

---

# Máxima subgráfica común inducida mediante una heurística inspirada en ACO

---

***Estudiantes:***  
Ortiz Montiel Diego Iain  
319072369

***Profesor:***  
Canek Peláez Valdés

***Ayudantes:***  
Leslie Ramírez Gallegos

# Índice

<b>1. Introducción</b>	<b>2</b>
<b>2. Formulación del problema</b>	<b>2</b>
2.1. La gráfica . . . . .	3
2.2. Subgráficas inducidas . . . . .	3
2.3. Isomorfismo inducido . . . . .	3
2.4. Soluciones . . . . .	3
2.5. Soluciones factibles y no factibles . . . . .	4
2.6. Criterio de optimización . . . . .	4
2.7. Complejidad del problema . . . . .	4
<b>3. Heurística basada en Colonias de Hormigas</b>	<b>5</b>
3.1. Preliminares . . . . .	5
3.2. Representación del espacio de búsqueda . . . . .	5
3.3. Construcción de soluciones . . . . .	6
3.4. Actualización de feromonas . . . . .	6
3.5. Criterios de paro . . . . .	6
3.6. Comentarios sobre el comportamiento de la heurística . . . . .	7
<b>4. Implementación</b>	<b>7</b>
4.1. Lenguaje de programación . . . . .	7
4.2. Organización del sistema . . . . .	7
4.3. Herramientas de compilación y ejecución . . . . .	8
4.4. Representación de las gráficas . . . . .	8
4.5. Representación de soluciones . . . . .	8
4.6. Manejo de la factibilidad . . . . .	8
4.7. Construcción paralela de soluciones . . . . .	8
4.8. Parámetros y reproducibilidad . . . . .	9
4.9. Limitaciones de la implementación . . . . .	9
<b>5. Experimentación</b>	<b>9</b>
5.1. Equipo de prueba: . . . . .	9
5.2. Tipos de instancias . . . . .	10
5.3. Configuración de instancias . . . . .	10
5.4. Parámetros y semillas . . . . .	10
5.5. Métricas . . . . .	11
5.6. Resultados . . . . .	11
5.7. Análisis del parámetro de <i>cutoff</i> . . . . .	11
<b>6. Conclusiones generales</b>	<b>12</b>

## Resumen

Se estudia el problema de la máxima subgráfica común inducida (MCIS) y se propone una heurística inspirada en colonias de hormigas para su resolución en grafos grandes y ruidosos. Se presentan resultados experimentales y un análisis del comportamiento del algoritmo.

## 1. Introducción

La comparación estructural entre gráficas es un problema central en múltiples áreas de la ciencia computacional y aplicada [1]. En disciplinas como la bioinformática, por ejemplo, es común modelar moléculas, proteínas o redes de interacción como gráficas, donde los vértices representan entidades químicas o biológicas y las aristas representan relaciones estructurales o funcionales [2], [3]. Determinar similitudes entre estas estructuras permite identificar patrones conservados, inferir funciones desconocidas o comparar comportamientos entre sistemas complejos. Problemas análogos aparecen también en visión por computadora, análisis de redes sociales, detección de fraude y recuperación de información estructurada [4], [5].

Dentro de este contexto, el problema de la *Máxima Subgráfica Común Inducida* (Maximum Common Induced Subgraph, MCIS) surge como una formulación natural para medir similitud estructural entre dos gráficas [1], [6]. Dado un par de gráficas, el objetivo es encontrar la subgráfica inducida más grande que sea isomorfa en ambas, preservando tanto la adyacencia como la no adyacencia entre vértices. Esta restricción hace que el MCIS sea una medida especialmente estricta de similitud, pero también incrementa considerablemente su complejidad computacional.

El problema del MCIS es conocido por ser NP-difícil, lo que implica que no se conocen algoritmos exactos eficientes que lo resuelvan para instancias grandes [6]. En aplicaciones reales, donde las gráficas pueden contener cientos o miles de vértices y presentar ruido estructural, los métodos exactos resultan impracticables. Por esta razón, en la práctica se recurre a técnicas aproximadas y heurísticas que buscan soluciones de alta calidad en tiempos razonables, aun sin garantizar optimalidad [7].

Entre estas técnicas, las metaheurísticas inspiradas en procesos naturales han mostrado ser particularmente efectivas para problemas de optimización combinatoria [8]. En este trabajo se explora una heurística inspirada en el comportamiento de colonias de hormigas (Ant Colony Optimization, ACO) para aproximar soluciones al problema del MCIS inducido.

