Universidad Nacional Autónoma de México

Facultad de Ciencias

Comprendiendo y atacando la corrupción

un análisis mediante teoría de redes y teoría de juegos a los sistemas sociales y su evolución

T E S I S

Para obtener el título de:

M A T E M Á T I C O

Presenta:

Carlos Alberto Villarreal Noyola

Directora de tesis:

Dra. Bibiana Obregón Quintana

Ciudad Universitaria, Ciudad de México, México

C
D
C
(3,3)
(0,5)
D
(5,0)
(1,1)
A
B

Agradecimientos

A mis padres Carlos y Elsa
Por su apoyo en todo momento.

A los proyectos PAPIIT IN111822 y PAPIIT IN114724
Por el apoyo brindado para la conclusión de este proyecto, espero más científicos alcancen las mismas oportunidades.

Resumen

Se propone un nuevo modelo matemático que representa la propagación y persistencia de la corrupción en las sociedades. Se integra al juego evolutivo de (Ubeda & Dueñez-Guzman, 2010) la dinámica de redes de (Scatà et al., 2016).
Mediante simulaciones computacionales, se recrearon sociedades con distintas características, como la alta confianza entre las personas de Dinamarca, el estricto control de las autoridades en Singapur y el equilibrio inestable de la sociedad en México, logrando replicar su evolución ante diversos escenarios.
Los resultados revelan que la corrupción persiste cuando se controla la comunidad central de la red; sin embargo, para erradicarla, una intervención aleatoria es más efectiva que un ataque centralizado. Estos resultados ofrecen una justificación teórica para la implementación de mecanismos para desmantelar redes corruptas.

Abstract

A new mathematical model is proposed that represents the spread and persistence of corruption in societies. The evolutionary game of (Ubeda & Dueñez-Guzman, 2010) is integrated into the dynamics of (Scatà et al., 2016).
Through computer simulations, societies with different social characteristics were recreated, such as high trust among the people of Denmark, the strict control of the authorities in Singapore and the unstable balance of society in Mexico, managing to replicate their evolution in various scenarios.
The results reveal that corruption persists when the network's core community is controlled; however, to eradicate it, a random intervention is more effective than a centralized attack. These results offer a theoretical justification for the implementation of mechanisms to dismantle corrupt networks.

Bienvenidos al arte de conectar los puntos

Índice

C
D
C
(3,3)
(0,5)
D
(5,0)
(1,1)
A
B

Introducción

La corrupción es un problema persistente dentro de la sociedad, genera inquietudes sobre la estructura de nuestras instituciones y de la misma naturaleza humana. Se considera un término "sombrilla", ya que engloba una gran cantidad de conductas dentro de un mismo concepto. A menudo solo vemos sus síntomas, como un soborno o un contrato otorgado a una empresa fantasma, pero no visualizamos la estructura oculta que la mantiene operando. Ante tal desafío nace una pregunta: ¿cómo podemos combatirla?

La respuesta tradicional, basada en la persecución de casos individuales, se ha mostrado insuficiente. En los últimos años gracias a la popularización de los ordenadores personales los modelos matemáticos han emergido como herramientas en la lucha contra la corrupción. En especial la modelación mediante teoría de redes y teoría de juegos.

Este trabajo recoge el esfuerzo de varios investigadores en el estudio y lucha contra el fenómeno. Se recrean y ponen a prueba los límites de sus modelos matemáticos, para comprender no solo sus aciertos, también sus defectos: ¿qué aspectos cruciales del fenómeno dejan sin explicar?

Finalmente, a partir de este análisis crítico, esta tesis propone un nuevo modelo, donde las estrategias empleadas para reducir la corrupción puedan ser aplicadas en nuestros sistemas políticos y sociales.

Todos los modelos presentados en este trabajo pueden ser consultados y descargados libremente para su análisis y mejora (*A.4 - Código*).

El objetivo no es solo académico; es ofrecer una nueva lente a través de la cual podamos diseñar políticas más inteligentes y efectivas para desmantelar las redes de corrupción desde sus raíces.

Objetivo principal

Crear una dinámica de propagación de información en redes complejas que emule el comportamiento documentado en casos de corrupción.

Objetivos secundarios

- Modelar por medio de la teoría de redes escándalos de corrupción contemporáneos en México.
- Crear una red que generalice las características de las redes analizadas.
- Encontrar modelos de difusión que se ajusten a la propagación de la corrupción en la red creada.
- Caracterizar mediante teoría de juegos las sociedades exitosas en la lucha contra la corrupción, basándose en índices de percepción de la corrupción y de desarrollo.
- Recrear el caso mexicano y encontrar estrategias adecuadas para reducir la propagación de la corrupción dentro del modelo.

Justificación de la investigación

Las matemáticas no pueden explicar por qué una persona decide ser corrupta, pero puede usarse para analizar los datos globales e identificar patrones en nuestros sistemas sociales.

En particular, la teoría de redes y la teoría de juegos ofrecen nuevos enfoques al combate de la corrupción, nos ayudan a desarrollar estrategias novedosas; por ejemplo, permiten identificar si una institución o empresa es vulnerable, también ayudan en la creación de redes de confianza y la remoción de algún funcionario clave.

C D
C (3,3) (0,5)
D (5,0) (1,1)
A B

1 - Antecedentes

1.1 - El combate a la corrupción a lo largo del tiempo

Las primeras estrategias documentadas para contrarrestar la corrupción yacen en el código de Hammurabi escrito hace casi 3,800 años en la antigua Mesopotamia, en él se discuten reglas sobre el soborno y la malversación de fondos públicos.
En la antigua Roma los emperadores usaron los sobornos para gobernar sobre los funcionarios, originando inestabilidad política a largo plazo por la pérdida de confianza en su gobierno y con ello la fractura del imperio. Un caso similar sucedió en la iglesia católica durante la edad media, cuando los obispos usaron su poder para enriquecerse, lo cual culminó en la reforma protestante. El filósofo Montesquieu mediante un estudio exhaustivo de la caída del imperio romano, propuso una serie de reformas para combatirla: la separación de poderes, la libertad de expresión y la educación pública, ideas que hoy en día perduran (Geltner et al., 2018).
En la actualidad, la corrupción sigue siendo un problema relevante. Acorde al Índice de Percepción de la Corrupción (*Figura 1.1*) creado por la organización no gubernamental Transparencia Internacional, en el 2023 la nación con el menor índice de corrupción fue Dinamarca junto con varios países escandinavos, los cuales implementaron estrategias similares, posteriormente en el índice les siguió Singapur el cual implementó una estrategia distinta pero igualmente efectiva (Transparencia Internacional, 2024).

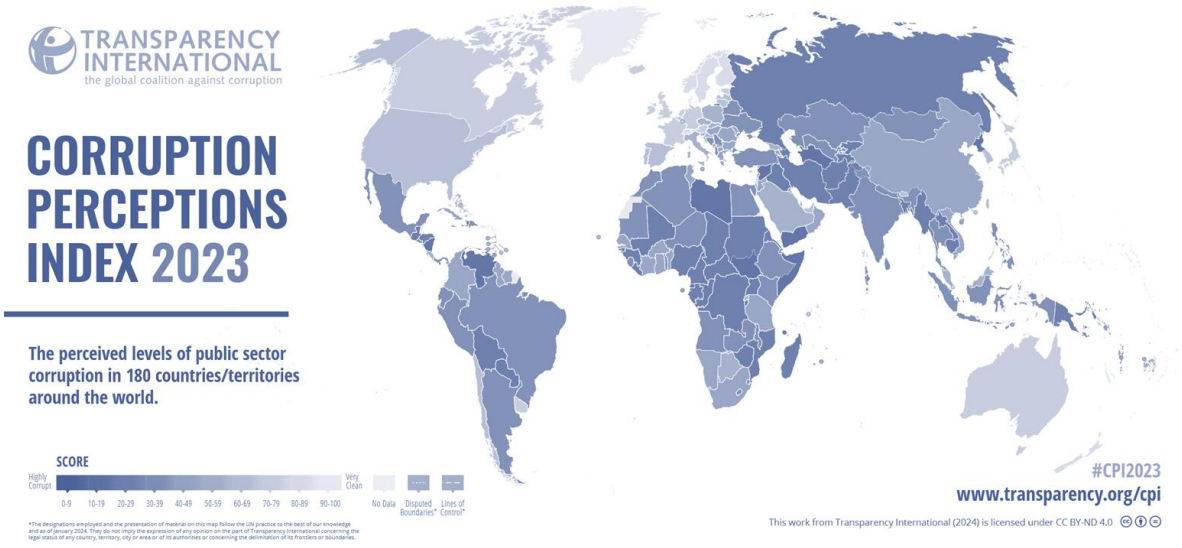


Figura 1.1 – Dinamarca y los países escandinavos se posicionaron en los lugares menos corruptos del índice ocupando los primeros lugares, México se posicionó en el lugar 126 de 180. Siria, Venezuela y Somalia fueron evaluados como los más corruptos ocupando los últimos lugares del índice (Transparencia Internacional, 2024).

1.2 - Historias de éxito

1.2.1 - Caso Dinamarca

Se puede comprender el éxito de Dinamarca a través de sus normas sociales y acceso a la información. El gobierno de Dinamarca implementó la ley de transparencia financiera, esta ley obliga a los funcionarios públicos a declarar sus bienes e ingresos cada mes, mismos que pueden ser consultados en línea por cualquier ciudadano. Además, todos los movimientos de dinero en el país pueden ser consultados, un ejemplo son las publicaciones en internet sobre la situación financiera de cada empresa.

Desde un punto de vista social, los daneses son conocidos por expresar sus opiniones directamente, la sinceridad es un valor que permite una fuerte confianza por sus compatriotas. Tal confianza se ve reflejada hacia el gobierno, pues la mayoría de la población paga sus impuestos voluntariamente.

El éxito de Dinamarca en la lucha contra la corrupción ha sido el promover la transparencia financiera y la rendición de cuentas, con ello los gobiernos pueden crear un clima de confianza, dificultando a los funcionarios públicos participar en la corrupción (Chêne, 2011).

1.2.2 - Caso Singapur

"Si quieres derrotar la corrupción, debes estar preparado para enviar a la cárcel a tus amigos y familiares"

Lee Kuan Yew - Primer ministro de Singapur de 1959 a 1990.

Singapur es apodado por sus vecinos como "el imperio de la ley" gracias a las políticas impulsadas por el primer ministro Lee Kuan Yew hace varias décadas.

Implementó una serie de reformas al inicio de su administración, como la revisión continua de las cuentas bancarias de todos los empleados del gobierno, si existían indicios suficientes para ser condenado, se perdía el derecho a la pensión y no podían postularse a un cargo público nunca más.

También incrementó las condenas relacionadas a los actos de corrupción, efectuó la rotación de personal para evitar lazos cercanos y creó una oficina autónoma para investigar únicamente actos corruptos. Actualmente si los servidores públicos incurren en algún delito relacionado a estos temas, se les priva de su empleo, pensión y seguridad social. La sentencia en algunos casos puede llegar hasta la pena de muerte. Con todas estas medidas Lee Kuan transformó a Singapur en un país próspero en tan solo tres décadas (Gossaín, 2019).

1.3 - Un enfoque matemático

En los últimos años la lucha contra la corrupción ha encontrado una nueva aliada, la matemática, prueba de ello son la gran cantidad de artículos, libros y recursos recientes que tratan sobre el tema, se comparten algunos ejemplos en el anexo *Recursos recomendados*.

1.3.1 - Una perspectiva mexicana

La UNAM creó el Observatorio de la Corrupción e Impunidad (OCI), el cual aborda el tema mediante enfoques multidisciplinarios, colabora en conjunto con la Facultad de Ciencias Políticas y Sociales (FCPS) y con el Instituto de Investigaciones en Matemáticas Aplicadas y en Sistemas (IIMAS) analizando problemáticas nacionales e internacionales.

1.3.1.1 - Casos de estudio relevantes

1.3.1.1.1 - La red de Duarte (2010)

Figura 1.2 - Red de corrupción de las compañías e individuos (Luna-Pla y Nicolás-Carlock, 2020)

Los autores estudian un escándalo de corrupción mexicano que involucró a cientos de compañías utilizadas para malversar fondos (*Figura 1.2*), analizan características estructurales y temporales de la red (Luna-Pla y Nicolás-Carlock, 2020).

1.3.1.1.2 - La estafa maestra (2013)

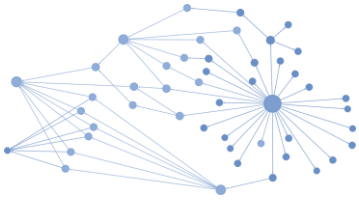


Figura 1.3 - Red de funcionarios, universidades y empresas correspondiente a la estafa maestra.

Se usó el mismo esquema que en el caso anterior, pero a una escala nacional, por medio de universidades públicas como intermediarias, creando una red (*Figura 1.3*) que involucró a diferentes organismos públicos (Castillo et al., 2017).

C
D
C
(3,3)
(0,5)
D
(5,0)
(1,1)
A
B

2 - Marco teórico

2.1 - ¿Qué es la corrupción?

La corrupción es un fenómeno complejo, acorde a (Transparencia Internacional, 2024) se define como:

> *"El abuso del poder confiado a una persona para obtener una ganancia privada".*

Esta definición conceptual encapsula dos conceptos clave que una definición matemática debe tomar en cuenta:

- El **poder** implica una red de relaciones, jerarquías y acceso a recursos.
- La **ganancia** es resultado de una decisión estratégica donde un individuo evalúa beneficios y riesgos.

La formulación matemática ha seguido principalmente dos caminos para recrear este comportamiento.
Por un lado, la teoría de redes ofrece un marco para analizar la estructura de la corrupción. Este enfoque permite identificar patrones, actores clave y vulnerabilidades en las redes sociales donde opera, como se explorará en *2.3.4 - Modelo de Martins et al., redes de tipo corruptas (2022).*

Por otro lado, la teoría de juegos proporciona las herramientas para modelar la dinámica de la decisión. Se enfoca en las interacciones estratégicas, donde los individuos evalúan los beneficios de un comportamiento deshonesto frente a las posibles sanciones, dicho enfoque se discutirá en *2.7.2 - El juego de la corrupción.*

Cada enfoque tiene sus limitaciones, pero al combinarlos es posible representar con mayor fidelidad el fenómeno.

2.2 - Breve panorama de la teoría de redes

2.2.1 - Los puentes de Königsberg (1736)

Los habitantes de Königsberg, Prusia (hoy en día llamada Kaliningrado, Rusia) se preguntaron si existía un camino capaz de recorrer la ciudad pasando por sus siete puentes en una única ocasión y regresando al mismo punto de inicio. El matemático suizo Leonhard Euler demostró que no era posible y para ello desarrolló el concepto de *grafo*.

> ***Definición 1 -*** *Grafo*
> *Una pareja ordenada $G = (V,E)$ compuesta por un conjunto finito $V = \{v_1, \dots, v_n\}$ de vértices o nodos y un conjunto $E = \{e_1, \dots, e_n\}$ de aristas o enlaces.*

En pocas palabras, es una representación abstracta (*Figura 2.1*) de las relaciones que existen entre objetos, los nodos representan los objetos a conectar y los enlaces su relación.

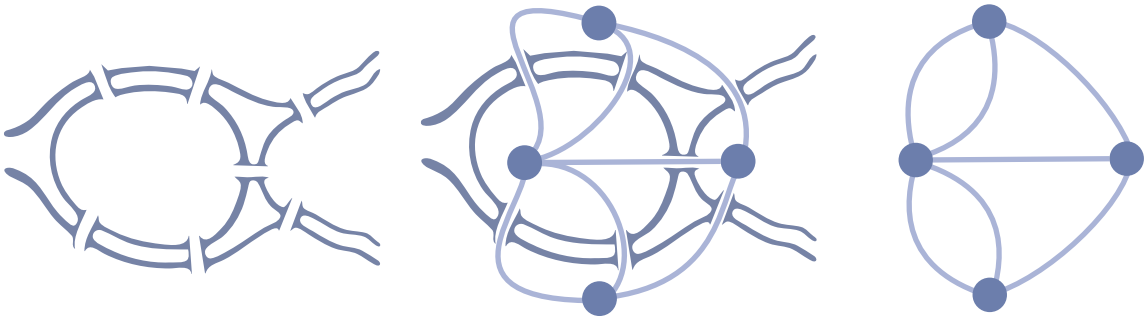


Figura 2.1 - Grafo ideado por Euler para representar el problema. La ciudad se divide en cuatro zonas representadas por nodos y los siete puentes como los enlaces.

Se percató que lo más relevante era únicamente el número de puentes. Si llegaba a un nodo desde un enlace el único modo de moverse era mediante un enlace distinto, es decir, es necesario un número par de enlaces si se quieren recorrer todos los puentes o también si existen dos vértices con un número impar de aristas.

Este problema fundó dos ramas de la matemática, la topología, la cual se enfoca únicamente en las propiedades de las conexiones de los objetos sin tomar en cuenta sus medidas y la teoría de grafos de la cual nació posteriormente la teoría de redes.

2.2.2 - Jacob Moreno propone el sociograma (1930)

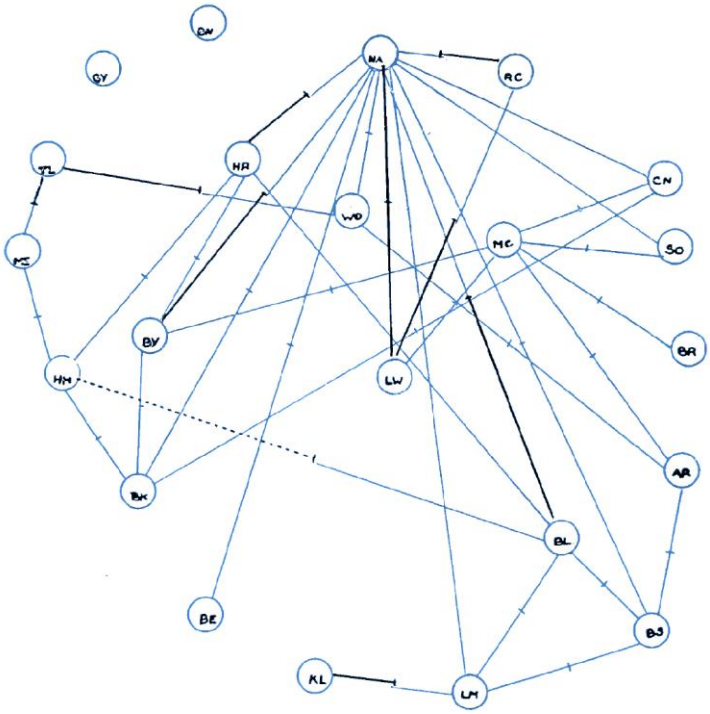


Figura 2.2 - Sociograma propuesto por Jacob Moreno (Moreno, 1934).

El sociólogo rumano Jacob Levy Moreno propuso representar las relaciones entre los miembros de un grupo por medio de un grafo (*Figura 2.2*), donde los nodos simbolizan las personas y los enlaces las relaciones del grupo. El objetivo fue analizar la cohesión, el liderazgo y la estructura del grupo (Moreno, 1934).

2.2.3 - Stanley Milgram y los 6 grados de separación (1967)

Milgram conjeturaba que solo seis contactos nos separaban de cualquier persona en el mundo. Desarrolló un experimento para comprobarlo, envió 60 cartas pidiendo que contactaran a una persona específica en Massachussets, los destinatarios debían enviar la carta a quien consideraban que podía conocerla, el promedio de envíos para alcanzar a la persona indicada resultó ser de cinco a siete contactos. Fue un experimento muy reducido, el cual se criticó en su momento por la comunidad matemática, pero popularizó la hipótesis (Milgram, 1967).

2.2.4 - Emerge la complejidad

El concepto de sistema complejo nace a mediados del siglo XX como resultado de la teoría general de sistemas y la teoría del caos. En estos sistemas el enfoque *reduccionista* (donde cada componente se analiza de manera aislada) no funcionaba para trabajar con los fenómenos. Fue necesario adoptar otro enfoque, donde se consideró cada parte del sistema como un todo, el enfoque *holístico* (Mitchell, 2009). Las propiedades fundamentales de un sistema complejo son:

- **Emergencia**: la aparición de nuevas propiedades o patrones debido a las interacciones no lineales entre los componentes del sistema.
- **Autoorganización**: la formación espontánea de estructuras ordenadas sin intervención externa.
- **Autorregulación**: la capacidad del sistema para mantener su estabilidad sin la necesidad de estímulos externos.
- **Adaptabilidad**: la habilidad del sistema para ajustarse a su entorno y modificar su estado en consecuencia.
- **No linealidad**: la imposibilidad de descomponer el sistema en elementos independientes para su análisis.

Este innovador enfoque ha facilitado la comprensión de diversos fenómenos, entre ellos el clima, los sistemas biológicos, las redes sociales y los mercados financieros.

2.2.5 - Nace la teoría de redes

> ***Definición 2 -*** *Red*
> *Un grafo que describe las interacciones de un sistema complejo.*
> *(Barabási A.-L. , 2016)*

Comenzó a tomar forma como una disciplina interdisciplinaria al ser utilizada para modelar fenómenos complejos en diversas áreas, como la biología, la informática, la sociología y la física, enfocando su análisis en las estructuras y patrones dentro de los sistemas, la distribución de la información, la propagación de ideas y la organización de los sistemas.
El desarrollo las computadoras y el avance en la recopilación de grandes volúmenes de datos fueron factores clave en la consolidación de la teoría de redes como un campo fundamental de estudio en las ciencias aplicadas desde finales del siglo XX, actualmente es una rama viva de las matemáticas (Barabási A.-L. , 2016).

2.3 - Modelos de redes

Son estructuras matemáticas usadas para analizar cómo se forman y evolucionan las redes, cada modelo crea una red con características únicas.

2.3.1 - Modelo de Erdös-Rényi, redes aleatorias (1959)

Figura 2.3 - Ejemplo de red aleatoria.

Paul Erdös & Alfred Renyi presentaron el primer algoritmo (Erdös & Rényi, 1959) para generar redes aleatorias *(Figura 2.3)*, fue usado durante décadas para crear redes complejas de una manera sencilla y práctica, posteriormente se desarrollaron algoritmos más sofisticados.

2.3.2 - Modelo de Watts-Strogatz, red de mundo pequeño (1998)

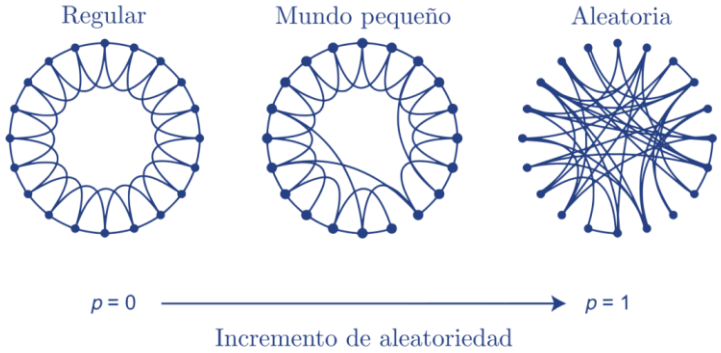


Figura 2.4 - Proceso con el cual se obtiene la red de mundo pequeño (Strogatz y Watts, 1998).

Duncan Watts y Steven Strogatz propusieron la *red de mundo pequeño* (*Figura 2.4*), su principal propiedad radica en que cualquier nodo está conectado a los demás mediante un número pequeño de enlaces entre ellos (Strogatz y Watts, 1998). Sus propiedades se encuentran en fenómenos del mundo real como las redes de neuronas, redes de aeropuertos, redes sociales (*2.2.3 - Stanley Milgram y los 6 grados de separación (1967)*) entre otras.

2.3.3 - Modelo de Barabási-Albert, redes libres de escala (1999)

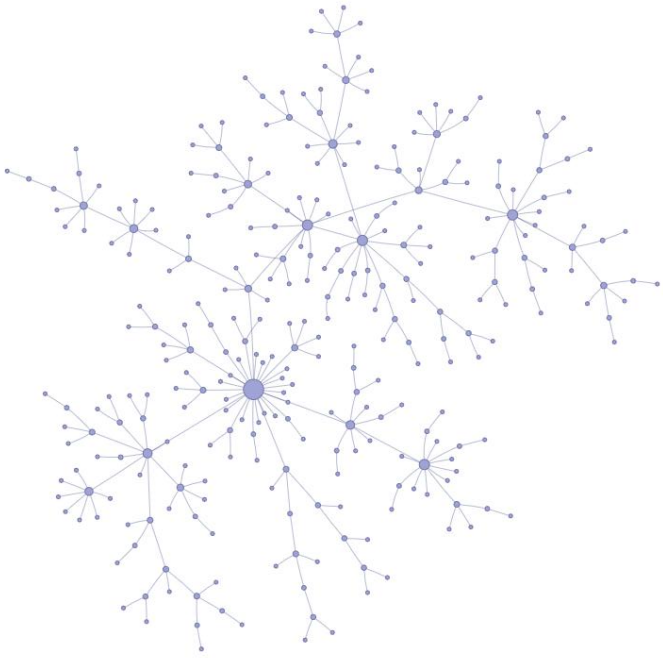


Figura 2.5 - Ejemplo de una red tipo Barabási-Albert.

Albert-László Barabási y Réka Albert propusieron un algoritmo para crear redes libres de escala (*Figura 2.5*) mediante un mecanismo de crecimiento y conexión preferencial (Barabási & Albert, 1999). En este modelo, los nuevos nodos se añaden progresivamente a la red y se conectan con mayor probabilidad a nodos que tienen más enlaces, generando una estructura donde unos pocos nodos concentran la mayoría de las conexiones.

Este proceso da lugar a una distribución de grado que sigue una *ley de potencias* (*2.4.1 - Grado y distribución de grado*), característica de muchas redes reales como Internet, redes sociales y sistemas biológicos. Este modelo permite comprender fenómenos como la robustez estructural y la propagación de información.

2.3.4 - Modelo de Martins et al., redes de tipo corruptas (2022)

Martins y colaboradores en recrearon redes asociadas a escándalos de corrupción en España y Brasil (*Figura 2.6*). Identificaron las propiedades comunes entre ellas y desarrollaron un modelo que replica las redes estudiadas (Martins et al., 2022).

En *3.1.1.1 - Propiedades que recrea el modelo de Martins et al.* se explica a profundidad las propiedades más importantes que recrea el modelo.

