AskOmics : Intégration et interrogation de réseaux de régulation génomique et post-génomique

Charles Bettembourg, Olivier Dameron, Anthony Bretaudeau, Fabrice Legeai

AskOmics : Intégration et interrogation de réseaux de régulation génomique et post-génomique

Charles Bettembourg[1,2], Olivier Dameron[3,1,2], Anthony Bretaudeau[4,5], Fabrice Legeai[4,6]

[1] CNRS, UMR IRISA, RENNES, FRANCE
[2] INRIA, UMR IRISA, DYLISS, RENNES, FRANCE
[3] UNIVERSITÉ DE RENNES 1, RENNES, FRANCE
[4] INRA, UMR IGEPP, BIPAA, RENNES, FRANCE
[5] INRIA, UMR IRISA, GENOUEST CORE FACILITY, RENNES, FRANCE
[6] INRIA, UMR IRISA, GENCALE, RENNES, FRANCE
charles.bettembourg@irisa.fr olivier.dameron@univ-rennes1.fr
anthony.bretaudeau@rennes.inra.fr fabrice.legeai@rennes.inra.fr

Résumé : Les études génomiques et post-génomiques génèrent de nombreuses données d'expression de plusieurs types d'ARN (messagers, micro, pi et lnc) ainsi que des informations sur la présence de marques épigénétiques (méthylation des histones, méthylation de l'ADN, accessibilité). L'identification des interactions entre ces différents acteurs biologiques par les biologistes à partir des fichiers tabulés contenant les données brutes est une tâche longue et difficile. AskOmics est un nouvel outil générique en cours de développement capable d'intégrer les données contenues dans des fichiers tabulés dans une base de données RDF, et de permettre aux biologistes de construire visuellement leurs requêtes sans avoir besoin de connaître la structure de la base de données ni de maitriser un langage de requêtes.

Mots-clés : RDF, post-génomique, données omiques, intégration, interrogation.

1 Introduction

L'étude des mécanismes de régulations génomiques et post-génomiques génère de nombreuses données omiques. Ces données concernent notamment des niveaux d'expression de plusieurs types d'ARN et des marques épigénétiques (Barrett *et al.*, 2013). L'interaction entre ces différents ARN et la présence de ces marques modulent l'expression des gènes (Gurtan & Sharp, 2013; Chang, 2013; Watanabe & Lin, 2014; Luco *et al.*, 2011). Un travail d'intégration est nécessaire pour identifier et analyser des associations entre ces jeux de données (Kim *et al.*, 2010).

L'Institut de Génétique, Environnement et Protection des Plantes (IGEPP, INRA, Rennes) s'est intéressé à la description des mécanismes de régulations géniques lors de l'embryogenèse chez le puceron du pois. Dans le cadre du projet ANR MiRNAdapt, l'IGEPP a généré de nombreuses données omiques à différents stades de l'embryogénèse du puceron. Ces données concernent des niveaux d'expression d'ARN messagers, de micro-ARN, de piARN et de longs ARN non codants d'une part et de marques épigénétiques (méthylation des histones, méthylation de l'ADN, accessibilité) d'autre part.

Cet exemple de données générées par des études post-génomiques pose le double problème de l'intégration des données et de leur analyse fine nécessitant leur interrogation par des biologistes n'ayant pas de connaissance dans un langage de requêtes. Les données omiques concernent de nombreux acteurs en interaction et doivent pouvoir être interrogées et analysées ensemble même si elles sont obtenues séparément (Gomez-Cabrero *et al.*, 2014). Il s'agit d'un problème

récurrent en bioinformatique, pour lequel nous proposons l'intégration des données dans une base de données RDF. L'utilisation d'un modèle RDF permet de relier facilement les différents agents d'une régulation génique (Jupp *et al.*, 2014). Ce modèle développé par le World Wide Web Consortium (W3C) permet l'implémentation simple d'une base de données par l'utilisation de triplets "sujet - prédicat - objet". Cela diffère du modèle relationnel classique dans lequel les données sont dépendantes d'une structure composée de nombreux champs de taille et contenus variables. L'implémentation, la maintenance et l'extensibilité d'une base de données RDF est ainsi moins contraignante que pour une base de données classique. La conversion de fichiers tabulés contenant les données générées par des biologistes vers une des syntaxes du modèle RDF est également aisée. De plus, toutes les bases de données RDF reposant sur la même structure composée de triplets, elles offrent une excellente inter-opérabilité, permettant de combiner ou de comparer des données locales avec des données et des connaissances issues de bases publiques.

