MetaPenta: visualización y análisis de redes metabólicas

Valerie Parra Cortés
Asesor: Jorge Alexander Duitama Castellanos

1. Resumen

Se han propuesto diversos acercamientos para abordar el problema de modelar, simular y representar redes metabólicas, los cuales se pueden resumir en tres principales modelos: continuos, discretos e híbridos. En este trabajo se propuso modelar redes metabólicas con redes de Petri. Una red de Petri es una representación de eventos discretos en forma de grafo que cumple propiedades de distribución, concurrencia y paralelismo. Se implementó MetaPenta, una herramienta que es capaz de realizar diferentes consultas sobre las redes metabólicas, por ejemplo, encontrar el camino mínimo entre dos metabolitos, identificar metabolitos claves como sumideros y fuentes, visualizar la red en forma de una red de Petri, entre otras funcionalidades. Se comparó la herramienta implementada con otros desarrollos de software como los *scripts* del grupo de investigación de Costas Maranas y la herramienta Fluxer. Se encontró que para el problema de *gap find* las salidas de los programas son muy parecidas, a pesar de que las aproximaciones para encontrar estos metabolitos son muy diferentes. Por otro lado, en términos de representación, Fluxer tiene varios tipos de gráficos para este propósito. Sin embargo, en la mayoría de las ocasiones dichas representaciones omiten gran parte de la información (por ejemplo, las enzimas de la reacción), mientras que la representación en red de Petri implementada en MetaPenta no la omite. A futuro se espera incrementar las funcionalidades de MetaPenta para manejar la reversibilidad de las reacciones y facilitar la reconstrucción automática de redes metabólicas.

Palabras clave: Redes metabólicas, bioinformática, redes de Petri.

2. Introducción

El conjunto de todas las reacciones que ocurren en una célula se denomina red metabólica. Una red metabólica se compone de tres elementos principales: metabolitos, enzimas y reacciones. Los metabolitos son las sustancias que se producen y se consumen en el interior de la célula. Este consumo y producción se da a través de las reacciones. En una reacción se rompen y se forman enlaces, lo que da lugar a nuevos compuestos que se denominan productos. Sin embargo, romper y formar enlaces son procesos que pueden requerir o liberar energía [1]. Generalmente se necesita una energía inicial que active la reacción, la cual puede ser muy alta y no encontrarse naturalmente en el ambiente. Debido a esto, la célula sintetiza proteínas denominadas enzimas, las cuales actúan como catalizadores para lograr que las reacciones ocurran. Las

enzimas reducen la energía de activación de la reacción de manera que esta se pueda iniciar [1]. Igualmente, una enzima también puede inhibir una reacción cuando hay desequilibrio de algún metabolito. Las enzimas son, en todo sentido, un regulador biológico.

Fig. 1: Muchos productos de consumo cotidiano son en realidad bioproductos. Entre estos está la cerveza, el *yogurt*, el pan y el chocolate. En todos los procesos anteriores se utilizan microorganismos para realizar fermentaciones. Todas las imágenes son tomadas de internet.

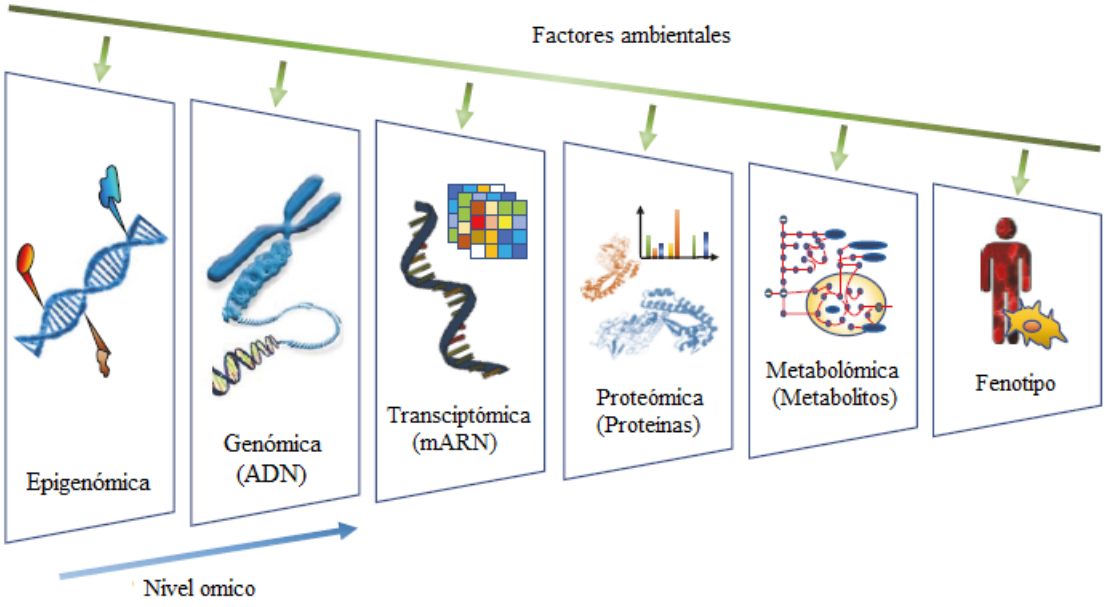


Fig. 2: Relación entre las distintas ómicas. Imagen tomada y modificada de [2]

Si se tiene en cuenta que una gran variedad de los productos que se consumen hoy en día se producen en ambientes biológicamente activos (Figure 1), conocer y entender a fondo las rutas metabólicas descritas por estas redes tiene aplicación en la industria química. Por ejemplo, en la elaboración de productos como el *yogurt*, la cerveza y el bioetanol se produce alcohol a través de la fermentación de azúcares. Por otro lado, algunos suplementos y vitaminas también son bioproductos ya que utilizan bacterias para su producción. En este contexto, visualizar y analizar redes metabólicas es una herramienta para entender a profundidad estos procesos. A este problema bioinformático se le conoce como **modelaje de redes metabólicas**.

