appelé **gi**.

<http://www.ncbi.nlm.nih.gov/>

PDB Protein Data Bank

Dépôt des structures biologiques moléculaires tridimensionnelles. La PDB permet la distribution de ces structures.

PSI-MI

Le groupe Proteomics Standards Initiative (PSI) a pour objectif de définir des standards communs pour la représentation des données en protéomique afin de faciliter la comparaison, l'échange et la vérification des données. PSI-MI est un format de fichier (type xml) visant à formaliser la manière de définir et documenter une interaction entre deux partenaires.

(<http://psidev.sourceforge.net/mi/xml/doc/user/>)

PubMed

PubMed est un service de "the National Library of Medicine". PubMed met à disposition plus de 15 millions de références d'articles biomédicaux remontant jusque dans les années 50. Ces références viennent de MEDLINE et des journaux de biologie. PubMed fournit aussi de nombreux liens vers des sites qui donnent l'accès au texte complet des articles.

PyMol (développé par V. DeLano <http://pymol.sourceforge.net/>)

PyMol est un système de visualisation graphique de molécule en 3D. PyMol est écrit en langage de programmation Python. Il permet une visualisation en « temps réel » et une régénération rapide des images graphiques moléculaires de haute qualité lorsqu'il y a animation. Il fournit aussi d'autres fonctionnalités, comme celle d'éditer des fichiers PDB pour assister les chercheurs dans leur recherche.

SIB Swiss Institute of Bioinformatics

Organisme européen fournissant les données des séquences protéiques. Chacune de ces séquences est théoriquement repérée par un identificateur appelé **sp**.

Ces données sont accessibles via le web par le serveur ExpASY (**Expert Protein Analysis System**).

<http://www.expasy.org/>

VOIE METABOLIQUE

Processus chimique qui se produit dans une cellule vivante.

ANNEXES

Annexe 1 :

exemple d'un fichier au format PSI-MI (tiré de la banque HPRD Human Protein Reference Database www.hprd.org).

Annexe 2 :

exemple de fiche FASTA provenant de la nr pour la séquence de l'IL2.

Annexe 3 :

exemple de format de fichier plat provenant de la Swissprot pour la séquence de l'IL2.

Annexe 4 :

script SQL pour la création de la table **proteins**.

Annexe 5 :

Script Awk pour pré-éditer les fichiers bruts de la Swissprot et TREMBL

Annexe 6 :

L'acteur `MintParser.java` et son action `OrgLMINTParser.java`

Annexe 7 :

`JDBCtest1.java` : Programme de test d'utilisation de la connexion JDBC

Annexe 8 :

exemple de Workflow sous PTOLEMY

ANNEXE 1

exemple d'un fichier au format PSI-MI (source : HPRD Human Protein Reference Database)

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE HPRDr2 SYSTEM "hprdr2.dtd">
<HPRDr2 xmlns="org:hprd:dtd:hprdr2">
  <protein id="P00002" annotator="anuradha">
    <title>FAU</title>
    <alt_title>FBR-MuSV associated ubiquitously expressed</alt_title>
    <alt_title>RPS30</alt_title>
    <alt_title>FAU1</alt_title>
    <alt_title>40S ribosomal protein S30 </alt_title>
    <alt_title>FAU encoded ubiquitin like protein</alt_title>
    <alt_title>Ubiquitin like protein FUBI</alt_title>
    <omim>134690</omim>
    <gene_map_locus>
      <title>11q13</title>
      <pubmed>8406491</pubmed>
    </gene_map_locus>
    <entry source="ref_seq">
      <entry_cdna>NM_001997</entry_cdna>
      <entry_protein>NP_001988.</entry_protein>
    </entry>
    <molecular_weight>14391</molecular_weight>
    <!-- Large sections begin here. -->
    <sequence>
      <cdna length="574">cagtgcgctg acacgcagcc cacggtctgt
actgacgcgc cctcgettct tcctctttct
cgactccatc ttcgcggtag ctgggaccgc
cgttcagtcg ccaatatgca gctctttgtc
cgcgccagc agctacacac cttcgaggtg
accggccagg aaacggctgc ccagatcaag
gctcatgtag cctcactgga gggcattgct
ccggaagatc aagtcgtgct cctggcaggc
gcgcccctgg aggatgaggc cactctgggc
cagtgcgggg tggaggccct gactaccctg
gaagtagcag gccgcatgct tggaggtaaa
gtccatggtt ccctggcccc tgctggaaaa
gtgagaggtc agactcctaa ggtggccaaa
caggagaaga agaagaagaa gacaggtcgg
gctaagcggc ggatgcagta caaccggcgc
tttgtcaacg ttgtgcccac ctttggaag
aagaagggcc ccaatgcaa ctcttaagtc
ttttgtaatt ctggctttct ctaataaaaa
agccacttag ttcagtcaaa aaaaaaaaaa
aaaa
</cdna>
      <cdna_utr5 start="1" end="105"/>
      <cdna_coding start="106" end="507"/>
      <cdna_utr3 start="508" end="574"/>
      <protein_sequence length="133">mqlfvraqel htfevtgqet vaqikahvas
legiapedqv vllagapled eatlgqcgve
alttleavagr mlggkvhgs1 aragkvrqgt
pkvakqekkk kktgrakrrm qynrrfvnv
ptfgkkkgpn ans
</protein_sequence>
    </sequence>
    <architecture>
      <domain source="sm" end="70" type="domain" start="1">
        <title>UBQ</title>
      </domain>
    </architecture>
  </protein>
</HPRDr2>
```

