UNIVERSIDAD POLITÉCNICA DE MADRID

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA
AGRONÓMICA, ALIMENTARIA Y DE BIOSISTEMAS

POLITÉCNICA

Generalización de procesos de consenso en redes complejas

Tesis doctoral

Miguel Rebollo Pedruelo

Doctor en Informática
Máster en Física de Sistemas Complejos

MADRID, 2019

GRUPO DE SISTEMAS COMPLEJOS

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA
AGRONÓMICA, ALIMENTARIA Y DE BIOSISTEMAS

Generalización de procesos de consenso en redes complejas

Tesis doctoral

Miguel Rebollo Pedruelo

Doctor en Informática

Máster en Física de Sistemas Complejos

Dirigida por:

Rosa María Benito Zafrilla

Doctora en CC. QUímicas

MADRID, 2019

Resumen

Los procesos de consenso en redes complejas permiten alcanzar acuerdos de forma distribuida teniendo en cuenta únicamente la información de los propios nodos y de los vecinos directos. Su comportamiento es bien conocido y las condiciones de convergencia en redes estáticas, donde no varía el número de nodos. Los únicos cambios permitidos se pueden producir en los enlaces (*switching topology*) o retardos en las comunicaciones para intercambiar los valores (*time-shifting*). Se asume que todos los nodos tienen el mismo comportamiento y siguen el protocolo establecido.

El objetivo de esta tesis es generalizar el modelo de consenso para adaptarlo a problemas en los que la red cambia durante el proceso. Las aportaciones principales son (i) la adaptación a redes dinámicas, en las que varían diversos factores del proceso, (ii) la inclusión de redes con signo que modelan relaciones positivas o negativas entre los nodos de la red, (iii) la extensión a n variables dependientes y a procesos de optimización multiobjetivo distribuidos, y (iv) la consideración de la presencia de nodos que no siguen el algoritmo y alteran el resultado del consenso. Con todo ello, se propone un modelo de consenso general que integra todos los aspectos anteriores.

Cada aportación se ha aplicado a problemas reales con el propósito de mostrar su funcionamiento, además de poner de manifiesto su utilidad en escenarios de interés en la actualidad.

Abstract

Consensus processes in complex networks allow agreements to be reached in a distributed way, taking into account only the information of the nodes and their direct neighbors. The behavior of this system and the conditions of convergence are well known for static networks, where the number of nodes does not vary. The only allowed changes can occur in the links (switching topology). Delays in communications to exchange values (time-shifting) are also considered. It is assumed that all nodes have the same behavior and follow the established protocol.

The objective of this thesis is to generalize the consensus model to adapt it to problems in which the network changes during the process. The main contributions are (i) the adaptation to dynamic networks, in which diverse factors vary during the process; (ii) the inclusion of signed networks that model positive or negative relations between the nodes of the network; (iii) the extension to n dependent variables and to distributed multiobjective optimization processes; and (iv) the consideration of the presence of nodes that do not follow the algorithm and alter the result of the consensus. With all this, a general consensus model is proposed that integrates all the preceding aspects.

Each contribution has been applied to real problems with the purpose of showing how they work, as well as demonstrating their usefulness in scenarios of current interest.

Tabla de contenidos

Resumen	I
Tabla de contenidos	V
Índice de figuras	VII
Índice de tablas	IX
1 Introducción	1
1.1. Motivación	1
1.2. Objetivos	2
1.3. Metodología de trabajo	3
1.4. Organización	3
2 Antecedentes	7
2.1. Modelo de consenso	7
2.2. Trabajos relacionados	10
3 Consenso en redes dinámicas	15
3.1. Introducción	15
3.2. Cambios estructurales	16
3.3. Cambio en los valores iniciales	20
3.4. Cambios en los pesos	21
3.5. Cambio en el factor de aprendizaje	24
3.6. Caso de estudio: Distribución de energía eléctrica	28
4 Consenso en redes con signo	41
4.1. Introducción	41
4.2. Consenso en redes con signo	42
4.3. Generalización a grafos m -partidos	44

4.4.	Coherencia como medida de la calidad de la división en comunidades	46
4.5.	Caso de estudio: detección de comunidades	48
5	Consenso multidimensional	59
5.1.	Introducción	59
5.2.	Consenso con variables multidimensionales	60
5.3.	Función de utilidad	63
5.4.	Combinación de consenso y gradiente	64
5.5.	Caso de estudio: Procesos analíticos jerárquicos	70
6	Consenso con nodos maliciosos	83
6.1.	Introducción	83
6.2.	Detección de desviaciones en el proceso de consenso	84
6.3.	Corrección de las desviaciones	85
6.4.	Casos de estudio	87
7	Modelo general de consenso	103
7.1.	Introducción	103
7.2.	Consenso en redes dirigidas	104
7.3.	Consenso acotado	106
7.4.	Modelo general de consenso	109
8	Conclusiones	115
	Referencias	121

Índice de figuras

2.1. Consenso en una red de de 4 nodos	8
3.1. Inserción y borrado de nodos	18
3.2. Cambio en los valores iniciales $x(0)$	21
3.3. Cambio en los pesos	23
3.4. Ajuste del factor de aprendizaje ε	25
3.5. Mapa de la red eléctrica balear	29
3.6. Patrón de la demanda eléctrica balear en un día	30
3.7. Distribución de grado de las estaciones	31
3.8. Cálculo del tamaño de la red	34
3.9. Adaptación a la demanda energética	37
3.10. Evolución del error relativo	38
3.11. Error ante eventos inesperados	39
3.12. Evolución del error tras un fallo	40
4.1. [Consenso en red balanceada con signo	43
4.2. Consenso en red desbalanceada	44
4.3. Grupos del Parlamento esloveno	49
4.4. Detección de comunidades por análisis espectral	49
4.5. Comunidades entre tribus de Nueva Guinea	50
4.6. NMI variando la mezcla entre comunidades	52
4.7. NMI variando el número de comunidades	54
4.8. Comunidades en guerras de edición en la Wikipedia	56
5.1. Grafo multicapa	61
5.2. Proceso de consenso con funciones de utilidad	64
5.3. Proceso de consenso y gradiente con funciones de utilidad	66
5.4. Consenso en una red con varios máximos	67
5.5. Consenso en una red con varios máximos	68
5.6. Entropía de la utilidad global	69

5.7. Ejemplo de jerarquía de criterios para un AHP	71
5.8. AHP mediante consenso para solución única	73
5.9. Evolución de los valores de cada criterio para cada uno de los participantes.	74
5.10. AHP con solución subóptima	75
5.11. Consenso para AHP con rotura de enlaces	76
5.12. AHP con 4 criterios. Comparación de consenso y media geométrica	81
6.1. Introducción de una desviación en el consenso	84
6.2. Corrección de las desviaciones	87
6.3. Votación distribuida por consenso	89
6.4. Introducción de desviaciones en proceso de votación	90
6.5. Corrección de las desviaciones en votación mediante consenso	91
6.6. Alteración de una imagen mediante ejemplos adversarios	91
6.7. Efecto en el consenso de la alteración de la imagen	92
6.8. Corrección de una imagen modificada mediante ejemplos adversarios	93
6.9. Problema de los generales bizantinos	95
6.10. Consenso para el problema de los generales bizantinos	96
6.11. Árbol de Merkle para almacenar los bloques de DLT	97
6.12. Proceso de consenso sobre un DLT.	99
6.13. Corrección de desviaciones en función del número de nodos maliciosos	100
6.14. Distribución de las correcciones completas	102
7.1. Grafo dirigido de cinco nodos y matriz de adyacencia asociada. Nótese que la matriz A no es una matriz simétrica	105
7.2. Consenso erróneo en redes dirigidas	105
7.3. Consenso en redes dirigidas	107
7.4. Configuración de un problema de consenso acotado	108
7.5. Procesos de consenso acotado	109

Índice de tablas

3.1. Características de la red eléctrica balear	31
3.2. Clasificación de las poblaciones por su centralidad	32
3.3. Diferencias en la velocidad de convergencia	36
4.1. Coherencia para todas las posibles particiones de V	47
4.2. Tiempo de ejecución de los diferentes algoritmos y configuraciones	53
4.3. Tiempo medio de ejecución de una iteración de consenso	55
5.1. Matriz de prioridades locales	71
5.2. Prioridades finales para los candidatos	72
5.3. Optimalidad de Pareto	78
5.4. Homogeneidad	79
5.5. Prioridades mediante consenso sobre juicios	80
6.1. Configuración de $x_i(0)$ para votaciones	88
6.2. Precisión de la clasificación de imágenes por distintos algoritmos .	94
6.3. Desviación media residual tras la corrección.	101
7.1. Ampliaciones del modelo de consenso	110

Capítulo 1

Introducción

1.1. Motivación

Hoy en día las redes son un elemento importante a la hora de intercambiar información. Las personas tomamos decisiones en función de las opiniones de nuestros contactos; existen redes de sensores que son capaces de medir magnitudes del mundo real y comunicarse entre ellas para intercambiar los datos leídos. También existen comunidades de robots que se comportan como una colectividad (enjambres o bandadas) para resolver problemas. Incluso podemos encontrarnos con sistemas de vehículos autoguiados que son capaces de adaptar su velocidad a la situación del tráfico. Los sistemas así modelados suelen ser sistemas complejos formados por un gran número de individuos que no hace factible la toma de decisiones centralizada debido al gran número de interacciones que serían necesarias. Además, este tipo de sistemas necesitan ser robustos ante fallos, de manera que sigan funcionando aunque algunos componentes fallen o la red cambie durante el proceso de toma de decisiones.

La descentralización puede suponer un problema si para poder operar los nodos de la red necesitan conocer toda la información. En esos casos, normalmente se habla de sistemas distribuidos, donde la información vital para el sistema está replicada en distintas zonas y cada parte de la red accede a la fuente de información más próxima, que se sincroniza con las demás para tratar de garantizar que los datos se encuentran actualizados, minimizando las posibles inconsistencias que puedan darse. En este tipo de sistemas, las fuentes de información suelen ser un cuello de botella y un punto débil que afecta al rendimiento global del sistema si alguno de estos recursos deja estar disponible por un fallo o incluso por un ataque deliberado. Además, si el entorno es muy dinámico y la topología de la red cambia frecuentemente, la sobrecarga del

sistema simplemente para propagar los cambios puede resultar excesiva.

Los sistemas descentralizados suelen ser una solución más flexible y robusta, aunque no es válida para cualquier tipo de problemas. Para que una solución descentralizada funcione, debe haber algún método que garantice su corrección trabajando con lo que se denomina **racionalidad acotada**: cada nodo de la red sólo dispone de su propia información y de la que le puedan proporcionar sus vecinos inmediatos. En general, cada nodo de la red tiene una visión parcial de la misma y no conoce su estructura, su topología o su tamaño. Este tipo de sistemas suele ser más robusto puesto que no hay ninguna entidad que haga las funciones de supervisor. El fallo de un nodo tiene efecto dentro de un alcance limitado, que dependerá en muchas ocasiones de la topología de la red.

Uno de los usos de este tipo de redes es determinar el valor global de una magnitud a partir de los valores individuales de esa misma magnitud en cada uno de los nodos de la red. Por ejemplo, podemos tener una red de sensores en la que cada nodo mide la temperatura de su entorno y se la intercambian para determinar cuál es la temperatura media en el área cubierta por la red. A este tipo de procesos se les conoce en general como procesos de difusión de información o también de propagación de creencias. El algoritmo de consenso, que se estudia en el presente trabajo, es uno de los métodos existentes para abordar este tipo de problemas.

Los procesos de consenso existentes asumen que la red permanece invariable durante el proceso. Solo se tienen en cuenta posibles retardos en el envío de la información entre los nodos de la red (*time delay*) y reconexiones de los enlaces en la red (*switching topology*), pero manteniendo el conjunto de nodos invariable.

Sin embargo, esta situación no es muy realista, especialmente cuando se trata de entornos dinámicos o de problemas en los que se necesita monitorizar de forma continua el valor de ciertas variables. En esos casos, es necesario tener en cuenta todos los cambios que se pueden dar en la red: cambios en los valores iniciales de los nodos, cambios en su peso en la red (si se trata de redes pesadas) o cambios en la topología, permitiendo la entrada de nodos nuevos o la desaparición de nodos durante el proceso.

1.2. Objetivos

El presente trabajo de tesis se titula «Generalización de procesos de consenso en redes complejas» y tiene por **objetivo**

Generalizar el modelo de consenso para que un grupo de entidades autónomas sea capaz de llegar a acuerdos de forma general cuando

se produzcan cambios en la red durante el proceso, intervenga más de un criterio y existan nodos que no siguen el protocolo.

Este objetivo se alcanzará a través de los siguientes **objetivos específicos**:

1. plantear el modelo general en redes dinámicas
2. extender el modelo a redes con signo
3. extender el modelo a redes dirigidas
4. ampliar el dominio a más de una variable, dependientes
5. detectar la presencia de nodos tramposos
6. proponer una versión continua de los modelos anteriores
7. integrar todas las propuestas en un solo modelo de consenso
8. demostrar la corrección aplicándolo a problemas reales

1.3. Metodología de trabajo

El presente trabajo trata de estudiar empíricamente el uso de procesos de consenso como un método de sincronización en redes válido para entornos dinámicos, donde la estructura de la red o los datos varían durante el proceso de la toma de decisiones.

Se extenderá el protocolo básico de consenso para combinarlo con otros métodos que permitan calcular valores agregados y para incluir factores de corrección que permitan utilizar el protocolo en redes dinámicas, manteniendo los criterios de convergencia de la red. Con estas correcciones se busca también poder eliminar cualquier tipo de datos global, de manera que se pueda utilizar en entornos completamente descentralizados.

Se construirá un modelo en MATLAB que permita la simulación de los procesos propuestos en el presente trabajo. Se aplicará a diversos problemas reales que sean representativos para la extensión que se proponga en cada apartado

1.4. Organización

En contenido del trabajo está estructurado de la siguiente manera

- Capítulo 2. Contiene una revisión de los trabajos relacionados con los procesos de sincronización y de propagación en redes mediante procesos de consenso, identificando las líneas de investigación actuales en este tema con el fin de justificar la aportación del presente trabajo de tesis en dicho campo.
- Capítulo 3. Una primera aproximación es la extensión del modelo a redes dinámicas, que van variando con el tiempo. El modelo básico de consenso solo contempla la posibilidad de redirigir enlaces, pero los nodos de la red y los valores iniciales deben permanecer constantes. En este capítulo se relajan estas condiciones de manera que se permiten cambios en los nodos participantes en el proceso, variaciones en los valores iniciales o en los pesos de la red. Su funcionamiento se muestra con un ejemplo de gestión de la demanda energética en la red eléctrica balear.
- Capítulo 4. Presenta una extensión a un caso particular de redes pesadas, cuando los pesos pueden tomar valores negativos (redes con signo). Se trata de una extensión particularmente interesante, ya que los trabajos actuales solo tiene en consideración redes balanceadas. En este capítulo se aplica también a aquellos casos en los que los pesos negativos no definen una partición perfecta en la red. Como ejemplo de aplicación, se propone un mecanismo para la detección de comunidades en redes con signo.
- Capítulo 5. Los procesos anteriores solo contemplan consensos sobre variables independientes. En este capítulo se estudia la extensión a n dimensiones. El proceso de consenso se combina con un proceso de descenso por gradiente con el que se consigue optimizar una función objetivo, definida a partir de las dependencias entre las variables de consenso. Para demostrar su funcionamiento, se aplica a los llamados procesos analíticos jerárquicos (AHP –*analytical hierarchical process*–), en los que un grupo de expertos tienen que llegar a un acuerdo acerca de los criterios para evaluar una serie de elementos y después tomar una decisión a partir de los pesos establecidos para dichos criterios.
- Capítulo 6. Los procesos de consenso requieren que todos los participantes sigan el mismo protocolo. En este capítulo se relaja esta condición y se permite que haya nodos que, de forma deliberada o accidental, introduzcan desviaciones en una o varias iteraciones del proceso de consenso. Se propone un mecanismo para detectar cuándo se producen estas desviaciones y para corregirlas, recuperando el consenso original. Para su evaluación se emplean una serie de escenarios de especial actualidad: un

sistema de votación electrónica, la detección de errores en sistemas de visión artificial por ejemplos adversarios y la solución del problema de los generales bizantinos, que es la base de los protocolos usados hoy en día en criptomonedas y cadenas de bloques (*blockchain*).

- Capítulo 7. En este último capítulo se unifican todas las aportaciones anteriores en un modelo único de consenso, estudiando cuáles de ellas son incompatibles entre sí. Se propone también una versión continua de todas las variantes.
- Capítulo 8. Presenta las conclusiones del trabajo y las líneas de investigación futura que quedan abiertas.

Capítulo 2

Antecedentes

2.1. Modelo de consenso

Dado un conjunto de entidades autónomas, el problema de consenso consiste en alcanzar un acuerdo en torno a una opinión, que se representará mediante un valor escalar, partiendo del conjunto inicial de opiniones de las entidades. Habitualmente, la comunicación está restringida de manera que no es posible el intercambio de opiniones entre cualquier par de entidades. Habitualmente, esta restricción viene dada por una red de comunicación que limita la comunicación a aquellos pares entre los que hay un canal explícito. Existen otros factores que limitan la comunicación, como el tener que compartir un espacio común, estar en cierto área de alcance o tener conocimiento de la existencia de la otra entidad (como ocurre por ejemplo en las relaciones de amistad existentes en las redes sociales).

Sea un grafo $G = (V, E)$ donde $V = \{1, 2, \dots, n\}$ es un conjunto de vértices o nodos y $E = \{e_{ij} | i, j \in V\}$ es un conjunto de enlaces donde e_{ij} , también denotado por (i, j) , indica que existe un enlace desde el nodo i hacia el nodo j . Dada una ordenación arbitraria de V , se define la matriz de adyacencia de G como

$$A_G(i, j) = \begin{cases} 1 & \text{si } e_{ij} \in E \\ 0 & \text{en otro caso} \end{cases}$$

Por simplicidad, denotaremos por $a_{ij} = A_G(i, j)$ a los elementos de la matriz de adyacencia.

Cada nodo i de la red tiene un valor $x_i \in \mathbb{R}$. La alcanza un consenso si $x_i = x_j \forall i, j \in V$. Si N_i denota los vecinos del nodo i , $N_i = \{j \in V | e_{ij} \in E\}$, un

Figura 2.1. Consenso en una red de de cuatro nodos. Los valores iniciales son $x = (0, 20, 40, 60, 8)^T$ y la red converge a $\bar{x} = 0,5$

proceso de consenso para alcanzar un acuerdo en el estado de n nodos puede representarse como un sistema lineal de orden n

$$x_i(t+1) = x_i(t) + \varepsilon \sum_{j \in N_i} x_j(t) - x_j(t) \quad (2.1)$$

La dinámica colectiva de los n nodos puede escribirse como

$$x(t+1) = Px(t)P = I_n - \varepsilon L \quad (2.2)$$

donde $x = (x_1, \dots, x_n)^T$ es el vector que incluye todos los valores de los nodos de la red, I_n es la matriz identidad de orden n y L es la matriz Laplaciana de la matriz de adyacencia del grafo.

$$l_{ij} = \begin{cases} -1 & j \in N_i \\ \deg(i) & i = j \end{cases}$$

La Figura 2.1 muestra un proceso de consenso en una red simple con cuatro nodos.

El proceso de consenso se origina en el trabajo de De Groot [DeG74], donde propone un método que permite a una comunidad de individuos llegar a un acuerdo sobre el valor de un determinado parámetro θ a partir de las distribuciones de probabilidad subjetivas y privadas de cada uno de los participantes. Se trataba de un proceso iterativo en el cual cada participante intercambiaba su valor con un subconjunto de compañeros y en cada iteración modificaba

su propio valor. Para que el proceso convergiera a un valor final común, debe cumplirse que la matriz que regula el intercambio sea estocástica.

Olfati y Murray [OSM04, OSFM07] han sido quienes dieron el impulso definitivo a este método, aplicándolo a redes de agentes que deben coordinarse y habitualmente relacionados con problemas de control, como redes de sensores, control de vehículos autoguiados o control de formaciones. La formulación propuesta es semejante a la utilizada a los problemas de sincronización en osciladores acoplados [MP04, BLM⁺06], especialmente con los osciladores de Kuramoto, aproximaciones finalmente han podido unificarse [LDCH10].

Aunque los consensos se pueden utilizar para calcular el valor de distintas funciones (por ejemplo, un valor medio, el máximo o el mínimo), no sirve para cualquier cálculo. En concreto, no puede realizar cálculos de valores agregados, como puede ser la suma de todos los valores de la red. Todos ellos se consideran métodos desestructurados (no se asume ninguna estructura de control que rijan el proceso) y pueden clasificarse en tres grandes grupos

1. algoritmos de inundación, en los que en cada paso se propaga la información a toda la red desde un punto
2. algoritmos basados en paseos aleatorios (*random walks*), que introducen un elemento aleatorio a la hora de seleccionar los nodos con los que se intercambia información;
3. algoritmos de *gossip* o de propagación de rumores, en los que cada nodo intercambia parte de la información que tiene con un subconjunto de nodos seleccionados.

Los trabajos de Jesus et al. [JBA11] y de Rajagopalan y Varshney [RV06] estudian con detenimiento y caracterizan las técnicas que se emplean para la agregación de datos distribuida.

La convergencia de los procesos de consenso es una de las características que más ampliamente se ha estudiado. Inicialmente se pensaba que no se podía garantizar [FLM86] y fueron los propios Olfati y Murray los que realizaron un primer estudio de la convergencia bajo ciertas condiciones. Posteriormente, se han conseguido definir las propiedades necesarias y suficientes para que se puedan alcanzar consensos en la red [TSJ08]. El presente trabajo, en el que se emplean redes no dirigidas, cumple los criterios de convergencia del modelo de Olfati y Murray. Una de las líneas de más interés relacionada con la convergencia es estudiar métodos que permitan acelerarla [OT09]. En cuanto a dominios en los que se ha aplicado, se puede destacar su uso en redes sociales [LSTT11], la aplicación a sistemas de grandes dimensiones [ASJ13], su

utilización como técnica de *clustering* [LF12], la resolución de los problemas derivados de comunicación con errores [FCFZ09] o la extensión a grafos no dirigidos arbitrarios [CI12].

2.2. Trabajos relacionados

A continuación se detallan los trabajos relacionados con las principales líneas de investigación que se han seguido durante el trabajo de tesis.

2.2.1. Redes con signo

La mayoría de las redes sociales presentan una estructura comunitaria, por lo que entender sus propiedades se ha convertido en una prioridad en las redes complejas [New03]. En consecuencia, la detección de comunidades y la agrupación de gráficos son un tema de investigación de larga data en las redes [For10]. Este problema consiste en encontrar un grupo de nodos densamente conectados con pocas conexiones a nodos que no pertenecen al grupo [HBG⁺14]. Los nodos de una misma comunidad suelen compartir atributos, y la identificación de estas comunidades es relevante para descubrir cómo se propaga la información a través de la red o por qué algunos comportamientos surgen de la interacción de los participantes.

Se puede utilizar un amplio conjunto de técnicas para identificar las comunidades existentes en una red [For10, NdC11, Sch07]. Las principales preocupaciones sobre los algoritmos actuales utilizados para este propósito en redes complejas se refieren a su rendimiento [DNF14, GB13]. Las redes sociales pueden contener millones de usuarios y su análisis para encontrar la distribución óptima de las comunidades puede resultar inalcanzable. Además, algunos métodos requieren conocer a priori el número de comunidades existentes [CLSW10, DNF14].

Además, la mayoría de ellos necesitan un grafo etiquetado con vínculos positivos, pero hay casos en los que la naturaleza de las relaciones implica un sentido negativo, como por ejemplo gustos similares -desprecio, amor - odio o confianza - desconfianza[AP13]. Las redes firmadas son la representación natural de este tipo de relaciones, permitiendo que los enlaces tengan un peso negativo asociado. Los enfoques existentes para identificar comunidades en este tipo de redes se basan en adaptaciones de diferentes métodos, tales como el modelo de Pott, [TB09, BTU12], la modularidad, [GoJA09], el análisis espectral[AMI12, WYW⁺12], la descomposición del kernel y los *k-means* [CWD12], la agrupación de correlaciones[CBV12] y los algoritmos genéticos[AP13]. Otros métodos que no utilizan gráficos firmados,

pero que pueden ser fácilmente ampliados incluyen enfoques de teoría de juegos [CLSW10]. Estos métodos utilizan la red completa para identificar a las comunidades, y no pueden ser aplicados en ambientes donde los participantes tienen una visión limitada de la red. En estos entornos se utilizan métodos basados en la propagación de la información. Por ejemplo, Yang et al. [YCL07] utilizan paseos aleatorios para cubrir la red, recuperando información de los nodos visitados.

Una de las contribuciones del presente trabajo de tesis es un nuevo algoritmo de detección basado en un mecanismo de consenso. Es un proceso iterativo en el que cada participante intercambia su valor con un subconjunto de socios. Todos los participantes modifican su valor en cada iteración. El trabajo de De Groot [DeG74] puede considerarse seminal para procesos de consenso, mientras que Olfati-Saber y Murray [OSM04, OSFM07] dieron el impulso final a este método con una condición para la convergencia hacia un valor final común. Los trabajos que aplican procesos de consenso en redes se centran en el análisis de la velocidad de convergencia [OT09], su uso en redes sociales [LSTT11] y su aplicación en sistemas a gran escala [ASJ13]. Existe un trabajo que utiliza el consenso como técnica de agrupamiento para detectar comunidades [LF12], pero los autores definen el consenso como un tipo particular de red y no como un proceso dinámico.

Cha et al. [CCC⁺18] generalizó la detección de comunidades en redes con atributos, donde los nodos y enlaces tienen atributos que representan características de ambos. Los gráficos firmados son casos especiales de redes atribuidas. Los autores utilizan medidas de centralidad para identificar a las comunidades. Sin utilizar este nombre, el trabajo de Guo y Zhang [GZ16] también propone un método válido para múltiples tipos de redes utilizando una función de disimilitud y propagando la afinidad para detectar los centros de cada comunidad.

Una línea de investigación reciente es la combinación de técnicas de inteligencia artificial y redes complejas. Che et al. [CWWL16] y Zhu et al. [ZML18] utilizan métodos evolutivos (algoritmos genéticos), que han tenido éxito en la inteligencia artificial para problemas de *clustering*. La detección comunitaria es un caso particular de agrupamiento y, por lo tanto, los resultados pueden aplicarse. También se pueden utilizar otras técnicas de agrupación en racimos [WZLJ16]. Cai et al. [CMGT16] han estudiado diferentes técnicas de computación evolutiva aplicadas a la detección de comunidades en redes.

Otra técnica bien conocida de la inteligencia artificial es la optimización de enjambres, en la que un grupo de entidades autónomas, siguiendo reglas simples, hacen emerger un comportamiento de grupo más complejo, que se utiliza en problemas de optimización. Li et al. [LHL16] lo usan para detectar comuni-

dades en redes con signo. La principal diferencia con nuestra propuesta es que el trabajo de Li varía la topología de la red en cada corrida para acercarse a la división óptima de la comunidad. Por lo tanto, el algoritmo necesita un conocimiento completo y acceso a la red, y no es válido para entornos distribuidos reales.

2.2.2. Consenso multivariable y aplicación a problemas de optimización

La principal aportación en esta línea es la aplicación al proceso jerárquico analítico de grupo (GAHP), que es la generalización de AHP para un grupo de tomadores de decisiones. Cada agente tiene su matriz de juicio y sus prioridades finales y negocia para lograr una solución única y acordada. Existen principalmente dos métodos para agregar preferencias individuales: la agregación de juicios individuales (AIJ) y la agregación de preferencias individuales (AIP) [RG94]. La primera agrega la matriz de decisión completa, mientras que la segunda sólo agrega las prioridades. Ossadnik et al. [OSK16] hacen un análisis comparativo de los diferentes enfoques de GAHP. Otros métodos utilizados con éxito para resolver GAHP incluyen el trabajo de Srdjevic [SS13], o Dong y Saarly [DS14], en los que el proceso se centra en asegurar la consistencia de la solución. Huang et al. [HCLL13] combina juicios y preferencias, lo que lleva a una decisión de grupo más satisfactoria. También se han utilizado otras técnicas, como el recocido simulado, tratando de resolver el GAHP cuando existen diferencias significativas entre las opciones [BSSZ16]. Finalmente, la palabra *consensus* aparece en otros trabajos relacionados con la toma de decisiones en grupo y GAHP, pero con un significado general, y no se refiere al método utilizado en este documento [DS14, WCFHV17].

2.2.3. Detección y corrección de fallos

Finalmente, en esta tesis se propone una extensión de un proceso de consenso, que pertenece a la familia de los algoritmos del cotilleo, para mantener un libro mayor distribuido. Trabajos anteriores [RBLG18] muestran su aplicabilidad en áreas como el aprendizaje automático, con ejemplos adversarios [GSS14], o los mecanismos de votación distribuida, un área de interés real en los sistemas distribuidos [CFL09]. Hoy en día, el voto electrónico combinado con tecnologías distribuidas como blockchain han puesto al día para su aplicación a procesos de toma de decisiones en grupos [MH17]. El interés en la cadena de bloques crece porque el fortalecimiento de la protección de la privacidad [Ksh17][TDSHJ16], y se puede aplicar a diversas áreas como sis-

temas de rastreo de alimentos [Tia16], contratos administrativos [WDM15] o gestión de la red eléctrica [PCA⁺18]. Su uso en Internet de los objetos requiere ciertas adaptaciones [JCdAdAR18][MDB17]. La seguridad es una de las preocupaciones más importantes de esta tecnología. Los principales protocolos implementan diferentes defensas contra los ataques bizantinos, que pueden comprometer la estabilidad del sistema [TLP17][JCdAdAR18].

