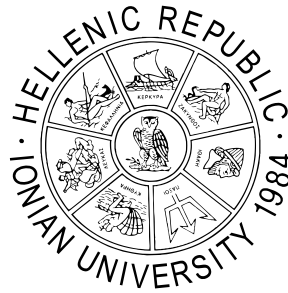


IONIAN UNIVERSITY

**Faculty of Information Science
and Informatics**

Department of Informatics



Master Thesis

Research Directions in Informatics

**Study of Cascading Behaviour in Various Network
Topologies**

Andreana Styliou

September 10, 2021

Supervisor

Michael Stefanidakis, *Associate Professor,*
Ionian University

Advisory Committee Members

Michael Stefanidakis, *Associate Professor,*
Ionian University

Konstantinos Oikonomou, *Professor,*
Ionian University

Christoforos Ntantogian, *Assistant Professor,*
Ionian University

To my mother, Ina.

Abstract

In the world of social networks, the diffusion of innovation, ideas and technologies, is studied under Cascade Models, which captures the way in which individuals adopt these new behaviours by observing the actions of their peers in their social circle. In network theory, the term information cascade describes a process, in which the spread of information is modelled as a new behaviour entering a network. This behaviour is adopted by the network nodes based on the number of nodes already following the new behaviour.

Meanwhile, information dissemination in traditional dynamic routing protocols is accomplished using flooding algorithms, which guarantees that the knowledge regarding changes in the topology is received by all nodes participating in the routing process. However, the overhead produced by blindly flooding the network with messages contributes to the storm problem, and many variations to classic routing propose more resource-conservative techniques for the dissemination of routing updates.

The main idea of this Thesis focuses on incorporating a variation of a cascade model, namely the Linear Threshold Model, in a dynamic routing algorithm, in order to minimise overhead. The Linear Threshold Model captures the influence node's link differently, and also assigns a different adoption threshold to each node. Thus, these metrics can represent the importance of the links and the nodes to the routing process, when derived based on traffic and structure related properties, respectively. Consequently, link loads are used to derive the weights of the model, while degree centrality is used to define individual adoption thresholds. Link load indicates the frequency of communication with the node on the end of that link and the degree to which the node participates in the routing process through that link. Degree centrality is used based on the idea that central nodes need to be informed about the network, due to them participating in the routing of network traffic with

high probability.

This cascade-based approach is experimentally evaluated in a simulation environment across random geometric graphs of varying sizes and densities. Its performance is evaluated with respect to its ability to accurately route traffic after a link in the network fails, and is compared to the performance achieved by informing all (through flooding) and none of the network’s nodes, which correspond to the best and worst-case scenarios, respectively. Two different traffic distributions are generated prior to testing the cascade model, uniform traffic distribution and Poisson traffic distribution. Additionally, two different link weight assignment methods are considered under these traffic approaches, with one of them taking into account only load from incoming traffic, and the second one both incoming and outgoing traffic. Ideally, the desired model would maximise routing accuracy, while minimising overhead in the dissemination of routing updates.

Through experimentation, it is shown that the cascade-based model is highly capable of spreading routing updates to few, but important nodes. In particular, it achieves close to 100% routing accuracy, delivering almost all of the traffic as effectively as flooding and significantly outperforming the worst-case scenario. In contrast, this close-to-optimal routing accuracy is achieved by covering a fairly small fraction of the network’s nodes, which is inversely proportional to the network’s size. The results are consistent with both uniform and Poisson traffic patterns in the network. However, the two link weight assignment methods do not differ much in their performance across the experiments.

Based on the acquired results, it is concluded that the proposed approach incorporating cascading behaviours to spread routing updates accomplishes its intended goal of minimising overhead without sacrificing routing accuracy, under the considered scenarios. Nonetheless, more thorough experimentation needs to take place under more diverse scenarios, before these results can be generalised. In particular, more extreme traffic distributions could highlight differences between the two link weight assignment methods, or indicate that more sophisticated adoption thresholds need to be considered. Last but not least, the structural properties that different topologies would present could likely affect performance, which is not investigated here.

Περίληψη

Η διάδοση των καινοτομιών, ιδεών και τεχνολογικών τάσεων έχει μελετηθεί στην επιστήμη των κοινωνικών δικτύων, χρησιμοποιώντας τα λεγόμενα μοντέλα cascade. Η λογική τους έχει να κάνει με την επιρροή που ασκούν οι διάφοροι κοινωνικοί δεσμοί των ατόμων στις επιλογές τους σχετικά με μια νέα συμπεριφορά που εξαπλώνεται. Τα μοντέλα cascade μπορούν επίσης να εφαρμοστούν στη μελέτη των δικτύων υπολογιστών. Ουσιαστικά, μια νέα πληροφορία που εισέρχεται στο δίκτυο, μοντελοποιείται ως συμπεριφορά η οποία υιοθετείται από τους κόμβους, βάσει ενός κατώφλιου. Το κατώφλι αυτό προκύπτει από τον αριθμό των γειτόνων του κόμβου που ακολουθούν την καινούργια συμπεριφορά.

Η διάδοση της πληροφορίας στα δίκτυα γενικότερα, είναι μια βασική διαδικασία, η οποία χρησιμοποιείται από πλήθος πρωτόκολλων και αλγορίθμων. Στα δυναμικά πρωτόκολλα δρομολόγησης, η διάδοση της πληροφορίας που αφορά στις αλλαγές της τοπολογίας, πραγματοποιείται με τη βοήθεια του αλγορίθμου flooding. Ο αλγόριθμος αυτός, αν και αποδοτικός ως προς τον σκοπό του, ‘πλημμυρίζει’ το δίκτυο με μεγάλο αριθμό μηνυμάτων, γεγονός που οφείλεται στο ότι ο κάθε κόμβος του δικτύου, αποστέλλει αντίγραφα του μηνύματος σε όλους τους γείτονές του. Για τον λόγο αυτό, κατά καιρούς έχουν προταθεί εναλλακτικές για τη διάδοση πληροφοριών που αφορούν αλλαγές στην τοπολογία με αποδοτικότερους τρόπους.

Η παρούσα διπλωματική προτείνει την αξιοποίηση μιας παραλλαγής του Linear Threshold Model των cascades, και την εφαρμογή της στη διάχυση των αλλαγών της τοπολογίας για την εξυπηρέτηση πρωτοκόλλων δρομολόγησης, με σκοπό την ελαχιστοποίηση των περιττών μηνυμάτων που παράγονται λόγω του flooding. Ένα από τα χαρακτηριστικά του μοντέλου είναι η δυνατότητά του να διαφοροποιήσει τα βάρη των ζεύξεων ενός κόμβου, σύμφωνα με την επιρροή που ασκούν οι γείτονές του στην απόφασή του, αλλά και να ορίσει διαφορετικό κατώφλι αποδοχής της συμπεριφοράς για κάθε κόμβο. Αυτά τα βάρη των ζεύξεων και τα κατώφλια προσαρμόζονται, ώστε να

γίνουν αντιπροσωπευτικά για τις ανάγκες της δρομολόγησης, με βάση τα μηνύματα και τη δομή του δικτύου. Έτσι, το φορτίο της ζεύξης χρησιμοποιείται για τον υπολογισμό του βάρους, αφού κόμβοι που ανταλλάσσουν συχνότερα μηνύματα μεταξύ τους είναι πιθανότερο να συμμετάσχουν στην διαδικασία της δρομολόγησης ενός του άλλου. Όσον αφορά τα κατώφλια για την αποδοχή της συμπεριφοράς, η μετρική που χρησιμοποιείται προκύπτει βάσει του degree centrality, που δηλώνει το βαθμό συνδεσιμότητας του κόμβου, και βασίζεται στην ιδέα ότι οι κεντρικοί κόμβοι συμμετέχουν στην δρομολόγηση σε μεγαλύτερο βαθμό.

Η αξιολόγηση του μοντέλου πραγματοποιείται πειραματικά σε περιβάλλον προσομοίωσης δικτύων, για διάφορους τύπους τυχαίων γεωμετρικών γραφών, αλλάζοντας το μέγεθος του δικτύου και την ακτίνα συνδεσιμότητάς του. Η αξιολόγηση επικεντρώνεται στη δυνατότητα του δικτύου να ανακάμψει έπειτα από χαμένες ζεύξεις, η οποία υποδηλώνεται από το ποσοστό μηνυμάτων που χάνονται, λόγω της μη αποδοχής της καινούργιας συμπεριφοράς από κάποιους κόμβους. Το ποσοστό αυτό συγκρίνεται με την διάδοση των αλλαγών της τοπολογίας χρησιμοποιώντας το flooding (καλύτερη περίπτωση), αλλά και τη μη-διάδοση αλλαγών με κανέναν τρόπο (χειρότερη περίπτωση). Δύο κατανομές βάσει των οποίων παράγονται τα μηνύματα στο δίκτυο εξετάζονται στα πειράματα, ομοιογενής κατανομή και κατανομή Poisson. Για τον προσδιορισμό των βαρών λαμβάνονται υπόψιν δύο μέθοδοι: (α) χρησιμοποιώντας το φορτίο των εισερχόμενων μηνυμάτων μόνο, (β) χρησιμοποιώντας τα εισερχόμενα σε συνδυασμό με τα εξερχόμενα. Ο σκοπός της προσέγγισης είναι η μεγιστοποίηση της ακρίβειας του πρωτοκόλλου δρομολόγησης και παράλληλα, η ελαχιστοποίηση των παραγόμενων μηνυμάτων.

Τα πειράματα δείχνουν πως το μοντέλο τελικά καταφέρνει να διαχύσει την απαιτούμενη πληροφορία σε κόμβους επαρκείς για την επίτευξη σχεδόν βέλτιστης δρομολόγησης. Συγκεκριμένα, επιτυγχάνει σχεδόν 100% ακρίβεια, όσο δηλαδή και το flooding, και πολύ καλύτερα της χειρότερης περίπτωσης. Το πιο σημαντικό όμως είναι, πως αυτό το επιτυγχάνει με την κάλυψη σχετικά μικρού ποσοστού των κόμβων, το οποίο όμως συμβάλλει σημαντικά στη διαδικασία της δρομολόγησης και επιπλέον, είναι αντιστρόφως ανάλογο του μεγέθους του δικτύου. Τα αποτελέσματα ισχύουν και για τα δύο είδη κατανομής μηνυμάτων και μάλιστα, ανεξάρτητα από τη μέθοδο που επιλέγεται για τον προσδιορισμό των βαρών των ζεύξεων για τα συγκεκριμένα πειράματα.