```

</architecture>
<expressions>
  <expression>
    <title>Ubiquitous</title>
    <pubmed>1326960</pubmed>
  </expression>
</expressions>
<functions>
  <fclass>Ubiquitin proteasome system protein</fclass>
  <fprocess>Protein synthesis, processing and protein fate</fprocess>
</functions>
<localizations>
  <localization>
    <title>Ribosome</title>
    <pubmed>1326960</pubmed>
  </localization>
</localizations>
<modifications>
</modifications>
<substrates>
</substrates>
<interactions>
<entrySet xmlns="net:sf:psidev:mi"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="net:sf:psidev:mi
http://psidev.sourceforge.net/mi/xml/src/MIF.xsd"
  level="1" version="1">
  <entry>
    <source>
      <names>
        <shortLabel>HPRD</shortLabel>
        <fullName>Human Protein Reference Database</fullName>
      </names>
    </source>
    <availabilityList>
      <availability id="copyright">
        This data is copyrighted by Johns Hopkins University.
        Commercial entities may not use this without prior licensing
        authorization. Other databases must agree to enforce the same
        licensing guidelines before making this data public on their
website.
      </availability>
    </availabilityList>
    <experimentList>
      <experimentDescription id="experimental">
        <interactionDetection>
          <names>
            <shortLabel>experimental</shortLabel>
          </names>
          <xref>
            <primaryRef db="PSI-MI" id="MI:0045"/>
          </xref>
        </interactionDetection>
      </experimentDescription>
    </experimentList>
    <interactorList>
      <proteinInteractor id="ID_HPRD_00002">
        <names>
          <shortLabel>FAU</shortLabel>
        </names>

```

```

<xref>
  <primaryRef db="HPRD" id="HPRD_00002"/>
  <secondaryRef db="PubMed" id="8406491"/>
  <secondaryRef db="Ref-Seq" id="NP_001988."/>
</xref>
<organism ncbiTaxId="9606">
  <names>
    <shortLabel>Human</shortLabel>
    <fullName>Homo sapiens</fullName>
  </names>
</organism>
<sequence>mqlfvraqel htfevtgqet vaqikahvas
legiapedqv vllagapled eatlgqcgve
alittlevagr mlggkvhgsl aragkvrgqt
pkvakqekkk kktgrakrrm qynrrfvnvv
ptfgkkkgpn ans
</sequence>
  </proteinInteractor>
</interactorList>

</entry>
</entrySet>
</interactions>
  <disease></disease>
  <comments></comments>
</protein>
</HPRDr2>

```

ANNEXE 2

exemple de fiche FASTA provenant de la nr pour la sequence de l'IL-2