L'utilisateur final d'une base de données omiques est soit biologiste, soit bioinformaticien. Mais l'interrogation d'une base de données RDF se fait via des requêtes SPARQL, qui n'est pas un langage couramment maîtrisé par les bioinformaticiens et moins encore par les biologistes. Nous présentons dans cet article AskOmics, un nouvel outil permettant d'intégrer des données à partir de fichiers tabulés ou structurés dans une base de données RDF et d'interroger cette base de données en construisant visuellement des requêtes sans nécessiter la maîtrise de SPARQL. AskOmics est actuellement en développement et utilise comme données de test le réseau de régulations ARN du projet MiRNAdapt. Néanmoins, le problème auquel il répond est général et notre mise en oeuvre vise la généricité.

2 AskOmics

AskOmics est conçu selon un modèle multi-couches séparant les données, la maintenance (incluant l'insertion et la suppression de données), le module de construction des requêtes et l'affichage du graphe de requête et des résultats. Cette modularité facilité la conception, le développement, l'évolution et le déploiement de l'application. Chaque partie peut être modifiée séparément et de nouveaux modules (par exemple un nouveau visualisateur) peuvent être aisément intégrés. La figure 1 détaille l'architecture d'AskOmics. AskOmics étant encore en développement, tous ses modules ne sont pas finalisés au moment de la rédaction de cet article.

2.1 Génération de la base et maintenance

La couche de maintenance permet de générer la base de données RDF via le module *Data addition*. Les fichiers tabulés initiaux sont convertis au format turtle lors de cette opération. Chaque type d'entité biologique décrit par dans les têtes de colonne des fichiers initiaux devient ainsi une classe dans la base de données, dont les instances sont les données correspondantes. Ce module *Data addition* sert ensuite à ajouter de nouvelles données dans la base. Les opérations de suppression de données sont assurées par le module *Data removal*. Le module *Statistics* permet de calculer le nombre de triplets total et par classe dans la base de données. Les modules *Domain knowledge generator* et *Database abstraction generator* écrivent chacun un fichier turtle utilisé dans la couche métier et décrits dans la section suivante.

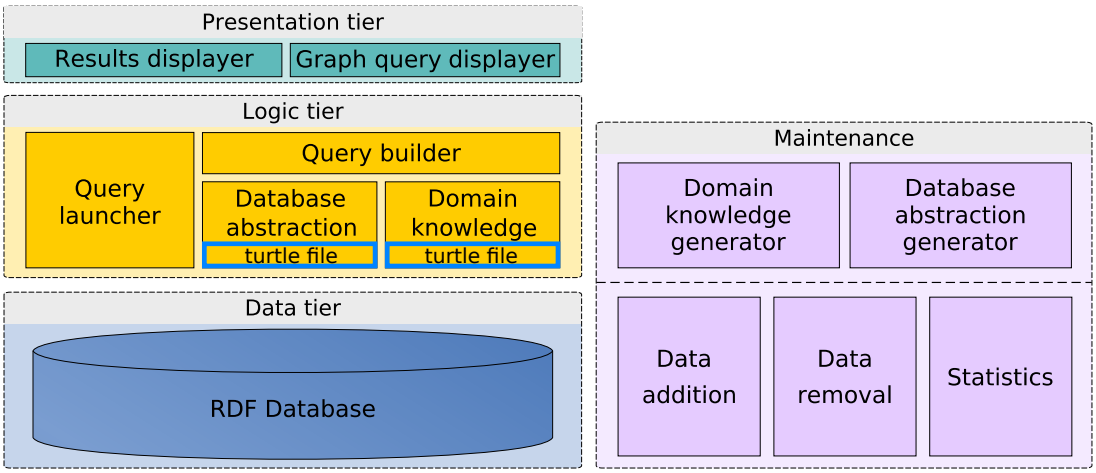

FIGURE 1 – Architecture d'AskOmics. La couche de données est générée en utilisant le module d'addition de données de la couche de maintenance. La couche logique permet de procéder à la construction de requêtes visualisables sous forme de graphe dans la couche de présentation. Lorsque l'utilisateur a terminé la construction de sa requête, son graphe est converti en SPARQL par le *Query launcher* de la couche logique et envoyée à la base de données. La couche de présentation affiche alors les résultats.