Actualmente en la construcción de un modelo de red metabólica se pueden generar al menos dos tipos de inconsistencias. La primera, denominada *root no production* es aquella donde aparecen metabolitos que no son producidos por una reacción de la red y sin embargo aparecen en el modelo. La segunda inconsistencia, denominada *root no consumption*, consiste en la presencia de metabolitos que no son consumidos por ninguna reacción. Como se asume que la célula está en estado estacionario, hay una reacción faltante donde se consume este metabólito o debe ser expulsado por una reacción extracelular. Estos dos tipos de metabolitos marcan lo que se denomina la frontera de la red [3]. La representación gráfica de este problema puede verse en la Figura 4. Otro análisis que es de interés en metabolómica es el de encontrar una ruta mínima para producir un metabolito. Esto es de particular interés en ingeniería metabólica, ya que en esta rama de la ingeniería se buscan y optimizan rutas metabólicas [3]. Otro gran reto en las redes metabólicas es lograr una correcta visualización. La visualización más común es el mapa de Escher (Figura 5a), aunque se han intentado otras representaciones como grafos de reacciones (Figura 5b). Por último, un análisis que consideramos importante, aunque no haya mucho en el estado del arte, es el de encontrar componentes conectadas de la red metabólica, esto puede ayudar sobre todo en modelos *de novo* altamente desacoplados cuyos análisis metabólicos basados en programación lineal no convergen.

En el campo médico las rutas metabólicas de un organismo ayudan a predecir biomarcadores [2]. Los biomarcadores son sustancias que se cuantifican y se usan para determinar el estado normal de un proceso biológico. Por otro lado, conocer la red metabólica también ayuda a entender los mecanismos biológicos que determinan la aparición de diferentes enfermedades como la diabetes, el cáncer y algunas enfermedades neurodegenerativas [2]. En la mitad del proceso de expresión de la información genética en fenotipos observables se encuentran las redes metabólicas. Por lo tanto, entender a fondo estas redes nos ayuda a dilucidar la relación entre genotipo y fenotipo [2] (ver Fig. 2).

Se han realizado varios acercamientos computacionales para modelar las redes metabólicas: los modelos continuos, discretos e híbridos. En los modelos continuos, se modela la red como una ecuación global que se denomina *análisis de balance de flujo* [4]. El principal problema de este método es que no se modelan los datos de la dinámica molecular y se hace la suposición de que el sistema se encuentra en estado estacionario. Para intentar corregir estas falencias se han utilizado modelos discretos basados en lenguajes formales [3]. Entre los modelos discretos más famosos para las redes metabólicas están los grafos. Sin embargo, esta representación tiene sus limitantes ya que, no deja incorporar el uso de las enzimas, ni el hecho de que se necesitan todos los reactivos en cierta proporción estequiométrica para que se dé una reacción. Una segunda aproximación para estos modelos discretos es tratar las redes como un grafo bipartito [5] entre reacciones

y metabolitos. Por último, se han propuesto modelos híbridos que intentan combinar modelos discretos y continuos.

Dentro de los modelos discretos se ha propuesto modelar la red metabólica como una red de Petri [6]. Una Red de Petri es un formalismo orientado a grafos que permite modelar sistemas con propiedades de concurrencia, paralelismo y distribución. Está constituida por lugares y transiciones. De manera gráfica los lugares se representan con un círculo, mientras que una transición se representa con un rectángulo [7]. Los lugares contienen *tokens*, los cuales se utilizan para activar transiciones. Dichas cantidades se encuentran en el peso de los ejes (Figura 3), si todos los lugares que llevan a una transición tienen una cantidad de *tokens* igual o mayor a la indicada en el eje, la transición puede activarse.

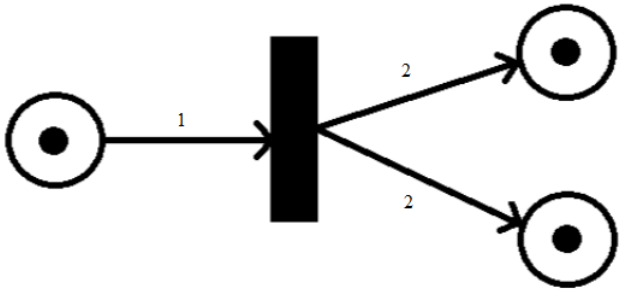


Fig. 3: Representación de una red de Petri. Los círculos son los *places* o lugares y son los que contienen los tokens que activan las *transitions* o transiciones que en la figura están representadas con rectángulos