El objetivo de este reporte es presentar la formulación del problema, describir la implementación del sistema propuesto y analizar experimentalmente su comportamiento en distintas instancias, poniendo especial énfasis en los efectos del tamaño de la instancia, las restricciones inducidas y las decisiones de diseño que afectan la exploración del espacio de búsqueda.

## 2. Formulación del problema

En este capítulo se acota y formaliza el problema de la Máxima Subgráfica Común Inducida (MCIS). El objetivo es establecer con precisión la estructura matemática del

problema que se abordará, así como las restricciones que determinan qué soluciones son válidas y cuáles no. Esta formulación servirá como base para el diseño del sistema computacional descrito en secciones posteriores [1].

## 2.1. La gráfica

Sea  $G = (V, E)$  una gráfica simple no dirigida, donde  $V$  es un conjunto finito de vértices y  $E \subseteq \{\{u, v\} \mid u, v \in V, u \neq v\}$  es el conjunto de aristas. En este trabajo se consideran únicamente gráficas simples, sin lazos ni aristas múltiples [2].

Sean  $G_1 = (V_1, E_1)$  y  $G_2 = (V_2, E_2)$  dos gráficas dadas. Estas gráficas representan las instancias de entrada del problema [5].

## 2.2. Subgráficas inducidas

Dada una gráfica  $G = (V, E)$  y un subconjunto de vértices  $S \subseteq V$ , se define la subgráfica inducida por  $S$ , denotada como  $G[S]$ , como la gráfica:

$$G[S] = (S, E_S),$$

donde

$$E_S = \{\{u, v\} \in E \mid u \in S, v \in S\}.$$

La subgráfica inducida conserva exactamente todas las relaciones de adyacencia y no adyacencia entre los vértices de  $S$  presentes en la gráfica original, lo cual la distingue de otras nociones más laxas de subestructura [7].

## 2.3. Isomorfismo inducido

Sean  $G_1[S_1]$  y  $G_2[S_2]$  subgráficas inducidas de  $G_1$  y  $G_2$ , respectivamente. Diremos que estas subgráficas son isomorfas si existe una biyección

$$f : S_1 \rightarrow S_2$$

tal que, para todo par de vértices distintos  $u, v \in S_1$ , se cumple:

$$\{u, v\} \in E_1 \iff \{f(u), f(v)\} \in E_2.$$

Esta condición impone la preservación simultánea de la adyacencia y de la no adyacencia, lo que da lugar a la noción de isomorfismo inducido, característica central del problema del MCIS [6].

## 2.4. Soluciones

Dadas dos gráficas  $G_1$  y  $G_2$ , una solución al problema del MCIS consiste en dos subconjuntos de vértices  $S_1 \subseteq V_1$  y  $S_2 \subseteq V_2$ , junto con una biyección  $f : S_1 \rightarrow S_2$ , tal que las subgráficas inducidas  $G_1[S_1]$  y  $G_2[S_2]$  sean isomorfas [7].

A diferencia de otros problemas de comparación estructural, no toda correspondencia parcial entre vértices constituye una solución válida, ya que cualquier violación a la condición de isomorfismo inducido invalida completamente la solución.

## 2.5. Soluciones factibles y no factibles

Clasificaremos las soluciones candidatas en dos tipos: factibles y no factibles.

[Solución factible] Una solución  $(S_1, S_2, f)$  será factible si y sólo si  $f$  define un isomorfismo inducido entre  $G_1[S_1]$  y  $G_2[S_2]$ .

Si existe al menos un par de vértices  $u, v \in S_1$  tal que la relación de adyacencia entre  $u$  y  $v$  no se preserve bajo  $f$ , la solución se considerará no factible. Esta distinción es fundamental en el MCIS, ya que incluso violaciones locales invalidan completamente la solución [1].

## 2.6. Criterio de optimización

El objetivo del problema del MCIS es maximizar el tamaño de la subgráfica común inducida. Formalmente, se busca maximizar el valor:

$$|S_1| = |S_2|.$$

El número de aristas inducidas queda determinado de manera implícita por la estructura de los subconjuntos seleccionados y no se optimiza de forma independiente, pero son un criterio de desempate, como es habitual en formulaciones clásicas del problema [7].

