Nuevos algoritmos metaheurísticos para el análisis de la influencia de los usuarios en las redes sociales

Isaac Lozano-Osorio
Universidad Rey Juan Carlos
Madrid, Spain
isaac.lozano@urjc.es

Jesús Sánchez-Oro
Universidad Rey Juan Carlos
Madrid, Spain
jesus.sanchezoro@urjc.es

Abraham Duarte
Universidad Rey Juan Carlos
Madrid, Spain
abraham.duarte@urjc.es

Óscar Cordón
Universidad de Granada
Granada, Spain
ocordon@decsai.ugr.es

***Resumen*—La evolución y la difusión de las redes sociales han atraído el interés de la comunidad científica en los últimos años. En concreto, han surgido nuevos problemas interesantes y difíciles de resolver. En el contexto del marketing viral las empresas y los investigadores tratan de encontrar los elementos que maximizan los beneficios, incrementan visualizaciones, etc. Esta familia de problemas suele conocerse como problemas de maximización de la influencia en redes sociales (SNIM, del inglés *Social Network Influence Maximization*), cuyo objetivo es encontrar los usuarios más influyentes (comúnmente conocidos como semillas) en la red social, simulando un modelo de difusión de influencia. Como se sabe que SNIM es un problema $\mathcal{NP}$-difícil, y la mayoría de las redes a analizar son considerablemente grandes, este problema se ha convertido en un verdadero reto para la comunidad científica. Diferentes trabajos han intentado resolverlo mediante enfoques heurísticos o metaheurísticos, como los algoritmos evolutivos basados en simulaciones del mecanismo de propagación, pero no son capaces de resolver redes de gran tamaño del mundo real debido a sus limitaciones en tiempo de computación. El principal inconveniente de este problema de optimización reside en el esfuerzo computacional necesario para evaluar una solución. Dado que la infección de un nodo se evalúa de forma probabilística, el valor de la función objetivo requiere de una simulación de Monte Carlo para analizar la robustez del método, resultando un proceso computacionalmente complejo. Nuestra propuesta trata de superar estas limitaciones considerando un algoritmo metaheurístico basado en el marco del *Greedy Randomized Adaptive Search Procedure* (GRASP). Esta metodología se divide en dos fases: la fase de construcción, basada en características estáticas de la red para aumentar su eficiencia al no requerir realizar ninguna simulación, y la fase de optimización local, que consiste en una búsqueda local limitada para encontrar a los usuarios más influyentes basándose en movimientos de intercambio. Los experimentos realizados en 6 conocidos conjuntos de datos de redes sociales con 5 tamaños diferentes de conjuntos de semillas confirman que el algoritmo propuesto es capaz de proporcionar resultados competitivos en términos de calidad y tiempo de computación al compararlo con los mejores algoritmos encontrados en el estado del arte.**

***Index Terms*—Sistemas de información, redes sociales, maximización de la influencia, teoría de grafos, marketing viral, GRASP.**

I. Introducción

Hoy en día, las redes sociales (RRSS) son utilizadas diariamente por millones de usuarios, mientras que el número de usuarios sigue creciendo exponencialmente. Este crecimiento se debe a la cantidad de datos generados por los usuarios y, por lo tanto, los problemas clásicos relacionados con las RRSS se están volviendo computacionalmente más difíciles. Una red social puede definirse como la representación de las interacciones sociales que puede utilizarse para estudiar la propagación de ideas, la detección de grupos, la dinámica de los vínculos sociales, la propagación de enfermedades, el marketing viral o la publicidad, entre otros [1, 2, 3, 4, 5]. Formalmente, una red social se modela con un grafo $G(V, E)$ donde el conjunto de nodos V representa a los usuarios y cada relación entre dos usuarios se modela como un par $(u, v) \in E$, con $u, v \in V$ que indica que el usuario u puede transmitir información (o infectar) al usuario v.

Kempe et al. [6] formalizaron originalmente el modelo de influencia para analizar cómo se transmite la información entre los usuarios de una red social. Dada una red con n nodos donde las aristas representan el proceso de difusión o propagación en dicha red, la tarea consiste en elegir un conjunto de semillas S de tamaño $k < n$ con el objetivo de maximizar el número de nodos de la red que son influenciados por el conjunto de semillas S.

