Politecnico di Torino

Politecnico di Torino

Master's Degree in Mathematical Engineering

2025/2026

Intervention Design for Influence Maximization in Linear Threshold Model on Networks

Supervisors:
Giacomo Como
Fabio Fagnani

Candidate:
Alessio Dematttia

Abstract

Understanding the mechanisms governing the diffusion of behaviors, opinions, and innovations in social groups represents a major challenge in social science and network theory. The Linear Threshold Model (LTM) on graphs provides a principled mathematical framework for this phenomenon, modelling each individual as a node that adopts a new behavior if the cumulative influence from already-active neighbors exceeds a personal resistance threshold. In this thesis we study the optimization problem faced by a social planner aiming to induce complete adoption through propagation while minimizing the total cost of intervention. We consider two complementary intervention strategies: targeting, which selects a subset of individuals for direct activation, and partial incentive, which reduces individual thresholds to facilitate organic diffusion. Both problems are known to be NP-hard on arbitrary graphs; nevertheless, we establish that specific graph topologies admit polynomial-time exact algorithms. We propose novel efficient algorithms for complete graphs with heterogeneous costs, including a reduction to minimum-weight perfect matching and a dynamic programming solution for the time-constrained variant, and derive optimal algorithms for complete multipartite graphs. We further extend existing results to paths, trees, and cycles, providing a unified treatment of low-degree topologies via dynamic programming. For the general intractable case, we develop and analyze a distributed, iterative, randomized algorithm based on simulated annealing over the space of permutations, derive theoretical guarantees on its mixing and convergence rate via spectral gap analysis, and demonstrate its practical effectiveness through extensive experiments on large-scale real-world social networks.

Pour la Patrie, les Sciences et la Gloire

Motto of the École Polytechnique

Table of Contents

Introduction

The study of how behaviors, opinions, and innovations spread through a population is a central theme in sociology and economics. In most real-world settings, individuals do not make decisions in isolation: the choice to adopt a new technology, join a political movement, or purchase a product is shaped not only by personal preferences, but also by the behavior of peers. This simple observation — that social influence is a primary driver of collective behavior — has profound consequences for how cascades of adoption unfold across a network, and raises a natural question: who should be targeted, and at what cost, to trigger large-scale adoption?

A foundational contribution in this direction is due to Granovetter [1], who was the first to formalize a model of behavioral diffusion capturing this tension. In his framework, each individual facing a binary choice is characterized by a personal *threshold*: the minimum number of peers who must adopt before the individual does so, reflecting the point at which social pressure overcomes intrinsic resistance. Granovetter applied this model to a wide range of collective phenomena — riot participation, strikes, voting, and the diffusion of innovations — observing that groups with similar average preferences can produce dramatically different aggregate outcomes depending on the precise distribution of thresholds across individuals. A key consequence is that the network structure of social interactions, not just individual attitudes, is a primary determinant of collective behavior.

The breadth of this framework is reflected in its empirical reach. In the commercial domain, Domingos and Richardson [2] were among the first to exploit the network structure of customer interactions for viral marketing, arguing that the value of a customer should account for the indirect influence they exert on future adopters through word-of-mouth. In the political domain, González-Bailón et al. [3] provided direct empirical evidence of threshold dynamics in collective action, studying recruitment patterns on Twitter during the 2011 Spanish protests and finding clear signatures of social influence and complex contagion: participation spread in cascades, with each individual's decision shaped by the prior choices of their contacts. Similar mechanisms have been documented in the adoption of medical innovations and new technologies [4], and in the emergence of social norms more broadly.

Despite this long empirical tradition, the field underwent a decisive change of perspective in the early 2000s, driven by the rise of large-scale online social platforms. For the first time, it became possible to observe diffusion processes at massive scale with fine-grained data: network structure, adoption timing, and information cascades became measurable across millions of individuals. This empirical revolution shifted attention from descriptive models to computational and algorithmic questions. The landmark contribution in this direction is due to Kempe et al. [5], who reformulated influence maximization as a discrete combinatorial optimization problem: given a social network and a budget k, which k individuals should be initially targeted to maximize the expected spread of adoption? Proving NP-hardness under both the Linear Threshold and the Independent Cascade models, they showed that a greedy algorithm achieves a $(1 - 1/e)$-approximation, when thresholds are random, by exploiting the submodularity of the influence function. This work established influence maximization as a central problem at the intersection of algorithms and network science, and triggered a large body of subsequent research.

Subsequent work established that the deterministic version of the problem, in which individual thresholds are fixed rather than drawn at random, is significantly harder. Chen [6] introduced the Target Set Selection (TSS) problem, i.e. find the minimum-cardinality set of nodes whose initial activation guarantees complete adoption under fixed thresholds. Chen proved that TSS is NP-hard and cannot be approximated within a factor of $2^{\log^{1-\varepsilon}|V|}$ for any fixed $\varepsilon > 0$, unless NP $\subseteq$ DTIME($|V|^{\text{polylog}|V|}$), making it one of the hardest combinatorial optimization problems from an approximation standpoint.

This thesis is situated within Chen's framework. We study the LTM intervention problem from an algorithmic standpoint, seeking to minimize the cost of intervention necessary to guarantee complete adoption.

Related work

Two intervention problems have been studied in the literature. The first is the *influence maximization* problem: given a budget, find the intervention that maximizes the number of nodes eventually adopting. The second, which is the focus of this thesis, is the *cost minimization* problem: find the cheapest intervention guaranteeing full adoption, or a certain partial adoption. The latter is also known as *Least Cost Intervention Problem* (LCIP). Two complementary intervention strategies are considered: *targeting*, which directly activates a subset of individuals, and *partial incentive*, which reduces individual thresholds to facilitate organic diffusion through the network.

On the influence maximization side, a large body of work builds on the approximation framework of Kempe et al. [5]. Chen et al. [7] developed scalable heuristics for the Linear Threshold model based on local DAG approximations. More recently, Cordasco et al. [8] addressed the problem of finding small sets that influence networks under time constraints.

On the cost minimization side exact polynomial-time algorithms are known for specific graph families. Como et al. [9, 10] introduced a resistance distribution function characterizing LTM dynamics on complete graphs, obtaining an $O(n)$ algorithm for the Target Set Selection (TSS) with identity costs and establishing a combinatorial reformulation of the intervention problem over the symmetric group $\mathcal{S}_n$ as $\min_{\sigma \in \mathcal{S}_n} C(\sigma)$. Cordasco et al. [11] proposed an $O(|E| \log |V|)$ greedy algorithm for the Weighted TSS (WTSS) with provable upper bounds, and that solves exactly the problem on complete graphs under a monotonicity assumption on costs. For trees, Günnec et al. [12] developed an $O(n)$ dynamic programming algorithm for the LCIP with linear costs and established NP-hardness for the general weighted case. Raghavan et al. [13] obtained linear-time algorithms for TSS and WTSS on trees and cycles. For general cases, branch-and-cut methods were proposed by Günnec et al. [14] and Fischetti et al. [15].

Thesis objectives and structure

The primary objective of this thesis is to develop new exact algorithms and metaheuristics for the LTM cost-minimization intervention problem, extending the state of the art along four directions. On complete graphs with homogeneous edge weights, our contributions are threefold. We resolve the WTSS problem without any monotonicity assumption on costs, obtaining an $O(n \log n)$ algorithm that improves the $O(n^2 \log n)$ bound of Cordasco et al. [11]. Building on the resistance distribution function of Como et al. [9], we generalize the LCIP to arbitrary non-decreasing cost functions via a reduction to minimum-weight perfect matching, solvable in $O(n^3)$ by the Hungarian algorithm. Finally, we propose a dynamic programming algorithm for a time-constrained variant of WTSS, a problem not previously addressed in the literature, running in $O(T \cdot n^4)$ time. Building on the combinatorial reformulation of the intervention problem over the symmetric group $\mathcal{S}_n$ introduced by Como et al. [10], we exploit this framework to derive optimal algorithms for complete bipartite and multipartite graphs, graph families not previously addressed in the literature. For low-degree topologies, we generalize the approach of Günnec et al. [12] to arbitrary edge weights and general cost functions, obtaining $O(n \cdot 2^d)$ algorithms for bounded-degree trees and linear-time algorithms for paths and cycles, the latter being novel. Finally, we formalize a Simulated Annealing algorithm over $\mathcal{S}_n$ with a transposition-based neighborhood and derive an explicit mixing time bound. The algorithm is designed to handle the full generality of the LCIP,

arbitrary graph topology, arbitrary edge weights, and arbitrary non-decreasing cost functions. An extensive experimental evaluation across complete graphs, random trees, small-world graphs, and real-world collaboration and social networks shows that the algorithm consistently achieves gaps below 1% relative to exact references when available, and,across all graph families and cost structures tested, significantly outperforms all competing heuristics. These results establish Simulated Annealing over the permutation space as a robust and general-purpose method for the LTM intervention problem.

To address these objectives, the thesis is organized as follows:

- **Chapter 1** introduces the mathematical framework: graph theory background, the Linear Threshold Model and its fundamental properties, the connection to network coordination games, and the formal definitions of TSS, WTSS, and LCIP, together with the main complexity results.
- **Chapter 2** addresses the complete graph, presenting the $O(n \log n)$ WTSS algorithm, the $O(n^3)$ LCIP algorithm with general costs via minimum-weight perfect matching, and the dynamic programming approach for time-constrained WTSS.
- **Chapter 3** presents the combinatorial reformulation over $\mathcal{S}_n$ due to Como et al. [10], and exploits it to derive optimal algorithms for complete bipartite and multipartite graphs, and to discuss the acyclic orientation perspective.
- **Chapter 4** presents exact algorithms for low-degree topologies: bounded-degree trees, paths, cycles, unicyclic graphs, and figure-eight graphs.
- **Chapter 5** develops the local search and simulated annealing algorithms with the full mixing time analysis.
- **Chapter 6** reports the experimental evaluation across synthetic and real-world graph families.

Chapter 1

Problem Statement

In this chapter, we introduce the mathematical framework underlying this thesis. We begin by recalling the basic definitions of graph theory and game theory, then introduce the Linear Threshold Model (LTM) and establish its fundamental properties: monotonicity and finite-time convergence. We subsequently connect the LTM dynamics to the framework of network coordination games, showing that convergence to full adoption admits a natural game-theoretic interpretation. Finally, we formalize the intervention problem, in which a central controller modifies individual thresholds to guarantee complete adoption at minimum cost, called LCIP, and introduce its two main specializations: the Target Set Selection problem (TSS) and the LCIP with linear costs, together with the main complexity results.

1.1 Graph Theory

To study social models, the natural mathematical object to use is the graph. In this representation, nodes stand for units of a population that interact with one another; these interactions are encoded as edges, and weights allow one to further characterise the strength of each tie.

A weighted graph is defined as a triple

$$G = (V, E, W)$$

where

- V is a finite set of *nodes* (or *vertices*);
- $E \subseteq V \times V$ is a set of *edges* (or *links*);
- $W \in \mathbb{R}_+^{V \times V}$ is the *weight matrix*, such that $W_{ij} > 0$ if and only if $(i, j) \in E$.

We denote by $n = |V|$ the *order* of the graph. Edges of the form (i, i) are called *self-loops*. We consider only graphs with at most one edge between any ordered pair of nodes; multiple edges are not allowed.

Moreover, we say that a graph $G = (V, E, W)$ is

- *undirected* when the weight matrix W is symmetric, i.e. $W_{ij} = W_{ji}$ for every $i, j \in V$. In this case, undirected edges are denoted as unordered pairs $\{i, j\}$;
- *unweighted* when $W_{ij} \in \{0,1\}$ for every $i, j \in V$. In this case, $W \in \{0,1\}^{V \times V}$ is called the *adjacency matrix* and is in bijective correspondence with the edge set E;
- *simple* when it is undirected, unweighted, and contains no self-loops.

We now introduce some useful notation. Let $G = (V, E, W)$ be a graph. Then

- the *out-neighborhood* and the *in-neighborhood* of a node $i \in V$ are the sets

$$\mathcal{N}_i = \{j \in V : (i, j) \in E\}, \qquad \mathcal{N}_i^- = \{j \in V : (j, i) \in E\}$$

  for undirected graphs these two sets coincide and are simply denoted $\mathcal{N}_i$;
- a node i with no out-neighbors ($\mathcal{N}_i = \emptyset$) is called a *sink*; a node with no in-neighbors ($\mathcal{N}_i^- = \emptyset$) is called a *source*;
- the *out-degree* and *in-degree* of a node i are the quantities

$$w_i = \sum_{j \in V} W_{ij} \qquad w_i^- = \sum_{j \in V} W_{ji}$$

  for undirected graphs these coincide and are simply written w_i. For unweighted graphs, w_i reduces to the cardinality $|\mathcal{N}_i|$.

Let $G = (V, E, W)$ be a graph. We introduce the following notions of reachability and cycle structure:

- a *walk* is a finite sequence of nodes $\gamma = (i_0, i_1, \ldots, i_l)$ such that $(i_{h-1}, i_h) \in E$ for all $h = 1, \ldots, l$; the integer l is called the *length* of γ;
- a *path* is a walk $\gamma = (i_0, i_1, \ldots, i_l)$ in which all nodes are distinct, i.e. $i_h \neq i_k$ for all $0 \leq h < k \leq l$, except for possibly $i_0 = i_l$;
- a *cycle* is a path $\gamma = (i_0, i_1, ..., i_l)$ with $i_0 = i_l$ and $l \geq 3$.

We say that G is *strongly connected* if for every pair $i, j \in V$ there exists a walk from i to j. For undirected graphs, strong connectivity reduces to the usual notion of *connectivity*. We say that G is *acyclic* (or *cycle-free*) if it contains no cycle.

1.1.1 Special Graphs

We now recall several families of simple graphs that appear throughout this work.

Complete Graph. A graph $G = (V, E, W)$ is a *complete graph*, denoted K_n, when all edges between distinct nodes are present, i.e. $(i, j) \in E$ if and only if $i \neq j$. Every node has degree $w_i = n - 1$.

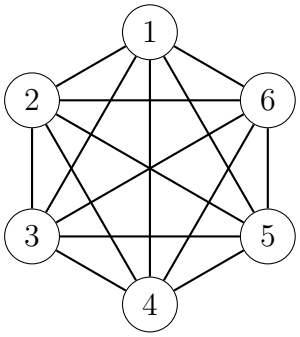


Figure 1.1: Complete graph K_6.

Path Graph. A graph $G = (V, E, W)$ is a *path graph*, denoted L_n, when the nodes can be ordered as $1, 2, \ldots, n$ such that $(i, j) \in E$ if and only if $|i - j| = 1$. The two nodes of degree one are called the *endpoints* of the path.

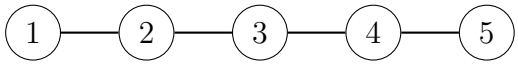


Figure 1.2: Path graph L_5.

Ring Graph. A graph $G = (V, E, W)$ is a *ring graph* (or *cycle graph*), denoted C_n, when the nodes can be ordered as $1, 2, \ldots, n$ such that $(i, j) \in E$ if and only if $|i - j| = 1$, together with the additional edge $\{1, n\}$. In other words, C_n is obtained from the path graph L_n by connecting the two endpoints.

Tree Graph. A graph $G = (V, E, W)$ is a *tree*, denoted with T_n, when it is connected and acyclic. Nodes of degree one are called *leaves*. A *rooted tree* is a tree with a designated node $r \in V$ called the *root*, which induces a natural parent-child ordering on the remaining nodes.

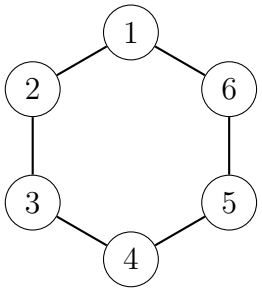


Figure 1.3: Cycle graph C_6.

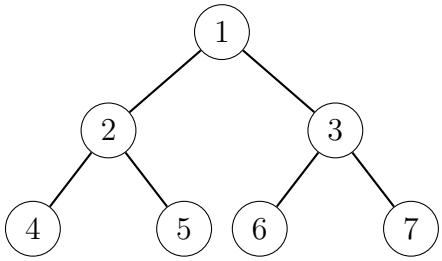


Figure 1.4: A rooted tree T_7 with root node 1.

Bipartite Graph. A graph $G = (V, E, W)$ is *bipartite* when there exists a partition $V = V_1 \cup V_2$, with $V_1 \cap V_2 = \emptyset$, such that $(i, j) \in E$ implies $i \in V_1$ and $j \in V_2$, or $i \in V_2$ and $j \in V_1$. In other words, all edges run between the two parts and no edge is internal to either V_1 or V_2. A bipartite graph is *complete bipartite*, denoted $K_{n,m}$, when every node in V_1 is connected to every node in V_2, where $n = |V_1|$ and $m = |V_2|$. Note that $K_{n,m}$ is a special case of the complete multipartite graph $K_{2\times m}$ when $n = m$.

Complete Multipartite Graph. A graph $G = (V, E, W)$ is *complete multipartite* when there exists a partition $V = V_1 \cup V_2 \cup \cdots \cup V_k$, with $k \geq 2$ and $V_i \cap V_j = \emptyset$ for $i \neq j$, such that $(i, j) \in E$ if and only if i and j belong to *different* parts of the partition. That is, every node in one part is connected to every node in every other part, but there are no edges within any part. The parts $V_1, \ldots, V_k$ are called the *classes* of the graph. When each class has the same cardinality m, the graph is called *balanced* and is denoted $K_{k\times m}$. Note that the complete graph K_n is the special case $k = n$, $|V_i| = 1$ for all i.

1.2 Game Theory

Another important mathematical framework for modeling social phenomena is game theory, which describes strategic interactions among multiple rational agents,

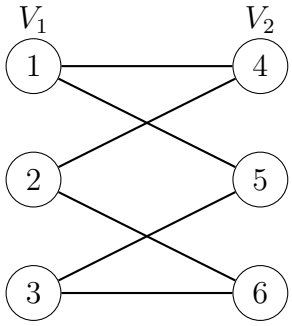


Figure 1.5: A bipartite graph with $V_1 = \{1,2,3\}$ and $V_2 = \{4,5,6\}$.

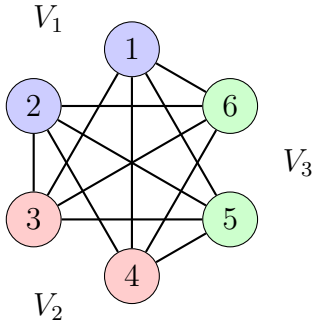


Figure 1.6: Complete multipartite graph $K_{3\times 2}$.

each seeking to maximize their own payoff.

We now formalize these concepts for games in strategic form.

Definition 1 (Strategic Form Game)**.** *A game in strategic form (or normal form) is a triple* $(V, \{\mathcal{A}_i\}, \{u_i\})$ *consisting of:*

(i) A finite set of players $V = \{1, 2, \ldots, n\}$*.*

(ii) For each player $i \in V$*, a set of actions* $\mathcal{A}_i$*. The action set* $\mathcal{A}_i$ *may be finite or continuous (e.g., an interval of real numbers).*

(iii) For each player $i \in V$*, a utility function (also called payoff or reward function)*

$$u_i : \mathcal{X} \to \mathbb{R}$$

where $\mathcal{X} = \prod_{j \in V} \mathcal{A}_j$ *is the* configuration space *(or action profile space).*

An element $x \in \mathcal{X}$ is called an *action profile* or *configuration*, representing a complete assignment of actions to all players. The utility function $u_i(x)$ specifies the payoff that player i receives when the action profile x is played. We write

$$x_{-i} = (x_1, x_2, ..., x_{i-1}, x_{i+1}, ..., x_{n-1}, x_n)$$

to indicate the vector x without the i-th entry, then we can express player i's utility as

$$u_i(x_i, x_{-i}) = u_i(x)$$

making explicit the dependence on both player i's own action and the actions of others.

Each player i is modeled as a rational agent who chooses an action $x_i \in \mathcal{A}_i$ to maximize their utility $u_i(x_i, x_{-i})$, given the actions x_{-i} of the other players. To formalize this, we introduce the *best response function*:

Definition 2 (Best Response)**.** *The best response of player i to an action profile x_{-i} of the other players is the set*

$$B_i(x_{-i}) = \arg\max_{x_i \in \mathcal{A}_i} u_i(x_i, x_{-i})$$

In general, $B_i(x_{-i})$ may be a set, as multiple actions can simultaneously achieve the maximum utility.

A central question in game theory concerns the identification of stable strategy profiles, known as Nash equilibria, in which no player has a unilateral incentive to deviate from their chosen action. Formally:

Definition 3 (Nash Equilibrium)**.** *Let $(V, \{\mathcal{A}_i\}, \{u_i\})$ be a game in strategic form. An action profile $x^* \in \mathcal{X}$ is a* Nash equilibrium *(NE) if*

$$x_i^* \in B_i(x_{-i}^*) \quad \textit{for all } i \in V$$

That is, each player's action is a best response to the actions of all other players.

A stronger notion of stability is given by strict Nash equilibria, where each player has a unique best response:

Definition 4 (Strict Nash Equilibrium)**.** *A Nash equilibrium $x^* \in \mathcal{X}$ is* strict *if for every player $i \in V$,*

$$u_i(x_i^*, x_{-i}^*) > u_i(x_i, x_{-i}^*) \quad \textit{for all } x_i \in \mathcal{A}_i \setminus \{x_i^*\}$$

Equivalently, $B_i(x_{-i}^*) = \{x_i^*\}$ for all $i \in V$.

In a strict Nash equilibrium, no player is indifferent between their equilibrium action and any alternative.

1.2.1 Network Games

A natural and important specialization of the general game-theoretic framework arises when the strategic interactions among players are structured by an underlying graph. In such settings, each player's utility depends not on the actions of all other players, but only on those of their immediate neighbors in the network.

Definition 5 (Network Game)**.** *A network game on the graph $G = (V, E, W)$ is a game in strategic form $(V, \{\mathcal{A}_i\}, \{u_i\})$ such that the utility functions satisfy the following locality property: for any player $i \in V$ and any two configurations $x, y \in \mathcal{X}$ such that $x_j = y_j$ for every $j \in \mathcal{N}_i \cup \{i\}$, it holds that*

$$u_i(x) = u_i(y)$$

This locality property reflects a fundamental feature of many real-world systems: agents are not globally aware of the entire population, but rather interact directly only with a limited set of peers. Network games thus provide a natural model for social and economic phenomena, including public goods game, quadratic games, and majority/minority dynamics.

A particularly tractable and widely studied subclass of network games arises when interactions are *pairwise*, meaning that the utility of each player can be decomposed as a sum of contributions from individual interactions with each neighbor. Formally, the utility of player i takes the form

$$u_i(x) = \sum_{j \in \mathcal{N}_i} \phi_i(x_i, x_j)$$

where $\phi_i : \mathcal{A}_i \times \mathcal{A}_j \to \mathbb{R}$ is a local payoff function capturing the outcome of the interaction between player i and neighbor j. Moreover, when the local payoff function ϕ_i is the same for all players, the game is said to be *symmetric*, a condition that arises naturally in many applications.

1.2.2 Game Dynamics

One main aim of game theory is understanding how a system evolves over time under rational choice and the possibility of strategy revision. To this end, several dynamics have been proposed in the literature to describe how players adaptively update their actions in response to the current state of the system. Among these, we focus on *best-response dynamics*, in which at each time step a player updates their action by selecting one that maximizes their utility given the current actions of the others.

Definition 6. *Let $(V, \{\mathcal{A}_i\}, \{u_i\})$ be a game in strategic form. The* asynchronous best-response dynamics *is a discrete-time Markov chain $(X_t)_{t \in \mathbb{N}}$ with state space*

$\mathcal{X}$ defined as follows: at each time step t, a player $i \in V$ is selected uniformly at random and updates their action by choosing uniformly at random from their current best-response set $B_i((X_t)_{-i})$, while all other players maintain their current actions. Formally, for any $x, y \in \mathcal{X}$,

$$\mathbb{P}(X_{t+1} = y \mid X_t = x) = \begin{cases} \frac{1}{|V|} \cdot \frac{1}{|B_i(x_{-i})|} & \text{if } y_i \in B_i(x_{-i}) \text{ and } y_{-i} = x_{-i} \\ 0 & \text{otherwise} \end{cases}$$

In the asynchronous dynamics, only one player updates at each time step, reflecting a sequential or decentralized decision-making process. An alternative model is the synchronous dynamics, where all players update simultaneously:

Definition 7. *The* synchronous best-response dynamics *is a discrete-time Markov chain $(X_t)_{t\in\mathbb{N}}$ where at each time step t, all players simultaneously update their actions by choosing uniformly at random from their respective best-response sets $B_i((X_t)_{-i})$. Formally, for any $x, y \in \mathcal{X}$,*

$$\mathbb{P}(X_{t+1} = y \mid X_t = x) = \begin{cases} \prod_{i\in V} \frac{1}{|B_i(x_{-i})|} & \text{if } y_i \in B_i(x_{-i}) \text{ for all } i \in V \\ 0 & \text{otherwise} \end{cases}$$

The synchronous dynamics models settings where all players observe the current state and react simultaneously, in contrast to the asynchronous case where updates occur sequentially. Both dynamics are natural models of learning and adaptation in games, and they can exhibit different convergence properties and long-run behavior.

1.3 Linear Threshold Model

The Linear Threshold Model (LTM) describes how behaviors, opinions, or adoptions spread through a social network. The model captures a fundamental social phenomenon: individuals face a binary choice between a default conservative option and an alternative behavior, with their decision driven by two competing forces. The first is an intrinsic resistance, reflecting personal cost-benefit considerations or individual preferences that make adoption inherently more or less costly. The second is social influence: the cumulative pressure exerted by peers who have already adopted. An individual adopts the alternative when the social influence received from their network is sufficient to overcome their personal resistance. This framework naturally models a wide range of real-world phenomena: the diffusion of innovations and new technologies, the adoption of social norms and collective behaviors, the spread of political opinions, or the decision to join a protest or

purchase a product. The key insight of the model is that the interplay between the heterogeneity of individual thresholds and the topology of the interaction network gives rise to rich collective dynamics: sometimes a single early adopter triggers a cascade of adoption throughout the entire network, while in other cases adoption remains confined to a small subset of the population.

1.3.1 Dynamic system

The population is modeled by a weighted graph $G = (V, E, W)$, where nodes represent individuals, edges encode social ties, and the weight $W_{ij} > 0$, called the *influence weight*, quantifies how strongly individual i is influenced by individual j.

Each individual i is characterized by two parameters: a *normalized threshold* $\theta_i \in [0,1]$, representing the minimum fraction of total potential influence required for adoption, and a *resistance* $r_i = \theta_i w_i$, the absolute amount of peer influence needed to activate node i. The state of individual i at time t is $x_i(t) \in \{0,1\}$, where $x_i(t) = 0$ denotes inactivity and $x_i(t) = 1$ denotes adoption.

The dynamics evolve in discrete time: individual i adopts at time $t+1$ if and only if the cumulative influence received from currently active neighbors meets or exceeds its resistance r_i. Formally, introducing the threshold function $f_\alpha : \mathbb{R}_+ \to \{0,1\}$

$$f_\alpha(x) = \begin{cases} 0 & \text{if } x < \alpha \\ 1 & \text{if } x \geq \alpha \end{cases} \tag{1.1}$$

the update rule reads

$$x_i(t+1) = f_{r_i}\left(\sum_{j \in V} W_{ij} x_j(t)\right) = f_{\theta_i w_i}\left(\sum_{j \in V} W_{ij} x_j(t)\right) \tag{1.2}$$

Writing $x(t) \in \{0,1\}^n$, the global dynamics take the compact form

$$x(t+1) = F(x(t)) \tag{1.3}$$

where $F : \{0,1\}^n \to \{0,1\}^n$ is the global update function.

Definition 8 (LTM Instance)**.** *An instance of the Linear Threshold Model is a pair $(G, \{\theta_i\})$, consisting of a weighted graph $G = (V, E, W)$ and a threshold distribution $\{\theta_i\}$. Unless stated otherwise, the dynamics start from the null adoption state $x(0) = \mathbf{0}$.*

A state $\bar{x} \in \{0,1\}^n$ is an *equilibrium point* if $F(\bar{x}) = \bar{x}$. The state $\bar{x} = \mathbf{1}$ is always an equilibrium, since $r_i = \theta_i w_i \leq w_i$ for all $i \in V$; it represents the target configuration of the intervention problems studied in the remainder of this chapter.

We illustrate three characteristic behaviors of the LTM dynamics: convergence to an equilibrium, periodic oscillations, and sensitivity to initial conditions.

Example 1 (Convergence on a Tree)**.** *Consider a simple tree with 5 nodes shown in Figure 1.7. Node 1 has resistance $r_1 = 0$ (activates spontaneously), while nodes 2, 3, 4, and 5 have resistance $r_i = 1$ (requiring influence from at least one active neighbor). All edges have unit weight $W_{ij} = 1$. Starting from the initial configuration where all nodes are inactive, the dynamics evolve as follows:*

- *$t = 1$: Node 1 becomes active (resistance $r_1 = 0$).*
- *$t = 2$: Nodes 2 and 3 receive influence from node 1 and become active.*
- *$t = 3$: Nodes 4 and 5 receive influence from nodes 2 and 3, respectively, and become active.*
- *$t \geq 4$: All nodes remain active; the system has reached the equilibrium $\bar{x} = \mathbf{1}$.*

This example illustrates convergence to complete adoption on a tree structure.

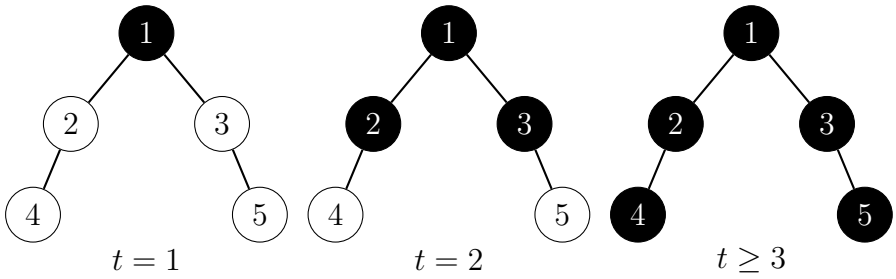


Figure 1.7: Convergence to complete adoption on a tree.