## 2.7. Complejidad del problema

El problema de la Máxima Subgráfica Común Inducida (MCIS) es conocido por ser NP-difícil. Esta dificultad puede justificarse observando que el MCIS generaliza otros problemas clásicos de alta complejidad. En particular, el problema del *clique máximo* puede reducirse en tiempo polinomial a una instancia del MCIS: dada una gráfica  $G$ , encontrar un clique de tamaño máximo en  $G$  es equivalente a encontrar una subgráfica común inducida máxima entre  $G$  y una gráfica completa del mismo tamaño. Como el problema del clique máximo es NP-completo, se sigue que el MCIS es, al menos, NP-difícil [6].

La dificultad del MCIS se ve acentuada por la restricción de inducción. A diferencia de formulaciones más laxas de subestructura común, la inducción exige la preservación simultánea de la adyacencia y de la no adyacencia entre todos los pares de vértices seleccionados. Esta condición introduce dependencias globales entre las decisiones locales de selección de vértices, provocando que el espacio de soluciones factibles se reduzca de manera drástica y altamente no lineal conforme crece el tamaño de la instancia.

Es importante señalar que el problema del MCIS, en su formulación de optimización, no pertenece a la clase NP en el sentido clásico. Aunque una solución candidata puede describirse mediante un mapeo entre subconjuntos de vértices, verificar que dicha solución es óptima requiere comparar su tamaño con el de todas las demás soluciones factibles posibles, lo cual no puede realizarse en tiempo polinomial conocido. En contraste, la versión de decisión del problema —determinar si existe una subgráfica común inducida de tamaño al menos  $k$ — sí pertenece a NP, ya que un certificado consistente en el mapeo puede verificarse en tiempo polinomial.

Debido a esta complejidad inherente, el uso de algoritmos exactos resulta impráctico para instancias de tamaño moderado o grande. En consecuencia, en este trabajo

se recurre al uso de heurísticas inspiradas en optimización combinatoria para aproximar soluciones de alta calidad en tiempos razonables, sin garantizar optimalidad global [8]. Este enfoque permite explorar de manera efectiva el espacio de soluciones factibles, aun cuando no sea posible asegurar la obtención de la solución óptima.

### 3. Heurística basada en Colonias de Hormigas

Dado que el problema de la Máxima Subgráfica Común Inducida es NP-difícil y que su formulación de optimización no admite algoritmos exactos eficientes para instancias grandes, en este trabajo se adopta una heurística inspirada en el comportamiento colectivo de colonias de hormigas, conocida como *Ant Colony Optimization* (ACO). Esta metaheurística fue propuesta originalmente por Marco Dorigo a principios de la década de 1990 y ha sido aplicada con éxito a diversos problemas de optimización combinatoria [8].

La idea central de ACO consiste en modelar la construcción de soluciones como un proceso estocástico guiado por información acumulada colectivamente, denominada feromonía, y por información heurística local. A lo largo de múltiples iteraciones, una colonia de agentes simples (hormigas) construye soluciones candidatas, reforzando progresivamente aquellas decisiones que conducen a soluciones de mayor calidad [8].

#### 3.1. Preliminares

Sea  $P$  un problema de optimización clasificado como NP-difícil y sea  $S$  el conjunto de soluciones factibles para una instancia dada de  $P$ . Se asume la existencia de una función objetivo

$$f : S \rightarrow \mathbb{R}^+,$$

tal que valores menores (o mayores, según la formulación) indican soluciones de mejor calidad [6].

En el caso del MCIS, una solución consiste en un mapeo inducido entre subconjuntos de vértices de dos gráficas de entrada, y la función objetivo busca maximizar el número de vértices de la subgráfica común inducida [7].

A diferencia de heurísticas de búsqueda local, ACO no requiere definir explícitamente una relación de vecindad entre soluciones completas. En su lugar, las soluciones se construyen incrementalmente a partir de componentes elementales, lo que resulta especialmente adecuado para problemas donde las soluciones deben satisfacer restricciones estructurales globales, como ocurre en el MCIS inducido [8].

#### 3.2. Representación del espacio de búsqueda

En la heurística propuesta, el espacio de búsqueda se modela como el conjunto de correspondencias posibles entre vértices de las gráficas  $G_1$  y  $G_2$ . Cada componente elemental de una solución corresponde a una pareja  $(u, v)$ , donde  $u \in V_1$  y  $v \in V_2$ , que representa la decisión de mapear el vértice  $u$  al vértice  $v$ .