Βάσει των εξαγόμενων αποτελεσμάτων, συμπεραίνεται ότι η χρήση του μοντέλου των cascades είναι κατάλληλη για την εξάπλωση των αλλαγών της τοπολογίας σε πρω-

τόκολλα δρομολόγησης, και πετυχαίνει υψηλή ακρίβεια με παράλληλα χαμηλό κόστος. Παρά ταύτα, υπάρχουν περιθώρια βελτίωσης και χρειάζεται να δοκιμαστεί σε πλήθος ακόμα δικτύων και μοτίβων ανταλλαγής μηνυμάτων μεταξύ των κόμβων, για να μπορέσει κανείς να βγάλει ένα πιο σίγουρο συμπέρασμα. Για παράδειγμα, ύπο άλλες συνθήκες, οι διαφορετικοί τρόποι υπολογισμού των βαρών των ζεύξεων μπορεί να διαφοροποιηθούν παραπάνω ως προς την αποτελεσματικότητά τους. Τέλος, η δομή των γράφων μπορεί να διαδραματίζει σημαντικό ρόλο στην απόδοση, πράγμα το οποίο δεν λαμβάνεται υπόψιν στην παρούσα εργασία.

Acknowledgments

A big thank you to my family, who supported and encouraged me, always. My mother, Ina, my father, Andreas, and all four of my grandparents, who can't wait to see me getting my Master's degree.

I also thank a very special person to me, Spiros. You were always the first to tell me how much you believed in me. Thank you for your patience, you made things much easier.

Of course I thank my sister and my friends from Kozani. Even though we were cities apart, we always managed to make time for each other. They knew what I was doing was important to me, and always cared about my progress and inspired me through their support. For that I am very thankful.

I would also like to thank my supervisors, Dr. Michael Stefanidakis, Dr. Konstantinos Oikonomou, and Dr. Chistoforos Ntantogian for all their valuable support and advice. A special thank you to Dr. Oikonomou, for encouraging me to pursue my dreams, and giving me the opportunity to experience, first hand, the highs and lows of research. Sometimes this journey was difficult, but always exciting.

Last, I want to thank all my friends and colleagues from my department, with whom we experienced this journey together. Alex, Stelios, Kostas, Korina, Spiros, and John. Each and every one of you, in your own way, was an irreplaceable part of our "lab team", as we call it, and we had much fun together and learned many exciting things. Now that this journey has come to its end, who knows what the future holds? Every one will follow their own path, and I know I will miss everyone dearly. But one thing is certain: this is only the beginning!

Contents

A	Introduction	1
A.1	Information Cascades	2
A.2	Information Dissemination & Routing	3
A.3	Past Related Work	4
A.4	Contribution	5
A.5	Thesis Structure	6
B	Theoretic Background	9
B.1	Cascades	9
B.1.1	Network Coordination Games	9
B.1.2	Linear Threshold Model	13
B.2	Routing Protocols	14
B.2.1	Distance-Vector Routing	15
B.2.2	Link-State Routing	19
C	Proposed Approach	23
C.1	Overview	23
C.2	System Model and Assumptions	25
C.3	Cascade Model	27
C.3.1	Progressive	28
C.3.2	Weights	29
C.3.3	Threshold	29
D	Evaluation	31
D.1	Setup	31

D.2	Results	33
D.2.1	Uniform Traffic	33
D.2.2	Poisson Traffic	35
E	Conclusions	39
I	Glossary	47
II	Notations	49
III	Abbreviations	50
	Index	51

List of Figures

B.1	An illustration of the decision making step for node v , based on the number of neighbours following behaviour A and B	12
B.2	A simple graph network.	15
B.3	Distance-vector routing: A graph example along with a matrix that shows the initial distances available to each node in the graph.	17
B.4	Distance-vector routing: Final distances available to each node in the graph.	18
B.5	An example of flooding in a graph: Source node initiates the transmission of flooding messages, and every node that receives it retransmits a copy.	20
B.6	An example of a link failure in large-scale network.	22
C.1	Example of cascading behaviour in a graph.	24
C.2	Example of traffic flow in a graph. The dotted lines represent the number of messages sent from one node to another.	26
C.3	Chain topology: node 0 initiates the cascading of a new behaviour.	28
D.1	The actual delivery ratio of traffic generated according to the uniform distribution for different network sizes and densities when using incoming link load for weight computation (left), and both incoming and outgoing load (right). The dashed line indicates the worst-case performance achieved by not updating any nodes.	34
D.2	Fraction of nodes covered by the model for traffic generated according to the uniform distribution using incoming load for weight computation (left), and both incoming and outgoing load (right).	34

D.3	Expected ratio achieved by the model for traffic generated according to the uniform distribution using incoming load for weight computation (left), and both incoming and outgoing load (right).	35
D.4	The actual delivery ratio of traffic generated according to the Poisson distribution for different network sizes and densities when using incoming link load for weight computation (left), and both incoming and outgoing load (right). The dashed line indicates the worst-case performance achieved by not updating any nodes.	36
D.5	Fraction of nodes covered by the model for traffic generated according to Poisson distribution using incoming load for weight computation (left), and both incoming and outgoing load (right).	36
D.6	Expected ratio for traffic generated according to Poisson distribution achieved by the model using incoming load for weight computation (left), and both incoming and outgoing load (right).	37

List of Tables

<i>B.1 The payoff matrix of the game. If players v and w both choose A, they receive a payoff of q. If they both choose B, they receive a payoff of $q - 1$. If they follow opposite behaviours, they each receive a payoff of 0.</i>	11
<i>B.2 Initial routing table of node A.</i>	18
<i>B.3 Final routing table of node A.</i>	19
<i>C.1 Distance matrix.</i>	26
<i>C.2 From left to right: traffic matrix and load matrix.</i>	27

Chapter A

Introduction

THERE are numerous examples of real life situations in which individuals influence and are influenced by others. Such examples include consumer behaviour, the formation of political views, people's ideology, cultural values and many others. It is far more likely for an individual to adopt a new idea or a new behaviour, despite their own personal beliefs, if they observe a large number of people in their social circle also adopting it. This can be explained by people inferring additional information from other's people individual choices, and therefore public opinion may shift their beliefs in favour of the most prominent behaviour and vice versa.

These concepts of human behaviour are studied by social psychology and sociology, and the social ties that link people together are referred to, in these fields, as social networks. Social networks are used to formally describe and differentiate between the relationships that govern people's behaviour and decision-making, and study the dynamics of these structures as a whole.

An important aspect that is prominent in collective behaviour studies, is that the spreading of new ideas or behaviours often emerges from just a few individuals and as more and more people begin to adopt them, influenced by their peers, the ideas may eventually spread to the population contagiously, taking the dimensions of an epidemic [1]. This spread of ideas is a form of information diffusion. The initial research on this topic started out as early as in the middle of the 20th century, and eventually developed into concrete mathematical models that describe how ideas and behaviours diffuse through a population [2, 3]. The fundamental understanding

of how the success or failure of a new idea can be predicted based on some prior information can, for example, among others, be used to increase the potential success of new products, when entering a market. A widespread approach to modelling information diffusion is proposed under game-theoretic considerations, where each individual has a certain number of friends, and the payoff they receive, increases or decreases from coinciding their behaviour with the behaviours of their friends.

It can be argued that the mechanisms of information diffusion in social networks, can motivate applications in other areas. For instance, routing, congestion control, and resource sharing in computer networks have been modelled under game-theoretic perspectives [4], where agents must decide their best move, from a group of actions, taking into account the possible moves that other agents can also make, in an attempt to maximise their payoff. In cases where the agents have common interests, they play a cooperative game, whereas when their interactions are of conflicting interest, the game is called non-cooperative. An interesting concept in non-cooperative games is the Nash equilibrium, which is used to define their solution. In a Nash equilibrium, every player of the game is playing their best move, and no player has anything to gain by changing their action.

A.1 Information Cascades

There are two main approaches when it comes to modelling diffusion: *probabilistic* and *decision-based*. Probabilistic models suggest the information spreading takes place based on some predefined probability; the nodes are passive and probabilistically pass the information to their neighbours. Because of the randomness that is involved in these models, such approaches are suited for modelling epidemic spreading. On the other hand, in decision-based models, each node comes to face a decision as to whether to activate based on its neighbours' current states. This approach suggests that the nodes are active players of a network game and the outcome of the game, assuming the initial conditions are known, is a deterministic process.

The work of Banerjee [5], was one of the firsts to describe the concept of individuals imitating the choice of others by inferring that other decision makers may have important information that is unknown to the former. It uses the term *herding*, although other work that develops around that time [6, 7] refers to the concept as *information cascade*.

In network theory, the term *information cascade* describes a process, in which the spread of information is modelled as a new behaviour entering a network. This behaviour is adopted by the network nodes based on some factors that influence their decision. Modelling the spread of information is a valuable tool in a wide range of real networks. Apart from social sciences and economics where it is used to determine the spreading mechanisms of new ideas in populations and the adoption of new technologies, simulating cascading behaviour in computer networks may shed light to traditional information dissemination approaches.

A.2 Information Dissemination & Routing

Information dissemination in its pure form, is needed for propagating knowledge in a variety of computer networks' distributed algorithms and techniques. An example of this, is routing, which refers to the process of finding optimum paths for delivering traffic across the network. In small networks with static topologies, pre-computing these paths and assigning the knowledge to each node, can be easily accomplished manually. Large networks with unstable topologies on the other hand, require dynamic routing protocols, which propagate real-time information about changes in the network topology, in order to ensure the accurate construction of the paths by the nodes in a distributed way. This is extremely challenging to achieve correctly, as distributed systems need to operate in sync, and all nodes must have the latest copy of the information.

The most prominent dynamic routing protocols utilise some form of a flooding algorithm, in order to distribute copies of topology information to all nodes. Flooding is an extremely costly approach to information dissemination. Although it guarantees the propagation to every node in the network, depending on the requirements of each system, cost may sometimes be more important than the accurate distribution of information. In a large-scale network, correctly updating the routing tables in a subset of the nodes, may result in an overall accurate flow of traffic, without flooding the network with a plethora of topology-related messages. That is due to fact that because of the topology, certain nodes may play little to no role in the routing processes of other nodes, so the propagation of the routing information is not required to reach them. It could also be the case, that due to the needs of certain applications, traffic in a subset of the network may be prioritised, so ensuring

accurate distribution of information to just this subset would be sufficient.

