Universidad Rey Juan Carlos

TESIS DOCTORAL

Design and Implementation of Metaheuristic Algorithms for Social Network Influence Problems

Autor:

Isaac Lozano Osorio

Directores:

Abraham Duarte Muñoz

Jesús Sánchez-Oro Calvo

Programa de Doctorado en Tecnologías de la Información y las Comunicaciones

Escuela Internacional de Doctorado

2024

This thesis had been founded by the following projects:

- Project "Cybersecurity, Network Analysis and Monitoring for the Next Generation Internet.", Comunidad de Madrid y Fondos Estructurales de la Unión Europea with grant ref. P2018/TCS-4566.
- Project "Metaheurísticas eficientes para la optimización en grafos", funded by Spanish Ministerio de Ciencia, Innovación y Universidades under grant ref. PGC2018-095322-B-C22.
- Project "Nueva metodología holística para la configuración, comparación y evaluación de metaheurísticas.", grant ref PID2021-126605NB-I00 funded by MCIN/AEI/10.13039/501100011033, and ERDF A way of making Europe.

Abuelos, Papá, Mamá y Luz María.
Esta Tesis es posible gracias a vosotros.

Agradecimientos

El camino durante estos últimos cuatro años no ha sido fácil, ha resultado en una montaña rusa de emociones: felicidad, tristeza, éxito, fracaso, frustración o satisfacción. Tampoco ha sido un trabajo individual, han participado una gran cantidad de personas, "si quieres llegar rápido, ve sólo, si quieres llegar lejos, ve acompañado". Gracias a todas ellas, de una u otra manera, me han marcado y me han hecho evolucionar, tanto en lo profesional como en lo personal, por lo que quiero agradecer cada una de sus aportaciones en las diferentes etapas que me han llevado a finalizar este trabajo.

No puedo empezar de otra forma que no sea agradeciendo a mis padres, a mi hermana y a Sicilia. Por haberme educado y enseñado el camino correcto. Por los esfuerzos realizados para que siempre, pasase lo que pasase, tuviera la oportunidad de dedicarme a lo que más me gusta, a pesar de resultar en un gran esfuerzo para vosotros. Como bien sueles decir papá, no pueden faltar todas esas estrellas que hoy iluminan el cielo, que hoy estarán orgullosos y sin ellos esto no hubiese sido posible. Sicilia, durante el último año la tesis doctoral, la paz que he sentido ha sido gracias a ti, esto se debe a tu constante apoyo y comprensión, gracias por estar siempre para cualquier cosa, pronto estoy seguro de que se romperá mi hipótesis.

Durante esta época y la distancia, hace difícil disponer de tiempo para dedicar a mis amigos de toda la vida. Quiero agradecer a Virginia, Mario, Noemí, Carlos, Jaime, Verdu, Luis e Iván por haber estado gran parte de mi vida, ya sean momentos de felicidad o tristeza. En este listado se incluyen también innumerables compañeros del mundo de ajedrez, a los que me gustaría destacar a José Miguel Pérez por todo lo que haces por mí cada día y a Clara Gallego con quien no solo comparto el ajedrez, también la investigación. Sin olvidarme de los compañeros de casa, Agustín, Carlos y Juan.

También quiero mencionar a Jorge Cuadrado y Guillermo Martín, junto a ellos descubrí qué es el emprendimiento, me formé y tuve una experiencia de la misma duración de la tesis con Muzpic.

No puede faltar, Estefanía Martín, al grupo Vortic3 y Lite. Ellos me ofrecieron mi primera oportunidad en el mundo de la investigación, a pesar de ser otra área diferente a la presente tesis, gracias a ellos pude asistir a mi primer congreso y realizar varias investigaciones como BlueThinking y TutoApp.

En la portada de la tesis doctoral se puede apreciar que no solo está mi nombre. También aparecen los que denomino mis "padres" académicos. Gracias por vuestro tiempo, vuestra dedicación y sobre todo por vuestra paciencia, sé que no resulto fácil a veces. Espero que el “diamante en bruto muy bruto” se haya pulido con el tiempo. Durante el desarrollo de la tesis he aprendido mucho de vosotros y no solo sobre optimización.

Sin el grupo de investigación GRAFO, su experiencia y ambiente dentro del mismo, estoy seguro de que hubiese sido inviable finalizar la tesis. Recuerdo el primer año en el despacho con Sergio Pérez y Raúl Martín. De hecho, quizás con el que he pasado más tiempo durante estos años es Sergio Cavero, casi todos los días nos venían a decir a altas horas de la tarde que iban a cerrar el departamental. Pero me gustaría agradecer a todos los compañeros del grupo: Alberto, Andrea, Alejandra, Alf, Antonio, Chema, Enrique, Eduardo, Juanma, Marcos, Nico, Rubén y Yuste. Permitidme incluir dentro de este listado también a las Anas, ellas forman parte de la familia GRAFO y también ha sido un placer trabajar con vosotras. Sin olvidarme de la familia laboral que compone el departamento de informática y estadística. En algún momento me han ayudado con mis gestiones o con los problemas que me han podido surgir: Almudena Sierra, Antonio Sanz, Juanjo Pantrigo y a todos aquellos buenos compañeros y docentes que he tenido durante todos estos años.

Quiero agradecer a David Dixon por ser mi guía en el mundo de la programación competitiva. Gracias a él, conocí qué es un problema de optimización, he tenido la oportunidad de viajar, participar en numerosos campeonatos y adquirir conocimientos que hoy en día son fundamentales en mi entorno laboral. Además, quiero agradecer a Raúl Martín, Sergio Pérez, Jakub, Sergio Salazar, Alicia, Sara y Raúl por ser compañeros de este viaje, aportando su valioso tiempo, apoyo y contribuyendo de manera significativa a mi crecimiento personal y profesional.

No puedo olvidarme del otra área que durante los últimos años ha tenido gran importancia en mi día a día, la ciberseguridad. De hecho, las primeras clases que impartí en la universidad fueron del Grado en Ingeniería de la Ciberseguridad. Gran parte de culpa la tienen de nuevo Raúl Martín y Sergio Pérez, ya que su implicación en las competiciones, conocimientos y pasión hacen que cualquier persona muestre interés en querer aprender sobre ciberseguridad. Gracias a ellos y a Sergio Cavero, Javier Yuste y Jakub hemos tenido la suerte de ganar/participar en múltiples competiciones como la National Cyber League, Bug Bounty de Meta, BitUp o JNIC. También he tenido la suerte de conocer, compartir tiempo y aprender de quienes ya están siendo grandes profesionales del área y estoy seguro de que les queda un gran futuro: Diego, Ismael, Pablo, etc.

Gracias a Kenneth Sorensën por su hospitalidad, generosidad y dedicación durante la última fase de la tesis realizada en Amberes (Bélgica), ha sido un placer trabajar con un investigador de su renombre y calidad. Esta experiencia no hubiese sido posible sin la acogida de Felipe y Mylena quienes me acogieron como uno más dentro de un departamento diferente.

Hace ya muchos años que comencé a estudiar, durante este viaje, la tesis doctoral es una estación que marca un nuevo inicio, estoy seguro de que rodeado de vosotros vendrá lleno de retos, aciertos, fallos, logros y aprendizaje.

Contents

List of Figures

List of Tables

List of Acronyms

ACO	Ant Colony Optimization.
APR	Adaptive Path Relinking.
BIMP	Budgeted Influence Maximization Problem.
CELF	Cost Effective Lazy Forward.
CL	Candidate List.
DPR	Dynamic Path Relinking.
EMO	Multi-objective Optimization.
EP	Evolutionary Path Relinking.
EPR	Exterior Path Relinking.
GA	Genetic Algorithm.
GRASP	Greedy Randomized Adaptive Search Procedure.
ICM	Independent Cascade Model.
IDM	Influence Diffusion Model.
IPR	Interior Path Relinking.
JCR	Journal Citation Reports.
LP	Linear Programming.
LTM	Linear Threshold Model.
MORK	Metaheuristic Optimization frameworRK.
OP	Optimization Problem.
PR	Path Relinking.

PSO	Particle Swarm Optimization.
RCL	Restricted Candidate List.
RPR	Reactive Path Relinking.
SA	Simulated Annealing.
SIM	Social Influence Maximization.
SJR	Scimago Journal & Country Rank.
SN	Social Network.
SNA	Stanford Network Analysis.
SNAP	Stanford Network Analysis Project.
SNI	Social Network Influence.
SNIMP	Social Network Influence Maximization Problem.
SPR	Static Path Relinking.
TSS	Target Set Selection.
TV	Tri-Valency Model.
VNS	Variable Neighborhood Search.
WCM	Weighted Cascade Model.

Abstract

Optimization has been a constant concern throughout history, from ancient Greeks seeking the most efficient way to organize cities to modern algorithms optimizing business processes. The significance of optimization lies in its ability to solve complex problems, enhance efficiency, and make informed decisions. Over centuries, optimization has proven to be fundamental for human progress.

Nowadays, optimization has gained even greater importance across various fields, owing to the increasing complexity of the challenges we face. From business logistics to route planning in navigation, optimization has become an essential tool for tackling ever-evolving issues. The ability to efficiently and accurately solve problems is crucial in an increasingly interconnected world that heavily relies on technology.

To address these challenges, there are various methodologies in optimization, including exact methods, approximations, genetic, and heuristic algorithms. These approaches offer flexible and adaptive solutions for a variety of problems, enabling researchers and professionals to find the best possible solution in different contexts.

This thesis focuses specifically on problems related to Social Network Analysis, an area of study that has gained prominence in the digital age. Within this discipline, various problems are identified, with particular attention directed towards the concept of influence. The central problem involves selecting users within a social network in a way that maximizes or minimizes influence on other users, considering potential constraints such as maximum budgets.

Defining influence within the context of social networks presents a significant challenge due to the diversity of available methods. The ability to strategically select users has practical applications in marketing campaigns, disease eradication, and the detection of misinformation campaigns. The complexity of these problems is exacerbated by the $\mathcal{NP}-hard$ nature of many of them, implying that finding exact solutions is impractical for large social networks.

While approximate algorithms exist, in certain cases, it is crucial to have quick and high-quality information, such as in disease detection. Therefore, this thesis focuses on the use of heuristics and metaheuristics to address influence problems in social networks. These approaches provide efficient and adaptable solutions, particularly in situations where speed and precision are paramount.

This thesis proposes different heuristic and metaheuristic algorithms to address the most widespread variants of influence problems in social networks. Various methodologies, such as Greedy Randomized Adaptive Procedure Search (GRASP) or Path Relinking (PR), have been applied and evaluated on real-world networks to verify their utility and applicability in these contexts. The results obtained surpass current proposals in all studied variants of social network influence problems.

Part I

PhD Dissertation

Contents

Chapter 1

Introduction

This Doctoral Thesis unfolds within the field of optimization. In both informal and scientific contexts, "optimization" is often synonymous with improvement. However, in the scientific realm, it assumes a more precise definition: "the systematic process of searching for the best solution to a given problem among a vast set of possibilities". More specifically, this document is focused on solving problems included in Social Network Influence (SNI) from an optimization point of view. This chapter contextualizes optimization on its fundamental principles and features and concludes with the presentation of the central hypothesis and objectives that propel the research forward.

1.1 Optimization

Optimization, rooted in the human pursuit of improvement, has a rich historical tapestry. In ancient Greece, mathematicians like Euclid grappled with optimization questions embedded in their geometric studies. Around 300 BC, Euclid explored problems such as determining the minimal distance between a point and a line. Additionally, his mathematical insights extended to proving that, among rectangles with a given total length of edges, a square possesses the maximum area.

The formalization of optimization problems gained prominence in the 18th and 19th centuries with the advent of calculus and mathematical analysis. The pioneering work in 1744 of Euler, Lagrange, and others provided a theoretical framework for describing and solving optimization problems [1], introducing the notion of a mathematical function. Lagrange's formulation of constrained optimization, known as Lagrangian optimization [2], marked a significant milestone in addressing problems subject to specific conditions.

As mathematical optimization advanced, real-world applications emerged, notably in engineering and economics. The industrial revolution spurred the need for optimizing manufacturing processes, and economists sought to maximize utility and profit within constrained resources. This historical backdrop underscores the dual nature of optimization—arising organically in practical challenges while simultaneously evolving as a formal discipline.

Fast forward to the 20th century, optimization became an integral part of operations research and management science. Linear Programming (LP) is arguably the key base methodology in mathematical optimization as we know it today. LP was first introduced by Leonid Kantorovich in 1939 [3] and then independently reintroduced by George Dantzig in 1947 [4]. The simplex method, developed by George Dantzig in 1947 [5], revolutionized linear programming, making it applicable to a wide range of practical problems. This period witnessed the transition from ad-hoc problem-solving to systematic algorithmic approaches.

In the contemporary landscape of optimization, a myriad of methods have proliferated, each catering to distinct problem domains and computational scenarios. Heuristics [6], which offer efficient, though not necessarily optimal, solutions, provide a flexible and practical approach to navigate combinatorial spaces. Metaheuristics [7], a more advanced class of algorithms, guide the search process by intelligently balancing exploration and exploitation. Approximation methods [8] in optimization are techniques that provide solutions that are close to the optimal solution without the guarantee of being absolutely optimal. Genetic algorithms [9], inspired by natural selection processes, demonstrate the efficacy of exploring solution spaces with a genetic-inspired search strategy. Mathematical models and metaheuristics [10] continue to complement each other (denoted matheuristics), creating a rich tapestry of methodologies to address optimization challenges. In this dynamic era, the optimization community witnesses a synthesis of classical algorithmic approaches with cutting-edge techniques, reflecting the evolving nature of problem-solving strategies.

Figure 1.1 provides an overview of the number of research works aimed to solve optimization problems published and indexed in the Web of Science directory per year, focusing on the research areas of Computer Science, Mathematics, and Operations Research Management Service [1].

Figure 1.1 illustrates the annual growth of optimization publications. The data show a consistent increase in the number of publications over the years, signaling a growing interest and relevance in research within this field. This pattern suggests a heightened focus from the scientific community, underscoring the significance and dynamism of optimization research.

Optimization problems

The definition of an Optimization Problem (OP) requires the concept of an objective function, often referred to as a cost function, utility function, or criterion, which is a mathematical expression that quantifies the performance, desirability, or effectiveness of a solution. The objective function serves as the guiding criterion for evaluating and comparing different solutions, helping in the systematic search for the optimal solution to the given problem.

An OP can be defined as the minimization or maximization of the value of an

[1]The data was generated on 22/11/2023 and consist of 287880 articles `https://www.webofscience.com/wos/woscc/summary/e6f055d0-e484-45e6-bcba-5877a7793a49-b719e1f5/sort-group-background-citingcount/1`

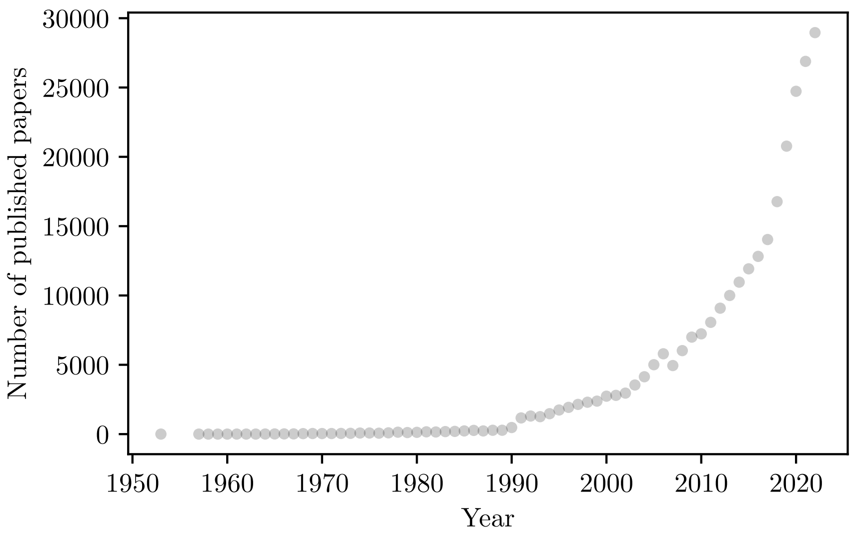


Figure 1.1: Number of published and indexed articles in Web of Science related to optimization field.

objective function $f(S)$, where S represents a solution in the set of feasible solutions SS, those which satisfy every problem constraint [11]. The aim in optimization is to either minimize or maximize this objective function, depending on the nature of the problem—minimization when seeking the smallest value, and maximization when pursuing the largest. Mathematically, the optimization problem can be succinctly expressed as follows:

$$\begin{gathered}\text{Minimize } f(S)\\ \text{subject to } S \in \mathcal{SS}\end{gathered} \tag{1.1}$$

When an optimization problem is addressed, the sought-after solution is termed the optimal solution. This optimal solution, within the feasibility bounds, is characterized by providing the minimum or maximum value for the objective function being assessed, depending on whether the problem involves minimization or maximization, respectively. Solving an optimization problem basically consists of finding the best values of a set of decision variables in terms of an objective function while satisfying some given constraints.

Figure 1.2 illustrates a possible graphical representation of a minimization optimization problem, as well as the most relevant concepts presented so far. Specifically, in this figure, the search space of an optimization problem is represented by the x-axis, which also contains all feasible solutions to the problem. The y-axis represents the value of the objective function associated with each of the solutions. As an example, two solutions S and $S^\star$ are illustrated, where $S^\star$ is also the optimal solution to the problem since there is no solution with a lower value of the objective function in the search space.

The preceding figure elucidates key concepts in optimization problems. A shoulder denotes a region on the optimization landscape where the objective function un-

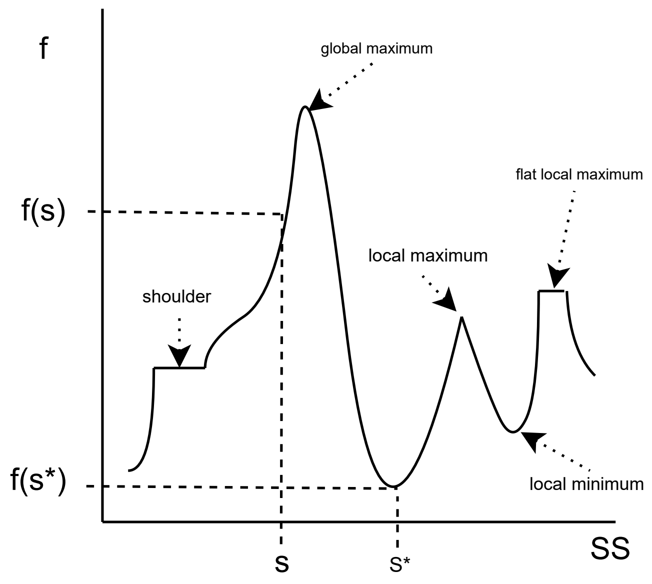


Figure 1.2: Graphical representation of an optimization problem where the x-axis represents the search space SS, and the y-axis represents the objective function value of each solution s, $s^\star \in SS$.

dergoes a gradual or slow change in value. Typically, it represents a flat or gently sloping area where the function remains relatively constant before encountering a more pronounced change, such as a peak or valley. Moreover, the visual representation highlights the global maximum, denoting the maximum optimal value. A local maximum is a point in the solution space where the objective function attains a value equal to or greater than the values at the surrounding points. However, note that a local maximum does not imply that the value at that point is the global maximum of the function across its entire domain, as depicted in the figure. Analogously, a local minimum follows a similar definition, but pertains to a point where the objective function reaches a minimum value. Lastly, a flat local maximum characterizes a scenario in which solutions yield nearly identical or similar objective function values, resulting in a relatively flat or plateau-like region. This scenario indicates that the objective function has attained its maximum possible values within the search space.

Optimization classification

The study of optimization has been conducted in various fields of study, with Mathematics and Computer Science being two of the most prominent ones. A traditional categorization of the different optimization problems, based on the type of variables and the mathematical expression of the objectives and constraints, divides Mathematical Programming into three main areas: Linear Programming, where the variables are real (continuous) numbers and the objectives and constraints are linear; Integer Linear Programming, where the variables are integers in addition; and Non-Linear Programming, where the variables are real numbers and there are no restrictions on the expression of objectives and constraints. This classification assumes that the mathematical expression of the objectives and constraints is known, which is not always the case. Many real-world problems in industry and business are difficult to formulate mathematically and they are challenging to solve because of their combinatorial nature.

Classifying optimization problems according to their computational complexity is a common practice. This is done by measuring the time it takes for a machine to solve the problem, usually using the Bachmann-Landau notation, also known as Big-O notation. This notation defines the time required by an algorithm to solve a certain problem in the worst case [12, 13, 14].

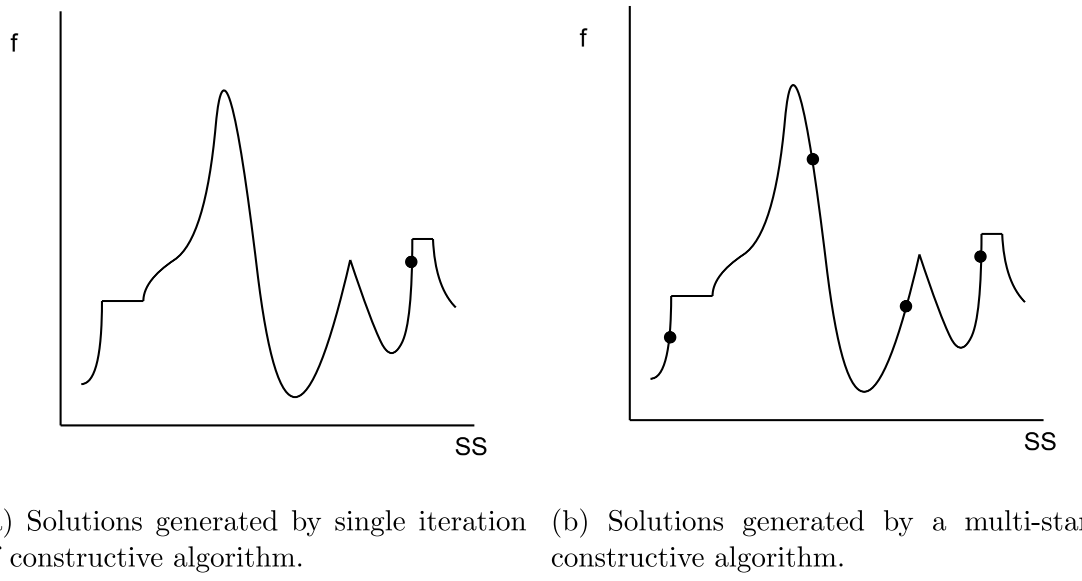
Using this notation, certain problems can be solved by an existing algorithm in polynomial time. The time an algorithm needs to solve a problem is related to its input, expressed as a polynomial function. This is denoted as $O(n^t)$, where t is the highest exponent in the polynomial. Problems that meet this criterion are classified as $\mathcal{P}$ problems, such as the Monte Carlo algorithm [15] introduced by Nicholas Metropolis in 1947. Most combinatorial optimization problems cannot be solved in a reasonable time, as no algorithm is known to obtain an optimal solution in polynomial time. These problems are part of the $\mathcal{NP}$ set, with the $\mathcal{P}$ problems being a subset. Another subset of $\mathcal{NP}$ problems is the $\mathcal{NP}$-complete set, where no polynomial algorithm exists to solve them, but a polynomial algorithm can verify if a solution solves the problem, such as the Traveling Salesman Problem [16]. Finally, the $\mathcal{NP}$-hard class includes all problems that do not have a polynomial algorithm to verify if a given solution solves the problem, for example the Social Network Influence Maximization Problem [17].

Heuristic and Metaheuristic methods in optimization

Heuristic methods [18] are a practical choice for quickly addressing problems, although they do not guarantee an optimal solution. This is the primary distinction between exact and heuristic methods. The exact algorithms are designed to find the best solution for a given problem [19], while heuristic algorithms do not provide this assurance. The main issue with exact algorithms is that many real-world problems are too complex to solve, or no exact algorithm is known to solve them. On the other hand, the main issue with heuristic algorithms is that they can easily become stuck in a local optimum of the solution space.

Generally, they start with an empty solution, gradually adding individual components (e.g., vertices, edges, variables) to eventually form a feasible solution. Among these heuristic construction algorithms, greedy algorithms are particularly proficient in producing high-quality solutions [20]. In a basic greedy construction, each step involves adding a solution component based on the optimal value determined by a defined heuristic function [21]. Despite the determinism of greedy construction, the importance of solution diversity for a thorough exploration of the solution space cannot be ignored. To address this, researchers have proposed semi-greedy heuristics [22] or incorporated randomization into the construction process [23]. Multiple construction of solutions is beneficial when the construction is not entirely deterministic, allowing exploration of different points in the search space. This strategy is especially useful when the knowledge gained from previous solutions helps generate subsequent solutions. However, caution should be taken when repeatedly constructing solutions in scenarios where the initial construction steps are computationally intensive compared to the subsequent steps, as it may not be time-efficient [21].

Figure 1.3 shows two figures related to constructive heuristic approaches. On the one hand, Figure 1.3a shows a greedy solution. On the other hand, Figure 1.3b shows multiple constructions with several generated solutions, where the one that minimizes the objective function the most will be selected.



(a) Solutions generated by single iteration of constructive algorithm.

(b) Solutions generated by a multi-start constructive algorithm.

Figure 1.3: Two constructive methods on a solution search space.

Typically, the solution obtained through constructive procedures serves as a starting point for improvement methods, such as local search [24, 25]. Local search begins with a feasible solution and then attempts to replace it with a better solution from a reduced set of solutions within the solution space. This subset of solutions, known as a neighborhood, encompasses all solutions achievable by specific moves (or a set of moves).

Figure 1.4 illustrates a single neighborhood with a solution S in the search space, which could be generated by a heuristic approach. A local search will try to improve this solution with two alternatives: the first improvement and the best improvement. The former, when an improvement is found, is accepted, and the search continues with this improvement, leading to the solution S_1, which is a local minimum. The latter is named best improvement, where all possibilities are explored before accepting a solution as an improvement, the solution S_2 is obtained, which is the global minimum.

Figure 1.5 displays two neighborhoods, represented by circles highlighted with different shades of gray gradient. These circles represent the domain of achievable solutions. Then, a solution S is depicted that belongs to a neighborhood different from the previous figure, so it is not possible to reach the global minimum. To do so, a neighborhood change can be performed, as shown in the solution S_2. Through a local search, the optimal solution S_5 can be reached. The solutions S_2, S_3, and S_4 show possible moves (new solutions that the local search can reach with a single move) before reaching the global minimum.

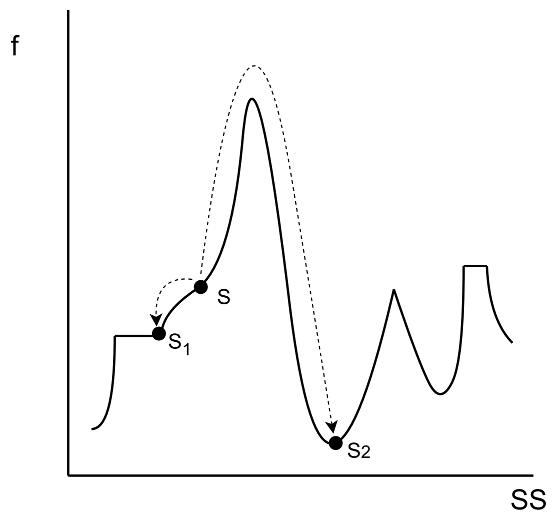


Figure 1.4: A local search with a solution representing a first-improvement and best-improvement local search.

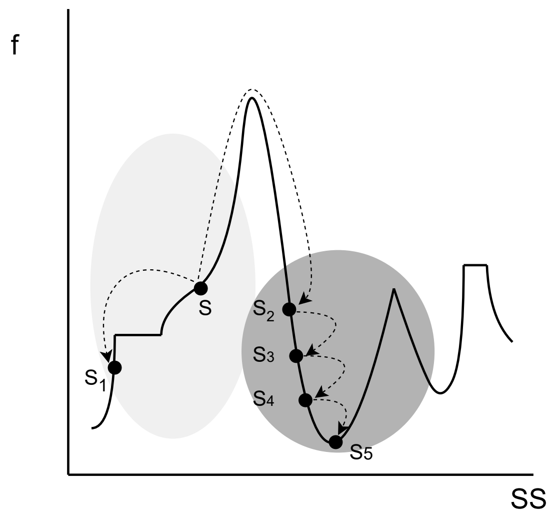


Figure 1.5: A neighbourhood change to achieve the global minimum through local search moves.

When dealing with large neighborhoods, termination criteria can be a significant factor to consider. Furthermore, detecting when it is time to end an improvement process to prevent over-exploring a region of the search space is a very effective strategy. Other approaches suggest reduction techniques to reduce the number of solutions in a neighborhood. The objective function is a key component of local search algorithms, since it is calculated multiple times to evaluate solutions within a neighborhood. In some cases, evaluating the objective function requires a long calculation time, thus, researchers have developed different approaches to address this critical issue. For instance, using approximations of the original objective functions or other simpler heuristics with similar characteristics. In this dissertation, this problem is addressed through an efficient or intelligent evaluation of the objective function, avoiding complete recalculation after a move (see Section 2.2.1). Another aspect to consider when applying local search strategies is the influence or sensitivity of the initial solution. In some cases, the quality of the final solution found by a local search algorithm can be highly dependent on the initial solution used as a starting point. This implies that only a few regions are explored, and it is necessary to introduce some diversity in the generated solutions. Therefore, local search algorithms must be able to balance the need to explore new solutions with the need to exploit the current best solution. If the algorithm focuses too much on exploitation, it may easily get stuck in a local optimum, while if it focuses too much on exploration, it may waste time exploring bad-quality solutions. This difficulty is closely related to the definition of a suitable neighborhood. In some cases, it may be difficult to define an appropriate neighborhood for a given problem, which can make it hard to use local search algorithms effectively.

Under this circumstance, metaheuristics emerge as a new kind of algorithms that guide the heuristics during their search, with the aim of escaping from local optima and improving the solutions found. The metaheuristic concept was introduced by Glover in 1986 [7] and was defined as an “strategy that guides and modifies other heuristics to explore solutions beyond local optimality”. In the same definition, Glover explains that “the heuristics guided by such a meta-strategy may be high-level procedures or may embody nothing more than a description of available moves for transforming one solution into another, together with an associated evaluation rule”.

Attending to this definition, metaheuristics are considered as a set of approximated algorithms that guide the heuristic procedure exploration in a smart way, combining intensification (exploitation) and diversification (exploration) of the search space of the problem under solution. By the intensification, a limited but promising region of the search space is explored, looking for improvements in the incumbent solution. This procedure is traditionally associated with local search procedures. Regarding exploration, a large region of the search space is explored with the aim of increase the diversity among the solutions explored. Metaheuristics have been successfully applied in a large number of optimization problems, reaching high-quality solutions in a reasonable computational time.

In this PhD Thesis, some of the most widely used metaheuristics have been studied to solve optimization problems. Some of them are as follows. Greedy Randomized Adaptive Search Procedure (GRASP) [23, 26] and Path Relinking (PR) [27]. Concretely, these metaheuristics are used to provide high-quality solutions to optimization problem that belongs to the Stanford Network Analysis (SNA) family.

1.2 Optimization problems in Social Network Influence

Nowadays, millions of users are involved in Social Network (SN), growing exponentially the number of active users. This growth is extended to the amount of behavioral data, and, therefore, all classical network-related problems are becoming computationally harder. SNs have become one of the most widely used sources of information, mainly due to their ability to provide the user with real-time content. They are not only a new form of communication, but also a powerful tool that can be used to gather information related to relevant questions, for example: which is the favorite political party for the upcoming elections, what are the most talked about movies of the past year, which is the best-rated restaurant in a given area, etc. Extracting relevant information from social networks is a topic of interest, mainly due to the enormous amount of potential data available. However, traditional network analysis techniques are becoming obsolete due to the exponential growth of social networks, in terms of the number of active users and the relations among them.

The different areas within the Social Network Analysis family are illustrated in Figure 1.6 [28].

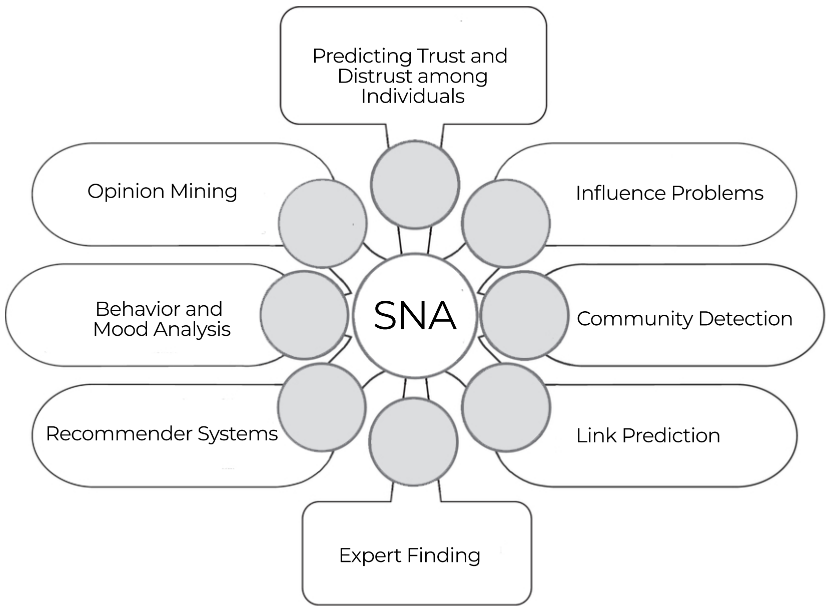


Figure 1.6: Social Network Analysis: Family of Problems.

A concise overview of the primary topics within the field of Social Network Analysis is provided in a clockwise direction: Predicting Trust and Distrust among Individuals, involves developing models to forecast trust and distrust levels in social network relationships. This aims to enhance our ability to anticipate the dynamics of interpersonal connections; Influence Problems, refers to the spread of information, behaviors, or opinions across a social network, influencing the attitudes or actions of its nodes; Community Detection, identifies groups or clusters of nodes within a social network that exhibit higher connectivity among themselves than with the rest, unveiling the underlying structure and relationships; Link Prediction, forecasts future connections between entities, utilizing network structure and historical interactions; Expert Finding, identifies individuals with specific knowledge, skills, or expertise in a subject within a social network; Recommender Systems, analyze user preferences to provide personalized suggestions; Behavior and Mood Analysis, studies patterns related to user behavior and emotional states in social networks; Opinion Mining, extracts subjective information from social media or reviews to determine sentiments and opinions expressed by individuals or groups.

Influence propagation plays an important role in the spread of information, opinion, idea, innovation, rumor, or misinformation on a large scale [29]. This spreading process has a huge practical importance in viral marketing [30, 31]. Consider the case of promoting a brand by of a commercial house through online marketing, where the goal is to attract the users for purchasing a particular product. The best way to do this is to select a set of highly influential users and distribute them free samples. Many of them will like the item and influence their neighbors to try the product. These newly informed users will influence their neighbors. This cascading process will continue and, ultimately, a large fraction of users will try for the product, leading to a significant improvement in the earned revenue. Naturally, the number of free sample products will be limited due to economic reasons. Hence, this process will be fruitful if the free samples can be distributed among the highly influential users and the problem here bottoms down to select influential users from the network.

SNs are used not only to spread positive information but also malicious information. In general, research devoted to maximize the influence of positive ideas is called Influence Maximization [32]. Thus, solving successfully this problem allows the decision-maker to decide the best way to propagate information about products and/or services. On the contrary, SNs can also be used to spread malicious information such as derogatory rumors, disinformation, hate speech, or fake news. These examples motivate research about how to reduce the influence of negative information. This family of problems is usually known as influence minimization [33, 34, 35].

There are different approaches to solve these problems: approximation algorithms guarantee the worst-case bound for influence spread, but the main drawback is scalability issues, which means, with the increase in the network size; heuristic solutions, while lacking a worst-case bound on influence spread, often exhibit superior scalability and improved running time compared to the approximation algorithms; metaheuristic methods based on evolutionary computation techniques, also lack a worst-case bound on influence spread; community-based solutions leverage community detection in the underlying social network as an intermediate step to break down the problem into the community level, enhancing scalability, but most algorithms in this category are heuristic and do not provide a worst-case bound on influence spread.

A complete survey on Social Network Influence can be found in [36] where it is stated that metaheuristics scarce in SNI. Finally, another recent survey states different metaheuristic algorithms used in SNI [37].

1.3 Hypothesis and objectives

After identifying the problems to be solved, the following hypothesis supports the development of the research project.

The hypothesis proposed for the development of this Doctoral Thesis is aboard since a point of view that SNI problems are $\mathcal{NP}$-hard problems, with practical interest in many scientific disciplines. Therefore, it is interesting to develop algorithms that can generate high-quality solutions in a reasonable time. The scalability property required since the real-word increase the number of data generated via SN.

Metaheuristic techniques are also proposed, which have been shown to be effective procedures when dealing with optimization problems. In particular, trajectory metaheuristics constitute a subfamily of this type of techniques, characterized by considering more than one solution simultaneously and providing combination mechanisms among them. Dispersed Search is a clear exponent of this type of metaheuristics. On the other hand, the so-called trajectory metaheuristics start from an initial solution and are capable of generating a trajectory in the solution space. Greedy Randomized Adaptive Search Procedure (GRASP) is an example of a trajectory metaheuristic.

The proposed heuristic algorithm will be complemented with the metaheuristic techniques best suited to the problem, in particular GRASP and PR will be considered.

To achieve the main objectives mentioned above, the following objectives must be addressed:

- Study the state of the art of the problem, analyzing current algorithm proposals.
- Design and develop heuristic algorithms to solve problems related to the Social Network Influence problems using different Influence Diffusion Models.
- Configure the parameters of the algorithm developed. The algorithms developed must be configured to use the best parameters to achieve the best results. To configure the parameters, you must conduct a preliminary experiment.
- Validate the heuristic algorithm. The algorithms will be experimentally compared with the state-of-the-art algorithms to provide a fair comparison.
- Analyze the results obtained by the new algorithm and compare them versus the state-of-the-art algorithms.
- Adapt the proposed algorithms to the real-known problems related to: pandemic evolution containment, rumor, and fake news minimization, where the results will be validated.
- Publish all the source code, instances, and results in public repositories to ease further comparisons.
- The partial results will be submitted to JCR journals with review processes by independent institutions. Thus, provide an expert evaluation to improve our research and verify that it contributes to the academic area related to SNI.

Throughout the dissertation, conclusions of the dissertation are presented in order to confirm the previous hypothesis.

1.4 Memory structure

This document summarizes the research that has been conducted and is organized as follows:

- Part I, the developed research is shown, describing the addressed problem, as well as the followed methodology and the obtained results. Finally, the conclusions derived from the work are exposed.
  - Chapter 1, optimization research area, combinatorial Optimization Problems, heuristics and metaheuristics are described. The hypotheses and objectives for this work are also defined.
  - Chapter 2, the Social Network Influence is presented, defining it in a detailed way, explaining each of its variants.
  - Chapter 3, the methodology applied to the previously described problem is explained. The main metaheuristic and all the other techniques that lead to the obtained results are exposed.
  - Chapter 4, the obtained results for each variant of the problem and the main contributions made are analyzed and discussed.
  - Chapter 5, the conclusions derived from the research and possible future work are presented.
- Part II the published articles associated with this Thesis are summarized and gathered, together with information about the journal in which they are published.
- Finally, Part III includes a brief summary of the dissertation in the native language, Spanish.

Chapter 2

Social Network Influence Problem

This chapter introduces the classic Social Network Influence Problem. Then, most extended Influence Diffusion Models (IDM) are presented, which are responsible of determining when a user is influenced or not. Finally, for each problem, the objective function, the goal of the problem, and an illustrative example of the evaluation of a solution are described. In particular, the problems studied are: Social Network Influence Maximization (Section 2.3), Budgeted Influence Maximization (Section 2.4), and finally, the Target Set Selection Problem (Section 2.5).

2.1 Social Network Influence Problems

The evolution and expansion of social networks have begun to attract the interest of the scientific community in recent years. Due to this, numerous problems that are difficult to solve with classical techniques have arisen, such as viral marketing [38], disease propagation [39], misinformation [40], among others. The rationale behind this is the exponential growth of users in the last decades. Figure 2.1 shows the number of Internet users per year according to the number of users in millions.

The Social Network Analysis is usually divided into several research fields, each of them focused in a different area. This thesis is focused on one of these sub-areas: the Social Network Influence Maximization. This family of problems aims to find the most influential users within a social network.

A social network is conformed with a set of users and the social interactions among them. Then, the aim is to strategically choose users to optimize specific criteria, such as maximizing or minimizing influence, meeting a certain budget, etc.

Social Networks are usually modeled as a graph $G = (V, E)$ where the set of nodes V represents the users and each relation between two users is modeled as a pair $(u, v) \in E$, with $u, v \in V$ indicating that user u is connected to or even can transmit information to user v. Depending on the nature of the network, the graph can be directed (if it is a one-way relation), weighted (if the users or relations have different relevance), etc.

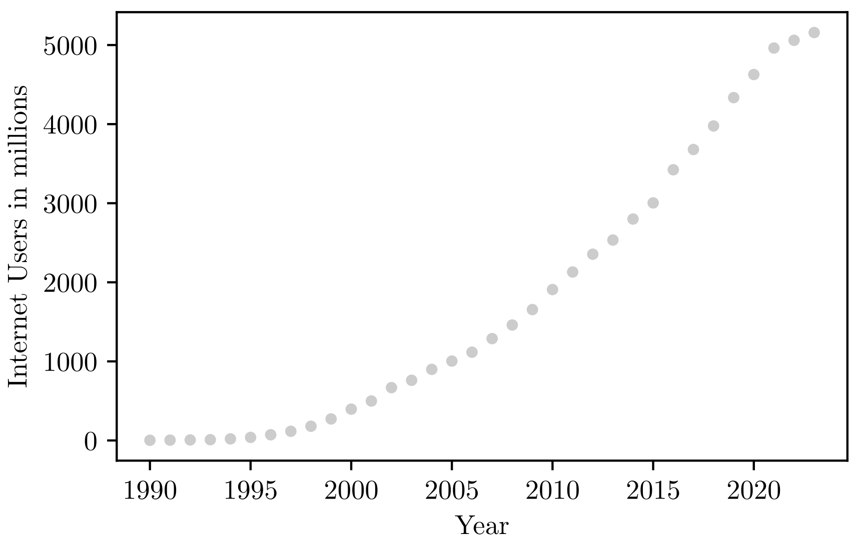


Figure 2.1: Internet Users Timeline based on We Are Social company reports.

Information propagation is the phenomenon with which information moves from one user to another through network relationships. In the literature, there are many works that are interested in the study and/or simulation of this phenomenon. In fact, many works [41, 42] studied the information propagation process to understand and explain this phenomenon [43]. A diffusion model, also known as propagation model, is a model that simulates and describes the entire propagation process and determines which node in the network will receive the propagated message. A detailed discussion on the most extended IDMs is presented in Section 2.2.

Most social networks have a high number of active users, becoming their analysis a real challenge for academics and practitioners. The main drawback in SNI is related to the evaluation of a solution, since solutions are usually evaluated with an IDM, and IDMs require high computational effort. As stated before, this family of problems is known to be $\mathcal{NP}$-hard, and therefore an exact algorithm is not suitable to solve it optimally in a reasonable computing time. The application of efficient metaheuristics in the field of social networks will allow the community to have tools able to analyze the influence of users in real time, which represents a very significant advance in this field. The greedy selection function considers the redundancy between likely-influenced nodes and does not include those reached by the already selected seed nodes to provide a better estimation of the total spread. Some heuristics have been proposed as time-saving solutions for greedy decisions: random, degree, and centrality [17].

Every day, online social networks collect a huge amount of data, and the influence maximization process is time-consuming in some cases. Some real-world applications, such as disease propagation analysis actions, play an important role where every minute is crucial, as the spread can be exponential. The longer it takes to have a solution, the higher the risk of affecting a larger number of people. Similarly, in the business world, the planning of marketing campaigns requires from real-time information extracted from the social network to analyze the effect of the political campaign. Moreover, in the dynamic world of social networks, where trends change continuously, the ability to quickly find a solution is essential.

The increase in the interest of the scientific community in Social Network Influence have lead researchers to present a survey on this topic [36]. In that work, the authors experimentally compared the results obtained by the most recent algorithms such as approximation algorithms [44], heuristic solutions [17], community-based solutions [45] and metaheuristic solutions [46], trying to solve Social Network Influence Maximization Problem (SNIMP). They concluded by stating that metaheuristics are scarce but interesting techniques in Social Network Influence. Finally, a recent survey related to Social Network Influence with metaheuristics is presented in [37].

2.2 Influence Diffusion Models

The evaluation of the influence of a given seed set S over a network G requires the definition of an Influence Diffusion Model (IDM) [47]. An IDM is responsible for modeling how the information is transmitted through the network, thus indicating which nodes are influenced. Each node has two states depending whether the information has already influenced it (active) or not (inactive). The most extended IDMs in the literature are: Independent Cascade Model (ICM) (see [17]), Weighted Cascade Model (WCM) (see [17]), Linear Threshold Model (LTM) (see [17]) and Tri-Valency Model (TV) (see [48]). All of them are based on assigning an influence probability to each relational link in the SN, since a relation in a network does not necessarily mean that a user influences another one in a certain period of time.

Since in every diffusion model the edges have a certain probability of influence, it is not possible to perform a deterministic evaluation of the active nodes after the propagation process has finished. Instead, a probabilistic method is usually considered to evaluate the nodes influenced by a certain set of seeds. Algorithm 1 shows the Monte Carlo algorithm, which is a well-known method used in the literature to estimate the number of activated users. The algorithm requires four input parameters: the social network $G = (V, E)$, a seed of infected/activated users S, the number of iterations ev, and, the IDM criteria represented by ψ.

The algorithm starts by initializing the set which stores the number of infected users (step 1). It then performs a number of predefined iterations ev (steps $2 - 18$), finding in each iteration the influenced nodes by the given seed set S. Initially, the set of nodes $A^\star$ reached by the initial seed set, S, is actually the seed set (step 3). Then, the method iterates until no new nodes are influenced (steps 5-16). In each iteration of the inner for-loop, the neighbors of each node reached in the previous one are traversed (steps 8-12). For each neighbor, the specific IDM criteria ψ is performed and, if it is valid, then the neighboring node becomes infected (steps 9-11). At the end, the set of infected nodes is updated (step 14) as well as the nodes infected in the previous iteration (step 15). Finally, the algorithm returns the mean number of infected nodes among all the simulations performed (step 19). Notice that this value is considered as the objective function to be optimized when solving a SNI problem. It is worth mentioning that, as infection is an stochastic process, the ICM must be run several times (ev in our case) to achieve an appropriate estimation.

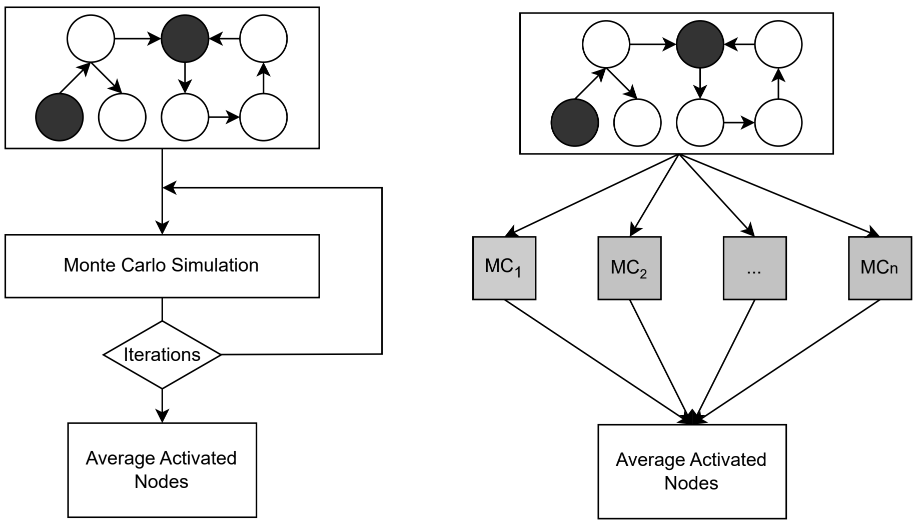
Algorithm 1 *MonteCarlo*$(G = (V, E), S, \psi, ev)$

```
1: $I \leftarrow \emptyset$
2: for $i \in 1 \ldots ev$ do
3:     $A^\star \leftarrow S$
4:     $A \leftarrow S$
5:     while $A \neq \emptyset$ do
6:         $B \leftarrow \emptyset$
7:         for $v \in A$ do
8:             for $(u, v) \in E$ do
9:                 if $\psi$ then
10:                    $B \leftarrow B \cup \{u\}$
11:                end if
12:            end for
13:        end for
14:        $A^\star \leftarrow A^\star \cup B$
15:        $A \leftarrow B$
16:    end while
17:    $I \leftarrow I + |A^\star|$
18: end for
19: return $I/ev$
```

Numerous studies have extensively explored the most adequate number of iterations in Monte Carlo simulations [49], since it should be large enough to be precise, but small enough to avoid requiring large computing times. In the context of problems related to social network influence, researchers performs ranging from 100 to 100000 iterations, reflecting the diversity in simulation [46, 50]. Some studies have introduced alternative convergence criteria for Monte Carlo algorithms, such as Gaussian distribution fitting, Chebyshev Inequality Stopping Rule, Estimation of the Variance, among others [49]. The motivation behind these studies lies in the recognition that specifying a fixed number of iterations may lead to inaccuracies if chosen too small or result in excessive computational burden if set too high. To address this, certain methods aim to approximate the Monte Carlo algorithm [51], improving its efficiency and mitigating the challenges associated with arbitrary iteration counts.

Lastly, some approaches proposed the parallelization of Monte Carlo simulation [52], since each iteration performs independently. Figure 2.2 illustrates two approaches to obtain activated nodes, both figures receive as input a social network with selected seed set nodes (represented as black nodes). Figure 2.2a outlines the sequential methodology, where all iterations are executed consecutively and the activated nodes are collectively returned at the end. Then Figure 2.2b illustrates the independent execution of several iterations, with the activated nodes returned separately for each iteration. Subsequently, an average is computed to derive the mean count of activated nodes.

These methods ensure a balance between computational time and quality in the realm of social network analysis. The following sections will detail each of the IDMs considered.