A cada posible correspondencia se le asocia una cantidad de feromonía  $\tau(u, v)$ , que representa el grado de preferencia colectiva de la colonia por dicha asignación, así como un

valor heurístico  $\eta(u, v)$ , que codifica información local sobre la compatibilidad estructural entre los vértices involucrados [8].

### 3.3. Construcción de soluciones

Cada hormiga construye una solución de manera incremental, comenzando desde una solución vacía. En cada paso, la hormiga selecciona una nueva correspondencia  $(u, v)$  entre las candidatas disponibles, de acuerdo con una regla probabilística que combina feromona e información heurística.

Formalmente, la probabilidad de seleccionar la correspondencia  $(u, v)$  en un estado dado está determinada por:

$$P(u, v) = \frac{\tau(u, v)^\alpha \cdot \eta(u, v)^\beta}{\sum_{(u', v') \in C} \tau(u', v')^\alpha \cdot \eta(u', v')^\beta},$$

donde  $C$  es el conjunto de correspondencias factibles en el estado actual, y  $\alpha, \beta \in \mathbb{R}^+$  son parámetros que controlan la influencia relativa de la feromona y de la heurística [8].

El proceso de construcción continúa hasta que no existan correspondencias adicionales que puedan añadirse sin violar las restricciones del problema, o hasta que se cumpla alguna condición de paro adicional.

### 3.4. Actualización de feromonas

Al finalizar una iteración, las soluciones construidas por las hormigas son evaluadas de acuerdo con la función objetivo. Posteriormente, la feromona se actualiza siguiendo dos mecanismos complementarios: evaporación y refuerzo.

La evaporación se aplica de forma global para evitar la convergencia prematura:

$$\tau(u, v) \leftarrow (1 - \rho) \tau(u, v),$$

donde  $\rho \in (0, 1)$  es el coeficiente de evaporación [8].

El refuerzo se aplica a las correspondencias que forman parte de soluciones de alta calidad, incrementando su valor de feromona en proporción a la calidad de la solución encontrada. Este mecanismo introduce un proceso de retroalimentación positiva que favorece la reutilización de buenas decisiones en iteraciones posteriores [8].

### 3.5. Criterios de paro

La ejecución del algoritmo se detiene cuando se cumple alguna de las siguientes condiciones:

- se alcanza un número máximo de iteraciones;
- la colonia converge a una solución estable durante un número determinado de iteraciones;
- se alcanza un límite de tiempo de ejecución.

Durante toda la ejecución se mantiene registro de la mejor solución encontrada, la cual es reportada como resultado final del sistema.

### 3.6. Comentarios sobre el comportamiento de la heurística

La calidad del sistema depende críticamente de los parámetros  $\alpha$ ,  $\beta$  y  $\rho$ , así como del número de hormigas y del criterio de paro utilizado. No existe un consenso general en la literatura sobre valores óptimos universales para estos parámetros; por ello, en este trabajo se determinan de manera experimental [8].

Asimismo, el desempeño de la heurística está fuertemente influenciado por la forma en que se define la información heurística y por la manera en que se restringe el conjunto de correspondencias factibles durante la construcción de soluciones. Estos aspectos se detallan en la sección de Implementación.

## 4. Implementación

En esta sección se describen los aspectos prácticos de la implementación del sistema propuesto, incluyendo el lenguaje de programación utilizado, la organización del código, las herramientas de compilación y las principales decisiones de diseño adoptadas para materializar la heurística basada en Colonias de Hormigas aplicada al problema del MCIS inducido.

### 4.1. Lenguaje de programación

El sistema fue implementado en C++, aprovechando su eficiencia en tiempo de ejecución y su control explícito sobre la gestión de memoria, características fundamentales para abordar problemas de optimización combinatoria sobre gráficas de gran tamaño. El uso de C++ permite además una integración directa con bibliotecas estándar de concurrencia y facilita la paralelización del algoritmo.

### 4.2. Organización del sistema

El código se organiza en módulos claramente diferenciados, cada uno con una responsabilidad específica dentro del sistema. Entre los componentes principales se encuentran:

- un módulo de lectura y representación de gráficas, encargado de construir las estructuras de datos internas a partir de los archivos de entrada;
- un módulo que implementa la estructura de soluciones y el verificador de factibilidad inducida;
- un módulo dedicado a la heurística ACO, que coordina la construcción de soluciones, la actualización de feromonas y la evaluación de resultados;
- un módulo de visualización y exportación, utilizado para generar representaciones gráficas de las soluciones encontradas.