Example 2 (Periodic Behavior on a Cycle)**.** *Consider a cycle graph C_4 with 4 nodes, where each node has resistance $r_i = 2$ (requiring influence from at least two active neighbors) and all edges have unit weight $W_{ij} = 1$. Starting from the initial configuration where nodes 1 and 3 are active (shown in black) and nodes 2 and 4 are inactive (shown in white), the dynamics evolve periodically:*

- *$t = 0$: Nodes 1 and 3 are active.*
- *$t = 1$: Nodes 2 and 4 each receive influence from two active neighbors (nodes 1 and 3) and become active. Nodes 1 and 3 receive influence from only one active neighbor each and become inactive.*
- *$t = 2$: Nodes 1 and 3 each receive influence from two active neighbors (nodes 2 and 4) and become active again. Nodes 2 and 4 become inactive.*
- *$t \geq 3$: The system alternates between states $x(0)$ and $x(1)$ indefinitely.*

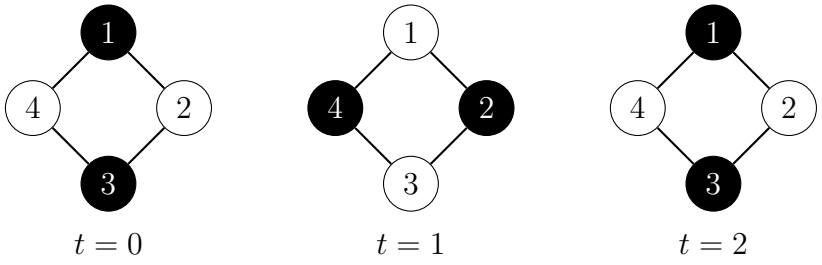


Figure 1.8: Periodic behavior on a cycle graph with period 2.

This example illustrates periodic behavior on a cycle, where the system does not converge to a fixed equilibrium.

Example 3 (Granovetter's Threshold: Sensitivity to Initial Conditions)**.** *Consider a complete graph K_6 with 6 nodes, where each node has degree 5 and all edges have unit weight $W_{ij} = 1$. We compare two scenarios that differ only in the resistance of node 2.*

Scenario A: *Resistances are $r_1 = 0, r_2 = 1, r_3 = 2, r_4 = 3, r_5 = 4, r_6 = 5$. Starting from $x(0) = \mathbf{0}$:*

- *$t = 1$: Node 1 activates spontaneously ($r_1 = 0$).*
- *$t = 2$: Node 2 receives influence 1 and activates ($r_2 = 1$).*
- *$t = 3$: Node 3 receives influence 2 and activates ($r_3 = 2$).*
- *$t = 4$: Node 4 receives influence 3 and activates ($r_4 = 3$).*
- *$t = 5$: Node 5 receives influence 4 and activates ($r_5 = 4$).*
- *$t = 6$: Node 6 receives influence 5 and activates ($r_6 = 5$).*
- *$t \geq 7$: Complete adoption $\bar{x} = \mathbf{1}$.*

Scenario B: *Resistances are $r_1 = 0, r_2 = 2, r_3 = 2, r_4 = 3, r_5 = 4, r_6 = 5$. Starting from $x(0) = \mathbf{0}$:*

- *$t = 1$: Node 1 activates spontaneously ($r_1 = 0$).*
- *$t \geq 2$: Node 2 requires influence 2 but only receives 1, so it remains inactive. Similarly, all other nodes remain inactive.*
- *Result: Only node 1 is active; $\bar{x} = (1, 0, 0, 0, 0, 0)$.*

This example illustrates how a small change in a single individual's resistance (node 2 changing from $r_2 = 1$ to $r_2 = 2$) can dramatically alter the collective behavior, shifting from complete adoption to near-complete rejection.

Scenario A: $r_2 = 1$
Complete adoption

Scenario B: $r_2 = 2$
Only node 1 active

Figure 1.9: Sensitivity to initial conditions on K_6.

1.3.2 Monotonicity and Convergence

The LTM dynamics enjoys a fundamental monotonicity property with respect to the pointwise partial order on $\{0,1\}^n$: for $x, y \in \{0,1\}^n$, we write $x \leq y$ if $x_i \leq y_i$ for all $i \in V$.

Lemma 1 (Monotonicity)**.** *Let $(G, \{\theta_i\})$ be any instance of the LTM dynamics. For any two initial conditions $x(0), y(0) \in \{0,1\}^n$ with $x(0) \leq y(0)$, it holds that $x(t) \leq y(t)$ for all $t \geq 0$.*

Proof. By induction on t. The case $t = 0$ holds by hypothesis. Assume $x(t) \leq y(t)$ for some $t \geq 0$. Then, for each $i \in V$, since $W_{ij} \geq 0$,

$$\sum_{j \in V} W_{ij}\, x_j(t) \leq \sum_{j \in V} W_{ij}\, y_j(t)$$

and since f_{r_i} is nondecreasing, it follows that

$$x_i(t+1) = f_{r_i}\left(\sum_{j \in V} W_{ij}\, x_j(t)\right) \leq f_{r_i}\left(\sum_{j \in V} W_{ij}\, y_j(t)\right) = y_i(t+1)$$

for all $i \in V$, i.e., $x(t+1) \leq y(t+1)$. □

In some alternative formulations of the LTM, adoption is irreversible: once $x_i(t) = 1$, then $x_i(s) = 1$ for all $s \geq t$. Our model allows nodes to revert to the inactive state; however, this distinction does not affect the subsequent analysis. Indeed, by the monotonicity of the dynamics, starting from $x(0) = \mathbf{0}$, if node i becomes active at time t then $x_i(s) = 1$ for all $s \geq t$: both formulations yield equivalent behavior when initialized at $\mathbf{0}$.

Corollary 1. *The LTM dynamics starting from $x(0) = \mathbf{0}$ converges to an equilibrium point in at most n steps.*

Proof. By Lemma 1, the sequence $(x(t))_{t\geq 0}$ is nondecreasing in $\{0,1\}^n$. Since each coordinate can switch from 0 to 1 at most once, a fixed point is reached within at most n steps. □

By the monotonicity property established in Lemma 1, if the dynamics starting from $x(0) = \mathbf{0}$ converges to the all-active configuration $\mathbf{1}$, then any initial condition $x(0) \geq \mathbf{0}$ also converges to $\mathbf{1}$. In other words, if full adoption is reachable from $\mathbf{0}$, it is reachable from any starting point.

Before introducing the intervention problems that constitute the focus of this thesis, we establish an important connection between the LTM dynamics and network coordination games via best-response dynamics, which provides a game-theoretic interpretation of the diffusion process.

1.4 Network Coordination Games

A particularly relevant example of pairwise network game is the *network coordination game*, which generalizes the majority game. Consider a weighted graph $G = (V, E, W)$. A network coordination game is defined by the triple $(V, \mathcal{A}, \{u_i\})$, where V is the set of players, $\mathcal{A} = \{0,1\}$ is the common action space, and the utility function of player i is

$$u_i(x_i, x_{-i}) = \sum_{j \in \mathcal{N}_\rangle} W_{ij}\Big((1 - x_i)(1 - x_j) + x_i x_j\Big) + c_i x_i \tag{1.4}$$

where $c_i \in [-w_i, w_i]$. The first term rewards coordination: player i gains payoff W_{ij} for each neighbor j playing the same action. The parameter c_i captures an intrinsic bias toward action 1; when $c_i = 0$ for all i and the graph is unweighted, the game reduces to the majority game.

To derive the best response, observe that

$$\sum_{j \in \mathcal{N}_i : x_j = 0} W_{ij} + \sum_{j \in \mathcal{N}_i : x_j = 1} W_{ij} = w_i,$$

so player i prefers action 1 over action 0 if and only if

$$\sum_{j \in \mathcal{N}_i} W_{ij} x_j + c_i \geq w_i - \sum_{j \in \mathcal{N}_i} W_{ij} x_j$$

which simplifies to

$$\frac{1}{w_i} \sum_{j \in \mathcal{N}_i} W_{ij} x_j \geq \frac{w_i - c_i}{2w_i}$$

Defining the *coordination threshold* $\theta_i = \frac{w_i - c_i}{2w_i} \in [0,1]$, the best response of player i takes the form

$$B_i(x_{-i}) = \begin{cases} \{0\} & \text{if } \dfrac{1}{w_i} \displaystyle\sum_{j \in \mathcal{N}_i} W_{ij} x_j < \theta_i \\ \{0,1\} & \text{if } \dfrac{1}{w_i} \displaystyle\sum_{j \in \mathcal{N}_i} W_{ij} x_j = \theta_i \\ \{1\} & \text{if } \dfrac{1}{w_i} \displaystyle\sum_{j \in \mathcal{N}_i} W_{ij} x_j > \theta_i \end{cases} \tag{1.5}$$

We now establish the connection between the LTM dynamics and the network coordination game. The key observation is that the LTM update rule (1.2) coincides with the synchronous best-response dynamics under a specific tie-breaking convention: when the best-response set is $\{0,1\}$ (i.e. when the cumulative influence exactly equals the threshold), the LTM selects the maximum action 1. Formally, for each player $i \in V$,

$$x_i(t+1) \;=\; f_{r_i}\!\left(\sum_{j \in V} W_{ij}\, x_j(t)\right) \;=\; \max B_i\big(x_{-i}(t)\big)$$

Proposition 1. *Consider a network coordination game on $G = (V, E, W)$ with coordination thresholds $\{\theta_i\}$. Then:*

(i) Every equilibrium point of the LTM dynamics is a Nash equilibrium. Conversely, every strict Nash equilibrium is an equilibrium point of the LTM dynamics.

(ii) If all best responses are unique, i.e. $|B_i(x_{-i})| = 1$ for all $i \in V$ and all $x \in \{0,1\}^n$, then the sets of Nash equilibria and LTM equilibrium points coincide.

(iii) The LTM dynamics starting from $x(0) = \mathbf{0}$ converges to $\mathbf{1}$ if and only if the asynchronous best-response dynamics reaches $\mathbf{1}$ with probability 1.

Proof. (i) Let $\bar{x}$ be an LTM equilibrium point, so $F(\bar{x}) = \bar{x}$, i.e. $f_{r_i}\big(\sum_j W_{ij}\, \bar{x}_j\big) = \bar{x}_i$ for all $i \in V$. Since $\bar{x}_i = \max B_i(\bar{x}_{-i})$, we have $\bar{x}_i \in B_i(\bar{x}_{-i})$ for all i, hence $\bar{x}$ is a Nash equilibrium. Conversely, let x^* be a strict Nash equilibrium, so $B_i(x^*_{-i}) = \{x^*_i\}$ for all $i \in V$. From (1.5), this implies $f_{r_i}\big(\sum_j W_{ij}\, x^*_j\big) = x^*_i$ for all i, which is $F(x^*) = x^*$.

(ii) When $|B_i(x_{-i})| = 1$ for all i and x, the max tie-breaking is never invoked, so $\max B_i(x_{-i}) = B_i(x_{-i})$ for all i, and the equivalence $F(x^*) = x^* \iff x^*_i \in B_i(x^*_{-i})\ \forall\, i$ follows immediately.

(iii) $(\Rightarrow)$: Suppose the LTM dynamics converges to $\mathbf{1}$ from $x(0) = \mathbf{0}$. By Lemma 1, the same convergence holds from any initial condition $x(0) \geq \mathbf{0}$, so from

every state there exists a monotone path to **1** under the LTM dynamics. Every LTM transition is a valid best-response move. Therefore, the LTM trajectory can be unrolled into a sequence of asynchronous best-response moves, showing that **1** is reachable from every state. By the theory of absorbing states of Markov chains, the asynchronous dynamics reaches **1** with probability 1.

($\Leftarrow$): Suppose the asynchronous best-response dynamics reaches **1** with probability 1 from **0**. Then there exists a path

$$\mathbf{0} = x^{(0)}, x^{(1)}, \ldots, x^{(k)} = \mathbf{1}$$

where each step activates exactly one node: $x^{(t+1)} = x^{(t)} + e_i$ for some $i \in V$ with $\sum_{j \in \mathcal{N}_i} W_{ij}\, x_j^{(t)} \geq r_i$. The path is monotone by construction. Condensing consecutive activations into simultaneous time steps yields a trajectory consistent with the LTM update rule. By Lemma 1, the LTM trajectory from **0** dominates $x^{(t)}$ componentwise at every step, so the LTM converges to **1**. □

This result provides a game-theoretic interpretation of the LTM: the dynamics is the synchronous best-response dynamics with maximum tie-breaking, its equilibria are Nash equilibria, and convergence to **1** is equivalent to the asynchronous dynamics reaching the socially optimal equilibrium with probability 1. For a more extensive discussion, see [9].

1.5 Intervention Problem

Throughout this thesis, we study intervention problems in the context of the Linear Threshold Model. The term *intervention problem* is an umbrella term encompassing a broad class of optimization problems in which a central controller seeks to achieve a desired outcome by strategically modifying certain characteristics of the system, typically subject to budget constraints or with the goal of minimizing the cost required to reach a target configuration.

In the context of network games, possible interventions include altering the network structure by adding or removing edges, modifying edge weights, or adjusting the utility functions of individual players. In the context of the Linear Threshold Model, the natural parameters available for intervention are the graph topology, the edge weights, and the threshold distribution. The choice of which parameters to modify depends on the application: in commercial settings, for instance, a firm may lower individual adoption thresholds by offering coupons or free samples, effectively reducing the resistance of targeted nodes. In this work, we focus exclusively on threshold interventions.

1.5.1 Least Cost Intervention Problem

Given an LTM dynamics $(G, \{\theta_i\})$, we want to ensure that the system converges to complete adoption $\mathbf{1}$ starting from the all-inactive configuration $x(0) = \mathbf{0}$. By Corollary 1, the dynamics always converges to some equilibrium in at most n steps; however, this equilibrium need not be $\mathbf{1}$. To promote full adoption, we are allowed to modify the resistance of each node $i \in V$ via an *intervention vector* $h \in \mathbb{R}^n_+$, where $h_i \geq 0$ denotes the reduction applied to node i. The modified resistance becomes $r'_i = r_i - h_i = \theta_i w_i - h_i$, and the dynamics is replaced by

$$x_i(t+1) = f_{r_i - h_i}\left(\sum_{j \in V} W_{ij}\, x_j(t)\right) \tag{1.6}$$

represented compactly as

$$x(t+1) = F_h(x(t)) \tag{1.7}$$

Definition 9 (Successful Intervention)**.** *An intervention $h \in \mathbb{R}^n_+$ is successful if the dynamics* (1.6) *starting from $x(0) = \mathbf{0}$ converges to $\mathbf{1}$. The set of all successful interventions is*

$$\mathcal{H} = \left\{ h \in \mathbb{R}^n_+ \;:\; \lim_{t \to \infty} x(t) = \mathbf{1}, \quad x(0) = \mathbf{0} \right\} \tag{1.8}$$

Since each intervention carries a cost, we assign to each node $i \in V$ a non-decreasing cost function $C_i : \mathbb{R}_+ \to \mathbb{R}_+$, where $C_i(h_i)$ represents the cost of reducing node i's resistance by h_i. An instance of the problem is denoted by a triple $(G, \{\theta_i\}, \{C_i\})$. We refer to this as the *Least Cost Influence Problem* (LCIP), or equivalently the *LTM intervention problem*. The optimization problem reads

$$\min_{h \in \mathcal{H}} \sum_{i \in V} C_i(h_i) \tag{1.9}$$

Although $\mathcal{H}$ may have infinite cardinality, the monotonicity of the cost functions allows us to restrict attention to a finite subset. For each node $i \in V$, define

$$\mathcal{H}_i := \left\{ h_i \in \mathbb{R}_+ \;:\; h_i = w_i\theta_i - \sum_{j \in \mathcal{N}_i} W_{ij} y_j \text{ for some } y \in \{0,1\}^n \right\} \tag{1.10}$$

Proposition 2. *There exists an optimal solution $h^* \in \prod_{i \in V} \mathcal{H}_i \subseteq \mathcal{H}$.*

Proof. Under any successful intervention $h \in \mathcal{H}$, each node i activates at some time t when a configuration $y \in \{0,1\}^n$ of neighbors is already active, so that

$$\sum_{j \in N_i} W_{ij} y_j \geq w_i\theta_i - h_i$$

If $h_i > w_i\theta_i - \sum_{j\in N_i} W_{ij}y_j$, setting $h_i' = w_i\theta_i - \sum_{j\in N_i} W_{ij}y_j$ achieves the same activation at lower cost, since C_i is non-decreasing. Hence any intervention strictly exceeding the minimum necessary can be reduced without compromising success, and an optimal solution lies in $\prod_{i\in V} \mathcal{H}_i$. Since each $\mathcal{H}_i$ is finite (containing at most 2^{n^2} elements), the problem (1.9) restricted to this set possesses an optimal solution. □

The choice of cost functions $\{C_i\}$ is an important modeling decision that depends on the application scenario and determines the computational complexity of the problem. Two main formulations have been studied in the literature. The first is the *fixed cost* formulation, $C_i(h_i) = c_i\mathbb{1}_{\{h_i>0\}}$, introduced by Chen [6], known as the Target Set Selection problem (TSS). The second is the *linear cost* formulation, $C_i(h_i) = c_i \cdot h_i$. These two cases are studied in Sections 1.5.2 and 1.5.3 respectively.

1.5.2 Target Set Selection

In the Target Set Selection problem, the cost functions take the form $C_i(h_i) = c_i\mathbb{1}_{\{h_i>0\}}$: any positive intervention on node i incurs a fixed cost c_i, regardless of the intervention magnitude. Under this structure, partially lowering a resistance without fully activating the node is always suboptimal. The intervention problem thus reduces to identifying the minimum-cost subset of nodes that, when fully activated, triggers complete adoption through propagation.

Remark 1 (Integer-valued resistances)**.** *We restrict our analysis to simple graphs and assume without loss of generality that resistances take integer values $r_i \in \{0, 1, \ldots, d_i\}$, or equivalently, thresholds satisfy $\theta_i \in \{0, 1/d_i, \ldots, (d_i - 1)/d_i, 1\}$, where $d_i = \deg(i)$. Indeed, since partially lowering a resistance without activating the node is suboptimal, a node is either directly targeted or requires at least $\lceil r_i \rceil$ active neighbors to activate. There is therefore no loss of generality in assuming $r_i \in \mathbb{Z}$.*

The set of nodes receiving positive interventions is called the *target set*, defined as

$$\mathcal{W} = \{i \in V : h_i > 0\} \tag{1.11}$$

A target set $\mathcal{W}$ is *successful* if the LTM dynamics starting from $x(0) = \mathbf{0}$ with all nodes in $\mathcal{W}$ fully activated converges to $\mathbf{1}$. The intervention problem (1.9) specializes to

$$\min_{\mathcal{W}\subseteq V} \left\{ \sum_{i\in\mathcal{W}} c_i \ : \ \mathcal{W} \text{ is a successful target set} \right\} \tag{1.12}$$

An instance $(G, \{\theta_i\}, \{c_i\mathbb{1}_{h_i>0}\})$ is referred to as *Target Set Selection* (TSS) when costs are homogeneous, i.e., $c_i = c$ for all $i \in V$; without loss of generality we set

$c = 1$. When costs are heterogeneous, the problem is referred to as *Weighted Target Set Selection* (WTSS).

The TSS and WTSS problems are computationally challenging in general.

Theorem 1 ([9])**.** *The Target Set Selection problem is NP-hard, even for homogeneous costs and simple graphs.*

Nevertheless, polynomial-time algorithms exist for specific graph topologies. Raghavan et al. [13] developed linear-time algorithms for both TSS and WTSS on trees and rings. For complete graphs, Cordasco et al. [16] proposed a polynomial-time greedy algorithm for WTSS under a monotonicity assumption on costs, running in $O(n^2 \log n)$ time. Como et al. [9] introduced an elegant approach for the homogeneous case on complete graphs, solvable in $O(n)$ time. In this thesis, we contribute state-of-the-art results for the complete graph setting: we develop an $O(n \log n)$ algorithm for WTSS that requires no additional assumptions on the cost structure, and a polynomial-time algorithm for a time-constrained variant of the problem. These results are presented in Chapter 2.

1.5.3 LCIP with Linear Costs

In the Least Cost Influence Problem (LCIP) with linear costs, we consider cost functions of the form $C_i(h_i) = c_i \cdot h_i$, where $c_i > 0$ is a per-unit cost associated with node i. Unlike TSS, where the controller either fully targets a node or leaves it untouched, the LCIP with linear costs allows for *partial incentives*: the resistance of a node can be reduced by any amount $h_i \geq 0$, with cost proportional to the reduction. This models settings such as coupon-based marketing, where a firm may offer a partial discount to lower an individual's adoption threshold without fully subsidizing the product. The optimization problem (1.9) specializes to

$$\min_{h \in \mathcal{H}} \sum_{i \in V} c_i h_i \tag{1.13}$$

When costs are homogeneous, $c_i = c$ for all $i \in V$, we set $c = 1$ without loss of generality. An equivalent formulation, due to Gunnec et al. [12], introduces binary variables $y_{it} \in \{0,1\}$ denoting whether node i is active at time period t, and continuous variables $p_i \geq 0$ representing the incentive paid to node i. The problem

can be written as

$$\begin{aligned} \min \quad & \sum_{i \in V} p_i & & \\ \text{s.t.} \quad & y_{i0} = 0 & & \forall i \in V \\ & p_i + \sum_{j \in N_i} W_{ij}\, y_{j,t-1} \geq r_i\, y_{it} & & \forall i \in V,\ t = 1, \ldots, n \\ & \sum_{i \in V} y_{in} = |V| & & \\ & y_{it} \in \{0,1\},\ p_i \geq 0 & & \forall i \in V,\ t = 1, \ldots, n \end{aligned} \tag{1.14}$$

The LCIP is computationally challenging in general.

Theorem 2 ([12])**.** *The LCIP is NP-hard. Moreover, it cannot be approximated within a factor of* $O\left(2^{\log^{1-\varepsilon}|V|}\right)$ *for any fixed* $\varepsilon > 0$.

The NP-hardness holds even on bipartite graphs and on simple graphs. The inapproximability result follows by reduction from the TSS problem [6].

Nevertheless, polynomial-time algorithms exist for specific graph topologies. Gunnec et al. [12] developed two exact algorithms for the LCIP with linear costs on trees under equal influence: a greedy algorithm running in $O(n \log n)$ time, and a dynamic programming algorithm achieving $O(n)$ time. For complete graphs, Como et al. [10] provided a polynomial-time algorithm under the equal influence assumption. In this thesis, we contribute to this line of work by developing algorithms for the LCIP on complete graphs and related structures under general cost functions, via the combinatorial reformulation presented in Chapter 3 and expanding results on paths and cycles with linear-time algorithms in Chapter 4.

Chapter 2

Algorithms for Complete Graph

In this chapter, we address the LTM intervention problem on complete graphs. The complete graph topology is of particular interest as a natural model of fully connected social networks, where every individual is directly influenced by all others. Beyond its theoretical relevance, it serves as a benchmark in the experimental evaluation of Chapter 6, where the exact solutions derived here are used as reference optima to assess the performance of the heuristic methods developed in Chapter 5.

The literature offers several results for this topology, all restricted to simple graphs with homogeneous edge weights, a setting we adopt throughout this chapter as well. For the targeting problem, Como et al. [9] proposed an elegant $O(n)$ algorithm for the TSS based on a resistance distribution function. Cordasco et al. [11] addressed a restricted variant of the WTSS, requiring the activation costs to increase monotonically with resistance, and solved it in $O(n^2 \log n)$ time. For the LCIP, Como et al. [10] solved the special case of identity cost functions $C_i(h_i) = h_i$ in $O(n \log n)$ time. To the best of our knowledge, no result is available in the literature for general non-decreasing cost functions.

Our contributions are threefold. First, building upon the resistance distribution function of Como et al. [9], we solve the general WTSS in $O(n \log n)$ time via a greedy algorithm, removing the monotonicity assumption of Cordasco et al. [11] and improving their complexity from $O(n^2 \log n)$ to $O(n \log n)$. Second, we strictly generalize the LCIP result of Como et al. [10] to arbitrary non-decreasing cost functions $\{C_i\}$, with no assumptions on the threshold distribution or the cost structure; this is achieved via a reduction to minimum-weight perfect matching on a complete bipartite graph, solvable in $O(n^3)$ by the Hungarian algorithm. Third, we propose a dynamic programming algorithm for the time-constrained WTSS, a variant, running in $O(T \cdot n^4)$ time.

Throughout this chapter, the graph is assumed to be the simple complete graph K_n on n nodes.

2.1 Targeting

As discussed in the first chapter, the target set selection problem is NP-hard even in the homogeneous cost case [9]; nevertheless, exact polynomial-time algorithms exist for complete graphs. We present here the work of Como et al. [9], which introduces a resistance distribution function to characterize the limit behavior of the LTM dynamics on K_n and exploits it to solve any instance of the TSS problem $(K_n, \{\theta_i\}, \{\mathbb{1}_{h_i>0}\})$. These results serve as the foundation for the sequent sections.

Before introducing the notation, we recall that in the targeting setting, resistances can be assumed integer-valued without loss of generality, so in a complete graph K_n $r_i = (n-1)\theta_i \in \{0, \dots, n-1\}$ for all $i \in V$.

Definition 10. *Given an instance $(K_n, \{\theta_i\})$ of the LTM dynamics on a complete graph, the* cumulative resistance distribution function $\Phi : \{1, \dots, n\} \to \{0, 1, \dots, n\}$ *is defined by*

$$\Phi(k) = |\{i \in V : r_i < k\}| \tag{2.1}$$

that is, $\Phi(k)$ counts the number of nodes whose resistance is strictly less than k.

Example 4. *Consider an instance $(K_5, \{\theta_i\})$ with $n = 5$ nodes and resistances $r_1 = 0,\ r_2 = 1,\ r_3 = 1,\ r_4 = 3,\ r_5 = 3$. The cumulative resistance distribution function $\Phi : \{1, \dots, 5\} \to \{0, 1, \dots, 5\}$ is computed as follows:*

$$\begin{aligned}
\Phi(1) &= |\{i \in V : r_i < 1\}| = |\{1\}| = 1 \\
\Phi(2) &= |\{i \in V : r_i < 2\}| = |\{1, 2, 3\}| = 3 \\
\Phi(3) &= |\{i \in V : r_i < 3\}| = |\{1, 2, 3\}| = 3 \\
\Phi(4) &= |\{i \in V : r_i < 4\}| = |\{1, 2, 3, 4, 5\}| = 5 \\
\Phi(5) &= |\{i \in V : r_i < 5\}| = |\{1, 2, 3, 4, 5\}| = 5
\end{aligned}$$

In the following lemma, we use the resistance distribution function to characterize the limit behavior of the LTM dynamics.

Lemma 2. *Let $(K_n, \{\theta_i\})$ be an instance of the LTM dynamics, starting from $x(0) = \mathbf{0}$. The state of complete adoption $x = \mathbf{1}$ is reached if and only if*

$$\Phi(k) \geq k \qquad \forall k \in \{1, \dots, n\} \tag{2.2}$$

Proof. We prove both directions separately.

(⇒) Necessity: Suppose condition (2.2) fails, i.e., there exists $k^* \in \{1, \ldots, n\}$ with $\Phi(k^*) < k^*$, and let i^* be a node with $r_{i^*} = k^*$. Since every node receives the same amount of influence from the network, nodes activate in non-decreasing order of resistance, so at any time t the influence received by i^* is at most

$$\Phi(k^*) < k^* = r_{i^*}$$

which is insufficient for activation. Hence complete adoption cannot be reached.

(⇐) Sufficiency: Assume condition (2.2) holds. We prove by strong induction on k that all nodes with resistance at most k eventually become active.

Base case ($k = 0$): Nodes with $r_i = 0$ satisfy the activation condition regardless of the state of their neighbors, so they activate at time $t = 1$.

Inductive step: Assume all nodes with resistance at most k are active at some time t. By the induction hypothesis and condition (2.2), at least $\Phi(k+1) \geq k+1$ nodes are active at time t. Since the graph is complete, every node with resistance $r_i = k+1$ receives influence at least $k+1 = r_i$ and therefore activates at time $t+1$. Previously active nodes remain active by monotonicity.

By induction, all nodes eventually become active, completing the proof. □

Corollary 2. *Let $(K_n, \{\theta_i\})$ be an instance of the LTM dynamics, deciding whether it will reach complete adoption can be done in $O(n)$ time.*

Proof. Simulating the dynamics directly would require $O(n^2)$ time. Instead, we can compute the resistance distribution function Φ in $O(n)$ time as follows: through a linear scan, count the number of nodes at each resistance level $r_i \in \{0, \ldots, n-1\}$, then compute Φ via a cumulative sum. Finally, verify condition (2.2) in linear time by checking $\Phi(k) \geq k$ for all $k \in \{1, \ldots, n\}$. □

We now analyze how a targeting intervention affects the dynamics. A target set $\mathcal{W} \subseteq V$ induces a modified LTM instance $(K_n, \{\theta_i'\})$, where targeted nodes have their resistance set to zero, i.e., $r_i' = 0$ for $i \in \mathcal{W}$, while all other nodes retain their original resistances.

Definition 11. *Given an instance $(K_n, \{\theta_i\})$ of the LTM dynamics and a target set $\mathcal{W} \subseteq V$, we denote by $\Phi_{\mathcal{W}} : \{1, \ldots, n\} \to \{0, 1, \ldots, n\}$ the resistance distribution function of the modified instance $(K_n, \{\theta_i'\})$ induced by $\mathcal{W}$.*

To derive an explicit expression for $\Phi_{\mathcal{W}}$, consider first the case where a single node l with resistance $r_l = g$ is targeted. Since $r_l' = 0$, node l transitions from contributing to $\Phi(k)$ only for $k > g$ to contributing for all $k \in \{1, \ldots, n\}$, yielding

$$\Phi_{\{l\}}(k) = \begin{cases} \Phi(k) + 1 & \text{for all } k \in \{1, \ldots, g\} \\ \Phi(k) & \text{for all } k > g \end{cases} \tag{2.3}$$

A key observation is that each targeted node modifies Φ independently of the others, so for an arbitrary target set $\mathcal{W} \subseteq V$ the modified distribution function takes the form

$$\Phi_{\mathcal{W}}(k) = |\{i \in V : r_i < k\}| + |\{i \in \mathcal{W} : r_i \geq k\}| \tag{2.4}$$

that is, $\Phi_{\mathcal{W}}(k)$ counts all nodes with resistance strictly less than k, plus the nodes in $\mathcal{W}$ whose resistance is at least k.

Example 5. *Consider an instance $(K_5, \{\theta_i\})$ with resistances $r_1 = 0,\ r_2 = 1,\ r_3 = 2,\ r_4 = 4,\ r_5 = 4$. The resistance distribution function takes the values*

$$\Phi(1) = 1, \quad \Phi(2) = 2, \quad \Phi(3) = 3, \quad \Phi(4) = 3, \quad \Phi(5) = 5$$

Since $\Phi(4) = 3 < 4$, condition (2.2) fails and the dynamics does not reach complete adoption from $x(0) = \mathbf{0}$. Targeting node 4 (with $r_4 = 4$) and setting $r_4' = 0$, by equation (2.4) the modified distribution becomes

$$\Phi_{\{4\}}(1) = 2, \quad \Phi_{\{4\}}(2) = 3, \quad \Phi_{\{4\}}(3) = 4, \quad \Phi_{\{4\}}(4) = 4, \quad \Phi_{\{4\}}(5) = 5$$

so that $\Phi_{\{4\}}(k) \geq k$ for all $k \in \{1, \ldots, 5\}$ and complete adoption is guaranteed by Lemma 2.

Lemma 3. *Let $(K_n, \{\theta_i\}, \{\mathbb{1}_{h_i>0}\})$ be an instance of the TSS problem on a complete graph with resistance distribution function Φ. The value of the optimal solution, i.e., the minimum cardinality of a successful target set, equals*

$$M = \max_{k \in \{1,\ldots,n\}} [k - \Phi(k)]^+ \tag{2.5}$$

Proof. We prove both directions separately.