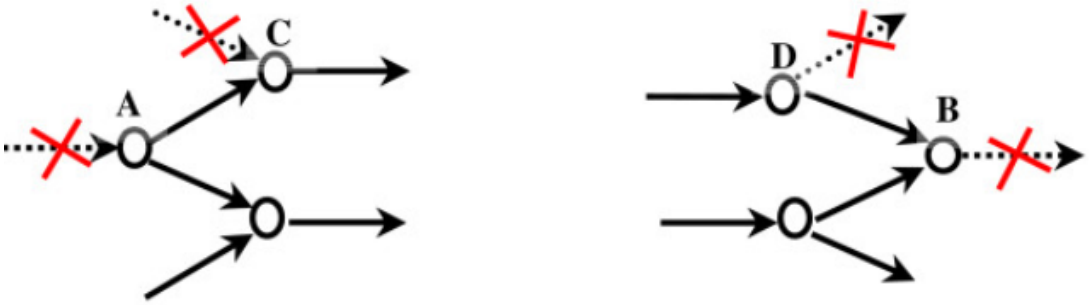
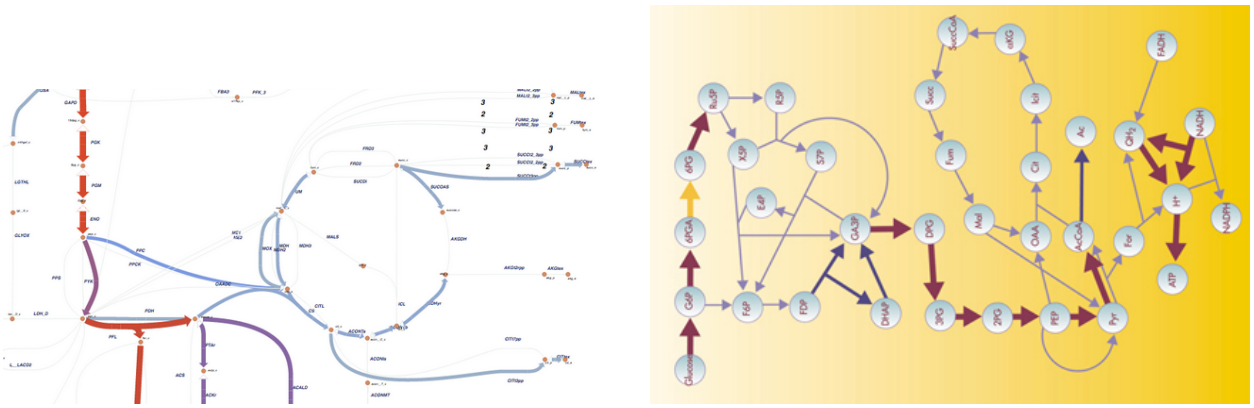


Fig. 4: Problemas de *root no production* y *root no consumption* [8]

La ventaja principal que tiene esta representación sobre una representación de grafos es que permite modelar completamente el concepto de reacción metabólica como una transición. Además de esto, hay una profunda teoría sobre las redes de Petri, incluidas poderosas herramientas para su simulación [9, 10] Finalmente, las redes de Petri permiten integrar información continua para construir modelos híbridos [7].



(a) Mapa de Escher

(b) Grafo de una red metabólica

Fig. 5: Diferentes representaciones de una red metabólica. En la figura A vemos un mapa de Escher, una de las representaciones más comunes y utilizadas para redes metabólicas. Esta imagen fue tomada y modificada de *Escher library* Por otro lado, en la Figura B tenemos una representación de grafo que se le puede hacer a una red metabólica. Imagen tomada y modificada de *Wikipedia*

A pesar de las ventajas de este modelo, no hay muchas implementaciones con este formalismo de redes metabólicas. Aunque hay una variedad de software y páginas web que permiten hacer análisis sobre las redes metabólicas, muchos ya no cuentan con mantenimiento. Algunas de las herramientas más útiles para trabajar con las redes metabólicas se enumeran a continuación:

1. **RAVEN**: Una de las librerías más conocidas para la reconstrucción, curación y análisis de redes metabólicas es RAVEN. Esta librería de MatLab cuenta con dos versiones y su último artículo publicado, el de la versión 2.0, tiene más de 25 citaciones. Está librería cuenta con mantenimiento y está siendo actualizada constantemente [11].

2. **Network Analysis Tools (NeAT)**:Esta es una de las herramientas más completas para hacer análisis sobre redes metabólicas. Permite encontrar rutas metabólicas, sacar estadísticas de la red, curvas de sensitividad/especificidad (ROC) entre otras funciones [12]. Su principal desventaja es que las funciones relacionadas con la visualización de las redes pueden ser algo lentas y tiene algunos enlaces caídos. Fue lanzada en el año 2008 y es constantemente renovada. El artículo de su lanzamiento tiene 37 citas [13].

3. **WEbInterViewer**: Es un software (aunque también cuentan con página web) que permite visualizar redes metabólicas como un gráfico por capas. Este software es pago, fue lanzado en 2004 y sólo está disponible para Windows. El artículo de su lanzamiento tiene 5 citaciones externas y esta herramienta es renovada constantemente.

4. **GraphWeb**: Es una página web que fue lanzada en 2007, es gratis y el artículo de su lanzamiento tiene 41 citas. Su última actualización fue en el año 2008 por lo que no tiene mantenimiento. Su principal función es el descubrimiento de módulos a través de filtros topológicos [14].

5. **Fluxer**: Fluxer es una herramienta que permite a los usuarios visualizar redes metabólicas, encontrar árboles de expansión mínima y rutas metabólicas [15]. Se encuentra disponible en versión Web. En particular se comparó su funcionalidad de encontrar caminos mínimos con la implementada en MetaPenta..

6. **Scripts del grupo Costas Maranas**: Costas Maranas es un grupo que hace scripts en GAMS para hacer diferentes análisis sobre redes metabólicas [8], incluidos análisis de flujo, *gap find*, *gap fill*, función de biomasa, entre otras.

3. MetaPenta: Una nueva herramienta para análisis de redes metabólicas

Dado el estado actual de las herramientas para análisis de redes metabólicas, en este trabajo desarrollamos una nueva solución de software para realizar diferentes análisis de redes metabólicas. Esta sección describe las funcionalidades implementadas, el diseño de la solución y la interfaz gráfica.

3.1. Funcionalidades

Ruta mínima

Para determinar un camino mínimo en una red de Petri se mantuvo la idea de activación de una transición (sólo puede ser activada si se cuenta con todos los recursos de los que depende), el cual a su vez es el mismo comportamiento biológico esperado. Así mismo, fue necesario almacenar la distancia desde el nodo origen, la existencia del metabolito y la reacción por la que se llegó. Se llevó una cola de prioridad con la distancia mínima y se iba recorriendo la red en este orden. Cada vez que se explora un metabolito se mira cuáles son las reacciones que se pueden activar y se van añadiendo los metabolitos que se producen a la cola de prioridad para posteriormente ser analizados. Al final el programa pinta la sub-red asociada a la mínima ruta encontrada.