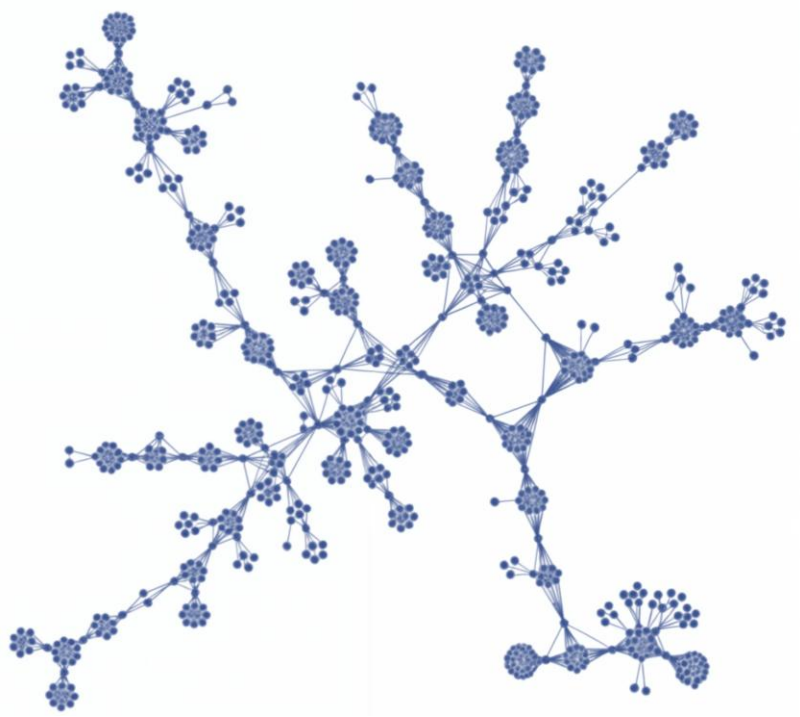


Figura 2.6 - Red generada por los investigadores, sus propiedades se sitúan dentro de los mismos rangos que las redes corruptas documentadas. La red tiene una estructura con comunidades claramente definidas e interconectadas entre sí con un coeficiente de agrupamiento alto y una distribución de grado exponencial (Martins et al., 2022).

2.4 - Métricas y propiedades fundamentales de redes

Las métricas son medidas para analizar, comprender y comparar el comportamiento de cualquier red.

2.4.1 - Grado y distribución de grado

El *grado* de un nodo se define como el número de conexiones o enlaces que tiene con otros nodos dentro de una red. La *distribución de grado* describe la probabilidad $P(k)$ de seleccionar al azar un nodo con grado k. En ciertas redes, esta distribución sigue una ley de potencias:

$$P(k) \sim k^{-\gamma}$$

Donde

- γ es el exponente de la ley (también conocido como índice de escala).

En las cuales la mayoría de los nodos tienen pocas conexiones, mientras que unos pocos nodos, concentran un número alto de enlaces. Un rasgo característico de redes libres de escala (*2.3.3 - Modelo de Barabási-Albert, redes libres de escala (1999)*).

2.4.2 - Comunidades

Una comunidad es un grupo de nodos que están conectados entre sí con una alta densidad y tienen pocas conexiones con nodos fuera del grupo.

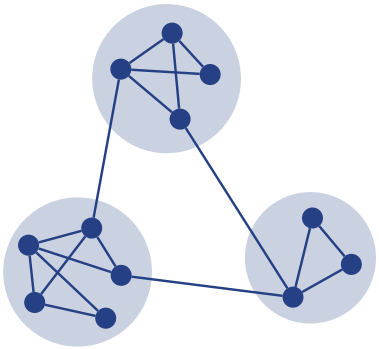


Figura 2.7 – Ejemplo de 3 comunidades en una red.

Una red está dividida en comunidades (*Figura 2.7*) cuando es posible detectar estas estructuras mediante un algoritmo como *el método del corte mínimo* o por *agrupamiento jerárquico*. Seleccionar el mejor algoritmo para grandes redes es un problema abierto hoy en día.

2.4.3 - Camino y longitud de camino media

Un *camino* es una secuencia de nodos y enlaces que conectan a dos nodos en la red. La *longitud de camino media* es el promedio de las distancias más cortas entre cualquier par de nodos. Se define como:

$$L = \frac{1}{N(N-1)} \sum_{i \neq j} d(i,j)$$

Donde:

- $d(i,j)$ es la longitud del camino más corto entre los nodos i y j.
- N es el número total de nodos.

Una longitud de camino media pequeña significa que la información puede propagarse de manera muy eficiente a través de la red.

2.4.4 - Agrupamiento (clustering)

El *coeficiente de agrupamiento* mide la tendencia de los nodos a formar grupos densos. Se define como:

$$C_i = \frac{2 \times \Delta_{\mathrm{N}_i}}{k_i \times \ k_i - 1}$$

Donde:

- Δ_{N_i} es el número de grafos completos de 3 nodos alrededor del nodo i.
- k_i es el grado del nodo i.

Esta medida cuantifica la probabilidad de que dos vecinos de un nodo también sean vecinos entre sí.

El *coeficiente de agrupación global* de la red es el promedio de los coeficientes de agrupación locales de todos los nodos. Y está definido como:

$$C = \frac{1}{N} \sum_{i=1}^{N} C_i$$

Donde:

- N es el número total de nodos en la red.

Un alto coeficiente de agrupamiento es una característica distintiva de las redes de mundo pequeño.

2.4.5 - Propiedad de mundo pequeño

Una red tiene la propiedad de mundo pequeño si su longitud de camino media es corta y tiene alto coeficiente de agrupamiento. Esta estructura es común en redes sociales (*2.3.2 - Modelo de Watts-Strogatz, red de mundo pequeño (1998)*).

2.4.6 - Densidad

La densidad de una red mide que tan conectada está entre sí, comparando el número de enlaces existentes con el número máximo de enlaces posibles.

$$\text{Densidad} = \frac{2 \times \text{ Número de enlaces}}{\text{Número de nodos} \times (\text{Número de nodos} - 1)}$$

Una red con densidad alta se asemeja a una red completa, mientras que una densidad baja indica una red dispersa. Esta métrica útil para caracterizar la cohesión de pequeños grupos en estructuras sociales más grandes.

2.4.7 - Asortatividad

El *coeficiente de asortatividad* describe la preferencia de los individuos a conectarse con otros que son similares a ellos en algún aspecto (McNulty, 2022).
Definido como:

$$r = \frac{\sum_{jk} jk(e_{jk} - q_j q_k)}{\sigma_q^2}$$

Donde:

- $\sum_{jk} jk(e_{jk} - q_j q_k)$ es la suma de las diferencias entre la probabilidad conjunta observada (e_{jk}) de la red y la probabilidad conjunta esperada si no hubiera correlación $(q_j q_k)$ entre los nodos.
- $\sigma_q^2 = \sum_k k^2 q_k - \left(\sum_k k q_k\right)^2$ es el valor de normalización, representa la variancia de la distribución de grado normalizada (q_k).

El valor de $\boldsymbol{r}$ estará comprendido entre los valores -1 y 1,

- $r = 1$, indica una red asortativa (nodos con alto grado se conectan a otros nodos con alto grado).
- $r = 0$, indica una red sin correlación de grado (no hay preferencia en la conexión de nodos).
- $r = -1$, indica una red disortativa (nodos con alto grado se conectan a nodos con bajo grado).

Este concepto ayuda a entender cómo se forman y aíslan los grupos de poder dentro de una red.

2.5 - Procesos de propagación de información en redes

Son procesos que permiten simular y entender cómo se disemina la información dentro de una red, son fundamentales para analizar la difusión de enfermedades, ideas o comportamientos.

2.5.1 - Dinámica del umbral (1978)

Mark Granovetter propuso que las decisiones individuales no dependen solo de las preferencias personales, también están fuertemente influenciadas por las decisiones del círculo cercano (Granovetter, 1978). Esta dinámica es útil para explicar fenómenos sociales, en particular la propagación de comportamientos dentro de una comunidad (*Figura 2.8*).

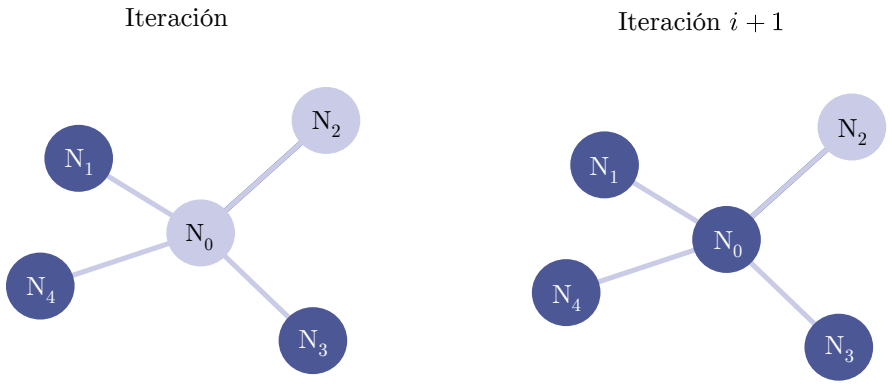


Figura 2.8 - Se ejemplifica una iteración del modelo del umbral, el nodo N0 con $\theta_{N_0} = \frac{1}{2}$ pasó de un estado inactivo a uno activo, debido a que tres de cuatro nodos conectados a él tenían un estado activo, sobrepasado su umbral.

La dinámica funciona de la siguiente manera:

- A cada nodo N_i se le asigna un umbral θ_{N_i}, porcentaje que representa la cantidad de influencia que necesita recibir de sus vecinos para adoptar un nuevo estado.
- Cada nodo N_i adquirirá el comportamiento de sus vecinos si el porcentaje es mayor que su umbral θ_{N_i}.
- Se repite el proceso hasta que la red llegue a un equilibrio.

Esta dinámica ayuda a explicar cómo las decisiones inicialmente aisladas se transforman en patrones colectivos dentro de una comunidad.

2.5.2 - Dinámica del umbral con perfiles (2018)

En (Milli et al., 2018) refinan la dinámica del umbral, incluyen preferencias individuales dentro de la adopción del nuevo comportamiento. Se añaden nuevos roles, como individuos que son reacios a adoptar un comportamiento e individuos que espontáneamente lo adoptan.

2.5.3 - Dinámica del balance social (2005)

En (Antal et al., 2005) se estudia la evolución de redes sociales (*Figura 2.9*) por medio de una dinámica que simula las relaciones de amistad y enemistad en una población.

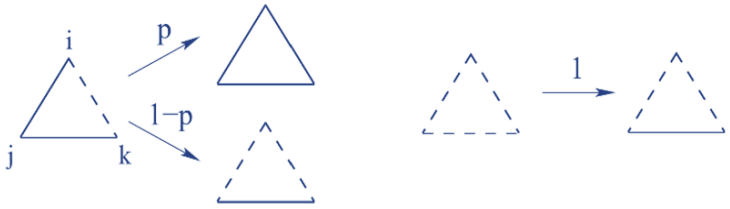


*Figura 2.9 - Los nodos **i**, **j**, **k** representan personas, un grupo de tres personas es llamada **triada** representada mediante △, la línea continua representa amistad y la línea punteada enemistad, **p** representa la probabilidad de crear enlaces amistosos. Existen dos casos de configuraciones inestables, cuando coexiste un enemigo en una triada $\triangle_1$ (imagen izquierda) o cuando todos son enemigos $\triangle_3$ (imagen derecha). Los triadas evolucionan a configuraciones estables, cuando todos son amigos $\triangle_0$ o cuando coexisten dos enemigos y un amigo $\triangle_2$ (Antal et al., 2005).*

El modelo funciona de la siguiente manera:

- Se elige una tríada al azar:
    - Si la tríada está desequilibrada ($\triangle_1$ o $\triangle_3$), mediante la probabilidad $\boldsymbol{p}$ se cambia el signo de uno de sus enlaces para equilibrarla.
    - Si la tríada está equilibrada ($\triangle_0$ o $\triangle_2$), no hay cambio.
- Se repite el proceso hasta que la red llegue a un equilibrio.

Cabe destacar que son reglas muy sencillas basadas en la experiencia cotidiana, su objetivo fue comprender y caracterizar los equilibrios encontrados.

2.6 - Breve panorama de la teoría de juegos

La teoría de juegos es una rama de las matemáticas que estudia las situaciones de conflicto y cooperación entre individuos racionales.
Fue propuesta por el matemático húngaro John Von Neumann y por Oskar Morgenstern en 1944, posteriormente John Nash contribuyó a la rama con el desarrollo de teoremas matemáticos que permitieron aplicar la teoría de juegos a problemas con más de una solución, conocidos como juegos con equilibrios múltiples. Actualmente, la teoría de juegos se usa en diferentes campos, incluyendo economía, ciencias políticas, psicología y ciencias de la computación (Zapata Lillo, 2016).

2.6.1 - El dilema del prisionero

Una situación hipotética popularizada por Albert W. Tucker y Harold Kuhn, donde dos personas son sentenciadas por un delito, si ambos confiesan serán sentenciados a tres años de prisión, si ninguno confiesa únicamente obtendrán un año de condena.
Lo interesante pasa cuando una persona confiesa el delito y la otra persona no lo hace, quien confiesa sale libre pero el compañero que decidió callar es sentenciado a cinco años de prisión, ninguno conoce que decisión tomará su compañero.

Tabla 2.1 - Matriz de pagos que representa el dilema del prisionero.

	Confesar	No confesar
Confesar	3	0
No confesar	5	1

Si ambos jugadores eligieran únicamente su propio beneficio entonces confesarían, lo cual los llevaría a ambos a la cárcel. Pero si buscaran el beneficio comunitario entonces no confesarían, con ello reduciendo considerablemente la sentencia a la otra persona (Zapata Lillo, 2016). Aquí florece un concepto importante dentro de la teoría de juegos, aquellas personas que deciden favorecer el bien común son llamados *jugadores cooperativos* y aquellas que únicamente ven por su propio beneficio son llamados *jugadores no cooperativos*.

2.7 - Juegos evolutivos

Es una extensión de la teoría de juegos que incorpora principios de la evolución para analizar cómo ciertas estrategias se mantienen o desaparecen dentro de una población a lo largo del tiempo. A diferencia del enfoque clásico, donde se asume que los individuos toman decisiones de manera racional, las estrategias se ajustan en función de su éxito relativo en interacciones con otros (Cressman, 2003). El sistema busca una *estrategia evolutivamente estable*, análogo al equilibrio de Nash en juegos clásicos.

2.7.1 - Dinámica del replicador

La dinámica del replicador es un modelo matemático utilizado en la teoría de juegos evolutivos para describir cómo cambian las frecuencias de diferentes estrategias en una población. Donde las estrategias que consiguen un pago por encima del promedio de la población aumentan en frecuencia, mientras que las estrategias con un pago por debajo del promedio disminuyen (Cressman, 2003).

La ecuación del replicador en su forma continua es:

$$\frac{dx_i}{dt} = x_i[f_i(\boldsymbol{x}) - \overline{f}(\boldsymbol{x})]$$

Donde:

- $\frac{dx_i}{dt}$ es la tasa de cambio de la frecuencia de la estrategia i con respecto al tiempo.
- x_i es la frecuencia actual de la estrategia i.
- $f_i(\boldsymbol{x})$ es el pago medio de la estrategia i cuando se juega contra la población actual de n estrategias $\boldsymbol{x} = (x_1, x_2, \dots, x_n)$.
- $\overline{f}(\boldsymbol{x})$ es el pago medio de toda la población.

Esta dinámica es empleada para estudiar *2.7.2 - El juego de la corrupción*. Un análisis a profundidad se puede consultar en *A.1 - Replicador dinámico del juego de la corrupción.*

2.7.2 - El juego de la corrupción

En (Ubeda & Dueñez-Guzman, 2010) extienden el dilema del prisionero a un juego con cuatro estrategias (*Figura 2.10*). Formulado para permitir las asimetrías de poder entre personas. El juego se analizó bajo el marco de los juegos evolutivos para conocer sus equilibrios y condiciones de estabilidad.

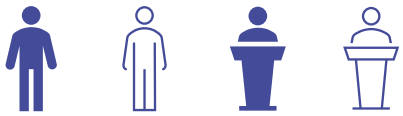


Figura 2.10 - Estrategias propuestas en el juego:
C (Civil cooperativo): *Los ciudadanos que eligen la estrategia que beneficie a ambos jugadores. Los que siguen las leyes para el bien común de la sociedad.*
D (Civil no cooperativo): *Los ciudadanos que eligen la estrategia que únicamente los beneficie a ellos sobre los demás. Personas que rompen las leyes.*
H (Poder cooperativo): *Individuos con poder que eligen la estrategia que beneficie a ambos jugadores. Asociado con las personas honestas en cargos públicos.*
K (Poder no cooperativo): *Individuos con poder que no cooperan con los demás, es decir, las personas corruptas en puestos de gobierno o empresas.*

La matriz de pagos (*Tabla 2.2*) está constituida por los valores r, s, t correspondientes a *2.6.1 - El dilema del prisionero*, los valores p y c pertenecen al *juego del castigo*, una extensión del dilema del prisionero en donde los policías intervienen y finalmente *el juego de la corrupción* con los valores q y d.

Tabla 2.2 - Matriz de pagos del juego de la corrupción (Ubeda & Dueñez-Guzman, 2010).

	Ciudadano cooperativo (C)	Ciudadano no cooperativo (D)	Poder cooperativo (H)	Poder no cooperativo (K)
Ciudadano cooperativo (C)	r	$-s$	r	$-s$
Ciudadano no cooperativo (D)	t	0	$t-p$	$-p$
Poder cooperativo (H)	r	$-s-c$	r	$-s-d$
Poder no cooperativo (K)	t	$-c$	$t-q$	$-q-d$

2.7.2.1 - Interpretación de los pagos

Estos pagos se crearon con la finalidad de permitir castigos diferentes entre los ciudadanos (cooperativos o no) y las personas con poder (cooperativas o no cooperativas), se considera que el castigo dado a un ciudadano no cooperativo $(-p)$ es diferente del castigo a una persona con poder no cooperativa $(-q)$ y el costo para castigar a un ciudadano no cooperativo $(-c)$ puede diferir del costo de castigar a una persona con poder no cooperativa $(-d)$.

2.7.2.1.1 - Dilema del prisionero

- $\boldsymbol{r}$, recompensa por la cooperación mutua, interacciones sociales comunes entre personas que respetan las leyes (acuerdos, negocios, etc.).
- $-\boldsymbol{s}$, pérdida por ser cooperador cuando el otro jugador no lo es, es decir una ruptura del contrato social (cuando una persona es usada en beneficio de otra, sufrir un delito, etc.).
- $\boldsymbol{t}$, ganancia por no cooperar cuando el otro jugador es cooperador (cuando una persona se aprovecha de las personas, cometer un delito, etc.).
- $\mathbf{0}$, la ganancia es nula entre dos ciudadanos no cooperativos ya que no existe confianza entre ellos y no hacen tratos. Se considera un punto muerto. Es importante notar que, a diferencia de la matriz de pagos clásica (*Tabla 2.1*), aquí se introducen pagos negativos para representar castigos y costos, el valor 0 se convierte en la línea base de referencia, manteniendo la jerarquía de pagos del dilema del prisionero $t > r > 0 > -s$.

2.7.2.1.2 - Juego del castigo

- $-\boldsymbol{p}$, el castigo a los civiles por romper las leyes, por ejemplo, los años de cárcel por robo o delitos mayores.
- $-\boldsymbol{c}$, el costo por hacer cumplir las leyes, parametriza numéricamente la dificultad para procesar delitos, comúnmente llamado *el estado de derecho*, una sociedad con un buen estado de derecho tendrá bajos costos, mientras que en una sociedad con impunidad el costo será mayor, ya que se necesita de un mayor esfuerzo por hacer valer la ley.

2.7.2.1.3 - Juego de la corrupción

- $-\boldsymbol{q}$, castigo infligido a una persona con poder, las leyes contra la corrupción asignan los castigos, dependerá del país y sus leyes la severidad de la sanción aplicada.
- $-\boldsymbol{d}$, el costo de castigar a una persona con poder, dependerá del estado de derecho del país, para países en desarrollo aún con leyes anticorrupción el costo será más alto que en países desarrollados.

2.7.2.2 - Interpretación del juego

En una colmena de avispas el contrato social se conforma entre la avispa reina, la única delegada para poner huevos y las obreras (**C**), que trabajan para criar a sus hermanas. Sin embargo, existe una tentación biológica: una obrera puede romper el contrato (**D**) y poner sus propios huevos no fertilizados para pasar sus genes, obteniendo un beneficio genético (t) a costa de las leyes de la colonia. Para evitar este caos, existen obreras con el rol de policía. Si una policía es honesta (**H**), gastará energía y tiempo ($-c$) en inspeccionar las celdas y comerse los huevos ilegales de sus compañeras egoístas, aplicando un castigo ($-p$) al destruir su descendencia.

El problema surge cuando la avispa policía es corrupta (**K**), esto significa que tiene el poder de vigilar, pero decide no hacerlo para ahorrarse el costo energético (c) o, peor aún, usa su posición de poder para poner sus propios huevos impunemente (t). Si las obreras comunes notan que las policías no están vigilando (es decir, que el poder no cooperativo es la estrategia dominante), se desata el caos: todas empiezan a poner huevos de manera egoísta en lugar de trabajar, y la productividad de la colmena colapsa.

El juego demuestra que, al igual que en las sociedades humanas, si el costo de vigilar es muy alto o el castigo es ineficaz, la colmena evoluciona hacia un estado de anarquía.

2.8 - Uniendo redes con juegos

2.8.1.1 - Modelo de Scatà et al (2016)

En (Scatà et al., 2016) se exploró la evolución de la cooperación humana por medio de un modelo (*Figura 2.11*) que combinó *2.6.1 - El dilema del prisionero* y las redes libres de escala.

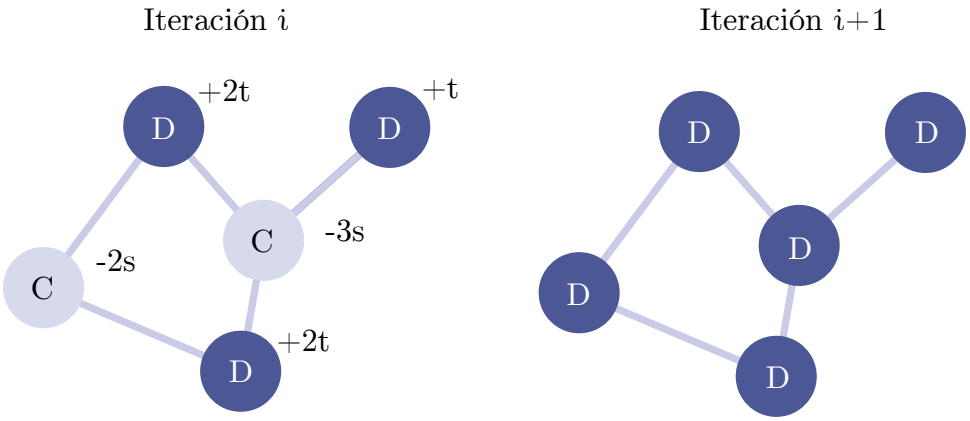


Figura 2.11 - Los nodos cooperadores (C) y no cooperadores (D) juegan al dilema del prisionero, en cada ronda los nodos calculan su ganancia total, en este ejemplo los nodos cooperadores (C) eligen cambiar su estrategia a la no cooperación (D) para obtener mejores pagos.

Con el objetivo de analizar como influía un grupo mínimo de cooperadores (C) (llamados *la masa crítica*) en la evolución de una red con población no cooperadora (D) fueron introducidos en tres configuraciones espaciales iniciales (*Figura 2.12*). El modelo incluyó el concepto de la homofilia (la tendencia a cooperar con individuos similares) como parte del análisis. Posteriormente se compararon los resultados con un segundo juego llamado *A.2 - El dilema de la nieve acumulada* donde los jugadores son más propensos a cooperar.

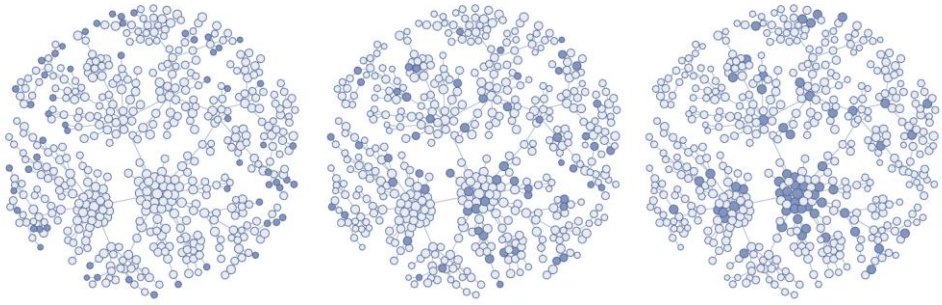


Figura 2.12 – Configuraciones espaciales iniciales de masa crítica cooperadora (en color obscuro) periférica, aleatoria y central (Scatà et al., 2016).

Este modelo destaca la interacción esencial entre la estructura social, la distribución inicial de cooperadores y el grado de homofilia como elementos clave para comprender la emergencia y estabilidad de la cooperación humana.

C
D
C
(3,3)
(0,5)
D
(5,0)
(1,1)
A
B

3 - Metodología

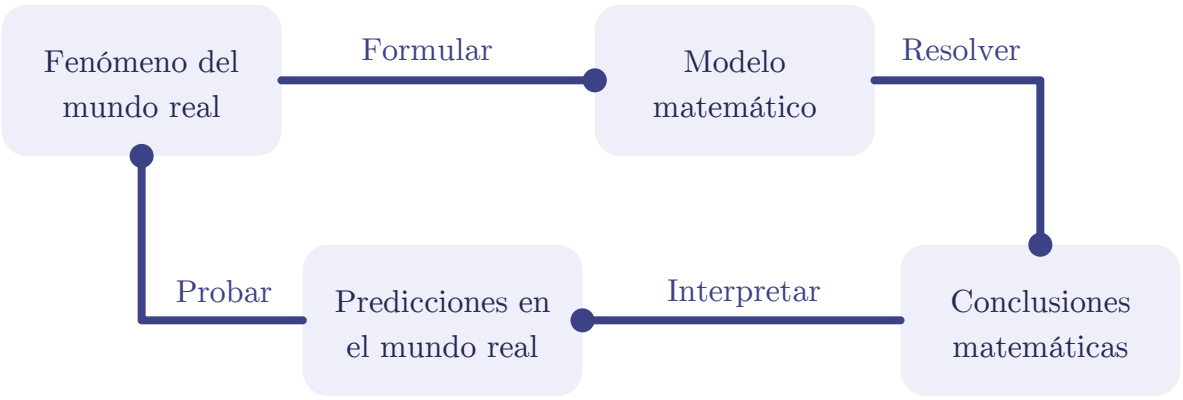


Figura 3.1 – Método de creación y mejora de modelos matemáticos (Stewart et al., 2020).

El proceso de creación del modelo (*Figura 3.1*), se estructuró en dos partes principales:

- *3.1 - Análisis e implementación de modelos previos:* Se recrearon tres modelos con el objetivo de entender la caracterización de la corrupción desde las distintas perspectivas de los investigadores, tomar las ideas más importantes y comprender las limitaciones de los modelos.

- *3.2 - Desarrollo de un nuevo modelo:* Con la experiencia adquirida en la recreación de los anteriores modelos, se propuso un modelo que replicara los aspectos sociales presentados en *1.2 - Historias de éxito* y *1.3.1.1 - Casos de estudio relevantes*.
  En *3.2.1.2 - Experimentación con diferentes escenarios* se probó la resiliencia de las sociedades respecto a un ataque corrupto, e intervenciones para erradicar dicho comportamiento en sociedades inicialmente corruptas.

Este enfoque integral permitió reproducir los resultados de investigaciones anteriores, y aportar una perspectiva nueva para comprender y atacar el fenómeno.

