



UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA MATEMÁTICA

CONSTRUCCIÓN DE UNA BASE PARA EL ESPACIO DE BUBBLES DE
UN DIGRAFO

MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL MATEMÁTICO

DIEGO ANTONIO GRAMUSSET HEPP

PROFESOR GUÍA:
VICENTE ACUÑA

MIEMBROS DE LA COMISIÓN:
ALEJANDRO MAASS
JOSÉ SOTO
JOSÉ ALISTE

SANTIAGO DE CHILE
13 DE JUNIO DE 2021

RESUMEN DE LA MEMORIA PARA OPTAR
AL TÍTULO DE INGENIERO CIVIL MATEMÁTICO
POR: DIEGO ANTONIO GRAMUSSET HEPP
FECHA: 13 DE JUNIO DE 2021
PROF. GUÍA: VICENTE ACUÑA

CONSTRUCCIÓN DE UNA BASE PARA EL ESPACIO DE BUBBLES DE UN DIGRAFO

En un digrafo, una bubble se define como un par de caminos dirigidos con vértices extremos comunes, pero internamente disjuntos. Aparecen naturalmente en el grafo de ensamblaje generado con el fin de realizar la secuenciación del genoma de una especie, y su estudio posee diversas aplicaciones en este campo de la biología.

El conjunto de bubbles en un grafo puede tener, en el peor de los casos, un tamaño exponencial sobre la cantidad de nodos presentes en él, y en el contexto de este trabajo, este número puede variar desde los cientos de miles hasta los miles de millones de nodos, por lo que se hace necesario obtener un modo de representación razonable de ellas.

En el presente trabajo se diseñó un algoritmo que obtiene en tiempo polinomial una base para el espacio de bubbles construida a partir del conjunto de bubbles elementales. Luego se estudió algunas propiedades topológicas del grafo de ensamblaje que permiten diseñar un algoritmo que lleva a cabo esta misma tarea de manera diferente. Se comparan ambos métodos y se determinó cual de ellos es más eficiente.

A todo lo que perdimos en el camino.
Sobre todo tú, Mentés.

Agradecimientos

Esta sección no podría empezarla mencionar a mi familia, son la base de todo lo que soy y por ello les estaré siempre agradecido. Les agradezco también su paciencia. Anne, Pauli y Liss, este trabajo nunca se habría terminado si no fuera por ustedes.

A los profesores y auxiliares que tanto me enseñaron. Mención especial a Sebastián Pérez aka Mario, de quien aprendí mucho, aunque mis notas no lo mostraran.

A mis compañeros, por compartir tantas alegrías y tristezas, fueron un bálsamo en los momentos más duros de la carrera. Tengo una lista de seis planas mencionándolos a todos, si me conociste en Beauchef, tu nombre está en ellas, lo aseguro.

A la gente del Roble, con ustedes el ocio siempre fue dulce. Cada momento con ustedes es un tesoro.

A los cabros de PNZ, Rubén, Candia, Pozo, Franco, Vicencio, Freddy, Vicho, Plutón, a todos, porque me quieren aún conociendo mis peores facetas. Los quiero mucho.

A Alday, Barraza, Kevin, Pablo, Yayo, Turco y Moritas, son mis ratas favoritas y excelentes amigos, siempre es fascinante compartir con ustedes y el amor que les tengo nunca va a cambiar.

A Super Especial, Gabo, Claus, Manu y Chamo, hacer música con ustedes es una maravilla y me encanta todo lo que hemos creado. Me han ayudado a crecer mucho como intérprete, músico y persona.

A mis amigos Caleuchanos, Camilo, Negro, Feña, Javi y Mauricio. Porque han estado ahí siempre, y siempre ahí van a estar. Es esa sempiterna mística que El Caleuche tiene.

A Morales nuevamente, has sido mi mejor amigo por años y tu amistad ha alterado el curso de mi vida de múltiples formas que ni siquiera puedo dimensionar. Gracias por tanto.

Por último, a la Cami y el Sebita. Nuestra amistad es lejos lo mejor que encontré en Beauchef. Son la compañía perfecta y con ustedes sobran buenos consejos, conversaciones interesantes, momentos divertidos, palabras de consuelo y tantas cosas más.

Tabla de Contenido

Introducción	1
1. Antecedentes Preliminares	3
1.1. Teoría de Grafos	3
1.2. Teoría de la computación	7
2. Genoma y modelamiento	9
3. Bubbles en el espacio de ciclos	13
3.1. Espacio de ciclos	13
3.2. Espacio de bubbles	15
3.3. Conjunto fuente y conjunto destino	16
3.4. Generador unión de bases para cada fuente	19
3.5. Un generador de bubbles elementales	19
3.6. Base de $\mathcal{B}(G)$ por Eliminacion Gaussiana	21
3.6.1. Algoritmo con eliminación gaussiana	21
3.6.2. Análisis del algoritmo	22
4. Algoritmo topológico para encontrar un generador	25
4.1. Descomposición del grafo	25
4.2. Eligiendo bubbles	34
4.3. Algoritmo general	41
5. Discusión	46
Conclusión	47
Anexos	56

Índice de Ilustraciones

1.1. Ejemplo de camino.	4
1.2. Grafo conexo.	5
1.3. Camino dirigido y un grafo que no es camino dirigido.	6
2.1. Representación de un SNP en el grafo de ensamblaje.	11
2.2. Ejemplo de generación de bubbles anidadas.	12
3.1. Ejemplo de la suma de un par de grafos.	13
3.2. Representación genérica de una (s,t)-bubble	15
3.3. Ejemplo de bubble elemental no simple.	21
4.1. Representación general del subgrafo G_i , con sus nodos frontera y destino.	26
4.2. G_1 , G_2 y los nodos fronteras del grafo G_2	27
4.3. Esquema del teorema 4.6.	29
4.4. Esquema del teorema 4.7.	31
4.5. Ejemplo de construcción del multigrafo \overline{G}	36
4.6. Ejemplo grafo \hat{E}_i	38

Introducción

La genómica es el área de la biología que se dedica al estudio de la estructura y funcionamiento del genoma de los seres vivos. Por ello, es fundamental para su desarrollo el ser capaz de poder analizar el genoma de una especie, lo que comienza con la *secuenciación*, que consiste en la lectura de las cadenas de nucleótidos componen el genoma. La técnica que permite llevar a cabo esta tarea normalmente requiere fragmentar aleatoriamente el ADN (o ARN) de un conjunto de células, y leer la secuencia de nucleótidos de cada uno de los fragmentos.

Este procedimiento da origen entonces a lo que se conoce en la literatura como el problema del ensamblaje del genoma, es decir, a la reconstrucción del genoma en su totalidad a partir de la lectura de cada una de las secuencias obtenidas de los fragmentos. La existencia de fragmentos repetidos del genoma hacen de éste un problema complejo, del que se han propuesto variadas técnicas algorítmicas y computacionales.

Algunas técnicas para la resolución de este problema utilizan modelos en grafos, donde cada nodo representa una secuencia de nucleótidos y que se conecta con los nodos que podrían ser posibles continuaciones de dicha secuencia en el genoma estudiado. Dos ejemplos de estos modelos son el grafo de De Bruijn y el grafo de traslape (overlap graph).

Si bien el objetivo original de estos modelos en grafos era la reconstrucción del genoma de la especie del cual se generó, o ensamblaje *denovo* [5], con los años surgieron nuevas aplicaciones interesantes. Por ejemplo, la creación de algoritmos para la detección de repeticiones en el genoma y su clasificación [7, 11, 9], enumeración de bubbles [10, 2], y la detección de superbubbles [12, 4, 8].

El interés de este trabajo se centra en las bubbles. Una bubble es un par de caminos dirigidos en el grafo, los cuales comparten su origen y su destino, pero simultáneamente, son totalmente disjuntos en su interior. Este objeto es relevante pues una bubble en el grafo de ensamblaje puede representar diversos fenómenos biológicos importantes como lo son, por ejemplo, los SNPs, indels o splicing de ARN. Sin embargo, no toda bubble tiene tal origen biológico.

Por otra parte, gran número de las bubbles surge “algebraicamente” a partir de la interacción entre bubbles que sí corresponden a eventos de origen biológico.

co. Por ejemplo, cuando un evento se produce en una subsecuencia dentro de otro evento mayor. Esto dificulta la identificación de las bubbles correctas. Sin embargo, su estructura algebraica permite definir conjuntos de bubbles que son capaces de generar otras. De esta propiedad surge naturalmente la pregunta sobre la existencia de un conjunto pequeño de bubbles que pueda generar todo el espacio de bubbles, es decir el conjunto de todas las bubbles presentes en el grafo.

En el trabajo de Acuña et al.[1] se demostró la existencia de un generador del espacio de bubbles de tamaño polinomial y con descomposición calculable en tiempo polinomial. Sin embargo, se plantea que dicho generador no es necesariamente minimal, y queda propuesto el problema de encontrar en tiempo polinomial un generador minimal del espacio de bubbles. Este trabajo busca dar respuesta a ese problema, y tiene por objetivo crear un algoritmo que a partir de un grafo construya una base de su espacio de bubbles. Para ello se utilizará dos enfoques, el primero de ellos buscará construir un conjunto linealmente independiente a partir del conjunto generador dado en [1], y el segundo utilizará propiedades topológicas del grafo para construir una base de similares características.

El objetivo general del presente trabajo es el diseño de un algoritmo que construya una base del espacio de bubbles, la cual esta compuesta en su totalidad por bubbles, conforme a la definición de bubbles previamente expuesta.

Para ello, se seguirá el siguiente esquema:

En el primer capítulo se introducen algunas nociones básicas de teoría de grafos y de computación.

En el segundo capítulo se presenta información sobre genómica que da origen al problema en cuestión y se explica el modelo.

En el tercer capítulo se enfrenta el problema desde una perspectiva gaussiana, construyéndose y analizando un primer algoritmo.

En el cuarto capítulo se estudian propiedades topológicas del grafo del modelo y se construye y analiza un algoritmo a partir de ellas.

En el quinto capítulo se discuten los resultados obtenidos.

Capítulo 1

Antecedentes Preliminares

1.1. Teoría de Grafos

Para entender el trabajo a desarrollarse posteriormente, es necesario tener conocimiento de algunos conceptos y resultados básicos de teoría de grafos. Estos se presentan a continuación.

Definición 1.1 Un **grafo** es un par de conjuntos $G = (V, E)$ que satisface que $E \subseteq [V]^2 := \{e \subseteq V; |e| = 2\}$. Los elementos de V se denominan **vértices** (o **nodos**) de G y los elementos de E se denominan sus **aristas**. La arista $e = \{u, v\}$ se denota por $e = uv$, y los vértices u y v se denominan los **extremos** de e .

Al número de vértices de un grafo es llamado su **orden**, se denota como $n = |G| := |V|$, y el número de aristas se llama **tamaño**, y se denota como $m = \|G\| := |E|$.

Definición 1.2 Sean $G = (V, E), G' = (V', E')$ grafos tales que $V' \subseteq V$ y $E' \subseteq E$, entonces se dirá que G' es **subgrafo** de G .

Además, si E' es exactamente el conjunto de aristas de G cuyos extremos están en V' , se dirá que G' es el **grafo inducido** por V' sobre G .

Si en cambio, V' es exactamente el conjunto de extremos de las aristas de E' , se dirá que G' es el **grafo inducido** por E' sobre G .

Por último, si $V' = V$, se dirá que G' es un subgrafo **recubridor** (o simplemente cubridor) de G .

Definición 1.3 Sean $G = (V, E), G' = (V', E')$ grafos, se define el grafo

$$G \cup G' := (V \cup V', E \cup E')$$

Además, sea $e = uv$; $u, v \in V$, se define $G \cup e := G \cup (\{u, v\}, \{e\})$

Definición 1.4 Sea $G = (V, E)$, $v \in V$ un vértice de G . Sea $e \in E$ tal que $v \in e$, entonces se dira que v es **incidente** en e .

El conjunto de aristas en las que v es incidente se denota por $E(v)$. Al número de aristas en las que v es incidente se denomina el **grado** de v y se denota por $d(v) := |E(v)|$. Un vértice con grado nulo se dice **aislado**.

Un par de vértices $u, v \in V$ se dicen **adyacentes** o **vecinos** si $uv \in E$.

Definición 1.5 Un **camino** es un grafo no vacío $P = (V, E)$ tal que

$$V = \{v_0, v_1, \dots, v_n\} \quad E = \{v_0v_1, v_1v_2, \dots, v_{n-1}v_n\}$$

A los vértices v_0 y v_n se les dice los extremos de P , al resto se les dice interiores y al número de aristas se le llama el **largo** de P . Tambien se dira que P une a los vértices u y v .

En general, se referirá a un camino mediante la secuencia de sus vértices de la forma $P = v_0v_1\dots v_n$ y para $0 \leq i \leq j \leq n$ se denota por:

$$\begin{aligned} Pv_i &:= v_0\dots v_i \\ v_iPv_j &:= v_i\dots v_j \\ v_jP &:= v_j\dots v_n \end{aligned}$$

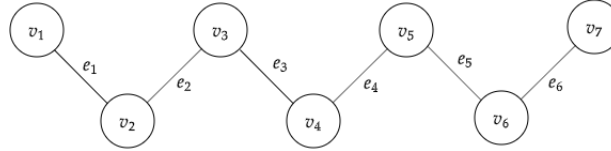


Figura 1.1: Ejemplo de camino.

Definición 1.6 Sea $P = v_0\dots v_n$ un camino, al grafo $C = P \cup v_nv_0$ se le llama un **ciclo**, y, del mismo modo, se puede denotar como $C = v_0\dots v_nv_0$.

Definición 1.7 Un grafo $G = (V, E)$ se dira:

- **conexo** si para todo par de vértices $u, v \in V$ existe un camino $P \subseteq G$ tal que u y v son sus extremos.

- **acíclico** si no contiene ningún ciclo. A un grafo acíclico también se le llama **bosque**.
- **árbol** si es un bosque conexo. En general, se denota un grafo que es árbol con la letra T .
- **árbol enraizado** si es un árbol y existe un vértice $r \in V$ denominado raíz.

Además se llama **componente conexa** de un grafo G a todo subgrafo conexo maximal.

En la siguiente figura se presenta un grafo conexo, el cual es acíclico, y por ende árbol, si se remueve la arista e_1 , y deja de ser conexo si se remueve la arista e_2 .

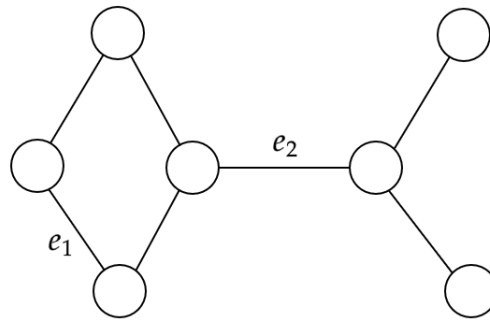


Figura 1.2: Grafo conexo.

Proposición 1.8 Sea $T = (V, E)$ un árbol, entonces para todo par de vértices $u, v \in V$ existe un único camino $P_{uv} \subseteq T$ que los une

Definición 1.9 Sea $G = (V, E)$ un grafo conexo, se define como **árbol recubridor** de G a todo subgrafo recubridor T , que es árbol.

Definición 1.10 Se define un **grafo dirigido** o **digrafo** como un par de conjuntos $G = (V, E)$ con $E \subseteq V^2$ llamados conjunto de vértices y de arcos, respectivamente. Se puede interpretar un digrafo como un grafo en el cual el orden en que se define una arista es relevante.

A un arco de la forma $uu \in E$ se le dira bucle, y al par de arcos $uv, vu \in E$ se les llama arcos paralelos. En el presente trabajo se consideraran grafos dirigidos sin bucles ni arcos paralelos.

Se define un par de funciones sobre las aristas de un digrafo

$$i : E \longrightarrow V \quad ; \quad t : E \longrightarrow V$$

donde para $e = uv \in E$ se tiene que $i(e) = u$; $t(e) = v$.

Además se dicen **vecinos de salida** (resp. entrada) de u a los nodos $v \in V$ tales que $uv \in E$ (resp. $vu \in E$). Este conjunto se denotará por $N_+(u)$ (resp. $N_-(u)$).

Se pueden extender los conceptos de camino y ciclo a grafos dirigidos, considerando además que para todo par de arcos consecutivos, e_i, e_{i+1} , se tiene que satisfacer la condición $t(e_i) = i(e_{i+1})$. Esta condición lo que garantiza es que los arcos consecutivos posean una orientación coherente en el orden en que está definido el camino.

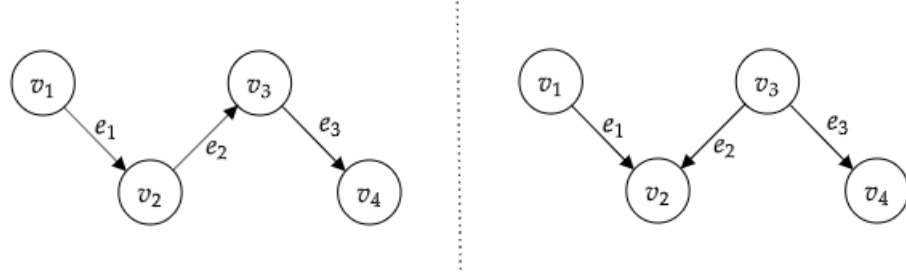


Figura 1.3: Camino dirigido y un grafo que no es camino dirigido.

Definición 1.11 Sea G un grafo dirigido sin bucles ni aristas paralelas. Se llama el **grafo subyacente** de G al grafo obtenido a partir de este cuando se considera el conjunto de aristas sin sus direcciones.