Metabolitos clave

Con el fin de identificar los metabolitos fuente y sumidero, se procedió a buscar aquellos metabolitos o lugares donde no hay ejes de entrada o salida. Aquellos lugares que en la red de Petri no tienen ningún eje de entrada son los metabolitos que se consumen, pero nadie los está produciendo, estos metabolitos representan un error y deben ser curados. Así mismo los lugares que en la red de Petri no tienen ningún eje de salida son aquellos metabolitos que se producen en la célula pero no son consumidos en ninguna actividad celular y deben ser o eliminados o curados según corresponda (*root no production* y *root no consumption*).

Metabolitos en común

Para identificar los metabolitos en común entre dos redes metabólicas se realizó una búsqueda lineal entre la lista de metabolitos de ambas redes. Para la identificación de las reacciones donde participa un metabolito hay que encontrar el lugar asociado al metabolito y mirar los ejes de entrada y salida que son los que lo conectan con las reacciones donde se produce y donde se consume respectivamente.

Componentes débilmente conectadas

Por último, para encontrar las componentes débilmente conectadas se empieza en un lugar de la red de Petri al azar, se selecciona número que representará el ID de la componente conexa. Se recorrió el grafo haciendo BFS desde dicho nodo y se fueron escribiendo los metabolitos con el ID correspondiente a la componente conexa. Cuando se termina este procedimiento, se selecciona un nodo que no haya sido clasificado, se actualiza el ID y se repite el procedimiento.

Generación de archivos en diversos formatos para explorar la red en otras herramientas

Para darle flexibilidad al usuario en cuanto a las herramientas, MetaPenta ofrece la opción de exportar todo a distintos formatos estándar para que pueda ser visualizado o manejado en otras herramientas. Por ejemplo, MetaPenta convierte el formato XML a un formato CSV que pueda ser visualizado en CytoScape. Además genera todos los archivos de entrada a GAMS para *gap Find* [16].

3.2. Arquitectura de la aplicación

Toda la aplicación está implementada en Java, además cuenta con tres paquetes: la red metabólica, la red de Petri y la interfaz gráfica. Esta última a su vez está dividida en dos: la visualización de la red de Petri y la de los paneles. El primer paquete, la red metabólica, tiene toda la lógica de la aplicación, este paquete cuenta con un cargador, para leer los archivos XML, y todas las clases asociadas a la red metabólica (Figura 6, clases de color rosado). Todo el procesamiento se hace en la clase *MetabolicNetwork* ya que es la que conoce tanto el modelo metabólico como la red de Petri, de la cual se vale para ejecutar todas las funcionalidades (sección 3.1).

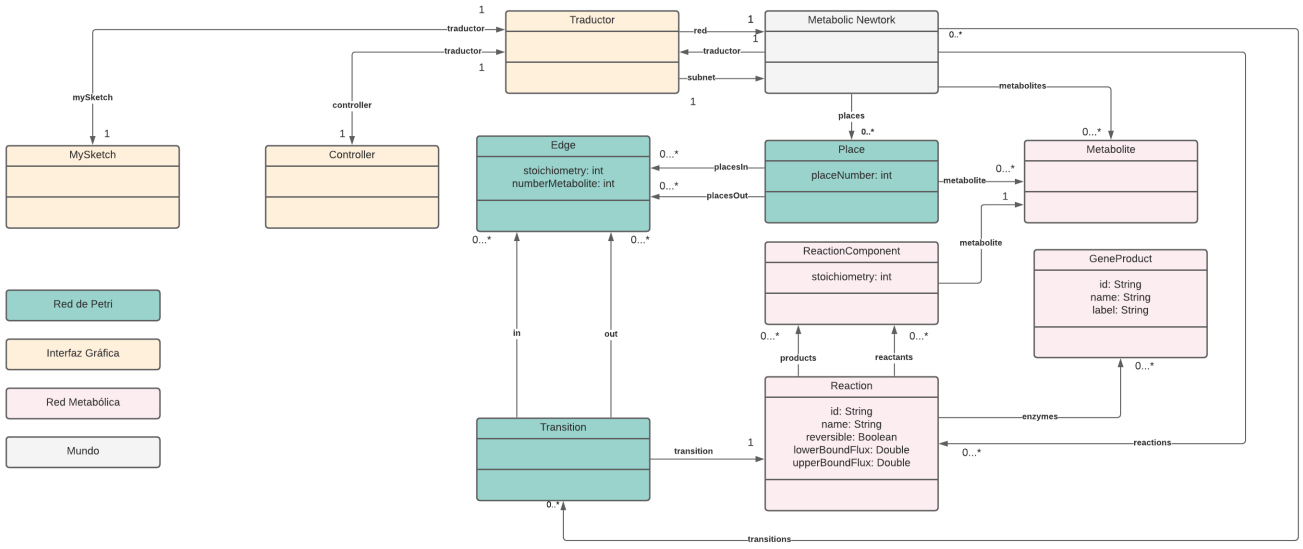


Fig. 6: Clases principales de la aplicación. En verde se muestra todo lo relacionado con la red de Petri, en amarillo todo lo encargado de realizar el manejo de eventos. En rosado, el modelo de la red metabólica. Por último, el mundo y su lógica están contenidos en la clase *Metabolic Network* que está en gris.

Por otro lado, el segundo paquete, el que contiene la red de Petri, consta de tres clases principales: *Edge*, *Place* y *Transitions*. La clase *Place* representa los metabolitos de la red metabólica (para detalle de una red de Petri ver Figura 3), mientras que la clase *Transitions* representa las reacciones. Esta parte del programa sólo estructura los datos en la red ya que, como se mencionó antes, el modelo del mundo es el que contiene

toda la lógica de la aplicación.