```
>gi|47682793|gb|AAH70338.1| Interleukin 2, precursor [Homo sapiens]gi|42542685|
gb|AAH66257.1| Interleukin 2, precursor [Homo sapiens]gi|42542582|gb|AAH66255.1|
Interleukin 2, precursor [Homo sapiens]gi|45593462|sp|P60568|IL2_HUMAN
Interleukin-2 precursor (IL-2) (T-cell growth factor) (TCGF) (Aldesleukin)gi|
45593463|sp|P60569|IL2_HYLLA Interleukin-2 precursor (IL-2) (T-cell growth
factor) (TCGF)gi|5729676|gb|AAD48509.1| interleukin 2 precursor [Homo sapiens]
gi|5032248|gb|AAD14263.2| interleukin-2 [Homo sapiens]gi|69702|pir||ICGI2
interleukin-2 precursor - common gibbongi|69701|pir||ICHU2 interleukin-2
precursor [validated] - humangi|13447388|gb|AAK26665.1| interleukin 2 [Homo
sapiens]gi|1836111|gb|AAB46883.1| interleukin-2; IL-2 [Homo sapiens]gi|33781|
emb|CAA23827.1| interleukin-2 [Homo sapiens]gi|33809|emb|CAA25742.1| human
interleukin 2 [Homo sapiens]gi|33784|emb|CAA25292.1| interleukin [Homo sapiens]
gi|386819|gb|AAA98792.1| interleukin 2gi|177015|gb|AAA35454.1| interleukin-2gi|
177013|gb|AAA35453.1| interleukin-2gi|28178861|ref|NP_000577.2| interleukin 2
precursor; T cell growth factor; aldesleukin [Homo sapiens]gi|223637|prf||
0904306A interleukin 2
MYRMQLLSICIALSLALVTNSAPTSSSTKKTQLQLEHLLLDLQMLNGINNYKNPKLTRMLTFKfympkkatELKHLQCLE
EELKPLEEVLNLAQSKNFHLRPRDLISNINVIVLELKGSETTFMCEYADETATIVEFLNRWITFCQSIISTLT
```

ANNEXE 3

exemple de format de fichier plat provenant de la Swissprot pour la sequence de l'IL-2

ID IL2_HUMAN STANDARD; PRT; 153 AA.
AC P60568; P01585;
DT 21-JUL-1986 (Rel. 01, Created)
DT 21-JUL-1986 (Rel. 01, Last sequence update)
DT 05-JUL-2004 (Rel. 44, Last annotation update)
DE Interleukin-2 precursor (IL-2) (T-cell growth factor) (TCGF)
DE (Aldesleukin).
GN Name=IL2;
OS Homo sapiens (Human).
OC Eukaryota; Metazoa; Chordata; Craniata; Vertebrata; Euteleostomi;
OC Mammalia; Eutheria; Primates; Catarrhini; Hominidae; Homo.
OX NCBI_TaxID=9606;
RN [1]
RP SEQUENCE FROM N.A.
RX MEDLINE=84247353; PubMed=6330695;
RA Holbrook N.J., Lieber M., Crabtree G.R.;
RT "DNA sequence of the 5' flanking region of the human interleukin 2
RT gene: homologies with adult T-cell leukemia virus.";
RL Nucleic Acids Res. 12:5005-5013(1984).
RN [2]
RP SEQUENCE FROM N.A.
RX MEDLINE=83167472; PubMed=6403867;
RA Taniguchi T., Matsui H., Fujita T., Takaoka C., Kashima N.,
RA Yoshimoto R., Hamuro J.;
RT "Structure and expression of a cloned cDNA for human interleukin-2.";
RL Nature 302:305-310(1983).
RN [3]
RP SEQUENCE FROM N.A.
RX MEDLINE=84023840; PubMed=6312994;
RA Maeda S., Nishino N., Obaru K., Mita S., Nomiyama H., Shimada K.,
RA Fujimoto K., Teranishi T., Hirano T., Onoue K.;
RT "Cloning of interleukin 2 mRNAs from human tonsils.";
RL Biochem. Biophys. Res. Commun. 115:1040-1047(1983).
RN [4]
RP SEQUENCE FROM N.A.
RX MEDLINE=83246551; PubMed=6306584;
RA Devos R., Plaetinck G., Cheroutre H., Simons G., Degraeve W.,
RA Tavernier J., Remaut E., Fiers W.;
RT "Molecular cloning of human interleukin 2 cDNA and its expression in
RT E. coli.";
RL Nucleic Acids Res. 11:4307-4323(1983).
RN [5]
RP SEQUENCE FROM N.A.
RX MEDLINE=84170356; PubMed=6608729;
RA Holbrook N.J., Smith K.A., Fornace A.J. Jr., Comeau C.M.,
RA Wiskocil R.L., Crabtree G.R.;
RT "T-cell growth factor: complete nucleotide sequence and organization
RT of the gene in normal and malignant cells.";
RL Proc. Natl. Acad. Sci. U.S.A. 81:1634-1638(1984).
RN [6]
RP SEQUENCE FROM N.A.
RX MEDLINE=84170243; PubMed=6324170;
RA Fujita T., Takaoka C., Matsui H., Taniguchi T.;
RT "Structure of the human interleukin 2 gene.";
RL Proc. Natl. Acad. Sci. U.S.A. 80:7437-7441(1983).
RN [7]
RP SEQUENCE FROM N.A.
RX MEDLINE=95239150; PubMed=7722480;
RA Eizenberg O., Faber-Elman A., Lotan M., Schwartz M.;
RT "Interleukin-2 transcripts in human and rodent brains: possible
RT expression by astrocytes.";
RL J. Neurochem. 64:1928-1936(1995).
RN [8]
RP SEQUENCE FROM N.A.
RC TISSUE=Placenta;
RX MEDLINE=96422299; PubMed=8824916;
RA Chernicky C.L., Tan H., Burfeind P., Ilan J., Ilan J.;
RT "Sequence of interleukin-2 isolated from human placental poly A+ RNA:
RT possible role in maintenance of fetal allograft.";
RL Mol. Reprod. Dev. 43:180-186(1996).
RN [9]

RP SEQUENCE FROM N.A.
 RA Rieder M.J., Carrington D.P., Chung M.-W., Lee K.L., Poel C.L., Yi Q.,
 RA Nickerson D.A.;
 RT "SeattleSNPs. NHLBI HL66682 program for genomic applications, UW-
 RT FHCRC, Seattle, WA (URL: <http://pga.gs.washington.edu>).";
 RL Submitted (MAR-2001) to the EMBL/GenBank/DDBJ databases.
 RN [10]
 RP SEQUENCE OF 21-153 FROM N.A.
 RX MEDLINE=89062420; PubMed=3264184;
 RA Weir M.P., Chaplin M.A., Wallace D.M., Dykes C.W., Hobden A.N.;
 RT "Structure-activity relationships of recombinant human interleukin
 RT 2.";
 RL Biochemistry 27:6883-6892(1988).
 RN [11]
 RP SEQUENCE OF 1-69 FROM N.A.
 RX MEDLINE=87064618; PubMed=3491296;
 RA Siebenlist U., Durand D.B., Bressler P., Holbrook N.J., Norris C.A.,
 RA Kamoun M., Kant J.A., Crabtree G.R.;
 RT "Promoter region of interleukin-2 gene undergoes chromatin structure
 RT changes and confers inducibility on chloramphenicol acetyltransferase
 RT gene during activation of T cells.";
 RL Mol. Cell. Biol. 6:3042-3049(1986).
 RN [12]
 RP SEQUENCE OF 1-68 FROM N.A.
 RA Nishino N., Obaru K., Maeda S., Shimada K., Onoue K.;
 RT "Organization of the DNA regions flanking the human interleukin 2
 RT gene.";
 RL Biomed. Res. 6:197-205(1985).
 RN [13]
 RP SEQUENCE OF 21-153, DISULFIDE BOND, AND CARBOHYDRATE-LINKAGE SITE.
 RX MEDLINE=85038540; PubMed=6333684;
 RA Robb R.J., Kutny R.M., Panico M., Morris H.R., Chowdhry V.;