Este problema se formuló inicialmente en [7] y posteriormente se demostró que es $\mathcal{NP}$-difícil para la mayoría de los modelos de difusión de influencia (MDI) [8]. Al igual que con muchos otros problemas $\mathcal{NP}$-difíciles, se han considerado algoritmos heurísticos y metaheurísticos, como los algoritmos voraces y evolutivos, para resolver el problema explorando eficazmente el espacio de soluciones, evitando el análisis de cada posible subconjunto de nodos.

La información representa todo aquello que puede ser transmitido entre pares conectados dentro de una red. El nivel de influencia de un nodo viene determinado por el proceso de adopción o propagación. Se pueden encontrar varios MDI en la literatura, pero el más extendido es el Modelo de Cascada Independiente [6], un MDI probabilístico donde la arista (u, v) tiene un peso P que representa la probabilidad con la que u es capaz de influir en v.

Este trabajo presenta una novedosa aproximación metaheurística para abordar este problema, que nos permite encontrar soluciones de alta calidad en el contexto del SNIM en tiempos de computación reducidos. Nuestro principal objetivo es diseñar un algoritmo eficiente en el que se minimice el uso de la simulación Monte Carlo necesaria para la aplicación del MDI, aumentando así la eficiencia del algoritmo.

El resto del trabajo está estructurado como sigue. La sección

2 formaliza el problema y revisa la literatura relacionada. El enfoque propuesto se describe con detalle en la sección 3. En la sección 4 se presentan los resultados experimentales considerando un conjunto de datos público que ha sido utilizado previamente para esta tarea. Los resultados obtenidos se comparan con los algoritmos más populares para SNIM y demuestran claramente la eficacia de la metodología propuesta. Por último, en la sección 5 se extraen algunas conclusiones derivadas de esta investigación.

II. Estado del arte

En esta sección se introducen algunos trabajos relacionados con SNIM y MDI y se realiza un breve estudio de los métodos existentes para resolver este problema, basados en heurísticas o en algoritmos de inteligencia computacional.

Richardson y Domingos [7] formularon inicialmente el problema de selección de nodos objetivo en RRSS. Sin embargo, Kempe et al. [6] intentaron resolver el problema SNIM, formulándolo como un problema de optimización discreto. Se ha demostrado que el cálculo de la propagación de la influencia de un conjunto de nodos dado es $\mathcal{NP}$-difícil [8]. Kempe et al. [6] propusieron un algoritmo voraz llamado *greedy hill-climbing algorithm* con una aproximación de $1-1/e$. Además, también demostraron que este algoritmo voraz puede obtener una solución dentro del 63 % de la optimalidad bajo tres MDIs comúnmente utilizados como el Modelo de Cascada Independiente (MCI), el Modelo de Cascada Ponderada (MCP), y el modelo de Umbral Lineal (UL). El MCI es uno de los MDI más populares y es el que se utiliza en esta contribución. Considera la transmisión de la influencia a través de la red en forma de árbol, donde los nodos semilla son las raíces.