2.2 Couche logique

Le module *Query builder* se base sur deux fichiers turtle *Domain knowledge* et *Database abstraction* et échange avec la couche de visualisation pour construire la requête de l'utilisateur. Il commence par envoyer à la couche de présentation les classes marquées comme points de départs possibles pour la construction de requêtes. Ce marquage est défini dans le fichier turtle *Domain knowledge* généré par le module *Domain knowledge generator* de la couche de maintenance. Pour le moment, c'est l'utilisateur qui paramétre ce générateur. Il est prévu d'ajouter ultérieurement un module capable d'enregistrer les requêtes préférées des biologistes et d'adapter en fonction le fichier de *Domain knowledge* généré. Ce fichier contient actuellement 3 types d'informations :

— Les classes pouvant servir de point de départ pour les requêtes.
— Des raccourcis entre deux classes pour accélérer la construction des requêtes grâce à l'utilisation des *property paths* de SPARQL 1.1. Cela permet de formuler plus rapidement une question fréquemment posée.
— Des classes que l'on souhaite masquer à l'utilisateur afin de simplifier la manipulation des données.

Voici le contenu d'un fichier *domain knowledge* :

```
@prefix : <http://www.irisa.fr/dyliss/base/> .
@prefix d: <http://www.irisa.fr/dyliss/display/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

:microRNA d:startPoint "true"^^xsd:boolean .
```

```
:messengerRNA d:startPoint "true"^^xsd:boolean .
:lncRNA d:startPoint "true"^^xsd:boolean .
:piRNA d:startPoint "true"^^xsd:boolean .

:microRNA d:shortcut ":studied_in/:has_results/:trend" .
:messengerRNA d:shortcut ":studied_in/:has_results/:trend" .
:lncRNA d:shortcut ":studied_in/:has_results/:trend" .
:piRNA d:shortcut ":studied_in/:has_results/:trend" .

:study d:hidden "true"^^xsd:boolean .
:experimentsComparison d:hidden "true"^^xsd:boolean .
```

Il permet de définir :

— 4 types d'ARN comme points de départ possibles pour les requêtes grâce à la relation *startPoint*.

— 4 raccourcis via des *property paths* grâce à la relation *shortcut*.

— 2 classes masquées pour l'utilisateur via la relation *hidden*.

À partir d'un point de départ, le *Query builder* construit la requête en fonction des choix de l'utilisateur. Il interroge le fichier *Domain knowledge* pour proposer des raccourcis à l'utilisateur et pour savoir quelles classes masquer. Il interroge aussi le fichier turtle *Database abstraction* généré par le module *Database abstraction generator* de la couche de maintenance. Ce fichier permet de trouver les classes en relation avec la classe sélectionnée par l'utilisateur afin de les lui proposer pour construire sa requête. L'utilisation d'un fichier d'abstraction est beaucoup plus rapide pour cette étape que l'interrogation directe de la base de données. En voici un extrait pour les données de MiRNAdapt, définissant la classe `microRNA` et ses relations :

```
@prefix : <http://www.irisa.fr/dyliss/base/> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

:microRNA rdf:type owl:Class ;
          rdfs:label "microRNA" ;
          rdfs:subClassOf [ rdf:type owl:Restriction ;
                            owl:onProperty rdfs:label ;
                            owl:someValuesFrom xsd:string
                          ] ,
                          [ rdf:type owl:Restriction ;
                            owl:onProperty :part_of_pairing ;
                            owl:someValuesFrom :pairing
                          ] ,
                          [ rdf:type owl:Restriction ;
                            owl:onProperty :sequence ;
                            owl:someValuesFrom xsd:string
                          ] ,
                          [ rdf:type owl:Restriction ;
```

```
    owl:onProperty :has_target ;
    owl:someValuesFrom :messengerRNA
  ] ,
  [ rdf:type owl:Restriction ;
    owl:onProperty :studied_in ;
    owl:someValuesFrom :study
  ] .
```

Quand l'utilisateur a terminé la construction de sa requête, le module *Query launcher* se charge de la convertir en SPARQL, l'envoie à la base de données RDF et envoie les réponses de celle-ci au module *Results displayer* de la couche de présentation. La communication entre la couche de présentation et la couche logique est assurée par un *framework* pyramid.

2.3 Couche de présentation

La couche de présentation consiste en une interface web utilisant la librairie de représentation de graphes d3.js. Dans le *Graph query displayer*, les classes de la base de données sont représentées par des nœuds et les relations entre ces classes par des arêtes reliant les nœuds. L'utilisateur commence par choisir un des points de départs obtenus en interrogeant le *Query builder* de la couche logique. Ce point de départ forme le premier nœud du graphe. On peut lancer des expansions à partir de ce point de départ, c'est à dire demander au *Query builder* de trouver les classes en relation avec ce nœud pour les proposer à l'utilisateur. Le graphe se construit ainsi visuellement par expansions successives. Une fois l'utilisateur satisfait de son graphe de requête, il peut la lancer, ce qui met à contribution le *Query launcher*. Le module *Results displayer* met en forme et affiche les réponses obtenues par le *Query launcher*.