3.1 - Análisis e implementación de modelos previos

Se realizó un análisis detallado del fenómeno apoyado en la sección *1.3.1.1 - Casos de estudio relevantes*, posteriormente se replicaron los tres modelos clave: el modelo de red propuesto por (Martins et al., 2022), el juego de la corrupción planteado por (Ubeda & Dueñez-Guzman, 2010) y el modelo híbrido de (Scatà et al., 2016).

3.1.1 - Recreación de la red corrupta

El objetivo principal fue comprender el algoritmo de (Martins et al., 2022) para crear redes con una estructura similar a las asociadas en escándalos de corrupción y entender su estructura por medio de sus métricas.

3.1.1.1 - Propiedades que recrea el modelo de Martins et al.

Acorde a los datos recabados por los investigadores, las redes corruptas de Brasil y España presentaron características muy similares en su estructura (*Tabla 3.1*).

Tabla 3.1 – Métricas más relevantes de una red de tipo corrupta (Martins et al., 2022).

Distribución de grado exponencial	Si
Grado promedio de nodo	17 - 20
Estructura en comunidades	Si
Tamaño promedio de las comunidades	7
Coeficiente de agrupamiento global	0.91 - 0.93
Coeficiente de agrupamiento local	0.93 - 0.94
Camino medio	2.99 - 5.11
Propiedad de mundo pequeño	Si
Coeficiente de asortatividad global	0.53 - 0.74
Coeficiente de asortatividad local	0.50 - 0.59
Densidad global	0.0
Densidad local	0.18

3.1.1.2 - Explicación detallada del algoritmo

1. **Asignar parámetros iniciales:**

   - $\boldsymbol{\lambda}$, tamaño característico de los escándalos.
   - $\boldsymbol{\alpha}$, proporción de nodos reincidentes.
   - $\boldsymbol{\beta}$, cantidad de nodos mínima para la creación de nodos reincidentes.
   - $\boldsymbol{p}$, probabilidad de reincidencia.
   - $\boldsymbol{N}$, número de iteraciones.

2. **Se crea una red vacía.**

3. **Se añade una red completa:**

   El tamaño característico (λ) se obtuvo de forma empírica mediante investigación periodística, el modelo selecciona aleatoriamente el tamaño de cada nuevo escándalo mediante la distribución exponencial:

   $$P(\lambda) \sim \ e^{-\frac{x}{\lambda}}$$

4. **Se eligen nodos reincidentes:**

   Algunos nodos añadidos a la red se conectan con las redes completas de anteriores iteraciones, llamados nodos "reincidentes" porque corresponden a las personas que estuvieron involucradas en un escándalo y vuelven a involucrarse en otro.
   Son creados mediante la ecuación:

   $$r \ = \ \alpha n \ - \ \beta$$

   Cuando se crean nodos reincidentes, se seleccionan nodos ya presentes en la red con una probabilidad p, o nodos que se convertirán en reincidentes por primera vez con una probabilidad $1 - p$.

5. **Iterar nuevamente:**

   Se repite el paso 3 y 4 la cantidad de iteraciones elegidas.

6. **Termina el algoritmo.**

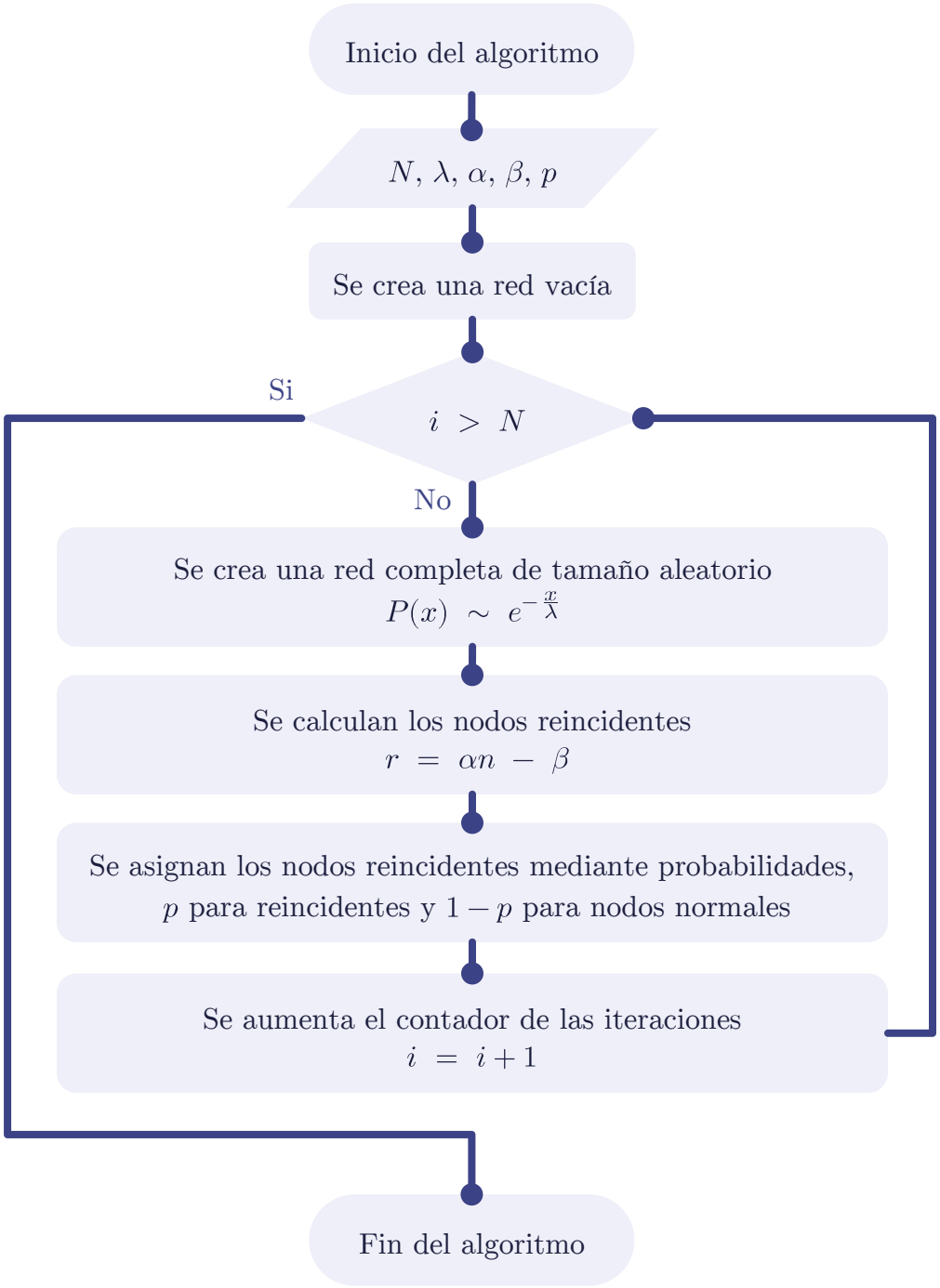


Figura 3.2 - Diagrama de flujo que representa las partes más importantes del algoritmo propuesto por Martins et al.

3.1.2 - Recreación del juego de la corrupción

Los parámetros de la matriz del juego (*Tabla 2.2)* fueron ajustados para representar el comportamiento social en tres países: Dinamarca, Singapur y México.

3.1.2.1 - De lo general a lo particular

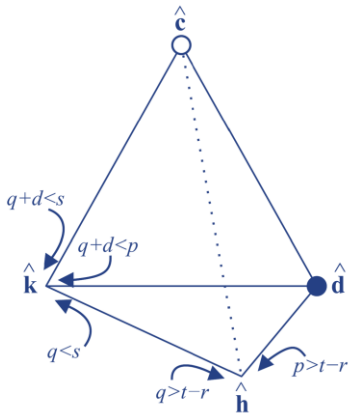


Figura 3.3 - Dinámica del juego de la corrupción, representada en un complejo simplicial, se indican los equilibrios del sistema para diferentes valores de los pagos. (Ubeda & Dueñez-Guzman, 2010).

La dinámica general del juego representada en el complejo simplicial (*Figura 3.3*) muestra los posibles equilibrios del sistema y las trayectorias evolutivas dependiendo de los pagos. Diferencias de poder pequeñas, como cuando: $s < q + d < p$, crean sociedades con poca corrupción, donde conviven personas cooperadoras y personas corruptas. Cuando existen grandes diferencias de poder: $0 < q + d < p$, la corrupción es persistente, creando únicamente sociedades con personas corruptas.
Si el castigo para los civiles es mayor que el castigo a una persona con poder: $p > q$, el sistema es propenso a un equilibrio entre personas cooperadoras y corruptas (Ubeda & Dueñez-Guzman, 2010).

Por medio del *A.1 - Replicador dinámico del juego de la corrupción* se implementaron las ecuaciones diferenciales que modelaban el juego en su forma continua. Estas ecuaciones, capturaron la competencia evolutiva entre las cuatro estrategias consideradas. Dinamarca y Singapur fueron modeladas mediante un equilibrio estable entre personas con poder honestas y personas cooperadoras. México por otra parte, se modeló mediante el equilibro (x) descrito en el artículo, una población de personas cooperativas y personas corruptas.

3.1.3 - Recreación del modelo mixto de Scatà et al.

Se recreó el algoritmo de (Scatà et al., 2016) con el fin de comprender la unión entre la teoría de juegos y la teoría de redes en un solo modelo.

3.1.3.1 - Explicación del algoritmo

1. **Asignar parámetros iniciales:**
   - Tamaño de la red $\boldsymbol{R}$: Cantidad de nodos totales dentro de la red.
   - Masa crítica $\boldsymbol{(M)}$: Cuántos nodos serán cooperativos al inicio.
   - Posición inicial $\boldsymbol{(c)}$: La posición inicial de la masa crítica dentro de la red (centrales, periféricos o aleatorios).
   - Tipo de juego $\boldsymbol{(J)}$: Se escoge entre el dilema del prisionero o el juego del dilema de la nieve acumulada.
   - Nivel de homofilia $\boldsymbol{(H)}$: Controla la adopción de estrategias vecinas.
   - Nivel de ruido al adoptar estrategias $\boldsymbol{(K)}$: Controla la aleatoriedad de los nodos al actualizar sus estrategias.
   - Iteraciones $\boldsymbol{(N)}$: El número de veces que se repetirá el algoritmo.
2. **Crear la red:** Se crea una red libre de escala de tamaño $\boldsymbol{R}$ con $\boldsymbol{M}$ nodos de masa crítica, se asignan las estrategias aleatoriamente a cada nodo.
3. **Jugar:** Cada nodo juega una ronda de $\boldsymbol{J}$ con sus nodos vecinos.
4. **Ganancias:** Cada nodo calcula la ganancia total $(\boldsymbol{g_i})$ de sus interacciones.
5. **Actualizar estrategias:** Cada nodo decide cambiar o mantener su estrategia. La probabilidad $(\boldsymbol{P})$ de cambiar a la estrategia de un vecino depende de:
   5.1. **Diferencia de ganancia:** Si un vecino tiene una ganancia mayor $(\boldsymbol{g_j})$, la probabilidad que el nodo cambie a la estrategia de ese vecino aumenta.
   $$P_{\text{base}} = \frac{1}{1 + e^{-\frac{g_j - g_i}{K}}}$$
   5.2. **Homofilia:** Si un vecino tiene la misma estrategia la probabilidad de copiarlo se refuerza $P_{\text{final}} = P_{\text{base}} \times (1 + H)$. Si el vecino tiene una estrategia diferente la homofilia actúa como una resistencia disminuyendo la probabilidad de copiarlo $P_{\text{final}} = P_{\text{base}} \times (1 - H)$.
6. **Repetir:** Se repiten los pasos 3, 4 y 5 durante $\boldsymbol{N}$ rondas para conocer cómo evoluciona la cooperación en la red.
7. Termina el algoritmo.

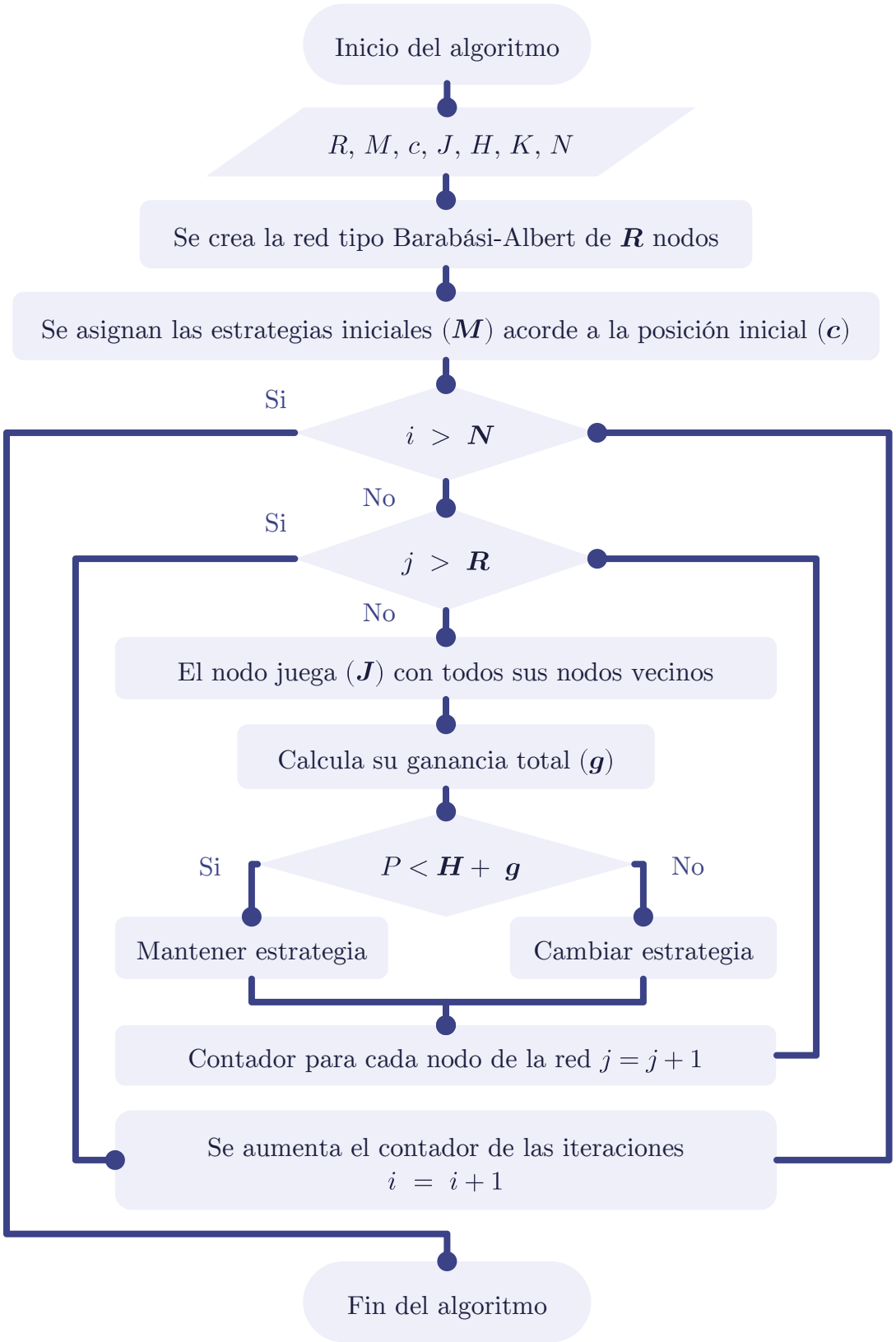


Figura 3.4 - Diagrama de flujo que representa las partes más importantes del algoritmo propuesto por Scatà et al.

3.2 - Desarrollo de un nuevo modelo

Se evaluaron enfoques alternativos para modelar la difusión de la corrupción, incluyendo modelos basados en redes multicapa y procesos de difusión en redes como los presentados en *2.5 - Procesos de propagación de información en redes.* Dichas aproximaciones fueron descartadas por no replicar comportamientos empíricos clave (*A.3 - Modelos descartados)*.

Con la experiencia previa de los modelos descartados, y comprendiendo sus limitaciones, se trabajó con el algoritmo de *2.8.1.1 - Modelo de Scatà et al (2016)*, reemplazando el juego originalmente propuesto por los autores (*2.6.1 - El dilema del prisionero*) por *2.7.2 - El juego de la corrupción*.

3.2.1 - Modelo refinado de Scatà et al.

3.2.1.1 - Recreación del juego evolutivo discreto

Con el objetivo de comprobar que el nuevo modelo reprodujera los resultados obtenidos mediante la dinámica del replicador, se implementó la versión discreta del juego con una red completa de 1000 nodos durante 100 iteraciones.

3.2.1.2 - Experimentación con diferentes escenarios

Se diseñaron escenarios experimentales para analizar el comportamiento del modelo en situaciones donde una población cooperadora evolucionara a una población corrupta y donde una población corrupta evolucionara a una cooperadora.

Para la red se trabajó con el *2.3.3 - Modelo de Barabási-Albert, redes libres de escala (1999)* comúnmente asociadas a estructuras sociales (Scatà et al., 2016). En todos los escenarios, se utilizó la misma red base con mil nodos y un parámetro de conexión inicial ($m = 1$).

Para estos escenarios propuestos se añadieron casos adicionales de la masa crítica:

- **Masa crítica cooperativa:** compuesta por nodos tipo C y H.
- **Masa crítica no cooperativa:** compuesta por nodos tipo D y K.
- **Masa crítica corrupta:** exclusivamente nodos tipo K.

3.2.1.2.1 - Evaluación de la robustez social

Se analizó la resistencia de las sociedades a la propagación de la corrupción. Inicialmente, cada red se configuró con una población cooperadora y con una masa crítica corrupta pequeña, incrementando progresivamente en cada simulación el porcentaje de la masa crítica, desde el 5% hasta el 95% de la población total, en incrementos del 5%.

El modelo se probó durante 100 iteraciones y en algunos casos 500 iteraciones, según lo requerido para llegar a un estado de equilibrio. Esto permitió identificar si existía un umbral mínimo a partir del cual la red colapsara hacia un estado predominantemente corrupto.

Cada configuración espacial inicial de masa crítica (nodos centrales, nodos periféricos y nodos aleatorios) fue sometida a análisis para determinar su influencia en la evolución de la corrupción en la red. Asimismo, durante todo el proceso se modificaron los niveles de homofilia de la red ($H = 0.0, 0.5, 0.8, 0.9, 1.0$) en cada sociedad para analizar su impacto en la evolución final.

Finalmente, en cada red creada se le filtró su subred conformada exclusivamente por nodos corruptos, a fin de analizar y comparar sus métricas con las redes documentadas y creadas en *2.3.4 - Modelo de Martins et al., redes de tipo corruptas (2022)*.

3.2.1.2.2 - Intervenciones en sociedades corruptas

En un escenario inverso, se trabajaron con redes inicialmente corruptas (con mayoría de nodos con estrategia K) con el objetivo de convertirlas a redes con población cooperadora. Estas intervenciones consistieron en la modificación de parámetros clave del modelo, como la introducción de una masa crítica cooperadora o un aumento en la homofilia, lo que simuló reformas estructurales y políticas públicas aplicadas a una sociedad.

3.2.1.2.2.1 - Intervención mediante masa crítica cooperadora

Se introdujeron masas críticas en dos variantes:

- **Masa crítica de poder honrado:** compuesta por nodos tipo H.
- **Masa crítica mixta cooperativa:** compuesta por nodos tipo C y H.

Cada configuración fue evaluada en las tres distribuciones espaciales (nodos centrales, nodos periféricos y nodos aleatorios) usando la misma metodología que en el experimento de la robustez social, es decir, una simulación de 100 a 500 rondas, incrementando progresivamente en cada simulación la proporción de masa crítica inicial del 5% hasta el 95%, en incrementos del 5%, con el fin de determinar si existía un umbral mínimo que propagara las estrategias cooperadoras y si era equivalente al de la masa crítica corrupta.

3.2.1.2.2.2 - Alteración de la homofilia

Se exploró si una mayor homofilia facilitaba la dispersión de estrategias cooperativas y en consecuencia la transformación de la red, repitiendo las intervenciones con masa crítica para distintos valores de homofilia ($H = 0.0, 0.5, 0.8, 0.9, 1.0$).

C
D
C
(3,3)
(0,5)
D
(5,0)
(1,1)
A
B

4 - Resultados

4.1 - Recreación de modelos previos

4.1.1 - Recreación de la red corrupta

Se recreó el algoritmo en Python (*A.4.1 - Recreación de la red corrupta*), usando los siguientes parámetros:

$$\lambda = 7.33, \qquad \alpha = 0.1, \qquad \beta = -12, \qquad p = 0.025, \qquad N = 100.$$

Se verificó que la tasa de reincidencia α fue el factor más importante para la estructura de la red (*Figura 4.1*). Cuando tomaba valores pequeños $\alpha < 0.09$ la red se desintegraba por completo, creando comunidades asiladas, por el contrario, cuando $\alpha > 0.19$, la red se conectaba excesivamente, perdiendo las características asociadas a las redes corruptas, replicando así los resultados del estudio.

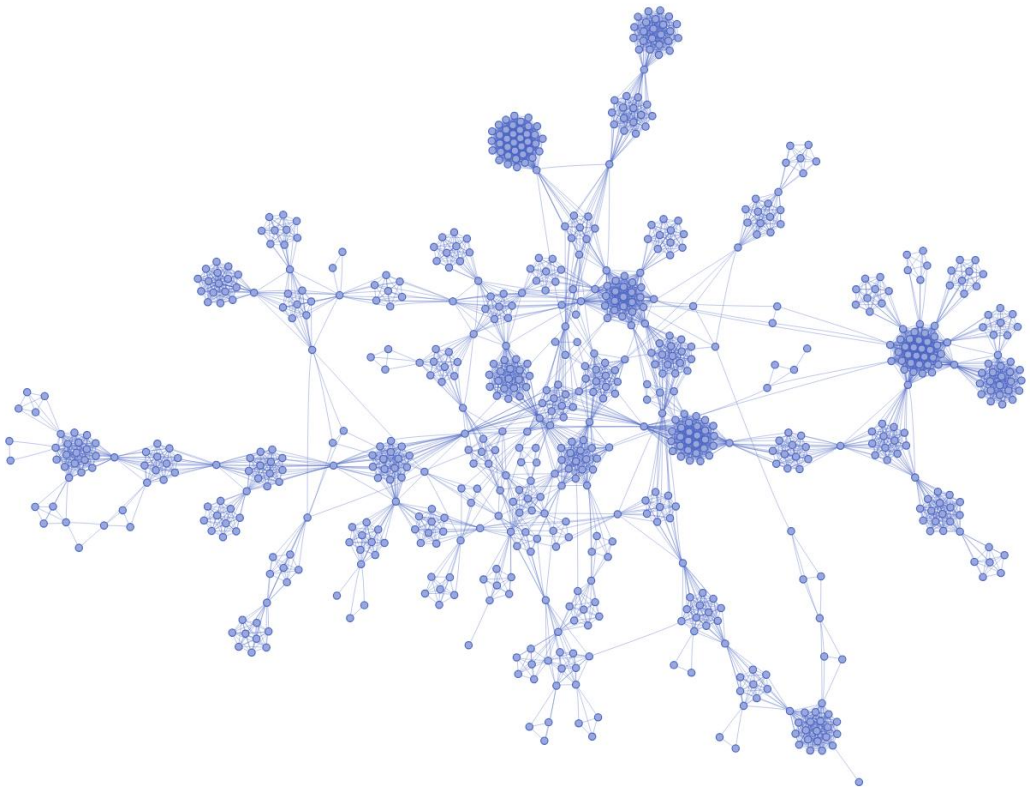


Figura 4.1 – Red creada mediante el algoritmo propuesto por (Martins et al., 2022).

4.1.1.1 - Métricas obtenidas

Las métricas de la red creada (*Tabla 4.1* y *Figura 4.2*) fueron muy cercanas a las métricas descritas por los investigadores para las redes de España y Brasil (*3.1.1.1 - Propiedades que recrea el modelo de Martins et al.*).

Tabla 4.1 – Métricas resultantes de la red de tipo corrupta modelada.

Distribución de grado exponencial	Si
Exponente de la ley de potencias (γ)	-0.77
Número de comunidades	42
Comunidad más grande	66 nodos
Tamaño promedio de las comunidades	18.62
Coeficiente de agrupamiento global	0.93
Coeficiente de agrupamiento local	0.93
Camino medio	2.52
Propiedad de mundo pequeño	Si
Coeficiente de asortatividad global	0.87
Coeficiente de asortatividad local	0.31
Densidad global	0.02
Densidad local	0.18

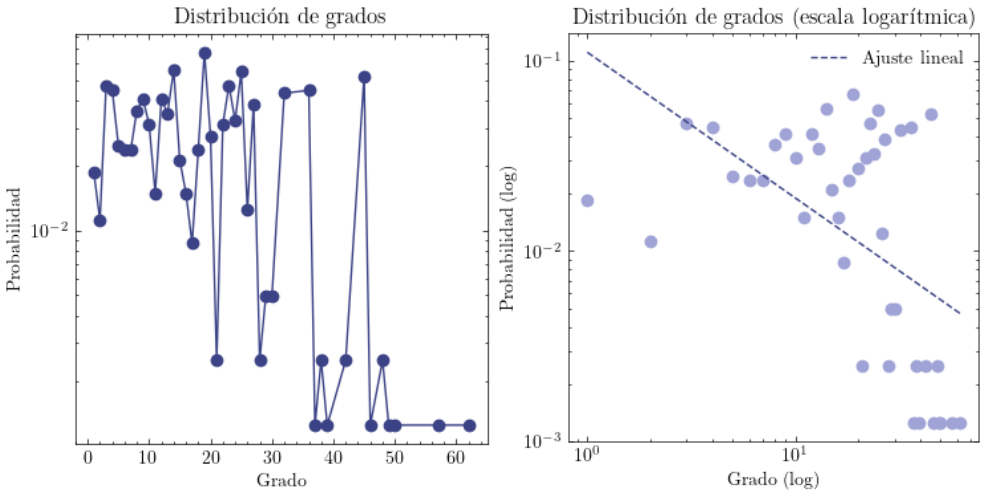
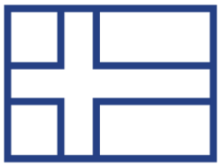


Figura 4.2 – La red presentó una distribución de grados de tipo exponencial.

4.1.2 - Computo de las matrices del juego de la corrupción

Se propusieron pagos numéricos que recobraran el comportamiento social expuesto en *1.2 - Historias de éxito* y en (Legatum Institute Foundation, 2023), para las sociedades de Dinamarca, Singapur y México.

4.1.2.1 - Dinamarca



Dinamarca (*Figura 4.3*) corresponde a una sociedad donde el sentido de comunidad es más importante que el individual, ello se reflejó en el pago de la cooperación $r = 2.5$, el cual es alto respecto a la no cooperación $s = 1,\ t = 3$.
Al tener un estado de derecho en buenas condiciones $c = 0.2,\ d = 0.5$, no es necesario tener castigos altos $p = 8,\ q = 11$, para generar un equilibrio cooperativo.