Sin embargo, seguir con precisión el algoritmo es obligatorio para todos los nodos. Un comportamiento bien conocido es el de seguir al líder [OPPU16], en el que uno de los nodos permanece inalterado o cambia arbitrariamente. Entonces, la red completa converge al valor de este nodo. Otro caso aparece cuando sólo un nodo falsifica el valor transmitido. La consecuencia de ello es que la red converge a un valor diferente del promedio. Por lo tanto, es demasiado fácil para un nodo alterar el resultado final del consenso.

Capítulo 3

Consenso en redes dinámicas

3.1. Introducción

El modelo de consenso asume que la red no varía durante el proceso. El número de nodos inicial permanece constante y los únicos cambios que se tienen en cuenta son cambios en la topología. Este comportamiento se conoce como *switching topology*. En estos casos, varía la velocidad de convergencia, pero no su valor final. Tanto los pesos de la red como los valores iniciales permanecen sin cambios.

Sin embargo, existen escenarios en los que resulta adecuado relajar estas restricciones, permitiendo

- Escenario 1: cambios en el conjunto de nodos: se debe permitir la entrada de nuevos nodos durante el proceso de consenso o su eliminación. En cualquier caso, el valor de convergencia debe considerar solo el de los participantes en el momento de finalización del consenso.
- Escenario 2: cambios en los valores iniciales: es el caso de la monitorización continua de una variable de control. Es el caso de una red de sensores que monitoriza un parámetro como la temperatura. Los nodos de la red comparten las lecturas y calculan la temperatura media en un momento dado, pero esa lectura varía con las fluctuaciones de la temperatura.
- Escenario 3: cambios en los pesos de los nodos: cuando la red incluya pesos en los nodos, estos pueden variar para reflejar cambios en la importancia del nodo, lo que afecta a la media ponderada.
- Escenario 4: cálculo del factor de aprendizaje ε : eliminando así la necesidad de un parámetro global, comúnmente conocido por la red. Aplicable

a cualquier ejemplo de consenso en un entorno distribuido real, ya que el valor de consenso

A continuación se desarrollan cada uno de estos casos. En todos ellos, para la obtención de la solución, se tendrá en cuenta la propiedad de **conservación de la suma** en el proceso de consenso, según la cual

$$\sum_i x_i(0) = \sum_i x_i(t) \forall t \geq 0 \quad (3.1)$$

3.2. Cambios estructurales

Cuando se producen cambios estructurales en la red, deben reflejarse en la matriz de adyacencia A y, en consecuencia, las matrices P y L (o sus variantes pesadas P_W y L_W respectivamente) deben recalcularse. Esta operación tiene un coste computacional del orden de $O(n^2)$ para cada una de ellas. Incluso en el caso de que estemos usando matrices dispersas, tendrá un coste de $O(m)$ siendo m el número total de enlaces de la red.

Estos cambios afectarán únicamente al modelo matricial. En el caso de las fórmulas distribuidas no es necesaria ninguna operación extra más allá de añadir los enlaces con los nodos nuevos en la red o eliminar los enlaces con los nodos que han desaparecido.

El principal problema de los cambios estructurales es que pueden provocar cambios en el factor de aprendizaje ε que hagan que la matriz de Perron no sea primitiva y, por lo tanto, que el algoritmo de consenso no converja a un valor único en la red. En principio, y puesto que el valor de ε es una cota inferior, asumiremos que los cambios estructurales son tales que el valor de ε sigue siendo válido. La Sección 3.5 modifica el modelo para permitir ajustarse a valores cambiantes de ε manteniendo la convergencia del algoritmo.

3.2.1. Inserción de nodos

El cambio más simple que se puede producir en la red es que, durante el proceso de consenso, se añadan nuevos nodos a la red o se retire alguno de los participantes.

Insertar nodos nuevos en la red es trivial. Basta con empezara intercambiar valores con sus vecinos y aplicar la Ecuación 2.1 (o su equivalente en el caso de redes pesadas). Debido a la conservación de la suma, el proceso convergerá a la media de los valores iniciales $x_i(0)$, incluyendo el nuevo valor. No hay

límite en el número de nodos que se pueden añadir a la red simultáneamente (véase Figura 3.1(a))

3.2.2. Borrado de nodos

Sin embargo, cuando un nodo abandona la red, no puede eliminarse sin más. Después de haber ejecutado algunos ciclos, el valor actual de $x_i(t) = x_i(0) + \Delta x_i$ incorpora un incremento (positivo o negativo) hacia la media global de la red $\bar{x}$. Si eliminamos el nodo i sin más consideración, estaremos eliminando una cantidad diferente a su valor inicial $x_i(0)$ y en consecuencia la suma no se conservaría.

$$\sum_i x_i(t) = \sum_i [x_i(0) + \Delta x_i] = \sum_i x_i(0) + \sum_i \Delta x_i = \sum_i x_i(t) + \sum_i \Delta x_i$$

por la conservación de la suma, luego en todo momento debe cumplirse que $\sum_i \Delta x_i = 0$. Pero si eliminamos el valor $x_i(t)$ de la red entonces

$$\sum_{j \neq i} x_j(t) = \sum_i x_i(t) - x_i(t) = \sum_i x_i(0) - x_i(0) - \Delta x_i = \sum_{j \neq i} x_j(0) - \Delta x_i$$

con lo no se conservaría la suma. Para corregirlo, debemos añadir a la red la cantidad Δx_i .

La forma más inmediata de conseguirlo es sumar la cantidad Δx_i a un nodo cualquiera de la red, pero eso introduciría un elemento aleatorio en el modelo. Podemos evitarlo dividiendo Δx_i a partes iguales entre todos los vecinos del nodo i . En ambos casos la suma se conserva, por lo que las dos soluciones son equivalentes. De cara a una implementación en un sistema real, es más eficiente pasar el excedente completo Δx_i a un solo vecino, pues se intercambian menos mensajes. Por lo contrario, repartir el excedente entre todos los vecinos simplifica la expresión matricial. Usaremos esta última para que el modelo dinámico sea homogéneo y considere de la misma forma las correcciones que es necesario realizar para tener en cuenta la dinámica de la red (véase Figura 3.1(b)).

Sea $x_i(t)$ el valor actual del nodo que va a abandonar la red y $d_i = \sum_j a_{ij}$ su grado. Denotamos por $N_i = \{j \mid a_{ij} \neq 0\}$ el conjunto de vecinos del nodo i .

$$x_i(t) = x_i(0) + [x_i(t) - x_i(0)] = x_i(0) + \sum_{j \in N_i} \frac{x_i(t) - x_i(0)}{d_i} \quad (3.2)$$

Definimos $T_d = [td_i]^{N \times 1}$ un vector columna con

$$T_d = \frac{x_i(t) - x_i(0)}{d_i} A_i \quad (3.3)$$

Figura 3.1. Ejemplo de inserción y borrado de nodos. A partir de una configuración inicial como la que aparece en la Figura 2.1, los nodos llegan o abandonan la red con una frecuencia que sigue una distribución exponencial con $\beta = 1/50$, es decir, cada 50 iteraciones de media (a) llegada de nodos en las iteraciones $t = 40$ y $t = 60$ con valores $x_5(0) = 1$ y $x_6(0) = 0,1$. (b) borrado de los nodos 2 y 4 en las iteraciones $t = 49$ y $t = 67$ respectivamente.

como el vector de transformación que habrá que aplicar cuando un nodo abandone la red. El proceso de consenso quedará ampliado con la corrección por el borrado de nodos haciendo

1. $x(t+1) = P x(t)$
2. $x = x + T_d$

3.2.3. Recálculo de las matrices

Si bien en la versión descentralizada del algoritmo la modificación de la red afecta únicamente a las estructuras de datos del propio nodo y de sus vecinos, en el caso de la versión matricial requiere un recálculo de todas las matrices. Añadir o eliminar un nodo varia la matriz de adyacencia A , por lo que será necesario volver a recalcular la laplaciana L y esto conlleva a su vez el recálculo de la matriz de Perron P que se emplea en cada iteración. Si la dinámica produce cambios frecuentes en la red, este proceso puede resultar muy costoso.

Por otro lado, en aquellos casos en los que varíe el factor de aprendizaje ε , la matriz de Perron P también debe recalcularse para ajustarla al nuevo valor

Sea $A^{N \times N}$ la matriz de adyacencia de una red con N nodos. Un cambio en la estructura de la red significa añadir un vector de unos y ceros $A_{N+1} = [a]^{N \times 1}$ donde $a_i = 1$ si hay una conexión entre el nuevo $N+1$ y un nodo existente en la red i . La nueva matriz ampliada será

$$\hat{A} = \left(\begin{array}{c|c} A & A_{N+1} \\ \hline A_{N+1}^T & 0 \end{array} \right)$$

En la matriz laplaciana ampliada $\hat{L}$ aparecerá la nueva fila y columna A_{N+1} cambiada de signo y además se incrementará el valor de $l_{ii} \forall i \in N_{N+1}$, siendo las matrices resultantes

$$\hat{L} = \left(\begin{array}{c|c} L + D_{A_{N+1}} & -A_{N+1} \\ \hline -A_{N+1}^T & \sum_i a_{N+1,i} \end{array} \right)$$

donde $D_{A_{N+1}}$ denota la matriz diagonal formada por los elementos de A_{N+1} . La matriz de Perron se modifica en el mismo sentido y queda como resultado

$$\hat{P} = \left(\begin{array}{c|c} P - \varepsilon D_{A_{N+1}} & \varepsilon A_{N+1} \\ \hline \varepsilon A_{N+1}^T & 1 - \varepsilon \sum_i a_{N+1,i} \end{array} \right)$$

De esta manera, podemos modificar solo aquellos elementos de las matrices asociadas al proceso de consenso que se han visto afectados por el cambio, resultando un proceso más eficiente que si hubiera que recalcular las matrices completas en cada paso.

De forma análoga, la eliminación de un nodo de la red debe reflejarse en las tres matrices. la modificación de la matriz de adyacencia A es trivial, pues basta con eliminar la fila y la columna correspondientes al nodo eliminado. Si representamos mediante $A_{i,j}^-$ la matriz A eliminado la fila i y la columna j , la matriz de adyacencia resultante será:

$$\hat{A} = A_{i,i}^-$$

La laplaciana que le corresponde puede obtenerse mediante la expresión

$$\hat{L} = (L - D_{A_i})_{i,i}^-$$

Y de forma similar, la matriz de Perron resultante se obtiene haciendo

$$\hat{P} = (P + \varepsilon D_{A_i})_{i,i}^-$$

3.3. Cambio en los valores iniciales

La ejecución reiterada Ecuación 2.1 en cada uno de los nodos de la red hace que el valor final $x_i(t)$ converja al valor medio de los valores iniciales $\bar{x} = \frac{\sum_i x_i(0)}{N}$. El proceso de consenso se caracteriza por la conservación de la suma de los valores iniciales durante todo el proceso, cumpliéndose que

$$s = \sum_i x_i(0) = \sum_i x_i(t) \quad \forall t \quad (3.4)$$

Y dado que el tamaño de la red N no varía, la media global $\bar{x}$ también se conserva. Haciendo uso de esta propiedad, si un nodo cambia su valor inicial $x_i(0)$ por otro valor $z_i(0)$, la nueva suma conservada debe ser

$$\hat{s} = z_i(0) + \sum_{j \neq i} x_j(0)$$

si sumamos y restamos $x_i(0)$ a la expresión y reorganizamos los términos tenemos que

$$\hat{s} = z_i(0) - x_i(0) + \sum_j x_j(0) = z_i(0) - x_i(0) + s \quad (3.5)$$

es decir, la diferencia en la suma global únicamente depende del cambio que se ha producido en el nodo i y es el incremento producido en el valor inicial. Si el proceso de consenso ya ha ejecutado algunos pasos, el valor acumulado en $x_i(t)$ procedente de los otros nodos sería distinto al valor $z_i(t)$ que tendría en nodo i en la iteración t si el valor inicial hubiera sido $z_i(0)$. Pero puesto que la suma se conserva, el valor final de consenso será el mismo que si el valor del nudo hubiera sido $z_i(0)$ en el inicio.

Dado que la variación que introduce en la suma global en cambio en el valor inicial de un nodo puede corregirse de forma local. Cuando se produce un cambio en los valores iniciales de la red, basta con que cada nodo actualice su valor con el incremento que le corresponde y continúe el proceso de consenso normalmente. El cambio se propagará por toda la red, que convergerá a la media de los valores actualizados.

Sea $T_x = [tx_i]^{N \times 1}$ un vector columna donde

$$tx_i = \begin{cases} z_i(0) - x_i(0) & \text{si } x_i(0) \text{ cambia} \\ 0 & \text{en otro caso} \end{cases} \quad (3.6)$$

Podemos representar el modelo completo mediante la siguiente expresión para el algoritmo de consenso:

Figura 3.2. Cambio en los valores iniciales. En la iteración $t = 47$ los valores de los nodos cambian a $z(0) = \{\}$, con lo que la red converge al nuevo valor de consenso $\bar{z} = 0,517$

1. $x(t+1) = P x(t)$
2. $x = x + T_x$

3.4. Cambios en los pesos

Como se ha comentado en la Sección 2.1, la fórmula de consenso en redes pesadas queda definida por la Ecuación 3.7

$$x_i(t+1) = x_i(t) + \frac{\varepsilon}{w_i} \sum_{j \in N_i} [x_j(t) - x_i(t)] \quad (3.7)$$

que converge a la suma ponderada de los valores iniciales

$$\bar{x} = \frac{\sum_i w_i x_i}{\sum_i w_i}$$

Si el peso del nodo i cambia durante el proceso de consenso, w_i deja de ser constante y debería reemplazarse por $w_i(t)$, quedando la expresión

$$x_i(t+1) = x_i(t) + \frac{\varepsilon}{w_i(t)} \sum_{j \in N_i} [x_j(t) - x_i(t)] \quad (3.8)$$

En redes pesadas con pesos constantes se sigue manteniendo la conservación de la suma

$$s = \sum_i w_i x_i(0) = \sum_i w_i x_i(t) \quad \forall t \quad (3.9)$$

puesto que $\sum_i w_i$ es independiente del tiempo. Sin embargo, si permitimos que el peso de un nodo cambie durante el proceso, la suma pesada variará en función de las variaciones de los pesos. Asumamos que se produce el cambio en el peso en nodo i de la red, pasando de ser w_i a un nuevo peso v_i . El valor $x_i(t)$ puede expresarse como la suma de su valor inicial más la variación hacia el valor medio que ha sufrido en las t iteraciones anteriores:

$$x_i(t) = x_i(0) + [x_i(t) - x_i(0)] \quad (3.10)$$

por lo que su contribución a la suma pesada será

$$w_i x_i(t) = w_i x_i(0) + w_i [x_i(t) - x_i(0)] \quad (3.11)$$

es decir, que en el nodo i tenemos acumuladas $w_i [x_i(t) - x_i(0)]$ unidades de la suma ponderada total que se deben mantener en el nodo i . El nuevo valor $z_i(t)$ que debe tener el nodo i para que se conserve la nueva suma debe cumplir

$$v_i z_i(t) = v_i x_i(0) + w_i [x_i(t) - x_i(0)]$$

por lo que

$$z_i(t) = x_i(0) + \frac{w_i [x_i(t) - x_i(0)]}{v_i}$$

Si en lugar de expresar el nuevo valor a partir del valor inicial $x_i(0)$ queremos hacerlo a partir del último valor $x_i(t)$ basta con sustituirlo usando la Ecuación 3.6

$$\begin{aligned} z_i(t) &= x_i(t) - [x_i(t) + x_i(0)] + \frac{w_i [x_i(t) - x_i(0)]}{v_i} = \\ &= x_i(t) + [x_i(0) - x_i(t)] - \frac{w_i [x_i(0) - x_i(t)]}{v_i} = \\ &= x_i(t) + \frac{v_i}{v_i} [x_i(0) - x_i(t)] - \frac{w_i [x_i(0) - x_i(t)]}{v_i(t)} = \\ &= x_i(t) + \frac{v_i [x_i(0) - x_i(t)] - w_i [x_i(0) - x_i(t)]}{v_i} = \\ &= x_i(t) + \frac{(v_i - w_i) [x_i(0) - x_i(t)]}{v_i} \end{aligned} \quad (3.12)$$

Figura 3.3. Cambio en los pesos. Inicialmente, los pesos de la red son $w = \{0,5 \ 0,2 \ 0,3 \ 0,1\}$, siendo la media pesada $\bar{x} = y$ en la iteración $t = 23$ los pesos cambian a $w'(0) = \{0,15 \ 0,35 \ 0,27 \ 0,25\}$, con lo que la red converge al nuevo valor de consenso $\bar{x}' = 0,6478$

Tal y como refleja la Ecuación 3.8, el cambio del peso de un nodo puede compensarse de forma local en el propio nodo sin que sea necesario usar información de los vecinos. En las siguientes iteraciones, el cambio se va propagando por el resto de la red. Esto hace que sea posible que un número indeterminado de nodos puedan cambiar de peso de forma simultánea, incluso la red completa.

Se puede representar este cambio en la expresión matricial del algoritmo de consenso. Definimos $T_w = [tw_i]^{N \times 1}$ un vector columna donde cada tw_i está definido según la Ecuación 3.8

$$T_w = (v - w)(x^0 - x)v^{-1} = (I - WV^{-1})(X^0 - X) \quad (3.13)$$

siendo V el nuevo vector de pesos y X^0 el vector que contiene los valores iniciales. Podemos representar el modelo completo mediante la siguiente expresión para el algoritmo de consenso:

1. $x(t+1) = P x(t)$
2. $x = x + T_w$

3.5. Cambio en el factor de aprendizaje

El peso w_i de un nodo es una constante en la ecuación que rige el consenso en redes pesadas, lo mismo que ε . Sin embargo, ε es un dato global de la red que depende de su estructura (del grado de cada nodo) y, en el caso de una red pesada, de los pesos de los nodos.

$$\varepsilon < \min_i \frac{w_i}{d_i}$$

tomando $w_i = 1 \forall i$ en el caso de redes no pesadas.

Si se produce un cambio en cualquier de estos valores que haga que el valor de ε se vea modificado y este cambio no se realiza en todos los nodos, perderemos la garantía de la convergencia. Otro problema que aparece relacionado con este valor es que se debe conocer de antemano por toda la red, perdiendo la característica de racionalidad acotada, pues hay al menos un dato global que hay que transmitir o que asumimos que los nodos de la red conocen de antemano.

Supongamos que en el instante inicial $t = 0$ se desconoce el valor de ε y que cada nodo tiene un $\varepsilon_i = \frac{w_i}{d_i}$ propio. Podemos obtener el valor de ε haciendo un proceso de consenso sobre mínimos, donde

$$\varepsilon_i(t+1) = \min \left(\varepsilon_i(t), \min_{j \in N_i} \varepsilon_j(t) \right) \quad (3.14)$$

El proceso que recoge la Ecuación 3.12 consiste en que cada nodo intercambia con sus vecinos su valor de ε_i y se queda con el menor de todos ellos. Si la red está conectada, converge al valor mínimo de ε_i . De esta forma, realizando primero un consenso para determinar el valor de ε ya estamos en las condiciones adecuadas para que la Ecuación 2.1 converja.

De forma matricial, sea

$$E = \begin{pmatrix} \varepsilon_1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \varepsilon_n \end{pmatrix}$$

una matriz $N \times N$ en la que tenemos en la diagonal principal los valores iniciales $\varepsilon_i(0)$. La expresión matricial de este proceso es

$$E = \text{diag} \left(\min_j E(I + A) \right) \quad (3.15)$$

Figura 3.4. Ajuste del factor de aprendizaje ε sin que sea necesario conocerlo de antemano. La propia red lo calcula durante el proceso haciendo un consenso a mínimos. La red converge converge al valor medio de los valores iniciales.

donde $\min$ obtiene el valor mínimo por filas y diag es una matriz diagonal a partir de los elementos especificados (en este caso los mínimos en cada iteración). La aplicación iterativa de la Ecuación 3.13 converge a un valor único para ε .

Esta solución resuelve el problema para redes estáticas y nos permite construir un algoritmo descentralizado en dos fases: la primera calcula el valor de ε mediante un proceso iterativo sobre la Ecuación 3.13 y la segunda implementa el algoritmo de consenso normalmente usando como ε el valor calculado en la fase anterior.

El problema de esta primera solución es determinar la condición de parada para la primera fase en el caso descentralizado (Ecuación 3.12), ya que si no tenemos información global no podemos determinar cuándo el valor mínimo se ha propagado por toda la red y se puede comenzar la fase de consenso. Una segunda aproximación consiste en intercalar ambas fases, utilizando un valor $\varepsilon_i(t)$ que varía con el tiempo hasta que alcanza su valor definitivo. Como en los casos anteriores, al corregir el valor de $\varepsilon_i(t)$ habrá que compensar de alguna manera el error introducido por en la iteración anterior. Si en la Ecuación 3.7 dejamos fuera del sumatorio el término que varía con el tiempo, podemos ver que el cambio en el peso de un nodo y el cambio en el valor de ε tienen la misma forma.

$$\begin{aligned}
x_i(t+1) &= x_i(t) + \frac{1}{w_i} \sum_{j \in N_i} \varepsilon [x_j(t) - x_i(t)] \\
x_i(t+1) &= x_i(t) + \varepsilon \sum_{j \in N_i} \frac{1}{w_i(t)} [x_j(t) - x_i(t)]
\end{aligned}$$

Sea e_i el nuevo valor calculado para ε_i en la iteración actual. Puesto que el sumatorio en el proceso de consenso pesado está multiplicado por $1/w_i$ (véase Ecuación 3.7) y para corregir el cambio en el peso usamos su inverso, aplicando el mismo razonamiento que en la Ecuación 3.4 llegamos a la expresión

$$z(t) = x(t) + \frac{\left(\frac{1}{e_i} - \frac{1}{\varepsilon_i}\right) [x_i(0) - x_i(t)]}{\frac{1}{e_i}} \quad (3.16)$$

Para construir la expresión matricial, trataremos de simplificar la expresión. Definimos $T_e = [te]^{N \times 1}$ un vector columna donde

$$\begin{aligned}
te_i &= \frac{\left(\frac{1}{e_i} - \frac{1}{\varepsilon_i}\right) [x_i(0) - x_i(t)]}{\frac{1}{e_i}} = \\
&= e_i \frac{\varepsilon_i - e_i}{e_i \varepsilon_i} [x_i(0) - x_i(t)] = \\
&= \frac{\varepsilon_i - e_i}{\varepsilon_i} [x_i(0) - x_i(t)] = \\
&= \left(1 - \frac{e_i}{\varepsilon_i}\right) [x_i(0) - x_i(t)] \quad (3.17)
\end{aligned}$$

y de forma compacta

$$T_e = (I - D_{E'} D_E^{-1})(x^0 - x) \quad (3.18)$$

La adaptación del valor de ε hace que sea necesario recalcular la matriz de Perron P en ese paso. La matriz de adyacencia A y la laplaciana L no se ven modificadas (a no ser que el cambio que se haya producido en la red lo requiera). El cambio afectará a la fila y a la columna completa del nodo cuyo valor de ε haya cambiado.

$$\hat{P} = P + (E - E')L$$

Y el proceso de consenso completo queda de la siguiente forma¹

¹En este caso el paso de adaptación se ha realizado antes del paso de consenso ya que tiene que corregir la posible desviación producida por el cambio en ε .

1. $E' = \text{diag}(\min_j E(I + A))$
2. $x = x + T_e$
3. $P + (E - E')L$
4. $E = E'$
5. $x = P \cdot x$

En el caso de tener redes dinámicas, donde se producen cualquiera de los cambios que se han analizado en las secciones previas, la problemática es similar: nunca podremos estar seguros de estar usando el valor adecuado de ε debido a los cambios que se producen en el grado de los nodos o en su peso.

- Si los nodos pueden entrar y salir de la red, se modifica el grado de los nodos restantes. Eliminar nodos no es un problema porque si el grado disminuye $1/d_i$ aumenta y el valor de ε sigue estando por debajo de dicha cota.
- Si el nodo más conectado recibe nuevos vecinos (común en redes con conexión preferencial— *preferential attachment*—) su grado aumenta y en consecuencia la cota para ε disminuye. Si el valor actual de ε queda por encima de la cota ya no se puede garantizar la convergencia.
- La redirección de enlaces afecta al grado de los nodos de la red de la misma forma que lo hace el borrado de nodos cuando el nodo más conectado de la red está involucrado y gana enlaces.
- En el caso de redes pesadas, puesto que la cota de ε depende del peso de los nodos, también se debe tener en cuenta cuando varíen los pesos de la red. Si la variación de los pesos es monótona creciente (nunca disminuye) el valor mínimo de $\frac{w_i}{d_i}$ anterior seguirá siendo una cota inferior, por lo que no es necesario hacer ningún cambio. El problema vuelve a surgir cuando el peso de los nodos puede disminuir durante el proceso de consenso

Independientemente de como se haya conseguido el valor ε actual, cuando el nuevo valor e_i en alguno de los nodos está por debajo del valor previo hay que corregir el valor de $x(t)$ calculado con el valor ε_i usando la Ecuación 3.16

3.6. Caso de estudio: Distribución de energía eléctrica

El problema de la gestión de una red eléctrica ha ido creciendo en interés e importancia. Representa una parte significativa del PIB de un país y es un sector estratégico esencial para el resto de las actividades económicas. La energía eléctrica no puede almacenarse en grandes cantidades, por lo que la cantidad de energía requerida debe generarse cuando se demanda. ¿Cómo se puede distribuir esta demanda entre los componentes de la red para que se satisfaga? y, ¿cómo se pueden superar los fallos en cualquier subestación para que el resto de la red disponible pueda compensar las pérdidas? en esta sección se abordan estos problemas, con la dificultad añadida de resolverlos solo con información local.

La red eléctrica se ha modelado como una red compleja, donde cada subestación es un nodo de la red. Los nodos solo pueden comunicarse con los otros nodos correspondientes a las subestaciones a las que está directamente conectada. Se utiliza un proceso de consenso para ajustar la potencia suministrada a la demanda actual, mientras que un algoritmo de *gossip* calcula al mismo tiempo el número total actual de estaciones activas y la capacidad requerida de la red.

El presente caso de estudio muestra la aplicabilidad de las redes de consenso dinámicas al escenario anteriormente expuesto. Se ha construido un simulador que utiliza la configuración de la red eléctrica balear y la demanda real durante 15 días. Con esos datos, se ha simulado cómo la energía proporcionada por las subestaciones se ajusta para satisfacer la demanda utilizando el algoritmo de consenso. En segundo lugar, se estudia el efecto de un fallo en una subestación, comprobando como el resto de la red asume la carga para seguir atendiendo a la demanda energética de las islas.

la dinámica que se considera en el presente caso es :

- **cambios en el valor inicial** de los nodos, adaptándose en todo momento a la demanda requerida,
- **cambios estructurales** por la eliminación de subestaciones cuando se producen fallos o la inserción de estaciones nuevas cuando se recuperan
- se emplea una **red pesada**, donde los pesos son la capacidad relativa de las subestaciones. Estos pesos también cambian cuando se produce la caída de una subestación.

Figura 3.5. Mapa de la red eléctrica balear. El color de las líneas indica su capacidad

La red eléctrica de las Islas Baleares está compuesta por 57 subestaciones y 82 líneas de 30 kV a 220 kV (véase Figura 3.5). La red de transporte une las cuatro islas que forman el archipiélago: Mallorca, Menorca, Ibiza y Formentera. Además, una conexión con la península española compensa el saldo negativo de la demanda. Por ejemplo, durante 2013, la producción neta de las islas fue de 4 405 GWh, mientras que la demanda fue de 5 671 GWh.

La demanda no es uniforme a lo largo del día y aparece un patrón característico (Figura 3.6). Los datos con la demanda real pueden obtenerse en *Red Eléctrica de España*². Los datos relativos a la generación programada, la demanda real y la prevista se actualizan cada diez minutos. Esta información ha sido utilizada en este caso para validar la propuesta con datos de la demanda real de los usuarios en el período analizado.

²<http://www.ree.es>

Figura 3.6. Patrón de la demanda eléctrica balear en un día. (Fuente: Red Eléctrica Española. Recuperado de <http://www.ree.es>)

3.6.1. Caracterización de la red

Como otras redes artificiales, y a pesar de su pequeño tamaño, la red de energía balear presenta la mayoría de las características de las redes del mundo pequeño. Esta estructura proporciona a la red eficiencia en la difusión y robustez ante perturbaciones o daños. El grado medio indica que cada subestación está conectada de media a otras 2,8 subestaciones. El diámetro total de la red es de 14. Eso significa que necesitamos 14 pasos para transportar electricidad entre las dos subestaciones más lejanas. Sin embargo, la longitud media del trayecto es de sólo cuatro, lo que significa que, como promedio, sólo necesitamos cuatro pasos para transportar energía entre cualquier par de subestaciones. El coeficiente de agrupamiento o *clustering* mide el número de triángulos que hay en la red: si una subestación está conectada a otras dos subestaciones, el agrupamiento mide si estas subestaciones también están conectadas entre sí. La combinación de un agrupamiento alto y caminos cortos le da a una red la propiedad de un mundo pequeño.

Sin embargo, hay otra característica típica de muchas redes complejas: la distribución del grado. Mide cómo se distribuye el grado (número de vecinos) a lo largo de la red. Por lo general, aparece una distribución de leyes de potencia.

Tabla 3.1

Características de la red eléctrica balear

medida	valor
grado medio	2.877
longitud de camino medio	4.786
diámetro	14
<i>clustering</i>	0.337

Figura 3.7. Distribución de grado de las estaciones (en escala logarítmica). A diferencia de otras redes, en las que la distribución del grado sigue una ley de potencias, en este caso se ajusta a una distribución de Poisson con media $\mu = 2,877$ y varianza $\sigma^2 = 2,931$.

Esto significa que hay unos pocos nodos en la red con un grado excepcionalmente alto (*hubs*), mientras que la mayoría de los nodos tienen un grado bajo. Pero en el caso de la red de energía balear, se ha observado una distribución de probabilidad de Poisson (véase Figura 3.7). Por lo tanto, la red estudiada no es una red libre de escalas y la distribución de grados es similar a la distribución de los grafos aleatorios.