3 Application

Nous avons implémenté une première version d'AskOmics en utilisant les données d'expression d'ARN messagers, de micro-ARN, de piARN et de longs ARN non codants obtenues par le projet MiRNAdapt. Les propriétés et relations des différents ARN peuvent être représentés par le modèle proposé par la Figure 2. Actuellement, la base contient 44 688 228 triplets.

La Figure 3 illustre l'état actuel du développement d'un outil d'interrogation de la base de données RDF. Dans cet exemple, l'utilisateur a choisi comme point de départ la classe `messengerRNA`. Parmi les classes en relation avec ce premier nœud obtenues grâce au *Query builder*, l'utilisateur a choisi d'ajouter `microRNA`, `lncRNA` et `trend` à son graphe de requête. La classe `trend` est ici proposée en suivant un raccourci grâce à la propriété *shortcut* du *Domain knowledge*. En effet, cette classe `trend` est situé dans la boîte `Difference` de la Figure 2 et n'est donc pas directement liée à `messengerRNA`. Un raccourci suivant `studied in` et `has differential results` pour arriver à `trend` permet de construire plus rapidement cette partie de la requête qui correspond à une demande fréquente de la part des biologistes. L'utilisateur a ensuite choisi de procéder à l'expansion des nœuds `lncRNA` et `microRNA`, et a ajouté pour chacun de ces nœuds la classe `trend`.

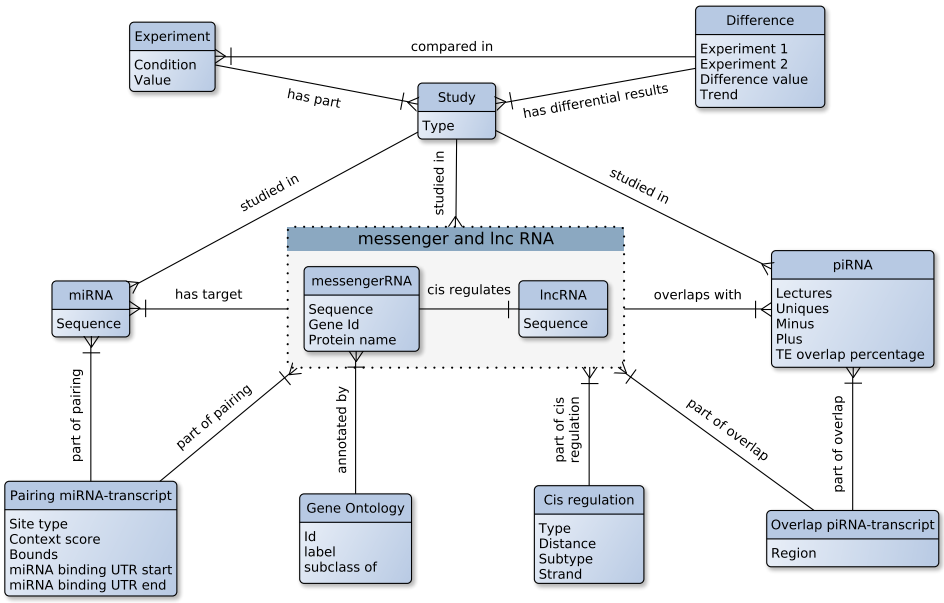

FIGURE 2 – Structure de la base de données RDF d'après les proriétés et relations des classes qu'elle contient.