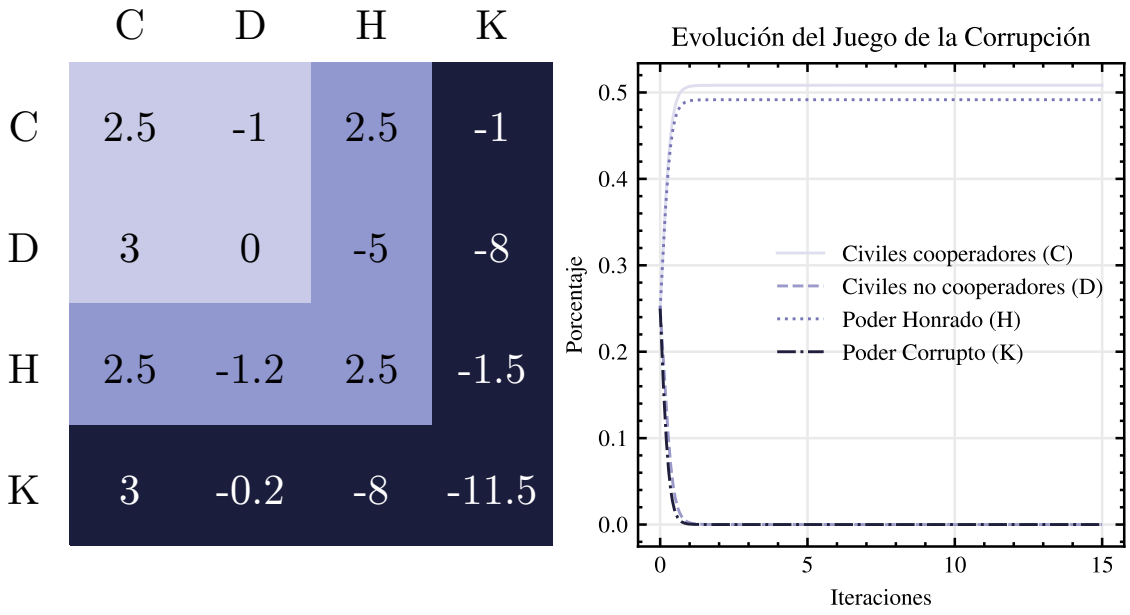


Figura 4.3 – Matriz de pagos de Dinamarca y gráfica de la evolución de la población. En pocas iteraciones la sociedad danesa llega a un equilibrio estable donde la mitad de la población corresponde a civiles cooperadores (C) y la otra mitad al poder honrado (H).

4.1.2.2 - Singapur

Singapur (*Figura 4.4*) se modeló como una sociedad susceptible a la corrupción, tiene los parámetros tradicionales del dilema del prisionero $s = 1$, $t = 4$, $r = 2$. Pero, gracias a sus leyes estrictas, reflejadas como castigos altos $p = 10$, $q = 15$ y un excelente estado de derecho $c = 0.2$, $d = 0.2$, las personas deciden cooperar.

	C	D	H	K
C	2	-1	2	-1
D	4	0	-6	-10
H	2	-1.2	2	-1.2
K	4	-0.2	-11	-15.2

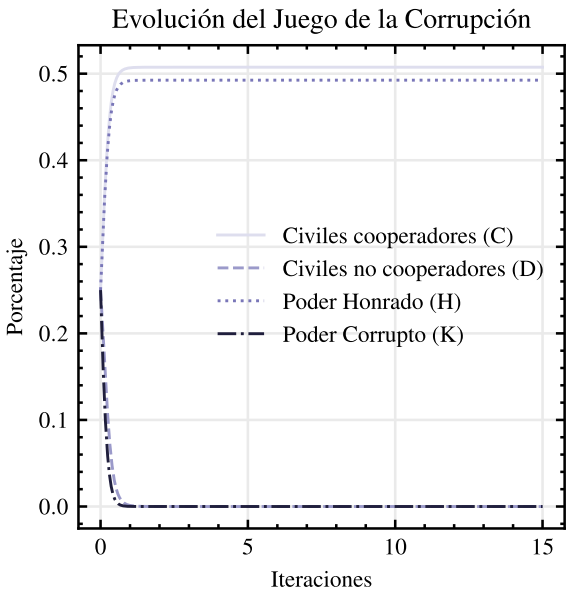


Figura 4.4 – Matriz de pagos de Singapur y gráfica de la evolución de la población. Las leyes estrictas y su fácil aplicación generan el equilibrio estable cooperativo de civiles cooperadores (C) y poder honrado (H) en la sociedad de Singapur.

4.1.2.3 - México

México (*Figura 4.5*) se propuso como una sociedad susceptible a la corrupción, contiene los parámetros usuales del dilema del prisionero $r = 2$, $s = 1$ y $t = 4$.

El castigo a los civiles es mayor al castigo a las personas con poder $(p > q)$ con $p = 10$ y $q = 5$, junto con un estado de derecho deteriorado $c = 2$ y $d = 4$, propicia la corrupción dentro de la población.

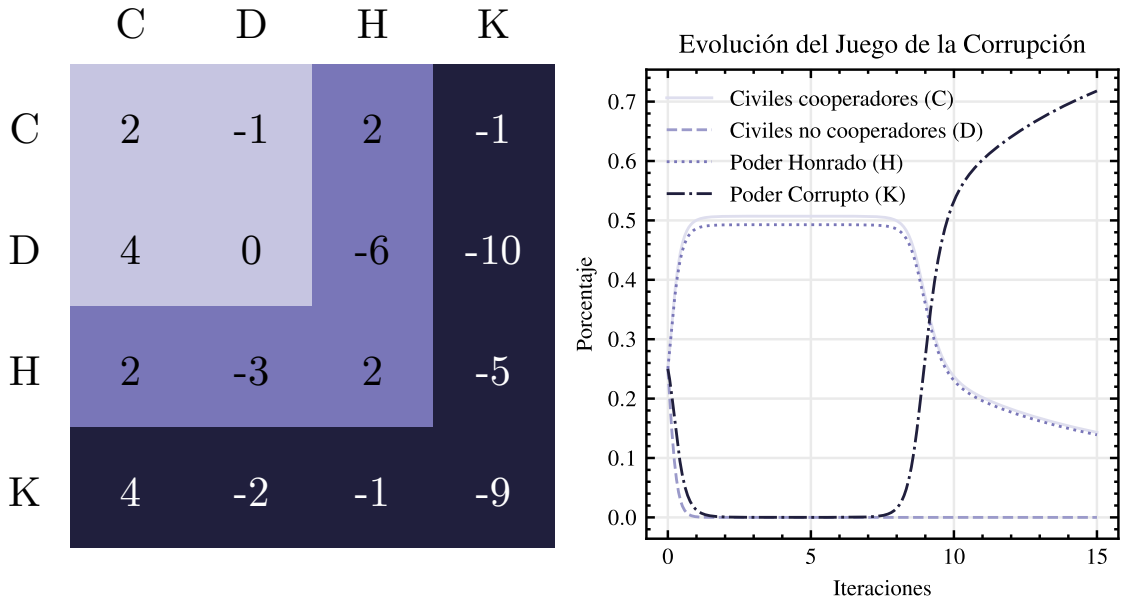


Figura 4.5 – Matriz de pagos de México y gráfica de la evolución del sistema. Se puede notar el equilibrio inestable de la población, a partir de la novena iteración la población cambia a una población mixta de civiles cooperadores (C), poder honrado (H) y corrupto (K), el sistema converge al equilibrio estable $(\boldsymbol{x})$ analizado en el artículo de (Ubeda & Dueñez-Guzman, 2010).

4.1.3 - Recreación del modelo de Scatà et al.

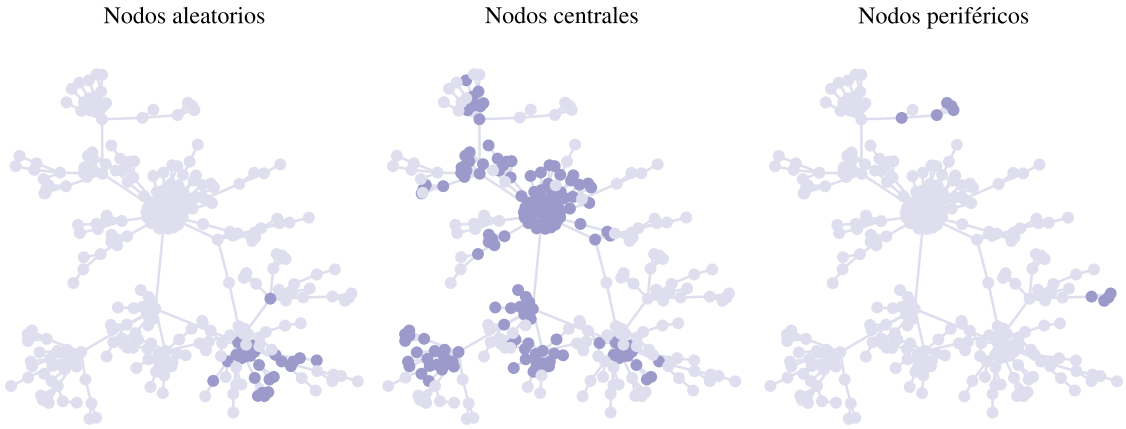


Figura 4.6 – Evolución final de las personas cooperadoras (en color oscuro) en distintas configuraciones dentro de la red para el juego del dilema del prisionero. La cooperación total de la red fue mayor cuando la masa crítica estuvo situada en los nodos centrales de la red.

Los resultados mostraron que para el juego del dilema del prisionero una alta homofilia ($H > 0.8$) y una masa crítica ubicada en los nodos centrales maximizaba la cooperación. Para el juego de la nieve acumulada la homofilia no fue un factor clave para maximizar la cooperación (*Figura 4.6* y *Figura 4.7*).

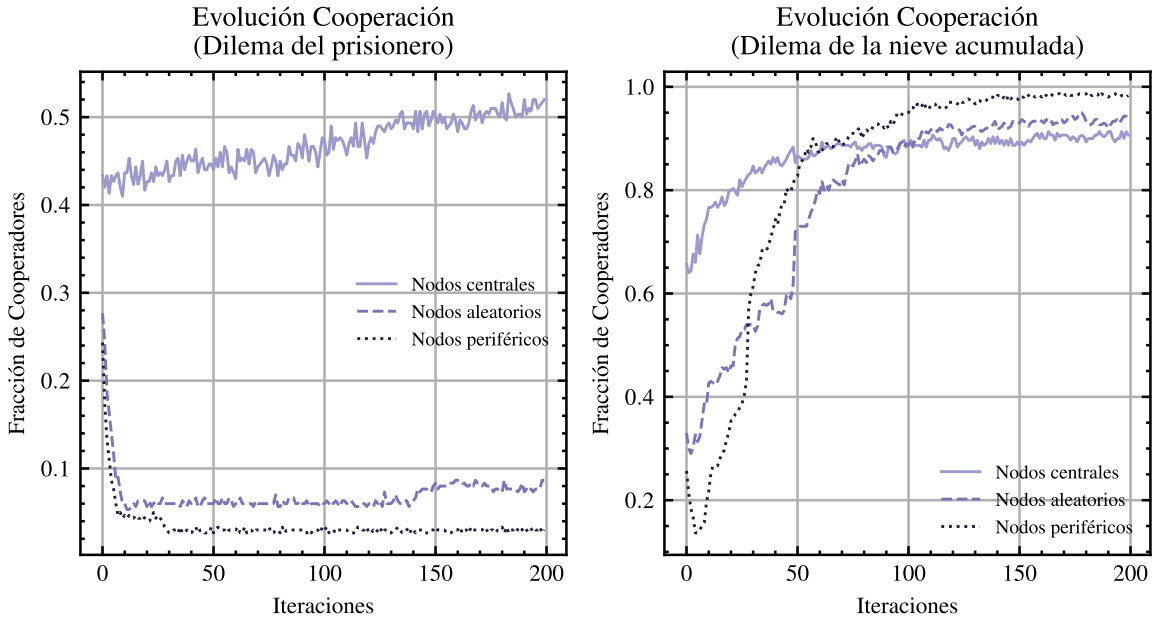


Figura 4.7 - Se configuraron 300 nodos con una masa crítica inicial del 50%. Para el juego del dilema del prisionero la propagación de la cooperación fue más lenta en comparación con juego de la nieve acumulada, replicando los resultados del estudio.

4.2 - Modelo refinado de Scatà et al.

Todas las gráficas de evolución, redes y subredes generadas se pueden encontrar en el repositorio en línea expuesto en el apéndice *A.4 - Código.*

4.2.1 - Recreación del caso continuo

El nuevo modelo se probó sin homofilia en una red completa (*Figura 4.8*), lo cual creó una versión discreta de la dinámica del replicador. Se lograron equilibrios similares a los obtenidos en *4.1.2 - Computo de las matrices*.

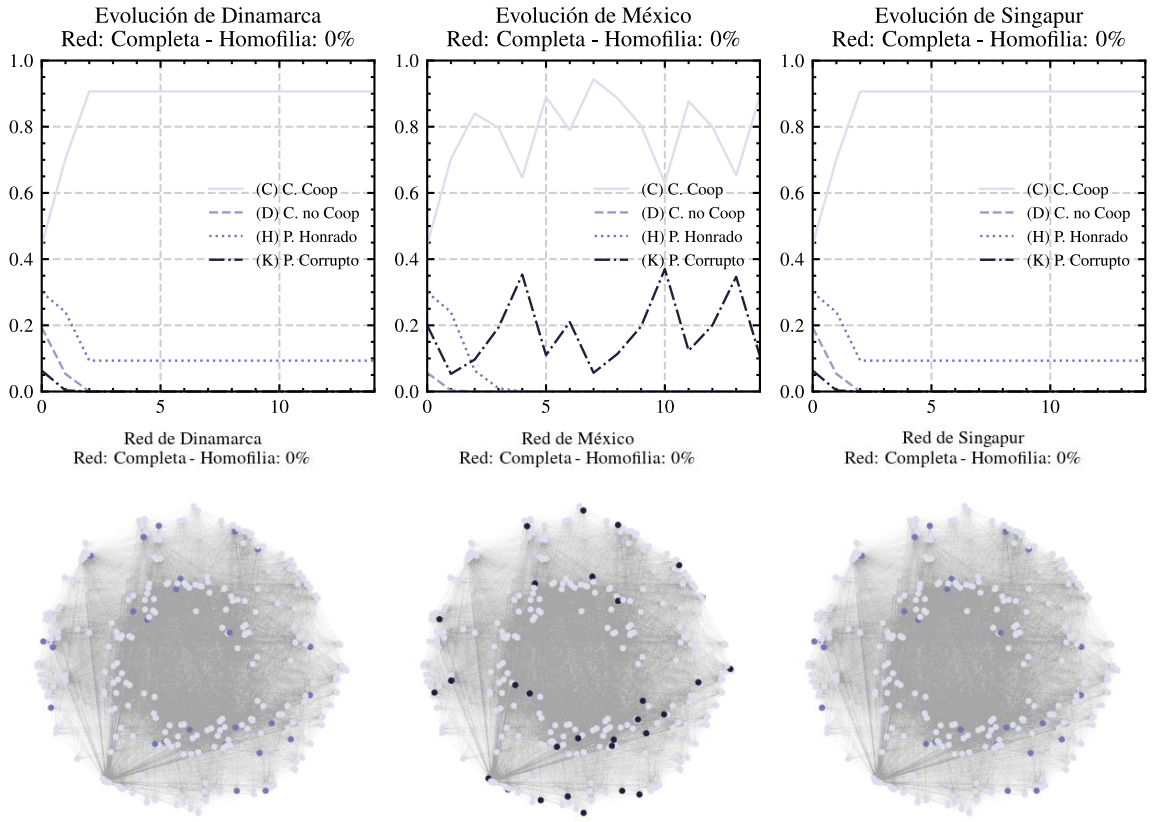


Figura 4.8 – El algoritmo refinado de Scatà et al. recreó los resultados obtenidos mediante la dinámica del replicador. El caso mexicano pertenece al equilibrio ($\boldsymbol{x}$) de cooperadores (C) corruptos (K), por otra parte, los casos de Dinamarca y Singapur corresponden a equilibrios estables de estrategias cooperadoras (C,H).

4.2.2 - Experimentación con diferentes escenarios

4.2.2.1 - Evaluación de la robustez social

4.2.2.1.1 - La configuración espacial importa

Una masa crítica en una disposición espacial de nodos centrales (*Figura 4.9*) produce la mayor propagación de la estrategia (K), independientemente de la sociedad.

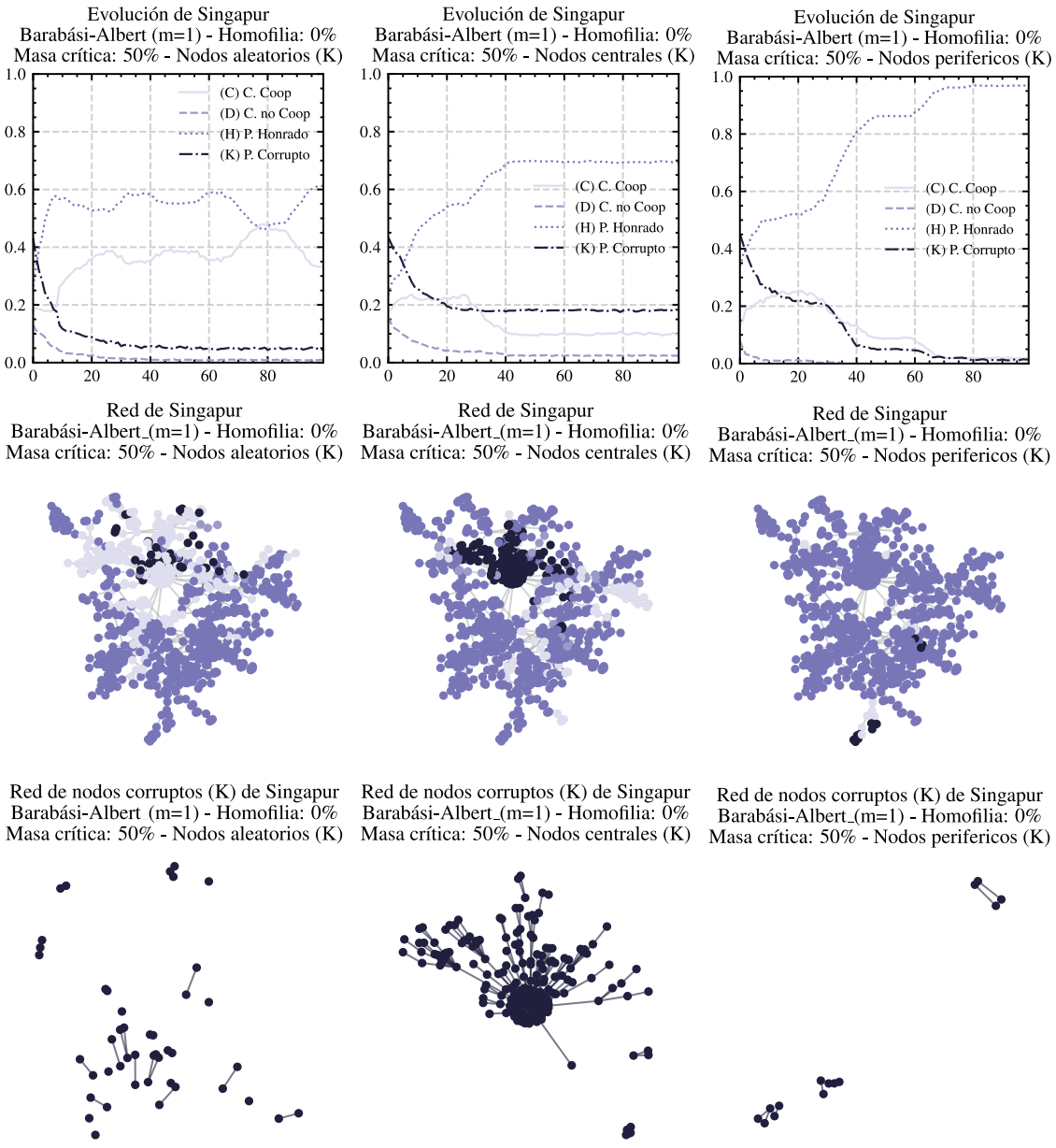


Figura 4.9 - Ejemplo con la sociedad de Singapur, para una masa crítica corrupta del 50% con disposición inicial central, la estrategia corrupta logra permanecer en el 20% de la red, en los otros dos casos la masa crítica es eliminada casi en su totalidad en las primeras 100 iteraciones. Dinamarca presentó resultados similares.

4.2.2.1.2 - El umbral mínimo

En el caso de Dinamarca y Singapur, no existe un umbral mínimo en el cual la red evolucione a una población total de nodos corruptos (K) (*Figura 4.10*), en la disposición inicial central existe un porcentaje mínimo (aproximadamente entre el 20% y 25%) en el cual la masa crítica persiste en la red. Para porcentajes mayores al 50%, la masa crítica se reduce, pero permanece en el 40% de la población total.

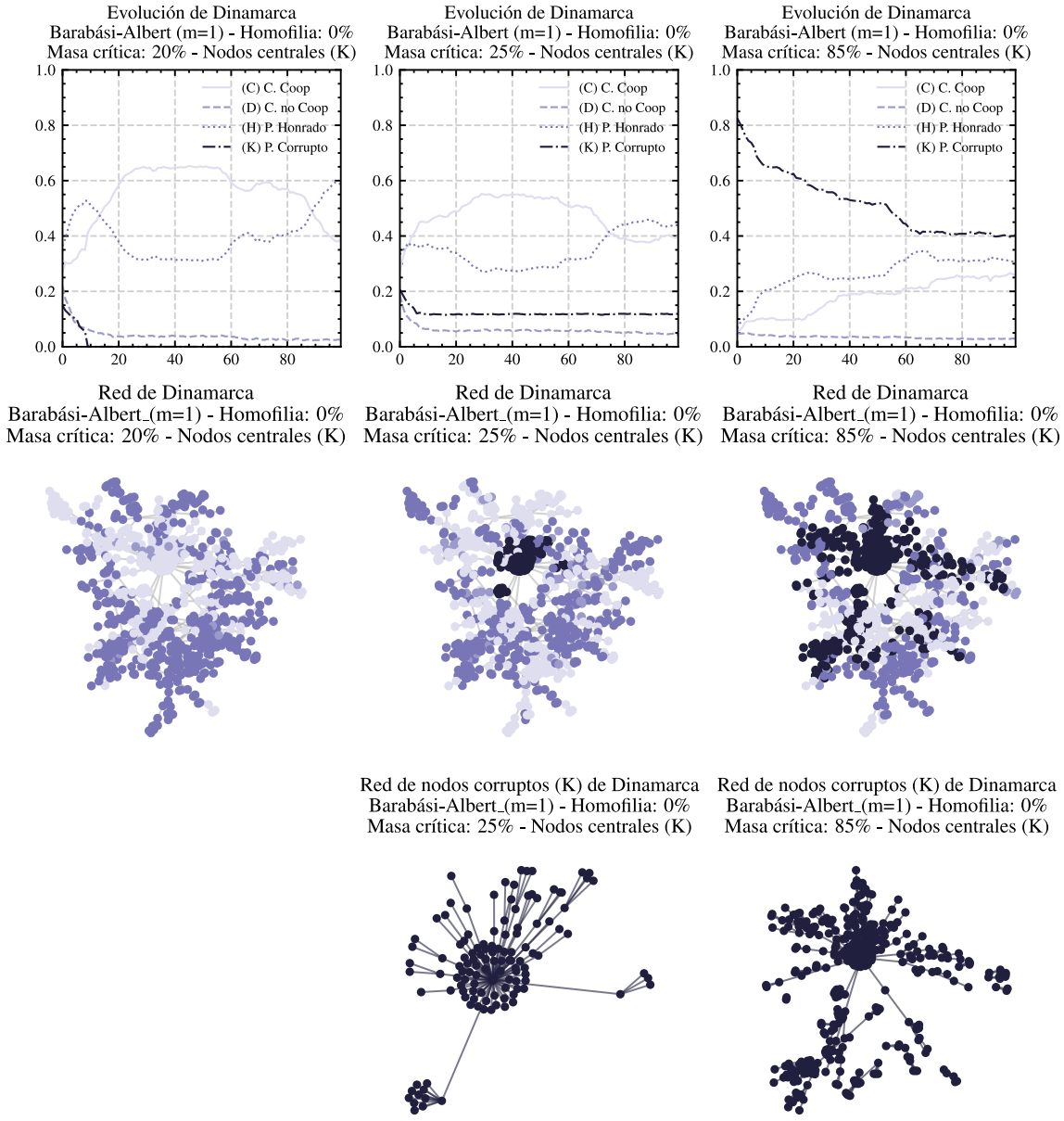


Figura 4.10 - Ejemplo con la sociedad de Dinamarca, para una masa crítica del 20%, 25% y 80%, los nodos con estrategia corrupta (K) persisten dentro de la comunidad central de la red. Singapur presentó la misma dinámica.

4.2.2.1.3 - Caso extremo

Cuando el 95% de la masa crítica es corrupta (K) (*Figura 4.11*), las sociedades más resilientes evolucionaron a una red con aproximadamente el 60% a 70% de la población con la estrategia del poder honrado (H). Caso contrario, cuando una sociedad no cuenta con castigos severos el porcentaje inicial de masa crítica se mantiene constante.