(a) Sequential (b) Parallel

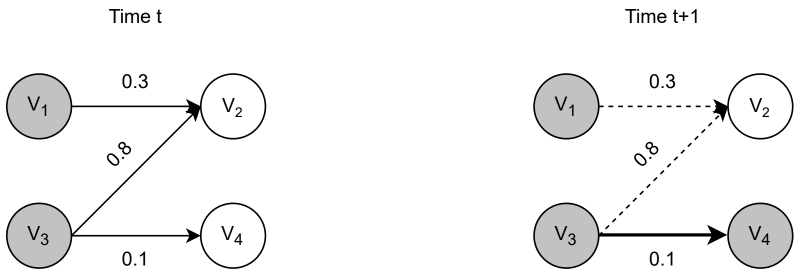
Figure 2.2: Monte Carlo Simulation.

2.2.1 Independent Cascade Model

This model serves as a framework for simulating information cascades within social networks [17, 53, 54], so that a node can only influence those to which it is directly connected (see Algorithm 1). A vertex v is said to be active when it receives the information and accepts it. It is said to be inactive when it does not receive or rejects the information. An inactive node becomes active if it receives and accepts the message. A node, after been activated, can transmit information to its neighbor nodes with activation probability $p_{v,u} \in [0, 1]$. It is worth mentioning that transitions occur exclusively from an inactive to an active state. Specifically, within the ICM, a node denoted as v that becomes active at time t, has the chance to activate its neighbor next time $t+1$. The activated node then generates a random number between $[0, 1]$ for each neighbor u. If and only if $p_{v,u} \leq p$ the neighbor u will be activated. The diffusion process continues until no other new nodes can be activated.

Figure 2.3 illustrates a social network composed of four nodes $\{v_1, v_2, v_3, v_4\}$ and three edges $\{(v_1, v_2), (v_3, v_2), (v_3, v_4)\}$, each number on the top of an edge corresponds to an activation probability (generated by a random number between $[0, 1]$). Nodes shaded in gray denote their activated status.

On the one hand, Figure 2.3a shows the initial step, where nodes v_1 and v_3 are activated. On the other hand, Figure 2.3b shows dashed lines denoting that the relation has not activated a new node and a bold black line indicating that the node is influenced. The ICM sets a fixed value of p for all edges, assuming a p value of 1% (in this case, 0.01) commonly employed in the literature. There is only one node meeting the criterion $p_{v,u} \leq p$, which is node v_4. Since node v_4 has not neighbors the process ends, resulting in a total of 3 activated nodes.



(a) SN with 4 nodes and 3 relations. (b) Resulting SN in t+1.

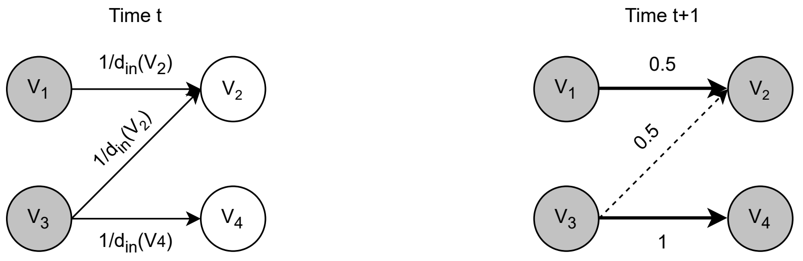
Figure 2.3: Independent Cascade Influence Diffusion Model.

2.2.2 Weighted Cascade Model

The Weighted Cascade Model [17] considers that the probability of a user v for being influenced by user u is proportional to the in-degree of user v, i.e., the number of users that can eventually influence user v. That is to say, a node with large degree is relatively hard to be activated by a single connection. Therefore, the probability of influencing user v is defined as $1/d_{in}(v)$, where $d_{in}(v)$ is the in-degree of user v. A node will be activated if $(1 - 1/d_{in}(v)) \leq r$, given a randomly generated number r.

In the standard ICM, the probability of propagation between any two nodes is the same. However, this assumption can be altered so that the probability of propagation is related to the number of neighbors a node has. This implies that a popular individual with many connections is more difficult to activate with a single neighbor, which is a more realistic approach. If an individual receives input information from several sources, then the probability of being influenced by just a single source should be smaller than if it is influenced by several sources. The spread process is similar to ICM, except that the propagation probability is dependent on the degree.

Figure 2.4 represents the same social network as previously stated. Notice that, the edges have a form derived by the WCM model, which would be the activation probability.



(a) SN with 4 nodes and 4 relations. (b) Resulting SN in t+1.

Figure 2.4: Weighted Cascade Influence Diffusion Model.

Figure 2.4a represents the initial social network at time t where the same nodes as before are activated. Then, the activation probability of edge (v_1, v_2) is denoted as

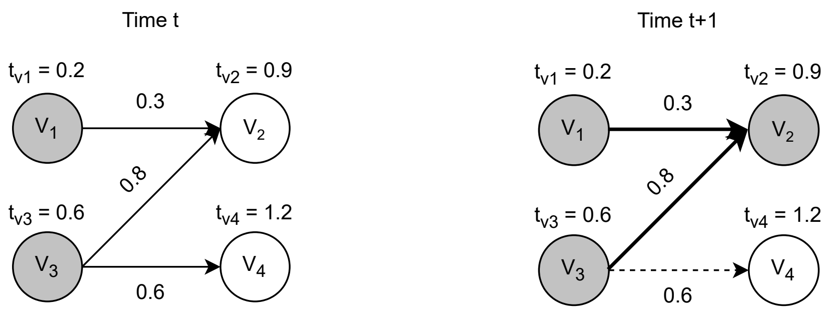
$1/d_{in}(v_2)$ being $d_{in}(v_2) = 2$. Therefore, all the relations to node v_2 would result in the activation probability of 1/2, so edge (v_3, v_2) also results in the activation probability 1/2. However, edge (v_3, v_4) is different, since node v_4 only has an incident edge, so the activation probability would be 1. Finally, assuming that r are: 0.6 for (v_1, v_2), 0.4 for (v_3, v_2) and 0 for (v_3, v_4), Figure 2.2b represents that which nodes have been successfully activated, where v_4 is activated $(1 - 1/1 \leq 0)$ and node v_2 is activated $(1 - 1/2 \leq 0.6)$. Note that node v_2 is activated just by one edge (v_1, v_2), as a random number r is generated, it could not activate one edge with same probability. The total number of activated nodes in the depicted example will be 4.

2.2.3 Linear Threshold Model

The Linear Threshold Model [17, 48] together with the Independent Cascade Model, were the first simulation models used to simulate the information propagation process. In the LTM, a weight $\omega(u, v)$ is associated to each edge (u, v) and a threshold θ_u to each node u. A node u will be activated if the total weight between itself and its activated neighbors is, at least, θ_u. More formally:

$$\sum_{v} \omega(u, v) \geq \theta_u \tag{2.1}$$

The depicted Figure 2.5 represents a similar social network as before. The represented social network 2.5a has a number in each vertex representing the threshold θ and in each edge denoting the weight $\omega(u, v)$.



(a) SN with 4 nodes and 4 relations. (b) Resulting Social Network in t+1.

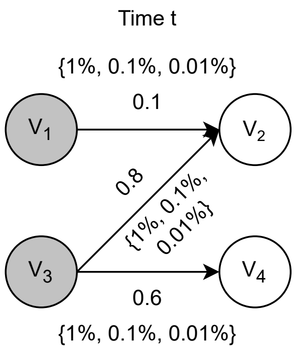
Figure 2.5: Linear Threshold Model.

As can be observed in Figure 2.5b node v_2 is activated. The reason behind this is that node v_2 has activated neighbors whose relations sum more than the threshold at node v_2, $\theta_{v_2} = 0.9$, specifically $\omega(v_1, v_2) + \omega(v_3, v_2) = 0.3 + 0.8 = 1.1 \geq \theta_{v_2}$. The remaining node v_4 is not activated since the sum of its relations does not reach the threshold value, i.e., $0.6 < 1.2$, thus the total number of activated nodes is 3.

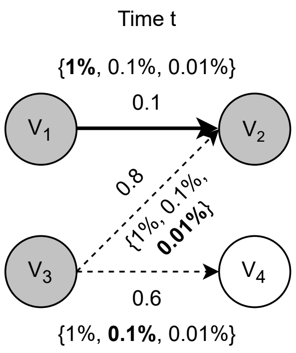
2.2.4 Tri-Valency Model

The last model is called Tri-Valency Model and it was developed by [48]. This propagation model randomly selects the edge probability from a set of probabilities $p = \{1\%, 0.1\%, 0.001\%\}$. The activated node v then generates a random number between $[0, 1]$ for each neighbor u. If and only if $p_{v,u} \leq p$ the neighbor u will be activated. The main objective in TV is to introduce variability in the strength of connections within the model. These values represent different probabilities associated with the activation of nodes through their edges. By incorporating this range of probabilities, the model can simulate scenarios in which edges have varying degrees of impact on the diffusion of influence through the network. This adds a layer of realism to the model, capturing the diversity in the strength of connections observed in real-world social networks and other complex systems.

Figure 2.6 includes two sub-figures. The former Figure 2.6a is related to the values assigned to the different edges. The latter Figure 2.6b shows the selection of random probabilities on each edge between the candidate values and the activated nodes according to their relations.



(a) SN with 4 nodes and 4 relations.



(b) Resulting Social Network in t+1.

Figure 2.6: Tri-Valency Model.

The random probability associated with each edge is shown in bold in Figure 2.6b. Then, just edge (v_1, v_2) meets the criteria $p_{v,u} \leq p$ and activates node v_2, so the total number of active nodes is 3.

2.3 Social Network Influence Maximization Problem

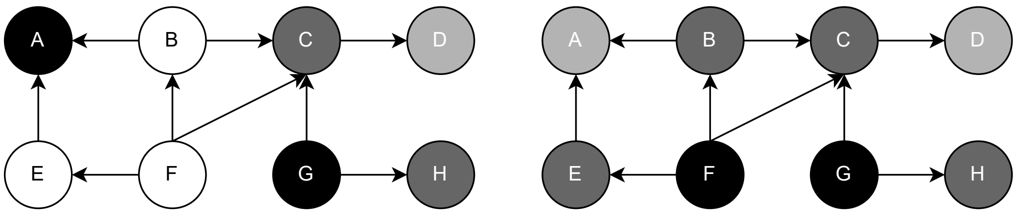
The Social Network Influence Maximization Problem (SNIMP) is the first problem considered in the framework of this Doctoral Thesis. It was initially formulated in [55] and was later proven to be $\mathcal{NP}$-hard for most ICMs in [47]. Given a SN with $|V| = n$ nodes where the edges (relational links) represent the spreading or propagation process on that network, the task is to choose a seed node set S of size $k < n$ (where k is an input parameter) with the aim of maximizing the number of nodes in the network that are influenced by the seed set S.

In mathematical terms,

$$S^{\star} \leftarrow \arg\max_{S \in \mathcal{S}} ICM(G, S, p, ev) \tag{2.2}$$

where $\mathcal{S}$ is the set of all possible solutions (i.e., seed set setups with size k), p is the probability of a user to be influenced, and ev is the number of iterations of the Monte Carlo simulation used to run the ICM.

Figure 2.7 shows an example of an SN with 8 nodes and 9 directed edges, where each pair (u, v) denotes that the user v may be influenced by u. Information represents anything that can be passed across connected peers within a network. The influence level given by a node is determined by the adoption or propagation process. Let us consider $k = 2$ for this example graph.



(a) $S_1 = \{\texttt{A, G}\}$ (b) $S_2 = \{\texttt{F, G}\}$

Figure 2.7: SNIMP example with a SN with 8 nodes and 9 edges. Two feasible solutions S_1 and S_2 are represented, each of them resulting in a different set of influenced users.

Figure 2.7a shows the solution S_1 where the seed set is conformed with nodes `A` and `G`. Without loss of generality, for this example it is assumed that $p = 1$ (i.e., all the nodes are always infected by their neighbors). By simulating diffusion model we can see how nodes `C` and `H` are directly influenced by node G. It is worth mentioning that a leaf in a graph (node without neighbors) will not be able to activate any other node but itself, as can be seen in the figure depicted with node A. After that, node `C` influences node `D`. Different levels of influence are represented by a gray gradient from black to white. Therefore, if we consider a single evaluation of the Monte Carlo simulation, the objective function value of S_1 is $ICM(G, S_1, p, 1) = 5$. Figure 2.7b depicts the solution $S_2 = \{\texttt{F,G}\}$. Similarly to Figure 2.7a, a gray gradient indicates the process of influence over the network, resulting in an objective function value of $ICM(G, S_2, p, 1) = 8$, since all nodes are influenced, being the optimal solution. Notice that, following this evaluation, S_2 is better than S_1.

A promising starting point for such problems involves targeting nodes with higher degrees (in-degree or out-degree). The reason behind that is that they have a large number of neighbors to potentially activate [47]. It is worth mentioning that considering different communities is also crucial. Selection of all seed nodes from a single community may be effective if it is densely populated. However, exploring nodes in other communities can be valuable in alternative scenarios [45].

In a real-world application, one notable field is marketing [56]. The goal of marketing campaigns is to identify users, often referred to as influencers, who can deliver the highest performance for the campaign. Given that the number of influencers is limited and that they may share common followers, determining the most strategic influencers becomes pivotal in reaching a larger audience. Activating more users could increase sales, visibility, ad impressions, or brand generation.

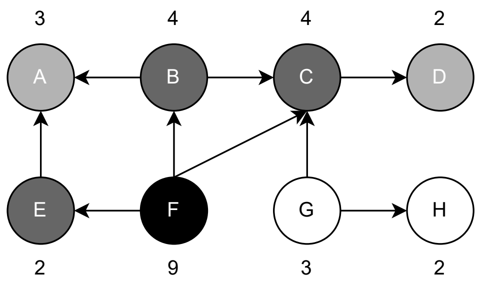
2.4 Budgeted Influence Maximization Problem

The second problem addressed within this Doctoral Thesis is the Budgeted Influence Maximization Problem (BIMP), originally defined in [57], which, instead of selecting a fixed number of initial users, allows us to invest a certain budget in users of the SN, considering that the cost of selecting users is not uniform. This variant is closer to real SN than SNIMP. In BIMP, the traditional model of SN is still considered. However, a function $C : V \rightarrow \mathbb{Z}^+$ is introduced, which assign a non-uniform positive integer cost to every user of the network. Additionally, an initial budget B is given, which is the maximum investment that can be used to select nodes. Each selected node u will decrease the available budget in $C(u)$ units. Then, the BIMP consists of selecting a set of seed nodes $S^\star$ that maximizes the information diffusion throughout the network without exceeding the given budget B. More formally,

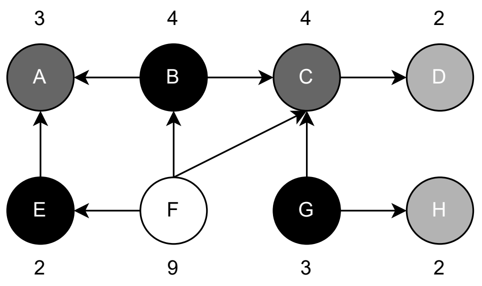
$$S^\star \leftarrow \underset{S \in \mathbb{SS}}{\arg\max}\, ICM(G, S, p, ev) : \sum_{u \in S} C(u) \leq B \tag{2.3}$$

where $\mathbb{SS}$ represents all possible combinations of seed sets that can be generated.

Figure 2.8 shows an SN similar to Figure 2.7. The SN consists of 8 nodes (where each node has a number denoting the cost of activating it) and 9 directed edges. Let us consider a budget of $B = 10$ for this example graph.



(a) $S_1 = \{$`F`$\}$



(b) $S_2 = \{$`B, E, G`$\}$

Figure 2.8: BIMP example considering a budget of 10 and an SN with 8 nodes and 9 edges. Two feasible solutions S_1 and S_2 are represented, each of them resulting in a different set of influenced users.

Then, Figure 2.8a shows the solution S_1 where the seed set is conformed by node `F`. Since this node requires a budget of 9, no more nodes can be activated. Without loss of generality, for this example it is assumed that $p = 1$ (i.e., all the nodes are always infected by their neighbors). By simulating diffusion model, we can see how nodes `B`, `C` and `E` are directly influenced by node F. After that, node `C` influences node `D`. Finally, both nodes `B` and `E` activate node `A`. Different levels of influence are represented by a gray gradient from black to white. Therefore, if we consider a single evaluation of the Monte Carlo simulation, the objective function value of S_1 is $ICM(G, S_1, p, 1) = 6$. Figure 2.8b shows the solution $S_2 = \{\texttt{B,E,G}\}$, with a total cost of $4 + 2 + 3 = 9$. It is worth mentioning that the BIMP problem has not fixed seed set size. Similarly to Figure 2.8a, a gray gradient indicates the influence process over the network, resulting in an objective function value of $ICM(G, S_2, p, 1) = 7$. Notice that, following this evaluation, S_2 is better than S_1. As can be derived from the problem, it is a good idea to activate it when there are remaining nodes that the budget can afford.

The large amount of data and interest in SN have aroused the interest of both the scientific community and companies in considering BIMP to optimize the spreading process of a certain message, product, or idea to clients. Marketing agencies like BrandWatch (see [58]) have limited budget by their clients and use this approach when their customers need a commercial campaign based on a certain budget to determine the most effective users to initialize the campaign.

2.5 Target Set Selection Problem

The third problem studied is the maximum effort-reward GAP Target Set Selection (TSS), proposed by [59] and demonstrated to be $\mathcal{NP}$-hard. TSS is somewhat similar to SNIMP and BIMP since it also requires to select a seed set with the maximum influence. The primary distinctions lie in the fact that, unlike the SNIMP, the TSS does not entail a fixed number of nodes but each of these nodes has an effort cost. This might initially suggest similarities with the BIMP, but a key disparity arises in the manner in which node activation is propagated. TSS does not use a probabilistic Monte Carlo-based method for influence propagation; instead, it adopts a deterministic approach as influence propagation.

Given a SN and a maximum budget K, the TSS problem tries to find a subset of users $S \subseteq V$ whose effort does not exceed the maximum value K, with the aim of maximizing the number of influenced users in the social network. Since the concept of influencing can be ambiguous, it is necessary to perform some initial definitions to clarify it.

Starting from a set of initially activated nodes S, this process consists of activating all those non-activated nodes which are influenced by the activated ones. In the context of TSS, the influence of a user is modeled by a rational influence function, denoted as $\psi : V \times V \to [0, 1]$ applied to all pairs of users, assessing how one user influences another. Note that if two users u, v are not connected, then $\psi(v, u) = 0$. Without loss of generality, a solution for the TSS is given by the set of initially activated nodes S. Notice that this process is iteratively applied until no new nodes are activated. Given a step t and a set of activated nodes S^t, the set of nodes which are activated after applying a single iteration of the influence propagation process is denoted as S^{t+1}.

The influence propagation process then stops when no new nodes are activated, i.e., $S^t = S^{t-1}$.

In TSS, each node has an associated effort to activate it, which is defined by the function $\alpha : V \to \mathbb{Z}^+$, as well as a reward obtained when it is influenced or initially activated which is defined by $\beta : V \to \mathbb{Z}^+$. Thus, the effort α is only considered for those nodes which are part of the set of the initially activated users S, and the reward β is earned whether a user u is initially activated or subsequently activated by the set of activated users. Another fact in TSS is that, when a node is activated at step t, it remains activated through the entire influence propagation process. Let us denote x_v^t as a binary variable which takes the value 1 if the node v is activated at step t and 0 otherwise (we refer the reader to [60] to a more detailed description of the model). Then,

$$x_v^{t-1} \leq x_v^t, \quad \forall v \in V, 1 \leq t \leq T \tag{2.4}$$

where T indicates the maximum number of steps in the influence propagation process, i.e., number of iterations in which new nodes are activated.

A solution S for the TSS is feasible if and only if the sum of the efforts of the initially activated nodes is smaller than or equal to K, which is a constraint of the problem, i.e., $\sum_{v\in V} \alpha(v) \cdot x_v^0 \leq K$. The objective function for the TSS is then evaluated as the sum of rewards obtained by the active nodes in the last iteration of the influence propagation process. In mathematical terms,

$$TSS(S) = \sum_{v \in V} \beta(v) \cdot x_v^T \tag{2.5}$$

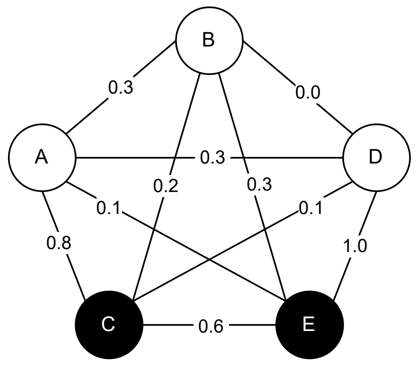
Notice that evaluating the TSS is a computationally demanding procedure. In particular, the computational complexity of this evaluation is $O(n^2)$ since it is necessary to traverse all the nodes and, for each new activated node, it is required to traverse again the set of nodes searching for new potential nodes to be activated. The TSS then seeks for a solution $S^\star$ with the maximum objective function value. More formally,

$$S^\star \leftarrow \arg\max_{S \in \mathbb{SS}} TSS(S) \tag{2.6}$$

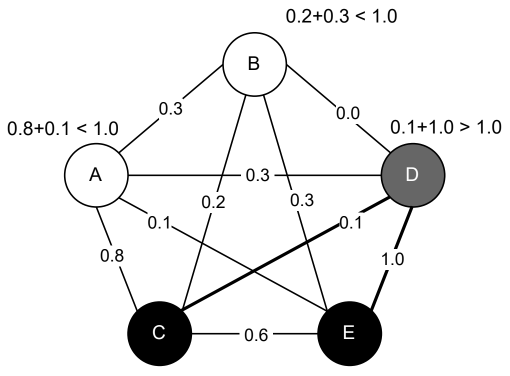
where $\mathbb{SS}$ represents the set of all feasible solutions, i.e., all possible combination of nodes whose sum of effort is smaller than or equal to K.

Figure 2.9 depicts an example of the complete influence propagation process over a network with 5 nodes and 10 edges. The solution under evaluation is conformed by nodes C and E, i.e., $S = \{\mathsf{C}, \mathsf{E}\}$. Without loss of generality, let us suppose that the sum of costs $\alpha(\mathsf{C}) + \alpha(\mathsf{E})$ is smaller than or equal to K.

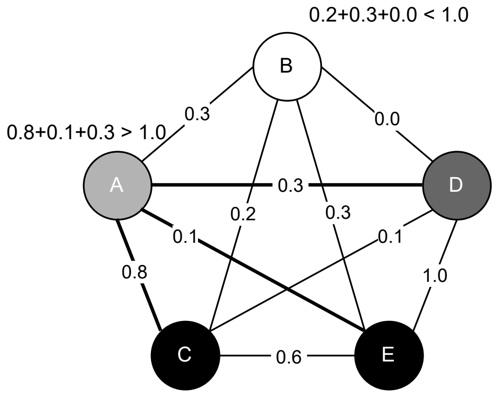
Figure 2.9a shows the initial step $t = 0$, where the activated nodes are the ones initially selected C and E, i.e., $S = \{\mathsf{C}, \mathsf{E}\}$. Then, in the first iteration of the influence propagation process, depicted in Figure 2.9b, it is evaluated whether non-influenced nodes A, B, and D are influenced or not. Starting with node A, it is necessary to evaluate $\psi(\mathsf{C}, \mathsf{A}) + \psi(\mathsf{E}, \mathsf{A}) = 0.8 + 0.1 = 0.9 < 1.0$. Therefore, node A is not influenced in this step. A similar evaluation is performed with node B, resulting in $\psi(\mathsf{C}, \mathsf{B}) + \psi(\mathsf{E}, \mathsf{B}) = 0.2 + 0.3 = 0.5 < 1.0$, indicating that node B is not activated. Finally, when performing



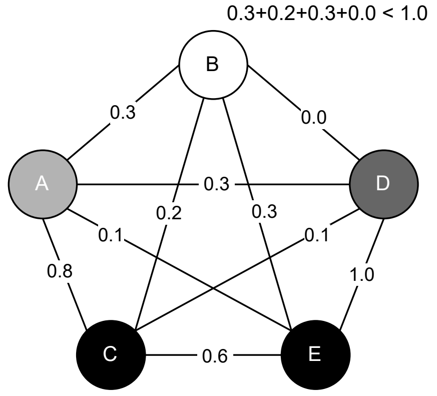
(a) Step I: Initial solution with $S = \{\mathtt{C}, \mathtt{E}\}$



(b) Step II: $S^1 = \{\mathtt{C}, \mathtt{E}, \mathtt{D}\}$



(c) Step III: $S^2 = \{\mathtt{C}, \mathtt{E}, \mathtt{D}, \mathtt{A}\}$



(d) Step IV: $S^3 = \{\mathtt{C}, \mathtt{E}, \mathtt{D}, \mathtt{A}\}$

Figure 2.9: Target Set Selection Influence propagation process over a network with 5 nodes and 10 edges, considering the solution $S = \{\mathtt{C}, \mathtt{E}\}$.

the evaluation of node D, we obtain $\psi(\mathtt{C}, \mathtt{B}) + \psi(\mathtt{E}, \mathtt{B}) = 0.1 + 1.0 = 1.1 \geq 1.0$. Then, after the first iteration, node D is included in the set of activated nodes, resulting in $S^1 = \{\mathtt{C}, \mathtt{D}, \mathtt{E}\}$.

Since $S^0 \neq S^1$, it is necessary to continue with the influence propagation process. If we now evaluate the second iteration, Figure 2.9c, starting with node A, $\sum_{v \in S^1} \psi(v, \mathtt{A}) = 0.8 + 0.1 + 0.3 = 1.2 \geq 1.0$. Then, node A will be included in the set of activated nodes in the next iteration, S^2. In the case of node B, the evaluation is $\sum_{v \in S^1} \psi(v, \mathtt{B}) = 0.2 + 0.3 + 0.0 = 0.5 < 1.0$, so the node B is not activated. After this iteration, $S^2 = \{\mathtt{A}, \mathtt{C}, \mathtt{D}, \mathtt{E}\} \neq S^1$, so an additional iteration is required. Figure 2.9d illustrates the last iteration of the influence propagation process. In this case, it is only required to evaluate node B, resulting in $\sum_{v \in S^2} \psi(v, \mathtt{B}) = 0.3{+}0.2{+}0.3{+}0.0 = 0.8 < 1.0$. Therefore, node B is not activated. Since no new nodes are included in the set of activated nodes, i.e., $S^2 = S^3$, the influence propagation process stops in this iteration, returning the reward associated to the activated nodes.

Although the main application of this problem is to increase the impact of advertising a product for a company [61, 62], there are several applications in different fields. For instance, in politics, the Max-TSS can be used for reducing the impact of fake news or misinformation from two opposite approaches: identifying the individuals which are mostly spreading fake news through the network, or to boost those individuals which are transmitting reliable information [63]. Even more, this application is closely related to disease control, since it has been proven that the diseases spreads following the same model as information through Social Networks [64].

2.6 Real World Social Networks

The set of SNs considered in this paper have been entirely obtained from the most relevant works found in the literature, in order to provide a fair comparison among the analyzed algorithms.

Instance	$\|N\|$	$\|E\|$	**Ref.**	**Instance**	$\|N\|$	$\|E\|$	**Ref.**
KNOKI	10	100	[65, 60]	BKOFFC	40	1600	[65, 60]
KNOKM	10	100	[65, 60]	BKHAMC	44	1936	[65, 60]
RDGAM	14	196	[65, 60]	BKFRAC	58	3364	[65, 60]
RDPOS	14	196	[65, 60]	Prision	67	4489	[65, 60]
RDHLP	14	196	[65, 60]	email-Eu-core	1005	25571	[65, 66]
KAPFMU	15	225	[65, 60]	facebook	4039	88234	[65, 67]
KAPFMM	15	225	[65, 60]	ca-GrQc	5242	14496	[65, 68]
THURA	15	225	[65, 60]	WikiVote	7115	103689	[69, 70]
THURM	15	225	[65, 60]	Twitch EN	7126	35324	[65, 71]
NEWC [1...15]	17	289	[65, 60]	LastFM	7624	27806	[65, 72]
DAVIS	18	252	[65, 60]	cit-HepTh	9877	25998	[65, 73]
SAMPLK [1...3]	18	324	[65, 60]	BlogCatalog3	10312	333983	[65, 73]
SAMPES	18	324	[65, 60]	NetHEPT	15233	32235	[74, 75]
SAMPIN	18	324	[65, 60]	ca-AstroPh	18772	198110	[69, 76]
KRACKAD [1...21]	21	441	[65, 60]	ca-CondMat	23133	93497	[69, 76]
KRACKFR [1...21]	21	441	[65, 60]	cit-HepPh	34546	421578	[69, 76]
ZACHE	34	1156	[65, 60]	email-Enron	36692	183831	[69, 76]
ZACHC	34	1156	[65, 60]	HC Twitter	54836	89059	[65]
BKTECC	34	1156	[65, 60]	p2p-Gnutella31	62586	147892	[69, 76]
KAPFTI [1-2]	39	1521	[65, 60]	Flixster	95969	484865	[74, 77]
KAPFTS [1-2]	39	1521	[65, 60]	email-EuAll	265214	420045	[76, 78]

Table 2.1: Nodes and edges of the instances used in this thesis.

All of them are publicly available in: `https://snap.stanford.edu/`, `http://vlado.fmf.uni-lj.si/pub/networks/data/UciNet/UciData.htm` and `http://datasets.syr.edu/datasets/BlogCatalog3.html`. Relevant information about these instances is collected in Table 2.1, where some papers in which each instance has been used are included.

As shown in the previous Table 2.1, there are instances of diverse sizes, ranging from 10 to more than 250000 nodes. This variability verify the algorithm's scalability and if it is suitable develop mathematical models. Additionally, these instances are derived from well-known social networks, contributing to a comprehensive evaluation that considers the algorithm's performance across different network scales and structures.

Lastly, the density (the number of existing connections divided by the total number of possible connections) and the different number of connected components in the SNs are analyzed. Social networks with low densities often result in sparsely connected networks, indicating that influence might require more time to propagate through the network due to a lack of direct connections. The average density of the used SNs is 0.0032, suggesting that, as expected, SN have sparse connectivity. Regarding the number of connected components, the average is 1362 components. It is noteworthy that some SN consist of a single component, as previous studies recommend selecting the largest connected component within the social network. The instance with the largest number of components present a total of 15837 connected components, showcasing a range of SN with diversity in terms of components.

Chapter 3

Methodology

This Doctoral Thesis proposes various heuristic and metaheuristic procedures to solve the presented SNIMP. The use of heuristics allows us to find good solutions efficiently when exact methods are not able to find feasible solutions or require too much computational time. Metaheuristics have been demonstrated to be an effective way of solving complex combinatorial optimization problems. Examples of metaheuristics include GRASP or Path Relinking, among others. These algorithms are able to provide high-quality solutions in a short amount of time.
At present, Metaheuristics have become a field of research that combines multiple disciplines such as Computer Science or Operations Research. This has been evidenced by the numerous journal and conference papers, books, and conferences like the Metaheuristics International Conference (MIC). This chapter is devoted to the metaheuristics used to solve the $\mathcal{NP}$-hard problems discussed in Chapter 2. Specifically, we will discuss algorithms based on Greedy Randomized Adaptive Search Procedure (Section 3.1) and Path Relinking (Section 3.2).

3.1 Greedy Randomized Adaptive Search Procedure

Greedy Randomized Adaptive Search Procedure (GRASP) is a metaheuristic framework developed in the late 1980s [26] and formally introduced in Feo et al. [23]. We refer the reader to [79, 80] for a complete review of the recent advances in this methodology. Figure 3.1 shows a graphical representation of the different phases involved in the GRASP methodology, offering a comprehensive overview of it. GRASP is a multi-start framework divided into two distinct stages, where N iterations are performed. The first is a greedy, randomized, and adaptive construction phase of a solution, left section of the illustration. The second stage applies an improvement method to obtain a local optimum with respect to a certain neighborhood, starting from the constructed solution, right section of the illustration shows the swaps movement between nodes. This methodology is able to find a trade-off between the diversification of the randomized construction phase and the intensification of the local search procedure, allowing the algorithm to escape from local optima and perform a wider exploration of the search space. These two phases are repeated until a termination criterion is met, returning the best global solution found during the search.

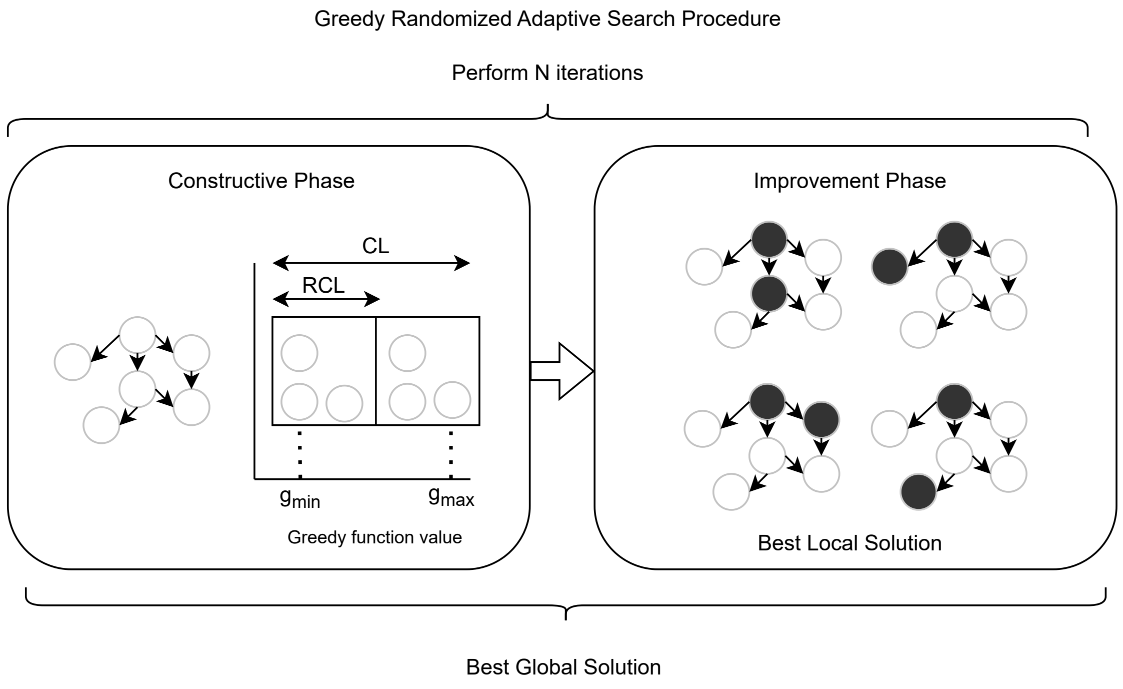


Figure 3.1: GRASP framework.

Moreover, the pseudocode presented in Algorithm 2 provides a detailed view of the main components of the GRASP framework. Without loss of generality, this pseudocode is designed for a maximization problem.

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

```
1: $v \leftarrow rnd(V)$
2: $S \leftarrow \{v\}$
3: $CL \leftarrow V \setminus \{v\}$
4: while $not\ isFeasible(S)$ do
5:     $g_{\min} \leftarrow \min_{u \in CL} g(u)$
6:     $g_{\max} \leftarrow \max_{u \in CL} g(u)$
7:     $\mu \leftarrow g_{\max} - \alpha \cdot (g_{\max} - g_{\min})$
8:     $RCL \leftarrow v \in CL : g(v) \geq \mu$
9:     $u \leftarrow rnd(RCL)$
10:    $S \leftarrow S \cup \{u\}$
11:    $CL \leftarrow CL \setminus \{u\}$
12: end while
13: $S \leftarrow Improve(S)$
14: return $S$
```

The algorithm starts by the constructive phase whose objective is to generate an initial solution. In order to favor diversification, the first node included in the solution S in the pseudocode depicted is selected at random from the set of nodes V (step 1) and included in the solution (step 2). Notice that this initial node can be selected by greedy criteria. Then, all feasible nodes that can be included in the solution are included in the Candidate List (CL) (step 3). In this example, it is assumed that all nodes are feasible, so only the initial node must be removed from the set of possible nodes to be incorporated. Then, the constructive method iteratively adds new elements to the solution until it becomes feasible (steps 4-12). In each iteration, the minimum and maximum value based on a greedy heuristic function is evaluated (steps 5-6). Then, a threshold μ is calculated (step 7), which is required for creating the Restricted Candidate List (RCL) with the most promising nodes (step 8). This threshold directly depends on the value of the input parameter α, which is in the range $[0, 1]$. Notice that this parameter indicates the greediness or randomness of the constructive procedure. On the one hand, if $\alpha = 0$, then the threshold is evaluated as $g_{\max}$, becoming a totally greedy algorithm (i.e. the *RCL* only includes the best choice in each iteration). On the other hand, if $\alpha = 1$ then $\mu = g_{\min}$, resulting in a completely random method (i.e. the *RCL* includes every feasible choice in each iteration). Finally, the next node is selected at random from the *RCL* (step 9), including it in the solution (step 10) and updating the *CL* with the feasibles nodes (step 11).

Different strategies have been proposed to select the α value, including randomly selecting it from a uniform distribution of discrete probability (in the general case) [81].

- Adapt the α value in response to the quality of the recently obtained solutions (reactive GRASP) [82, 83].
- Select the α value from a non-uniform discrete descending distribution, where there is a higher chance of selecting the best values [82, 84].
- Set the α value to a specific number (similar to the pure greedy selection) [85, 86].

This initial solution is not necessarily a local optimum, as there are numerous random selections. This implies that the construction phase does not guarantee the solution's optimality with respect to a certain neighborhood structure. The second phase of GRASP involves improving the solution generated by the constructive procedure in each iteration with the aim of reaching a local (ideally global) optimum (step 13). This phase usually enhances the initial solution, but it does not guarantee its optimality. However, experiments have shown that the quality is significantly improved. The standard GRASP implementation uses a local search procedure for the improvement phase.

This requires the definition and exploration of a neighborhood structure to move to a neighbor that produces an improvement in the objective function. This movement must keep the solution feasible, and the local improvement method is executed until no better solutions can be found in the explored neighborhood.

In [87], several elements are identified as having a significant impact on local search: the quality of the initial solution, the structure of the neighborhood, and the search strategy.

- The neighborhood structure, that is recommended to be simple to avoid large computational times.

- The local optimization algorithm applied for the neighborhood. Depending on the type of movement, the improvement methods can be classified into:
  - First improvement, that consists of selecting the movement that leads to the first neighbor that improves the incumbent solution.
  - Best improvement, which evaluates all movements in the neighborhood and selects the one with the highest profit.

  While both alternatives yield similar quality solutions, the first method is more common due to its lower computational effort. Empirical observations suggest that the second option may lead GRASP to converge more frequently to non-global optima.
- Evaluation of the greedy function of the candidates.
- The initial solution by itself. The objective is to build high-quality solutions to minimize as much as possible this fact.

Figure 3.2 illustrates the search space after four iterations and the completion of both phases. The figure showcases shaded areas with distinct patterns, each representing a different neighborhood. Additionally, four points within the search space are depicted, each resulting from the construction of the initial solution. Subsequently, depending on whether a point resides in one neighborhood or another, the intended direction of the local search is indicated by a line, assuming the objective of the problem is to minimize. It is noteworthy that, as depicted in the figure, if two points are within the same neighborhood both would converge to the same local minimum. The solution returned in this scenario would be the global minimum, representing the minimum across all local minima.

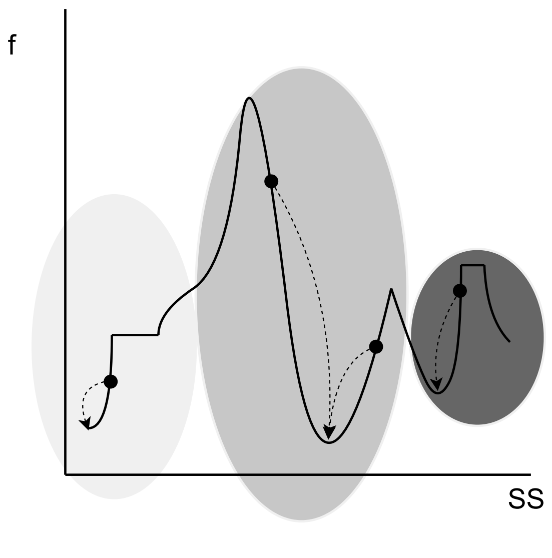


Figure 3.2: Search space when GRASP metaheuristic is applied.

The improvement phase ends when there are no movements that lead to a better solution. This termination condition ensures that the algorithm has iterated through the available movements and has reached a point where additional iterations do not yield superior outcomes. The algorithm then performs multiple iterations, systematically selecting the best local solution from each iteration. This iterative process allows for a thorough exploration of the search space.

3.2 Path Relinking

Path Relinking (PR) was originally presented as a framework for combining intensification and diversification strategies in the context of Tabu Search [88]. PR relies on the idea of connecting two high-quality solutions creating a path between them, with the expectation of finding promising solutions during the exploration of the path. The algorithm tries to iteratively include in the first solution, usually named as initial solution, attributes of the second solution, named as guiding solution. Both the initial and the guiding solutions present a high quality, and, therefore, it is expected that the path created between them explores new promising regions of the search space. Then, a path is generated from origin to destination, traversing the search space. The combination method in path relinking is based on the generation of the trajectory among solutions in the search space. The way in which the path between the initiating solution (S_i) and the guiding solution (S_g) is built is by performing movements that try to reach S_i from S_g. The path is generated by gradually removing the attributes of the initiating solution to introduce attributes that belong to the guiding solution. The goal is to find a better solution than S_i and S_g in the built trajectory between them.

To provide a more specific example of how Path Relinking works with a pair of solutions (initiating solution and guiding solution), Figure 3.3 shows how the metaheuristic works when minimizing an objective function. The figure shows the trajectory between the initiating solution and the guiding solution (gray color points) as it traverses the search space and eventually finds a solution with a better objective function value than the two from which it is based. In the particular example of the figure, in a maximization problem S_2 would be the best solution found, while in a minimization problem S_5 would be the chosen one.

The process of selecting the node to be removed and included in each iteration can be performed randomly (Random Path Relinking) [89], greedily (Greedy Path Relinking) [90], or in a more elaborated manner (Greedy Randomized Path Relinking) [91]. Notice that both Greedy Path Relinking and Greedy Randomized Path Relinking are more computationally demanding than Random Path Relinking. This is mainly because they require to generate the complete set of feasible solutions in each step of the path and also evaluate each one of them.

There are also various PR strategies designed to transition between paths, each providing a distinct perspective on traversing the search space:

- Interior Path Relinking (IPR): creates a path that connects two high-quality feasible solutions, exploring new solutions while traversing the path. The rationale behind this is that if the initiating and guiding solutions are promising, then there is a high probability of finding good solutions on the path that connects them [92].

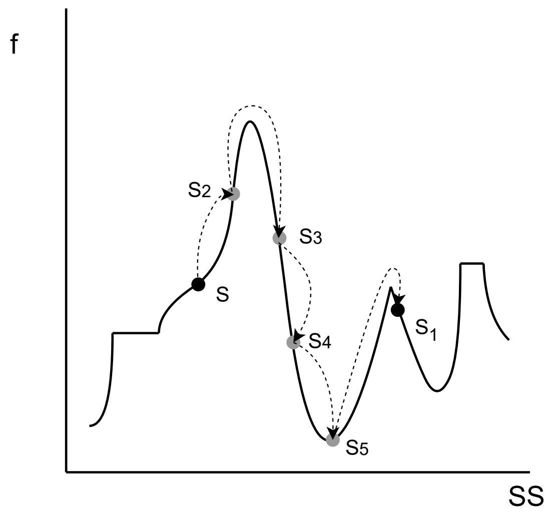


Figure 3.3: Path Relinking path between two solutions in the search space.

- Exterior Path Relinking (EPR): follows the opposite idea of IPR by focusing on the removal of nodes that are in both solutions, exploring different and diversified solution sets. If the solutions in the efficient set are similar among them, then the path created with IPR will be rather short, thus leading to a small number of new solutions explored. This behavior results in maintaining the similarity among the solutions in the efficient set. Then, it is necessary to include some diversity in the search, to explore a different region of the search space [93].
- Reactive Path Relinking (RPR): combines both strategies previously described, IPR and EPR. RPR analyzes the similarity between the initial and guiding solutions to intensify or diversify the solution. Then, if the similarity between them does not exceed a certain threshold, they are not different enough, and EPR is applied to diversify. Otherwise, the IPR is applied to intensify the search, since the initiating and guiding solutions are promising, there is a high probability of finding good diverse solutions on the path that connects them [94]. This strategy emerges as a related work from this doctoral thesis.

Algorithm 3 shows the pseudocode for the general IPR algorithm.

Algorithm 3 IPR(S_i, S_g)

```
1: $S_b \leftarrow \emptyset$
2: while $S_i \neq S_g$ do
3:     $add \leftarrow S_g - S_i$
4:     $remove \leftarrow S_i - S_g$
5:     $S_i \leftarrow S_i \setminus PRStrategy(remove)$
6:     $S_i \leftarrow S_i \cup PRStrategy(add)$
7:     if $f(S_i)$ is better than $f(S_b)$ then
8:         $S_b \leftarrow S_i$
9:     end if
10: end while
11: return $S_b$
```

Two high-quality initial solutions are required: the initiating solution S_i and the guide solution S_g. The first step of the algorithm (step 1) defines the solution that will store the best solution within the path that the algorithm will construct. Notice that neither the initiating nor the guiding one will be considered as returning solution. Then, while the initiating solution is different from the guiding solution (steps 2-10), movements will be performed. To perform these movements, it is necessary to know which elements should be added (step 3). In particular, all the elements belonging to the guiding solution which are not already in the initiating one are considered. Similarly, it is required to know which elements should be removed (step 4), that is, all elements that are in the initiating solution but not in the guiding one. Once the set of elements is obtained, one of these elements is removed according to the chosen strategy (random or greedy) (step 5). In the same way, a node is added using the chosen strategy (these can be different, and different combinations can also be tried to detect which one works better experimentally) (step 6). Next, it is needed to check whether the current solution on the path is better than the best solution during the path or not (steps 7-9). In case the solution is better by evaluating the objective function (either lower in the case of minimization or higher in the case of maximization), the current solution is placed in the S_b variable (step 8). Once the guide solution is reached, the algorithm ends and returns the best solution found during the path (without considering the initial solution and the guide solution).

PR has several variants that determine how the exploration of the search space unfolds. Some of these well-known approaches in the literature are the following:

- Static Path Relinking (SPR): This is the classical form of path relinking, given an ES, the method makes a simple path between two solutions, gradually exploring the search space between them. The best solution generated in both paths undergoes a local search method to enhance the overall outcomes [95]. The method ends when every pair of solutions in the ES has been combined.

- Dynamic Path Relinking (DPR): DPR adapts the relinking strategy in real-time on the basis of the algorithm's evolution. The dynamic nature considers updating the elite set, evaluating if the resulting solution is able to enter in the elite set following a certain criterion. If so, it can serve as a guiding solution in subsequent PR applications [79].

- Adaptive Path Relinking (APR): APR incorporates adaptability into the relinking process to adapt to the specific characteristics of the problem or the evolution of the algorithm [96]. APR departs from traditional PR implementations by considering multiple solutions simultaneously. It aims to identify, select, and systematically combine promising solution components encountered during the search, providing a diversified and initial sampling of promising areas in the solution space.

- Evolutionary Path Relinking (EP): Introduced by Resende and Werneck [97] as a post-processing phase for ES. It is applied to each pair of solutions in ES. Solutions obtained with this application of PR are considered candidates to be included in the ES, and PR is applied again to them as long as new solutions are included in the ES, contributing to the ongoing evolution of ES Solutions in the ES evolve, analogously to the reference set in Scatter Search [98].

In this Doctoral Thesis, PR combined with the GRASP [27] is used. In particular, SPR and DPR are considered. This methodology adapts the PR to the GRASP improvement phase with the objective of obtaining high-quality solutions.

Chapter 4

Joint discussion of results

This Chapter shows a review of the results provided by the addressed problems in this Thesis. The next sections explain the algorithm proposal, as well as a thorough comparison with the best method found in the literature for each problem. In every case, a final table is presented, composed of the following metrics: Avg. denoting the average objective function value in all instances, Dev. (%) that represents the average deviation to the best known value, Time (s) that shows the average computation time required by each algorithm, and finally, #Best which matches if the solution is the best solution or #Optimal that counts the number of exact values found by the algorithm. Additionally, all tables show highlighted in black the best results. This results are derived from Part II where Section 4.1 addressed the results of the SNIMP, Section 4.2 studied the results in the context of BIMP and finally Section 4.3 shows the results resulting from TSS.

4.1 Results on the Social Network Influence Maximization Problem

Richardson and Domingos [55] initially formulated the problem of selecting target nodes in SNs. However, it was not until Kempe et al. [17] were the SNIMP was solved formulating it as a discrete optimization problem later proving that it is $\mathcal{NP}$-hard [47].

According to several surveys [36, 37], greedy and approximation algorithms are proposed since this problem is showed as $\mathcal{NP}$-hard problem. In Section 2.2 several IDMs are explained being ICM one of the most extended IDMs. This probabilistic method reports the average number of activated nodes in a propagation simulation. The results of the SNIMP are published in a high-impact journal indexed in the JCR. More details can be found in Chapter 6 in Part II.

The proposed algorithm to solve SNIMP is based on the GRASP methodology [26] (see Section 3.1 for further details). Notice that, in the context of SNA, the local search method is a rather computationally demanding procedure, since real SNs need huge amount of nodes and edges, so highly scalable methods are required.

Surveys [36, 37] stated that metaheuristic algorithms are scarce in terms of SNI problems, so it is interesting to evaluate the behavior of a metaheuristic when dealing with this hard combinatorial optimization problem. In particular, GRASP uses the diversification versatility to reach high-quality solutions iteratively applying two different phases: construction and local improvement.

The constructive phase is designed to generate an initial solution, and it is usually guided by a greedy selection function which helps the constructive method to select the next element to be included in the partial solution. The first node is selected at random to favor diversification, and then the remaining nodes until reaching k elements are selected by a greedy criterion. In this work, two greedy methods are studied: the first is based on the local information of the neighbors, while the other is based on the clustering coefficient [99]. All nodes are evaluated according this criterion, conforming a candidate list CL. Then, a threshold $\mu = g_{\min} + \alpha \cdot (g_{\max} - g_{\min})$ is used to select the most promising nodes from the CL, creating a RCL. This threshold depends directly on the value of the input parameter α, which is in the range $[0, 1]$. Notice that this parameter indicates the greediness or randomness of the constructive procedure. On the one hand, if $\alpha = 0$, then the threshold is evaluated as $g_{\max}$, becoming a totally greedy algorithm (i.e., the *RCL* only includes the best choice in each iteration). On the other hand, if $\alpha = 1$ then $\mu = g_{\min}$, resulting in a completely random method (i.e., the *RCL* includes every feasible choice in each iteration).