Finalmente, existen otras medidas relacionadas con la importancia de un nodo dentro de la red. Este concepto se conoce como centralidad. Hay cuatro medidas principales de centralidad: el grado, la cercanía (*closeness*), la intermediación (*betweenness*) y el valor propio. La centralidad de grado mide la importancia de un nodo por su grado (número de conexiones). La centralidad

de la cercanía determina la distancia a los otros nodos de la red: el nodo más central es el que tiene una distancia mínima a todo el resto de la red. La intermediación está relacionado con el número de trayectorias que pasan por un nodo, que actúa como puente entre muchos nodos de la red. Finalmente, la centralidad de valor propio es una medida de la influencia del nodo en la red, dando una puntuación alta a aquellos nodos conectados con otros nodos que a su vez tiene una puntuación alta. La Tabla 3.2 muestra los valores de estas medidas de centralidad para las diez primeras poblaciones en cada valor. Se ha añadido una quinta medida: el k -core, que crea grupos de nodos que están vinculados a otros nodos con grado k o superior.

En la red balear se identifica claramente un grupo de cinco subestaciones que obtienen sistemáticamente la alta puntuación en todas las medidas de centralidad: Son Reus, Llubí, Es Bessons, Valldurgent y Con Orlandis. Además, las subestaciones de Son Reus y Llubí son las más críticas, ya que son las subestaciones más centrales independientemente de la medida que se tome en consideración. Eso quiere decir que esas cinco subestaciones son las más críticas del sistema balear. Si observamos el mapa de la Figura 3.5, podemos darnos cuenta de que se trata de las estaciones conectadas a las líneas de mayor capacidad (de color verde).

Tabla 3.2

Clasificación de las poblaciones por su centralidad

grado		k -core		closeness		betweenness		valor propio	
pob.	valor	pob.	valor	pob.	valor	pob.	valor	pob.	valor
Son Reus	9	Son Reus	8	Son Reus	0.326	Son Reus	776	Son Reus	0.055
Llubí	7	Llubí	6	Llubí	0.318	Llubí	628	Llubí	0.043
Es Bessons	7	Es Bessons	6	Son Orlandis	0.308	Valldurgent	567	Es Bessons	0.043
Valldurgent	6	Son Orlandis	5	Es Bessons	0.289	Es Bessons	555	Son Orlandis	0.037
Son Orlandis	6	Valldurgent	5	Valldurgent	0.281	Son Orlandis	392	Valldurgent	0.036
Llucmajor	5	Llucmajor	4	Llucmajor	0.280	Santa Ponsa	344	Llucmajor	0.031
Coliseo	5	Coliseo	4	Marratxí	0.264	Llucmajor	307	Coliseo	0.030
Rafal	4	Cas Tresorer	3	Can Picafort	0.264	Torrent	301	Ciudadela	0.025
Torrent	4	Ciudadela	3	Coliseo	0.259	Calviá	243	Arenal	0.024
Ibiza	4	San Antonio	3	Ses Veles	0.256	Cala Millor	193	Inca	0.024

Nota. Importancia de las poblaciones en función de distintas medidas de centralidad: grado, k -core, closeness, betweenness y valor propio. Las cinco primeras se repiten en todas las medidas. Se corresponden con las centrales conectadas con líneas de 220 kV.

3.6.2. Proceso de consenso en la red eléctrica balear

Para coordinar la red, se ha diseñado un algoritmo descentralizado que realiza tres procesos de consenso independientes en paralelo.

1. un consenso para calcular el tamaño de la red
2. otro consenso para determinar la capacidad total de la red
3. un tercer y último consenso para obtener la demanda actual

Por lo tanto cada nodo intercambia una tupla formada por tres valores: (s_i, c_i, x_i) siendo s_i la variable para calcular el numero de nodos, c_i la variable para calcular la capacidad de la red y x_i la demanda actual.

El primer consenso aprovecha el comportamiento del algoritmo para contar el número de subestaciones activas. la formulación básica converge al valor medio de los valores iniciales. De modo que si el valor inicial de todos los nodos excepto de uno es cero

$$s_1(0) = 1, s_i(0) = 0 \forall i > 1 \quad (3.19)$$

la red converge a

$$\bar{s} = \frac{\sum_i s_i(0)}{n} = \frac{1}{n}, \quad n = \frac{1}{s_i(t)} \quad (3.20)$$

. Así que cada nodo puede calcular el tamaño de la red como el inverso del valor de consenso (véaseFigura 3.8).

El segundo proceso de consenso no hace nada especial. Para calcular la capacidad, cada nodo toma como valor inicial su capacidad $c_i(0)$. Cuando converge, el valor que se obtiene es la capacidad media de la red. Pero como con el primer proceso se ha obtenido el tamaño n , basta con multiplicar ambos valores para conocer la capacidad total de la red C .

$$C = \bar{c} \cdot n = c_i(t) \cdot n = \frac{c_i(t)}{s_i(t)} \quad (3.21)$$

Estos valores se actualizan constantemente. Si se produce un cambio en la red (un fallo en la red o un cambio en la capacidad global), los vecinos lo detectan automáticamente y lo propagan al resto de la red.

Esta información es utilizada por el algoritmo de consenso para ajustar el rendimiento de las subestaciones. Cada subestación tiene que proporcionar solamente la parte proporcional de la capacidad total de la red. Por lo tanto, el peso de la subestación i en el sistema se obtiene como

Figura 3.8. Cálculo del tamaño de una red con cuatro nodos. Los valores iniciales son $s(0) = \{1, 0, 0, 0\}$ y el resultado converge a $s_i(t) = 0,25$, luego $n = 1/s_i(t) = 1/0,25 = 4$.

$$w_i(t) = \frac{c_i(0)}{C} = \frac{c_i(0)s_i(t)}{c_i(t)} \quad (3.22)$$

Cuando todos los mensajes han sido recibidos de sus vecinos, los nodos de la red actualizan sus valores siguiendo la Ecuación 2.1. Tras una fase transitoria, $x_i(t)$ contiene la demanda total y, por tanto,

$$D_i(t) = w_i(t)x_i(t) = \frac{c_i(0)}{C}x_i(t) = \frac{c_i(0)s_i(t)x_i(t)}{c_i(t)} \quad (3.23)$$

es la parte de la demanda proporcionada por la subestación i .

Hay que subrayar que estos tres valores ($s_i(t)$, $c_i(t)$ y $x_i(t)$) se calculan al mismo tiempo, de forma que cualquier cambio en el número de nodos y en la capacidad de la red puede ser detectado y el sistema se adapta a ellos.

Se ha modelado una aplicación demo, donde cada una de estas subestaciones son agentes, y tenemos un agente adicional que es el *leader agent*. Este agente está indicando la demanda global de la red de energía. Hemos comprobado cómo el sistema multiagente que sigue nuestro método se acerca a esta demanda incluso cuando está cambiando (siguiendo los datos reales de la demanda existente en las Islas Baleares) como se puede ver en la Figura 3.6. Esta demo muestra cómo la red se adapta por sí misma a los cambios en la demanda y detecta y corrige los efectos de posibles fallos. La única condición para alcanzar la solución exacta es que la red debe permanecer conectada. Si

alguna parte de la red está aislada del resto, seguirá funcionando, pero los valores calculados mostrarán alguna desviación.

3.6.3. Adaptación a la demanda

La convergencia de la red está teóricamente asegurada por el algoritmo, por lo que no es relevante para nuestro propósito. Sin embargo, el mecanismo "seguir al líder" nos proporciona una forma elegante y eficiente de adaptar la salida del sistema a una demanda dinámica y cambiante en tiempo real.

Veamos la demanda eléctrica real de las Islas Baleares durante dos semanas. Un nodo ficticio modelará esta demanda, actualizando su valor cada 10 minutos (de modo que un día está representado por 144 datos). Este nodo está conectado a un subconjunto de todas las posibles subestaciones. El modelo es equivalente a tener m nodos que mantengan una parte de la demanda. Por ejemplo, cada nodo podría ser una población. No se ha optado por esta opción porque (i) no se dispone de información acerca de qué subestación proporciona servicio a qué población, y (ii) tampoco se dispone información de la demanda segregada por poblaciones.

Las subestaciones ejecutan cada algoritmo de consenso con la misma frecuencia que se actualizan los datos de la demanda. Las subestaciones conectadas con el nodo de demanda reciben el valor por directamente. Desde su punto de vista, el nodo de demanda es una subestación más, por lo que no es necesario implementar ningún comportamiento especial.

Para medir la eficiencia del método, se calcula un error

$$e = \frac{\sum_i w_i(t)x_i(t) - D_G}{D_G} \quad (3.24)$$

donde D_G es la demanda global y $C = \sum_i w_i(t)x_i(t)$ es la capacidad total que puede asumir la red. Una convergencia lenta produce un alto error en la estimación de la demanda. La tabla muestra los resultados obtenidos por dos implementaciones diferentes del protocolo de consenso con frecuencias de actualización diferentes. El consenso estándar se calcula utilizando $P = I - \varepsilon L$. Se puede usar una formulación alternativa con $P = D^{-1}A$. El algoritmo de consenso estándar necesita intercambiar información varias veces por ciclo para converger adecuadamente a la demanda real (cada diez segundos). Por otro lado, la formulación alternativa obtiene un rendimiento comparable manteniendo la información original y el tipo de cambio energético. Así que hemos optado por la segunda versión del algoritmo.

Una vez seleccionado el algoritmo adecuado, hemos estudiado cómo se adapta el algoritmo a los cambios en la demanda. La Figura 3.9 muestra el

Tabla 3.3

Diferencias en la velocidad de convergencia

modelo	T energia	T inform.	ratio inform./energia	media	desv. std
$P = I - \varepsilon L$	600	300	1:2	10.5705	80.1273
$P = I - \varepsilon L$	600	200	1:3	6.3020	57.9475
$P = I - \varepsilon L$	600	150	1:4	6.5643	53.2062
$P = I - \varepsilon L$	600	100	1:5	7.7443	52.9254
$P = I - \varepsilon L$	600	50	1:12	0.0031	0.0516
$P = I - \varepsilon L$	600	10	1:60	-0.5e-04	0.0088
$P = D^{-1}A$	600	600	1:1	1.6e-04	0.0121

Nota. Diferencias en la velocidad de convergencia usando diferentes ratios entre el número de intercambios de información y número de intercambios de energía entre subestaciones. Los periodos de intercambio (T) están medidos en segundos. El cálculo de la matriz de Perron como una matriz de caminantes aleatorios $P = D^{-1}A$ converge más rápido con un ratio de intercambio de información más bajo.

proceso de adaptación. Hay una fase transitoria al principio, debido a la falta de información hasta que se ha calculado el tamaño y la capacidad total de la red (véase Figura 3.9(a)). Después de 20 o 25 iteraciones, todas las subestaciones han comenzado a converger a la demanda. El zoom proporcionado en la Figura 3.9(b) nos muestra las diferencias en la velocidad de la convergencia a la demanda. Este hecho se debe a las diferencias en la topología de la red y a las medidas de centralidad explicadas.

La evolución del error se muestra en la Figura 3.10. La Figura 3.10(a) muestra el error asociado a la demanda real. Aparece una pauta regular como consecuencia de la periodicidad de la demanda. Los picos corresponden a los puntos con cambios más rápidos en la demanda. Durante este período, las redes deben transmitir la información y modificar rápidamente los valores de referencia. Sin embargo, en los máximos y mínimos locales los valores son bastante estables y el error tiende a cero.

Para comparar el comportamiento del algoritmo, se ha ejecutado con una demanda aleatoria. Para simularla, se introducen cambios aleatorios de 200 MWh en cada iteración (10 minutos). Podemos ver en la Figura 3.10(b) que el patrón ha desaparecido del error. Sin embargo, el rango del error es aproximadamente el mismo.

En ambos casos, la distribución del error se ajusta mejor a la distribución bimodal, como mezcla de dos poblaciones gaussianas. En el caso de la adapta-

Figura 3.9. Adaptación a la demanda energética. Las líneas representan la estimación de cada una de las estaciones sobre la demanda total requerida en el sistema eléctrico. Cada subestación proporciona el porcentaje de esta capacidad total que le corresponde según su peso en la red.

Figura 3.10. Evolución del error relativo. Asumimos que la información se intercambia al mismo ratio que la electricidad.

ción a los datos reales, se obtienen valores de $\mu_1 = -0,0098$ y $\sigma_1 = 0,0069$ para la media y la desviación estándar para la primera población; y $\mu_2 = 0,0027$ y $\sigma_2 = 0,0103$ para la segunda, con una proporción de mezcla de 0.23/0.77. En el caso de una demanda generada aleatoriamente, $\mu_1 = -0,0103$ y $\sigma_1 = 0,0068$ son la media y la desviación estándar para la primera población y $\mu_2 = 0,0084$ y $\sigma_2 = 0,0085$ para la segunda, con proporción de mezcla de 0.49/0.51

3.6.4. Adaptación a fallos

El aspecto más interesante del consenso dinámico es la tolerancia a fallos. Además de la robustez de la red, el algoritmo debe funcionar correctamente incluso en caso de fallo de una subestación, actualizando la información para que las subestaciones activas restantes puedan asumir la carga perdida por el fallo, de forma que se pueda satisfacer toda la demanda.

Se consideran dos desviaciones de los valores para comprobar el comportamiento del algoritmo y aislar sus efectos. La primera está relacionada con los cambios en la demanda y no se considera un fallo. Este comportamiento aparece cada vez que se obtiene un nuevo valor para la demanda y se propaga a través de la red para que las subestaciones se ajusten a ella. La Figura 3.11(a) muestra este efecto. La demanda global desciende en 1 000 MWh y la red vuel-

Figura 3.11. Error relativo ante eventos inesperados, debidos a cambios bruscos en la demanda o a la caída de una subestación.

ve a converger sin problemas al nuevo valor de referencia en 10 iteraciones. El segundo efecto está relacionado con el fallo de una subestación (Llucmajor) y se muestra en la Figura 3.11(b). En este caso, se observa una oscilación hasta que las redes compensan la potencia perdida por la subestación fallida.

En ambos casos, este comportamiento es realizado automáticamente por los vecinos de los nodos que fallan o que detectan el cambio en la demanda. No se necesita un control central ni información global.

Finalmente, la Figura 3.12 compara el comportamiento de la red sin fallos con el efecto del fallo en una subestación mientras se actualiza la demanda, por lo que todos los efectos se consideran al mismo tiempo. La línea roja muestra la diferencia entre ambos errores (con y sin fallos). Dado que las condiciones iniciales son las mismas, antes del fallo en la subestación, ambas redes se comportan de la misma manera. Cuando la subestación falla, se observa una oscilación y después de un corto tiempo de transición, la red parece adaptarse a la nueva situación. Aparece una suave diferencia entre las dos redes, debido al cambio en la topología introducida por la fallida subestación. Llucmajor es una de las subestaciones más centrales, tal y como aparece en la Tabla 3.2).

En el instante 1 600, la subestación se reactiva y la red recupera la evolución normal. Además, se recupera totalmente del fallo y continúa con la evolución normal como si la subestación nunca hubiera fallado, como indica la diferencia cero entre los errores.

Con este caso de estudio se ha evaluado la adaptación de los procesos

Figura 3.12. Evolución relativa de los errores tras un fallo en comparación con el comportamiento normal de la red. La línea roja indica la diferencia entre ambos errores. En la iteración 1 600 (día 11), la subestación se reactiva y la red recupera la evolución normal.

de consenso en redes dinámicas, en las que varios de los parámetros de la red varían mientras tiene lugar el proceso de consenso. En este ejemplo, se producen cambios en los valores iniciales, borrados e inserción de nodos y cambios en los pesos.

Capítulo 4

Consenso en redes con signo

4.1. Introducción

El objetivo de este capítulo es proponer un modelo de consenso que sea aplicable cuando las relaciones entre los nodos de la red pueden ser positivas o negativas. La principal aportación es la creación de una variante del algoritmo de consenso para detectar comunidades en redes con signo. El proceso es completamente descentralizado y considera exclusivamente la información proporcionada por sus vecinos directos. Cuando converge, cada nodo sabrá calculado cuáles de sus vecinos pertenecen a su misma comunidad y cuáles a otros grupos.

Sin embargo, el proceso de consenso no puede aplicarse directamente a las redes con signo. La matriz utilizada en el consenso debe ser definida positiva. Para garantizarlo, la matriz de adyacencia que modela la red no puede contener pesos negativos. El mismo problema ya ha sido resuelto para el análisis espectral, en el que el estudio de los grafos se realiza usando métodos de álgebra lineal. Habitualmente, este método se emplea como una técnica de agrupamiento (*clustering*) para analizar comunidades, segmentación de imágenes, minería de datos y otros problemas de clasificación [NdC11]. El proceso se inicia a partir de dos matrices: la matriz de adyacencia y la matriz laplaciana. La mayoría de los métodos se basan en el cálculo de valores propios y sus correspondientes vectores propios. Pero esta técnica también necesita grafos no ponderados o ponderados pero positivos.

En este dominio, Kunegis et al. [KSL⁺10] proponen una alternativa para calcular la matriz laplaciana que evita que L sea una matriz no definida. La matriz laplaciana con signo, $\bar{L}$ de un grafo G , con una matriz de adyacencia A , viene dada por

$$\bar{L} = \bar{D} - A \quad (4.1)$$

donde la matriz $\bar{D}$ es una matriz diagonal cuyos elementos en la diagonal son

$$\bar{d}_{ii} = \sum_{j \neq i} |a_{ij}| \quad (4.2)$$

La diferencia es que $\bar{D}$ mantiene el grado de cada nodo, mientras que en la matriz de grados original D pueden ocurrir cancelaciones. Después de este cambio, el análisis espectral se puede utilizar como una técnica de agrupación en gráficos con signo. Pero todavía necesita una visión global de la red.

4.2. Consenso en redes con signo

Sea $G = (V, E)$ un grafo ponderado, con signo y no dirigido, con $W = [w_{ij}]$ la matriz de adyacencia pesada asociada a G . w_{ij} es el peso asociado al enlace $(i, j) \in E$, es real y está definido en el intervalo $[-1, 1]$. El proceso de consenso en redes con signo está guiado por la matriz de Perron con signo $\bar{P}$

$$x(t+1) = \bar{P}x(t), \quad \bar{P} = I - \varepsilon \bar{L} \quad (4.3)$$

siendo $\bar{L}$ la laplaciana con signo definida por la Ecuación 4.1. La expresión local del proceso de consenso en redes con signo es

$$x_i(t+1) = x_i(t) + \varepsilon \sum_{j \in N_i} [w_{ij}x_j(t) - x_i(t)] \quad (4.4)$$

Cuando el proceso converge, el sistema se estabiliza alrededor de dos valores x y $-x$, y el signo es el atributo que establece la pertenencia a uno u otro grupo.

El proceso de consenso lleva a la partición de los nodos de la red V en dos subconjuntos V_1, V_2 tales que $V_1 \cup V_2 = V$ y $V_1 \cap V_2 = \emptyset$, siguiendo los enlaces positivos y negativos. Dos nodos i, j pertenecen al mismo subconjunto si están unidos por un enlace de peso positivo $w_{ij} > 0$ y pertenecen a subconjuntos distintos si $w_{ij} < 0$. De esta forma, E puede dividirse también en dos subconjuntos: uno de aceptación E_A y otro de rechazo E_R , que se definen a partir de las siguientes dos condiciones:

1. si $(i, j) \in E_A$ entonces $i, j \in V_1$ o $i, j \in V_2$
2. si $(i, j) \in E_R$ entonces $i \in V_1$ y $j \in V_2$, o $i \in V_2$ y $j \in V_1$

Figura 4.1. Ejemplo de consenso en una red balanceada con signo. Las líneas de puntos son enlaces negativos. La división óptima es $V_1 = \{1, 3, 4\}$ y $V_2 = \{2\}$. El proceso de consenso identifica a los dos grupos con $x_1 = x_3 = x_4 = 0,3571$ y $x_2 = -0,3571$.

La Figura 4.1 muestra un ejemplo de consenso en una red con signo. Las líneas punteadas (Figura 4.1(a)) corresponden a enlaces con pesos negativos. En este caso, puede observarse que existe una partición óptima. La evolución del proceso de consenso aparece en la Figura 4.1(b) muestra un ejemplo de consenso en una red con signo. Los valores iniciales de x_i se eligen aleatoriamente. Cuando el proceso converge, el signo de los valores de los nodos indica el grupo al que pertenecen. En este caso, $x(t) = \pm 0,3571$. Se puede ver que el nodo 2 está separado del resto de la red. La mejor partición es $V_1 = \{1, 3, 4\}$, $V_2 = \{2\}$, así que E_A está formada por los enlaces $E_A = \{(1, 3), (1, 4), (3, 4)\}$ y $E_R = \{(1, 2), (2, 3), (2, 4)\}$. El resultado que da el proceso de consenso deja $x_1 = x_3 = x_4 = 0,3571$ y $x_2 = -0,3571$, es decir, el mismo en valor absoluto, pero con distinto signo.

Este resultado se obtiene cuando existe una partición en la que todos los enlaces que pertenecen a E_A tienen peso positivo y todos los enlaces que pertenecen a E_R tienen peso negativo (como en el ejemplo de la Figura 4.1). Para ello, se deben cumplir cualquiera de estas condiciones equivalentes:

1. G es un grafo balanceado
2. el grafo generado por los enlaces negativos $G_R = (V, E_R)$ debe ser un grafo bipartido completo

Figura 4.2. Ejemplo de consenso en una red desbalanceada. El enlace entre los nodos dos y cuatro es positivo, pero pertenecen a grupos distintos. La división óptima es $V_1 = \{1, 3, 4\}$ y $V_2 = \{2\}$, pero el consenso ahora converge a cero por la izquierda y por la derecha.

3. G no tiene ciclos negativos de longitud impar
4. el mayor valor propio de $\bar{P}$ es $\lambda_n = 1$

En este caso, el proceso de consenso converge a

$$\lim_{t \rightarrow \infty} = \frac{\sum_{i \in V_1} x_i(0) - \sum_{i \in V_2} x_i(0)}{n} \quad (4.5)$$

La principal contribución en este aspecto del presente trabajo es que el algoritmo de consenso propuesto también encuentra una partición cuando la red no es balanceada. En ese caso, el valor de todos los nodos converge a cero: $\lim_{t \rightarrow \infty} = 0$. Esto se debe a que $\lambda_n < 1$

4.3. Generalización a grafos m -partidos

La generalización para detectar comunidades de m se define por un proceso iterativo que aplica el proceso de consenso sobre la partición identificada en el paso anterior. Dada una red $G = (V, E)$, queremos encontrar la partición V en los conjuntos $V_1, \dots, V_m$, de tal manera que:

1. si (i, j) está en E_A entonces $i, j \in V_k, k = 1, \dots, m$, y

2. si (i, j) está en E_R entonces $i \in V_k, j \in V_l$ con $k \neq l$.

Supongamos que se ha aplicado el método propuesto, obteniendo dos comunidades. El algoritmo se ha detenido con un valor x_i para cada nodo. como la permutación que mantiene el orden descendente de x y $P_p i$ la matriz de permutación asociada. Si se ordena el vector y la matriz de adyacencia se permuta de la misma manera, se obtiene una matriz de bloques.

$$P_\pi A P_\pi = A^P = \left(\begin{array}{c|c} C_1 & D_1 \\ \hline D_2 & C_2 \end{array} \right) \quad (4.6)$$

donde C_1 y C_2 son las submatrices que representan los enlaces dentro de cada grupo y D_1 y D_2 mantienen los enlaces entre grupos. Si G es un gráfico no dirigido, la matriz A^P es simétrica y, por lo tanto, $D_2 = D_1^T$. Además, si es un gráfico con signo equilibrado, todos los elementos en C_1 y C_2 serán positivos y los elementos en D_1 y D_2 serán negativos. Si hay un grupo C_k en el que hay pesos positivos y negativos, este es un indicador de que posiblemente más de una comunidad se superponen en ese grupo. Por lo tanto, se puede ejecutar una nueva iteración del método. Podemos dividir A^P en dos matrices de bloques

$$A^P = \left(\begin{array}{c|c} C_1 & D_1 \\ \hline D_2 & C_2 \end{array} \right) = \left(\begin{array}{c|c} C_1 & 0 \\ \hline 0 & C_2 \end{array} \right) + \left(\begin{array}{c|c} 0 & D_1 \\ \hline D_2 & 0 \end{array} \right) \quad (4.7)$$

y aplicando el proceso sobre la matriz del puño obtenemos una nueva subdivisión

$$C_1 = \left(\begin{array}{c|c} C_{11} & D_{11} \\ \hline D_{12} & C_{12} \end{array} \right), \quad C_2 = \left(\begin{array}{c|c} C_{21} & D_{21} \\ \hline D_{22} & C_{22} \end{array} \right) \quad (4.8)$$

Seguiremos haciéndolo hasta que todos los elementos de todos los C_k sean positivos o hasta que $x_i > 0 \forall i$. La matriz final de A^P es

$$A^P = \left(\begin{array}{ccc} C_1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & C_m \end{array} \right) + \left(\begin{array}{ccc} 0 & \dots & D_{1m} \\ \vdots & \ddots & \vdots \\ D_{m1} & \dots & 0 \end{array} \right) \quad (4.9)$$

donde C_k son las comunidades y D_{ij} denota los enlaces existentes entre los grupos C_i y C_j . Por lo tanto, los conjuntos $E_A = \{e_{ij} | e_{ij} \in C_k, k = 1, \dots, m\}$ y $E_R = \{e_{ij} | e_{ij} \in D_{kl}, k, l = 1, \dots, m\}$.

El proceso completo se resumen en el Algoritmo 4.1

Algoritmo 4.1*Detección de comunidades*

```

1:  $x$  = resultado del consenso
2: while  $\exists x_i < 0$  do
3:   desactivar los enlaces  $(i, j) \mid x_i \times x_j < 0$ 
4:    $x$  = resultado del consenso
5: end while

```

4.4. Coherencia como medida de la calidad de la división en comunidades

El proceso de consenso conduce a la división de la red en dos grupos, siguiendo los pesos positivo y negativo, porque equivale a resolver un problema de coherencia. La coherencia puede describirse como un problema de satisfacción de restricciones (CSP–*constraint satisfaction problem*–), donde el valor de coherencia de la red debe ser maximizado [TV98]. El problema se representa en un grafo ponderado, no dirigido $G = (V, E)$. Cada enlace tiene asociado un peso $w_{ij} \in [-1, 1]$. Los enlaces representan restricciones que se deben cumplir. Un enlace positivo significa que los nodos implicados deben estar en el mismo grupo, mientras que un enlace negativo indica que los nodos deben pertenecer a grupos diferentes. Así, E puede dividirse en un conjunto de restricciones aceptadas E_A y un conjunto de restricciones rechazadas E_R , tal y como se ha definido en el proceso de consenso para redes con signo.

Sean $E_A, E_R \subseteq E$ la partición del conjunto de restricciones que cumplen las condiciones 1 y 2. Se define el valor de coherencia de la partición de E en (E_A, E_R) como

$$Coh = \frac{\sum_{e_{ij} \in E_A} w_{ij} - \sum_{e_{ij} \in E_R} w_{ij}}{\sum_{\forall i,j} |w_{ij}|} \quad (4.10)$$

La coherencia es óptima cuando no hay ninguna otra partición que produzca un mayor valor de coherencia

$$Coh^{opt} = \arg \max_{E_A \subseteq E} Coh \quad (4.11)$$

La coherencia es máxima Coh^* cuando el grafo está balanceado y todos los enlaces con peso positivo están en E_A y todos los enlaces con peso negativo

Tabla 4.1

Coherencia para todas las posibles particiones de V

partición	<i>Coh</i>
{1, 2, 3, 4}	0.500
{1, 2, 3} {4}	-0.750
{1, 2, 4} {3}	0.250
{1, 2} {3, 4}	-0.250
{1, 3, 4} {2}	0.625
{1, 3} {2, 4}	0.125
{1, 4} {2, 3}	0.000
{1} {2, 3, 4}	0.375

Nota. Coherencia para todas las particiones posibles de V . El óptimo $Coh^{opt} = 0,625$ corresponde a la partición $V_1 = \{1, 3, 4\}$, $V_2 = \{2\}$ Como la red no está balanceada, no se alcanza el valor máximo de coherencia

en E_R . En ese caso, $Coh = Coh^{opt} = Coh^* = 1$. El valor de coherencia del ejemplo que aparece en la Figura 4.1 sería

$$Coh = \frac{0,5 + 0,5 + 1 - (-0,5 - 0,5 - 1)}{4} = 1 = Coh^{opt} = Coh^* \quad (4.12)$$

Cualquier otra partición arroja un valor menor para la coherencia. Por ejemplo, si $V_1 = \{1, 2\}$ y $V_2 = \{3, 4\}$, los enlaces quedan repartidos de la siguiente forma: $E_A = \{(1, 2), (3, 4)\}$, $E_R = \{(1, 3), (1, 4), (2, 3), (2, 4)\}$ y el valor de coherencia

$$\begin{aligned} Coh &= \frac{(w_{12} + w_{34}) - (w_{13} - w_{14} - w_{23} - w_{24})}{4} = \\ &= \frac{(-0,5 + 1) - (0,5 + 0,5 - 0,5 - 1)}{4} = \frac{1}{4} = 0,25 \end{aligned} \quad (4.13)$$

En cambio, en el ejemplo de la Figura 4.2 no se puede alcanzar la puntuación máxima, pues el enlace positivo $w_{24} = 1$ hace que la red no esté balanceada. En ese caso, los valores de coherencia que se obtienen para las distintas particiones son los calculados en la Tabla 4.1. La partición detectada en la Figura 4.2 es la que tiene el valor de coherencia óptimo para la red dada $Coh^{opt} = 0,625$.