 RT "Amino acid sequence and post-translational modification of human
 RT interleukin 2.";
 RL Proc. Natl. Acad. Sci. U.S.A. 81:6486-6490(1984).
 RN [14]
 RP CARBOHYDRATE-LINKAGE SITE.
 RX MEDLINE=90008901; PubMed=2793860;
 RA Conradt H.S., Nimtz M., Dittmar K.E.J., Lindenmaier W., Hoppe J.,
 RA Hauser H.;
 RT "Expression of human interleukin-2 in recombinant baby hamster kidney,
 RT Ltk-, and Chinese hamster ovary cells. Structure of O-linked
 RT carbohydrate chains and their location within the polypeptide.";
 RL J. Biol. Chem. 264:17368-17373(1989).
 RN [15]
 RP X-RAY CRYSTALLOGRAPHY (3.0 ANGSTROMS).
 RX MEDLINE=88070646; PubMed=3500515;
 RA Brandhuber B.J., Boone T., Kenney W.C., McKay D.B.;
 RT "Three-dimensional structure of interleukin-2.";
 RL Science 238:1707-1709(1987).
 RN [16]
 RP X-RAY CRYSTALLOGRAPHY.
 RX MEDLINE=92335891; PubMed=1631562;
 RA Bazan J.F.;
 RT "Unraveling the structure of IL-2.";
 RL Science 257:410-412(1992).
 RN [17]
 RP RESPONSE TO ABOVE LETTER.
 RA McKay D.B.;
 RL Science 257:412-413(1992).
 RN [18]
 RP STRUCTURE BY NMR.
 RX MEDLINE=92379010; PubMed=1510960;
 RA Mott H.R., Driscoll P.C., Boyd J., Cooke R.M., Weir M.P.,
 RA Campbell I.D.;
 RT "Secondary structure of human interleukin 2 from 3D heteronuclear NMR
 RT experiments.";
 RL Biochemistry 31:7741-7744(1992).
 RN [19]
 RP 3D-STRUCTURE MODELING.
 RX MEDLINE=95111955; PubMed=7529123;
 RA Bamborough P., Hedgecock C.J., Richards W.G.;
 RT "The interleukin-2 and interleukin-4 receptors studied by molecular
 RT modelling.";
 RL Structure 2:839-851(1994).

CC -!- FUNCTION: Produced by T-cells in response to antigenic or
CC mitogenic stimulation, this protein is required for T-cell
CC proliferation and other activities crucial to regulation of the
CC immune response. Can stimulate B cells, monocytes, lymphokine-
CC activated killer cells, natural killer cells, and glioma cells.
CC -!- SUBCELLULAR LOCATION: Secreted.
CC -!- DISEASE: Involved in a form of T-cell acute lymphoblastic leukemia
CC (T-ALL) by a chromosomal translocation t(4;16)(q26;p13) which
CC involves TNFRSF17 and IL2.
CC -!- PHARMACEUTICAL: Available under the name Proleukin (Chiron). Used
CC in patients with renal cell carcinoma or metastatic melanoma.
CC -!- SIMILARITY: Belongs to the IL-2 family.
CC -!- DATABASE: NAME=R&D Systems' cytokine source book: IL2;
CC WWW="http://www.rndsystems.com/asp/g_sitebuilder.asp?bodyId=206".
CC -----
CC This SWISS-PROT entry is copyright. It is produced through a collaboration
CC between the Swiss Institute of Bioinformatics and the EMBL outstation -
CC the European Bioinformatics Institute. There are no restrictions on its
CC use by non-profit institutions as long as its content is in no way
CC modified and this statement is not removed. Usage by and for commercial
CC entities requires a license agreement (See <http://www.isb-sib.ch/announce/>
CC or send an email to license@isb-sib.ch).
CC -----
DR EMBL; J00264; AAD48509.1; -.
DR EMBL; X01586; CAA25742.1; -.
DR EMBL; V00564; CAA23827.1; -.
DR EMBL; X00695; CAA25292.1; -.
DR EMBL; K02056; AAA98792.1; -.
DR EMBL; M13879; AAA59141.1; -.
DR EMBL; K03174; AAA35453.1; -.
DR EMBL; S77834; AAD14263.2; -.
DR EMBL; S82692; AAB46883.1; -.
DR EMBL; AF359939; AAK26665.1; -.
DR EMBL; M22005; AAA59140.1; ALT_INIT.
DR EMBL; M33199; AAA59139.1; -.
DR EMBL; A14844; CAA01199.1; -.
DR PIR; A01849; ICHU2.
DR PDB; 1ILM; 26-JAN-95.
DR PDB; 1ILN; 26-JAN-95.
DR PDB; 1IRL; 07-DEC-95.
DR PDB; 1M47; 25-FEB-03.
DR PDB; 1M48; 25-FEB-03.
DR PDB; 1M49; 25-FEB-03.
DR PDB; 1M4A; 25-FEB-03.
DR PDB; 1M4B; 25-FEB-03.
DR PDB; 1NBP; 18-DEC-02.
DR PDB; 3INK; 31-OCT-93.
DR GlycoSuiteDB; P60568; -.
DR Genew; HGNC:6001; IL2.
DR MIM; 147680; -.
DR GO; GO:0005615; C:extracellular space; TAS.
DR GO; GO:0008189; F:apoptosis inhibitor activity; TAS.
DR GO; GO:0005134; F:interleukin-2 receptor binding; TAS.
DR GO; GO:0019209; F:kinase activator activity; TAS.
DR GO; GO:0019735; P:antimicrobial humoral response (sensu Verte...; TAS.
DR GO; GO:0007155; P:cell adhesion; TAS.
DR GO; GO:0007267; P:cell-cell signaling; TAS.
DR GO; GO:0006955; P:immune response; TAS.
DR GO; GO:0030101; P:natural killer cell activation; TAS.
DR GO; GO:0030307; P:positive regulation of cell growth; TAS.
DR GO; GO:0008284; P:positive regulation of cell proliferation; TAS.
DR GO; GO:0030217; P:T-cell differentiation; TAS.
DR InterPro; IPR000779; Interleukin-2.
DR Pfam; PF00715; IL2; 1.
DR PRINTS; PR00265; INTERLEUKIN2.
DR ProDom; PD003649; Interleukin-2; 1.
DR SMART; SM00189; IL2; 1.
DR PROSITE; PS00424; INTERLEUKIN_2; 1.
KW Cytokine; Glycoprotein; Immune response; Signal; Growth factor;
KW T-cell; Chromosomal translocation; Proto-oncogene; Pharmaceutical;
KW 3D-structure; Direct protein sequencing.
FT SIGNAL 1 20
FT CHAIN 21 153 Interleukin-2.
FT CARBOHYD 23 23 O-linked (GalNAc...).
FT /FTid=CAR_000051.

FT	DISULFID	78	125	
FT	VARIANT	21	21	Missing (in FT-IL2-A and FT-IL2-B).
FT				/FTid=VAR_003967.
FT	VARIANT	22	22	Missing (in FT-IL2-B).
FT				/FTid=VAR_003968.
FT	HELIX	27	49	
FT	TURN	50	50	
FT	HELIX	53	59	
FT	TURN	60	61	
FT	STRAND	64	64	
FT	STRAND	67	67	
FT	HELIX	73	76	
FT	HELIX	77	80	
FT	TURN	81	82	
FT	HELIX	83	93	
FT	HELIX	102	117	
FT	STRAND	127	127	
FT	STRAND	132	132	
FT	HELIX	134	149	
FT	TURN	150	153	
SQ	SEQUENCE	153 AA; 17628 MW; 59E2F40F25860F84 CRC64;		
	MYRMQLLS	ALSLALVTNS	APTSSTKKT	QLQLEHLLLD LQMILNGINN YKNPKLTRML
	TFKFYMPKKA	TELKHLQCLE	EELKPLEEVL	NLAQSKNFHL RPRDLISNIN VIVLELKGSE
	TTFMCEYADE	TATIVEFLNR	WITFCQSIIS	TLT
//				

ANNEXE 4 : Script SQL pour la création de la table proteins