($\leq$) Lower Bound: Let $\mathcal{W} \subseteq V$ be a successful target set. By Lemma 2, full adoption requires

$$\Phi_{\mathcal{W}}(k) \geq k \quad \text{for all } k \in \{1, \ldots, n\}$$

For any k where $\Phi(k) < k$, substituting equation (2.4) gives

$$\Phi(k) + |\{i \in \mathcal{W} : r_i \geq k\}| \geq k \implies |\{i \in \mathcal{W} : r_i \geq k\}| \geq k - \Phi(k)$$

Since $\mathcal{W}$ must satisfy this inequality for all $k \in \{1, \ldots, n\}$, we obtain

$$|\mathcal{W}| \geq \max_{k \in \{1,\ldots,n\}} [k - \Phi(k)]^+ = M$$

($\geq$) Upper Bound: Let $k^* \in \arg\max_k [k - \Phi(k)]^+$, so that $M = k^* - \Phi(k^*)$, and define $\mathcal{W} = \{i_1, \ldots, i_M\}$ as the M nodes with the largest resistances. We verify

that $\mathcal{W}$ satisfies condition (2.2). Since $\mathcal{W}$ consists of the M highest-resistance nodes, we have

$$|\{i \in \mathcal{W} : r_i \geq k\}| = \min(M,\, n - \Phi(k))$$

where $n-\Phi(k)$ denotes the number of nodes with resistance at least k. If $n-\Phi(k) \geq M$, then

$$\Phi_{\mathcal{W}}(k) = \Phi(k) + M \geq \Phi(k) + (k - \Phi(k)) = k$$

where the inequality follows from $M \geq k - \Phi(k)$ by definition of M. If instead $n - \Phi(k) < M$, then all nodes with resistance at least k belong to $\mathcal{W}$, and

$$\Phi_{\mathcal{W}}(k) = \Phi(k) + (n - \Phi(k)) = n \geq k$$

In both cases $\Phi_{\mathcal{W}}(k) \geq k$, so by Lemma 2, $\mathcal{W}$ achieves full adoption. Since $|\mathcal{W}| = M$ and M nodes are necessary, the optimal solution value is M. □

The previous lemma allows to compute both the optimal value and an optimal solution for the homogeneous target set selection problem in $O(n)$ time, and provides the structural foundation to tackle the more general weighted target set selection problem on complete graphs.

2.2 Weighted Targeting

The weighted target set selection problem on the complete graph K_n was studied by Cordasco et al. [16], who proposed a polynomial-time greedy algorithm. However, their approach has two notable limitations. First, it requires an additional assumption on the cost structure: in any instance $(K_n, \{\theta_i\}, \{c_i \mathbb{1}_{h_i>0}\})$, costs must increase monotonically with resistance, i.e., $c_i \leq c_j$ whenever $\theta_i \leq \theta_j$. Second, the algorithm runs in $O(|E| \log |V|)$ time, which on K_n amounts to $O(n^2 \log n)$. Here is proposed an $O(n \log n)$ greedy algorithm that solves the WTSS problem on K_n without any assumption on the cost structure.

Lemma 4. *Let $(K_n, \{\theta_i\}, \{c_i \mathbb{1}_{h_i>0}\})$ be an instance of the WTSS problem, and let k^* be the largest value in $\{1, \ldots, n\}$ such that $\Phi(k^*) < k^*$. Then there exists an optimal solution that includes a minimum-cost node among those with resistance at least k^*, i.e., a node $i^* \in \arg\min\{c_i : r_i \geq k^*\}$.*

Proof. Consider an optimal solution $\mathcal{W}^* \subseteq V$ and suppose that $i^* \notin \mathcal{W}^*$. By Lemma 3, any successful target set must satisfy $\Phi_{\mathcal{W}^*}(k) \geq k$ for all $k \in \{1, \ldots, n\}$. In particular, for $k = k^*$, since $\Phi(k^*) < k^*$ by hypothesis,

$$|\{i \in \mathcal{W}^* : r_i \geq k^*\}| \geq k^* - \Phi(k^*) \geq 1$$

so $\mathcal{W}^*$ must contain at least one node ℓ with $r_\ell \geq k^*$. Since $i^* \notin \mathcal{W}^*$ and i^* has minimum cost among nodes with resistance at least k^*, we have $c_\ell \geq c_{i^*}$.

Consider the solution $\mathcal{W}' = (\mathcal{W}^* \setminus \{\ell\}) \cup \{i^*\}$. Its cost satisfies $\sum_{i\in\mathcal{W}'} c_i \leq \sum_{i\in\mathcal{W}^*} c_i$. Moreover, $\mathcal{W}'$ remains feasible: for $k \leq k^*$, both ℓ and i^* contribute identically to $\Phi_{\mathcal{W}}(k)$; for $k > k^*$, the condition $\Phi(k) \geq k$ holds by definition of k^*, so the activation condition is already satisfied independently of the target set. Hence $\mathcal{W}'$ is an optimal solution containing i^*, which proves the lemma. □

Algorithm 1 Greedy Algorithm for WTSS on Complete Graphs

Input: Complete graph K_n, resistances $\{r_i\}$, costs $\{c_i\}$
Output: Optimal target set $\mathcal{W}$ and optimal cost C^*
Compute cumulative resistance distribution $\Phi(k)$ for $k \in \{1, \ldots, n\}$
Initialize $\mathcal{W} \leftarrow \emptyset$, $C^* \leftarrow 0$, ext $\leftarrow 0$
Initialize min-heap H ordered by cost c_i
for $k = n - 1$ **down to** 1 **do**
    Insert all nodes with $r_i = k$ into H
    while $\Phi(k) + \text{ext} < k$ **do**
        $i^* \leftarrow$ extract-min(H)
        $\mathcal{W} \leftarrow \mathcal{W} \cup \{i^*\}$
        $C^* \leftarrow C^* + c_{i^*}$
        ext $\leftarrow$ ext $+ 1$
    end while
end for
return $\mathcal{W}$, C^*

Theorem 3. *Let $(K_n, \{\theta_i\}, \{c_i \mathbb{1}_{h_i>0}\})$ be an instance of the WTSS problem. Algorithm 1 computes an optimal target set and its optimal cost.*

Proof. By Lemma 4, there exists an optimal solution containing the node i^* selected at each iteration. The remaining problem reduces to a new WTSS instance with modified distribution $\Phi_{\mathcal{W}}$, to which the same argument applies inductively. □

Corollary 3. *Let $(K_n, \{\theta_i\}, \{c_i \mathbb{1}_{h_i>0}\})$ be an instance of the WTSS problem. Algorithm 1 solves the problem in $O(n \log n)$ time.*

Proof. Computing the cumulative resistance distribution Φ requires $O(n)$ time. The minimum-cost node selection is implemented via a min-heap ordered by cost c_i, supporting insertions and extractions in $O(\log n)$ time. Since there are at most n insertions and n extractions in total, the heap contributes $O(n \log n)$ to the overall complexity. The auxiliary variable ext avoids recomputing $\Phi_{\mathcal{W}}$ at each

iteration, reducing the update cost to $O(1)$ per iteration. Therefore, the overall time complexity is $O(n \log n)$. □

Corollary 4. *Let $(K_n, \{\theta_i\}, \{c_i \mathbb{1}_{h_i>0}\})$ be an instance of the WTSS problem. Every optimal solution activate a number of nodes equal to*

$$M = \max_{k \in \{1,\dots,n\}} \Big[k - \Phi(k)\Big]^+$$

Proof. By Lemma 3, any successful target set must contain at least M nodes. Suppose by contradiction that there exists an optimal solution $\mathcal{W}^*$ with $|\mathcal{W}^*| > M$. Consider the subset $\mathcal{W}' \subseteq \mathcal{W}^*$ consisting of the M nodes in $\mathcal{W}^*$ with the highest resistances. Assume that $\mathcal{W}'$ is not a successful target set. Then there exists k such that

$$\Phi_{\mathcal{W}'}(k) = \Phi(k) + |\{i \in \mathcal{W}' : r_i \geq k\}| < k$$

Since $\mathcal{W}^*$ is successful, we have

$$\Phi_{\mathcal{W}^*}(k) = \Phi(k) + |\{i \in \mathcal{W}^* : r_i \geq k\}| \geq k$$

Note that $|\{i \in \mathcal{W}^* : r_i \geq k\}| > |\{i \in \mathcal{W}' : r_i \geq k\}|$ only if $|\{i \in \mathcal{W}^* : r_i \geq k\}| > M$ and $|\{i \in \mathcal{W}' : r_i \geq k\}| = M$, since $\mathcal{W}'$ consists of the M highest resistance nodes from $\mathcal{W}^*$. Therefore,

$$\Phi_{\mathcal{W}'}(k) = \Phi(k) + M < k$$

which implies $k - \Phi(k) > M$. This contradicts the definition

$$M = \max_{k \in \{1,\dots,n\}} [k - \Phi(k)]^+$$

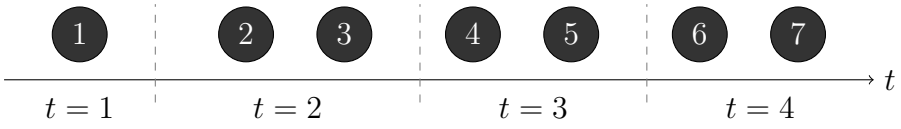
Therefore, $\mathcal{W}'$ must be a successful target set. Since $|\mathcal{W}'| = M < |\mathcal{W}^*|$ and all costs are positive, $\mathcal{W}'$ has strictly lower cost than $\mathcal{W}^*$, contradicting the optimality of $\mathcal{W}^*$. This proves the corollary. □

2.3 General Cost Influence Problem

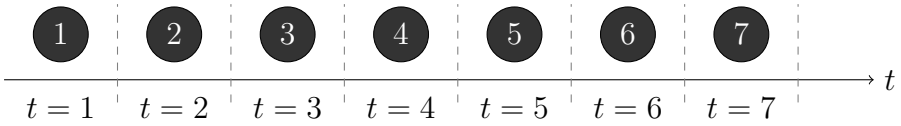
Beyond the targeting setting, Como et al. [10] solved the LCIP with identity cost functions $C_i(h_i) = h_i$ in $O(n \log n)$ time. In this section, we present an original contribution that strictly generalizes this result: an $O(n^3)$ algorithm for the intervention problem on complete graphs under arbitrary threshold distributions $\{\theta_i\}$ and general non-decreasing cost functions $\{C_i\}$, with no assumptions required on either. The key idea is a reduction to minimum-weight perfect matching on a complete bipartite graph. The formal justification relies on the permutation-based

reformulation of Proposition 4, proved in Chapter 3; here we give a self-contained informal argument for the complete graph that motivates the reduction and makes the present section readable independently.

Consider any successful intervention $h \in \mathcal{H}$: starting from $x(0) = \mathbf{0}$, nodes become active in groups at each time step, with possibly multiple nodes activating simultaneously.



We can stretch this process by splitting each time step into as many sub-steps as there are nodes activating at that time, ordering nodes within each group arbitrarily. This assigns a unique position to every node and does not affect the success of the intervention: nodes activated later in a group receive at least as much influence as in the original dynamics, since more nodes are already active before them.



The stretched process is naturally encoded by a permutation $\sigma \in S_n$, where $\sigma(i) = t$ means that node i is activated at position t. On the complete graph, node i at position t receives influence exactly $t - 1$ from the preceding nodes, so the minimum intervention required to ensure its activation is

$$h_i(\sigma) = \Big[r_i - (t-1)\Big]^+ = \Big[r_i - (\sigma(i)-1)\Big]^+$$

A crucial observation is that this cost depends only on $\sigma(i)$, independently of which specific nodes occupy the other positions. The intervention costs therefore *decouple* across nodes, and the problem reduces to finding the permutation $\sigma \in S_n$ minimizing

$$\sum_{i \in V} C_i\Big([r_i - (\sigma(i)-1)]^+\Big)$$

This decoupling structure immediately suggests a reduction to a combinatorial assignment problem. The formal equivalence between this permutation-based formulation and the original LTM intervention problem is established in Chapter 3; here we exploit it directly to derive an efficient algorithm.

Proposition 3. *Any LTM intervention problem instance on a simple complete graph $(K_n, \{\theta_i\}, \{C_i\})$ can be reduced to an instance of the minimum-weight perfect matching problem on a complete bipartite graph in $O(n^2)$ time.*

Proof. The minimum-weight perfect matching problem, consists in searching in a complete bipartite graph $K_{n,n}$ a perfect matching (i.e., a set of n pairwise disjoint edges covering all vertices) whose total weight is minimum, where the weight is computed summing over the weight of edges selected.

Consider any instance of the problem $(K_n, \{\theta_i\}, \{C_i\})$. Construct a bipartite graph where one side represents all nodes $V = \{1, \dots, n\}$ and the other side represents positions $t = 1, \dots, n$. Each real node is connected to every position by an edge with weight

$$W_{it} = C_i\Big([r_i - (t-1)]^+\Big) = C_i\Big([\theta_i(n-1) - (t-1)]^+\Big)$$

Now notice that every perfect matching corresponds to a permutation, and by construction, the weight of the matching equals the total cost of the corresponding permutation. Then a solution to the matching problem is also a solution to the LTM intervention problem. The reduction require $O(n^2)$ steps for the computation of the weight matrix. □

Corollary 5. *Any LTM intervention problem $(K_n, \{\theta_i\}, \{C_i\})$ on a complete graph can be solved exactly in $O(n^3)$ time.*

Proof. By Proposition 3, the problem reduces to finding a minimum-weight perfect matching in a complete bipartite graph with n nodes on each side. This can be solved optimally using the Hungarian algorithm [17, 18], which has time complexity $O(n^3)$. Adding the $O(n^2)$ reduction time yields overall complexity $O(n^3)$. □

2.4 Time-Constrained Weighted Targeting

The WTSS problem with a time constraint on propagation is relevant from an applicative perspective: in many economic and social contexts, the speed of diffusion is as important as its extent. For instance, in the context of viral marketing, a company launching a new product must achieve widespread adoption before a competitor enters the market and captures the audience. Similarly, in the modeling of social unrest, the rapid spread of a protest or collective action may be disrupted by institutional interventions that become effective after a certain time window.

We consider the time-constrained problem $(K_n, \{\theta_i\}, \{c_i \mathbb{1}_{h_i>0}\}, T)$, where T is the time limit within which complete adoption must be reached. When $T \geq n$, the constraint is never binding and the problem reduces to the standard WTSS. For $T < n$, however, a solution that is optimal for the unconstrained problem may fail to achieve complete adoption in time, so the optimal cost of the time-constrained variant is in general higher.

We begin by partitioning the nodes into groups according to their resistance value:

$$\mathcal{G}(k) = \{i \in V : r_i = k\} \qquad k \in \{0, 1, \ldots, n-1\} \tag{2.6}$$

Since on a complete graph the optimal choice of which nodes to target within a group is determined solely by cost, if an optimal solution requires z nodes from $\mathcal{G}(k)$, one should always select the z cheapest ones. We therefore precompute, for each group $\mathcal{G}(k)$ and each $z \leq |\mathcal{G}(k)|$, the quantity $c(z, k)$ as the sum of the z smallest costs among nodes in $\mathcal{G}(k)$. Similarly, we define $ct(z, k)$ as the sum of the z smallest costs among all nodes with resistance at most k, taken across all groups $\mathcal{G}(0), \ldots, \mathcal{G}(k)$.

We solve the time-constrained problem via dynamic programming. The key idea is to process groups from the highest resistance down to the lowest, deciding at each step how many nodes to target in the current group.

Definition 12. *Consider an instance $(K_n, \{\theta_i\}, \{c_i \mathbb{1}_{h_i>0}\}, T)$ of the time-constrained WTSS problem. We denote by*

$$\mathrm{OPT}(t, m, r_{\max}, k_{\text{ext}})$$

the subproblem of activating m nodes within t time steps from the subgraph induced by nodes with resistance at most $r_{\max}$, given that k_{ext} external nodes are already active and contribute both influence and to the count of m activated nodes. We denote by $\omega(\mathrm{OPT}(t, m, r_{\max}, k_{\text{ext}}))$ the minimum cost to solve this subproblem.

The original problem corresponds to $\mathrm{OPT}(T, n, n-1, 0)$. We assume without loss of generality that no node has resistance $r_i = 0$; if such nodes exist, they are treated as external nodes and the problem is reduced accordingly.

Feasibility. For each subproblem, we first check whether it is feasible. If the total number of available nodes is insufficient, i.e.,

$$\left| \bigcup_{g=1}^{r_{\max}} \mathcal{G}(g) \right| + k_{\text{ext}} < m$$

then $\omega(\mathrm{OPT}(t, m, r_{\max}, k_{\text{ext}})) = \infty$.

Recursive Case: $t > 2$**.**

Consider a feasible subproblem with $t > 2$. If $k_{\text{ext}} \geq r_{\max}$ or $k_{\text{ext}} \geq m$, all remaining nodes already receive sufficient influence to activate without further targeting, so the cost is 0. Otherwise, let $p = |\mathcal{G}(r_{\max})|$ and consider the three options for targeting nodes in $\mathcal{G}(r_{\max})$.

The first option is to target all p nodes in $\mathcal{G}(r_{\max})$, and recurse on the remaining groups:

$$\omega_1 = c(p, r_{\max}) + \omega(\mathrm{OPT}(t, m, r_{\max} - 1, k_{\text{ext}} + p))$$

The second option is to target $z < p$ nodes in $\mathcal{G}(r_{\max})$ and ensure that the remaining $p - z$ nodes in $\mathcal{G}(r_{\max})$ activate naturally at time t, which requires exactly $r_{\max}$ active nodes at time $t - 1$:

$$\omega_2 = \min_{z=0,\ldots,p-1} \Big[c(z, r_{\max}) + \omega(\mathrm{OPT}(t-1, r_{\max}, r_{\max} - 1, k_{\text{ext}} + z)) \Big]$$

The third option is to target $z < p$ nodes in $\mathcal{G}(r_{\max})$ and disregard the untargeted ones, focusing only on activating m nodes from lower resistance groups:

$$\omega_3 = \min_{z=0,\ldots,p-1} \Big[c(z, r_{\max}) + \omega(\mathrm{OPT}(t, m, r_{\max} - 1, k_{\text{ext}} + z)) \Big]$$

The recurrence is:

$$\omega(\mathrm{OPT}(t, m, r_{\max}, k_{\text{ext}})) = \begin{cases} \infty & \text{if } \left| \bigcup_{g=1}^{r_{\max}} \mathcal{G}(g) \right| + k_{\text{ext}} < m \\ 0 & \text{if } k_{\text{ext}} \geq r_{\max} \text{ or } k_{\text{ext}} \geq m \\ \min\{\omega_1, \omega_2, \omega_3\} & \text{otherwise} \end{cases} \tag{2.7}$$

Base Case: $t = 2$**.**

Consider a feasible subproblem with $t = 2$. If $k_{\text{ext}} \geq r_{\max}$, all nodes in $\bigcup_{g=1}^{r_{\max}} \mathcal{G}(g)$ already receive sufficient influence to activate at $t = 2$ without further targeting, so the cost is 0. Otherwise, the same three options as above apply, with the following modifications. The first and second options are analogous to the recursive case with $t = 2$:

$$\omega_1 = c(p, r_{\max}) + \omega(\mathrm{OPT}(2, m, r_{\max} - 1, k_{\text{ext}} + p))$$

$$\omega_2 = \min_{z=0,\ldots,p-1} \Big[c(z, r_{\max}) + \omega(\mathrm{OPT}(2, m, r_{\max} - 1, k_{\text{ext}} + z)) \Big]$$

The third option exploits the fact that, with only one time step remaining, one can directly target the $r_{\max} - k_{\text{ext}}$ cheapest nodes across all groups with resistance at most $r_{\max}$ to trigger full activation at $t = 2$:

$$\omega_3 = ct(r_{\max} - k_{\text{ext}},\, r_{\max})$$

The recurrence is:

$$\omega(\mathrm{OPT}(2, m, r_{\max}, k_{\text{ext}})) = \begin{cases} \infty & \text{if } \left| \bigcup_{g=1}^{r_{\max}} \mathcal{G}(g) \right| + k_{\text{ext}} < m \\ 0 & \text{if } k_{\text{ext}} \geq r_{\max} \\ \min\{\omega_1, \omega_2, \omega_3\} & \text{otherwise} \end{cases} \tag{2.8}$$

Base Case: $r_{\max} = 0$**.**

When no nodes remain to be processed:

$$\omega(\mathrm{OPT}(t, m, 0, k_{\mathrm{ext}})) = \begin{cases} 0 & \text{if } k_{\mathrm{ext}} \geq m \\ \infty & \text{otherwise} \end{cases} \tag{2.9}$$

Algorithm.

Algorithm 2 Dynamic Programming for Time-Constrained WTSS

Require: Complete graph K_n, resistances $\{r_i\}$, costs $\{c_i\}$, time limit T
Ensure: Optimal cost $\omega(\mathrm{OPT}(T, n, n-1, 0))$

1: Partition nodes into groups $\mathcal{G}(k) = \{i \in V : r_i = k\}$ for $k = 1, \ldots, n-1$
2: Precompute $c(z, k)$ for all k and $z \leq |\mathcal{G}(k)|$ by sorting each group by cost
3: Precompute $ct(z, k)$ for all k and z by sorting nodes with resistance at most k by cost
4: Initialize memoization table
5: **return** $\mathrm{DP}(T, n, n-1, 0)$

Theorem 4. *Let $(K_n, \{\theta_i\}, \{c_i \mathbb{1}_{h_i>0}\}, T)$ be an instance of the time-constrained WTSS problem. Algorithm 2 solves the problem in $O(T \cdot n^4)$ time.*

Proof. The preprocessing step computes $c(z, k)$ and $ct(z, k)$ by sorting nodes within each group and across groups, respectively, both requiring $O(n^2 \log n)$ time. The number of distinct subproblems is $O(T \cdot n^3)$, since $t \in \{1, \ldots, T\}$, $m \in \{1, \ldots, n\}$, $r_{\max} \in \{0, \ldots, n-1\}$, and $k_{\mathrm{ext}} \in \{0, \ldots, n\}$. Each subproblem requires evaluating $O(n)$ choices of z, each computable in $O(1)$ using the precomputed tables. The overall complexity is therefore $O(n^2 \log n) + O(T \cdot n^4) = O(T \cdot n^4)$. □

It is worth noting that, beyond the optimal cost, an optimal target set can be reconstructed efficiently by storing, for each subproblem, the optimal choice of z and the corresponding strategy at the time of computation, and then tracing back through the memoization table.

Although the algorithm is polynomial, the $O(T \cdot n^4)$ complexity makes it computationally heavy for large instances. A natural question is whether the tools developed for the unconstrained setting – in particular the greedy approach of Section 2.2 - can be extended or adapted to handle the time constraint more efficiently. We leave this as an open problem.

Algorithm 3 Recursive DP Function

function DP($t, m, r_{\max}, k_{\text{ext}}$)
  if $(t, m, r_{\max}, k_{\text{ext}})$ already computed **then**
    return $\omega(\text{OPT}(t, m, r_{\max}, k_{\text{ext}}))$
  end if
  if $r_{\max} = 0$ **then**
    return 0 if $k_{\text{ext}} \geq m$, else ∞
  end if
  if $|\bigcup_{g=1}^{r_{\max}} \mathcal{G}(g)| + k_{\text{ext}} < m$ **then**
    return ∞
  end if
  if $k_{\text{ext}} \geq r_{\max}$ **or** $k_{\text{ext}} \geq m$ **then**
    return 0
  end if
  $p \leftarrow |\mathcal{G}(r_{\max})|$
  if $t = 2$ **then**
    Compute $\omega_1, \omega_2, \omega_3$ using (2.8)
  else
    Compute $\omega_1, \omega_2, \omega_3$ using (2.7)
  end if
  return $\min\{\omega_1, \omega_2, \omega_3\}$
end function

Chapter 3

Combinatorial Reformulation

In this chapter, we present the combinatorial reformulation of the LTM intervention problem introduced by Como et al. [10], which transforms the search over intervention vectors into an optimization over the symmetric group $\mathcal{S}_n$. It serves a dual purpose in this thesis: it is the algebraic foundation underlying the exact algorithms for the complete graph developed in Chapter 2, and it provides the natural search space for the local search and simulated annealing algorithms of Chapter 5.

The literature offers several results building on this perspective. Como et al. [10] derived optimal permutations for complete graphs under identity costs, obtaining an $O(n \log n)$ greedy algorithm for the LCIP, characterized the structure of optimal permutations for path and cycle graphs via the reversed problem, and characterized the optimal solution for the uniform threshold case $\theta_i = 1/2$ on simple graphs. For general graphs, Günnec et al. [14] proposed a MILP formulation based on acyclic orientations as a tighter search space for the intervention problem.

Our contribution is the extension of this framework to complete bipartite, and complete multipartite graphs, interesting graph families, for which we derive an $O(n \log n)$ optimal algorithms under identity cost functions.

3.1 Reformulation

The LTM intervention problem (1.9) is defined over the set $\mathcal{H}$, which despite the reduction to $\prod_{i \in V} \mathcal{H}_i$ remains computationally intractable in general. We now present a combinatorial reformulation, that transforms the search over intervention vectors into an optimization over permutations in S_n, the symmetric group over the node set $V = \{1, \ldots, n\}$.

Given a permutation $\sigma \in S_n$, we define the *sequential activation dynamics* as follows: nodes are activated one at a time in the order prescribed by σ, and at step k the node i with $\sigma(i) = k$ becomes active if the cumulative influence received from all previously activated nodes meets or exceeds its resistance. The minimum intervention required to guarantee the activation of node i at position $\sigma(i)$, assuming all preceding nodes are already active, is

$$h_i(\sigma) = \Big[r_i - w_i(\sigma)\Big]^+, \qquad w_i(\sigma) = \sum_{j\,:\,\sigma(j)<\sigma(i)} W_{ij} \tag{3.1}$$

where $w_i(\sigma)$ is the cumulative influence received by node i from all preceding nodes. The total cost associated with σ is

$$C(\sigma) = \sum_{i\in V} C_i\Big(h_i(\sigma)\Big) \tag{3.2}$$

giving rise to the optimization problem

$$\min_{\sigma\in S_n} C(\sigma) \tag{3.3}$$

Proposition 4. *Let $(G, \{\theta_i\}, \{C_i\})$ be an instance of the LTM intervention problem, it holds that*

$$\min_{h\in\mathcal{H}} \sum_{i\in V} C_i(h_i) = \min_{\sigma\in S_n} C(\sigma) \tag{3.4}$$

Proof. We prove both inequalities separately.

Step 1. We show that $\min_{h\in\mathcal{H}} \sum_{i\in V} C_i(h_i) \leq \min_{\sigma\in S_n} C(\sigma)$ by proving that for every permutation $\sigma \in S_n$, the intervention vector $h(\sigma) \in \mathcal{H}$, i.e. guarantees full adoption under the standard LTM dynamics. Fix a permutation σ, we claim that under intervention $h(\sigma)$, all nodes i with $\sigma(i) \leq t$ are active at time t. The proof proceeds by induction on t. We start from $x(0) = \mathbf{0}$.

Base case ($t = 1$): Let i be the node with $\sigma(i) = 1$. By definition (3.1), $w_i(\sigma) = 0$ because there are no nodes before, so $h_i(\sigma) = r_i$ by (3.1). Under this intervention, the reduced resistance is $r_i - h_i(\sigma) = 0$, ensuring that $f_{r_i - h_i(\sigma)}(0) = 1$ and then $x_i(1) = 1$.

Inductive step: Assume all nodes j with $\sigma(j) \leq t$ are active at time t, i.e. $x_j(t) = 1$ for every j. At time $t + 1$, by monotonicity of the dynamics, all nodes j with $\sigma(j) \leq t$ remain active, i.e. $x_j(t + 1) = 1$ for every j. In addition, consider the node i with $\sigma(i) = t + 1$. At time $t + 1$ node i receives influence

$$\sum_j W_{ij} x_j(t) \geq \sum_{j\,:\,\sigma(j)\leq t} W_{ij} = \sum_{j\,:\,\sigma(j)<\sigma(i)} W_{ij} = w_i(\sigma)$$

By construction (3.1), $h_i(\sigma) = [r_i - w_i(\sigma)]^+$, which implies that the reduced resistance satisfies $f_{r_i - h_i(\sigma)}(\sum_j W_{ij} x_j(t)) = 1$. This completes the induction.

Therefore, all nodes eventually activate, confirming $h(\sigma) \in \mathcal{H}$.

Step 2. We show that $\min_{h \in \mathcal{H}} \sum_{i \in V} C_i(h_i) \geq \min_{\sigma \in S_n} C(\sigma)$ by constructing, for any successful intervention $h \in \mathcal{H}$, a permutation σ with $h(\sigma) \leq h$ (component wise), which implies $C(\sigma) \leq \sum_i C_i(h_i)$ by monotonicity of the cost functions. Given $h \in \mathcal{H}$, partition the set of nodes by activation time under the standard LTM dynamics:

$$\mathcal{X}(t) = \{i \in V \,:\, x_i(t) = 1 \text{ and } x_i(t-1) = 0\}$$

By monotonicity of the LTM dynamics, once a node activates, it remains active. Therefore, each node activates at most once, ensuring that the activation times partition the nodes. Since h is a successful intervention, $\bigcup_{t=1}^{n} \mathcal{X}(t) = V$, forming a partition of V. Under the original dynamics with intervention h, a node $i \in \mathcal{X}(t)$ receives influence

$$\sum_{s<t} \sum_{j \in \mathcal{X}(s)} W_{ij}$$

at time t (when it activates). Then, to activate node i at time t, we must have

$$h_i \geq r_i - \sum_{s<t} \sum_{j \in \mathcal{X}(s)} W_{ij}$$

Now, construct a permutation σ by ordering nodes first by activation time, then arbitrarily within each set $\mathcal{X}(t)$. Under permutation σ, node i receives influence

$$w_i(\sigma) = \sum_{j \,:\, \sigma(j) < \sigma(i)} W_{ij} \geq \sum_{s<t} \sum_{j \in \mathcal{X}(s)} W_{ij}$$

since the left-hand side may include additional nodes from $\mathcal{X}(t)$ that precede i in the ordering within $\mathcal{X}(t)$. Consequently if we consider the permutation intervention $h_i(\sigma)$,

$$h_i(\sigma) = [r_i - w_i(\sigma)]^+ \leq \left[r_i - \sum_{s<t} \sum_{j \in \mathcal{X}(s)} W_{ij}\right]^+ \leq h_i$$

From the first part of the proof, we know that each permutation σ induces a successful intervention vector $h(\sigma) \in \mathcal{H}$, which belongs to the set of permutation-based solutions. By monotonicity of the cost functions, we have $C(\sigma) \leq C(h)$. Since this holds for arbitrary $h \in \mathcal{H}$, the result follows. □

This equivalence reduces the search space from $|\mathcal{H}|$ candidate intervention vectors to $n!$ permutations, transforming the original problem into a combinatorial optimization over S_n. The symmetric group carries a rich algebraic structure —

in particular, the neighborhood structure induced by transpositions — that lends itself naturally to combinatorial algorithms and metaheuristic search, as developed in the remainder of this chapter and in Chapter 5. The search space can be further reduced to $|\mathcal{A}(G)| \leq n!$ acyclic orientations of G, as discussed in Section 3.4.