4.5. Caso de estudio: detección de comunidades

4.5.1. Aplicación a redes de referencia

El método se ha aplicado a casos típicamente utilizados en la literatura sobre redes con signo para comprobar la validez de la propuesta. El primero es un ejemplo del análisis del Parlamento esloveno [KM96], en el que se estudian las relaciones entre los partidos políticos que componen el Parlamento. Se pidió a 74 de los 90 miembros del Parlamento que calcularan la distancia entre los diez partidos parlamentarios entre -3 (muy diferente) y +3 (muy similar). La Figura 4.3 muestra las comunidades obtenidas con el proceso de consenso, que son los mismos grupos obtenidos en la investigación original con el valor de coherencia óptimo $Coh = 0,954 = Coh^{opt}$. Tenga en cuenta que la partición perfecta no es posible ya que la parte SNS tiene una relación negativa con algunas partes de ambos grupos.

La Figura 4.3 muestra las comunidades que se obtienen en el ejemplo, que son las mismas que las que se obtienen en el estudio original. El proceso comienza con una orden arbitrario para la red. El proceso de consenso en la red de partidos eslovenos converge con un valor de $\bar{x}_i(t) = \pm 0,0610$. Si reordenamos los partidos políticos de acuerdo al valor final, apreciamos claramente la división en grupos $V_1 = \{ZS, SLS, SDSS, SPS/SNS, SKD\}$ y $V_2 = \{SNS, ZESS, DS, LDS, ZLSD\}$. El partido SNS pertenece a V_2 aunque se ha considerado diferente a ZESS y DS.

El valor de coherencia obtenido por esta división es el óptimo $Coh = 0,954 = Coh^{opt}$. Se debe tener en cuenta que la partición perfecta no es posible ya que la parte SNS tiene una relación negativa con algunas partes de ambos grupos y el grafo no está balanceado.

Un segundo ejemplo muestra el proceso cuando aparecen más de dos grupos. El caso modela las relaciones entre las tribus nativas de Nueva Guinea [Rea54]. La red está formada por 16 tribus, donde los pesos positivos y negativos denotan la amistad o la enemistad respectivamente. También se considera la distancia entre las tribus, ya que Read afirma en su artículo que la relación es más fuerte entre vecinos que con otras tribus distantes. Fig. 4.4 muestra el resultado cuando se utiliza un método de análisis espectral [DM04], calculando los k primeros vectores propios de la matriz laplaciana con signo para identificar las comunidades. EL primer vector propio (representado en el eje x) separa inequívocamente a la comunidad V_2 , pero el segundo eje no es capaz de separar a los otros dos grupos.

Las comunidades detectadas están formadas por los grupos

- $V_1 = \{OVE, ALIKA, GAHUK, MASIL, UKUDZ, GEHAM, ASARO\}$

Figura 4.3. Grupos del Parlamento esloveno.

$V_1 = \{ZS, SLS, SDSS, SPS/SNS, SKD\}$ y

$V_2 = \{SNS, ZESS, DS, LDS, ZLSD\}$ con $Coh = 0,954$. La red converge a $\bar{x} = \pm 0,0610$

Figura 4.4. Detección de comunidades mediante análisis espectral usando los dos primeros vectores propios de L . Una de las comunidades está claramente identificada por λ_1 , pero el signo de λ_2 no separa la segunda.

Figura 4.5. Detección de comunidades para el caso de las tribus de Nueva Guinea. La red comienza en el estado inicial que muestra (c). Tras ejecutar el consenso de (a) se alcanza la primera división (d). Tras un segundo proceso de consenso (b), se llega a la división definitiva de comunidades (e)

- $V_2 = \{\text{GAVEV, KOTUN, NAGAD, GAMA}\}$
- $V_3 = \{\text{NAGAM, NOTOH, KOHIK, UHETO, SEUVE}\}$

La Figura 4.5 muestra el resultado obtenido mediante el proceso de consenso. Se necesitan dos iteraciones para clasificar las comunidades. La primera iteración aísla V_2 con un valor de consenso alrededor de $\pm 10^{-5}$ después de 150 pasos. La segunda iteración divide el grupo restante en dos comunidades V_1 y V_3 . Una vez más, como la segunda partición no es perfecta, el valor de consenso tiende a ser $\pm 10^{-5}$, mientras que la comunidad V_2 converge a un valor diferente $\bar{x} = 0,51$.

4.5.2. Evaluación en redes sintéticas

Lancichinetti et al. [LFR08] han propuesto un conjunto de gráficos de referencia para evaluar los algoritmos de detección comunitaria. La métrica de evaluación utilizada es la Información Mutua Normalizada (NMI) entre la partición real (debe ser conocida) y la partición obtenida por el algoritmo estudiado. Nuestro algoritmo ha sido comparado con la modularidad clásica y los métodos de análisis espectral. Además, se ha incluido el modelo de Pott de resolución local sin límite [RN10] y una técnica basada en la optimización local de una función de ajuste (FLK) [LFK09]. Finalmente, se ha añadido una clasificación arbitraria a los resultados a modo de referencia. Para aquellos algoritmos que trabajan sólo con ponderaciones positivas, la matriz de adyacencia se ha renormalizado en $[0,1]$ y los pesos inferiores a 0,5 se consideran negativas.

Los experimentos se han dividido en dos grupos. El primer conjunto estudia la influencia de la mezcla entre comunidades (ver Fig. 4.6). Se define por un parámetro mx que varía de 0.2 a 0.5. Se ha utilizado un número fijo de comunidades $q = 2$ de igual tamaño (el tamaño de la comunidad es irrelevante para el proceso de consenso). En el segundo grupo, el parámetro de mezcla es fijo, y se estudia el efecto del número de comunidades. En todos los casos, el tamaño de la red varía de 120 a 480 nodos.

El algoritmo de consenso obtiene la distribución correcta de la comunidad independientemente del tamaño de la red y de la mezcla de las comunidades, mientras que el rendimiento del resto de los algoritmos decae cuando la mezcla de las comunidades aumenta, especialmente para las redes pequeñas, por orden: Pott, modularidad, análisis espectral y LFK. Pero cuando el número de comunidades aumenta, el rendimiento del algoritmo de consenso disminuye, mientras que el resto de los métodos detecta correctamente a casi todas las comunidades. El análisis espectral es el único algoritmo cuyo rendimiento disminuye a medida que aumenta el número de comunidades. La razón es que los gráficos generados en el punto de referencia tienen comunidades superpuestas. El NMI es un factor que tiene en cuenta la densidad de los vínculos y el consenso es un método centrado en el valor de los vínculos y trata de mantener el número de comunidades lo más reducido posible.

Además del NMI, se ha analizado el tiempo de ejecución de los diferentes algoritmos. El tiempo medio de ejecución y la desviación estándar se calculan a partir de 5 repeticiones de cada configuración. No hay diferencias significativas entre los dos conjuntos (variando el parámetro de mezcla y el número de comunidades) y la información no se incluye debido a limitaciones de espacio. En todos los casos, el algoritmo de consenso es más eficiente que el resto,

Figura 4.6. La información mutua normalizada (NMI) para los diferentes algoritmos en función del tamaño de la red. Todas las redes tienen comunidades $q = 2$ y el parámetro mixto entre comunidades varía: (a) $mx = 0.2$, (b) $mx = 0.3$, (c) $mx = 0.4$, (d) $mx = 0.5$. El algoritmo de consenso identifica correctamente las comunidades para cualquier tamaño de red en todos los casos. A medida que aumenta el factor de mezcla, el rendimiento del resto de los métodos es peor para las redes pequeñas.

especialmente cuando se compara con los modelos LFK y Potts.

4.5.3. Conflictos en la Wikipedia

El último experimento analiza la escalabilidad del algoritmo. Se ha utilizado un conjunto de datos que registra los conflictos positivos y negativos entre usuarios de Wikipedia implicados en guerras de edición sobre un artículo [KON13, BL10, Kun13]. Los nodos representan usuarios y enlazan un conflicto entre dos usuarios. La red tiene 118.100 nodos y más de 4×10^6 enlaces. Para ese tamaño de red, no hay ningún método para extraer la distribución óptima de la comunidad en un tiempo factible. El número de combinaciones posibles es de 2^m , siendo m el número de enlaces en la red. Este problema

Tabla 4.2

Tiempo de ejecución de los diferentes algoritmos y configuraciones

mezcla	nodos	coherencia	espectral	modularidad	LFK	Pott
	120	0.0045±0.00097	0.016±0.0024	0.01±0.0023	4.1±1.1	0.41±0.01
0.2	180	0.0058±0.00047	0.02±0.0073	0.024±0.00083	15±5.5	1±0.029
	240	0.0062±0.00026	0.018±0.0015	0.052±0.00063	19±3.7	2.3±.014
	300	0.0077±0.00023	0.017±0.00028	0.096±0.0046	34±7.7	4.3±0.077
	360	0.0093±0.00022	0.019±0.0018	0.16±0.0036	62±18	7.1±0.039
	420	0.011±0.00029	0.026±0.0054	0.25±0.0059	109±35	11±0.099
	480	0.013±0.00055	0.026±0.0041	0.39±0.027	135±35	15±.22
0.3	120	0.0048±0.00063	0.021±0.0045	0.0085±0.00055	8.8 ± 2.3	0.4±0.0095
	180	0.0057±0.00032	0.019±0.003	0.024±0.0049	16±3.8	1.1±.038
	240	0.0059±0.00012	0.019±0.0046	0.049±.0036	33±3	2.2±0.036
	300	0.007±0.00023	0.018±0.00094	0.09±0.0017	35±8	4.1±0.023
	360	0.0085±0.0004	0.018±0.0007	0.15±0.00087	43±8.2	7±0.17
	420	0.011± 0.00038	0.025±0.0066	0.25±0.0056	98±28	11±0.1
	480	0.013±0.0014	0.024±0.0015	0.38±0.0096	92±20	15±.31
0.4	120	0.0061±0.001	0.019±0.0017	0.0085±0.00047	11±4	0.4±0.0032
	180	0.0066±0.00069	0.023±0.004	0.024±0.0041	29±2.5	1.1±0.021
	240	0.0067±0.00034	0.022±0.0056	0.055±0.0089	39±6	2.4±.077
	300	0.0073±0.0004	0.02±0.0049	0.091±0.0025	32±20	4.2±0.068
	360	0.0084±0.00018	0.019±0.0011	0.15±0.0023	49±24	7±.12
	420	0.01± 0.00043	0.021±0.0017	0.25±0.0071	87±20	11±.1
	480	0.013±0.0023	0.025±0.0025	0.39±0.014	90±0	15±0.12
0.5	120	0.0099±0.0012	0.018±0.0014	0.013±0.0058	11±4.6	0.4±0.012
	180	0.0082±0.0014	0.024±0.0037	0.022±0.00032	35±10	1.1±0.03
	240	0.0076±0.00078	0.021±0.0037	0.047±0.00065	56±12	2.4±0.07
	300	0.0078±0.00091	0.021±0.0046	0.091±0.0028	73±22	4.4±0.14
	360	0.0087±0.00022	0.02±0.00058	0.16±.0056	70±35	6.9±0.12
	420	0.01±0.00034	0.025±0.0045	0.26±0.012	87±40	11±0.21
	480	0.013±0.00046	0.03±0.0064	0.4±0.018	59±13	15±0.2

Nota. Se consideran comunidades con $q = 2$. Los datos son el valor medio $\pm$ la desviación estándar calculada a partir de 5 ejecuciones en cada configuración. Los tiempos obtenidos para el segundo conjunto experimental (variando el número de comunidades) están en el mismo orden de magnitud. Los tiempos de ejecución obtenidos para el proceso de consenso son mejores en varios órdenes de magnitud que el resto, especialmente los algoritmos LFK y Potts.

pertenece a la categoría de problemas completos de NP. Por lo tanto, el valor de NMI no puede ser usado como una medida de calidad.

Otro criterio utilizado para medir la calidad de una división en comunidades es el índice de frustración $F(C)$ de una partición $C = \{C_1, \dots, C_k\}$ en k comunidades como la suma de enlaces negativos dentro de una comunidad y los enlaces positivos entre nodos de diferentes comunidades [YCL07, AP13].

Figura 4.7. Información mutua normalizada (NMI) para los diferentes algoritmos en función del tamaño de la red. Todas las redes tienen el mismo parámetro $mx = 0,2$ y el número de comunidades varía: (a) $q = 2$, (b) $q = 4$, (c) $q = 6$, (d) $q = 8$. El algoritmo de consenso falla a medida que aumenta el número de comunidades, mientras que el restablecimiento de los métodos identifica correctamente las comunidades para cualquier tamaño de red. Los resultados son peores para las redes pequeñas

$$F(C) = \sum_{i,j \in V} A_{i,j}^- \delta(c_i, c_j) + A_{i,j}^+ (1 - \delta(c_i, c_j)) \quad (4.14)$$

donde c_i es la comunidad a la que pertenece el nodo i y $\delta(c_i, c_j)$ es la función del ta de Kronecker, que toma valor uno cuando i y j pertenecen a la misma comunidad y cero en caso contrario. Nótese la similitud con la ecuación del valor de coherencia de la partición Coh . Por lo tanto, calcularemos el valor de coherencia de la división de la comunidad como una medida de la calidad de la solución.

Pero, ¿es el valor de coherencia un buen estimador del valor real? Si se conoce el valor de coherencia para la partición óptima Coh^* , entonces un valor más alto para la relación Coh/Coh^* significa una mejor solución, siendo igual a 1 cuando la división encontrada es la óptima. Se puede demostrar que algunos

métodos para resolver problemas de coherencia logran un ratio de optimización $> 0,87856$ [TV98]. Por lo tanto, hemos considerado que las soluciones que superan este umbral son una partición suficientemente buena.

Fig. 4.8 muestra los resultados obtenidos para el conjunto de datos de conflictos del wiki. Los resultados han sido promediados después de diez ejecuciones. El consenso comienza con valores aleatorios para $x_i(0)$. La detección de comunidad se detiene después de iteraciones $t = 100$. Los resultados comparan el rendimiento del proceso usando diferentes precisiones para el paso de consenso, usando un error $e = 10^{-3}, 10^{-4}, 10^{-5}$. Esto significa que el proceso de consenso se detiene cuando todos los nodos tienen una diferencia con su valor anterior inferior al error e .

El número de comunidades crece hasta la iteración 50 aproximadamente, independientemente de la precisión. Después de ese momento, el número de comunidades detectadas permanece inalterado. Hay diferencias en el número de comunidades encontradas que dependen de la precisión. En los pasos iniciales, cada comunidad identificada se divide en dos grupos. Pero después de diez divisiones, un pequeño número de grupos se redefinen en cada iteración. No hay diferencias significativas basadas en la precisión.

En cuanto a la calidad de la solución, la precisión afecta a la rapidez con la que la solución converge hacia la mejor. Usando una mayor precisión se consigue más rápidamente un valor de partición casi óptimo. Pero se necesita un número mínimo de iteraciones para llegar a un número estable de comunidades. Cualquier solución que llegue al punto en que el número de comunidades se estabilice con el mejor valor de coherencia puede considerarse una respuesta válida. El ratio de estimación de la optimización coincide con los criterios de 0,878 en todos los casos después de un número suficiente de iteraciones. Si se tienen en cuenta los tiempos de ejecución (ver Tabla de tiempos de ejecución), una precisión de 10^{-4} da la mejor solución posible con un tiempo de ejecución asequible, a pesar de que sobreestima el número de comunidades.

Tabla 4.3

<i>Tiempo medio de ejecución de una iteración de consenso</i>						<i>Nota.</i>
error	10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}	
time	0.79 ± 0.04	1.0 ± 0.16	9.14 ± 0.65	142 ± 12	564 ± 27	
Tiempo $\bar{t} \pm \sigma$, en segundos						

En este capítulo se ha propuesto y probado un método basado en una red de consenso como mecanismo para formar comunidades de manera descentralizada, teniendo en cuenta exclusivamente la información local sobre el propio

Figura 4.8. Realización del algoritmo de detección de comunidades para el conjunto de datos de conflicto de wikipedia. Los criterios de parada para cada consenso han sido probados para un error máximo de $e = 10^{-3}, 10^{-4}, 10^{-5}$ (a) El número de comunidades detectadas se estabiliza después de 50 iteraciones. (b) Evolución del tamaño medio de la comunidad dividido por 2 en las iteraciones iniciales. (c) Evolución de la calidad de la partición utilizando medidas de coherencia. Un error superior a $e = 10^{-4}$ no mejora sustancialmente el resultado. (d) Calidad óptima estimada. Utilizando un error limitado por $e = 10^{-4}$ se obtiene un buen resultado

agente y sus vecinos directos. Un proceso de consenso es un proceso descentralizado que puede converger al mismo valor para un parámetro. En cada iteración, todos los nodos de la red intercambian sus valores con sus vecinos directos, y modifican su valor. Los agentes sólo tienen que comunicarse con sus vecinos inmediatos, y no es necesario propagar información adicional sobre la red. No se requiere preparación, ni organización previa en el gráfico ni conocimientos a priori. Además, sólo el valor actual del agente para el parámetro y los valores enviados por sus vecinos en cada ronda.

Este trabajo contribuye en tres aspectos principales: primero, la extensión de la red de consenso para tratar una matriz de pesos negativos (gráficos fir-

mados). En segundo lugar, cómo utilizar este modelo modificado para detectar comunidades como problemas de coherencia. Finalmente, una generalización de los métodos como mecanismo de detección de comunidades para dividir una población de nodos en grupos de m , maximizando el valor de coherencia.

Los resultados sugieren que el modelo propuesto es válido ya que las soluciones obtenidas son comparables a las obtenidas por otros métodos bien conocidos y centralizados, basados en el análisis espectral o la modularidad, entre otros. Se han realizado diferentes pruebas para comprobar el rendimiento del proceso y la calidad de la división de la comunidad sobre conocidos ejemplos de gráficos firmados, redes artificiales creadas a partir de los gráficos de referencia del LFR y con un escenario real con grandes redes para probar la escalabilidad. Los resultados sugieren que el enfoque de consenso es un método válido para identificar las comunidades existentes, recuperando la partición óptima para gráficos balanceados. Para las comunidades que se superponen, funciona mejor a medida que aumenta el tamaño de la red, pero otros métodos proporcionan una mejor solución para gráficos pequeños con un gran número de comunidades.

Capítulo 5

Consenso multidimensional

5.1. Introducción

El proceso de consenso converge al valor medio de la red para una variable x o para m variables $x^1, \dots, x^m$ independientes. En este caso, se pueden hacer m procesos de consenso en paralelo, intercambiando un vector con m componentes en cada iteración. La expresión matricial sigue siendo la misma, aunque en este caso se sustituye el vector $x^{(n \times 1)}$ por una matriz $X^{(n \times m)}$

$$X(t) = P \cdot X(t-1) \quad (5.1)$$

La expresión individual de cada nodo (véase Ecuación 2.1) también se mantiene, pero ahora los términos $x_i(t)$ y $x_j(t)$ son vectores $x^{(1 \times m)}$

Sin embargo, es habitual que exista algún tipo de dependencia entre las variables, de manera que el consenso al que se llegue en una de ellas afecte a los valores aceptables para otra. Este tipo de fenómenos suele aparecer en escenarios donde un grupo tiene que llegar a un acuerdo sobre una decisión en común y hay más de un criterio. Por ejemplo, vamos a abrir una sucursal de nuestro negocio y tenemos que decidir en qué ciudad en función de sus habitantes, la renta *per cápita*, sus posibilidades de comunicación y el número de empresas relacionadas que están establecidas ya en ese lugar.

En casos como ese, los procesos de consenso deben incluir algún tipo de optimización sobre una función de utilidad $g : \mathbb{R}^m \rightarrow \mathbb{R}$ que otorgue una valoración a la solución que se alcanza. Los métodos centralizados requieren fusionar la información que se encuentra distribuida por toda la red, lo que supone un alto coste tanto computacional como de comunicación cuando el tamaño de la red aumenta. Las aproximaciones descentralizadas tienen la ventaja de una mayor escalabilidad, adaptación a cambios en la topología de la

red y pueden proporcionar privacidad en los datos. Existen varios algoritmos que pueden resolver problemas de optimización acoplados. La combinación de algoritmos de consenso y de descenso por gradiente es una de ellas y puede expresarse como un proceso de dos etapas [YLY14]

$$x_i(t+1) = \sum_j w_{ij} x_j(t) - \alpha \nabla g_i(x_i(t)) \quad (5.2)$$

donde $W = [w_{ij}]$ es una matriz simétrica y doble estocástica y $\nabla g_i(x_i(t))$ es una función que modela un descenso por gradiente para minimizar una función de coste.

Es este capítulo se propone una extensión del modelo de consenso en este sentido, aplicado a la consecución de consensos cuando cada nodo tiene una función de utilidad que quiere maximizar. La extensión permite que cada nodo mantenga en privado su utilidad, intercambiando únicamente con los vecinos el valor de las variables de consenso (que en este caso se encuentran acopladas a través de la función de utilidad).

5.2. Consenso con variables multidimensionales

Supongamos una red formada por n nodos que desean alcanzar un acuerdo sobre m variables acopladas. Cada nodo tiene una función de utilidad $u_i = g_i(x_i^1, \dots, x_i^m)$ monótona no decreciente definida a partir de los valores de las variables. El proceso consistirá en m procesos de consenso en paralelo, entre los que se intercala un ascenso por gradiente sobre $g_i(x_i)$ de manera que cada nodo trata de compensar en su propia función e utilidad la desviación que le ha producido el paso de consenso ejecutado. Sobre este modelo, el proceso combinado de consenso y ascenso por gradiente tiende a alcanzar el punto de convergencia situado en el máximo de una función de utilidad global, definida como la suma de las funciones de utilidad individuales, pero que nunca se calcula ni, por supuesto, conoce ninguno de los nodos. Bajo determinadas circunstancias, el método converge al valor máximo de esa función de utilidad global, proporcionando un mecanismo válido en situaciones de información incompleta.

La generalización del proceso de consenso al caso multidimensional requiere la consideración de redes multicapa. Las redes multicapa son un formalismo creado para modelar de forma más realista ciertos fenómenos que aparecen en las redes complejas. Normalmente, las relaciones no ocurren de forma aislada en una sola red. Nociones como red de redes", redes multicapa, redes multiplex o redes interdependientes se definen para poder representarlos. En las redes

Figura 5.1. Grafo multicapa

multicapa, entre los nodos existen enlaces de distinto tipo. Por ejemplo, en un grupo de personas, los enlaces pueden representar amistad, relaciones laborales o lazos familiares. En otros casos, como en un modelo de comunicación, las capas pueden ser distintos medios, como llamadas de teléfono o correos electrónicos. Cada uno de estos tipos de enlaces forma una red (una capa). La interdependencia entre capas se define a través de enlaces cruzados entre los nodos que representan a la misma entidad en distintas capas. Estos enlaces modelan el trasvase de información de una capa a otra.

Una red multicapa se define formalmente como un par $M = (G, C)$ donde $G = \{G^1, \dots, G^m\}$ es una familia de grafos $G^\alpha = (V^\alpha, E^\alpha), \forall \alpha \in [1, m]$ llamados capas, y $C = \{L^{\alpha\beta} \subseteq E^\alpha \times E^\beta, \forall \alpha, \beta \in [1, m], \alpha \neq \beta\}$ es el conjunto de enlaces entre dos capas diferentes G^α y G^β [BBC⁺14]. Los elementos de cada E^α se denominan conexiones *intracapa* y los elementos de C son las conexiones *intercapa* o enlaces cruzados. La característica especial de las redes multiplex es que todas las capas tienen el mismo conjunto de nodos $E^1 = \dots = E^m = E$ y los enlaces cruzados se establecen entre nodos equivalentes en capas diferentes $L^{\alpha\beta} = \{(v^\alpha, v^\beta), \forall v \in V; \alpha, \beta \in [1, m]\}$.

La representación de este sistema será a través de una red multicapa, formada por m capas y en cada una de ellas n nodos (véase Figura 5.1). Cada nodo i asigna un valor de importancia a cada capa α a través del peso $w_i^\alpha \in [0, 1]$. Sin perder generalidad, podemos asumir que $\sum_\alpha w_i^\alpha = 1$. Asumiremos también que todas las funciones están idénticamente distribuidas, aunque con diferentes parámetros.

Un consenso multidimensional entre n nodos sobre m variables acopladas

se modela como una tupla $(G, C, X(t), U)$ donde

- $G = \{G^1, \dots, G^m\}$ es una red multiplex formada por m capas, cada una de las cuales trabajará sobre una de las dimensiones del consenso., siendo $G^\alpha = (V^\alpha, E^\alpha)$ el grafo asociado a cada capa. Todas las capas tienen el mismo conjunto de nodos $E^1 = \dots = E^m = E$, formando una red multiplex.
- $C = \{(v^\alpha, v^\beta), \forall v \in V; \alpha, \beta \in [1, p]\}$ es el conjunto de enlaces cruzados entre las capas
- $x(t) = \{x_1(t), \dots, x_n(t)\}, x_i(t) \in \mathbb{R}^m$, es el conjunto de valores de cada nodo en el instante t . $x_i(t) = \{x_i^1(t), \dots, x_i^m(t)\}$ tiene m componentes, una por cada dimensión.
- $U = \{u_1(x_1), \dots, u_n(x_n)\}, u_i : \mathbb{R}^m \rightarrow \mathbb{R}$, es el conjunto de las funciones de utilidad de los nodos de la red

Cada capa realiza un consenso sobre las relaciones definidas en el grafo G^α . Cada nodo i intercambia el valor x_i^α con sus vecinos directos y actualiza su valor con la información recibida siguiendo el proceso básico de consenso (véase Ecuación 2.1). Estos procesos independientes de m llevan a un consenso considerando que los términos son independientes, lo cual no es realista. La interdependencia existente entre los valores de las variables para cada nodo se define en las capas cruzadas en C . Por lo tanto, después del paso de consenso, se realiza una fase de compensación para corregir el valor de consenso hacia el valor preferido según la función de utilidad definida para el nodo i (ver Apartado 5.3).

El consenso final $X^* = \{x_1, \dots, x_n\}$, con $x_1 = \dots = x_n$ se alcanza cuando todos los nodos tienen los mismos valores para todas las dimensiones a través de un proceso de dos pasos que puede resumirse como muestra la Ecuación 5.3 :

$$\begin{aligned} x_i^\alpha(t+1) &= x_i^\alpha(t) + f_i(x_1^\alpha(t), \dots, x_n^\alpha(t)) \\ &\quad + g_i(x_i^1(t), \dots, x_i^m(t)) \end{aligned} \quad (5.3)$$

donde f_i modela el proceso intracapa (consenso en cada dimensión) y g_i modela la transferencia entre capas, usando los enlaces de capas cruzadas (ascenso por gradiente).

El proceso de consenso con variables independientes puede considerarse un caso especial, en el que todos los nodos tienen una función de utilidad

constante. Su gradiente será siempre cero, por lo que el término correspondiente a $g(x)$ desaparece y el resultado es la implementación inicial del consenso para m variables.

5.3. Función de utilidad

Las funciones de utilidad tienen un conjunto de propiedades comunes en cualquier problema de optimización: independencia, integridad, transitividad y continuidad. Además, la función debe ser monótona no creciente. La distribución gaussiana cumple todas estas propiedades y, por lo tanto, modela la función de utilidad u_i de los nodos. Asumiremos que las funciones de utilidad tienen un máximo y este máximo será el punto inicial $x_i(0)$ para todos los nodos, representando la solución individual preferida.

El peso asignado a cada capa define la medida de dispersión de la gaussiana. Un nodo no desea cambios en su término más relevante. Por lo tanto, cualquier cambio en su valor debe disminuir drásticamente su utilidad. Por otro lado, los nodos permitirían cambios en términos de baja importancia, lo que podría disminuir ligeramente sus utilidades. En el caso de una distribución gaussiana, la desviación estándar es el parámetro que rige este comportamiento. Si usamos $\sigma_i^\alpha = 1 - w_i^\alpha$ obtenemos el comportamiento deseado. La función de utilidad se define de la siguiente manera:

$$u_i^\alpha(x_i^\alpha) = e^{-\frac{1}{2} \left(\frac{x_i^\alpha - x_i^\alpha(0)}{1 - w_i^\alpha} \right)^2} \quad (5.4)$$

Todas estas funciones individuales se combinan en una única función de utilidad para el nodo.

$$u_i(x_i) = \prod_{\alpha} u_i^\alpha(x_i^\alpha) \quad (5.5)$$

Esta definición en la Ecuación 5.5 corresponde a una distribución gaussiana renormalizada, multidimensional, de manera que la utilidad máxima para el nodo i es $u_i(x_i(0)) = 1$. La utilidad global del sistema es la suma de las utilidades individuales de los agentes.

$$U = \sum_i u_i(x_i) \quad (5.6)$$

Los nodos nunca calculan esta utilidad global. La función de utilidad se mantiene desconocida para todos los nodos en el proceso de consenso y el

Figura 5.2. Proceso de consenso sobre funciones de utilidad. Cada nodo tiene su propia función que mantiene privada. Los valores iniciales son $x_i(0) = \{0,65 \ 0,75 \ 0,8 \ 0,5\}$ y $\sigma = 0,1$ para todos los nodos. El proceso de consenso converge a $x_i(t) = 0,675$

descenso por gradiente se guía únicamente por la función de utilidad local u_i de cada nodo.

La Figura 5.2(a) muestra las funciones de utilidad y la utilidad global de un ejemplo formado por cuatro nodos con valores iniciales $x_i(0) = \{0,65 \ 0,75 \ 0,8 \ 0,5\}$. Las funciones de utilidad locales son gaussianas centradas en $x_i(0)$ con un valores de $\sigma = 0,1$ para todos los nodos. El proceso de consenso (véaseFigura 5.2(b)) converge al valor medio de los valores iniciales sin tener en cuenta las funciones de utilidad, convergiendo en el punto $x_i(t) = 0,675$.