A.3 Past Related Work

In terms of its application in computer networks, game theory has been used as a theoretical decision making framework for routing, congestion control, and resource sharing [4]. Network routing games have been discussed in [8, 9, 10]. Non-cooperative network games have been applied in order to derive paths in routing [4], where both the source and destination aim to maximise the benefit for themselves, and also compete with other source, destination pairs in the network. Conditions for the existence of Nash equilibrium in non-cooperative routing games, have been derived for wired networks [11, 12, 13], and in [14], non-cooperative routing games are studied in wireless network environments.

Routing schemes that are adaptive to traffic patterns have been proposed in the past to accommodate the need for optimal traffic flow in the network, and also minimising bandwidth usage by routing updates. Mobile ad hoc networks (MANETs) are the most prominent example of a decentralised networks where nodes do not rely on any existing infrastructure. Changes in topology happen rapidly, and it is up to the nodes to determine the best routes to forwarding data, based on dynamic routing algorithms. A very prominent problem in ad hoc networks, in general, is the broadcast storm problem described in [15], which is caused by flooding, and its main three issues are: redundant rebroadcasts, contention, and collisions. Multiple solutions have been proposed in [16] for suppressing the storm caused by the traditional flooding algorithm. In [17], a neighbour coverage-based probabilistic rebroadcast is proposed for reducing routing overhead in MANET. [18] proposes distance and location based techniques for reducing redundant routing packages in vehicular ad hoc networks. In [19], a heuristic, neighbour knowledge-based approach is proposed to select multiple nodes for flooding broadcast messages in MANET environments.

Other proposed approaches for routing in MANETs exploit the nature of wireless communication, by broadcasting routing information to nodes in range, without relying on pre-existing routing paths [20]. Other variations include back-off based broadcast [21, 22], in which certain nodes, that are selected by others to forward the routing packets, use a back-off time depending on the distance in hop count from the destination.

A.4 Contribution

Routing protocols are used to construct the forwarding paths of the routers in networks, and coordinate their communication in events of topology changes, in order to keep these paths up to date. The dissemination of topology changing information, or routing packets, traditionally takes place using some kind of flooding algorithm. The most important disadvantage of flooding algorithms, also pointed out in the literature [15], is the high overhead due to the extremely large amount of messages it produced. Messages are produced by all nodes of the network causing the generation of multiple duplicates.

This thesis proposes an alternative approach to disseminating topology updates in routing protocols, motivated by the Linear Threshold Model used to describe the cascading behaviour in social networks. This model is used for studying the diffusion of innovations, knowledge, and ideas among populations in which individuals are influenced by the choices of their peers. The new idea is seen as a behaviour, that gets adopted by the individuals of the network based on the number of their peers who have also adopted it. Depending on the adoption thresholds, used to describe the susceptibility of each individual to adoption, the initial number of adopters, the structure of the social network, and the degree of influence the peers exerts to one another, the cascading behaviour might get adopted by most or all the individuals in the network, or might stop with only a small number of converted individuals.

In the case of routing, the knowledge of a lost link is considered a behaviour emerging from a node in the network. The weights the nodes assign to their links is derived taking under consideration the traffic flow on that link. This is due to the fact that nodes communicating more frequently with each other, are more likely to participate in the routing process of each other's packages. The adoption threshold of each node is based on its degree centrality, as more central nodes are more likely to route more packets. Nodes receive a message containing the lost link information, and make the decision to adopt the new behaviour if their adoption condition holds true. The goal is to minimise the number of transmitted messages during a topology update, whilst also achieving a good coverage of the nodes that are more interested in the lost link information, and by extension maintaining accurate routing paths.

The evaluation of the proposed approach takes place in random geometric graphs and the algorithms are developed in the OMNeT++ simulation environment for dif-

ferent sizes and densities. The initial routing tables are established and a traffic matrix is constructed for each run. Two different methods are used for the construction of the traffic matrix: uniform traffic distribution and Poisson traffic distribution. The matrix is used for establishing the weights of the links needed for the model. Two different ways of calculating the weights based on traffic are used. After a link goes down, the spreading of the new behaviour is started by the nodes of the affected link. In the end of the cascade, traffic based on the existing traffic matrix is exchanged one more time, in order to evaluate the coverage achieved. The evaluation metrics compare the number of messages successfully delivered based on the proposed model, to the best and worst case scenarios (flooding and no dissemination).

A.5 Thesis Structure

The rest of this thesis is structured as follows. Chapter B covers the theoretical background of this thesis. Section B.1.1 familiarises the reader with the basic, game-theoretic concepts, through a brief definition of game theory and network coordination games. A basic information diffusion game is then modelled in a graph, and its game related properties, such as the adoption threshold, conversion, and contagion threshold are discussed. In B.1.2, a more general approach to cascading behaviour is presented, in which the influences between the nodes are captured in the form of weights. Additionally, in B.2, routing protocols are briefly introduced, and two of the main classes of routing protocols are presented using descriptive examples: distance-vector routing and link-state routing.

The thesis proceeds with Chapter C, in which the proposed approach to incorporate decision-based information diffusion in traditional routing protocols is presented. The main idea is that by disseminating network topology changes, based on a cascading behaviour model, routing can still recover from link failures and some of the overhead of flooding can be avoided. The model uses a modified version of a Linear Threshold Model, which is presented in C.3, along with the weights and adoption thresholds that are derived based on link load and degree centrality, respectively. The intuition behind the selection of these metrics is also discussed.

In Chapter D, the setup of the experiments, the implementation details, and the evaluation metrics of the thesis' approach are presented, and the results obtained are

presented and discussed. Two different approaches to calculating the weight links of the model are considered. It is shown that under the presented conditions, the proposed approach is highly capable of disseminating routing updates to the nodes of the network, in regards to the traffic that can be maintained after the recovery of the network by flooding (best-case scenario), and not updating the network at all (worst-case scenario).

Finally, Chapter E outlines the summary of the achieved results of the thesis, and also draws conclusions. Potential future directions are briefly discussed in the end.

Chapter B

Theoretic Background

THIS chapter lays the theoretic foundation for later describing the proposed approach of this thesis. Section B.1 introduces the terminology and the basic concepts regarding game-theoretic cascade modelling in a computer network. Information diffusion is studied as a game, with nodes being the players that make decisions based on the choices of other nodes in the network. Section B.2 proceeds with describing routing in networks, by using two of the most prominent classes of routing protocols as examples: distance vector and link state.

B.1 Cascades

Cascading behaviour processes are network models that capture the diffusion of information in a population. This information can be interpreted as the spread of new ideas or technological innovations and their applications include viral marketing [23], online blogs [24] and contagion models [25].

B.1.1 Network Coordination Games

When individuals that are connected via a network interact, their behaviour is influenced by the neighbours around them. From a game-theoretic point of view, these interactions can be modelled as payoffs that the individuals receive, depending on how their own choices coincide with those of their neighbours.

Game theory is a mathematical field in which social situations are modelled among decision-making players that can have mutual or conflicting interests. It has

most notably been used in economics, with the most prominent example being that of the competition between companies in trying to enter a new market. In this case, companies are players in a game, and each company is trying to decide its best strategy, taking into account that its competitors could make similar or different moves.

Due to the ever-growing need for efficient communication techniques that could help to resolve the challenges of modern networking, game theory has been used as a tool, in order to incorporate complex decision-making techniques into the communication systems [26]. The various types of network games can be classified into two main classes, depending on whether the payoff of a selected player's action is increasing or decreasing, with respect to the number of neighbours who take the same action. Network games in which a player will increase their payoff, when they select the same action as another player are of *strategic complements* (as opposed to *strategic substitutes*), and the most canonical example of this class is that of *coordination games* [27].

The basic game-theoretic model for studying information diffusion in a network is a simple coordination game. Each node v has certain neighbours, and the payoff of v for adopting the new behaviour that enters the network increases, as more and more of these neighbours adopt it. The key intuition behind the game-theoretic model is that a node will gain more payoff if its neighbours adopt the same behaviour. The following description of this model is based on [1].

Information diffusion game

Consider, for the case of social diffusion, a graph $G = (V, E)$ in which nodes are the individuals in a population, and there exists an edge (v, w) if v and w are neighbours, that is, they interact and their decisions affect each other. This kind of graph is could be considered a social network, and its edges could represent the social ties of the entities within it.

At a certain time instant a new behaviour emerges in the network, and each node has a choice between keeping the existing “old” behaviour, or switching to the “new” one. The “old” behaviour is labelled A , and the “new” behaviour B . There is an incentive for v and w (on each edge (v, w)) to have their behaviours match, that is modelled by a real number q , with $0 < q < 1$.

Table B.1: *The payoff matrix of the game. If players v and w both choose A , they receive a payoff of q . If they both choose B , they receive a payoff of $q - 1$. If they follow opposite behaviours, they each receive a payoff of 0.*

		w	
		A	B
v	A	q	0
	B	0	$1 - q$

Table B.1 numerically demonstrates the mechanics of the game between each pair of nodes v and w that takes place during a decision making step. Now, assuming each node is playing this simple game with each of its neighbours in G , its total payoff is simply the sum of the payoffs from all of those individual games.

To be able to capture the dynamics of these interactions on a more comprehensive scale for node v , its degree needs to be considered. Let d_v denote the degree of node v , that is, the number of neighbours of node v . Now, those neighbours of v that follow behaviour A can be denoted as d_v^A and neighbours that follow behaviour B as d_v^B . Based on the rules of the described game, the total payoff node v would receive from choosing behaviour A is $P^A = qd_v^A + 0 \cdot d_v^B$, and the total payoff from choosing behaviour B is $P^B = 0 \cdot d_v^A + (1 - q)d_v^B$. When faced with the decision of switching, the node will always choose the behaviour that results it with the biggest payoff. Let us now assume that $P^B > P^A$. In that case:

$$\begin{aligned}
 (1 - q)d_v^B &> qd_v^A \\
 (1 - q)d_v^B &> q(d_v - d_v^B) \\
 d_v^B - qd_v^B &> qd_v - qd_v^B \\
 d_v^B &> qd_v.
 \end{aligned}$$

Similarly, $P^A > P^B$ would have yielded $d_v^B < qd_v$. It is now clear that v should adopt a certain behaviour if at least a q fraction of its neighbours have adopted it. Therefore, q is the *adoption threshold* for behaviour B . The smaller the value of q , the more likely it is for a node to adopt the new behaviour, and thus, it can spread through the network much easier.