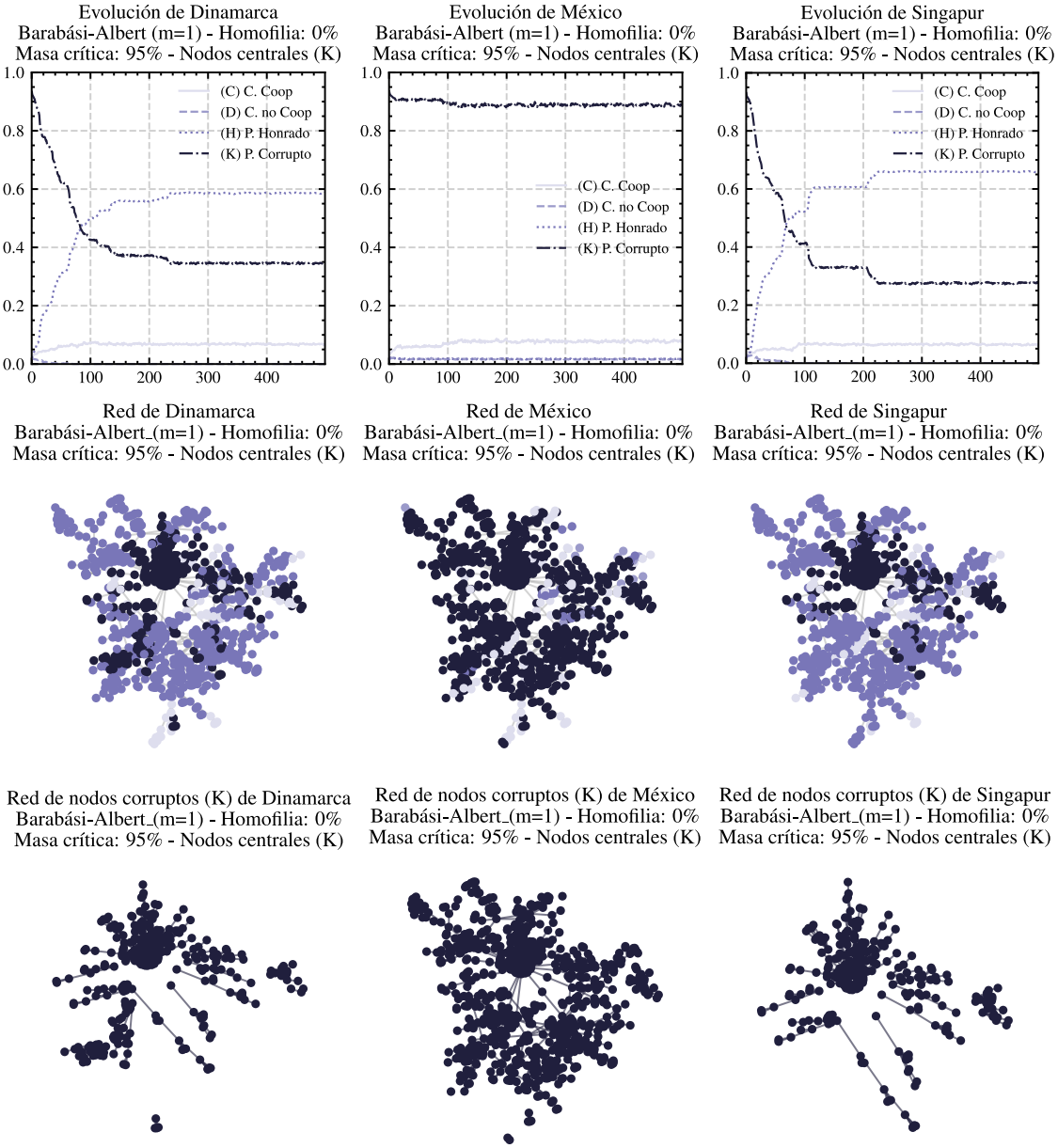


Figura 4.11 – Se comparan las tres sociedades, las redes de Dinamarca y Singapur logran erradicar la estrategia mayoritaria (K), Singapur tiene una mayor efectividad al modificar un 67% de la red por una estrategia cooperadora, por otro lado, la dinámica mexicana únicamente logra disminuir un 5% la estrategia en las zonas periféricas.

4.2.2.1.4 - La influencia de la homofilia

Una mayor homofilia ralentizaba la evolución de las estrategias (*Figura 4.12*), pero no influía en una mayor propagación ni en el equilibrio final de las poblaciones.

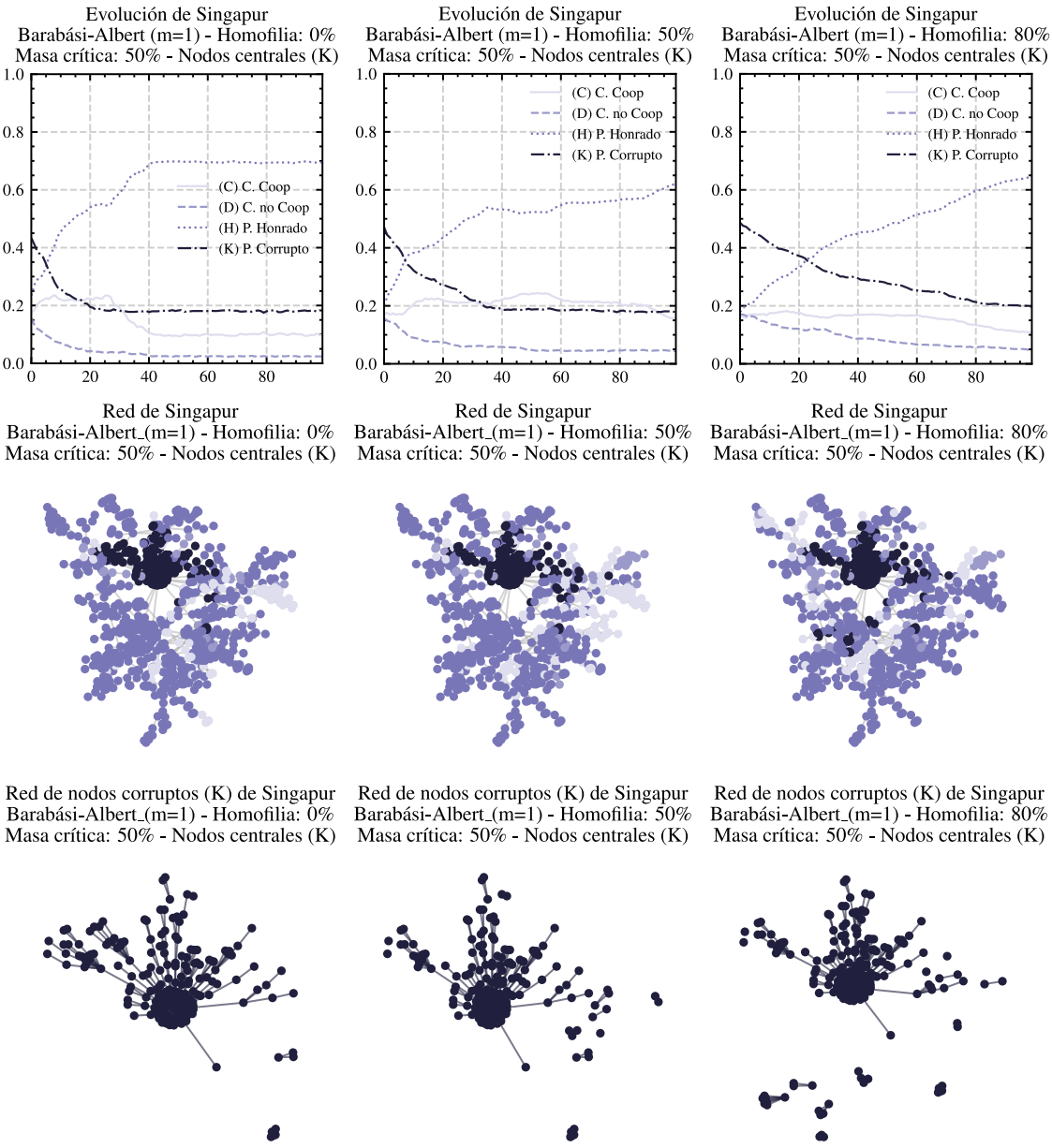
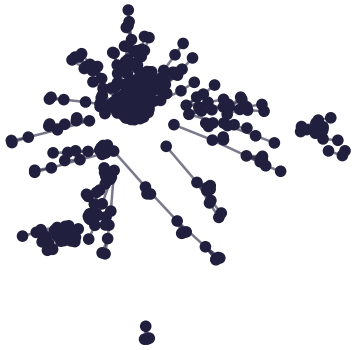


Figura 4.12 – Ejemplo con la red de Singapur, en todos los escenarios la población evoluciona a las mismas proporciones. Los mismos resultados se obtuvieron para México y Dinamarca.

4.2.2.1.5 - Métricas de las subredes corruptas de nodos (K)

Las subredes corruptas creadas en *4.2.2.1.3 - Caso extremo* fueron analizadas (*Figura 4.13* y *Figura 4.14*) acorde a las métricas de *3.1.1.1 - Propiedades que recrea el modelo de Martins et al.*

Red de nodos corruptos (K) de Dinamarca
Barabási-Albert_(m=1) - Homofilia: 0%
Masa crítica: 95% - Nodos centrales (K)

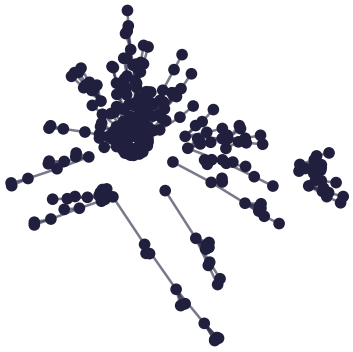


Distribución de grado exponencial	Si
Exponente de la ley de potencias (γ)	-1.84
Número de comunidades	106
Comunidad más grande	35 nodos
Tamaño promedio de las comunidades	3.3
Coeficiente de agrupamiento global	0.0
Coeficiente de agrupamiento local	0.0
Camino medio	2.5
Propiedad de mundo pequeño	No
Coeficiente de asortatividad global	-0.206
Coeficiente de asortatividad local	-0.594
Densidad global	0.004
Densidad local	0.057

Figura 4.13 – Red de Dinamarca con 350 nodos y 245 enlaces.

La red de Singapur tuvo métricas similares (*Figura 4.14*).

Red de nodos corruptos (K) de Singapur
Barabási-Albert_(m=1) - Homofilia: 0%
Masa crítica: 95% - Nodos centrales (K)



Distribución de grado exponencial	Si
Exponente de la ley de potencias (γ)	-2.14
Número de comunidades	101
Comunidad más grande	20 nodos
Tamaño promedio de las comunidades	2.8
Coeficiente de agrupamiento global	0.0
Coeficiente de agrupamiento local	0.0
Camino medio	2.7
Propiedad de mundo pequeño	No
Coeficiente de asortatividad global	-0.229
Coeficiente de asortatividad local	-0.576
Densidad global	0.004
Densidad local	0.101

Figura 4.14 - Red de Singapur, con 279 nodos y 179 enlaces.

4.2.2.2 - Intervenciones a una sociedad corrupta

El caso mexicano, al no contar con una dinámica que erradicara la población corrupta (*Figura 4.11*) fue necesario implementar estrategias alternativas que permitieran a la población transitar hacia un equilibrio basado en la cooperación.

4.2.2.2.1 - Intervención mediante masa crítica

4.2.2.2.1.1 - La configuración espacial importa

Una masa crítica con una configuración espacial de nodos aleatorios (*Figura 4.15*) resultó ser la más efectiva para la propagación de las estrategias cooperativas.

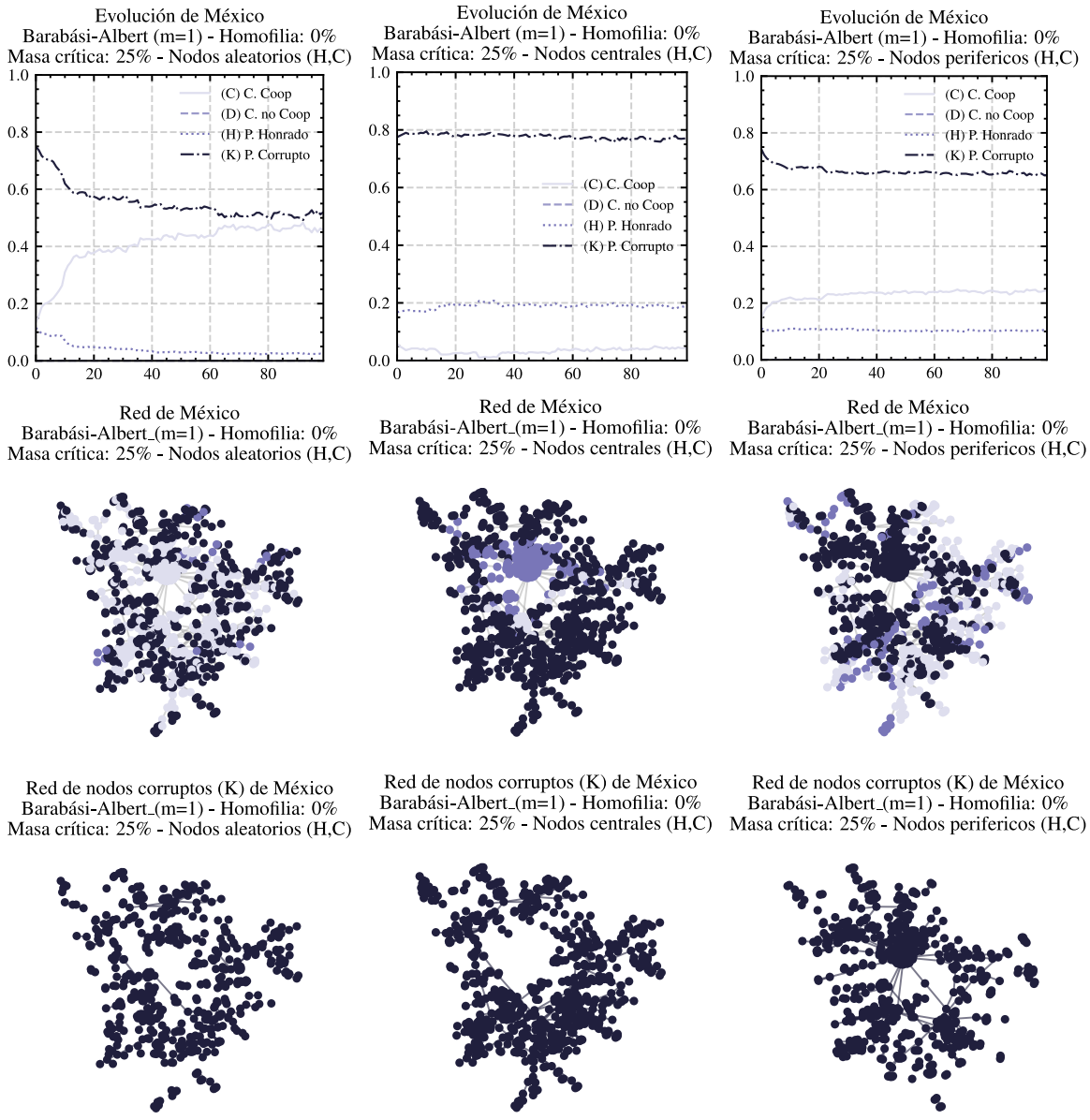


Figura 4.15 – Comparativa de las 3 configuraciones espaciales de masa crítica para el caso mexicano. La configuración aleatoria logró reducir del 75% al 52% la estrategia mayoritaria.

4.2.2.2.1.2 - Diferentes masas críticas

La propagación de las estrategias cooperativas fue más efectiva con la masa crítica mixta (C,H) que con la masa crítica constituida únicamente de la estrategia del poder honrado (H) (*Figura 4.16*).

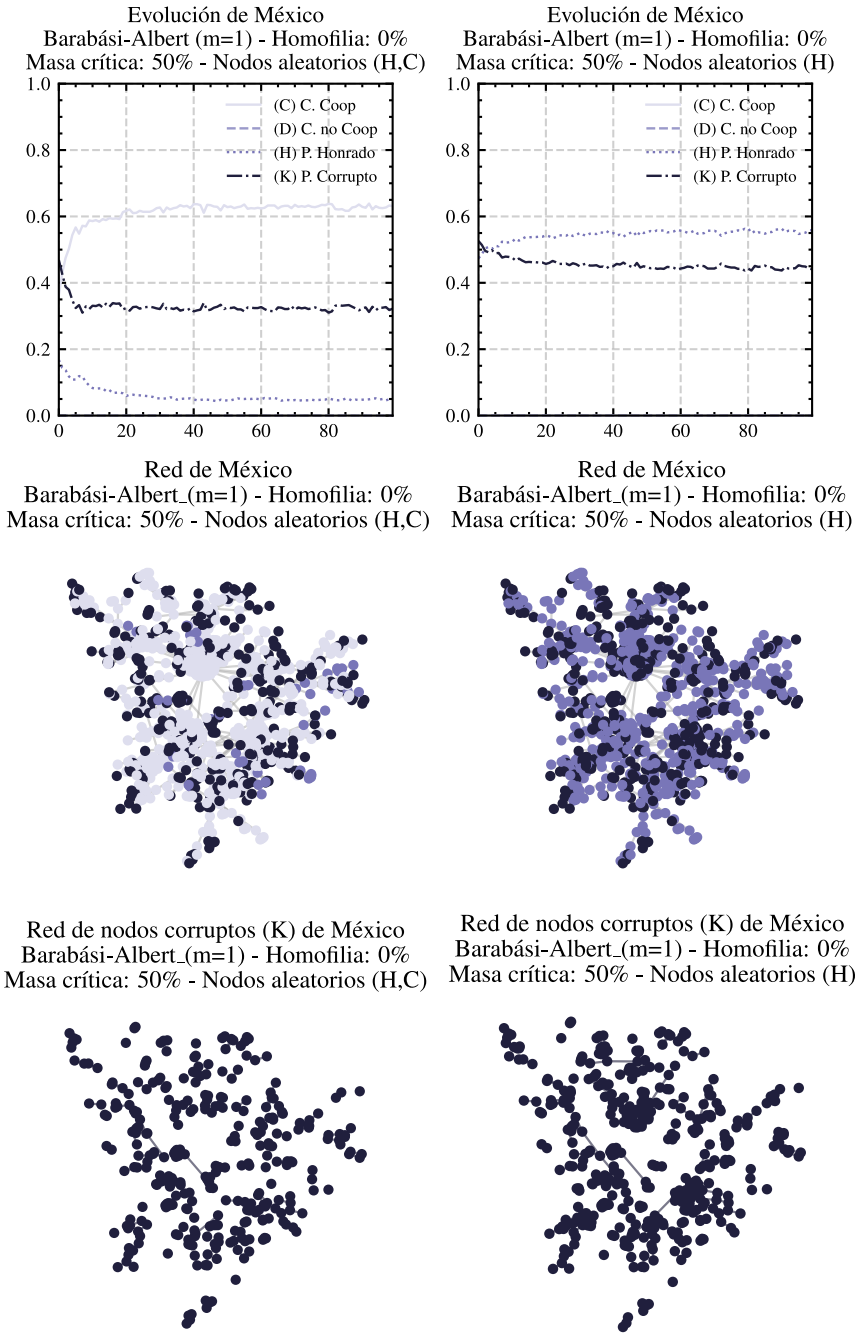


Figura 4.16 – Ambos escenarios con el mismo porcentaje inicial de masa crítica (50%), la masa crítica combinada (C,H) propagó las estrategias de cooperación a un mayor porcentaje de la red (64%) respecto a la masa crítica (H) (55%).

4.2.2.2.1.3 - El umbral mínimo

No existe un umbral mínimo donde toda la red evolucione a una población cooperativa, pero existen porcentajes mínimos donde se genera una mayor propagación. Por ejemplo, con un 15% de masa crítica cooperadora la estrategia se logra propagar al 40% de la red (*Figura 4.17*). A partir del 30% de masa crítica la red evoluciona a una mayoría cooperativa del 60%, pero para porcentajes de masas críticas mayores al 60% no existe una mayor propagación de estrategias cooperadoras.

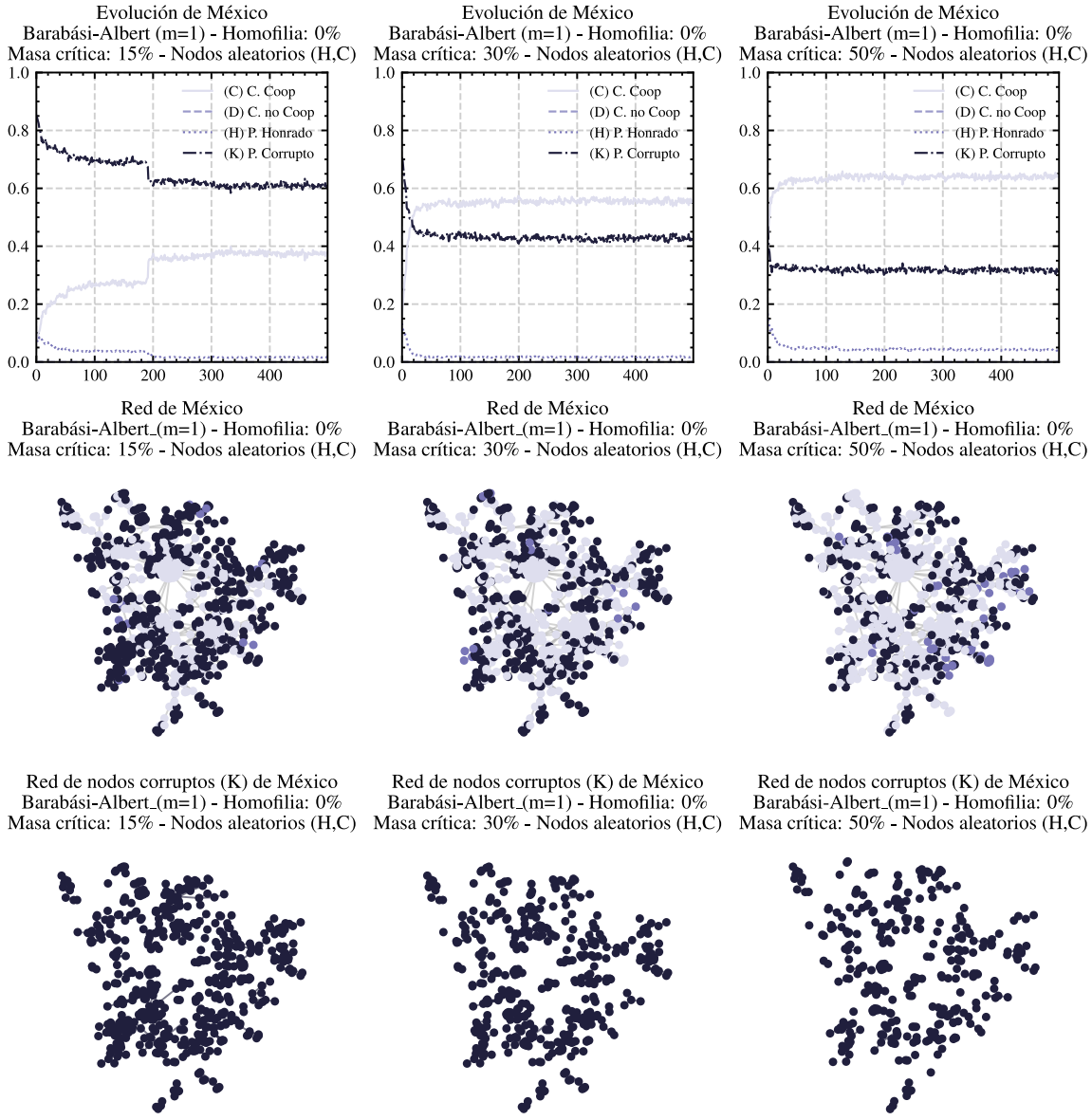


Figura 4.17 – Tres diferentes casos de masa crítica 15%, 30%, 50%), se puede notar que a partir de cierto porcentaje (60%) la difusión de las estrategias cooperativas se estanca dentro de la red.

4.2.2.2.2 - Homofilia

Se tuvo un comportamiento similar a la prueba de la robustez, al aumentar la homofilia (*Figura 4.18*) se ralentizaba el equilibrio final pero no influía en una mayor o menor propagación de las estrategias, únicamente cuando la homofilia era total ($H = 100\%$) la dinámica se bloqueaba en los valores iniciales.

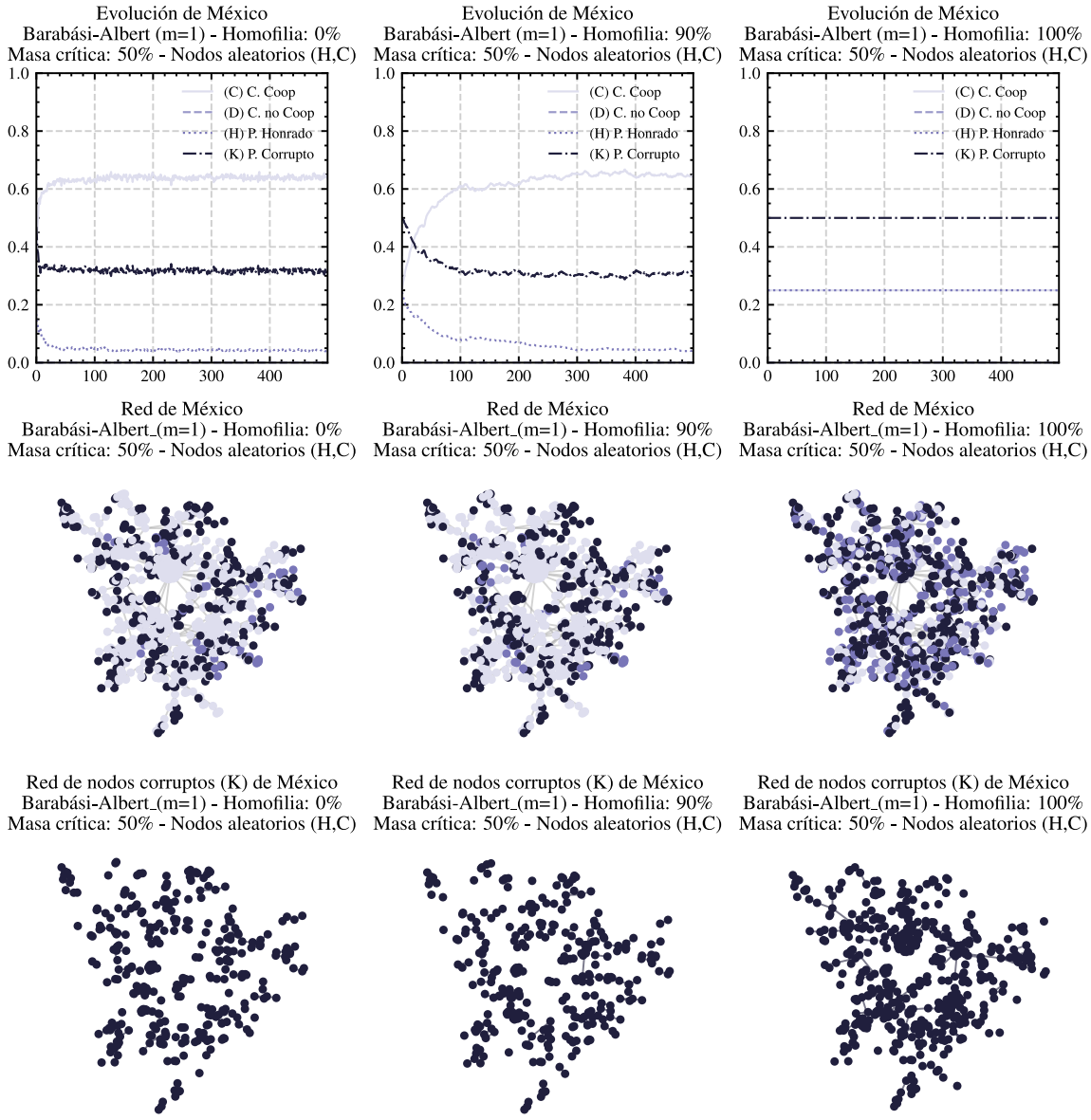


Figura 4.18 - En el caso de una masa crítica del 50% para una homofilia del 0% el equilibrio se obtiene en las primeras 30 iteraciones, al aumentar la homofilia al 90% el equilibrio se alcanza en 100 iteraciones, esta tendencia se presenta en los demás porcentajes iniciales de la masa crítica.

4.3 - Análisis

4.3.1 - Dinámica del modelo refinado

El modelo propuesto proporcionó una dinámica de propagación más realista en comparación con modelos de difusión convencionales (*A.3 - Modelos descartados*). A diferencia de los modelos del umbral, donde una mayoría inicial suele determinar el estado final de la red, este modelo demostró que la matriz de pagos tiene una influencia más decisiva que el porcentaje inicial de nodos corruptos. Un ejemplo contundente se observó en la simulación de sociedades resilientes (*Figura 4.9*), donde incluso con una masa crítica corrupta del 95%, la población evoluciona hacia un equilibrio mayoritariamente cooperativo, demostrando que una estructura de incentivos robusta con leyes y castigos severos puede revertir un estado casi totalmente corrupto.