5.4. Combinación de consenso y gradiente

Un proceso distribuido que combina consenso y ascenso de gradiente converge a un valor cercano al máximo de la función de utilidad global cuando no hay otro valor máximo local. Esta solución utiliza una iteración de dos pasos. En la primera, cada agente actualiza su valor hacia sus vecinos, siguiendo un proceso de consenso. En el segundo paso, se realiza un ascenso con gradiente que desplaza de nuevo al nodo i hacia su nivel óptimo. Por lo tanto, la Ecuación 5.3 puede reescribirse como sigue

$$\begin{aligned}
x_i^\alpha(t+1) = & x_i^\alpha + \overbrace{\frac{\varepsilon}{w_i^\alpha} \sum_{j \in N_i^\alpha} (x_j^\alpha(t) - x_i^\alpha(t))}^{f_i} + \\
& \underbrace{+\varphi \nabla u_i(x_i^1(t), \dots, x_i^m(t))}_{g_i}
\end{aligned} \tag{5.7}$$

Si se elimina uno de los pasos, el resultado será uno no deseado.

- si $g_i = 0$ y se ejecuta un proceso de consenso, el acuerdo converge al promedio (ponderado) de los términos, que puede no coincidir con el punto máximo de utilidad.
- si $f_i = 0$ y el proceso está guiado por un ascenso de gradiente, los agentes permanecen en sus posiciones iniciales (o seguirán cualquier otra función de utilidad) aislados del resto.

En el caso de las funciones de utilidad seleccionadas (distribuciones gaussianas),

$$\nabla u_i(x_i) = \left(\frac{\partial u_i(x_i)}{\partial x_i^1}, \dots, \frac{\partial u_i(x_i)}{\partial x_i^m} \right) \tag{5.8}$$

y cada uno de los términos de ∇u_i puede calcularse fácilmente a partir de la Ecuación 5.5.

$$\frac{\partial u_i(x_i)}{\partial x_i^\alpha} = -\frac{x_i^\alpha(t) - x_i^\alpha(0)}{(1 - w_i^\alpha)^2} u_i(x_i) \tag{5.9}$$

La convergencia de este método depende del tamaño del escalón φ utilizado en el ascenso del gradiente. Debe cumplir con eso

$$\varphi \leq \min_i \frac{1}{L_{u_i}} \tag{5.10}$$

donde L_{u_i} es la constante de Lipschitz de cada función de utilidad. En el caso de una distribución gaussiana, es directo que el valor máximo del derivado aparece en su punto de inflexión, que se encuentra en $x_i^\alpha \pm (1 - w_i^\alpha)$. Sustituyendo en la Ec. 5.9

$$\begin{aligned}
\frac{\partial u_i(x_i)}{\partial x_i^\alpha} &= \frac{\partial u_i(x_i^\alpha - (1 - w_i^\alpha))}{\partial x_i^\alpha} = \\
&= -\frac{x_i^\alpha - (1 - w_i^\alpha) - x_i^\alpha(0)}{(1 - w_i^\alpha)^2} u_i(x_i) = \frac{1}{1 - w_i^\alpha} e^{-\frac{p}{2}}
\end{aligned} \tag{5.11}$$

Figura 5.3. Proceso de consenso combinado con gradiente sobre funciones de utilidad. Los valores iniciales son $x(0) = \{0,65 \ 0,75 \ 0,8 \ 0,5\}$ y $\sigma = 0,1$ para todos los nodos. El proceso de consenso converge a $x_i(t) \approx 0,75$, que es el punto donde está el máximo de la función de utilidad global U

y combinando todas las expresiones

$$L_{u_i} = \left(\sum_{\alpha} \frac{e^{-p/2}}{1 - w_i^{\alpha}} \right)^{1/2} = \|\nabla u_i(x_i)\|_2 \quad (5.12)$$

Finalmente, sustituimos las expresiones correspondientes en la Ecuación 5.7 obtenemos la ecuación final (Ecuación 5.13) que rige el proceso de consenso multidimensional. Utiliza la función de utilidad local para realizar el ascenso del gradiente y el valor actual de los vecinos directos para converger a un valor común casi óptimo.

$$x_i^{\alpha}(t+1) = x_i^{\alpha} + \frac{\varepsilon}{w_i^{\alpha}} \sum_{j \in N_i^{\alpha}} (x_j^{\alpha}(t) - x_i^{\alpha}(t)) - \frac{1}{\max_i \|\nabla u_i(x_i)\|_2} \cdot \frac{x_i^{\alpha}(t) - x_i^{\alpha}(0)}{(1 - w_i^{\alpha})^2} u_i(x_i) \quad (5.13)$$

Nótese que todos los términos en la Ecuación 5.13 se refieren a los valores del nodo i o sólo a los valores x de los vecinos directos de i (no se necesita ni siquiera la función de utilidad de los vecinos).

En la Figura 5.3(b) se aprecia como el proceso de consenso cuando se introduce el gradiente varía respecto al modelo de consenso solo (véase Figura 5.2).

Figura 5.4. Consenso en una red con varios máximos. Los valores iniciales son $x(0) = \{0,60,10,450,5\}$ y $\sigma = 0,1$ para todos los nodos. Tanto el proceso de consenso básico como el gradiente convergen a zonas subóptimas al tratar de converger a un único punto. U

En esta ocasión, el nodo $i = 4$ se ve atraído por el resto de los nodos y convergen a un valor que está en torno al máximo de la función de utilidad global U . Hay que recalcar que esta función no se conoce ni se calcula en ningún punto (salvo para representarla gráficamente) y el proceso combinado de consenso y gradiente es el que va tomando valores que convergen en el valor máximo. En este caso, en lugar de converger al valor medio de los valores iniciales $\bar{x} = 0,675$, la red converge a un valor $\bar{x} \approx 0,75$.

El resultado del proceso es una prioridad común y acordada para la alternativa evaluada en cada capa. Todas las alternativas pueden ser evaluadas al mismo tiempo usando un proceso de consenso independiente intercambiando un vector de preferencias.

Si la función de utilidad global es suave y todos los nodos tienen una utilidad $u_i > 0$ para cualquier decisión final, el método propuesto converge hacia la decisión óptima de la red. Sin embargo, si no hay un punto en el que todos los participantes tengan una utilidad positiva, la función de utilidad global resultante tiene al menos un máximo local adicional que puede alterar el proceso de convergencia. La Figura 5.4 muestra este caso cuando los valores iniciales son $x(0) = \{0,60,10,450,5\}$. En este caso, el nodo $i = 2$ se encuentra separado del resto lo suficiente como para .

En esos casos, permitimos que los nodos rompan los enlaces con aquellos vecinos que los están arrastrando a un área no deseada. Para hacer eso, es suficiente con romper las comunicaciones y dejar de intercambiar información

Figura 5.5. Consenso en una red con varios máximos. Los valores iniciales son $x(0) = \{0,6 \ 0,1 \ 0,45 \ 0,5\}$ y $\sigma = 0,1$ para todos los nodos. Tanto el proceso de consenso básico como el gradiente convergen a zonas subóptimas al tratar de converger a un único punto. U

con ellos. En este caso, la red puede dividirse en varios grupos, llegando cada uno de ellos a una decisión diferente. Es el caso que muestra la Figura 5.5.

En ese caso, es necesario definir una función que permita medir la utilidad alcanzada por la red. Si únicamente definimos la utilidad global como la suma de las utilidades, cuando los nodos se desplazan a un punto para alcanzar el consenso, siempre van a reducir su valor de utilidad. Esto ocurre porque en el instante inicial todos los nodos están en $x_i(0)$, que es el máximo de su función de utilidad. Cualquier otro valor de consenso hace que la utilidad disminuya, por ser una función monótona no decreciente $u_i(x_i(t)) \leq u_i(x_i(0)) \ \forall t > 0$.

La función de utilidad tiene que aumentar cuanto

1. más aumentan las utilidades individuales
2. menos grupos hay y mayor es el grupo mayoritario
3. más pequeña es la distancia entre los valores de los nodos

Estos tres factores están relacionados directamente con (1) la utilidad U , (2) la entropía E de los grupos que se forman, y (3) el tamaño de los grupos. Con estos criterios, se define la utilidad de la red cuando se permite la rotura de enlaces U_B como

$$U_B = \frac{\sum_i u_i}{(1 + \sum_{i,j} |x_i - x_j|)e^H} \quad (5.14)$$

Figura 5.6. Valor de utilidad de la solución. La línea verde es de utilidad cuando todos los nodos están en su máximo local, y la roja es el valor de utilidad cuando todos los nodos alcanzan el mismo acuerdo sobre el máximo global en una gran coalición.

Esta expresión es compatible con la formulación inicial de la utilidad, pues si no se rompen los enlaces todos los nodos acaban en un único valor de consenso, que tendrá entropía $E = 1$ y la suma de las diferencias entre los valores es cero, con lo que el denominador de la Ecuación 5.14 es uno y el valor de $U_B = U$. En la Figura 5.6 se puede ver la evolución de U_B a medida que el proceso de consenso converge. Cuando el nodo $i = 2$ detecta que su utilidad en el punto de consenso está por debajo de un mínimo, se desconecta de sus vecinos y el ascenso por gradiente hace que vuelva a su posición inicial. El resto de nodos continúa con el proceso de consenso combinado hasta que converge en el punto óptimo de la suma de sus utilidades.

La ventaja de este enfoque distribuido es que evita los problemas de cuellos de botella que surgen en las soluciones mediadas. Los agentes individuales no son conscientes de una solución global y definitiva, sino de la convergencia hacia un compromiso acordado entre sus vecinos cercanos. Además, el sistema es escalable ya que se pueden añadir nuevos nodos sin necesidad de notificaciones adicionales al resto de la red.

5.5. Caso de estudio: Procesos analíticos jerárquicos

El Proceso Jerárquico Analítico (AHP) es un método de optimización multiobjetivo. Los responsables de la toma de decisiones proporcionan evaluaciones subjetivas sobre la importancia relativa de los diferentes criterios y la preferencia de cada alternativa para cada criterio [IL11]. El resultado es un ranking de las opciones consideradas que incluye la puntuación relativa asignada a cada una de estas opciones. La principal ventaja de este proceso es que permite (i) organizar la información de manera eficiente y precisa, incluso para problemas complejos; y (ii) sintetizar y visualizar los efectos de los cambios en los niveles o preferencias. El PSA puede utilizarse para que un solo usuario tome una decisión, pero también para que un grupo de personas, como un comité o un grupo de expertos, llegue a un acuerdo conjunto. Sin embargo, estos enfoques asumen que todos los actores se conocen entre sí y pueden intercambiar información. Además, se suele introducir algún facilitador centralizado. Este trabajo propone un método para la toma de decisiones en grupo basado en AHP (GAHP), donde los participantes se conectan a través de una red e interactúan exclusivamente con sus vecinos directos.

El AHP comienza con la definición de los criterios utilizados para evaluar las alternativas, organizados de forma jerárquica. La importancia de cada criterio se define a través de su peso $w^\alpha \in [0, 1]$. Por ejemplo, supongamos que un nuevo líder tiene que ser elegido entre tres candidatos: Tom, Dick y Harry. La edad, experiencia, educación y carisma de estos candidatos son considerados para evaluarlos. La jerarquía de criterios y las ponderaciones asociadas a cada criterio α se muestran en la Figura 5.7. Una vez definidos los criterios, se crea una matriz de pares, asignando un valor relativo de juicio o preferencia a cada

Algoritmo 5.1

Algoritmo de consenso ejecutado por cada nodo a_i

- 1: **repeat**
 - 2: Recibe los valores de los vecinos en cada capa: $[x_j^1, x_j^2, \dots, x_j^n], layers = 1..n$
 - 3: Aplica el consenso para cada capa para obtener el nuevo valor $f(x_i^\alpha)$
 - 4: Apply gradient adjustment to all the new values for each layer of the agent x_i : $x_i^\alpha(t+1) = g(x_i^\alpha) + h(x_i^\alpha)$
 - 5: Send all the agent's layers values to all his neighbours: $[x_i^1, x_i^2, \dots, x_i^n], layers = 1..n$
 - 6: **until** the system converges or maxiter reached
-

Figura 5.7. Ejemplo de jerarquía de criterios para un AHP

Tabla 5.1

Matriz de prioridades locales

	Tom	Dick	Harry	prioridad (l_i^α)
Tom	1	1/4	4	0.217
Dick	4	1	9	0.717
Harry	1/4	1/9	1	0.066

Nota. Matriz de prioridades locales con la importancia relativa de cada candidato en relación a su experiencia.

par de alternativas. El valor a_{ij} representa la preferencia de la alternativa i sobre la j . Al todas las prioridades locales se sintetizan a través de todos los criterios para calcular la prioridad global final m_i para cada alternativa usando el promedio ponderado $m_i = \sum w^\alpha l_i^\alpha$.

El Proceso Jerárquico Analítico de Grupo (GAHP) es la generalización de AHP para un grupo de tomadores de decisiones. Cada agente tiene su matriz de juicio y sus prioridades finales y negocia para lograr una solución única y acordada. Existen principalmente dos métodos para agregar preferencias individuales: la agregación de juicios individuales (AIJ) y la agregación de prioridades individuales (AIP) [RG94]. La primera agrega la matriz de decisión completa, mientras que la segunda sólo agrega las prioridades. Ossadnik et al. [OSK16] hacen un análisis comparativo de los diferentes enfoques de GAHP. Otros métodos utilizados con éxito para resolver GAHP incluyen el trabajo de Srdjevic y Srdjevic,[SS13] o Dong y Saarly [DS14], en los que el proceso se centra en asegurar la consistencia de la solución. Huang et al. [HCLL13] combina juicios

Tabla 5.2

Prioridades finales para los candidatos

	exp.	edu.	car.	edad	total
Tom	0.119	0.024	0.201	0.015	<i>0.358</i>
Dick	0.392	0.010	0.052	0.038	0.492
Harry	0.036	0.093	0.017	0.004	<i>0.149</i>

Nota. Prioridades finales para cada criterio: experiencia (exp.), educación (edu.), carisma (car.) y edad. Dick es el candidato elegido.

y preferencias, lo que lleva a una decisión de grupo más satisfactoria. También se han utilizado otras técnicas, como el recocido simulado, tratando de resolver el GAHP cuando existen diferencias significativas entre las opciones [BSSZ16]. Finalmente, la palabra *consenso* aparece en otros trabajos relacionados con la toma de decisiones en grupo y GAHP, pero con un significado general, y no se refiere al método utilizado en este documento [DS14, WCFHV17].

En los siguientes apartados se presentan 3 ejemplos sintéticos con 9 nodos con diferentes configuraciones iniciales que van a tomar una decisión utilizando GAHP. Estos ejemplos ilustran que, en ocasiones, la mejor solución se alcanza cuando todos los agentes están de acuerdo (experimento 1). Sin embargo, en otras ocasiones es mejor dividir a todo el grupo en coaliciones más pequeñas, para que cada uno de los grupos pequeños conduzca a un consenso diferente y todo el sistema alcance una utilidad mayor (experimento 3). Se ha elegido un ejemplo bidimensional para poder representarlo gráficamente, por lo que sólo se consideran 2 criterios.

5.5.1. Experimento 1: solución conjunta

Este experimento presenta un conjunto de 9 nodos que llegan a una solución consensuada en la que todos están de acuerdo en dos criterios. La Figura 5.8(b) muestra las utilidades que definen las preferencias de cada nodo según la Ecuación. Con estas funciones se puede determinar la utilidad global y cómo los nodos están situados en los puntos que marca su estado inicial (Figura 5.8(b)). Cuando el proceso combinado de consenso y gradiente se detiene, todos los nodos han llegado al mismo punto, que corresponde a la decisión común acordada por la red (véase Figura 5.8(c)). Para que esta solución exista, la única condición es que todos los participantes tengan una utilidad positiva $u_i > 0$ a lo largo de todo el espacio de la solución.

Figura 5.8. (a) Funciones de utilidad locales u_i de cada nodo. (b) Función de utilidad global y situación inicial de la red. Esta función no se calcula nunca ni la conocen los nodos. (c) situación final de consenso en un punto

Figura 5.9. Evolución de los valores de cada criterio para cada uno de los participantes. La convergencia está garantizada si $\forall i u_i > 0$ en todo el espacio de la solución

La Figura 5.9 muestra la evolución del valor de cada criterio para cada uno de los participantes (en un color diferente) a lo largo del proceso. Convierte en la decisión final. Si estos valores se consideran como las coordenadas x y y , coinciden con el punto que corresponde a la solución en la Figura 5.8.

5.5.2. Experimento 2: consenso en situaciones subóptimas

Muestra cómo la evolución del sistema puede conducir a un consenso que no corresponde al valor óptimo cuando hay nodos cuya utilidad es 0 en la proximidad de la solución global. En ese caso, la forma de la función de utilidad global muestra picos y valles, con valores locales óptimos. En ese caso, la convergencia hacia la solución óptima no está garantizada y, como se muestra en la Figura 5.10, el proceso se detiene en cualquier valor, dependiendo de las preferencias y de la distribución de las funciones de la utilidad en el espacio de la solución.

5.5.3. Experimento 3: rotura de la red en acuerdos separados

La propuesta para resolver el problema mostrado en el experimento 2 es permitir la ruptura de los enlaces entre los nodos, de manera que no todos los nodos puedan estar de acuerdo con los mismos valores. Cuando un nodo detecta que la solución guía hacia un punto con cero utilidad, el nodo puede decidir romper el vínculo con aquellos vecinos que se están alejando de las preferencias. Como muestra la Figura 5.11, en este caso, la red está dividida

Figura 5.10. Ejemplo de convergencia hacia una solución subóptima porque los participantes no son capaces de converger en la mejor solución para el grupo ya que no tiene ninguna utilidad para algún nodo individual.

en grupos, cada uno de los cuales converge en un acuerdo diferente. La decisión óptima la toma el grupo formado por aquellos nodos cuya función de utilidad es positiva en la mejor solución. Esta solución se alcanza si los nodos con utilidad cero se acaban de retirar del sistema. A pesar de ello, permitimos que estos participantes tomen otra decisión formando un grupo separado.

El centro Figure 5.11 muestra la evolución de los criterios en tal caso. Se puede observar cómo se toma más de una decisión. En este caso, la red se divide en 2 grupos, cada uno en un valor máximo local.

Figura 5.11. Estados iniciales y finales de un proceso de AHP que permite romper los enlaces y volver a conectarse con los vecinos cercanos. Esta solución garantiza la convergencia de un subgrupo con la mejor decisión posible, junto con otros acuerdos sobre soluciones subóptimas.

5.5.4. Comparación con otras aproximaciones

En esta sección se muestran los resultados obtenidos con el método propuesto en comparación con otros enfoques de GAHP. Los resultados de la comparación fueron tomados del trabajo de Ossadnik et al.[OSK16]. El conjunto de experimentos está formado por 4 agentes que tienen que estar de acuerdo en los criterios de evaluación para una decisión. En este conjunto experimental, los agentes mantienen privada su matriz de juicio y negocian sobre los valores finales. Se incluyen dos versiones del proceso de consenso: consenso sobre los juicios (COJ) y sobre las prioridades (COP).

Los experimentos comparan nuestro método con los métodos examinados en la referencia mencionada anteriormente. Estos métodos son (i) la agregación de juicios individuales (AIJ), (ii) la agregación de prioridades individuales (AIP), (iii) el enfoque de la función de pérdidas (LFA), y (iv) GAHP con diferencias y clasificaciones preferenciales (PDR). En todos los casos se consideran medios aritméticos y geométricos (excepto GAHP).

Los escenarios 1 y 2 evalúan la optimalidad de Pareto, que dice que si todos los miembros del grupo prefieren la alternativa A_1 a la alternativa A_3 , entonces el grupo debería preferir A_3 también. La diferencia entre ambos escenarios es que, en el escenario 2, A_1 y A_3 están más cerca uno del otro (ver los valores de DM_1 en la Tabla 5.3).

Otra medida es la condición de homogeneidad, que dice que si cada individuo juzga una alternativa A_1 μ -veces más grande que otra alternativa A_2 , entonces la sentencia sintetizada A_1 debería ser μ -veces más grande que otra alternativa A_2 también. Los escenarios 3 y 4 evalúan esta propiedad. En el escenario 3, todos los agentes prefieren A_1 a A_3 con un factor $\mu_{1,3} = 2$. En el escenario 4, todos los agentes juzgan la alternativa A_4 como 1,6 veces más preferible que A_3 , de modo que $\mu_{4,3} = 1,6$. La tabla muestra los resultados relativos a la condición de homogeneidad.

Los resultados muestran que sólo el AIP (con media aritmética o geométrica) y nuestra propuesta (COP y COJ) funcionan correctamente en todos los casos, asegurando la optimización de Pareto y la homogeneidad de los resultados. El COJ falla en la condición de homogeneidad para el escenario 4, con un valor de 1,57 (en lugar de 1,6). Sin embargo, es el mejor de los resultados que falla esta prueba y se acerca lo suficiente a la solución correcta.

En un segundo conjunto experimental, comparamos nuestra propuesta con el método de consenso grupal propuesto por Dong y Saarly [DS14]. En este caso, la palabra consenso se utiliza en un sentido general de acuerdo entre los participantes, con diferencias significativas con el método propuesto. Los resultados de Dong se comparan en el artículo con otros ejemplos bien conocidos,

Tabla 5.3

Optimalidad de Pareto

Método <i>Pref. individuales</i>	Escenario	A_1	A_2	A_3	A_4	Opt. Pareto
DM_1	1	0.236	0.418	0.164	0.181	$A_1 > A_3$
	2	0.161	0.421	0.159	0.259	$A_1 > A_3$
DM_2	1	0.490	0.127	0.173	0.210	$A_1 > A_3$
	2	0.485	0.140	0.204	0.174	$A_1 > A_3$
DM_3	1	0.238	0.262	0.063	0.437	$A_1 > A_3$
	2	0.190	0.248	0.104	0.458	$A_1 > A_3$
Método <i>Pref. grupales</i>	Escenario	A_1	A_2	A_3	A_4	Opt. Pareto ... satisfecho?
AIJ(WAMM)	1	0.290	0.321	0.133	0.254	$A_1 > A_3$ Sí
	2	0.181	0.353	0.184	0.281	$A_1 < A_3$ No
AIJ(WGMM)	1	0.279	0.344	0.130	0.246	$A_1 > A_3$ Sí
	2	0.164	0.370	0.190	0.276	$A_1 < A_3$ No
AIP(WAMM)	1	0.312	0.300	0.146	0.241	$A_1 > A_3$ Sí
	2	0.264	0.302	0.161	0.272	$A_1 > A_3$ Sí
AIP(WGMM)	1	0.318	0.288	0.149	0.244	$A_1 > A_3$ Sí
	2	0.252	0.297	0.172	0.279	$A_1 > A_3$ Sí
LFA(WAMM)	1	0.253	0.343	0.128	0.275	$A_1 > A_3$ Sí
	2	0.177	0.370	0.165	0.288	$A_1 > A_3$ Sí
LFA(WGMM)	1	0.289	0.343	0.133	0.234	$A_1 > A_3$ Sí
	2	0.165	0.375	0.190	0.269	$A_1 < A_3$ No
PDR	1	0.370	0.264	0.041	0.325	$A_1 > A_3$ Sí
	2	0.329	0.276	0.063	0.331	$A_1 > A_3$ Sí
COJ	1	0.238	0.262	0.063	0.436	$A_1 > A_3$ Sí
	2	0.288	0.365	0.152	0.193	$A_1 > A_3$ Sí
COP	1	0.313	0.299	0.146	0.241	$A_1 > A_3$ Sí
	2	0.505	0.048	0.312	0.174	$A_1 > A_3$ Sí

así que hemos usado estos resultados para comparar nuestro método también.

La matriz de juicios final (G_C) con un ratio de consistencia $CR = 0,0449$

Tabla 5.4

Homogeneidad

Método <i>Pref. individuales</i>	Escenario	A_1	A_2	A_3	A_4	homogeneidad
DM_1	3	0.332	0.338	0.166	0.164	$\mu_{1,2} = 2,00$
	4	0.264	0.364	0.143	0.229	$\mu_{4,3} = 1,60$
DM_2	3	0.458	0.119	0.229	0.194	$\mu_{1,2} = 2,00$
	4	0.445	0.139	0.160	0.257	$\mu_{4,3} = 1,60$
DM_3	3	0.205	0.257	0.102	0.436	$\mu_{1,2} = 2,00$
	4	0.232	0.290	0.183	0.295	$\mu_{4,3} = 1,60$
Método <i>Pref. grupales</i>	Escenario	A_1	A_2	A_3	A_4	¿homogeneidad ... satisfecha?
AIJ(WAMM)	3	0.290	0.271	0.212	0.228	$\mu_{1,3} = 1,37$ No
	4	0.268	0.304	0.149	0.279	$\mu_{4,3} = 1,88$ No
AIJ(WGMM)	3	0.302	0.311	0.171	0.216	$\mu_{1,3} = 1,77$ No
	4	0.268	0.320	0.145	0.267	$\mu_{4,3} = 1,84$ No
AIP(WAMM)	3	0.344	0.256	0.172	0.228	$\mu_{1,3} = 2,00$ Sí
	4	0.312	0.282	0.156	0.250	$\mu_{4,3} = 1,60$ Sí
AIP(WGMM)	3	0.352	0.248	0.176	0.223	$\mu_{1,3} = 2,00$ Sí
	4	0.312	0.270	0.161	0.258	$\mu_{4,3} = 1,60$ Sí
LFA(WAMM)	3	0.273	0.293	0.163	0.271	$\mu_{1,3} = 1,68$ No
	4	0.243	0.331	0.139	0.286	$\mu_{4,3} = 2,05$ No
LFA(WGMM)	3	0.287	0.322	0.171	0.220	$\mu_{1,3} = 1,68$ No
	4	0.266	0.330	0.146	0.259	$\mu_{4,3} = 1,77$ No
PDR	3	0.352	0.238	0.073	0.337	$\mu_{1,3} = 4,83$ No
	4	0.371	0.267	0.074	0.288	$\mu_{4,3} = 3,91$ No
COJ	3	0.331	0.338	0.165	0.166	$\mu_{1,3} = 2,00$ Sí
	4	0.199	0.330	0.183	0.287	$\mu_{4,3} = 1,57$ (*)
COP	3	0.362	0.423	0.181	0.032	$\mu_{1,3} = 2,00$ Sí
	4	0.208	0.260	0.203	0.328	$\mu_{4,3} = 1,60$ Sí

obtenida por consenso es

$$G_C = \begin{pmatrix} 1 & 3,9978 & 5,1335 & 6,1379 \\ 0,2501 & 1 & 3,0022 & 3,1379 \\ 0,1948 & 0,3331 & 1 & 2,0000 \\ 0,1629 & 0,3187 & 0,5000 & 1 \end{pmatrix}$$

Tabla 5.5

Prioridades mediante consenso sobre juicios

método	prioridades finales
Dong et al (2010)	$w = (0,5416, 0,2597, 0,1299, 0,0688)^T$
Wu & Zu (2012)	$w = (0,5437, 0,2722, 0,1172, 0,0669)^T$
Dong & Saarly al (2014)	$w = (0,5919, 0,2423, 0,1021, 0,0637)^T$
COJ	$w = (0,6000, 0,2222, 0,1067, 0,0711)^T$

Nota. Prioridades finales obtenidas por los diferentes enfoques indicados y nuestro propio método (COJ), Las prioridades están el mismo orden de magnitud y la ordenación obtenida no presenta ninguna variación con respecto a los estudios de referencia.

La Figura 5.12 muestra la diferencia entre el uso de la media geométrica y el valor de consenso. Cuando se utiliza el consenso, la solución acordada converge a un valor máximo en la función de utilidad global. La solución fundada por consenso tiene un mejor valor para la comunidad, en lugar de sacrificar la mejor opción para tener en cuenta las opciones con un menor valor de los participantes marginales.

En este capítulo se ha presentado un método basado en una combinación de consenso y ascenso de gradiente y se ha aplicado para resolver GAHP en un entorno descentralizado, donde los participantes en el proceso de toma de decisiones intercambian sus preferencias con sus vecinos directos para llegar a un acuerdo que permita al equipo seleccionar la alternativa de mayor utilidad global.

El modelo permite consensuar no sólo las prioridades, sino también los juicios. Se ha comprobado que las soluciones propuestas cumplen con la condición de optimalidad de Pareto y homogeneidad. Además, obtiene soluciones similares a las de otros enfoques, pero de forma distribuida, donde todos los agentes participantes mantienen sus preferencias privadas y únicamente intercambian sus valores.

Por lo general, los métodos GAHP convergen a la media geométrica, mientras que esta propuesta converge a la utilidad global máxima manteniendo la relación de consistencia por debajo de 0,1. Los métodos distribuidos existentes utilizan un moderador para guiar el proceso de decisión del grupo, pero nuestro método es totalmente descentralizado, evitando el uso de este elemento. Además, en los casos en que la función de utilidad global presenta puntos máximos locales, nuestro método sugiere la división del grupo en diferentes coaliciones.

Figura 5.12. Prioridades finales calculadas por consenso (en azul) o por la media geométrica (en rojo). El consenso converge hacia el valor de utilidad global más alto

Capítulo 6

Consenso con nodos maliciosos

6.1. Introducción

Hasta ahora se ha asumido que todos los nodos de la red deben seguir exactamente el proceso de consenso de forma obligatoria para que el resultado sea correcto. Para alterar el resultado, basta con que uno de los nodos altere el valor que intercambia en una de las iteraciones para que el valor de consenso se desvíe del valor esperado. Existen mecanismos para detectar cuándo se produce una variación, pero requieren guardar el estado completo de la red en cada iteración y además es necesario mantener una visión global de la red y conocer su estructura, pues se basan en matrices de observación que se calculan a partir de la matriz de adyacencia de la red completa [SSH11]. Otras alternativas requieren la existencia de un nodo que hace las funciones de coordinador [Raf17]. También están relacionados otros trabajos enfocados a reducir el efecto del ruido en redes de sensores, pero sus soluciones están centradas en obtener la topología de red óptima para minimizar el efecto del ruido [MZ10, WT16]

En este capítulo se propone un método completamente descentralizado que no requiere información global de la red y que es independiente de la topología, si bien es cierto que existen determinadas restricciones que se deben cumplir para poder anular por completo las desviaciones producidas.