El último paquete que tenemos es el de la visualización. En este paquete tenemos una clase traductor que es la que convierte el modelo de la red de Petri que es retornado por el mundo, al modelo de nodos que maneja Processing, por lo que existe una relación fuerte entre las clases Traductor, MetabolicNetwork y Controller (Figura 6).

3.3. Tecnologías

Las tecnologías utilizadas por la aplicación unto con su respectiva versión se numeran a continuación:

1. Java 11.0.7
2. Processing 2.0
3. JavaFX 15

La visualización de la red de Petri está apoyada en Processing 2.0, una librería de Java que además tiene un lenguaje independiente que permite realizar visualizaciones fácilmente. La visualización de los paneles se hace con JavaFX. Esta es una tecnología que nos permite crear interfaces gráficas para aplicaciones Java. Dado que Processing necesita atributos específicos para pintar tanto los metabolitos como las reacciones, este paquete tiene sus propios nodos que contienen atributos como los colores, el tamaño y todo lo necesario para poder ser dibujado en la interfaz. Adicionalmente, Processing tiene acceso directo a la escena de JavaFX, por lo que puede cambiar textos, paneles, tamaño, etc. Esta clase es la que directamente actualiza las áreas de texto que contienen la información del nodo actual.

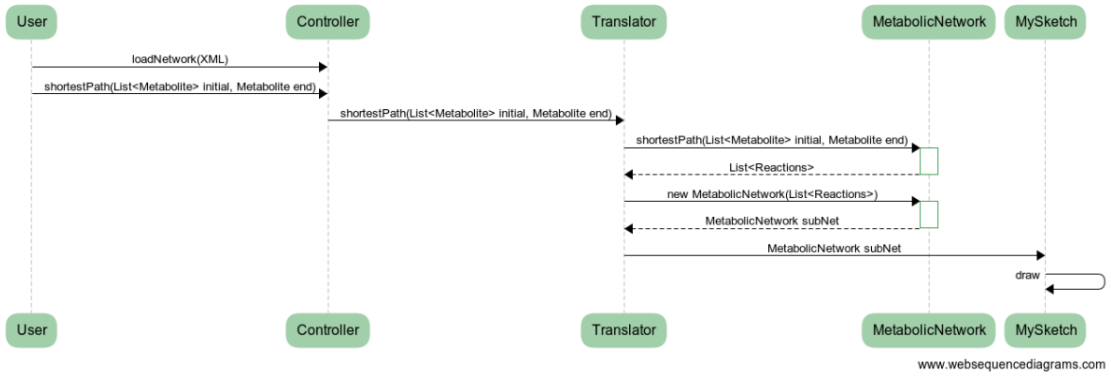


Fig. 7: Diagrama de secuencia para el algoritmo de ruta mínima

Por otro lado, se buscó seguir el patrón MVC ya que esta es la arquitectura utilizada por JavaFX. Sin embargo, con la introducción de Processing, fue necesario modificar esta arquitectura para que JavaFX se pudiera embeber en Processing. Por lo tanto, se añadió al modelo un traductor que se encarga de hacer la traducción entre los datos del mundo y las clases de Processing. Esta clase es muy importante ya que hace que el mundo y la interfaz se puedan comunicar. Por otro lado, se procuró segmentar al máximo la

aplicación para favorecer la mantenibilidad. Además, esto permite que, ya que la aplicación es de código abierto, los usuarios puedan reutilizar las partes del código que sean de su interés, por ejemplo, sólo la parte de visualización para pintar las redes de Petri utilizando Processing.

3.4. Interfaz de Usuario

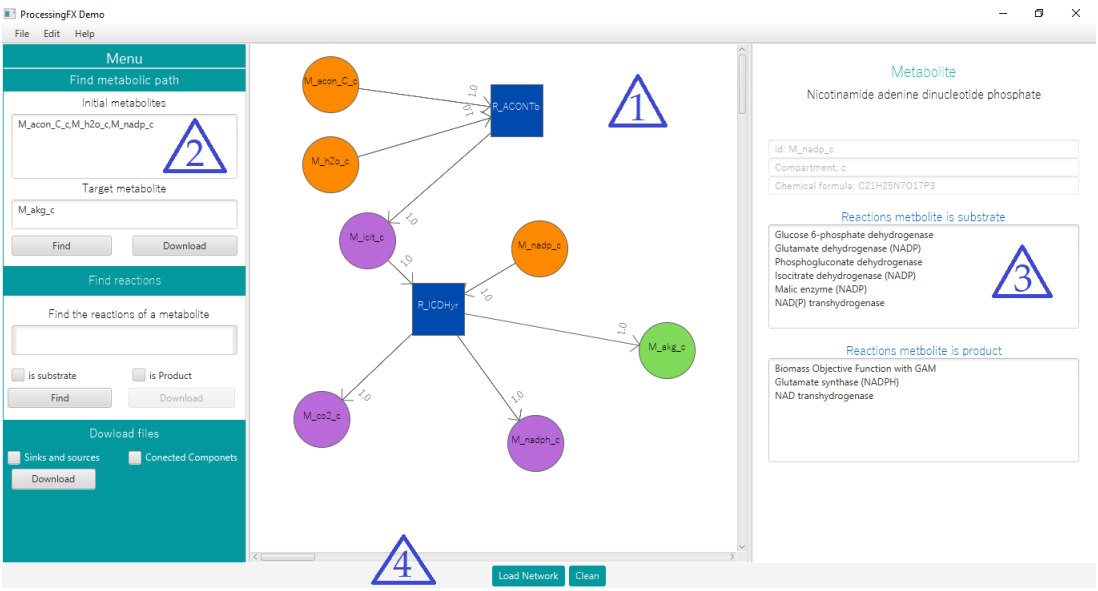


Fig. 8: Interfaz de usuario diseñada para MetaPenta

3.4.1. Panel 1

Todo este panel está hecho en Processing 2.0 y tiene una lógica completamente independiente del resto de la aplicación, aquí sólo se maneja visualización. El usuario puede mover y arrastrar los metabolitos y las reacciones que están en este panel para lograr la conformación de la red que más le facilite el análisis. Adicionalmente, según los requerimientos, los nodos pueden mostrarse de distintos colores. Por ejemplo, en el requerimiento de ruta mínima, los metabolitos iniciales se muestran en naranja, el metabolito objetivo se muestra en verde y los metabolitos intermedios se muestran en morado.