4.3.2 - Robustez social y estructura de las redes corruptas

Al analizar la resiliencia de las sociedades de Dinamarca y Singapur, se encontró que la configuración espacial inicial de la masa crítica corrupta es una variable fundamental. La corrupción solo lograba mantenerse y propagarse cuando los nodos corruptos iniciales controlaban la comunidad central de la red. Este fenómeno es evidente en la *Figura 4.9* donde se observa que una masa crítica de nodos corruptos (K) ubicados en la periferia o de forma aleatoria es neutralizada por la población cooperativa, mientras que una masa crítica en los nodos centrales logra persistir e incluso propagarse.

Las subredes corruptas generadas (*Figura 4.13* y *Figura 4.14*) replicaron características clave descritas en *3.1.1.1 - Propiedades que recrea el modelo de Martins et al.*, como una distribución de grado exponencial, longitud de camino medio corta y una densidad baja. Sin embargo, varias de estas propiedades son heredadas de la red principal Barabási-Albert (con $m = 1$), la cual no genera de forma natural comunidades densas, lo que es una característica de escándalos de corrupción más complejos (Martins et al., 2022).

4.3.3 - Análisis de estrategias para erradicar la corrupción

El caso mexicano, caracterizado por un equilibrio (x) entre nodos cooperadores (C) y corruptos (K), sirvió como un laboratorio ideal para probar estrategias de intervención.

El resultado más notable fue la mayor efectividad de una intervención con masa crítica cooperadora dispuesta de forma aleatoria en comparación con masas críticas centralizadas o periféricas (*Figura 4.15*), además se observó que la homofilia fue un factor determinante para ralentizar el equilibrio de la propagación (*Figura 4.18*) pero no determinante para cambiar las proporciones finales de las estrategias.

Una causa probable del comportamiento de la homofilia es debido a los incentivos económicos del juego de la corrupción, estos superan la predisposición de la dinámica a asociarse con similares, la cual es más dominante en juegos simples como el dilema del prisionero o el dilema de la nieve acumulada estudiado en *4.1.3 - Recreación del modelo de Scatà et al.*

4.4 - Discusión

Los hallazgos de este trabajo ofrecen un puente entre dos enfoques teóricos diferentes. Por un lado, el *2.3.4 - Modelo de Martins et al., redes de tipo corruptas (2022)* ofrece un algoritmo para crear redes con métricas similares a las redes corruptas documentadas (*4.1.1 - Recreación de la red corrupta)*. Sin embargo, al asumir que todos los nodos son corruptos desde el inicio, no explica por qué surgen estas redes. En cambio, el modelo propuesto en este trabajo ofrece una dinámica evolutiva donde la creación de una red corrupta no es un punto de partida, sino un estado emergente que depende de la interacción entre la red y el juego.

Por otro lado, *2.7.2 - El juego de la corrupción* demuestra que es posible alcanzar un equilibrio cooperativo incluso en presencia de corrupción (*4.1.2.3 - México*). Sin embargo, al ser un modelo no espacial, asume que todos los individuos interactúan entre sí, lo cual es una simplificación de la realidad social. Un hallazgo tan importante como la vulnerabilidad de los nodos periféricos (discutido en la sección *4.2.2.1.1 - La configuración espacial importa*) no puede ser explicado sin un componente de red, subrayando la necesidad de integrar ambos enfoques. El análisis de la robustez y de las intervenciones revela una dualidad en la lucha contra la corrupción.

La persistencia de la corrupción requiere control central. Como se demostró en *4.2.2.1.2 - El umbral mínimo*, una masa crítica corrupta solo es funcional si parte de los nodos centrales de la red. Esto sugiere que las redes de corrupción consolidadas dependen de una estructura jerárquica y de actores clave que mantienen la cohesión en la red. Validando los resultados expuestos en *1.3.1.1 - Casos de estudio relevantes.*

La erradicación de la corrupción requiere una disrupción aleatoria. La estrategia más efectiva para desmantelar una red corrupta no fue un ataque directo a su centro o desde la periferia, sino una intervención aleatoria (*4.2.2.2.1.1 - La configuración espacial importa*). Un ataque centralizado tiende a fallar porque, en redes libres de escala, la estructura es robusta: si se elimina un nodo central, los nodos de jerarquía media reorganizan las conexiones, absorbiendo la intervención. Por otro lado, una intervención periférica carece de la influencia necesaria para revertir la tendencia mayoritaria.

En cambio, la intervención aleatoria demostró ser superior debido a que fragmenta la comunidad central. La eliminación aleatoria de nodos degrada rápidamente la conectividad global de la red. Al atacar al azar, estadísticamente se rompen enlaces que son vitales para unir diferentes comunidades. Así la red corrupta se fractura en múltiples comunidades aisladas posibilitando que la estrategia cooperativa emerja y se expanda hasta llegar a un equilibrio.

Identificar con precisión qué enlaces son los más importantes requiere un mapa de todas las interacciones, debido a que la red corrupta está oculta en la mayoría de los casos, encontrar esos enlaces clave es costoso. Una intervención aleatoria surge como la alternativa costo-efectiva: logra destruir la cohesión estructural sin necesidad de invertir muchos recursos en desenmascarar la totalidad de la red antes de actuar.

4.4.1 - ¿Cuál es la mejor estrategia?

A la luz de los hallazgos de este trabajo, la búsqueda de una solución única parece insuficiente. La corrupción es un sistema adaptativo. Por lo tanto, una estrategia óptima deberá ser híbrida, atacando simultáneamente tanto a los incentivos individuales como a la cohesión estructural.

4.4.1.1 - Modificar el juego

Si la matriz de pagos sigue favoreciendo la corrupción, ninguna intervención estructural será permanente. Es necesario que la corrupción sea una estrategia perdedora a largo plazo. Esto implica fortalecer el estado de derecho ($-c$) para que los castigos ($-q$) sean severos, y la impunidad ($-d$) sea baja.

Es el camino seguido por países como Singapur, creando un entorno fundamentalmente hostil a la corrupción (*1.2.2 - Caso Singapur*).

4.4.1.2 - Atacar aleatoriamente la red

Simultáneamente, crear políticas que permitan reemplazar nodos en cualquier parte de la red de forma aleatoria y sencilla. Por medio de rotación obligatoria de personal en puestos de alto riesgo (ej. aduanas, compras públicas, gestión de licencias) se impediría la formación de redes clientelares a largo plazo.

Una manera de hacerlo es implementar políticas que promuevan la digitalización y apertura total de datos (*1.2.1 - Caso Dinamarca*), al fomentar la transparencia se reduce el costo de monitoreo para la sociedad y se hacen visibles la mayoría de las redes asociadas a contrataciones o flujo de recursos (*1.3.1.1 - Casos de estudio relevantes*). Con ello, se facilita la identificación de los enlaces clave, permitiendo que las intervenciones aleatorias puedan complementarse con remociones específicas.

Cabe mencionar que dichas políticas han sido propuestas anteriormente pero no han proliferado en su implementación debido a temas políticos (Casar y Cejudo, 2023).

C D
C (3,3) (0,5)
D (5,0) (1,1)
A B

Conclusiones

Los límites de los modelos matemáticos

La construcción de modelos constituye una tarea compleja. Un modelo demasiado simple, con pocas variables, difícilmente logra representar de manera adecuada la realidad; mientras que un modelo excesivamente complejo puede generar parámetros difíciles de controlar y conducir a una falta de comprensión del fenómeno. Por ello, resulta fundamental encontrar un equilibrio que permita, al mismo tiempo, representar con fidelidad el fenómeno y facilitar su interpretación.

La corrupción es un fenómeno social influenciado por una gran variedad de factores, incluyendo la economía, la política y la cultura. Este trabajo demuestra que la corrupción no debe combatirse únicamente como una serie de delitos individuales, sino como un sistema complejo con propiedades estructurales específicas. La principal aportación de esta tesis no es solo el modelo matemático, sino la revelación de la fragilidad oculta de estas redes: su dependencia crítica de la organización centralizada.

Al integrar la teoría de redes con la teoría de juegos, se evidenció que las estrategias convencionales para enfocar los ataques hacia los nodos centrales son ineficientes. En contraste, se demuestra que las intervenciones aleatorias son el mecanismo más potente para fracturar la cohesión de la red.

Trabajo futuro

Finalmente, este trabajo abre nuevas vías de investigación, la vía más clara es validar este modelo con datos abiertos de contrataciones públicas o con registros judiciales para comprobar si dentro de las empresas o departamentos de gobierno se presenta una dinámica similar, con ello se completaría el proceso de la modelación matemática (*Figura 3.1*) lo cual permitiría validar las observaciones aquí presentadas. Al transformar un modelo teórico en una herramienta de diagnóstico y pronóstico para la formulación de políticas públicas más efectivas y posteriormente a la recuperación del estado de derecho en México, así como en otros países que enfrentan problemáticas estructurales similares.

C D
C (3,3) (0,5)
D (5,0) (1,1)
A B

Bibliografía

B.1 - Artículos científicos

Antal, T., Krapivsky, P., & Redner, S. (2005). Dynamics on social balance on networks. *Physical Review E*. doi: https://doi.org/10.1103/PhysRevE.72.036121

Barabási, A.-L., & Albert, R. (1999). Emergence of Scaling in Random Networks. *Science*, 509-512. doi: https://doi.org/10.1126/science.286.5439.509

Erdös, P., & Rényi, A. (1959). On Random Graphs I. *Publicationes Mathematicae*, 290–297. doi: https://doi.org/10.1126/science.286.5439.509

Granovetter, M. (1978). Threshold Models of Collective Behavior. *American Journal of Sociology*. doi: http://dx.doi.org/10.1086%2F226707

Luna-Pla, I., & Nicolás-Carlock, J. (2020). Corruption and complexity: a scientific framework for the analysis of corruption networks. *Applied Network Science*. doi: https://doi.org/10.1007/s41109-020-00258-2

Martins, A., da Cunha, B., Hanley, Q., Gonçalves, S., Perc, M., & Ribeiro, H. (2022). Universality of political corruption networks. *Nature Scientific Reports*. doi: https://doi.org/10.1038/s41598-022-10909-2

Milgram, S. (1967). The Small World Problem. *Psychology Today*. Ziff-Davis Publishing Company.

Milli, L., Rossetti, G., Pedreschi, D., & Giannotti, F. (2018). Active and passive diffusion processes in complex networks. *Applied network science*, 1-15. doi: https://doi.org/10.1007/978-3-319-72150-7_25

Scatà, M., Di Stefano, A., La Corte, A., Liò, P., Catania, E., Guardo, E., & Pagano, S. (2016). Combining evolutionary game theory and network theory to analyze human cooperation patterns. *Chaos, Solitons and Fractals*, 17–24. doi: https://doi.org/10.1016/j.chaos.2016.04.018

Strogatz, S. H., & Watts, D. J. (1998). Collective dyanamis of 'small-world' networks. *Nature*, 440-442. doi: https://doi.org/10.1038/30918

Ubeda, F., & Dueñez-Guzman, E. (2010). Power and Corruption. *Evolution*, 1127-1139. doi: https://doi.org/10.1111/j.1558-5646.2010.01194.x

B.2 - Libros

Barabási, A.-L. (2016). *Network Science.* Cambridge University Press.

Cressman, R. (2003). *Evolutionary Dynamics and Extensive Form Games.* The MIT Press.

Geltner, G., Kroeze, R., & Vitoria, A. (2018). *Anticorruption in History: From Antiquity to the Modern Era.* Oxford University Press.

McNulty, K. (2022). *Handbook of Graphs and Networks in People Analytics.* CRC Press.

Mitchell, M. (2009). *Complexity: A Guided Tour.* Oxford University Press.

Moreno, J. (1934). *Who Shall Survive?* New York.

Stewart, J., Clegg, D., Watson, S., & Redlin, L. (2020). *Single Variable Calculus, Early Transcendentals* (9 ed.).

Zapata Lillo, P. (2016). *Economía, política y otros juegos.* Ciudad de México: Las Prensas de Ciencias.

B.3 - Artículos periodísticos

Casar, M. A., & Cejudo, Q. (Diciembre de 2023). México: Anatomía de la Corrupcion. *Mexicanos Contra la Corrupción y la Impunidad* https://contralacorrupcion.mx/anatomia-de-la-corrupcion-cuarta-edicion/

Castillo, M., Roldán, N., Ureste, M., Bobadilla, O., Alberro, Y. C., Santamaría, J., Moreno Chávez, D. (4 de Septiembre de 2017). La Estafa Maestra. Graduados en desaparecer dinero público. *Animal Político* https://www.animalpolitico.com/estafa-maestra/

Gossaín, J. (Julio de 2019). ¿Quiere saber cómo fue que acabaron con la corrupción en Singapur? *El Tiempo.* https://www.eltiempo.com/colombia/otras-ciudades/como-se-acabo-la-corrupcion-en-singapur-386920

Chêne, M. (Noviembre de 2011). What makes New Zealand, Denmark, Finland, Sweden and others 'cleaner' than most countries? *Transparencia Internacional* https://www.transparency.org/en/blog/what-makes-new-zealand-denmark-finland-sweden-and-others-cleaner-than-most-countries

B.4 - Informes

Legatum Institute Foundation. (2023). *The 2023 Legatum Prosperity Index.* Obtenido de https://index.prosperity.com/about/resources

Transparencia Internacional. (2024). *Índice de Percepción de la Corrupción 2023.* Obtenido de https://www.transparency.org/en/cpi/2023

C
D
C
(3,3)
(0,5)
D
(5,0)
(1,1)
A
B

Apéndices

A.1 - Replicador dinámico del juego de la corrupción

El vector de estrategias corresponderá a un vector de cuatro entradas,

$$\boldsymbol{x} = \begin{pmatrix} C \\ D \\ H \\ K \end{pmatrix}$$

Y la matriz $\mathbf{A}$ corresponderá a la tabla de pagos del juego,

$$\mathbf{A} = \begin{pmatrix} r & -s & r & -s \\ t & 0 & t-p & -p \\ r & -s-c & r & -s-d \\ t & -c & t-q & -q-d \end{pmatrix}$$

El objetivo es calcular las ecuaciones diferenciales asociadas a cada estrategia.

A.1.1 - Cálculo de los pagos medios

Se calcula como el producto de la matriz de pagos y el vector de estrategias

$$f(\boldsymbol{x}) = A\boldsymbol{x} = \begin{pmatrix} r & -s & r & -s \\ t & 0 & t-p & -p \\ r & -s-c & r & -s-d \\ t & -c & t-q & -q-d \end{pmatrix} \begin{pmatrix} C \\ D \\ H \\ K \end{pmatrix} = \begin{pmatrix} f_C(\boldsymbol{x}) \\ f_D(\boldsymbol{x}) \\ f_H(\boldsymbol{x}) \\ f_K(\boldsymbol{x}) \end{pmatrix}$$

Por ejemplo, el pago medio para los Civiles Cooperadores, $f_C(\boldsymbol{x})$, se calcula como:

$$f_C(\boldsymbol{x}) = r \cdot C + \ -s \ \cdot D + r \cdot H + \ -s \ \cdot K$$

Y así sucesivamente para las otras estrategias.

$$f_D(\boldsymbol{x}) = t \cdot C + 0 \cdot D + (t-p) \cdot H + (-p) \cdot K$$

$$f_H(\boldsymbol{x}) = r \cdot C + \ -s-c \ \cdot D + r \cdot H + \ -s-d \ \cdot K$$

$$f_K(\boldsymbol{x}) = t \cdot C + (-c) \cdot D + (t-q) \cdot H + (-q-p) \cdot K$$

A.1.2 - Cálculo del pago medio de la población

Se calcula el pago medio de toda la población, $\overline{f}(\boldsymbol{x})$, tomando el producto interno del vector de estrategias $\boldsymbol{x}$ y el vector de pagos medios $f(\boldsymbol{x})$.

$$\overline{f}(\boldsymbol{x}) = \boldsymbol{x}^T A\boldsymbol{x} = \boldsymbol{x}^T f(\boldsymbol{x}) = (C\ D\ H\ K)\begin{pmatrix} f_C(\boldsymbol{x}) \\ f_D(\boldsymbol{x}) \\ f_H(\boldsymbol{x}) \\ f_K(\boldsymbol{x}) \end{pmatrix}$$

$$= C \cdot f_C(\boldsymbol{x}) + D \cdot f_D(\boldsymbol{x}) + H \cdot f_H(\boldsymbol{x}) + K \cdot f_K(\boldsymbol{x})$$

Este pago medio de la población es un promedio de los pagos medios de cada estrategia, ponderado por la frecuencia de cada estrategia en la población.

A.1.3 - Cálculo de las ecuaciones diferenciales

Siguiendo la forma general de la ecuación del replicador, para cada estrategia:

- **Civiles Cooperadores (C):**

$$\frac{dC}{dt} = C[f_C(\boldsymbol{x}) - \overline{f}(\boldsymbol{x})] = \ C(rC - sD + rH - sK - \overline{f}(\boldsymbol{x}))$$

- **Civiles No Cooperadores (D):**

$$\frac{dD}{dt} = D[f_D(\boldsymbol{x}) - \overline{f}(\boldsymbol{x})] = \ D(tC + (t-p)H - pK - \overline{f}(\boldsymbol{x}))$$

- **Poder Honrado (H):**

$$\frac{dH}{dt} = H[f_H(\boldsymbol{x}) - \overline{f}(\boldsymbol{x})] = \ H(rC - (s+c)D + rH - (s+d)K - \overline{f}(\boldsymbol{x}))$$

- **Poder Corrupto (K):**

$$\frac{dK}{dt} = K[f_K(\boldsymbol{x}) - \overline{f}(\boldsymbol{x})] = \ K(tC - cD + (t-q)H - (q+d)K - \overline{f}(\boldsymbol{x}))$$

Estas ecuaciones describen la evolución de cada estrategia considerando la competencia entre ellas en el juego de la corrupción. Dichas ecuaciones fueron programadas en *A.4.2 - Recreación del juego de la corrupción* para estudiar la dinámica general del juego.

A.2 - El dilema de la nieve acumulada

También conocido como el *juego de la gallina* o *juego del halcón y la paloma* (Zapata Lillo, 2016), es una situación hipotética en la que dos conductores se encuentran atrapados en una carretera bloqueada por un banco de nieve. Ambos desean cruzar, pero para hacerlo es necesario despejar el camino removiendo la nieve. Aunque ambos quieren llegar a su destino, prefieren no realizar el esfuerzo físico.

Lo interesante ocurre al analizar la dependencia de las decisiones: si ninguno la remueve, ambos se quedan atascados y obtienen el peor resultado. Si ambos deciden retirarla, comparten el trabajo y cruzan más rápido. Sin embargo, si uno decide quitarla y el otro decide esperar, quien espera obtiene el beneficio de cruzar sin el costo del esfuerzo, mientras que la otra carga con todo el trabajo.

A diferencia del dilema del prisionero, aquí la mejor estrategia depende estrictamente de lo que haga el otro jugador. Si el otro conductor se niega a ayudar, la elección racional es hacerlo uno mismo, pues es preferible trabajar solo (1 punto) que quedarse atascado indefinidamente (0 puntos). Pero si el otro conductor decide ayudar, la mejor opción individual es no hacer nada para obtener el máximo beneficio (3 puntos).

Tabla A.1 - Matriz de pagos que representa el dilema de la nieve acumulada

	Remover	Esperar
Remover	2	1
Esperar	3	0

Este escenario ilustra una dinámica entre el beneficio mutuo y el personal: la cooperación es posible, pero la tentación de aprovecharse del esfuerzo ajeno genera una tensión constante.

A.3 - Modelos descartados

En etapas tempranas de la tesis, se propuso un modelo basado en *2.5.2 - Dinámica del umbral con perfiles (2018)* y *2.5.3 - Dinámica del balance social (2005)*, simulaba la propagación de la corrupción como un proceso de difusión. Bajo ciertas condiciones toda la red quedaba infectada (*Figura A.3.1*). Generaba resultados muy limitados debido a los pocos parámetros con los que se podían trabajar.

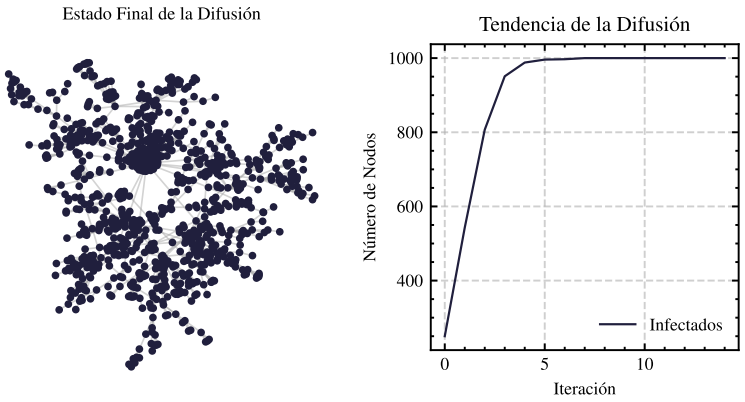


Figura A.3.1 – El modelo del umbral con perfiles propagaba cualquier estrategia a toda la red en muy pocas iteraciones, lo cual no era realista para el fenómeno de la corrupción. Fue necesario añadir más restricciones a los nodos para contener la propagación.

Posteriormente se trabajó con una red multiplex (*Figura A.3.2*), añadiendo una segunda red para simular 2 procesos diferentes. La primera red se destinó a modelar el concepto del poder, mediante una red libre de escala con la dinámica del umbral con perfiles, la segunda red fue enfocada en modelar las relaciones personales mediante una red creada con la *2.5.3 - Dinámica del balance social (2005).*

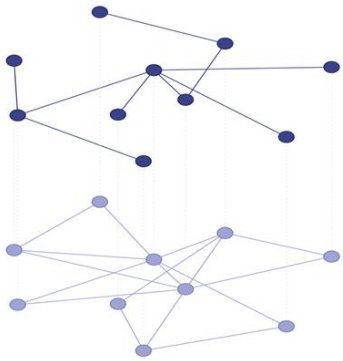


Figura A.3.2 – Red multiplex de 2 capas propuesta.

El modelo se descartó debido a su complejidad innecesaria y las limitaciones en sus resultados, aun teniendo más parámetros para trabajar no podía simular la robustez social (*4.2.2.1 - Evaluación de la robustez social*) y otros escenarios.

A.4 - Código

Repositorio en línea donde se pueden acceder a todos los archivos para su descarga libre.

https://github.com/ElCaballerodelaTierra/Corrupcion/

A continuación se comparten las partes más importantes del código empleado para recrear los modelos estudiados en *3.1 - Análisis e implementación de modelos previos* y *3.2 - Desarrollo de un nuevo modelo.*

A.4.1 - Recreación de la red corrupta

El principal cambio respecto al código original de Martins et al (*3.1.1 - Recreación de la red corrupta*) fue usar la librería *NetworkX* para eliminar los enlaces duplicados y visualizar las redes, con ello no es necesario crear un entorno virtual para usar la librería *graph-tool* en *Windows*, lo cual permite reproducir el código en otros sistemas operativos.