Esta separación facilita el mantenimiento del código y permite modificar componentes individuales sin afectar al resto del sistema.

### 4.3. Herramientas de compilación y ejecución

El proyecto utiliza el sistema de construcción `Meson` para la gestión del proceso de compilación. `Meson` permite una configuración clara de dependencias, opciones de compilación y generación de binarios optimizados, lo cual resulta conveniente para proyectos de investigación que requieren múltiples iteraciones experimentales.

La paralelización del sistema se realiza mediante `OpenMP`, lo que permite ejecutar múltiples hormigas de manera concurrente sobre arquitecturas multinúcleo. El número de hilos utilizados puede configurarse dinámicamente a través de la variable de entorno `OMP_NUM_THREADS`, lo que facilita la experimentación bajo distintas configuraciones de hardware.

### 4.4. Representación de las gráficas

Las gráficas de entrada se representan internamente mediante listas de adyacencia. Esta elección permite verificar relaciones de adyacencia en tiempo constante promedio y resulta especialmente adecuada para el MCIS inducido, donde la verificación repetida de adyacencia y no adyacencia entre pares de vértices es una operación crítica [2].

Cada vértice se identifica mediante un índice entero, y las aristas se almacenan de manera simétrica, respetando la naturaleza no dirigida de las gráficas consideradas.

### 4.5. Representación de soluciones

Una solución parcial o completa se representa como un mapeo inyectivo entre vértices de la primera gráfica y vértices de la segunda. Este mapeo define implícitamente los subconjuntos de vértices inducidos y la correspondencia entre ellos.

La representación permite verificar de manera incremental la factibilidad inducida: al intentar extender una solución parcial con una nueva correspondencia, se comprueba que las relaciones de adyacencia y no adyacencia se preserven con respecto a todos los vértices previamente mapeados.

### 4.6. Manejo de la factibilidad

El sistema mantiene de manera estricta la factibilidad inducida durante todo el proceso de construcción de soluciones. Cualquier correspondencia que viole las condiciones de isomorfismo inducido es descartada inmediatamente.

Esta decisión de diseño reduce de forma significativa el espacio de búsqueda y evita la exploración de soluciones estructuralmente inválidas. Aunque este enfoque limita la capacidad de escapar de mínimos locales, experimentos preliminares mostraron que permitir soluciones temporalmente no factibles no produjo mejoras consistentes en la calidad de las soluciones finales, mientras que incrementó el costo computacional enormemente [7].

### 4.7. Construcción paralela de soluciones

En cada iteración, múltiples hormigas construyen soluciones de manera independiente siguiendo la misma regla probabilística, pero utilizando fuentes de aleatoriedad distintas.

Este esquema favorece la exploración del espacio de búsqueda y permite aprovechar el paralelismo disponible en el hardware subyacente.

El paralelismo introduce variabilidad controlada en el proceso de búsqueda, pero no elimina las dificultades inherentes al crecimiento del espacio de soluciones factibles en instancias grandes o altamente simétricas.

#### 4.8. Parámetros y reproducibilidad

El comportamiento del sistema depende de parámetros como el número de hormigas, los coeficientes  $\alpha$  y  $\beta$  que controlan la influencia de la feromonía y de la información heurística, y el coeficiente de evaporación  $\rho$ . Dado que no existe un consenso general sobre valores óptimos universales, dichos parámetros se determinaron de manera experimental [8].

Para garantizar la reproducibilidad de los resultados, el sistema permite fijar explícitamente la semilla del generador de números aleatorios. Esto facilita la repetición exacta de experimentos y el análisis comparativo entre distintas configuraciones.

#### 4.9. Limitaciones de la implementación

Si bien la implementación prioriza la corrección estructural y la claridad del modelo, presenta limitaciones inherentes al enfoque adoptado. En particular, la construcción incremental estrictamente factible dificulta la recuperación de subgráficas inducidas muy grandes en escenarios con alta simetría estructural o núcleos densos.

Estas limitaciones no se deben a errores de implementación, sino a compromisos deliberados de diseño orientados a garantizar soluciones válidas y un comportamiento controlable del sistema. Su impacto se analiza experimentalmente en la sección siguiente.