Then, when a feasible solution is obtained, the local search phase explores the neighborhood conformed by all solutions that can be reached by performing a single movement. The neighborhood of a solution S is defined as the set of solutions that can be reached by performing a single move over S. In the context of SNIMP, we propose a swap move $\mathit{Swap}(S, u, v)$ where node u is removed from the seed set, being replaced by v, with $u \in S$ and $v \notin S$. This swap move is formally defined as:

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

Thus, the neighborhood N_s of a given solution S consist of the set of solutions that can be reached from S by performing a single swap move. More formally,

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

As stated, scalability is completely necessary in this work, and performing all possible moves would result in a rather time consuming procedure. In order to reduce the computational complexity of the search, a surrogate local search is proposed. This work propose an intelligent neighborhood exploration strategy with the aim of reducing the number of solutions explored within each neighborhood. This reduction in the size of the search space is performed by exploring just a small fraction, δ, of the available nodes for the swap node.

The selected IDM is, as in the previous method, the ICM algorithm with the corresponding Monte Carlo simulation, performing 100 iterations with a probability of influence of 0.01. These parameter values are the most extended ones in the related literature. The number of seed nodes k to conform a solution is selected in the range $k = \{10, 20, 30, 40, 50\}$ as stated in [17, 70, 100], thus obtaining $7 \cdot 5 = 35$ different problem instances (resulting from the combination of 7 networks and 5 seed set sizes).

In order to analyze the quality of the proposed algorithm, a competitive testing has been performed with the best methods found in the state of the art by considering the complete set of 35 instances derived from the SNAP[1] repository. Three algorithms are considered: CELF [44], the well-known greedy hill-climbing algorithm; CELF++ [101], the improved version of CELF [44]; and PSO [102], the particle swarm optimization algorithm which is considered the state of the art for social influence analysis according to the recent experimental study developed in [36]. Table 4.1 collects the final results obtained in this competitive testing. Notice that Avg. is not an integer value since it is the average value of the 100 runs of the ICM in the Montecarlo simulation. A final row has been also included in this table (G.Avg.) with the average values of the objective function and Time(s), computed across the set of 35 instances.

		CELF		**CELF++**		**PSO**		**GRASP**	
k	**Name**	**Avg.**	**Time (s)**	**Avg.**	**Time (s)**	**Avg.**	**Time (s)**	**Avg.**	**Time (s)**
	CA-AstroPh	157.60	**2.51**	171.81	9.40	169.85	232.40	**187.47**	8.28
	CA-CondMat	35.73	**0.67**	35.73	2.15	33.40	4.60	**36.15**	2.56
	Cit-HepPh	46.63	**1.16**	46.63	3.29	35.27	1.71	**47.20**	4.20
10	Email-Enron	383.95	**25.23**	469.63	87.68	465.24	1756.84	**489.67**	41.41
	Email-EuAll	132.96	**6.03**	130.28	307.98	107.41	37.42	**144.57**	24.42
	Wiki-Vote	108.50	**0.39**	108.50	1.00	92.16	16.40	**109.10**	6.32
	p2p-Gnutella31	16.24	**1.46**	16.23	7.83	13.38	0.95	**16.27**	1.63
	CA-AstroPh	222.63	**2.69**	234.36	9.76	222.92	889.79	**259.25**	18.53
	CA-CondMat	59.72	**0.66**	59.87	2.13	45.46	8.67	**61.05**	6.00
	Cit-HepPh	81.75	**1.11**	81.75	3.25	68.51	2.58	**82.11**	18.97
20	Email-Enron	451.24	**25.71**	547.96	88.47	544.57	4394.46	**589.65**	74.23
	Email-EuAll	214.66	**5.68**	214.54	303.01	162.32	99.98	**224.10**	28.88
	Wiki-Vote	162.49	**0.49**	162.49	1.45	141.66	41.44	**165.32**	26.03
	p2p-Gnutella31	30.82	**1.30**	30.86	7.43	24.69	0.99	**30.92**	3.80
	CA-AstroPh	266.77	**2.85**	276.69	10.48	259.90	1005.17	**312.68**	51.32
	CA-CondMat	80.87	**0.70**	82.18	2.30	66.27	11.09	**82.54**	14.11
	Cit-HepPh	113.39	**1.16**	113.39	3.45	86.22	3.69	**113.63**	42.30
30	Email-Enron	501.78	**25.49**	608.63	88.62	553.25	7594.67	**652.48**	140.71
	Email-EuAll	277.40	**5.86**	275.36	298.66	212.84	183.58	**281.30**	59.58
	Wiki-Vote	208.18	**0.64**	208.18	2.03	150.40	97.75	**214.97**	80.83
	p2p-Gnutella31	44.75	**1.24**	**44.81**	7.42	35.30	1.34	**44.81**	6.08
	CA-AstroPh	319.52	**3.11**	302.86	11.58	288.92	1492.82	**360.34**	66.97
	CA-CondMat	100.96	**0.76**	101.80	2.54	75.61	17.40	**104.38**	16.37
	Cit-HepPh	140.63	**1.27**	140.63	3.81	113.46	4.94	**141.20**	58.03
40	Email-Enron	549.64	**25.95**	658.38	92.09	634.58	9032.87	**705.03**	216.65
	Email-EuAll	323.85	**6.17**	312.47	302.47	258.46	230.12	**337.39**	165.23
	Wiki-Vote	246.02	**0.83**	246.02	2.83	182.88	115.05	**252.15**	34.60
	p2p-Gnutella31	58.26	**1.28**	58.22	7.48	51.26	1.85	**58.37**	12.69
	CA-AstroPh	361.51	**3.50**	338.28	13.11	340.54	2267.98	**399.92**	132.35
	CA-CondMat	119.29	**0.86**	120.72	2.87	106.10	10.85	**124.57**	26.51
	Cit-HepPh	165.47	**1.38**	165.47	4.26	126.77	6.35	**166.77**	65.20
50	Email-Enron	597.26	**27.02**	680.29	96.71	662.67	10063.47	**744.38**	157.26
	Email-EuAll	361.51	**6.68**	357.43	304.03	258.15	321.81	**375.03**	161.59
	Wiki-Vote	277.65	**1.09**	277.65	3.76	188.82	181.07	**287.66**	86.39
	p2p-Gnutella31	71.80	**1.34**	71.90	7.76	64.63	2.74	**72.08**	17.42
	G.Avg.	208.33	**5.55**	221.49	60.09	195.54	1146.71	**236.37**	39.04

Table 4.1: Competitive testing of the proposed GRASP algorithm with respect to the best algorithms found in the literature: CELF, CELF++, and PSO. Best results are highlighted with bold font.

First, the results obtained with PSO are highlighted, since it is ranking the last one even being considered the state of the art for this problem. The rationale behind

[1]`http://snap.stanford.edu/`

this is that the original work [102] considers small size instances (from 410 to 15233 nodes) and the quality of the solutions provided by PSO deteriorates when the instance size increases, as it can be derived from Table 4.1. Meanwhile, CELF and its improved version CELF++ are able to reach better solutions, still being competitive with the state-of-the-art algorithms. However, only CELF++ is able to match the best-known solution in 1 instance (out of 35). Finally, the best results are obtained with the proposed GRASP algorithm, which is able to reach the best solution found in all the 35 instances. Furthermore, the computing time is smaller than the second best algorithm, CELF++ (39.04 versus 60.09 seconds on average).

Analyzing the computing time required for each algorithm, it can be clearly seen that CELF and CELF++, as completely greedy approaches, are not really affected by increasing the size of the seed set. On the contrary, the computing time required for PSO and GRASP is affected by the size of the seed set since larger k-values lead the local improvement method to perform a larger number of iterations. However, if a closer vision of the obtained results is taken with GRASP, it can be concluded that the increase in the number of iterations and, therefore, in the computing time allows the algorithm to reach better solutions. In the case of PSO, the increase of computing time is even much more noticeable but it does not usually result in better solutions, suggesting that the PSO algorithm is particularly suitable for solving small size instances.

In order to validate these results, a non-parametric Friedman test has been conducted for ranking all the compared algorithms. The p-value obtained, smaller than 0.0005, confirms that there are statistically significant differences among the algorithms. The algorithms sorted by ranking are GRASP (1.00), CELF++ (2.44), CELF (2.79), and PSO (3.77). Finally, it is performed the well-known non-parametric Wilcoxon statistical test for pairwise comparisons, which answers the question: do the solutions generated by both algorithms represent two different populations? The resulting p-value smaller than 0.0005 when comparing GRASP with each other algorithm confirms the superiority of the proposed GRASP algorithm. Therefore, GRASP emerges as one of the most competitive algorithms for the SNIMP, being able to reach high-quality solutions in small computing time.

4.2 Results on the Budgeted Influence Maximization Problem

Companies can spend a specific budget in their marketing campaigns, which is not modeled in the classic SNIMP. With the aim of including this feature, Nguyen [57] formally defined the BIMP inspired by SNIMP, showing its $\mathcal{NP}$-hardness based on SNIMP. As it was aforementioned there are several IDM, this work uses: ICM, WCM and TV.

Banerjee [36] published the latest survey in SIM, becoming one of the most relevant research works in the area of influence maximization problems. The algorithm named ComBIM proposed by [45] is considered the state of the art for BIMP. ComBIM provides a community-based solution that provides the best results in the literature as far as our knowledge, so it will be considered as the algorithm to benchmark our proposal. More details can be found in Chapter 7 in Part II.

The algorithmic proposal is based on Greedy Randomized Adaptive Search Procedure (GRASP) methodology where a novel efficient and effective heuristic for selecting the seed set in the constructive phase is designed. This greedy criterion, named g_{dist} leverages the node seed distribution, it prioritizes nodes that do not have selected neighbors as a seed node, with the aim of reaching a larger number of non-influenced users by exploring regions that have been mainly ignored until that point. In order to do so, the value of a node is directly its degree, but penalizing it if some of its neighbor nodes have already been selected. The penalization has been experimentally set by halving the degree. More formally

$$g_{dist} = \begin{cases} d_u^+ & \text{if } v \notin S,\ \forall v \in N_u^+ \\ \frac{d_u^+}{2} & \text{otherwise} \end{cases}$$

Once the constructive phase ends a local search phase is performed. The main difference with SNIMP is the budget (B), since in BIMP there is not an exact number of nodes required. Then, a movement that removes one node and inserts another one may result in an unfeasible solution which exceeds the available budget. The neighborhood of a solution S is defined as the set of solutions that can be reached by performing a single move over S. Then, it is necessary to define the move that will be considered in the context of BIMP. Specifically, the move, named as *Replace*(S, u, P), involves removing node u from the solution and replacing it with the set of nodes in P, with $P \in V \setminus S$. Notice that, in order to reach a feasible solution, the sum of the cost of nodes in P must be smaller or equal than $B + C(u)$ (since u will be removed, its cost must not be taken into account). More formally,

$$\mathit{Replace}(S, u, P) = S \setminus \{u\} \cup P$$

Then, given a solution S, the neighborhood $N_R(S)$ is defined as the set of feasible solutions that can be reached with a single *Replace* move. In mathematical terms,

$$N_R(S) = \left\{ S' \leftarrow \mathit{Replace}(S, u, P)\ \forall u \in S \land \forall P \in V \setminus S : \sum_{p \in P} C(p) \leq B + C(u) \right\}$$

Even considering an efficient implementation of the objective function evaluation, the vast size of the resulting neighborhood makes the complete exploration of the neighborhood not suitable for the BIMP. Therefore, number of evaluations that the local search performs is limited with the aim of having a computationally efficient method. It is worth mentioning that, if the number of iterations used in the IDMs are limited, then it is interesting to firstly explore the most promising neighbors of the considered neighborhood.

The parameters used are as follow: as it is customary in SIM problems, 100 Monte Carlo simulations are performed on all IDMs models. The total budget B to conform a solution is selected in the range $B = \{2000, 6000, 10000, 140000, 180000, 22000, 26000\}$ as stated in [45], thus obtaining $3 \cdot 7 = 21$ different problem instances for each IDM.

Taking into account that 4 IDMs are considered, the total number of instances are $21 \cdot 4 = 84$. In order to analyze the quality of the proposed algorithm, a competitive testing is performed with the best method found in the state of the art, ComBIM. Table 4.2 collects the final results obtained in this competitive testing, for each IDM.

IDM	Algorithm	Avg.	Time (s)	Dev (%)	#Best
ICM(1%)	ComBIM	8319.68	214.97	17.64%	0
	GRASP	**8872.61**	**117.06**	**0.00%**	**21**
ICM(2%)	ComBIM	14467.65	215.31	6.49%	3
	GRASP	**14828.77**	**146.21**	**0.07%**	**18**
WCM	ComBIM	2277.79	214.04	57.49%	0
	GRASP	**10087.08**	**97.80**	**0.00%**	**21**
TV	ComBIM	1976.11	214.68	39.10%	0
	GRASP	**2677.58**	**69.65**	**0.00%**	**21**
Summary	ComBIM	6760.31	214.75	30.18%	3
	GRASP	**9116.51**	**107.68**	**0.02%**	**81**

Table 4.2: Competitive testing of the proposed GRASP algorithm with respect to state of the art algorithm ComBIM. Best results are highlighted with bold font.

The results show how GRASP is able to obtain high-quality solutions (81 best solutions out of 84), requiring half of the computing time (107.68 seconds vs 214.75 seconds). Although GRASP is able to outperform ComBIM in all IDMs considered, the most remarkable results in terms of quality are obtained when using WCM and TV. Specifically, ComBIM is able to reach the best solution just in three instances when using ICM (2%). In this case, the deviation of GRASP is 0.07%, indicating that it is really close to that best solution. In view of these results, GRASP emerges as one of the most competitive algorithms for BIMP.

Finally, the well-known non-parametric Wilcoxon statistical test for pairwise comparisons is performed, which answers the question: do the solutions generated by both algorithms represent two different populations? The resulting p-value smaller than 0.0001 when comparing GRASP with ComBIM confirms the superiority of the proposed GRASP algorithm. In particular, GRASP is able to obtain 81 out of 84 positive ranks, 3 negative ranks, and 0 ties.

4.3 Results on the Target Set Selection Problem

Finally, the latest solved problem is related to the family of Target Set Selection Problems where two variants can be distinguished: guaranteeing reaching the complete network (or even a certain part of it) with the minimum number of initial users or maximizing the number of users reached while not exceeding an initial budget. This proposal focuses on solving the latter, which is usually named the maximum effort-reward GAP Target Set Selection problem (Max-TSS), a $\mathcal{NP}$-hard problem [59]. More details can be found in Chapter 8 in Part II.

Our proposal is related to Path Relinking to solve the Max-TSS problem [103]. Path Relinking requires from a method to generate high-quality and diverse solutions in order to create promising paths during the search in both static and dynamic variants. Although these solutions can be generated at random, it has been experimentally shown in several works that designing a specific constructive and local improvement method for the problem under consideration usually leads to better results [92, 104, 105, 106].

In the context of influence maximization problems, the Greedy Randomized Adaptive Search Procedure has been shown to be an effective and efficient method to generate them [69]. The constructive method proposed for Max-TSS problem follows the GRASP philosophy of diversification by avoiding totally greedy decisions. With the aim of increasing diversity, the method selects the first node to be included at random from the set of users V, initializing the solution under construction S. Then, the CL is created with all the nodes but v. The constructive method iteratively adds a node to the solution while the budget is not exceeded and the CL is not empty. In each iteration, the minimum and maximum value of a certain greedy function are computed. The aim of the greedy function is to evaluate how promising a candidate is, and it is a key part of the constructive procedure. With this threshold, the RCL is created, containing all the nodes whose greedy function value is larger than or equal to the threshold μ, considering that they do not exceed the maximum budget. Once the RCL is constructed, the next element is selected at random from it (since all the nodes in RCL are promising) to favor diversity. The selected node is then added to the incumbent solution, updating the CL by removing it.

The greedy function is traditionally considered in the GRASP literature, and it consists of directly evaluating the objective function value if the node under evaluation were added to the incumbent solution, i.e., it represents the direct contribution of the node to the solution under construction. More formally,

$$g_{of}(c, S) \leftarrow TSS(S \cup \{c\})$$

The main drawback is that the evaluation of the objective function for the Max-TSS is a rather computationally demanding process, so a new greedy function is proposed with the aim of reducing the computational effort of the evaluation, since it will be performed in each iteration of the construction process. The second greedy function proposed, named $g_{dg}(c, S)$, considers that the relevance of a node is directly proportional to its degree. In other words, if a node is connected to several nodes, then it will probably influence a large amount of its adjacent nodes. Then, this greedy function is evaluated as the degree of the evaluated node:

$$g_{dg}(c, S) \leftarrow |u \in V : (c, u) \in E|$$

The second phase of GRASP consists of a local improvement method that finds a local optimum starting from the initial solution. From the initial solution S, it is not possible to add new nodes, since the constructive procedure stops when the maximum budget is exceeded with any of the remaining nodes. Therefore, the proposed move operator is defined in two steps: remove and add. In particular, the move operator removes a node from the solution and then iteratively adds nodes until the maximum budget is reached.

In the context of TSS, the computational effort is a critical part of the algorithm, so the first improvement method has been decided to be used with the aim of reducing the computing time for performing a local search method. With the aim of avoiding biasing the search, the neighborhood is explored at random, performing the first movement that results in a better solution based on a move operator which removes a node from the solution and replaces it with all the nodes that can be added without exceeding the allowed budget.

With the aim of further reducing the computational effort of the local search method, three improvements are proposed. The first improvement tries to escape cycling the search by avoiding the exploration of already visited solutions. In order to do so, each visited solution is associated with a unique number, i.e., hash code, evaluated following a hash function. Then, every time a solution is visited, it is evaluated if its corresponding hash code has not already been included in the set of visited solutions. If so, the method undoes the move and continues with the next iteration, avoiding repeating the exploration of the same region of the search space. The second improvement is devoted to limit the nodes explored during the search, discarding those nodes which will result in an unfeasible solution. Then, the candidate nodes to be added are sorted with respect to their effort value in ascending order. Only those nodes whose effort value is smaller than or equal to the available budget are explored. Additionally, to favor diversity, the exploration is performed at random among all nodes that satisfy this constraint. The objective of the last improvement is to reduce the computing time required to evaluate the influence of a node by caching it. Specifically, the influence of a node (i.e., those nodes that are affected by its activation), is calculated at the beginning of the local search method. Then, every time a node is selected to be removed or added to the solution, the influence of that node over the other nodes of the graph is updated. As a result, it is not necessary to completely evaluate the objective function in each iteration but to check the corresponding pre-calculated influence.

After that, SPR and DPR are used to improve the solution. Path Relinking strategies require from a set of high-quality solutions, usually known as Elite Set, which are combined. In the context of TSS, given two solutions S_i and S_g to be combined, the path-creation method designed for the Max-TSS problem iteratively removes nodes belonging to S_i but not to S_g, i.e., $S_i \setminus S_g$, and includes nodes which are in S_g but not in S_i, i.e., $S_g \setminus S_i$. The method stops when S_i has completely become S_g and no more nodes can be removed / added. Since the computational effort is a critical part of TSS, Random Path Relinking has been selected which, additionally, increases the diversity of the search. In the proposed method, every pair of solutions in the Elite Set are combined.

The dataset used to perform the experiments was derived from the best algorithm found in the literature to provide a fair comparison. This set of instances is conformed with 82 instances derived from real-life social networks which have been extensively used in social network analysis. The main drawback of this dataset is that the largest network is conformed with 58 nodes, which might not be challenging enough considering the current size of social networks. To mitigate this drawback, 8 additional instances have been added, with sizes ranging from 67 to 10312 nodes.

The best previous approach is an exact method which shows its limits when dealing with larger and more complex instances. In order to evaluate the contribution of our proposal, an additional metaheuristic algorithm for performing a comparison with SPR and DPR has also been included. In particular, Simulated Annealing (SA) has been selected, which is a metaheuristic based on the analogy between an optimization process and a thermodynamic process known as annealing. It is a search method which tries to escape from local optima allowing to explore worse solutions if those solutions satisfy certain criteria. SA was originally proposed by Kirkpatrick et al. [107] and it has been successfully applied in a wide variety of hard combinatorial optimization problems. SA has been successfully applied in several works related to influence maximization problems [108, 109]. Additionally, the well-known CELF selection algorithm [44], which has been widely used in the context of influence maximization problems and, in particular, in Max-TSS [110], is included in the comparison. CELF is a greedy procedure which leverages the submodularity property of the network to considerably reduce the computational effort of the greedy hill-climbing algorithm. The main objective of this optimization is to scale to large problems, reaching near optimal placements. This improvement makes CELF approximately 700 times faster than the original procedure. There exists several implementation of SA which are publicly available. For this work, the one provided by Metaheuristic Optimization framewoRK (MORK) has been selected [111], which has been tested over several hard optimization problems [112, 113].

The results are divided into two different experiments. First of all, SPR and DPR are evaluated when considering the set of original instances in which the exact method is able to reach the optimal value. Table 4.3 shows the results obtained. As it can be derived from the results, SPR performs slightly better than DPR in this set of instances, being able to reach 79 out of 82 optimal solutions, while DPR reaches 76. It is important to remark that the average deviation of both methods, smaller than 0.05, indicates that in those instances in which neither SPR nor DPR are able to reach the optimal value, they stay really close to it. In order to confirm this hypothesis, a pairwise non-parametric Wilcoxon statistical test between SPR and Gurobi solver has been conducted, obtaining a p-value equal to 0.109, which indicates that, with a confidence interval of 95%, there are not statistically significant differences between those methods. Regarding the SA, it is worth mentioning that it is able to reach 76 out of 82 instances with a deviation of 4.86%, requiring negligible time such as SPR and DPR. With respect to CELF, the algorithm requires from irrelevant computing times as DPR, SPR and SA, but it only reaches 61 out of 82 optimal solutions, with a deviation of 12.19%. From these results, two main conclusions can be obtained: SA is a competitive algorithm for the Max-TSS, and the proposed DPR and SPR significantly contribute to the quality of the obtained solutions, as it can be seen in the smaller deviation with respect to the optimal value.

The last experiment is devoted to evaluate the performance of the proposed algorithms and the Gurobi solver when considering the most challenging and realistic instances. Table 4.4 shows the results obtained in the set of large instances. In this case, the results are shown disaggregated, since it is conformed with 8 instances that can be individually analyzed. It is worth mentioning that the Gurobi solver is only

Algorithm	Avg.	Dev. (%)	Time (s)	#Optimal
Gurobi	**45.38**	**0.00**	117.14	**82**
DPR	44.34	2.58	**0.01**	76
SPR	44.54	1.82	**0.01**	79
SA	46.31	**4.86**	**0.01**	76
CELF	42.07	12.19	**0.01**	61

Table 4.3: Comparison of SPR, DPR, SA, CELF and Gurobi solver when considering the original dataset in which Gurobi is able to reach the optimal value.

able to provide the optimal solution for 2 out of 8 instances derived from the new set of complex instances marked with an asterisk in the corresponding instance name. For the remaining instances, Gurobi is not even able to load the model in memory, which highlights the need to consider metaheuristic algorithms for this set of challenging instances. In particular, in those instances where Gurobi reaches the optimal value, SA, SPR and DPR are also able to find it. However, CELF is not able to reach the optimal value for these two instances. Additionally, for the instance EMAIL-EUCORE, Gurobi requires almost 30h to find the optimal value, while SA requires 268s, DPR 262s and SPR only 85s.

	CELF				SA			
Instance	Avg.	Dev. (%)	Time (s)	#Best	Avg.	Dev. (%)	Time (s)	#Best
PRISON*	240	11.11	0.02	0	**270**	**0.00**	0.34	**1**
EMAIL-EU-CORE*	4672	1.79	22.71	0	**4757**	**0.00**	267.78	**1**
EGO-FACEBOOK	**19462**	**0.00**	3609.75	**1**	**19462**	**0.00**	1044.35	**1**
CA-GRQC	22487	5.05	12441.33	0	**23684**	**0.00**	5364.14	**1**
TWITCH_EN	25060	0.22	16833.33	0	23853	5.03	5885.88	0
LASTFM_ASIA	**25005**	**0.00**	26017.00	**1**	23000	8.02	6160.10	0
CA-HEPTH	44451	1.16	165624.79	0	**44972**	**0.00**	9105.73	**1**
BLOG_CATALOG3	**46732**	**0.00**	44674.80	**1**	46418	0.67	8407.40	0
Summary	23514.13	2.23	33652.97	3	23302.00	1.71	4534.97	5

	SPR				DPR			
Instance	Avg.	Dev. (%)	Time (s)	#Best	Avg.	Dev. (%)	Time (s)	#Best
PRISON*	**270**	**0.00**	**0.07**	**1**	**270**	**0.00**	0.16	**1**
EMAIL-EU-CORE*	**4757**	**0.00**	**84.48**	**1**	**4757**	**0.00**	262.59	**1**
EGO-FACEBOOK	**19462**	**0.00**	**279.47**	**1**	**19462**	**0.00**	769.70	**1**
CA-GRQC	23630	0.23	**442.07**	0	**23684**	**0.00**	2166.29	**1**
TWITCH_EN	24853	1.06	**680.77**	0	**25116**	**0.00**	2230.12	**1**
LASTFM_ASIA	24556	1.80	**760.97**	0	24780	0.90	2409.55	0
CA-HEPTH	44909	0.14	**2140.92**	0	**44972**	**0.00**	7743.06	**1**
BLOG_CATALOG3	46595	0.29	**1336.91**	0	46692	0.09	8705.44	0
Summary	23629.00	0.44	**715.71**	3	**23716.63**	**0.12**	3035.87	**6**

Table 4.4: Comparison of CELF, SA, SPR and DPR over the set of largest and most complex instances.

Analyzing the instances in which Gurobi is not able to even load the model, SPR requires from smaller computing time than DPR in general, but it provides worse results in terms of quality. Regarding SA, it is able to provide competitive results in these challenging instances. Specifically, SPR reaches the best solution in 3 out of 8 instances, SA reaches 5 out of 8 best solutions, and, finally, DPR reaches all the best solutions but for two instances in which CELF is able to provide slightly better results. It is worth mentioning that CELF requires from approximately five times the computing time required by DPR, thus being DPR much more scalable for large scale networks. In terms of deviation, CELF provides the worst results with a 2.23%, followed by SA with 1.71%, but it is considerably larger than the one obtained by SPR and DPR. Specifically, the average deviation obtained by SPR is considerably small (0.44%), and DPR is able to reach a deviation of 0.12%. Since the deviation of DPR is really close to 0%, a pairwise non-parametric Wilcoxon statistical test has been conducted to evaluate if there are statistically significant differences between SPR and DPR. The resulting p-value of 0.04, smaller than 0.05, indicates that DPR is statistically better than SPR. These results highlights the contribution of SPR and DPR to the state of the art of Max-TSS.

As a conclusion, both SPR and DPR are able to provide promising solutions for the TSS, each one of them being suitable for different situations. On the one hand, if the computing time is a hard constraint, we do recommend considering SPR since the quality of the solutions is not drastically worse. On the other hand, if the maximum computing time is not a critical part of the problem, DPR is able to provide better results in terms of quality. The proposed algorithms have been compared with a Simulated Annealing implementation, which has been successfully applied in several influence maximization problems, and with CELF, which is a widely used method in the context of influence maximization and, particularly, in Max-TSS. The results obtained highlight the appropriateness of designing a specific algorithm for solving the TSS such as the proposal of this research.

Chapter 5

Conclusions and future work

In this chapter, the conclusions of each variant of the problems tackled and general future work of this Doctoral Thesis are presented. Section 5.1, shows the conclusions of Social Network Influence Maximization Problem (SNIMP), Section 5.2 highlight the conclusions related to Budgeted Influence Maximization Problem (BIMP) and finally Section 5.3 remarks the main conclusions about Target Set Selection (TSS). Last Section 5.4 presents the general future devised from this Doctoral Thesis related to Social Influence Problems. It is worth mentioning that in each section a link to a repository (with source code, instances, and results) is presented to ease further comparisons.

5.1 Conclusions on the Social Network Influence Maximization Problem

In this paper a quick Greedy Randomized Adaptive Search Procedure (GRASP) algorithm for solving the SNIMP has been presented. Two constructive procedures are proposed, with the one based on the two-step neighborhood being more competitive one than that based on the clustering coefficient. Furthermore, the idea of using local information allows the algorithm to construct a complete solution in a small computing time. Then, a local search based on swap moves is presented. Since an exhaustive exploration of the search space is not suitable for this problem, we propose an intelligent neighborhood exploration strategy which limits the region of the search space to be explored, focusing on the most promising areas. This rationale leads us to provide high-quality solutions in reasonable computing time, even for the largest instances derived from real-world SNs commonly considered in the SNIMP area. Since the intelligent neighborhood exploration strategy is parameterized, if the computing time is not a relevant factor, the region explored can be easily extended to find better solutions, thus increasing the required computational effort. This fact makes the proposed GRASP algorithm highly scalable. The results obtained are supported by Friedman test and then the pairwise Wilcoxon test, confirming the superiority of the proposal with respect to the classical and state-of-the-art solution procedures in the area.

This work was presented and published in the Journal of Ambient Intelligence and Humanized Computing (JCR Q2) entitled as *A quick GRASP-based method for influence maximization in social networks* [69]. The impact factor of this journal is 3.662, located at 68/145 in the area of Computer Science, Artificial Intelligence and 73/164 in Information Systems. The source code has also been made publicly available[1] to ease further comparison. It is worth mentioning that this research has also been presented in: *Parallel Problem Solving from Nature–PPSN XVI: 16th International Conference, PPSN* held in Leiden, The Netherlands from September 5-9 2020; *International Conference on Variable Neighborhood Search (ICVNS)* held in Abu Dhabi, United Arab Emirates from 22-24 Mar 2021; and *XIX Conferencia de la Asociación Española para la Inteligencia Artificial (CAEPIA 2021): avances en Inteligencia Artificial* held in Málaga from 22-24 Sep 2021. Further details and the complete manuscript can be found in Chapter 6.

5.2 Conclusions on the Budgeted Influence Maximization Problem

An efficient, effective and scalable algorithm based on Greedy Randomized Adaptive Search Procedure (GRASP) framework was developed. This framework can be configured according to the time requirements. Scalability is achieved by using local information in constructive methods and avoiding an exhaustive search in local search (selecting the most promising nodes). The results and the comparison with the previous algorithms are developed using three different IDMs for BIMP, considering the probabilistic Monte Carlo algorithm for the evaluation of the objective function.

Finally, an infodemic case study is analyzed from the influence maximization perspective. Specifically, an instance is built based on 386384 tweets about the American Health Care Act (AHCA). An experiment is performed, showing the superiority of GRASP when comparing it with ComBIM in 21 out of 27 available instances. The most influential users are analyzed, showing their relevance in the topic studied, being most of them senators, comedians, writers, or newspapers.

Chapter 7 shows the accepted work entitled as *An efficient and effective GRASP algorithm for the Budget Influence Maximization Problem* [65], in the Journal of Ambient Intelligence and Humanized Computing. This journal has an impact factor of 3.662, located at 68/145 in the area of Computer Science, Artificial Intelligence and 73/164 in Information Systems. As the previous work, to ease further comparisons source code, instances and the complete results can be found publicly available in the following link [2].

[1]`https://grafo.etsii.urjc.es/SNIMP`
[2]`https://grafo.etsii.urjc.es/BIMP`

5.3 Conclusions on the Target Set Selection problem

This work presents an algorithm based on Path Relinking [27] for solving the TSS problem. The potential of GRASP coupled with a Path Relinking combination strategy has been proven. Specially, regarding to the computing time required by the algorithm to reach high-quality solutions, the proposal emerged as the best method in the state of the art.

In particular, the main contributions of this work are as follows. Two different variants of GRASP + Path Relinking have been proposed: Static Path Relinking and Dynamic Path Relinking. Both SPR and DPR are able to provide promising solutions for the TSS, each one of them being suitable for different situations. On the one hand, if the computing time is a hard constraint, we do recommend considering SPR since the quality of the solutions is not drastically worse. On the other hand, if the maximum computing time is not a critical part of the problem, DPR is able to provide better results in terms of quality.

It is worth mentioning that a Path Relinking path strategy (Reactive Path Relinking) was proposed and accepted in a related work in this thesis [94] (see Section 3.2).

Novel improvements to reduce the computational effort of the local search method: escape from cycling the search by avoiding the exploration of already visited solutions using hash function in solutions. Then, every time a solution is visited, it is evaluated if its corresponding hash code has not already been included in the set of visited solutions. The explored nodes are limited during the search, and discarding those nodes which will result in an unfeasible solution. They are sorted with respect to their effort value in ascending order. Then, only those nodes whose effort value is smaller than or equal to the available budget are explored. The objective of the last improvement is to reduce the computing time required to evaluate the influence of a node by caching it.

Experiments have shown that the combination of GRASP with PR results in high-quality solutions. The efficient implementation of the algorithm and the quality of applied heuristics allow the algorithm to overcome the previous work, supported by statistical tests.

The incumbent work was published results in a publication, *Dynamic Path Relinking for the Target Set Selection problem* [114] in Knowledge-Based Systems which has an impact factor of 8.800 and it is located at 19/145 in the field of Computer Science, Artificial Intelligence. The full document is included in Chapter 8 of Part II, the complete source code and instances are publicly available to ease further comparisons, as well as the complete results[3]. It is worth mentioning that this research has also been presented in the *XL Congreso Nacional de Estadística e Investigación Operativa (SEIO)* held in Elche, Spain from 7-10 Nov 2023.

[3]https://grafo.etsii.urjc.es/TSS

5.4 Future Work

During this thesis related to the analysis of influence maximization problems, several future works have been found on different areas, such as follows: instances, diffusion models and algorithmic proposals.

The main drawback during the research is the open access to the instances to provide reproducibility against other proposals, all the works related to this doctoral thesis are public in different repositories for their access. To improve scientific works, researchers should facilitate the same. Nowadays through data mining techniques or the use of APIs it is possible to get social networks, however sometimes it is unfeasible to obtain a complete social network, society advances, data increases and the use of static social networks is becoming outdated. The use of dynamic social networks in real time is a field of research where adapting these algorithms is very interesting, this will allow to execute marketing campaigns in real time varying according to the iterations and being able to improve the results.

Current influence propagation models are one of the main drawbacks when evaluating algorithms. Studies take advantage of parallelization through algorithmic techniques and it is a field of interest to reduce computational costs. Another way to improve the current system would be to create more robust models to evaluate influence propagation. Currently models are approximations based mainly on structural information that may fail to generate an accurate diffusion pattern. Pei et al. [115] shows that the actual pattern of information propagation is likely to be affected by factors such as human behavior, common preferences or beliefs, and social reinforcement. Creating a model that evaluates diffusion that take these factors into account can model more realistically.

Researchers [36, 37] shows that Metaheuristic algorithms and Deep Learning are scarce in Social Influence Problems. One of the reasons for the use of current algorithms is the structure of the network. Existing research essentially assumes that the network structure is fixed and does not change over time. This can help in finding solutions, but one can easily realize that more dynamic elements in the network structure can make it more realistic to capture the actual interactions in real-world social networks. To be successful in solving these problems, more attention should be paid to the efficiency and scalability of the proposed methods. In practical terms, it will not be feasible for a method to take longer to find a solution that maximizes the propagation of influence than the time required for it to propagate, as there is a risk that by the time a solution has been found the situation will have changed completely.

Nowadays, the proliferation of fake news and the spread of rumors are on the rise, prompting a heightened interest in issues related to influence minimization. Toward the conclusion of this doctoral thesis, a research stay was conducted in Antwerp (Belgium) with Professor Kenneth Sörensen. During this research stay, the problem of influence minimization was addressed, which includes blockers (nodes that, when receiving information, do not transmit it). A novel methodology using machine learning was employed, in which, given the characteristics of a solution, it is possible to determine the features that make good solutions. This is particularly noteworthy as it aligns with the objective of any heuristic. The study has yielded promising results, although it remains an ongoing work in progress. It is worth mentioning that it has also been accepted in the *38th Annual Conference of the Belgian Operational Research Society (ORBEL38)* held in Antwerp, Belgium, from 8-9 Feb 2024.

Finally, SNI problems are growing due to the real-world use that is obtained from them. This makes the problem interesting to both researchers and industry, having a need for scalability and the need of fast and efficient algorithms, allows the use of metaheuristic algorithms to be useful due to their particular characteristics.

Part II

Publications

Contents

Chapter 6

A quick GRASP-based method for influence maximization in social networks

The journal paper associated to this part is:

- Lozano-Osorio, I.; Sánchez-Oro, J.; Duarte, A; Cordón, O. A quick GRASP-based method for influence maximization in social networks. Journal of Ambient Intelligence and Humanized Computing (2023). `https://doi.org/10.1007/s12652-021-03510-4` [69]
- Submitted: 04 February 2021.
- Accepted: 13 September 2021.
- Impact Factor (JCR 2021): 3.662.
- Subject Category: Computer Science, Artificial Intelligence - 68/145 (Q2).

It is worth mentioning that this research has also been presented in the following conferences:

- Lozano-Osorio, I.; Sánchez-Oro, J.; Duarte, A; Cordón, O. New metaheuristic algorithms for the analysis of the user influence in social networks. Parallel Problem Solving from Nature–PPSN XVI: 16th International Conference, PPSN 2020, Leiden, The Netherlands, September 5-9.
- Lozano-Osorio, I.; Sánchez-Oro, J.; Duarte, A; Cordón, O. Measuring the influence of users in social networks using Variable Neighborhood Search. In International Conference on Variable Neighborhood Search (ICVNS). 22-24 Mar 2021, Abu Dhabi, United Arab Emirates.
- Lozano-Osorio, I.; Sánchez-Oro, J.; Duarte, A; Cordón, O. Nuevos algoritmos metaheurísticos para el análisis de la influencia de los usuarios en las redes sociales. In XIX Conferencia de la Asociación Española para la Inteligencia Artificial (CAEPIA 2021): avances en Inteligencia Artificial. 22-24 de septiembre de 2021 Málaga, España.

Journal of Ambient Intelligence and Humanized Computing
https://doi.org/10.1007/s12652-021-03510-4

ORIGINAL RESEARCH

A quick GRASP-based method for influence maximization in social networks

Isaac Lozano-Osorio[1] · Jesús Sánchez-Oro[1] · Abraham Duarte[1] · Óscar Cordón[2]

Received: 4 February 2021 / Accepted: 13 September 2021

Abstract
The evolution and spread of social networks have attracted the interest of the scientific community in the last few years. Specifically, several new interesting problems, which are hard to solve, have arisen in the context of viral marketing, disease analysis, and influence analysis, among others. Companies and researchers try to find the elements that maximize profit, stop pandemics, etc. This family of problems is collected under the term Social Network Influence Maximization problem (SNIMP), whose goal is to find the most influential users (commonly known as seeds) in a social network, simulating an influence diffusion model. SNIMP is known to be an $\mathcal{NP}$-hard problem and, therefore, an exact algorithm is not suitable for solving it optimally in reasonable computing time. The main drawback of this optimization problem lies on the computational effort required to evaluate a solution. Since each node is infected with a certain probability, the objective function value must be calculated through a Monte Carlo simulation, resulting in a computationally complex process. The current proposal tries to overcome this limitation by considering a metaheuristic algorithm based on the Greedy Randomized Adaptive Search Procedure (GRASP) framework to design a quick solution procedure for the SNIMP. Our method consists of two distinct stages: construction and local search. The former is based on static features of the network, which notably increases its efficiency since it does not require to perform any simulation during construction. The latter involves a local search based on an intelligent neighborhood exploration strategy to find the most influential users based on swap moves, also aiming for an efficient processing. Experiments performed on 7 well-known social network datasets with 5 different seed set sizes confirm that the proposed algorithm is able to provide competitive results in terms of quality and computing time when comparing it with the best algorithms found in the state of the art.

Keywords Information systems · Social networks · Influence maximization · Network science · Viral marketing · GRASP

1 Introduction

Nowadays, millions of users are involved in social networks (SNs), growing exponentially the number of active users. This growth is extended to the amount of behavioral data and, therefore, all classical network-related problems are becoming computationally harder. SNs can be defined as the representation of social interactions that can be used to study the propagation of ideas, social bond dynamics, disease propagation, viral marketing, or advertisement, among others (D'angelo et al. 2009; Klovdahl 1985; Barabási and Pósfai 2016; Reza et al. 2014).

SNs are used not only for spreading positive information but also malicious information. In general, research devoted to maximize the influence of positive ideas is called Influence Maximization (Nguyen Hung et al. 2016). Thus, solving successfully this problem allows the decision-maker to decide the best way to propagate information about products and/or services. On the contrary, SNs can be also used for the diffusion of malicious information like derogatory rumors, disinformation, hate speech, or fake news. These examples motivate research about how to reduce the

✉ Óscar Cordón
ocordon@decsai.ugr.es

Isaac Lozano-Osorio
isaac.lozano@urjc.es

Jesús Sánchez-Oro
jesus.sanchezoro@urjc.es

Abraham Duarte
abraham.duarte@urjc.es

[1] Universidad Rey Juan Carlos, Móstoles, Spain

[2] Universidad de Granada, Granada, Spain

influence of negative information. This family of problems is usually known as Influence Minimization (Khalil Elias et al. 2013; Luo et al. 2014; Xinjue et al. 2018; Qipeng et al. 2015).

A SN is usually modeled with a graph $G(V, E)$ where the set of nodes V represents the users and each relation between two users is modeled as a pair $(u, v) \in E$, with $u, v \in V$ indicating that user u is connected to or even can transmit information to user v. Kempe et al. (2003) originally formalized the influence model to analyze how the information is transmitted among the users of a SN. Given a SN with $|V| = n$ nodes where the edges (relational links) represent the spreading or propagation process on that network, the task is to choose a seed node set S of size $k < n$ with the aim of maximizing the number of nodes in the network that are influenced by the seed set S. This results in a combinatorial optimization problem known as the Social Network Influence Maximization problem (SNIMP).

The evaluation of the influence of a given seed set S requires the definition of an Influence Diffusion Model (IDM) (Kempe et al. 2015). This model is responsible for deciding which nodes are affected by the information received from their neighboring nodes in the SN. The most extended IDMs are: Independent Cascade Model (ICM), Weighted Cascade Model (WCM), and Linear Threshold Model (LTM). All of them are based on assigning an influence probability to each relational link in the SN. ICM, which is one of the most used IDMs, considers that the influence probability is the same for each link, and it is usually a small probability, being 1% a widely accepted value. On the contrary, WCM considers that the probability of a user v for being influenced by user u is proportional to the in-degree of user v, i.e., the number of users that can eventually influence user v. Therefore, the probability of influencing user v is defined as $1/d_{in}(v)$, where $d_{in}(v)$ is the in-degree of user v. The latter model, LTM, requires a specific activation weight for each link in the SN. Given those weights, a user will be influenced if and only if the sum of the weights of its neighbors if larger than or equal to a given threshold. In this paper we consider the ICM since it is one of the most popular IDMs in the literature. In particular, ICM views influence as being transmitted through the network in a tree-like fashion, where the seed nodes are the roots.

The SNIMP then involves finding a seed set S, with $|S| = k$ (where k is an input parameter), that maximizes the number of users influenced and, as a consequence, the spread of information through the network. In mathematical terms,

$$S^{\star} \leftarrow \arg\max_{S \in \mathcal{S}} ICM(G, S, p, ev)$$

where $\mathcal{S}$ is the set of all possible solutions (i.e., seed set setups), p is the probability of a user to be influenced, and ev is the number of iterations of the Monte Carlo simulation used to run the ICM (see Sect. 3.1).

The SNIMP was initially formulated in Richardson et al. (2003) and it was later proven to be $\mathcal{NP}$-hard for most IDMs in Kempe et al. (2015). As with many other $\mathcal{NP}$-hard problems, heuristic and metaheuristic algorithms, such as greedy and evolutionary algorithms, have been considered to solve the problem by effectively exploring the solution space, avoiding the analysis of every possible subset of nodes (Banerjee et al. 2020).

This work presents a novel metaheuristic approach for dealing with the SNIMP, allowing us to find high quality solutions in short computing time. Our main goal is to design an efficient algorithm where the use of Monte Carlo simulation required for the IDM application is minimized, thus increasing the efficiency of the algorithm. To do so, we make use of the Greedy Randomized Adaptive Search Procedure (GRASP) framework, characterized for its efficiency when designing solutions for $\mathcal{NP}$-hard combinatorial optimization problems. Our procedure is based on two stages. On the one hand, a greedy constructive procedure based on the 2-step neighborhood which is randomized to diversify the search with the aim of exploring a wider portion of the solution space. On the other hand, we introduce an efficient local search method. Specifically, it relies on an intelligent neighborhood exploration strategy for finding local optima with respect to the constructed solutions. The proposed procedure is validated over a set of 35 instances widely used in the context of social influence maximization, and benchmarked against both the classical methods based on greedy hill-climbing strategies (Goyal et al. 2011; Leskovec et al. 2007) and the state-of-the-art solution procedure for SNIMP based on particle swarm optimization (Banerjee et al. 2020). The results obtained clearly demonstrate the efficacy of the proposed methodology.

The remainder of the work is structured as follows. Section 2 reviews the related literature, detailing the different approaches followed to deal with different problems derived from the social influence maximization. The proposed approach is described in Sect. 3, where Sect. 3.1 introduces the influence diffusion model selected in this research, Sect. 3.2 presents the construction method proposed for providing high quality initial solutions, and Sect. 3.3 describes the local search proposed for finding local optimum with respect to a given neighborhood structure. Section 4 presents the experimental results considering a public dataset which has been previously used for this task, divided into preliminary experiments devoted to adjust the search parameters (Sect. 4.1), and final experiments to perform a competitive testing to evaluate the quality of the proposal (Sect. 4.2). Finally, Sect. 5 draws some conclusions derived from this research.

2 Literature review

In this section, we introduce some related work about SNIMP and IDM as well as a brief survey of existing methods for solving this problem, either based on heuristics or computational intelligence algorithms.

Richardson et al. (2003) initially formulated the problem of selecting target nodes in SNs. However, Kempe et al. (2003) were the first to solve the SNIMP formulating it as a discrete optimization problem. It has been shown that the SNIMP is $\mathcal{NP}$-hard (Kempe et al. 2015). Kempe et al. (2003) proposed a greedy hill-climbing algorithm with an approximation of $1 - 1/e - \epsilon$, being e the base of the natural logarithm and ϵ any positive real number. This result indicates that the algorithm is able to find solutions which are always within a factor of at least 63% of the optimal value under the three IDMs described in Sect. 1.

As a consequence of the computational effort required to evaluate the ICM, Kempe et al. (2003) also proposed several greedy heuristics based on SN analysis metrics such as degree and closeness centrality (Stanley and Katherine 1994). These methods only require one run of a Monte Carlo simulation to validate the single solution obtained using heuristic functions, thus increasing the efficiency at the cost of a loss of efficacy. When the considered metric is the degree of the node, the algorithm is called *high-degree heuristic*.

Several extensions of those first greedy algorithms were later proposed. In particular, Leskovec et al. (2007) introduced the Cost-Effective Lazy Forward (*CELF*) selection which exploited the submodularity property to significantly reduce the run time of the greedy hill-climbing algorithm, becoming over 700 times faster than the original procedure. The rationale is that the expansion of each node is computed a priori and it only needs to be recomputed for a few nodes. Meanwhile, Chen et al. (2009) used the concept of degree-discount heuristics to optimize the high-degree heuristic. The greedy selection function considers the redundancy between likely influenced nodes and does not include those reached by the already selected seed nodes to provide a better estimation of the total spread.

In Goyal et al. (2011) a new algorithm called *CELF++* was proposed with the aim of improving the efficiency of the original CELF. It leans on the property of submodularity of the spread function for IDM, avoiding unnecessary computations. According to the authors it is 35-55% faster than *CELF*.

A large number of works have been developed in the area since those first proposals (Şimşek and Kara 2018). Different kinds of heuristic and metaheuristic algorithms have been considered to solve the SNIMP. Table 1 summarizes the most recent approaches, including the algorithm type and the specific IDM considered.

Analyzing previous studies we can conclude that more complex metaheuristic approaches usually result in better solutions than simple greedy approaches. Yang and Weng (2012) proposed an Ant Colony Optimization (ACO) algorithm based on a parameterized probabilistic model to address the SNIMP. They used the degree centrality, distance centrality, and simulated influence methods for determining the heuristic values.

Meanwhile, the method based on Simulated Annealing (SA) presented in Li et al. (2017) applied two heuristic methods to accelerate the convergence process of SA, along with a new method of computing influence spread to speed up the proposed algorithm. In Bucur et al. (2017), an Evolutionary Multi-objective Optimization (EMO) algorithm (Lamont et al. 2007) was proposed for SNIMP, where the two considered objectives were maximizing the influence of a seed set and minimizing the number of nodes in the seed set jointly.

As said before, some heuristics have been proposed as time-saving solutions for greedy decisions: random, degree, and centrality (Kempe et al. 2003). The random heuristic selects nodes randomly, without considering node influence, to form the seed set in the network. Degree and centrality heuristics derive from the definition of the node influence in SN analysis (Stanley and Katherine 1994). Degree centrality heuristic usually produces less accurate results to the SNIMP. Alternatively, the high-degree heuristic targets the SNIMP by taking into account prior knowledge of the node's neighbors (Chen et al. 2009).

Recently, a complete survey on SNIMP has been presented in Banerjee et al. (2020). In that work, authors experimentally compare the results obtained by the most recent algorithms. The survey concludes that the particle swarm optimization approach by Gong et al. (2016) obtains the best results in the literature, so we will use that algorithm to benchmark our proposal. This survey has become one of the most relevant research in the area of influence maximization problems.

The aforementioned metaheuristic methods are able to obtain good results but usually require large computational times. Our proposal considers a method combining the use of heuristic functions and Monte Carlo simulations for limiting the number of ICM evaluations that tries to balance the quality of the obtained solutions and the required computing time. With the aim of reducing the required computing time, we consider a GRASP, combining good heuristic solutions that are quickly generated and an efficient local search method which minimizes the number of Monte Carlo evaluations to further improve the initial solution. The use of GRASP in Graph Theory and Network Science has led to several successful research in the last years (Pérez-Peló et al. 2019, 2020; Gil-Borrás et al. 2020).