As time advances, the new behaviour spreads through the network in a cascading fashion. That is, when a node switches from behaviour A to B , it consequently

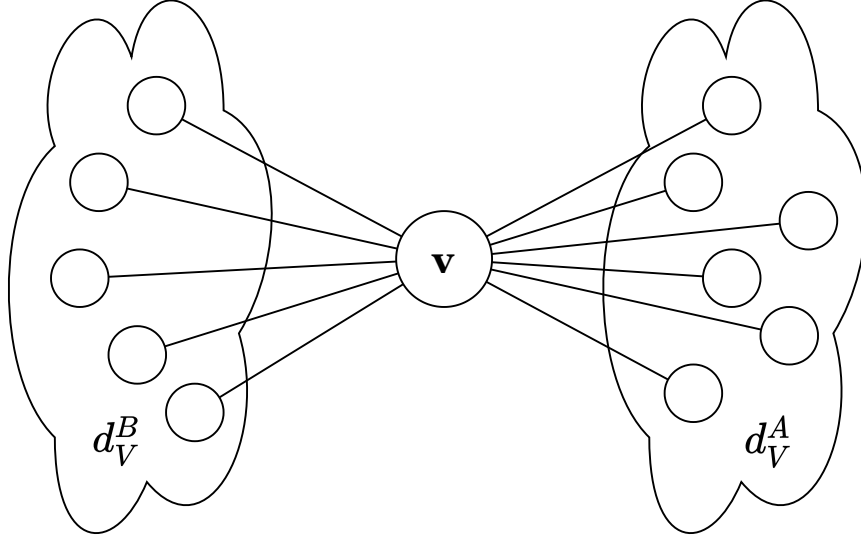


Figure B.1: An illustration of the decision making step for node v , based on the number of neighbours following behaviour A and B .

causes more of its neighbours to also switch, and so forth. Note that up until now, the described model did not consider whether a node could switch back to behaviour A , after it has already switched to behaviour B . To this end, there exist two main categories in the literature [28] that capture this particular aspect: *progressive models* and *non-progressive models*. According to the progressive models, nodes converted by the new behaviour will maintain that behaviour in all future time-steps, while in non-progressive models nodes can, under certain conditions, convert back to the old behaviour. To contextualise this, progressive models can be associated with monopolistic settings of services or technologies diffused over social networks, meanwhile non-progressive models allow for multiple providers competing against each other to tip users to switch back and forth among their services [29]. Studying these two variations of diffusion models, is not only beneficial for capturing particular settings of propagation, but also helps to better understand the process itself, as the contagion thresholds of the two models are highly related [30]. In this thesis, progressive models will be of the main focus.

Let us now assume that the model described earlier is of progressive nature, and nodes cannot switch back to A after adopting B . The cascade which originates as

nodes gradually switch to B has two obvious outcomes. Either all nodes eventually adopt behaviour B , or the spread of B stops, with some parts of the network covered by the new behaviour and others remaining with the old behaviour A . One can easily observe that the progress of the described game depends, apart from the adoption threshold, on the initial parameters of the process, such as the structure of the network itself, and also on the number of the initial adopters. A number of research works [] focus on finding the appropriate initial conditions of spreading behaviours that would be able to cover entire populations, while keeping the resources used for such processes at minimum levels.

Node conversion

In a graph G , with countably infinite set of nodes, and where each node has a finite number of neighbours, a node w is *converted* by a set S of initial adopters, if for some time-step t this node adopts the new behaviour with respect to the threshold q . If every node of the graph is eventually converted by S , then this set is called *contagious* with respect to the threshold q . For small values of q , it is easier for a set S to be contagious. Following the above, one can define the *contagion threshold* of G to be the maximum q , for which there exists a finite contagious set. This definition of the contagion threshold implies its relation to the topology of G ; the bigger the contagion threshold of a network, the easier it is for behaviour to reach the full population. It has in fact been proven [30], that there doesn't exist a graph with a contagion threshold bigger than $1/2$, whether a progressive or a non-progressive process is considered. This essentially means that if the strict majority of a node's neighbours is required to switch, before the node also switches to the new behaviour, it is impossible for this behaviour to spread.

B.1.2 Linear Threshold Model

In a real world scenario, different friends or colleagues may exert greater or lesser influence upon certain individuals, based on their relationship, their social status or even their mutual share of beliefs. Due to this, it would be desirable to capture those relationships when modelling diffusion for certain network applications that exhibit a more complex nature of relationships between the nodes.

The Linear Threshold Model is a more general cascade model, in which relationships and rates of influence between members of a population are captured in the form of weights related to each node's neighbour. In addition, each node has its own threshold for adopting the new behaviour.

More specifically, the weight on each edge (w, v) is denoted as a non-negative weight b_{vw} and indicates the influence of w upon v , with $\sum_{w \in N(v)} b_{vw} \leq 1$, where $N(v)$ is the set of nodes with edges to v . Each node v is assigned with a threshold θ_v uniformly at random from $[0, 1]$, and this associated weight is the neighbour fraction of node v that is required to follow the behaviour in order for v to also do so.

The cascading behaviour process starts, as it did in the previous model, from a set of initial adopters S , and as time progresses in discrete time-steps $t = 1, 2, 3, \dots$, a node adopts the new behaviour B if at any given time-step:

$$\sum_{w \in N^B(v)} b_{vw} \geq \theta_v.$$

The Linear Threshold Model is an approach that allows nodes to weight the influences of their neighbours differently, and thus, formulate more advanced rules for adoption. It is, however, still strict, in the sense that neighbour's influences are additive. There are, in fact, more inclusive approaches, like the General Threshold Model, that take into account relationships between nodes that are neighbouring more than 1-hop away.

B.2 Routing Protocols

Network routing is essentially the process by which the appropriate paths for network packets are selected in a network. This process is based on the routing tables, which comprise records of the routes to various network destinations. In the case of small static networks, routing tables may be manually configured. Bigger and more complex networks however, require automated solutions to computing the routes, as topology can change rapidly, making the computation of such information by hand impossible. A solution to this problem is dynamic routing, where the routing tables are constructed autonomously based on routing protocols.

In general, there is a technical distinction between a routing table and a forwarding table. The routing table contains the complete information about every route in

the network in the form of network addresses and the corresponding next hop's address. They are the result of routing protocols and it is from them that a forwarding table is then constructed. Routing tables are periodically reconstructed based on topology changes within the network. Forwarding tables on the other hand, exist within the router and contain all the information to outgoing destinations of the network and the specifics of how to reach them.

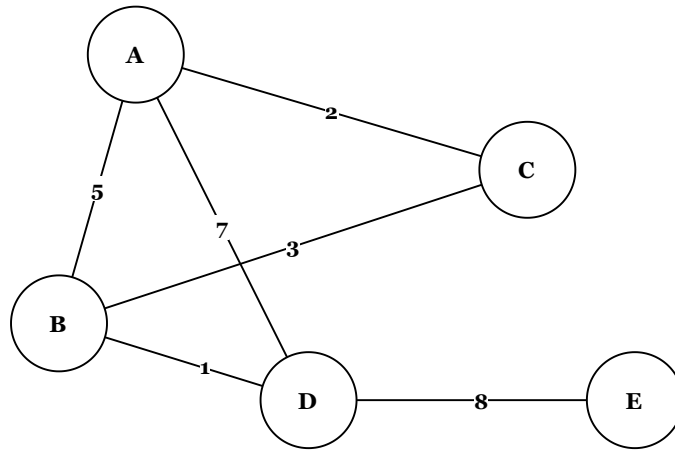


Figure B.2: *A simple graph network.*

Routing is primarily a problem of graph theory. Consider the graph B.2, which is an example based on [31], where each node represents a router in the network and each edge is a network link with an assigned cost. In this example, the edges are undirected and have a single cost, that is, if a link from *A* to *B* exists, then a link from *B* to *A* also exist and has the same cost (this is not always the case when dealing with real networks scenarios). The cost from a node of the network to any other is the sum of the costs of all the links that connects them. A routing algorithm tackles the problem of finding the lowest-cost path between all the nodes of the network in a distributed and dynamic manner. This chapter examines the two main classes of routing protocols: *distance vector* and *link state*.

B.2.1 Distance-Vector Routing

In the distance-vector routing algorithm each router constructs a vector with the costs to all other nodes of the network and shares this information at regular intervals with his 1-hop neighbours. Various widely used routing protocols are built on the

distance-vector algorithm. The most notable example is the Routing Information Protocol (RIP) used in IP networks. Distance-vector requires prior knowledge of the network topology which is usually distributed in a centralised manner at the start of the algorithm. The process is asynchronous, in the sense that the nodes do not operate in synchronised steps with each other. The costs assigned to each neighbour may contain special values depending on the circumstances. For example, when a link to a node in the network doesn't exist, the cost is set to a high enough integer and is interpreted as infinite.

The distance from a node x to a node y is calculated based on the Bellman-Ford equation which is used to determine single source shortest paths in a graph with negative edge weights (but no negative cycles). After a node computes its distance vector, that is, the distance from its neighbours, it transmits it to all of its immediate neighbours in the form of a routing packet. Nodes then, receive these routing packets and save the most recently received distance vector from each of its neighbours. In this manner the system stays updated, as nodes recalculate their distance vectors when they receive different information, or discover that a link is down.

Figure B.3 depicts an example of distance-vector routing. The costs of the links between the nodes are denoted and the least-cost paths can be easily observed. The table of the figure is essentially a distance matrix, and it shows the information initially available to the nodes about the network. Each row of the table is an array that corresponds to one node. The values of the array represent the distance of the shortest-path to each other node in the network. A node's distance to itself is denoted as 0, and distance to nodes that are not immediate neighbours is initially unknown and is denoted as infinite.

Based on the distance matrix which depicts the knowledge that the nodes possess at each time-step, Table B.2 depicts an initial routing table that node A would have stored. Each row contains the destination identifier, the cost needed to reach said destination, and the next hop in order to reach it. Given that at the initial part of the process the nodes have inadequate knowledge to reach destinations beyond their immediate neighbours, nodes F and E in the routing table of A are assigned an unlimited cost; ergo, these destinations are unreachable.