### 5. Experimentación

Esta sección describe el diseño experimental utilizado para evaluar la heurística propuesta para el problema de la Máxima Subgráfica Común Inducida. El objetivo es analizar el comportamiento del sistema bajo distintos escenarios estructurales, así como estudiar su escalabilidad, estabilidad y sensibilidad a parámetros de diseño.

#### 5.1. Equipo de prueba:

Todas las pruebas se hicieron en un equipo cuyas características relevantes son:

1. Procesador: Ryzen 9 8945HS, 5.2GHZ
2. Sistema operativo: Fedora 42
3. RAM: 16gb

## 5.2. Tipos de instancias

Se consideraron cuatro tipos de instancias, cada una diseñada para evaluar un aspecto distinto del algoritmo:

- **Núcleo compartido pequeño:** ambas gráficas contienen un núcleo común inducido reducido, inmerso en una cantidad significativa de ruido estructural.
- **Núcleo compartido grande:** ambas gráficas contienen un núcleo común inducido de gran tamaño, lo que permite evaluar la escalabilidad del sistema.
- **Par isomorfo mediano:** ambas gráficas son isomorfas, utilizadas como referencia para el mejor resultado esperable.
- **Caso incompatible mediano:** una gráfica mediana se compara contra una gráfica casi plana (muy poco densa), diseñada para no compartir una subgráfica inducida grande.

## 5.3. Configuración de instancias

Las configuraciones específicas utilizadas en los experimentos se resumen en la Tabla 1.

Instancia	$ V_1 $	$ V_2 $	Núcleo común	Tipo
NC-P	200	250	80	Núcleo pequeño
NC-G	800	1000	400	Núcleo grande
ISO-M	300	300	300	Isomorfo mediano
INC-M	300	300	—	Incompatible

Cuadro 1: Configuración de las instancias utilizadas.

En el caso incompatible, la segunda gráfica corresponde a una gráfica casi plana generada mediante un modelo Erdős–Rényi con probabilidad de arista muy baja.

## 5.4. Parámetros y semillas

Para todas las ejecuciones se utilizaron los siguientes parámetros:

- Número de hormigas: 14
- Iteraciones máximas: 1000
- $\alpha = 1,0$ ,  $\beta = 3,0$ ,  $\rho = 0,1$

Debido al carácter no determinista del algoritmo, cada instancia se ejecutó múltiples veces con distintas semillas, cabe destacar que dada la naturaleza de la eucarística basada en aprendizaje nos convino mas pocas semillas largas que muchas rápidas:

- 10 semillas para instancias NC-P e ISO-M.
- 5 semillas para NC-G.
- 10 semillas para INC-M.

## 5.5. Métricas

Para cada ejecución se registraron las siguientes métricas:

- Número de vértices del MCIS inducido encontrado.
- Número de aristas inducidas.
- Tiempo total de ejecución.

## 5.6. Resultados

Los resultados se resumen en la Tabla 2. Los valores corresponden al promedio y al mejor resultado obtenido entre todas las semillas.

Instancia	Corridas	Prom. vértices	Mejor vértices	Mejor aristas	Mejor semilla	Tiempo prom. (s)
INC-M	10	0.00	0	0	1	0.05
ISO-M	10	300.00	300	17923	1	69.25
NC-G	5	18.00	19	17	3	1.10
NC-P	10	66.60	67	356	5	22.89

Cuadro 2: Resultados experimentales de la heurística ACO para distintas instancias del problema MCIS inducido.