Henceforth, we refer to the LTM intervention problem in both formulations interchangeably: a solution is represented either as an intervention vector h^* or as a permutation σ^*, with the understanding that the latter induces the intervention vector $h(\sigma^*)$.

3.2 Reversed Problem

A natural symmetry of the combinatorial reformulation relates any problem instance to its *reversed* counterpart. Throughout this section, we assume G is undirected, i.e. $W_{ij} = W_{ji}$ for all $i, j \in V$.

Definition 13. *Given a permutation $\sigma \in S_n$, its* reversed permutation $\bar{\sigma}$ *is defined by*

$$\bar{\sigma}(i) = n - \sigma(i) + 1 \qquad \forall\, i \in V$$

i.e. $\bar{\sigma}$ reverses the activation order induced by σ.

Definition 14. *Given an instance $(G, \{\theta_i\}, \{C_i\})$, the* reversed problem *is defined as $(G, \{1 - \theta_i\}, \{C_i\})$.*

Lemma 5. *Let $(G, \{\theta_i\}, \{C_i\})$ be an instance of the LTM intervention problem on an undirected graph, and let $(G, \{1 - \theta_i\}, \{C_i\})$ be its reversed problem. Denote by $C(\sigma)$ and $\bar{C}(\sigma)$ the costs in the original and reversed problems, respectively. Then for every $\sigma \in S_n$,*

$$\bar{C}(\bar{\sigma}) = C(\sigma) - \left(\sum_{i \in V} \theta_i w_i - W^* \right) \tag{3.5}$$

where $W^* = \frac{1}{2} \sum_{i \in V} w_i$ is the total edge weight.

Proof. Since G is undirected, the influence received by node i under $\bar{\sigma}$ satisfies

$$w_i(\bar{\sigma}) = \sum_{j \,:\, \bar{\sigma}(j) < \bar{\sigma}(i)} W_{ij} = \sum_{j \,:\, \sigma(j) > \sigma(i)} W_{ij} = w_i - w_i(\sigma)$$

Using the identity $[a]^+ - [a]^- = a$ and $[-a]^+ = [a]^-$, the cost of $\bar{\sigma}$ in the reversed problem is

$$\bar{C}(\bar{\sigma}) = \sum_{i \in V} \Big[(1 - \theta_i) w_i - w_i(\bar{\sigma}) \Big]^+ = \sum_{i \in V} \Big[\theta_i w_i - w_i(\sigma) \Big]^-$$

$$= \sum_{i \in V} \Big[\theta_i w_i - w_i(\sigma)\Big]^+ - \sum_{i \in V} \Big(\theta_i w_i - w_i(\sigma)\Big) = C(\sigma) - \Big(\sum_{i \in V} \theta_i w_i - \sum_{i \in V} w_i(\sigma)\Big)$$

Finally, for any permutation σ,

$$\sum_{i \in V} w_i(\sigma) = \sum_{i \in V} \sum_{j\,:\,\sigma(j) < \sigma(i)} W_{ij} = \sum_{\{i,j\} \in E} W_{ij} = W^*$$

which completes the proof. □

Corollary 6. *If σ^* is optimal for $(G, \{\theta_i\}, \{C_i\})$, then $\bar{\sigma}^*$ is optimal for the reversed problem $(G, \{1 - \theta_i\}, \{C_i\})$.*

Proof. The quantity $\sum_{i \in V} \theta_i w_i - W^*$ is independent of σ, so the result follows immediately from Lemma 5. □

Paths and Cycles

Throughout this and the following subsections, we assume $G = (V, E, W)$ is a simple graph, i.e. $W_{ij} = W_{ji} \in \{0,1\}$ for all $i, j \in V$, so that w_i coincides with the degree of node i. We further assume linear costs $C_i(h_i) = h_i$.

Example 6 (Path Graphs)**.** *Consider a path graph L_n with node set $V = \{1, \ldots, n\}$ ordered sequentially, where nodes 1 and n are the endpoints (degree 1) and all intermediate nodes have degree 2. The identity permutation $\sigma = \mathrm{id}$ (activating nodes in order $1, 2, \ldots, n$) has cost*

$$C(\mathrm{id}) = \theta_1 + \sum_{i=2}^{n-1} [2\theta_i - 1]^+ \tag{3.6}$$

Optimal permutations can be characterized by the threshold values.

Case 1. If $\theta_i = \theta \leq \frac{1}{2}$ for all $i \in V$, then $\sigma = \mathrm{id}$ is optimal with cost $C(\mathrm{id}) = \theta$.

Case 2. If $\theta_i \leq \frac{1}{2}$ for all $i \in V$, an optimal permutation is obtained by selecting $i^* \in \arg\min_{i \in V} w_i \theta_i$ as the first node, then progressively activating adjacent nodes.

Case 3. If $\theta_i \geq \frac{1}{2}$ for all $i \in V$, an optimal permutation σ^* is obtained by solving the reversed problem with thresholds $\{1 - \theta_i\}$ (which falls under Case 2), obtaining τ^*, and setting $\sigma^* = \bar{\tau}^*$ via Corollary 6.

Remark 2. *Analogous characterizations hold for cycle graphs C_n. Since every node has degree 2, the same case analysis applies, with the additional observation that the ring has no endpoints, so the starting node can be chosen freely among those minimizing $w_i \theta_i$.*

3.3 Complete and Related Graphs

On the complete graph, the permutation reformulation takes a particularly tractable form: the influence received by any node depends only on its position in the activation order, independently of which specific nodes precede it. This decoupling structure admits exact polynomial-time algorithms for several variants of the problem. The result for the LCIP with identity cost functions, $C_i(h_i) = h_i$ for all $i \in V$, is due to Como et al. [10]; the extensions to complete bipartite and complete multipartite graphs under the same cost structure are original contributions of this thesis. Throughout this section, all graphs are assumed simple.

3.3.1 Complete Graphs

Consider the complete graph K_n. Every node has degree $w_i = n - 1$, so the resistance of node i is $r_i = \theta_i(n-1)$. A key structural observation is that, under any permutation $\sigma \in S_n$, the influence received by node i at position $\sigma(i) = t$ is

$$w_i(\sigma) = t - 1 \tag{3.7}$$

since every pair of nodes is connected with unit weight $W_{ij} = 1$. In particular, $w_i(\sigma)$ depends on σ only through $\sigma(i)$, not through the identities of the preceding nodes. The total cost therefore takes the form

$$C(\sigma) = \sum_{i \in V} \Big[\theta_i(n-1) - (\sigma(i) - 1)\Big]^+ \tag{3.8}$$

Proposition 5. *Let $(K_n, \{\theta_i\}, \{\mathrm{id}\})$ be an instance of the LTM intervention problem. There exists an optimal permutation σ^* in which nodes are ordered by non-decreasing resistance:*

$$\sigma^*(i) < \sigma^*(j) \implies r_i \le r_j \qquad \text{for all } i, j \in V \tag{3.9}$$

Consequently, an optimal solution can be computed in $O(n \log n)$ time.

Proof. Let $\sigma \in S_n$ be any permutation and suppose $r_i < r_j$ but $\sigma(i) = h > k = \sigma(j)$ for some $i, j \in V$. Define σ' by swapping the positions of i and j: $\sigma'(i) = k$, $\sigma'(j) = h$, and $\sigma'(\ell) = \sigma(\ell)$ for all $\ell \neq i, j$. Since all other positions are unchanged, it suffices to compare the contributions of i and j to $C(\sigma)$ and $C(\sigma')$. Under σ, these are

$$\Big[r_i - (h-1)\Big]^+ + \Big[r_j - (k-1)\Big]^+$$

and under σ',

$$\Big[r_i - (k-1)\Big]^+ + \Big[r_j - (h-1)\Big]^+$$

Since $r_i < r_j$ and $k < h$, we have

$$\Big[r_i - (k-1)\Big]^+ + \Big[r_j - (h-1)\Big]^+ \leq \Big[r_i - (h-1)\Big]^+ + \Big[r_j - (k-1)\Big]^+$$

hence $C(\sigma') \leq C(\sigma)$. Repeated application of this swap transforms any permutation into one satisfying (3.9) without increasing cost, proving the existence of an optimal solution of this form. Such a permutation is obtained by sorting nodes by non-decreasing resistance, which requires $O(n \log n)$ time. □

3.3.2 Complete Bipartite Graphs

We now extend this analysis to complete bipartite graphs, presenting an original contribution of this thesis.

Consider $K_{n,m}$ with parts V_1 and V_2, with $|V_1| = n$ and $|V_2| = m$. The weight matrix satisfies $W_{ij} = 1$ if and only if i and j belong to different partitions, and $W_{ij} = 0$ otherwise. For the LTM intervention problem on $(K_{n,m}, \{\theta_i\}, \{id\})$, we construct an optimal permutation via a greedy approach.

Given any permutation σ, define the induced orderings τ_1 and τ_2 on V_1 and V_2 respectively, where $\tau_k(i)$ denotes the relative position of node $i \in V_k$ among nodes from V_k in σ.

Proposition 6. *Let $(K_{n,m}, \{\theta_i\}, \{id\})$ be an LTM intervention problem on a bipartite complete graph, then there exists an optimal permutation σ^* such that both τ_1 and τ_2 order nodes by increasing threshold within their respective parts, i.e.,*

$$\tau_k(i) < \tau_k(j) \iff r_i \leq r_j \quad \textit{for all } i, j \in V_k,\ k \in \{1{,}2\}$$

Proof. Observe that modifying the internal ordering within V_1 (or V_2) does not affect the intervention requirements for nodes in V_2 (or V_1), since the graph is complete bipartite with $W_{ij} = 1$ for all existing edges.

Consider $i, j \in V_k$ with $r_i < r_j$ and $\tau_k(i) = h > l = \tau_k(j)$. By the same exchange argument as in Proposition 5, swapping their relative positions within V_k does not increase cost. Therefore, there exists an optimal solution where each part is internally ordered by threshold. □

It remains to determine how to interleave nodes from V_1 and V_2.

Proposition 7. *Let $(K_{n,m}, \{\theta_i\}, \{id\})$ be an instance of the LTM intervention problem on a complete bipartite graph. The greedy Algorithm 4 constructs an optimal permutation σ^*.*

Proof. By Proposition 6, there exists an optimal permutation in which nodes within each partition are ordered by increasing resistance. Therefore, we restrict our attention to permutations of this form.

Let $i_1^* \in V_1$ and $i_2^* \in V_2$ denote the nodes with minimum resistance in their respective partitions. Without loss of generality, assume $r_{i_1^*} \leq r_{i_2^*}$, so the greedy algorithm selects $i^* = i_1^*$ first. Consider any permutation σ with $\sigma(i_2^*) = 1$ and $\sigma(i_1^*) = k$ for some $k > 1$. By Proposition 6 and the minimality of $r_{i_1^*}$ in V_1, all nodes at positions $2, \ldots, k-1$ must belong to V_2, ordered by increasing resistance. The total intervention cost for positions 1 through k under σ is

$$\sum_{j:\sigma(j)<k} r_j + [r_{i_1^*} - (k-1)]^+$$

Now consider the permutation σ' obtained by placing i_1^* at position 1, i_2^* at position 2, and shifting the nodes previously at positions $2, \ldots, k-1$ to positions $3, \ldots, k$. The total intervention cost for positions 1 through k under σ' is

$$r_{i_1^*} + [r_{i_2^*} - 1]^+ + \sum_{j:3\leq\sigma'(j)\leq k} [r_j - 1]^+$$

Since $r_{i_1^*} \leq r_{i_2^*} \leq r_j$ for all nodes j at positions $2, \ldots, k-1$ in σ, the cost under σ' is at most that under σ. Therefore, choosing the minimum-resistance node first is never suboptimal. After activating i^*, all adjacent nodes in the opposite partition have their resistance reduced by 1, yielding a reduced problem on the remaining graph. Applying the greedy selection inductively completes the proof. □

3.3.3 Complete Multipartite Graphs

The analysis of the complete bipartite case extends naturally to complete multipartite graphs. The key observation is that Algorithm 4, already introduced for $K_{n,m}$, applies unchanged to this more general setting.

Theorem 5. *Let $(G, \{\theta_i\}, \{\mathrm{id}\})$ be an instance of the LTM intervention problem, where G is a complete multipartite graph with parts $V_1, \ldots, V_K$. Algorithm 4 constructs an optimal permutation.*

Proof. The proof generalizes the two-step argument of Proposition 7. First, by the same exchange argument as in Proposition 6, within each part V_k there exists an optimal permutation in which nodes are ordered by non-decreasing resistance: swapping two nodes $i, j \in V_k$ with $r_i < r_j$ and $\tau_k(i) > \tau_k(j)$ does not alter the intervention costs of nodes in any other part, since all inter-part weights are equal to 1. Second, restricting attention to permutations of this form, the interleaving across parts is determined by the same greedy argument as in Proposition 7: placing the global minimum-resistance node i^* first is never suboptimal, and after its activation the resistances of all adjacent nodes are reduced by 1, yielding a reduced instance on the same complete multipartite structure. □

Algorithm 4 Greedy Algorithm for Complete Multipartite Graphs

Input: Complete multipartite graph G, resistances $\{r_i\}$
Output: Optimal permutation σ
Initialize $S \leftarrow V$, $\sigma \leftarrow \emptyset$, $t \leftarrow 1$
while $S \neq \emptyset$ **do**
    $i^* \leftarrow \arg\min_{i \in S} r_i$
    $\sigma(i^*) \leftarrow t$
    $S \leftarrow S \setminus \{i^*\}$
    for each $j \in S$ adjacent to i^* **do**
        $r_j \leftarrow r_j - 1$
    end for
    $t \leftarrow t + 1$
end while
return σ

Remark 3. *The complete graph K_n is the special case $K = n$ with $|V_k| = 1$ for all k, consistently with Proposition 5.*

Remark 4. *The decoupling structure of* (3.7) *extends beyond the linear cost setting: for arbitrary non-decreasing cost functions $\{C_i\}$, the intervention cost of node i at position t depends only on i and t, independently of the remaining nodes. This observation yields a reduction to minimum-weight perfect matching and an $O(n^3)$ exact algorithm via the Hungarian method, which is developed in full in Chapter 2.*

3.4 Acyclic Orientations

The permutation reformulation admits a further structural reduction. Recall that $w_i(\sigma) = \sum_{j\,:\,\sigma(j)<\sigma(i)} W_{ij}$ depends on σ only through the set of predecessors of i, not through their internal ordering. This observation motivates the following definition.

Definition 15. *An* acyclic orientation *of G is an assignment of a direction to each edge $\{i, j\} \in E$ such that the resulting directed graph is acyclic. We denote by $\mathcal{A}(G)$ the set of all acyclic orientations of G.*

Every permutation $\sigma \in S_n$ induces an acyclic orientation of G by directing each edge $\{i, j\}$ from i to j if $\sigma(i) < \sigma(j)$. Conversely, every acyclic orientation is induced by at least one permutation, namely any topological ordering of the resulting DAG; in general, multiple permutations may induce the same orientation. Acyclic orientations have a natural interpretation in the LTM context: the direction of each edge encodes the direction of influence propagation, with $i \to j$ meaning

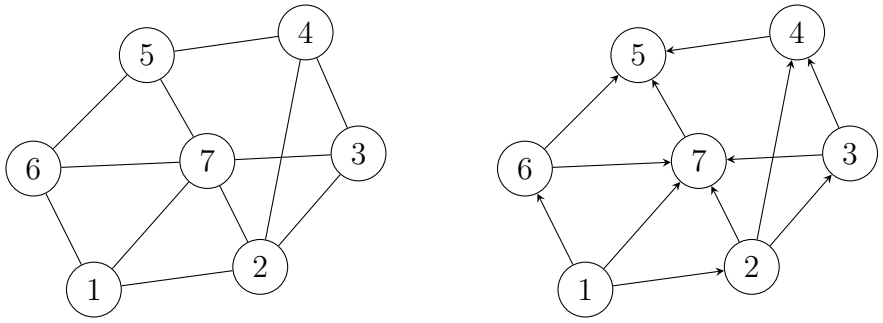


Figure 3.1: An undirected graph G on seven nodes and one of its acyclic orientations O.

that i activates before j and thus contributes to j's cumulative influence. Every acyclic orientation therefore represents a feasible propagation scheme, in which influence flows along a well-defined partial order and no circular dependency arises. A non-acyclic orientation, by contrast, contains a directed cycle, which would require a group of nodes to mutually activate one another before any of them has been activated, a logical impossibility under the dynamics of the LTM.

For a fixed acyclic orientation $O \in \mathcal{A}(G)$, the in-neighborhood of each node i is well-defined as the set of predecessors $\mathcal{N}_O^-(i) = \{j \in \mathcal{N}_i : (j,i) \in O\}$. The corresponding intervention vector is

$$h_i(O) = \Big[r_i - \sum_{j \in \mathcal{N}_O^-(i)} W_{ij} \Big]^+ \qquad \forall\, i \in V \tag{3.10}$$

which is minimal in the sense that no component can be reduced without violating full adoption. Since $w_i(\sigma)$ depends on σ only through $\mathcal{N}_O^-(i)$, any two permutations inducing the same orientation O yield the same intervention vector $h(O)$ and hence the same cost $C(O)$. The optimization problem $\min_{\sigma \in S_n} C(\sigma)$ therefore admits the equivalent formulation

$$\min_{O \in \mathcal{A}(G)} C(O) \tag{3.11}$$

Since $|\mathcal{A}(G)| \leq n!$, with equality only on the complete graph, this search space is strictly tighter than S_n for most graphs. However, despite the smaller search space, acyclic orientations are harder to handle directly than permutations.

Günneç et al. [14] address this difficulty by reformulating (3.11) as a mixed-integer linear program, introducing for each edge $\{i,j\} \in E$ a binary variable $y_{ij} \in \{0,1\}$ equal to 1 if node i precedes node j in the activation order, and a continuous variable $p_i \geq 0$ representing the incentive paid to node i. Adapted to

the general weight setting of this thesis, the formulation reads:

$$\begin{aligned}
\min \quad & \sum_{i \in V} p_i \\
\text{s.t.} \quad & y_{ij} + y_{ji} = 1 && \forall\, \{i,j\} \in E \\
& p_i + \sum_{j \in \mathcal{N}_i} W_{ij}\, y_{ji} \geq r_i && \forall\, i \in V \\
& \sum_{\{i,j\} \in C} y_{ij} \leq |C| - 1 && \forall\, \text{directed cycle } C \subseteq G(y) \\
& p_i \geq 0, \quad y_{ij} \in \{0,1\} && \forall\, i \in V,\, \{i,j\} \in E
\end{aligned} \tag{3.12}$$

The first constraint enforces that each edge is oriented in exactly one direction. The second ensures that p_i covers the gap between r_i and the influence received by i from its predecessors. The third family, known as *k-dicycle inequalities*, enforces acyclicity of the orientation; since directed cycles are exponentially many, these constraints are separated dynamically in the branch-and-cut procedure.

Como et al. [10] exploit the acyclic orientation perspective from a different angle, characterizing the optimal cost for simple graphs with uniform threshold $\theta_i = 1/2$. In this special case the problem reduces to computing the *oriented path number* of G, defined as the minimum number of vertex-disjoint directed paths needed to cover a DAG realization of G.

Chapter 4

Exact Algorithms for Low-Degree Graph Topologies

In this chapter we focus on the LTM intervention problems $(G, \{\theta_i\}, \{C_i\})$ with general costs, within low-degree graphs characterized by elementary topologies; we refer specifically to trees, paths, and unicyclic graphs and their simple compositions. The literature offers several results regarding exact and efficient algorithms for these fundamental structures, but all are restricted to simple graphs and to either linear cost functions $C_i(h_i) = c_i h_i$ or targeting cost functions $C_i(h_i) = c_i \mathbb{1}_{h_i>0}$. Instead, we generalize these results to general weights and general cost functions.

Specifically, regarding the WTSS problem under the equal influence scenario, Raghavan et al. [13] proposed distinct linear-time algorithms for trees and cycles. The LCIP with linear costs has also been investigated. While Cordasco et al. [16] initially provided a polynomial-time solution for simple trees with identity cost functions, Günnec et al. [12] later proposed a more efficient and general approach. Precisely, they developed a dynamic programming algorithm capable of addressing simple trees with general linear costs; furthermore, as seen in the introduction, the authors established the NP-hardness of the problem when transitioning to general trees with heterogeneous influence on edges. Moreover, the time-bounded variant of the problem has been addressed by Cordasco et al. [11], with specialized algorithms for trees and paths under conditions of equal influence and identity cost functions.

Our contributions are threefold. First, we generalize the dynamic programming approach of Günnec et al. [12] to bounded-degree trees with arbitrary edge weights and general non-decreasing cost functions $\{C_i\}$, removing all structural assumptions on both the influence weights and the cost structure. Second, we develop linear-time algorithms for paths and, notably, for cycles. Third, we show that the approach

naturally extends to composite graph structures, opening the way to tackling intervention problems on graphs that can be decomposed into simpler topological components.

We start by recalling the dynamic programming algorithm proposed by Günnec et al. [12], whose underlying principles serve as the basis for our generalizations.

4.1 Trees

We consider instances of the LTM intervention problem on simple trees with linear cost functions $C_i(h_i) = c_i \cdot h_i$. For simple graphs, without loss of generality, we may assume that the resistance r_i of each node satisfies $r_i \in [1, \deg(i)]$. Indeed, if $r_i < 1$, there is no benefit to providing partial intervention to node i: either it activates through propagation at no cost, or it requires full intervention at cost $C_i(r_i)$. In the latter case, we can equivalently set $c_i' = C_i(r_i)$ and $r_i' = 1$ without affecting the optimal solution. We formalize this as follows.

Definition 16 (Standard Form)**.** *An instance* $(G, \{\theta_i\}, \{C_i\})$ *of the LTM intervention problem on a simple graph is in* standard form *if all resistances satisfy* $r_i \in [1, \deg(i)]$ *for all* $i \in V$*.*

Throughout this section, we assume all instances are in standard form. We now analyze the problem on star graphs, which serve as the building block for the general tree algorithm.

4.1.1 Star Graphs

Consider a simple star graph S_n with a central node c and $n-1$ leaf nodes, with linear costs $C_i(h_i) = c_i \cdot h_i$. We assume the instance is in standard form, so $r_i = 1$ for every leaf node, since each leaf has degree 1.

The key observation is that activating the central node c triggers complete diffusion throughout the star, since all leaf nodes immediately receive sufficient influence. The problem therefore reduces to minimizing the cost of activating c.

Let $r_c = \theta_c \cdot \deg(c)$ denote the resistance of the central node. Each leaf node i, once active, provides unit influence to c and can be activated at cost $c_i \cdot r_i = c_i$. The central node requires at least $n_c = \lceil r_c \rceil$ active neighbors to activate without direct intervention. Since providing one unit of influence to c costs c_c via direct intervention and c_i via activating leaf i, it is cost-efficient to activate leaf i rather than intervene on c directly if and only if $c_i \leq c_c$. We therefore define

$$S = \{i \in \mathcal{N}_c : c_i \leq c_c\}$$

as the set of leaves whose activation is preferable to direct intervention on c.

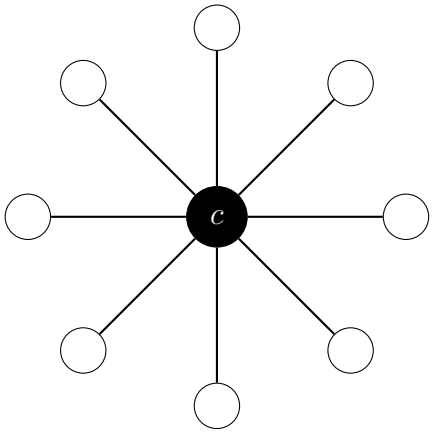


Figure 4.1: A star graph S_n with central node c and $n-1$ leaf nodes.

Two cases arise. If $|S| \geq n_c$, the central node can be activated entirely through leaf activations. Sorting the leaves in S by increasing cost as $c_{(1)} \leq c_{(2)} \leq \cdots \leq c_{(|S|)}$, we activate the $n_c - 1$ cheapest leaves fully, while for the n_c-th leaf we choose between full activation and providing the remaining resistance directly to c:

$$\mathcal{K} = \sum_{j=1}^{n_c-1} c_{(j)} + \min\Big(c_{(n_c)},\ c_c \cdot (r_c - (n_c - 1))\Big) \tag{4.1}$$

If instead $|S| < n_c$, all leaves in S are activated and the remaining resistance of c is covered by direct intervention:

$$\mathcal{K} = \sum_{i \in S} c_i + c_c \cdot (r_c - |S|) \tag{4.2}$$

Algorithm 5 Optimal Activation for Star Graphs

Input: Resistance r_c, cost coefficient c_c of the central node, leaf costs $\{c_i\}_{i \in \mathcal{N}_c}$
Output: Minimum activation cost $\mathcal{K}$
$n_c \leftarrow \lceil r_c \rceil$
$S \leftarrow \{i \in \mathcal{N}_c : c_i \leq c_c\}$
if $|S| \geq n_c$ **then**
    Sort S by cost: $c_{(1)} \leq c_{(2)} \leq \cdots \leq c_{(|S|)}$
    $\mathcal{K} \leftarrow \sum_{j=1}^{n_c-1} c_{(j)} + \min(c_{(n_c)},\ c_c \cdot (r_c - (n_c - 1)))$
else
    $\mathcal{K} \leftarrow \sum_{i \in S} c_i + c_c \cdot (r_c - |S|)$
end if
return $\mathcal{K}$

The algorithm runs in $O(n)$ time: computing S requires a linear scan, and identifying the n_c cheapest elements can be done in $O(n)$ via the Quick-select algorithm [12], avoiding the need for a full sort.

4.1.2 Simple Trees with Linear Costs

The dynamic programming algorithm for simple trees with linear costs is based on a recursive decomposition that reduces the tree problem to a sequence of star subproblems.

Consider a tree T_n rooted at an arbitrary node r. For each node v, let T_v denote the subtree rooted at v. For each subtree, we compute two optimal activation costs: ω_v^{NI}, the minimum cost to activate all nodes in T_v when v receives no influence from its parent, and ω_v^I, the minimum cost to activate all nodes in T_v when v receives influence from its parent, which reduces its resistance to $r'_v = [r_v - 1]^+$. Clearly $\omega_v^I \leq \omega_v^{NI}$, since receiving external influence can only reduce the intervention required at v.

The algorithm proceeds bottom-up from the leaves to the root. At each node v, once all subtrees T_u for $u \in \mathcal{C}_v$ have been solved, the problem at v reduces to a star subproblem. Each child u becomes a leaf hypernode with cost $\Delta_u = \omega_u^{NI} - \omega_u^I$, representing the additional cost incurred when v does not provide influence to u. The total cost at v is the sum of the base costs $\sum_{u \in \mathcal{C}_v} \omega_u^I$, which are always incurred, plus the solution to the star subproblem at v:

$$\omega_v^{NI} = \sum_{u \in \mathcal{C}_v} \omega_u^I + \textsc{SolveStar}(r_v,\, c_v,\, \{\Delta_u\}_{u \in \mathcal{C}_v}) \tag{4.3}$$

$$\omega_v^I = \sum_{u \in \mathcal{C}_v} \omega_u^I + \textsc{SolveStar}([r_v - 1]^+,\, c_v,\, \{\Delta_u\}_{u \in \mathcal{C}_v}) \tag{4.4}$$

Theorem 6. *Let $(T_n, \{\theta_i\}, \{c_i h_i\})$ be an instance of the LTM intervention problem on a simple tree with linear cost functions. The problem can be solved in $O(n)$ time.*

Proof. See Günnec et al. [12], Theorem 5. The algorithm visits each node exactly once in a bottom-up traversal. The bottleneck at each node v is solving the star subproblem, which requires identifying the $\lceil r_v \rceil$ cheapest children among $\mathcal{C}_v$. By the Quick-select algorithm, this takes $O(\deg(v))$ time. Summing over all nodes,

$$\sum_{v \in V} \deg(v) = 2|E| = 2(n-1) = O(n)$$

which gives the claimed complexity. □

Algorithm 6 Dynamic Programming for Trees

Input: Tree T_n rooted at r, resistances $\{r_i\}$, cost coefficients $\{c_i\}$
Output: Minimum activation cost ω_r^{NI}
function TreeSolve(v)
    if v is a leaf **then**
        $\omega_v^{NI} \leftarrow c_v r_v$
        $\omega_v^{I} \leftarrow c_v[r_v - 1]^+$
        return $(\omega_v^{NI}, \omega_v^{I})$
    end if
    Let $\mathcal{C}_v$ be the set of children of v
    base $\leftarrow 0$
    for each $u \in \mathcal{C}_v$ **do**
        $(\omega_u^{NI}, \omega_u^{I}) \leftarrow$ TreeSolve(u)
        $\Delta_u \leftarrow \omega_u^{NI} - \omega_u^{I}$
        base $\leftarrow$ base $+ \omega_u^{I}$
    end for
    $\omega_v^{NI} \leftarrow$ base+ SolveStar(r_v, c_v, $\{\Delta_u\}$)
    $\omega_v^{I} \leftarrow$ base+ SolveStar($[r_v - 1]^+$, c_v, $\{\Delta_u\}$)
    return $(\omega_v^{NI}, \omega_v^{I})$
end function
return TreeSolve(r)[0]

4.1.3 General Trees

For general trees with arbitrary edge weights W_{ij} and general cost functions C_i, the LTM intervention problem is NP-hard, as shown by Günnec et al. [12] (Theorem 3). However, when the maximum degree of the tree is bounded by a constant d, a polynomial-time algorithm exists.

The recursive decomposition of Section 4.1.2 carries over unchanged to this setting. The only difference lies in the star subproblem: with general cost functions and arbitrary edge weights, the cost-efficient subset of children to activate can no longer be identified by a simple sorting argument. Instead, one must enumerate all $2^{\deg(v)}$ subsets of children of v to find the optimal activation set. When the maximum degree is bounded by a constant d, this enumeration costs $O(2^d)$ per node, which is $O(1)$ for fixed d.

Theorem 7. *Let $(T_n, \{\theta_i\}, \{C_i\})$ be an instance of the LTM intervention problem on a tree with maximum degree bounded by a constant d. The problem can be solved in $O(n \cdot 2^d)$ time.*

Proof. The algorithm follows the same bottom-up recursive decomposition as in

Theorem 6, replacing SolveStar with an exhaustive search over all $2^{\deg(v)}$ subsets of children of v. Each subset evaluation requires $O(1)$ time, so the star subproblem at node v costs $O(2^{\deg(v)})$. Summing over all nodes and using $\deg(v) \leq d$,

$$\sum_{v \in V} O(2^{\deg(v)}) \leq n \cdot O(2^d) = O(n \cdot 2^d)$$

For constant d, this reduces to $O(n)$. □

This result extends the applicability of the tree algorithm beyond the linear cost setting, and serves as a building block for the composite graph structures discussed in Section 4.3.