Nótese que se requiere que el digrafo no posea bucles ni aristas paralelas para que al remover las direcciones de todas las aristas, el resultado sea efectivamente un grafo y posea el mismo tamaño que el grafo G .

Definición 1.12 Se define también un multigrafo como un par $G = (V, E)$ de conjuntos disjuntos y una función $t : E \longrightarrow [V]^2$ que a cada elemento de E le asigna un subconjunto de 2 elementos de V que se denominan sus extremos. Esta definición busca representar un grafo en el cual es posible la existencia de múltiples aristas entre un mismo par de vértices.

Nuevamente los conceptos de grafos se extienden naturalmente a multigrafos, pero debido a la existencia de multiples aristas entre un par de nodos, los caminos deben estar caracterizados por la secuencia de aristas que lo conforman, pues la secuencia de vértices puede generar ambigüedades.

1.2. Teoría de la computación

A continuación se definiran algunos conceptos básicos de teoría de computación.

Definición 1.13 Un **problema** se define como un subconjunto $\mathcal{P} \subseteq I_{\mathcal{P}} \times S_{\mathcal{P}}$, donde $I_{\mathcal{P}}$ corresponde al conjunto de instancias del problema y $S_{\mathcal{P}}$ al conjunto de soluciones del problema. Luego, diremos que $y \in S_{\mathcal{P}}$ es solución de \mathcal{P} en la instancia $x \in I_{\mathcal{P}}$ si $(x, y) \in \mathcal{P}$.

Definido lo que es un problema, se definirá el concepto de algoritmo de manera intuitiva, pues para realizar una definición formal, se requeriría ahondar en conceptos de teoría de la computación que se escapan del alcance de esta tesis.

Definición 1.14 Se dice que una secuencia de operaciones básicas bien definida A es un **algoritmo** que resuelve el problema \mathcal{P} si, dada cualquier instancia $x \in I_{\mathcal{P}}$ como entrada a A , su valor de retorno, $A(x)$ posee una o más soluciones de \mathcal{P} en x .

Es decir, un algoritmo es una herramienta que encuentra soluciones para un problema. No obstante, esto no basta en la realidad, un algoritmo debe encontrar soluciones a un problema en un tiempo razonable. ¿Qué quiere decir esto último? Para entenderlo nos serán útiles los siguientes conceptos.

Definición 1.15 Llamamos **tamaño de la entrada** al largo que posea en el sistema de codificación utilizado por el algoritmo la instancia x sobre la que se ejecutará, es decir $|x|$.

En general se supone que la entrada está codificada de manera razonable, lo que quiere decir que no se introduce información redundante, y también se supone que para un par de codificaciones distintas de la misma entrada x , el tamaño de ellas no puede diferir mucho.

Dado un algoritmo A para un problema \mathcal{P} , con una instancia x de tamaño $|x| = n$, queremos determinar, en términos de n , cuántas operaciones llevará a cabo A al momento de retornar una solución. A este valor le diremos *tiempo de ejecución*.

El valor del tiempo de ejecución depende obviamente del tamaño de la entrada n , pero también depende de x , es decir, para entradas diferentes del mismo tamaño el tiempo de ejecución puede variar de manera impredecible. Luego, para poder asignar al algoritmo A un tiempo de ejecución independiente de la

entrada de manera conservadora, siempre se considerará el mayor tiempo de ejecución para entradas de un mismo tamaño n . Este procedimiento se denomina análisis del peor escenario.

Por último, para un algoritmo A , su tiempo de ejecución para una entrada de tamaño n , que se denota $t_A(n)$, dependerá también de la máquina mediante la cual será ejecutado, por lo que se realiza la siguiente definición.

Definición 1.16 *Sea A un algoritmo con tiempo de ejecución $t_A(n)$, n el tamaño de la entrada. Sea $g : \mathbb{N} \longrightarrow \mathbb{N}$ una función. Se dirá que el tiempo de ejecución de A es de orden $g(n)$, denotado por $O(g(n))$, si existe un natural n_0 tal que para todo $n > n_0$ se cumple que*

$$t_A(n) < cg(n)$$

para algún valor constante c .

De esta forma se puede expresar el tiempo de ejecución de un algoritmo netamente en terminos del tamaño de su entrada, considerando siempre el peor escenario y de manera independiente a la tecnología mediante la cual se ejecutará.

Capítulo 2

Genoma y modelamiento

El genoma corresponde al conjunto de genes presentes en las células de una especie o individuo en particular. Se puede considerar como la cadena de nucleótidos presente en cada uno de sus cromosomas, y su estudio es la base de la genómica.

Los nucleótidos son en general las unidades fundamentales del material genético, y corresponden a ensambles de una molécula de pentosa (ribosa o desoxirribosa), una molécula de ácido fosfórico, y alguna base nitrogenada. Este último elemento es el que le da el carácter a cada una de las bases utilizadas en el código genético de una célula. Las bases presentes en el ADN de una célula corresponden a adenina, citosina, guanina y timina. Este conjunto de bases nitrogenadas será modelado como el alfabeto $\mathcal{A} := \{A, C, G, T\}$, indicando cada base nitrogenada por su letra inicial. En el ARN, la timina es reemplazada por uracilo, por lo que la letra T podría ser reemplazada por su parte por la letra U .

Un gen corresponde a una secuencia de bases nitrogenadas que ocupan un lugar particular dentro de un cromosoma (*locus*) y que codifica la información necesaria para la síntesis de alguna molécula que desarrolle alguna función fisiológica específica. A las diversas secuencias de bases que se pueden encontrar en un mismo gen se les llama *alelos*. Es decir, hablar de un gen en específico hace alusión a todas las posibles cadenas de bases nitrogenadas que se pueden encontrar en una región específica del genoma de una especie, y que sintetizan moléculas que cumplen la misma función biológica (aunque con posibles variaciones en como se desarrolla esa función debido a las variaciones en el gen). Cabe destacar que a raíz de esto, dentro de una población de individuos de una misma especie, o incluso dentro del mismo individuo, se puede observar diversidad genética, la cual puede o no estar expresada en características físicas. Al conjunto de genes presentes en un individuo se le denomina genotipo, y al resultado de la expresión de dichos genes en combinación con la interacción con el medio ambiente se le denomina fenotipo.

Por esto, se busca estudiar el genoma de una especie, y para ello es necesario obtener la secuencia de bases nitrogenadas que componen cada uno de sus cromosomas y sus funciones. Actualmente, las distintas tecnologías existentes permiten realizar lecturas de hebras de ADN (o ARN) de diverso largo, en cualquier caso mucho menor que un genoma completo. Por ello, para poder llevar a cabo análisis y estudios genéticos, se requiere construir, a partir de dichas lecturas, secuencias de bases más largas, intentando completar el genoma de un organismo. Estas lecturas se realizan mediante una técnica llamada reacción de cadena de polimerasa, o PCR, por su sigla en inglés.

Teniendo lecturas, la reconstrucción del genoma se podría realizar estudiando las coincidencias entre los finales y principios de las diferentes lecturas, sin embargo, el gran número de lecturas hace que la comparación entre todas ellas sea un problema extremadamente costoso en términos de tiempo. Por ejemplo, para un conjunto de 10^4 lecturas de largo 10^3 , se tendrían que realizar $o(10^3)$ operaciones para comparar cada par de lecturas, y son $o(10^8)$ pares de lecturas, por lo que en total, sólo en esta parte del proceso se requeriría de $o(10^{11})$ operaciones.

Una manera alternativa de abordar el problema, evitando este excesivo número de comparaciones, es considerar el conjunto de todas las secuencias de largo k , para algún valor fijo k , que se encuentran dentro de alguna lectura. A estas cadenas se les llama k – *meros*, y a partir de ellas se puede construir un grafo dirigido, en el que cada nodo es un $(k - 1)$ – *mero*, y los arcos serán los k – *meros* encontrados, de forma que para cada arco, su nodo inicial corresponde a sus primeras $k - 1$ bases y su nodo terminal, sus últimas $k - 1$ bases. Un grafo construido de esta forma recibe el nombre de grafo de De Bruijn, y en el caso particular de este problema se le dirá grafo de ensamblaje. Además, cada arco entre un par de nodos aparece múltiples veces, según la frecuencia relativa con la que el k – *mero* aparece dentro de las lecturas. De este modo, la hebra original de ADN corresponde a un camino euleriano dentro de este digrafo.

El grafo de De Bruijn obtenido de las lecturas, es el paso previo para intentar obtener, no sin cierta dificultad, la secuencia real del genoma. Sin embargo, esta estructura también puede reflejar otros fenómenos biológicos que son de alto interés para el mundo científico. Estos fenómenos biológicos se encuentran representados en patrones específicos de la estructura del grafo y pueden ser estudiados sin necesidad de obtener un ensamblaje particular. Algunos ejemplos de especial interés en este trabajo corresponden a tres fenómenos: splicing alternativo, indels y SNPs.

El splicing alternativo corresponde al fenómeno biológico a través del cual diversas secuencias de ARN mensajero pueden ser obtenidas a partir de una misma secuencia de ADN. El ARN mensajero se obtiene a partir del ADN, mediante el descarte de segmentos no codificantes denominadas *intrones*, y la

unión de segmentos codificantes denominadas *exones*. Es decir, dentro de una misma sección de ADN, pueden estar codificadas una multiplicidad de genes, y según cuales sean las regiones que se encuentran activas al momento de realizar la síntesis del ARN, la molécula obtenida finalmente como resultado variará en su función. Esto en el grafo de De Bruijn se puede observar como aparición de ciclos dirigidos.

Un indel, por otra parte, corresponde a la inserción(in) o eliminación(del) de un número variable de nucleótidos en un sector del genoma de un organismo. Estas variaciones pueden ir desde las unidades hasta las decenas de miles y tienen aplicaciones en el estudio de filogenética como marcadores de población[13]. En el grafo se observan como caminos dirigidos alternativos entre un par de nodos de un camino dirigido mayor, si el camino genera un ciclo dirigido corresponde a una inserción, si no corresponde a una eliminación.

Por último, un polimorfismo de un solo nucleótido, o SNP por su sigla en inglés, corresponde a la variación de un solo nucleótido en una posición específica del genoma, lo que da lugar a alelos diferentes. Por ejemplo, el factor V Leiden es un transtorno de hipercoagulabilidad que se genera por un SNP *G* a *A* en el nucleótido 1961 del gen *F5*[6]. Este fenómeno se observa como una bifurcación de largo $k - 1$ en un camino dirigido en el grafo de ensamblaje.

Como se puede observar, estos tres tipos de fenómenos se expresan de manera similar en el grafo de ensamblaje. Cada uno de ellos generan una o más estructuras que en la literatura reciben el nombre de bubble (o burbuja), y es en estas estructuras donde se enfoca esta tesis. El concepto de burbuja será definido con rigurosidad posteriormente.

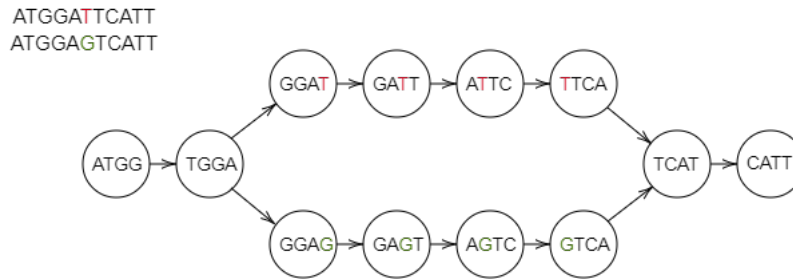


Figura 2.1: Representación de un SNP en el grafo de ensamblaje.

Sin embargo, la presencia de estos fenómenos biológicos en el genoma puede dar origen a un número mucho mayor de bubbles cuando estos eventos se encuentran anidados unos dentro de otros. La siguiente figura presenta como tan solo dos SNP consecutivos dan origen a un total de cinco bubbles.

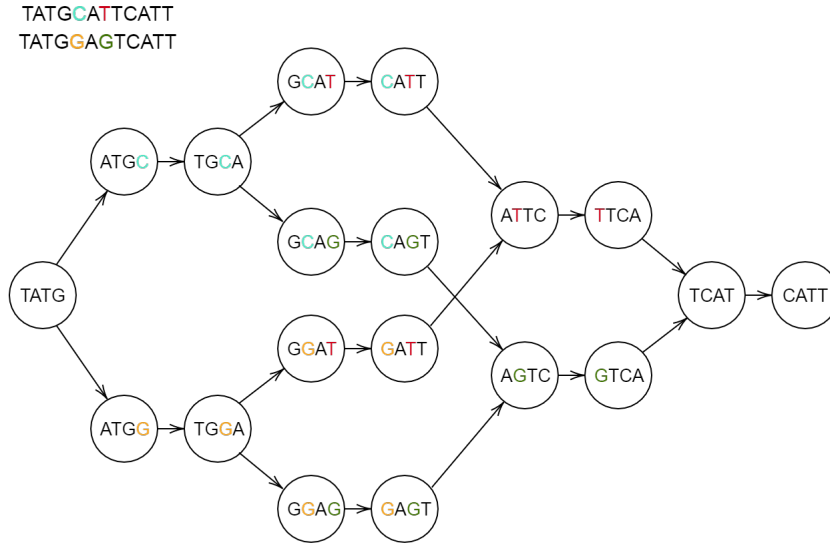


Figura 2.2: Ejemplo de generación de bubbles anidadas.

Por este motivo se han desarrollado algoritmos que buscan y enumeran bubbles [2, 10], que reconocen repeticiones y palíndromos de secuencias dentro de las lecturas [7, 11], y que detectan superbubbles dentro del grafo [4, 8, 12]. Gracias a estos trabajos, es posible reconocer y detectar las bubbles existentes dentro de una región del grafo de ensamblaje. A pesar de ello, dado que el número de bubbles en un grafo de De Bruijn es enorme, estas soluciones no permiten identificar fácilmente un conjunto pequeño de ellas que describa los eventos biológicos mencionados.

El presente trabajo busca generar un algoritmo que encuentre un conjunto de ellas de tamaño reducido, pero que mediante herramientas algebraicas sea capaz de representar a todas las bubbles del grafo de ensamblaje. Esto puede dar una primera aproximación a determinar el número de bubbles cuyo origen se encuentra en un fenómeno biológico y quizás en un futuro poder encontrarlas.

Capítulo 3

Bubbles en el espacio de ciclos

En este capítulo se presenta el espacio de ciclos de un grafo, y se presenta un resultado útil de este espacio vectorial. Luego se define de manera análoga para un digrafo el concepto de espacio de bubbles, se presentan algunos conjuntos generadores de dicho espacio, y se presenta un algoritmo que a partir de cualquiera de dichos generadores, es capaz de encontrar un generador minimal.

3.1. Espacio de ciclos

Definición 3.1 Sean $G_1 = (V_1, E_1)$ y $G_2 = (V_2, E_2)$ subgrafos de un grafo G , se define $G_1 + G_2$ como el subgrafo inducido por el conjunto de aristas $E_1 \Delta E_2$.

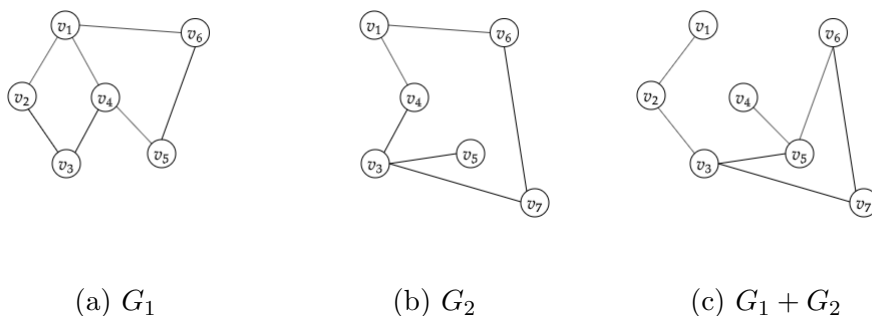


Figura 3.1: Ejemplo de la suma de un par de grafos.

Definida esta suma, el espacio de conjuntos de aristas de un grafo G posee estructura de espacio vectorial sobre el cuerpo \mathbb{F}_2 , el cuerpo de dos elementos.

Definición 3.2 Sea $G = (V, E)$ un grafo, al subespacio vectorial generado por el conjunto de sus ciclos se le denomina **espacio de ciclos** de G y se denota por $\mathcal{C}(G)$. Notar que no todos los elementos del espacio de ciclos son necesariamente ciclos.

En terminos generales, un subgrafo $G' = (V, F)$ pertenece al espacio de ciclos de G si y solo si todo vértice $v \in V$ tiene grado par en G' .

Definición 3.3 Sea T un árbol de un grafo G , decimos que un ciclo está asociado a T si se forma a partir de aristas en T más una arista fuera de T .

Proposición 3.4 Sea T un árbol recubridor de un grafo conexo. El conjunto β_T de los ciclos asociados a T corresponde a una base del espacio de ciclos.

DEMOSTRACIÓN. Consideremos primero un grafo G conexo y T un árbol recubridor de G . Para cada arista $e \notin T$ podemos construir un único ciclo con las aristas del árbol T .

Sea β el conjunto de los ciclos generados de esta forma, es claro que $|\beta| = \|G\| - \|T\| = m - n + 1$.

Es evidente que este conjunto es linealmente independiente, pues cada uno de sus elementos posee una arista que es única. Para determinar que es de hecho una base del espacio de ciclos, basta con demostrar que es generador de los ciclos del grafo.

Llevemos a cabo esta demostración por inducción sobre el número de aristas que no se encuentran en T del ciclo en cuestión.