La forma más extendida de evaluar la propagación en MCI es una simulación de Monte Carlo. Sin embargo, incluso una sola iteración de la simulación en MCI consume bastante tiempo. El algoritmo 1 muestra el pseudocódigo de la simulación Monte Carlo para evaluar la propagación de la información a través de la red dado un conjunto de semillas. El algoritmo realiza un número de iteraciones predefinidas IT, encontrando en cada iteración cuáles son los nodos que reciben la información dado un conjunto de semillas S (pasos $2-18$). Inicialmente, el conjunto de nodos $A^\star$ alcanzado por el conjunto semilla inicial, S, es realmente el conjunto semilla (paso 3). A continuación, el método itera hasta que no se infecten nuevos nodos (pasos 5-16). En cada iteración, se recorren los nuevos nodos infectados para comprobar si son capaces de transmitir información a los nodos no infectados (pasos 8-12). Para cada nodo no infectado adyacente a uno de los nuevos nodos infectados, se genera un número aleatorio. Si este número es menor que la probabilidad de infección P, entonces se infecta (pasos 9-11). Al final de la iteración, se actualiza el conjunto de nodos infectados (paso 14) así como los nodos infectados en la iteración anterior (paso 15). Finalmente, el algoritmo devuelve el número medio de nodos infectados durante el número predefinido de iteraciones (paso 19). Este valor, llamado *difusión de la influencia*, se considera como la función objetivo a optimizar por el algoritmo voraz. Es decir, el conjunto de semillas que maximiza el valor de propagación compondría la solución óptima del problema. Obsérvese que, como la infección es un proceso estocástico, el MCI debe ejecutarse varias veces para lograr una estimación adecuada de la media de la variable, lo que da lugar a una simulación de Monte Carlo.

Como consecuencia de coste computacional, Kempe et al [6] también propusieron varias heurísticas voraces basadas en medidas de análisis de redes sociales como la centralidad de grado y de centralidad de cercanía [9]. Cuando la medida considerada es el grado del nodo, el algoritmo se denomina *heurística de alto grado*.

Algorithm 1 $MCI(G=(V,E),S,P,IT)$

```
1:  influencia ← ∅
2:  for i ∈ 1...IT do
3:      A* ← S
4:      A ← S
5:      while A ≠ ∅ do
6:          B ← ∅
7:          for v ∈ A do
8:              for (u,v) ∈ E do
9:                  if rnd(0,1) ≥ p then
10:                     B ← B ∪ u
11:                 end if
12:             end for
13:         end for
14:         A* ← A* ∪ B
15:         A ← B
16:     end while
17:     influencia ← influencia + |A*|
18: end for
19: return influencia/IT
```

Posteriormente se propusieron varias extensiones de esos primeros algoritmos voraces. En particular, Leskovec et al. [10] introdujeron la selección perezosa rentable (CELF), que explota la propiedad de submodularidad para reducir significativamente el tiempo de ejecución del algoritmo *greedy hill-climbing*, llegando a ser más de 700 veces más rápido que el algoritmo *greedy hill-climbing*. La razón es que la expansión de cada nodo se calcula a priori y sólo es necesario volver a calcularla para unos pocos nodos. Por otra parte, Chen, Wang y Yang [11] utilizaron el concepto de heurística de descuento de grado para optimizar la heurística de alto grado. La función de selección voraz considera la redundancia entre los nodos probablemente influenciados y no cuenta los alcanzados por los nodos semilla ya seleccionados para desarrollar una mejor estimación de la propagación total.

En [12] se propuso un nuevo algoritmo llamado CELF++ con el objetivo de mejorar la eficiencia del CELF original, siendo este un 35-55 % más rápido que CELF. En este trabajo se reimplementó cuidadosamente el algoritmo CELF++ siguiendo las instrucciones de [12] y se probó con diferentes instancias, confirmando los resultados publicados en [13].

Analizando el estado del arte, los enfoques metaheurísticos más complejos suelen dar lugar a mejores soluciones que los enfoques voraces simples. Yang et al. [14] propusieron un algoritmo de optimización de colonias de hormigas basado en un modelo probabilístico parametrizado para abordar el problema SNIM. Utilizaron los métodos de centralidad de grado, centralidad de distancia e influencia simulada para determinar los valores heurísticos.

Mientras tanto, el método basado en temple simulado presentado en [15] aplica dos métodos heurísticos para acelerar el proceso de convergencia, junto con un nuevo método de cálculo de la propagación de la influencia para acelerar el algoritmo propuesto. En [16], se propuso un algoritmo de Optimización Evolutiva Multiobjetivo [17] para SNIM, donde los dos objetivos considerados eran (i) maximizar la influencia de un conjunto de semillas, y (ii) minimizar el número de nodos en el conjunto de semillas.