4.2 Paths and Cycles

With the same dynamic programming approach developed for trees, we specialize the algorithm for path graphs. More significantly, we use this as a foundation to design a linear-time algorithm for cycle graphs. An existing linear-time algorithm for paths with time constraints was proposed by Cordasco et al.[11], but it is restricted to simple graphs, i.e. unweighted edges, and unitary cost functions, i.e. $C_i(h_i) = h_i$. The algorithms we propose are fully general: they apply to arbitrary edge weights W_{ij} and arbitrary cost functions C_i.

We consider general LTM intervention problem instances $(L_n, \{\theta_i\}, \{C_i\})$ and $(C_n, \{\theta_i\}, \{C_i\})$ on path and cycle graphs with n nodes, respectively, without restrictions on edge weights or cost functions.

4.2.1 Path Graphs

The algorithm is based on solving subproblems defined on subpaths of L_n. We introduce two types of subproblems depending on whether the rightmost node receives external influence.

Definition 17. *Let $(L_n, \{\theta_i\}, \{C_i\})$ be an instance of the LTM intervention problem on a path graph. For $1 \leq m \leq n$, we define:*

1. *$L^{NI}_{1 \to m}$: the subproblem of activating all nodes on the subpath from node 1 to node m, where node m receives no influence from node $m+1$, equivalenty node $m+1$ receives influence from node m.*

2. *$L^{I}_{1 \to m}$: the subproblem of activating all nodes on the subpath from node 1 to node m, where node m receives influence from node $m+1$. In this case, the resistance of node m is reduced to $r'_m = [r_m - W_{m,m+1}]^+$.*

We denote by $\omega(L^{NI}_{1\to m})$ and $\omega(L^{I}_{1\to m})$ the minimum cost required to solve the respective subproblems.

The algorithm can be interpreted through acyclic orientations of the path graph. At each step, we decide the direction of influence propagation, which corresponds to orienting an edge. Transitioning from $L_{1\to m}$ to $L^{I}_{1\to m-1}$ is equivalent to orienting edge $\{m-1, m\}$ as $(m, m-1)$, with node m influencing node $m-1$. Conversely, transitioning to $L^{NI}_{1\to m-1}$ corresponds to orienting the edge as $(m-1, m)$, with node $m-1$ influencing node m.

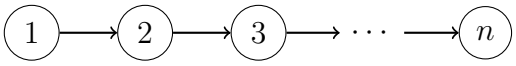


Figure 4.2: An acyclic orientation on a path.

Recursive formulation. For the subproblem $L^{NI}_{1\to m}$, we distinguish two cases based on whether the influence from node $m-1$ is sufficient to activate node m. If $W_{m,m-1} \geq r_m$, one may either let node m activate by propagation after solving $L^{NI}_{1\to m-1}$, at cost $\omega(L^{NI}_{1\to m-1})$, or activate node m directly and solve $L^{I}_{1\to m-1}$, at cost $\omega(L^{I}_{1\to m-1}) + C_m(r_m)$. If instead $W_{m,m-1} < r_m$, one may either provide the partial incentive $r_m - W_{m,m-1}$ after solving $L^{NI}_{1\to m-1}$, at cost $\omega(L^{NI}_{1\to m-1}) + C_m(r_m - W_{m,m-1})$, or activate node m fully after solving $L^{I}_{1\to m-1}$, at cost $\omega(L^{I}_{1\to m-1}) + C_m(r_m)$. Formally:

$$\omega(L^{NI}_{1\to m}) = \begin{cases} \min\{\omega(L^{NI}_{1\to m-1}), \omega(L^{I}_{1\to m-1}) + C_m(r_m)\} & \text{if } W_{m,m-1} \geq r_m \\ \min\{\omega(L^{NI}_{1\to m-1}) + C_m(r_m - W_{m,m-1}), \\ \qquad \omega(L^{I}_{1\to m-1}) + C_m(r_m)\} & \text{if } W_{m,m-1} < r_m \end{cases} \tag{4.5}$$

For the subproblem $L^{I}_{1\to m}$, we replace r_m with the reduced resistance $r'_m = [r_m - W_{m,m+1}]^+$ and apply analogous reasoning. Since we have assumed $r_i \leq w_i = W_{i,i+1} + W_{i,i-1}$, the case $W_{m,m-1} < r'_m$ cannot occur, and the recurrence simplifies to:

$$\omega(L^{I}_{1\to m}) = \begin{cases} \omega(L^{I}_{1\to m-1}) & \text{if } r'_m \leq 0 \\ \min\{\omega(L^{NI}_{1\to m-1}),\ \omega(L^{I}_{1\to m-1}) + C_m(r'_m)\} & \text{if } W_{m,m-1} \geq r'_m > 0 \end{cases} \tag{4.6}$$

Base cases.

$$\omega(L^{NI}_{1\to 1}) = C_1(r_1), \qquad \omega(L^{I}_{1\to 1}) = C_1([r_1 - W_{1,2}]^+).$$

The original problem is to compute $\omega(L^{NI}_{1\to n})$.

Algorithm 7 Dynamic Programming for Path Graphs

Input: Path graph L_n, resistances $\{r_i\}$, cost functions $\{C_i\}$
Output: Minimum activation cost $\omega(L^{NI}_{1\to n})$
function PathSolve(m, influence)
    if $m = 1$ **then**
        if influence = false **then**
            return $C_1(r_1)$
        else
            return $C_1([r_1 - W_{1,2}]^+)$
        end if
    end if
    if influence = false **then**
        Compute $\omega(L^{NI}_{1\to m})$ using (4.5)
        return $\omega(L^{NI}_{1\to m})$
    else
        Compute $\omega(L^{I}_{1\to m})$ using (4.6)
        return $\omega(L^{I}_{1\to m})$
    end if
end function
return PathSolve(n, false)

Proposition 8. *Let $(L_n, \{\theta_i\}, \{C_i\})$ be an instance of the LTM intervention problem on a path graph. The problem can be solved exactly in $O(n)$ time.*

Proof. With memoization, the algorithm solves $2n$ subproblems (two for each node), each requiring $O(1)$ operations given the solutions to previous subproblems. Therefore, the total time complexity is $O(n)$. □

4.2.2 Cycle Graphs

Recall that relevant feasible interventions correspond to acyclic orientations of the graph. On a path, every orientation is acyclic, but on a cycle C_n, an acyclic orientation must have at least one node that acts as a source (all edges oriented outward) and one as a sink (all edges oriented inward), thereby breaking the cycle.

A straightforward algorithm iterates over all nodes, designates each as a sink (or source), orients its adjacent edges accordingly, and solves the resulting path subproblem. Since there are n nodes and each path subproblem requires $O(n)$ time, the total complexity is $O(n^2)$.

However, we can achieve $O(n)$ complexity by selecting a node i and considering all possible orientations of its adjacent edges, while ensuring that the resulting

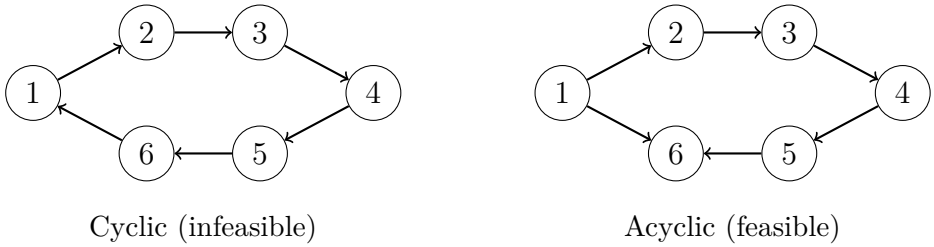


Figure 4.3: Cyclic versus acyclic orientations on a 6-node cycle.

orientation is acyclic. To avoid cyclic orientations, we modify the path algorithm to assign infinite cost to orientations that would create a directed cycle. Specifically, if the path from $i+1$ to $i-1$ forms a single directed path without any reversal, the orientation would be cyclic and is therefore forbidden.

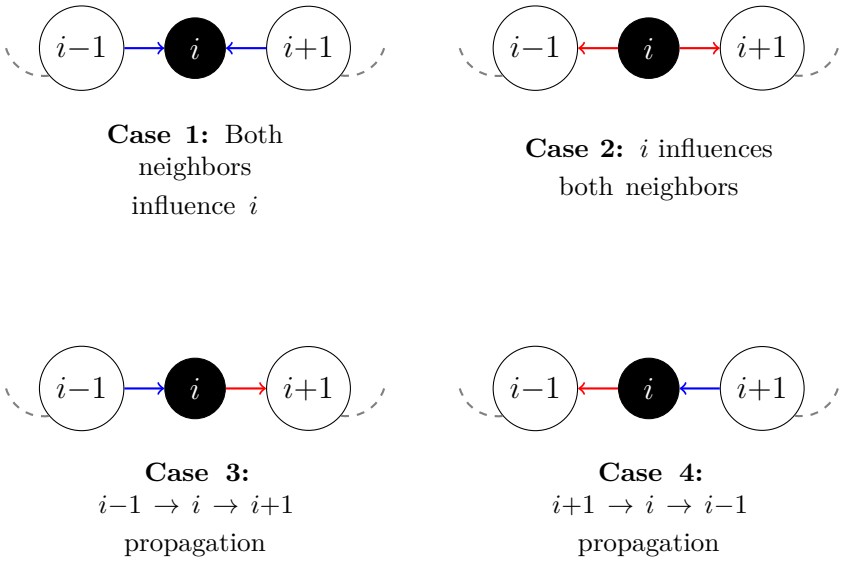


Figure 4.4: Possible orientations of edges adjacent to node i in a cycle.

By considering all possible edge orientations around a fixed node i and solving the remaining path for each configuration, we capture all feasible acyclic orientations of the cycle. Therefore, the choice of node i is not relevant. We now analyze each of the four cases illustrated in Figure 4.4, determining the intervention cost at node i, how influence propagates to its neighbors, and whether there are restrictions on the orientation of the remaining edges.

Case 1. Node i receives influence from both neighbors $i-1$ and $i+1$. Assuming the resistance of i is sufficiently low, no intervention is required at node i. The

remaining problem is a path from $i+1$ to $i-1$ without external influence from i, which can be solved in linear time using the path algorithm. Every solution to this path subproblem is feasible when combined with node i acting as a sink, as the resulting orientation is acyclic.

Case 2. Node i is fully activated and influences both neighbors. This requires an intervention cost of $C_i(r_i)$ at node i. Both adjacent nodes receive influence from i, reducing their effective resistances to $r'_{i-1} = [r_{i-1} - W_{i-1,i}]^+$ and $r'_{i+1} = [r_{i+1} - W_{i+1,i}]^+$. The remaining problem is a path from $i-1$ to $i+1$, where both endpoints receive external influence. Every solution is feasible when combined with node i acting as a source, as the resulting orientation is acyclic.

Case 3. Node i receives influence from node $i-1$ and provides influence to node $i+1$. The intervention cost at node i is $C_i([r_i - W_{i,i-1}]^+)$ if the influence from $i-1$ is insufficient. Node $i+1$ receives influence from i, so its effective resistance becomes $r'_{i+1} = [r_{i+1} - W_{i+1,i}]^+$. The remaining problem is a path from $i+1$ to $i-1$. The total cost is the intervention cost at node i plus the cost of the reduced path problem. However, we cannot use the standard path algorithm directly, because there exists an orientation of this path that forms a directed path from $i+1$ to $i-1$, which when combined with the edges at i creates a directed cycle. Such an orientation does not correspond to a feasible intervention. Therefore, we must modify the path algorithm to assign infinite cost to orientations that would create such a cycle.

Case 4. Case 4 is symmetric to Case 3, with node i receiving influence from node $i+1$ and providing influence to node $i-1$. The same considerations apply regarding the modification of the path algorithm to avoid cyclic orientations.

To handle Cases 3 and 4, we modify the standard path algorithm by introducing a binary flag that tracks whether at least one edge has been oriented in reverse (i.e., from a higher-indexed node to a lower-indexed node). This flag ensures that we avoid orientations that would form a directed cycle from 1 to n. Once a reverse orientation occurs, the flag is set and all remaining nodes can be solved using the standard recurrence.

Forbidden (flag = 0) Allowed (flag = 1)

Figure 4.5: Path orientations without and with reverse edges.

Consider a path from node 1 to node m. We augment the subproblems as follows:

Definition 18. *For a path subproblem, we define:*

- $L^{I,0}_{1\to m}$*: the subproblem where no edge has yet been oriented in reverse (i.e., no edge* $(k, k-1)$ *with* $k > 1$ *has been selected). This represents a partial solution that is still at risk of forming a directed path from* n *to* 1*.*
- $L^{I,1}_{1\to m}$*: the subproblem where at least one edge has been oriented in reverse, ensuring that the resulting path orientation is not a single directed path and thus avoids creating a cycle.*

Similarly, we define $L^{NI,0}_{1\to m}$ *and* $L^{NI,1}_{1\to m}$ *for subproblems without external influence.*

The recurrence relations are equal to those of the basic path algorithm, with the modification that when an edge is oriented in reverse (from node k to node $k-1$), the flag transitions from 0 to 1. Orienting an edge in reverse corresponds to transitioning to an influenced subproblem $L^{I}_{1\to m-1}$, as the lower-indexed node receives influence from the higher-indexed node. To enforce that at least one reversal must occur, we set $\omega(L^{I,0}_{1\to 1}) = \infty$ and $\omega(L^{NI,0}_{1\to 1}) = \infty$ as a base case, while $\omega(L^{I,1}_{1\to 1})$ and $\omega(L^{NI,1}_{1\to 1})$ take usual values.

For Case 3, where node i receives influence from $i-1$ and provides influence to $i+1$, we solve the augmented path subproblem from $i+1$ to $i-1$ using the modified algorithm with the flag. The flag ensures that at least one edge is oriented from a higher-indexed node to a lower-indexed node, preventing a directed path from $i+1$ to $i-1$ that would close the cycle. Similarly, for Case 4, we solve the path from $i-1$ to $i+1$ with the flag ensuring that at least one edge is oriented backward.

The modified algorithm requires $O(n)$ time. There are $4n$ augmented subproblems (two types of external influence $\{I, NI\}$ times two flag values $\{0, 1\}$ times n nodes), each solvable in $O(1)$ time given the solutions to previous subproblems. The total cost is computed as the sum of the intervention cost at node i and the minimum cost of the reduced path problem.

Corollary 7. *Let* $(C_n, \{\theta_i\}, \{C_i\})$ *be an instance of the LTM intervention problem on a cycle graph. The problem can be solved exactly in* $O(n)$ *time.*

Proof. The algorithm fixes an arbitrary node i and partitions all acyclic orientations into four cases based on the orientations of edges adjacent to i. Cases 1 and 2 use the standard path algorithm on the remaining $n-1$ nodes, each requiring $O(n)$ time with memoization. Cases 3 and 4 use the augmented path algorithm with $4n$ subproblems, also requiring $O(n)$ time total. Since we evaluate four cases, each requiring $O(n)$ time, the total complexity is $O(n)$. □

Algorithm 8 Dynamic Programming for Cycle Graphs

Input: Cycle graph C_n, resistances $\{r_i\}$, cost functions $\{C_i\}$
Output: Minimum cost for complete activation
function CycleSolve
    Fix an arbitrary node i (e.g., $i = 1$)
    // Case 1: Both neighbors influence node i
    $c_1 \leftarrow$ PathSolve$(i + 1, i - 1, \text{false})$
    // Case 2: Node i influences both neighbors
    $c_2 \leftarrow C_i(r_i)+$ PathSolve$(i - 1, i + 1, \text{true})$
    // Case 3: Propagation from $i - 1$ through i to $i + 1$
    $c_3 \leftarrow C_i([r_i - W_{i,i-1}]^+)+$ AugmentedPathSolve$(i + 1, i - 1)$
    // Case 4: Propagation from $i + 1$ through i to $i - 1$
    $c_4 \leftarrow C_i([r_i - W_{i,i+1}]^+)+$ AugmentedPathSolve$(i - 1, i + 1)$
    return $\min(c_1, c_2, c_3, c_4)$
end function
return CycleSolve

4.3 Composite Graph Structures

The algorithms developed for trees, paths, and cycles can be combined to handle more complex graph topologies through a systematic decomposition approach. The key idea is to identify subgraphs that can be solved independently, compress each into a hypernode encoding its optimal activation costs under all relevant influence profiles, and then apply the cycle or path algorithm to the contracted graph. We illustrate this approach on two families of graphs and conclude with open questions.

Throughout this section, given a subgraph H adjacent to at most two external nodes, we define its *influence profile* as the tuple $(\omega^{LL}, \omega^{LR}, \omega^{RL}, \omega^{RR})$, where each entry denotes the minimum activation cost of H under no external influence, influence from the left only, influence from the right only, and influence from both sides, respectively.

Proposition 9. *Let $(G, \{\theta_i\}, \{C_i\})$ be an instance of the LTM intervention problem on a unicyclic graph where each node on the cycle is the root of a tree with maximum degree bounded by a constant d. The problem can be solved in $O(n)$ time.*

Proof. For each node i on the cycle, let T_i denote the subtree rooted at i. Since T_i has maximum degree d, its influence profile can be computed in $O(|T_i|)$ time by running the bounded-degree tree algorithm of Theorem 7 under each of the four influence configurations at i. Each subgraph T_i is then replaced by a hypernode on the cycle, with activation costs given by its influence profile. The cycle algorithm of Corollary 7 is applied to the contracted cycle of k hypernodes, where $k \leq n$.

The total complexity is

$$\sum_{i=1}^{k} O(|T_i|) + O(k) = O(n)$$

since $\sum_{i=1}^{k} |T_i| = n$ and $k \leq n$. □

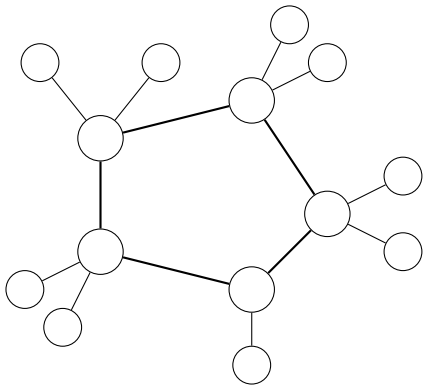


Figure 4.6: A unicyclic graph where each cycle node is the root of a tree with maximum degree $d = 2$.

Proposition 10. *Let $(G, \{\theta_i\}, \{C_i\})$ be an instance of the LTM intervention problem on a figure-eight graph consisting of two cycles C_{left} and C_{right} sharing a central node i. The problem can be solved in $O(n)$ time.*

Proof. Let $n_{\text{left}} = |C_{\text{left}}|$ and $n_{\text{right}} = |C_{\text{right}}|$, so that $n_{\text{left}} + n_{\text{right}} - 1 = n$. We compute the influence profile of C_{left} (including node i) under the four possible influence configurations that i may receive from C_{right}, using the cycle algorithm of Corollary 7. This requires $O(n_{\text{left}})$ time. The left cycle is then replaced by a hypernode on C_{right}, and the cycle algorithm is applied to the resulting cycle of n_{right} nodes in $O(n_{\text{right}})$ time. The total complexity is $O(n_{\text{left}}) + O(n_{\text{right}}) = O(n)$. □

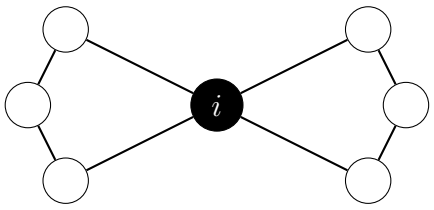


Figure 4.7: A figure-eight graph consisting of two cycles sharing a central node i.

Remark 5. *More generally, the decomposition approach extends to any graph that can be recursively contracted into a single cycle or path by replacing subgraphs with hypernodes, provided that each subgraph can be solved in polynomial time and has at most two attachment points to the rest of the graph. This suggests that graphs formed by compositions of cycles and bounded-degree trees admit polynomial-time algorithms for the LTM intervention problem. Characterizing the complete class of graphs tractable under this decomposition remains an open question. A concrete open problem is whether the grid graph is solvable in polynomial time, and under what structural conditions.*

Chapter 5

Local Search and Simulated Annealing

In this chapter, we address the LTM intervention problem on general graphs, where, as established in Chapter 1, no polynomial-time algorithm with provable guarantees is known. For arbitrary graphs, any practical approach must accept a trade-off between computational efficiency and solution quality.

Several methods have been proposed in the literature to cope with this hardness. On the exact side, Fischetti et al. [15] and Günnec et al. [14] developed branch-and-cut approaches that perform well in practice on sparse graphs, but retain exponential worst-case complexity. On the heuristic side, Cordasco et al. [11] introduced a greedy algorithm for simple graphs with a provable upper bound on the solution cost, and proposed several refinements based on degree-based node selection. Moreover, Günnec et al. [12] employ a greedy algorithm, exact for trees, as a heuristic for the LCIP on general graphs, under the assumption of uniform edge weights and linear cost functions. These methods, however, are either restricted to specific cost structures or do not scale to large and dense instances. A detailed experimental comparison is deferred to Chapter 6.

We exploit the combinatorial reformulation of the intervention problem over the symmetric group $\mathcal{S}_n$, recalled in Chapter 3, which recasts the problem as $\min_{\sigma \in \mathcal{S}_n} C(\sigma)$ over a discrete space equipped with a natural neighborhood structure. This perspective suggests Local Search as a baseline strategy, operating on the transposition meta-graph of $\mathcal{S}_n$. We then refine this into Simulated Annealing by allowing controlled moves to worse solutions via the Metropolis–Hastings acceptance criterion, enabling escape from local optima.

Our contributions are twofold. First, we formalize a Simulated Annealing algorithm via the Metropolis–Hastings chain on the transposition meta-graph of $\mathcal{S}_n$, designed to handle the full generality of the LCIP: arbitrary graph topology,

arbitrary edge weights, and arbitrary non-decreasing cost functions. This method is capable of addressing this level of generality without restrictions on the cost structure. Second, for the LCIP with general costs and arbitrary edge weights, we prove convergence to the global optimum and derive an explicit upper bound on the mixing time, by combining the Diaconis–Shahshahani spectral theorem for the random transposition shuffle [19] with the comparison theorem for reversible Markov chains.

5.1 Local Search

Recall from Chapter 3 that the LTM intervention problem is equivalent to the optimization problem over the permutation space $\mathcal{S}_n$:

$$\min_{\sigma \in \mathcal{S}_n} C(\sigma) = \min_{\sigma \in \mathcal{S}_n} \sum_{i \in V} C_i(h_i(\sigma))$$

where $\sigma(i)$ represents the position of node i in the activation sequence and $h_i(\sigma) = [r_i - w_i(\sigma)]^+$ is the minimum intervention required to ensure the activation of node i at position $\sigma(i)$. The combinatorial structure of $\mathcal{S}_n$ naturally suggests equipping the solution space with a neighborhood structure, formalized through the notion of a meta-graph.

Definition 19. *Given an optimization problem* $\min_{x \in \mathcal{X}} f(x)$*, a* meta-graph *is a strongly connected graph* $G = (\mathcal{X}, E)$ *whose vertex set coincides with the solution space.*

Given a meta-graph, Local Search (LS) is one of the most widely used meta-heuristics. Starting from an initial candidate solution x_0, the algorithm iteratively moves to the best neighboring solution until no further improvement is possible, as formalized in Algorithm 9.

For the algorithm to be practical, the meta-graph must satisfy two fundamental requirements:

- **Connectivity.** The meta-graph must be strongly connected, ensuring that any optimal solution is reachable from any starting point. Without this property, the algorithm may be confined to a suboptimal region of the solution space regardless of initialization.
- **Efficiency.** Two computational costs must be kept low at each iteration: the cost of *generating* a neighboring solution, and the cost of *evaluating* the objective function at the new solution. The latter is best handled by maintaining an incremental representation of f, updating its value from the current solution rather than recomputing it from scratch. If either cost is

Algorithm 9 Local Search

Input: Objective function $f : \mathcal{X} \to \mathbb{R}$, neighborhood structure $G = (\mathcal{X}, E)$, initial solution $x_0 \in \mathcal{X}$
Output: Local optimum x^*
$x \leftarrow x_0$
while improvement found **do**
    $x' \leftarrow \arg\min_{y \in \mathcal{N}(x)} f(y)$
    if $f(x') < f(x)$ **then**
        $x \leftarrow x'$
    else
        break
    end if
end while
return x

significant, the metaheuristic becomes impractical even for moderate problem sizes.

In the following section, we propose a transposition-based meta-graph on $\mathcal{S}_n$ that satisfies both requirements, and for which sharp theoretical results on convergence rates are available.

Despite its simplicity, Local Search has two well-known limitations: it can become trapped in local optima, and it provides no guarantees on the quality of the solution relative to the global optimum. In this thesis, we address both issues through *Simulated Annealing* (SA), a memory-less stochastic extension of Local Search. The key idea is to allow moves to worse solutions with a controlled, non-zero probability, enabling escape from local optima. This is implemented via the Metropolis-Hastings algorithm, which we introduce in Section 5.2.

5.2 Simulated Annealing

The Metropolis-Hastings algorithm is a classical Markov Chain Monte Carlo (MCMC) method originally designed to sample from a target probability distribution. The key idea is to construct a Markov chain whose stationary distribution coincides with the desired target, then simulate the chain until it reaches stationarity.

This idea adapts naturally to optimization: it suffices to choose a target distribution that concentrates its mass on optimal solutions. For a minimization problem with cost function $C(\sigma)$, the classical choice is the Boltzmann distribution

$$\pi_\beta(\sigma) \propto \exp\left(-\beta C(\sigma)\right)$$

where $\beta > 0$ is the inverse temperature parameter. As $\beta \to \infty$, the distribution concentrates on minimum-cost solutions. Given the meta-graph structure on $\mathcal{S}_n$, one can construct a reversible Markov chain admitting π_β as its stationary distribution, which is both theoretically tractable and easy to simulate. Crucially, the Markov chain formalism is not merely decorative: it enables explicit guarantees on the probabilistic quality of the solution via ergodic theory.

The Metropolis-Hastings algorithm is the core of Simulated Annealing (SA), which augments it with a *cooling schedule* that progressively increases β during execution. Starting from a low value of β (high temperature), the algorithm gradually raises β over time, balancing exploration and exploitation: at high temperature, worse solutions are accepted with non-negligible probability, enabling escape from local optima; as the temperature decreases, the algorithm becomes increasingly selective and concentrates on the most promising regions of the solution space.

In the following sections, we first define the transposition-based meta-graph and the associated sampling chain, then introduce the Metropolis-Hastings dynamics and analyze its mixing time.

5.2.1 Transposition Meta-graph

A natural neighborhood structure on $\mathcal{S}_n$ is provided by *transpositions*. A transposition (i, j) with $1 \leq i, j \leq n$ is a permutation that swaps the values at positions i and j while leaving all other positions unchanged, i.e.,

$$\tau(i) = j, \quad \tau(j) = i, \quad \tau(\ell) = \ell \quad \text{for all } \ell \neq i, j$$

We also regard (i, i) as a transposition for every i, corresponding to the identity permutation; this convention yields n identity transpositions and is adopted for notational clarity. We denote by $\mathcal{T}$ the set of all pairs on $\{1, \ldots, n\}$:

$$\mathcal{T} = \{(i, j) : 1 \leq i, j \leq n\}, \qquad |\mathcal{T}| = n^2$$

Given a permutation $\sigma \in \mathcal{S}_n$, a neighboring permutation is obtained by right-composing σ with a transposition $(i, j) \in \mathcal{T}$:

$$\sigma' = \sigma \cdot (i, j)$$

Example 7. *Consider* $n = 10$ *and the permutation* σ *defined by*

i	1	2	3	4	5	6	7	8	9	10
$\sigma(i)$	3	7	1	5	9	2	8	4	10	6

Applying the transposition (2,5) *swaps the values at positions* 2 *and* 5*, yielding*

$\sigma' = \sigma \cdot (2{,}5)$:

i	1	2	3	4	5	6	7	8	9	10
$\sigma'(i)$	3	**9**	1	5	**7**	2	8	4	10	6

where the entries at positions 2 *and* 5 *have been exchanged* ($\sigma'(2) = 9,\ \sigma'(5) = 7$)*, while all other values remain unchanged.*

By Cayley's theorem, every permutation can be expressed as a composition of transpositions, which guarantees that the resulting meta-graph on $\mathcal{S}_n$ is strongly connected: any permutation is reachable from any other in a finite number of transposition steps. Each permutation σ has exactly $\binom{n}{2}$ distinct neighbors, corresponding to nontrivial transpositions (i, j) with $i \neq j$, together with a self-loop induced by the n identity transpositions (i, i).

Other neighborhood structures on $\mathcal{S}_n$ are of course possible. For instance, one could define a neighborhood by selecting a node i and moving it to the last position, shifting all other elements accordingly. Alternatively, given a strongly connected graph $G = (V, E)$, one could restrict transpositions to edges of G, defining

$$\mathcal{T}_G = \{(i, j) : (i, j) \in E\}$$

Our choice of using the full set $\mathcal{T}$ is motivated by three considerations. First, applying a transposition to a permutation is computationally inexpensive, as it requires swapping only two values. Second, the change in the objective function $C(\sigma)$ induced by a transposition can be computed incrementally from the current value, without recomputing the cost from scratch, satisfying the efficiency requirement discussed in Section 5.1. Third, sharp spectral results are available for the associated Markov chain, which will allow us to characterize the convergence rate of the algorithm precisely in Section 5.3.

5.2.2 Random Shuffling Chain

The transposition meta-graph introduced in the previous section naturally gives rise to a Markov chain on $\mathcal{S}_n$, known as the *random transposition shuffle* [20]. Let $(X_t)_{t\in\mathbb{N}}$ be a Markov chain with state space $\mathcal{S}_n$. At each time step t, a transposition (i, j) is sampled uniformly at random from $\mathcal{T}$, independently of the current state $X_t = \sigma$, and the chain moves to

$$X_{t+1} = X_t \cdot (i, j)$$

The transition probabilities are given by

$$P(\sigma, \sigma') = \begin{cases} \dfrac{1}{n} & \text{if } \sigma' = \sigma \\ \dfrac{2}{n^2} & \text{if } \sigma' = \sigma \cdot (i,j) \text{ for some } 1 \le i < j \le n \\ 0 & \text{otherwise} \end{cases} \tag{5.1}$$

where the probability $\frac{1}{n}$ accounts for the n identity transpositions (i,i) that leave σ unchanged, and $\frac{2}{n^2}$ accounts for each nontrivial transposition, which can be sampled as either (i,j) or (j,i). The chain is irreducible by strong connectivity of the meta-graph, and aperiodic due to the presence of self-loops.