- Cuando el ciclo posee solo 1 arista fuera de T , luego se trata de un ciclo perteneciente a β , por lo que es trivial.
- Supongamos ahora que se tiene el resultado para ciclos con $k - 1$ aristas fuera de T , y sea C un ciclo con k aristas fuera de T , sea $\bar{e} \in C \setminus T$ una de ellas, y sea $C_{\bar{e}}$ el ciclo de β asociado a ella. Luego $C + C_{\bar{e}}$ es un elemento del espacio de ciclos con $k - 1$ aristas fuera de T , por lo que se puede generar con elementos de β , es decir,

$$C + C_{\bar{e}} = \sum_{e \notin T} \alpha_e C_e$$

De este modo

$$C = C_{\bar{e}} + \sum_{e \notin T} \alpha_e C_e$$

Por lo que C puede generarse con elementos de β

Esto nos permite concluir entonces que el conjunto β es de hecho generador y, por ende, base del espacio de ciclos. □

Corolario 3.5 La dimensión del espacio de ciclos de un grafo es $\dim(\mathcal{C}(G)) = m - n + c$, donde c es el número de componentes conexas del grafo.

DEMOSTRACIÓN. Para un grafo G no necesariamente conexo, aplicamos el resultado anterior para cada componente conexa de G y se construye una base de cardinal $m - n + c$.

□

3.2. Espacio de bubbles

En este trabajo se busca llegar a un resultado similar al presentado en la sección anterior para una estructura presente en digrafos, íntimamente relacionada con los ciclos de su grafo subyacente.

Primero, cabe notar que un grafo dirigido de De Bruijn construido a partir de lecturas del material genético sí puede tener bucles y arcos paralelos. Esto presenta problemas al momento de querer construir su grafo subyacente, pero en general, este problema se puede reparar reemplazando los bucles por 2 nodos auxiliares y convirtiéndolos en un ciclo de largo 3, y reemplazando un arco de cada par de aristas paralelas por un nodo extra y convirtiendo el arco en un camino de largo 2.

Supondremos de ahora en adelante entonces que los grafos dirigidos no poseen bucles ni arcos paralelos, pues sabemos que si los poseen, se pueden reparar.

Definición 3.6 Sea G un grafo dirigido. Un subgrafo B de G se denomina **bubble** si es la unión de dos caminos p y q ambos comenzando en un vértice s y terminando en un vértice t y que no poseen ningún otro vértice en común (es decir, sus vértices internos son disjuntos). Los caminos p y q se denominan las *piernas* de B y se dirá que B es una (s, t) -bubble.

Si p y q son las piernas de una (s, t) -bubble B , denotamos $B = (p, q)$ y se define $\text{Inicio}(B) = s$ y $\text{Terminal}(B) = t$.

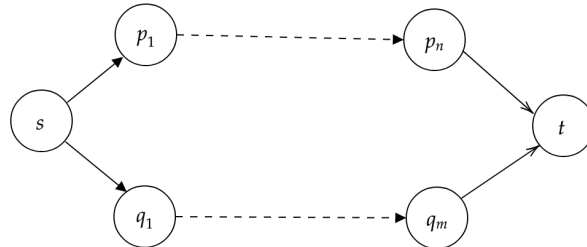


Figura 3.2: Representación genérica de una (s, t) -bubble

Definición 3.7 Sea G un grafo dirigido. Definimos $\mathbb{B}(G)$ como el conjunto de

todas las bubbles de G .

Notamos que el grafo subyacente de toda bubble de G está en el espacio de ciclos del grafo subyacente de G . Es por esto que decimos, abusando un poco de la terminología, que $\mathbb{B}(G)$ está en el espacio de ciclos de G . Notar que $\langle \mathbb{B}(G) \rangle$ es el subespacio vectorial más pequeño que contiene a $\mathbb{B}(G)$ y puede contener elementos que no son bubbles. Nuestro objetivo es encontrar un conjunto β compuesto sólo por bubbles, que sea generador de $\mathbb{B}(G)$, pero de cardinal mucho menor que $\mathbb{B}(G)$. Es decir, $\langle \beta \rangle = \langle \mathbb{B}(G) \rangle$. Idealmente se busca que β sea linealmente independiente, es decir que β sea base. Al espacio generado por todas las bubbles de un grafo se le denominará **espacio de bubbles** y se denotará $\mathcal{B}(G)$.

Claramente, si β es base del espacio de bubbles, su cardinal estará acotado superiormente por la dimensión del espacio de ciclos, es decir por $m - n + c$. Más aún, a partir de $\mathbb{B}(G)$ o a partir de cualquier generador de $\mathbb{B}(G)$ compuesto por bubbles, es posible encontrar una base de $\mathcal{B}(G)$ realizando eliminación Gaussiana. Sin embargo esto puede ser computacionalmente costoso dependiendo del tamaño del generador inicial.

Surge la pregunta: así como es relativamente sencillo encontrar una base del espacio de ciclos a partir de un árbol recubridor, ¿existirá un algoritmo para encontrar un generador β del espacio de bubbles a partir de la topología del grafo G ?

El objetivo será entonces encontrar, a partir de la topología del grafo G , un conjunto β , compuesto por bubbles, que sea base o al menos un conjunto generador de todas las bubbles de un grafo G y que tenga un tamaño cercano al de una base en el espacio de ciclos, i.e $O(n + m)$.

3.3. Conjunto fuente y conjunto destino

Vamos a introducir algunas nociones relacionadas con la existencia de caminos entre vértices del grafo.

Definición 3.8 *Sea $G = (V, E)$ un grafo dirigido. Dado dos vértices $u, v \in V$ se dice que u es alcanzable desde v si existe un camino de v a u en G . Se denota por $\text{Alcanzable}(v) \subseteq V$ al conjunto de todos los vértices alcanzables desde v .*

Proposición 3.9 *Para $v_1, v_2 \in V$ tales que $v_2 \in \text{Alcanzable}(v_1)$, entonces se tiene que $\text{Alcanzable}(v_2) \subseteq \text{Alcanzable}(v_1)$.*

DEMOSTRACIÓN. Sea $u \in \text{Alcanzable}(v_2)$, luego existe P_2 camino en G que va de v_2 a u , del mismo modo existe P_1 camino en G que va de v_1 a v_2 . Sea w

el primer nodo de P_1 que se encuentra contenido en P_2 , entonces el conjunto $v_1 P_1 w P_2 u$ es un camino en G que va de v_1 a u , lo que permite concluir que $u \in \text{Alcanzable}(v_1)$. □

Definición 3.10 Sea $G = (V, E)$ un grafo dirigido, un conjunto de vértices $S \subseteq V$ se denomina conjunto **fuentes** si para todo vértice $u \in V$ existe un vértice $v \in S$ tal que $u \in \text{Alcanzable}(v)$. Si un conjunto fuentes S no contiene un conjunto más pequeño que sea fuentes se dira que S es un conjunto fuentes minimal.

Definición 3.11 Sea $G = (V, E)$ un grafo dirigido, un conjunto de vértices $D \subseteq V$ se denomina conjunto **destino** si para todo vértice $u \in V$ existe un vértice $v \in D$ tal que $v \in \text{Alcanzable}(u)$. Si un conjunto destino D no contiene un conjunto más pequeño que sea destino se dira que D es un conjunto destino minimal.

Proposición 3.12 Todo conjunto fuentes minimal contiene exactamente un vértice de cada componente fuertemente conexa de G que no posea arcos entrantes de otras componentes conexas.

DEMOSTRACIÓN. Sea $S \subseteq V$ un conjunto de vértices de G tales que S es conjunto fuentes minimal. Sea C una componente fuertemente conexa de G tal que C no posee arcos entrantes. Sea $u \in C$, como S es conjunto fuentes, existe $v \in S$ tal que $u \in \text{Alcanzable}(v)$, es decir, existe un camino en G de v a u , pero como C es una componente conexa sin arcos entrantes, y $u \in C$, podemos concluir que $v \in C$. Esto nos permite concluir que en S hay al menos un vértice por cada componente fuertemente conexa sin arcos entrantes.

Además es fácil notar que debe ser exactamente un vértice por cada componente fuertemente conexa sin nodos entrantes. Supongamos que no es así, es decir que existe una componente fuertemente conexa sin arcos entrantes C' tal que existen $u_1, u_2 \in C'$ y $u_1, u_2 \in S$. Como C' es fuertemente conexa, existe P camino dirigido en C' que va de u_1 a u_2 , por lo que $u_2 \in \text{Alcanzable}(u_1)$, y por la proposición 3.9, tenemos que $\text{Alcanzable}(u_2) \subseteq \text{Alcanzable}(u_1)$, lo que implica que el conjunto $S \setminus \{u_2\}$ es conjunto fuentes, contradiciendo la minimalidad de S .

De forma similar, si S contiene un elemento w tal que no pertenece a ninguna componente fuertemente conexa sin arcos entrantes. Sea C la componente fuertemente conexa que contiene a w , e un arco entrante a C y $w' = i(e)$. Sea C' la componente fuertemente conexa que contiene a w' , si C' no tiene arcos entrantes, entonces existe $u \in S$ tal que $u \in C'$ y por ende existe un camino

P' entre u y w' en C' y un camino P entre w' y w que contiene al arco e , de modo que existe un camino entre u y w en G , por lo que nuevamente, por 3.9, se contradice la minimalidad de S . Si C' tiene arcos entrantes, este argumento puede reiterarse nuevamente, y debido a que las componentes fuertemente conexas de un grafo son finitas y no pueden reiterarse, eventualmente llega a una sin arcos entrantes, reduciéndose al caso anterior.

□

Proposición 3.13 *Todo conjunto destino minimal contiene exactamente un vértice de cada componente fuertemente conexa de G que no posea arcos salientes hacia otras componentes conexas.*

DEMOSTRACIÓN. Este resultado es corolario del resultado anterior, considerando el grafo con sus direcciones invertidas.

□

Corolario 3.14 *Todo conjunto fuente minimal de un grafo tiene el mismo cardinal.*

Corolario 3.15 *Todo conjunto destino minimal de un grafo tiene el mismo cardinal.*

Definición 3.16 *Sea G un grafo dirigido. Si $k \in \mathbb{N}$ es el cardinal de cualquier conjunto fuente minimal, entonces se dice que G es un grafo k -fuente. Si $m \in \mathbb{N}$ es el cardinal de cualquier conjunto destino minimal, entonces G es un grafo m -destino.*

Teorema 3.17 *Sea $G = (V, E)$ un grafo dirigido 1-fuente (1-destino), entonces el espacio de ciclos y el espacio de bubbles coincide y se puede construir una base compuesta sólo por bubbles de G en $\mathcal{O}(n + m)$.*

DEMOSTRACIÓN. Recordemos que en general, el espacio de bubbles es un subespacio vectorial del espacio de ciclos. Sea $v \in V$ un nodo fuente de G . Se puede construir un árbol recubridor de G con raíz en v (es decir, sus arcos están orientados desde v a las hojas), el cual a su vez permite construir, una base del espacio de ciclos del grafo subyacente de G como en la demostración de la proposición 3.4. Como los elementos de esa base son bubbles, entonces claramente el espacio de bubbles es exactamente el espacio de ciclos.

Para el caso 1-destino se puede considerar una inversión de las direcciones del grafo y se reduce el problema al caso anterior.

□

Como ya se ha mencionado en el capítulo anterior, las bubbles aparecen de manera segura en el grafo de ensamblaje y su estudio está vinculado a diversos fenómenos biológicos que son de interés en la genómica. A continuación se presentan algunos conjuntos generadores del espacio de bubbles, y posteriormente se propone un método que construye una base del espacio a partir de ellos.

3.4. Generador unión de bases para cada fuente

Ya se estableció como encontrar una base compuesta por bubbles para el caso de un grafo 1-fuente y 1-destino, por lo que, de aquí en adelante, se supondrá que el grafo no es 1-fuente ni 1-destino. Se dirá entonces que G tiene k fuentes, con $k \geq 2$.

Definición 3.18 Sea G un grafo dirigido, $S = \{s_1, \dots, s_k\}$ un conjunto fuente minimal, denotaremos por G_i al grafo inducido por $\text{Alcanzable}(s_i)$ para todo $i \in \{1, \dots, k\}$. Cabe notar que G_i es un grafo 1-fuente con s_i una fuente del mismo.

Una manera de encontrar un generador compuesto solo por bubbles que surge naturalmente consiste en calcular una base para cada G_i con $i \in \{1, \dots, k\}$ y considerar la unión de todas las bases obtenidas. Es evidente que el conjunto obtenido es generador del espacio de bubbles, por lo tanto, si usamos eliminación gaussiana sobre él, obtendremos una base de $\mathcal{B}(G)$.

3.5. Un generador de bubbles elementales

En [1] se presentó un generador de bubbles que cumplía con ciertas propiedades que permiten generar cualquier bubble aunque la suma esté restringida a la obtención de bubbles. Es decir, toda bubble es generada por una suma sucesiva de bubbles, cuyos pasos intermedios también son bubbles. Sin entrar en detalles sobre esta propiedad específica del generador, el resultado también es un generador en el sentido de esta tesis, esto es, sin restringir la suma. Presentamos las nociones necesarias para definir tal generador.

Definición 3.19 Sea G un grafo dirigido y un orden total de sus arcos. Sean dos caminos p y q en G . Se dice que $p <_{\text{lex}} q$ si el largo de p es menor que el largo de q o si siendo del mismo largo la secuencia de arcos de p es lexicográficamente menor que la de q . Si $p <_{\text{lex}} q$ se dice que p es lexicográficamente más corta que q .

De aquí en adelante se supondrá que hay un orden total (arbitrario) de los

arcos, y que por lo tanto define un orden total de los caminos en G .

Definición 3.20 Sea G un grafo dirigido. Sea $B = (p, q)$ a la bubble compuesta por las piernas p y q . Se denota por $l(B)$ (resp. $\mathcal{L}(B)$) la pierna lexicográficamente menor (resp. mayor) de las piernas de B .

Definición 3.21 Sean A y B dos bubbles cualesquiera. Se dice que A es más pequeña que B si una de las siguientes condiciones se cumple:

- (i) $\mathcal{L}(A) <_{lex} \mathcal{L}(B)$;
- (ii) $\mathcal{L}(A) = \mathcal{L}(B)$ y $l(A) <_{lex} l(B)$

Definición 3.22 Sea B una bubble tal que no puede ser descompuesta como la suma de dos bubbles más pequeñas, B se denomina una bubble **simple**.

Definición 3.23 Sea B una (s, t) -bubble, se dirá que B es una bubble **elemental** si

- (i) $l(B)$ corresponde al camino más corto entre s y t , y
- (ii) Para $\mathcal{L}(B) = s, v_1, \dots, v_l, t$, entonces s, v_1, \dots, v_l es el camino lexicográficamente más corto entre s y v_l en G

Al conjunto de las bubbles elementales de G se denota por $\mathbb{E}(G)$.

Se ha demostrado en [1] que toda bubble simple es además elemental y que $|\mathbb{E}(G)| \leq mn$. Además, se demostró que toda bubble puede ser expresada como la suma de una secuencia de bubbles simples, por lo que el conjunto de bubbles simples correspondería a un generador del espacio de bubbles. A raíz de esto, podemos concluir que dentro de las bubbles elementales existe un conjunto generador del espacio de bubbles, cuyo cardinal esta acotado polinomialmente sobre el tamaño y orden del grafo.

En la siguiente figura puede observarse un grafo en el que la bubble compuesta por los nodos v_1, v_3, v_4, v_5, v_6 y v_7 es elemental, más no es simple. Con este pequeño ejemplo podemos concluir que el espacio de bubbles simples es subconjunto propio de $\mathbb{E}(G)$.

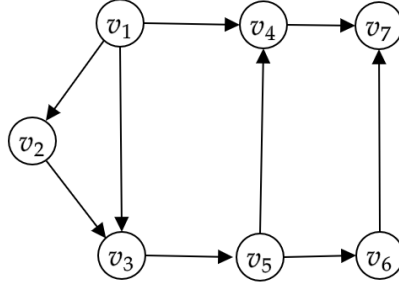


Figura 3.3: Ejemplo de bubble elemental no simple.

En [1] se consideró una operatoria sobre las bubbles en la cual sólo era posible definir la suma si el resultado obtenido correspondía también era bubble. Se descartó esta restricción en el presente trabajo con el objetivo de recuperar las propiedades de espacio vectorial análogas al espacio de ciclos de un grafo y teniendo en cosideración que los resultados relevantes obtenidos en dicha publicación se conservan al realizar esta relajación.

3.6. Base de $\mathcal{B}(G)$ por Eliminacion Gaussiana

Teniendo ya un par de conjuntos generadores del espacio de bubbles, es natural notar entonces que se puede obtener un conjunto linealmente independiente a partir de cada uno de ellos que siga siendo generador, es decir, que sea base de $\mathcal{B}(G)$. Esto es posible aplicando un método de resolución de sistemas de ecuaciones lineales clásico llamado eliminación gaussiana.

Mientras menor sea el tamaño del generador inicial, más eficiente será el proceso de eliminación gaussiana. Se debe tener en cuenta que si se contruye una base para cada subgrafo G_i mediante la aplicación de BFS, la unión de dichas bases corresponde a un conjunto de bubbles elementales, de modo que se sabe que un generador construido de esta forma es de menor tamaño que el generador de bubbles elementales.