3.4.2. Panel 2

En este panel es donde ocurre la mayor parte de la interacción con el usuario. En la parte superior del panel, el usuario puede introducir los IDs de los metabolitos para hacer la búsqueda de la ruta mínima. Si bajamos un poco, llegamos al campo de texto que nos permite filtrar y mostrar la subnet asociada a un metabolito. Para realizar este filtro, el usuario debe colocar el ID del metabolito y seleccionar si quiere que se muestren las reacciones donde el metabolito es substrato, producto o ambas en el *checkbox* que se encuentra justo debajo. En la última parte de este panel podemos seleccionar, también por medio de un *checkbok*, los documentos que queremos descargar y el programa generará los archivos en la ruta con el prefijo especificado cuando el usuario oprima el botón *"Download"*.

3.4.3. Panel 3

Este panel es el que tiene menos interacción directa con el usuario debido a que, sólo se encarga de mostrar la información del nodo seleccionado. Si es un metabolito, se muestra su id, el compartimento, su fórmula química y las reacciones donde participa. Por otro lado, si es una reacción se muestra su nombre y los IDs de los metabolitos que en ella participan.

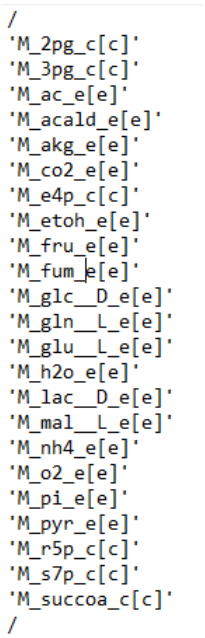
3.4.4. Panel 4

Este panel tiene dos botones. El primero, el de *"load"*, permite cargar el archivo en formato XML de la red metabólica, mientras que el botón de *çlean"* elimina la red que se encuentre cargada para una nueva red pueda ser cargada.

3.5. Comparación con soluciones existentes

3.5.1. Root no production y Root no consumption

(a) Salida del Script de MetaPenta



(b) Salida del Script de GAMs *gapFind*

Fig. 9: Salidas de Meta

Gap find resuelve los problemas *root no production*, y *root no consumption*. Uno de los algoritmos desarrollado para identificar estos metabolitos clave es *gap find*. Una de las implementaciones que hay de este algoritmos está hecho en GAMS y es del grupo de investigación de *Costas Maranas*.

El primer problema que encontramos al utilizar este *script* es la entrada. El programa no recibe entradas estándar de redes metabólicas (JSON o XML) sino que define su propio formato. Este formato está definido por ocho archivos. El primer archivo es el de componentes, donde se guarda un arreglo con todos los

metabolitos de la red. Entre paréntesis se define el compartimiento del metabolito y al inicio y al final el carácter /", lo cual es la forma de definir un arreglo en GAMS. Los otros dos archivos relacionados a metabolitos tienen el mismo formato, pero uno contiene sólo los compuestos citosolicos y el otro sólo los extracelulares. Adicionalmente, se reciben dos archivos que contienen las restricciones para los flux de cada una de las reacciones, un archivo para el límite superior y otro para el límite inferior. También se recibo en un archivo que contenga un arreglo de GAMS con todas las reacciones y otro aparte con un arreglo de las reacciones reversibles. Por último, se recibe en un archivo los ejes de la matriz estequiométrica.

El segundo problema que tenemos para usar este *script* es que al estar ligado a GAMS, está ligado a la versión de este programa que tengamos y dado que GAMS es pago, tenemos o que comprar el programa, o limitarnos a redes muy pequeñas para el análisis. En este caso, dado que no se contaba con una versión paga de GAMS, optamos por correrlo con redes pequeñas, específicamente con el modelo e-coli-core.

El resultado de *gap find* es un archivo con los metabolitos clave. La salida de MetaPenta se puede ver en la Figura 9a mientras que la salida del *script* de GAMS está en la Figura 9b. Este tuvo que correrse asumiendo que la función de biomasa era reversible ya que el script no permitía que no hubiera ninguna reacción no. Como vemos en la Figura 9a estos dos programas sólo difieren en dos metabolitos, lo cual puede deberse a la reacción reversible que pide GAMS. De resto, todos los metabolitos de la sección de *sources* son identificados por MetaPenta. Sin embargo, MetaPenta también identifica dos *sinks* que no son identificados por textitgapFind. Asimismo, textitgapFind identifica otros dos metabolitos (no se sabe si *no production* o *no consumption*) que no son identificados por MetaPenta.