Theorem 8 (Ergodic Theorem for Random Shuffling)**.** *The random transposition shuffle is irreducible, aperiodic, and reversible. Its unique stationary distribution is the uniform distribution on $\mathcal{S}_n$:*

$$\pi(\sigma) = \frac{1}{n!} \qquad \textit{for all } \sigma \in \mathcal{S}_n$$

Furthermore, for any initial distribution, the chain converges to π as $t \to \infty$:

$$\lim_{t\to\infty} \mathbb{P}(X_t = \sigma) = \frac{1}{n!} \qquad \textit{for all } \sigma \in \mathcal{S}_n$$

Proof. This is a standard result in Markov chain theory; see [20] for a detailed proof. □

This chain is called a *sampling chain* because, at stationarity, it generates permutations uniformly at random over $\mathcal{S}_n$. As such, it does not directly serve as an optimization algorithm. However, it provides the base structure upon which the Metropolis-Hastings dynamics is built in the next section, by biasing the stationary distribution toward low-cost permutations.

5.2.3 Metropolis–Hastings Algorithm

Starting from the random shuffling chain $(X_t)_{t\in\mathbb{N}}$, we define a parametrized family of Markov chains $(Z_t^{(\beta)})_{t\in\mathbb{N}}$ on $\mathcal{S}_n$, indexed by the inverse temperature parameter $\beta > 0$. The chain incorporates the optimization objective by equipping the base chain with an accept/reject mechanism based on the change in the cost function between the current state and the proposed one.

The chain is designed so that its stationary distribution is the *Boltzmann measure*

$$\pi_\beta(\sigma) = \frac{1}{Z_\beta} \exp\Big(-\beta C(\sigma)\Big) \tag{5.2}$$

where

$$Z_\beta = \sum_{\sigma \in \mathcal{S}_n} \exp\Big(-\beta C(\sigma)\Big) \tag{5.3}$$

is the partition function. The Boltzmann measure assigns higher probability to permutations with lower cost, with the degree of concentration controlled by β.

Given the current state σ, a candidate σ' is proposed by sampling a transposition (i, j) uniformly from $\mathcal{T}$, as in the base chain. The candidate is then accepted or rejected according to the following rule:

- if $C(\sigma') < C(\sigma)$, the move is accepted with probability 1;
- if $C(\sigma') \geq C(\sigma)$, the move is accepted with probability

$$\alpha(\sigma, \sigma') = \exp\Big(-\beta \left(C(\sigma') - C(\sigma)\right)\Big)$$

Formally, for $\sigma \neq \sigma'$, the transition probabilities are

$$P_\beta(\sigma, \sigma') = \begin{cases} \dfrac{2}{n^2} & \text{if } \sigma' = \sigma \cdot (i,j) \text{ and } C(\sigma') < C(\sigma) \\ \dfrac{2}{n^2} \exp\Big(-\beta \left(C(\sigma') - C(\sigma)\right)\Big) & \text{if } \sigma' = \sigma \cdot (i,j) \text{ and } C(\sigma') \geq C(\sigma) \\ 0 & \text{otherwise} \end{cases} \tag{5.4}$$

while the probability of remaining in σ is determined by the normalization constraint $\sum_{\sigma'} P_\beta(\sigma, \sigma') = 1$.

Theorem 9 (Ergodic Theorem for Metropolis–Hastings)**.** *For any $\beta > 0$, the Metropolis–Hastings chain $(Z_t^{(\beta)})_{t \in \mathbb{N}}$ is irreducible, aperiodic, and reversible with respect to the Boltzmann measure π_β defined in* (5.2)*. Consequently, π_β is its unique stationary distribution, and for any initial distribution the chain converges to π_β as $t \to \infty$:*

$$\lim_{t \to \infty} \mathbb{P}\Big(Z_t^{(\beta)} = \sigma\Big) = \pi_\beta(\sigma) \qquad \textit{for all } \sigma \in \mathcal{S}_n$$

Moreover, as $\beta \to +\infty$, the Boltzmann measure concentrates on the set of global minimizers of C:

$$\lim_{\beta \to \infty} \pi_\beta(\sigma) = \begin{cases} \dfrac{1}{|\mathcal{S}^*|} & \textit{if } \sigma \in \mathcal{S}^* = \arg\min\limits_{\sigma' \in \mathcal{S}_n} C(\sigma') \\ 0 & \textit{otherwise} \end{cases}$$

Proof. See [21]. □

Algorithm 10 Metropolis–Hastings for LTM Intervention

Input: Initial permutation $\sigma_0 \in \mathcal{S}_n$, inverse temperature $\beta > 0$, number of iterations T
Output: Permutation σ_T
$\sigma \leftarrow \sigma_0$
for $t = 1$ **to** T **do**
    Sample (i, j) uniformly at random from $\mathcal{T}$
    $\sigma' \leftarrow \sigma \cdot (i, j)$
    $\Delta C \leftarrow C(\sigma') - C(\sigma)$
    if $\Delta C < 0$ **then**
        $\sigma \leftarrow \sigma'$
    else
        Sample $u \sim \text{Uniform}(0,1)$
        if $u < \exp(-\beta\, \Delta C)$ **then**
            $\sigma \leftarrow \sigma'$
        end if
    end if
end for
return σ

Theorem 9 guarantees that, for sufficiently large β, the chain visits optimal solutions with high probability. In practice, however, choosing β large from the outset slows convergence, as the chain becomes nearly deterministic and exploration is suppressed. Simulated Annealing addresses this by adopting a *cooling schedule* that progressively increases β during execution, starting with broad exploration at low β and gradually concentrating on optimal solutions as β grows. The design of the cooling schedule and the empirical performance of the algorithm are discussed in Chapter 6.

5.3 Mixing Time

Theorem 9 guarantees that the Metropolis–Hastings chain converges to the Boltzmann measure π_β as $t \to \infty$, and that π_β concentrates on optimal solutions for large β. However, convergence in the limit is not sufficient from an algorithmic perspective: we need the chain to reach a distribution close to π_β in a *reasonable* number of steps, ideally polynomial in the problem size n. The *mixing time* formalizes this requirement, quantifying how many iterations are needed for the chain to be within a prescribed distance from its stationary distribution, regardless of the starting point.

We begin by introducing the standard notion of distance between probability distributions used in this context.

Definition 20. *Let μ and ν be two probability measures on a finite set S. The* total variation distance *between μ and ν is defined as*

$$\|\mu - \nu\|_{\mathrm{TV}} = \sup_{A \subseteq S} \big|\mu(A) - \nu(A)\big|$$

The total variation distance measures the maximum discrepancy between the probabilities assigned by two distributions to the same event, providing a natural notion of closeness between probability measures. It satisfies $0 \leq \|\mu - \nu\|_{\mathrm{TV}} \leq 1$ for all μ, ν, and is convex in the sense that for any $\lambda \in [0,1]$,

$$\|\lambda\mu_1 + (1-\lambda)\mu_2 - \nu\|_{\mathrm{TV}} \leq \lambda\|\mu_1 - \nu\|_{\mathrm{TV}} + (1-\lambda)\|\mu_2 - \nu\|_{\mathrm{TV}}$$

To measure convergence uniformly over all possible starting points, we introduce the worst-case distance function.

Definition 21. *Let P be an irreducible and aperiodic transition matrix on a finite state space S with stationary distribution π. The* distance function *is defined for $t \geq 0$ as*

$$d(t) = \max_{x \in S} \|P^t(x, \cdot) - \pi\|_{\mathrm{TV}}$$

The distance function $d(t)$ provides a uniform upper bound on the convergence to stationarity over all initial distributions. Indeed, any probability measure μ on S can be written as $\mu = \sum_{x \in S} \mu(x)\, \delta_x$, and by convexity of the total variation distance,

$$\|\mu P^t - \pi\|_{\mathrm{TV}} \leq \sum_{x \in S} \mu(x)\, \|P^t(x, \cdot) - \pi\|_{\mathrm{TV}} \leq d(t)$$

Hence, once $d(t)$ is small, the chain is close to stationarity regardless of where it started.

Definition 22. *Let P be an irreducible and aperiodic transition matrix on a finite state space S. For any $0 < \varepsilon < 1$, the* mixing time *at level ε is defined as*

$$t_{\mathrm{mix}}(\varepsilon) = \inf\Big\{t \geq 0 : d(t) \leq \varepsilon\Big\}$$

The mixing time thus quantifies the number of steps required for the chain to be within total variation distance ε of its stationary distribution, uniformly over all initial states. Characterizing $t_{\mathrm{mix}}(\varepsilon)$ for the Metropolis–Hastings chain is the main objective of the following sections.

5.3.1 Spectral Analysis

A classical approach to bounding the mixing time of a reversible Markov chain is via spectral analysis of its transition matrix. The key quantities are the spectral gap and the associated relaxation time.

Definition 23. *Let P be an irreducible, aperiodic, and reversible transition matrix on a finite state space S with eigenvalues $1 = \lambda_1 > \lambda_2 \geq \cdots \geq \lambda_{|S|} > -1$. Define $\lambda^* = \max\{\lambda_2, |\lambda_{|S|}|\}$. The* absolute spectral gap *is $\gamma^* = 1 - \lambda^*$ and the* relaxation time *is*

$$t_{\mathrm{rel}} = \frac{1}{\gamma^*}$$

The relaxation time controls the mixing time through the following classical bound [20].

Theorem 10. *Let P be an irreducible, aperiodic, and reversible transition matrix with stationary distribution π, and let $0 < \varepsilon < 1$. Then*

$$(t_{\mathrm{rel}} - 1)\,\log\!\left(\frac{1}{2\varepsilon}\right) \;\leq\; t_{\mathrm{mix}}(\varepsilon) \;\leq\; t_{\mathrm{rel}}\,\log\!\left(\frac{1}{2\varepsilon\sqrt{\pi_{\min}}}\right)$$

where $\pi_{\min} = \min_{x \in S} \pi(x)$.

To apply this bound to the Metropolis–Hastings chain, we need the relaxation time of the base random transposition shuffle, which is given by the following result due to Diaconis and Shahshahani [19]; see also [22, Section 9.2].

Theorem 11 (Diaconis–Shahshahani [19])**.** *For the random transposition shuffle on $\mathcal{S}_n$, the second-largest and smallest eigenvalues of the transition matrix are*

$$\lambda_{\max} = 1 - \frac{2}{n}, \qquad \lambda_{\min} = \frac{2}{n} - 1$$

Consequently, $\lambda^* = \max\{\lambda_{\max}, |\lambda_{\min}|\} = 1 - \frac{2}{n}$, $\gamma^* = \frac{2}{n}$, and

$$t_{\mathrm{rel}} = \frac{n}{2}$$

5.3.2 Comparison Theorem

Theorem 11 characterizes the spectral gap of the base chain, but our goal is to bound the mixing time of the Metropolis–Hastings chain, which has a different stationary distribution. The *comparison theorem* provides a tool to transfer spectral bounds from one chain to another on the same state space.

For a reversible Markov chain with transition matrix P and stationary distribution π, define the *edge stationary measure*

$$Q(x,y) = \pi(x)\,P(x,y) \qquad x,y \in S$$

By reversibility, $Q(x,y) = Q(y,x)$, so $Q(x,y)$ represents the stationary probability flux along the edge (x,y).

Theorem 12 (Comparison Theorem [23])**.** *Let P and $\hat{P}$ be the transition matrices of two irreducible, aperiodic, and reversible Markov chains on the same finite state space S, with stationary distributions π and $\hat{\pi}$ and edge measures Q and $\hat{Q}$, respectively. Suppose there exist constants $\alpha, \gamma > 0$ such that for all $x, y \in S$,*

$$\hat{Q}(x,y) \le \alpha\, Q(x,y) \qquad \text{and} \qquad \pi(x) \le \gamma\, \hat{\pi}(x)$$

Then the relaxation times t_{rel} of P and $\hat{t}_{\mathrm{rel}}$ of $\hat{P}$ satisfy

$$t_{\mathrm{rel}} \le \alpha\,\gamma\,\hat{t}_{\mathrm{rel}}$$

5.3.3 Main Bound

We now combine the tools developed above to derive an explicit upper bound on the mixing time of the Metropolis–Hastings chain. Recall that $\Delta = \max_{\sigma\in\mathcal{S}_n} C(\sigma) - \min_{\sigma\in\mathcal{S}_n} C(\sigma)$ denotes the range of the cost function over the permutation space.

Theorem 13. *Let $(Z_t^{(\beta)})_{t\in\mathbb{N}}$ be the Metropolis–Hastings chain on $\mathcal{S}_n$ with stationary distribution π_β. For any $\beta > 0$,*

$$t_{\mathrm{rel}} = O\Big(n\, e^{\beta\Delta}\Big)$$

Proof. Let $\hat{P}$ denote the transition matrix of the random transposition shuffle, with uniform stationary distribution $\hat{\pi}(\sigma) = 1/n!$, and let P denote the Metropolis–Hastings transition matrix with stationary distribution $\pi_\beta(\sigma) \propto e^{-\beta C(\sigma)}$. For $\sigma \neq \sigma'$ related by a transposition, the MH transition satisfies

$$P(\sigma,\sigma') = \hat{P}(\sigma,\sigma')\,\min\!\left(1,\, \frac{\pi_\beta(\sigma')}{\pi_\beta(\sigma)}\right)$$

so the corresponding edge flows are

$$Q(\sigma,\sigma') = \pi_\beta(\sigma)\,P(\sigma,\sigma') = \hat{P}(\sigma,\sigma')\,\min\Big(\pi_\beta(\sigma), \pi_\beta(\sigma')\Big)$$

For the base chain,

$$\hat{Q}(\sigma,\sigma') = \hat{\pi}(\sigma)\,\hat{P}(\sigma,\sigma') = \frac{\hat{P}(\sigma,\sigma')}{n!}$$

To apply Theorem 12, we choose

$$\alpha = \frac{1}{n!} \cdot \frac{1}{\min_\sigma \pi_\beta(\sigma)} = \frac{Z_\beta}{n!}\, e^{\beta \max_\sigma C(\sigma)}$$

which ensures $\hat{Q}(\sigma, \sigma') \le \alpha\, Q(\sigma, \sigma')$ for all σ, σ', and

$$\gamma = \frac{\max_\sigma \pi_\beta(\sigma)}{\min_\sigma \hat{\pi}(\sigma)} = n!\, \frac{e^{-\beta \min_\sigma C(\sigma)}}{Z_\beta}$$

so that $\pi_\beta(\sigma) \le \gamma\, \hat{\pi}(\sigma)$ for all σ. By Theorem 12 and using $\hat{t}_{\mathrm{rel}} = n/2$ from Theorem 11,

$$t_{\mathrm{rel}} \le \alpha\, \gamma\, \hat{t}_{\mathrm{rel}} = e^{\beta(\max C - \min C)} \cdot \frac{n}{2} = O\Big(n\, e^{\beta \Delta}\Big)$$

where we used $\alpha\, \gamma = e^{\beta \Delta}$. □

Corollary 8. *For any $0 < \varepsilon < 1$, the mixing time of the Metropolis–Hastings chain satisfies*

$$t_{\mathrm{mix}}(\varepsilon) = O\Big(n\,(n \log n + \beta \Delta)\, e^{\beta \Delta}\Big)$$

Proof. By Theorem 10,

$$t_{\mathrm{mix}}(\varepsilon) \le t_{\mathrm{rel}}\, \log\!\left(\frac{1}{2\varepsilon \sqrt{\pi_{\min}}}\right)$$

Since $\pi_{\min} = \min_\sigma \pi_\beta(\sigma) \ge \frac{1}{n!}\, e^{-\beta \Delta}$, we have

$$\log\!\left(\frac{1}{2\varepsilon \sqrt{\pi_{\min}}}\right) = O(n \log n + \beta \Delta)$$

and combining with $t_{\mathrm{rel}} = O(n\, e^{\beta \Delta})$ from Theorem 13 yields the result. □

The bound in Corollary 8 shows that the mixing time grows exponentially in $\beta \Delta$, where Δ is the range of the cost function. This reflects a fundamental tension in Simulated Annealing: large β is needed to concentrate the stationary distribution on optimal solutions, but it also slows down convergence.

Chapter 6

Experiments

This chapter evaluates the practical performance of the algorithms developed in Chapter 5 on the full range of graph families, threshold parameters, weight distributions, and cost structures introduced in Chapter 1. The experimental comparison includes Simulated Annealing, Local Search, Local Search with Reheating, four greedy constructive heuristics, the DOT algorithm of Como et al. [9], and the Hungarian algorithm as an exact reference on complete graphs.

The results confirm that SA is the strongest overall method among those tested, and the first capable of handling effectively any combination of edge weights and intervention costs without restrictions on the cost structure. Its performance is robust across synthetic topologies for $l \in \{1/3, 1/2\}$, consistently achieving gaps below 1% w.r.t. the exact optimum or best observed solution. The cases $l = 0$ and majority threshold expose systematic failure modes, shared to varying degrees by all metaheuristics. On real-world networks, performance is strongly dependent on graph density and the iteration budget.

6.1 Algorithms

We compare nine algorithms. The primary object of study are three metaheuristics operating on the permutation space S_n, introduced in Chapter 5: Simulated Annealing (SA), Local Search (LS), and Local Search with Reheating (LS+R). These are benchmarked against several comparison methods: four greedy constructive heuristics (`inf`, `cinf`, `thr`, `ginf`), the Distributed Optimal Targeting algorithm (DOT) of Como et al. [9] included exclusively in the targeting experiments, and the Hungarian algorithm, which provides an exact reference solution on complete graph instances. Random Search is included as a sanity check.

Simulated Annealing (SA). SA operates on S_n with the transposition neighborhood. At each iteration, a random transposition (a, b) is proposed as the candidate move, which costs $O(1)$ to generate. Computing the resulting cost update ΔC requires recomputing the influence received by a and b from each other and from the nodes occupying intermediate positions in the current permutation; this step costs $O(d_a + d_b)$, where d_a and d_b denote the in-degrees of the two swapped nodes. The initial temperature is estimated adaptively from the instance by sampling 1000 random transpositions and setting

$$T_0 = -\frac{\overline{|\Delta|}}{\log p_0}, \qquad p_0 = 0.8,$$

where $\overline{|\Delta|}$ is the mean absolute cost change over the bottom 10% of sampled moves. The temperature decreases geometrically to $T_f = 10^{-7}$ over the full iteration budget N_{iter}, specified in Section 6.2. An early stopping criterion halts the search if the relative improvement over 1% of the budget falls below 0.5% for 20 consecutive checkpoints.

Local Search (LS). LS operates on S_n with the same transposition neighborhood as SA, accepting only non-worsening moves ($\Delta C \leq 0$). It is susceptible to trapping in local optima. The same early stopping criterion as SA applies.

Local Search with Reheating (LS+R). LS+R extends LS by monitoring stagnation: when the relative improvement falls below 0.5% for 10 consecutive checkpoints, the temperature is reset from the current best solution with acceptance probability $p_0 = 0.25$, and a new geometric cooling schedule is applied over the remaining iterations. This allows the search to escape shallow local optima while concentrating the search in the neighborhood of the current best solution.

Greedy Heuristics. The three greedy heuristics — `inf`, `cinf`, and `thr` — share a common constructive structure, described in Algorithm 11, and differ only in the node selection criterion ϕ. Starting from the full graph, nodes are placed one at a time into the permutation; after each placement, influence is propagated in cascade and residual resistances are updated. The overall running time is $O(n^2)$. The criterion `inf` selects the node exerting the greatest residual influence on its neighbors. The criterion `cinf` normalizes this influence by the activation cost, trading off influence against cost-efficiency. The criterion `thr` ignores influence entirely and selects the cheapest node to activate, acting as a pure cost-minimization greedy.

The three selection criteria are:

Algorithm 11 Greedy Heuristic

Require: Graph G, selection criterion ϕ
Ensure: Permutation σ, total cost C
$I \leftarrow V$, $C \leftarrow 0$, $k \leftarrow 1$
while $I \neq \emptyset$ **do**
  $i^* \leftarrow \arg\max_{i \in I} \phi(i)$
  $C \leftarrow C + C_{i^*}(r_{i^*})$
  $\sigma(i^*) \leftarrow k$, $k \leftarrow k + 1$
  Propagate influence of i^*: for each $j \in \mathcal{N}_{i^*} \cap I$, set $r_j \leftarrow r_j - W_{ji^*}$
  for all $j \in I$ with $r_j \leq 0$ **do**
    $\sigma(j) \leftarrow k$, $k \leftarrow k + 1$
    Propagate influence of j recursively
  end for
  Remove all newly activated nodes from I
end while
return σ, C

- `inf`: $\phi(i) = \sum_{j \in \mathcal{N}_i} \min(W_{ji},\, r_j)$, the residual influence exerted by i on its out-neighbors, with each arc contribution capped at the residual resistance of the target node;
- `cinf`: $\phi(i) = \sum_{j \in \mathcal{N}_i} \min(W_{ji},\, r_j) \,/\, C_i(r_i)$, the ratio of residual influence exerted to activation cost;
- `thr`: $\phi(i) = -C_i(r_i)$, the negative activation cost, so that the node with minimum cost is selected.

Gunnec Heuristic (`ginf`). Applicable to simple graphs with linear costs $C_i(x) = c_i x$, this heuristic follows the same constructive structure as Algorithm 11 with selection criterion $\phi(i) = -c_i$, selecting at each step the node with minimum cost coefficient. Before each selection, the instance is reduced to standard form (see Section 4.1): all nodes with $r_i \leq 0$ are activated for free and their influence is propagated in cascade. This coincides with the greedy algorithm of Günnec et al. [14] and yields an exact solution on simple trees with linear costs.

DOT. The Distributed Optimal Targeting algorithm of Como et al. [9] operates on binary activation states $\{0,1\}^n$ rather than on permutations, following a dual approach: starting from the full seed set V, nodes are iteratively removed when their activation is no longer necessary. At each step a node i is selected uniformly at random. If i is active and already receives sufficient influence from its active

neighbors (i.e. $\sum_{j \in \mathcal{N}_i} W_{ij} \geq r_i$), it is removed from the seed set at no cost. If i is inactive, it is added to the seed set with probability $e^{-C_i(r_i)/T}$, where T is the current temperature following a geometric cooling schedule from T_0 to $T_f = 0.1$. DOT is included exclusively in the targeting experiments, for which it was originally designed.

Hungarian Algorithm. For simple complete graphs, the LCIP reduces to a minimum-weight perfect matching on $K_{n,n}$, solved exactly in $O(n^3)$ by the Hungarian algorithm, as established in Chapter 2. This provides the exact optimum and serves as the reference for gap computation on complete graph instances.

Random Search. Ten independent random permutations are generated and the best solution retained, with no local improvement applied. It is included as a sanity check to confirm that all other methods produce non-trivial solutions.

6.2 Graph Instances

All graphs considered in this work are undirected in origin. As a preprocessing step, each undirected edge $\{u, v\}$ is replaced by two directed arcs (u, v) and (v, u), and self-loops are removed. The resulting simple directed graph is the instance on which the LTM and the intervention problem are defined.

Complete Graphs. We generate complete graphs K_n for $n \in \{100, 200, 400\}$. These instances are included because the Hungarian algorithm provides an exact reference solution under uniform weights and linear costs, enabling evaluation of the metaheuristics in terms of absolute gap from the optimum.

Random Trees. We generate random trees T_n for $n \in \{100, 200, 400\}$. On these instances the Gunnec Heuristic solves the LCIP exactly under uniform weights and linear costs, again permitting exact gap evaluation.

Small-World Graphs. To probe performance on sparse, clustered topologies, we use Watts–Strogatz small-world graphs [24], constructed starting from a ring lattice in which each node is connected to its k nearest neighbors on each side, with each edge then independently rewired with probability $p = 0.1$. The resulting graphs retain high local clustering while acquiring short average path lengths, making them a natural model for social influence networks. We use $k \in \{10, 20\}$ and $n \in \{100, 200, 400\}$.

Real-World Graphs. Real-world networks are the primary target of influence maximization applications, as they model the social structures in which diffusion phenomena naturally arise. Unlike synthetic graphs, they exhibit heterogeneous degree distributions, emergent community structure, and irregular topologies that cannot be replicated by construction. We include two collections with complementary structural properties, described below. For each graph, we report the *average clustering coefficient*

$$C = \frac{1}{|V|} \sum_{i \in V} c_i, \qquad c_i = \frac{|\{\{j,k\} \subseteq \mathcal{N}_i \,:\, \{j,k\} \in E\}|}{\binom{d_i}{2}}$$

where d_i is the degree of node i and c_i is the fraction of pairs of neighbors of i that are themselves connected.

The first collection consists of five co-authorship networks from the Stanford Network Analysis Project (SNAP) [25], where each node represents an author and each edge connects two authors who co-authored at least one paper on arXiv:

- `CA-GrQc`,
- `CA-HepTh`,
- `CA-HepPh`,
- `CA-CondMat`,
- `CA-AstroPh`.

These networks are sparse, with low average degree and high clustering coefficient.

The second collection consists of five university friendship networks from the Facebook100 dataset, available through the Network Data Repository [26], where each node represents a student and each edge a mutual Facebook friendship:

- `socfb-Cornell5`,
- `socfb-Penn94`,
- `socfb-Rice31`,
- `socfb-WashU32`,
- `socfb-JohnsHopkins55`.

These networks are significantly denser than the collaboration graphs, with average degree in the range $[100, 190]$, providing a complementary stress test on high-connectivity instances. Their statistics are reported in Table 6.1.

Collection	Graph	$\lvert V\rvert$	$\lvert E\rvert$	$\bar{d}$	C
SNAP	CA-GrQc	5,242	14,484	11.05	0.530
	CA-HepTh	9,877	25,973	10.52	0.471
	CA-HepPh	12,008	118,489	39.47	0.611
	CA-CondMat	23,133	93,439	16.16	0.633
	CA-AstroPh	18,772	198,050	42.20	0.631
Facebook	socfb-Rice31	4,087	184,828	180.89	0.294
	socfb-JohnsHopkins55	5,180	186,586	144.08	0.268
	socfb-WashU32	7,755	367,541	189.58	0.261
	socfb-Cornell5	18,660	790,777	169.51	0.219
	socfb-Penn94	41,536	1,362,220	131.18	0.212

Table 6.1: Statistics of the real-world graphs. $|V|$: number of nodes; $|E|$: number of undirected edges; $\bar{d}$: average degree; C: average clustering coefficient.

6.3 Experimental Settings

All experiments share a common set of parameters. The resistance of each node i is drawn as

$$r_i \sim \text{Uniform}(l \cdot w_i,\ w_i)$$

where $w_i = \sum_{j \in \mathcal{N}_i} W_{ij}$ and $l \in \{0, \frac{1}{3}, \frac{1}{2}\}$ is a hardness parameter: larger values of l push resistances closer to w_i, making cascade propagation harder and requiring greater intervention to achieve full adoption. We additionally consider the special case $r_i = w_i/2$ for all i, corresponding to the majority threshold of the network coordination game studied in Chapter 1. Each configuration is replicated over three independent random seeds and all reported metrics are averaged over the three replications. The iteration budget is set to $N_{\text{iter}} = 40\,n^2$ for synthetic instances and $N_{\text{iter}} = 1000\,|V|$ for real-world instances.

Evaluation Metric. All results are reported in terms of gap percentage with respect to a reference solution. For each seed s, the gap of algorithm $\mathcal{A}$ is defined as

$$\text{gap}(\mathcal{A}) = \frac{C(\mathcal{A}) - C^{\text{ref}}}{C^{\text{ref}}} \times 100$$

where $C(\mathcal{A})$ is the cost returned by $\mathcal{A}$ and C^{ref} is the reference cost on the same seed. When an exact algorithm is available — the Hungarian algorithm on K_n in Experiments 1 and 3, and `ginf` on T_n in Experiment 1 — the reference is the exact optimum C^*. Otherwise, the reference is the best solution found across all algorithms on the same seed. The reported gap is the average over the three seeds.

Experiment 1: Simple/Linear. Edge weights are uniform ($W_{ij} = 1$ for all $(i, j) \in E$), i.e. simple graphs, and cost functions are linear, $C_i(x) = c_i x$, with coefficients $c_i \sim \text{Uniform}(1, 50)$ drawn independently for each node. All graph families are included. In addition to the metaheuristics and greedy heuristics, the Gunnec heuristic (`ginf`) is included as a comparison method. For complete graphs K_n, the Hungarian algorithm provides an exact reference solution; for random trees T_n, the Gunnec heuristic (`ginf`) is exact and serves as the reference optimum.

Experiment 2: Weighted/General. Edge weights are drawn independently from Uniform$(1, 50)$. Two cost regimes are considered:

- **Linear costs**: $C_i(x) = x$.
- **Mixed costs**: each node independently receives one of three cost functions chosen uniformly at random: linear $C_i(x) = x$, indicator $C_i(x) = c_i \mathbf{1}_{x>0}$, or piecewise $C_i(x) = \max(x, c_i)\mathbf{1}_{x>0}$, where $c_i \sim \text{Uniform}(1, w_i)$ in each case.

All graph families are included. The Gunnec heuristic and the hungarian algorithms are not applicable in this setting. The mixed cost regime tests the robustness of the algorithms to heterogeneous and nonlinear cost structures, corresponding to the full generality of the LCIP.

Experiment 3: Simple/Targeting. Edge weights are uniform ($W_{ij} = 1$), i.e. simple graphs, and cost functions are of fixed-cost type. Two variants are considered:

- **Homogeneous**: $C_i(x) = \mathbf{1}_{x>0}$ for all i, corresponding to the unweighted TSS problem.
- **Heterogeneous**: $C_i(x) = c_i \mathbf{1}_{x>0}$, with $c_i \sim \text{Uniform}(1, 50)$, corresponding to the WTSS problem.

All graph families are included. In addition to the metaheuristics and greedy heuristics, the DOT algorithm is included as comparison method. For complete graphs K_n, the Hungarian algorithm provides an exact reference.

6.4 Results

We report results in terms of *gap percentage* with respect to a reference solution. For complete graphs in Experiment 1 and 3, the reference is the exact optimum provided by the Hungarian algorithm. For random trees in Experiment 1, the reference is the exact optimum provided by the Gunnec tree algorithm. For all

Experiment	Weights	Costs	Exact reference
1: Simple/Linear	Uniform	Linear	Hungarian (K_n), `ginf` (T_n)
2: Weighted/General	[1,50]	Linear Mixed	—
3: Simple/Targeting	Uniform	Homogeneous Heterogeneous	Hungarian (K_n)

Table 6.2: Summary of the experimental matrix.

other instances, the reference is the best solution found across all algorithms on the same run. Not all results are reported in this section; additional tables are deferred to Appendix A.

6.4.1 Complete Graphs

Complete graphs K_n are the densest family considered and the only one, with trees, for which an exact reference solution is available in the first and third experiments via the Hungarian algorithm.