Table 1 Summary of the literature published for the SNIMP in the last 20 years

References	Algorithm	IDM
Kempe et al. (2003)	Greedy	ICM-LTM
Chen et al. (2009)	Greedy	ICM
Lappas et al. (2010)	Dynamic Programming	ICM
Goyal et al. (2011)	Greedy	ICM
Yang and Weng (2012)	Ant Colony	ICM
Nguyen and Zheng (2013)	Greedy	ICM
Jiaguo et al. (2014)	Greedy	ICM
Li et al. (2014)	Greedy	Polarity-Related-ICM
Liu et al. (2014)	Greedy	Latency Aware-ICM
Lee and Chung (2015)	Greedy	ICM
Song et al. (2015)	Greedy	ICM
Bucur and Iacca (2016)	Genetic	ICM
Gong et al. (2016)	Particle Swarm	ICM
Ok et al. (2016)	Greedy	Independent Poisson Clock
Tong et al. (2016)	Greedy	ICM
Zhang et al. (2016)	Greedy	LTM-ICM
Bucur et al. (2017)	Multi-objective evolutionary	ICM
Li et al. (2017)	Simulated Annealing	Polarity-Related-ICM
Peng et al. (2017)	Greedy	SI
Tong et al. (2017)	Greedy	ICM
Zhang et al. (2017)	Genetic	LTM
Bucur et al. (2018a)	Multi-objective evolutionary	ICM
Bucur et al. (2018)	Multi-objective evolutionary	ICM
Samadi et al. (2018)	Mixed Integer Programming	Partial Parallel Cascade
Liu et al. (2019)	Evolutionary	SIR epidemic spreading model
Salavati and Abdollahpouri (2019)	Ant Colony	ICM
Robles et al. (2020)	Multi-objective evolutionary	Viral marketing model

3 Algorithmic approach

The method presented in this work aims at finding high quality solutions in reasonable computing time. In order to do so, we propose a metaheuristic algorithm. Metaheuristics are one of the most extended techniques for solving hard optimization problems. They are able to guide the search in order to escape from local optima, with the goal of finding high quality solutions in a reduced computing time.

GRASP is a metaheuristic framework developed in the late 1980s Feo and Resende Mauricio (1989) and formally introduced in Feo and Resende Mauricio (1994). We refer the reader to Resende Mauricio et al. (2010), Resende Mauricio and Celso (2013) for a complete survey of the last advances in this methodology. GRASP is a multi-start framework divided into two distinct stages. The first one is a greedy, randomized, and adaptive construction of a solution. The second stage applies an improvement method to obtain a local optimum with respect to a certain neighborhood, starting from the constructed solution. This methodology is able to find a trade-off between the diversification of the randomized construction phase and the intensification of the local search procedure, allowing the algorithm to escape from local optima and perform a wider exploration of the search space. These two phases are repeated until a termination criterion is met, returning the best solution found during the search.

3.1 Influence diffusion model

Before defining the algorithmic proposal, it is necessary to provide a formal definition of the IDM considered in this work, which is the ICM introduced in Sect. 1. Due to the probabilistic nature of ICM, the most extended way of evaluating the spread is by conducting a Monte Carlo simulation. However, even a single iteration of the simulation in ICM is rather time-consuming. Algorithm 1 shows the pseudocode of the Monte Carlo simulation to evaluate the spread of information through a SN named G given a seed set S. Specifically, it receives four input parameters: the graph which models the SN, a tentative solution, the probability of a user to be influenced, and the number of iterations of the corresponding Monte Carlo simulation.

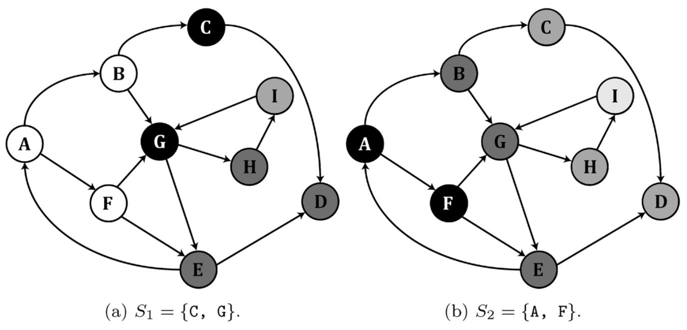


Fig. 1 SN with 9 nodes and 13 edges. Two feasible solutions S_1 and S_2 are represented, each of them resulting in a different set of influenced users

Algorithm 1 $ICM(G = (V, E), S, p, ev)$

```
1: $I \leftarrow \emptyset$
2: for $i \in 1 \ldots ev$ do
3:     $A^\star \leftarrow S$
4:     $A \leftarrow S$
5:     while $A \neq \emptyset$ do
6:         $B \leftarrow \emptyset$
7:         for $v \in A$ do
8:             for $(u, v) \in E$ do
9:                 if $rnd(0, 1) \leq p$ then
10:                    $B \leftarrow B \cup \{u\}$
11:                end if
12:            end for
13:        end for
14:        $A^\star \leftarrow A^\star \cup B$
15:        $A \leftarrow B$
16:    end while
17:    $I \leftarrow I + |A^\star|$
18: end for
19: return $I/ev$
```

The algorithm starts by initializing the set which stores the number of infected users (step 1). It then performs a number of predefined iterations *ev* (steps $2-18$), finding in each iteration which are the influenced nodes by the given seed set *S*. Initially, the set of nodes $A^\star$ reached by the initial seed set, *S*, is actually the seed set (step 3). Then, the method iterates until no new nodes are influenced (steps 5–16). In each iteration of the inner for-loop, the neighbors of each node reached in the previous one are traversed (steps 8–12). For each neighbor, a random number is generated. If this number is smaller than the probability of infection *p*, then the neighboring node becomes infected (steps 9–11). At the end, the set of infected nodes is updated (step 14) as well as the nodes infected in the previous iteration (step 15). Finally, the algorithm returns the mean number of infected nodes among all the simulations performed, i.e., *I* divided by *ev* (step 19). Notice that this value is considered as the objective function to be optimized when solving the SNIMP. That is, the seed set maximizing the *spread value* over the network would compose the optimal solution to the problem. It is worth mentioning that, as infection is a stochastic process, the ICM must be run several times (*ev* in our case) to achieve an appropriate estimation, thus resulting in a Monte Carlo simulation.

In order to illustrate the evaluation of a solution under the ICM, Fig. 1 shows an example of a SN with 9 nodes and 13 directed edges, where each pair (u, v) denotes that the user *v* may be influenced by *u*. Information represents anything that can be passed across connected peers within a network. The influence level given by a node is determined by the adoption or propagation process. Let us consider $k = 2$ for this example graph.

Figure 1a shows solution S_1 where the seed set is conformed with nodes `C` and `G`. Without loss of generality we will assume, for this example, that $p = 1$ (i.e., all the nodes are always infected by their neighbors). Simulating the diffusion model we can see how nodes `D`, `E`, and `H` are directly influenced by the seed set. After that, node `H` influences node `I`. Different levels of influence are represented by a gray gradient from black to white. Therefore, if we consider a single evaluation of the Monte Carlo simulation, the objective function value of S_1 is $ICM(G, S_1, p, 1) = 6$. Figure 1b depicts solution $S_2 = \{\mathtt{A}, \mathtt{F}\}$. Similarly to Fig. 1a, a gray gradient indicates the process of influence over the network, resulting in an objective function value of $ICM(G, S_2, p, 1) = 9$, since all nodes are influenced. Notice that, following this evaluation, S_2 is better than S_1. However, it is important to remark that a single iteration for the Monte Carlo simulation is not significant, so we have decided to perform ev =100 evaluations of the simulation as it is customary in the literature for the SNIMP (Bucur and Iacca 2016; Bucur et al. 2018). Additionally, we set $p = 0.01$ as stated in previous works Bucur and Iacca 2016; Kempe et al. 2003; Gong et al. 2016. Then, for the simplicity, we refer to $ICM(G, S, 0.01, 100)$ as $ICM(G, S)$ in the remaining of the paper.

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

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

3.2 Construction phase

The construction phase of GRASP is designed to generate an initial solution and it is usually guided by a greedy selection function which helps the constructive method to select the next elements to be included in the partial solution (see Algorithm 2).

In order to favor diversification, the first node to be included in the solution S is selected at random from the set of SN nodes V (step 1). The candidate list CL is created with all the nodes but v (step 2) and the node v is included as the only node in the initial solution S (step 3). Then, the constructive method iteratively adds new elements to the solution until it becomes feasible, being composed of k nodes (steps 4–12). In each iteration, the minimum and maximum value of the greedy heuristic function is evaluated (steps 5–6). Then, a threshold μ is calculated (step 7), which is required for creating the Restricted Candidate List (RCL) with the most promising nodes (step 8). This threshold directly depends on the value of the input parameter α, which is in the range [0, 1]. Notice that this parameter indicates the greediness or randomness of the constructive procedure. On the one hand, if $\alpha = 0$, then the threshold is evaluated as $g_{\max}$, becoming a totally greedy algorithm (i.e., the RCL only includes the best choice in each iteration). On the other hand, if $\alpha = 1$ then $\mu = g_{\min}$, resulting in a completely random method (i.e., the RCL includes every feasible choice in each iteration). Since this parameter is tuned experimentally, we refer the reader to Sect. 4 to analyze the experiments performed to select the best value for α. Finally, the next node is selected at random from the RCL (step 9), including it in the solution (step 10) and updating the CL (step 11). The method ends when k elements are included in the seed set, returning the constructed solution S (step 13).

The greedy heuristic function g used in steps 5-6 is one of the key features when designing a constructive procedure in the context of GRASP. In this work we propose two different greedy functions to generate initial solutions. The first one, named g_{ne}, is a heuristic based on the first and second degree neighbors of a given node (usually known as 2-step in SN analysis Stanley and Katherine 1994). Given a node u, its out-degree is defined as $d^+_u = |N^+_u = |$, where $N^+_u = \{w \in V : (u, w) \in E\}$ is the set of adjacent nodes to u.

Thus, the first greedy function calculates the sum of the out-degree of u plus the out-degree of its neighbors. This heuristic function is based on the two-hop area (Sen et al. 2014) and three-degree theory (Christakis and Fowler 2009) hypothesis, indicating that there exists an intrinsic decay when increasing the maximum neighborhood level. More formally,

$$g_{ne}(u) = d^+_u + \sum_{v \in N^+_u} d^+_v$$

If we analyze this metric over the graph depicted in Fig. 1, the value of g_{ne} over vertex $\mathtt{F}$, for instance, is evaluated as $g_{ne}(\mathtt{F}) = d^+_{\mathtt{F}} + d^+_{\mathtt{E}} + d^+_{\mathtt{G}} = 2 + 2 + 2 = 6$. In order to reduce the computational effort of updating the value of this greedy function for each node, the method updates the value just for the affected nodes, which are those nodes to which the selected one is directly connected. For instance, in case of selecting node $\mathtt{F}$, only nodes $\mathtt{E}$ and $\mathtt{G}$ must be updated, by subtracting the out-degree of the selected node $\mathtt{F}$. This efficient update mechanism allows the algorithm to minimize the relevance of those vertices that are directly influenced by one of the selected nodes.

The second greedy function, named g_{cc}, considers the nodes clustering coefficient as a heuristic value. It estimates the likelihood that nodes in a graph tend to cluster together. In mathematical terms, the clustering coefficient is defined as (in a directed network):

$$g_{cc}(u) = \frac{\left|\{(v,w) \in E \,:\, v,w \in N^+_u\}\right|}{d^+_u \cdot (d^+_u - 1)}$$

It is worth mentioning that, to speed up this computation, every value is pre-calculated before the execution of the algorithm. We refer the reader to Watts and Strogatz (1998) for a detailed description of the implementation of this metric.

Let us illustrate how we can evaluate this value with an example. Considering again the SN depicted in Fig. 1, the evaluation of the clustering coefficient of node $\mathtt{F}$ is performed as $g_{cc} = \dfrac{|(\mathtt{G},\mathtt{E})|}{d^+_{\mathtt{F}} \cdot (d^+_{\mathtt{F}} - 1)} = \dfrac{1}{2} = 0.5$.

Given g_{ne} and g_{cc} as greedy functions, we propose two different constructive procedures C_{ne} and C_{cc}, each one based on a different greedy function. The impact and influence of each constructive procedure in the generated solutions will be deeply analyzed in Sect. 4.

3.3 Local search phase

The second phase of GRASP involves improving the solution generated by the constructive procedure in each iteration with the aim of reaching a local (ideally global) optimum. In the context of GRASP, this phase can be accomplished by using simple local search procedures or more complex heuristics like Tabu Search or even a hybrid metaheuristic (Martí et al. 2018). The high complexity of the problem under consideration has led us to propose a low time consuming local search procedure.

Before defining a local search method, it is necessary to introduce the neighborhood to be explored. The neighborhood of a solution S is defined as the set of solutions that can be reached by performing a single move over S. In the context of SNIMP, we propose a swap move $Swap(S, u, v)$ where node u is removed from the seed set, being replaced by v, with $u \in S$ and $v \notin S$. This swap move is formally defined as:

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

Thus, the neighborhood N_s of a given solution S consist of the set of solutions that can be reached from S by performing a single swap move. More formally,

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

The next step to define the proposed local search procedure consists in indicating the way in which the neighborhood is explored. Even considering an efficient implementation of the objective function evaluation, the vast size of the resulting neighborhood, $k \cdot (n - k)$, makes the complete exploration of the neighborhood not suitable for the SNIMP. Therefore, an intelligent neighborhood exploration strategy is presented with the aim of reducing the number of solutions explored within each neighborhood. This reduction in the size of the search space is performed by exploring just a small fraction, δ, of the available nodes for the swap node.

Since we are limiting the number of nodes considered in the local search approach, it is recommended to select the most promising ones to be involved in the swap moves. In the context of SNIMP, a node with a large out-degree can eventually influence a larger amount of nodes. Following this idea, we sort the candidate nodes to be included in the seed set in descending order with respect to their out-degree, while the candidate nodes to be removed from the seed set are sorted in ascending order with respect to their out-degree. Sect. 4 details the impact of the number of selected nodes, determined by δ, in the results obtained with the local search procedure.

Finally, we need to indicate which moves will be accepted during the search. In particular, two strategies are usually considered: Best Improvement and First Improvement

Table 2 Nodes and edges of the instances used in this work

Instance	Nodes	Edges	References
WikiVote	7115	103689	Bucur and Iacca (2016), Lawyer (2015)
ca-AstroPh	18772	198110	Liu et al. (2019), Lawyer (2015)
ca-CondMat	23133	93497	Liu et al. (2019), Lawyer (2015)
cit-HepPh	34546	421578	Liu et al. (2019), Lawyer (2015)
email-Enron	36692	183831	Liu et al. (2019)
p2p-Gnutella31	62586	147892	Liu et al. (2019), Lawyer (2015)
email-EuAll	265214	420045	Liu et al. (2019), Lawyer (2015)

Table 3 Results of the constructive procedure when generating 100 solutions, considering different α values for both heuristic functions

Heuristic	α	Avg.	Time (s)	Dev (%)	#Best
C_{ne}	0.25	236.62	**6.23**	15.27	0
	0.50	258.57	7.07	2.17	1
	0.75	262.11	7.15	0.31	6
	RND	**262.38**	6.55	**0.11**	**8**
C_{cc}	0.25	96.49	77.21	63.24	0
	0.50	83.06	77.21	65.41	0
	0.75	93.18	78.96	64.57	0
	RND	95.04	77.21	63.24	0

Best results are highlighted with bold font

Table 4 Influence of the number of nodes δ explored in each iteration of the local search procedure when coupled with the constructive procedure

δ	Avg.	Time (s)	Dev. (%)	#Best
10	272.28	47.28	0.89	5
20	272.46	**43.20**	0.56	**8**
30	**274.42**	101.31	**0.51**	4
40	273.72	69.17	0.54	4

Best results are highlighted with bold font

(Hansen and Mladenović 2006). On the one hand, the former explores, in each iteration, the complete neighborhood, moving to the best solution in it. On the other hand, the latter moves to the first solution that achieves an improvement in the objective function value, without requiring to explore the complete neighborhood. Due to the computational effort required to evaluate a solution for the SNIMP, we propose a First Improvement approach, which does not need to explore all the solutions in the neighborhood, thus reducing the number of objective function evaluations required and consequently the overall run time.

Notice that the objective function evaluation consists in a Monte Carlo simulation, being the most computationally demanding part of the proposed algorithm. For this reason, the proposed local search aims to limit the number of required simulations, thus leading to a more efficient procedure.

4 Computational experiments and analysis of results

The aim of this section is to describe the computational experiments designed to evaluate the performance of the proposed algorithms and to analyze the obtained results. All the experiments have been performed in an Intel Core i7 (2.6 GHz) with 8GB RAM and the algorithms were implemented using Java 9. The source code has also been made publicly available.[1]

The set of SNs considered in this paper have been entirely obtained from the most relevant works found in the literature, in order to provide a fair comparison among the analyzed algorithms. All of them are publicly available in Stanford Network Analysis Project (SNAP): https://snap.stanford.edu/. Relevant information about these instances is collected in Table 2, where some papers in which each instance has been used are included.

First of all, it is important to indicate which values are used for the ICM algorithm with the corresponding Monte Carlo simulation. In all the experiments, as stated in Sect. 3.2, 100 simulations of the ICM are performed with a probability of influence of 0.01. These parameter values are the most extended ones in the related literature. The number of seed nodes k to conform a solution is selected in the range $k = \{10, 20, 30, 40, 50\}$ as stated in Bucur and Iacca (2016), Kempe et al. (2003), Salavati and Abdollahpouri (2019), thus obtaining $7 \cdot 5 = 35$ different problem instances (resulting from the combination of 7 networks and 5 seed set sizes).

The experiments are divided into two parts: preliminary and final experimentation. The former (Sect. 4.1) refers to those experiments performed to select the best parameters to set up our algorithm, while the latter (Sect. 4.2) validates the best configuration, comparing it with the best methods found in the state of the art.

All the experiments developed report the following metrics: Avg., the average objective function value (i.e., the number of influenced nodes, in average, after 100 simulations); Time (s), the average computing time required by the algorithm in seconds; Dev(%), the average deviation with respect to the best solution overall found in the experiment; and #Best, the number of times that the algorithm matches the best solution.

4.1 Preliminary experimentation to setup the final GRASP method

The preliminary experimentation has been performed with a small set of 10 instances out of 35 to avoid overfitting. This selection comprises, approximately 30% of the global set and it provides enough variability in instances and values of k.

The first preliminary experiment is designed to find the best value for the α parameter in each of the proposed constructive procedures (see Sect. 3.2). Table 3 shows the detailed results for each constructive procedure when considering $\alpha = \{0.25, 0.50, 0.75, RND\}$, where *RND* indicates

[1] https://grafo.etsii.urjc.es/SNIMP.

Table 5 Competitive testing of the proposed GRASP algorithm with respect to the best algorithms found in the literature: CELF, CELF++, and PSO

k		CELF		CELF++		PSO		GRASP	
	Name	Avg.	Time (s)	Avg.	Time (s)	Avg.	Time (s)	Avg.	Time (s)
10	CA-AstroPh	157.60	**2.51**	171.81	9.40	169.85	232.40	**187.47**	8.28
	CA-CondMat	35.73	**0.67**	35.73	2.15	33.40	4.60	**36.15**	2.56
	Cit-HepPh	46.63	**1.16**	46.63	3.29	35.27	1.71	**47.20**	4.20
	Email-Enron	383.95	**25.23**	469.63	87.68	465.24	1756.84	**489.67**	41.41
	Email-EuAll	132.96	**6.03**	130.28	307.98	107.41	37.42	**144.57**	24.42
	Wiki-Vote	108.50	**0.39**	108.50	1.00	92.16	16.40	**109.10**	6.32
	p2p-Gnutella31	16.24	**1.46**	16.23	7.83	13.38	0.95	**16.27**	1.63
20	CA-AstroPh	222.63	**2.69**	234.36	9.76	222.92	889.79	**259.25**	18.53
	CA-CondMat	59.72	**0.66**	59.87	2.13	45.46	8.67	**61.05**	6.00
	Cit-HepPh	81.75	**1.11**	81.75	3.25	68.51	2.58	**82.11**	18.97
	Email-Enron	451.24	**25.71**	547.96	88.47	544.57	4394.46	**589.65**	74.23
	Email-EuAll	214.66	**5.68**	214.54	303.01	162.32	99.98	**224.10**	28.88
	Wiki-Vote	162.49	**0.49**	162.49	1.45	141.66	41.44	**165.32**	26.03
	p2p-Gnutella31	30.82	**1.30**	30.86	7.43	24.69	0.99	**30.92**	3.80
30	CA-AstroPh	266.77	**2.85**	276.69	10.48	259.90	1005.17	**312.68**	51.32
	CA-CondMat	80.87	**0.70**	82.18	2.30	66.27	11.09	**82.54**	14.11
	Cit-HepPh	113.39	**1.16**	113.39	3.45	86.22	3.69	**113.63**	42.30
	Email-Enron	501.78	**25.49**	608.63	88.62	553.25	7594.67	**652.48**	140.71
	Email-EuAll	277.40	**5.86**	275.36	298.66	212.84	183.58	**281.30**	59.58
	Wiki-Vote	208.18	**0.64**	208.18	2.03	150.40	97.75	**214.97**	80.83
	p2p-Gnutella31	44.75	**1.24**	**44.81**	7.42	35.30	1.34	**44.81**	6.08
40	CA-AstroPh	319.52	**3.11**	302.86	11.58	288.92	1492.82	**360.34**	66.97
	CA-CondMat	100.96	**0.76**	101.80	2.54	75.61	17.40	**104.38**	16.37
	Cit-HepPh	140.63	**1.27**	140.63	3.81	113.46	4.94	**141.20**	58.03
	Email-Enron	549.64	**25.95**	658.38	92.09	634.58	9032.87	**705.03**	216.65
	Email-EuAll	323.85	**6.17**	312.47	302.47	258.46	230.12	**337.39**	165.23
	Wiki-Vote	246.02	**0.83**	246.02	2.83	182.88	115.05	**252.15**	34.60
	p2p-Gnutella31	58.26	**1.28**	58.22	7.48	51.26	1.85	**58.37**	12.69
50	CA-AstroPh	361.51	**3.50**	338.28	13.11	340.54	2267.98	**399.92**	132.35
	CA-CondMat	119.29	**0.86**	120.72	2.87	106.10	10.85	**124.57**	26.51
	Cit-HepPh	165.47	**1.38**	165.47	4.26	126.77	6.35	**166.77**	65.20
	Email-Enron	597.26	**27.02**	680.29	96.71	662.67	10063.47	**744.38**	157.26
	Email-EuAll	361.51	**6.68**	357.43	304.03	258.15	321.81	**375.03**	161.59
	Wiki-Vote	277.65	**1.09**	277.65	3.76	188.82	181.07	**287.66**	86.39
	p2p-Gnutella31	71.80	**1.34**	71.90	7.76	64.63	2.74	**72.08**	17.42
	G.Avg.	208.33	**5.55**	221.49	60.09	195.54	1146.71	**236.37**	39.04

Best results are highlighted with bold font

that the value of α is selected at random in each construction (thus allowing the algorithm for a higher and balanced search space diversification). The method is executed 100 independent times per instance, returning the best constructed solution.

As it can be drawn from the table, the best results are consistently provided by the greedy function based on the two-step neighbors, C_{ne}, both in quality and run time. In particular, the best results are obtained when considering $\alpha = RND$, with 8 best solutions and 0.11% of average deviation. The small deviation value indicates that, even in the cases in which it is not able to reach the best solution, it remains really close to it. Besides, it is the second best choice in terms of run time, with a slight difference with respect to the best option. Therefore, we select C_{ne} as the best constructive procedure with $\alpha = RND$.

The second preliminary experiment is devoted to analyze the influence of the number of explored nodes in the neighborhood for the local search phase. In particular, we have tested $\delta = \{10, 20, 30, 40\}$, being δ the number of nodes explored in each local search iteration. Table 4 shows the influence that the number of nodes explored δ has in the performance of the local search procedure. Notice that an independent local search is applied to each one of the 100 constructed solutions, returning the best solution found overall.

The obtained results show that the best δ value is 30, reaching the smallest deviation and the largest number of best solutions found, although in this case the results are more similar in every metric. However, it is more computationally demanding than the remaining values, being two or even three times slower than the other δ values. Furthermore, $\delta = 20$ is the quickest variant (almost 2.5 times faster than $\delta = 30$) and it presents a very good performance: a promising average objective function value and average deviation with respect to the best value (272.46 versus 274.42 and 0.56% versus 0.51%, respectively). For this reason, we select $\delta = 20$ as our design choice for the final algorithm.

It is important to remark the relevance of the introduced intelligent neighborhood exploration strategy in the performance of the whole algorithm. Specifically, the best identified local search method (with $\delta = 20$) spends 26.03 seconds on solving the smallest instance (WikiVote), obtaining an objective function value of 165.32. If we do not consider the δ parameter and execute an exhaustive local search, it needs 15793.09 seconds (i.e., more than 600 times longer) to find the local optima with a value of 165.59. We do not extend this experiment for the remaining instances since the computing time is unacceptable.

Having performed the preliminary experiments, we can conclude that the best results are obtained with the configuration greedy function=g_{ne}, $\alpha = RND$, and $\delta = 20$. These parameter values will be used to set up the final version of the algorithm.

4.2 Final experimentation to benchmark the final GRASP method with state-of-the-art results

In order to analyze the quality of the proposed algorithm, we perform a competitive testing with the best methods found in the state of the art by considering the complete set of 35 instances. In this experiment, three additional algorithms are considered: CELF (Leskovec et al. 2007), the well-known greedy hill-climbing algorithm; CELF++ (Goyal et al. 2011), the improved version of CELF (Leskovec et al. 2007); and PSO (Gong et al. 2016), the particle swarm optimization algorithm which is considered the state of the art for social influence analysis according to the recent experimental study developed in Banerjee et al. (2020). Table 5 collects the final results obtained in this competitive testing, where we report for each method and instance the value of the objective function (Avg.) and the associated computing time in seconds (Time(s)). Notice that Avg. is not an integer value since it is the average value of the 100 runs of the ICM in the Montecarlo simulation. We have also included a final row in this table (G.Avg.) with the average values of the objective function and Time (s), computed across the set of 35 instances.

We would like to first highlight the results obtained with PSO, since it is ranking the last one even being considered the state of the art for this problem. The rationale behind this is that the original work (Gong et al. 2016) considers small size instances (from 410 to 15233 nodes) and the quality of the solutions provided by PSO deteriorates when the instance size increases, as it can be derived from Table 5. Meanwhile, CELF and its improved version CELF++ are able to reach better solutions, still being competitive with the state-of-the-art algorithms. However, only CELF++ is able to match the best-known solution in 1 instance (out of 35). Finally, the best results are obtained with the proposed GRASP algorithm, which is able to reach the best solution found in all the 35 instances. Furthermore, the computing time is smaller than the second best algorithm, CELF++ (39.04 versus 60.09 seconds on average).

As can be observed in Table 5, CELF is able to provide high quality solutions in small computing time. In order to further analyze these results, we conduct an additional experiment to compare our constructive procedure (i.e., only using the first stage of the GRASP procedure and not considering the application of the local search) with CELF. In this case, the average objective function for C_{ne} is 227.83, which compares favorably to the result achieved with CELF, which is 208.33. In both cases, the computing time is approximately 6 seconds.

Analyzing the computing time required for each algorithm, we can clearly see that CELF and CELF++, as completely greedy approaches, are not really affected by increasing the size of the seed set. On the contrary, the computing time required for PSO and GRASP is affected by the size of the seed set since larger k-values lead the local improvement method to perform a larger number of iterations. However, if we take a closer vision of the results obtained with GRASP, we can conclude that this increase in the number of iterations and, therefore, in the computing time allows the algorithm to reach better solutions. In the case of PSO, the increase of computing time is even much more noticeable but it does not usually result in better solutions, suggesting that the PSO algorithm is particularly suitable for solving small size instances.

In order to validate these results, we have conducted a non-parametric Friedman test for ranking all the compared algorithms. The p-value obtained, smaller than 0.0005, confirms that there are statistically significant differences among the algorithms. The algorithms sorted by ranking are GRASP

(1.00), CELF++ (2.44), CELF (2.79), and PSO (3.77). We finally perform the well-known non-parametric Wilcoxon statistical test for pairwise comparisons, which answers the question: do the solutions generated by both algorithms represent two different populations? The resulting *p*-value smaller than 0.0005 when comparing GRASP with each other algorithm confirms the superiority of the proposed GRASP algorithm. Therefore, GRASP emerges as one of the most competitive algorithms for the SNIMP, being able to reach high quality solutions in small computing time.

5 Conclusions

In this paper a quick GRASP algorithm for solving the SNIMP has been presented. Two constructive procedures are proposed, with the one based on the two-step neighborhood being more competitive one than that based on the clustering coefficient. Furthermore, the idea of using local information allows the algorithm to construct a complete solution in a small computing time. Then, a local search based on swap moves is presented. Since an exhaustive exploration of the search space is not suitable for this problem, we propose an intelligent neighborhood exploration strategy which limits the region of the search space to be explored, focusing on the most promising areas. This rationale leads us to provide high quality solutions in reasonable computing time, even for the largest instances derived from real-world SNs commonly considered in the SNIMP area. Since the intelligent neighborhood exploration strategy is parameterized, if the computing time is not a relevant factor, the region explored can be easily extended to find better solutions, thus increasing the required computational effort. This fact makes the proposed GRASP algorithm highly scalable. The results obtained are supported by Friedman test and then the pairwise Wilcoxon test, confirming the superiority of the proposal with respect to the classical and state-of-the-art solution procedures in the area.

In our future work, we plan to study the adaptation of techniques developed in this work to influence minimization problems. This adaptation can be useful for minimizing the impact of fake news and monitor those users which can eventually transmit misinformation through the network (Wei-Neng et al. 2019; Wang et al. 2017).

Acknowledgements This research was funded by "Ministerio de Ciencia, Innovación y Universidades" under grant ref. PGC2018-095322-B-C22, "Comunidad de Madrid" and "Fondos Estructurales" of European Union with grant refs. S2018/TCS-4566, Y2018/EMT-5062. It is also supported by the Spanish Agencia Estatal de Investigación, the Andalusian Government, the University of Granada, and European Regional Development Funds (ERDF) under grants EXASOCO (PGC2018-101216-B-I00) and AIMAR (A-TIC-284-UGR18).

Funding Funding for open access charge: Universidad de Granada / CBUA.

Declaration

Conflict of interest The authors declare that they have no conflict of interest.

References

Banerjee S, Jenamani M, Pratihar DK (2020) A survey on influence maximization in a social network. Knowl Inf Syst 62:3417–3455. https://doi.org/10.1007/s10115-020-01461-4

Barabási, Albert-László, Pósfai Márton (2016) Network science. Cambridge University Press, Cambridge. ISBN: 9781107076266 1107076269, http://barabasi.com/networksciencebook/

Bucur D, Iacca G (2016) Infuence maximization in social networks with genetic algorithms. In: Applications of evolutionary computation. Springer International Publishing, Berlin, pp 379–392. https://doi.org/10.1007/978-3-319-31204-0_25

Bucur D, Giovanni I, Andrea M, Giovanni S, Al-berto T (2017) Multi-objective evolutionary algorithms for influence maximization in social networks, pp 221–233. https://doi.org/10.1007/978-3-319-55849-3_15

Bucur D, Giovanni I, Andrea M, Giovanni S, Al-berto T (2018a) Evaluating surrogate models for multi-objective in-fluence maximization in social networks. In: Proceedings of the genetic and evolutionary computation conference companion on GECCO 18. ACM Press. https://doi.org/10.1145/3205651.3208238

Bucur D, Giovanni I, Andrea M, Giovanni S, Al-berto T (2018b) Improving multi-objective evolutionary in fluence maximization in social networks. In: Applications of evolutionary computation. Springer International Publishing, Berlin, pp 117–124. https://doi.org/10.1007/978-3-319-77538-8_9

Chen W, Yajun W, Siyu Y (2009) Efficient influence maximization in social networks. In: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining-KDD 09. ACM Press. https://doi.org/10.1145/1557019.1557047

Christakis NA, Fowler JH (2009) Connected: the surprising power of our social networks and how they shape our lives. Little, Brown Spark

D'angelo A, Aditya A, Kang-Xing J, Yun-Fang J, Levy K, Oleksandr M, Yishan W (2009) Targeting advertisements in a social network. US Patent App. 12/195321

Feo T, Resende Mauricio AGC (1989) A probabilistic heuristic for a computationally difficult set covering problem. Oper Res Lett 8(2):67–71. https://doi.org/10.1016/0167-6377(89)90002-3

Feo TA, Resende Mauricio GC, Stuart HS (1994) A greedy randomized adaptive search procedure for maximum independent set. Oper Res 42(5):860–878. https://doi.org/10.1287/opre.42.5.860

Gil-Borrás SG, Pardo E, Antonio A-A, Abraham D (2020) GRASP with Variable Neighborhood Descent for the on- line order batching problem. J Glob Optim 2020:1–31

Gong M, Chao S, Chao D, Lijia M, Bo S (2016) An efficient memetic algorithm for influence maximization in social networks. IEEE Comput Intell Mag 11(3):22–33. https://doi.org/10.1109/mci.2016.2572538

Goyal A, Wei L, Laks VSL (2011) CELF++: optimizing the greedy algorithm for influence maximization in social networks. In: Proceedings of the 20th international conference companion on World wide web WWW 11. ACM Press. https://doi.org/10.1145/1963192.1963217

Hansen P, Mladenović N (2006) First vs. best improvement: an empirical study. In: Discrete Applied Mathematics 154.5. IV ALIO/EURO workshop on applied combinatorial optimization, pp 802–817. Issn: 0166-218X. https://doi.org/10.1016/j.dam.2005.05.020, http://www.sciencedirect.com/science/article/pii/S0166218X05003070

Jiaguo L, Guo J, Yang Z, Zhang W, Jocshi A (2014) Improved algorithms of CELF and CELF++ for influence maximization. J Eng Sci Technol Rev 7(3):32–38. https://doi.org/10.25103/jestr.073.05

Kempe D, Jon K, Éva T (2003) Maximizing the spread of influence through a social network. In: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining KDD 03. ACM Press. https://doi.org/10.1145/956750.956769

Kempe D, Jon K, Eva T (2015) Maximizing the spread of influence through a social network. Theory Comput 11(1):105–147. https://doi.org/10.4086/toc.2015.v011a004

Khalil EB, Dilkina B, Song L (2013)'CuttingEdge: influence minimization in networks. In: Workshop on frontiers of network analysis: methods, models, and applications at NIPS. url: files/papers/CuttingEdge.pdf

Klovdahl AS (1985) Social networks and the spread of infectious diseases: the AIDS example. Soc Sci Med 21(11):1203–1216. https://doi.org/10.1016/0277-9536(85)90269-2

Lamont CCC, Gary B, van Veldhuizen DA (2007) Evolutionary algorithms for solving multi-objective problems. Springer, US. https://doi.org/10.1007/978-0-387-36797-2

Lappas T, Evimaria T, Dimitrios G, Heikki M (2010) Finding effectors in social networks. In: Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining KDD 10. ACM Press. https://doi.org/10.1145/1835804.1835937

Lawyer G (2015) Understanding the influence of all nodes in a network. Sci Rep 5:1. https://doi.org/10.1038/srep08665

Lee J-R, Chung C-W (2015) A Query approach for influence maximization on specific users in social networks. IEEE Trans Knowl Data Eng 27(2):340–353. https://doi.org/10.1109/tkde.2014.2330833

Leskovec J, Andreas K, Carlos G, Christos F, Jeanne Van B, Natalie G (2007) Cost-effective outbreak detection in networks. Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining KDD 07. ACM Press. https://doi.org/10.1145/1281192.1281239

Li D, Zhi-Ming X, Nilanjan C, Anika G, Katia S, Sheng L (2014) Polarity related influence maximization in signed social networks. PLoS ONE 9(7):e102199. https://doi.org/10.1371/journal.pone.0102199 ((**Ed. by Sergio Gómez**))

Li D, Wang C, Zhang S, Zhou G, Chu D, Chong W (2017) Positive influence maximization in signed social networks based on simulated annealing. Neurocomputing 260:69–78. https://doi.org/10.1016/j.neucom.2017.03.003

Liu B, Gao C, Yifeng Z, Dong X, Yeow MC (2014) Influence spreading path and its application to the time constrained social influence maximization problem and beyond. IEEE Trans Knowl Data Eng 26(8):1904–1917. https://doi.org/10.1109/tkde.2013.106

Liu Y, Xi W, Jurgen K (2019) Framework of evolutionary algorithm for investigation of influential nodes in complex networks. IEEE Trans Evol Comput 23(6):1049–1063. https://doi.org/10.1109/tevc.2019.2901012

Luo C, Kainan C, Xiaolong Z, Zeng D (2014) Time critical disinformation influence minimization in online social networks. In: 2014 IEEE joint intelligence and security informatics conference, pp 68–74

Martí R, Anna M-G, Jesús S-O, Abraham D (2018) Tabu search for the dynamic Bipartite Drawing Problem. Comput Oper Res 91:1–12 ((**issn: 0305-0548**))

Nguyen H, Zheng R (2013) On budgeted influence maximization in social networks. IEEE J Sel Areas Commun 31(6):1084–1094. https://doi.org/10.1109/jsac.2013.130610

Nguyen Hung T, Thai My T, Dinh Thang N (2016) Stop-and-stare: optimal sampling algorithms for viral marketing in billion-scale net- works. In: Proceedings of the 2016 international conference on management of data. SIGMOD '16. San Francisco, California, USA: association for computing machinery, pp 695–710. isbn: 9781450335317. https://doi.org/10.1145/2882903.2915207. url: https://doi.org/10.1145/2882903.2915207

Ok J, Youngmi J, Jinwoo S, Yung Y (2016) On maximizing diffusion speed over social networks with strategic users. IEEE/ACM Trans Netw 24(6):3798–3811. https://doi.org/10.1109/tnet.2016.2556719

Peng S, Aimin Y, Lihong C, Shui Y, Dongqing X (2017) Social influence modeling using information theory in mobile social networks. Inf Sci 379:146–159. https://doi.org/10.1016/j.ins.2016.08.023

Pérez-Peló S, Jesús S-O, Abraham D (2020) Finding weaknesses in networks using greedy randomized adaptive search procedure and path relinking. Expert Syst 2020:e12540

Pérez-Peló S, Jesús S-O, Raúl M-S, Abraham D (2019) On the analysis of the influence of the evaluation metric in community detection over social networks. Electronics 8(1):23

Qipeng Y, Shi R, Zhou C, Wang P, Guo L (2015) Topic-aware social influence minimization. In: Proceedings of the 24th International Conference on World Wide Web. WWW '15 Companion. Florence, Italy: Association for Computing Machinery, pp 139–140. ISBN: 9781450334730. https://doi.org/10.1145/2740908.2742767

Resende M, Celso GC, Ribeiro C (2013) GRASP: greedy randomized adaptive search procedures. In: Search methodologies. Springer US, pp 287–312. https://doi.org/10.1007/978-1-4614-6940-7_11

Resende M, Rafael-Martí GC, Micael G, Abraham D (2010) GRASP and path relinking for the max-min diversity problem. Comput Oper Res 37(3):498–508. https://doi.org/10.1016/j.cor.2008.05.011

Reza Z, Abbasi MA, Liu H (2014) Social media mining: an introduction. Cambridge University Press, Cambridge. https://doi.org/10.1017/CBO9781139088510

Richardson M, Rakesh A, Pedro D (2003) Trust management for the semantic web. In: Lecture notes in computer science. Springer, Berlin, Heidelberg, pp 351–368. https://doi.org/10.1007/978-3-540-39718-2_23

Robles JF, Manuel C, Oscar C (2020) Evolutionary multiobjective optimization to target social network influentials in viral marketing. In: Expert systems with applications, vol 147, p 113183. ISSN: 0957-4174. https://doi.org/10.1016/j.eswa.2020.113183, http://www.sciencedirect.com/science/article/pii/S0957417420300099

Salavati C, Abdollahpouri A (2019) Identifying influential nodes based on ant colony optimization to maximize profit in social networks. Swarm Evol Comput 51:100614. https://doi.org/10.1016/j.swevo.2019.100614

Samadi M, Nagi R, Semenov A, Nikolaev A (2018) Seed activation scheduling for influence maximization in social networks. Omega 77:96–114. https://doi.org/10.1016/j.omega.2017.06.002

Sen P, Lev M, José SA, Zheng ZA, Makse H (2014) Searching for superspreaders of information in real-world social media. Sci Rep 4:1. https://doi.org/10.1038/srep05547

Şimşek A, Kara R (2018) Using swarm intelligence algorithms to detect influential individuals for influence maximization in social networks. Expert Syst Appl 114:224–236. https://doi.org/10.1016/j.eswa.2018.07.038

Song G, Xiabing Z, Yu W, Xie K (2015) Influence maximization on large-scale mobile social network: a divide- and-conquer method. IEEE Trans Parallel Distrib Syst 26(5):1379–1392. https://doi.org/10.1109/tpds.2014.2320515

Wasserman S, Faust K (1994) Social network analysis. Cambridge University Press, Cambridge. https://doi.org/10.1017/cbo9780511815478

Tong G, Weili W, Shaojie T, Ding-Zhu D (2017) Adaptive influence maximization in dynamic social networks. IEEE/ACM Trans Netw 25(1):112–125. https://doi.org/10.1109/tnet.2016.2563397

Tong GA, Shasha L, Weili W, Ding-Zhu D (2016) Effector detection in social networks. IEEE Trans Comput Soc Syst 3(4):151–163. https://doi.org/10.1109/tcss.2016.2627811

Wang B, Ge C, Luoyi F, Li S, Xinbing W (2017) Drimux: dynamic rumor influence minimization with user experience in social networks. IEEE Trans Knowl Data Eng 29(10):2168–2181

Watts DJ, Strogatz SH (1998) Collective dynamics of small-world networks. Nature 393(6684):440–442. https://doi.org/10.1038/30918

Wei-Neng C, Tan D-Z, Yang Q, Gu T, Zhang J (2019) Ant colony optimization for the control of pollutant spreading on social networks. IEEE Trans Cybern 50(9):4053–4065

Xinjue W, Deng K, Li J, Yu JX, Jensen SC, Yang X (2018) Targeted influence minimization in social networks. In: Phung D, Tseng VS, Webb GI, Ho B, Ganji M, Rashidi L (eds) Advances in knowledge discovery and data mining. Springer International Publishing, Cham, pp 689– 700. ISBN: 978-3-319-93040-4

Yang W-S, Weng S-X (2012) Application of the ant colony optimization algorithm to competitive viral marketing. In: Lecture notes in computer science. Springer, Berlin, Heidelberg, pp 1–8. https://doi.org/10.1007/978-3-642-30448-4_1

Zhang H, Dung TN, Huiling Z, My TT (2016) Least cost influence maximization across multiple social networks. IEEE/ACM Trans Netw 24(2):929–939. https://doi.org/10.1109/tnet.2015.2394793

Zhang K, Haifeng D, Feldman MW (2017) Maximizing influence in a social network: improved results using a genetic algorithm. Phys A 478:20–30. https://doi.org/10.1016/j.physa.2017.02.067

Chapter 7

An efficient and effective GRASP algorithm for the Budget Influence Maximization Problem

The journal paper associated to this part is:

- Lozano-Osorio, I., Sánchez-Oro, J., Duarte, A. et al. An efficient and effective GRASP algorithm for the Budget Influence Maximization Problem. Journal of Ambient Intelligence and Humanized Computing (2023). https://doi.org/10.1007/s12652-023-04680-z [65]
- Status: **Published**
- Submitted: 01 February 2022.
- First Review: 11 March 2023.
- Accepted: 21 August 2023.
- Published: 21 September 2023.
- Impact Factor (JCR 2021): 3.662.
- Subject Category: Computer Science, Artificial Intelligence - 68/145 (Q2).

Journal of Ambient Intelligence and Humanized Computing
https://doi.org/10.1007/s12652-023-04680-z

ORIGINAL RESEARCH

An efficient and effective GRASP algorithm for the Budget Influence Maximization Problem

Isaac Lozano-Osorio[1] · Jesús Sánchez-Oro[1] · Abraham Duarte[1]

Received: 1 February 2022 / Accepted: 21 August 2023

Abstract
Social networks are in continuous evolution, and its spreading has attracted the interest of both practitioners and the scientific community. In the last decades, several new interesting problems have aroused in the context of social networks, mainly due to an overabundance of information, usually named as infodemic. This problem emerges in several areas, such as viral marketing, disease prediction and prevention, and misinformation, among others. Then, it is interesting to identify the most influential users in a network to analyze the information transmitted, resulting in Social Influence Maximization (SIM) problems. In this research, the Budget Influence Maximization Problem (BIMP) is tackled. BIMP proposes a realistic scenario where the cost of selecting each node is different. This is modeled by having a budget that can be spent to select the users of a network, where each user has an associated cost. Since BIMP is a hard optimization problem, a metaheuristic algorithm based on Greedy Randomized Adaptive Search (GRASP) framework is proposed.

Keywords Information systems · Social networks · Budget Influence maximization · Viral marketing · GRASP

1 Introduction

The continuous growth of social networks is increasing the data generated by active users exponentially in such a way that problems related to social networks are becoming a challenging task for traditional algorithms. A Social Network (SN) is defined as a set of social interactions among users with the aim of transmitting ideas, propagation of diseases, misinformation detection, or viral marketing, among others (see Reza et al. 2014; Barabási and Pòsfai 2016; Bello-Orgaz et al. 2017; Chen et al. 2020; Tretiakov et al. 2022).

The massive information available nowadays hinders the task of differentiating real from false information. Most of the research related to detecting fake news and misinformation are based on the analysis of the publication content and context-oriented methods, mainly tackled from the Natural Language Processing area. This research is focused on identifying the most influential users in a Social Network, which may help the algorithms to identify if the source of a piece of information has credibility or not (Noguera-Vivo et al. 2023).

Traditionally, an SN is represented by a graph $G = (V, E)$, where the users are modeled as the set of nodes V and the relation between two users $u, v \in V$ is modeled as an edge $(u, v) \in E$. If there is a relation between two users, then information can be transmitted between them, following one of the Influence Diffusion Models (IDM) which will be described in Sect. 3. Since information can be transmitted in several ways depending on the social network analyzed, or the nature of the relations, Kempe et al. (2003) proposed two different models of information spreading, which have led to a wide variety of new models in the last years.

This paper is intended to deal with a problem in the family of Social Influence Maximization (SIM). It is assumed that if in an SN there exists a relation between two users, then the information can be transmitted from one to another. Without loss of generality, the objective of each variant of SIM is to find a set S of users to start the diffusion of information with the aim of maximizing the scope of the information in terms of the number of users influenced. In the context of infodemics, identifying these users will allow other

✉ Jesús Sánchez-Oro
jesus.sanchezoro@urjc.es

Isaac Lozano-Osorio
isaac.lozano@urjc.es

Abraham Duarte
abraham.duarte@urjc.es

[1] Universidad Rey Juan Carlos, Móstoles, Spain

algorithms to elucidate if it is relevant to analyze the veracity of the information due to the capacity to spread of the user.

The most common variant is named as Social Network Influence Maximization Problem (SNIMP). The objective in this problem is to select a set of k nodes, with $k < n$, in such a way that the number of nodes in the network which are influenced is maximum. This problem has been widely studied in the literature (see Gong et al. 2016; Lozano-Osorio et al. 2021).

However, this variant is not even close to the real SN behavior. In particular, if a company is trying to spread information of their product through the network, the cost of selecting one or another user is not uniform, i.e., some users, which are usually known as influencers, will require a larger budget to be selected than any other anonymous user. The rationale behind this is that the influencer guarantees a larger spreading of the information than the anonymous user.

This paper deals with the Budgeted Influence Maximization Problem (BIMP), originally defined in Nguyen and Zheng (2013), which, instead of selecting a fixed number of initial users, allows us to invest a certain budget in users of the SN, considering that the cost of selecting users is not uniform. Notice that this variant is closer to real SN than SNIMP. In BIMP, the traditional model of SN is still considered, defining a network as a graph $G = (V, E)$, where V is the set of users and E the set of relations among them. However, a function $C : V \rightarrow \mathbb{Z}^+$ is introduced, which assign a non-uniform positive integer cost to every user of the network. Additionally, an initial budget B is given, which is the maximum investment that can be used to select nodes. Each selected node u will decrease the available budget in $C(u)$ units. Then, the BIMP consists of selecting a set of seed nodes $S^\star$ that maximizes the information diffusion throughout the network without exceeding the given budget B. More formally,

$$S^\star \leftarrow \underset{S \in \mathbb{SS}}{\arg\max}\, IDM(G, S) \, : \, \sum_{u \in S} C(u) \leq B$$

where $\mathbb{SS}$ represents all possible combinations of seed sets that can be generated, and *IDM* is one of the Influence Diffusion Models presented in Sect. 3.

The large amount of data and interest in SN have aroused the interest of both the scientific community and companies in considering BIMP for optimizing the spreading process of a certain message, product, or idea to clients. Marketing agencies like BrandWatch (see Hayes et al. 2021) use this approach when their customers need a commercial campaign based on a certain budget to determine the most effective users to initialize the campaign. Last years, a wide variety of works related to infodemics are focused on pandemic prediction and vaccination discussions (Chen et al. 2020; Bello-Orgaz et al. 2017). The results on BIMP will be able to identify the most influential users in this context, with the aim of validating their credibility when spreading information and their scope.

In the original definition of BIMP (Nguyen and Zheng 2013), an approximation algorithm is presented which guarantees an approximation ratio of $1 - 1/\sqrt{e}$ is presented. Additionally, they proposed a directed acyclic graph-based heuristic for this problem. This problem has been widely studied mainly due to its practical applications. We refer the reader to Sect. 2 for a detailed review of the literature about BIMP. The main contributions of this research are the following:

- A solution framework based on the Greedy Randomized Adaptive Search Procedure methodology.
- A novel efficient and effective heuristic for selecting the seed set in the constructive phase. This heuristic is experimentally compared with the best previous approaches to show its contribution.
- Three influence diffusion models are tested, instead of just one IDM as in the previous research, showing the robustness of the proposal.
- A scalable algorithm for solving BIMP. Since SNs are exponentially growing, it is necessary to provide a highly-scalable algorithm able to deal with eventually large SNs.
- A comparison of the proposed algorithm with the best methods found in the literature using three publicly available social network datasets which were originally considered in previous works.
- A real-life instance directly related to infodemics is generated based on tweets retrieved from the publicly available dataset called Tweetsets in the area of Healthcare.
- A public repository[1] with the developed code to ease further comparisons.