```
CREATE TABLE proteins (  
  idProtein      integer PRIMARY KEY,      -- identificateur interne de la proteine  
  idStandard     text,                    -- l'Id standard de la proteine dans la SwissProt  
  sp             text,                    -- le sp de la proteine (l'Identificateur  
SwissProt)  
  premierNom_Prot text,                  -- Premier nom de la proteine  
  listeNom_Prot  text[],                 -- Liste des noms secondaires de la proteine  
  premierNom_Gene text,                  -- Premier nom du gene  
  listeNom_Gene  text[],                 -- Liste de tous les noms de genes  
  premierTaxon   text,                  -- Le premier nom entier taxon  
  listeTaxon     text[],                 -- Liste des autres noms de taxon  
  idTaxon        integer,               -- identificateur de taxon  
  listeEMBL      text[],                 -- liste du ou des identificateurs EMBL  
  listePIR       text[],                 -- liste du ou des identificateurs PIR  
  listePDB       text[],                 -- liste du ou des identificateurs PDB  
  listeGO        text[],                 -- liste du ou des identificateurs GO  
  listeInterPro  text[],                 -- liste du ou des identificateurs InterPro  
  listePfam      text[],                 -- liste du ou des identificateurs Pfam  
  listePRINTS    text[],                 -- liste du ou des identificateurs PRINTS  
  listeProDom    text[],                 -- liste du ou des identificateurs ProDom  
  listeSMART     text[],                 -- liste du ou des identificateurs SMART  
  listePROSITE   text[],                 -- liste du ou des identificateurs PROSITE  
  listeSignal    text[],                 -- liste du ou des intervalles SIGNAL  
  listeGlyc      text[],                 -- liste du ou des intervalles Glyc  
  listeHelix     text[],                 -- liste du ou des intervalles HELIX  
  listeStrand    text[],                 -- liste du ou des intervalles STRAND  
  longueur       integer,               -- la longueur de la chaine proteique  
  sequenceAA     text                    -- la chaine des acides amines de la proteine  
);
```

ANNEXE 5

Script AWK permettant de pré-éditer les fichiers Swissprot et Trembl