Experiment 1: Simple/Linear

In Experiment 1, weights are uniform and costs are linear; the Hungarian algorithm provides the exact optimum as reference.

l	n	SA	LS	LS+R	CInf	Thr	GInf	Rnd
0	100	**12.67**	24.03	18.10	65.37	65.37	59.60	11099.63
	200	7.99	**4.48**	7.89	126.10	126.10	58.86	42332.85
	400	**3.68**	13.14	12.98	111.12	111.12	113.80	84221.59
1/3	100	**0.21**	0.45	0.47	45.78	45.78	8.86	300.44
	200	**0.06**	0.30	0.55	48.98	48.98	5.53	304.89
	400	**0.04**	0.14	0.19	57.39	57.39	4.86	311.60
1/2	100	**0.01**	0.10	0.14	27.38	27.38	3.88	157.77
	200	**0.01**	0.09	0.14	25.66	25.66	4.33	145.12
	400	**0.03**	0.05	0.08	29.41	29.41	3.31	155.77
Maj.	100	**0.00**	0.00	0.00	0.00	0.00	0.00	169.37
	200	**0.00**	0.00	0.00	0.00	0.00	0.00	152.42
	400	**0.00**	0.00	0.00	0.00	0.00	0.00	161.44

Table 6.3: Gap (%) w.r.t. Hungarian exact optimum, K_n, Experiment 1. Best heuristic result per row in **bold**.

For $l \in \{1/3, 1/2\}$, SA and LS (LS+R) are the best algorithms (0.01–0.21% w.r.t. the Hungarian exact optimum), with `ginf` close behind at 3–9%, consistent

with the matching reduction on K_n. Note that `cinf` and `thr` produce identical gaps under uniform weights, as the residual influence is the same for all nodes at each step. For $l = 0$, all methods degrade; SA remains among the best heuristics (4–13%). The majority threshold is trivial on K_n: all algorithms except Random Search find the exact optimum.

Experiment 2: Weighted/General

In Experiment 2, weights are drawn uniformly at random; the gap is computed w.r.t. the best observed solution across all algorithms.

l	n	SA	LS	LS+R	Inf	CInf	Thr	Rnd
0	100	2.82	6.11	**1.92**	729.68	13.12	13.43	19752.31
	200	**0.00**	4.25	3.55	113.18	24.46	25.39	982.59
	400	**0.00**	1.99	1.89	142.20	23.05	23.59	1863.39
1/3	100	**0.00**	1.48	1.29	18.78	15.88	16.76	77.30
	200	**0.00**	0.94	1.20	13.68	12.12	12.63	52.90
	400	**0.00**	0.96	0.93	9.90	9.42	9.68	52.34
1/2	100	**0.00**	1.29	1.25	11.59	12.46	13.57	34.83
	200	**0.00**	0.99	0.85	9.27	9.53	10.18	25.22
	400	**0.00**	0.77	0.84	6.14	7.46	7.79	23.79
Maj.	100	**0.44**	1.76	1.33	14.85	5.93	8.81	23.07
	200	**0.00**	1.61	1.21	10.57	4.79	6.39	16.78
	400	**0.00**	1.06	1.05	7.60	3.51	4.79	11.95

Table 6.4: Gap (%) w.r.t. best observed, K_n, Experiment 2, linear costs. Best result per row in **bold**.

Under linear costs (Table 6.4), SA and LS (LS+R) are the best algorithms by a wide margin. LS and LS+R close the gap with SA as l increases, reaching below 1.5% for $l \in \{1/3, 1/2\}$. Among the greedy heuristics, `cinf` performs best, incurring 7–15% degradation relative to SA, with the gap narrowing under the majority threshold (3.51–5.93%).

Under mixed costs (Table A.1), SA remains optimal for $l \in \{1/3, 1/2, \text{Maj.}\}$ (0–0.49%), with the same qualitative pattern as the linear case; LS and LS+R degrade slightly (1–5%). The single exception is $l = 0$, where SA collapses to 87–309% above the best observed solution while `cinf` and `thr` are the best algorithms: the combination of low thresholds and the high connectivity of K_n renders most transpositions cost-neutral.

Experiment 3: Simple/Targeting

In Experiment 3, weights are uniform and costs are indicator functions; the Hungarian algorithm provides the exact optimum as reference.

l	n	SA	LS	LS+R	CInf	Thr	DOT	Rnd
0	100	58.33	40.28	76.39	197.22	**29.17**	702.78	726.39
	200	167.32	169.61	171.24	341.83	**55.23**	1612.75	1629.41
	400	271.67	274.26	289.63	249.63	**50.37**	2605.37	2589.07
1/3	100	**0.98**	0.98	1.96	48.04	17.65	58.82	81.37
	200	**0.47**	0.94	2.40	44.24	14.56	66.37	88.23
	400	**0.00**	0.00	0.74	13.16	7.46	70.22	92.55
1/2	100	**0.00**	0.00	0.67	38.00	16.67	21.33	41.33
	200	**0.00**	0.33	0.33	27.72	9.24	25.82	44.34
	400	**0.00**	0.00	0.00	8.49	3.16	27.45	45.92

Table 6.5: Gap (%) w.r.t. Hungarian exact optimum, K_n, Experiment 3, homogeneous costs. Best result per row in **bold**.

Under homogenous costs (Table 6.5), for $l \in \{1/3, 1/2\}$, SA and LS are the best algorithms (0–0.98% w.r.t. the Hungarian exact optimum), with LS+R close behind. Under targeting on simple complete graph, `cinf` and `thr` should theoretically produce identical results; small differences in the reported gaps arise from tie-breaking. DOT is not competitive (21–70%). For $l = 0$, SA and LS collapse to 40–272%, while `thr` is the most robust method (29–55%).

Under heterogeneous costs (Table A.2), SA and LS recover to below 2% for $l \in \{1/3, 1/2\}$. For $l = 0$ SA degrades to 68–393% while `cinf` and `thr` achieve 34–49%. DOT remains ineffective throughout.

6.4.2 Random Trees

Trees are the sparsest graph family considered: with $n - 1$ edges and no cycles.

Experiment 1: Simple/Linear

In Experiment 1, weights are uniform and costs are linear; `ginf` provides the exact optimum as reference.

Under simple graph ans linear costs (Table 6.6), SA performs very well across all threshold regimes, achieving below 0.65% for $l \in \{0, 1/3, 1/2\}$ and consistently outperforming LS and LS+R. The greedy heuristics `cinf` and `thr` are reasonably competitive at 2–15%, substantially better than on complete graphs. The only problematic regime for SA is the majority threshold, where `thr` dominates (0.17–1.89%) while SA degrades to 6–12%.

Experiment 2: Weighted/General

In Experiment 2, weights are drawn uniformly at random; the gap is computed w.r.t. the best observed solution.

l	n	SA	LS	LS+R	CInf	Thr	Rnd
0	100	**0.40**	2.60	3.81	10.51	2.16	253.44
	200	**0.58**	2.64	2.65	10.72	1.76	221.93
	400	**0.55**	3.92	2.55	7.19	2.21	234.77
1/3	100	**0.34**	1.23	1.15	7.05	3.91	106.36
	200	**0.47**	1.63	1.55	7.12	2.81	115.57
	400	**0.65**	1.70	1.44	6.60	3.88	121.03
1/2	100	**0.26**	0.74	0.30	5.62	4.49	77.03
	200	**0.02**	1.06	0.37	3.81	4.14	87.53
	400	**0.29**	0.69	0.88	5.06	3.93	83.79
Maj.	100	11.86	25.11	29.52	10.38	**1.17**	459.14
	200	7.18	28.66	25.86	15.27	**0.17**	415.57
	400	6.05	22.73	20.85	13.64	**1.89**	411.66

Table 6.6: Gap (%) w.r.t. `ginf` exact optimum, T_n, Experiment 1. Best heuristic result per row in **bold**.

l	n	SA	LS	LS+R	CInf	Thr	Rnd
0	100	**0.04**	1.56	1.23	10.85	17.55	187.32
	200	**0.00**	1.13	1.46	14.53	13.95	209.74
	400	**0.00**	1.42	1.80	14.46	17.01	205.21
1/3	100	**0.02**	0.48	1.13	13.36	21.42	121.12
	200	**0.08**	0.28	1.58	17.10	20.24	118.16
	400	**0.06**	0.19	1.07	14.03	22.93	124.33
1/2	100	**0.00**	0.41	0.50	10.69	25.47	87.89
	200	0.28	0.27	**0.25**	12.13	19.91	92.08
	400	0.22	**0.11**	0.27	12.24	24.96	92.68
Maj.	100	**0.00**	9.22	8.51	27.03	37.56	256.15
	200	**0.00**	3.94	6.28	26.69	29.99	257.27
	400	**0.00**	1.75	5.76	28.92	37.17	257.05

Table 6.7: Gap (%) w.r.t. best observed, T_n, Experiment 2, linear costs. Best result per row in **bold**.

Under linear costs (Table 6.7), SA is the best algorithm throughout, achieving below 0.08% for $l \in \{0, 1/3\}$ and near-zero for $l = 1/2$. LS and LS+R follow at 0.1–1.8%, a slight degradation relative to SA that is consistent across all sizes. `cinf` incurs approximately 10–17% across all non-majority regimes, with the gap widening under the majority threshold (27–29%).

Under mixed costs (Table A.3), the same pattern holds with SA near 0% throughout. `cinf` incurs 15–47%, a further deterioration relative to the linear case, confirming that SA's advantage over greedy heuristics grows with cost complexity.

Experiment 3: Simple/Targeting

In Experiment 3, weights are uniform and costs are indicator functions; the gap is computed w.r.t. the best observed solution.

l	n	SA	LS	LS+R	CInf	Thr	DOT	Rnd
0	100	1.52	**0.00**	1.52	46.97	58.76	51.52	91.11
	200	**0.00**	0.00	0.71	41.15	52.92	64.94	102.72
	400	0.37	0.37	**0.00**	45.92	65.67	72.02	105.59
1/3	100	**0.00**	0.00	0.00	25.17	49.38	14.69	57.29
	200	0.51	0.51	**0.00**	26.25	49.37	21.14	65.00
	400	**0.00**	0.00	0.00	24.89	58.82	22.87	63.62
1/2	100	**0.00**	0.00	0.00	19.00	47.78	3.34	37.25
	200	**0.00**	0.00	0.00	14.25	43.43	5.05	45.70
	400	**0.00**	0.00	0.00	18.02	52.77	5.69	46.42
Maj.	100	4.95	**0.00**	0.00	17.03	70.70	124.73	168.86
	200	3.49	**1.11**	4.68	21.51	63.33	113.65	175.32
	400	0.57	**0.00**	1.19	15.68	75.38	130.54	184.41

Table 6.8: Gap (%) w.r.t. best observed, T_n, Experiment 3, homogeneous costs. Best result per row in **bold**.

Under homogeneous costs (Table 6.8), SA, LS, and LS+R are the best algorithms across all regimes, with gaps at or near 0% for $l \in \{0, 1/3, 1/2\}$. Under the majority threshold, LS dominates (0–1.11%) while SA degrades slightly (0.57–4.95%). Among the greedy heuristics, `cinf` is the best, incurring 14–47% degradation; notably, DOT is competitive for $l = 1/2$ (3–6%), its best performance across all topologies.

Under heterogeneous costs (Table A.4), SA remains the best algorithm (0–0.28%) across all l values except majority, where SA incurs 0–4.80% and `cinf` and `thr` achieve 7–12%. DOT remains competitive for $l = 1/2$ (6.50–8.61%) but degrades for other threshold values.

6.4.3 Small-World Graphs

Small-world graphs combine high local clustering with short average path lengths, making them a natural model for social influence networks. We report results for Watts–Strogatz graphs with $k = 10$ nearest neighbors and with $k = 20$.

Experiment 1: Simple/Linear

In Experiment 1, weights are uniform and costs are linear; the gap is computed w.r.t. the best observed solution.

For $k = 10$ (Table 6.9), SA is the best algorithm for $l \in \{1/3, 1/2\}$ (0% gap), with LS following at 0.83–2.60%. For $l = 0$, SA is unstable: it achieves 0.32% for $n = 400$ but degrades to 21.90% for $n = 200$, competing with `ginf` (3.94–10.32%)

l	n	SA	LS	LS+R	CInf	Thr	GInf	Rnd
0	100	11.89	10.75	22.14	22.85	22.00	**4.75**	3710.71
	200	21.90	**10.15**	32.72	20.35	16.97	10.32	4868.91
	400	**0.32**	3.88	8.57	11.82	13.34	3.94	3464.19
1/3	100	**0.00**	2.60	5.27	32.55	32.56	2.22	235.67
	200	**0.00**	1.14	2.63	28.87	33.11	2.63	239.12
	400	**0.00**	1.70	1.87	29.86	34.94	2.53	295.40
1/2	100	**0.00**	1.19	0.25	17.89	21.78	0.75	119.99
	200	**0.00**	1.45	0.91	18.33	24.71	1.64	126.10
	400	**0.00**	0.83	0.85	18.13	23.93	1.87	152.15
Maj.	100	15.49	26.55	36.08	1.73	**0.85**	52.68	477.85
	200	37.47	55.74	43.50	8.26	**2.90**	80.39	673.37
	400	34.69	30.36	35.62	**0.00**	8.22	60.33	809.56

Table 6.9: Gap (%) w.r.t. best observed, Watts–Strogatz ($k = 10$), Experiment 1. Best result per row in **bold**.

which is the most reliable method in this regime. The majority threshold inverts the ranking: `thr` dominates (0.85–8.22%) while SA collapses to 15–37%.

For $k = 20$ (Table A.5), the pattern is qualitatively preserved: SA achieves near-zero for $l \in \{1/3, 1/2\}$ (0–0.04%) and competes with `ginf` for $l = 0$. Under the majority threshold, `thr` again dominates (0–4.27%) while SA incurs 5–19%, with the degradation growing with n.

Experiment 2: Weighted/General

In Experiment 2, weights are drawn uniformly at random; the gap is computed w.r.t. the best observed solution.

l	n	SA	LS	LS+R	CInf	Thr	Rnd
0	100	**3.29**	7.57	3.57	13.91	19.50	1427.63
	200	**0.00**	1.22	9.27	19.45	22.65	1362.63
	400	**0.00**	11.57	7.61	23.23	33.63	1470.49
1/3	100	**0.00**	3.82	3.30	22.03	24.33	80.75
	200	**0.00**	3.05	4.10	25.95	31.58	93.94
	400	**0.00**	2.98	3.46	25.27	29.32	90.96
1/2	100	**0.00**	1.88	2.76	17.83	22.96	46.37
	200	**0.00**	2.17	2.16	19.11	26.33	48.22
	400	**0.00**	2.06	2.67	19.94	25.40	50.03
Maj.	100	1.31	**0.00**	10.07	6.90	12.65	155.73
	200	25.73	29.26	35.42	**0.93**	4.44	231.06
	400	61.12	63.21	52.18	**0.00**	6.01	319.23

Table 6.10: Gap (%) w.r.t. best observed, Watts–Strogatz ($k = 10$), Experiment 2, linear costs. Best result per row in **bold**.

l	n	SA	LS	LS+R	CInf	Thr	Rnd
0	100	21.23	28.65	23.46	32.20	**29.47**	1244.05
	200	328.88	330.72	527.61	**0.00**	10.64	6437.39
	400	38.00	86.91	119.49	**6.76**	16.19	2184.42
1/3	100	**0.00**	13.50	11.83	32.19	35.11	178.86
	200	**0.00**	12.63	12.82	18.48	23.99	202.30
	400	**0.00**	14.60	12.90	24.34	29.07	192.48
1/2	100	**0.00**	5.32	3.07	21.64	30.93	101.45
	200	**0.00**	6.54	3.48	13.15	19.59	110.32
	400	**0.00**	9.33	10.32	17.07	24.39	114.39
Maj.	100	22.21	36.68	27.44	27.49	**3.65**	357.84
	200	31.77	43.07	52.21	13.10	**10.86**	457.29
	400	36.18	40.14	48.63	**1.48**	8.59	500.88

Table 6.11: Gap (%) w.r.t. best observed, Watts–Strogatz ($k = 10$), Experiment 2, mixed costs. Best result per row in **bold**.

For $k = 10$ and under linear costs (Table 6.10), with $l \in \{1/3, 1/2\}$ SA achieves 0% gap throughout, with LS and LS+R following at 2–4% — a slightly larger gap than on complete graphs, reflecting the irregular topology. `cinf` and `thr` incur 18–32%. For $l = 0$, SA is near-optimal for $n \in \{200, 400\}$ and incurs 3.29% for $n = 100$, remaining the best algorithm. The majority threshold produces strong instability in SA: from 1.31% at $n = 100$ to 61% at $n = 400$, while `cinf` achieves 0–6.90%.

Under mixed costs (Table 6.11), with $l \in \{1/3, 1/2\}$ SA is again the best algorithm (0%), while LS and LS+R degrade more noticeably (3–15%) and `cinf` and `thr` incur 13–35%. For $l = 0$, no method dominates consistently: SA ranges from 21% to 329% with high variance across sizes, while `cinf` and `thr` are more stable (0–32%). The majority threshold under mixed costs is problematic for all metaheuristics (22–36%), with `thr` the most reliable (3.65–10.86%).

For $k = 20$ (Tables A.6 and A.7), the linear cost pattern is preserved: SA achieves 0% for $l \in \{1/3, 1/2\}$ and dominates for $l = 0$; the majority threshold produces the same instability (1–42%). Under mixed costs and $l = 0$, the failure is dramatically amplified: SA reaches 1441% for $n = 400$, the largest gap observed across all synthetic topologies, while `thr` achieves 0%. For $l \in \{1/3, 1/2\}$, SA recovers to 0%.

Experiment 3: Simple/Targeting

In Experiment 3, weights are uniform and costs are indicator functions. The gap is computed w.r.t. the best observed solution across all algorithms.

Under homogeneous costs (Table 6.12), SA, LS, and LS+R are the best algorithms across all regimes. For $l \in \{1/3, 1/2\}$, SA achieves 0–0.85% and LS

l	n	SA	LS	LS+R	CInf	Thr	DOT	Rnd
0	100	**4.76**	17.68	32.33	40.33	72.66	347.35	342.59
	200	**9.09**	17.70	24.52	25.60	88.43	386.98	414.37
	400	4.17	**1.71**	9.23	32.96	113.33	415.54	441.98
1/3	100	**0.00**	5.45	5.45	23.43	109.19	71.21	103.33
	200	**0.52**	1.12	5.46	22.27	113.91	76.65	108.37
	400	0.85	**0.58**	2.00	27.33	126.01	84.54	113.88
1/2	100	**0.74**	3.88	2.33	18.93	97.17	35.92	61.93
	200	**0.00**	1.97	3.97	18.92	110.80	40.84	68.67
	400	**0.60**	1.01	1.60	23.85	116.43	46.03	70.66
Maj.	100	**1.39**	2.62	9.51	21.57	49.32	62.89	84.15
	200	3.04	**0.79**	7.80	29.26	60.84	91.89	118.84
	400	4.56	**1.65**	7.59	19.96	77.63	108.49	132.71

Table 6.12: Gap (%) w.r.t. best observed, Watts–Strogatz ($k = 10$), Experiment 3, homogeneous costs. Best result per row in **bold**.

l	n	SA	LS	LS+R	CInf	Thr	DOT	Rnd
0	100	53.24	15.75	41.78	15.97	**8.67**	882.99	1031.96
	200	**15.86**	21.37	39.34	41.33	24.73	733.76	861.27
	400	9.04	10.51	16.61	19.47	**8.64**	1007.34	1158.78
1/3	100	3.06	**1.03**	22.44	24.08	33.43	162.16	220.58
	200	**1.47**	2.94	6.35	27.24	29.53	158.60	230.08
	400	**0.56**	1.54	7.43	27.10	24.45	186.89	260.59
1/2	100	5.00	**0.00**	7.15	25.21	43.12	77.64	125.47
	200	**0.89**	1.27	3.77	24.34	28.40	82.55	136.29
	400	**0.00**	3.79	3.17	28.66	32.91	88.50	140.11
Maj.	100	6.21	**2.32**	20.69	20.96	6.61	176.39	328.47
	200	16.55	27.69	22.64	18.02	**0.89**	246.32	445.15
	400	18.72	26.47	29.69	16.04	**3.65**	350.63	584.67

Table 6.13: Gap (%) w.r.t. best observed, Watts–Strogatz ($k = 10$), Experiment 3, heterogeneous costs. Best result per row in **bold**.

follows at 0.58–3.88%; `cinf` is the best greedy heuristic at 19–27%. For $l = 0$, the metaheuristics remain dominant (1.71–9.09%) though with some instability across the methods. The majority threshold does not produce the inversion observed in Experiments 1 and 2: SA and LS remain competitive (1–5%) while `cinf` incurs 20–29%. For $k = 20$ (Table A.8), the pattern is qualitatively preserved with SA and LS near 0% for $l \in \{1/3, 1/2\}$ and slightly larger instability for $l = 0$ (2–16%).

Under heterogeneous costs (Table 6.13), SA, LS, and LS+R dominate for $l \in \{1/3, 1/2\}$, with SA achieving 0–5% and LS 0–3.79%; `cinf` and `thr` incur 24–43%. For $l = 0$, results are highly unstable: SA ranges from 9% to 53% with no consistent winner among SA, LS, and `thr`. Under the majority threshold, `thr` dominates (0.89–6.61%) while SA degrades to 6–19%. For $k = 20$ (Table A.9), the same structure holds with slightly larger gaps for SA at $l = 0$ (15–62%).

6.4.4 Real-World Networks

Real-world networks are evaluated with an iteration budget of $N_{\text{iter}} = 1000\,|V|$, substantially smaller than the $40\,n^2$ budget used for synthetic instances. This choice was motivated by computational constraints. The results reported here should therefore be interpreted as a lower bound on SA performance under unconstrained computation; runtime analysis is presented at the end of this chapter.

Experiment 1: Simple/Linear

In Experiment 1, weights are uniform and costs are linear; the gap is computed w.r.t. the best observed solution.

Graph	SA	LS	LS+R	CInf	Thr	GInf	Rnd
$l = 0$							
CA-AstroPh	17.92	25.51	25.41	33.39	26.97	**0.00**	2994.01
CA-CondMat	5.59	11.35	12.86	19.86	13.12	**0.00**	944.33
CA-GrQc	0.69	3.97	6.25	11.91	9.40	**0.00**	585.19
CA-HepPh	5.38	10.94	11.69	38.00	47.79	**0.00**	2237.22
CA-HepTh	3.88	10.14	10.54	14.37	7.41	**0.00**	636.86
$l = 1/3$							
CA-AstroPh	0.91	2.59	3.11	37.18	68.59	**0.00**	286.47
CA-CondMat	0.82	3.31	4.17	21.05	32.26	**0.00**	221.33
CA-GrQc	**0.00**	2.52	2.63	16.61	23.00	1.27	189.03
CA-HepPh	**0.00**	1.32	1.67	35.76	59.47	0.85	260.64
CA-HepTh	0.89	4.51	4.76	16.97	21.66	**0.00**	194.75
$l = 1/2$							
CA-AstroPh	0.14	1.05	1.37	20.36	47.71	**0.00**	150.24
CA-CondMat	0.15	1.67	1.91	13.29	29.68	**0.06**	131.32
CA-GrQc	**0.00**	1.47	1.74	10.99	20.82	1.00	118.61
CA-HepPh	**0.00**	0.71	0.84	20.10	39.58	0.64	142.31
CA-HepTh	0.46	2.43	2.31	11.46	22.46	**0.00**	121.09

Table 6.14: Gap (%) w.r.t. best observed, SNAP networks, Experiment 1. Best result per row in **bold**.

Under uniform weights and linear costs (Table 6.14 and Table 6.15), `ginf` finds the best observed solution for $l = 0$. On SNAP, SA achieves near-optimal results for $l \in \{1/3, 1/2\}$ (0–0.91%), closely matching `ginf`, while LS and LS+R follow at 0.71–4.76%. For $l = 0$, SA degrades to 0.69–17.92%, with performance varying across graphs; CA-GrQc, the sparsest graph, yields the smallest gap while CA-AstroPh, the densest, the largest. These are encouraging results, though it would be interesting to evaluate SA under a budget scaling as n^2, which is expected to further close the gap with `ginf`.

On Facebook networks, SA degrades more substantially for $l \in \{1/3, 1/2\}$ (0.61–2.71%) and collapses for $l = 0$ (87–127%), while `ginf` remains the best observed

Graph	SA	LS	LS+R	CInf	Thr	GInf	Rnd
$l = 0$							
Cornell5	105.04	114.68	112.46	130.43	106.53	**0.00**	18412.10
JohnsHopkins	108.39	125.15	126.75	60.26	56.67	**0.00**	23271.84
Penn94	124.52	141.61	141.57	105.04	70.22	**0.00**	19733.82
Rice31	126.84	141.85	143.27	71.85	65.82	**1.68**	26110.48
WashU32	86.86	97.10	98.21	127.03	113.63	**0.00**	16088.31
$l = 1/3$							
Cornell5	2.08	2.79	3.47	55.68	84.45	**0.00**	323.44
JohnsHopkins	1.20	1.72	2.36	53.94	80.01	**0.00**	324.55
Penn94	2.71	3.47	4.01	53.71	85.22	**0.00**	322.89
Rice31	1.14	1.69	2.52	54.16	77.36	**0.07**	326.38
WashU32	1.96	2.49	3.05	55.04	78.21	**0.00**	319.08
$l = 1/2$							
Cornell5	1.00	1.37	1.55	27.13	51.61	**0.00**	159.57
JohnsHopkins	0.71	1.13	1.39	27.00	49.58	**0.00**	161.00
Penn94	1.18	1.70	1.88	26.92	53.26	**0.00**	160.17
Rice31	0.61	1.04	1.34	26.95	47.24	**0.00**	161.49
WashU32	0.75	1.18	1.32	27.18	47.32	**0.00**	156.86

Table 6.15: Gap (%) w.r.t. best observed, Facebook networks, Experiment 1. Best result per row in **bold**.

solution. This performance drop is consistent with the higher density and larger size of Facebook graphs: with a budget linear in $|V|$.

Experiment 2: Weighted/General

In Experiment 2, weights are drawn uniformly at random. The gap is computed w.r.t. the best observed solution across all algorithms.

Under linear costs (Table 6.16 and Table 6.17), for $l \in \{1/3, 1/2\}$, SA achieves near-optimal results on both network families. On SNAP, SA incurs 0–2.33%; notably, LS+R is frequently more effective than LS on individual instances (e.g. CA-GrQc $l = 1/3$: LS+R=0.37% vs LS=5.26%), suggesting that reheating helps escape local optima on sparse irregular topologies. `cinf` is the best greedy heuristic at 8–46% on SNAP. On Facebook, SA achieves 0–0.14% for $l \in \{1/3, 1/2\}$, with `cinf` incurring 27–57%.

For $l = 0$, both families show severe degradation of SA. On SNAP, performance is topology-dependent: SA is near-optimal on CA-CondMat, CA-GrQc, and CA-HepTh (0.52–1.68%) but degrades on CA-AstroPh and CA-HepPh (33–50%), the two densest graphs; `thr` dominates on the latter. On Facebook, SA collapses uniformly to 52–235% while `thr` achieves 0% on all graphs. The majority threshold produces the familiar inversion on both families: `cinf` achieves 0–1.61% on SNAP and 0% on Facebook while SA incurs 11–131% and 17–105% respectively.

Under mixed costs (Tables A.10 and A.11), SA remains near-optimal for $l \in \{1/3, 1/2\}$ on SNAP (0–0.16%) and partially recovers on Facebook (0–4.35%),

Graph	SA	LS	LS+R	CInf	Thr	Rnd
$l = 0$						
CA-AstroPh	50.11	53.87	56.75	13.39	**0.00**	12340.66
CA-CondMat	1.51	16.14	16.84	8.10	**0.06**	3415.53
CA-GrQc	1.68	7.74	**0.90**	15.75	15.56	1963.67
CA-HepPh	32.78	42.32	34.28	8.92	**0.00**	17546.35
CA-HepTh	**0.52**	13.57	1.37	8.11	1.37	2048.00
$l = 1/3$						
CA-AstroPh	**0.04**	1.71	2.68	33.22	22.86	642.29
CA-CondMat	0.74	5.15	**0.00**	19.37	12.28	688.31
CA-GrQc	2.33	5.26	**0.37**	23.40	22.48	470.33
CA-HepPh	1.14	1.48	**0.27**	46.30	41.08	534.72
CA-HepTh	**0.00**	5.28	1.35	16.77	14.49	563.90
$l = 1/2$						
CA-AstroPh	**0.08**	0.96	1.75	25.52	24.92	354.51
CA-CondMat	**0.58**	2.22	0.66	17.69	17.20	423.71
CA-GrQc	1.27	1.44	**0.16**	18.88	21.77	271.08
CA-HepPh	0.34	1.11	**0.23**	30.19	31.16	275.05
CA-HepTh	**0.00**	4.39	1.17	16.63	19.28	365.34

Table 6.16: Gap (%) w.r.t. best observed, SNAP networks, Experiment 2, linear costs. Best result per row in **bold**.

Graph	SA	LS	LS+R	CInf	Thr
$l = 0$					
Cornell5	234.87	203.89	217.00	20.16	**0.00**
JohnsHopkins	51.32	61.25	62.34	13.71	**0.00**
Penn94	220.85	249.57	220.24	23.47	**0.00**
Rice31	87.78	109.09	112.07	14.55	**0.00**
WashU32	227.52	278.45	271.39	10.64	**0.00**
$l = 1/3$					
Cornell5	**0.00**	1.76	2.45	52.45	39.25
JohnsHopkins	**0.14**	1.42	2.39	54.19	43.44
Penn94	**0.00**	3.29	3.05	48.15	34.37
Rice31	**0.00**	1.67	2.35	56.55	43.17
WashU32	**0.00**	1.39	3.14	54.38	40.04
$l = 1/2$					
Cornell5	**0.00**	1.56	1.89	29.77	28.38
JohnsHopkins	**0.00**	0.58	1.21	30.04	29.80
Penn94	**0.13**	1.13	1.36	26.53	24.80
Rice31	**0.00**	0.66	1.57	31.30	27.66
WashU32	**0.00**	1.28	2.29	31.00	28.52

Table 6.17: Gap (%) w.r.t. best observed, Facebook networks, Experiment 2, linear costs. Best result per row in **bold**.

where `cinf` dominates (0–3.19%). For $l = 0$, the failure is amplified on both families: SA reaches 56% on the densest SNAP graphs and 302–618% on Facebook, the largest gaps in the entire study.

Experiment 3: Simple/Targeting

In Experiment 3, weights are uniform and costs are indicator functions. The gap is computed w.r.t. the best observed solution across all algorithms.

Graph	SA	LS	LS+R	CInf	Thr	DOT
$l = 0$						
CA-AstroPh	0.45	**0.00**	1.82	9.02	25.06	341.39
CA-CondMat	0.30	**0.05**	1.35	26.68	43.97	222.09
CA-GrQc	0.20	**0.17**	1.27	27.86	36.97	128.65
CA-HepPh	0.73	**0.07**	1.67	20.85	36.64	249.91
CA-HepTh	**0.19**	0.27	1.62	28.39	42.09	162.57
$l = 1/3$						
CA-AstroPh	0.26	**0.00**	1.92	27.16	62.44	145.33
CA-CondMat	**0.09**	0.20	1.25	30.34	56.80	97.01
CA-GrQc	**0.11**	0.15	0.83	23.87	41.17	59.56
CA-HepPh	**0.07**	0.23	1.32	27.06	54.28	107.27
CA-HepTh	0.21	**0.00**	1.18	26.38	51.68	74.85
$l = 1/2$						
CA-AstroPh	0.12	**0.00**	1.74	24.94	55.53	86.90
CA-CondMat	**0.14**	0.17	0.81	22.71	46.42	54.21
CA-GrQc	0.26	**0.00**	0.51	16.36	33.90	30.68
CA-HepPh	**0.00**	0.31	1.24	21.70	47.55	61.20
CA-HepTh	0.14	**0.03**	0.62	19.08	42.25	41.16

Table 6.18: Gap (%) w.r.t. best observed, SNAP networks, Experiment 3, homogeneous costs. Best result per row in **bold**.