Para mostrar su utilidad, se aplica a tres casos: un mecanismo de voto distribuido, la difusión de imágenes a través de una red y el problema de los generales bizantinos, que es la base de las cadenas de bloques. En todos ellos nos centraremos en cómo detectar y corregir las posibles manipulaciones sobre el proceso de consenso para alterar el resultado de la votación, modificar la imagen para que no se clasifique correctamente (usando ejemplos adversarios–

Figura 6.1. Introducción de una desviación en el proceso de consenso. Inicialmente $x(0) = \{0,2 \ 0,4 \ 0,6 \ 0,8\}$. En la iteración $t = 5$ el nodo $i = 2$ añade 0,1 a su valor ($u_2(5) = 0,1$), lo que hace que la red converja a $\bar{x} = 0,525$.)

adversarial examples—) o evitar la difusión de mensajes falseados en el caso de los generales bizantinos.

6.2. Detección de desviaciones en el proceso de consenso

El proceso de consenso definido en la Ecuación 2.1 asume que todos los participantes ejecutan el mismo algoritmo. Si un nodo desea alterar el resultado basta con enviar a los vecinos cualquier otro valor. Por ejemplo, si un nodo i envía un valor fijo $x_i(t) = x_i(0) \ \forall t$, la red completa convergerá a dicho valor $x_i(0)$. En general, podemos modelar el proceso de consenso con desviaciones como

$$x_i(t+1) = x_i(t) + \varepsilon \sum_{j \in N_i} [x_j(t) - x_i(t)] + u_i(t) \quad (6.1)$$

donde $u_i(t)$ es el error introducido por el nodo i en la iteración t . Por ejemplo, en la Figura 6.1 podemos ver qué ocurre cuando el nodo $i = 2$ añade una desviación de 0,1 en la iteración $t = 5$, es decir, $u_2(5) = 0,1$. En este caso, se ve claramente el pico del cambio del valor y cómo la red converge a la nueva media $\bar{x} = 0,525$, como si el valor inicial del nodo hubiera sido $x_2(0) = 0,6$ en lugar de su valor original $x_2(0) = 0,4$

La pregunta que se plantea es si los vecinos de i pueden detectar cuándo $u_i(t) \neq 0$. Despejando la Ecuación 2.1 podemos obtener la siguiente expresión:

$$dv_i(t+1) = x_i(t+1) + (\varepsilon d_i - 1)x_i(t) - \varepsilon \sum_{j \in N_i} x_j(t) \quad (6.2)$$

siendo d_i el grado del nodo i . Si los nodos de la red siguen el algoritmo, cada iteración debe cumplir

$$dv_i(t) = 0 \quad \forall t \geq 0 \quad (6.3)$$

y cuando $dv_i(t) > 0$ el nodo puede cuestionar la validez de los datos recibidos. La primera parte de la expresión depende solo de los valores de i y el último término es la suma de los valores recibidos en la iteración anterior, así que basta con guardar esta suma para controlar el proceso. La única restricción del método es que el primer intercambio debe ser el real, es decir, el valor inicial no contendrá errores.

De esta manera se detecta que el resultado obtenido en el proceso de consenso no es correcto. La siguiente pregunta que se plantea es ¿puede corregirse esta desviación?

6.3. Corrección de las desviaciones

Una primera aproximación es compensar la desviación cuando se detecta en los nodos de la red. Vamos a asumir, sin perder generalidad y para facilitar la explicación, que la desviación se produce una única vez y en uno de los nodos i . Todos sus vecinos $j \in N_i$ lo detectan según (6.2), con $dv_j(t) > 0$. En la siguiente iteración, estos nodos descuentan la desviación de su valor calculado, haciendo

$$z_j(t+1) = x_j(t+1) - dv_j(t+1) \quad (6.4)$$

siendo $z_j(t+1)$ el valor a intercambiar en la siguiente iteración. Si todos los nodos se comportan de esta forma, la suma se conserva y además se puede demostrar que $u(t) = \sum_i dv_i(t+1)$. El principal problema de esta solución es que el nodo que introduce la desviación debe colaborar para compensar el resultado, lo que no es probable.

De (6.2) se puede deducir cómo se reparte el exceso entre los vecinos. El nodo tramposo i se guarda $(1 - \varepsilon d_i)u_i(t)$ para él mismo y reparte $\varepsilon u_i(t)$ para cada uno de sus vecinos, luego

$$dv_j(t+1) = \varepsilon \sum_{k \in N_j} u_k(t), \quad \frac{dv_j(t+1)}{\varepsilon} = \sum_{k \in N_j} u_k(t) \quad (6.5)$$

Dividiendo la desviación obtenida en el nodo por el factor ε , cualquier nodo conoce cuál es el error total que se ha introducido en el consenso. Sin embargo, dado que los valores de $\sum_{k \in N_j} u_k(t)$ están agregados, lo que no es posible es determinar qué nodo ha introducido el error.

Esta solución permite a los vecinos del nodo con la desviación saber que algo está mal en el proceso de consenso y corregirlo. Para que esta corrección llegue a todos los nodos se debe propagar también por la red, por lo que se efectúa un proceso de consenso paralelo sobre $(D_i(t)|w_i(t))$ donde $D_i(t) = \sum_{s=0}^t dv_i(s)$ es la suma de las desviaciones detectadas por el nodo i , y $w_i(t) = 1$ si $dv_i(t) \neq 0$ en la iteración t y cero en otro caso (de forma que a través de w podemos determinar cuántos nodos han detectado la desviación). Este proceso se realiza en paralelo al realizado sobre la variable $x(t)$, pero solo los nodos confiables la realizan. Cuando el proceso converge, el valor alterado obtenido en $x(t)$ puede corregirse de la siguiente manera

$$c_i(t) = x_i(t) - \frac{D_i(t)/\varepsilon}{w_i(t)} \quad (6.6)$$

y (6.6) da el resultado final de la aplicación de un proceso de consenso acumulado con la Ecuación (6.9) y la corrección calculada en la Ecuación (6.5). La Figura 6.2 muestra cómo se produce la propagación de la corrección y cada nodo obtiene el valor $dv_i(t)$ que le permite corregir el resultado desviado al que ha llegado el consenso (véase Figura 6.1).

El algoritmo final que ejecuta cada nodo de la red puede resumirse como

Consenso con nodos tramposos

- 1: $x_i(0) = (x_i^1, \dots, x_i^m \mid 0)$
 - 2: **while** $x_i(t)$ no converge **do**
 - 3: $x_i(t+1) = x_i(t) + \sum_{j \in N_i} [x_j(t) - x_i(t)] + u_i(t)$
 - 4: $dv_i(t) = x_i(t+1) + (\varepsilon d_i - 1)x_i(t) - \varepsilon \sum_{j \in N_i} x_j(t)$
 - 5: acumula las desviaciones $dv_k(t)$ en $D_i(t)$
 - 6: $D_i(t+1) = D_i(t) + \sum_{j \in N_i} [D_j(t) - D_i(t)]$
 - 7: **end while**
 - 8: $c_i^k(t) = x_i^k(t) - \frac{D_i^k(t)/\varepsilon}{w_i^k(t)} \quad \forall k$
 - 9: el resultado es $x_i(t) - c_i(t)$
-

Figura 6.2. Corrección de las desviaciones. En la situación de la Figura 6.1, los decinos de nodo $i = 2$ detectan la desviación y comienzan su propagación. El consenso sobre dv converge a $\bar{dv} = 0,015$ y aplicando la Ecuación 6.6 conseguimos calcular el exceso que se ha introducido en la red.)

Es importante destacar que este método no impide la manipulación ni lo corrige cuando lo detecta. El proceso converge al valor manipulado. Sin embargo, cuando finaliza el proceso, los nodos disponen de la información necesaria para saber si se ha producido alguna desviación de los datos y cómo corregirla.

6.4. Casos de estudio

A continuación se van a presentar tres casos de aplicación de la detección y corrección de fallos a problemas para los que disponer de sistemas robustos y seguros es de vital importancia: (i) un sistema de votación electrónica, (ii) la detección de imágenes manipuladas con ejemplos adversarios y (iii) el problema de los generales bizantinos, base de las cadenas de bloques.

6.4.1. Votación distribuida

Supongamos que se desea obtener el resultado de un votación que se realiza de forma distribuida en una red. Supongamos que se dispone de m opciones y cada nodo solo puede votar a una de ellas. El proceso de consenso consistirá en intercambiar con los vecinos un vector $x_i = (x_i^1, \dots, x_i^m)$, donde $x_i^k = 1$ si k es la opción seleccionada por i and $x_i^j = 0 \forall j \neq k$. Es decir, el vector contiene un 1

Tabla 6.1

Configuración de $x_i(0)$ para votaciones

	x_i^1	x_i^2	x_i^3	y_i
x_0	0	0	0	1
x_1	0	0	1	0
x_3	1	0	0	0
x_4	0	0	1	0
x_5	0	1	0	0
x_6	1	0	0	0
x_7	1	0	0	0
x_8	0	1	0	0
x_9	0	0	1	0
x_{10}	0	0	1	0

Nota. Los nodos del 1 al 10 indican con un uno la opción votada y el resto se queda a cero. Un nodo ficticio X_0 se inicializa con todas las opciones a cero (no vota) y marca como uno la variable auxiliar para contar el número de participantes..

en la posición de la opción votada y ceros en todas las demás (véase Tabla 6.1). Cuando se realiza el proceso de consenso, el resultado converge al valor medio de la votación. Para obtener el resultado general, basta con multiplicar el resultado por el número de nodos, que puede obtenerse de forma sencilla con un proceso de consenso en paralelo inicializado de una forma particular.

El proceso de consenso converge a la media de los valores iniciales. Si $\sum_i y_i(0) = 1$, el proceso de consenso convergerá para algún instante de tiempo t a $y_i(t) = \frac{1}{n}$ por lo que el tamaño de la red $n = \frac{1}{y_i(t)}$. Ambos procesos de consenso son independientes y pueden ejecutarse en paralelo.

Extendemos el vector de votos con una columna adicional

$$(x_i|y_i) = (x_i^1, \dots, x_i^m | y_i) \quad (6.7)$$

e, inicialmente, $y_i = 0 \forall i$. Sin perder generalidad, podemos introducir un nodo adicional en la red cuyos valores iniciales sean

$$(x_0|y_0) = (\underbrace{0, \dots, 0}_m | 1) \quad (6.8)$$

Este nodo no afecta al resultado de la votación puesto que no vota por ninguna opción. Cuando el proceso de consenso converge, basta con que cada nodo

Figura 6.3. Evolución del proceso de consenso para calcular el resultado de una votación de forma distribuida. red aleatoria con $n = 10$ nodos y $m = 3$ opciones. Los votos son $x(0) = \{3, 3, 1, 3, 2, 1, 1, 2, 3, 3\}$, lo que da un resultado de $\{3, 2, 5\}$.

divida los valores finales x_i por el valor obtenido en la última columna y_i para determinar el resultado general de la votación tal y como muestra (6.9)

$$\begin{aligned} \frac{x_i(t)}{y_i(t)} &= \frac{(x^1, \dots, x^m)}{1/n} = (nx^1, \dots, nx^m) \\ &= \left(\sum_j x_j^1, \dots, \sum_j x_j^m \right) \end{aligned} \quad (6.9)$$

La Figura 6.3 muestra un ejemplo de la aplicación de este proceso sobre una red aleatoria de diez nodos. Los nodos eligen entre 3 opciones y la elección de cada uno es $x(0) = \{3, 3, 1, 3, 2, 1, 1, 2, 3, 3\}$. Por lo tanto, $(x_1|y_1) = (0, 0, 1|0)$, $(x_2|y_2) = (0, 0, 1|0)$, $(x_3|y_3) = (1, 0, 0|0)$, y así sucesivamente. Las líneas muestran la evolución del cociente $\frac{x_i^k(t)}{y_i(t)}$. Puede observarse que la red converge al resultado de la votación $\{3, 2, 5\}$.

Supongamos que el nodo $i = 7$ desea añadir dos votos a la primera opción y eliminar un voto de la tercera: $u_7 = (2, 0, -1)$. El resultado del consenso convergerá a $\{5, 2, 4\}$ y la opción ganadora sería la primera, tal y como deseaba el nodo 7 (ver Figura 6.4).

Aplicando (6.2), la red puede detectar esta manipulación. La Figura 6.5 muestra el proceso de consenso sobre los valores de $dv_i(t)$. Para corregir las desviación en el proceso de voto, el consenso de las desviaciones también se

Figura 6.4. El nodo $i = 7$ añade 2 votos a la primera opción y elimina un voto de la tercera en la iteración $t = 6$. El resultado alterado converge a $\{5, 2, 4\}$.

extiende con un valor w_i adicional que en este caso se emplea para determinar cuántos nodos han detectado la desviación. El resultado obtenido al aplicar (6.5) debe dividirse por este valor w_i . En el caso de la Figura 6.5, $\varepsilon = 0,1650$ y la manipulación es detectada por cuatro nodos, lo que arroja un valor de $w = 0,4$, con lo que la desviación detectada a partir de los valores de $dv(t)$ obtenidos será $(2, 0, -1)$ en todos los nodos.

De esta forma, al terminar el proceso de consenso, aunque el resultado esté manipulado, todos los nodos de la red tienen la información que les permite corregirlo.

6.4.2. Alteración de imágenes con ejemplos adversarios

Un segundo caso de aplicación es la transmisión de imágenes a través de una red. Asumiremos que hay un nodo que tiene la imagen (por ejemplo, una imagen capturada en una cámara, como la matrícula de un coche) y quiere propagarla a todos los demás. Sin perder generalidad, será $i = 1$.

La representación de la imagen será a través de un vector que contiene el mapa de bits, con tantos elementos como resolución tiene la imagen y con el valor de RGB de cada pixel. El proceso de consenso se debe modificar de forma semejante al apartado anterior. A dicho vector, se le añade una columna y_i que valdrá cero para todos los nodos excepto para el que contenga la imagen, para el que $y_1 = 0$. Si la imagen contiene p píxeles, $(x_1|y_1) = (x_1^1, \dots, x_1^p | 1)$ y para el resto de nodos $(x_i|y_i) = (0, \dots, 0 | 0)$. Cuando el proceso de consenso

Figura 6.5. El proceso de consenso sobre las desviaciones detectadas converge a $dv(t) = (0,1320, 0, -0,0660)$. Aplicando (6.5) adaptada a votaciones, con $\varepsilon = 0,1650$ y $w = 0,4$ se obtiene la corrección deseada $\frac{dv/\varepsilon}{w} = (2, 0, -1)$

Figura 6.6. Ejemplo adversario para engañar a los algoritmos de aprendizaje profundo. Si a la imagen (a) se le añade la perturbación (b) multiplicada por un factor de 0,07, da como resultado una imagen que es erróneamente clasificada como un gibón (c)

converge, basta con dividir el valor obtenido para cada pixel entre $y_i(t)$ para recuperar la imagen original.

Sin embargo, un nodo malicioso puede tratar de modificar la imagen para que el sistema no la reconozca o la identifica erróneamente. Una de las técnicas empleadas se conoce como ejemplos adversarios [GSS14] y consiste en añadir a la imagen una ligera distorsión de manera que los algoritmos de aprendizaje profundo fallen y clasifiquen la imagen de forma incorrecta. En el ejemplo que

muestra la Figura 6.6, la imagen inicial (6.6(a)) se clasifica correctamente como un oso panda. Sin embargo, al añadirle una capa a la imagen con un ruido (6.6(b)) que parece aleatorio y con una intensidad muy baja (multiplicada por un factor de 0.07), el resultado es indistinguible para una persona de la imagen original. En cambio, los primeros clasificadores que empleaban técnicas de aprendizaje profundo lo clasificaban erróneamente como un mono de la familia de los gibones (6.6(c)).

En este caso, el nodo $i = 5$ aplica una perturbación a la imagen en la iteración $t = 20$, tal y como muestra la Figura 6.6. A la perturbación se le aplica un factor de 0.007, que es suficiente para hacer que los clasificadores que no contemplan *adversarial example* fallen, aunque para el ojo humano parece que la imagen no ha sido alterada. La Figura 6.7 muestra el proceso de consenso, donde el valor es la suma del mapa de bits asociado a la imagen. Se aprecia la perturbación introducida. Cuando converge, todos los nodos tienen la imagen alterada.

Figura 6.7. Alteración del consenso en la iteración 20 para incluir la imagen modificada con ejemplos adversarios. El valor de x_i es la suma del mapa de bits asociado a la imagen

Pero de la misma forma que en el caso de la votación, la alteración se detecta inmediatamente por los vecinos utilizando (6.5) y esta desviación respecto a la imagen original se puede corregir cuando el proceso ha finalizado.

En esta ocasión, vamos a relajar la corrección, de manera que no deseamos

recuperar la imagen original, si no simplemente tratar de paliar el efecto de la perturbación para que la imagen vuelva a ser clasificada correctamente. El resultado que muestra la Figura 6.8 se ha obtenido haciendo un consenso sobre las desviaciones dv_i obtenidas. De esta forma, la parte de la perturbación que se queda el nodo que modifica la imagen $\varepsilon(1 - d_i)u_i(t)$ no se puede recuperar.

Figura 6.8. Imagen corregida una vez ha concluido el proceso de consenso. Se clasifica como panda con una precisión del 0.9986 (googleNet)

Pero como se pone de manifiesto den lo resultados de la Tabla 6.2, aunque no se recupere la imagen original, la corrección es lo suficientemente buena como para que la imagen vuelva a ser correctamente clasificada.

Los métodos actuales suelen incorporar bibliotecas que consideran el tratamiento ejemplos adversarios, por lo que frecuentemente el único efecto que tiene es disminuir la precisión del clasificador en la imagen seleccionada, pero sigue clasificándose de forma correcta. La imagen corregida por consenso muestra un pixelado mayor, perceptible a simple vista (véaseFigura 6.8). Sin embargo, la precisión que obtiene por el clasificador es equivalente a la imagen original. La implementación de los distintos algoritmos corresponden a las versiones de Matlab para la *Neural Network Toolbox*. En todos los casos excepto VGG-16, las imágenes son clasificadas como panda gigante, la imagen modificada obtiene una menor precisión y la imagen corregida, sin ser una corrección total, logra resultados comparables a la imagen original. En el caso cd VGG-16, se observa el comportamiento detectado en el trabajo de Goodfellow [GSS14] y la imagen manipulada se clasifica como un gibón. La corrección vuelve a clasificarse correctamente, y esta vez con una precisión notablemente superior. La Tabla 6.2 muestra la precisión de la imagen original, la alterada mediante ejemplo adversario y la imagen corregida durante el proceso de consenso.

Tabla 6.2

Precisión de la clasificación de imágenes por distintos algoritmos

	Google Net	Res-Net 50	Alex Net	VGG-16
original	0.9873	0.9844	0.7479	0.5715 (panda)
alterada	0.9313	0.9429	0.6887	0.4439 (gibón)
corregida	0.9986	0.9863	0.9220	0.8302 (panda)

6.4.3. Problema de los generales bizantinos

Un tercer caso de aplicación de particular interés: el problema de los generales bizantinos. La primera solución la plantean Lamport et al. [LSP82] y propone una solución basada en prueba de trabajo (*proof of work* –*POW*–) como las que incorporan en la actualidad los algoritmos para cadenas de bloques y algunas criptomonedas, como bitcin.

El problema de los generales bizantinos consiste en lo siguiente. Supongamos que un conjunto de generales estn sitiando una ciudad y tienen que coordinarse para atacar. Si no lo hacen todos al mismo tiempo, fracasarn. Dentro del grupo hay algunos generales traidores que tratarn de prevenir el acuerdo. Para poder atacar, uno de los generales mandar un mensaje al resto (los llamaremos tenientes para diferenciarlos) con la hora a la que van a atacar o indicando si se van a retirar. Cada teniente leal transmitir el mensaje tal y como lo ha recibido, Pero los traidores alterarn el mensaje indicando una hora distinta. Por ejemplo, si el ataque es a las 9:00, un general traidor mandar un mensaje para que el ataque sea a las 8:00 Cuando lleguen las 8:00, los generales que hayan sido engnados atacarn y, al no ser suficientes, perdern. Luego, los generales restantes atacarn a las 9:00 y de nuevo perdern por no ser suficientes.

Cmo puede el algoritmo de consenso resolver este problema? En primer lugar, veamos cmo se pueden sincronizar mediante consenso. Supongamos que no hay traidores. La inicializacin ser la siguiente: el general (nodo $i = 1$) la hora de ataque como valor inicial $x_1(0) = 8$ y la columna adicional con $y_1 = 1$, con un vector extendido $(x_1|y_1) = (8 | 1)$. El resto de nodos se inicializan a cero: $(x_i|y_i) = (0 | 0), \forall i \neq 1$. Una vez alcanzada la convergencia, todos los generales sabrn la hora de ataque, que ser $\frac{x_i(t)}{y_i(t)} = 8$.

Si hay algn traidor, como muestra la Figura 6.10(a), la red converge al valor modificado. En este caso, el nodo $i = 15$ a incrementado en una la hora

Figura 6.9. Problema de los generales bizantinos con 4 generales y con un mensaje simple de atacar o retirarse. Los generales solo atacarán si están seguros de que van a hacerlo todos.

de ataque y la red converge a 9.

En este caso necesitamos obtener la hora precisa, por lo que es necesario hacer una corrección completa siguiendo (6.5). De esta forma, los generales determinan que la hora recibida de ataque tiene una desviación de 1 hora adicional sobre la hora que contenía el mensaje original, luego la hora de ataque serán las 8:00 (Figura 6.10).

6.4.4. Consenso como alternativa a *blockchain* en IoT

Finalmente, mostraremos la aplicación de la detección de fallos en el proceso de consenso a un escenario real: la tecnología de libro mayor distribuido (DLT—*distributed ledger technology*—). Los libros de contabilidad distribuidos registran, replican, comparten y sincronizan los datos sin un administrador central y sobre la base de redes igual a igual (*peer-to-peer*). Las cadenas de bloques (*blockchain*) es una posible utilidad del DLT. La base de datos se distribuye entre un conjunto de nodos de una red. Cada uno de ellos conserva una copia idéntica del libro de contabilidad. Cuando un nodo genera un nuevo bloque de datos, se extiende a toda la red y los nodos votan para decidir qué copia es la correcta.

Para aplicar el proceso de consenso presentado en este trabajo, asumimos que sólo un agente tiene la autoridad para emitir bloques con la información que el agente genera. Por ejemplo, para la IO, sólo el sensor crea un bloque con sus lecturas, o una institución académica genera certificados de sus estudiantes, o una agencia es la que firma los contratos. El problema que surge en esos

Figura 6.10. Consenso para el problema de los generales bizantinos (a) Proceso de consenso para determinar la hora de ataque en una red aleatoria de 50 nodos. El nodo $i = 15$ es un traidor y cambia la hora de ataque de las 8:00 a las 9:00. (b) Evolución del factor de corrección $dv_i(t)$. El valor de convergencia es de 0.0085 y aplicando (6.5) se obtiene que la diferencia con la hora de ataque tiene un exceso de una hora

escenarios es cómo asegurar que los participantes intercambien los datos tal como están.

La estructura elegida que almacena los bloques de datos en cada agente son los árboles de Merkle. En estos árboles, los nodos terminales (hojas) son los bloques que contienen los datos, y los *hash* de los nodos internos se calculan a partir de los *hash* de sus hijos (ver Figura 6.11). Para nuestra propuesta

- cada agente se queda con una copia del árbol de Merkle,
- cada agente emite bloques con la información generada (lecturas del sensor)
- el resto de los nodos distribuyen el bloque y lo almacenan en su propio árbol de Merkle

La solución adoptada para el BGP es válida para este escenario. Consideremos una red de sensores con agentes que mantienen el registro de las lecturas y utilizan DLT con árboles de Merkle para distribuir y sincronizar los datos entre los sensores mediante un proceso de consenso.

1. el nodo i crea un bloque con todas sus lecturas s

Figura 6.11. Árbol de Merkle para almacenar los bloques de DLT. Las hojas son los bloques que contienen los datos. Los nodos internos contienen los *hash* calculados a partir de los *hash* de sus hijos. La validez del árbol se puede comprobar usando solo el *hash* de la raíz

2. calcula la *hash* que corresponde a la cadena con las lecturas
3. inserta el bloque en su copia del árbol de Merkle
4. obtiene la nueva clave para la raíz h_r
5. compone un vector con la cadena y la *hash* ($s|h_r|y_i = 1$)
6. el resto a nodos establecen su estado inicial a ($""|y_i = 0$)

Después de la inicialización, comienza un proceso de consenso. Cuando converge, todos los nodos obtienen la cadena original y el valor *hash* haciendo s/y_i y h/y_i .

Por ejemplo, consideremos una red con cuatro sensores. El nodo $i = 1$ genera un bloque con una lectura, es decir

```

{ "@iot.count": 1,
  "value": [{
    "@iot.id": 1,
    "phenomenonTime": 2018-10-05T08:55.08.504Z,
    "result": 0.9058}]
}

```

A este bloque, una vez se ha añadido al árbol de Merkle, le corresponde una raíz

18F0E3BC9EFA59DFC42916D32818B38ACC495EFCA374AD8D7577C8DD661CD4C9

Cuando finaliza el proceso de consenso, todos los agentes han recibido el mismo bloque de datos y el *hash* para la raíz. Cada agente añade el bloque a su árbol y recupera el nuevo *hash* para la raíz, que debe coincidir con el recibido. La figura 6.12 muestra la convergencia de dicho proceso de consenso. La trama representa la suma de todos los caracteres del bloque. Cuando el consenso converge, significa que el bloque ha llegado a todos los agentes y que pueden extraer los datos y añadirlos a sus árboles.

Este proceso funciona como se esperaba y mantiene copias idénticas de los árboles de Merkle en todos los nodos. Sin embargo, ¿qué sucede si un nodo malicioso intenta modificar el bloque? El agente necesita un bloque que pueda pasar como válido, por lo que tiene que

1. modificar el bloque con una nueva lectura s'
2. insertar el bloque modificado en el árbol
3. recuperar el nuevo *hash* de la raíz h'_r
4. componer el mensaje $(s'|h'_r|y_i(t))$ y enviarlo a sus vecinos

Siguiendo con el ejemplo, el bloque modificado con el resultado alterado es el siguiente

```
{ "@iot.count": 1,
  "value": [{
    "@iot.id": 1,
    "phenomenonTime": 2018-10-05T08:55.08.504Z,
    "result": 0.9312}]
}
```

cuya raíz es

0048862E318D17CF1D55019AE6AC3881910368984CC551DD63137B517A0C3AAC

En este caso, la modificación es detectada automáticamente por los vecinos (ver Figura 6.12(a)). Una vez que el proceso de consenso converge, el bloque propagado es

```
{"@iot.count":1,
 "value":{
  "@iot.id":1,
```

Figura 6.12. Proceso de consenso sobre un DLT. Los valores son la suma de todos los caracteres de las lecturas y el *hash*. (a) Un nodo introduce un bloque modificado en $t = 20$. (b) Convergencia del valor de dv para la corrección.

```
"phenomenonTime": "2018-10-05T08:55.08.504Z",
"result": 0.9057}
}
1EE51B5B3846E0C@@D42B3B0E@89?61FDD422?D948652:CC285@B77B48A5875C'
```

y se puede corregir sobre los bloques recibidos usando el valor de desviación dv_i propagado a través de la red y la ecuación 6.4. El bloque recuperado es el original, incluyendo la *hash*.

```
{"@iot.count": 1,
"value": {
"@iot.id": 1,
"phenomenonTime": "2018-10-05T08:55.08.504Z",
"result": 0.9058}
}
18F0E3BC9EFA59DFC42916D32818B38ACC495EFCA374AD8D7577C8DD661CD4C9
```

6.4.5. Validación en redes sintéticas

Finalmente, se han realizado un conjunto de experimentos para validar cómo se degrada la solución a medida que aumentan los nodos que no siguen el algoritmo de consenso. Las pruebas se han realizado sobre redes aleatorias

Figura 6.13. Desviaciones sin corregir en una red de 100 nodos variando el número de nodos tramposos. La línea central es el valor medio y el área sombreada la desviación típica.

de 100 nodos. Se ha variado el número de nodos que no siguen el proceso desde 5 hasta 50 (representa el 50% de la red). Para cada caso, se han generado 20 redes diferentes y en cada una de ellas se han ejecutado 50 experimentos variando qué nodos hacen trampas, con un total de 1 000 ejecuciones para cada caso. En la Figura 6.13 se muestra el valor medio (la línea central) y la desviación típica (el área sombreada). Puede verse que siempre hay una desviación mínima, debido a la ubicación de los nodos tramposos.

La Figura 6.14 muestra la evolución del número de experimentos en los que se logra recuperar completamente el porcentaje de votos original. Cuando solo hay un nodo tramposos, se recupera en todos los casos. A medida que aumenta el número de nodos tramposos, el número de casos en los que se logra corregir la desviación decrece siguiendo una ley de potencias de parámetro $\alpha = -0,44$.

Para que la solución de Lamport para problema de los generales bizantinos o la corrección planteada por Sundaram funcionen, se requiere que

1. en la red haya al menos $3m + 1$ nodos si hay m nodos tramposos
2. al menos la mitad de los vecinos de cada nodo deben comportarse correctamente

Puesto que para estas pruebas se han generado redes aleatorias sin ningún control sobre el tipo de nodos y su conectividad, y que los ejemplos llegan a

Tabla 6.3

Desviación media residual tras la corrección.

#desviación	media	σ	max.
1	0.0000	0.0000	0
10	0.8240	0.8460	5
20	1.1420	1.0034	5
30	1.4420	1.1798	7
40	1.5810	1.3104	7
50	1.9000	1.5239	10
60	2.0960	1.6448	8
70	2.1920	1.6965	11
80	2.4660	1.9516	13
90	2.5840	2.0541	11
100	2.5970	2.0304	12

una distribución de nodos de confianza y tramposos de 50–50, es de esperar que haya casos en los que la red no sea capaz de compensar la desviación. Resulta interesante ver que el proceso de consenso sigue funcionando aunque se supere el límite de un tercio de nodos tramposos, si bien es algo que debe estudiarse en más profundidad y queda para trabajos futuros.