3.6.1. Algoritmo con eliminación gaussiana

Se diseñó el siguiente algoritmo, el cual construye el conjunto de bubbles elementales y procede a aplicar eliminación gaussiana para obtener una base compuesta de bubbles del espacio $\mathcal{B}(G)$.

```

Data:  $G = (V, E)$ 
1 begin
2    $S \leftarrow \emptyset;$ 
3   for  $v \in V$  do
4      $T \leftarrow \text{BFS}(G, v);$ 
5     for  $e = (u_1, u_2) \notin T$  do
6       if  $vTu_1 \cap vTu_2 == \emptyset;$  /* Si los caminos son disjuntos
7         */
8         then
9         |  $S \leftarrow S + (vTu_1 + vTu_2 + e);$  /* Se agrega la bubble
10        */
11       end
12     end
13   end
14    $\mathcal{G} \leftarrow \text{Eliminación\_Gaussiana}(S);$ 
15   return  $\mathcal{G}$ 
16 end

```

En este contexto, uTv corresponde al único camino entre los vértices u y v , siguiendo los arcos del árbol T .

La subrutina de eliminación gaussiana se realiza modelando el conjunto S como una matriz en la que cada columna representa una bubble a través de su indicatriz sobre el conjunto E , es decir S es una matriz de m filas y mn columnas, donde cada fila representa un elemento de E y cada columna un elemento de $\mathbb{E}(G)$.

De este modo, el conjunto de columnas linealmente independiente representado por una solución no trivial del sistema $Sx = 0$ corresponde a una base del espacio $\mathcal{B}(G)$. Este sistema puede ser resuelto mediante la aplicación de eliminación gaussiana.

3.6.2. Análisis del algoritmo

Se procederá a demostrar la correctitud del algoritmo, así como a calcular su complejidad. Para poder asegurar que el conjunto de bubbles retornado \mathcal{G} es generador, bastará demostrar que toda bubble elemental puede ser construida a partir de él, pues $\mathbb{E}(G)$ es generador, luego para toda bubble existe una descomposición en bubbles elementales.

Sea $B = (p, q)$ una (s, t) -bubble elemental, luego se sabe que $l(B)$ es el camino más corto entre s y t , así como $\mathcal{L}(B)$ es el camino más corto hasta el vértice previo a t , el cual se llamará v_t . El algoritmo al realizar BFS desde s , considerará ambos caminos al ser los más cortos entre sus extremos y la arista $(v_t, t) \notin T$. Como p y q son disjuntos, B entrará al conjunto S . Esto permite concluir que $\mathcal{E}(G) \subseteq S$ y es fácil notar que toda bubble en S es elemental, por lo que $\mathcal{E}(G) = S$. Esto garantiza que el conjunto S construido por el algoritmo

es en efecto el conjunto de las bubbles elementales.

Es evidente ahora que toda bubble elemental puede generarse a partir de \mathcal{G} . Sea B^* bubble elemental, luego $B \in \mathcal{S}$. Se tiene así dos casos

- $B^* \in \mathcal{G}$ y el resultado es trivial.
- $B^* \notin \mathcal{G}$, luego al momento de realizar el algoritmo la iteración de la eliminación gaussiana sobre la columna representante de B^* , no existía un pivote posible en dicha columna, lo que dictamina la existencia de una descomposición de B^* en las bubbles utilizadas como pivote previamente.

Para finalizar, \mathcal{G} es entonces un conjunto linealmente independiente, capaz de generar toda bubble en $\mathbb{E}(G)$, y por ende, todo elemento de $\mathcal{B}(G)$. Además, la eliminación gaussiana obtiene un conjunto de columnas linealmente independiente dentro de una matriz con m filas, por lo que está asegurado que tiene un cardinal $|\mathcal{G}| \leq m$, luego es lineal sobre la entrada del algoritmo. \square

Para calcular la complejidad del algoritmo hay que notar que este consta de dos partes, la primera de ella corresponde a la construcción del conjunto S , lo que se lleva a cabo mediante una aplicación de BFS por cada vértice del grafo y posteriormente una actualización del conjunto por cada arista no presente en la salida de BFS. Este proceso posee una complejidad de orden $O(n(n+m))$.

La segunda parte corresponde a la aplicación del algoritmo de eliminación gaussiana, cuya complejidad al aplicarse sobre una matriz de r filas y c columnas tiene una complejidad de $O(cr^2)$. En este caso, al modelarse el conjunto generador como una matriz de $m \times nm$, la complejidad de esta parte del algoritmo esta dada por $O(nm^3)$.

Es evidente entonces que la complejidad del algoritmo se encuentra dominada por la eliminación gaussiana y es por ende $O(nm^3)$. Esta complejidad es problemática pues para grafos de tamaño y orden razonable, llega a valores muy elevados. Por ejemplo, el genoma de la bacteria E. Coli-K12, posee alrededor de 4.6 millones de pares de nucleotidos [3], por lo que su grafo de ensamblaje debe poseer un orden similar. Es decir, incluso para genomas pequeños, $O(nm^3)$ es un valor demasiado elevado.

Otro ejemplo más extremo es el caso del grafo completo K_n , para el cual se tiene que

$$|\mathcal{G}(K_n)| = n^2(n-1)$$

por lo que la eliminación Gaussiana realizará alrededor de $n^4(n-1)^3$ operaciones.

Alternativamente, este algoritmo puede modificarse para utilizar un generador a partir de bases construidas desde un conjunto de fuentes. Esto disminuye el tamaño del generador y el conjunto de fuentes puede obtenerse previamente y considerarse como una entrada adicional del algoritmo. Con estas modificaciones, siendo s el cardinal del conjunto fuente, la complejidad de la eliminación gaussiana es de $o(sm^3)$, pues el generador es de tamaño $o(sm)$, por lo que la disminución de la complejidad estará determinada por la razón entre los valores s y n . Es evidente que, dado que las fuentes son nodos del grafo, este valor siempre es menor o igual que 1.

Para simplificar el análisis y comparación de los algoritmos, se acotará superiormente el valor de m por su máximo posible en función de n , que corresponde a un valor del orden de n^2 . Con esto en consideración la complejidad del algoritmo del grafo es $O(n^7)$.

Independientemente de estas dificultades, se puede concluir a partir de este algoritmo, que existen conjuntos linealmente independientes de bubbles, de cardinal lineal sobre el orden y tamaño del grafo G , capaces de generar el espacio de bubbles. Sin embargo, la elevada complejidad que posee este método lleva a cuestionarse si será posible la construcción de un conjunto de similares características sin tener que recurrir a un procedimiento tan costoso como lo es la eliminación gaussiana.

Capítulo 4

Algoritmo topológico para encontrar un generador

En la sección anterior se presentó un algoritmo que a partir de un conjunto generador, construye una base de $\mathcal{B}(G)$ formada exclusivamente por bubbles, vía eliminación gaussiana de los elementos linealmente dependientes.

Se tiene entonces una manera de encontrar una base de $\mathcal{B}(G)$ compuesta en su totalidad por bubbles. El desafío es entonces entender las propiedades topológicas de las bubbles para poder generar un conjunto generador pequeño, ojalá un base, sin tener que recurrir a eliminación gaussiana.

El algoritmo propuesto en este trabajo examina en cada iteración una familia creciente y finita de subgrafos de G , comenzando por un subgrafo 1-fuente y terminando en G . Así, a partir de un conjunto generador del subgrafo 1-fuente inicial, en cada iteración se extiende este conjunto de manera de generar las bubbles del subgrafo dicha iteración. Finalmente lo que se obtiene es una base de $\mathcal{B}(G)$ compuesta en su totalidad de bubbles.

4.1. Descomposición del grafo

Como se mencionó en el capítulo anterior, es posible encontrar un generador compuesto por bubbles de manera sencilla para el caso 1-fuente y 1-destino. Por lo que, en adelante, se supondrá que el grafo no es 1-fuente ni 1-destino. Supondremos entonces que G tiene k fuentes, con $k \geq 2$.

Definamos G_i el subgrafo alcanzable desde la fuente i , con $i \in \{1, \dots, k\}$. Definamos además G_i^* el subgrafo alcanzable desde las primeras i fuentes, con $i \in \{1, \dots, k\}$. Es decir, $G_i^* = \bigcup_{j=1}^i G_j$. Bajo el supuesto que para G_{i-1}^* ya se construyó un conjunto generador β_{i-1} de bubbles compuesto sólo por bubbles, se busca extender este conjunto a un generador β_i que genere las bubbles del grafo G_i^* .

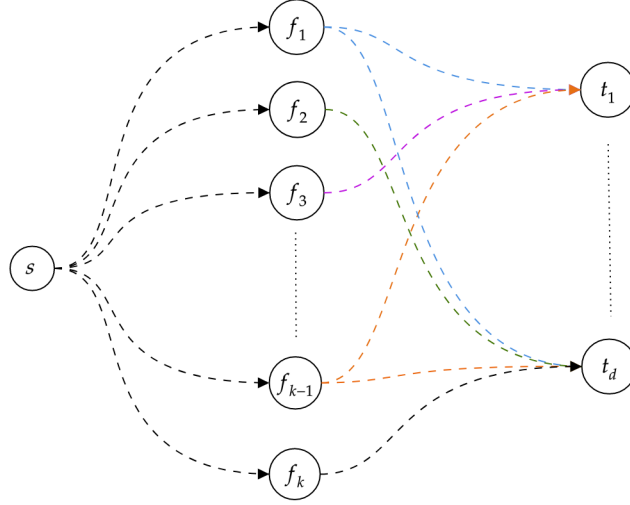


Figura 4.1: Representación general del subgrafo G_i , con sus nodos frontera y destino.

Consideremos el conjunto de nuevas aristas de la iteración i , es decir $E(G_i) \setminus E(G_{i-1}^*)$ y sea ΔG_i el subgrafo inducido por este conjunto de aristas.

Proposición 4.1 ΔG_i es un grafo 1-fuente, donde s_i es una fuente.

DEMOSTRACIÓN. Basta demostrar que para cualquier nodo v de ΔG_i existe un camino entre s_i y v completamente contenido en ΔG_i . Como v está en ΔG_i existe un nodo u tal que (u, v) está en $E(G_i) \setminus E(G_{i-1}^*)$. Como u está en G_i (pues ΔG_i es subgrafo de G_i) existe un camino L entre s_i y u contenido en G_i . Veamos que este camino más (u, v) es camino entre s_i y v completamente contenido en ΔG_i . Supongamos que no. Entonces existe algún arco de ese camino que está en $E(G_{i-1}^*)$ es decir es alcanzable por algún s_j con $j < i$. Pero entonces u y v también son alcanzables por ese s_j con lo que (u, v) pertenecería a $E(G_{i-1}^*)$ lo que es una contradicción. □

Sea T un árbol dirigido recubridor de ΔG_i con raíz en s_i . Sea $\tilde{\beta}_i$ el generador de ΔG_i definido por los arcos de ΔG_i que no están en T .

Podemos clasificar las bubbles de G_i^* en 3 tipos:

- Tipo 1: Bubbles compuestas por arcos en ΔG_i ,
- Tipo 2: Bubbles compuestas por arcos en G_{i-1}^* , y
- Tipo 3: Bubbles que mezclan arcos de ambos ΔG_i y G_{i-1}^* .

Es claro que se pueden generar las bubbles de tipo 1 y 2 con las bubbles de β_{i-1} y $\tilde{\beta}_i$, sólo resta encontrar las bubbles restantes que permitan generar las de tipo 3 para completar la extensión de la base de G_{i-1}^* a G_i^* . El foco del resto del capítulo se centrará en realizar esta tarea. La dificultad principal es que el conjunto de bubbles de tipo 3 necesarias para esa extensión no es fácilmente identificable. Para eso, vamos a buscar identificar propiedades de dependencia entre bubbles, de manera de no incluir bubbles de tipo 3 que pueden ser generadas por una combinación entre las bubbles ya escogidas y los conjuntos $\tilde{\beta}_{i-1}$ y β_i .

Vamos a definir un conjunto de nodos que es esencial a la hora de establecer propiedades de dependencia entre bubbles de tipo 3. Estas bubbles, al tener arcos en ΔG_i y G_{i-1}^* necesariamente tendrán en cada pierna un nodo que separa los arcos entre estos dos conjuntos.

Sea $v \in G_i$, diremos que v es **nodo frontera** de G_i si $v \in G_{i-1}^* \cap \Delta G_i$. Al conjunto de nodos frontera de G_i se denota por \mathcal{F}_i .

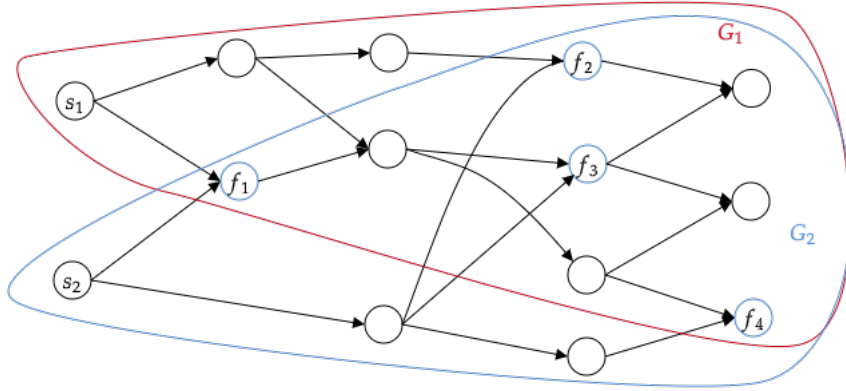


Figura 4.2: G_1 , G_2 y los nodos fronteras del grafo G_2 .

Necesariamente los nodos frontera de G_i corresponden a hojas de T , aunque no necesariamente toda hoja de T es un nodo frontera.

Definición 4.2 Sea $D = \{d_1, \dots, d_l\}$ un conjunto de destinos minimal del grafo G , se define un etiquetado sobre los nodos de G

$$c : G \longrightarrow \mathcal{P}(D)$$

$$c(u) = D \cap \text{Alcanzable}(u)$$

Es decir, $c(u)$ es el conjunto de **destinos alcanzables** por u en G . Extenderemos esta noción también a bubbles. Los destinos alcanzables de una bubble b será la intersección de los destinos alcanzables por sus nodos. Es fácil ver que esto es equivalente al conjunto de destinos alcanzables por su nodo

final. Es decir, si b es una bubble en G_i con nodo final t , entonces $c(b) = c(t)$. Diremos que $c(b)$ es el conjunto de destinos alcanzables de b en G .

Proposición 4.3 *Sea b una bubble en G_i de tipo 3, es decir, tal que no está completamente contenida en ΔG_i ni completamente contenida en G_{i-1}^* . Entonces, su nodo inicio está en $\Delta G_i \setminus \mathcal{F}_i$ y su nodo término está en G_{i-1}^* . Además, si f_1 y f_2 son respectivamente los primeros nodos de cada pierna que pertenecen a G_{i-1}^* entonces ambos son nodos frontera y tales que $f_1 \neq f_2$.*

DEMOSTRACIÓN. Sea $b = (p, q)$ una (s, t) -bubble de G_i de tipo 3.

Supongamos que $s \notin \Delta G_i \setminus \mathcal{F}_i$, entonces $s \in (\Delta G_i)^c \cup \mathcal{F}_i$, luego, o bien $s \in \mathcal{F}_i \subseteq G_{i-1}^*$, o bien $s \in G_{i-1}^* \setminus \Delta G_i \subseteq G_{i-1}^*$. Es decir, $s \in G_{i-1}^*$, pero esto implica que existe un $j < i$ tal que $s \in \text{Alcanzable}(s_j)$ y como $\forall u \in b, u \in \text{Alcanzable}(s)$, por la proposición 3.9, $u \in \text{Alcanzable}(s_j)$, luego b es burbuja en $G_j \subseteq G_{i-1}^*$, por lo que b es de tipo 2. Esto contradice que b sea bubble de tipo 3, por lo que $s \in \Delta G_i \setminus \mathcal{F}_i$.

Por otra parte, sabemos que b es de tipo 3, luego existe un arco $e \in b \cap G_{i-1}^*$, sea $v = t(e)$, es claro que $v \in G_{i-1}^*$, luego existe $j < i$ tal que $v \in \text{Alcanzable}(s_j)$ y como e participa de la bubble b , $t \in \text{Alcanzable}(v)$, se concluye que $t \in \text{Alcanzable}(s_j)$, es decir, $t \in G_{i-1}^*$.

Consideremos ahora f_1 el primer nodo de la pierna q perteneciente a G_{i-1}^* , sea w el nodo anterior, entonces sabemos que $w \notin G_{i-1}^*$, luego el arco $\bar{e} = wf_1 \notin E(G_{i-1}^*)$. Esto último nos indica que $\bar{e} \in E(G_i) \setminus E(G_{i-1}^*)$, por lo que $f_1 \in \Delta G_i$. Se concluye entonces que $f_1 \in \mathcal{F}_i = G_{i-1}^* \cap \Delta G_i$. De manera análoga se prueba que $f_2 \in \mathcal{F}_i$. Resta demostrar que $f_1 \neq f_2$, lo que es fácil de notar, pues si $f_1 = f_2$, la bubble b sería de tipo 1. □

Notar que el resultado anterior permite que una bubble de tipo 3 termine en un nodo frontera, pero ese nodo no puede ser el primer nodo de ambas piernas que está en G_{i-1}^* .