Graph	SA	LS	LS+R	CInf	Thr	DOT
$l = 0$						
Cornell5	169.66	163.76	162.91	**0.00**	135.80	1679.60
JohnsHopkins	138.43	131.10	131.87	**0.00**	112.44	1508.02
Penn94	145.36	140.39	140.34	**0.00**	122.87	1642.85
Rice31	213.06	205.21	212.27	**0.00**	148.87	2019.77
WashU32	174.71	171.17	171.11	**0.00**	145.28	1682.51
$l = 1/3$						
Cornell5	8.35	7.91	7.90	**0.00**	115.72	210.26
JohnsHopkins	2.78	2.88	4.53	**0.72**	115.07	207.69
Penn94	4.82	3.83	4.33	**0.00**	115.07	217.10
Rice31	8.45	8.60	10.91	**0.00**	106.03	213.92
WashU32	9.11	9.17	8.21	**0.00**	106.54	206.58
$l = 1/2$						
Cornell5	0.61	**0.00**	0.88	3.08	85.59	111.08
JohnsHopkins	0.53	**0.19**	1.36	5.48	93.54	119.28
Penn94	0.23	**0.00**	0.55	7.37	92.33	118.41
Rice31	1.35	**1.23**	3.44	0.94	81.28	114.86
WashU32	0.46	**0.19**	1.73	1.87	80.80	108.07

Table 6.19: Gap (%) w.r.t. best observed, Facebook networks, Experiment 3, homogeneous costs. Best result per row in **bold**.

Under homogeneous targeting (Table 6.18 and Table 6.19), on SNAP SA and LS achieve near-optimal results across all regimes including $l = 0$, with SA incurring 0.19–0.73% and LS 0.00–0.27%. For $l \in \{1/3, 1/2\}$ both drop below 0.26%. `cinf` is the best greedy heuristic at 9–28% for $l \in \{0, 1/3, 1/2\}$, `cinf` achieves 0% on all graphs while SA degrades to 0.61–52.01%.

On Facebook, performance is strongly threshold-dependent. For $l = 1/2$, SA and LS are near-optimal (0.23–1.35% and 0.00–1.23% respectively). For $l = 1/3$, `cinf` dominates (0–0.72%) while SA incurs 2.78–9.11%. For $l = 0$, all metaheuristics collapse to 131–213% while `cinf` achieves 0% on all graphs; DOT reaches 1508–2020%. The contrast with SNAP is stark and consistent with the budget limitation hypothesis: indicator costs on dense graphs produce a nearly flat landscape that $1000\,|V|$ iterations cannot navigate.

Under heterogeneous costs (Tables A.12 and A.13), SA achieves 0% on all SNAP graphs for $l \in \{1/3, 1/2\}$ and 0–1.30% for $l = 0$. On Facebook, `cinf` dominates throughout (0–0.65%) while SA incurs 0.98–28.64% for $l \in \{1/3, 1/2\}$ and 168–289% for $l = 0$.

6.5 Discussion

Simulated annealing emerges as the strongest overall method among those tested, and to the best of our knowledge the first capable of handling effectively any combination of edge weights and intervention costs. Its performance is however sensitive to the underlying graph topology. The experiments reveal that structural irregularity, rather than density per se, is the primary driver of difficulty: complete graphs, despite being the densest family considered, respond very well to SA across all threshold regimes, while irregular high-degree graphs such as the Facebook networks pose a significantly harder optimization problem.

On synthetic instances with $l \in \{1/3, 1/2\}$, SA is never beaten by any competing method across all topologies and cost structures, consistently achieving gaps below 1% w.r.t. the exact optimum or best observed solution. This holds for both simple and weighted graphs, and for linear, mixed, and indicator cost functions, making SA the only method in our comparison capable of handling the full generality of the LCIP effectively. LS and LS+R follow at a distance, with reheating providing occasional improvements on sparse irregular topologies but no systematic advantage over SA.

The cases $l = 0$ and majority threshold are harder for all metaheuristics. Under very low thresholds, a small number of correctly targeted nodes is sufficient to trigger the entire cascade: as a consequence, most transpositions in the permutation space are cost-neutral and the optimization landscape is nearly flat. The majority threshold produces a qualitatively different inversion: the combinatorial structure

of the problem shifts in a way that favors greedy heuristics such as `cinf` and `thr`, which achieve near-zero gaps while SA becomes unreliable. Both failure modes are amplified by graph density and nonlinear costs.

On real-world networks, results are consistent with the synthetic picture for $l \in \{1/3, 1/2\}$. For weighted and targeting experiments, SA achieves near-optimal performance on SNAP graphs, in line with synthetic instances of comparable structure. For the homogeneous weight with linear costs experiment, SA closely matches `ginf`, a heuristic that exploits the specific structure of the LCIP with linear costs. On Facebook graphs, performance degrades across all experiments, most severely for $l = 0$ and targeting costs. This degradation is consistent with the budget limitation: with $N_{\text{iter}} = 1000\,|V|$, SA receives far fewer iterations per node than under the synthetic budget of $40\,n^2$, and the results reported here should be interpreted as a lower bound on SA performance under unconstrained computation.

All experiments were conducted using a Python implementation, which imposed significant runtime constraints and makes direct comparison with methods such as the branch-and-cut of [12], implemented in C, inadvisable. We believe that a reimplementation in a compiled language would substantially reduce execution time, enabling iteration budgets scaling as n^2 and, consequently, improved results particularly on the large dense graphs where the current budget is most constraining.

Conclusions

This thesis has studied the LTM intervention problem from an algorithmic standpoint, deriving exact polynomial-time algorithms for structured graph families and designing effective metaheuristics for the general case. On complete graphs, we derived an $O(n \log n)$ WTSS algorithm, reduced the general-cost LCIP to minimum-weight perfect matching solvable in $O(n^3)$, and obtained a novel $O(T \cdot n^4)$ dynamic programming algorithm for the time-constrained variant. For general graphs, we exploit the combinatorial reformulation to solve complete bipartite and multipartite graphs, generalized the dynamic programming approach of Günnec to arbitrary weights and costs, and derived linear-time exact algorithms for paths and cycles. On the metaheuristic side, we formulated SA as a Metropolis–Hastings chain on $\mathcal{S}_n$, derived a mixing time bound, and confirmed experimentally that SA delivers strong performance, achieving optimality gaps below 1% across all graph families and cost structures, and consistently ranked as the best method among those tested.

Several directions remain open. On the exact algorithmic side, the general LCIP and the WTSS on complete bipartite graphs remain open, though their structural regularity suggests that reductions analogous to those of Chapter 2 may yield efficient exact algorithms. The transition from tractability to intractability as graph structure grows more complex is not yet understood. Starting from the low-degree topologies treated in this chapter, it remains open under what conditions the LCIP becomes hard as one moves to denser or more irregular graphs: for instance, whether the problem is solvable in polynomial time on grid graphs under uniform or heterogeneous cost and weight assumptions, or at what degree of cycle complexity or irregularity NP-hardness first appears.

On the metaheuristic side, our implementation constrained the iteration budget; whether a more efficient implementation, allowing budget scaling as n^2, would substantially close the remaining gaps is an open experimental question. The mixing time bound derived in Chapter 5 may also admit improvement: the random transposition chain on $\mathcal{S}_n$ is known to exhibit a cutoff phenomenon at $\frac{1}{2}n \log n$ steps [20], and exploiting this sharper estimate in the comparison argument could yield a tighter bound.

Finally, the time-constrained variant of the intervention problem, introduced

in Chapter 2 for complete graphs, raises substantially harder questions on general topologies. This direction opens a new chapter in the study of LTM intervention, where the goal is not only full adoption, but rapid propagation within a prescribed time horizon.

Appendix A

Table of Results

l	n	SA	LS	LS+R	Inf	CInf	Thr	Rnd
0	100	86.86	35.06	91.02	371.47	7.93	**7.58**	2295.61
	200	87.90	77.23	73.93	206.28	**9.25**	10.31	1867.63
	400	309.40	275.13	287.67	502.74	**0.37**	1.35	6412.33
1/3	100	**0.49**	2.66	3.21	70.63	20.45	22.51	160.05
	200	**0.00**	5.22	4.20	84.00	15.84	16.44	173.00
	400	**0.00**	3.58	5.26	86.96	16.33	16.14	197.25
1/2	100	**0.19**	2.59	1.26	58.81	13.32	12.55	92.56
	200	**0.00**	1.98	1.47	53.82	11.19	11.48	97.06
	400	**0.49**	1.20	1.01	63.25	10.50	11.55	109.63
Maj.	100	**0.09**	1.15	1.27	93.17	5.85	6.99	110.82
	200	**0.00**	0.75	0.87	112.56	4.42	5.31	126.11
	400	**0.00**	0.64	0.78	139.41	4.25	4.54	142.13

Table A.1: Gap (%) w.r.t. best observed, K_n, Experiment 2, mixed costs. Best result per row in **bold**.

l	n	SA	LS	LS+R	CInf	Thr	DOT	Rnd
0	100	67.50	**32.91**	76.15	49.29	34.14	6238.76	6366.84
	200	190.66	153.27	186.58	47.03	**47.03**	12803.72	12859.86
	400	392.77	334.18	641.79	38.97	**38.97**	36093.86	36234.48
1/3	100	**0.57**	1.56	5.29	10.04	10.04	309.88	379.29
	200	**1.22**	1.70	4.66	27.85	27.06	310.13	388.08
	400	**1.89**	1.89	2.57	15.56	15.56	336.23	400.97
1/2	100	**0.41**	0.51	1.24	16.16	16.16	121.38	170.84
	200	1.00	**0.60**	1.68	12.54	12.54	124.62	163.36
	400	1.06	**0.81**	1.24	6.39	5.19	133.81	173.53

Table A.2: Gap (%) w.r.t. Hungarian exact optimum, K_n, Experiment 3, heterogeneous costs. Best result per row in **bold**.

l	n	SA	LS	LS+R	CInf	Thr	Rnd
0	100	0.36	0.36	**0.00**	33.19	23.63	220.66
	200	**0.00**	0.32	0.95	34.68	27.60	180.10
	400	**0.00**	2.30	2.06	35.32	30.54	206.64
1/3	100	1.33	2.42	**1.29**	14.91	29.72	128.08
	200	**0.10**	1.36	0.90	33.46	32.07	116.23
	400	**0.00**	0.96	1.28	26.78	32.45	123.86
1/2	100	**0.00**	0.26	0.26	24.66	31.98	105.03
	200	**0.06**	0.84	0.24	26.24	28.33	88.57
	400	**0.04**	0.44	0.93	22.24	32.70	95.75
Maj.	100	**0.06**	0.35	2.18	32.88	48.22	272.66
	200	**1.35**	2.19	3.71	47.43	50.34	284.95
	400	**0.43**	1.97	3.76	46.28	43.89	269.37

Table A.3: Gap (%) w.r.t. best observed, T_n, Experiment 2, mixed costs. Best result per row in **bold**.

l	n	SA	LS	LS+R	CInf	Thr	DOT	Rnd
0	100	**0.00**	2.49	1.44	48.50	23.02	57.08	133.01
	200	**0.13**	2.73	2.40	39.03	28.25	72.20	182.35
	400	**0.00**	1.20	3.18	40.76	25.98	74.14	179.23
1/3	100	**0.00**	2.03	0.00	25.52	23.49	24.74	92.59
	200	**0.28**	0.33	0.93	31.34	30.88	23.65	105.48
	400	**0.00**	1.08	0.89	28.74	27.04	26.52	98.86
1/2	100	**0.00**	1.76	0.00	21.61	24.74	6.50	71.34
	200	**0.05**	0.81	0.03	20.80	25.89	7.04	80.02
	400	**0.00**	1.04	0.87	23.50	29.13	8.61	76.25
Maj.	100	**4.80**	20.55	7.59	7.09	7.49	176.19	376.80
	200	**0.40**	2.02	12.84	11.61	8.70	175.78	390.53
	400	**0.00**	12.04	13.85	12.44	12.52	204.14	396.69

Table A.4: Gap (%) w.r.t. best observed, T_n, Experiment 3, heterogeneous costs. Best result per row in **bold**.

l	n	SA	LS	LS+R	CInf	Thr	GInf	Rnd
0	100	**2.67**	7.40	28.03	29.70	32.49	25.00	6135.26
	200	2.12	**1.94**	3.55	48.21	53.64	9.20	2831.78
	400	**0.00**	5.41	14.44	26.27	30.20	4.98	4417.84
1/3	100	**0.02**	1.35	3.30	43.17	42.56	5.27	256.78
	200	**0.00**	1.31	2.03	38.19	42.77	2.77	275.44
	400	**0.00**	2.84	2.15	37.86	42.76	4.77	274.71
1/2	100	**0.04**	0.77	0.97	23.38	25.29	4.56	134.96
	200	**0.00**	0.53	0.60	20.83	26.02	1.63	146.95
	400	**0.00**	0.59	0.91	23.03	27.90	2.60	146.29
Maj.	100	11.44	9.59	**1.64**	7.70	4.27	45.13	314.91
	200	5.24	16.72	13.30	5.33	**0.00**	55.67	467.23
	400	19.16	43.64	26.95	12.78	**0.67**	96.90	610.57

Table A.5: Gap (%) w.r.t. best observed, Watts–Strogatz ($k = 20$), Experiment 1. Best result per row in **bold**.

l	n	SA	LS	LS+R	CInf	Thr	Rnd
0	100	**0.00**	5.36	6.04	12.45	14.52	2064.23
	200	**0.00**	4.84	11.32	27.77	29.47	1554.55
	400	**0.00**	4.53	12.19	22.38	25.01	2354.19
1/3	100	**0.00**	1.85	1.88	19.91	22.59	74.59
	200	**0.00**	2.32	3.35	20.32	20.96	69.02
	400	**0.00**	1.72	2.07	20.53	22.37	74.77
1/2	100	**0.00**	1.30	1.77	15.80	18.20	35.90
	200	**0.00**	1.42	2.15	15.04	17.33	36.84
	400	**0.00**	2.05	1.83	16.12	19.53	39.11
Maj.	100	**0.71**	2.67	2.27	7.32	9.23	48.30
	200	**1.37**	3.45	11.61	14.53	16.70	146.45
	400	42.36	46.39	56.40	**0.15**	4.97	216.22

Table A.6: Gap (%) w.r.t. best observed, Watts–Strogatz ($k = 20$), Experiment 2, linear costs. Best result per row in **bold**.

l	n	SA	LS	LS+R	CInf	Thr	Rnd
0	100	167.65	57.70	259.76	0.40	**0.34**	4249.88
	200	44.58	52.24	41.55	13.03	**0.39**	1171.06
	400	1440.53	930.69	1586.03	2.31	**0.00**	25132.46
1/3	100	**0.00**	6.52	5.51	22.05	19.07	156.71
	200	**0.00**	7.85	7.53	19.45	19.98	153.73
	400	**0.00**	11.52	14.03	10.89	15.46	190.14
1/2	100	**0.00**	4.93	1.54	14.31	18.81	97.71
	200	**0.00**	6.20	3.12	18.33	20.57	100.59
	400	**0.00**	8.04	6.17	8.51	12.36	109.77
Maj.	100	**0.00**	4.98	5.26	9.56	14.77	174.13
	200	12.06	19.85	14.92	**5.94**	7.07	261.57
	400	43.98	77.54	61.99	5.69	**0.09**	547.19

Table A.7: Gap (%) w.r.t. best observed, Watts–Strogatz ($k = 20$), Experiment 2, mixed costs. Best result per row in **bold**.

l	n	SA	LS	LS+R	CInf	Thr	DOT	Rnd
0	100	11.11	15.28	26.64	**10.23**	21.34	375.46	397.47
	200	**2.44**	2.44	6.33	15.00	75.67	281.44	292.89
	400	15.87	19.66	24.79	**19.56**	92.76	462.17	488.20
1/3	100	2.22	**0.00**	5.42	17.29	80.97	69.38	99.17
	200	1.97	**1.06**	4.06	20.95	108.33	71.29	94.55
	400	0.53	**0.00**	3.19	23.14	117.62	79.64	104.93
1/2	100	**0.72**	0.72	1.47	12.38	65.73	30.66	51.11
	200	1.47	**0.38**	4.08	16.53	93.68	39.27	60.75
	400	**0.77**	0.95	2.71	16.83	103.77	41.56	65.32
Maj.	100	1.28	**0.00**	7.64	30.44	55.75	62.11	79.82
	200	**1.31**	4.88	15.94	36.86	62.00	76.36	93.36
	400	7.31	**0.00**	15.36	41.63	68.95	93.82	111.71

Table A.8: Gap (%) w.r.t. best observed, Watts–Strogatz ($k = 20$), Experiment 3, homogeneous costs. Best result per row in **bold**.

l	n	SA	LS	LS+R	CInf	Thr	DOT	Rnd
0	100	61.86	55.12	72.00	15.00	**6.78**	1315.73	1419.57
	200	23.19	39.17	33.73	**9.43**	10.43	931.73	1049.51
	400	14.89	25.89	33.22	10.18	**6.84**	1025.28	1168.28
1/3	100	**0.97**	2.95	6.21	14.65	16.40	204.29	282.37
	200	**0.48**	2.33	5.41	12.07	16.89	190.06	252.95
	400	**0.00**	2.26	7.78	12.08	13.35	193.31	261.01
1/2	100	**1.07**	1.20	7.80	9.04	12.43	96.10	141.81
	200	**0.00**	2.16	3.94	13.32	24.54	98.92	139.78
	400	**0.00**	1.64	2.63	15.65	19.83	99.57	150.42
Maj.	100	**1.86**	6.61	14.01	24.85	26.10	161.72	254.29
	200	13.73	**12.36**	12.56	40.84	40.87	235.78	398.03
	400	16.15	10.27	**6.36**	54.35	30.07	273.96	441.27

Table A.9: Gap (%) w.r.t. best observed, Watts–Strogatz ($k = 20$), Experiment 3, heterogeneous costs. Best result per row in **bold**.

Graph	SA	LS	LS+R	CInf	Thr	Rnd
$l = 0$						
CA-AstroPh	55.96	69.48	65.72	**0.00**	4.20	1169.41
CA-CondMat	**0.00**	8.26	8.31	6.79	7.70	486.91
CA-GrQc	**0.00**	6.28	6.43	14.25	16.35	411.52
CA-HepPh	40.46	48.69	57.07	**0.00**	10.15	1083.51
CA-HepTh	**0.00**	12.40	11.18	16.88	14.72	449.31
$l = 1/3$						
CA-AstroPh	**0.00**	4.68	5.59	4.55	26.65	157.72
CA-CondMat	**0.00**	5.45	5.88	19.20	35.27	166.36
CA-GrQc	**0.00**	4.71	5.14	19.09	30.89	155.71
CA-HepPh	**0.04**	2.41	3.85	0.23	22.86	139.68
CA-HepTh	**0.00**	5.90	7.90	21.21	36.46	169.26
$l = 1/2$						
CA-AstroPh	**0.00**	3.16	3.41	4.88	28.36	97.97
CA-CondMat	**0.00**	4.12	5.46	18.10	38.71	112.97
CA-GrQc	**0.00**	3.27	2.86	15.97	31.60	105.89
CA-HepPh	**0.16**	1.31	4.00	2.09	25.73	87.68
CA-HepTh	**0.00**	4.35	4.56	20.26	38.55	117.21
Maj.						
CA-AstroPh	61.74	73.07	67.99	**0.00**	28.58	425.12
CA-CondMat	28.03	37.94	45.85	**0.00**	16.20	437.49
CA-GrQc	1.68	7.72	7.72	**0.56**	11.37	256.67
CA-HepPh	40.92	53.42	53.02	**0.00**	21.49	320.45
CA-HepTh	9.89	18.18	20.37	**0.00**	21.08	372.56

Table A.10: Gap (%) w.r.t. best observed, SNAP networks, Experiment 2, mixed costs. Best result per row in **bold**.

Graph	SA	LS	LS+R	CInf	Thr
$l = 0$					
Cornell5	572.74	553.90	576.64	1.77	**1.12**
JohnsHopkins	618.26	683.02	630.65	**0.00**	6.14
Penn94	456.47	467.69	463.73	3.19	**0.00**
Rice31	301.53	302.58	322.62	**0.00**	7.49
WashU32	393.96	372.84	406.64	**4.11**	6.53
$l = 1/3$					
Cornell5	2.85	7.86	8.86	**0.00**	16.12
JohnsHopkins	4.35	7.81	8.96	**0.00**	16.26
Penn94	2.85	6.39	8.08	**0.00**	17.66
Rice31	3.19	5.61	6.39	**0.00**	12.97
WashU32	2.14	4.96	8.06	**0.00**	13.52
$l = 1/2$					
Cornell5	0.97	4.43	4.87	**0.15**	16.76
JohnsHopkins	1.09	3.63	4.76	**0.09**	16.19
Penn94	**0.00**	3.22	3.74	0.01	18.38
Rice31	0.73	3.48	4.27	**0.63**	13.33
WashU32	0.45	3.71	5.08	**0.10**	14.92

Table A.11: Gap (%) w.r.t. best observed, Facebook networks, Experiment 2, mixed costs. Best result per row in **bold**.

Graph	SA	LS	LS+R	CInf	Thr	DOT
$l = 0$						
CA-AstroPh	**1.30**	4.77	5.65	2.20	5.53	417.17
CA-CondMat	**0.00**	3.01	5.46	20.92	27.45	259.88
CA-GrQc	**0.00**	3.80	5.02	27.26	26.56	157.85
CA-HepPh	**0.00**	4.49	6.70	13.41	18.93	295.74
CA-HepTh	**0.00**	3.75	5.47	27.08	30.80	182.68
$l = 1/3$						
CA-AstroPh	**0.00**	4.01	5.99	17.11	46.69	176.58
CA-CondMat	**0.00**	3.79	5.16	26.97	49.18	115.83
CA-GrQc	**0.00**	3.01	4.87	24.13	37.03	73.75
CA-HepPh	**0.00**	4.52	5.98	22.20	49.83	132.24
CA-HepTh	**0.00**	3.97	4.84	28.02	44.60	84.97
$l = 1/2$						
CA-AstroPh	**0.00**	3.27	4.86	20.40	52.74	103.40
CA-CondMat	**0.00**	3.62	4.55	24.70	47.65	66.91
CA-GrQc	**0.00**	3.23	4.55	21.26	35.45	41.44
CA-HepPh	**0.00**	3.32	4.55	21.00	49.94	74.37
CA-HepTh	**0.00**	3.40	4.19	23.48	44.02	47.85
Maj.						
CA-AstroPh			*no data*			
CA-CondMat	48.58	56.41	57.83	**0.00**	45.30	284.86
CA-GrQc	13.25	16.38	20.58	**0.00**	13.82	138.11
CA-HepPh	51.16	58.78	61.60	**0.00**	65.25	290.39
CA-HepTh	32.32	37.50	41.92	**0.00**	28.52	239.15

Table A.12: Gap (%) w.r.t. best observed, SNAP networks, Experiment 3, heterogeneous costs. Best result per row in **bold**.

Graph	SA	LS	LS+R	CInf	Thr	DOT
$l = 0$						
Cornell5	195.80	206.85	208.01	**0.00**	16.56	2493.05
JohnsHopkins	167.64	178.47	174.04	**0.61**	5.61	2218.99
Penn94	176.71	186.05	186.28	**0.00**	16.83	2390.00
Rice31	288.83	304.69	293.78	**0.00**	14.81	3377.51
WashU32	199.43	219.75	213.92	**0.00**	15.00	2508.22
$l = 1/3$						
Cornell5	28.60	33.39	35.09	**0.00**	49.60	367.90
JohnsHopkins	19.79	26.16	28.03	**0.00**	43.00	349.79
Penn94	23.47	28.20	29.46	**0.00**	49.79	364.34
Rice31	28.64	30.96	35.62	**0.00**	41.43	365.36
WashU32	27.98	32.24	33.35	**0.00**	46.91	359.63
$l = 1/2$						
Cornell5	4.95	8.23	10.06	**0.00**	44.30	163.43
JohnsHopkins	0.98	4.80	5.97	**0.65**	45.89	160.04
Penn94			*no data*			
Rice31	5.59	9.51	9.78	**0.00**	44.67	169.77
WashU32	5.58	7.97	10.46	**0.00**	43.11	162.64

Table A.13: Gap (%) w.r.t. best observed, Facebook networks, Experiment 3, heterogeneous costs. Best result per row in **bold**.

Bibliography

[1] Mark Granovetter. «Threshold Models of Collective Behavior». In: *American Journal of Sociology* 83.6 (1978), pp. 1420–1443.

[2] Pedro Domingos and Matthew Richardson. «Mining the Network Value of Customers». In: *Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2001, pp. 57–66.

[3] Sandra González-Bailón et al. «The Dynamics of Protest Recruitment through an Online Network». In: *Scientific Reports* 1 (2011), p. 197.

[4] H. Peyton Young. «Innovation Diffusion in Heterogeneous Populations: Contagion, Social Influence, and Social Learning». In: *American Economic Review* 99.5 (2009), pp. 1899–1924.

[5] David Kempe, Jon Kleinberg, and Éva Tardos. «Maximizing the Spread of Influence through a Social Network». In: *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2003, pp. 137–146.

[6] Ning Chen. «On the approximability of influence in social networks». In: *SIAM Journal on Discrete Mathematics* 23.3 (2009), pp. 1400–1415.

[7] Wei Chen, Yifei Yuan, and Li Zhang. «Scalable Influence Maximization in Social Networks under the Linear Threshold Model». In: *Proceedings of the 10th IEEE International Conference on Data Mining*. 2010, pp. 88–97.

[8] Ferdinando Cicalese et al. «Spread of influence in weighted networks under time and budget constraints». In: *Theoretical Computer Science* 586 (2015). Fun with Algorithms, pp. 40–58. ISSN: 0304-3975.

[9] Giacomo Como, Stéphane Durand, and Fabio Fagnani. «Optimal Targeting in Super-Modular Games». In: *IEEE Transactions on Automatic Control* 67.12 (2022), pp. 6366–6380.

[10] Giacomo Como, Stéphane Durand, and Fabio Fagnani. «Intervention problems in the Linear Threshold Model: a general formulation and new results». Working paper. 2025.

[11] Gennaro Cordasco et al. «Optimizing Spread of Influence in Social Networks via Partial Incentives». In: *Structural Information and Communication Complexity (SIROCCO 2015)*. Vol. 9439. Lecture Notes in Computer Science. Springer, 2015, pp. 119–134.

[12] Dilek Günneç, S. Raghavan, and Rui Zhang. «Least-Cost Influence Maximization on Social Networks». In: *INFORMS Journal on Computing* 32.2 (2019), pp. 289–302.

[13] S. Raghavan and Rui Zhang. «Weighted target set selection on trees and cycles». In: *Networks* 77.4 (2021), pp. 587–609.

[14] Dilek Günneç, S. Raghavan, and Rui Zhang. «A branch-and-cut approach for the least cost influence problem on social networks». In: *Networks* 76.1 (2020), pp. 84–105.

[15] Matteo Fischetti et al. «Least cost influence propagation in (social) networks». In: *Mathematical Programming. Series B* 170.2 (2018), pp. 293–325.

[16] Gennaro Cordasco et al. *Time-Bounded Influence Diffusion with Incentives*. arXiv preprint arXiv:1807.06921. 2018.

[17] Harold W. Kuhn. «The Hungarian Method for the Assignment Problem». In: *Naval Research Logistics Quarterly* 2.1–2 (1955), pp. 83–97.

[18] James Munkres. «Algorithms for the Assignment and Transportation Problems». In: *Journal of the Society for Industrial and Applied Mathematics* 5.1 (1957), pp. 32–38.

[19] Persi Diaconis and Mehrdad Shahshahani. «Generating a random permutation with random transpositions». In: *Z. Wahrscheinlichkeitstheorie verw. Gebiete* 57 (1981), pp. 159–179.

[20] Nathanaël Berestycki. *Mixing Times of Markov Chains: Techniques and Examples. A Crossroad between Probability, Analysis and Geometry*. Lecture notes, University of Cambridge. 2016.

[21] James R. Norris. *Markov Chains*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 1997.

[22] Laurent Saloff-Coste. *Random Walks on Finite Groups and Mixing Times*. Lecture notes, available at Cornell University. 2004.

[23] Vincenzo Auletta et al. «Convergence to equilibrium of logit dynamics for strategic games». In: *Proceedings of the 23rd ACM Symposium on Parallelism in Algorithms and Architectures*. New York, NY, USA: Association for Computing Machinery, 2011, pp. 197–206.

[24] Duncan J. Watts and Steven H. Strogatz. «Collective dynamics of 'small-world' networks». In: *Nature* 393.6684 (1998), pp. 440–442.

[25] Jure Leskovec and Andrej Krevl. *SNAP Datasets: Stanford Large Network Dataset Collection*. `http://snap.stanford.edu/data`. June 2014.

[26] Ryan A. Rossi and Nesreen K. Ahmed. *The Network Data Repository with Interactive Graph Analytics and Visualization*. `https://networkrepository.com`. 2015.

Ringraziamenti

Desidero ringraziare il Professor Como e il Professor Fagnani per la guida durante questo lavoro, per la loro costante disponibilità e per i tanti confronti stimolanti che hanno dato forma sia alla tesi che al mio modo di vedere la ricerca.

Sono grato al Politecnico di Torino per questi anni di eccellente formazione, e ad HPC@PoliTO per aver fornito le risorse computazionali che hanno reso possibile la parte sperimentale di questo lavoro.

Un ringraziamento profondo va alla mia famiglia, soprattutto per aver creduto nella mia formazione e nella mia crescita non a parole ma con i fatti — questo è il regalo più grande che abbia ricevuto. A mia madre, per aver inconsapevolmente definito il mio carattere e indirizzato verso questo percorso. A mio padre e mia sorella, per credere in me molto più di quanto faccia io.

Voglio ringraziare le persone che mi hanno tenuto compagnia in questi anni e che mi hanno sempre offerto nuovi spunti, facendomi uscire dalla mia testa. Francesco e Joe, per il sostegno reciproco e per aver condiviso questo percorso passo dopo passo. Mario, per i momenti di spensieratezza e divertimento di questi anni. Marco, per essere stato un punto fisso nei miei anni a Torino e per le tante partite a scacchi. Giuseppe, per riportarmi sempre alla concretezza e per la sua visione lucida. Domenico, non solo per l'indispensabile supporto tecnico, ma anche per essermi stato vicino nonostante i mille chilometri di distanza, e per avermi costantemente aperto la mente a modi di pensare che prima trascuravo.

Ringrazio inoltre Domenico, Francesco e Claudio per un'ulteriore revisione della tesi.