TDA Clinvar

Jesús Jiménez Sánchez

Introducción

ClinVar (https://www.ncbi.nlm.nih.gov/clinvar/) es una Base de Datos mantenida por el Instituto de Salud de USA (National Institute of Health) en la que se agrega información sobre mutaciones del genoma humano y sus relaciones con enfermedades. Con el objetivo de facilitar el acceso a estos datos, en esta práctica se propone construir una nueva interfaz que permita consultas más flexibles. Además, impondremos ciertos requisitos sobre la eficiencia de los métodos de gestión y acceso a la información de ClinVar, ya que la escalabilidad es un aspecto muy importante de esta nueva interfaz, que debe responder a un alto número de consultas.

El nuevo tipo de dato Clinvar cumple los siguientes requisitos:

- Representa las mutaciones con el TDA mutación y las enfermedades con el TDA enfermedad especificados en la práctica 2: "TDA Mutación y TDA Enfermedad".
- Obtiene las mutaciones ordenadas por cromosoma/posición.
- Permite un acceso eficiente a las mutaciones por ID.
- Permite recuperar todas las mutaciones que afectan a un gen determinado.
- Obtiene las enfermedades que hay en la base de datos.
- Dado un identificador de enfermedad, identifica las mutaciones que están asociadas con la enfermedad. Si una enfermedad no dispone de identificador, puede ser ignorada.
- Consulta todas las enfermedades que contienen un determinado término en su nombre (por ejemplo, “cancer" o “prostate_cancer").
- Dado un determinado término para el nombre de una enfermedad (ejemplo, "cancer" o “prostate_cancer"), obtiene la secuencia ordenada de las K mutaciones que con mayor probabilidad están relacionadas (son causantes) de la misma.
- Dado el elevado número de mutaciones en la base de datos (más de 130.000), se debe evitar duplicar la información de la base de datos, esto es, NO está permitido mantener copias redundantes de los datos en distintas estructuras y TDAs. En su lugar, la información completa de cada mutación se almacenará una única vez y habrá estructuras adicionales que nos permitan alcanzar los objetivos propuestos.
- Dota de la capacidad de iterar sobre los conjuntos de enfermedades y mutaciones, para este último considerando diversos criterios, el primero iterar en orden creciente cromosoma/posición y el segundo iterar considerando orden creciente de identificador de gen.

Clinvar

```
using namespace std;

typedef string IDgen;
typedef string IDmut;
typedef string IDenf;

class Clinvar{
private:
        set<Mutacion> mutDB;          //Base de datos que contiene toda la información asociada a una mutación
        unordered_map<IDmut, set<Mutacion>::iterator> IDm_map; // Asocia IDmutacion con mutación
        map<IDgen, list< set<Mutacion>::iterator> > gen_map;         // Asocia genes con mutaciones
        map<IDenf, Enfermedad> EnfDB;         // Base de datos de enfermedades
        multimap<IDenf, set<Mutacion>::iterator> IDenf_mmap;       // Asocia enfermedad con mutaciones
public:
        class iterator {
                . . .
        };
        typedef map<IDenf, Enfermedad>::iterator enfermedad_iterator;          // Nos vale utilizar el iterador del map
        class gen_iterator {
                . . .
        };

        //Functor
        class ProbMutaciones{
                . . .
        };

        void load (string nombreDB);
        void insert (const Mutacion & x);
        bool erase (IDmut ID);

        iterator find_Mut(IDmut ID);
        enfermedad_iterator find_Enf(IDenf ID);

        vector<Enfermedad> getEnfermedades(Mutacion & mut);
        list<IDenf> getEnfermedades(string keyword);
        set<IDmut> getMutacionesEnf (IDenf ID);
        set<IDmut> getMutacionesGen (IDgen ID);

        set<Mutacion, ProbMutaciones> topKMutaciones (int k, string keyword);

        /* Métodos relacionados con los iteradores */
        iterator begin();
        iterator end();

        iterator lower_bound(string cromosoma, unsigned int posicion);
        iterator upper_bound(string cromosoma, unsigned int posicion);

        enfermedad_iterator ebegin();
        enfermedad_iterator eend();

        gen_iterator gbegin();
        gen_iterator gend();

        //Clases amigas
private:
        friend class iterator;
        friend class gen_iterator;
        friend class ProbMutaciones;
};
```

Iteradores

Se usarán tres iteradores, del clinvar al completo, de genes y de enfermedades.

```
class iterator {
        . . .
};

typedef map<IDenf, Enfermedad>::iterator enfermedad_iterator;        // Nos vale utilizar el iterador del map

class gen_iterator {
        . . .
};
```

En cada iterador se implementan todos los operadores de los que dispone un iterador normal: ++, =, ==, - -, …

ProbMutaciones

Para ordenar las mutaciones en los conjuntos y las colas con prioridad se usa el functor ProbMutaciones, que ordena las mutaciones según el valor de CAF:

```
(1 - a.getCaf()[0]) > (1 - b.getCaf()[0])
```

Métodos

- load (string nombreDB)

Se encarga de leer los elementos de un fichero dado por el argumento nombreDB, e insertar toda la información en ClinVar.