After the construction of the initial routing tables, the algorithm calls for communicating each node's knowledge with the rest of the network. In the first step of

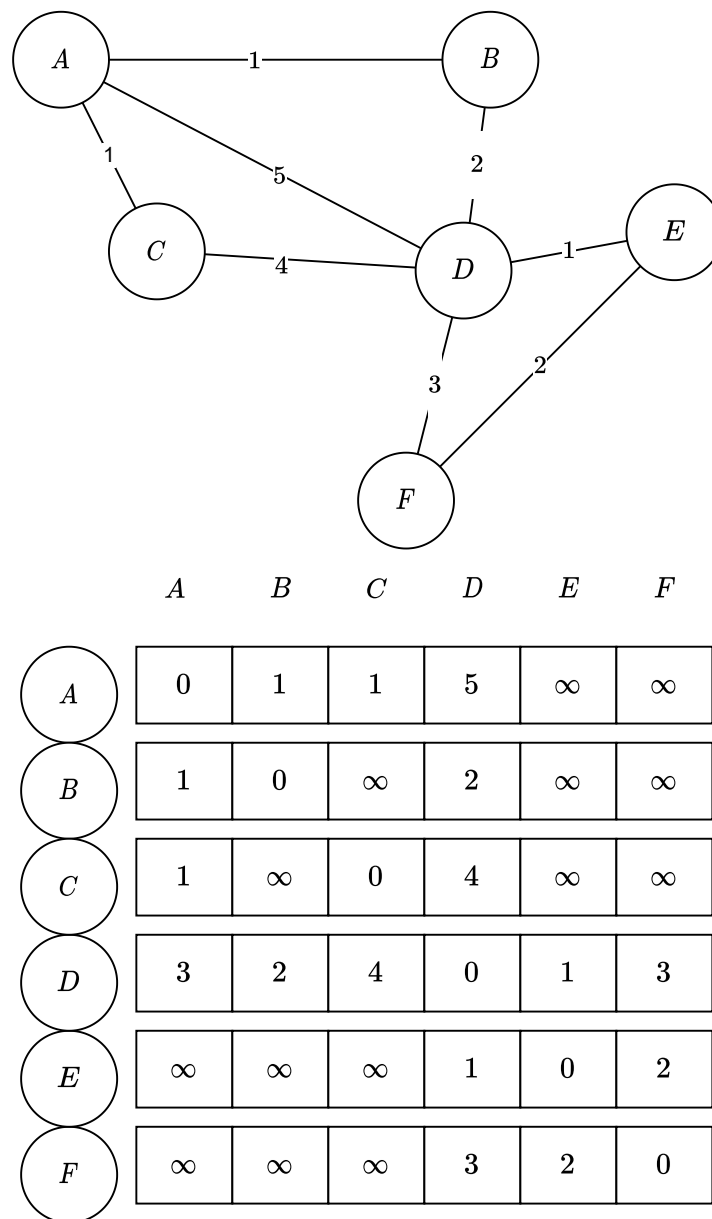


Figure B.3: *Distance-vector routing: A graph example along with a matrix that shows the initial distances available to each node in the graph.*

this process, nodes send a message with the list of their distances to their immediate neighbours. As nodes get new information about the network from their neighbours, they can add up the learned cost of reaching a certain destination with their own, and afterwards decide if the newly obtained route is more cost-efficient than their

Destination	Cost	Next Hop
B	1	B
C	1	C
D	5	D
F	∞	—
E	∞	—

Table B.2: *Initial routing table of node A.*

current one. After a finite number of exchanged messages, each nodes finally ends up with complete, consistent information about the whole network in a distributed manner, as shown in Figure B.4.

	<i>A</i>	<i>B</i>	<i>C</i>	<i>D</i>	<i>E</i>	<i>F</i>
<i>A</i>	0	1	1	3	4	6
<i>B</i>	1	0	2	2	3	5
<i>C</i>	1	2	0	4	5	7
<i>D</i>	3	2	4	0	1	3
<i>E</i>	4	3	5	1	0	2
<i>F</i>	6	5	7	3	2	0

Figure B.4: *Distance-vector routing: Final distances available to each node in the graph.*

Consequently, Table B.3 depicts the final routing table of node A. Certain routes already contained in the initial table are discarded, as more cost-efficient ones are obtained, and destinations that were previously considered unreachable are now formed through their neighbouring nodes.

Destination	Cost	Next Hop
B	1	B
C	1	C
D	3	B
F	6	B
E	4	B

Table B.3: *Final routing table of node A.*

As it was already discussed, the distance-vector algorithm requires an exchange of messages between the nodes, to ensure that information about the whole network is acknowledged by all the nodes. This exchange also happens in regular time intervals, such that even when nothing has changed, other nodes in the network continue to get this node's information in case they need to change their routes, and also ensure the link is still active. There is, additionally, another kind of a message update, the *triggered* update. Triggered updates take place whenever a node discovers that a link is down, or receives a triggered update about a link failure from its neighbours. If this update compels the node to change its own routes, it also notifies its own neighbours about the change, and so on.

B.2.2 Link-State Routing

Link-state routing protocols are of the main focus of this thesis, as the described approach in Chapter C considers a modification of Open Shortest Path First (OSPF), a link-state based routing protocol. They are in many ways similar to distance-vector routing protocols, as each node still learns about the network through its neighbours.

The basic difference lies in the fact that link-state routing relies on disseminating every node's information about the costs of reaching their immediate neighbours to every other node in the network. For this purpose, link state utilises a *flooding* algorithm, where a node send its link-state information to its immediate neighbours, and the neighbours then forward a copy of the information to their immediate neighbours, so that eventually the information propagates through the entire network in this manner (see Figure B.5).

The updates that nodes exchange are called *Link-State Packets* (LSP) and con-

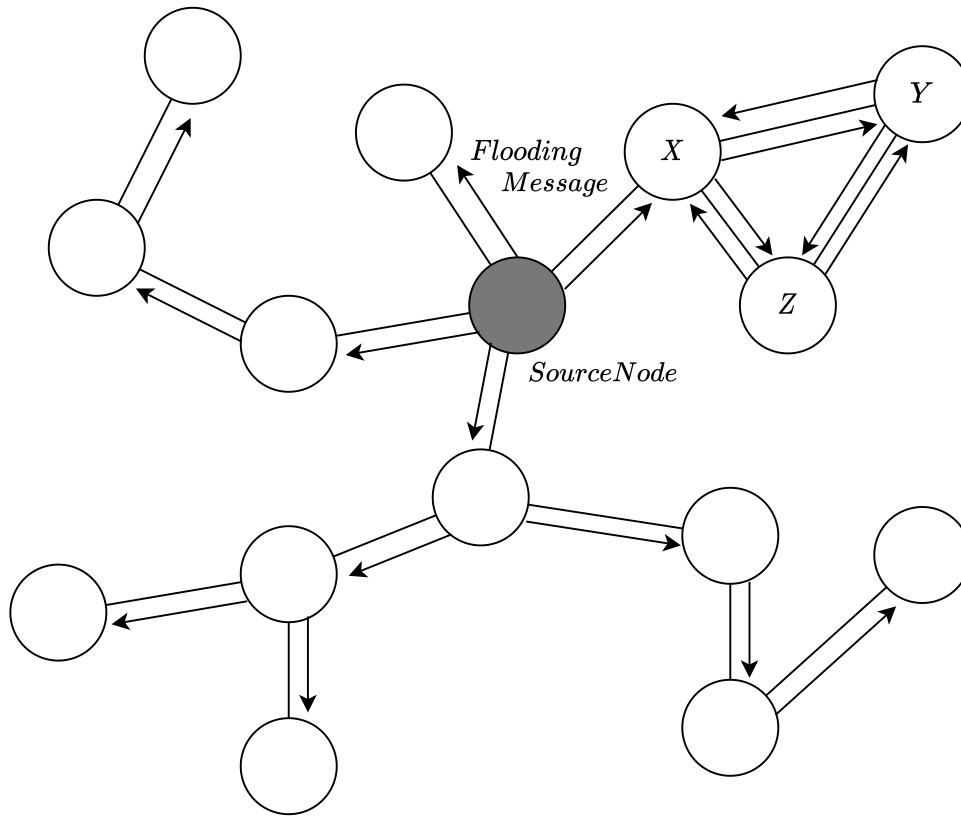


Figure B.5: An example of flooding in a graph: Source node initiates the transmission of flooding messages, and every node that receives it retransmits a copy.

tain the subsequent information: the ID of the node the packet originates from, an array of immediate links to the node and their corresponding costs, a sequence number, and the packet's time-to-live (TTL).

It is crucial the information is reliably transmitted to all nodes. For this purpose the nodes use the packet's sequence number and TTL. This ensures that even when multiple, conflicting LSPs arrive from the same source, the recipient is still able to correctly sort them out. The transmitted packets also make use of acknowledgements and corresponding retransmissions to avoid packet loss.

Once an LSP copy of every node in the network is collected, a given node constructs a graphical representation of the network. This includes the links existing in the graph, and the cost associated with each link. The goal is to derive the shortest-path to each destination. For this purpose, Dijkstra algorithm is used in link-state

routing protocol. Dijkstra's algorithm is an iterative process of finding the shortest path from one specific starting node to all other nodes of a graph.

Flooding in Link-State Routing

Flooding is the process by which the source node, essentially, floods the network with some information, as every node that receives the flood message, retransmits a copy to every other one of its neighbours. Aside from that, to ensure the reliability of this process, nodes wait for respective acknowledgements after sending LSP packets, and if none is received from a neighbour, they proceed to packet retransmission.

The sequence numbers that are present in the LSP packets help with distributing the latest copy of the information. Consider, for example that a node receives an LSP packet from some other node in the network. If no copy of this information is yet stored, the node accepts the copy and stores it in its database. If, however, there is a copy already present from the same source, the node compares between the two sequence numbers, and keeps the newer version. In case the node replaces its copy, the next step is to retransmit the new version to the rest of its neighbours. Other nodes follow the same procedure, and eventually all nodes end up with the same copy of information.

Additionally, the TTLs that LSP packets carry, are used to put an end to the old information that is stored in the network. In every discrete time-step, the TTL value of the copies travelling across nodes, or copies already existing in the database of the nodes is decreased. Once TTL reaches zero, nodes initiate a flooding process to notify the network that the obsolete LSP copy must be removed.