3.5.2. Fluxer

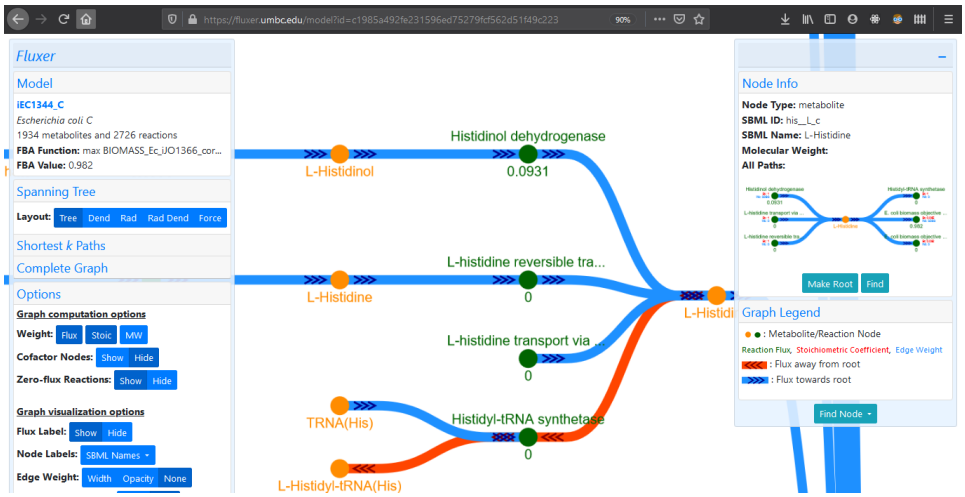


Fig. 10: Interfaz principal de Fluxer

Fluxer es una herramienta que nos permite visualizar las redes metabólicas. La visualización que ofrece Fluxer, aunque es similar al de MetaPenta, no está enfocado a redes de Petri. Fluxer ofrece mucho tipo de

visualizaciones, una de ellas muy similar a una red de Petri. Una de las principales desventajas de Fluxer es que, aunque no consume recursos propios del computador, tiene muchos problemas de *lagging* ya que cargar la visualización puede ser demorado. Muchas veces la página se bloquea lo suficiente para que se ralentice el navegador.

De las características más interesantes de Fluxer está el algoritmo *k-shortest path*, estos algoritmos nos permite encontar el k-ésimo camino más corto. Una versión simplificada de este algoritmo (el 1er camino más corto) es la que está implementada en MetaPenta. Otra diferencia ente MetaPenta y Fluxer es que mientras que Metapenta pide un grupo de metabolitos de entrada, fluxer tiene sólo un metabolito inicial y uno final.

Ejemplo: E-coli core

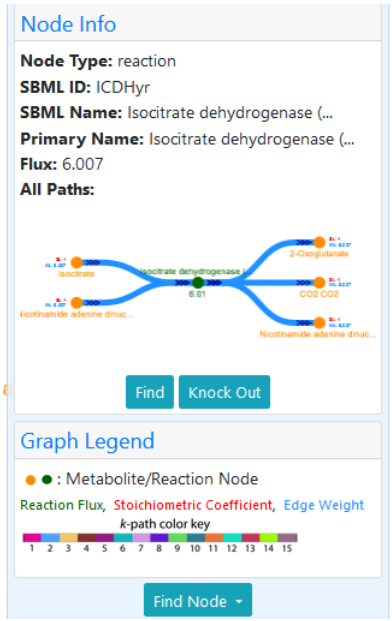


Fig. 11: Detalle de una reacción en Fluxer

Fig. 12: Camino encontrado por fluxer para e-coli core

Esta funcionalidad se probó inicialmente usando el modelo core de e-coli. El camino que se probó fue de *Cis-Aconitate* a *2-Oxoglutarate*. Este es uno de los caminos más sencillos y cortos, pero que nos permiten ver la diferencia en la visualización entre ambos programas. La salida de este ejemplo para MetaPenta se puede ver

en la Figura 8, mientras que en la Figura 12 se ve la salida de Fluxer para la mejor ruta. Como podemos ver, ambos algoritmos encuentran exactamente la misma ruta, pero la representan de manera diferente. Mientras que MetaPenta en su visualización base permite ver todos los metabolitos y las reacciones involucradas en el proceso, Fluxer sólo nos muestra dos metabolitos por reacción y para ver en detalle cuáles sería todos los metabolitos necesarios tocaría ir reacción por reacción anotando los metabólitos de entrada. En MetaPenta esto es más fácil de visualizar, ya que en naranja se muestran cuáles fueron los metabolitos iniciales, en verde el metabolito final y en morado los metabolitos intermedios que se producirán. Es importante mirar los metabolitos intermedios porque esto nos puede llevar a tener productos no deseados que en el peor de los casos pueden ser una fuente de contaminación para el metabolito final, por esta razón es bueno identificarlos. Adicionalmente, Fluxer ni en la visualización básica, ni en visualización por nodo (Figura 11), permite saber cuáles son las enzimas involucradas en la reacción, cosa que también puede ser de interés al usuario.

4. Discusión

MetaPenta es una herramienta que permite realizar diferentes análisis en redes metabólicas. Como parte de este trabajo de grado se implementaron funcionalidades para encontrar patologías como metabolitos sin producir o sin consumir. También se desarrolló una funcionalidad para identificar rutas de producción de metabolitos. Esta funcionalidad es muy importante en diferentes industrias para identificar alternativas innovadoras de generación de diferentes bioproductos. La arquitectura de la solución incluyendo tecnologías de última generación como JavaFX y Processing para visualización e interacción facilita la mantenibilidad y evolución de Metapenta. Si comparamos esta herramienta con otros desarrollos como *Fluxer* se pierde menos información al representar las reacciones como redes de Petri, además de que en esta representación, en el panel asociado a información, se pueden ver las enzimas asociadas a las reacciones, información que Fluxer no muestra. Otra ventaja que tiene MetaPenta es que recibe solamente el XML estándar, a diferencia de *gapFind* que recibe archivos en un formato único para su análisis y toca transformar los archivos en formato estándar a este nuevo formato. Por lo que MetaPenta ofrece varias ventajas que no están considerando algunas de las otras herramientas existentes para analizar redes metabólicas.