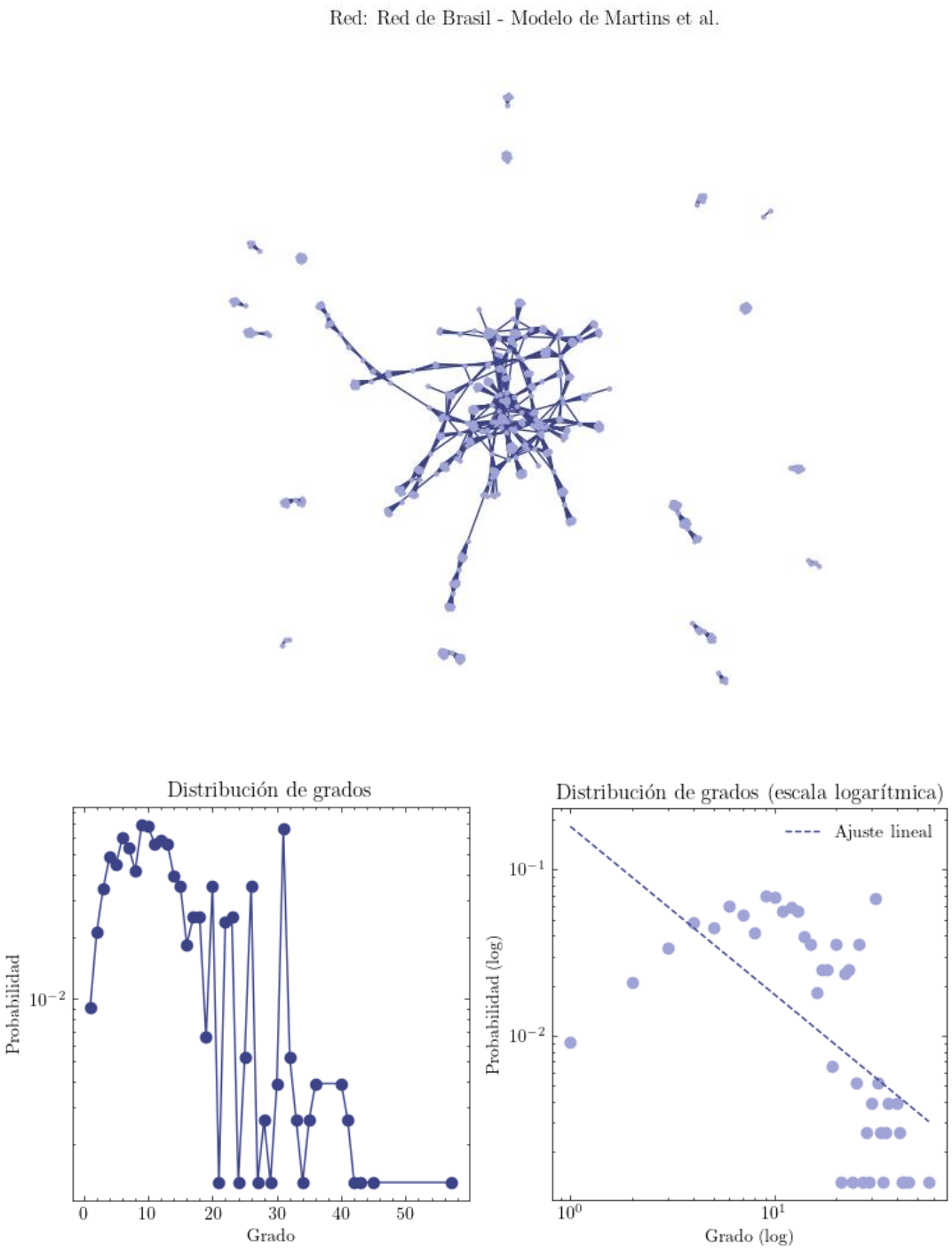
```
#Código original de los autores con varias modificaciones para adaptarlo a la versión actual de NetworkX, además
se mejorara la eficiencia del código

#Se intalan las librerias necesarias para todos los programas
%pip install numpy scipy mathplotlib networkx ndlib scienceplots

import numpy as np
import networkx as nx
import matplotlib.pyplot as plt  # Importar matplotlib para visualización con networkx
from itertools import combinations

def generate_net_links(tmax = 100, lambda_ = 7.33, a = 0.09, b = -11.5, proba = 0.024):
    t                         = 0
    links                     = []
    links_list                = []
    agent_names               = set()
    last_agent_index          = 0
    repeated_agent_names      = set()
    total_repeated_agents     = 0

    while t < tmax:
        n_new_agents = int(np.round(np.random.exponential(lambda_)))

        if n_new_agents > 1:

            new_agent_names     = [x for x in np.arange(last_agent_index, last_agent_index + n_new_agents)]
            last_agent_index   += n_new_agents

            new_repeated_agents = int(np.round((a*(len(agent_names)) + b - total_repeated_agents)))

            if new_repeated_agents > 0:
                for i in range(min(new_repeated_agents, len(new_agent_names))):

                    if (np.random.uniform() <= proba) & (len(repeated_agent_names) > 0):
                        #select from repeated_agent_names
                        repeated_agent = np.random.choice(list(repeated_agent_names))
                    else:
                        #select from agent_names
                        repeated_agent = np.random.choice(list(agent_names.difference(repeated_agent_names)))
                        #Corrección: usar diferencia de conjuntos
                        total_repeated_agents += 1

                    new_agent_names[i] = repeated_agent
                    repeated_agent_names.add(repeated_agent)

            for agent_ in new_agent_names:
                agent_names.add(agent_)

            new_links = list(combinations(new_agent_names, 2))

            links_list += [new_links]
            t          += 1

    return links_list

edge_lists = generate_net_links(tmax = 100, a = 0.142)
# e.g, [[(0, 1), (0, 3), ... (5, 6)], [(7, 8), (7, 9) ... (14, 15)], ... ]

#Modificación del código original para trabajar con networkx
def network(t, edges_list):
    g           = nx.Graph() # Crear grafo de networkx
    edge_list_t = np.concatenate(edges_list[:t])
    g.add_edges_from(edge_list_t) # Añadir aristas con networkx
```

```
    # Eliminar self loops con networkx

    g.remove_edges_from(nx.selfloop_edges(g))

    # networkx no tiene una función directa para eliminar aristas paralelas
    # En este caso, add_edges_from no añade aristas paralelas duplicadas por defecto, por lo que este paso puede
omitirse en networkx si se desea el mismo comportamiento.
    # Si se necesitara eliminar aristas paralelas explícitamente (en caso de que se generaran de otra forma), se
requeriría un manejo adicional.

    return g

final_network = network(100, edge_lists)
final_network

# Visualización de la red con networkx y matplotlib

# Calcular la disposición de los nodos con networkx (spring_layout es similar a SFDP)
pos = nx.spring_layout(final_network)

# Dibujar la red con networkx y matplotlib
nx.draw(final_network, pos, node_size=5, width=0.5, with_labels=False) # Ajustar parámetros para networkx

plt.savefig("corruption_network_networkx.png") # Guardar la figura usando matplotlib
plt.show() # Mostrar la figura (opcional)

#Termina código modificado de los autores, a continuación, se crea la función para calcular las métricas de la red

#Función que calcula todas las métricas de la red

import networkx as nx
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import linregress
import scienceplots
from networkx.algorithms import community

#Funcion para contar el número de nodos y aristas en la red

def contarNodosAristas(G):
    num_nodos = G.number_of_nodes()
    num_aristas = G.number_of_edges()
    print(f"Número de nodos: {num_nodos}")
    print(f"Número de aristas: {num_aristas}")

#Funcion para graficar la red

def mostrarRed(G,nombre):

    plt.style.use('science')
    plt.figure(figsize=(6, 6))
    pos = nx.spring_layout(G)
    nx.draw(G, pos, node_size=5, node_color='#9FA3D5', edge_color='#3B4285', with_labels=False)
    plt.title(f'Red: {nombre}')
    plt.savefig(f'red_{nombre}.svg', dpi=300)
    plt.show()

#Funcion para graficar la distribución de grados

def distribucionDeGrado(G, nombre):
    # Obtener los grados de los nodos y calcular la distribución de frecuencia
    degrees = list(dict(nx.degree(G)).values())
    unique_degrees, counts = np.unique(degrees, return_counts=True)
    probs = counts / len(G)

    # Eliminar grados con frecuencia cero
    non_zero_probs = probs[probs > 0]
    non_zero_degrees = unique_degrees[probs > 0]

    # Tomar el logaritmo de las probabilidades y los grados
    log_probs = np.log(non_zero_probs)
    log_degrees = np.log(non_zero_degrees)

    # Realizar un ajuste lineal
    slope, intercept, r_value, p_value, std_err = linregress(log_degrees, log_probs)

    # Calcular el exponente de la ley de potencias
    gamma = round(slope,2)
```

```
    # Crear el gráfico

    plt.style.use('science')

    # Graficar con escala logarítmica únicamente en el eje y (Distribución de grados en una red)
    plt.figure(figsize=(4, 4))

    plt.plot(non_zero_degrees, non_zero_probs, 'o-', color='#3B4285')
    plt.yscale('log')
    plt.xlabel('Grado')
    plt.ylabel('Probabilidad')
    plt.title('Distribución de grados')
    plt.legend()

    plt.savefig(f'distribucion_de_grados_{nombre}.svg', dpi=300)
    plt.show()

    plt.figure(figsize=(4, 4))

    plt.loglog(unique_degrees, probs, 'o', color='#9FA3D5')
    plt.plot(unique_degrees, np.exp(intercept + slope * np.log(unique_degrees)), '--', label='Ajuste lineal',
color='#3B4285')

    plt.xlabel('Grado (log)')
    plt.ylabel('Probabilidad (log)')
    plt.legend()
    plt.title('Distribución de grados (escala logarítmica)')

    # Guardar el gráfico con mayor resolución
    plt.savefig(f'distribucion_de_grados_log_{nombre}.svg', dpi=300)
    plt.show()

    # Imprimir el exponente
    print("Exponente de la ley de potencias (gamma):", gamma)

#Función para calcular la estructura en comunidades

def estructuraComunidades(G, nombre):
    # Detectar comunidades usando el algoritmo de detección de comunidades de Louvain
    comunidades = community.louvain_communities(G)

    # Número de comunidades
    num_comunidades = len(comunidades)

    #Cantidad de nodos en cada comunidad
    nodos_comunidad = [len(comunidad) for comunidad in comunidades]

    # Promedio de nodos por comunidad
    promedio_nodos = round(np.mean(nodos_comunidad),2)

    # Crear un diccionario de colores para las comunidades
    color_map = {}
    for i, comunidad in enumerate(comunidades):
        for node in comunidad:
            color_map[node] = i

    # Crear el gráfico de la red con colores para las comunidades
    plt.figure(figsize=(6, 6)) # Tamaño del gráfico
    pos = nx.spring_layout(G)
    nx.draw(G, pos, node_size=5, node_color=list(color_map.values()), edge_color='gray', with_labels=False,
cmap=plt.cm.tab20)
    plt.title(f'Red: {nombre} (Comunidades)')
    plt.savefig(f'comunidades_{nombre}.svg', dpi=300)
    plt.show()

    # Imprimir el número de comunidades y el tamaño de cada una
    print(f"Número de comunidades: {num_comunidades}")
    print(f"Número de nodos por comunidad: {nodos_comunidad}")
    print(f"Promedio de nodos por comunidad: {promedio_nodos}")

    # Encontrar la comunidad más grande
    max_comunidad = max(nodos_comunidad)

    print(f"Comunidad más grande: {max_comunidad} nodos")

    #Etiquetar los nodos de la comunidad más grande
    nodos_max_comunidad = [node for node in G.nodes() if node in comunidades[np.argmax(nodos_comunidad)]]
```

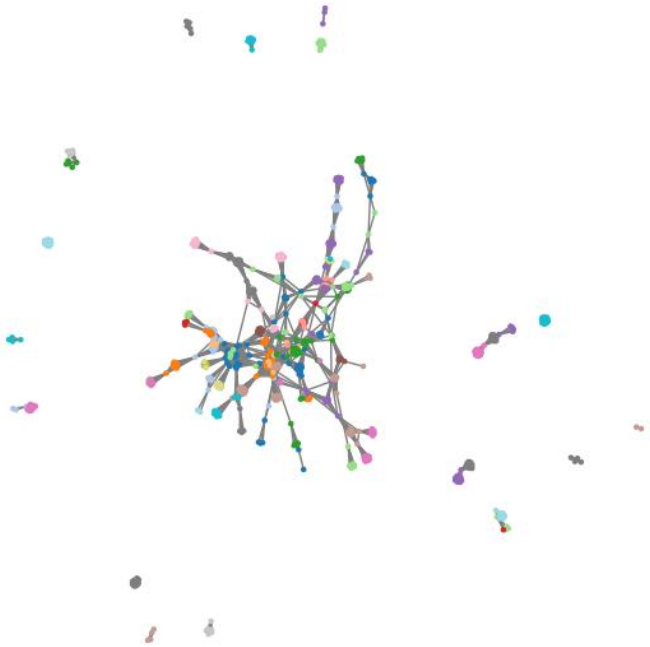
```
    #Seleccionar la red de la comunidad más grande
    G_max_comunidad = G.subgraph(nodos_max_comunidad)
    return G_max_comunidad


# Propiedad de mundo pequeño

def mundoPequeno(G, G_max_comunidad, nombre):

    # Calcular el coeficiente de agrupamiento global
    clustering_coeff = round(nx.average_clustering(G),2)

    # Calcular el coeficiente de agrupamiento local (comunidad más grande)
    clustering_coeff_local = round(nx.average_clustering(G_max_comunidad),2)

    # Calcular el camino medio más corto de la red local
    try:
        average_path_length = round(nx.average_shortest_path_length(G_max_comunidad),2)
    except nx.NetworkXError:
        average_path_length = round(float('inf'),2)

    print(f"Coeficiente de agrupamiento global: {clustering_coeff}")
    print(f"Coeficiente de agrupamiento local: {clustering_coeff_local}")
    print(f"Camino medio más corto: {average_path_length}")

    if  (clustering_coeff_local > .90 and average_path_length < 5):
        print(f"La red '{nombre}' tiene la propiedad de mundo pequeño.")
    else:
        print(f"La red '{nombre}' no tiene la propiedad de mundo pequeño.")

#Asortaividad

def asortatividad(G, G_max_comunidad):
    # Calcular el coeficiente de asortatividad de grado global
    assortativity_coeff = round(nx.degree_assortativity_coefficient(G),2)
    # Calcular el coeficiente de asortatividad de grado local
    assortativity_coeff_local = round(nx.degree_assortativity_coefficient(G_max_comunidad),2)
    # Imprimir los coeficientes de asortatividad
    print(f"Coeficiente de Asortatividad Global: {assortativity_coeff}")
    print(f"Coeficiente de Asortatividad Local: {assortativity_coeff_local}")

def densidad(G, G_max_comunidad):
    # Calcular la densidad de la red global
    density = round(nx.density(G), 2)
    # Calcular la densidad de la red local
    density_local = round(nx.density(G_max_comunidad), 2)
    # Imprimir la densidad
    print(f"Densidad de la red global: {density}")
    print(f"Densidad de la red: {density_local}")

#Función global que calcula todas las métricas

def metricasRed(G, nombre):
    # Contar nodos y aristas
    contarNodosAristas(G)
    # Mostrar la red
    mostrarRed(G, nombre)
    # Obtener los grados de los nodos y calcular la distribución de frecuencia
    distribucionDeGrado(G, nombre)
    # Estructura en comunidades
    G_max_comunidad = estructuraComunidades(G, nombre)
    # Propiedad de mundo pequeño
    mundoPequeno(G, G_max_comunidad, nombre)
    # Asortatividad
    asortatividad(G, G_max_comunidad)
    # Densidad
    densidad(G, G_max_comunidad)

#Calcular las métricas del modelo de red de corrupción de Brasil
nombre = 'Red de Brasil - Modelo de Martins et al.'
metricasRed(final_network, nombre)
```

```
Número de nodos: 765
Número de aristas: 5215
```




```
Exponente de la ley de potencias (gamma): -1.01
```

Red: Red de Brasil - Modelo de Martins et al. (Comunidades)



```
Número de comunidades: 35
Número de nodos por comunidad: [55, 23, 13, 32, 34, 23, 21, 49, 11, 7, 44, 8, 13, 3, 26, 44, 2, 4, 11, 32, 46, 36,
23, 33, 31, 39, 5, 10, 4, 7, 7, 36, 4, 14, 15]
Promedio de nodos por comunidad: 21.86
Comunidad más grande: 55 nodos
Coeficiente de agrupamiento global: 0.94
Coeficiente de agrupamiento local: 0.96
Camino medio más corto: 2.14
La red 'Red de Brasil - Modelo de Martins et al.' tiene la propiedad de mundo pequeño.
Coeficiente de Asortatividad Global: 0.72
Coeficiente de Asortatividad Local: 0.64
Densidad de la red global: 0.02
Densidad de la red: 0.26
```

A.4.2 - Recreación del juego de la corrupción

Se implementó el modelo del replicador con las ecuaciones diferenciales descritas en *A.1.3 - Cálculo de las ecuaciones diferenciales*. Se define la función que modela y grafica el juego de la corrupción, posteriormente se ingresan los parámetros para cada sociedad.

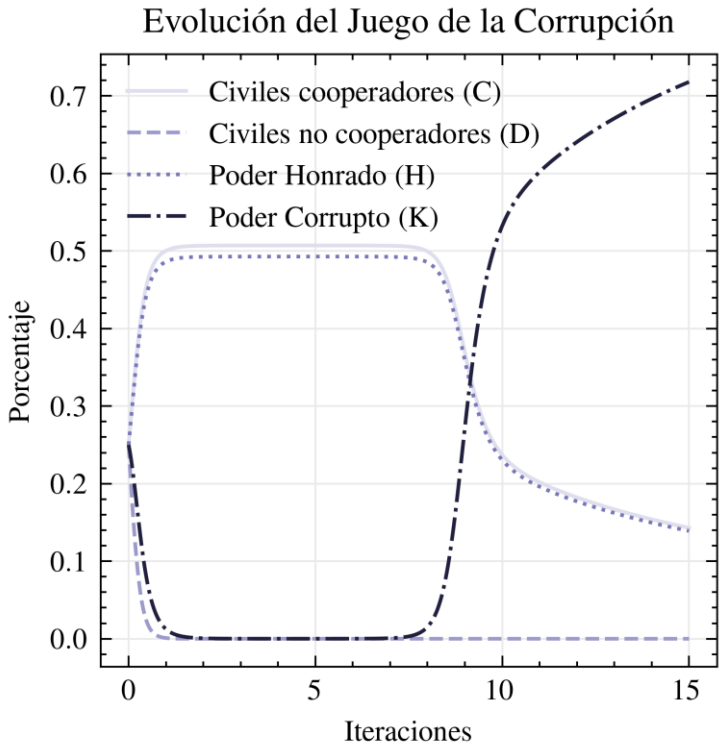
```
import numpy as np
import matplotlib.pyplot as plt
from scipy.integrate import odeint
import scienceplots
import matplotlib.patches as patches
import os

plt.style.use(('science', 'ieee'))

# Función para simular y graficar el juego de la corrupción
def run_corruption_simulation(r, s, t, p, q, c, d, tiempo_inicial, tiempo_final, num_puntos, filename="default"):
    """
    Simula el juego de la corrupción y genera ambas gráficas.
    Args:
        r, s, t, p, q, c, d: Parámetros del juego.
        tiempo_inicial (int): Valor inicial del rango de tiempo.
        tiempo_final (int): Valor final del rango de tiempo.
        num_puntos (int): Cantidad de puntos de tiempo.
        filename (str): Nombre base para los archivos de salida.
    """
    """
    # Función para verificar las desigualdades
    #Se desactivó porque en los resultados de los investigadores usaron parametros para el juego del dilema del
prisionero que no cumplian estas desigualdades

    def check_inequalities(r, s, t, p, q, c, d):
        if not (t > r > t - s):
            print("Error: No se cumple t > r > t - s")
            return False
        if not (t - s > 0):
            print("Error: No se cumple t - s > 0")
            return False
        if not (p > 0):
            print("Error: No se cumple p > 0")
            return False
        if not (c > 0):
            print("Error: No se cumple c > 0")
            return False
        if not (q > 0):
            print("Error: No se cumple q > 0")
            return False
        if not (d > 0):
            print("Error: No se cumple d > 0")
            return False
        return True
    """
    # Ecuaciones diferenciales para el modelo del Juego de la Corrupción
    def corruption_game_equations(x, t):
        c, d, h, k = x
        payoff_matrix = np.array([
            [r,   -s,  r,  -s],
            [t,    0, t-p, -p],
            [r, -s-c, r, -s-d],
            [t,  -c, t-q, -q-d]
        ])
        strategy_vector = np.array([c, d, h, k])
        mean_payoffs = payoff_matrix @ strategy_vector
        population_mean_payoff = np.dot(strategy_vector, mean_payoffs)
        dc_dt = c * (mean_payoffs[0] - population_mean_payoff)
        dd_dt = d * (mean_payoffs[1] - population_mean_payoff)
        dh_dt = h * (mean_payoffs[2] - population_mean_payoff)
        dk_dt = k * (mean_payoffs[3] - population_mean_payoff)
        return [dc_dt, dd_dt, dh_dt, dk_dt]
```

```
def simulate_corruption_game(initial_conditions, time_points):
    results = odeint(corruption_game_equations, initial_conditions, time_points, rtol=1e-8, atol=1e-8)
    return results

def plot_results(results, time_points, filename):
    c_results = results[:, 0]
    d_results = results[:, 1]
    h_results = results[:, 2]
    k_results = results[:, 3]

    plt.figure(figsize=(3, 3))

    colors = ['#DEDEEE', '#9B9ACA', '#7976B8', '#201F3D']
    plt.plot(time_points, c_results, label='Civiles cooperadores (C)', color=colors[0])
    plt.plot(time_points, d_results, label='Civiles no cooperadores (D)', color=colors[1])
    plt.plot(time_points, h_results, label='Poder Honrado (H)', color=colors[2])
    plt.plot(time_points, k_results, label='Poder Corrupto (K)', color=colors[3])

    plt.xlabel('Iteraciones')
    plt.ylabel('Porcentaje')
    plt.title('Evolución del Juego de la Corrupción')
    plt.legend()
    plt.grid(True, color="#EAEAEA")

    # Guardar la gráfica
    #os.makedirs(os.path.dirname(filename), exist_ok=True) # Crear directorio si no existe
    plt.savefig(f'{filename}_juegoDeLaCorrupcion.svg', dpi=300)
    plt.show()

def plot_payoff_table(r, s, t, p, q, c, d, filename):
    payoff_matrix = np.array([
        [r, -s, r, -s],
        [t, 0, t - p, -p],
        [r, -s - c, r, -s - d],
        [t, -c, t - q, -q - d]
    ])

    fig, ax = plt.subplots(figsize=(2, 1))
    ax.axis('off')

    strategies = [' C ', ' D ', ' H ', ' K ']
    colors = ['lavender', 'midnightblue']

    table = ax.table(cellText=payoff_matrix,
                     colLabels=strategies,
                     rowLabels=strategies,
                     cellLoc='center',
                     loc='center',
                     fontsize=12)
    table.auto_set_font_size(True)

    plt.title('Tabla de Pagos del Juego de la Corrupción')
    plt.savefig(f'{filename}_tablaPagos.png', dpi=300)
    plt.show()

    # Imprime la tabla de pagos numericamente
    print("\nTabla de Pagos (Valores Numéricos):\n")
    print("   ", end="")
    for strat in strategies:
        print(f"{strat}   ", end="")
    print()

    for i, row in enumerate(payoff_matrix):
        print(strategies[i], end=" ")
        for val in row:
            print(f"{val:>4} ", end="")
        print()
    print("\n")
```

```
# Condiciones iniciales
    initial_conditions = [0.25, 0.25, 0.25, 0.25]
    time_points = np.linspace(tiempo_inicial, tiempo_final, num_puntos)

    """
    # Verificar las desigualdades antes de simular
    if check_inequalities(r, s, t, p, q, c, d):
        # Simula el juego
        results = simulate_corruption_game(initial_conditions, time_points)
        # Grafica la tabla de pagos
        plot_payoff_table(r, s, t, p, q, c, d, filename)
        # Grafica los resultados
        plot_results(results, time_points, filename)
    else:
        # Grafica la tabla de pagos
        plot_payoff_table(r, s, t, p, q, c, d, filename)
        print("Error: Los parámetros no cumplen las desigualdades. La simulación no se ejecutará.")
    """
    # Simula el juego
    results = simulate_corruption_game(initial_conditions, time_points)
    # Grafica la tabla de pagos
    plot_payoff_table(r, s, t, p, q, c, d, filename)
    # Grafica los resultados
    plot_results(results, time_points, filename)

# Parámetros del juego propuesto para México

# Parametros sugeridos para el caso mexicano
# p > q castigo de las personas con poder es menor que para los civiles
# s < q + d < p el equilibrio entre (K) y (C) existe y es estable
# q + d << p
# 0 < c << 1

# Pagos del dilema del prisionero
r = 2 #Recompensa por cooperación mutua (Convivencia de acuerdo al contrato social)
s = 1 #Pérdida por ser cooperador cuando el otro no coopera (que se aprovechen de ti, sufrir una estafa, robo,
etc.)
t = 4 #Ganancia por ser no cooperador cuando el otro es cooperador (aprovecharse del otro, estafar, robar, etc.)

p = 10 #Castigo a los civiles (por ejemplo, cárcel por robo)
#Se ajusta el valor de p para que se cumpla la inecuación s < q + d < p
#Así se promueve una sociedad más cooperativa y menos corrupta (ver la sección de "Discusión" dentro del artículo
de Dueñez et al. 2010)
c = 2 #Costo por hacer cumplir las leyes, un buen estado de derecho implica bajos costos.

q = 5 #Castigo a las personas con poder (por ejemplo, cárcel por corrupción)
#Se ajusta el valor de q para que sea mayor que p
#Si p > q entonces la corrupción florece (ver la sección de "Discusión" del artículo de Dueñez et al. 2010)
d = 4 #Costo por castigar a alguien corrupto (por ejemplo, costo de tiempo y recursos para investigar un caso de
corrupción) Una sociedad con bajos índices de corrupcion tendrá costos muy bajos.

tiempo_inicial = 0
tiempo_final = 15
num_puntos = 1000
filename = "México"

# Llama a la función para ejecutar la simulación
run_corruption_simulation(r, s, t, p, q, c, d, tiempo_inicial, tiempo_final, num_puntos, filename)
```

Tabla de Pagos (Valores Numéricos):

	C	D	H	K
C	2	-1	2	-1
D	4	0	-6	-10
H	2	-3	2	-5
K	4	-2	-1	-9



A.4.3 - Modelo refinado de Scatà et al.

En la primera parte del código se definen los parámetros correspondientes a las tres sociedades junto con la función de pagos, después se crean las funciones que replican el algoritmo de Scatà et al. expuesto en la *Figura 3.4*, posteriormente se establecen las funciones de graficado y los valores para la simulación.

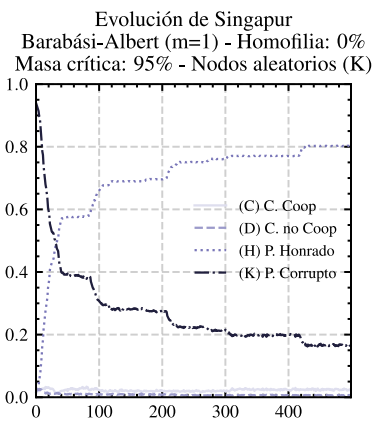
```
import networkx as nx
import numpy as np
import random
import matplotlib.pyplot as plt
import scienceplots
import os
from collections import Counter
import time
# Opcional para mostrar SVG en notebooks
try:
    from IPython.display import SVG, display, Image
except ImportError:
    SVG, display, Image = None, None, None
# --- Parámetros Juego y Funciones Auxiliares  ---
PARAMS_DINAMARCA = {'r': 2.5, 's': 1, 't': 3, 'p': 8, 'c': 0.2, 'q': 11, 'd': 0.5}
PARAMS_SINGAPUR =  {'r': 2, 's': 1, 't': 4, 'p': 10, 'c': 0.2, 'q': 15, 'd': 0.2}
PARAMS_MEXICO =    {'r': 2, 's': 1, 't': 4, 'p': 10, 'c': 2, 'q': 5, 'd': 4}

def calcular_pago_individual_corrupcion(strat_i, strat_j, params):
    r, s, t, p, c, q, d = params['r'], params['s'], params['t'], params['p'], params['c'], params['q'],
params['d']
    payoffs = { ('C', 'C'): r, ('C', 'D'): -s, ('C', 'H'): r, ('C', 'K'): -s, ('D', 'C'): t, ('D', 'D'): 0, ('D',
'H'): t-p, ('D', 'K'): -p, ('H', 'C'): r, ('H', 'D'): -s-c, ('H', 'H'): r, ('H', 'K'): -s-d, ('K', 'C'): t, ('K',
'D'): -c, ('K', 'H'): t-q, ('K', 'K'): -q-d }
    return payoffs.get((strat_i, strat_j), 0)

def actualizar_estrategia_corrupcion_red_homofilia(red, nodo, estrategias, recompensas_totales, K,
homophily_strength):
    estrategia_actual = estrategias[nodo]; recompensa_total_actual = recompensas_totales[nodo]
    vecinos = list(red.neighbors(nodo))
    if not vecinos: return estrategia_actual
    vecino_elegido = random.choice(vecinos); estrategia_vecino = estrategias[vecino_elegido];
recompensa_total_vecino = recompensas_totales[vecino_elegido]
    probabilidad_adoptar_base = 0.0
    try:
        if K <= 1e-9: probabilidad_adoptar_base = 1.0 if recompensa_total_vecino > recompensa_total_actual else
0.0
        else: delta_recompensa_scaled = np.clip((recompensa_total_actual - recompensa_total_vecino) / K, -700,
700); probabilidad_adoptar_base = 1 / (1 + np.exp(delta_recompensa_scaled))
    except (OverflowError, FloatingPointError): probabilidad_adoptar_base = 1.0 if recompensa_total_vecino >
recompensa_total_actual else 0.0
    if estrategia_actual == estrategia_vecino: probabilidad_adoptar_final = probabilidad_adoptar_base * (1 +
homophily_strength)
    else: probabilidad_adoptar_final = probabilidad_adoptar_base * (1 - homophily_strength)
    probabilidad_adoptar_final = np.clip(probabilidad_adoptar_final, 0.0, 1.0)
    return estrategia_vecino if random.random() < probabilidad_adoptar_final else estrategia_actual

def calcular_centralidad(red):
    try: centrality = nx.eigenvector_centrality_numpy(red); print("  Usando centralidad de Eigenvector.")
    except Exception as e: print(f"  Advertencia: Eigenvector falló ({e}). Usando Grado."); centrality =
nx.degree_centrality(red)
    return centrality

def seleccionar_nodos_cm(centrality_dict, cm_size, placement_type):
    if cm_size > len(centrality_dict): raise ValueError(f"CM size ({cm_size}) > num nodos.")
    if cm_size == 0: return []
    reverse_sort = (placement_type == 'Nodos centrales')
    nodos_ordenados = sorted(centrality_dict, key=centrality_dict.get, reverse=reverse_sort)
    if placement_type in ['Nodos centrales', 'Nodos perifericos']: cm_nodes = nodos_ordenados[:cm_size]
    elif placement_type == 'Nodos aleatorios': cm_nodes = random.sample(list(centrality_dict.keys()), cm_size)
    else: raise ValueError(f"Colocación CM desconocida: {placement_type}")
    return cm_nodes
```