Definición 4.4 *Sea b una bubble en G_i de tipo 3, es decir, tal que no está completamente contenida en ΔG_i ni completamente contenida en G_{i-1}^* . Decimos que f_1 y f_2 son los **nodos de cruce** de b si son respectivamente los primeros nodos de cada pierna que pertenecen a G_{i-1}^* .*

Proposición 4.5 *Sean f_1 y f_2 dos nodos frontera de G_i tales que $f_1 \neq f_2$ y sea d un destino alcanzable por ambos nodos (i.e. $d \in c(f_1) \cap c(f_2) \neq \emptyset$), entonces existe una bubble b de tipo 3 tal que f_1 y f_2 son sus nodos de cruce y d es un destino alcanzable de b .*

DEMOSTRACIÓN. Sea $d \in c(f_1) \cap c(f_2)$, luego existen dos caminos P_1 y P_2 en G_i tales que P_1 va desde f_1 a d y P_2 desde f_2 a d . Notar que ambos caminos están en

G_{i-1}^* . Sea $t \in V(G_{i-1})$ el primer nodo en común entre P_1 y P_2 . Por otro lado sea $s \in V(\Delta G_i)$ el nodo más profundo de T tal que $\{f_1, f_2\} \subseteq \text{Alcanzable}(s)$. Como f_1 y f_2 son hojas de T , s es distinto de ambos nodos frontera y por lo tanto no es hoja. Luego, s no es nodo frontera. Es claro que podemos construir 2 caminos internamente disjuntos desde s hasta t , que pasan por f_1 y f_2 respectivamente, y que forman una bubble en G_i^* .

□

Teorema 4.6 Sea $i \in \{1, \dots, k\}$, dado un par de bubbles b_1, b_2 de tipo 3 en G_i que poseen los mismos nodos de cruce f_1, f_2 y que comparten un destino alcanzable ($c(b_1) \cap c(b_2) \neq \emptyset$), entonces existen bubbles r_1, r_2 en G_{i-1}^* y ℓ_1, ℓ_2 en ΔG_i tales que $b_1 = r_1 + \ell_1 + b_2 + r_2 + \ell_2$.

Antes de ver la demostración formal del teorema, veamos el resultado en el siguiente esquema:

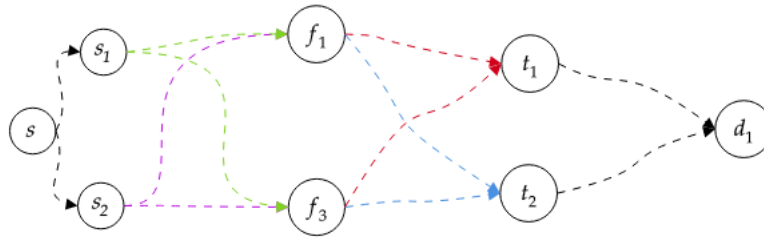


Figura 4.3: Esquema del teorema 4.6.

Si se observa la figura, se puede notar que la (s_1, t_1) -bubble representada por las secciones verde y roja es la suma de la (s_2, t_2) -bubble (representada por las secciones violeta y azul) con otras bubbles completamente contenidas en ΔG_i o en G_{i-1}^* .

DEMOSTRACIÓN. Sean t_1 y t_2 los terminales de las burbujas b_1 y b_2 . Sea $d_1 \in c(b_1) \cap c(b_2)$, consideremos los siguientes nodos:

- $f_{1,r}$ el último nodo en común entre los caminos $f_1 t_1$ en b_1 y $f_1 t_2$ en b_2 .
- $f_{2,r}$ el último nodo en común entre los caminos $f_2 t_1$ en b_1 y $f_2 t_2$ en b_2 .
- \bar{t} el primer nodo en común entre los caminos $t_1 d_1$ y $t_2 d_1$.

Se pueden construir las bubbles:

- $r_1 = f_{1,r} t_1 + t_1 \bar{t} + f_{1,r} t_2 + t_2 \bar{t}$
- $r_2 = f_{2,r} t_1 + t_1 \bar{t} + f_{2,r} t_2 + t_2 \bar{t}$

Es claro que r_1, r_2 son bubbles del grafo G_{i-1}^* y que

$$r_1 + r_2 = f_{1,r}t_1 + f_{1,r}t_2 + f_{2,r}t_1 + f_{2,r}t_2$$

Análogamente se consideran los nodos

- $\bar{s} \in V$ el último nodo en común entre los caminos ss_1 y ss_2 .
- $f_{1,l}$ el primer nodo en comun entre los caminos s_1f_1 y s_2, f_1 .
- $f_{2,l}$ el primer nodo en comun entre los caminos s_1f_2 y s_2, f_2 .

y las bubbles:

- $l_1 = \bar{s}s_1 + s_1f_{1,l} + \bar{s}s_2 + s_2f_{1,l}$
- $l_2 = \bar{s}s_1 + s_1f_{2,l} + \bar{s}s_2 + s_2f_{2,l}$

Se tiene que l_1, l_2 son bubbles del grafo $G_i \setminus E(G_{i-1}^*)$ y que

$$l_1 + l_2 = s_1f_{1,l} + s_2f_{1,l} + s_1f_{2,l} + s_2f_{2,l}$$

Notando que

$$\begin{aligned} b_1 &= s_1f_{1,l} + f_{1,l}f_1 + f_1f_{1,r} + f_{1,r}t_1 + s_1f_{2,l} + f_{2,l}f_2 + f_2f_{2,r} + f_{2,r}t_1 \\ r_1 + r_2 &= f_{1,r}t_1 + f_{1,r}t_2 + f_{2,r}t_1 + f_{2,r}t_2 \\ l_1 + l_2 &= s_1f_{1,l} + s_2f_{1,l} + s_1f_{2,l} + s_2f_{2,l} \\ \text{Sea } \alpha &= l_1 + l_2 + b_1 + r_1 + r_2 \\ \alpha &= s_2f_{1,l} + f_{1,l}f_1 + f_1f_{1,r} + f_{1,r}t_2 + s_2f_{2,l} + f_{2,l}f_2 + f_2f_{2,r} + f_{2,r}t_2 \\ \alpha &= b_2 \end{aligned}$$

Es decir,

$$b_2 = r_1 + \ell_1 + b_1 + r_2 + \ell_2$$

con esto, se da por demostrada la proposición. □

Este resultado implica que, en conjunto con las bubbles en β_{i-1} y $\tilde{\beta}_i$, incluir en el generador una bubble b de tipo 3 con etiquetado $c(b)$ es suficiente para generar todas las bubbles de tipo 3 que tengan los mismos nodos de cruce y un etiquetado no disjunto con $c(b)$. Así, sólo basta incluir una de ellas en el generador β_i .

Teorema 4.7 *Sea $i \in \{2, \dots, k\}$ y sean b_1, b_2 bubbles de tipo 3 en G_i ambas con los mismos nodos de cruce f_1 y f_2 . Si existe $j \in \{1, \dots, i-1\}$ tal que $f_1, f_2 \in \text{Alcanzable}(s_j)$ entonces existen bubbles $\hat{b}_1, \hat{b}_2 \in G_{i-1}^*$ y $\ell_1, \ell_2 \in \Delta G_i$ tal que $b_2 = \hat{b}_1 + \ell_1 + b_1 + \hat{b}_2 + \ell_2$.*

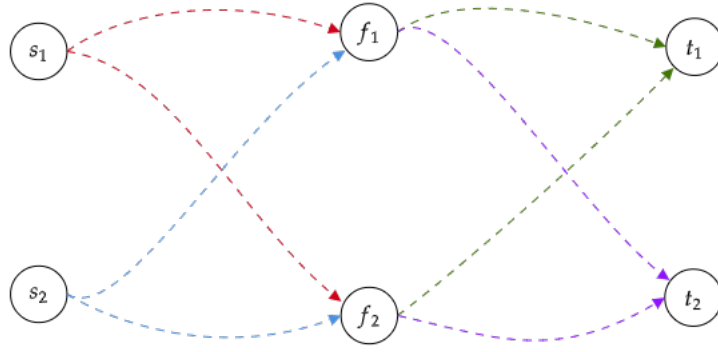


Figura 4.4: Esquema del teorema 4.7.

DEMOSTRACIÓN. l_1 y l_2 son las mismas bubbles construidas en 4.6. Por otra parte, sea $\bar{s} \in G_j$ tal que $f_1, f_2 \in \text{Alcanzable}(\bar{s})$ y $\bar{s}f_1, \bar{s}f_2$ sean caminos disjuntos. Sea t_1, t_2 los nodos terminales de b_1 y b_2 respectivamente, $f_k t_l$ el segmento de la bubble b_l que va de f_k a t_l con $k, l = 1, 2$, se construyen las bubbles en $G_j \subseteq G_{i-1}^*$:

- $\hat{b}_1 = \bar{s}f_1 + f_1t_1 + \bar{s}f_2 + f_2t_1$
- $\hat{b}_2 = \bar{s}f_1 + f_1t_2 + \bar{s}f_2 + f_2t_2$

Que satisfacen que:

$$\begin{aligned}
 b_1 &= s_1f_1 + f_1t_1 + s_1f_2 + f_2t_1 \\
 \hat{b}_1 &= \bar{s}f_1 + f_1t_1 + \bar{s}f_2 + f_2t_1 \\
 \hat{b}_2 &= \bar{s}f_1 + f_1t_2 + \bar{s}f_2 + f_2t_2 \\
 l_1 + l_2 + b_1 &= s_2f_1 + f_1t_1 + s_2f_2 + f_2t_1 \\
 \text{Sea } \alpha &= l_1 + l_2 + b_1 + \hat{b}_1 + \hat{b}_2 \\
 \alpha &= s_2f_1 + f_1t_2 + s_2f_2 + f_2t_2 \\
 \alpha &= b_2
 \end{aligned}$$

Lo que concluye el resultado □

Este resultado nos dice que si un par de nodos frontera del grafo G_i se encuentran simultáneamente en un grafo G_j para $j < i$, entonces para generar todas las bubbles de G_i que tienen ese par de nodos de cruce basta incluir sólo una de ellas en el generador.

Teorema 4.8 Sea $i \in \{1, \dots, k\}$ y sean $F = \{f_1, f_2, f_3\}$ un conjunto de nodos frontera distintos en G_i y d un destino alcanzable por todos (es decir, $d \in c(f_1) \cap c(f_2) \cap c(f_3)$). Entonces existen bubbles b_1, b_2, b_3 que satisfacen las siguientes propiedades:

- sus nodos de cruce son respectivamente $\{f_1, f_2\}$, $\{f_2, f_3\}$ y $\{f_1, f_3\}$,
- $d \in c(b_1) \cap c(b_2) \cap c(b_3)$,
- $b_1 + b_2 = b_3$.

DEMOSTRACIÓN. Como f_1, f_2, f_3 son nodos frontera de G_i , y tienen el destino común d , existen los caminos P_1, P_2 y P_3 tales que parten en s_i , terminan en d , y pasan por f_1, f_2 y f_3 , respectivamente. Sea u_1 el último nodo en comun entre P_1 y P_2 antes de pasar por f_1 o f_2 y v_1 el primer nodo en comun entre P_1 y P_2 después de pasar por f_1 o f_2 , respectivamente. Es claro que el par de subcaminos de P_1 y P_2 que van desde u_1 a v_1 corresponden a una bubble, b_1

De manera análoga, existen u_2, v_2 , que se encuentran en los caminos P_2 y P_3 y tales que el par de caminos que van de u_2 a v_2 construyen una bubble, b_2 .

Es evidente que los nodos de paso de b_1 y b_2 son $\{f_1, f_2\}$ y $\{f_2, f_3\}$ respectivamente, y que, como ambos nodos terminales se encuentran en caminos dirigidos a d , $d \in c(b_1) \cap c(b_2)$. Sin perdida de generalidad, se puede decir que u_1 se encuentra antes que u_2 en el camino P_2 . Se tienen entonces dos casos:

1. v_1 es previo a v_2 en P_2 :

En este caso, el camino que $u_2 P_2 v_1$ está totalmente contenido en el camino $u_2 P_2 v_2$. Considérese entonces la suma $b_1 + b_2$. Teniendo en cuenta que:

$$b_1 = u_1 P_1 v_1 + (u_1 P_2 u_2 + u_2 P_2 v_1)$$

$$b_2 = (u_2 P_2 v_1 + v_1 P_2 v_2) + u_2 P_3 v_2$$

se puede notar que:

$$b_1 + b_2 = (u_1 P_1 v_1 + v_1 P_2 v_2) + (u_1 P_2 u_2 + u_2 P_3 v_2)$$

Ahora, notando que $(u_1 P_1 v_1 + v_1 P_2 v_2)$ es un camino entre u_1 y v_2 , $(u_1 P_2 u_2 + u_2 P_3 v_2)$ tambien es un camino entre u_1 y v_2 , y que ambos caminos son disjuntos por su construcción, se define $b_3 = b_1 + b_2$ que corresponde a una bubble con nodos de cruce $\{f_1, f_3\}$ y nodo terminal v_2 , lo que implica que $d \in c(b_3)$

2. v_1 es posterior a v_2 en P_2 :

Del mismo modo, con la salvedad que ahora el camino $u_2 P_2 v_2$ esta totalmente contenido en el camino $u_2 P_2 v_1$, se considera la suma $b_1 + b_2$. Esta

vez, se tienen las igualdades:

$$b_1 = u_1 P_1 v_1 + (u_1 P_2 u_2 + u_2 P_2 v_2 + v_2 P_2 v_1)$$

$$b_2 = u_2 P_2 v_2 + u_2 P_3 v_2$$

Por lo que

$$b_1 + b_2 = u_1 P_1 v_1 + (u_1 P_2 u_2 + u_2 P_3 v_2 + v_2 P_2 v_1)$$

Lo que de la misma forma corresponde a un bubble b_3 , la cual cumple que $d \in C(b_3)$ y tiene nodos de cruce $\{f_1, f_3\}$.

Así, en ambos casos se puede construir una bubble b_3 , tal que tiene nodos de cruce $\{f_1, f_3\}$, $d \in c(b_3)$ y $b_3 = b_1 + b_2$, lo que concluye el resultado \square

Corolario 4.9 *Sea d un destino alcanzable y sea $F = \{f_1, f_2, \dots, f_\ell\}$ el conjunto de todos los nodos frontera de G_i que pueden alcanzar a d con $\ell \geq 2$. Entonces existe un conjunto β_d de $\ell - 1$ bubbles de tipo 3 tal que $\tilde{\beta}_i \cup \beta_{i-1} \cup \beta_d$ generan cualquier bubble de tipo 3 que alcanza a d en G_i .*

DEMOSTRACIÓN. Sea $E_F \subseteq F^2$ una familia de pares de nodos frontera tal que el grafo (F, E_F) es un árbol, para cada elemento $e = f_p f_q$ de E_F , se elige una bubble b_{pq} de tipo 3 que tenga como nodos de cruce f_p y f_q tal que $d \in c(b_{pq})$, cuya existencia esta garantizada por 4.5, se define $\beta_d = \{b_{pq} | (f_p, f_q) \in E_F\}$. Es claro que $|\beta_d| = |F| - 1 = \ell - 1$

Sea b una bubble de tipo 3 tal que $d \in c(b)$. Sean f_p, f_q los nodos de cruce de b , si el par $(f_p, f_q) \in E_F$, entonces, por el teorema 4.6 se sabe que b puede ser generada a partir de b_{pq} y bubbles contenidas en $\tilde{\beta}_i \cup \beta_{i-1}$.

Por otra parte, si $(f_p, f_q) \notin E_F$, se sabe que la arista (f_p, f_q) genera un único ciclo C en el grafo (F, E_F) . Se demuestra a continuación inductivamente sobre el largo de C , que el conjunto $\beta_C = \{b_e | e \in C \cap E_F\} \subseteq \beta_d$, puede generar a b con la participación de bubbles de $\tilde{\beta}_i \cup \beta_{i-1}$.

Si C tiene largo 3, participan 3 nodos frontera, f_p, f_q y un tercer nodo f_r . Aplicando el teorema 4.8 existen $\tilde{b}_{pr}, \tilde{b}_{qr}, \tilde{b}_{pq}$ de tipo 3 que alcanzan d tales que $\tilde{b}_{pq} = \tilde{b}_{pr} + \tilde{b}_{qr}$. Ahora, aplicando el teorema 4.6 sobre las bubbles b_{pr} y b_{qr} , se generan las bubbles $\tilde{b}_{pr}, \tilde{b}_{qr}$, sumándolas, se obtiene las bubble \tilde{b}_{pq} , y aplicando nuevamente el teorema 4.6, a partir de \tilde{b}_{pq} se puede generar la bubble b . Es evidente entonces que a partir de $\beta_C \cup \tilde{\beta}_i \cup \beta_{i-1}$ se puede generar b .

Sea $n > 3$ el largo de C , suponiendo ahora que para ciclos de largo $n - 1$, se puede generar la bubble en cuestión, sea f_r el nodo vecino a f_p en E_F que participa en C , como E_F es acíclico, se sabe que la arista $(f_q, f_r) \notin E_F$, y el largo del ciclo generado por dicha arista es $n - 1$, luego existe una forma de generar todas las bubbles de tipo 3 que tienen por nodo de cruce f_q y f_r y alcanzan d , utilizando bubbles de $\beta_i \cup \beta_{i-1} \cup \beta_d$. Realizando una aplicación de los teoremas 4.6 y 4.8 de manera idéntica al caso base, se concluye el resultado. \square

4.2. Eligiendo bubbles

Recordemos que en la iteración i estamos bajo el supuesto que ya se construyó un conjunto generador β_{i-1} para las bubbles de G_{i-1}^* y un generador $\tilde{\beta}_i$ para las bubbles de ΔG_i , ambos generadores compuesto sólo por bubbles. Queremos encontrar un conjunto $\hat{\beta}_i$ de bubbles de tipo 3 tal que $\beta_i = \beta_{i-1} \cup \tilde{\beta}_i \cup \hat{\beta}_i$ es un generador de todas las bubbles de G_i^*

Para poder hacer una selección de las bubbles que ingresan a $\hat{\beta}_i$, intentaremos hacerlo de manera de que este conjunto sea pequeño, es decir, evitando incluir bubbles linealmente dependientes. Los resultados de la sección anterior nos aseguran que:

- **Observación 1:** Si $\beta_{i-1} \cup \tilde{\beta}_i \cup \hat{\beta}_i$ puede generar una bubble de tipo 3 con nodos de cruce $F = \{f_1, f_2\} \subseteq \mathcal{F}_i$ y con d un destino alcanzable, entonces puede generar todas las bubbles con los mismos nodos de cruce y con d entre sus destinos alcanzables.
- **Observación 2:** Si $\beta_{i-1} \cup \tilde{\beta}_i \cup \hat{\beta}_i$ puede generar una bubble de tipo 3 con nodos de cruce $F = \{f_1, f_2\} \subseteq \mathcal{F}_i$ y F es un conjunto alcanzable desde un origen s_j con $j \in \{1, \dots, i-1\}$, entonces puede generar todas las bubbles con los mismos nodos de cruce (cualquiera sea su conjunto de destinos alcanzables).
- **Observación 3:** Si $\beta_{i-1} \cup \tilde{\beta}_i \cup \hat{\beta}_i$ puede generar dos bubbles B_1, B_2 de tipo 3 con nodos de cruce $\{f_1, f_2\}$ y $\{f_2, f_3\}$ y tales que d es un destino alcanzable de ambas, entonces puede generar todas las bubbles con nodos de cruce $\{f_1, f_3\}$ y destino alcanzable d .