The remainder of the work is structured as follows. Section 2 reviews the related literature, detailing the different approaches followed to deal with different problems derived from SIM. Then, Sect. 3 introduces the most extended IDMs in the literature, which are also used in this research. The proposed approach is described in Sect. 4, where Sect. 4.1 presents the construction method to provide high-quality initial solutions, and Sect. 4.2 describes the proposed local search to find local optima with respect to a given neighborhood structure. Section 5 presents the experimental results considering a public dataset which has been previously used for this task in order to have a fair comparison, divided into preliminary experiments, which are devoted to adjust the search parameters (Sect. 5.1),

[1] https://grafo.etsii.urjc.es/BIMP.

and final experiments, with the aim of performing a competitive testing to evaluate the quality of the proposal (Sect. 5.2). An infodemic case study is developed based on Healthcare tweets in Sect. 5.3. Finally, Sect. 6 draws some conclusions derived from this research.

2 Literature review

The problem of selecting target nodes in SNs to spread information was introduced by Richardson et al. (2003) proposing the first problem formulation. Kempe et al. (2003) presented a heuristic approach to solve the SNIMP, and in Kempe et al. (2015) proved that SNIMP is $\mathcal{NP}$ -hard.

Nguyen and Zheng (2013), formally defined the BIMP inspired by SNIMP, showing its $\mathcal{NP}$-hardness based on SNIMP. They developed an approximation algorithm which guarantees an approximation ratio of $1 - 1/\sqrt{e}$, being e the base of the natural logarithm. Most of the heuristic proposals for BIMP are inspired by the original algorithms for SNIMP, mainly due to its similarities and the computational effort required to evaluate the IDM. Kempe et al. (2003) presented several greedy heuristics with an approximation of $1 - 1/e - \epsilon$, where ϵ is any positive real number. When the considered greedy function of the heuristic is the degree of the node, the algorithm is called *high-degree heuristic*. Based on the node degree idea, Chen et al. (2010) proposed a new greedy function to optimize the high-degree heuristic, such as the greedy selection function considering the redundancy between likely influenced nodes, but discarding those reached by the already selected seed nodes, in order to provide a better estimation of the total spread.

Han et al. (2014) proposed a set of heuristics for optimizing BIMP by considering influential nodes and cost-effective nodes to increase both accuracy and effectiveness. Later on, Guney et al. (2015) proposed a sample average approximation method for BIMP, which is able to reach almost near optimal solutions.

Banerjee et al. (2020) published the latest survey in SIM, becoming one of the most relevant research works in the area of influence maximization problems. The algorithm named ComBIM proposed by Banerjee et al. (2019) is considered the state of the art for BIMP. ComBIM provides a community-based solution that provides the best results in the literature as far as our knowledge, so it will be considered as the algorithm to benchmark our proposal. Recently, Lozano-Osorio et al. (2021) proposed a new heuristic method for selecting the seed set in the context of SNIMP. This work adapts this heuristic with the aim of evaluating the performance of the proposal over a different variant of the same family of problems.

3 Influence diffusion model

The evaluation of the influence of a given seed set S over a network G requires the definition of an Influence Diffusion Model (IDM). An IDM is responsible for deciding which nodes are affected or influenced by the information received from their neighbors in the SN. The most extended IDMs in the literature are: Independent Cascade Model (ICM) (see Kempe et al. 2003; Goyal et al. 2011), Weighted Cascade Model (WCM) (see Kempe et al. 2003), and Tri-Valency Model (TV) (see Granovetter 1978). All of them are based on assigning an influence probability to each relational link in the SN, since a relation in a network does not necessarily implies that a user influence another one in a certain period of time.

- ICM, which is one of the most used IDMs, considers that the influence probability is the same for each link.
- WCM considers that the probability of a user v for being influenced by user u is proportional to the in-degree of user v, i.e., the number of users that can eventually influence user v. Therefore, the probability of influencing user v is defined as $1/d_{in}(v)$, where $d_{in}(v)$ is the in-degree of user v.
- TV select the edge probability randomly from the set of probabilities $(1\%, 0.1\%, 0.001\%)$.

Following the recommendations of the literature and, more specifically, the state of the art for BIMP, the three aforementioned IDMs are evaluated in this work. The only IDM that requires for an input parameter is ICM, where the probability values are set to 1% and 2%, as stated in Banerjee et al. (2019).

Due to the probabilistic nature of the IDM, the most extended way of evaluating the spread is by conducting a Monte Carlo simulation (MC). However, even a single iteration of the simulation model is rather time-consuming when considering large graphs derived from SN. Algorithm 1 shows the pseudocode of the Monte Carlo simulation used to evaluate the spread of information through an SN named G given a certain seed set S. Specifically, it receives four input parameters: the graph which models the SN, G; a solution, S; the IDM used to select the diffusion probability of a node to be influenced, *IDM*; and the number of repetitions r performed to avoid the impact of randomness.

Algorithm 1 $MC(G = (V, E), S, IDM, r)$

```
1:  $I \leftarrow 0$
2:  for $i \in 1 \ldots r$ do
3:      $A^\star \leftarrow S$
4:      $A \leftarrow S$
5:      while $A \neq \emptyset$ do
6:          $D \leftarrow \emptyset$
7:          for $v \in A$ do
8:              for $(u, v) \in E$ do
9:                  if $rnd(0, 1) \leq IDM(u)$ then
10:                     $D \leftarrow D \cup \{u\}$
11:                 end if
12:             end for
13:         end for
14:         $A \leftarrow D \setminus A^\star$
15:         $A^\star \leftarrow A^\star \cup D$
16:     end while
17:     $I \leftarrow I + |A^\star|$
18: end for
19: return $I/r$
```

The algorithm starts by initializing the total number of infected users (step 1). Then, the algorithm performs a number of predefined simulations r (steps 2–18), finding in each iteration which are the influenced nodes starting from the given seed set S. Initially, the set of nodes $A^\star$ reached by the initial seed set, S, is actually the seed set (step 3). Then, the method iterates until no new nodes are influenced (steps 5–16). In each iteration of the inner for-loop, the method evaluates the IDM for each node directly related to a recently influenced node (steps 8–12). For each neighbor, a random number is generated. If this number is smaller than the probability of infection p that is determined by the IDM used, then it is considered that the neighbor becomes infected (steps 9–11). At the end, the set of the nodes infected in the previous iteration (step 14) that are not just analyzed as well as infected nodes is updated (step 15). Finally, the algorithm returns the average number of infected nodes among all simulations performed (step 19). Notice that this value is considered as the objective function to be optimized when solving the BIMP or, generally, any SIM problem. Therefore, the seed set which maximizes the *spread value* over the network would compose the optimal solution to the problem. It is worth mentioning that, as infection is a stochastic process, the IDM must be executed a considerably large number of iterations to achieve an appropriate estimation, thus resulting in a Monte Carlo simulation.

4 Algorithmic approach

The proposed algorithm follows the Greedy Randomized Adaptive Search Procedure (GRASP) methodology, which was originally introduced by Feo and Resende (1989) and formally defined in Feo et al. (1994). We refer the reader to Resende Mauricio et al. (2010), Resende Mauricio and Ribeiro (2013) for a complete survey of the last advances in this methodology.

GRASP is a multistart metaheuristic, divided into two distinct phases: construction and local improvement. The first phase consists of a greedy, random, and adaptive construction of a solution, in order to provide a promising starting point. The second phase consists of a method to locally improve the constructed solution to a local optimum with respect to a given neighborhood.

A recent proposal by Lozano-Osorio et al. (2021) uses GRASP method to solve SNIMP, since GRASP methodology is able to find a trade-off between diversification in the stochastic construction phase and the intensification of the local search process, enabling the algorithm to escape from local optima and perform a wider search space exploration.

These two phases are repeated until a termination criterion is met. Notice that this criterion makes the algorithm scalable to eventually large social networks, since the termination criterion can be tuned to perform a smaller number of iterations.

In the context of BIMP, a novel heuristic for selecting the most promising nodes in the construction phase is proposed. Additionally, the local improvement phase considers a move based on the replacement of nodes and it can be limited to avoid large computing times without significantly deteriorating the quality of the solutions found.

4.1 Construction phase

The purpose of the GRASP construction phase is to generate a promising initial solution in a short computing time. In order to do this, the construction phase is usually guided by a greedy selection function, which helps the constructive method to select the most promising elements to be included in the partial solution (see Algorithm 2). It is worth mentioning that, in the context of BIMP, the computational effort required to evaluate the greedy function value should be minimal, since the size of the social networks might lead the algorithm to be extremely slow when a solution is constructed.

Algorithm 2 *Construct*$(G = (V, E), \alpha, B)$

```
1: $v \leftarrow rnd(V)$
2: $S \leftarrow \{v\}$
3: $B \leftarrow B - C(v)$
4: $CL \leftarrow \{c \in V : c \notin S \wedge C(c) \leq B\}$
5: while $CL \neq \emptyset$ do
6:     $g_{\min} \leftarrow \min_{u \in CL} g(u)$
7:     $g_{\max} \leftarrow \max_{u \in CL} g(u)$
8:     $\mu \leftarrow g_{\max} - \alpha \cdot (g_{\max} - g_{\min})$
9:     $RCL \leftarrow \{v \in CL : g(v) \geq \mu\}$
10:    $u \leftarrow rnd(RCL)$
11:    $S \leftarrow S \cup \{u\}$
12:    $B \leftarrow B - C(u)$
13:    $CL \leftarrow \{c \in CL : c \notin S \wedge C(c) \leq B\}$
14: end while
15: return $S$
```

The algorithm starts by randomly selecting the first node to be included in the solution S at random from the set of nodes V (steps 1–2). The random selection of the first element to be included in the solution is customary in GRASP since it favors diversification. When selecting the first node v, the available budget is decremented with the cost of v, $C(v)$ (step 3) The candidate list CL is then created with all nodes whose cost is smaller than the available budget B which are not already in the solution S (step 4). Then, the constructive method iteratively adds new elements to the solution until no candidate nodes with a cost smaller than B are available to be selected (steps 5–14). In each iteration, the minimum and maximum value of the greedy heuristic function is evaluated (steps 6–7). Since the greedy function is a key feature of GRASP, hereinafter each greedy function considered is described. Then, a threshold μ is calculated (step 8), which is required for creating the Restricted Candidate List (*RCL*) with the most promising nodes (step 9). This threshold directly depends on the value of the input parameter α, with $0 \leq \alpha \leq 1$. Notice that this parameter indicates the greediness or randomness of the constructive procedure. On the one hand, if $\alpha = 0$, then the threshold is evaluated as $g_{\max}$, becoming a totally greedy algorithm (i.e., the *RCL* only includes those nodes with the maximum greedy function value). On the other hand, if $\alpha = 1$ then $\mu = g_{\min}$, resulting in a completely random method (i.e., the *RCL* includes every candidate node whose cost is smaller than the available budget). Since this parameter is experimentally tuned, we refer the reader to Sect. 5 to analyze the effect of different values for the α parameter in the final algorithm. Then, the next node is selected at random from the *RCL* (step 10), including it in the solution under construction (step 11), updating the budget by reducing it with the cost of the selected node (step 12). The *CL* is also updated (step 13) in the same way as (step 4), being candidate nodes whose associated cost is smaller than the remaining budget. The method ends when no more elements can be included in the seed set (i.e., there is no candidate node whose cost is smaller than the available budget), returning the constructed solution S (step 15).

As it was aforementioned, the greedy heuristic function g used in steps 6–7 is one of the key features when designing a constructive procedure in the context of GRASP. In particular, this greedy function must select the most promising nodes without requiring large computing times. In this work, we adapt two existing greedy functions originally proposed for SNIMP, and propose a novel one to analyze its performance against the well-established ones.

The first greedy function, named g_{deg}, considers the out-degree of a node as heuristic value. Given a node u, let us define out-degree as $d_u^+ = |N_u^+|$, where $N_u^+ = \{v \in V : (u, v) \in E\}$ In mathematical terms,

$$g_{deg}(u) = d_u^+$$

The second greedy function, named g_{2step}, was originally used for SNIMP (Lozano-Osorio et al. 2021). It is a heuristic based on the first and second degree neighbors of a given node, usually known as 2-step neighbors in the context of SN analysis (Stanley and Katherine 1994). The evaluation of this greedy function over a certain node u can be formally defined as:

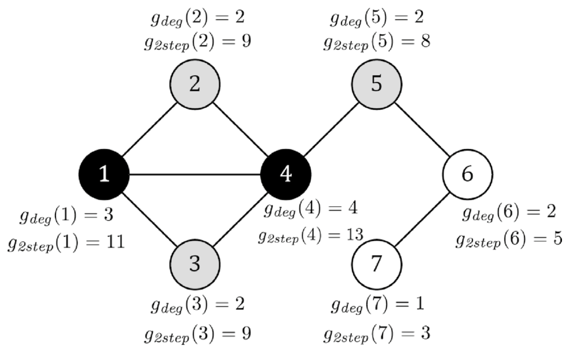
$$g_{2step}(u) = d_u^+ + \sum_{v \in N_u^+} d_v^+$$

Additionally, this work proposes a novel heuristic, named g_{dist} which leverages the node seed distribution. This method prioritizes nodes that do not have selected neighbors as a seed node, with the aim of reaching a larger number of non-influenced users by exploring regions that have been mainly ignored until that point. In order to do so, the greedy function value of a node is directly its degree, but penalizes it if some of its neighbor nodes have already been selected. The penalization has been experimentally set by halving the degree. More formally

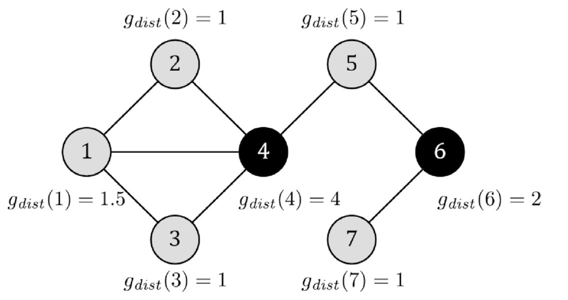
$$g_{dist} = \begin{cases} d_u^+ & \text{if } v \notin S,\ \forall v \in N_u^+ \\ \frac{d_u^+}{2} & \text{otherwise} \end{cases}$$

Let us illustrate the behavior of each proposed greedy heuristic function with an example of SN with 7 nodes and 8 relations, depicted in Fig. 1. The value of each heuristic function is presented close to each node.

Figure 1a shows the evaluation of g_{deg} and g_{2step}, since both of them result in the same solution. In the case of g_{deg}, the first selected node is 4, since it is the node with the largest degree. Then, node 1 is selected as the one with the second largest degree. This seed set is able to directly influence up to three nodes, reducing the possibility of influencing



(a) Solution $S_1 = \{\texttt{1, 4}\}$, considering g_{deg} and g_{2step} heuristic, which may influence up to three nodes: 2, 3, and 5.



(b) Solution $S_2 = \{\texttt{4, 6}\}$, considering the g_{dist} heuristic, which may influence all nodes: 1, 2, 3, 5 and 7.

Fig. 1 Evaluation of the three considered heuristic functions to select the initial seed set over a SN with 7 nodes and 8 relations

nodes 6 and 7. In particular, there will be a possibility of influencing nodes 6 and 7 if and only if node 5 is influenced. The same behavior can be seen when considering g_{2step}: the first selected node is 4, which presents the largest value and, then, node 1 is selected. Finally, Fig. 1b shows the resulting solution when considering g_{dist} heuristic. In this case, the first node is selected with respect to its degree, resulting in node 4. Then, the heuristic value of the nodes directly connected to node 4 is evaluated as their degree reduced by half, while the heuristic value of non-directly connected nodes still remains as their degree. Then, the second node selected is 6, which presents the largest value for g_{dist}. Notice that, in this example, both g_{deg} and g_{2step} reports the same solution, although the value of each greedy function is different. However, the idea of penalizing those nodes connected to already selected ones, used in g_{dist}, lead the constructive procedure to reach a better solution. The impact and the influence of each greedy constructive procedure will be deeply analyzed in Sect. 5.

4.2 Local improvement

The second phase of GRASP is responsible for locally improving each solution generated by the constructive procedure with the aim of reaching a local (ideally global) optimum. In the context of GRASP, this phase can be accomplished by using a simple local search procedure or a more complex heuristic (even a complete metaheuristic) like Tabu Search (see Martí et al. 2018). The elevated complexity of the problem under consideration has led us to propose a simple yet effective local search procedure to reduce the computational effort required.

Before defining a local search method, it is necessary to introduce the neighborhood to be explored. The neighborhood of a solution S is defined as the set of solutions that can be reached by performing a single move over S. Then, it is necessary to define the move that will be considered in the context of BIMP. Specifically, the move, named as $Replace(S, u, P)$, involves removing node u from the solution and replacing it with the set of nodes in P, with $P \in V \setminus S$. Notice that, in order to reach a feasible solution, the sum of the cost of nodes in P must be smaller or equal than $B + C(u)$ (since u will be removed, its cost must not be taken into account). More formally,

$$Replace(S, u, P) = S \setminus \{u\} \cup P$$

Then, given a solution S, the neighborhood $N_R(S)$ is defined as the set of feasible solutions that can be reached with a single $Replace$ move. In mathematical terms,

$$N_R(S) = \left\{ S' \leftarrow Replace(S,u,P)\ \forall u \in S \wedge \forall P \in V \setminus S : \textstyle\sum_{p \in P} C(p) \leq B + C(u) \right\}$$

Having defined the neighborhood which will be explored in the local search, the next step consists of defining the way in which the neighborhood $N_R(S)$ is explored. Even considering an efficient implementation of the objective function evaluation, the vast size of the resulting neighborhood makes the complete exploration of the neighborhood not suitable for the BIMP. Therefore, we limit the number of evaluations that the local search performs with the aim of having a computationally efficient method. It is worth mentioning that, if the number of iterations Ψ is limited, then it is interesting to firstly explore the most promising neighbors of the considered neighborhood. Therefore, an intelligent neighborhood exploration strategy is presented.

Hansen and Mladenović (2006) performed an empirical study on the well-known Traveling Salesman Problem to compare first and best improvement strategies in the context of local search. The authors conclude that both strategies present similar results in terms of quality, but the first improvement approach is faster when considering randomness in the constructive phase. Following their recommendations and due to the computational effort required to evaluate a solution for the BIMP, we propose a first improvement approach. This strategy does not need to explore all solutions in the neighborhood for each iteration, thus reducing the number of objective function evaluations required and consequently the overall run time.

The proposed strategy filters the nodes that are involved in every iteration of the local search method. In particular, the nodes considered for removal from the solution S are selected at random, but the ones to be later included are selected by their contribution to the objective function value if they are included in the incumbent solution. Notice that evaluating the contribution requires performing a Monte Carlo simulation, which is rather time consuming. With the aim of reducing the computational effort of this evaluation, a single Monte Carlo execution is performed, i.e., the value of r in Algorithm 1 is an input parameter of the local search method named Δ (see Sect. 5.1 where an experiment to analyze the performance of r value is is done). Furthermore, in order to increase the efficiency, the Monte Carlo simulation is not performed from scratch. Instead, since the solution has already been evaluated, the influenced nodes are known. Then, to evaluate the contribution of inserting a new node v in the solution, it is only required to evaluate which are the new nodes influenced by v, resulting in an efficient way of estimating the contribution of including v in the incumbent solution. Then, the candidate nodes to be included are those with the largest contribution to the objective function value. The number of candidates to be evaluated is determined by the maximum number of evaluations Ψ (see Sect. 5.1 where an experiment with different Ψ values is carried out). The pseudocode of the local search LS is shown in Algorithm 3.

Algorithm 3 LS$(G, IDM, S, \Psi, \Delta)$

```
1:  $R \leftarrow rnd(S, \Psi)$
2:  for $u \in R$ do
3:      $\Psi = \Psi - 1$
4:      $P^\star \leftarrow \arg\max_{P \in V \setminus S} MC(G, S, IDM, \Delta)$ : $\sum_{p \in P} C(p) \leq B + C(u)$
5:      $S' \leftarrow Replace(S, u, P^\star)$
6:      if $MC(G, IDM, S', \Delta) > MC(G, IDM, S, \Delta)$ then
7:          $S \leftarrow S'$
8:          go to 1                                  ▷ Improvement found
9:      end if
10: end for
11: return $S$
```

The method starts by creating the set of nodes whose removal is tested (step 1). In particular, it consists of a random set of Ψ nodes extracted from the nodes which are not already in the solution. If the number of available nodes is smaller than Ψ, then all nodes are candidates. For each candidate node (steps 2–10), the available number of evaluations is decremented (step 3) and, then, the set of most promising nodes to be included $P^\star$ is created as those maximizing the contribution to the objective function value if included in the solution satisfying the cost constraint when removing u from S (step 4). Once both the candidate node to be removed u and the set of most promising nodes to be included $P^\star$ are selected, the *Replace* move is performed, resulting in a neighbor solution S' (step 5). Then, S is updated if S' results in a better solution (step 7), restarting the search since the local search method follows a first improvement strategy (step 8). The method ends returning the best solution found during the search (step 11).

5 Computational experiments and analysis of results

The aim of this section is to describe the computational experiments designed to evaluate the performance of the proposed algorithms and to analyze the obtained results. All experiments have been performed in an Intel Core i7-9750 H (2.6 GHz) with 16GB RAM and the algorithms were implemented using Java 17 and the *Metaheuristic Optimization framewoRK*[2] (MORK) 10, designed to facilitate the implementation of algorithms for solving $\mathcal{NP}$-hard problems. The source code of the proposed methods has also been made publicly available.[3]

The set of SNs considered in this paper has been entirely obtained from the best algorithm found for BIMP in the literature, to provide a fair comparison among the analyzed algorithms. All of them are publicly available in Stanford Network Analysis Project (SNAP).[4] The datasets used were the Epinions dataset (Richardson et al. 2003; Xu et al. 2016) which consists of 75879 nodes and 508837 edges, the dataset HepTh with 27770 nodes, and 352807 edges and finally CondMat which has 23133 and 93497 edges (Leskovec et al. 2007).

Since this research is designed for improving the analysis of users in an infodemics context, we have generated a real-life instance with tweets related to a health case published about the announcement of the American Health Care Act (AHCA) in 2017. The new dataset contains 54836 nodes and 89059 edges (we refer the reader to Sect. 5.3 for a more detailed description about this instance).

Table 1 shows the following details of the datasets used: number of nodes in the largest connected-component (|*LCC*|), total number of connected-components

[2] https://github.com/rmartinsanta/mork/.

[3] https://grafo.etsii.urjc.es/BIMP.

[4] https://snap.stanford.edu/.

Table 1 Metrics of the used datasets

Instance	Nodes	Edges	$\|LCC\|$	TC	ADPN
soc-Epinions1	75879	405740	75877	2	10.69
HC Twitter	54836	89059	47257	3060	3.25
CA-CondMat	23133	93497	21363	567	8.08
CA-HepT	9877	25998	8638	429	5.26

(TC) and average out degree for all nodes (more formally $\frac{1}{n}\sum_{i=1}^{n}(deg(v_i))$).

It is worth mentioning that the original SNs derived from SNAP do not have any weight in the nodes. In the context of BIMP, every node has an associated cost to be selected, so it is necessary to perform this assignment. With the aim of having a fair comparison, we contact the authors of the previous work for their exact cost assignment and the source code to execute the algorithm in the same platform. Unfortunately, we did not receive any response, so we implement their algorithm carefully following the detailed description provided in the manuscript (Banerjee et al. 2019). Additionally, we generate a random uniform cost for each node following the suggestions of the previous authors. In order to ease further comparisons, we have made publicly available the exact instances used in this work.

First of all, it is important to indicate the number of repetitions performed in the Monte Carlo simulation. As it is customary in SIM problems, 100 Monte Carlo simulations are performed on all IDMs models. The total budget B to conform a solution is selected in the range $B = \{2000, 6000, 10000, 140000, 180000, 22000, 26000\}$ as stated in Banerjee et al. (2019), thus obtaining $3 \cdot 7 = 21$ different problem instances for each IDM. Taking into account that 4 IDMs are considered as described in Sect. 3, the total number of instances are $21 \cdot 4 = 84$.

The experiments are divided into two parts: preliminary and final experimentation. The former (Sect. 5.1) refers to those experiments performed to select the best parameters to set up our algorithm, while the latter (Sect. 5.2) validates the best configuration, comparing it with the best method found in the state of the art.

All experiments developed report the following performance metrics: Avg., the average of the number of influenced nodes; Time (s), the average execution time of the algorithm measured in seconds; Dev (%), the average deviation with respect to the best solution found in the experiment, evaluated as $\frac{f_{best}-f_a}{f_{best}} \cdot 100.$, where f_{best} is the objective function of the best solution found in the experiment and f_a is the objective function value of the best solution found by the algorithm; and finally, #Best, the number of times that the algorithm matches the best solution in the experiment. Tables report a summary to provide a global view of each algorithm by averaging the results obtained along all the considered instances. Individual results per instance are included in the public repository, where the code is also available.

Table 2 Results of the constructive procedure when generating 50 solutions, considering different α values for every heuristic function

Greedy function	α	Avg.	Time (s)	Dev (%)	#Best
g_{dist}	0.25	3446.91	14.96	0.87	6
	0.50	3430.71	14.67	2.06	4
	0.75	3346.41	14.00	7.14	1
	RND	**3462.20**	**14.67**	**0.81**	**10**
g_{deg}	0.25	3337.21	9.77	5.54	2
	0.50	3338.72	9.38	5.73	2
	0.75	3332.81	9.06	6.96	0
	RND	3417.53	9.68	3.44	6
g_{2step}	0.25	3146.76	9.60	11.92	0
	0.50	3158.04	9.20	12.28	0
	0.75	3172.05	9.38	11.60	0
	RND	3246.42	9.72	9.05	0

Best results are highlighted with bold font

5.1 Preliminary experimentation to setup the final GRASP method

In the preliminary experiments, 6 representative SNs are evaluated with each IDM, resulting in 24 instances. The set of preliminary instances comprehends the 28% of the total set of 84 instances. This selection of instances is done to avoid overfitting in the model.

The purpose of the first preliminary experiment is to obtain the best greedy heuristic function together with the value of α. For this purpose, all greedy heuristic methods, g_{deg}, g_{2step}, and g_{dist}, have been analyzed when considering $\alpha = (0.25, 0.50, 0.75, RND)$. Notice that $\alpha = RND$ indicates that a random value in the range 0-1 is selected for each construction. The GRASP method used 50 iterations in all experiments.

Table 2 collects the final results from this competitive testing. Notice that Avg. is not an integer value since it is the average value of the 100 repetitions of the Monte Carlo simulation.

As it can be drawn from the table, the best results are consistently provided by the greedy function based on the new heuristic procedure, g_{dist}. In particular, the best results are obtained when considering $\alpha = RND$, with 10 best solutions and 0.81% of average deviation. The small deviation value indicates that, even in the cases in which it is not able to reach the best solution, it remains really close to it. It is worth mentioning that the heuristic g_{2step}, which is the

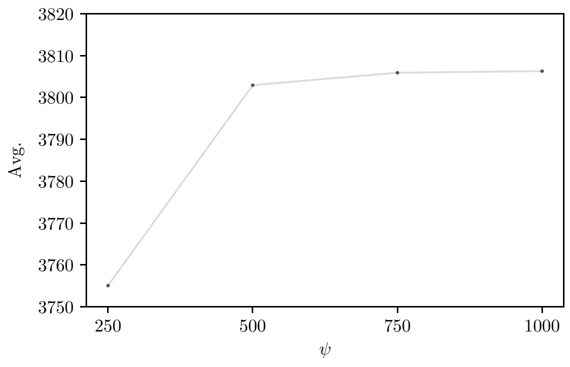


Fig. 2 Analysis of the effect of the value of Ψ in the local search phase

Table 4 Competitive testing of the proposed GRASP algorithm with respect to state of the art algorithm ComBIM

IDM	Algorithm	Avg	Time (s)	Dev (%)	#Best
ICM(1%)	ComBIM	8319.68	214.97	17.64%	0
	GRASP	**8872.61**	**117.06**	**0.00%**	**21**
ICM(2%)	ComBIM	14467.65	215.31	6.49%	3
	GRASP	**14828.77**	**146.21**	**0.07%**	**18**
WC	ComBIM	2277.79	214.04	57.49%	0
	GRASP	**10087.08**	**97.80**	**0.00%**	**21**
TV	ComBIM	1976.11	214.68	39.10%	0
	GRASP	**2677.58**	**69.65**	**0.00%**	**21**
Summary	ComBIM	6760.31	214.75	30.18%	3
	GRASP	**9116.51**	**107.68**	**0.02%**	**81**

Best results are highlighted with bold font

Table 3 Results of the GRASP algorithm when generating 50 solutions, considering $\alpha = RND$ versus the constructive procedure executed isolatedly

IDM	Method	Avg.	Time (s)	Dev (%)	#Best
ICM(1%)	g_{dist}	3379.44	**18.79**	15.62	0
	GRASP	**4005.01**	31.95	**0.00**	**9**
ICM(2%)	g_{dist}	7246.65	**26.98**	7.94	0
	GRASP	**7872.00**	62.43	**0.00**	**9**
WC	g_{dist}	2477.73	**15.24**	4.81	0
	GRASP	**2603.01**	21.07	**0.00**	**9**
TV	g_{dist}	615.81	**8.85**	39.75	0
	GRASP	**1022.11**	12.11	**0.00**	**9**
Summary	g_{dist}	3461.23	17.47	9.96	0
	GRASP	**3844.21**	**31.89**	**0.00**	**36**

Best results are highlighted with bold font

best greedy function in the context of SNIMP in Lozano-Osorio et al. (2021), produces the worst values in the context of BIMP, independently of the selected α-value. This result highlights the relevance of proposing new heuristics for this problem. Regarding the computing time, although g_{dist} requires slightly larger computing times on average, the difference with the other greedy heuristic functions are negligible. Therefore, we select g_{dist} as the best constructive procedure with $\alpha = RND$.

The next experiment is devoted to analyze the effect of the maximum number of iterations Ψ in the local search phase in terms of quality and computing time. Figure 2 shows the improvement when increasing the value of Ψ. Notice that the quality of the solutions significantly improves upon reaching $\Psi = 500$. At that point, the search seems to stagnate and no considerable improvement is found, thus leading us to select $\Psi = 500$ for the local search phase.

The third preliminary experiment is devoted to analyze the contribution of the local search phase in the complete GRASP algorithm. In order to do so, the constructive procedure considering g_{dist} and $\alpha = RND$ is executed and compared with the complete GRASP framework. The results are shown in Table 3.

As it can be derived from the results, the local search requires twice the computational time than the constructive procedure isolatedly, on average, in each IDM. This increase is justified since it is able to reach a considerably better solution, as it can be seen in the large average deviation values presented by the constructive procedure without local search. In particular, the average deviation is 9.96% on average, reaching a maximum value of 39.75% in the case of TV. Notice that, in the case of TV, the random selection of the probability of being influenced affects on the obtained results, being ICM and WC are more robust in the comparison. Regarding the number of best solutions found, it can be seen how the local search phase is able to improve the initial solutions, since the constructive procedure is not able to reach any best value.

Having performed the preliminary experiments, the best results are obtained with the following values: the greedy heuristic function selected is g_{dist}, the parameter of the constructive procedure is set to $\alpha = RND$, the maximum number of evaluations is $\Psi = 500$, and the number of Monte Carlo simulations is set to $\Delta = 10$ in the local search (this value has been set since no significant differences have been found testing different values in the range [10,100]). These parameter values will be used to set up the final version of the algorithm.

5.2 Competitive testing

In order to analyze the quality of the proposed algorithm, a competitive testing is performed with the best method found in the state of the art, ComBIM, by considering the complete set of 84 instances.

Table 4 collects the final results obtained in this competitive testing, where for each IDM the same metrics as in the preliminary experimentation are reported: Avg., Time (s), Dev(%), and #Best.

The results show how GRASP is able to obtain high-quality solutions (81 best solutions out of 84), and this values are obtained in half of the computing time (107.68 s vs 214.75 s). Although GRASP is able to outperform ComBIM in all IDMs considered, the most remarkable results in terms of quality are obtained when using WC and TV. Specifically, ComBIM is able to reach the best solution just in three instances when using ICM (2%). In this case, the deviation of GRASP is 0.07%, indicating that it is really close to that best solution. In view of these results, GRASP emerges as one of the most competitive algorithms for BIMP.

We finally perform over all instances the well-known non-parametric Wilcoxon statistical test for pairwise comparisons, which answers the question: do the solutions generated by both algorithms represent two different populations? The resulting p-value smaller than 0.0001 when comparing GRASP with ComBIM confirms the superiority of the proposed GRASP algorithm. In particular, GRASP is able to obtain 81 out of 84 positive ranks, 3 negative ranks, and 0 ties. Therefore, GRASP emerges as one of the most competitive algorithms for the BIMP, being able to reach high-quality solutions in small computing times.

5.3 An infodemic case study

This section shows an infodemic case study, based on tweets retrieved from the George Washington University's publicly available dataset called Tweetsets (Wrubel et al. 2020). Existing tweets where a user shares a tweet, that means that the user has been influenced by the original tweet, are used to build this instance. The tweetset used is related to infodemics in the area of Healthcare, related to the announcement of the American Health Care Act (AHCA) in 2017. This dataset consists of 386384 tweets, where 284131 are retweets. The original dataset contains all the identifiers of the tweets and, in order to generate this instance, we have retrieved it from Twitter, resulting in 96705 tweets. Notice that 187426 tweets have been removed from Twitter due to fake news filters or suspended accounts (Tretiakov et al. 2022). The final dataset has 54836 users, 96705 tweets, and 2060 components, where the largest component has 47257 nodes. The available budget for BIMP is generated following the same procedure of the previous instances: a random uniform cost is generated for each node. In order to compare our proposal, a competitive testing if performed.

Table 5 Competitive testing of the proposed GRASP algorithm with respect to state-of-the-art algorithm ComBIM

IDM	Algorithm	Avg	Time (s)	Dev (%)	#Best
ICM(1%)	ComBIM	6621.69	318.91	0.65%	1
	GRASP	**6663.23**	**42.39**	**0.04%**	**6**
ICM(2%)	ComBIM	**12833.06**	319.01	**0.00%**	**6**
	GRASP	12722.24	**55.70**	0.86%	1
WC	ComBIM	23898.77	319.32	23.09%	0
	GRASP	**31267.49**	**95.12**	**0.00%**	**7**
TV	ComBIM	1845.83	318.69	17.48%	0
	GRASP	**2247.41**	**95.12**	**0.00%**	**7**
Summary	ComBIM	11299.84	318.98	10.31%	7
	GRASP	**13225.09**	**72.08**	**0.22%**	**21**

Best results are highlighted in bold font

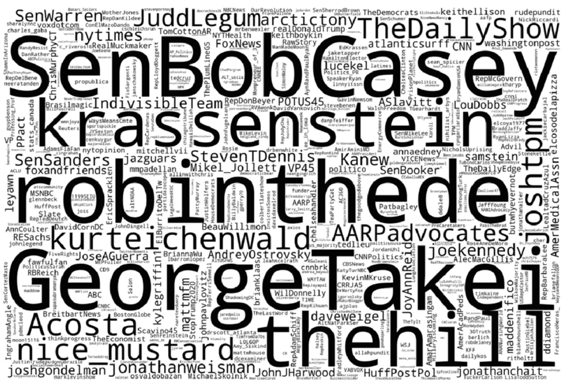


Fig. 3 Most influential users in the HC Twitter dataset detected by GRASP over different IDM

Table 5 shows the results resulting from the comparison of GRASP and ComBIM over this case study. As it can be derived, GRASP obtains 21 best solutions out of 28, requiring less than four times the computing time from ComBIM (72.08 vs 318.98 s). ComBIM method performs better with the ICM influence diffusion model than WC and TV, showing the same behavior as in the previous instances. On the contrary, our method adapts to each IDM due to the evaluation of the objective function with the Monte Carlo simulation. We perform the Wilcoxon non-parametric statistical test resulting in a p-value smaller than 0.0001, confirming that GRASP is statistically better than ComBIM.

Having selected the most influential users with GRASP, it is necessary to analyze who are those users and how they are related to the context under evaluation. For this purpose, a word cloud has been constructed so that the weight of a user is directly proportional to the times that it has selected by GRASP as an influential user (seed node) for each IDM. This weight determines the size of the font used in the word cloud.

Figure 3 highlighted the most influential users. For instance, *SenBobCasey* is an US senator from the Democratic Party which was really active in the context of Health Care; *krassenstein* is the social account of an famous independent investigative journalist focused on detecting hate in infomedics; *robinthede* and *GeorgeTakei* are writers and famous comedians who published some viral jokes about this context; and *thehill*, which is an American newspaper and political journalism website published in Washington D. C. since 1994. All these accounts were really active with the Health Care proposal (both supporting or opposing it), with more than 9 million followers as a whole. This result suggests that the information spread by these users should be carefully analyzed mainly due to the high impact of diffusion.

6 Conclusions

In this paper, an efficient and effective GRASP algorithm for solving the BIMP has been presented. Three different diffusion models are used for BIMP considering the probabilistic algorithm Monte Carlo for the evaluation of the objective function.

Three greedy heuristic functions have been proposed for generating the initial solutions of GRASP. The first one is based on the two-step neighborhood, recently published in a problem of the same family, with the aim of analyzing how it adapts to a similar problem. The second one is based on the degree of each node and, finally, a new heuristic that penalizes those nodes whose neighbors have been already selected as a seed node is proposed, with the aim of expanding to new regions in the graph. Furthermore, the idea of using local information allows the algorithm to construct a complete solution in small computing time.

The local search method proposed is based on a new move named *Replace*, whose objective is to remove a seed node replacing it with the most promising ones. To make a scalable algorithm, and to avoid an exhaustive search which is not suitable for this problem, the local search is limited and can be configured according to the time requirements.

Comparing the presented GRASP algorithm versus the best algorithm found in the state of the art, GRASP obtained the best solution in 81 out of 84 available instances by requiring half of the computing time. The results reported are supported by the well-known pairwise Wilcoxon statistical test, confirming the superiority of the proposal with respect to the classical and state-of-the-art solution procedures for the BIMP.

Finally, an infodemic case study is analyzed from the influence maximization perspective. Specifically, an instance is built based on 386384 tweets about the American Health Care Act (AHCA). An experiment is performed, showing the superiority of GRASP when comparing it with ComBIM in 21 out of 27 available instances. The most influential users are analyzed, showing their relevance in the topic studied, being most of them senators, comedians, writers, or newspapers.

Acknowledgements The authors acknowledge support from the Spanish Ministry of "Ciencia, Innovaciòn MCIN/AEI/10.13039/501100011033/FEDER, UE) under grant ref. PID2021-126605NB-I00 and PID2021-125709OA-C22, the "Comunidad de Madrid" and "Fondos Estructurales" of the European Union with grant reference S2018/TCS-4566.

Data availability Data availability is public at https://grafo.etsii.urjc.es/BIMP/.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

References

Banerjee S, Jenamani M, Pratihar DK (2019) ComBIM: a community-based solution approach for the budgeted influence maximization problem. Expert Syst Appl 125:1–13. https://doi.org/10.1016/j.eswa.2019.01.070

Banerjee S, Jenamani M, Pratihar DK (2020) A survey on influence maximization in a social network. Knowl Inf Syst 62:3417–3455. https://doi.org/10.1007/s10115-020-01461-4

Barabási A-L, Pòsfai M (2016) Network science. Cambridge University Press, Cambridge

Bello-Orgaz G, Hernandez-Castro J, Camacho D (2017) Detecting discussion communities on vaccination in twitter. Future Gener Comput Syst 66:125–136. https://doi.org/10.1016/j.future.2016.06.032

Chen W, Wang C, Wang Y (2010) Scalable influence maximization for prevalent viral marketing in large-scale social networks. In: Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining—KDD '10. ACM Press, Hoboken. https://doi.org/10.1145/1835804.1835934

Chen E, Lerman K, Ferrara E (2020) Tracking social media discourse about the covid-19 pandemic: development of a public coronavirus twitter data set. JMIR Public Health Surveill 6(2):e19273. https://doi.org/10.2196/19273

Feo TA, Resende MGC (1989) A probabilistic heuristic for a computationally difficult set covering problem. Oper Res Lett 8(2):67–7. https://doi.org/10.1016/0167-6377(89)90002-3

Feo TA, Resende MGC, Smith SH (1994) A greedy randomized adaptive search procedure for maximum independent set. Oper Res 42(5):860–878. https://doi.org/10.1287/opre.42.5.860

Gong M, Song C, Duan C, Ma L, Shen B (2016) An efficient memetic algorithm for influence maximization in social networks. IEEE Comput Intell Mag 11(3):22–33. https://doi.org/10.1109/mci.2016.2572538

Goyal A, Lu W, Lakshmanan Laks VS (2011) CELF++. In: Proceedings of the 20th international conference companion on World wide web—WWW '11. ACM Press, Hoboken. https://doi.org/10.1145/1963192.1963217

Granovetter M (1978) Threshold models of collective behavior. Am J Sociol 83(6):1420–1443

Guney E, Cakir V, Ozdemir Y, Duzdar I (2015) Budgeted influence maximization in social networks with independent cascade diffusion model. In: Proceedings of the 4th international symposium & 26th national conference on operational research, pp 291–296

Han S, Zhuang F, He Q, Shi Z (2014) Balanced seed selection for budgeted influence maximization in social networks. In: *Advances in knowledge discovery and data mining*, pp 65–77. Springer International Publishing, Berlin. https://doi.org/10.1007/978-3-319-06608-0_6

Hansen P, Mladenović N (2006) First vs. best improvement: an empirical study. IV. ALIO/EURO workshop on applied combinatorial optimization. Discrete Appl Math 154(5):802–817. https://doi.org/10.1016/j.dam.2005.05.020

Hayes JL, Britt BC, Evans W, Rush SW, Towery NA, Adamson AC (2021) Can social media listening platforms' artificial intelligence be trusted? Examining the accuracy of crimson hexagon's (now brandwatch consumer research's) ai-driven analyses. J Advert 50(1):81–91. https://doi.org/10.1080/00913367.2020.1809576

Jos N-V, del Mar G-PM, Villar-Rodríguez G, Martín A, Camacho D (2023) Disinformation and vaccines on social networks: behavior of hoaxes on twitter. Rev Latina Comun Soc 81:44–62

Kempe D, Kleinberg J, Tardos E (2015) Maximizing the spread of influence through a social network. Theory Comput 11(1):105–147. https://doi.org/10.4086/toc.2015.v011a004

Kempe D, Kleinberg J, Tardos É (2003) Maximizing the spread of influence through a social network. In: *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining—KDD '03*. ACM Press, Hoboken. https://doi.org/10.1145/956750.956769

Leskovec J, Kleinberg J, Faloutsos C (2007) Graph evolution. ACM Trans Knowl Discov Data 1(1):2. https://doi.org/10.1145/1217299.1217301

Lozano-Osorio I, Sánchez-Oro J, Duarte A, Cordón Ó (2021) A quick GRASP-based method for influence maximization in social networks. J Ambient Intell Hum Comput 1:1. https://doi.org/10.1007/s12652-021-03510-4

Nguyen H, Zheng R (2013) On budgeted influence maximization in social networks. IEEE J Sel Areas Commun 31(6):1084–1094. https://doi.org/10.1109/jsac.2013.130610

Rafael M, Anna M-G, Jesús S-O, Abraham D (2018) Tabu search for the dynamic bipartite drawing problem. Comput. Oper. Res. 91:1–12

Resende Mauricio GC, Martí R, Gallego M, Duarte A (2010) GRASP and path relinking for the max-min diversity problem. Comput Oper Res 37(3):498–508. https://doi.org/10.1016/j.cor.2008.05.011

Resende Mauricio GC, Ribeiro Celso C (2013) GRASP: Greedy randomized adaptive search procedures. In: Search methodologies, pp 287–312. Springer, New York. https://doi.org/10.1007/978-1-4614-6940-7_11

Reza Z, Ali AM, Huan L (2014) Social media mining: an introduction. Cambridge University Press. Cambridge. https://doi.org/10.1017/CBO9781139088510

Richardson M, Agrawal R, Domingos P (2003) Trust management for the semantic web. In: Lecture notes in computer science, pp 351–368. Springer, Berlin Heidelberg. https://doi.org/10.1007/978-3-540-39718-2_23

Stanley W, Katherine F (1994) Social network analysis. Cambridge University Press, Cambridge. https://doi.org/10.1017/cbo9780511815478

Tretiakov A, Martín A, Camacho D (2022) Detection of false information in spanish using machine learning techniques. In: Hujun Y, David C, Peter T (eds) Intelligent data engineering and automated learning—IDEAL 2022, pp 42–53. Springer International Publishing, Cham

Wenzheng X, Weifa L, Lin Xiaola Yu, Jeffrey X (2016) Finding top-k influential users in social networks under the structural diversity model. Information Sciences 355–356:110–126. https://doi.org/10.1016/j.ins.2016.03.029

Wrubel L, Littman J, Bonnett W, Kerchner D (2020) gwu-libraries/tweetsets: Version 1.1.1. https://zenodo.org/record/1289426

Chapter 8

Dynamic Path Relinking for the Target Set Selection problem

The journal paper associated to this part is:

- Lozano-Osorio, I.; Oliva-Garcia, A. and Sánchez-Oro, J. Dynamic Path Relinking for the Target Set Selection problem. Knowledge-Based Systems (2023). `https://doi.org/10.1016/j.knosys.2023.110827` [114]
- Status: **Published**.
- Submitted: 12 April 2023.
- Accepted: 20 July 2023.
- Published: 1 August 2023.
- Impact Factor (JCR 2022): 8.800.
- Subject Category: Computer Science, Artificial Intelligence - 19/145 (Q1).

It is worth mentioning that this research has also been presented in the following conference:

- Lozano-Osorio, I.; Oliva-García, A.; Sánchez-Oro, J. Path Relinking Dinámico para el problema Target Set Selection. In XL Congreso Nacional de Estadística e Investigación Operativa (SEIO). 7-10 Nov 2023, Elche, Spain.

Knowledge-Based Systems 278 (2023) 110827

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Dynamic Path Relinking for the Target Set Selection problem

Isaac Lozano-Osorio, Andrea Oliva-García, Jesús Sánchez-Oro [*]

Department of Computer Science and Statistics, University Rey Juan Carlos, C/Tulipán, S/N, Móstoles, 28933, Spain

A R T I C L E   I N F O

Article history:
Received 12 April 2023
Received in revised form 18 July 2023
Accepted 20 July 2023
Available online 1 August 2023

Dataset link: https://grafo.etsii.urjc.es/TSS/

Keywords:
Target Set Selection problem
Influence maximization
Dynamic Path Relinking
GRASP
Social networks
Metaheuristics

A B S T R A C T

This research proposes the use of metaheuristics for solving the Target Set Selection (TSS) problem. This problem emerges in the context of influence maximization problems, in which the objective is to maximize the number of active users when spreading information throughout a social network. Among all the influence maximization variants, TSS introduces the concept of reward of each user, which is the benefit associated to its activation. Therefore, the problem tries to maximize the reward obtained among all active users by selecting an initial set of users. Each user has also associated an activation cost, and the total sum of activation costs of the initial set of selected users cannot exceed a certain budget. In particular, two Path Relinking approaches are proposed, comparing them with the best method found in the state of the art. Additionally, a more challenging set of instances are derived from real-life social networks, where the best previous method is not able to find a feasible solution. The experimental results show the efficiency and efficacy of the proposal, supported by non-parametric statistical tests.

1. Introduction

The evolution of Social Networks has been in continuous growth in the last years. Nowadays, almost everyone frequently uses one or more Social Networks for both posting and gathering information. Due to the relevance of Social Networks in several contexts, such as politics, marketing, or disease control, among others, scientists and practitioners have put all their efforts in designing and developing algorithms to automatically analyze and collect the most relevant information from them.

Among all the different problems that emerge from Social Network Analysis, this research is focused on Influence Maximization Problems. This is a large family of hard combinatorial optimization problems where it is necessary to select a set of users from a Social Network with the aim of maximizing the spread of information throughout the network. In particular, this study is focused on the Target Set Selection Problem in which, given a certain budget, it is necessary to select a subset of users whose total cost is smaller than the given budget, with the aim of maximizing the reach of information dissemination through the network.

In the context of Target Set Selection Problem (TSS), there are two main variants: guaranteeing reaching the complete network (or even a certain part of it) with the minimum number of initial users or maximizing the number of users reached while not exceeding an initial budget. This proposal is focused on solving the latter, which is usually named Max-TSS.

Although the main application of this problem is to increase the impact of advertising a product for a company [1,2], there are several applications in different fields. For instance, in politics, the Max-TSS can be used for reducing the impact of fake news or misinformation from two opposite approaches: identifying the individuals which are mostly spreading fake news through the network, or to boost those individuals which are transmitting reliable information [3]. Even more, this application is closely related to disease control, since it has been proven that the diseases spreads following the same model as information through Social Networks [4].

In this work, a metaheuristic algorithm based on a combination of Greedy Randomized Adaptive Search Procedure and Path Relinking is presented with the aim of providing high-quality solutions for the Max-TSS in reasonable computing times when considering large real-life Social Networks. Metaheuristics were originally defined by [5] as "a high-level problem-independent algorithmic framework that provides a set of guidelines or strategies to develop heuristic optimization algorithms". In the last decades, metaheuristics have become one of the most extended approximate type of algorithms for solving hard and complex combinatorial optimization problems. These are robust algorithms which are able to provide high-quality

* Corresponding author.
E-mail addresses: isaac.lozano@urjc.es (I. Lozano-Osorio), andrea.oliva@urjc.es (A. Oliva-García), jesus.sanchezoro@urjc.es (J. Sánchez-Oro).
URLs: https://grafo.etsii.urjc.es/en/author/isaac-lozano-osorio/ (I. Lozano-Osorio), https://grafo.etsii.urjc.es/en/author/andrea-oliva-garcia/ (A. Oliva-García), https://jesussanchezoro.github.io/ (J. Sánchez-Oro).

solutions in reasonable computing times. However, they cannot guarantee the optimality of the solution. Therefore, they are recommended when it is not possible to compute the optimal solution with an exact algorithm. This algorithm is tested with large-scale graphs derived from real social networks, analyzing the contribution of each part of the algorithm. This analysis will allow researchers and practitioners to select and adapt some parts of the proposed algorithm to other related problems. One of the main issues when dealing with Target Set Selection consists of getting stuck in local optima. The algorithm based on Dynamic Path Relinking proposed in this research is able to escape from local optima by creating a path between two high-quality solutions, which will lead the algorithm to explore a wider portion of the search space. In particular, Dynamic Path Relinking will be able to identify the most promising subsets of influential users and strategically combine them in a path to generate new diverse and high-quality solutions.