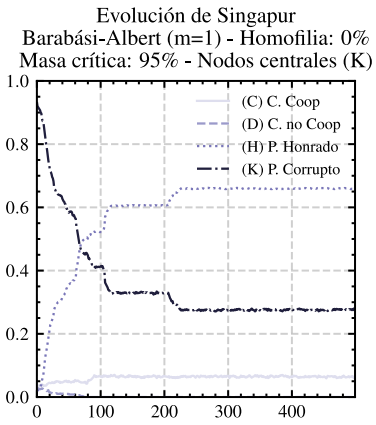
```
def inicializar_estrategias_cm_flexible(n, cm_nodes, cm_strategies, non_cm_strategies, seed=None):
    if seed is not None: random.seed(seed)
    estrategias = {}; non_cm_nodes = list(set(range(n)) - set(cm_nodes))
    if not cm_strategies: raise ValueError("cm_strategies vacía.")
    if not non_cm_strategies: raise ValueError("non_cm_strategies vacía.")
    random.shuffle(cm_nodes); random.shuffle(non_cm_nodes)
    for i, nodo in enumerate(cm_nodes): estrategias[nodo] = cm_strategies[i % len(cm_strategies)]
    for i, nodo in enumerate(non_cm_nodes): estrategias[nodo] = non_cm_strategies[i % len(non_cm_strategies)]
    return dict(sorted(estrategias.items()))
# --- Ejecución Simulación Modificada ---
def ejecutar_simulacion_corrupcion(
    num_nodos=100,
    network_type='Completa', network_params=None,
    cm_size=50, cm_percentage=50,
    cm_placement_type='Nodos aleatorios',
    cm_strategies = ['C', 'H'], non_cm_strategies = ['D', 'K'],
    params_juego=PARAMS_MEXICO, K=0.1, homophily_strength=0.0,
    rondas=15, seed=None
    ):
    if seed is not None: random.seed(seed); np.random.seed(seed)
    print(f"Creando red tipo '{network_type}'...")
    network_params_str_out = ""
    if network_type == 'Completa': red = nx.complete_graph(num_nodos)
    elif network_type == 'Barabási-Albert':
        if network_params is None: network_params = {}
        m = network_params.get('m', 2)
        if m >= num_nodos: m = max(1, num_nodos // 2)
        red = nx.barabasi_albert_graph(num_nodos, m, seed=seed); network_params_str_out = f"_(m={m})"
    else: raise ValueError(f"Tipo de red desconocido: {network_type}")
    print(f"Red creada: {red.number_of_nodes()} nodos, {red.number_of_edges()} enlaces.")
    print(f"Calculando centralidad...")
    centrality = calcular_centralidad(red)
    print(f"Seleccionando {cm_size} nodos CM ({cm_placement_type})...")
    cm_nodes = seleccionar_nodos_cm(centrality, cm_size, cm_placement_type)
    print(f"Inicializando estrategias: CM={cm_strategies}, No-CM={non_cm_strategies} con seed={seed}...")
    estrategias = inicializar_estrategias_cm_flexible(num_nodos, cm_nodes, cm_strategies, non_cm_strategies,
seed=seed)
    counts_init = Counter(estrategias.values()); print(f"  Estrategias iniciales: {counts_init}")
    historia_fracciones = {'C': [], 'D': [], 'H': [], 'K': []}; nodos_lista = list(red.nodes())
    print(f"Iniciando {rondas} rondas...")
    for r in range(rondas):
        recompensas_totales = {nodo: 0 for nodo in nodos_lista}
        for nodo_i in nodos_lista:
            strat_i = estrategias[nodo_i]; vecinos_i = list(red.neighbors(nodo_i))
            if not vecinos_i: continue
            for nodo_j in vecinos_i: strat_j = estrategias[nodo_j]; recompensas_totales[nodo_i] +=
calcular_pago_individual_corrupcion(strat_i, strat_j, params_juego)
        siguientes_estrategias = {}; nodos_actualizar = nodos_lista[:]; random.shuffle(nodos_actualizar)
        for nodo in nodos_actualizar: siguientes_estrategias[nodo] =
actualizar_estrategia_corrupcion_red_homofilia(red, nodo, estrategias, recompensas_totales, K, homophily_strength)
        estrategias = siguientes_estrategias
        conteo_actual = Counter(estrategias.values())
        for strat_key in ['C', 'D', 'H', 'K']: fraccion = conteo_actual.get(strat_key, 0) / num_nodos if num_nodos
> 0 else 0; historia_fracciones[strat_key].append(fraccion)
    print("Simulación completada.")
    cm_strat_str_file = "".join(sorted(cm_strategies))
    if cm_placement_type == 'Nodos centrales': cm_placement_label_file = "Cen"
    elif cm_placement_type == 'Nodos perifericos': cm_placement_label_file = "Per"
    elif cm_placement_type == 'Nodos aleatorios': cm_placement_label_file = "Ale"
    else: cm_placement_label_file = cm_placement_type[:3].lower()
    cm_info_str = f"_CM{cm_percentage}"
    cm_title_str = f"Masa crítica: {cm_percentage}\\% - {cm_placement_type} ({','.join(cm_strategies)})"
    return historia_fracciones, red, estrategias, network_type, network_params_str_out, cm_info_str, cm_title_str
# --- Funciones de Graficado  ---
def graficar_evolucion_corrupcion(historia_fracciones, titulo, network_type_str, homophily_str, cm_title_str,
filename):
    plt.style.use(['science', 'ieee']); plt.figure(figsize=(2.5, 2.5))
    colores = {'C': '#DEDEEE', 'D': '#9B9ACA', 'H': '#7976B8', 'K': '#201F3D'}
    estilos = {'C': '-', 'D': '--', 'H': ':', 'K': '-.'}; nombres_legenda = {'C': '(C) C. Coop', 'D': '(D) C. no
Coop', 'H': '(H) P. Honrado', 'K': '(K) P. Corrupto'}
    rondas_totales = len(historia_fracciones['C']); eje_x = range(rondas_totales)
    for strat in ['C', 'D', 'H', 'K']: plt.plot(eje_x, historia_fracciones[strat], label=nombres_legenda[strat],
color=colores[strat], linestyle=estilos[strat], linewidth=1)
    plt.title(f"Evolución de {titulo}\n{network_type_str} - Homofilia: {int(float(homophily_str) * 100)}\\%
\n{cm_title_str}")
    plt.ylim(0, 1.0); plt.xlim(0, rondas_totales -1 if rondas_totales > 1 else 1); plt.legend(fontsize=7,
loc='best'); plt.grid(True, linestyle='--', alpha=0.6)
    os.makedirs(os.path.dirname(filename), exist_ok=True); plt.savefig(filename, format="svg", dpi=300,
bbox_inches='tight'); print(f"Gráfica de evolución guardada como '{filename}'"); plt.close()
```

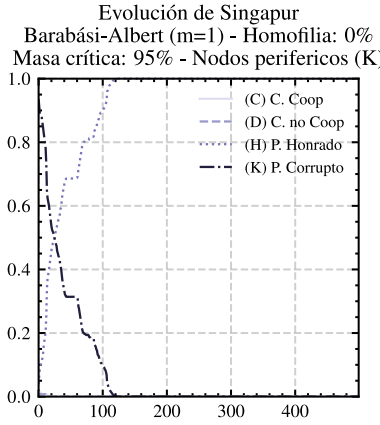
```
def graficar_red_final(red, estrategias_finales, titulo, network_type, network_params_str, homophily_str,
cm_title_str, filename):
    if not isinstance(red, nx.Graph) or not red.nodes(): return None
    plt.style.use(['science', 'ieee']); fig_size = (3.5, 3.5) if network_type == 'Completa' else (2.5, 2.5);
plt.figure(figsize=fig_size); n = red.number_of_nodes()
    color_map = {'C': '#DEDEEE', 'D': '#9B9ACA', 'H': '#7976B8', 'K': '#201F3D', 'Unknown': '#808080'}
    node_colors = [color_map.get(estrategias_finales.get(nodo, 'Unknown'), '#808080') for nodo in red.nodes()]
    print(f"  Calculando layout para red {network_type} ({n} nodos)..."); start_time = time.time()
    try: pos = nx.spring_layout(red, seed=42, k=0.6/np.sqrt(n) if n>0 else 0.1, iterations=50 if
network_type=='Barabási-Albert' else 30)
    except Exception: pos = nx.circular_layout(red)
    print(f"  Layout calculado en {time.time() - start_time:.2f} seg.")
    print(f"  Dibujando red final {network_type}..."); node_size = 10 if network_type == 'Completa' else 10;
nx.draw_networkx_nodes(red, pos, node_color=node_colors, node_size=node_size, linewidths=0.1, edgecolors='face')
    edge_alpha = 0.03 if network_type == 'Completa' else 0.5; edge_width = 0.03 if network_type == 'Completa' else
0.3; nx.draw_networkx_edges(red, pos, edge_color="#A7A7A7", alpha=edge_alpha, width=edge_width)
    title_net_part = f"{network_type}{network_params_str}" if network_type=='Barabási-Albert' else network_type;
plt.title(f"Red de {titulo}\n{title_net_part} - Homofilia: {int(float(homophily_str) * 100)}\\%
\n{cm_title_str}"); plt.axis('off')
    os.makedirs(os.path.dirname(filename), exist_ok=True); save_format = "png" if network_type == 'Completa' else
"svg"; save_dpi = 150 if network_type == 'Completa' else 300; plt.savefig(filename, format=save_format,
dpi=save_dpi, bbox_inches='tight'); print(f"  Gráfica de red final guardada como '{filename}' (Formato:
{save_format})"); plt.close()
    return pos

def graficar_subgrafo_k(red_original, estrategias_finales, pos_original, titulo, network_type, network_params_str,
homophily_str, cm_title_str, filename):
    nodos_k = [nodo for nodo, strat in estrategias_finales.items() if strat == 'K']
    if not nodos_k: print(f"  No nodos 'K' p/ subgrafo en {titulo}."); return None
    print(f"  Creando subgrafo 'K' ({len(nodos_k)} nodos)..."); subgrafo = red_original.subgraph(nodos_k)
    pos_k = {nodo: pos_original[nodo] for nodo in nodos_k if nodo in pos_original}
    if len(pos_k) != len(nodos_k): print(" Adv: No pos para todos K."); nodos_a_dibujar = list(pos_k.keys());
subgrafo = red_original.subgraph(nodos_a_dibujar);
    else: nodos_a_dibujar = nodos_k
    if not nodos_a_dibujar: return None
    plt.style.use(['science', 'ieee']); plt.figure(figsize=(2.5, 2.5))
    nx.draw_networkx_nodes(subgrafo, pos_k, nodelist=nodos_a_dibujar, node_color='#201F3D', node_size=10,
linewidths=0.1, edgecolors='face')
    nx.draw_networkx_edges(subgrafo, pos_k, edge_color="#201F3D", alpha=0.6, width=0.4)
    title_net_part = f"{network_type}{network_params_str}" if network_type=='Barabási-Albert' else network_type;
plt.title(f"Red de nodos corruptos (K) de {titulo}\n{title_net_part} - Homofilia: {int(float(homophily_str) *
100)}\\% \n{cm_title_str}"); plt.axis('off')
    os.makedirs(os.path.dirname(filename), exist_ok=True); plt.savefig(filename, format="svg", dpi=300,
bbox_inches='tight'); print(f"  Gráfica de subgrafo K guardada como '{filename}'"); plt.close()
    return filename

# --- Función Principal Modificada ---
def simular_y_graficar_pais(
    nombre_pais, params_juego, num_nodos,
    network_type, network_params,
    cm_size, cm_percentage, # <--- Nuevo parámetro de porcentaje
    cm_placement_type, cm_strategies, non_cm_strategies,
    K, homophily_strength, rondas, seed,
    output_dir="Resultados_Corrupcion",
    graficar_redes=True, graficar_subgrafo_k_flag=True
    ):
    cm_strats_str_label = "".join(sorted(cm_strategies))
    if cm_placement_type == 'Nodos centrales': cm_placement_label = "Cen"
    elif cm_placement_type == 'Nodos perifericos': cm_placement_label = "Per"
    elif cm_placement_type == 'Nodos aleatorios': cm_placement_label = "Ale"
    else: cm_placement_label = cm_placement_type[:3].lower()
    print(f"\n--- Sim: {nombre_pais} / {network_type}{network_params} / CM:{cm_placement_label}({cm_percentage}%
{cm_strats_str_label}) / H={homophily_strength} ---")
    historia, red_final, estrategias_finales, net_type_used, net_params_str_out, cm_info_str, cm_title_str =
ejecutar_simulacion_corrupcion(
        num_nodos=num_nodos, network_type=network_type, network_params=network_params,
        cm_size=cm_size, cm_percentage=cm_percentage,
        cm_placement_type=cm_placement_type,
        cm_strategies=cm_strategies, non_cm_strategies=non_cm_strategies,
        params_juego=params_juego, K=K, homophily_strength=homophily_strength,
        rondas=rondas, seed=seed
    )
    homophily_file_str = f"_h{homophily_strength:.2f}".replace('.', '')
    homophily_title_str = f"{homophily_strength:.2f}"
    nombre_base = f"{nombre_pais.lower().replace(' ',
'_')}_{net_type_used}{net_params_str_out}{cm_info_str}{homophily_file_str}"
    network_title_str = f"{net_type_used}{net_params_str_out.replace('_', ' ')}"
    path_grafica_evolucion = os.path.join(output_dir, f"evolucion_{nombre_base}.svg")
```

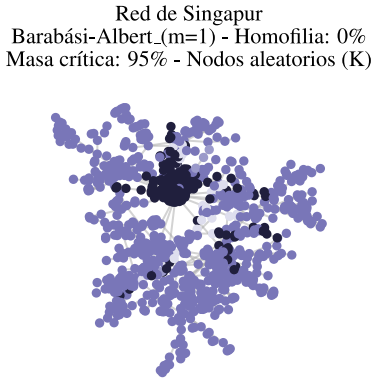
```
    graficar_evolucion_corrupcion(historia, nombre_pais, network_title_str, homophily_title_str, cm_title_str,
path_grafica_evolucion)
    pos_calculado = None; path_grafica_red_final = None
    if graficar_redes:
        format_red = 'png' if net_type_used == 'Completa' else 'svg'
        path_grafica_red_final = os.path.join(output_dir, f"red_final_{nombre_base}.{format_red}")
        pos_calculado = graficar_red_final(red_final, estrategias_finales, nombre_pais, net_type_used,
net_params_str_out, homophily_title_str, cm_title_str, path_grafica_red_final)
    path_grafica_subgrafo_k = None
    if graficar_subgrafo_k_flag:
        if pos_calculado is None and graficar_redes:
             print("  Calculando pos solo para subgrafo K...")
             try: pos_calculado = nx.spring_layout(red_final, seed=seed, k=0.6/np.sqrt(num_nodos) if num_nodos>0
else 0.1, iterations=50 if net_type_used=='Barabási-Albert' else 30)
             except Exception: pos_calculado = None
        if pos_calculado:
            path_grafica_subgrafo_k = os.path.join(output_dir, f"subgrafo_k_{nombre_base}.svg")
            graficar_subgrafo_k(red_final, estrategias_finales, pos_calculado, nombre_pais, net_type_used,
net_params_str_out, homophily_title_str, cm_title_str, path_grafica_subgrafo_k)
        else: print("  No pos, omitiendo subgrafo K.")
    print(f"--- Simulación completada ---")
    return path_grafica_evolucion, path_grafica_red_final, path_grafica_subgrafo_k


# --- Ejemplo de Uso  ---
if __name__ == "__main__":
    # --- Parámetros de Simulación y Control ---
    NODOS = 1000
    RUIDO_K = 0.0
    RONDAS = 500
    SEED_GLOBAL = 42
    OUTPUT_DIR_GLOBAL = f'N{NODOS}_I{RONDAS}'
    GRAFICAR_REDES_FINALES = True
    GRAFICAR_SUBGRAFO_K = True

    # --- Selección de Escenarios a Ejecutar ---
    PAISES_A_SIMULAR = ['Singapur']  # Puedes ajustar esta lista para simular solo algunos países ['México',
'Dinamarca', 'Singapur']
    config_paises = {
        'Dinamarca': PARAMS_DINAMARCA,
        'Singapur': PARAMS_SINGAPUR,
        'México': PARAMS_MEXICO
    }

    # ****  Lista de porcentajes de masa crítica a probar ****
    CM_PORCENTAJES_A_PROBAR = [
        #5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60, 65, 70, 75, 80, 85, 90,
        95
        #100
        ] # En porcentajes (15 para 15%)

    niveles_homofilia = [
        0.0,
        #0.5,
        #0.8,
        #0.9,
        #1.0
        ] # Niveles de homofilia a probar [0.0, 0.1, 0.2, 0.3, 0.4, 0.5] etc.
    config_redes = [
        #{'network_type': 'Completa', 'network_params': {}},
        {'network_type': 'Barabási-Albert', 'network_params': {'m': 1}}
    ]
    config_cm_placement = ['Nodos centrales', 'Nodos perifericos', 'Nodos aleatorios'] #['Nodos centrales', 'Nodos
perifericos', 'Nodos aleatorios'] # Puedes ajustar esto según tus necesidades
    config_cm_strategies = [
        #{'label': 'Equilibrio', 'cm_strats': ['C', 'H', 'D', 'K'], 'non_cm_strats': ['C', 'H', 'D', 'K']},
        #{'label': 'Coop', 'cm_strats': ['C', 'H'], 'non_cm_strats': ['D', 'K']},
        #{'label': 'NoCoop', 'cm_strats': ['D', 'K'], 'non_cm_strats': ['C', 'H']},
        {'label': 'SoloK', 'cm_strats': ['K'], 'non_cm_strats': ['C', 'D', 'H']}
        #{'label': 'HvsK', 'cm_strats': ['H'], 'non_cm_strats': [ 'K']},
        #{'label': 'CHvsK', 'cm_strats': ['H', 'C'], 'non_cm_strats': [ 'K']}
    ]
```

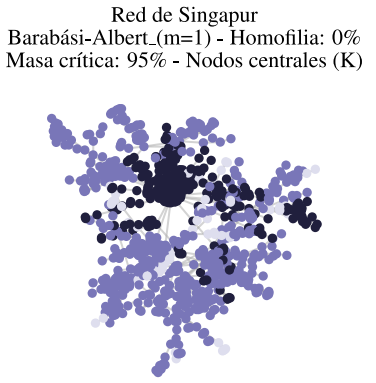
```
    # --- Lógica de Ejecución ---
    paths_graficas = {}
    paises_filtrados = {pais: config_paises[pais] for pais in PAISES_A_SIMULAR if pais in config_paises}

    # Añadir nuevo bucle para porcentajes de CM
    for cm_percentage in CM_PORCENTAJES_A_PROBAR:
        cm_size = round((cm_percentage / 100) * NODOS)
        if cm_size == 0 and NODOS > 0:
            print(f"Omitiendo CM %={cm_percentage} porque resulta en 0 nodos de masa crítica.")
            continue

        print(f"\n=================== CM PORCENTAJE = {cm_percentage}% (Tamaño={cm_size}) ====================")
        for pais, params_juego_pais in paises_filtrados.items():
            print(f"\n=================== PAÍS: {pais.upper()} ====================")
            for cm_strat_config in config_cm_strategies:
                cm_label = cm_strat_config['label']
                cm_strats_list = cm_strat_config['cm_strats']
                non_cm_strats_list = cm_strat_config['non_cm_strats']
                print(f"\n========== MASA CRÍTICA = {cm_label} ({','.join(cm_strats_list)}) ==========")
                for homophily_level in niveles_homofilia:
                    print(f"\n===== HOMOFILIA = {homophily_level:.2f} =====")
                    for sim_config_red in config_redes:
                        net_type = sim_config_red['network_type']
                        net_params = sim_config_red['network_params']
                        print(f"\n===== RED: {net_type} {net_params} =====")
                        for cm_place_type in config_cm_placement:
                            print(f"\n===== CM Placement: {cm_place_type} =====")

                            if cm_place_type == 'Nodos centrales': cm_place_label_dir = "Cen"
                            elif cm_place_type == 'Nodos perifericos': cm_place_label_dir = "Per"
                            elif cm_place_type == 'Nodos aleatorios': cm_place_label_dir = "Ale"
                            else: cm_place_label_dir = cm_place_type[:3].lower()

                            net_params_str_dir = "".join([f'_{k}{v}' for k,v in net_params.items()]) if net_params
else ""
                            homophily_str_dir = f"_h{homophily_level:.2f}".replace('.', '')
                            cm_strat_str_dir_label = f"_CM{cm_label}"
                            cm_percent_str_dir = f"_P{cm_percentage}" # Para el nombre del directorio

                            # Estructura de directorios: Pais / CM_Tipo / CM_Porcentaje / Red_Tipo+Params /
Placement+Homofilia
                            output_subdir_base = os.path.join(OUTPUT_DIR_GLOBAL, pais, f"CM_{cm_label}",
f"P{cm_percentage}")
                            output_subdir = os.path.join(output_subdir_base,
f"Red_{net_type}{net_params_str_dir}", f"{cm_place_label_dir}{homophily_str_dir}")

                            clave_base =
f"{pais}_{net_type}{net_params_str_dir}_{cm_place_label_dir}_{cm_strat_str_dir_label}{cm_percent_str_dir}{homophil
y_str_dir}"

                            path_evol, path_red, path_sub_k = simular_y_graficar_pais(
                                 pais, params_juego_pais, NODOS, net_type, net_params,
                                 cm_size, cm_percentage, # <-- Pasar ambos, tamaño y %
                                 cm_place_type,
                                 cm_strats_list, non_cm_strats_list,
                                 RUIDO_K, homophily_level, RONDAS, SEED_GLOBAL, output_subdir,
                                 graficar_redes=GRAFICAR_REDES_FINALES,
                                 graficar_subgrafo_k_flag=GRAFICAR_SUBGRAFO_K
                             )
                            paths_graficas[f"{clave_base}_evol"] = path_evol
                            if path_red: paths_graficas[f"{clave_base}_red"] = path_red
                            if path_sub_k: paths_graficas[f"{clave_base}_subgrafo_k"] = path_sub_k

    print(f"\nTodas las simulaciones completadas. Resultados en '{OUTPUT_DIR_GLOBAL}'.")
```

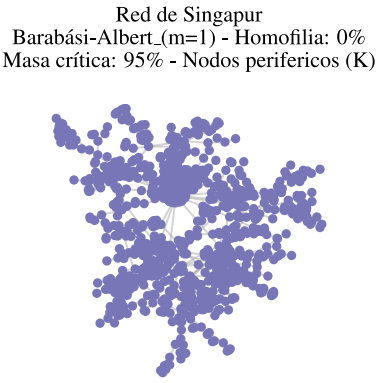

Evolución de Singapur
Barabási-Albert (m=1) - Homofilia: 0%
Masa crítica: 95% - Nodos aleatorios (K)
(C) C. Coop
(D) C. no Coop
(H) P. Honrado
(K) P. Corrupto

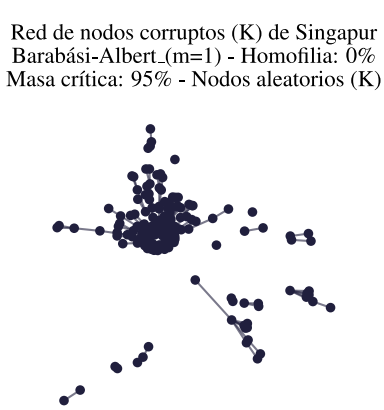

Evolución de Singapur
Barabási-Albert (m=1) - Homofilia: 0%
Masa crítica: 95% - Nodos centrales (K)
(C) C. Coop
(D) C. no Coop
(H) P. Honrado
(K) P. Corrupto

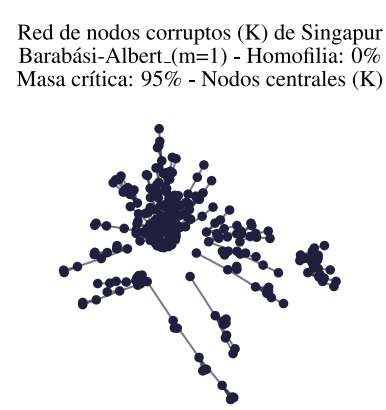

Evolución de Singapur
Barabási-Albert (m=1) - Homofilia: 0%
Masa crítica: 95% - Nodos perifericos (K)
(C) C. Coop
(D) C. no Coop
(H) P. Honrado
(K) P. Corrupto


Red de Singapur
Barabási-Albert_(m=1) - Homofilia: 0%
Masa crítica: 95% - Nodos aleatorios (K)


Red de Singapur
Barabási-Albert_(m=1) - Homofilia: 0%
Masa crítica: 95% - Nodos centrales (K)


Red de Singapur
Barabási-Albert_(m=1) - Homofilia: 0%
Masa crítica: 95% - Nodos perifericos (K)


Red de nodos corruptos (K) de Singapur
Barabási-Albert_(m=1) - Homofilia: 0%
Masa crítica: 95% - Nodos aleatorios (K)


Red de nodos corruptos (K) de Singapur
Barabási-Albert_(m=1) - Homofilia: 0%
Masa crítica: 95% - Nodos centrales (K)

C
D
C
(3,3)
(0,5)
D
(5,0)
(1,1)
A
B

Anexos

Recursos recomendados

Se comparte contenido para adentrarse en los temas de la teoría de redes, la teoría de juegos y la corrupción.

Libros

Granados, O., & José R., N.-C. (2021). *Corruption Networks.* Springer Nature.

Presenta diferentes artículos donde se ha usado la teoría de redes y sistemas complejos para la detección, control y predicción de la corrupción.

Páginas web

Los enlaces y nombres pueden cambiar con el tiempo.

***Network Science* de Albert-László Barabási.** Excelente página web interactiva donde explica los conceptos básicos de la teoría de redes.

https://networksciencebook.com/

Software especializado

NetworkX. Librería de Python enfocada en la creación, manipulación y estudio de la estructura, dinámica y funciones de redes complejas.

https://networkx.org/

NDlib - Network Diffusion Library. Librería de Python que permite describir, simular y estudiar procesos de difusión en redes complejas.

https://ndlib.readthedocs.io/

En memoria de **María del Carmen Morales**
y de todas las madres buscadoras

Asesinada junto a uno de sus hijos la noche del 23 de abril en el municipio de Tlajomulco de Zúñiga, Jalisco. Integrante del colectivo Guerreros Buscadores quienes revelaron lo ocurrido en el rancho Izaguirre en Teuchitlán, Jalisco. Donde integrantes del Cártel Jalisco Nueva Generación (CJNG), entrenaban, asesinaban y desaparecían personas.

C
D
C
(3,3)
(0,5)
D
(5,0)
(1,1)
A
B