It is important to mention that, even though precautions such as TTL counters and dropping of duplicates are taken, to prevent endless circulation of flooding messages in the network, flooding is still one of the costliest algorithms for information dissemination in terms of bandwidth. The main reason for that is the large number of potentially unnecessary retransmissions caused by the algorithm.

An example of unnecessary retransmissions in flooding can be seen in Figure B.5, due to the loop formed between the nodes X , Y , and Z . Node X sends a flooding message to nodes Y and Z . Both nodes X and Z do not resend back to node X , they do, however, send a flooding message to each other, as they have no knowledge that the same message was already received by both of them. Furthermore, by

the nature of the algorithm, they each would additionally need to retransmit the message back to node X , as a message must be sent to every other node except the source node. To prevent this superfluous circulation in these topologies, nodes are prevented from sending flooding messages to nodes they have already received from.

Messages generated by flooding must be sent to every other node in the network, even when the information has one potential destination. Imagine in a large, dense network, a link between node X and its only neighbour Y is lost, as shown in Figure B.6. It could be that X is an important end node, and shares traffic with a big portion of the network, in which case it would be useful to notify the network of the lost connection. In the case, however, where little to no traffic is shared between the node X and the rest of the network, notifying the network about the lost connection may not be even worth it at all. Due to the density of the network, a large number of flooding messages would be spawned, with little to no use to routing traffic in the network. Later in Chapter C, a more resource-conservative approach to dissemination in link state is discussed, in an attempt to tackle this issue.

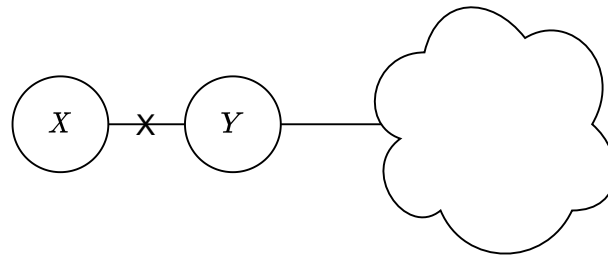


Figure B.6: *An example of a link failure in large-scale network.*

Chapter C

Proposed Approach

THE approach proposed in this thesis incorporates the cascading behaviour in traditional routing protocols, in order to minimise the dissemination of unnecessary information in events of topology changes, which are used to maintain the node's forwarding tables up to date.

C.1 Overview

The cascade model is used in this approach in order to disseminate a link failure event across the network in an efficient manner. The general idea is that the cascading behaviour of dissemination produces less overhead than the traditional flooding, as the originated cascade terminates in certain points of the network, based on certain parameters such as the source of the cascade, which is hopefully sufficient to yield a reliable routing table. This approach, however, results in a compromise between the reliability of routing based on node coverage, and the saving of resources, that in this case is translated in the number of produced information dissemination messages.

The knowledge of the link failure is modelled as a behaviour in a linear threshold cascade model; a node that discovers a link failure is adopting the new behaviour, and other nodes in the network are deciding to whether or not, also adopt this new behaviour, and consequently update their routing tables, based on their adoption thresholds, and the weights of the links they hold to their neighbours.

The decision for adoption is partially based upon statistical information that nodes derive about the network as time progresses. Thus, the initial construction

of the routing tables, still calls for a flooding dissemination. It is only after a finite period of time, that nodes collect sufficient information, in order for the cascading algorithm to take effect.

The intuition behind the weights assignment is to take into account the frequency of communication, or traffic, between the nodes. Therefore, a metric needs to be derived, which would indicate the probability of a node x communicating with a node y . This communication could be either direct; x decides to send a message to y , or vice versa, or indirect, that is x needs to route its message to y , in order to reach a third node z . This metric is particularly useful when examining if the behaviour cascading from a given neighbouring node is to be accepted.

Another important aspect of the model, is that each node maintains its own adoption threshold. This individual adoption threshold is static, and is based solely on the node's *degree centrality*. Therefore, the more popular the node, the more likely the node may need to update its routing information, due to its high connectivity.

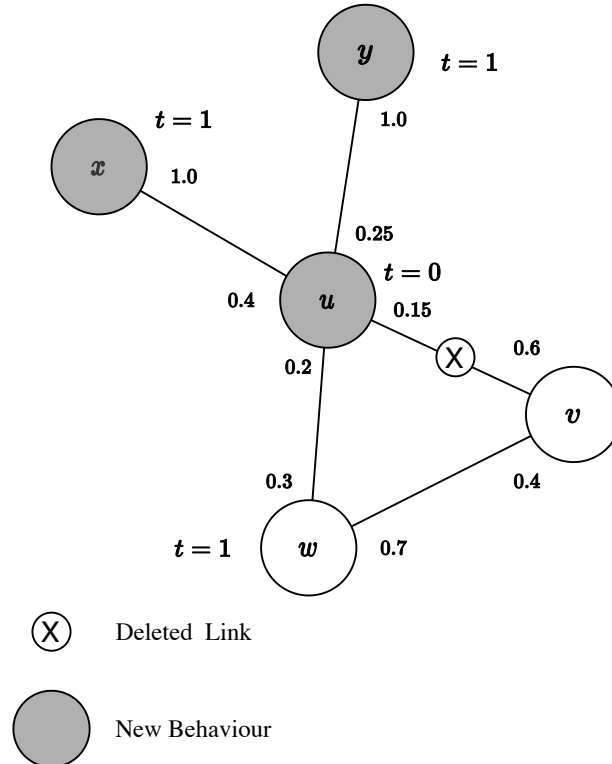


Figure C.1: Example of cascading behaviour in a graph.

Figure C.1 depicts the general idea of the cascading model proposed for the routing information update. At some point in time $t = 0$ node u tries to communicate with v and finds out that the link is lost. Node u then starts cascading this information about the lost link to his neighbours. Nodes update their behaviour if the portion of neighbours with the new behaviour is greater than some threshold. The weight of the nodes are shown in the figure. Note, also, that these weights are directed, and the weight u assigns to the link with v does not necessarily equal the weight v assigns to the same link. Based on the weights depicted in the figure, in $t = 1$, nodes x and y adopt the behaviour and update their routing tables, as u is their only neighbour. In the same time-step node w doesn't adopt the behaviour, thus, causing the cascade to stop. After this partial update of routing tables in the network, nodes x , y , and u are still able to communicate with v , by re-routing their packages through w . Therefore, the accuracy of the routing is not affected, despite w not being informed about the lost link between u and v .

The depicted process is a general example of how the cascades can be used for modelling routing updates. The rest of this Chapter is concerned with introducing the necessary notations and explicitly stating assumptions about the system, before eventually laying out the proposed approach in full detail.

C.2 System Model and Assumptions

The exploration of the model takes place in a network topology, which is modelled by an undirected graph $G(N, L)$, with N and L being the sets of nodes and links (or edges), respectively. Furthermore, A is an $|N| \times |N|$ adjacency matrix of G , with element $A_{u,v}$ being one, if nodes u and v are adjacent, and zero otherwise. The number of nodes adjacent to $u \in N$ is denoted by $\delta(u) = \sum_{v \in N} A_{u,v}$.

Assume all nodes have a forwarding table accurately constructed by some routing algorithm. Through this algorithm the nodes acquire information in order to derive their routing tables. For the purposes of routing, each link has a cost of 1, so that the total cost between two nodes is simply the number of hops separating them. Then $d_{u,v}$ denotes the distance in hops between node u and v , and $p_{u,v}$ the path between u and v , such as $\{u, i, j, \dots, v\}$.

The nodes of the network exchange information, therefore it is useful to quantitatively capture the traffic of the network. The traffic patterns between the network

nodes are assumed to be known and static. As such, a traffic matrix τ is formed, such that the number of packets originating from node u to v is denoted as $\tau_{u,v}$. The load of the link (u, v) is denoted as $l_{u,v}$, which indicates the number of packets traversing this link. Note that, $l_{u,v}$ does not necessarily equal $l_{v,u}$. The load is also partially determined by the routes produced by the routing algorithm, as well as the traffic patterns of the network.

Figure C.2 illustrates the traffic flow in an example graph. The link costs are equal and traffic is unevenly distributed across the nodes. The packets from node u to w are routed through node v , therefore loads $l_{u,v}$ and $l_{v,w}$ increase accordingly. The respective distance, traffic, and load matrices are additionally shown in Tables C.1 and C.2.

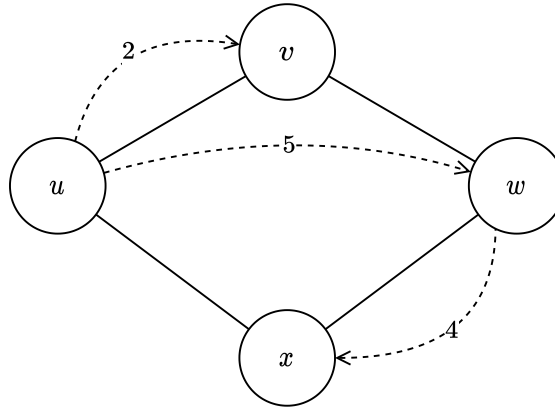


Figure C.2: Example of traffic flow in a graph. The dotted lines represent the number of messages sent from one node to another.

	u	v	w	x
u	0	1	2	1
v	1	0	1	2
w	2	1	0	1
x	1	2	1	0

Table C.1: Distance matrix.

	u	v	w	x
u	0	2	5	0
v	0	0	0	0
w	0	0	0	4
x	0	0	0	0

	u	v	w	x
u	0	7	0	0
v	0	0	5	0
w	0	0	0	4
x	0	0	0	0

Table C.2: From left to right: traffic matrix and load matrix.

C.3 Cascade Model

The cascade process is modelled after the Linear Threshold Model, as presented in Chapter B. The goal is to assign context-specific weights to the links of the network, and to also derive an appropriate adoption threshold for each node, in order to appropriately disseminate topology changes and accommodate routing protocols.

Links are assigned weights based on the traffic they carry. That is, an edge of the graph between the nodes v and w bears a non-negative weight denoted as b_{vw} , with $\sum_{w \in N(v)} b_{vw} = 1$, where $N(v)$ is the set of nodes with edges to v . This weight is directly related to the traffic that flows between the nodes of the link. The general idea is that the greater the traffic, the more inclined a neighbouring node is to the new routing information, as the node may frequently need to utilise the failed link for its traffic. In general, the inability to cover nodes that frequently communicate via the lost link would result in a more dramatic loss of routing accuracy.