Es claro que en la medida que incluimos bubbles en $\hat{\beta}_i$ debemos considerar los nodos de cruce y los destinos alcanzables no sólo de las bubbles incluidas en $\hat{\beta}_i$ sino también de las bubbles *generadas* por ellas, pues éstas pueden llegar a disminuir el número final de bubbles necesarias. Por ejemplo, según uno de los resultados, incluir una bubble B con $\{f_1, f_2\}$ como nodos de cruce que tenga a d como único destino alcanzable es suficiente para generar todas las bubbles que, como ella, tengan a F como nodos de cruce y a d como destino alcanzable. Sin embargo, si una de esas bubbles generadas también tiene a d' como destino alcanzable, entonces incluir B permite generar también todas las bubbles con los mismos nodos de cruce pero con d' como conjunto alcanzable. Esto aunque

B no tenga a d' como conjunto alcanzable.

Basados en este ejemplo vamos a construir una relación de equivalencia entre los destinos d y d' .

Definición 4.10 Sea $\{f_1, f_2\} \subseteq \mathcal{F}_i$ dos nodos del conjunto frontera de la iteración i , y sea $D = c(f_1) \cap c(f_2)$ el conjunto de los destinos alcanzables por ambos nodos. Definimos en D la relación \sim_{f_1, f_2} entre dos destinos alcanzables $d_1, d_2 \in D$ de la siguiente manera: $d_1 \sim_{f_1, f_2} d_2$ si se cumple al menos una de las siguientes:

- $\{f_1, f_2\}$ es un conjunto alcanzable desde un origen s_j con $j \in \{1, \dots, i-1\}$
- existe una bubble B de tipo 3 con nodos de cruce $\{f_1, f_2\}$ y tales que d_1 y d_2 son ambos destinos alcanzables por B .

Claramente \sim_{f_1, f_2} es refleja y simétrica. La relación definida nos ayuda a identificar bubbles que no es necesario incluir en el generador según lo descrito en las observaciones 1 y 2, puesto que al generar una bubble de tipo 3 con destino d se generan todas las bubbles que tengan los mismos nodos de cruce y algún destino relacionado con d . A su vez, las bubbles generadas pueden generar otras bubbles con destinos relacionados a ellos, y así sucesivamente. Esto implica que podemos considerar la clausura transitiva de la relación para definir clases de destinos equivalentes.

Definición 4.11 Sea $\{f_1, f_2\} \subseteq \mathcal{F}_i$ dos nodos del conjunto frontera de la iteración i , definimos la relación \sim_{f_1, f_2} como la clausura transitiva de la relación \sim_{f_1, f_2} .

Lo interesante de esta relación de equivalencia es que, si consideramos las bubbles con $\{f_1, f_2\} \subseteq \mathcal{F}_i$ como nodos de cruce, basta con generar una bubble que tenga d como destino alcanzable para generar todas las bubbles con destino alcanzable en la clase de equivalencia de d .

Ahora bien, esta relación no considera la Observación 3, que relaciona la generación de bubbles con distintos nodos de cruce. Para poder integrar esta información vamos a definir un multigrafo que considera los conjuntos de paso, destinos alcanzables y clases de equivalencias.

Definición 4.12 Sea \mathcal{F}_i el conjunto frontera de la iteración i . Definimos \hat{G}_i el multigrafo cuyos nodos corresponden a \mathcal{F}_i y cuyo conjunto de aristas \hat{E}_i es tal que, dado un par $f_1, f_2 \in \mathcal{F}_i$ hay una arista entre ellos por cada clase de equivalencia de la relación \sim_{f_1, f_2} . Es decir, hay tantas aristas entre f_1 y f_2 como clases de equivalencia de \sim_{f_1, f_2} . Dada una arista $e \in \hat{E}_i$, la etiqueta $\mathcal{L}(e)$ denota la clase de equivalencia asociada a la arista e .

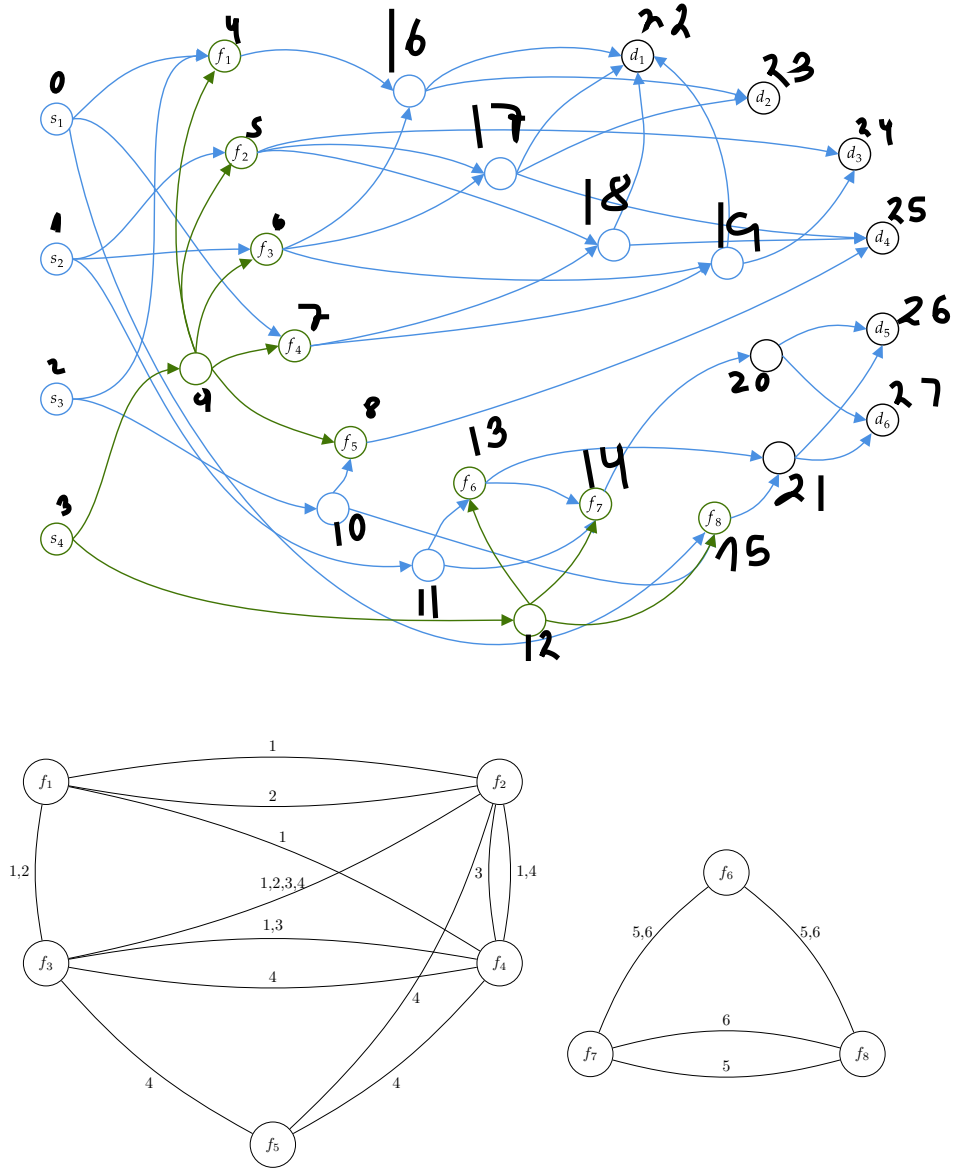


Figura 4.5: Ejemplo de construcción del multigrafo \overline{G} .

Es claro por construcción, que para cada bubble B de tipo 3 de la iteración i que tenga a $\{f_1, f_2\}$ como nodos de paso existe una y sólo una arista e del multigrafo que conecta a f_1 con f_2 y cuya etiqueta contiene a todos los destinos alcanzables por B . Diremos que esta arista *representa* a B . Es claro además que B puede representar a varias bubbles.

Además, gracias a las observaciones 1 y 2, si consideramos el conjunto de todas las bubbles representadas por una arista del multigrafo, es claro que basta que $\beta_{i-1} \cup \tilde{\beta}_i \cup \hat{\beta}_i$ pueda generar una de ellas para generarlas todas. Así diremos que $\beta_{i-1} \cup \tilde{\beta}_i \cup \hat{\beta}_i$ *genera la arista* e si puede generar una bubble representada por e (y por lo tanto todas las representadas por la misma arista).

Antes de explicar como considerar la Observación 3 en el multigrafo, vea-

mos una propiedad de las aristas de éste que comparten a un destino d en su etiqueta.

Proposición 4.13 *Sea $d \in D_i$, un destino de la iteración i . En el multigrafo \hat{G}_i el conjunto de aristas que tienen a d en su etiqueta, $\hat{E}_{i,d} := \{e \in \hat{E}_i; d \in \mathcal{L}(e)\}$, es un clique.*

DEMOSTRACIÓN. Por definición de la relación, los únicos nodos del multigrafo que pueden tener arcos incidentes con d en su etiqueta son los correspondientes a nodos en \mathcal{F}_i que pueden alcanzar al destino d en el grafo original. Vemos que entre cada par de esos nodos hay un y sólo un arco con d en su etiqueta.

Por la proposición 4.5, para cada par de nodos f_1, f_2 de \hat{G}_j tales que $d \in \text{Alcanzable}(f_1) \cap \text{Alcanzable}(f_2)$ siempre existe una bubble con d como destino alcanzable. Así, existe una arista $e \in \hat{E}_j$ entre f_1 y f_2 tal que $d \in \mathcal{L}(e)$. Además, por la construcción del multigrafo, esta arista es única, puesto que las etiquetas de dos aristas paralelas siempre tienen intersección vacía por ser clases de equivalencia de la misma relación. Luego el conjunto $\hat{E}_{j,d}$ genera el grafo completo sobre los nodos $\hat{V}_{j,d} := \{v \in \hat{V}_j; d \in \text{Alcanzable}(v)\}$. □

Notar que gracias a la observación 3, en la iteración i podemos prescindir de incluir en el generador una bubble con cruce $\{f_1, f_3\}$ y destino d si ya se pueden generar dos bubbles con destino d y conjuntos de cruce $\{f_1, f_2\}$ y $\{f_2, f_3\}$ respectivamente para algún f_2 en la frontera. Esto en términos del multigrafo definido, sucede cuando existe un ciclo de largo tres donde todas sus aristas están etiquetadas por d . Generar las bubbles representadas por dos de sus arcos generan también las bubbles del arco restante.

Es claro que se puede extender esta noción a ciclos de largo n , donde es suficiente generar sólo $n - 1$ bubbles correspondientes a $n - 1$ aristas del ciclo para generarlo completo.

Antes de formalizar esta idea, notar que cada vez que existe un camino etiquetado con un destino d también existe un arco etiquetado con d entre los extremos del camino, puesto que, por propiedad 4.13 dado un destino d , las aristas del multigrafo con d en su etiqueta forman un subgrafo completo, es decir, un clique.

Proposición 4.14 *Sea E_β un conjunto de aristas del multigrafo \hat{G}_i . Sean $e_1 = (u, v)$ y $e_2 = (u, w)$ dos aristas adyacentes en u generadas por $E_\beta \cup \beta_{i-1} \cup \tilde{\beta}_i$ y d un destino tal que $d \in \mathcal{L}(e_1) \cap \mathcal{L}(e_2)$. Entonces existe una única arista e_3 entre v y w que tiene a d en su etiqueta. Además, e_3 también es generada por $E_\beta \cup \beta_{i-1} \cup \tilde{\beta}_i$.*

DEMOSTRACIÓN. Este resultado es consecuencia directa del teorema 4.8. Basta

notar que e_1 , e_2 y e_3 representan todas las bubbles que tienen nodos de cruce u, v , u, w y v, w respectivamente, en particular las obtenidas al aplicar dicho resultado a estos pares de fronteras. Es así como se puede generar una bubble que llega a d con nodos de cruce v, w , y por ende, se puede generar toda su clase de equivalencia.

□

Esta propiedad, junto con la Observación 3, nos dice que dado un destino d , tomar una bubble por cada arista de cualquier árbol T recubridor de $\hat{E}_{i,d}$ en el multigrafo será suficiente para generar todas las bubbles que alcanzan a d en la iteración i .

Sin embargo, como las aristas pueden estar etiquetadas con más de un destino, *incluir* en el generador una bubble por cada arista de un árbol recubridor y hacerlo para *cada* destino alcanzable puede terminar construyendo un generador mucho más grande de lo necesario. Efectivamente, al generar las bubbles de un destino estamos también generando indirectamente bubbles de otros destinos.

Una primera idea sería ir destino por destino completando sólo las aristas necesarias para completar un árbol recubridos del grafo $\hat{E}_{i,d}$. Sin embargo, con esta estrategia igual podríamos agregar aristas de más. Considérese un grafo \hat{E}_i , como en el siguiente ejemplo

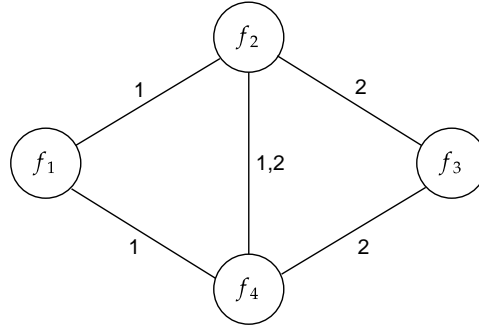


Figura 4.6: Ejemplo grafo \hat{E}_i .

Al considerar un árbol recubridor para el subgrafo $\hat{E}_{i,2}$, se incluye en el generador la arista entre f_2 y f_4 etiquetada con $(1, 2)$, la cual ya puede ser generada con las aristas incluidas en la base dada por un árbol recubridor de $\hat{E}_{i,1}$.

Así proponemos la siguiente subrutina para encontrar las bubbles que agregaremos a β_i en la iteración i del algoritmo general. La idea es iterar sobre todas las aristas verificando para cada una de ellas si ya es generada por $\beta_i \cup \beta_{i-1} \cup \tilde{\beta}_i$ o si debe ser incluida en β_i . Esto se realiza revisando cada arista incidente a un nodo, y verificando si la arista no comparte etiquetado con las aristas ya

revisadas por el algoritmo en iteraciones anteriores incidentes a alguno de sus extremos. Una vez revisadas todas las aristas del nodo, se pasa a algún nodo vecino a algún nodo ya revisado, de este modo, el multigrafo se recorre de forma conexa y sin dejar aristas sin revisar antes de pasar al siguiente nodo. Gracias a esto, verificar que una arista $e = (a, b)$ ya es generada es bastante simple. Sólo debemos mirar si existe algún d en la etiqueta de e que también aparezca en algún arco ya generado incidente a a y en algún arco ya generado incidente a b , gracias al resultado 4.14.

3 Algorithm 1: Subrutina Extension

Data: multigrafo $\hat{G} = (\hat{V}, \hat{E})$, conjunto de destinos alcanzables D ,
función etiquetado $L : E \rightarrow \mathcal{P}(D)$, nodo inicial $a_0 \in \hat{V}$

```

1 begin
2    $V' = \emptyset$ ; # nodos examinados
3    $E' = \emptyset$ ; # arcos generados
4    $a = a_0$ ; # nodo inicial
5    $T = \emptyset$ ; # arcos representando conjunto generador
6   while  $V' \neq \hat{V}$  do
7      $S = E(a) \setminus E'$ ; # arcos no generados adyacentes a  $a$ 
8     for  $e = (a, b) \in S$  do
9        $A = L(E(a) \cap E')$ ; # destinos arcos generados adyacentes a  $a$ 
10       $B = L(E(b) \cap E')$ ; # destinos arcos generados adyacentes a  $b$ 
11      if  $A \cap B \cap L(e) = \emptyset$  then
12         $T = T + e$ ;
13      end
14       $E' = E' + e$ ;
15    end
16     $V' = V' + a$ ;
17    if  $d(V') \neq \emptyset$  then
18      Elegir nuevo  $a \in d(V')$ ;
19    else
20      Elegir nuevo  $a \in \hat{V} \setminus V'$ ;
21    end
22    # Se actualiza  $a$  con cualquier nodo que esté conectado a  $V'$  por
    una arista revisada, o un nodo de una nueva componente
    conexas si no hay nodos de ese tipo.
23  end
24   $B = \text{Construir\_Bubbles}(T)$ ; # Subrutina que crea una bubble
    adecuada para cada representante en  $T$ 
25  return  $B$ 
26 end

```

Proposición 4.15 *El conjunto T construido por el algoritmo es generador del multigrafo*

DEMOSTRACIÓN. Supongamos que en una iteración del algoritmo el conjunto E' es efectivamente generado por el conjunto T calculado en ese momento. Veamos que, si el siguiente arco (a, b) no es ingresado, es porque efectivamente puede ser generado por T .