Como se ha dicho, se han propuesto algunas heurísticas como soluciones para ahorrar tiempo en las decisiones voraces: aleatoria, de grado y de centralidad [6]. La heurística aleatoria selecciona los nodos al azar, sin considerar la influencia de los nodos, para formar el conjunto de semillas en la red. Las heurísticas de grado y centralidad se derivan de la definición de la influencia de los nodos en el análisis de redes sociales [9]. La heurística de centralidad de grado suele producir resultados menos precisos para el problema SNIM. Basada en la heurística de alto grado, otra heurística, se dirige al problema SNIM teniendo en cuenta el conocimiento previo de los vecinos del nodo [11].

Los métodos metaheurísticos mencionados son capaces de obtener buenos resultados pero requieren grandes tiempos computacionales. Nuestra propuesta considera un método equilibrado basado en el MCI. Con el objetivo de reducir el tiempo de computación requerido, consideramos la metodología GRASP, que combina buenas soluciones heurísticas que se generan rápidamente y un método de búsqueda local eficiente que minimiza el número de evaluaciones de Monte Carlo para mejorar la solución inicial.

III. Greedy Randomized Adaptive Search Procedure

Greedy Randomized Adaptive Search Procedure (GRASP) es una metaheurística desarrollada a finales de los años 80 [18] e introducida formalmente en Feo et al. [19]. Remitimos al lector a [20, 21] para un estudio completo de los últimos avances en esta metodología. GRASP es una metaheurística multi-arranque dividida en dos etapas distintas. La primera es una construcción voraz, aleatoria y adaptativa de una solución. La segunda etapa aplica un método de mejora para obtener un óptimo local con respecto a una determinada vecindad a partir de la solución construida. Estas dos fases se repiten hasta que se cumple un criterio de terminación, devolviendo la mejor solución encontrada.

III-A. Fase de construcción

La fase de construcción de GRASP se dedica a generar una solución inicial y suele estar guiada por una función de selección voraz que ayuda al método constructivo a seleccionar los siguientes elementos a incluir en la solución parcial. El algoritmo 2 representa el pseudocódigo del procedimiento constructivo propuesto.

El método parte de una solución vacía S (paso 1), creando la lista de candidatos, LC, con el conjunto completo de vértices V (paso 2). A continuación, el método constructivo añade iterativamente nuevos elementos a la solución hasta que sea factible (pasos 3-11). En cada iteración, se evalúa el valor mínimo y máximo de la heurística voraz (pasos 4-5). A continuación, se calcula un umbral μ (paso 6). Este umbral es necesario para crear la Lista Restringida de Candidatos (LRC) con los nodos más prometedores (paso 7). Finalmente, se selecciona el siguiente nodo al azar de la LRC (paso 8), incluyéndolo en la solución (paso 9) y actualizando la LC (paso 10). El método termina devolviendo la solución construida S (paso 12).

Algorithm 2 $GRASP(G = (V, E), \alpha)$

1: $S \leftarrow \emptyset$
2: $LC \leftarrow$ V
3: **while** $|S| < K$ **do**
4: $\quad g_{min} \leftarrow min_{u \in LC} g(u)$
5: $\quad g_{max} \leftarrow max_{u \in LC} g(u)$
6: $\quad \mu \leftarrow g_{max} - \alpha \cdot (g_{max} - g_{min})$
7: $\quad LRC \leftarrow v \in LC : g(v) \geq \mu$
8: $\quad u \leftarrow rnd(LRC)$
9: $\quad S \leftarrow S \cup \{u\}$
10: $\quad LC \leftarrow LC \setminus \{u\}$
11: **end while**
12: **return** S