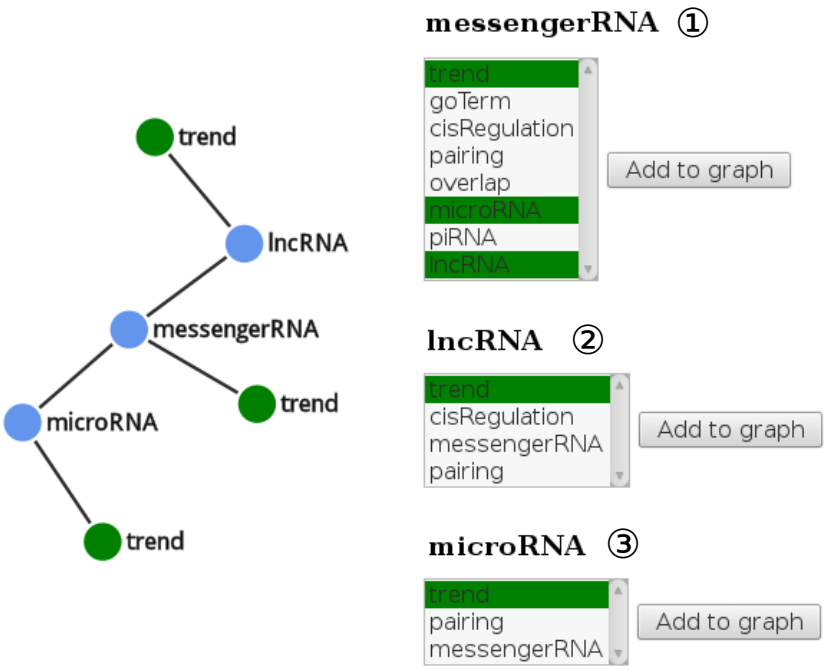

FIGURE 3 – Exemple de construction graphique de requête. On recherche les ARN messagers à la fois ciblés par un micro-ARN et cis-régulés par un long ARN non codant, chacun de ces ARN ayant montré une tendance différentielle au niveau des expériences dans lesquelles ils ont été étudiés.

Si l'utilisateur choisit de lancer la requête à ce stade, elle sera ainsi convertie en SPARQL :

```
PREFIX : <http://www.irisa.fr/dyliss/base/>
SELECT DISTINCT *
WHERE {?messengerRNA1 a :messengerRNA .
       ?lncRNA1 a :lncRNA .
       ?lncRNA1 :cis_regulates ?messengerRNA1 .
       ?microRNA1 a :microRNA .
       ?microRNA1 :has_target ?messengerRNA1 .
       ?messengerRNA1 :studied_in/:has_results/:trend ?trend1 .
       ?lncRNA1 :studied_in/:has_results/:trend ?trend2 .
       ?microRNA1 :studied_in/:has_results/:trend ?trend3 .
       }
```

Le *Query launcher* envoie cette requête à la base de données et en fait suivre les résultats au *Result displayer* qui les met en forme pour l'utilisateur.

4 Conclusion

AskOmics apporte une solution simple d'utilisation aux biologistes soucieux d'exploiter efficacement les données obtenues en génomique et post-génomique. Ce nouvel outil est générique ; il est capable de s'adapter pour l'intégration des données de nombreux types d'études biologiques différentes. Dans le contexte d'avalanches de données en biologies, il apporte une réponse à une forte demande de la communauté.

Références

BARRETT L. W., FLETCHER S. & WILTON S. D. (2013). *Untranslated Gene Regions and Other Non-coding Elements : Regulation of Eukaryotic Gene Expression*. Springer Science & Business Media.

CHANG H. Y. (2013). Genome regulation by long non-coding rnas. *Blood*, **122**(21), SCI–29.

GOMEZ-CABRERO D., ABUGESSAISA I., MAIER D., TESCHENDORFF A. E., MERKENSCHLAGER M., GISEL A., BALLESTAR E., BONGCAM-RUDLOFF E., CONESA A. & TEGNÉR J. (2014). Data integration in the era of omics : current and future challenges. *BMC Systems Biology*, **8**(S-2), I1.

GURTAN A. M. & SHARP P. A. (2013). The role of mirnas in regulating gene expression networks. *Journal of molecular biology*, **425**(19), 3582–3600.

JUPP S., MALONE J., BOLLEMAN J., BRANDIZI M., DAVIES M., GARCIA L. J., GAULTON A., GEHANT S., LAIBE C., REDASCHI N., WIMALARATNE S. M., MARTIN M. J., NOVÈRE N. L., PARKINSON H. E., BIRNEY E. & JENKINSON A. M. (2014). The ebi rdf platform : linked open data for the life sciences. *Bioinformatics*, **30**(9), 1338–1339.

KIM T. Y., KIM H. U. & LEE S. Y. (2010). Data integration and analysis of biological networks. *Current opinion in biotechnology*, **21**(1), 78–84.

LUCO R. F., ALLO M., SCHOR I. E., KORNBLIHTT A. R. & MISTELI T. (2011). Epigenetics in alternative pre-mrna splicing. *Cell*, **144**(1), 16–26.

WATANABE T. & LIN H. (2014). Posttranscriptional regulation of gene expression by piwi proteins and pirnas. *Molecular cell*, **56**(1), 18–27.