```
#Script de preedition de mes fichiers plats swispprot et trembl
BEGIN{
    premierDE=1;
    okDE=0;
    sequence=0;
    premierGN=1;
    okGN=0;
}

{
    #Pour le champs ID standard et sp (AC) on reecrit la ligne complete
    if ($1=="ID" || $1=="AC" )
    {printf("%s\n", $0);}

    #Pour le champ Nom de gene on ecrit toutes les lignes commençant par "GN"
    #sur une seule et même ligne avec comme prmeir champs GN
    if ($1=="GN")
    {
        okGN=1;
        gsub("\n", "", $0);
        if (premierGN==1)
        {
            premierGN=0;
            #s'il y a au moins une ligne DE (nom ou liste de nom de la prot.),
            #il faut s'assurer d'aller a la ligne.
            if (okDE==1) { printf("\n%s", $0);
                          okDE=0;}
            else        { printf("%s", $0);}
        }
        else
        {
            for(i=2; i<=NF; i++)
            {printf(" " $i);}
        }
    }

    if($1=="OS" || $1=="OC" || $1=="OX")
    {
        #pour aller a la ligne une fois que le traitement du nom
        #et liste de noms de la prot est termine(champs DE)
        #et/ou celui du/des nom(s) du gene.
        if (okDE==1 || okGN==1)
        {
            printf("\n%s\n", $0);
            okDE=0;
            okGN=0;
        }
        else
        {
            printf("%s\n", $0);
        }
    }

    #une fois le champs "SQ" plus "SEQUENCE" rencontres,
    #j'ecris toutes les lignes completes du fichier d'entree
    #sur une seule et même ligne:
    #c'est la sequence proteique
    if (sequence==1 && $1!="//")
    {
        printf("%s", $0);
    }

    #a ce niveau je ne recupere que la longueur de la sequence: mon nombre d'acide amines
    if ($1=="SQ" && $2=="SEQUENCE" && $4=="AA;")
    {
        printf($1"\t"$2"\t"$3"\n");
        sequence=1;
    }

    #nous sommes a la fin des informations concernant une proteine
    #je remets tous les drapeaux a leur valeur initiale
    if ($1=="//")
}
```