En el presente capítulo ha desarrollado un mecanismo que permite (i) detectar cuándo se produce una desviación en el proceso de consenso y (ii) corregir la desviación sobre el valor final obtenido.

Se ha mostrado su utilidad en tres problemas de interés: un sistema de votación distribuido, la alteración de imágenes usando ejemplos adversarios y la resolución del problema de los generales bizantinos. En todos ellos, se ha realizado una adaptación del proceso de consenso para poder hacer cálculos sobre valores agregados, aprovechando la asignación concreta de los valores iniciales y realizando un consenso doble en paralelo sobre variables independientes. En los tres casos, se ha aplicado con éxito el algoritmo de consenso con detección de desviaciones. En el caso de las votaciones, se evita que un nodo intente añadir o eliminar votos a alguna de las opciones. Para la difusión de imágenes, se pueden anular los efectos en los errores de clasificación de imágenes a las que se les ha introducido una perturbación mediante ejemplos adversarios. Finalmente, en el problema de los generales bizantinos se corrigen los cambios que los traidores hacen en los mensajes y se recupera la hora original. De esta ma-

Figura 6.14. Número de experimentos que pueden corregir completamente el resultado. Se han ejecutado un total de 1 000 repeticiones. El rendimiento varía del 70 % con 5 nodos tramposos, al 10 % cuando todos los nodos falsean el resultado al menos una vez. Se distribuye siguiendo una ley de potencias con $\alpha = -0,44$

nera, se consigue un método robusto que permite seguir aplicando el consenso en situaciones de incertidumbre, en las que se pueden producir fallos de forma accidental o deliberada y los valores que se transmiten no son los correctos.

Una limitación del método expuesto es que los nodos necesitan tener un número mínimo de vecinos de confianza que propague la información siguiendo el algoritmo. La cantidad de nodos tramposos no debe superar un tercio del total de nodos de la red. Los resultados obtenidos experimentalmente indican que utilizando procesos de consenso se pueden reducir estos límites, pero es necesario un estudio teórico en profundidad sobre el comportamiento del algoritmo presentado. Por otra parte, no se contempla la posibilidad de que los nodos tramposos coordinen sus ataques. Se asume que cada nodo intenta afectar el resultado de forma individual frente a toda la red. Un planteamiento que resuelva estas situaciones se contempla como trabajo futuro.

Capítulo 7

Modelo general de consenso

7.1. Introducción

En los capítulos anteriores se han propuesto distintas extensiones a los procesos de consenso en redes complejas: se han considerado cambios en las redes durante el proceso de consenso, permitiendo introducir dinámica. El segundo lugar, se ha adaptado a redes con signo, en las que los pesos pueden ser positivo o negativos en función del tipo de relación que se establezca entre los nodos. Otra modificación ha consistido en ampliar el consenso a más de una variable acoplada, combinando el consenso con un proceso de ascenso por gradiente. Esto ha permitido aplicar el consenso a problemas de optimización multiobjetivo. Por último, se ha relajado la obligatoriedad de que todos los nodos sigan estrictamente las ecuaciones del proceso, si bien este caso es detectado por el resto de la red y se corrige para anular su efecto. Esto es particularmente interesante en escenarios realmente distribuidos, en los que los nodos de la red están programados de forma independiente y no se puede asegurar que no traten de sacar beneficio propio del proceso.

Todas ellas se basan en la característica del proceso de consenso de conservación de la suma (media), según la cual la suma (media) de los valores iniciales $x_i(0)$ no varía durante el proceso de consenso

$$\sum_i x_i(0) = \sum_i x_i(t) \quad \forall t \geq 0 \quad (7.1)$$

La estrategia general se basa en incluir un término en cada iteración que modele, propague o compense los valores actuales que se producen cuando

- se añaden o eliminan nodos

- se cambian los pesos de los nodos
- el factor de aprendizaje ε se calcula durante el proceso de consenso
- se incluye un término para el descenso por gradiente
- los nodos maliciosos introducen ruido en una o varias iteraciones
- los nodos que detectan un fallo introducen una compensación

para que se conserve la suma. En muchas de esas ocasiones, el término se aplica a una iteración en particular y en otros casos se trata de un término que paulatinamente converge a cero, de manera que tras un periodo transitorio solo se realiza el paso de consenso básico.

La consideración de pesos negativos es la única extensión que se basa en un cambio en la matriz laplaciana y no requiere la incorporación de ningún término adicional. En este caso, la consideración del signo no conserva la suma (media), pero sí lo hace su valor absoluto.

Para concluir el presente trabajo de tesis, se realizará una propuesta unificada de proceso de consenso que tenga en cuenta todas las modificaciones planteadas que sean compatibles entre sí.

Durante el desarrollo de la tesis se han explorado otras extensiones más sencillas, que no tienen la entidad suficiente como para hacer un análisis en profundidad pero sí son útiles para determinados escenarios y se tendrán en cuenta en el modelo general de consenso. Como las extensiones previas, su funcionamiento se basa en la conservación de la suma. La excepción son las redes dirigidas, que se basan en una modificación de la matriz laplaciana, como lo hace la extensión para redes con signo.

7.2. Consenso en redes dirigidas

Una red dirigida es aquella en la que los enlaces entre los nodos están orientados. Un enlace $(i, j) \in E$ es un enlace con origen en el nodo i y fin en el nodo j . EL enlace en dirección contraria $(j, i) \in E$ no existe si no se indica de forma explícita. La consecuencia inmediata es que la matriz de adyacencia que se genera no es una matriz simétrica y, por lo tanto, la matriz de Perron P que se emplea como base del proceso de consenso no es una matriz doble estocástica y esto hace que el consenso no se comporte de la forma esperada.

Dado el grafo dirigido de la Figura 7.1, tenemos dos opciones para construir la matriz laplaciana: usar el grado de entrada o el grado de salida. Llamaremos a las matrices diagonales D_{in} y D_{out} respectivamente. Si empleamos D_{out} , P

Figura 7.1. Grafo dirigido de cinco nodos y matriz de adyacencia asociada. Nótese que la matriz A no es una matriz simétrica

Figura 7.2. Consenso en redes dirigidas. Si la matriz es estocástica simple, no converge al valor medio, que para el estado inicial $x(0) = \{0,1 \ 0,3 \ 0,5 \ 0,7 \ 0,9\}$ sería $\bar{x} = 0,5$. A cambio, la red converge a $x_i(t) = 0,5777$

es una matriz estocástica por filas que converge, pero no al valor medio de $x_i(0)$. En cambio, si empleamos D_{in} , P es una matriz estocástica por columnas que no converge, aunque se conserva la suma de los valores iniciales. Otro inconveniente de usar el grado de entrada es que no se puede implementar en un proceso realmente distribuido, ya que cada nodo debería enviar información a través de enlaces de entrada, lo que resultaría imposible.

El proceso de consenso si usamos directamente una matriz estocástica simple es el que aparece en la Figura 7.2. Para el estado inicial $x(0) = \{0,1 \ 0,3 \ 0,5 \ 0,7 \ 0,9\}$ la red debería converger al valor medio $\bar{x} = 0,5$. Pero al no ser doble estocástica nos e conserva la suma en las iteraciones del proceso de consenso, por lo que la media va variando y finalmente se llega al valor $x_i(t) = 0,5777$.

La idea para conseguir una matriz válida para el consenso es utilizar un algoritmo de escalado que permita conseguir una matriz de Perron doble estocástica partiendo de una que no lo sea [DGH13]. En nuestro caso, se emplearán unas matices diferentes a la propuesta original que permiten su implementación de manera distribuida y la combinación con el proceso de consenso, de manera que se pueda ir construyendo la matriz de Perron final a la vez que se avanza en el consenso.

El proceso de escalado es un proceso iterativo que, a partir de una matriz estocástica por columnas inicial (para que se conserve la suma), va acercándose a una matriz doble estocástica siguiendo la Ecuación 7.2

$$P(t) = \bar{P}\Delta(t) + [I - \Delta(t)] \quad (7.2)$$

siendo

- $P(0) = (I + D_A)^{-1}(I + A)$ una variante de la matriz de Perron inicial
- $\Delta(k) = \text{diag}(\delta_1(k), \dots, \delta_n(k))$ una matriz diagonal que debe cumplir en todo momento que $0 \leq \delta_i(k) \leq 1$
- $\bar{P} = D_A^{-1}A$ es la matriz de adyacencia cuyas filas están divididas por el grado de salida de su nodo.

En nuestro caso, se ha utilizado el Algoritmo 1 propuesto por Domínguez y Hadjicostis [DGH13] para recalcular $\Delta(k)$.

La Figura 7.3 contiene el resultado del proceso de consenso con la matriz de Perron calculada con la Ecuación 7.2. En ambos casos se converge a la media de los valores iniciales. la diferencia es que la Figura 7.3(a) primero hace el cálculo de la matriz de Perron y luego el proceso de consenso, mientras que en la Figura 7.3(b) los dos procesos se realizan a la vez. Los valores de las iteraciones iniciales son distintos, pero al conservarse la suma, las dos son equivalentes.

7.3. Consenso acotado

Otra situación que puede darse en los procesos de consenso es que los nodos de la red tengan definido un intervalo dentro del cual están dispuesto a alcanzar el acuerdo, pero no si se sale de sus límites. Por ejemplo, un grupo de amigos van a salir de cena y cada uno tiene un presupuesto. Un consenso válido sería aquel que está dentro de los límites de todos los participante. Pero ¿qué ocurre si la solución se sale de los intervalos admisibles para algún nodo?

Figura 7.3. Consenso en redes dirigidas a partir del escalado de la matriz de Perron para que sea doble estocástica. El proceso funciona calculando la matriz de forma previa (a) o haciéndolo durante el propio proceso de consenso (b), ya que se conserva la suma y el resultado final es el mismo.

En el ejemplo de la cena, es posible que alguien del grupo no quiera o pueda disponer de tanto dinero y otro miembro del grupo, o todos de forma solidaria, pongan lo que le falta. A este proceso lo llamaremos consenso acotado.

Un problema de consenso acotado está definido por

- un grafo $G = (V, E)$
- un vector x que contiene una variable asociada a cada nodo
- un intervalo $[l, u] \in \mathbb{R}$ asociado a cada nodo.

En el estado inicial, asumiremos que el valor inicial de todos los nodos está dentro del intervalo correspondiente: $x_i(0) \in [l_i, u_i] \forall i$

Una primera aproximación para resolver el problema de consenso acotado es simplemente impedir que un nodo sobrepase sus límites. Si el proceso de consenso hace que un nodo quede por encima (debajo) de su límite superior (inferior), en ese caso se quedará con el límite superior

$$\exists x_i(t) > u_i \rightarrow x_i(t) = u_i \quad (7.3)$$

En esta situación, la red acaba teniendo un comportamiento de seguir al líder, como ya se ha puesto de manifiesto en este trabajo. En la Figura 7.5(a) podemos apreciar este comportamiento, y cómo el valor de consenso converge a 0.4, que es el límite superior del nodo $i = 1$.

nodo	$x_i(0)$	lim. inf.	lim. sup.
1	0.2	-0.2	0.4
2	0.4	0.0	0.8
3	0.6	0.2	1.0
4	0.8	0.4	1.2

Figura 7.4. Configuración de un problema de consenso acotado. Cada nodo, tiene asociado un rango dentro del que aceptará valores para el consenso. El valor medio $\bar{x} = 0,5$ está fuera del intervalo del nodo $i = 1$, cuyo límite superior es 0.4

El mecanismo para solucionarlo es una variación de la corrección que se realiza para el borrado de nodos. La única diferencia es que el nodo no se va de la red completamente y se queda con un valor residual.

Supongamos que en la iteración t el nodo i sobrepasa su límite superior: $x_i(t) > u_i$. El exceso que tiene que asumir el resto de la red es $dt_i(t) = x_i(t) - u_i$. Lo que hace el nodo i es introducir este término en la ecuación del proceso de consenso

$$x_i(t+1) = x_i(t) + \varepsilon \sum_{j \in N_i} [x_j(t) - x_i(t)] + 2dt_i(t) \quad (7.4)$$

donde

$$dt_i(t) = \begin{cases} x_i(t) - u_i & \text{si } x_i(t) > u_i \\ 0 & \text{en otro caso} \end{cases} \quad (7.5)$$

En la siguiente iteración, los vecinos de i recibirán el exceso y lo propagarán por toda la red. El efecto es que la red converge a un valor superior a la media de los valores iniciales para compensar el que el nodo i no llegue a ese valor. El valor se multiplica por dos porque la red tiene que asumir la parte de la media que el nodo i no incorpora en su valor para que se conserve la suma, y otro tanto que es la parte de más que tiene que aportar el resto de nodos de la red.

Es importante dos cosas.

1. la suma no se conserva estrictamente, ya que al duplicar el valor de $dt_i(t)$ estamos incrementando la suma.
2. este cambio es incompatible con la detección de nodos maliciosos. la forma de introducir la corrección es la misma que emplean los nodos para

Figura 7.5. Procesos de consenso acotado en el que los nodos de la red tienen límites. (a) El nodo $i = 1$ no sobrepasa su límite y fuerza al resto de la red a converger a su valor. (b) el nodo $i = 1$ introduce su exceso en la red para que el resto de nodos lo asuman de forma solidaria

hacer trampas, así que la red lo detectaría como un intento de alterar el resultado (como así es) y lo corregiría.

La Figura 7.5(b) muestra el funcionamiento de esta corrección. El nodo $i = 1$ sobrepasa su límite superior en la iteración $t = 3$, introduciendo un primer exceso en la red. La media de los valores iniciales es $\bar{x} = 0,5$ y $l_1 = 0,4$, por lo que la red tendrá que asumir 0.1 extra, que dividido entre los tres nodos restantes es 0.0333: exactamente la diferencia entre el valor de consenso final y la media de los valores iniciales.

7.4. Modelo general de consenso

Para completar el trabajo, a continuación se presenta un modelo general de consenso para redes complejas, que tiene en cuenta en su formulación todos los escenarios planeados hasta ahora. En su definición, se emplean dos mecanismos: la introducción de términos de corrección sobre el valor de los nodos en una iteración de consenso, y la construcción de la matriz laplaciana o la matriz de Perron. La Tabla 7.1 presenta a modo de resumen las estrategias que se han definido para la ampliación del modelo de consenso.

Tabla 7.1

Ampliaciones del modelo de consenso

escenario	método
inserción de nodos	término compensación
borrado de nodos	término compensación
cambio en valores iniciales	término compensación
red pesada	generalización matriz
cambios en los pesos	término compensación
cálculo de ε	término compensación
red con signo	generalización matriz
m variables dependientes	ampliación
consenso + gradiente	término compensación
introducción de ruido	término compensación
red dirigida	generalización matriz
consenso acotado	término compensación

Nota. comment

7.4.1. Generalización de la matriz de Perron

El primer conjunto de cambios afecta a la definición de la matriz de Perron. Los escenarios en los que se modifica directamente la matriz o el cambio proviene de una adaptación de la laplaciana, son los siguientes:

Redes pesadas. Cuando los nodos tienen asociado un peso w_i , la fórmula de consenso es la siguiente

$$x_i(t+1) = x_i(t) + \frac{\varepsilon}{w_i} \sum_{j \in N_i} [x_j(t) - x_i(t)] \quad (7.6)$$

y el término $\frac{\varepsilon}{w_i}$, cuando se trabaja con la expresión matricial, queda incluido dentro de la matriz de Perron

$$P_w = I - \varepsilon W^{-1} L \quad (7.7)$$

Redes con signo. En este caso, el cambio consiste en construir la matriz laplaciana utilizando el grado del nodo. La forma de hacerlo es utilizar la suma de los valores absolutos cuando se construye la diagonal de L

$$\overline{D}_A = [d_i] = \sum_i |a_{ij}| \quad (7.8)$$

Si la red no tiene pesos negativos $\sum_i |a_{ij}| = \sum_i a_{ij}$ por lo que se puede emplear esta expresión en todos los casos.

Redes dirigidas. La problemática en este tipo de redes es doble. Por un lado, la matriz es estocástica simple y para que se conserve la suma es necesario utilizar el grado de salida, lo que equivale a que sea estocástica por filas¹. En ese caso, el cálculo del grado se invierte para obtener la laplaciana

$$D_A^{out} = [d_{ii}] = \sum_j a_{ij} \quad (7.9)$$

Si la red es no dirigida, la matriz A es simétrica y el grado de entrada y de salida son iguales y coinciden con el grado del nodo $D_{in} = D_{out} = D$. Así que se puede construir siempre la matriz de Perron como una matriz estocástica por filas.

Integrando todas las opciones, podemos definir la matriz de Perron generalizada como

$$P_g = I + \varepsilon W^{-1}(\overline{D}_A^{out} + A), \quad \overline{D}_A^{out} = \sum_j |a_{ij}| \quad (7.10)$$

7.4.2. Generalización mediante términos de compensación

El segundo grupo de estrategias pueden resumirse como la inclusión de términos de compensación en la expresión de consenso con el fin de conservar la suma. La característica común de todas ellas es que se ha conseguido definir de forma que se puede calcular dichos términos teniendo en cuenta únicamente la información local disponible en el nodo.

Inserción de nodos. No es necesaria ninguna compensación. Los nuevos nodos simplemente empiezan a enviar sus valores a los vecinos y éstos lo tienen en cuenta en la siguiente iteración del proceso de consenso.

¹En el artículo de Domínguez y Hadjicostis [DGH13] las matrices están transpuestas y mantienen el grado de salida en columnas.

Borrado de nodos. Si se va a eliminar un nodo, este debe "devolver.^a la red la parte de la media que tiene acumulada en su variable. Denotaremos ese término como

$$b_i(t) = x_i(t) - x_i(0) \quad (7.11)$$

indicando que es la cantidad a compensar por el borrado del nodo i en la iteración t

Actualización del valor inicial. El planteamiento es semejante al borrado de los nodos. En este caso, el lugar de tener un valor acumulado por el proceso de consenso, tenemos una diferencia con respecto al valor inicial de partida. En este caso se introduce un término

$$z_i(t) = x'_i(0) - x_i(0) \quad (7.12)$$

que recoge la diferencia entre el valor inicial primero $x_i(0)$ y el nuevo valor inicial $x'_i(0)$ que se tiene que tomar como referencia en el proceso de consenso.

Cambios en los pesos. En redes pesadas dinámicas, el cambio en el peso de los nodos modifica el valor final de consenso pues converge a la media ponderada. Como las medidas anteriores para considerar la dinámica, el cambio en el peso de un nodo i debe compensar el cálculo que ha hecho hasta el momento con el peso anterior. Para ello, se introduce un término $p_i(t)$

$$p_i(t) = \frac{(v_i - w_i)[x_i(0) - x_i(t)]}{v_i} \quad (7.13)$$

donde w_i es el peso antiguo y v_i es el nuevo peso para el nodo i .

Cambios en el valor de ε . Los cambios en ε son algo fundamental para conseguir un proceso de consenso realmente distribuido. Tanto las ecuaciones originales de Olfati-Saber y Murray como la mayoría de sus variaciones incluyen términos globales que se conocen a priori. Una de las aportaciones de este trabajo es la posibilidad de calcular dicho valor de forma autónoma, partiendo del grado de cada uno de los nodos. Mientras ε varía para acercarse a su valor final, es necesario actualizar el valor de consenso para tener en cuenta el nuevo factor, ya que es una cota que se debe satisfacer para garantizar la convergencia del proceso de consenso. La expresión para compensar los cambios es

$$e_i(t) = \left(1 - \frac{\varepsilon_i}{\varepsilon_i}\right) [x_i(0) - x_i(t)] \quad (7.14)$$

donde ε_i es el factor antiguo y ε_i es el nuevo factor para el nodo i .

Consenso + gradiente. La inclusión de variables acopladas introduce un nuevo término en el proceso de consenso que es la expresión de un ascenso por gradiente sobre una función de utilidad. Este factor está pesado por un parámetro ϕ decreciente, de manera que en las primeras iteraciones el gradiente tiene mucha importancia y trata de mantener a los nodos en sus zonas óptimas, y paulatinamente va decreciendo y la parte del consenso toma más importancia. De esta manera, en las primeras iteraciones los nodos quedan conectados con otros nodos semejantes en su proximidad, desconectándose de nodos poco relevantes. Si $\phi = 0$, el algoritmo se convierte en un proceso de consenso simple, con m variables independientes, o con una sola variable si $m = 1$. El término a incluir en la ecuación es

$$- \phi \frac{1}{\max_i \|\nabla u_i(x_i)\|_2} \cdot \frac{x_i^\alpha(t) - x_i^\alpha(0)}{(1 - w_i^\alpha)^2} u_i(x_i) \quad (7.15)$$

Introducción de ruido. Es el caso más simple, pues simplemente introduce un término $u_i(t)$ que indica la desviación que el nodo i introduce en la iteración t del proceso de consenso. El valor de este término queda libre, a discreción del nodo.

Con todos estos términos, la formulación completa del proceso de consenso generalizado queda

$$\begin{aligned} x_i(t+1) = & x_i(t) + \frac{\varepsilon_i}{w_i} \sum_{j \in N_i} [x_j(t) - x_i(t)] + b_i(t) + z_i(t) + p_i(t) + e_i(t) \\ & + u_i(t) - \phi \frac{1}{\max_i \|\nabla u_i(x_i)\|_2} \cdot \frac{x_i^\alpha(t) - x_i^\alpha(0)}{(1 - w_i^\alpha)^2} u_i(x_i) \end{aligned} \quad (7.16)$$

La única consideración a tener en cuenta es que el proceso de consenso robusto de detección y corrección de errores es incompatible con el uso de términos de compensación, ya que cualquier nodo lo detectará como un intento de manipulación del proceso de consenso y tratará de corregir el resultado final.

Capítulo 8

Conclusiones

En esta tesis se ha presentado la contribución a los procesos de consenso en redes complejas, en los que un conjunto de nodos son capaces de llegar a un acuerdo en el valor de un conjunto de variables de interés utilizando únicamente la información local de cada nodo y la de los vecinos directos. No existe ningún nodo que coordine el intercambio de información y no se conoce el tamaño, la topología ni otro tipo de información de la red.

El trabajo realizado y las principales conclusiones a las que se ha llegado se detallan a continuación.

1. Se ha planteado el modelo general en redes dinámicas, en las que distintos factores pueden variar mientras tiene lugar el proceso de consenso. La mayoría de los trabajos analizados mantenían la red constante y únicamente se permiten cambios en la topología de la red, siempre y cuando todos los nodos estén conectados eventualmente en una sola componente conexa. La necesidad de la componente conexa se mantiene, pero se han relajado el resto de restricciones y se ha conseguido un procedimiento válido para cualquier situación. En concreto, los cambios que se han introducido son
 - a) dinámica en los nodos: Es una de las restricciones que más se repite: el conjunto de nodos se establece al principio y permanece sin cambios durante el proceso de consenso. En este trabajo se permite, por un lado, que cambie el número de nodos que participa en el proceso de consenso, y el cálculo se actualiza para que en todo momento se tenga en cuenta solo a los nodos participantes. Si entran nuevos nodos, el valor de consenso los incluyen. De la misma manera, se elimina el efecto de los nodos que han abandonado el

proceso. Por otra parte, se tienen en cuenta cambios en los valores sobre los que se realiza el consenso. Es particularmente importante para aplicaciones como las redes de sensores, que están actualizando permanentemente sus lecturas. Sin esta aportación, habría que completar el proceso de consenso con las lecturas antiguas antes de comenzar un nuevo consenso para calcular el valor de la función correspondiente sobre las nuevas lecturas.

- b) *dinámica en los pesos*: Se ha introducido la posibilidad de emplear de redes pesadas, en las que los nodos tienen asociado un atributo que les aporta una importancia relativa en el proceso de consenso. No se tienen en cuenta pesos en los enlaces ya que su efecto se cancela en los nodos de sus extremos. Además de incluir los pesos, también se permite que varíen a lo largo del proceso y el consenso converge al la media ponderada utilizando los pesos más recientes.
 - c) *cambios en el factor de aprendizaje*, que permite aplicarlo a entornos completamente distribuidos. Es una de las mayores aportaciones, pues la mayoría de trabajos mantienen algún parámetro global que es conocido por toda la red (o incluso es necesario conocimiento de la red completa). Con ello se puede realizar una implementación completamente distribuida del proceso de consenso. Dado que el factor es una cota necesaria para que el proceso de consenso converja y se va actualizando continuamente, es muy importante que este valor se mantenga actualizado. Muchos de los cambios en la estructura de la red (inserción y borrado de enlaces o reconexiones) afectan al grado de los nodos y, por lo tanto, al valor de ε .
2. Se ha extendido el modelo a redes con signo, que permiten modelar relaciones positivas y negativas entre los nodos. Existen numerosos casos de la vida real en los que se emplean este tipo de relaciones. En la tesis se han expuesto varios de ellos, además de utilizar ejemplos sintéticos para validar la propuesta comparándola con el rendimiento de los algoritmos más habituales. Se ha planteado como un mecanismo para la detección de comunidades y se ha propuesto la coherencia como una medida de la calidad de la división en coaliciones. Las aportaciones principales son
- a) una propuesta para la división en dos comunidades incluso cuando la red no es balanceada. Los estudios existentes se limitan a casos en los que la red es k -partida, es decir, el subgrafo generado por los pesos negativos se puede dividir exactamente en k componentes. En esta tesis se ha comprobado empíricamente que el proceso de

consenso permite detectar comunidades también cuando la red no está balanceada. En ese caso, el proceso converge a cero por la derecha y por la izquierda, y es el signo de $x_i(t)$ quien establece la pertenencia a una u otra comunidad.

- b) a continuación se ha extendido el resultado a k comunidades mediante un proceso recursivo que va dividiendo las comunidades en dos hasta que en una iteración no se forman más grupos. Los resultados se han comparado con redes que son casos de referencia en los trabajos sobre detección de comunidades en redes con signo y también en bancos de pruebas sintéticos para medir el rendimiento y compararlo con los principales algoritmos. Finalmente, dados los buenos resultados obtenidos en tiempos de ejecución, se ha probado con datos reales en una red con millones de enlaces.
3. Se ha ampliado el dominio a más de una variable, dependientes entre sí, para abordar problemas más complejos. El problema se ha modelado como una red multicapa, donde en cada capa se consensua el valor de una de las variables y luego cada agente realiza un ascenso por gradiente entre las capas para tratar de optimizar una función objetivo que relaciona los valores de las variables entre sí según una función de utilidad que se mantiene privada en todo momento. Las contribuciones en este punto han sido
- a) utilización de gaussianas como funciones de utilidad que cumplen los criterios básicos para usarlas en criterios de optimización y su utilización en el proceso de descenso por gradiente simplifica la formulación y permite realizar los cálculos usando exclusivamente información local del nodo. Esto permite mantener las funciones de utilidad privadas para los nodos de la red y no es necesario compartir esa información ni siquiera con los vecinos directos.
 - b) conseguir que el proceso combinado de consenso y gradiente converja en el entorno del máximo de la función de utilidad global definida como la suma de las utilidades locales de los nodos, si que dicha función se calcule en ningún momento. Los nodos siguen por la función local para realizar el ascenso por gradiente y corregir las desviaciones que les produce el proceso de consenso. Al final, los nodos se agrupan en el entorno del punto óptimo si es único y no existen mínimos locales.
 - c) cuando los nodos forman grupos alejados entre sí, el formar que todos formen parte de una agrupación única no es eficiente. Por ese

motivo, se ha añadido la posibilidad de romper enlaces en la red cuando la utilidad de un nodo en el punto la que le dirige el proceso de consenso es cero o está por debajo de un umbral. Cuando esto ocurre, los nodos se dividen en grupo de forma natural y cada uno de ellos converge a puntos distintos.

4. Se ha detectado la presencia de nodos maliciosos que tratan de alterar el resultado del consenso. Esto ocurre en entornos distribuidos, donde cada uno de los elementos de la red ha sido diseñado o programado por personas diferentes, en las que no podemos confiar al cien por cien. En esos casos, puede haber nodos que traten de mejorar su provecho falseando el resultado al que converge el proceso. Para evitarlo, se ha diseñado una estrategia que permite
 - a) detectar las desviaciones que uno o más nodos tratan de introducir en la red de una iteración determinada. El mecanismo se basa de nuevo en la conservación de la suma y es capaz de detectar cuando la suma de todos los valores recibidos de sus vecinos se ha desviado del valor esperado a partir del dato recibido en la iteración inmediatamente anterior. Es importante remarcar que se detecta una desviación en el global de todos los valores recibidos, pero no se puede identificar con certeza cuál de los vecinos es el que ha falseado los datos, deliberada o accidentalmente.
 - b) corregir las desviaciones cuando ha finalizado el proceso de consenso este ha convergido a una solución que no es la media de los valores iniciales. La propuesta consiste en un segundo proceso de consenso en paralelo para propagar la desviación que los vecinos habrán detectado. El nodo malicioso debe colaborar en este proceso porque no hacerlo implica automáticamente el descubrir su engaño. Además, aunque no se pueda detectar con seguridad el nodo tramposo, si este insiste en falsear los resultados los nodos tienen mecanismos para asegurar su identidad y separarlo de la red o imponerle sanciones.
5. Por último, se han integrado todas las propuestas en un solo modelo de consenso que aúna estas aportaciones más otras aportaciones menores que también resultan de utilidad en determinados contextos. Se ha detectado que todas las modificaciones consisten básicamente en la modificación de la matriz de Perron o en la inclusión de términos de compensación. El presente trabajo de tesis concluye con un modelo generalizado de consenso que puede aplicarse en todas las situaciones anteriores de

forma independiente, o en diferentes combinaciones. La única restricción es que no se puede combinar los métodos que incluyen un término de compensación con la detección de nodos maliciosos, puesto que las desviaciones se realizan introduciendo un término de compensación, por lo que los nodos de la red detectarían cualquier otra adaptación como un intento de alteración del consenso.