- insert (const Mutacion & x)

Este método se encarga de insertar una nueva mutación en ClinVar. Esto implica actualizar todas las estructuras necesarias para mantener la coherencia interna de la representación propuesta.

- erase (IDmut ID)

No sólo borra la mutación del repositorio principal de datos sino que además se encarga de borrar toda referencia a dicho elemento dentro de él.
En el caso de que una enfermedad estuviese asociada únicamente a la mutación que está siendo eliminada, esta enfermedad también debe eliminarse de ClinVar.

- find_Mut(IDmut ID)

Busca la mutación con identificador ID dentro de ClinVar, si no lo encuentra devuelve end().

- find_Enf(IDenf ID)

Busca la enfermedad con identificador ID dentro de ClinVar, si no lo encuentra devuelve eend().

- getEnfermedades(Mutacion & mut)

Devuelve un vector con todas las enfermedades asociadas a una mutación en la base de datos clinvar.

- getEnfermedades(string keyword)

Devuelve una lista de los identificadores de enfermedad que contienen la palabra keyword como parte del nombre de la enfermedad.

- getMutacionesEnf (IDenf ID)

Devuelve un conjunto ordenado (en orden creciente de IDmut) de todas las mutaciones que se encuentran asociadas a la enfermedad con identificador ID. Si no tuviese ninguna enfermedad asociada, devuelve el conjunto vacío.

- getMutacionesGen (IDgen ID)

Devuelve un conjunto de todas las mutaciones que se encuentran asociadas a un gen determinado dado por ID.
Si no tuviese ninguno, devuelve el conjunto vacío.

- topKMutaciones (int k, string keyword)

Dado un string 'keyword', el sistema recupera todas las enfermedades cuyo nombre contiene keyword, y devuelve un set ordenado de mutaciones, en orden decreciente de probabilidad, con las k mutaciones más frecuentes en la población asociadas con esas enfermedades.

- begin(), ebegin(), gbegin()

Iteradores a la primera mutación, la primera enfermedad y el primer gen, respectivamente.

- end(), eend(), gend()

Iteradores al elemento que hay después de la última mutación, la última enfermedad y el último gen, respectivamente.

- lower_bound(string cromosoma, unsigned int posicion)

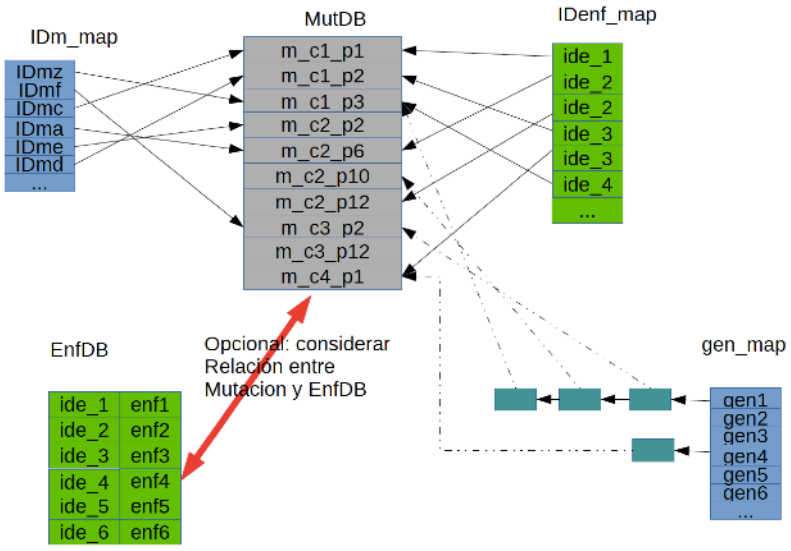
Búsqueda por rango considerando el par de valores cromosoma/posición y devuelve un iterador que apunta a la primera mutación que es mayor o igual a los parámetros.

- upper_bound(string cromosoma, unsigned int posicion)

Búsqueda por rango considerando el par de valores cromosoma/posición y devuelve un iterador que apunta a la primera mutación que es mayor estricta a los parámetros.

Datos

Los datos de la parte privada se relacionan entre sí de la forma que explica la siguiente figura:



- mutDB: Base de datos que contiene toda la información asociada a una mutación.
- IDm_map: Asocia IDmutacion con mutación.
- gen_map: Asocia genes con mutaciones.
- EnfDB: Base de datos de enfermedades.
- IDenf_mmap: Asocia enfermedad con mutaciones.