```

{
    printf("\n//\n");
    premierDE=1;
    sequence=0;
    okDE=0;
    premierGN=1;
    okGN=0;
}

#on ecrit toutes les lignes DE sur une seule et même ligne: plus facile a parser ensuite
if ($1=="DE" && premierDE==0)
{
    gsub("\", "", $0);
    for(i=2; i<=NF; i++)
        {printf(" "$i);}
}

if($1=="DE" && premierDE==1)
{
    gsub("\", "", $0);
    okDE=1;
    printf("%s", $0);
    premierDE=0;
}

#je recupere chacune des lignes "DR" qui m'interesse en foction du 2ieme champs
if ($1=="DR" && ($2=="EMBL;" || $2=="PIR;" || $2=="PDB;" || $2=="GO;" || $2=="InterPro;" ||
$2=="Pfam;" || $2=="PRINTS;" || $2=="ProDom;" || $2=="SMART;" || $2=="PROSITE;"))
printf("%s\n", $0);

#je recupere chacune des lignes "FT" qui m'interesse en foction du 2ieme champs
if ($1=="FT" && ($2=="SIGNAL" || $2=="CARBOHYD" || $2=="HELIX" || $2=="STRAND"))
printf("%s\n", $0);
}

```

ANNEXE 6

L'acteur MintParser.java et son action OrglMINTParser.java

L'ACTEUR : MintParser.java

```
package nom_projet.acteur.lib.interactionspp;

import ptolemy.actor.TypedAtomicActor;
import ptolemy.actor.TypedIOPort;
import ptolemy.data.expr.Parameter;
import ptolemy.kernel.CompositeEntity;
import ptolemy.kernel.util.IllegalActionException;
import ptolemy.kernel.util.NameDuplicationException;
import ptolemy.data.Token;

/**
 * @author perrine
 */
public class MINTParser extends TypedAtomicActor {

    /** ... */
    public Parameter _bdd_url;

    public Parameter _user;

    public Parameter _password;

    protected TypedIOPort _fichier_in;
    protected TypedIOPort _exit;

    private OrglMINTParser _mp;

    public MINTParser(CompositeEntity container, String name) throws
        NameDuplicationException, IllegalActionException {
        super(container, name);

        _bdd_url = new Parameter(this, "URL");
        _user = new Parameter(this, "User");
        _password = new Parameter(this, "Password");

        _fichier_in = new TypedIOPort(this, "data_in", true, false);
        _exit = new TypedIOPort(this, "exit", false, true);

        System.out.println("la on va cree un objet mp!");
        _mp = new OrglMINTParser();
        System.out.println("la on l' bien cree!!!");
    }

    public void initialize() throws IllegalActionException {
        super.initialize();
        _mp.initialize();
        System.out.println("Initialize");
    }

    public void fire() throws IllegalActionException {
        Token data;
        if (_fichier_in.hasToken(0)) {
            data = _fichier_in.get(0);
            String bdd = _bdd_url.getExpression().toString();
            String user = _user.getExpression().toString();
            String passwd = _password.getExpression().toString();
            String stdata = data.toString();

            System.out.println("\nbdd : " + bdd);
            System.out.println("user : " + user);
            System.out.println("passwd : " + passwd);
            System.out.println("data : " + stdata);
            _mp.fire(bdd, user, passwd, "/mnt/hda4/DataExp/mint/mint21full.txt");
            _exit.send(0, data);
        }
    }
}
```

l'ACTION : OrglMINTParser

```
package nom_projet.actor.lib.interactionspp;
import java.io.FileNotFoundException;
import java.io.File;
import java.io.FileReader;
import java.io.StreamTokenizer;
import java.lang.String;
import java.sql.*;

public class OrglMINTParser {
    public OrglMINTParser(){
    }

    public void initialize(){
        try
        {
            //Step 1: Load the JDBC driver.
            Class.forName("org.postgresql.Driver");
        }
        catch (Exception e)
        {
            System.err.println("Got an exception! ");
            System.err.println(e.getMessage());
        }
    }

    public void fire(String url, String usr, String pwd, String inName)
    {
        try
        {
            //Step 2: Establish the connection to the database.
            if (url.charAt(0) == '"' && url.charAt(url.length()-1) == '"')
            {
                url = url.substring(1, url.length()-1);
            }
            if (usr.charAt(0) == '"' && usr.charAt(usr.length()-1) == '"')
            {
                usr = usr.substring(1, usr.length()-1);
            }
            if (pwd.charAt(0) == '"' && pwd.charAt(pwd.length()-1) == '"')
            {
                pwd = pwd.substring(1, pwd.length()-1);
            }
            Connection conn = DriverManager.getConnection(url, usr, pwd);
            System.out.println("Connection a la base proteinDB etablie!!!!");