Supondremos que e no ingresa a T , $e = (a, b)$, y sin pérdida de generalidad supondremos que el algoritmo llegó a él examinando el nodo a . Como $e \notin T$, se sabe que $A \cap B \cap \mathcal{L}(e) \neq \emptyset$. Es decir, existe un destino $d_0 \in \mathcal{L}(e)$ que está en la etiqueta de un arco (v, a) adyacente a a y en la etiqueta de un arco

(b, w) adyacente a b , ambos arcos en E' . Esto quiere decir, en particular, que al menos uno de los nodos b y w ya fue examinado antes que a . Pero supusimos que llegamos a e desde a por lo que b aún no ha sido examinado. Así, w ya fue examinado por el algoritmo. Además, como el subgrafo $\hat{E}_{i,d}$ es un clique, debe existir una arista e' entre w y a que tiene a d_0 en su etiqueta. Como w fue examinado, e' necesariamente está en E' . Este arco junto con (b, w) aseguran, por Teorema 4.14, que e es generado por T . □

4.3. Algoritmo general

Esto permite concluir que toda bubble representada en el grafo \hat{G}_j , puede ser obtenida como una combinación lineal de bubbles representadas por aristas en T y bubbles ya presentes en el conjunto β . Considerando entonces los conjuntos β_{j-1} , β_j y las bubbles representadas por el resultado del algoritmo T , se puede construir un nuevo generador de bubbles para el grafo G_j , el cual se denotará por β_j .

Ya desarrollada esta subrutina para extender la base generada previamente, se diseña el siguiente algoritmo para la creación de una base del espacio de bubbles de un grafo G . Se consideran como entradas del método: los nodos y arcos del grafo $G = (V, E)$; el conjunto de nodos fuente y destino, S y D ; y las funciones que a cada nodo le asignan las fuentes que lo alcanzan y los destinos alcanzados, $c : V \rightarrow \mathcal{P}(D)$ y $\mathcal{S} : V \rightarrow \mathcal{P}(S)$. Estas últimas funciones pueden obtenerse previamente a partir del grafo, mediante una aplicación de algoritmos conocidos de búsqueda en grafos.

Algorithm 2: Algoritmo General

Data: grafo $G = (V, E)$, fuentes $S \subseteq V$, destinos $D \subseteq V$, funciones $S : V \rightarrow \mathcal{P}(S)$ y $c : V \rightarrow \mathcal{P}(D)$.

Result: β , conjunto de burbujas generadoras

```
1 Begin
2    $N\_OLD = \emptyset$ ; #Nodos visitados en alguna iteración
3    $\beta = \emptyset$ ;
4   for  $s \in S$  do
5        $[F, T, \beta_s] = BFS\_Modificado(s)$ ; #Crea  $T$  arbol generador de  $\Delta G$ ,  $F$  nodos frontera, y una base para  $\Delta G$ 
6        $\beta = \beta + \beta_s$ ; #Actualiza conjunto objetivo
7        $\hat{E} = \emptyset$ ; #Arcos del multigrafo  $\hat{G}$ 
8       for  $f \in F$  do
9            $T2 = Alcanzable(f)$ ; #Conjunto de nodos alcanzables desde  $f$ ;
10          for  $g \in F$ ,  $index(f) < index(g)$  do
11              #Nodos frontera posteriores a  $f$  en el listado de  $F$ 
12              if  $(s \neq \min S(f) \cap S(g))$  then
13                  # $f$  y  $g$  comparten fuente previa
14                   $\hat{E} = \hat{E} + [(fg, c(f) \cap c(g))]$ ; # Se crea un único arco entre  $f$  y  $g$  con etiqueta  $c(f) \cap c(g)$ 
15              else
16                   $Construccion\_Arcos$ ; #Construye la relacion  $\sim_{f,g}$  y los arcos entre  $f$  y  $g$ 
17              end
18          end
19      end
20       $\hat{G} = (F, \hat{E})$ ;
21       $Ext = Subrutina\_Extension(\hat{G}, D, L, F[1])$ ;
22       $\beta = \beta + Ext$ ; #Se actualiza el conjunto objetivo.
23  end
24 end
```

Handwritten annotations: A vertical line on the left with arrows pointing to lines 5, 16, and 21, labeled 1, 2, and 3 respectively. A curved arrow connects line 14 to line 16. A small circle is at the end of line 16.

1

Algorithm 3: BFS_Modificado

Data: s nodo raíz del árbol a generar

```

1 begin
2    $N\_V = [s]$ ; # Nodos visitados, se inicia con raíz  $s$ 
3   for  $u \in N\_V$  do
4      $OUT = N_+(u)$ ; # Vecinos de  $u$ 
5     for  $v \in OUT$  do
6       if  $(v \in N\_OLD \ \& \ v \notin F)$  then F not DEFINED prev
7         #  $v$  fue visitado por un source anterior y aún no se
          agrega a  $F$ 
8          $F = F + [v]$ ; # Se agrega  $v$  a  $F$  → Vertex
9          $T = T + [uv]$  # Se agrega  $uv$  a  $T$ ; → Edge? Camino?
10        else if  $(v \notin N\_OLD \ \& \ v \notin N\_V)$  then
11          #  $v$  no ha sido visitado aún
12           $N\_V = N\_V + [v]$  # Se agrega  $v$  a los nodos visitados
13           $T = T + [uv]$  # Se agrega  $uv$  a  $T$ ;
14        else / Ya se ha visitado  $v$ 
15          → #  $uv$  genera un ciclo en  $T$ 
16           $B = sTu + sTv + uv$ ; # Se contruye la bubble generada
              por  $uv$   $sTu$  = path that goes from  $s$  to  $u$  that is contained in  $T$ 
17          →  $\beta = \beta + [B]$  # Se agrega la bubble al generador;
18        end
19      end
20       $N\_OLD = N\_OLD + N\_V$ ; # Se actualiza el conjunto de nodos
        ya visitados por sources
21    end
22 end

```

Esta subrutina lo que realiza es la construcción de un árbol mediante BFS desde el nodo s , que tiene por hojas los nodos frontera. Con ello, reconoce los nodos fronteras y extiende el conjunto β con bubbles de tipo 1.

2 Algorithm 4: Construcción_Arcos

```

1 begin
2    $N\_V = [g]$ ; #Se inicia en  $g$ , nodos visitados
3    $T3 = \emptyset$ ; #Arcos recorridos
4    $R = \emptyset$ ; #Etiquetas de nodos alcanzables por  $f$  y  $g$ ?
5   for  $u \in N\_V$  &  $u \notin T2$  do / for  $u$  in  $N\_V$  and  $u$  not in  $T2$ 
6      $.OUT = N_+(u)$ ; #vecinos de salida de  $u$ 
7     for  $v \in OUT$  do .
8       if  $(v \notin N\_V)$  then
9          $T3 = T3 + [uv]$ ;
10        if  $v \in T2$  then
11           $R = R + [c(v)]$ ; # $v$  es alcanzable por ambos nodo
12          frontera, se guarda su etiquetado
13        end
14         $.N\_V = N\_V + [v]$ ; #Actualiza nodos visitados
15      end
16    end
17  end
18  for  $q \in R$  do
19     $r = q$ ;
20     $R = R - [q]$ ; #Se remueve  $q$  de  $R$ 
21    for  $p \in R$  do
22      if  $p \cap r \neq \emptyset$  then
23         $r = r \cup p$ ; #Si  $r$  tiene algun elemento en comun con  $p$  se
24        actualiza  $r$  uniéndolo con  $p$ 
25         $R = R - [p]$ ; #Se remueve  $p$  de  $R$ 
26      end
27    end
28     $E\_HAT = E\_HAT + [(fg, r)]$ ; #Se crea la arista  $fg$  con
29    etiqueta  $r$ 
30  end
31 end

```

• Cada etiqueta representa la equivalencia entre espacios, que parte desde un nodo frontera y llegan a nodos destinos comunes

. Esta subrutina encuentra todos los nodos que corresponde a un primer encuentro entre $Alcanzable(f)$ y $Alcanzable(g)$, con sus etiquetados crea las clases de equivalencia de la relación $\sim_{f,g}$ y construye una arista por cada una de ellas entre los nodos f y g .

La complejidad de este algoritmo se encuentra dominada principalmente por el proceso de construcción del grafo \hat{G} , cuya complejidad es $o(n^5)$, con lo que el algoritmo se ejecutará en $o(sn^5)$, con s el número de fuentes del grafo. Del mismo modo, considerando que s no puede ser mayor a n , la complejidad del algoritmo es $O(n^6)$.

Estandarizadas las complejidades de los algoritmos a una variable común, se pueden comparar entre sí. Es así como se puede concluir que el método topológico es más eficiente, pues su complejidad es al menos un orden de magnitud menor que la del método por eliminación gaussiana.

Capítulo 5

Discusión

De acuerdo a lo planteado, se logró efectivamente el objetivo principal de construir una base del espacio de bubbles a partir del conjunto de bubbles elementales del grafo.

En este contexto, se desarrolló el algoritmo presentado en 3.6.1, el cual recurre a un proceso de eliminación gaussiana. Esta implementación posee una elevada complejidad de $o(nm^3)$, lo que implica que su aplicación en escenarios reales sería inviable debido al prolongado tiempo de ejecución requerido. No obstante, este algoritmo sí nos permite garantizar la existencia de este tipo de bases.

A raíz de esto, se vuelve esencial la búsqueda de métodos alternativos para construir este tipo de bases de manera menos costosa. Se estudió así la relación de dependencia existente entre las bubbles del grafo a partir de sus propiedades topológicas. Esto es, la equivalencia entre burbujas cuyos inicios comparten un nodo fuente previo, y cuyos terminales convergen hacia un destino común. Como resultado se diseñó un algoritmo que construye una base con las propiedades buscadas a partir de los subgrafos generados desde cada nodo fuente, mediante la selección de una bubble representante para cada conjunto de bubbles equivalentes entre sí.

Si bien este segundo enfoque no realiza en ningún momento eliminación gaussiana, en su actual implementación posee una complejidad comparable ($o(sn^5)$) al primer algoritmo que sí la utiliza. Por esto, la ejecución del segundo método tampoco sería realizable sobre la totalidad del grafo de ensamblaje de un caso real.

Adicionalmente a lo anterior, se comparó la complejidad teórica de ambos

algoritmos y se determinó que el método topológico posee una complejidad un orden de magnitud menor que el método gaussiano. Este resultado debe ser verificado mediante la implementación de ambos métodos y la comparación de los tiempos de ejecución en casos reales.

Por último, el siguiente desafío es mejorar la implementación del segundo método, ya que actualmente la elevada complejidad viene dada por la construcción del grafo \hat{G} . Una construcción más eficiente de dicho objeto y la utilización de mejores estructuras de datos debería reducir de forma considerable la complejidad del algoritmo, aumentando considerablemente el número de casos en que sería más eficiente que el primer método por eliminación gaussiana.

Conclusión

Realizado este trabajo, se puede concluir que existen bases del espacio de bubbles de un grafo de ensamblaje compuestas en su totalidad por bubbles, y que existen distintos métodos para su construcción.

Uno de estos métodos requiere la construcción de un generador previo, y mediante el uso de eliminación gaussiana, construye un conjunto linealmente independiente.

Por otra parte, existe un algoritmo que utilizando propiedades topológicas del grafo, construye alguna de estas bases del espacio de bubbles.

Ambos métodos poseen elevadas complejidades, por lo que su aplicabilidad a casos reales es cuestionable, sin embargo, el método topológico posee una complejidad un orden de magnitud menor que el método gaussiano. Se espera que mediante la utilización de mejores estructuras de datos se reduzca considerablemente su complejidad.

Por último, los métodos son aplicables a regiones parciales del grafo, para los que el tiempo de ejecución se encuentre en rangos razonables.

Anexo 1: Ejemplo de aplicación

Se ejemplificará para dar a entender mejor el algoritmo mediante una aplicación sencilla sobre el siguiente digrafo.

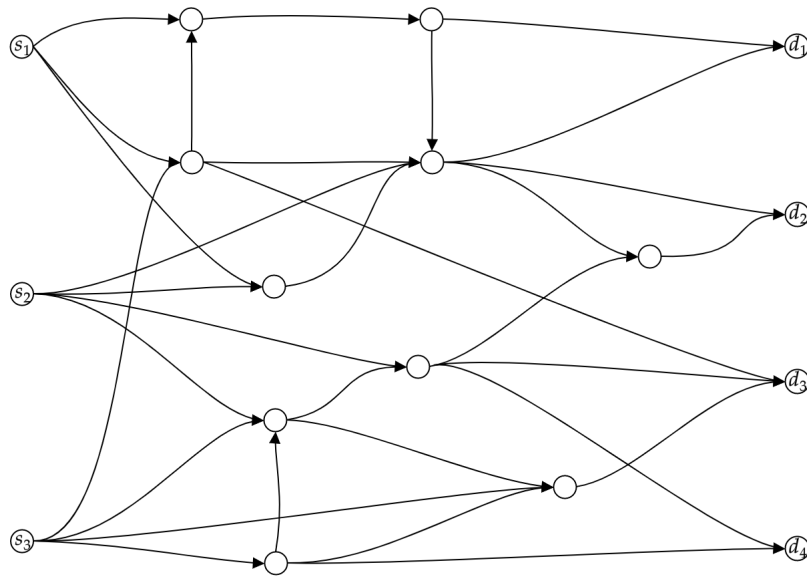


Figura 5.1

Para empezar, la primera iteración del algoritmo construye un árbol T_1 mediante una aplicación de BFS iniciando desde el nodo s_1 . En la siguiente figura, en verde se demarca el árbol y en rojo las aristas presentes en G_1 que no se encuentran en T_1 , y que por ende generan bubbles que ingresan al generador.

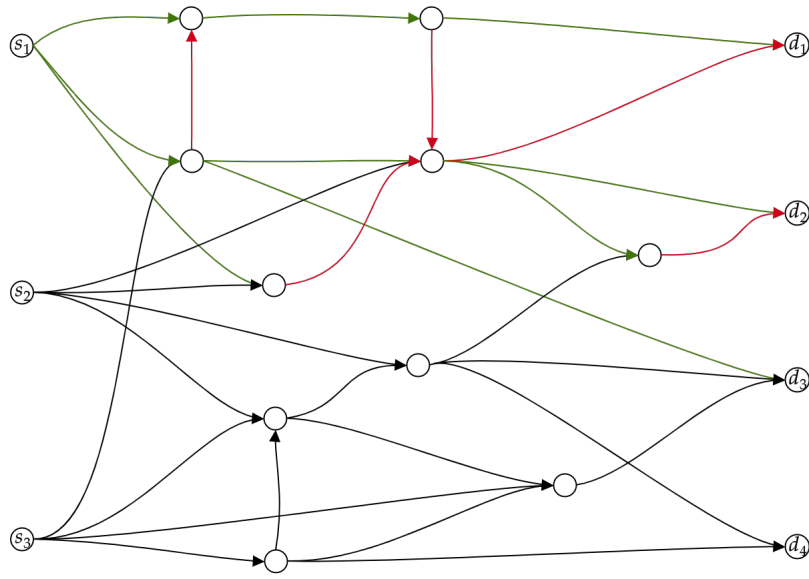


Figura 5.2

Hecho esto, todas las aristas de G_1 las marcamos con color azul, así sabremos cuales nodos ya han sido visitados previamente. Se realiza un BFS desde el nodo s_2 construyendo el árbol T_2 , sin embargo, cuando se llegue a un nodo ya visitado, es decir, que posea una arista de color azul entrando en él, BFS lo considera una hoja del árbol, lo denota como nodo frontera, y deja de iterar sobre esa rama. Así tenemos la siguiente figura, en la que del mismo modo se marca el árbol T_2 en verde, en azul el grafo G_1^* y en rojo las aristas en ΔG_2 que cierran bubbles que entran al generador. Además, se etiquetaron los nodos frontera como f_1, f_2, f_3 y f_4 .

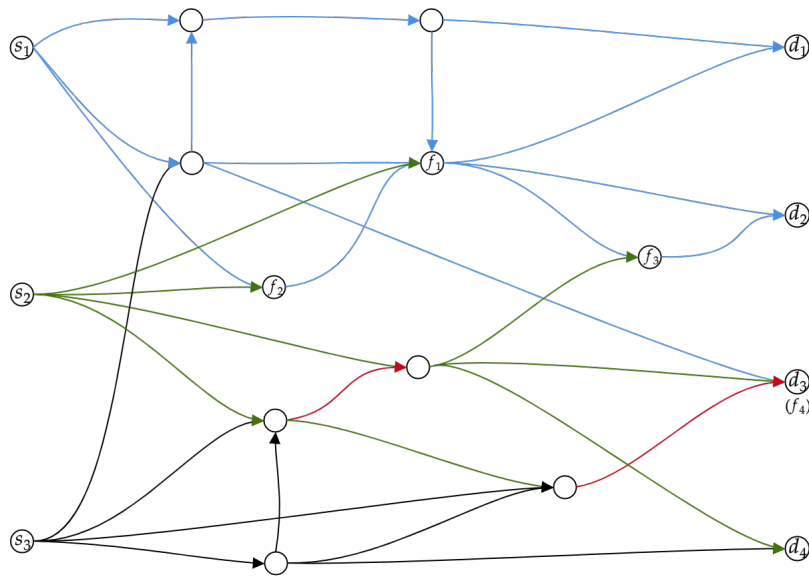


Figura 5.3

A partir de esta figura, se construye el multigrafo, presentado en la siguiente figura. Se aplica sobre ese multigrafo la subrutina de extensión y se para a la siguiente iteración. Esta subrutina se explica en el siguiente anexo.