The main contributions of this research can be summed up as follows:

- A constructive procedure based on a novel heuristic is proposed, being able to generate feasible solutions in small computing times.
- The proposed local search is optimized by reducing the evaluation of the objective function, which is the most computationally demanding part of the algorithm.
- Dynamic Path Relinking is proposed with the aim of creating paths between high-quality solutions and compared with the classical Static Path Relinking, Simulated Annealing and Cost-Effective Forward selection.
- The dataset of instances has been extended with real-life networks where the previous methods are not able to provide a solution in reasonable computing times.

Rest of the manuscript is structured as follows. Section 2 shows the related work with Target Set Selection Problem; Section 3 formally defines the Target Set Selection Problem; Section 4 thoroughly describes the algorithmic proposal and all the components designed; Section 5 presents a detailed computational experimentation to evaluate the contribution of each component and test the proposal with the best previous algorithm; and, finally, Section 6 draws some conclusions derived from the research, as well as highlights some future research lines.

2. Related work

In this section, we introduce some related work about TSS as well as a brief survey of existing methods for solving this problem, either based on exact methods, approximation methods, heuristics or computational intelligence algorithms.

Richardson et al. [6] initially formulated the problem of selecting target nodes in SNs. The TSS was originally proposed in [7], where it was proven to be $\mathcal{NP}$-hard, and the authors proposed a polynomial-time approximation algorithm for a probabilistic variant of the problem.

Kempe et al. [7] proposed three influence diffusion models that play an important role in understanding the diffusion phenomenon: the independent cascade model (ICM), the weighted cascade model (WCM), and the linear threshold model (LTM). In subsequent researches, several proofs of $\mathcal{NP}$-hardness were proposed [4,8–11], all of them supported by approximation results for specific types of network topologies [12,13].

The research on Social Network Influence problems has been focused on finding exact methods under restricted conditions. In particular, the TSS has been tackled from both exact [14–17], and heuristic perspectives [18,19].

The fact that the TSS problem when considering LTM as a diffusion model can be described as a hard combinatorial optimization problem has attracted the attention of both academic and practitioners. In fact, the problem can be stated as an Integer Linear Programming (ILP) problem, which is able to solve small problem instances. Two models have been proposed: a time-dependent ILP [20], which derives instantly from the definition, and a time-independent one [14].

While the previous results are certainly of interest, allowing researchers to extract information about graph properties which are a key part of the complexity of the problem, all of them deal with exactly solving the problem. This also holds for variations of the problem, such as the latency-bounded TSS (which aims to activate all the nodes of a graph in a bounded number of rounds), for which recent research is focused on exact methods for specific cases or making use of certain properties [21–23].

Additionally, some variants with specific constraints derived from real applications have also been considered, tackled with evolutionary metaheuristics [24,25]. Swarm intelligence algorithms have been applied to a multi-objective problem dealing with maximizing the spread of influence of a set while minimizing its size [26]. An evolutionary algorithm (EA) was applied in [19] to a variant of the TSS problem that was tackled in this research.

Ravelo et al. [24] proposed a new TSS variant denoted as the maximum effortreward GAP Target Set Selection problem (Max-TSS), a new $\mathcal{NP}$-hard version. To the best of our knowledge [27,28], the best approach for solving the Max-TSS is based on binary linear models and Lagrangian relaxations, which are solved by dynamic programming algorithms [19]. With the aim of improving the bounds, authors embed the dynamic programming algorithms in subgradient methods, which are used to generate feasible solutions for the problem. In the research, the authors highlight that their heuristic is able to reach near-optimal solutions in almost all the considered instances. Although the approach is able to find optimal solutions, its main drawback relies on the size of real-life networks. In the original work, authors optimally solved instances up to 58 nodes, while current social networks usually have more than 1000 nodes. With the aim of evaluating the limits of the exact proposal, this work tests the previous exact approach with graphs derived from real social networks such as Twitch or LastFM, among others.

Several works stated that metaheuristics scarce in the Social Network Influence Problems [25,27]. The use of Path Relinking in Graph Theory and Network Science has led to several successful research in the last years [29–31].

3. Problem definition

As it was aforementioned, the objective of the Target Set Selection Problem (TSS) is to find the most influential nodes in a social network. Let us define a social network as a graph $G = (V, E)$ where the set of vertices V, with $|V| = n$, represents the users of the social network and the set of edges E, with $|E| = m$, is conformed with pairs (u, v), with $u, v \in V$, indicating that there is any kind of relation between users u and v. Then, given a social network modeled as a graph $G = (V, E)$ and a maximum budget K, the TSS problem tries to find a subset of users $S \subseteq V$ whose effort does not exceed the maximum value K, with the aim of maximizing the number of influenced users in the social network. Since the concept of influencing can be ambiguous, it is necessary to perform some initial definitions to clarify it.

First of all, it is necessary to define how a node can potentially influence another one. In the context of TSS, the influence of a user follows a rational influence function $\psi : V \times V \rightarrow [0, 1]$ over every pair of users. This function measures the influence of a user over another one. Notice that if two users u, v are not connected, then $\psi(v, u) = 0$. Then, a set of activated users S^t influences a non-activated user u if and only if the sum of the potential influence of all users in S^t is larger than or equal to 1, i.e., $\sum_{v \in S^t} \psi(v, u) \geq 1$. This rational influence function was given by several authors [8–10,19] and it tries to model a behavior in which if several users simultaneously influence a certain user, then it will be activated. The specific value of the function is determined by [19], and it is based on the number of interactions among users, i.e., the larger the number of interactions, the larger the function value. We denote S^t as the set of activated users in a certain iteration t. This model suggest a different approach to the well-known Influence Diffusion Models which requires from smaller computing times than probabilistic methods which are rather time consuming to obtain robust solutions.

Having defined the process of influencing a node given a set of activated nodes, it is necessary to define the influence propagation process. Starting from a set of initially activated nodes S, this process consists of activating all those non-activated nodes which are influenced by the activated ones. Without loss of generality, a solution for the TSS is given by the set of initially activated nodes S. Notice that this process is iteratively applied until no new nodes are activated. Given a step t and a set of activated nodes S^t, the set of nodes which are activated after applying a single iteration of the influence propagation process is denoted as S^{t+1}. The influence propagation process then stops when no new nodes are activated, i.e., $S^t = S^{t-1}$.

In TSS, each node has an associated effort to activate it, which is defined by the function $\alpha : V \rightarrow \mathbb{Z}^+$, as well as a reward obtained when it is influenced or initially activated which is defined by $\beta : V \rightarrow \mathbb{Z}^+$. Thus, the effort α is only considered for those nodes which are part of the set of the initially activated users S, and the reward β is earned whether a user u is initially activated or subsequently activated by the set of activated users. Another fact in TSS is that, when a node is activated at step t, it remains activated through the entire influence propagation process. Let us denote x_v^t as a binary variable which takes the value 1 if the node v is activated at step t and 0 otherwise (we refer the reader to [19] to a more detailed description of the model). Then,

$$x_v^{t-1} \leq x_v^t, \quad \forall v \in V, 1 \leq t \leq T$$

where T indicates the maximum number of steps in the influence propagation process, i.e., number of iterations in which new nodes are activated.

A solution S for the TSS is feasible if and only if the sum of the efforts of the initially activated nodes is smaller than or equal to K, which is a constraint of the problem, i.e., $\sum_{v \in V} \alpha(v) \cdot x_v^0 \leq K$. The objective function for the TSS is then evaluated as the sum of rewards obtained by the active nodes in the last iteration of the influence propagation process. In mathematical terms,

$$TSS(S) = \sum_{v \in V} \beta(v) \cdot x_v^T$$

Notice that evaluating the TSS is a computationally demanding procedure. In particular, the computational complexity of this evaluation is $O(n^2)$ since it is necessary to traverse all the nodes and, for each new activated node, it is required to traverse again the set of nodes searching for new potential nodes to be activated. The TSS then seeks for a solution $S^\star$ with the maximum objective function value. More formally,

$$S^\star \leftarrow \underset{S \in \mathbb{SS}}{\arg\max}\, TSS(S)$$

where $\mathbb{SS}$ represents the set of all feasible solutions, i.e., all possible combination of nodes whose sum of effort is smaller than or equal to K.

Fig. 1 depicts an example of the complete influence propagation process over a network with 5 nodes and 10 edges. The solution under evaluation is conformed by nodes C and E, i.e., $S = \{\mathtt{C}, \mathtt{E}\}$. Without loss of generality, let us suppose that the sum of costs $\alpha(\mathtt{C}) + \alpha(\mathtt{E})$ is smaller than or equal to K.

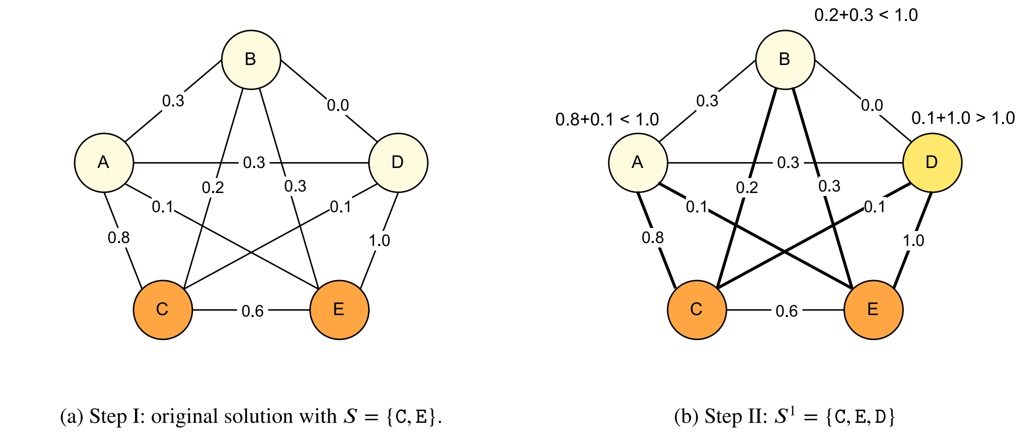
Fig. 1(a) shows the initial step $t = 0$, where the activated nodes are the ones initially selected C and E, i.e., $S = \{\mathtt{C}, \mathtt{E}\}$. Then, in the first iteration of the influence propagation process, depicted in Fig. 1(b), it is evaluated whether non-influenced nodes A, B, and D are influenced or not. Starting with node A, it is necessary to evaluate $\psi(\mathtt{C}, \mathtt{A}) + \psi(\mathtt{E}, \mathtt{A}) = 0.8 + 0.1 = 0.9 < 1.0$. Therefore, node A is not influenced in this step. A similar evaluation is performed with node B, resulting in $\psi(\mathtt{C}, \mathtt{B}) + \psi(\mathtt{E}, \mathtt{B}) = 0.2 + 0.3 = 0.5 < 1.0$, indicating that node B is not activated. Finally, when performing the evaluation of node D, we obtain $\psi(\mathtt{C}, \mathtt{B}) + \psi(\mathtt{E}, \mathtt{B}) = 0.1 + 1.0 = 1.1 \geq 1.0$. Then, after the first iteration, node D is included in the set of activated nodes, resulting in $S^1 = \{\mathtt{C}, \mathtt{D}, \mathtt{E}\}$.

Since $S^0 \neq S^1$, it is necessary to continue with the influence propagation process. If we now evaluate the second iteration, Fig. 1(c), starting with node A, $\sum_{v \in S^1} \psi(v, \mathtt{A}) = 0.8 + 0.1 + 0.3 = 1.2 \geq 1.0$. Then, node A will be included in the set of activated nodes in the next iteration, S^2. In the case of node B, the evaluation is $\sum_{v \in S^1} \psi(v, \mathtt{B}) = 0.2 + 0.3 + 0.0 = 0.5 < 1.0$, so the node B is not activated. After this iteration, $S^2 = \{\mathtt{A}, \mathtt{C}, \mathtt{D}, \mathtt{E}\} \neq S^1$, so an additional iteration is required. Fig. 1(d) illustrates the last iteration of the influence propagation process. In this case, it is only required to evaluate node B, resulting in $\sum_{v \in S^2} \psi(v, \mathtt{B}) = 0.3 + 0.2 + 0.3 + 0.0 = 0.8 < 1.0$. Therefore, node B is not activated. Since no new nodes are included in the set of activated nodes, i.e., $S^2 = S^3$, the influence propagation process stops in this iteration, returning the reward associated to the activated nodes.

4. Algorithmic approach

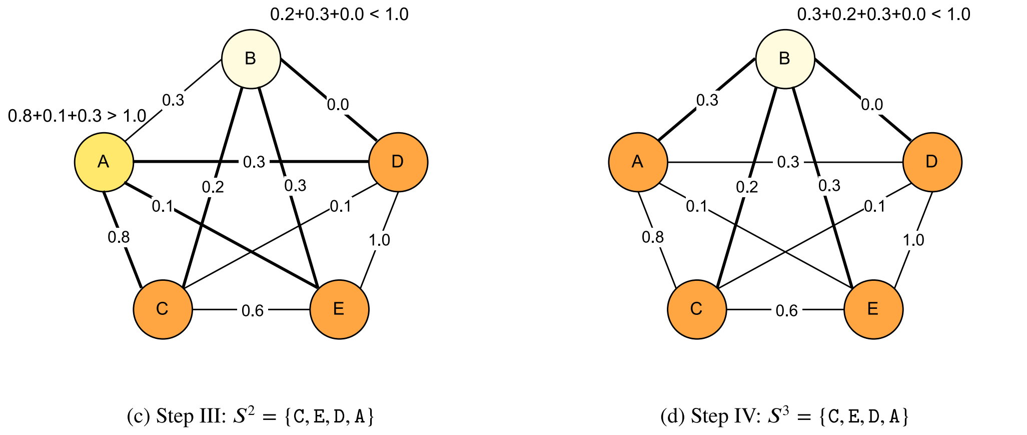
This work presents an algorithm based on Path Relinking (PR) [32] for solving the TSS problem. Path Relinking was originally presented as a framework for combining intensification and diversification strategies in the context of Tabu Search [33]. PR relies on the idea of connecting two high-quality solutions creating a path between them, with the expectation of finding promising solutions during the exploration of the path. The algorithm tries to include in the first solution, usually named as initial solution, attributes of the second solution, named as guiding solution. Both the initial and the guiding solutions present a high quality and, therefore, it is expected that the path created between them explores new promising regions of the search space. It has been traditionally combined with Greedy Randomized Adaptive Search Procedure (GRASP) since Laguna and Marti [34] adapted PR to increase the intensification phase of GRASP. There are two main PR strategies extended in the literature: Static PR and Dynamic PR. In this work, both strategies are tested in the context of TSS. First of all, it is necessary to design a specific path-creation method between two solutions in the context of TSS. Given two solutions S_i and S_g, which are initial and guiding solutions, the objective is to create a path from S_i to S_g by removing attributes from the initial solution which are not present in the guiding one, and replacing them with attributes which are in the guiding solution but not in the initial one.

The path-creation method designed for the TSS problem iteratively removes nodes belonging to S_i but not to S_g, i.e., $S_i \setminus S_g$, and includes nodes which are in S_g but not in S_i, i.e., $S_g \setminus S_i$. The process of selecting the node to be removed and included in each iteration can be performed randomly (Random Path Relinking), greedily (Greedy Path Relinking), or in a more elaborated manner



(a) Step I: original solution with $S = \{\mathtt{C}, \mathtt{E}\}$.

(b) Step II: $S^1 = \{\mathtt{C}, \mathtt{E}, \mathtt{D}\}$



(c) Step III: $S^2 = \{\mathtt{C}, \mathtt{E}, \mathtt{D}, \mathtt{A}\}$

(d) Step IV: $S^3 = \{\mathtt{C}, \mathtt{E}, \mathtt{D}, \mathtt{A}\}$

Fig. 1. Influence propagation process over a network with 5 nodes and 10 edges, considering the solution $S = \{\mathtt{C}, \mathtt{E}\}$.

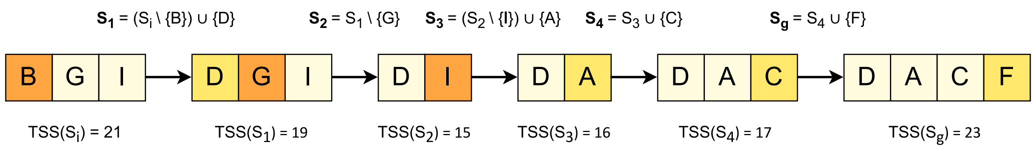


Fig. 2. Example of a path between solutions $S_i = \{\mathtt{B}, \mathtt{G}, \mathtt{I}\}$ and $S_g = \{\mathtt{D}, \mathtt{A}, \mathtt{C}, \mathtt{F}\}$.

(Greedy Randomized Path Relinking). Notice that both Greedy Path Relinking and Greedy Randomized Path Relinking are more computationally demanding than Random Path Relinking. This is mainly because they require to generate the complete set of feasible solutions in each step of the path, and also evaluate each one of them. Since the computational effort is a critical part of TSS, we have selected Random Path Relinking (RPR) which, additionally, increases the diversity of the search. Fig. 2 shows a possible path between the initial solution $S_i = \{\mathtt{B}, \mathtt{G}, \mathtt{I}\}$, with an objective function value of $TSS(S_i) = 21$, and the guiding solution $S_g = \{\mathtt{D}, \mathtt{A}, \mathtt{C}, \mathtt{F}\}$, with an objective function value of $TSS(S_g) = 23$.

The path starts by selecting the elements which need to be included during the path as $S_g \setminus S_i = \{\mathtt{D}, \mathtt{A}, \mathtt{C}, \mathtt{F}\}$, since S_i and S_g do not have any element in common. Then, it is required to select those elements which will be removed during the path creation, which are $S_i \setminus S_g = \{\mathtt{B}, \mathtt{G}, \mathtt{I}\}$. Then, in each iteration, an element is removed from the incumbent solution, and a new element is included in the solution if the maximum allowed budget K is not exceeded. Since Random Path Relinking is considered, the element to be removed and the one to be included is selected at random.

In the path created in the figure, the first step generates the solution $S_1 = (S_i \setminus \{\mathtt{B}\}) \cup \{\mathtt{D}\} = \{\mathtt{D}, \mathtt{G}, \mathtt{I}\}$, resulting in an objective function value of 19. In the next step, solution S_2 is created by removing node G, $S_2 = S_1 \setminus \{\mathtt{G}\} = \{\mathtt{D}, \mathtt{I}\}$, with an objective function value of 15. Notice that, in this case, no new element is included in the incumbent solution, assuming that, in this point, the available budget would be exceeded. Then, S_3 is generated as $S_3 = (S_2 \setminus \{\mathtt{I}\}) \cup \{\mathtt{A}\} = \{\mathtt{D}, \mathtt{A}\}$, with an objective function value of 16. At this point, there are no nodes to be removed from solution S_3, i.e., $S_3 \setminus S_g = \emptyset$. However, there are still nodes to be included, selecting in this stage node C, resulting in $S_4 = S_3 \cup \{\mathtt{C}\} = \{\mathtt{D}, \mathtt{A}, \mathtt{C}\}$ with an objective function value of 17. In the last step, the guiding solution is reached by including the last node F, finishing the path. It is worth mentioning that none of the solutions created during the path are necessarily a local optimum with respect to any neighborhood. Therefore, the local search method described in Section 4.3 is applied to the best solution found in the path (ties are broken randomly), excluding the initial and guiding solutions, i.e., S_1 in the figure. Once the path creation has been described, it is necessary to present the Path Relinking variants considered in this work. In particular, two of the most extended variants have been studied: Static Path Relinking (see Section 4.1) and Dynamic Path Relinking (see Section 4.2). The proposed algorithm will leverage the path creating of Path Relinking to select the most promising subsets of influential users of two different solutions and combine them to generate an eventually better set of influential users.

4.1. Static path relinking

Static Path Relinking (SPR) [35] is the most basic version of Path Relinking. SPR requires from an Elite Set (*ES*) of high quality and diverse solutions that can be generated either at random or by using a more elaborated procedure. In this problem, the *ES* is generated by using the GRASP algorithm presented in Section 4.3 which balances intensification and diversification. The number of solutions generated will be discussed later in the experimental results section (Section 5). Then, the *ES* is conformed with the best solutions found with GRASP. Again, the size of the *ES* is a parameter of the algorithm which will be adjusted in Section 5. Finally, all the solutions in the *ES* are combined with the Random Path Relinking method. Algorithm 1 shows the pseudocode of the method.

Algorithm 1 SPR($G = (V, E), \Delta, \delta$)

```
1:  $P \leftarrow \emptyset$
2:  for $i = 1 \dots \Delta$ do
3:      $S \leftarrow Construct(G)$
4:      $S' \leftarrow Improve(S)$
5:      if $S' \notin P$ then
6:          $P \leftarrow P \cup \{S'\}$
7:      end if
8:  end for
9:  $ES \leftarrow SelectBest(P, \delta)$
10: $S_b \leftarrow \text{argmax}_{S \in ES} TSS(S)$
11: for $i = 1 \dots \delta - 1$ do
12:     for $j = i + 1 \dots \delta$ do
13:         $S \leftarrow RPR(ES_i, ES_j)$
14:         $S' \leftarrow Improve(S)$
15:         if $TSS(S') > TSS(S_b)$ then
16:             $S_b \leftarrow S'$
17:         end if
18:     end for
19: end for
20: return $S_b$
```

The method starts by creating the initial population of solutions P (step 1). Then, SPR iterates until generating a set of Δ solutions (steps 2–8). In each iteration, a solution is generated (step 3) and then improved (step 4) using the constructive and local search methods presented in Section 4.3. The generated solution is then added to the initial population if and only if it has not been explored yet (steps 5–7).

The Elite Set *ES* is generated with the δ most promising solutions of P (step 9), and the best solution found is initialized (step 10). Then, all the solutions in the *ES* are combined using the Random Path Relinking method (step 13). The combined solution is later improved (step 14) and compared with the best solution found so far, updating it if necessary (steps 15–17). The method ends returning the best solution found during the search (step 20).

The analysis of computational complexity of SPR can be divided into two different phases: the one corresponding to GRASP phase (steps 2–8) and the one corresponding to the path creation itself (steps 11–19). The complexity of the GRASP phase is detailed in Section 4.3. The complexity of the second phase is evaluated as $O\left(\delta \cdot \delta \cdot \left(n + n^3 \log n + n^2 + n^2\right)\right)$. In particular, the first two δ factors indicate the loops in steps 11 and 12. Then, the complexity of RPR is $O(n)$ since, in the worst case (two completely different solutions), it will perform n iterations. Then, the complexity of the local search is $O\left(n^3 \log n\right)$ as stated in Section 4.3 and, finally, the complexity of comparing two solutions is $O\left(n^2\right)$ for each solution evaluation. Therefore, the final complexity of this method is $O\left(\delta^2 \cdot n^3 \log n\right)$.

4.2. Dynamic path relinking

Dynamic Path Relinking (DPR) [36] avoids the generation of a complete population of solutions and then combine them by dynamically creating new solutions and paths between them. In particular, the method starts by creating the *ES* with a fixed number of solutions created with GRASP, which is in continuous evolution during the process. Algorithm 2 details the proposed Dynamic Path Relinking method.

Algorithm 2 DPR($G = (V, E), \Gamma, \gamma$)

```
1:  $ES \leftarrow \emptyset$
2:  for $i = 1 \dots \Gamma$ do
3:      $S \leftarrow Construct(G)$
4:      $S' \leftarrow Improve(S)$
5:      if $S' \notin ES$ then
6:          $ES \leftarrow ES \cup \{S'\}$
7:      end if
8:  end for
9:  for $i = 1 \dots \gamma$ do
10:     $S \leftarrow Construct(G)$
11:     $S' \leftarrow Improve(S)$
12:     $PS \leftarrow \{S'\}$
13:     for $S_{ES} \in ES$ do
14:         $S_C \leftarrow RPR(S_{ES}, S')$
15:         $S'_C \leftarrow Improve(S_C)$
16:         $PS \leftarrow PS \cup \{S'_C\}$
17:     end for
18:     $UpdateES(ES, PS)$
19: end for
20: $S_b \leftarrow \text{argmax}_{S \in ES} TSS(S)$
21: return $S_b$
```

Similarly to SPR, DPR starts by creating the Elite Set *ES* (steps 1–8). However, contrary to SPR, the *ES* is initialized with the first Γ solutions generated by applying the constructive and local improvement methods. Then, for a fixed number of iterations γ (which is an input parameter of DPR), a new solution S' is constructed and improved (steps 10–11). The set *PS* will contain all the new solutions explored during the current iteration, and it is initialized with solution S' (step 12). The method then iterates over every solution $S_{ES} \in ES$ (steps 13–17), creating a path from S_{ES} to the newly generated solution S' using RPR (step 14). The best solution found in the path is then improved and added to *PS* (steps 15–16). Once all the paths have been created, the method $UpdateES(ES, PS)$ tries to insert every generated solution into the Elite Set (step 18). In particular, if the solution under evaluation S_P is better than the worst solution found in the *ES*, then S_P is included in the *ES*, replacing the most similar solution to S_P of the *ES* among those with worse objective function value than S_p (i.e., the one with the minimum distance to S_P).

In this point, it is important to define a distance metric between two solutions. Since the solution representation for the TSS is a set of nodes, the distance between two solutions S_1 and S_2 is measured as the number of different nodes. More formally,

$$d(S_1, S_2) = |S_1 \setminus S_2| + |S_2 \setminus S_1|$$

The algorithm ends with the best solution included in the Elite Set, which is also the best solution found in the search (step 21).

Having described the proposal, it is interesting to highlight the dynamic feature of this variant of Path Relinking. In the case of SPR, the initial set of solutions is generated and then combined, but the new solutions found in the path are never considered neither as initial nor guiding solutions. However, DPR dynamically generates new solutions which are considered in future paths, thus leading to a wider exploration of the search space. Section 5 will detailedly analyze the effect of this dynamic feature on the quality of the solutions found.

The computational complexity of this method is evaluated similarly to SPR. The first phase, which generates the ES, has the same complexity as GRASP, $O\left(\Gamma \cdot n^3 \log n\right)$. In the second phase, the dynamic feature of DPR increases the complexity since the local search is performed in each iteration, resulting in a final complexity of $O\left(\gamma \cdot \left(n + n^3 \log n + \Gamma \cdot \left(n + n^3 \log n\right)\right)\right)$, which can be reduced to $O\left(\gamma \cdot \Gamma \cdot \left(n^3 \log n\right)\right)$.

4.3. GRASP

Path Relinking requires from a method to generate high-quality and diverse solutions in order to create promising paths during the search in both static and dynamic variants. Although these solutions can be generated at random, it has been experimentally shown in several works that designing a specific constructive and local improvement method for the problem under consideration usually leads to better results [29–31,37]. In the context of influence maximization problems, the Greedy Randomized Adaptive Search Procedure (GRASP) has been shown to be an effective and efficient method to generate them [38].

GRASP is a trajectory-based metaheuristic originally proposed in [39], and formally defined in [40]. The metaheuristic is divided into two well-differenced phases: construction and local improvement. The key part of GRASP is the construction method, which is able to generate not only high-quality solutions but also diverse ones. Then, the local improvement method is responsible for finding a local optimum with respect to the initial solution. These two phases are iteratively applied until a certain termination criterion is reached, which is usually a maximum number of iterations. Notice that the computational complexity of GRASP directly depends on the complexity of the constructive procedure and the local search method, resulting in the maximum complexity between both of them.

Fig. 3 shows a general view of the main advantages of this metaheuristic. In particular, Fig. 3(a) shows the performance of a completely greedy algorithm, including the local search method. In this case, solution S_1 is generated and, then, the local search is able to reach a local optimum with respect to the considered neighborhood. However, the method stagnates in the local optimum, and it is not able to escape from it.

The advantage of GRASP is shown in Fig. 3(b). In this case, the construction phase of GRASP generates seven diverse and high-quality solutions instead of a single one. The diversification of GRASP increases the probability of reaching different regions of the search space. In the graphical example, this diversification finds solutions S_6 and S_7 which are not the best initial solutions (indeed, S_6 is the worst initial solution in terms of quality), but the application of the local search method ends in a better solution than the one found with the greedy approach.

Constructive method

The constructive method proposed for TSS problem follows the GRASP philosophy of diversification by avoiding totally greedy decisions. Algorithm 3 shows the pseudocode of the proposed method.

Algorithm 3 Construct($G = (V, E), K, \omega$)

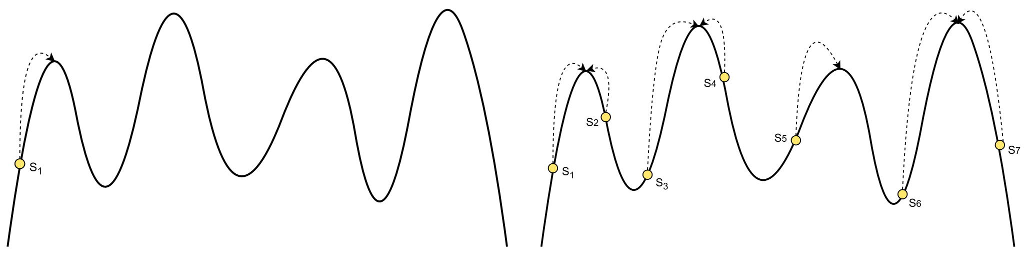
```
1:  $v \leftarrow Random(V)$
2:  $S \leftarrow \{v\}$
3:  $CL \leftarrow V \setminus \{v\}$
4:  while $\sum_{v \in S} \alpha(v) \leq K$ and $CL \neq \emptyset$ do
5:      $g_{\min} \leftarrow \min_{c \in CL} g(c, S)$
6:      $g_{\max} \leftarrow \max_{c \in CL} g(c, S)$
7:      $\mu \leftarrow g_{\max} - \omega \cdot (g_{\max} - g_{\min})$
8:      $RCL \leftarrow \{c \in CL : g(c) \geq \mu \wedge \sum_{v \in S} \alpha(v) + \alpha(c) \leq K\}$
9:      $v \leftarrow Random(RCL)$
10:     $S \leftarrow S \cup \{v\}$
11:     $CL \leftarrow CL \setminus \{v\}$
12: end while
13: return $S$
```

The algorithm requires from three input parameters: the input SN, $G = (V, E)$; the maximum allowed budget, K, and the parameter that controls the greediness/randomness of the search, ω. Notice that in the GRASP literature this parameter is usually referred as α. However, we have changed the notation to avoid confusion with the effort of a node, which is named as α.

With the aim of increasing diversity, the method selects the first node to be included at random from the set of users V (step 1), initializing the solution under construction S (step 2). Then, the candidate list CL is created with all the nodes but v (step 3). The constructive method iteratively adds a node to the solution while the budget is not exceeded and the candidate list is not empty (steps 4–12). In each iteration, the minimum and maximum value of a certain greedy function are computed (steps 5–6). The aim of the greedy function is to evaluate how promising a candidate is, and it is a key part of the constructive procedure. The proposed greedy functions will be described below. Then, a threshold μ is evaluated in order to establish a limit to consider whether a node is promising or not, which depends on the value of $g_{\min}$ and $g_{\max}$. Notice that the threshold completely depends on the value of the input parameter $\omega \in [0, 1]$. In particular, if $\omega = 0$, then $\mu = g_{\max}$ and the method becomes completely greedy, while if $\omega = 1$, then $\mu = g_{\min}$ and the method is totally random. Having this in mind, it is important to find a balance between randomness and greediness, so the value of this input parameter will be adjusted in the experiments (see Section 5.1). With this threshold, the restricted candidate list RCL is created (step 8), containing all the nodes whose greedy function value is larger than or equal to the threshold μ, considering that they do not exceed the maximum budget. Once the RCL is constructed, the next element is selected at random from it (since all the nodes in RCL are promising) to favor diversity (step 9). The selected node is then added to the incumbent solution (step 10), updating the CL by removing it (step 11). Finally, the method returns the constructed solution S (step 13). The computational complexity of this method is $O\left(n \cdot O(g)\right)$, where $O(g)$ indicates the complexity of the considered greedy function.

Having defined the constructive procedure, it is necessary to present the considered greedy functions. Specifically, two different greedy functions are evaluated in this research. The first greedy function is traditionally considered in the GRASP literature, and it consists of directly evaluating the objective function value if the node under evaluation were added to the incumbent solution. More formally,

$$g_{of}(c, S) \leftarrow TSS(S \cup \{c\})$$



(a) Example of a greedy algorithm. (b) Example of GRASP performance.

Fig. 3. Comparison between a greedy construction with local search and GRASP algorithm.

which has a computational complexity of $O\left(n^2\right)$ since it is directly the complexity of the objective function evaluation.

However, as it was stated in Section 3, the evaluation of the objective function for the TSS is a rather computationally demanding process, so a new greedy function is proposed with the aim of reducing the computational effort of the evaluation, since it will be performed in each iteration of the construction process.

The second greedy function proposed, named $g_{dg}(c, S)$, considers that the relevance of a node is directly proportional to its degree. In other words, if a node is connected to several nodes, then it will probably influence a large amount of its adjacent nodes. Then, this greedy function is evaluated as the degree of the evaluated node:

$$g_{dg}(c, S) \leftarrow |u \in V : (c, u) \in E|$$

with a computational complexity of $O(1)$ since it only requires to evaluate the degree of a node.

Therefore, the constructive method considering g_{of} has a computational complexity of $O\left(n^3\right)$, while considering g_{dg} reduces its complexity to $O\left(n\right)$. Both greedy functions will be tested in Section 5.1.

Local search

The solution generated with the constructive procedure is not necessarily a local optimum with respect to any neighborhood. For that reason, the second phase of GRASP consists of a local improvement method that finds a local optimum starting from the initial solution. In order to define a local search method, it is necessary to establish three main components: the move operator, the neighborhood explored, and the order in which it is explored.

Starting from the initial solution S, it is not possible to add new nodes, since the constructive procedure stops when the maximum budget is exceeded with any of the remaining nodes. Therefore, the proposed move operator is defined in two steps: remove and add. In particular, the move operator removes a node from the solution and then iteratively adds nodes until the maximum budget is reached:

$$move(S, u, V, K) = (S \setminus \{u\}) \cup \left\{ v \in V \setminus (S \cup \{u\}) : \sum_{s \in S \cup \{u\}} \alpha(s) + \alpha(v) \leq K \right\}$$

Having defined the move operator, the next step to propose a local search method is to define the neighborhood that will be explored during the search. In the case of TSS, the neighborhood is defined as the set of solutions that can be reached by performing a single move operator. More formally,

$$N(S) \leftarrow \{S' \leftarrow move(S, u, V, K) \quad \forall u \in S\}$$

Finally, it is necessary to indicate the order in which the neighborhood is explored. There are two main search strategies in local search methods: the first and best improvement. The former performs the first move that leads to an improvement in the current neighborhood, while the latter explores the complete neighborhood, performing the move that results in the best solution of the neighborhood. Best improvement is usually more computationally demanding than the first improvement, since it requires to explore the complete neighborhood in each iteration, although they have been shown to provide similar results for several combinatorial optimization problems [41,42].

In the context of TSS, the computational effort is a critical part of the algorithm, so we have decided to use the first improvement method with the aim of reducing the computing time to perform a local search method. With the aim of avoiding biasing the search, the neighborhood is explored at random, performing the first movement that results in a better solution.

To sum up, the local search method, denoted as Standard Local Search (SLS), follows a first improvement strategy based on a move operator which removes a node from the solution and replaces it with all the nodes that can be added without exceeding the allowed budget. The computational complexity of a single iteration of SLS is $O\left(|S| \cdot n \cdot n^2\right) = O\left(n^4\right)$ since for each element in S, it tries to perform the move operator with a non-selected node and, finally, it requires to evaluate the resulting solution to check if an improvement is found, which has a complexity of $O\left(n^2\right)$ as stated in Section 3.

With the aim of further reducing the computational effort of the local search method, three improvements are proposed, resulting in an Advanced Local Search (ALS). The first improvement tries to escape from cycling the search by avoiding the exploration of already visited solutions. In order to do so, each visited solution is associated with a unique number, i.e., hash code, evaluated following a hash function. Then, every time a solution is visited, it is evaluated if its corresponding hash code has not already been included in the set of visited solutions. If so, the method undoes the move and continues with the next iteration, avoiding repeating the exploration of the same region of the search space.

The second improvement is devoted to limit the nodes explored during the search, discarding those nodes which will result in an unfeasible solution. In order to do so, the candidate nodes to be added are sorted with respect to their effort value in ascending order. Then, only those nodes whose effort value is smaller than or equal to the available budget are explored. Additionally, to favor diversity, the exploration is performed at random among all nodes that satisfy this constraint.

The objective of the last improvement is to reduce the computing time required to evaluate the influence of a node by caching it. Specifically, the influence of a node (i.e., those nodes that are affected by its activation), is calculated at the beginning of the local search method. Then, every time a node is selected to be removed or added to the solution, the influence of that node over the other nodes of the graph is updated. As a result, it is not necessary to completely evaluate the objective function in each iteration but to check the corresponding pre-calculated influence.

In order to evaluate the optimization of ALS with respect to SLS, the computational complexity of ALS is calculated. In this case, the traversed nodes are sorted with a complexity of $O(n \log n)$ in each iteration. The last improvement reduces the complexity of evaluating a solution, from $O(n^2)$ to $O(n)$, since it only requires to traverse the nodes once for updating the value of ψ. Therefore, the final complexity of this method is $O(|S| \cdot n \log n \cdot n) = O(n^3 \log n)$. It is worth mentioning that this complexity refers to the worst case which, in the case of ALS, is hard to reach, since the second improvement limits the number of nodes explored during the search.

5. Results

This section is devoted to providing a detailed analysis of the performance of the proposed algorithm. In particular, the section is divided into two different subsets of experiments: preliminary and final. The former is designed to select the best configuration for the proposed algorithms in terms of components and parameter values, while the latter performs a competitive testing with the best algorithm found in the literature to analyze the efficiency and efficacy of the proposal. All experiments have been performed on an AMD EPYC 7282 16-core virtual CPU with 32 GB of RAM, using Java 17. All instances and source codes have been made publicly available at https://grafo.etsii.urjc.es/TSS.

The dataset used to perform the experiments has been derived from the best algorithm found in the literature to provide a fair comparison. This set of instances is conformed with 82 instances derived from real-life social networks which have been extensively used in social network analysis. The main drawback of this dataset is that the largest network is conformed with 58 nodes, which might not be challenging enough considering the current size of social networks. To mitigate this drawback, we have added 8 additional instances, with sizes from 67 to 10,312 nodes. The size of each instance is included in Table 8 in Appendix.

Given a node v, its effort $\alpha(v)$ and reward $\beta(v)$ is also given by the instance. The value of the effort and the reward for the original instances have been directly derived from the original dataset. For the new instances, we would like to thank the authors for kindly sending us the code [19] to calculate all the required parameters to generate the instance. Following this definition, the users with larger influence has a larger associated reward. Regarding the effort, those users which are easily influenced has a smaller effort.

5.1. Preliminary results

The preliminary experiments are designed to adjust the parameters of the proposal and to select the best configuration of elements to be included in the algorithm. The experiments have been designed to incrementally configure the algorithm following a sequential design. Although a full-factorial experimentation may better adjust the parameters, it has been shown that the results are usually equivalent [43].

With the aim of avoiding overfitting, the preliminary experiments are performed over a subset of 18 representative instances, which is a 20% of the complete set of 90 instances. All the experiments report the following metrics: Avg., the average objective function value; Dev. (%), the average deviation with respect to the best solution found in the experiment; Time (s), the average computing time required to execute the algorithm measured in seconds; and, # Best, the number of times that the algorithm reaches the best solution found in the experiment. Notice that in the tables in which the optimal value is known, # Best is replaced by # Optima.

Table 1
Comparison of the two proposed greedy functions when considering different values for ω.

Constructive	ω	Avg.	Dev. (%)	Time (s)	#Best
g_{of}	0.25	47.28	4604.56	400.07	15
	0.50	46.94	3985.67	400.04	15
	0.75	44.61	7259.96	400.03	14
	RND	45.00	5435.67	400.04	15
g_{dg}	0.25	3530.56	1.08	160.89	15
	0.50	3551.00	2.06	137.62	12
	0.75	3571.39	1.37	**121.09**	14
	RND	**3637.94**	**0.40**	122.42	**17**

Table 2
Comparison between the two proposed local search strategies.

Algorithm	Avg.	Dev. (%)	Time (s)	#Best
SLS	3546.22	0.95	1202.26	16
ALS	**3828.61**	**0.00**	**285.79**	**18**

The first experiment is designed to select the greedy function to be considered in the constructive phase of GRASP, as well as to select the best value for the ω parameter, which controls the greediness of the construction. In particular, greedy functions g_{of} and g_{dg} are tested, each one of them considering the values $\omega = \{0.25, 0.50, 0.75, RND\}$, with a fixed number of 100 iterations. Table 1 shows the results obtained by each combination of parameters. Notice that the maximum time allowed to each constructive is set to 3600 s.

The most remarkable thing about the results is that the greedy function based on the objective function value, g_{of}, consistently produces drastically worse results than the ones provided by g_{dg}. The rationale behind this is the inclusion of more challenging instances, in which g_{of} only generates a small number of solutions in the maximum allowed time, thus resulting in low-quality solutions.

If we now analyze the results obtained by g_{dg}, the computing time is clearly smaller than g_{of}, being able to solve all the instances without reaching the maximum allowed time. It is worth mentioning that the differences in computing time among the ω values are negligible. The best results in terms of quality are obtained with $\omega = RND$, with the smallest deviation (0.40%) and the largest number of best solutions found (17 out of 18). Therefore, we select g_{dg} and $\omega = RND$ for the constructive procedure.

The second experiment is designed for evaluating the performance of ALS with respect to SLS. Since the SLS is rather computationally demanding, the maximum allowed computing time is set to three hours per instance (10,800 s). Table 2 shows the results obtained when comparing both local search strategies.

Analyzing the quality of the local search methods, both are equivalent when considering the smallest instances. However, the differences appear when including the most challenging ones. In this case, SLS is not able to finish, so it is not able to reach the best solution in 2 out of 18 instances. Regarding the computing time, ALS is more than four times faster than SLS. Therefore, we select ALS for the remaining experiments as the local search method.

Having configured the constructive and local search methods, the next experiment is devoted to establishing the size of the initial population, Δ, and the Elite Set, δ, for the SPR. In order to do so, we have fixed $\delta = 10$ and vary Δ in the range [10, 50] with a step of 10. Table 3 shows the results obtained.

It can be derived from the results that even considering the smallest initial population of $\Delta = 10$ results in high-quality solutions. However, it can be seen how the efficacy of the algorithm

Table 3
Comparison of different sizes for the initial population of Static Path Relinking when fixing the Elite Set size to $\delta = 10$.

Δ	Avg.	Dev. (%)	Time (s)	#Best
10	3799.50	0.51	**86.61**	15
20	3845.00	0.59	94.93	15
30	3845.61	0.01	115.51	17
40	**3846.78**	**0.00**	165.64	**18**
50	**3846.78**	**0.00**	206.36	**18**

Table 4
Comparison of GRASP isolated and coupled with SPR to analyze the contribution of each part of the algorithm.

Algorithm	Avg.	Dev. (%)	Time (s)	#Best
GRASP	3637.94	1.97	**122.42**	15
SPR	**3846.78**	**0.00**	165.64	**18**

Table 5
Comparison between different size of the Dynamic Path Relinking Elite Set and the new solutions.

Γ	γ	Avg.	Dev. (%)	Time (s)	#Best
10	10	3842.78	25.05	**192.66**	16
10	20	**3846.78**	**0.00**	417.30	**18**
20	10	**3846.78**	**0.00**	411.66	**18**
20	20	**3846.78**	**0.00**	1098.50	**18**

increases with the size of the initial population, stagnating when $\Delta = 40$. In particular, $\Delta = 50$ provides the same results but requires almost 30% more computing time. Therefore, we select $\Delta = 40$ for the final configuration of SPR. We have performed the same experimentation for the value of δ fixing the initial population size to $\Delta = 40$, but there are no differences in quality among the different values. For this reason, we have selected an Elite Set size of $\delta = 10$.

Having selected the best GRASP configuration, i.e., constructive method using the greedy function g_{dg} and ALS as a local search, and the best SPR configuration, i.e., initial population size of 40, it is necessary to analyze the contribution of SPR to GRASP. In order to do so, an experiment is performed comparing the results obtained by GRASP isolated and then coupled with SPR. The results are shown in Table 4.

As expected, SPR is able to reach all the best solutions of the experiments, with a deviation of 0%. Furthermore, it does not considerably increase the computing time, being an efficient method for the TSS. On the contrary, GRASP fails to reach 3 out of 18 best solutions, remaining at a deviation of nearly 2% with respect to the solution found by SPR. These results confirms the contribution of SPR to the final algorithm.

The last preliminary experiment is designed to configure the dynamic variant of Path Relinking, DPR. In this case, there are two input parameters to adjust: Γ, the size of the Elite Set, and γ, the number of new solutions generated to combine with each solution of the Elite Set. The tested values are $\Gamma = \{10, 20\}$ and $\gamma = \{10, 20\}$, evaluating all possible combinations of these values. The results are shown in Table 5.

As expected, the computing time increases drastically with the size of the Elite Set and the initial population. However, the quality does not vary when considering pairs $(\Delta, \delta) = \{(10, 20), (20, 10), (20, 20)\}$. The combination $(10, 10)$ can be directly discarded due to the large deviation of 25% provided. Therefore, we select the configuration that provides the smallest computing time, which is $\Delta = 20$ and $\delta = 10$.

5.2. Final results

Since the dataset has been increased with more complex and challenging instances, it is necessary to establish a time limit for the exact algorithm. If the Gurobi solver reaches that time limit, it returns the best solution found so far, which is not necessarily the optimal value. In particular, the time limit given to the Gurobi solver is set to 108 000 s (approximately 30 h).

Table 6
Comparison of SPR, DPR, SA, CELF and Gurobi solver when considering the original dataset in which Gurobi is able to reach the optimal value.

Algorithm	Avg.	Dev. (%)	Time (s)	#Optimal
Gurobi	**45.38**	**0.00**	117.14	**82**
DPR	44.34	2.58	**0.01**	76
SPR	44.54	1.82	**0.01**	79
SA	46.31	**4.86**	**0.01**	76
CELF	42.07	12.19	**0.01**	61

The best previous approach is an exact method which shows its limits when dealing with larger and more complex instances. In order to evaluate the contribution of our proposal, we have also included an additional metaheuristic algorithm for performing a comparison with SPR and DPR. In particular, we have selected Simulated Annealing (SA), which is a metaheuristic based on the analogy between an optimization process and a thermodynamic process known as annealing. It is a search method which tries to escape from local optima allowing to explore worse solutions if those solutions satisfy certain criteria. SA was originally proposed by Kirkpatrick et al. [44] and it has been successfully applied in a wide variety of hard combinatorial optimization problems. SA has been successfully applied in several works related to influence maximization problems [45,46].

Additionally, the well-known Cost-Effective Lazy Forward (CELF) selection algorithm [47], which has been widely used in the context of influence maximization problems and, in particular, in TSS [48], is included in the comparison. CELF is a greedy procedure which leverages the submodularity property of the network to considerably reduce the computational effort of the greedy hill-climbing algorithm. The main objective of this optimization is to scale to large problems, reaching near optimal placements. This improvement makes CELF approximately 700 times faster than the original procedure.

There exists several implementation of SA which are publicly available. For this work, we have selected the one provided by the Metaheuristic Optimization FrameWoRK [49], which has been tested over several hard optimization problems [50,51] SA requires from several parameters, which has been set to the values recommended in the literature. The parameters used are: cooldown (C), which indicates the temperature variation, is set to 0.98; the initial temperature (T_i), which represents the worst value that can be found in the neighborhood, set to 100 000; the maximum number of iterations (I), set to 100; and the neighborhood considered during the search, which is based on the move operator defined in this work. The complexity of this implementation is divided into the construction phase and the proper SA algorithm. The complexity of the constructive phase is equal to GRASP complexity described in Section 4.3, while the complexity of SA is evaluated as $O\left(T_i \cdot I \cdot n^2\right)$. We refer the reader to [52] to a deeper analysis of the complexity of SA.

The results are divided into two different experiments. First of all, SPR and DPR are tests when considering the set of original instances in which the exact method is able to reach the optimal value. Table 6 shows the results obtained.

As it can be derived from the results, SPR performs slightly better than DPR in this set of instances, being able to reach 79 out of 82 optimal solutions, while DPR reaches 76. It is important to remark that the average deviation of both methods, smaller than 0.05, indicates that in those instances in which neither SPR nor DPR are able to reach the optimal value, they stay really close to it. In order to confirm this hypothesis, we have conducted a

Table 7
Comparison of CELF, SA, SPR and DPR over the set of largest and most complex instances.