            //def. et ouverture des fichiers
            File inputFile = new File(inName);
            //lecteur de fichier
            FileReader readFile = new FileReader(inputFile);
            //lecteur de 'token' dans le fichier
            StreamTokenizer token = new StreamTokenizer(readFile);
            token.eolIsSignificant(true);
            token.wordChars(32,123);
            token.wordChars(125,127);
            int tokenType;
            int compteurInteraction = 0;
            String tempoProtA = new String("");
            String tempoOrgA = new String("");
            String tempoA = new String("");
            String tempoProtB = new String("");
            String tempoOrgB = new String("");
            String tempoB = new String("");
            String tempoMethode = new String("");
            String tempoClef = new String("");
            int tempoVal = 0;
            tokenType = token.ttype;
            while(token.ttype != StreamTokenizer.TT_EOL)
            {
                token.nextToken();
            }
            token.nextToken();
            while ((token.ttype) != StreamTokenizer.TT_EOF)
            {
                compteurInteraction++;
                tempoProtA = token.sval;
                token.nextToken();
                token.nextToken();
            }
        }
    }
}
```

```

tempoOrgA = token.sval;
for(int i=0; i<tempoOrgA.length(); i++)
{
    if (tempoOrgA.charAt(i)=='\')
    {
        tempoA = tempoA+"\\";
        tempoA = tempoA + tempoOrgA.substring(i, i+1);
    }
}
Statement stA = conn.createStatement();
ResultSet rsA = stA.executeQuery("select idprotein from proteins where
sp='"+tempoProtA+"' and premierTaxon='"+tempoA+"'");

int i = 0;
while (i<15)
{
    if (token.ttype == 124)
    {
        i++;
        token.nextToken();
    }
}
tempoProtB = token.sval;
token.nextToken();
token.nextToken();
tempoOrgB = token.sval;
for(int j=0; j<tempoOrgB.length(); j++)
{
    if (tempoOrgB.charAt(j)=='\')
    {
        tempoB = tempoB+"\\";
        tempoB = tempoB + tempoOrgB.substring(j, j+1);
    }
}
Statement stB = conn.createStatement();
ResultSet rsB = stB.executeQuery("select idprotein from proteins where
sp='"+tempoProtB+"' and premierTaxon='"+tempoB+"'");

int j = 0;
while (j<23)
{
    if (token.ttype == 124)
    {
        j++;
        token.nextToken();
    }
}
tempoMethode = token.sval;
while(token.ttype != StreamTokenizer.TT_EOL)
{
    token.nextToken();
    if (rsA.next() && rsB.next())
    {
        Statement stUP = conn.createStatement();
        int rowCount = stUP.executeUpdate("insert into interactions
values ('MINT_"+compteurInteraction+"', 'MINT', '"+tempoMethode+"', "+rsA.getInt(1)+", "+rsB.getInt
(1)+")");

        stUP.close();
    }
}
token.nextToken();
tempoA = "";
tempoB = "";
rsA.close();
stA.close();
rsB.close();
stB.close();
}
conn.close();
}
catch (FileNotFoundException e)
{
    System.err.println("fichier "+ inName + " non trouve : " + e);
}
catch (Exception e)
{
    System.err.println("Got an exception! ");
    System.err.println(e.getMessage());
}
}
}

```

ANNEXE 7

JDBCtest1.java : Programme de test d'utilisation de la connexion JDBC

```
import java.sql.*;

public class JDBCtest1 {
    public static void main (String[] args) {
        try {
            // Step 1: Load the JDBC driver.
            Class.forName("org.postgresql.Driver");

            // Step 2: Establish the connection to the database.
            String url = "jdbc:postgresql:proteindb";
            Connection conn = DriverManager.getConnection(url,"postgres","bhul2");

            //requête d'interrogation de ma table proteindb:
            //je cherche l'enregistrement de cette table pour
            //lequel idstandard vaut IL2_HUMAN
            // et j'affiche les deux premier champs de cet enregistrement
            Statement st = conn.createStatement();
            ResultSet rs = st.executeQuery("select * from proteindb where
idstandard='IL2_HUMAN'");
            while(rs.next()){
                System.out.println(rs.getString(1)+ "\n");
                System.out.println(rs.getString(2)+ "\n");
            }
            rs.close();
            st.close();

            //requête de mise a jour, je remplace tous les champs
            //idstandard qui valent 'IL2_HUMAN' par 'ILdeux_HUMAN'
            Statement stUP = conn.createStatement();
            int rowCount = stUP.executeUpdate("update proteindb set
idstandard='ILdeux_HUMAN' where idstandard='IL2_HUMAN'");
            System.out.println(rowCount);
            stUP.close();
            System.out.println("reussie!!!\n");
        }
        catch (Exception e) {
            System.err.println("Got an exception! ");
            System.err.println(e.getMessage());
        }
    }
}
```

Annexe 8 : exemple de workflow