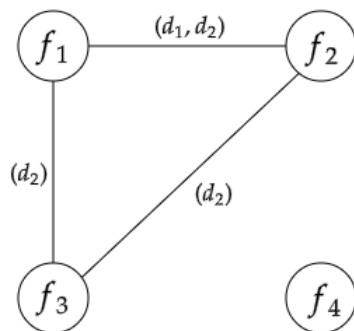


Figura 5.4

Por último, se repite el mismo procedimiento desde el nodo fuente s_3 , obteniéndose el siguiente resultado

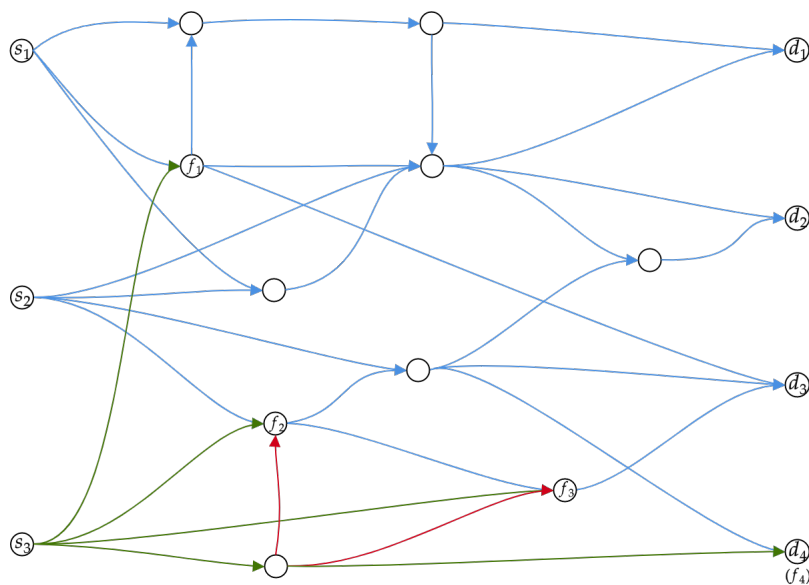


Figura 5.5

Y a partir de este, se contruye el siguiente multigrafo

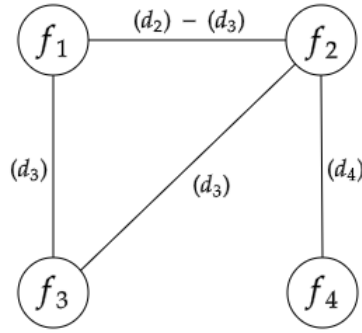


Figura 5.6

Reconocemos entonces 15 elementos que ingresan al conjunto generador:

1. En la figura 5.2, hay 5 aristas rojas, que con T_1 generan 5 bubbles que ingresan al generador.
2. En la figura 5.3, hay 2 aristas rojas, que del mismo modo generan 2 bubbles generadoras.
3. En la figura 5.5, hay 2 aristas rojas, por ende, 2 bubbles generadoras.
4. En la figura 5.4, tenemos bubbles generadoras representadas por las aristas entre f_1 y f_2 , y entre f_1 y f_3 .
5. En la figura 5.6, tenemos dos bubbles generadoras entre f_1 y f_2 , una etiquetada d_2 y otra etiquetada d_3 , una bubble entre f_1 y f_3 , y una entre f_2 y f_3 .

Estas dos últimas aseveraciones quedan explicadas mejor en la siguiente sección.

Anexo 2: Aplicación Subrutina de Extensión

Aplicaremos la subrutina de extensión al siguiente multigrafo:

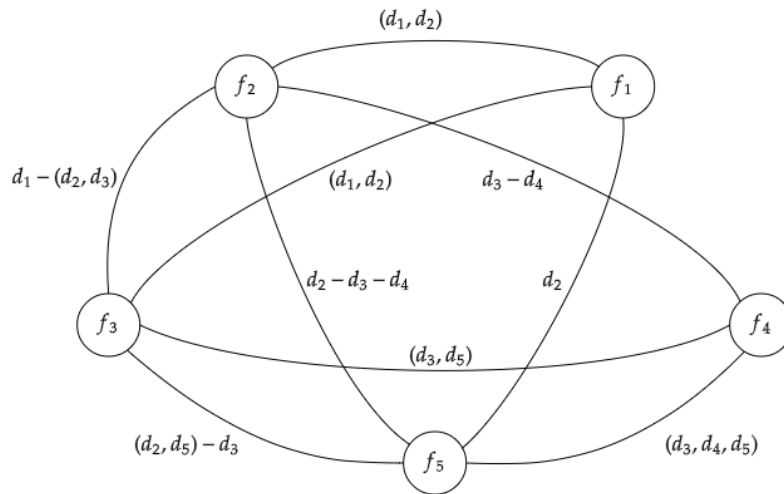


Figura 5.7

Cada arista del multigrafo esta representada por su etiquetado y aristas diferentes entre el mismo par de vértices están separadas por guiones. De este modo, $d_1 - (d_2, d_3)$ representa dos aristas entre el mismo par de vértices, una con etiqueta d_1 y otra con etiqueta (d_2, d_3) .

Se inicia la subrutina desde el vértice f_1 .

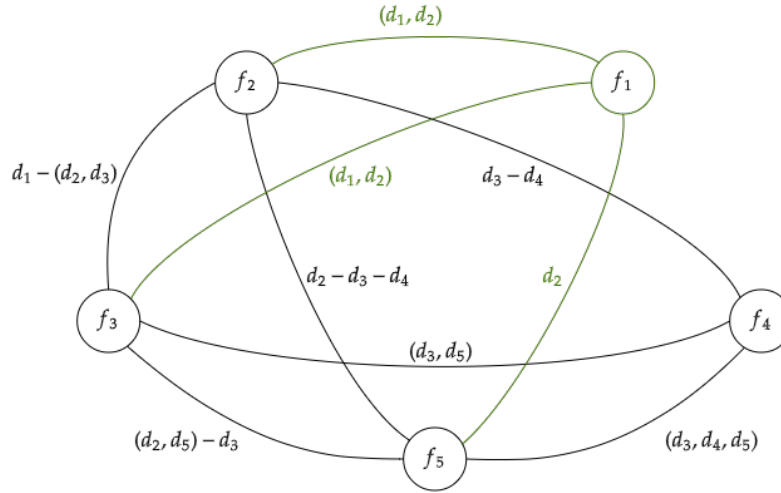


Figura 5.8

Se marcan en verde las aristas no revisadas que inciden en f_1 . Al ser la primera iteración, no hay ciclos posibles y por lo tanto todas las aristas ingresan al conjunto objetivo. Se marca un etiquetado con verde si entra al objetivo, y con rojo si no lo hace.

Se procede a la siguiente iteración, eligiendo algún vértice vecino a f_1 que aun no haya sido revisado. Se continúa por f_2 , y se marcan en verde las aristas incidentes a f_2 que aun no han sido revisadas.

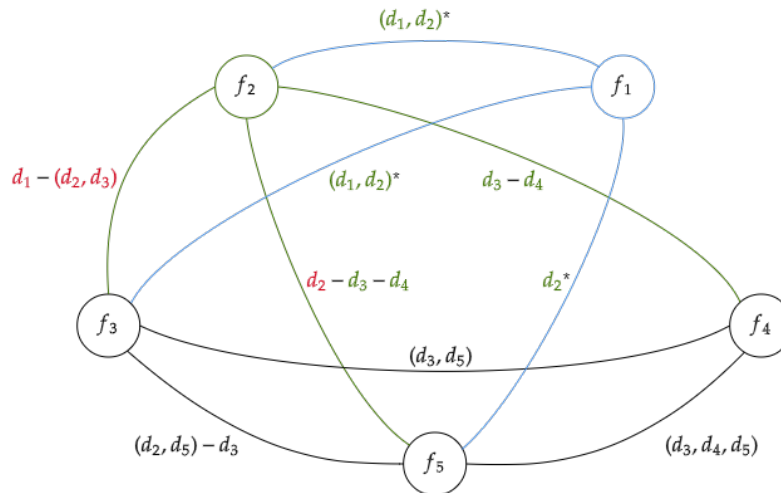


Figura 5.9

Del mismo modo, se revisan los etiquetados, y se marcan con verde aquellos que no produzcan ciclos con aristas de etiquetas compartidas ya revisadas, y

con rojo aquellas que lo hagan. Se destacan con asteriscos aquellas etiquetas ya revisadas que generan ciclos con las nuevas etiquetas marcadas en rojo, para clarificar la razón por la que no ingresan al generador.

Hecho esto, se elige un vértice vecino a f_1 o f_2 que aun no haya sido revisado, y se hace una nueva iteración. Eligiendo f_3 , y realizando el mismo análisis, se obtiene el siguiente resultado.

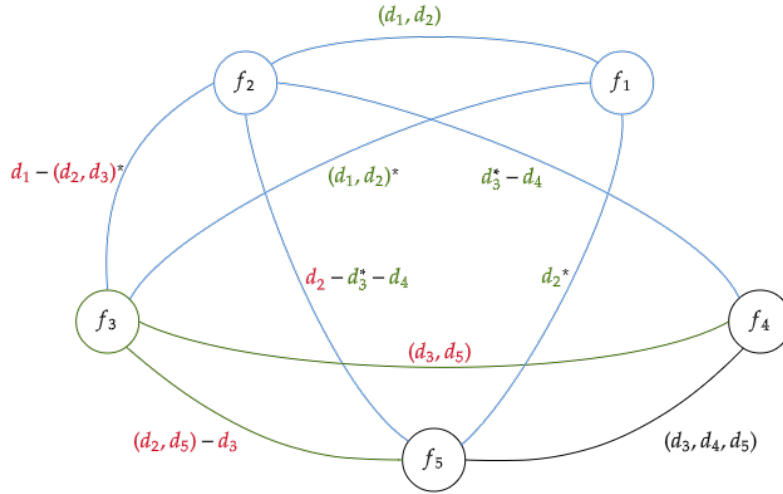


Figura 5.10

Se puede notar que la arista entre f_3 y f_4 etiquetada por (d_3, d_5) no ingresa a la base pues genera un ciclo con las aristas ya revisadas entre f_2 y f_3 con etiqueta (d_2, d_3) y la arista entre f_2 y f_4 con etiqueta d_3 . Cabe destacar que los elementos que se encuentran en el generador que dejan fuera la arista revisada, son el par existentes entre f_1, f_2 y f_3 con etiquetas (d_1, d_2) , es decir, no comparten ninguna etiqueta con ella, y sin embargo, participan en su generación.

Se procede con el siguiente nodo, en este caso f_5 , obteniéndose el siguiente resultado.

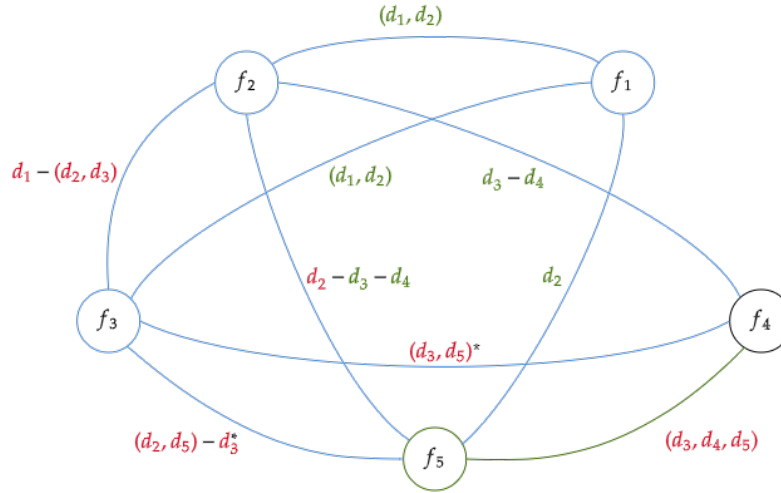


Figura 5.11

Se analizan las aristas entre el último par de vértices y la subrutina concluye marcando el último vértice, f_4 , verificando que ya se revisaron todos los vértices y retornando el resultado.

En la siguiente figura se presenta el multigrafo ya analizado, cada etiqueta se encuentra demarcada con algún color, verde si esta en el conjunto retornado, rojo si no. Sólo resta encontrar una bubble que represente cada arista con etiqueta verde.

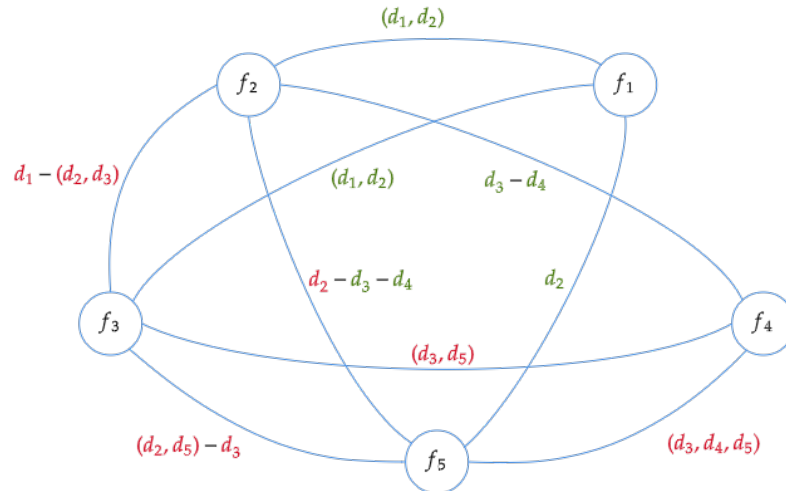


Figura 5.12

Con esto se concluye la subrutina, y el algoritmo general puede pasar a su siguiente iteración.

Bibliografía

- [1] Vicente Acuña, Roberto Grossi, Giuseppe F. Italiano, Leandro Lima, Romeo Rizzi, Gustavo Sacomoto, Marie-France Sagot, and Blerina Sinimeri. On bubble generators in directed graphs. In Hans L. Bodlaender and Gerhard J. Woeginger, editors, *Graph-Theoretic Concepts in Computer Science*, pages 18–31, Cham, 2017. Springer International Publishing.
- [2] Etienne Birmelé, Pierluigi Crescenzi, Rui Ferreira, Roberto Grossi, Vincent Lacroix, Andrea Marino, Nadia Pisanti, Gustavo Sacomoto, and Marie-France Sagot. Efficient bubble enumeration in directed graphs. In *International Symposium on String Processing and Information Retrieval*, pages 118–129. Springer, 2012.
- [3] Frederick R. Blattner, Guy Plunkett, Craig A. Bloch, Nicole T. Perna, Valerie Burland, Monica Riley, Julio Collado-Vides, Jeremy D. Glasner, Christopher K. Rode, George F. Mayhew, Jason Gregor, Nelson Wayne Davis, Heather A. Kirkpatrick, Michael A. Goeden, Debra J. Rose, Bob Mau, and Ying Shao. The complete genome sequence of escherichia coli k-12. *Science*, 277(5331):1453–1462, 1997.
- [4] Ljiljana Brankovic, Costas S Iliopoulos, Ritu Kundu, Manal Mohamed, Solon P Pissis, and Fatima Vayani. Linear-time superbubble identification algorithm for genome assembly. *Theoretical Computer Science*, 609:374–383, 2016.
- [5] Zamin Iqbal, Mario Caccamo, Isaac Turner, Paul Flicek, and Gil McVean. De novo assembly and genotyping of variants using colored de bruijn graphs. *Nature genetics*, 44(2):226, 2012.
- [6] Jody Lynn Kujovich. Factor v leiden thrombophilia. *Genetics in Medicine*, 13(1):1, 2011.
- [7] S Kurtz and C Schleiermacher. REPuter: fast computation of maximal repeats in complete genomes. *Bioinformatics*, 15(5):426–427, 05 1999.
- [8] Taku Onodera, Kunihiro Sadakane, and Tetsuo Shibuya. Detecting superbubbles in assembly graphs. In *International Workshop on Algorithms in Bioinformatics*, pages 338–348. Springer, 2013.

- [9] Paul A Pevzner, Haixu Tang, and Glenn Tesler. De novo repeat classification and fragment assembly. *Genome research*, 14(9):1786–1796, 2004.
- [10] Gustavo Sacomoto, Vincent Lacroix, and Marie-France Sagot. A polynomial delay algorithm for the enumeration of bubbles with length constraints in directed graphs and its application to the detection of alternative splicing in rna-seq data. In *International Workshop on Algorithms in Bioinformatics*, pages 99–111. Springer, 2013.
- [11] Gustavo Sacomoto, Blerina Sinimeri, Camille Marchet, Vincent Miele, Marie-France Sagot, and Vincent Lacroix. Navigating in a sea of repeats in rna-seq without drowning. In Dan Brown and Burkhard Morgenstern, editors, *Algorithms in Bioinformatics*, pages 82–96, Berlin, Heidelberg, 2014. Springer Berlin Heidelberg.
- [12] Wing-Kin Sung, Kunihiko Sadakane, Tetsuo Shibuya, Abha Belorkar, and Iana Pyrogova. An $O(m \log m)$ -time algorithm for detecting superbubbles. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, 12(4):770–777, 2015.
- [13] Ülo Väli, Mikael Brandström, Malin Johansson, and Hans Ellegren. Insertion-deletion polymorphisms (indels) as genetic markers in natural populations. *BMC genetics*, 9(1):8, 2008.