Instance	CELF				SA			
	Avg.	Dev. (%)	Time (s)	#Best	Avg.	Dev. (%)	Time (s)	#Best
PRISON*	240	11.11	0.02	0	**270**	**0.00**	0.34	**1**
EMAIL-EU-CORE*	4672	1.79	22.71	0	**4757**	**0.00**	267.78	**1**
EGO-FACEBOOK	**19462**	**0.00**	3609.75	**1**	**19462**	**0.00**	1044.35	**1**
CA-GRQC	22487	5.05	12441.33	0	**23684**	**0.00**	5364.14	**1**
TWITCH_EN	25060	0.22	16833.33	0	23853	5.03	5885.88	0
LASTFM_ASIA	**25005**	**0.00**	26017.00	**1**	23000	8.02	6160.10	0
CA-HEPTH	44451	1.16	165624.79	0	**44972**	**0.00**	9105.73	**1**
BLOG_CATALOG3	**46732**	**0.00**	44674.80	**1**	46418	0.67	8407.40	0
Summary	23514.13	2.23	33652.97	3	23302.00	1.71	4534.97	5

Instance	SPR				DPR			
	Avg.	Dev. (%)	Time (s)	#Best	Avg.	Dev. (%)	Time (s)	#Best
PRISON*	**270**	**0.00**	**0.07**	**1**	**270**	**0.00**	0.16	**1**
EMAIL-EU-CORE*	**4757**	**0.00**	**84.48**	**1**	**4757**	**0.00**	262.59	**1**
EGO-FACEBOOK	**19462**	**0.00**	**279.47**	**1**	**19462**	**0.00**	769.70	**1**
CA-GRQC	23630	0.23	**442.07**	0	**23684**	**0.00**	2166.29	**1**
TWITCH_EN	24853	1.06	**680.77**	0	**25116**	**0.00**	2230.12	**1**
LASTFM_ASIA	24556	1.80	**760.97**	0	24780	0.90	2409.55	0
CA-HEPTH	44909	0.14	**2140.92**	0	**44972**	**0.00**	7743.06	**1**
BLOG_CATALOG3	46595	0.29	**1336.91**	0	46692	0.09	8705.44	0
Summary	23629.00	0.44	**715.71**	3	**23716.63**	**0.12**	3035.87	**6**

pairwise non-parametric Wilcoxon statistical test between SPR and Gurobi solver, obtaining a *p*-value equal to 0.109, which indicates that, with a confidence interval of 95%, there are not statistically significant differences between those methods. Regarding the SA, it is worth mentioning that it is able to reach 76 out of 82 instances with a deviation of 4.86%, requiring negligible time such as SPR and DPR. With respect to CELF, the algorithm requires from negligible computing times as DPR, SPR and SA, but it only reaches 61 out of 82 optimal solutions, with a deviation of 12.19%. From these results, we can obtain two main conclusions: SA is a competitive algorithm for the TSS, and the proposed DPR and SPR significantly contribute to the quality of the obtained solutions, as it can be seen in the smaller deviation with respect to the optimal value.

The last experiment is devoted to evaluate the performance of the proposed algorithms and the Gurobi solver when considering the most challenging and realistic instances. Table 7 shows the results obtained in the set of large instances. In this case, we show the results disaggregated, since it is conformed with 8 instances that can be individually analyzed.

It is worth mentioning that the Gurobi solver is only able to provide the optimal solution for 2 out of 8 instances derived from the new set of complex instances marked with an asterisk in the corresponding instance name. For the remaining instances, Gurobi is not even able to load the model in memory, which highlights the need to consider metaheuristic algorithms for this set of challenging instances. In particular, in those instances where Gurobi reaches the optimal value, SA, SPR and DPR are also able to find it. However, CELF is not able to reach the optimal value for these two instances. Additionally, for the instance EMAIL-EU-CORE, Gurobi requires almost 30 h to find the optimal value, while SA requires 268 s, DPR 262 s and SPR only 85 s.

Analyzing the instances in which Gurobi is not able to even load the model, SPR requires from smaller computing time than DPR in general, but it provides worse results in terms of quality. Regarding SA, it is able to provide competitive results in these challenging instances. Specifically, SPR reaches the best solution in 3 out of 8 instances, SA reaches 5 out of 8 best solutions, and, finally, DPR reaches all the best solutions but for two instances in which CELF is able to provide slightly better results. It is worth mentioning that CELF requires from approximately five times the computing time required by DPR, thus being DPR much more scalable for large scale networks. In terms of deviation, CELF provides the worst results with a 2.23%, followed by SA with 1.71%, but it is considerably larger than the one obtained by SPR and DPR. Specifically, the average deviation obtained by SPR is considerably small (0.44%), and DPR is able to reach a deviation of 0.12%. Since the deviation of SPR is really close to 0%, we conduct a pairwise non-parametric Wilcoxon statistical test to evaluate if there are statistically significant differences between SPR and DPR. The resulting *p*-value of 0.04, smaller than 0.05, indicates that DPR is statistically better than SPR. These results highlights the contribution of SPR and DPR to the state of the art of TSS.

6. Conclusions

This research presents two different Path Relinking approaches for solving the Target Set Selection problem. In particular, the Static Path Relinking variant is compared with the Dynamic Path Relinking variant over a set of challenging instances derived from real-life social networks. Both methods are compared with the best approach found in the literature, which is an exact algorithm implemented in the Gurobi commercial solver. In the comparison, the limits of the exact approach are shown, being unable to even load the most complex instances, while SPR and DPR are able to provide high-quality solutions in reasonable computing time. Additionally, a complexity analysis has been included for each algorithm with the aim of analyzing the computational effort required to execute each one of them.

As a conclusion, both SPR and DPR are able to provide promising solutions for the TSS, each one of them being suitable for different situations. On the one hand, if the computing time is a hard constraint, we do recommend considering SPR since the quality of the solutions is not drastically worse. On the other hand, if the maximum computing time is not a critical part of the problem, DPR is able to provide better results in terms of quality.

The proposed algorithms have been compared with a Simulated Annealing implementation, which has been successfully applied in several influence maximization problems, and with CELF, which is a widely used method in the context of influence maximization and, particularly, in TSS. The results obtained highlight the appropriateness of designing an specific algorithm for solving the TSS such as the proposal of this research.

Table 8
Size of each instance considered in this work.

ID	Datasets	N	ID	Datasets	N
1	Knoke Bureaucracies KNOKI	10	73	Zachary Karate Club ZACHE	34
2	**Knoke Bureaucracies KNOKM**	10	74	Zachary Karate Club ZACHC	34
3	Roethlisberger & Dickson Bank RDGAM	14	**75**	**Bernard & Killworth Technical BKTECC**	34
4	Roethlisberger & Dickson Bank RDPOS	14	76–77	Kapferer Tailor Shop KAPFTI1 and KAPFTI2	39
5	**Roethlisberger & Dickson Bank RDHLP**	14	**78–79**	**Kapferer Tailor Shop KAPFTS1 and KAPFTS2**	39
6	Kapferer Mine KAPFMU	15	80	Bernard & Killworth Office BKOFFC	40
7	Kapferer Mine KAPFMM	15	81	Bernard & Killworth Ham Radio BKHAMC	44
8	Thurman Office THURA	15	**82**	**Bernard & Killworth Fraternity BKFRAC**	58
9	Thurman Office THURM	15	83	Gagnon & Macrae Prision	67
10–24	**Newcomb Fraternity NEWC1...NEWC15**	17	84	email-Eu-core network	1005
25	Davis Southern Club Women DAVIS	18	85	Social circles: Facebook	4039
26–28	Sampson Monastery SAMPLK1...SAMPLK3	18	**86**	**General Relativity and Quantum Cosmology collaboration network**	5242
29	Sampson Monastery SAMPES	18	87	Twitch EN	7126
30	Sampson Monastery SAMPIN	18	88	LastFM Asia Social Network	7624
31–51	**Krackhardt Office css KRACKAD1...KRACKAD21**	21	**89**	**High Energy Physics Theory collaboration network**	9877
52–72	**Krackhardt Office css KRACKFR1...KRACKFR21**	21	90	BlogCatalog3	10312

The successful application of Path Relinking metaheuristic to TSS lead us to propose several future lines of research. First of all, it would be interesting to evaluate the Path Relinking proposal over the opposite problem: minimization of information spreading, which is interesting in topics such as misinformation diffusion or disease control, among others. Another interesting line of research is the adaptation of Path Relinking to other well-known influence maximization problems, such as Budgeted Influence Maximization or Influence Spectrum Problem.

CRediT authorship contribution statement

Isaac Lozano-Osorio: Validation, Resources, Conceptualization. **Andrea Oliva-García:** Software, Investigation, Data curation. **Jesús Sánchez-Oro:** Writing – review & editing, Writing – original draft, Supervision, Methodology.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Jesus Sanchez-Oro reports financial support was provided by Spain Ministry of Science and Innovation. Isaac Lozano-Osorio reports financial support was provided by Spain Ministry of Science and Innovation. Andrea Oliva-Garcia reports financial support was provided by Spain Ministry of Science and Innovation.

Data availability

Data availability is public at https://grafo.etsii.urjc.es/TSS/.

Acknowledgments

The authors acknowledge support from the Spanish Ministry of "Ciencia, Innovación MCIN/AEI/10.13039/501100011033/FEDER, UE) under grant ref. PID2021-126605NB-I00 and PID2021-125709OA-C22, the "Comunidad de Madrid, Spain" and "Fondos Estructurales" of the European Union with grant reference S2018/TCS-4566.

Appendix

This Appendix shows the detailed size of each considered social network in Table 8. In particular, each instance is associated with an identification number (column ID), a name (column Datasets), and the number of nodes (column N). The subset of instances used in the preliminary experiments are those highlighted in bold font, specifically: [2, 5, 10, 20, 24, 31, 38, 43, 47, 51, 57, 65, 70, 75, 78, 82, 86, 89].

Instances with ID from 1 to 82 are directly derived from the best method found in the literature [19]. The remaining instances, which will also be included in our public repository https://grafo.etsii.urjc.es/TSS, have been derived from the following social network repositories:

- Instance 83: Same repository as the original dataset, http://vlado.fmf.uni-lj.si/pub/networks/data/UciNet/UciData.htm
- Instances from 84 to 89: SNAP dataset, https://snap.stanford.edu/data/
- Instance 90: BlogCatalog, http://datasets.syr.edu/datasets/BlogCatalog3.html

References

[1] J.V. Chen, B.c. Su, A.E. Widjaja, Facebook C2C social commerce: A study of online impulse buying, Decis. Support Syst. 83 (2016) 57–69, http://dx.doi.org/10.1016/j.dss.2015.12.008.

[2] S. Ryu, J. Park, The effects of benefit-driven commitment on usage of social media for shopping and positive word-of-mouth, J. Retail. Consum. Serv. 55 (2020) 102094, http://dx.doi.org/10.1016/j.jretconser.2020.102094.

[3] D.M.J. Lazer, M.A. Baum, Y. Benkler, A.J. Berinsky, K.M. Greenhill, F. Menczer, M.J. Metzger, B. Nyhan, G. Pennycook, D. Rothschild, M. Schudson, S.A. Sloman, C.R. Sunstein, E.A. Thorson, D.J. Watts, J.L. Zittrain, The science of fake news, Science 359 (6380) (2018) 1094–1096, http://dx.doi.org/10.1126/science.aao2998.

[4] P.A. Dreyer, F.S. Roberts, Irreversible k-threshold processes: Graph-theoretical threshold models of the spread of disease and of opinion, Discrete Appl. Math. 157 (7) (2009) 1615–1627, http://dx.doi.org/10.1016/j.dam.2008.09.012.

[5] K. Sörensen, F. Glover, Metaheuristics, Encycl. Oper. Res. Manag. Sci. 62 (2013) 960–970.

[6] M. Richardson, R. Agrawal, P. Domingos, Trust management for the semantic web, in: D. Fensel, K. Sycara, J. Mylopoulos (Eds.), The Semantic Web - ISWC 2003, Springer Berlin Heidelberg, Berlin, Heidelberg, 2003, pp. 351–368.

[7] D. Kempe, J. Kleinberg, E. Tardos, Maximizing the spread of influence through a social network, in: KDD '03: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2003, http://dx.doi.org/10.1145/956750.956769.

[8] N. Chen, On the approximability of influence in social networks, SIAM J. Discrete Math. 23 (3) (2009) 1400–1415, http://dx.doi.org/10.1137/08073617X.

[9] C.C. Centeno, M.C. Dourado, L.D. Penso, D. Rautenbach, J.L. Szwarcfiter, Irreversible conversion of graphs, Theoret. Comput. Sci. 412 (29) (2011) 3693–3700, http://dx.doi.org/10.1016/j.tcs.2011.03.029.

[10] F. Cicalese, G. Cordasco, L. Gargano, M. Milanič, J. Peters, U. Vaccaro, Spread of influence in weighted networks under time and budget constraints, Theoret. Comput. Sci. 586 (2015) 40–58, http://dx.doi.org/10.1016/j.tcs.2015.02.032.

[11] L. Narayanan, K. Wu, How to choose friends strategically, Theoret. Comput. Sci. 811 (2020) 99–111, http://dx.doi.org/10.1016/j.tcs.2018.07.013.

[12] C.Y. Chiang, L.H. Huang, B.J. Li, J. Wu, H.G. Yeh, Some results on the target set selection problem, J. Comb. Optim. 25 (4) (2012) 702–715, http://dx.doi.org/10.1007/s10878-012-9518-3.

[13] A. Nichterlein, R. Niedermeier, J. Uhlmann, M. Weller, On tractable cases of target set selection, Soc. Netw. Anal. Min. 3 (2) (2012) 233–256, http://dx.doi.org/10.1007/s13278-012-0067-7.

[14] E. Ackerman, O. Ben-Zwi, G. Wolfovitz, Combinatorial model and bounds for target set selection, Theoret. Comput. Sci. 411 (44–46) (2010) 4017–4022, http://dx.doi.org/10.1016/j.tcs.2010.08.021.

[15] M. Fardad, G. Kearney, On a linear programming approach to the optimal seeding of cascading failures, in: 2017 IEEE 56th Annual Conference on Decision and Control, CDC, IEEE, 2017, http://dx.doi.org/10.1109/CDC.2017.8263650.

[16] S. Raghavan, R. Zhang, A branch-and-cut approach for the weighted target set selection problem on social networks, INFORMS J. Optim. 1 (4) (2019) 304–322, http://dx.doi.org/10.1287/ijoo.2019.0012.

[17] F.G. Baghbani, M. Asadpour, H. Faili, Integer linear programming for influence maximization, Iran. J. Sci. Technol. Trans. Electr. Eng. 43 (3) (2019) 627–634, http://dx.doi.org/10.1007/s40998-019-00178-7.

[18] G. Cordasco, L. Gargano, M. Mecchia, A.A. Rescigno, U. Vaccaro, Discovering small target sets in social networks: A fast and effective algorithm, Algorithmica 80 (6) (2017) 1804–1833, http://dx.doi.org/10.1007/s00453-017-0390-5.

[19] S.V. Ravelo, C.N. Meneses, Generalizations, formulations and subgradient based heuristic with dynamic programming procedure for target set selection problems, Comput. Oper. Res. 135 (2021) 105441, http://dx.doi.org/10.1016/j.cor.2021.105441.

[20] M. Chopin, A. Nichterlein, R. Niedermeier, M. Weller, Constant thresholds can make target set selection tractable, Theory Comput. Syst. 55 (1) (2013) 61–83, http://dx.doi.org/10.1007/s00224-013-9499-3.

[21] F. Cicalese, G. Cordasco, L. Gargano, M. Milanič, U. Vaccaro, Latency-bounded target set selection in social networks, Theoret. Comput. Sci. 535 (2014) 1–15, http://dx.doi.org/10.1016/j.tcs.2014.02.027.

[22] M. Charikar, Y. Naamad, A. Wirth, On Approximating Target Set Selection, Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik GmbH, Wadern/Saarbruecken, Germany, 2016, http://dx.doi.org/10.4230/LIPICS.APPROX-RANDOM.2016.4.

[23] X. Liu, Z. Yang, W. Wang, Exact solutions for latency-bounded target set selection problem on some special families of graphs, Discrete Appl. Math. 203 (2016) 111–116, http://dx.doi.org/10.1016/j.dam.2015.09.005.

[24] S.V. Ravelo, C.N. Meneses, E.A. Anacleto, NP-hardness and evolutionary algorithm over new formulation for a Target Set Selection problem, in: 2020 IEEE Congress on Evolutionary Computation, CEC, IEEE, 2020, http://dx.doi.org/10.1109/CEC48606.2020.9185558.

[25] A.L. Serrano, C. Blum, A biased random key genetic algorithm applied to target set selection in viral marketing, in: Proceedings of the Genetic and Evolutionary Computation Conference, GECCO '22, Association for Computing Machinery, New York, NY, USA, 2022, pp. 241–250, http://dx.doi.org/10.1145/3512290.3528785.

[26] R. Olivares, F. Muñoz, F. Riquelme, A multi-objective linear threshold influence spread model solved by swarm intelligence-based methods, Knowl.-Based Syst. 212 (2021) 106623, http://dx.doi.org/10.1016/j.knosys.2020.106623.

[27] S. Banerjee, M. Jenamani, D.K. Pratihar, A survey on influence maximization in a social network, Knowl. Inf. Syst. 62 (9) (2020) 3417–3455, http://dx.doi.org/10.1007/s10115-020-01461-4.

[28] Z. Aghaee, M.M. Ghasemi, H.A. Beni, A. Bouyer, A. Fatemi, A survey on meta-heuristic algorithms for the influence maximization problem in the social networks, Computing 103 (11) (2021) 2437–2477, http://dx.doi.org/10.1007/s00607-021-00945-7.

[29] V. Campos, R. Martí, J. Sánchez-Oro, A. Duarte, GRASP with path relinking for the orienteering problem, J. Oper. Res. Soc. 65 (12) (2014) 1800–1813, http://dx.doi.org/10.1057/jors.2013.156.

[30] A. Duarte, J. Sánchez-Oro, M.G. Resende, F. Glover, R. Martí, Greedy randomized adaptive search procedure with exterior path relinking for differential dispersion minimization, Inform. Sci. 296 (2015) 46–60, http://dx.doi.org/10.1016/j.ins.2014.10.010.

[31] D. Cuellar-Usaquén, C. Gomez, D. Álvarez-Martínez, A GRASP/Path-Relinking algorithm for the traveling purchaser problem, Int. Trans. Oper. Res. 30 (2) (2021) 831–857, http://dx.doi.org/10.1111/itor.12985.

[32] M.G. Resende, C.C. Ribeiro, GRASP with path-relinking: Recent advances and applications, in: T. Ibaraki, K. Nonobe, M. Yagiura (Eds.), Metaheuristics: Progress As Real Problem Solvers, Springer US, Boston, MA, 2005, pp. 29–63, http://dx.doi.org/10.1007/0-387-25383-1_2.

[33] F. Glover, M. Laguna, Tabu Search, Springer, 1998.

[34] M. Laguna, R. Marti, GRASP and path relinking for 2-layer straight line crossing minimization, INFORMS J. Comput. 11 (1) (1999) 44–52, http://dx.doi.org/10.1287/ijoc.11.1.44.

[35] C.C. Ribeiro, M.G. Resende, Path-relinking intensification methods for stochastic local search algorithms, J. Heuristics 18 (2) (2012) 193–214, http://dx.doi.org/10.1007/s10732-011-9167-1.

[36] M. Resende, R.M.M. Gallego, A. Duarte, GRASP and path relinking for the max–min diversity problem, Comput. Oper. Res. 37 (3) (2010) 498–508, http://dx.doi.org/10.1016/j.cor.2008.05.011, Hybrid Metaheuristics.

[37] S. Pérez-Peló, J. Sánchez-Oro, A. Duarte, Finding weaknesses in networks using Greedy Randomized Adaptive Search Procedure and Path Relinking, Expert Syst. 37 (6) (2020) e12540, http://dx.doi.org/10.1111/exsy.12540.

[38] I. Lozano-Osorio, J. Sánchez-Oro, A. Duarte, Ó. Cordón, A quick GRASP-based method for influence maximization in social networks, J. Ambient Intell. Humaniz. Comput. (2021) http://dx.doi.org/10.1007/s12652-021-03510-4.

[39] T.A. Feo, M.G. Resende, A probabilistic heuristic for a computationally difficult set covering problem, Oper. Res. Lett. 8 (2) (1989) 67–71, http://dx.doi.org/10.1016/0167-6377(89)90002-3.

[40] T.A. Feo, M.G.C. Resende, S.H. Smith, A greedy randomized adaptive search procedure for maximum independent set, Oper. Res. 42 (5) (1994) 860–878, http://dx.doi.org/10.1287/opre.42.5.860.

[41] I. Lozano-Osorio, J. Sanchez-Oro, M.A. Rodriguez-Garcia, A. Duarte, Optimizing computer networks communication with the band collocation problem: A variable neighborhood search approach, Electronics 9 (11) (2020) http://dx.doi.org/10.3390/electronics9111860.

[42] I. Lozano-Osorio, A. Martínez-Gavara, R. Martí, A. Duarte, Max–min dispersion with capacity and cost for a practical location problem, Expert Syst. Appl. 200 (2022) 116899, http://dx.doi.org/10.1016/j.eswa.2022.116899.

[43] J. Sánchez-Oro, M. Laguna, R. Martí, A. Duarte, Scatter search for the bandpass problem, J. Global Optim. 66 (4) (2016) 769–790, http://dx.doi.org/10.1007/s10898-016-0446-0.

[44] S. Kirkpatrick, C.D. Gelatt, M.P. Vecchi, Optimization by simulated annealing, Science 220 (4598) (1983) 671–680, http://dx.doi.org/10.1126/science.220.4598.671.
[45] Q. Jiang, G. Song, C. Gao, Y. Wang, W. Si, K. Xie, Simulated annealing based influence maximization in social networks, Proc. AAAI Conf. Artif. Intell. 25 (1) (2011) 127–132, http://dx.doi.org/10.1609/aaai.v25i1.7838.
[46] S.J. Liu, C.Y. Chen, C.W. Tsai, An effective simulated annealing for influence maximization problem of online social networks, Procedia Comput. Sci. 113 (2017) 478–483, http://dx.doi.org/10.1016/j.procs.2017.08.306, The 8th International Conference on Emerging Ubiquitous Systems and Pervasive Networks (EUSPN 2017) / The 7th International Conference on Current and Future Trends of Information and Communication Technologies in Healthcare (ICTH-2017) / Affiliated Workshops.
[47] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, N. Glance, Cost-effective outbreak detection in networks, in: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2007, pp. 420–429, http://dx.doi.org/10.1145/1281192.1281239.
[48] C. Wang, L. Deng, G. Zhou, M. Jiang, A global optimization algorithm for target set selection problems, Inform. Sci. 267 (2014) 101–118, http://dx.doi.org/10.1016/j.ins.2013.09.033.
[49] R. Martín-Santamaría, S. Cavero, A. Herrán, A. Duarte, J.M. Colmenar, A practical methodology for reproducible experimentation: an application to the double-row facility layout problem, Evol. Comput. (2022) 1–35, http://dx.doi.org/10.1162/evco_a_00317.
[50] I. Lozano-Osorio, J. Sánchez-Oro, A. Martínez-Gavara, A.D. López-Sánchez, A. Duarte, An efficient fixed set search for the covering location with interconnected facilities problem, in: L. Di Gaspero, P. Festa, A. Nakib, M. Pavone (Eds.), Metaheuristics, Springer International Publishing, Cham, 2023, pp. 485–490.
[51] J. Yuste, E.G. Pardo, A. Duarte, Variable neighborhood descent for software quality optimization, in: L. Di Gaspero, P. Festa, A. Nakib, M. Pavone (Eds.), Metaheuristics, Springer International Publishing, Cham, 2023, pp. 531–536.
[52] G.H. Sasaki, B. Hajek, The time complexity of maximum matching by simulated annealing, J. ACM 35 (2) (1988) 387–403, http://dx.doi.org/10.1145/42282.46160.

Isaac Lozano-Osorio was born in Quintanar de la Orden (Spain) on August 11, 1997. He is a Ph.D. student at the University Rey Juan Carlos, Madrid, where he is part of the Group for Research in Algorithms For Optimization (GRAFO). His main research interests focus on the applicability of Artificial Intelligence (AI), focus specially on heuristics and metaheuristics, for solving hard optimization problems related to Social Network Influence Maximization Problems.

Andrea Oliva-García is a MEng student at Menéndez Pelayo International University, Madrid. She is part of the Group for Research in Algorithms For Optimization (GRAFO) at University Rey Juan Carlos. Her research interests focus on the applicability of Artificial Intelligence (AI) techniques to solve problems related to social network.

Jesús Sánchez-Oro was born in Madrid (Spain) on December 31, 1987. He holds a degree in Computer Science from the Universidad Rey Juan Carlos (2010), his Master's degree in Computer Vision from the Universidad Rey Juan Carlos in 2011, and his Ph.D. in Computer Science in 2016 from the Universidad Rey Juan Carlos. He is visiting professor at the Computer Science Department, and he is a member of the Group for Research on Algorithms For Optimization (GRAFO). His main research interests focus on Artificial Intelligence and Operations Research, specially in heuristics and metaheuristics for solving hard optimization problems.

Part III

Resumen en castellano

Capítulo 9

Resumen en castellano

Este capítulo sigue la siguiente estructura. La Sección 9.1 introduce los problemas de influencia en redes sociales, a continuación, en la Sección 9.2 se aborda la metodología empleada durante esta tesis, en la Sección 9.3 se presentan los objetivos y la hipótesis planteada en esta tesis doctoral, seguidamente en la Sección 9.4 se recogen los resultados más relevantes de esta investigación para, finalmente, en la Sección 9.5 concluir con las principales conclusiones.

Este anexo se incluye debido al artículo 22 de la Normativa Reguladora de los Estudios de Doctorado de la Universidad Rey Juan Carlos, aprobada en Consejo de Gobierno de 07/06/2019.

9.1 Antecedentes

La optimización ha sido una preocupación constante a lo largo de la historia, desde los antiguos griegos que buscaban la manera más eficiente de organizar las ciudades, hasta los modernos algoritmos que optimizan procesos empresariales. La importancia de la optimización radica en su capacidad para resolver problemas complejos, mejorar la eficiencia y tomar decisiones informadas. A lo largo de los siglos, se ha demostrado que la optimización es fundamental para el progreso humano.

En la actualidad, la optimización ha cobrado una importancia aún mayor en diversos campos, gracias a la creciente complejidad de los problemas que enfrentamos. Desde la logística empresarial hasta la planificación de rutas en la navegación, la optimización se ha convertido en una herramienta esencial para enfrentar desafíos en constante evolución. La capacidad de resolver problemas de manera eficiente y precisa es crucial en un mundo cada vez más interconectado y dependiente de la tecnología.

Para abordar estos desafíos, existen diversas metodologías en el campo de la optimización, como los métodos exactos, las aproximaciones y los algoritmos genéticos y heurísticos. Estos enfoques ofrecen soluciones flexibles y adaptativas para una variedad de problemas, permitiendo a los investigadores y profesionales encontrar la mejor solución posible en diferentes contextos.

Esta tesis se enfoca específicamente en problemas relacionados con el análisis de redes sociales, un área de estudio que ha ganado relevancia en la era digital. Dentro de esta disciplina, se identifican diversos problemas, y la atención se centra especialmente en el concepto de influencia. El problema central consiste en seleccionar usuarios dentro de una red social de manera que maximicen o minimicen la influencia sobre otros usuarios, considerando posibles restricciones, como presupuestos máximos.

Definir la influencia en el contexto de las redes sociales presenta un desafío significativo debido a la diversidad de métodos disponibles. La capacidad de seleccionar usuarios estratégicos tiene aplicaciones prácticas en campañas de marketing, en la erradicación de enfermedades y en la detección de campañas de desinformación. La complejidad de estos problemas se acentúa por la naturaleza $\mathcal{NP}$ de muchos de ellos, lo que implica que encontrar soluciones exactas es impracticable en grandes redes sociales.

Aunque existen algoritmos aproximativos, en ciertos casos es fundamental contar con información rápida y de alta calidad, como en la detección de enfermedades. Por lo tanto, esta tesis se centra en el uso de heurísticos y metaheurísticos para abordar problemas de influencia en redes sociales. Estos enfoques proporcionan soluciones eficientes y adaptables, especialmente en situaciones donde la velocidad y la precisión son fundamentales.

En este contexto, la influencia en redes sociales surge como uno de los problemas que está aumentando su popularidad en la actualidad y es el tema principal que concierne a esta Tesis.

9.2 Metodología

La metodología usada en esta Tesis Doctoral se basa en el método científico [116]. Esta metodología busca validar nuevos conocimientos siguiendo unos determinados pasos: observación, hipótesis, experimentación, medición, falsabilidad, reproducibilidad, revisión por pares y publicación.

El método científico es muy utilizado y se aplica a muchas áreas. En el ámbito de la optimización heurística y en esta Tesis Doctoral se ha utilizado como guía para proponer soluciones basadas en algoritmos heurísticos y algoritmos metaheurísticos. Determinar si una propuesta algorítmica es mejor que el estado del arte debe hacerse de una manera justa. Para evaluar un procedimiento metaheurístico desde un punto de vista objetivo en [117] se proponen tres pasos: diseño de la experimentación, ejecución y documentación de los resultados.

El diseño experimental busca definir los objetivos de los experimentos, mostrando las instancias (diferentes situaciones para cada uno de los problemas abordados) que se usarán para validar el algoritmo propuesto. Es recomendable usar las mismas instancias propuestas en los trabajos anteriores del estado del arte. De esa manera, la comparación entre el algoritmo propuesto y los métodos previos es más objetivo y justo.

Una vez diseñado el experimento, debemos conocer los indicadores de rendimiento del algoritmo. Para validar los resultados del algoritmo, los resultados son tratados estadísticamente para su análisis. En los métodos de optimización exacta, la eficiencia en términos de tiempo de ejecución es el indicador principal. Sin embargo, para evaluar el rendimiento de los procedimientos heurísticos y metaheurísticos deben considerarse otros indicadores: calidad de la solución, esfuerzo computacional y robustez del algoritmo [118, 119].

Finalmente, la interpretación de los resultados obtenidos muestra los resultados y realiza un análisis siguiendo los objetivos definidos. Para poder realizar una comparación justa y es importante en el área científica que los experimentos se tengan en un mismo entorno informático de experimentación. La interpretación de los resultados debe ser explícita y basarse en los objetivos definidos y las medidas de rendimiento consideradas.

Para cumplir el método científico en este ámbito, la reproducibilidad de los experimentos es algo importante. Para ello, se ha documentado todas las decisiones tomadas en los dos primeros pasos y se han publicado los resultados de la investigación en repositorios públicos de fácil acceso de cara a facilitar futuras comparaciones.

9.3 Hipótesis y objetivos

La hipótesis propuesta para el desarrollo de esta Tesis Doctoral se sustenta desde el punto de vista de que los problemas de influencia en redes sociales son problemas $\mathcal{NP}$—difíciles, con interés práctico en muchas disciplinas científicas. Por tanto, es interesante desarrollar algoritmos que puedan generar soluciones de alta calidad en un tiempo razonable. La escalabilidad es fundamental debido al tamaño de las redes sociales del mundo real.

También se proponen técnicas metaheurísticas, que han demostrado ser procedimientos eficaces a la hora de abordar problemas de optimización. En particular, las metaheurísticas trayectoriales constituyen una subfamilia de este tipo de técnicas, caracterizadas por considerar más de una solución simultáneamente y proporcionar mecanismos de combinación entre ellas. La búsqueda dispersa es un claro exponente de este tipo de metaheurísticas. Por otro lado, las metaheurísticas trayectoriales parten de una solución inicial y son capaces de generar una trayectoria en el espacio de solución. Greedy Randomized Adaptive Search Procedure (GRASP) es un ejemplo de metaheurística trayectorial.

El algoritmo heurístico propuesto se complementará con las técnicas metaheurísticas que mejor se adapten al problema, en particular se considerarán Greedy Randomized Adaptive Search Procedure y Path Relinking.

Para alcanzar los principales objetivos mencionados anteriormente, se deben abordar los siguientes objetivos:

- Estudiar el estado del arte del problema, analizando las propuestas algorítmicas actuales.
- Diseñar y desarrollar algoritmos heurísticos para resolver problemas relacionados con los problemas de influencia en redes sociales, utilizando diferentes modelos

de difusión de la influencia.

- Configurar los parámetros de los algoritmos desarrollados. Los algoritmos desarrollados deben configurarse con el fin de utilizar los mejores parámetros y lograr los mejores resultados. Para configurar los parámetros, se deben realizar experimentos preliminares.
- Validar el algoritmo heurístico. Los algoritmos se compararán experimentalmente con los mejores algoritmos en cada problema para proporcionar una comparación justa.
- Analizar los resultados obtenidos por el nuevo algoritmo y compararlos frente a los mejores algoritmos.
- Adaptar los algoritmos propuestos a los problemas reales conocidos relacionados con: contención de la evolución de pandemias, minimización de rumores y noticias falsas, donde se validarán los resultados.
- Publicar todo el código fuente, instancias y resultados en repositorios públicos para facilitar futuras comparaciones.
- Los resultados parciales serán enviados a revistas JCR con procesos de revisión por parte de instituciones independientes. De esta manera se obtendrá una evaluación experta para mejorar nuestra investigación y verificar que aporte al área académica relacionada con influencia en redes sociales.

9.4 Resultados

Esta sección muestra una revisión de los resultados arrojados por los problemas abordados en esta Tesis. Las siguientes secciones explican la propuesta del algoritmo, así como una comparación exhaustiva con el mejor método encontrado en la literatura para cada problema. En todos los casos se presenta una tabla final compuesta por las siguientes métricas: Pro., que denota el valor promedio de la función objetivo en todos los casos; Des. (%), que representa la desviación promedio al valor mejor conocido; Tiempo (s), que muestra el tiempo promedio de cálculo requerido por cada algoritmo; #Mejor, que coincide si la solución es la mejor solución u #Óptimo, que cuenta el número de valores exactos encontrados por el algoritmo. Además, todas las tablas muestran resaltados en negro los mejores resultados. Estos resultados se derivan de la Parte II donde la Sección 9.4.1 muestran los resultados del SNIMP, la Sección 9.4.2 estudia los resultados en el contexto de BIMP y, finalmente, Sección 9.4.3 resume los resultados derivados del TSS.

9.4.1 Resultados del problema de maximizar la influencia en redes sociales

Richardson y Domingos [55] formularon inicialmente el problema de la selección de nodos objetivo en SNs. Sin embargo, no fue hasta Kempe et al. [17] que se resolvió el

SNIMP formulándolo como un problema de optimización discreto, demostrando posteriormente que es $\mathcal{NP}$-difícil [47].

De acuerdo con varios trabajos de resumen [36, 37], algoritmos voraces y de aproximación se proponen, ya que este problema se mostró como un problema $\mathcal{NP}$-difícil. En la Sección 2.2 se explican varios modelos de propagación de la influencia (IDMs), siendo el modelo independiente por cascada (ICM) uno de los IDMs más extendidos. Este método probabilístico muestra el número medio de nodos activados en una simulación de propagación. Los resultados obtenidos sobre este problema se han publicado en una revista de alto impacto indexada en el JCR. Se pueden encontrar más detalles en el Capítulo 6 de la Parte II.

El algoritmo propuesto para resolver SNIMP se basa en la metodología GRASP [26] (véase la Sección 3.1 para más detalles). Obsérvese que, en el ámbito de las redes sociales, el método de búsqueda local es un procedimiento bastante exigente desde el punto de vista computacional, ya que las redes sociales reales necesitan una gran cantidad de nodos y aristas, por lo que se requieren métodos altamente escalables.

La fase constructiva en este problema se ha diseñado para generar una solución inicial y suele estar guiada por una función de selección voraz que ayuda al método constructivo a seleccionar el siguiente elemento que se incluirá en la solución parcial. El primer nodo se selecciona al azar para favorecer la diversificación, y luego los nodos restantes hasta llegar a k elementos se seleccionan mediante un criterio voraz. En este trabajo se estudian dos métodos voraces: el primero se basa en la información local de los vecinos, mientras que el otro se basa en el coeficiente de clustering [99]. Todos los nodos son evaluados según este criterio, conformando una lista de candidatos (CL). A continuación, se establece un umbral $\mu = g_{\text{mín}} + \alpha \cdot (g_{\text{máx}} - g_{\text{mín}})$ que se utiliza para seleccionar los nodos más prometedores de la CL, creando una lista de candidatos restringida (RCL). Este umbral depende directamente del valor del parámetro de entrada α, que está en el rango $[0, 1]$. Nótese que este parámetro indica la avaricia o aleatoriedad del procedimiento constructivo. Por un lado, sí $\alpha = 0$, entonces el umbral se evalúa como $g_{\text{máx}}$, convirtiéndose en un algoritmo totalmente voraz (es decir, la *RCL* solo incluye la mejor opción en cada iteración). Por otro lado, si $\alpha = 1$ entonces $\mu = g_{\text{mín}}$, resultando un método totalmente aleatorio (es decir, la *RCL* incluye cada elección factible en cada iteración).

A continuación, cuando se obtiene una solución factible, la fase de búsqueda local explora la vecindad conformada por todas las soluciones que pueden alcanzarse realizando un único movimiento. La vecindad de una solución S se define como el conjunto de soluciones que se pueden alcanzar realizando un único movimiento sobre S. En el contexto de SNIMP, proponemos un movimiento de intercambio donde el nodo u se elimina del conjunto de semillas, siendo sustituido por v, con $u \in S$ y $v \notin S$. Este movimiento de intercambio se define formalmente como:

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

Así, la vecindad N_s de una solución dada S consiste en el conjunto de soluciones que se pueden alcanzar desde S realizando un único movimiento de intercambio. Más formalmente,

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

Como se ha dicho, la escalabilidad es totalmente necesaria en este trabajo, y realizar todos los movimientos posibles daría lugar a un procedimiento que llevaría bastante tiempo. Para reducir la complejidad computacional de la búsqueda, se propone una búsqueda local surrogada. Este trabajo propone una estrategia inteligente de exploración de vecindarios con el objetivo de reducir el número de soluciones exploradas dentro de cada vecindario. Esta reducción del tamaño del espacio de búsqueda se realiza explorando solo una pequeña fracción, δ, de los nodos disponibles para el nodo de intercambio.

El IDM seleccionado es, como en el mejor trabajo anterior, el algoritmo ICM con la correspondiente simulación Monte Carlo, realizando 100 iteraciones con una probabilidad de influencia de 0.01. Estos valores de los parámetros son los más extendidos en la literatura relacionada. El número de nodos semilla k para conformar una solución se selecciona en el rango $k = \{10, 20, 30, 40, 50\}$ tal y como se indica en [17, 70, 100], obteniendo así $7 \cdot 5 = 35$ instancias de problema diferentes (resultantes de la combinación de 7 redes y 5 tamaños de conjunto semilla).

Para analizar la calidad del algoritmo propuesto, realizamos una prueba competitiva con los mejores métodos encontrados en el estado del arte, considerando el conjunto completo de 35 instancias derivadas del repositorio SNAP[1]. Se consideran tres algoritmos: CELF [44], el conocido algoritmo greedy hill-climbing; CELF++ [101], la versión mejorada de CELF [44]; y PSO [102], el algoritmo de optimización por enjambre de partículas que se considera el estado del arte para el análisis de la influencia social según el reciente estudio experimental desarrollado en [36]. La Tabla 9.1 recoge los resultados finales obtenidos en esta prueba competitiva. Nótese que Pro. no es un valor entero, ya que es el valor medio de las 100 ejecuciones del ICM en la simulación de Monte Carlo. También hemos incluido una fila final en esta tabla (G.Pro.) con los valores promedios de la función objetivo y Tiempo (s), calculados a través del conjunto de 35 instancias.

Nos gustaría destacar en primer lugar los resultados obtenidos con PSO, ya que es el último clasificado, incluso siendo considerado el estado del arte para este problema. La razón detrás de esto es que el trabajo original [102] considera instancias de tamaño pequeño (de 410 a 15233 nodos) y la calidad de las soluciones proporcionadas por PSO se deteriora cuando el tamaño de la instancia aumenta, como se puede derivar de la Tabla 9.1. Mientras tanto, CELF y su versión mejorada CELF++ son capaces de alcanzar mejores soluciones, siendo aún competitivos con los algoritmos de última generación. Sin embargo, solo CELF++ es capaz de igualar la solución más conocida en 1 instancia (de 35). Finalmente, los mejores resultados se obtienen con el algoritmo GRASP propuesto, que es capaz de alcanzar la mejor solución encontrada en los 35 casos. Además, el tiempo de cálculo es menor que el del segundo mejor algoritmo, CELF++ (39.04 frente a 60.09 segundos en promedio).

Al analizar el tiempo de cálculo requerido para cada algoritmo, podemos ver claramente que CELF y CELF++, como enfoques completamente voraces, no se ven

[1]`http://snap.stanford.edu`

		CELF		**CELF++**		**PSO**		**GRASP**	
k	**Instancia**	**Pro.**	**Tiempo (s)**	**Pro.**	**Tiempo (s)**	**Pro.**	**Tiempo (s)**	**Pro.**	**Tiempo (s)**
	CA-AstroPh	157.60	**2.51**	171.81	9.40	169.85	232.40	**187.47**	8.28
	CA-CondMat	35.73	**0.67**	35.73	2.15	33.40	4.60	**36.15**	2.56
	Cit-HepPh	46.63	**1.16**	46.63	3.29	35.27	1.71	**47.20**	4.20
10	Email-Enron	383.95	**25.23**	469.63	87.68	465.24	1756.84	**489.67**	41.41
	Email-EuAll	132.96	**6.03**	130.28	307.98	107.41	37.42	**144.57**	24.42
	Wiki-Vote	108.50	**0.39**	108.50	1.00	92.16	16.40	**109.10**	6.32
	p2p-Gnutella31	16.24	**1.46**	16.23	7.83	13.38	0.95	**16.27**	1.63
	CA-AstroPh	222.63	**2.69**	234.36	9.76	222.92	889.79	**259.25**	18.53
	CA-CondMat	59.72	**0.66**	59.87	2.13	45.46	8.67	**61.05**	6.00
	Cit-HepPh	81.75	**1.11**	81.75	3.25	68.51	2.58	**82.11**	18.97
20	Email-Enron	451.24	**25.71**	547.96	88.47	544.57	4394.46	**589.65**	74.23
	Email-EuAll	214.66	**5.68**	214.54	303.01	162.32	99.98	**224.10**	28.88
	Wiki-Vote	162.49	**0.49**	162.49	1.45	141.66	41.44	**165.32**	26.03
	p2p-Gnutella31	30.82	**1.30**	30.86	7.43	24.69	0.99	**30.92**	3.80
	CA-AstroPh	266.77	**2.85**	276.69	10.48	259.90	1005.17	**312.68**	51.32
	CA-CondMat	80.87	**0.70**	82.18	2.30	66.27	11.09	**82.54**	14.11
	Cit-HepPh	113.39	**1.16**	113.39	3.45	86.22	3.69	**113.63**	42.30
30	Email-Enron	501.78	**25.49**	608.63	88.62	553.25	7594.67	**652.48**	140.71
	Email-EuAll	277.40	**5.86**	275.36	298.66	212.84	183.58	**281.30**	59.58
	Wiki-Vote	208.18	**0.64**	208.18	2.03	150.40	97.75	**214.97**	80.83
	p2p-Gnutella31	44.75	**1.24**	**44.81**	7.42	35.30	1.34	**44.81**	6.08
	CA-AstroPh	319.52	**3.11**	302.86	11.58	288.92	1492.82	**360.34**	66.97
	CA-CondMat	100.96	**0.76**	101.80	2.54	75.61	17.40	**104.38**	16.37
	Cit-HepPh	140.63	**1.27**	140.63	3.81	113.46	4.94	**141.20**	58.03
40	Email-Enron	549.64	**25.95**	658.38	92.09	634.58	9032.87	**705.03**	216.65
	Email-EuAll	323.85	**6.17**	312.47	302.47	258.46	230.12	**337.39**	165.23
	Wiki-Vote	246.02	**0.83**	246.02	2.83	182.88	115.05	**252.15**	34.60
	p2p-Gnutella31	58.26	**1.28**	58.22	7.48	51.26	1.85	**58.37**	12.69
	CA-AstroPh	361.51	**3.50**	338.28	13.11	340.54	2267.98	**399.92**	132.35
	CA-CondMat	119.29	**0.86**	120.72	2.87	106.10	10.85	**124.57**	26.51
	Cit-HepPh	165.47	**1.38**	165.47	4.26	126.77	6.35	**166.77**	65.20
50	Email-Enron	597.26	**27.02**	680.29	96.71	662.67	10063.47	**744.38**	157.26
	Email-EuAll	361.51	**6.68**	357.43	304.03	258.15	321.81	**375.03**	161.59
	Wiki-Vote	277.65	**1.09**	277.65	3.76	188.82	181.07	**287.66**	86.39
	p2p-Gnutella31	71.80	**1.34**	71.90	7.76	64.63	2.74	**72.08**	17.42
	G.Avg.	208.33	**5.55**	221.49	60.09	195.54	1146.71	**236.37**	39.04

Tabla 9.1: Pruebas competitivas del algoritmo GRASP propuesto con respecto a los mejores algoritmos encontrados en la literatura: CELF, CELF++ y PSO. Los mejores resultados se resaltan en negrita.

realmente afectados por el aumento del tamaño del conjunto de semillas. Por el contrario, el tiempo de cálculo requerido para PSO y GRASP se ve afectado por el tamaño del conjunto de semillas, ya que valores k mayores llevan al método de mejora local a realizar un mayor número de iteraciones. Sin embargo, si miramos más de cerca los resultados obtenidos con GRASP, podemos concluir que este aumento en el número de iteraciones y, por tanto, en el tiempo de cálculo, permite que el algoritmo alcance mejores soluciones. En el caso de PSO, el aumento del tiempo de cálculo es aún mucho más notable, pero normalmente no resulta en mejores soluciones, lo que sugiere que el algoritmo PSO es particularmente adecuado para resolver instancias de tamaño pequeño.

Para validar estos resultados, hemos realizado una prueba de Friedman no paramétrica para clasificar todos los algoritmos comparados. El valor p obtenido, inferior a 0.0005, confirma que existen diferencias estadísticamente significativas entre los algoritmos. Los algoritmos ordenados por clasificación son GRASP (1.00), CELF++ (2.44), CELF (2.79) y PSO (3.77). Finalmente, realizamos la conocida prueba estadística no paramétrica de Wilcoxon para comparaciones por pares, que responde a la pregunta: ¿las soluciones generadas por ambos algoritmos representan dos poblaciones diferentes? El valor p resultante menor que 0.0005 al comparar GRASP entre sí confirma la superioridad del algoritmo GRASP propuesto. Por tanto, GRASP emerge como uno de los algoritmos más competitivos para el SNIMP, siendo capaz de alcanzar soluciones de alta calidad en poco tiempo de computación.

9.4.2 Resultados del problema de maximizar la influencia con un presupuesto

Las empresas suelen tener un presupuesto específico en sus campañas de marketing, esto no está modelado en el SNIMP clásico. Con el objetivo de incluir esta característica, Nguyen [57] definió formalmente el BIMP inspirado en SNIMP, siendo también $\mathcal{NP}$-difícil. Como ya se ha comentado existen varios IDM, este trabajo utiliza: ICM, WCM y TV.

Banerjee [36] publicó un artículo de resumen relacionado con los problemas de maximización de la influencia en redes sociales, convirtiéndose en uno de los trabajos de investigación más relevantes en el área. El algoritmo denominado ComBIM propuesto por Banerjee [45] se considera el estado del arte para BIMP. ComBIM proporciona una solución basada en la comunidad que proporciona los mejores resultados en la literatura hasta donde nuestro conocimiento alcanza, por lo que será considerado como el algoritmo para referenciar nuestra propuesta. Se pueden encontrar más detalles en el Capítulo 7 de la Parte II.

La propuesta algorítmica se basa en la metodología Greedy Randomized Adaptive Search Procedure (GRASP) donde se diseña una novedosa heurística eficiente y eficaz para la selección del conjunto de semillas en la fase constructiva. Este criterio voraz, denominado g_{dist} aprovecha la distribución de nodos semilla, es decir, prioriza los nodos que no tienen vecinos seleccionados como nodo semilla, con el objetivo de llegar a un mayor número de usuarios no influenciados, explorando regiones mayoritariamente ignoradas hasta ese momento.

Para ello, se valora directamente el grado de un nodo, pero penalizándolo si algunos de sus nodos vecinos ya han sido seleccionados. La penalización se ha fijado experimentalmente reduciendo el grado a la mitad. Más formalmente,

$$g_{dist} = \begin{cases} d_u^+ & \text{if } v \notin S,\ \forall v \in N_u^+ \\ \frac{d_u^+}{2} & \text{en otro caso} \end{cases}$$

Una vez finalizado el constructivo se realiza una fase de búsqueda local. La principal diferencia con SNIMP es el presupuesto (B), ya que en BIMP no se requiere un número exacto de nodos. Entonces, un movimiento que elimine un nodo e inserte otro, puede dar lugar a una solución inviable que supere el presupuesto disponible. La vecindad de una solución S se define como el conjunto de soluciones que se pueden alcanzar realizando un único movimiento sobre S. A continuación, es necesario definir el movimiento que se considerará en el contexto de BIMP. En concreto, el movimiento, denominado *Reemplazar*(S, u, P), consiste en eliminar el nodo u de la solución y sustituirlo por el conjunto de nodos en P, con $P \in V \setminus S$. Obsérvese que, para alcanzar una solución factible, la suma del coste de los nodos en P debe ser menor o igual que $B + C(u)$ (puesto que u se eliminará, su coste no debe tenerse en cuenta). Más formalmente,

$$\mathit{Reemplazar}(S, u, P) = S \setminus \{u\} \cup P$$

Entonces, dada una solución S, la vecindad $N_{textitR}(S)$ se define como el conjunto de soluciones factibles que pueden alcanzarse con un único movimiento *Reemplazar*. En términos matemáticos,

$$N_R(S) = \left\{ S' \leftarrow \mathit{Reemplazar}(S, u, P)\ \forall u \in S \land \forall P \in V \setminus S : \sum_{p \in P} C(p) \leq B + C(u) \right\}$$

Incluso considerando una implementación eficiente de la evaluación de la función objetivo, el gran tamaño de la vecindad resultante hace que la exploración completa de la vecindad no sea adecuada para el BIMP. Por ello, limitamos el número de evaluaciones que realiza la búsqueda local con el objetivo de disponer de un método computacionalmente eficiente. Cabe mencionar que, si el número de iteraciones utilizadas en los IDMs es limitado, entonces es interesante explorar en primer lugar los vecinos más prometedores de la vecindad considerada.

Los parámetros utilizados son los siguientes: como es habitual en los problemas SIMs, se realizan 100 simulaciones Monte Carlo en todos los modelos IDMs. El presupuesto total B para conformar una solución se selecciona en el rango $B = \{2000, 6000, 10000, 140000, 180000, 22000, 26000\}$ como se indica en el trabajo previo, obteniendo así $3 \cdot 7 = 21$ instancias de problema diferentes para cada IDM. Teniendo en cuenta que se consideran 4 IDMs, el número total de instancias son $21 \cdot 4 = 84$. Para analizar la calidad del algoritmo propuesto, se realiza una prueba competitiva con el mejor método encontrado en el estado del arte, ComBIM. La Tabla 9.2 recoge los resultados finales obtenidos en esta prueba competitiva para cada IDM.