Como trabajo futuro se espera implementar diversas funcionalidades, con especial énfasis en la reconstrucción automática de redes metabólicas. Se espera también aprovechar las características de distribución y paralelización de las redes de Petri para diseñar nuevos algoritmos que permitan encontrar nuevas rutas de producción de metabolitos [17]. Finalmente se espera mejorar la interfaz de usuario para permitir la visualización eficiente de redes metabólicas completas. A continuación se incluye una lista de las principales rutas de evolución que se proponen para MetaPenta:

1. Implementar el algoritmos *k-shortest path* a Metapenta y de esta manera poder calcular varias rutas, que es un requerimiento que esta más cerca de las necesidades del usuario
2. Se podría incluir un modelo termodinámicos para ampliar el criterio de lo que sería una buena ruta,
3. Dada la gran cantidad de formatos que cada una de las herramientas relacionadas, añadir más formatos estándar de entrada/salida a MetaPenta podría ser de mucha utilidad.
4. Una de las características más interesantes que podría añadirse a MetaPenta es conexión directa a las

bases de datos públicas, de esta manera poder descargar *on-demand* diferentes modelos de diferentes organismos sin necesidad de que el usuario tenga que introducir el XML sino el nombre del organismo.

5. Encontrar rutas mínima no sólo en el criterio de menor número de reacciones: se pueden encontrar reacciones por menor número de metabolitos, menor número de enzimas necesarias, que sólo ocurran en un compartimento, etc.

6. Algo que Metapenta no consideró ni manejo es el hecho de que hay reacciones reversibles. y al incluirlas hay que tener en cuenta como van a impactar los requerimientos actuales y futuros.

7. Otra cosa interesante para añadir al modelo son las reacciones de intercambio para ciertos análisis ya que estas reacciones son de vital importancia en los modelos.

8. Una de las razones por las cuales Metapenta no pintó la red metabólica completa es porque Processing no da la opción de redibujar sólo ciertas partes de la pantalla, sino que redibuja la pantalla ocmpleta. Con una versión simplificada de la red, o con otra librería podría lograrse la visualización de toda la red metabólica.

Referencias

[1] H. S. Fogler, *Elements of chemical reaction engineering*. Third edition. Upper Saddle River, N.J. : Prentice Hall PTR, [1999] ©1999, [1999]. Includes bibliographical references and index.;System requirements: Windows 95/NT, or Power Macintosh (cannot run executable modules); CD-ROM drive 4x or higher; Java-enabled HTML browser (latest version of Netscape is included); 36 MB of RAM and at least 16 MB of available hard drive space.

[2] C. Angione, "Human systems biology and metabolic modelling: A review—from disease metabolism to precision medicine," *BioMed Research International*, vol. 2019, p. 8304260, Jun 2019.

[3] C. D. Contreras Gamba, R. E. Gómez Díaz, and A. F. González Barrios, "Curación de redes metabólicas con teoría de grafos," 2016. 1 online resource (1 recurso en línea (73 hojas).).

[4] J. D. Orth, I. Thiele, and B. Ø. Palsson, "What is flux balance analysis?," *Nature biotechnology*, vol. 28, pp. 245–248, Mar 2010. 20212490[pmid].

[5] B. Veeramani and J. S. Bader, "Predicting functional associations from metabolism using bi-partite network algorithms," *BMC Systems Biology*, vol. 4, p. 95, Jul 2010.

[6] C. Chaouiya, E. Remy, and D. Thieffry, "Petri net modelling of biological regulatory networks," *Journal of Discrete Algorithms*, vol. 6, no. 2, pp. 165 – 177, 2008. Selected papers from CompBioNets 2004.

[7] E. Wingender, *Biological Petri Nets*, vol. 162. IOS press, 2011.

[8] V. S. Kumar, M. S. Dasika, and C. D. Maranas, "Optimization based automated curation of metabolic reconstructions," *Nature*, vol. 8, p. 212, 6 2007.

[9] K. M. Ng, M. B. I. Reaz, and M. A. M. Ali, "A review on the applications of petri nets in modeling, analysis, and control of urban traffic," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 2, pp. 858–870, 2013.

[10] K.-Q. Zhou and A. Zain, "Fuzzy petri nets and industrial applications: a review," *Artificial Intelligence Review*, vol. 45, pp. 1–42, 12 2015.

[11] H. Wang, S. Marcišauskas, B. J. Sánchez, I. Domenzain, D. Hermansson, R. Agren, J. Nielsen, and E. J. Kerkhoven, "Raven 2.0: A versatile toolbox for metabolic network reconstruction and a case study on streptomyces coelicolor," *PLoS computational biology*, vol. 14, pp. e1006541–e1006541, Oct 2018. 30335785[pmid].

[12] S. Brohée, K. Faust, and J. van Helden, "Network analysis tools (neat) tutorial," 10 2009.

[13] S. Broh—[eacute]—e, K. Faust, G. Lima-Mendez, G. Vanderstocken, and J. van Helden, "Network analysis tools: from biological networks to clusters and pathways," *Nature Protocols*, vol. 3, pp. 1616–1629, 09 2008.

[14] J. Reimand, L. Tooming, H. Peterson, P. Adler, and J. Vilo, "GraphWeb: mining heterogeneous biological networks for gene modules with functional significance," *Nucleic Acids Research*, vol. 36, pp. W452–W459, 05 2008.

[15] A. Hari and D. Lobo, "Fluxer: a web application to compute, analyze and visualize genome-scale metabolic flux networks," *Nucleic Acids Research*, vol. 48, pp. W427–W435, 2 2020.

[16] S.-A. Marashi and Z. Hosseini, "Discovering missing reactions of metabolic networks by using gene co-expression data," *PubMed*, vol. 7, pp. 41774–41774, 2 2017.

[17] S. M. Vahidipour, M. Meybodi, and M. Esnaashari, "Finding the shortest path in stochastic graphs using learning automata and adaptive stochastic petri nets," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 25, pp. 427–455, 06 2017.