6. Se ha demostrado la utilidad de las aproximaciones propuestas aplicándolas a problemas reales de interés en la actualidad. Cada uno de ellos pone de manifiesto la utilidad de las aportaciones que se han realizado en cada capítulo.
 - a) red eléctrica balear: es un caso que utiliza la dinámica de las redes para ajustarse a la demanda energética de las Islas Baleares. El modelo estudia cómo la red puede adaptarse de forma flexible a las variaciones en la demanda, así como cubrir los fallos ocasionales que se pueden producir de determinadas subestaciones, recuperando el régimen normal cuando éstas se reestablecen.
 - b) detección de comunidades en redes con signo, que permiten la identificación de grupos dentro de una red cuando las relaciones entre sus miembros son positivas o negativas, modelando relaciones de amistad/enemistad, confianza/desconfianza, atracción/rechazo, etcétera. El modelo ha sido capaz de trabajar con redes particularmente grandes, como es el caso de las guerras de edición en la Wikipedia.
 - c) procesos analíticos jerárquicos (AHP), como ejemplo de aplicación del consenso con variables acopladas combinado con un ascenso por gradiente. Esta aportación permite aplicar el consenso a una familia de problemas que se conoce como problemas de optimización multiobjetivo. EL proceso de consenso llega a soluciones coherentes y en rangos de decisiones equivalentes a los que se alcanzan con estrategias centralizadas. Además, permite resolver uno de los mayores problemas para aplicar esta técnica que es cómo grupos de expertos llega a un acuerdo sobre el peso de los juicios con los que se evalúan las diferentes alternativas.
 - d) en el caso del proceso de consenso robusto, se plantean tres escenarios de gran actualidad. En primer lugar, un sistema distribuido de votación electrónica que permite computar los votos obtenidos por las distintas opciones de forma distribuida. En segundo lugar, un caso que ha generado tremendas complicaciones a las aplicaciones

de reconocimiento de imágenes mediante aprendizaje profundos, como son los ejemplos adversarios, que con procesados aparentemente inofensivos de las imágenes, hacen que los clasificadores fallen estrepitosamente. Por último, se ha demostrado como se puede resolver el problema de los generales bizantinos: un problema mental que es la base de las cadenas de bloques y las criptomonedas. La solución que aporta el consenso eliminan la necesidad de incluir pruebas de trabajo (POW) para validar la corrección de los bloques, de manera que se puede optar por soluciones mucho más eficientes que garantizan el mismo nivel de seguridad, a la vez que mantienen la ventaja que aportan las bases de datos distribuidas.

Referencias

- [aH13] Hong-xiang et al Hu. Group consensus in multi-agent systems with hybrid protocol. *Journal of the Franklin Institute*, 350(3):575–597, 2013.
- [AMI12] Pranay Anchuri and Malik Magdon-Ismail. Communities and Balance in Signed Networks: A Spectral Approach. In *ASO-NAM'12*, pages 235–242, 2012.
- [AP13] Alessia Amelio and Clara Pizzuti. Community Mining in Signed Networks: A Multiobjective Approach. In *Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 95–99, New York, NY, USA, 2013. ACM.
- [ASJ13] Omid Askari-Sichani and Mahdi Jalili. Large-scale global optimization through consensus of opinions over complex networks. *Complex Adaptive Systems Modeling*, 1(1):11, 2013.
- [Atk05] Wei Ren Randal W Beard Ella M Atkins. A Survey of Consensus Problems in Multi-agent Coordination. In *In Proc. of 2005 American Control Conference*, 2005.
- [BBC⁺14] S. Boccaletti, G. Bianconi, R. Criado, C.I. del Genio, J. Gomez-Garde nes, M. Romance, I. Sendi na Nadal, Z. Wang, and M. Zanin. The structure and dynamics of multilayer networks. *Physics Reports*, 2014.
- [BGPS05] S Boyd, A Ghosh, B Prabhakar, and D Shah. Gossip algorithms: design, analysis and applications. In *IEEE INFOCOM 2005.*, pages 1653–1664, 2005.

- [BHOT05] V D Blondel, J M Hendrickx, A Olshevsky, and J N Tsitsiklis. Convergence in multiagent coordination, consensus, and flocking. In *In Proc. IEEE Conferene on Decision and Control*, pages 2996–3000, 2005.
- [BL10] Ulrik Brandes and Jürgen Lerner. Structural Similarity: Spectral Methods for Relaxed Blockmodeling. *J. Classification*, 27(3):279–306, 2010.
- [BLM⁺06] s. Boccaletti, v. Latora, y. Moreno, m. Chavez, and D.U. Hwang. Complex networks: Structure and dynamics. *Physics reports*, 424(4–5):175–308, 2006.
- [BMS⁺08] K Bhattacharya, G Mukherjee, J Saramäki, K Kaski, and S S Manna. The International Trade Network: weighted network analysis and modelling. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(02):P02002, 2008.
- [BSSZ16] Bosko Blagojevic, Bojan Srdjevic, Zorica Srdjevic, and Tihomir Zoranovic. Heuristic aggregation of individual judgments in AHP group decision making using simulated annealing algorithm. *Information Sciences*, 330:260–273, 2016.
- [BTU12] J Bruggemana, V A Traagb, and J Uitermarkc. Detecting Communities through Network Data. *American Sociological Review*, 77(6):1050–1063, 2012.
- [CBV12] N Cesa-Bianchi and F Vitale. A correlation clustering approach to link classification in signed networks. In *Proceedings of the 25th Annual Conference on Learning Theory.*, pages 1–34, 2012.
- [CCC⁺18] C Chang, C Chang, C Chang, D Lee, and P Lu. Exponentially Twisted Sampling for Centrality Analysis and Community Detection in Attributed Networks. *IEEE Transactions on Network Science and Engineering*, pages 1–1, 2018.
- [CFL09] C Castellano, S Fortunato, and V Loreto. Statistical physics of social dynamics. *Rev. Mod. Phys*, 81(2):591–646, 2009.
- [CI12] Kai Cai and H Ishii. Average consensus on arbitrary strongly connected digraphs with dynamic topologies. In *American Control Conference (ACC), 2012*, pages 14–19, 2012.

- [CLSW10] Wei Chen, Zhenming Liu, Xiaorui Sun, and Yajun Wang. A game-theoretic framework to identify overlapping communities in social networks. *Data Mining and Knowledge Discovery*, 21(2):224–240, 2010.
- [CLSW11] Wei Chen, Zhenming Liu, Xiaorui Sun, and Yajun Wang. Community Detection in Social Networks Through Community Formation Games. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Volume Three*, pages 2576–2581. AAAI Press, 2011.
- [CMB06] J Cortés, S Martinez, and F Bullo. Robust rendezvous for mobile autonomous agents via proximity graphs in arbitrary dimensions. *Automatic Control, IEEE Transactions on*, 51(8):1289–1298, 2006.
- [CMGT16] Qing Cai, Lijia Ma, Maoguo Gong, and Dayong Tian. A Survey on Network Community Detection Based on Evolutionary Computation. *Int. J. Bio-Inspired Comput.*, 8(2):84–98, 2016.
- [CWD12] Kai-Yang Chiang, Joyce Jiyoung Whang, and Inderjit S Dhillon. Scalable clustering of signed networks using balance normalized cut. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, pages 615–624, New York, NY, USA, 2012. ACM.
- [CWWL16] Jianrui Chen, Hua Wang, Lina Wang, and Weiwei Liu. A dynamic evolutionary clustering perspective: Community detection in signed networks by reconstructing neighbor sets. *Physica A: Statistical Mechanics and its Applications*, 447:482–492, 2016.
- [DeG74] M. DeGroot. Reaching a consensus. *Journal of the American Statistical Association*, 345(345):118–121, 1974.
- [DGH13] A. D. Dominguez-Garcia and C. N. Hadjicostis. Distributed matrix scaling and application to average consensus in directed graphs. *IEEE Transactions on Automatic Control*, 58(3):667–681, 2013.
- [DHS16] Shaofu Du, Li Hu, and Malin Song. Production optimization considering environmental performance and preference in the cap-and-trade system. *Journal of Cleaner Production*, 112:1600–1607, 2016.

- [DM04] Luca Donetti and Miguel A Muñoz. Detecting network communities: a new systematic and efficient algorithm. *Journal of Statistical Mechanics: Theory and Experiment*, 2004(10):P10012, 2004.
- [DNF14] Richard K Darst, Zohar Nussinov, and Santo Fortunato. Improving the performance of algorithms to find communities in networks. *Physical Review E*, 89:032809, 2014.
- [DS14] Qingxing Dong and Thomas L Saaty. An analytic hierarchy process model of group consensus. *Journal of Systems Science and Systems Engineering*, 23(3):362–374, 2014.
- [DZHX10] Yucheng Dong, Guiqing Zhang, Wei-Chiang Hong, and Yinfeng Xu. Consensus models for AHP group decision making under row geometric mean prioritization method. *Decision Support Systems*, 49(3):281–289, 2010.
- [FCFZ09] Paolo Frasca, Ruggero Carli, Fabio Fagnani, and Sandro Zampieri. Average consensus on networks with quantized communication. *Int J Robust Nonlin*, 19(16):1787–1816, 2009.
- [FLM86] Michael J Fischer, Nancy A Lynch, and Michael Merritt. Easy impossibility proofs for distributed consensus problems. *Distrib. Comput.*, 1(1):26–39, 1986.
- [For10] Santo Fortunato. Community detection in graphs. *Physics reports*, 486(3–5):75–174, 2010.
- [FRS09] Giorgio Fagiolo, Javier Reyes, and Stefano Schiavo. World-trade web: Topological properties, dynamics, and evolution. *Physical Review E*, 79:036115, 2009.
- [GB13] Prem K Gopalan and David M Blei. Efficient discovery of overlapping communities in massive networks. *Proceedings of the National Academy of Sciences*, 110(36):14534–14539, 2013.
- [Gea05] Diego Garlaschelli et al. The scale-free topology of market investments. *Physica A: Statistical Mechanics and its Applications*, 350(2-4):491–499, 2005.
- [GL05] Diego Garlaschelli and Maria I Loffredo. Structure and evolution of the world trade network. *Physica A: Statistical Mechanics and its Applications*, 355(1):138–144, 2005.

- [GL11] W.V. Gehrlein and D. Lepelley. *Voting Paradoxes and Group Coherence. The Condorcet Efficiency of Voting Rules*. Springer, 2011.
- [GoJA09] Sergio Gómez, Pablo Jensen, and Alex Arenas. Analysis of community structure in networks of correlated data. *Physical Review E*, 80:016114, 2009.
- [GPCR13] Yolanda Gómez, Alberto Palomares, Carlos Carrascosa, and Miguel Rebollo. Automatic deployment of a consensus networks mas. In Yves Demazeau, Toru Ishida, Juan M. Corchado, and Javier Bajo, editors, *Advances on Practical Applications of Agents and Multi-Agent Systems*, pages 279–282. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.
- [GSS14] Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and Harnessing Adversarial Examples. *arXiv*, 1412.6572, 2014.
- [GZ16] Wei-Feng Guo and Shao-Wu Zhang. A general method of community detection by identifying community centers with affinity propagation. *Physica A: Statistical Mechanics and its Applications*, 447:508–519, 2016.
- [HBG⁺14] Steve Harenberg, Gonzalo Bello, L Gjeltrema, Stephen Ranshous, Jitendra Harlalka, Ramona Seay, Kanchana Padmanabhan, and Nagiza Samatova. Community detection in large-scale networks: a survey and empirical evaluation. *Wiley Interdisciplinary Reviews: Computational Statistics*, 6(6):426–439, 2014.
- [HCLL13] Yeu-Shiang Huang, Wei-Chen Chang, Wei-Hao Li, and Zu-Liang Lin. Aggregation of utility-based individual preferences for group decision-making. *European Journal of Operational Research*, 229(2):462–469, 2013.
- [Hop82] J. J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *PNAS*, 79(8):2554–2558, 1982.
- [HRVW96] Christoph Helmberg, Franz Rendl, Robert J Vanderbei, and Henry Wolkowicz. An Interior-Point Method for Semidefinite Programming. *SIAM Journal on Optimization*, 6:342–361, 1996.

- [HWFZ18] W Hou, Z Wu, M Fu, and H Zhang. Constrained consensus of discrete-time multi-agent systems with time delay. *International Journal of Systems Science*, 49(5):947–953, 2018.
- [IL11] Alessio Ishizaka and Ashraf Labib. Review of the main developments in the analytic hierarchy process. *Expert Systems with Applications*, 38(11):14336 – 14345, 2011.
- [JBA11] Paulo Jesus, Carlos Baquero, and Paulo Sérgio Almeida. A Survey of Distributed Data Aggregation Algorithms. *CoRR*, abs/1110.0725, 2011.
- [JCdAdAR18] Emanuel Ferreira Jesus, Vanessa R L Chicarino, Célio V N de Albuquerque, and Antônio A de A Rocha. A Survey of How to Use Blockchain to Secure Internet of Things and the Stalker Attack. *Security and Communication Networks*, pages 1–27, 2018.
- [JJDQ15] Lyu Jianting, Qin Jiahu, Gao Dai, and Liu Qingqing. Consensus for constrained multi-agent systems with input saturation. *International Journal of Robust and Nonlinear Control*, 26(14):2977–2993, 2015.
- [JJS10] Sindhu Joseph, Carles Sierra Joseph, and Marco Schorlemmer. A coherence-driven action selection in dynamic environments. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: Volume 1 - Volume 1*, AAMAS '10, pages 1497–1498, Richland, SC, 2010. International Foundation for Autonomous Agents and Multiagent Systems.
- [JLM03] A Jadbabaie, Jie Lin, and A S Morse. Coordination of groups of mobile autonomous agents using nearest neighbor rules. *Automatic Control, IEEE Transactions on*, 48(6):988–1001, 2003.
- [JLY12] Zhijian Ji, Hai Lin, and Haisheng Yu. Leaders in multi-agent controllability under consensus algorithm and tree topology. *Systems & Control Letters*, 61(9):918–925, 2012.
- [KDG03] David Kempe, Alin Dobra, and Johannes Gehrke. Gossip-Based Computation of Aggregate Information. In *Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science*, pages 482–482, 2003.

- [KM96] Samo Kropivnik and Andrej Mrvar. An Analysis of the Slovene Parliamentary Parties Network. *Developments in Statistics and Methodology*, 12:209–216, 1996.
- [KON13] KONECT. Wikipedia conflict network dataset [on line] available at <http://konect.uni-koblenz.de/networks/wikiconflict>. Technical report, Universität Koblenz-Landau, 2013.
- [Ksh17] N Kshetri. Blockchain’s roles in strengthening cybersecurity and protecting privacy. *Telecommun. Policy*, 41(10):1027–1038, 2017.
- [KSL⁺10] J Kunegis, Schmidt S, A Lommatzsch, J Lerner, and L Sahin Albayrak Ernesto W de. Spectral Analysis of Signed Graphs for Clustering, Prediction and Visualization. In *Proc. of SDM*, pages 559–559, 2010.
- [Kun13] Jérôme Kunegis. KONECT – The Koblenz Network Collection. In *Proc. Int. Web Observatory Workshop*, 2013.
- [LA16] Young-Hun Lim and Hyo-Sung Ahn. Partial consensus of identical feedforward dynamic systems with input saturations. *International Journal of Robust and Nonlinear Control*, 26(11):2494–2510, 2016.
- [LADW06] L Li, D Alderson, J C Doyle, and W Willinger. Towards a Theory of Scale-Free Graphs: Definitions, Properties, and Implications. *Internet Mathematics*, 4(2):431–523, 2006.
- [LDCH10] Zhongkui Li, Zhisheng Duan, Guanrong Chen, and Lin Huang. Consensus of Multiagent Systems and Synchronization of Complex Networks: A Unified Viewpoint. *Circuits and Systems I: Regular Papers, IEEE Transactions on*, 57(1):213–224, 2010.
- [LEWV10] J. Li, E. Elhamifar, I. Wang, and R. Vidal. Consensus with robustness to outliers via distributed optimization. In *49th IEEE Conference on Decision and Control (CDC)*, pages 2111–2117, 2010.
- [LF12] Andrea Lancichinetti and Santo Fortunato. Consensus clustering in complex networks. *Sci. Rep.*, abs/1203.6093, 2012.

- [LFK09] Andrea Lancichinetti, Santo Fortunato, and János Kertész. Detecting the overlapping and hierarchical community structure in complex networks. *New Journal of Physics*, 11(3):033015, 2009.
- [LFR08] Andrea Lancichinetti, Santo Fortunato, and Filippo Radicchi. Benchmark graphs for testing community detection algorithms. *Physical Review E*, 78:046110, 2008.
- [LHL16] Zhaoxing Li, Lile He, and Yunrui Li. A novel multiobjective particle swarm optimization algorithm for signed network community detection. *Applied Intelligence*, 44(3):621–633, 2016.
- [LJ08] Peng Lin and Yingmin Jia. Average consensus in networks of multi-agents with both switching topology and coupling time-delay. *Physica A: Statistical Mechanics and its Applications*, 387(1):303–313, 2008.
- [LJ09] Peng Lin and Yingmin Jia. Consensus of second-order discrete-time multi-agent systems with nonuniform time-delays and dynamically changing topologies. *Automatica*, 45:2154–2158, 2009.
- [LL16] Dequan Li and Zongli Lin. Reaching consensus in unbalanced networks with coarse information communication. *International Journal of Robust and Nonlinear Control*, 26(10):2153–2168, 2016.
- [LLDF17] Yuezu Lv, Zhongkui Li, Zhisheng Duan, and Gang Feng. Novel distributed robust adaptive consensus protocols for linear multi-agent systems with directed graphs and external disturbances. *International Journal of Control*, 90(2):137–147, 2017.
- [LSP82] L Lamport, R Shostak, and M Pease. The Byzantine Generals Problem. *ACM TPLS*, 4(3):382–401, 1982.
- [LSTT11] B W K Ling, P Stewart, Kok-Lay Teo, and C K Tse. Study of near consensus complex social networks using eigen theory. In *Circuits and Systems (ISCAS), 2011 IEEE International Symposium on*, pages 2107–2110, 2011.
- [LYK⁺11] Kyu-Min Lee, Jae-Suk Yang, Gunn Kim, Jaesung Lee, Kwang-Il Goh, and In-mook Kim. Impact of the Topology of Global

- Macroeconomic Network on the Spreading of Economic Crises. *PLoS ONE*, 6(3):e18443, 2011.
- [MDB17] Axel Moinet, Benoit Darties, and Jean-Luc Baril. Blockchain based trust and authentication for decentralized sensor networks. *arXiv*, 1706.01/730v1, 2017.
- [Mea07] Lev Muchnik et al. Self-emergence of knowledge trees: Extraction of the Wikipedia hierarchies. *Physical Review E*, 76(1):016106, 2007.
- [MH17] S F Shahandashti McCorry and F Hao. A Smart Contract for Boardroom Voting with Maximum Voter Privacy. In *International Conference on Financial Cryptography and Data Security*, pages 357–375, 2017.
- [MP04] Y Moreno and A F Pacheco. Synchronization of Kuramoto oscillators in scale-free networks. *Europhysics Letters (epl)*, 68:603–609, 2004.
- [MZ10] C Li T Ma and J Zhang. Consensus control for leader-following multi-agent systems with measurement noises. *J Syst Sci Complex*, 23(1):35–49, 2010.
- [NdC11] Mariá C V Nascimento and André C P L F de Carvalho. Spectral methods for graph clustering – A survey. *European Journal of Operational Research*, 211(2):221–231, 2011.
- [New03] M. Newman. The structure and function of complex networks. *SIAM Review*, 45(2):167–256, 2003.
- [NOP10] A Nedic, A Ozdaglar, and P A Parrilo. Constrained Consensus and Optimization in Multi-Agent Networks. *IEEE TAC*, 55(4):922–938, 2010.
- [OPPU16] Y Orlov, A Pilloni, A Pisano, and E Usai. Consensus-based leader-follower tracking for a network of perturbed diffusion PDEs via local boundary interaction. *IFAC*, 49(8):228–233, 2016.
- [OSFM07] Reza Olfati-Saber, J A Fax, and Richard M Murray. Consensus and Cooperation in Networked Multi-Agent Systems. *Proceedings of the IEEE*, 95(1):215–233, 2007.

- [OSK16] Wolfgang Ossadnik, Stefanie Schinke, and Ralf H Kaspar. Group Aggregation Techniques for Analytic Hierarchy Process and Analytic Network Process: A Comparative Analysis. *Group Decision and Negotiation*, 25(2):421–457, 2016.
- [OSM04] Reza Olfati-Saber and Richard M Murray. Consensus problems in networks of agents with switching topology and time-delays. *Automatic Control, IEEE Transactions on*, 49(9):1520–1533, 2004.
- [OT09] Alex Olshevsky and John N Tsitsiklis. Convergence Speed in Distributed Consensus and Averaging. *SIAM J. Control Optim.*, 48(1):33–55, 2009.
- [PCA⁺18] Claudia Pop, Tudor Cioara, Marcel Antal, Ionut Anghel, Ioan Salomie, and Massimo Bertoncini. Blockchain Based Decentralized Management of Demand Response Programs in Smart Energy Grids. *Sensors*, 18(162):1–21, 2018.
- [PCRG13] A Palomares, C Carrascosa, M Rebollo, and Y Gómez. Implementing MAS Agreement Processes Based on Consensus Networks. In *Proc. of Distributed Computing and Artificial Intelligence*, pages 553–560. Springer International Publishing, 2013.
- [PPZ10] S S Pereira and A Pages-Zamora. Mean Square Convergence of Consensus Algorithms in Random WSNs. *Signal Processing, IEEE Transactions on*, 58(5):2866–2874, 2010.
- [PPZ11] S S Pereira and A Pages-Zamora. Consensus in Correlated Random Wireless Sensor Networks. *IEEE Sig. Proc.*, 59(12):6279–6284, 2011.
- [Raf17] M Rafailescu. Fault Tolerant Leader Election In Distributed Systems. *IJCSIT*, 9(1):13–20, 2017.
- [RBLG18] M Rebollo, R M Benito, J C Losada, and J Galeano. Robust Distributed Voting Mechanism by Consensus. In *Proc. IEEE/ACM Int. Conf. on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 1–6, 2018.
- [RCP14] Miguel Rebollo, Carlos Carrascosa, and Alberto Palomares. Consensus in Smart Grids for Decentralized Energy Management. In *Highlights of Practical Applications of Heterogeneous*

- Multi-Agent Systems. The PAAMS Collection*, pages 250–261. Springer, 2014.
- [Rea54] K.E. Read. Cultures of the central highlands, new guinea. *Southwestern Journal of Anthropology*, 10(1):1–43, 1954.
- [RG94] R Ramanathan and L S Ganesh. Group preference aggregation methods employed in AHP: An evaluation and an intrinsic process for deriving members’ weightages. *European Journal of Operational Research*, 79(2):249–265, 1994.
- [RN10] Peter Ronhovde and Zohar Nussinov. Local resolution-limit-free Potts model for community detection. *Physical Review E*, 81:046114, 2010.
- [RPC16] M. Rebollo, A. Palomares, and C. Carrascosa. Towards building agreement spaces using consensus networks. *AI Communications*, 29(1):17–30, 2016.
- [RV06] R Rajagopalan and P K Varshney. Data-aggregation techniques in sensor networks: a survey. *Communications Surveys Tutorials, IEEE*, 8(4):48–63, 2006.
- [SBna03] Ma Ángeles Serrano and Mari án Bogu n á. Topology of the world trade web. *Physical Review E*, 68:015101, 2003.
- [Sch07] Satu Elisa Schaeffer. Survey: Graph clustering. *Comput. Sci. Rev.*, 1(1):27–64, 2007.
- [SRgAA10] Norman Salazar-Ramirez, Juan A Rodr i guez Aguilar, and Josep Lluís Arcos. Robust Coordination in Large Convention Spaces. *AI Communications*, 23:357–372, 2010.
- [SS13] Bojan Srdjevic and Zorica Srdjevic. Synthesis of individual best local priority vectors in AHP-group decision making. *Applied Soft Computing*, 13(4):2045–2056, 2013.
- [SSH11] S S Sundaram and C N Hadjicostis. Distributed function calculation via linear iterative strategies in the presence of malicious agents. *IEEE TAC*, 56(7):1495–1508, 2011.
- [TB09] V A Traag and Jeroen Bruggeman. Community detection in networks with positive and negative links. *Physical Review E*, 80:036115, 2009.

- [TC07] Christopher Tricaud and YangQuan Chen. A consensus-based decision algorithm is used to manage water distribution into a parallel ponds network and achieve good quality of service and minimum water-loss. In *Proceedings of the 2007 IEEE International Conference on Mechatronics and Automation*, 2007.
- [TDSHJ16] C Tanas, S Delgado-Segura, and J Herrera-Joancomartí. An integrated reward and reputation mechanism for MCS preserving users'privacy. In *Proc. 10th Int. Workshop Data Privacy Manage., Secur. Assurance, vol. 9481. .*, pages 83–99, New York, 2016. Springer-Verlag.
- [Tia16] F Tian. An agri-food supply chain traceability system for China based on RFID a blockchain technology. In *Proc. 13th Int Conf. Service Syst. Service Manage. (ICSSSM)*, pages 1–6, 2016.
- [TLPH17] Michel Toulouse, Hai Le, Cao Vien Phung, and Denis Hock. Defense Strategies against Byzantine Attacks in a Consensus-Based Network Intrusion Detection System. *Informatica*, 41:193–207, 2017.
- [TSJ08] A Tahbaz-Salehi and A Jadbabaie. A Necessary and Sufficient Condition for Consensus Over Random Networks. *Automatic Control, IEEE Trans. on*, 53(3):791–795, 2008.
- [TV98] P Thagard and K Verburgt. Coherence as Constraint Satisfaction. *Cognitive Science*, 22(1):1–24, 1998.
- [UAKM09] Soumya Kar Usman A Khan and Jose M F Moura. Higher Dimensional Consensus Algorithms in Sensor Networks. In *In Proc. of 2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, 2009.
- [VZ16] M E Valcher and I Zorzan. New results on the solution of the positive consensus problem. *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 5251–5256, 2016.
- [WCFHV17] Jian Wu, Francisco Chiclana, Hamido Fujita, and Enrique Herrera-Viedma. A visual interaction consensus model for social network group decision making with trust propagation. *Knowledge-Based Systems*, 122:39–50, 2017.

- [WDM15] A Wright and F P De Mar. Decentralized Blockchain Technology and the Rise of Lex Cryptographia. *Accessed: Apr 2018 [Online]*, 10:2018, 2015.
- [WGAH14] Q Wang, H Gao, F Alsaadi, and T Hayat. An overview of consensus problems in constrained multi-agent coordination. *Systems Science and Control Engineering*, 2(1):275–284, 2014.
- [WMT08] James Ze Wang, Iven M. Y. Mareels, and Yap-Peng Tan. Robustness of distributed multi-agent consensus . In *Proceedings of the 17th World Congress The International Federation of Automatic Control*, 2008.
- [WS98] Duncan J Watts and Steven H. Strogatz. Collective dynamics of “small-world” networks. *Nature*, 393(6684):440–442, 1998.
- [WT16] J Z Mareels I Wang and Y Tan. Robustness of Distributed Multi-Agent Consensus. *IFAC Proceedings Volumes*, 41(2):1510–1515, 2016.
- [Wu07] Chai Wah Wu. *Synchronization in Complex Networks of Non-linear Dynamical Systems*. Word Scientific Publishing Co., Singapore, 2007.
- [WYW⁺12] Leting Wu, Xiaowei Ying, Xintao Wu, Aidong Lu, and Zhi-Hua Zhou. Examining spectral space of complex networks with positive and negative links. *IJSNM*, 1(1):91–111, 2012.
- [WZLJ16] Jianshe Wu, Long Zhang, Yong Li, and Yang Jiao. Partition signed social networks via clustering dynamics. *Physica A: Statistical Mechanics and its Applications*, 443:568–582, 2016.
- [Xia05] Lin Xiao. A scheme for robust distributed sensor fusion based on average consensus. In *Proc. IPSN*, pages 63–70, 2005.
- [YCL07] Bo Yang, William Cheung, and Jiming Liu. Community Mining from Signed Social Networks. *IEEE Transactions on Knowledge and Data Engineering*, 19(10):1333–1348, 2007.
- [YLY14] K. Yuan, Q. Ling, and W. Yin. On the convergence of decentralized gradient descent. Technical Report Report 13-61, UCLA CAM, 2014.

- [ZD18] Tianqiao Zhao and Zhengtao Ding. Distributed Agent Consensus-Based Optimal Resource Management for Microgrids. *IEEE Transactions on Sustainable Energy*, 9(1):443–452, 2018.
- [Zea11] Filippo Zanella et al. Newton-Raphson consensus for distributed convex optimization. In *Proc. of IEEE CDC-ECC’11*, pages 5917–5922, 2011.
- [Zea12a] Filippo Zanella et al. Asynchronous Newton-Raphson Consensus for Distributed Convex Optimization. In *Proc. of IFAC NecSys’12*, 2012.
- [Zea12b] Filippo Zanella et al. Multidimensional Newton-Raphson consensus for distributed convex optimization. In *Proc. of ACC’12*, pages 1079–1084, 2012.
- [ZML18] Xiaoyu Zhu, Yinghong Ma, and Zhiyuan Liu. A novel evolutionary algorithm on communities detection in signed networks. *Physica A: Statistical Mechanics and its Applications*, 503:938–946, 2018.
- [ZSL18] Qiu Zhirong, Liu Shuai, and Xie Lihua. Necessary and sufficient conditions for distributed constrained optimal consensus under bounded input. *International Journal of Robust and Nonlinear Control*, 28(6):2619–2635, 2018.
- [ZTK09] Jiandong Zhu, Yu-Ping Tian, and Jing Kuang. On the general consensus protocol of multi-agent systems with double-integrator dynamics. *Linear Algebra and Its Applications*, 431(5-7):701—715, 2009.