IDM	Algoritmo	Pro.	Tiempo (s)	Des (%)	#Mejor
ICM(1 %)	ComBIM	8319.68	214.97	17.64 %	0
	GRASP	**8872.61**	**117.06**	**0.00 %**	**21**
ICM(2 %)	ComBIM	14467.65	215.31	6.49 %	3
	GRASP	**14828.77**	**146.21**	**0.07 %**	**18**
WCM	ComBIM	2277.79	214.04	57.49 %	0
	GRASP	**10087.08**	**97.80**	**0.00 %**	**21**
TV	ComBIM	1976.11	214.68	39.10 %	0
	GRASP	**2677.58**	**69.65**	**0.00 %**	**21**
Resumen	ComBIM	6760.31	214.75	30.18 %	3
	GRASP	**9116.51**	**107.68**	**0.02 %**	**81**

Tabla 9.2: Pruebas competitivas de la metodología GRASP propuesta con respecto al algoritmo de última generación ComBIM. Los mejores resultados se destacan en negrita.

Los resultados muestran como GRASP es capaz de obtener soluciones de alta calidad (81 mejores soluciones de 84), requiriendo la mitad del tiempo de computación (107.68 segundos frente a 214.75 segundos). Aunque GRASP es capaz de superar a ComBIM en todos los IDMs considerados, los resultados más destacables en términos de calidad se obtienen al utilizar WCM y TV. En concreto, ComBIM es capaz de alcanzar la mejor solución solo en tres casos cuando se utiliza ICM (2 %). En este caso, la desviación de GRASP es del 0.07 %, lo que indica que está realmente cerca de esa mejor solución. A la vista de estos resultados, GRASP se perfila como uno de los algoritmos más competitivos para BIMP.

Por último, realizamos la conocida prueba estadística no paramétrica de Wilcoxon para comparaciones por pares, que responde a la pregunta: ¿representan las soluciones generadas por ambos algoritmos dos poblaciones diferentes? El valor p resultante inferior a 0.0001 al comparar GRASP con ComBIM confirma la superioridad del algoritmo GRASP propuesto. En concreto, GRASP es capaz de obtener 81 de 84 clasificaciones positivas, 3 negativas y 0 empates.

9.4.3 Resultados del problema de seleccionar el conjunto objetivo

El último problema resuelto está relacionado con la familia de problemas de selección del conjunto objetivo, en los que se pueden distinguir dos variantes: garantizar alcanzar la red completa (o incluso una cierta parte de ella) con el mínimo número de usuarios iniciales o maximizar el número de usuarios alcanzados sin exceder un presupuesto inicial. Esta propuesta se centra en resolver este último, que suele denominarse problema de selección del conjunto objetivo GAP de máximo esfuerzo-recompensa (Max-TSS), un problema $\mathcal{NP}$-difícil [59]. Se pueden encontrar más detalles en el Capítulo 8 de la Parte II.

Nuestra propuesta está relacionada con Path Relinking para resolver el problema Max-TSS [103]. Path Relinking requiere de un método para generar soluciones diversas y de alta calidad con el fin de crear caminos prometedores durante la búsqueda, tanto en variantes estáticas, como dinámicas. Aunque estas soluciones pueden generarse aleatoriamente, se ha demostrado experimentalmente en varios trabajos que el diseño de un método constructivo y de mejora local específico para el problema considerado suele conducir a mejores resultados [92, 104, 105, 106].

En el contexto de los problemas de maximización de influencia, la metodología Greedy Randomized Adaptive Search Procedure ha demostrado ser un método eficaz y eficiente para generarlos [69]. El método constructivo propuesto para el problema Max-TSS sigue la filosofía GRASP de diversificación evitando decisiones totalmente voraces. Con el objetivo de aumentar la diversidad, el método selecciona el primer nodo a incluir al azar del conjunto de usuarios V, inicializando la solución en construcción S. A continuación, se crea la lista de candidatos (CL) con todos los nodos menos v. El método constructivo añade iterativamente un nodo a la solución mientras no se supere el presupuesto y la CL no esté vacía. En cada iteración, se calcula el valor mínimo y máximo de una determinada función voraz. El objetivo de la función voraz es evaluar lo prometedor que es un candidato y es una parte clave del procedimiento constructivo. Con este umbral, se crea la lista de candidatos restringida (RCL), que contiene todos los nodos cuyo valor de la función voraz es mayor o igual que el umbral μ, considerando que no superan el presupuesto máximo. Una vez construida la RCL, se selecciona al azar el siguiente elemento del mismo (ya que todos los nodos de la RCL son prometedores) para favorecer la diversidad. A continuación, el nodo seleccionado se añade a la solución actual y se actualiza la CL eliminándolo.

La función voraz se considera tradicionalmente en la literatura GRASP y consiste en evaluar directamente el valor de la función objetivo si el nodo evaluado se añadiera a la solución actual, es decir, representa la contribución directa del nodo a la solución en construcción. Más formalmente,

$$g_{of}(c, S) \leftarrow TSS(S \cup \{c\})$$

El principal inconveniente es que la evaluación de la función objetivo para el Max-TSS es un proceso bastante exigente computacionalmente, por lo que se propone una nueva función voraz con el objetivo de reducir el esfuerzo computacional de la evaluación, ya que se realizará en cada iteración del proceso de construcción. La segunda función voraz propuesta, denominada $g_{dg}(c, S)$, considera que la relevancia de un nodo es directamente proporcional a su grado. En otras palabras, si un nodo está conectado a varios nodos, entonces probablemente influirá en una gran cantidad de sus nodos adyacentes. Entonces, esta función voraz se evalúa como el grado del nodo evaluado:

$$g_{dg}(c, S) \leftarrow |u \in V : (c, u) \in E|$$

La segunda fase de GRASP consiste en un método de mejora local que encuentra un óptimo local partiendo de la solución inicial. A partir de la solución inicial S, no es posible añadir nuevos nodos, ya que el procedimiento constructivo se detiene cuando se supera el presupuesto máximo con alguno de los nodos restantes. Por lo tanto, el operador de movimiento propuesto se define en dos pasos: eliminar y añadir. En concreto, el operador de movimiento elimina un nodo de la solución y, a continuación, añade nodos de forma iterativa hasta alcanzar el presupuesto máximo.

En el contexto de TSS, el esfuerzo computacional es una parte crítica del algoritmo, por lo que hemos decidido utilizar el primer método de mejora con el objetivo de reducir el tiempo computacional para realizar un método de búsqueda local. Con el objetivo de evitar sesgar la búsqueda, se explora el vecindario de forma aleatoria, realizando el primer movimiento que resulte en una solución mejor en base a un operador de movimiento que elimina un nodo de la solución y lo sustituye por todos los nodos que se puedan añadir sin exceder el presupuesto permitido.

Con el objetivo de reducir aún más el esfuerzo computacional del método de búsqueda local, se proponen tres mejoras. La primera mejora trata de evitar la exploración de soluciones ya visitadas. Para ello, se asocia a cada solución visitada un número único, es decir, un código hash. A continuación, cada vez que se visita una solución, se evalúa si su código hash correspondiente no se ha incluido ya en el conjunto de soluciones visitadas. En caso afirmativo, el método deshace el movimiento y continúa con la siguiente iteración, evitando repetir la exploración de la misma región del espacio de búsqueda. La segunda mejora se dedica a limitar los nodos explorados durante la búsqueda, descartando aquellos nodos que darán lugar a una solución inviable. A continuación, los nodos candidatos a ser añadidos se clasifican con respecto a su valor de esfuerzo en orden ascendente. Solo se exploran aquellos nodos cuyo valor de esfuerzo es menor o igual que el presupuesto disponible. Además, para favorecer la diversidad, la exploración se realiza de forma aleatoria entre todos los nodos que satisfacen esta restricción.

El objetivo de la última mejora es reducir el tiempo de cálculo necesario para evaluar la influencia de un nodo almacenándola en caché. En concreto, la influencia de un nodo (es decir, los nodos que se ven afectados por su activación), se calcula al principio del método de búsqueda local. Después, cada vez que se selecciona un nodo para eliminarlo o añadirlo a la solución, se actualiza la influencia de ese nodo sobre los demás nodos del grafo. Como resultado, no es necesario evaluar completamente la función objetivo en cada iteración, sino comprobar la influencia correspondiente precalculada.

A continuación, se utilizan SPR y DPR para mejorar la solución. Las estrategias de Path Relinking requieren de un conjunto de soluciones de alta calidad, normalmente conocido como conjunto élite que se combinan. En el contexto de TSS, dadas dos soluciones S_i y S_g a combinar, el método de creación de trayectorias diseñado para el problema Max-TSS elimina iterativamente nodos que pertenecen a S_i pero no a S_g, es decir, $S_i \setminus S_g$, e incluye nodos que están en S_g pero no en S_i, es decir, $S_g \setminus S_i$. El método se detiene cuando S_i se ha convertido completamente en S_g y no se pueden eliminar /añadir más nodos. Dado que el esfuerzo computacional es una parte crítica de TSS, hemos seleccionado Random Path Relinking que, además, aumenta la diversidad de la búsqueda. En el método propuesto, se combinan todos los pares de soluciones del conjunto élite.

El conjunto de datos utilizado para realizar los experimentos se ha derivado del mejor algoritmo encontrado en la literatura para proporcionar una comparación justa. Este conjunto de instancias está formado por 82 instancias derivadas de redes sociales reales que se han utilizado ampliamente en el análisis de redes sociales. El principal inconveniente de este conjunto de datos es que la red más grande está formada por 58 nodos, lo que podría no ser lo suficientemente exigente teniendo en cuenta el tamaño actual de las redes sociales. Para mitigar este inconveniente, hemos añadido 8 instancias adicionales, con tamaños de 67 a 10312 nodos.

El mejor enfoque anterior es un método exacto que muestra sus límites cuando se trata de instancias más grandes y complejas. Con el fin de evaluar la contribución de nuestra propuesta, también se ha incluido un algoritmo metaheurístico adicional para realizar una comparación con SPR y DPR. En concreto, hemos seleccionado Simulated Annealing (SA), que es una metaheurística basada en la analogía entre un proceso de optimización y un proceso termodinámico conocido como recocido. Se trata de un método de búsqueda que trata de escapar de los óptimos locales, permitiendo explorar soluciones peores si dichas soluciones satisfacen ciertos criterios. SA fue propuesto originalmente por Kirkpatrick et al. [107] y se ha aplicado con éxito en una amplia variedad de problemas de optimización combinatoria duros. SA se ha aplicado con éxito en varios trabajos relacionados con problemas de maximización de influencia [108, 109]. Además, el conocido algoritmo de selección CELF [44], que ha sido ampliamente utilizado en el contexto de los problemas de maximización de influencia y, en particular, en Max-TSS [110], se incluye en la comparación. CELF es un procedimiento voraz que aprovecha la propiedad de submodularidad de la red para reducir considerablemente el esfuerzo computacional del algoritmo greedy hill-climbing. El objetivo principal de esta optimización es escalar a problemas grandes, alcanzando colocaciones casi óptimas. Esta mejora hace que CELF sea aproximadamente 700 veces más rápido que el procedimiento original. Existen varias implementaciones de SA disponibles públicamente. Para este trabajo, hemos seleccionado la proporcionada por la herramienta Metaheuristic Optimization framewoRK (MORK) [111], que ha sido probada en varios problemas de optimización [112, 113].

Los resultados se dividen en dos experimentos diferentes. En primer lugar, se evalúan SPR y DPR al considerar el conjunto de instancias originales en las que el método exacto es capaz de alcanzar el valor óptimo. En la Tabla 9.3 se muestran los resultados obtenidos. Como puede deducirse de los resultados, SPR se comporta ligeramente mejor que DPR en este conjunto de instancias, siendo capaz de alcanzar 79 de las 82 soluciones óptimas, mientras que DPR alcanza 76. Es importante destacar que la desviación media de ambos métodos, inferior a 0.05, indica que en aquellas instancias en las que ni SPR ni DPR son capaces de alcanzar el valor óptimo, se quedan realmente cerca de él. Para confirmar esta hipótesis, hemos realizado un test estadístico no paramétrico de Wilcoxon por pares entre SPR y el solver Gurobi, obteniendo un valor p igual a 0.109, lo que indica que, con un intervalo de confianza del 95 %, no existen diferencias estadísticamente significativas entre dichos métodos. Respecto al SA, cabe destacar que es capaz de alcanzar 76 de 82 instancias con una desviación del 4.86 %, requiriendo un tiempo despreciable al igual que SPR y DPR. Respecto a

CELF, el algoritmo requiere de tiempos de computación irrelevantes como DPR, SPR y SA, pero solo alcanza 61 de 82 soluciones óptimas, con una desviación del 12.19 %. A partir de estos resultados, podemos obtener dos conclusiones principales: SA es un algoritmo competitivo para el Max-TSS, y los DPR y SPR propuestos contribuyen significativamente a la calidad de las soluciones obtenidas, como se aprecia en la menor desviación respecto al valor óptimo.

Algoritmo	Pro.	Des. (%)	Tiempo (s)	#Óptimo
Gurobi	**45.38**	**0.00**	117.14	**82**
DPR	44.34	2.58	**0.01**	76
SPR	44.54	1.82	**0.01**	79
SA	46.31	**4.86**	**0.01**	76
CELF	42.07	12.19	**0.01**	61

Tabla 9.3: Comparación de SPR, DPR, SA, CELF y el solver de Gurobi al considerar el conjunto de datos original en el que Gurobi es capaz de alcanzar el valor óptimo.

El último experimento está dedicado a evaluar el rendimiento de los algoritmos propuestos y del solucionador Gurobi cuando se consideran las instancias más desafiantes y realistas. La Tabla 9.4 muestra los resultados obtenidos en el conjunto de instancias grandes. En este caso, mostramos los resultados desagregados, ya que se conforma con 8 instancias que pueden ser analizadas individualmente. Cabe destacar que el solver de Gurobi solo es capaz de proporcionar la solución óptima para 2 de las 8 instancias derivadas del nuevo conjunto de instancias complejas marcadas con un asterisco en el nombre de la instancia correspondiente. Para el resto de instancias, Gurobi ni siquiera es capaz de cargar el modelo en memoria, lo que pone de manifiesto la necesidad de considerar algoritmos metaheurísticos para este conjunto de instancias desafiantes. En particular, en las instancias en las que Gurobi alcanza el valor óptimo, SA, SPR y DPR también son capaces de encontrarlo. Sin embargo, CELF no es capaz de alcanzar el valor óptimo para estos dos casos. Además, para la instancia EMAIL-EUCORE, Gurobi requiere casi 30h para encontrar el valor óptimo, mientras que SA requiere 268s, DPR 262s y SPR solo 85s.

Analizando las instancias en las que Gurobi ni siquiera es capaz de cargar el modelo, SPR requiere de menor tiempo de computación que DPR en general, pero proporciona peores resultados en términos de calidad. En cuanto a SA, es capaz de proporcionar resultados competitivos en estos casos difíciles. En concreto, SPR alcanza la mejor solución en 3 de las 8 instancias, SA alcanza 5 de las 8 mejores soluciones y, por último, DPR alcanza todas las mejores soluciones excepto en dos instancias en las que CELF es capaz de proporcionar resultados ligeramente mejores. Cabe mencionar que CELF requiere desde aproximadamente cinco veces el tiempo de computación requerido por DPR, siendo, por tanto, DPR mucho más escalable para redes de gran escala. En términos de desviación, CELF proporciona los peores resultados con un 2.23 %, seguido de SA con un 1.71 %, pero es considerablemente mayor que la obtenida por SPR y DPR. En concreto, la desviación media obtenida por SPR es considerablemente pequeña (0.44 %) y DPR es capaz de alcanzar una desviación del 0.12 %. Dado que la desviación de DPR es realmente cercana al 0 %, realizamos una prueba estadística no

	CELF				SA			
Instancia	Pro.	Dev. (%)	Tiempo (s)	#Mejor	Pro.	Dev. (%)	Tiempo (s)	#Mejor
PRISON*	240	11.11	0.02	0	**270**	**0.00**	0.34	**1**
EMAIL-EU-CORE*	4672	1.79	22.71	0	**4757**	**0.00**	267.78	**1**
EGO-FACEBOOK	**19462**	**0.00**	3609.75	**1**	**19462**	**0.00**	1044.35	**1**
CA-GRQC	22487	5.05	12441.33	0	**23684**	**0.00**	5364.14	**1**
TWITCH_EN	25060	0.22	16833.33	0	23853	5.03	5885.88	0
LASTFM_ASIA	**25005**	**0.00**	26017.00	**1**	23000	8.02	6160.10	0
CA-HEPTH	44451	1.16	165624.79	0	**44972**	**0.00**	9105.73	**1**
BLOG_CATALOG3	**46732**	**0.00**	44674.80	**1**	46418	0.67	8407.40	0
Resumen	23514.13	2.23	33652.97	3	23302.00	1.71	4534.97	5

	SPR				DPR			
Instancia	Pro.	Dev. (%)	Tiempo (s)	#Mejor	Pro.	Dev. (%)	Tiempo (s)	#Mejor
PRISON*	**270**	**0.00**	**0.07**	**1**	**270**	**0.00**	0.16	**1**
EMAIL-EU-CORE*	**4757**	**0.00**	**84.48**	**1**	**4757**	**0.00**	262.59	**1**
EGO-FACEBOOK	**19462**	**0.00**	**279.47**	**1**	**19462**	**0.00**	769.70	**1**
CA-GRQC	23630	0.23	**442.07**	0	**23684**	**0.00**	2166.29	**1**
TWITCH_EN	24853	1.06	**680.77**	0	**25116**	**0.00**	2230.12	**1**
LASTFM_ASIA	24556	1.80	**760.97**	0	24780	0.90	2409.55	0
CA-HEPTH	44909	0.14	**2140.92**	0	**44972**	**0.00**	7743.06	**1**
BLOG_CATALOG3	46595	0.29	**1336.91**	0	46692	0.09	8705.44	0
Resumen	23629.00	0.44	**715.71**	3	**23716.63**	**0.12**	3035.87	**6**

Tabla 9.4: Comparación de CELF, SA, SPR y DPR sobre el conjunto de instancias mayores y más complejas.

paramétrica por pares de Wilcoxon para evaluar si existen diferencias estadísticamente significativas entre SPR y DPR. El valor p resultante de 0.04, inferior a 0.05, indica que DPR es estadísticamente mejor que SPR.

Estos resultados ponen de relieve la contribución de SPR y DPR al estado del arte de Max-TSS.

Como conclusión, tanto SPR como DPR son capaces de proporcionar soluciones prometedoras para el TSS, siendo cada una de ellas adecuada para diferentes situaciones. Por un lado, si el tiempo de cálculo es una restricción dura, recomendamos considerar SPR ya que la calidad de las soluciones no es drásticamente peor. Por otro lado, si el tiempo máximo de computación no es una parte crítica del problema, DPR es capaz de proporcionar mejores resultados en términos de calidad. Los algoritmos propuestos se han comparado con una implementación de Simulated Annealing, que se ha aplicado con éxito en varios problemas de maximización de influencia, y con CELF, que es un método ampliamente utilizado en el contexto de la maximización de influencia y, en particular, en Max-TSS. Los resultados obtenidos ponen de manifiesto la conveniencia de diseñar un algoritmo específico para la resolución del TSS cómo el propuesto en esta investigación.

9.5 Conclusiones

En este capítulo se presentan las conclusiones de cada variante de los problemas abordados y el trabajo futuro general de esta Tesis Doctoral. Sección 9.5.1, muestra las conclusiones de Social Network Influence Maximization Problem (SNIMP), Sección 9.5.2 resalta las conclusiones relacionadas con Budgeted Influence Maximization Problem (BIMP) y finalmente Sección 9.5.3 comenta las principales conclusiones sobre Target Set Selection (TSS). Cabe mencionar que en cada sección se presenta un enlace a un repositorio (con código fuente, instancias y resultados) para facilitar mayores comparaciones.

9.5.1 Conclusiones del problema de maximizar la influencia en redes sociales

En este artículo se presenta un algoritmo eficiente Greedy Randomized Adaptive Search Procedure (GRASP) para resolver el SNIMP. Se proponen dos procedimientos constructivos, siendo más competitivo el basado en la vecindad de dos pasos que el basado en el coeficiente de agrupamiento. Además, la idea de utilizar información local permite al algoritmo construir una solución completa en un tiempo reducido. Luego se presenta una búsqueda local basada en movimientos de intercambio. Dado que una exploración exhaustiva del espacio de búsqueda no es adecuado para este problema, proponemos una estrategia inteligente de exploración de vecindarios que limite la región del espacio de búsqueda a explorar, centrándose en las áreas más prometedoras. Esta lógica nos lleva a proporcionar soluciones de alta calidad en un tiempo de cálculo razonable, incluso para las instancias más grandes derivadas de redes sociales del mundo real, comúnmente consideradas en el área de maximización de la influencia. Dado que la estrategia de exploración inteligente del vecindario está parametrizada, si el tiempo de cálculo no es un factor relevante, la región explorada se puede ampliar fácilmente para encontrar mejores soluciones, aumentando así el esfuerzo computacional requerido. Este hecho hace que el algoritmo GRASP propuesto sea altamente escalable. Los resultados obtenidos están respaldados por la prueba de Friedman y luego la prueba de Wilcoxon por pares, confirmando la superioridad de la propuesta con respecto a los procedimientos de solución clásicos y de última generación en el área.

Este trabajo fue presentado y publicado en la revista Journal of Ambient Intelligence and Humanized Computing (JCR Q2) titulado *A quick GRASP-based method for influence maximization in social networks* [69]. El factor de impacto de esta revista es 3.662, ubicándose en 68/145 en el área de Informática, Inteligencia Artificial y 73/164 en Sistemas de Información. El código fuente también se ha puesto a disposición del público[2] para facilitar una mayor comparación. Cabe mencionar que esta investigación también ha sido presentada en: *Parallel Problem Solving from Nature–PPSN XVI: 16th International Conference, PPSN*, celebrado en Leiden, Países Bajos, del 5 al 9 de septiembre de 2020; *International Conference on Variable Neighborhood Search (ICVNS)* celebrada en Abu Dhabi, Emiratos Árabes Unidos, del 22 al 24 de marzo

[2]`https://grafo.etsii.urjc.es/SNIMP`

de 2021; y *XIX Conferencia de la Asociación Española para la Inteligencia Artificial (CAEPIA 2021): avances en Inteligencia Artificial* celebrada en Málaga del 22 al 24 de septiembre de 2021. Más detalles y el artículo completo se pueden encontrar en el Capítulo 6.

9.5.2 Conclusiones del problema de maximizar la influencia con un presupuesto

Se desarrolló un algoritmo eficiente, efectivo y escalable basado en el marco Greedy Randomized Adaptive Search Procedure (GRASP). Este marco se puede configurar según los requisitos de tiempo. La escalabilidad se logra utilizando información local en métodos constructivos y evitando una búsqueda exhaustiva en la búsqueda local (seleccionando los nodos más prometedores). Los resultados y la comparación con los algoritmos anteriores se desarrollan utilizando tres modelos de difusión de la influencia diferentes para el problema, considerando el algoritmo probabilístico Monte Carlo para la evaluación de la función objetivo.

Finalmente, se analizó un estudio de caso de infodemia desde la perspectiva de la maximización de influencia. Específicamente, se construyó una instancia basada en 386384 tweets sobre la Ley Estadounidense de Atención Médica (AHCA). Se realizó un experimento que muestra la superioridad de GRASP al compararlo con el algoritmo previo llamado *ComBIM* en 21 de 27 instancias disponibles. Se analizaron los usuarios más influyentes, mostrando su relevancia en el tema estudiado, siendo la mayoría senadores, humoristas, escritores o periódicos.

El capítulo 7 muestra el trabajo aceptado titulado *An efficient and effective GRASP algorithm for the Budget Influence Maximization Problem* [65], en la revista *Journal of Ambient Intelligence and Humanized Computing.* Esta revista tiene un factor de impacto de 3.662, ubicándose en 68/145 en el área de Informática, Inteligencia Artificial y 73/164 en Sistemas de Información. Al igual que el trabajo anterior, para facilitar mayores comparaciones, el código fuente, las instancias y los resultados completos se pueden encontrar públicamente disponibles en el siguiente enlace [3].

9.5.3 Conclusiones del problema de seleccionar el conjunto objetivo

Este trabajo presenta un algoritmo basado en Path Relinking [27] para resolver el problema TSS. Se ha demostrado el potencial de GRASP junto con una estrategia combinada de Path Relinking. Especialmente, en cuanto al tiempo de cómputo requerido por el algoritmo para alcanzar soluciones de alta calidad, la propuesta surgió como el mejor método en el estado del arte.

En particular, las principales aportaciones de este trabajo son las siguientes. Se han propuesto dos variantes diferentes de GRASP usando dos variantes de Path Relinking: Path Relinking Estático (SPR) y Path Relinking Dinámico (DPR). Tanto SPR

[3]`https://grafo.etsii.urjc.es/BIMP`

como DPR pueden proporcionar soluciones prometedoras para TSS, siendo cada una de ellas adecuada para diferentes situaciones. Por un lado, si el tiempo de cálculo es una limitación importante, recomendamos considerar SPR ya que la calidad de las soluciones no son drásticamente peor. Por otro lado, si el tiempo máximo de cálculo no es una parte crítica del problema, DPR puede proporcionar mejores resultados en términos de calidad.

Cabe mencionar que la estrategia del camino entre dos soluciones de Path Relinking (Reactive Path Relinking) fue propuesta y aceptada en un trabajo relacionado en esta tesis [94] (ver Sección 3.2).

Se realizaron nuevas mejoras para reducir el esfuerzo computacional del método de búsqueda local. Para ello se usó un sistema para evitar la exploración de soluciones ya visitadas utilizando la función hash en las soluciones. Luego, cada vez que se visita una solución, se evalúa si su código hash correspondiente no se ha incluido ya en el conjunto de soluciones visitadas. También se limitaron los nodos explorados durante la búsqueda, descartándose aquellos nodos que darán como resultado una solución inviable. Estos nodos se encuentran ordenados con respecto a su valor de esfuerzo en orden ascendente. Por lo tanto, solo se exploran aquellos nodos cuyo valor de esfuerzo es menor o igual al presupuesto disponible. El objetivo de la última mejora es reducir el tiempo de cálculo necesario para evaluar la influencia de un nodo almacenándolo en caché.

Los experimentos han demostrado que la combinación de GRASP con PR da como resultado soluciones de alta calidad. La implementación eficiente del algoritmo y la calidad de las heurísticas aplicadas permiten que el algoritmo supere el trabajo anterior, apoyado en pruebas estadísticas.

Los resultados del trabajo fueron publicados en una revista con el título, *Dynamic Path Relinking for the Target Set Selection problem* [114] en Knowledge-Based Systems la cual tiene un factor de impacto de 8.800 y se ubica en 19/145 en el campo de la Informática, Inteligencia Artificial. El documento completo se incluye en el Capítulo 8 de la Parte II, el código fuente completo y las instancias están disponibles públicamente para facilitar comparaciones adicionales, así como los resultados completos[4]. Cabe mencionar que esta investigación también ha sido presentada en el *XL Congreso Nacional de Estadística e Investigación Operativa (SEIO)* celebrado en Elche, España, del 7 al 10 de noviembre de 2023.

[4]https://grafo.etsii.urjc.es/TSS

Bibliography

[1] W. Dunham, Euler: The master of us all, vol. 22. American Mathematical Society, 2022.

[2] B. Beavis and I. Dobbs, Optimisation and stability theory for economic analysis. Cambridge university press, 1990.

[3] A. Schrijver, Theory of linear and integer programming. John Wiley & Sons, 1998.

[4] G. Dantzig, Linear programming and extensions. Princeton university press, 1963.

[5] G. B. Dantzig, A. Orden, P. Wolfe, et al., “The generalized simplex method for minimizing a linear form under linear inequality restraints,” Pacific Journal of Mathematics, vol. 5, no. 2, pp. 183–195, 1955.

[6] S. H. Zanakis and J. R. Evans, “Heuristic “optimization”: Why, when, and how to use it,” Interfaces, vol. 11, no. 5, pp. 84–91, 1981.

[7] F. Glover, “Future paths for integer programming and links to artificial intelligence,” Computers & operations research, vol. 13, no. 5, pp. 533–549, 1986.

[8] L. N. Trefethen, Approximation Theory and Approximation Practice, Extended Edition. SIAM, 2019.

[9] J. H. Holland, Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence. MIT press, 1992.

[10] M. A. Boschetti, V. Maniezzo, M. Roffilli, and A. Bolufé Röhler, “Matheuristics: Optimization, simulation and control,” in International workshop on hybrid metaheuristics, pp. 171–177, Springer, 2009.

[11] C. H. Papadimitriou and K. Steiglitz, Combinatorial optimization: algorithms and complexity. Courier Corporation, 1998.

[12] J. Hartmanis, “Computers and intractability: a guide to the theory of np-completeness,” Siam Review, vol. 24, no. 1, p. 90, 1982.

[13] C. E. Leiserson, R. L. Rivest, T. H. Cormen, and C. Stein, Introduction to algorithms, vol. 3. MIT press Cambridge, MA, USA, 1994.

[14] S. S. Skiena, The algorithm design manual, vol. 2. Springer, 1998.

[15] N. Metropolis, “The beginning,” Los Alamos Science, vol. 15, pp. 125–130, 1987.

[16] M. R. Garey, R. L. Graham, and D. S. Johnson, “Some np-complete geometric problems,” in Proceedings of the eighth annual ACM symposium on Theory of computing, pp. 10–22, 1976.

[17] D. Kempe, J. Kleinberg, and É. Tardos, “Maximizing the spread of influence through a social network,” in Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 137–146, ACM Press, 2003.

[18] G. Polya, How to solve it: A new aspect of mathematical method, vol. 85. Princeton university press, 2004.

[19] L. A. Wolsey and G. L. Nemhauser, Integer and combinatorial optimization, vol. 55. John Wiley & Sons, 1999.

[20] E. A. Silver, R. Victor, V. Vidal, and D. de Werra, “A tutorial on heuristic methods,” European Journal of Operational Research, vol. 5, no. 3, pp. 153–162, 1980.

[21] T. Stützle and R. Ruiz, “Iterated greedy.,” Handbook of heuristics, pp. 547–577, 2018.

[22] J. P. Hart and A. W. Shogan, “Semi-greedy heuristics: An empirical study,” Operations Research Letters, vol. 6, no. 3, pp. 107–114, 1987.

[23] T. A. Feo, M. G. C. Resende, and S. H. Smith, “A greedy randomized adaptive search procedure for maximum independent set,” Operations Research, vol. 42, pp. 860–878, oct 1994.

[24] W. Michiels, E. H. Aarts, and J. Korst, “Theory of local search,” Handbook of heuristics, pp. 299–339, 2018.

[25] S. H. Zanakis, J. R. Evans, and A. A. Vazacopoulos, “Heuristic methods and applications: a categorized survey,” European Journal of Operational Research, vol. 43, no. 1, pp. 88–110, 1989.

[26] T. A. Feo and M. G. Resende, “A probabilistic heuristic for a computationally difficult set covering problem,” Operations Research Letters, vol. 8, pp. 67–71, apr 1989.

[27] M. Laguna and R. Marti, “Grasp and path relinking for 2-layer straight line crossing minimization,” INFORMS Journal on Computing, vol. 11, no. 1, pp. 44–52, 1999.

[28] S. S. Singh, V. Srivastava, A. Kumar, S. Tiwari, D. Singh, and H.-N. Lee, “Social network analysis: A survey on measure, structure, language information analysis, privacy, and applications,” ACM Trans. Asian Low-Resour. Lang. Inf. Process., vol. 22, may 2023.

[29] A. Tretiakov, A. Martín, and D. Camacho, “Detection of false information in spanish using machine learning techniques,” in Intelligent Data Engineering and

Automated Learning – IDEAL 2022 (H. Yin, D. Camacho, and P. Tino, eds.), (Cham), pp. 42–53, Springer International Publishing, 2022.

[30] Z. Reza, A. M. Ali, and L. Huan, Social Media Mining: An Introduction. Cambridge University Press, 2014.

[31] A.-L. Barabási and M. Pósfai, Network science. Cambridge: Cambridge University Press, 2016.

[32] T. Nguyen Hung, T. Thai My, and N. Dinh Thang, "Stop-and-stare: Optimal sampling algorithms for viral marketing in billion-scale networks," in Proceedings of the 2016 International Conference on Management of Data, SIGMOD '16, (New York, NY, USA), p. 695–710, Association for Computing Machinery, 2016.

[33] C. Luo, K. Cui, X. Zheng, and D. Zeng, "Time critical disinformation influence minimization in online social networks," 2014 IEEE Joint Intelligence and Security Informatics Conference, pp. 68–74, 2014.

[34] W. Xinjue, D. Ke, L. Jianxin, Y. J. Xu, S. Jensen Christian, and Y. Xiaochun, "Targeted influence minimization in social networks," in Advances in Knowledge Discovery and Data Mining (P. Dinh, T. V. S., W. G. I., H. Bao, G. Mohadeseh, and R. Lida, eds.), (Cham), pp. 689–700, Springer International Publishing, 2018.

[35] Y. Qipeng, S. Ruisheng, Z. Chuan, W. Peng, and G. Li, "Topic-aware social influence minimization," in Proceedings of the 24th International Conference on World Wide Web, WWW '15 Companion, (New York, NY, USA), p. 139–140, Association for Computing Machinery, 2015.

[36] S. Banerjee, M. Jenamani, and D. K. Pratihar, "A survey on influence maximization in a social network," Knowledge and Information Systems, vol. 62, no. 9, pp. 3417–3455, 2020.

[37] Z. Aghaee, M. M. Ghasemi, H. A. Beni, A. Bouyer, and A. Fatemi, "A survey on meta-heuristic algorithms for the influence maximization problem in the social networks," Computing, vol. 103, no. 11, pp. 2437–2477, 2021.

[38] A. D'angelo, A. Agarwal, K.-X. Jin, Y.-F. Juan, L. Klots, O. Moskalyuk, and Y. Wong, "Targeting advertisements in a social network," Mar. 2009. US Patent App. 12/195,321.

[39] A. S. Klovdahl, "Social networks and the spread of infectious diseases: The AIDS example," Social Science & Medicine, vol. 21, pp. 1203–1216, Jan. 1985.

[40] E. Chen, K. Lerman, and E. Ferrara, "Tracking social media discourse about the covid-19 pandemic: Development of a public coronavirus twitter data set," JMIR Public Health Surveill, vol. 6, p. e19273, May 2020.

[41] M. G. Rodriguez, D. Balduzzi, and B. Schölkopf, "Uncovering the temporal dynamics of diffusion networks," arXiv preprint arXiv:1105.0697, 2011.

[42] M. Gomez-Rodriguez, J. Leskovec, and A. Krause, "Inferring networks of diffusion and influence," ACM Transactions on Knowledge Discovery from Data (TKDD), vol. 5, no. 4, pp. 1–37, 2012.

[43] A. Guille, H. Hacid, C. Favre, and D. A. Zighed, "Information diffusion in online social networks: A survey," ACM Sigmod Record, vol. 42, no. 2, pp. 17–28, 2013.

[44] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and N. Glance, "Cost-effective outbreak detection in networks," in Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 420–429, ACM Press, 2007.

[45] S. Banerjee, M. Jenamani, and D. K. Pratihar, "ComBIM: A community-based solution approach for the budgeted influence maximization problem," Expert Systems with Applications, vol. 125, pp. 1–13, jul 2019.

[46] M. Gong, J. Yan, B. Shen, L. Ma, and Q. Cai, "Influence maximization in social networks based on discrete particle swarm optimization," Information Sciences, vol. 367-368, pp. 600–614, Nov. 2016.

[47] D. Kempe, J. Kleinberg, and E. Tardos, "Maximizing the spread of influence through a social network," Theory of Computing, vol. 11, no. 1, pp. 105–147, 2015.

[48] M. Granovetter, "Threshold models of collective behavior," American journal of sociology, vol. 83, no. 6, pp. 1420–1443, 1978.

[49] M. Bicher, M. Wastian, D. Brunmeir, and N. Popper, "Review on monte carlo simulation stopping rules: how many samples are really enough?," Simul. Notes Eur., vol. 32, no. 1, pp. 1–8, 2022.

[50] J. Xie, F. Zhang, K. Wang, X. Lin, and W. Zhang, "Minimizing the influence of misinformation via vertex blocking," 2023.

[51] K. Li, L. Zhang, and H. Huang, "Social influence analysis: models, methods, and evaluation," Engineering, vol. 4, no. 1, pp. 40–46, 2018.

[52] K. Esselink, L. Loyens, and B. Smit, "Parallel monte carlo simulations," Physical Review E, vol. 51, no. 2, p. 1560, 1995.

[53] J. Goldenberg, B. Libai, and E. Muller, "Talk of the network: A complex systems look at the underlying process of word-of-mouth," Marketing letters, vol. 12, pp. 211–223, 2001.

[54] T. M. Liggett and T. M. Liggett, Interacting particle systems, vol. 2. Springer, 1985.

[55] M. Richardson and P. Domingos, "Mining knowledge-sharing sites for viral marketing," in Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 61–70, 2002.

[56] W. Chen, C. Wang, and Y. Wang, "Scalable influence maximization for prevalent viral marketing in large-scale social networks," in Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '10, (New York, NY, USA), p. 1029–1038, Association for Computing Machinery, 2010.

[57] H. Nguyen and R. Zheng, "On budgeted influence maximization in social networks," IEEE Journal on Selected Areas in Communications, vol. 31, pp. 1084–1094, jun 2013.

[58] J. L. Hayes, B. C. Britt, W. Evans, S. W. Rush, N. A. Towery, and A. C. Adamson, "Can social media listening platforms' artificial intelligence be trusted? examining the accuracy of crimson hexagon's (now brandwatch consumer research's) ai-driven analyses," Journal of Advertising, vol. 50, no. 1, pp. 81–91, 2021.

[59] S. V. Ravelo, C. N. Meneses, and E. A. Anacleto, "NP-hardness and evolutionary algorithm over new formulation for a target set selection problem," 2020 IEEE Congress on Evolutionary Computation (CEC), jul 2020.

[60] S. V. Ravelo and C. N. Meneses, "Generalizations, formulations and subgradient based heuristic with dynamic programming procedure for target set selection problems," Computers & Operations Research, vol. 135, p. 105441, Nov. 2021.

[61] J. V. Chen, B. chiuan Su, and A. E. Widjaja, "Facebook c2c social commerce: A study of online impulse buying," Decision Support Systems, vol. 83, pp. 57–69, 2016.

[62] S. Ryu and J. Park, "The effects of benefit-driven commitment on usage of social media for shopping and positive word-of-mouth," Journal of Retailing and Consumer Services, vol. 55, p. 102094, 2020.

[63] D. M. J. Lazer, M. A. Baum, Y. Benkler, A. J. Berinsky, K. M. Greenhill, F. Menczer, M. J. Metzger, B. Nyhan, G. Pennycook, D. Rothschild, M. Schudson, S. A. Sloman, C. R. Sunstein, E. A. Thorson, D. J. Watts, and J. L. Zittrain, "The science of fake news," Science, vol. 359, pp. 1094–1096, mar 2018.

[64] P. A. Dreyer and F. S. Roberts, "Irreversible k-threshold processes: Graph-theoretical threshold models of the spread of disease and of opinion," Discrete Applied Mathematics, vol. 157, pp. 1615–1627, apr 2009.

[65] I. Lozano-Osorio, J. Sánchez-Oro, and A. Duarte, "An efficient and effective grasp algorithm for the budget influence maximization problem," Journal of Ambient Intelligence and Humanized Computing, Sep 2023.

[66] F. Jia, K. Zhou, C. Kamhoua, and Y. Vorobeychik, "Blocking adversarial influence in social networks," in Decision and Game Theory for Security: 11th International Conference, GameSec 2020, College Park, MD, USA, October 28–30, 2020, Proceedings 11, pp. 257–276, Springer, 2020.

[67] H. Loucif, M. e.-B. el Ibrahimi, A. Boubetra, and S. Akrouf, "A new recursive model for ranking web users in facebook based on their social influence," in Proceedings of the 3rd International Conference on Software Engineering and New Technologies, Hammamet, Tunisia, pp. 20–22, 2014.

[68] D. Bucur, G. Iacca, A. Marcelli, G. Squillero, and A. Tonda, "Evaluating surrogate models for multi-objective influence maximization in social networks," in Proceedings of the Genetic and Evolutionary Computation Conference Companion, pp. 1258–1265, ACM Press, 2018.

[69] I. Lozano-Osorio, J. Sánchez-Oro, A. Duarte, and Ó. Cordón, "A quick GRASP-based method for influence maximization in social networks," Journal of Ambient Intelligence and Humanized Computing, Sept. 2021.

[70] D. Bucur and G. Iacca, "Influence maximization in social networks with genetic algorithms," in Applications of Evolutionary Computation, pp. 379–392, Springer International Publishing, 2016.

[71] A. Gupta, I. Khatri, A. Choudhry, P. Chandhok, D. K. Vishwakarma, and M. Prasad, "A spreader ranking algorithm for extremely low-budget influence maximization in social networks using community bridge nodes," arXiv preprint arXiv:2211.09657, 2022.

[72] S. Kumar, A. Mallik, and B. Panda, "Influence maximization in social networks using transfer learning via graph-based lstm," Expert Systems with Applications, vol. 212, p. 118770, 2023.

[73] G. Cordasco, L. Gargano, and A. A. Rescigno, "Influence propagation over large scale social networks," in Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015, pp. 1531–1538, 2015.

[74] L. Sun, W. Huang, P. S. Yu, and W. Chen, "Multi-round influence maximization," in Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining, pp. 2249–2258, ACM, July 2018.

[75] N. Chen, "On the approximability of influence in social networks," in Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA '08, (USA), p. 1029–1037, Society for Industrial and Applied Mathematics, 2008.

[76] Y. Liu, X. Wang, and J. Kurths, "Framework of evolutionary algorithm for investigation of influential nodes in complex networks," IEEE Transactions on Evolutionary Computation, vol. 23, pp. 1049–1063, Dec. 2019.

[77] N. Barbieri, F. Bonchi, and G. Manco, "Topic-aware social influence propagation models," in 2012 IEEE 12th International Conference on Data Mining, pp. 81–90, 2012.

[78] G. Lawyer, "Understanding the influence of all nodes in a network," Scientific Reports, vol. 5, Mar. 2015.

[79] G. C. Resende Mauricio, R. Martí, M. Gallego, and A. Duarte, "GRASP and path relinking for the max–min diversity problem," Computers & Operations Research, vol. 37, pp. 498–508, Mar. 2010. Hybrid Metaheuristics.

[80] G. C. Resende Mauricio and C. C. Ribeiro, "GRASP: Greedy randomized adaptive search procedures," in Search Methodologies, pp. 287–312, Springer US, July 2013.

[81] G. C. Resende Mauricio and C. C. Ribeiro, "Greedy randomized adaptive search procedures: Advances, hybridizations, and applications," in Handbook of Metaheuristics, pp. 283–319, Springer US, 2010.

[82] C. C. Ribeiro, P. Hansen, S. Binato, and G. C. Oliveira, “A reactive grasp for transmission network expansion planning,” Essays and surveys in metaheuristics, pp. 81–100, 2002.

[83] M. Prais and C. C. Ribeiro, “Reactive grasp: An application to a matrix decomposition problem in tdma traffic assignment,” INFORMS Journal on Computing, vol. 12, no. 3, pp. 164–176, 2000.

[84] C. C. Ribeiro, P. Hansen, S. Binato, W. Hery, D. Loewenstern, and M. Resende, “A grasp for job shop scheduling,” Essays and surveys in metaheuristics, pp. 59–79, 2002.

[85] M. Prais and C. C. Ribeiro, “Parameter variation in grasp implementations,” in Extended abstracts of the third metaheuristics international conference, pp. 375–380, 1999.

[86] M. Prais and C. C. Ribeiro, “Parameter variation in grasp procedures,” Investigación Operativa, vol. 9, no. 1, pp. 1–20, 2000.

[87] A. Díaz, F. Glover, H. Ghaziri, J. González, M. Laguna, P. Moscato, and F. T. Tseng, “Optimización heurística y redes neuronales,” 2000.

[88] F. Glover and M. Laguna, Tabu search. Springer, 1998.

[89] R. M. Aiex, M. G. Resende, P. M. Pardalos, and G. Toraldo, “Grasp with path relinking for three-index assignment,” INFORMS Journal on Computing, vol. 17, no. 2, pp. 224–247, 2005.

[90] M. G. Resende, R. Martí, M. Gallego, and A. Duarte, “Grasp and path relinking for the max–min diversity problem,” Computers & Operations Research, vol. 37, no. 3, pp. 498–508, 2010.

[91] S. Binato, H. Faria Jr, and M. G. Resende, “Greedy randomized adaptive path relinking,” in Proceedings of the IV Metaheuristics International Conference, pp. 393–397, 2001.

[92] A. Duarte, J. Sánchez-Oro, M. G. Resende, F. Glover, and R. Martí, “Greedy randomized adaptive search procedure with exterior path relinking for differential dispersion minimization,” Information Sciences, vol. 296, pp. 46–60, Mar. 2015.

[93] A. Duarte, J. Sánchez-Oro, M. G. Resende, F. Glover, and R. MARTÍ, “Grasp with exterior path relinking for differential dispersion minimization,” Information Sciences, In press. TABLA V, 2014.

[94] I. Lozano-Osorio, J. Sánchez-Oro, A. D. López-Sánchez, and A. Duarte, “A reactive path relinking algorithm for solving the bi-objective p-median and p-dispersion problem,” Soft Computing, vol. 27, pp. 8029–8059, Mar. 2023.

[95] C. C. Ribeiro and M. G. Resende, “Path-relinking intensification methods for stochastic local search algorithms,” Journal of Heuristics, vol. 18, p. 193–214, apr 2012.

[96] F. Glover, “Scatter search and path relinking,” New ideas in optimization, vol. 138, 1999.

[97] M. G. Resende and R. F. Werneck, "A hybrid heuristic for the p-median problem," Journal of heuristics, vol. 10, pp. 59–88, 2004.

[98] R. Martí, M. Laguna, and F. Glover, "Principles of scatter search," european Journal of operational Research, vol. 169, no. 2, pp. 359–372, 2006.

[99] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world' networks," Nature, vol. 393, pp. 440–442, June 1998.

[100] C. Salavati and A. Abdollahpouri, "Identifying influential nodes based on ant colony optimization to maximize profit in social networks," Swarm and Evolutionary Computation, vol. 51, p. 100614, Dec. 2019.

[101] A. Goyal, W. Lu, and L. V. S. Lakshmanan, "*CELF++*: Optimizing the greedy algorithm for influence maximization in social networks," in Proceedings of the 20th international conference companion on World wide web - WWW '11, ACM Press, 2011.

[102] M. Gong, C. Song, C. Duan, L. Ma, and B. Shen, "An efficient memetic algorithm for influence maximization in social networks," IEEE Computational Intelligence Magazine, vol. 11, pp. 22–33, Aug. 2016.

[103] M. G. Resende and C. C. Ribeiro, GRASP with Path-Relinking: Recent Advances and Applications, pp. 29–63. Boston, MA: Springer US, 2005.

[104] V. Campos, R. Martí, J. Sánchez-Oro, and A. Duarte, "GRASP with path relinking for the orienteering problem," Journal of the Operational Research Society, vol. 65, pp. 1800–1813, Dec. 2014.

[105] S. Pérez-Peló, J. Sánchez-Oro, and A. Duarte, "Finding weaknesses in networks using greedy randomized adaptive search procedure and path relinking," Expert Systems, vol. 37, no. 6, p. e12540, 2020.

[106] D. Cuellar-Usaquén, C. Gomez, and D. Álvarez-Martínez, "A GRASP/path-relinking algorithm for the traveling purchaser problem," International Transactions in Operational Research, vol. 30, pp. 831–857, Apr. 2021.

[107] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," Science, vol. 220, no. 4598, pp. 671–680, 1983.

[108] Q. Jiang, G. Song, C. Gao, Y. Wang, W. Si, and K. Xie, "Simulated annealing based influence maximization in social networks," Proceedings of the AAAI Conference on Artificial Intelligence, vol. 25, pp. 127–132, Aug. 2011.

[109] S.-J. Liu, C.-Y. Chen, and C.-W. Tsai, "An effective simulated annealing for influence maximization problem of online social networks," Procedia Computer Science, vol. 113, pp. 478–483, 2017. The 8th International Conference on Emerging Ubiquitous Systems and Pervasive Networks (EUSPN 2017) / The 7th International Conference on Current and Future Trends of Information and Communication Technologies in Healthcare (ICTH-2017) / Affiliated Workshops.

[110] C. Wang, L. Deng, G. Zhou, and M. Jiang, "A global optimization algorithm for target set selection problems," Information Sciences, vol. 267, pp. 101–118, 2014.

[111] R. Martín-Santamaría, S. Cavero, A. Herrán, A. Duarte, and J. M. Colmenar, “A practical methodology for reproducible experimentation: an application to the Double-row Facility Layout Problem,” Evolutionary Computation, pp. 1–35, 11 2022.

[112] I. Lozano-Osorio, J. Sánchez-Oro, A. Martínez-Gavara, A. D. López-Sánchez, and A. Duarte, “An efficient fixed set search for the covering location with interconnected facilities problem,” in Metaheuristics (L. Di Gaspero, P. Festa, A. Nakib, and M. Pavone, eds.), (Cham), pp. 485–490, Springer International Publishing, 2023.

[113] J. Yuste, E. G. Pardo, and A. Duarte, “Variable neighborhood descent for software quality optimization,” in Metaheuristics (L. Di Gaspero, P. Festa, A. Nakib, and M. Pavone, eds.), (Cham), pp. 531–536, Springer International Publishing, 2023.

[114] I. Lozano-Osorio, A. Oliva-García, and J. Sánchez-Oro, “Dynamic path relinking for the target set selection problem,” Knowledge-Based Systems, vol. 278, p. 110827, 2023.

[115] S. Pei, L. Muchnik, S. Tang, Z. Zheng, and H. A. Makse, “Exploring the complex pattern of information spreading in online blog communities,” PLOS ONE, vol. 10, pp. 1–18, 05 2015.

[116] H. G. Gauch, Scientific method in practice. Cambridge University Press, 2003.

[117] E.-G. Talbi, Metaheuristics: from design to implementation. John Wiley & Sons, 2009.

[118] R. S. Barr, B. L. Golden, J. P. Kelly, M. G. Resende, and W. R. Stewart, “Designing and reporting on computational experiments with heuristic methods,” Journal of heuristics, vol. 1, pp. 9–32, 1995.

[119] B. L. Golden and A. A. Assad, “A decision-theoretic framework for comparing heuristics,” European journal of operational research, vol. 18, no. 2, pp. 167–171, 1984.