Behaviours are used to spread a change in the topology. In this way, multiple behaviours will eventually co-exist in the network. Behaviours are independent of each other, in the sense that a node may adopt multiple behaviours, regardless of previous adoptions, so that the adoption of a behaviour does not necessarily strip them of their already obtained knowledge.

To make this concrete, assume that at a certain point in time, a change in the topology of G emerges, that is a link has gone down between two nodes. Events take place in discrete time-steps, and in $t = 0$, those two nodes find out that the link is down (by exchanging periodic HELLO messages), convert to behaviour B and start spreading this behaviour to their neighbours in the form of a message. This message could be viewed as an LSP packet of a link-state protocol, and contains information about the affected link. Consequently, in time-step $t = 1$, each of their neighbouring nodes (u) receives this message, and updates its behaviour if the following holds

true:

$$\sum_{v \in \delta^B(u)} b_{uv} \geq q_u. \quad (\text{C.1})$$

C.3.1 Progressive

In special cases, such as in a chain topology, non-progressive cascades can stumble upon patterns of endlessly switching behaviours. To make this clearer, consider the graph depicted in Figure C.3. In time step $t = 0$, the initiator node 0 starts a new behaviour B with adoption threshold $q = 1/2$. Then in time step $t = 1$, nodes -1 and 1 will adopt B , whereas node 0, whose neighbours are still following A in time step 0, will switch to A . Then in $t = 2$, nodes 1 and -1 will adopt A , while nodes 0, 2, and -2 will adopt B . In this manner, as the behaviour spreads through both directions of the graph, nodes that are even-numbered and odd-numbered will always follow opposite behaviours and the system will never become stable.

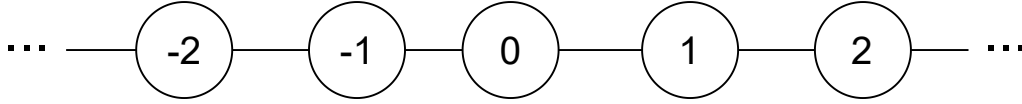


Figure C.3: Chain topology: node 0 initiates the cascading of a new behaviour.

In order to avoid this kind of patterns, and to also take full advantage of the information that is being disseminated, the model considered for this approach is progressive. That means that, no matter the circumstances, once a node adopts the behaviour that is being cascaded, it will not switch back to the old behaviour in some future time step. This property is similar to how link-vector routing manages the LSP packets using sequence numbers in order to avoid loops.

Additionally, as the cascading messages also contain the new routing information, nodes that reject the new behaviour, can still utilise this information. After all, disregarding information that is being directly transmitted to a node, would be wasteful. The challenge is to achieve the termination of the cascade process in parts of network where the routing information is not required.

C.3.2 Weights

The weights assigned to each node's link indicate how important the neighbour at the end of that link is. Importance can be determined in a number of ways and depends greatly on the needs of the application. In the case of routing, nodes that communicate more frequently with each other are much more likely to participate in the routing of each other packages.

One approach to calculating the weight of a link is to take into account the number of incoming packets of that link and dividing it with the total number of packets of all the nodes' links. The weight node u assigns to its link with v is denoted by b_{uv} and is derived by:

$$b_{uv} = \frac{l_{vu}}{\sum_{w \in N(u)} l_{uw}}, \quad (\text{C.2})$$

where l_{vu} the directed load of the link from v to u , which is divided by the sum of the loads of all links that u holds. In this way, it is ensured that

$$\sum_{v \in N(u)} b_{uv} \leq 1,$$

as is required by the Linear Threshold Model.

Another approach to weight calculation would be to take into account the outgoing, as well as the incoming, packets of the link. This would transform C.2 into:

$$b_{vw} = \frac{l_{vw} + l_{wv}}{\sum_{u \in N(v)} (l_{vu} + l_{uv})}, \quad (\text{C.3})$$

where apart from l_{vu} , l_{uv} is also considered.

C.3.3 Threshold

The adoption threshold is a metric that indicates how prone a node is to adopting the new behaviour. Large adoption thresholds require a larger portion of the nodes' neighbours to adopt the behaviour, in order for the node to also do so. In the context of routing, this threshold would need to reflect the node's predisposition to stay up to date with the changes to the network's topology.

According to the model, nodes follow separate adoption thresholds, so instead of arbitrarily assigning adoption thresholds to the nodes, the degree centrality of the nodes is taken into account. The idea is that the more central the node, the more

inclined it is to adopting any new information concerning the network, as it is more likely for it to participate in the routing processes of other nodes. So the adoption threshold needs to be inversely proportional to its degree centrality.

The degree centrality of a node indicates the importance, or the popularity, of the node to the network, based on the number of links it holds, and is

$$D_u = \frac{\delta(u)}{|N|},$$

where $\delta(u)$ the number of links of node u , and $|N|$ the total number of nodes in the network.

Chapter D

Evaluation

THIS Chapter is concerned with experimentally evaluating the proposed approach under diverse conditions and comparing the two different alternatives for link weight assignment. To quantify its effectiveness in informing nodes about topology changes, various metrics are introduced, while the simulation environment used during experimentation is thoroughly described. To better contextualise its performance, two baselines are considered, where on one extreme all nodes are informed about routing updates through flooding, whereas on the other no nodes are informed.

D.1 Setup

The evaluation of the proposed cascade model for the dissemination of routing updates, takes place over Random Geographic Graphs (RGGs). The algorithms capturing the model and its assumptions, as they were previously discussed in Chapter C, are developed and evaluated in the OMNeT++ simulation environment. Various RGGs with different number of nodes and connectivity radii (r_c) are considered, in order to take into account topological effects of the described model. The experiments are conducted for graphs of 10, 20, and 30 nodes. The different r_c values are derived based on the number of nodes in the network, and are $r_c = c_r * \hat{r}_c$, calculated for each $c_r \in \{1.2, 1.3, 1.4\}$, where $\hat{r}_c$ corresponds to connectivity threshold [32]:

$$\hat{r}_c = \sqrt{\frac{\ln(|N|)}{\pi|N|}},$$

in order to ensure the connectivity of the graphs with high probability.

Initially, the routes of the network are computed with the help of the Dijkstra algorithm. The traffic matrix τ is constructed by each node generating a random number of messages to all nodes (including itself). In one scenario, the number of messages sent by node i to node j is generated according to a uniform distribution in the range $[1, \frac{120^2}{|N|^2}]$, such that the total volume of traffic generated remains mostly the same, regardless of $|N|$. Another scenario is considered, where the number of messages sent by node i to node j is generated according to a Poisson distribution with mean $\frac{120^2}{2|N|^2}$, accumulating to a total similar to the previous scenario. These packets are routed from their sources to their destinations, and the link weights of the model are assigned according to the generated loads of the links. Two approaches for link weight assignments are compared: (i) the link's incoming packets are taken into account, as shown in Equation (C.2), (ii) both incoming and outgoing packets are taken into account, as shown in Equation(C.3). The adoption thresholds for each node is $q = 1 - (1/2 * D_u)$.

At a certain point in time, a connection gets lost in the network, and one of the two nodes holding that link converts to behaviour B , sending a message to the rest of its neighbours. In this manner, the cascading process starts. The message contains the ID of the nodes associated with the link. Nodes receiving this message, immediately update their routing tables, and then adopt the new behaviour if Equation (C.1) holds true. At some future point in time, once the cascade has ended, nodes proceed to send the traffic one more time, according to τ , in order to evaluate the cascading model dissemination and its effect on the routing paths.

For the sake of comparison, the maximum number of packets that can be delivered in the new topology, denoted by M , is calculated. Similarly, the maximum number of packets that can be delivered if no nodes update their routing tables, denoted by m , is calculated. Additionally, two packet *delivery ratios* are derived: The *actual* packet delivery ratio, calculated as the actual number of packets a delivered with routing updates propagation under cascade over the maximum that can be delivered, assuming all nodes have been updated, derived as

$$\frac{a}{M}.$$

The *worst-case* delivery ratio between the maximum number of packets that can be sent, assuming no nodes have been updated, over the maximum that can be

delivered, calculated as:

$$\frac{m}{M}.$$

Moreover, the *fraction of covered nodes* is used as a metric to measure how effective the cascade is at covering the network, derived as $\frac{N^B}{|N|}$, where N^B is the number of nodes that adopt behaviour B .

Last but not least, the *expected ratio* is considered, which measures the total number of packets that the network's nodes expected E to successfully deliver compared to the actual number of packets that could be delivered, assuming all nodes were updated, which is derived as

$$\frac{E}{M}.$$

This metric is clearly inversely correlated with the coverage ratio, as the more informed the nodes are, the more likely they are to not overestimate the number of packets that can be delivered. However, it also indicates how effective the cascade is at covering more important nodes, which is correlated with the actual delivery ratio. Thus, the expected ratio is an interesting metric that is hard to intuit.

D.2 Results

In this Section, the acquired results are presented across the different values of $|N|$ and c_r , showcasing the effectiveness of the proposed approach according to the described metrics and comparing the differences between the two different link weight assignment methods. It should be noted that all depicted values correspond to the mean of the values acquired from 10 repetitions for each set of parameter values.

D.2.1 Uniform Traffic

In Fig. D.1, the delivery ratios acquired from the experiments are depicted, across different values of c_r and for different network sizes, comparing the different link weight assignment methods considered. The dashed lines indicate the worst-case scenario performance, achieved by updating none of the network's nodes. It is observed that the proposed approach, based on cascades, is capable of achieving close to 100% actual delivery ratio, significantly outperforming the worst-case scenario.