La función heurística voraz (usada en los pasos 4-5) es una de las características que mejorarán la solución GRASP. La aproximación que utilizamos en este problema es una heurística basada en los vecinos de primer y segundo grado de un determinado usuario, normalmente conocido como 2 pasos en el análisis de RRSS. Dado un usuario u, este método calcula la suma del grado de salida de u, d_u^+ y el grado de salida de los vecinos descartando los nodos repetidos. En el gráfico representado en la figura 1, el valor heurístico del nodo 1 será $d_1^+ + d_5^+ = 1 + 3 = 4$, el valor heurístico del nodo 5 es 5 debido a $d_1^+ + d_2^+ + d_3^+ + d_4^+ + d_5^+ + d_6^+ = 0 + 1 + 0 + 0 + 4 = 5$. Por lo tanto, el valor de cada nodo se calcula como $[4, 5, 4, 4, 5, 2, 1, 1]$. Luego se elige el nodo con el valor más alto y se actualizan los nodos restantes restando el grado del nodo a sus vecinos (ignorando las aristas con usuarios que han sido seleccionados). En el ejemplo, el usuario seleccionado será el 5, porque es el valor más alto, y los grados de los nodos se recomponen como $[0, 1, 1, 1, X, 1, 1, 1]$. Cabe destacar que el nodo 5 ya está seleccionado y, por tanto, no es necesario calcularlo.

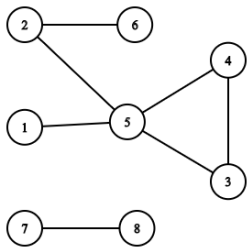

Figura 1. Figura de ejemplo de un grafo con 8 usuarios y 7 relaciones entre ellos para mostrar la función heurística propuesta para evaluar a cada usuario.

III-B. Fase de búsqueda local

La segunda fase de GRASP consiste en mejorar la solución generada por el procedimiento constructivo con el objetivo de alcanzar un óptimo local. Esta fase puede llevarse a cabo utilizando procedimientos de búsqueda local simples o heurísticas más complejas. Debido a la complejidad del problema considerado, proponemos un procedimiento de búsqueda local limitada.

Para definir correctamente una búsqueda local, primero tenemos que introducir la vecindad considerada. La vecindad de una solución puede definirse como el conjunto de nuevas soluciones que pueden alcanzarse realizando un único movimiento sobre la solución original. Por lo tanto, hay que definir un movimiento para presentar la vecindad. En el contexto de este problema, se considera el movimiento de intercambio, que sustituye un nodo seleccionado por otro no seleccionado. Este movimiento de intercambio se define formalmente como:

$$Swap(S, v, u) = S \setminus \{v\} \cup \{u\}$$

Entonces, la vecindad N_s de una solución dada S se define como el conjunto de soluciones que se pueden alcanzar realizando un único movimiento de intercambio a S. Más formalmente,

$$N_s(S) = \{Swap(S, v, u) \forall\ v \in S \wedge \forall\ u \in V \setminus S\}$$

Una vez definida la vecindad, hay que aclarar finalmente en qué orden se explora. En concreto, proponemos recorrer aleatoriamente la vecindad siguiendo un enfoque de *first improvement*, donde se realiza el primer movimiento de mejora.

Debido al gran tamaño del vecindario resultante, $k \times (n-k)$, se presenta una búsqueda local limitada con el objetivo de reducir el número de soluciones exploradas dentro de cada vecindario. El espacio de búsqueda explorado se limita reduciendo el porcentaje de nodos considerados para el movimiento de intercambio.

El número N de soluciones vecinas a explorar para cada solución se define mediante el mínimo entre el tamaño de la vecindad y un parámetro de búsqueda χ, es decir, $\min(|N_s(S)|, \chi)$. Cuanto mayor sea el valor de χ, mejor será la calidad de las soluciones, pero también será mayor el tiempo de cálculo necesario para alcanzarlas. El valor de χ se probó como {10, 25, 30}.

IV. EXPERIMENTOS

Esta sección describe el diseño experimental propuesto para validar esta aproximación y muestra los resultados obtenidos. La información relevante sobre las instancias utilizadas se recoge en la tabla I. Todas ellas están disponibles públicamente en Stanford Network Analysis Project (SNAP): https://snap.stanford.edu/. Todos los experimentos se han realizado en un ordenador con un Intel i7-4720HQ (2,50 GHz) y 16GB RAM.