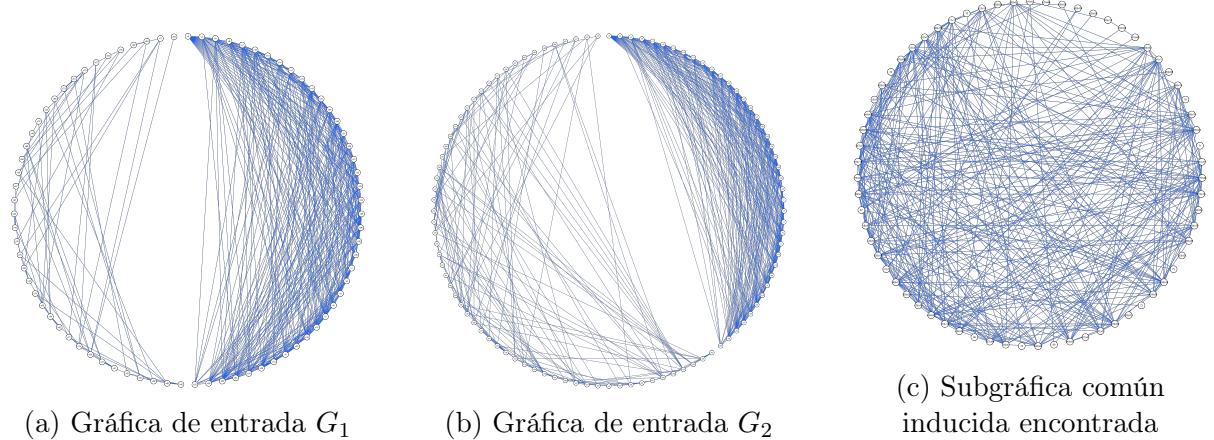


Figura 1: Ejemplo de una instancia del problema MCIS: dos gráficas de entrada y la subgráfica común inducida obtenida por la heurística.

## 5.7. Análisis del parámetro de *cutoff*

Durante las primeras versiones del sistema se incorporó un parámetro denominado *cutoff*, cuyo objetivo era permitir un grado limitado de no factibilidad estructural durante la construcción de soluciones. En particular, el *cutoff* controlaba la tolerancia a violaciones temporales del isomorfismo inducido, con la intención de aumentar la exploración del espacio de búsqueda y evitar decisiones demasiado restrictivas en etapas tempranas del algoritmo.

La motivación de este mecanismo era permitir que la heurística pudiera atravesar regiones del espacio de soluciones que no fueran estrictamente factibles, con la expectativa de que dichas violaciones pudieran corregirse posteriormente mediante la dinámica de feromonas y la selección iterativa de correspondencias.

Para evaluar su impacto, se realizaron experimentos variando el valor del *cutoff*, incluyendo ejecuciones con tolerancia nula (factibilidad estricta), tolerancia moderada y tolerancia alta. En todos los casos se mantuvieron constantes las semillas y el resto de los parámetros del sistema.

Los resultados experimentales mostraron que el valor del *cutoff* no tuvo impacto medible en la calidad final de las soluciones obtenidas. Para una misma semilla, las ejecuciones convergieron a soluciones idénticas en términos de número de vértices, número de aristas inducidas y mapeo final, independientemente del nivel de no factibilidad permitido.

Este comportamiento se explica por la naturaleza del problema del MCIS inducido y por la heurística estructural dominante utilizada. Las restricciones de inducción reducen de forma drástica el espacio de extensiones viables, lo que provoca que las soluciones parcialmente no factibles no puedan evolucionar hacia soluciones inducidas grandes. En consecuencia, la búsqueda colapsa hacia el mismo conjunto de decisiones estructuralmente válidas.

Dado que permitir no factibilidad temporal no produjo mejoras observables en explotación ni en calidad de solución, y únicamente incrementó la complejidad del algoritmo, el parámetro de *cutoff* fue eliminado en la versión final del sistema.

## 6. Conclusiones generales

## Referencias

- [1] D. Conte, P. Foggia, C. Sansone y M. Vento, “Thirty Years of Graph Matching in Pattern Recognition,” *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 18, n.º 3, págs. 265-298, 2004.
- [2] H. Bunke, “Graph Matching: Theoretical Foundations, Algorithms, and Applications,” *Proceedings of Vision Interface*, págs. 82-88, 2000.
- [3] K. M. Borgwardt y H.-P. Kriegel, “Shortest-Path Kernels on Graphs,” *IEEE International Conference on Data Mining*, págs. 74-81, 2005.
- [4] D. J. Cook y L. B. Holder, *Substructure Discovery Using Minimum Description Length and Background Knowledge*. Journal of Artificial Intelligence Research, 1994.
- [5] M. E. J. Newman, *Networks: An Introduction*. Oxford University Press, 2010.
- [6] M. R. Garey y D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.
- [7] H. Bunke y K. Shearer, “A Graph Distance Metric Based on the Maximal Common Subgraph,” *Pattern Recognition Letters*, vol. 19, n.º 3-4, págs. 255-259, 1998.
- [8] M. Dorigo, M. Birattari y T. Stützle, *Ant Colony Optimization*. IEEE Computational Intelligence Magazine, 2006.