Tabla I
INSTANCIAS

Instancia	***Nodos***	***Aristas***	***Diámetro***
p2p-Gnutella31	62586	147892	11
ca-AstroPh	18772	198110	14
ca-CondMat	23133	93497	14
cit-HepPh	34546	421578	12
email-Enron	36692	183831	11
email-EuAll	265214	420045	14

En primer lugar, es importante mostrar los valores utilizados para el algoritmo MCI con Monte Carlo y el número de iteraciones realizadas para obtener el mejor valor con GRASP. En todos los experimentos, GRASP construye 100 soluciones y MCI se ejecuta durante 100 iteraciones (IT) con una probabilidad de infección (P) de 0,01 %. Estos valores de los parámetros son los más extendidos en la literatura. El número de nodos semilla K para conformar una solución se selecciona en el rango $K = \{10, 20, 30, 40, 50\}$, obteniendo así $6 \times 5 = 30$ instancias de problemas diferentes (6 redes y 5 tamaños de conjuntos de semillas).

Los experimentos se dividen en dos partes: la experimentación preliminar y la final. La primera se refiere a los experimentos realizados para seleccionar los mejores parámetros para configurar el algoritmo, mientras que la segunda valida la mejor configuración, comparándola con los mejores métodos encontrados en el estado del arte.

Todos los experimentos reportan las siguientes métricas: Promedio, el valor promedio de la función objetivo; Tiempo (s), el tiempo de cómputo promedio requerido por el algoritmo en segundos; Desv(%), la desviación media con respecto a la mejor solución encontrada en el experimento; y # Mejores, el número de veces que el algoritmo encuentra la mejor solución del experimento.

La experimentación preliminar se ha realizado con un pequeño conjunto de 10 instancias (CA-AstroPh y CA-CondMat con $K = \{10, 20\}$; y Email-Enron y Email-EuAll con $K = \{30, 40, 50\}$) para evitar el sobreajuste. Esta selección abarca un 33,33 % del conjunto global y proporciona suficiente variabilidad en instancias y valores de K.

El propósito del experimento preliminar es establecer el valor de α en la fase constructiva del GRASP, probando los valores $\alpha = \{0.25, 0.5, 0.75, RND\}$, donde RND indica que se elige un valor aleatorio para α en cada construcción. Los resultados obtenidos se muestran en la tabla II. Como se puede observar, el constructivo con el valor de α aleatorio proporciona los mejores resultados tanto en calidad como en

tiempo de ejecución, por lo que es el valor seleccionado para utilizar en la fase de construcción.

Tabla II
RESULTADOS OBTENIDOS EN LA COMPARATIVA PARA ENCONTRAR EL MEJOR VALOR DE α EN LA FASE DE CONSTRUCCIÓN.

α	Promedio	Tiempo (s)	Desv(%)	# Mejores
0.25	406.58	**20.87**	2.12	7
0.50	407.29	23.23	1.87	7
0.75	409.23	28.42	1.02	9
RND	**411.75**	20.92	**0.00**	**10**

La evaluación comparativa entre el mejor diseño GRASP y CELF se reporta en la tabla III. Obsérvese que el tiempo de ejecución es menor para GRASP en todos los casos excepto en K=50 donde es mayor pero a costa de obtener una solución más precisa que CELF. En todas las ejecuciones, la calidad es igual o superior y el tiempo de ejecución es muy competitivo. La última columna muestra el número de instancias donde cada algoritmo, CELF y GRASP, obtiene el mejor resultado.

Tabla III
GRASP VS CELF

Algoritmo	K	Promedio	Tiempo (s)	Desv(%)	# Mejores
CELF	10	243.67	930.53	**0.00**	**6**
GRASP		**243.50**	**49.38**	0.20	5
CELF	20	289.33	890.46	0.06	**5**
GRASP		**289.17**	**252.96**	**0.05**	4
CELF	30	324.50	1209.12	**0.64**	4
GRASP		**325.50**	**570.39**	0.69	**5**
CELF	40	351.33	1409.75	1.65	2
GRASP		**358.00**	**1100.97**	**0.10**	**5**
CELF	50	376.00	**1157.66**	2.01	3
GRASP		**386.83**	1539.02	**0.09**	**5**

La figura 2 ilustra gráficamente que la calidad de GRASP crece más que CELF cuando el tamaño del conjunto de semillas aumenta, pero también podemos ver en la figura 3 cómo el tiempo de ejecución también crece y para $K = 50$ GRASP tiene un tiempo de ejecución mayor que CELF por primera y única vez.

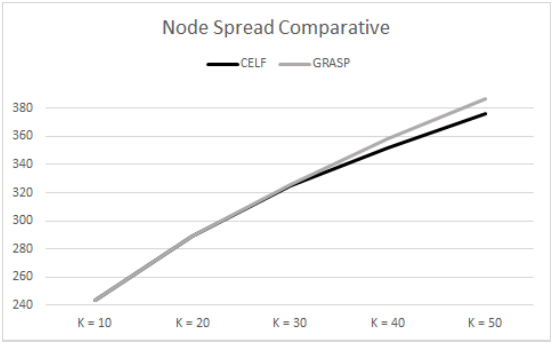

Figura 2. Influencia en CELF vs GRASP

Para confirmar que existen diferencias estadísticamente significativas entre ambos algoritmos, hemos realizado la conocida prueba no paramétrica de Wilcoxon. El valor p resultante, inferior a 0.001, confirma estas diferencias y, por tanto, la superioridad de la propuesta.

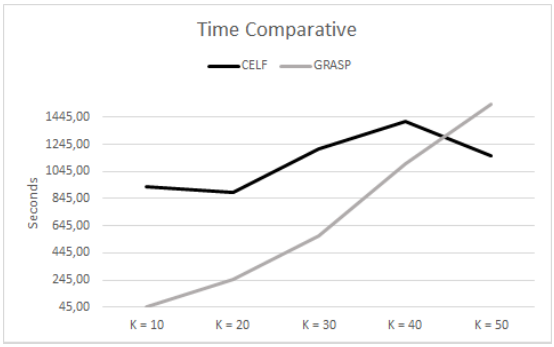

Figura 3. Tiempo en CELF vs GRASP

CONCLUSIONES

En este trabajo se ha propuesto un algoritmo GRASP para el problema SNIM. Se ha diseñado una fase de construcción eficiente que reduce el número de simulaciones de Monte Carlo necesarias, así como una búsqueda local limitada para reducir el número de soluciones a explorar y obtener mejores soluciones tanto en términos de calidad como de tiempo de computación. La función heurística proporciona flexibilidad en el diseño del método GRASP para alcanzar un compromiso entre la calidad y el tiempo de computación. GRASP ofrece resultados muy competitivos respecto a CELF, reduciendo drásticamente el tiempo de computación y proporcionando mejores soluciones en promedio.

Por tanto, se puede concluir que es necesario un diseño algorítmico cuidadoso en SNIM para poder tratar con grandes redes sociales sin requerir grandes tiempos de computación. La búsqueda local limitada presentada en este trabajo es capaz de equilibrar la calidad y el tiempo de computación mediante un único parámetro, lo que la hace adecuada para la mayoría de las redes sociales.

AGRADECIMIENTOS

Este trabajo ha sido parcialmente financiado por el "Ministerio de Ciencia, Innovación y Universidades" con la subvención ref. PGC2018-095322-B-C22 y "Comunidad de Madrid" y "Fondos Estructurales" de la Unión Europea con las subvenciones ref. S2018/TCS-4566 y Y2018/EMT-5062.

REFERENCIAS

[1] Stephen F. King y Thomas F. Burgess. «Understanding success and failure in customer relationship management». En: *Industrial Marketing Management* 37.4 (2008), págs. 421-431. DOI: 10.1016/j.indmarman.2007.02.005.

[2] Adam D'angelo et al. *Targeting advertisements in a social network*. US Patent App. 12/195,321. 2009.

[3] Alden S. Klovdahl. «Social networks and the spread of infectious diseases: The AIDS example». En: *Social Science & Medicine* 21.11 (1985), págs. 1203-1216. DOI: 10.1016/0277-9536(85)90269-2.

[4] Sergio Pérez-Peló et al. «On the Analysis of the Influence of the Evaluation Metric in Community Detection over Social Networks». En: *Electronics* 8.1 (2019), pág. 23.

[5] Sergio Pérez-Peló, Jesús Sánchez-Oro y Abraham Duarte. «Finding weaknesses in networks using Greedy Randomized Adaptive Search Procedure and Path Relinking». En: *Expert Systems* (2020), e12540.

[6] David Kempe, Jon Kleinberg y Éva Tardos. «Maximizing the spread of influence through a social network». En: *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '03*. ACM Press, 2003. DOI: 10.1145/956750.956769.

[7] Matthew Richardson, Rakesh Agrawal y Pedro Domingos. «Trust Management for the Semantic Web». En: *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2003, págs. 351-368. DOI: 10.1007/978-3-540-39718-2_23.

[8] David Kempe, Jon Kleinberg y Eva Tardos. «Maximizing the Spread of Influence through a Social Network». En: *Theory of Computing* 11.1 (2015), págs. 105-147. DOI: 10.4086/toc.2015.v011a004.

[9] Stanley Wasserman y Katherine Faust. *Social Network Analysis*. Cambridge University Press, 1994. DOI: 10.1017/cbo9780511815478.

[10] Jure Leskovec et al. «Cost-effective outbreak detection in networks». En: *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '07*. ACM Press, 2007. DOI: 10.1145/1281192.1281239.

[11] Wei Chen, Yajun Wang y Siyu Yang. «Efficient influence maximization in social networks». En: *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '09*. ACM Press, 2009. DOI: 10.1145/1557019.1557047.

[12] Amit Goyal, Wei Lu y Laks V.S. Lakshmanan. «CELF++». En: *Proceedings of the 20th international conference companion on World wide web - WWW '11*. ACM Press, 2011. DOI: 10.1145/1963192.1963217.

[13] Jiaguo Lv et al. «Improved Algorithms OF CELF and CELF++ for Influence Maximization». En: *Journal of Engineering Science and Technology Review* 7.3 (2014), págs. 32-38. DOI: 10.25103/jestr.073.05.

[14] Wan-Shiou Yang y Shi-Xin Weng. «Application of the Ant Colony Optimization Algorithm to Competitive Viral Marketing». En: *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2012, págs. 1-8. DOI: 10.1007/978-3-642-30448-4_1.

[15] Qing-lai Jiang et al. «Preparation of high active Pt/C cathode electrocatalyst for direct methanol fuel cell by citrate-stabilized method». En: *Transactions of Nonferrous Metals Society of China* 21.1 (2011), págs. 127-132. DOI: 10.1016/s1003-6326(11)60688-2.

[16] Doina Bucur et al. «Multi-objective Evolutionary Algorithms for Influence Maximization in Social Networks». En: *Applications of Evolutionary Computation*. Springer International Publishing, 2017, págs. 221-233. DOI: 10.1007/978-3-319-55849-3_15.

[17] Gary B. Lamont Carlos Coello Coello y David A. van Veldhuizen. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Springer US, 2007. DOI: 10.1007/978-0-387-36797-2.

[18] Thomas A Feo y Mauricio G.C Resende. «A probabilistic heuristic for a computationally difficult set covering problem». En: *Operations Research Letters* 8.2 (1989), págs. 67-71. DOI: 10.1016/0167-6377(89)90002-3.

[19] Thomas A. Feo, Mauricio G. C. Resende y Stuart H. Smith. «A Greedy Randomized Adaptive Search Procedure for Maximum Independent Set». En: *Operations Research* 42.5 (1994), págs. 860-878. DOI: 10.1287/opre.42.5.860.

[20] Mauricio G. C. Resende et al. «GRASP and path relinking for the max–min diversity problem». En: *Computers & Operations Research* 37.3 (2010), págs. 498-508. DOI: 10.1016/j.cor.2008.05.011.

[21] Mauricio G. C. Resende y Celso C. Ribeiro. «GRASP: Greedy Randomized Adaptive Search Procedures». En: *Search Methodologies*. Springer US, 2013, págs. 287-312. DOI: 10.1007/978-1-4614-6940-7_11.