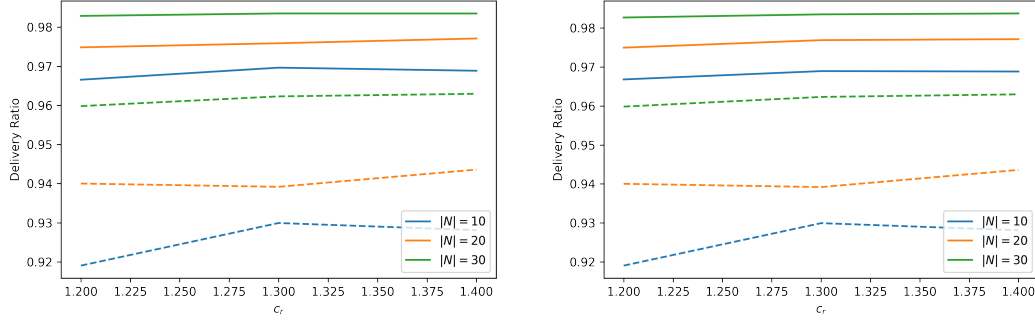


Figure D.1: The actual delivery ratio of traffic generated according to the uniform distribution for different network sizes and densities when using incoming link load for weight computation (left), and both incoming and outgoing load (right). The dashed line indicates the worst-case performance achieved by not updating any nodes.

This seems to be the case, regardless of the network's size or density. Furthermore, the different link assignment methods do not seem to affect the cascade's performance much, with respect to the actual delivery ratio.

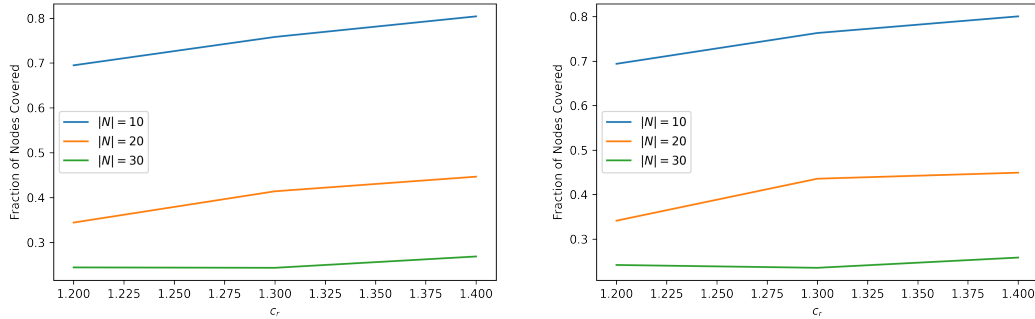


Figure D.2: Fraction of nodes covered by the model for traffic generated according to the uniform distribution using incoming load for weight computation (left), and both incoming and outgoing load (right).

On the other hand, in Fig. D.2, the fraction of the covered nodes is displayed across different values of $|N|$ and c_r . Evidently, there is a small increase in the coverage ratio as c_r increases, which indicates that as the network gets more dense, the more nodes adopt the cascading behaviour. However, it is noteworthy that as the network gets larger, the fraction of nodes adopting the behaviour is significantly

reduced. At the same time, though, for larger networks, the actual delivery ratio is the highest, indicating that the approach is effective at spreading updates to few but relevant nodes, which was the intended goal. Once again, the different link weight assignments do not differ much, perhaps due to the uniform traffic patterns considered.

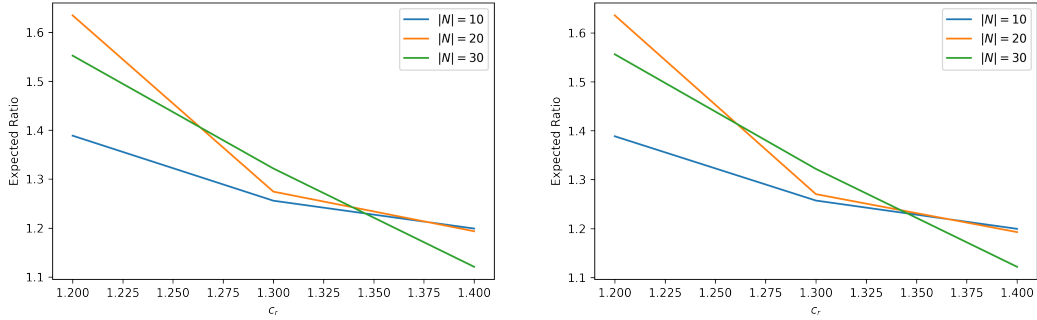


Figure D.3: *Expected ratio achieved by the model for traffic generated according to the uniform distribution using incoming load for weight computation (left), and both incoming and outgoing load (right).*

Finally, the expected ratio is shown in Fig. D.3. This metric is once again largely similar for both link weight assignment methods. Additionally, it seems to be mostly affected by the network's density, rather than its size, with higher densities leading to a lower expected ratio. It is also worth noting that the expected ratio is not purely correlated with coverage ratio, as it can be seen that despite coverage ratios being mostly dependent on $|N|$, this is not the case here. Nonetheless, the expected ratio almost reaches 1.0 for higher densities, which is the optimal value achievable by flooding.

D.2.2 Poisson Traffic

In the sequel, the traffic is generated according to the Poisson distribution, which actually results in less variance in the number of messages exchanged between nodes, due to the high mean of the Poisson Distribution, compared to the high variance in the uniform distribution brought about by the large range of possible values.

In Fig. D.4, the actual delivery ratio is depicted again. Interestingly enough, no significant differences are observed across the two link weight assignment methods,

with the proposed approach achieving close to optimal routing accuracy. In fact, the fraction of covered nodes, depicted in Fig. D.5, is largely similar to the one observed in the uniform traffic scenario, being inversely proportional to the network's size, with a slight increase observed as the network gets more dense. As such, the highest delivery ratio is achieved for $|N| = 30$, where the coverage fraction is also the lowest, indicating the approach's effectiveness.

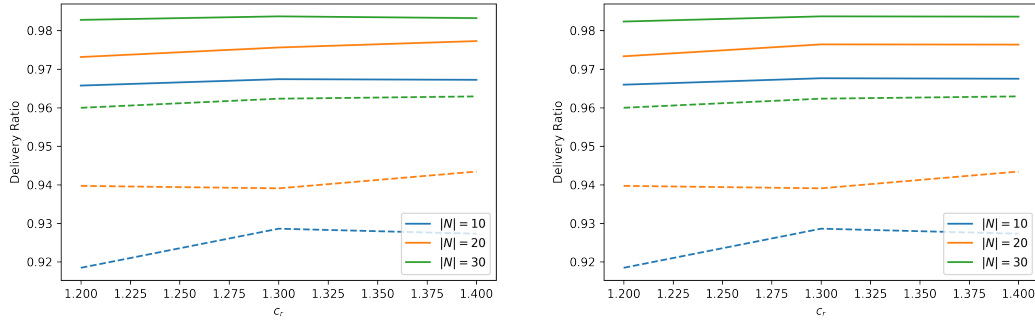


Figure D.4: The actual delivery ratio of traffic generated according to the Poisson distribution for different network sizes and densities when using incoming link load for weight computation (left), and both incoming and outgoing load (right). The dashed line indicates the worst-case performance achieved by not updating any nodes.

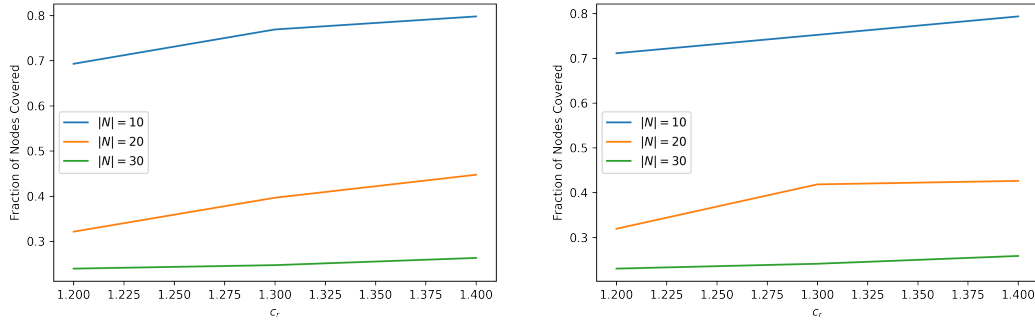


Figure D.5: Fraction of nodes covered by the model for traffic generated according to Poisson distribution using incoming load for weight computation (left), and both incoming and outgoing load (right).

The expected ratio acquired for the scenario with Poisson-generated traffic is shown in Fig. D.6. The two link weight assignment methods seem to perform the

same for the most part, as was also the case in the uniform traffic scenario. In particular, it is observed that the nodes' expectations become more realistic as the network gets more dense, although it is not affected much by the network's size. Thus, it seems that the traffic's variance may not significantly impact the proposed approach's effectiveness, unless more extreme traffic distributions are taken into account.

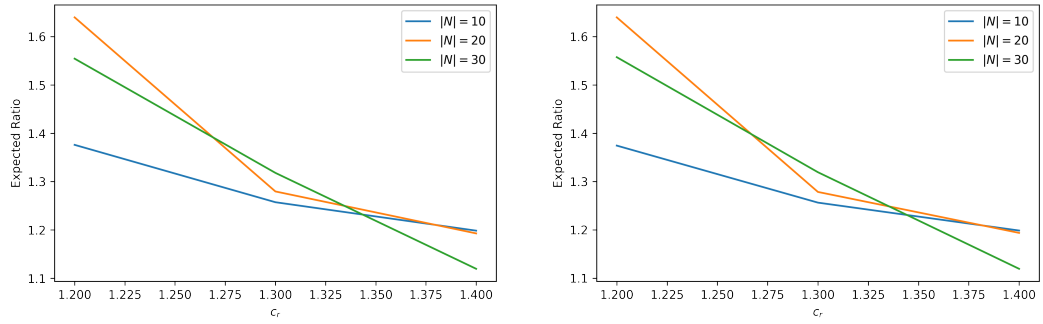


Figure D.6: *Expected ratio for traffic generated according to Poisson distribution achieved by the model using incoming load for weight computation (left), and both incoming and outgoing load (right).*

Chapter E

Conclusions

THIS Thesis proposed an approach for disseminating updates that occur to the network topology, which may be utilised by routing protocols to provide consistent routing tables for the network's nodes. This approach is motivated by interactions observed in social contexts, such as social networks, viewed under a game-theoretic lens. In particular, the approach extends the Linear Threshold Model, which capture the phenomenon of cascading behaviours, caused in part by peer pressure and herd behaviour, by assigning weights to nodes, based on their susceptibility, and to the connections established between them, according to the influence one node exerts to another.

Primarily, the Linear Threshold Model was extended by devising appropriate mechanisms to assign weights to nodes and links, taking into account the intricacies of routing. More specifically, weights were assigned to nodes according to their expected degree of participation in the routing process. To capture this, degree centrality was adopted, assigning lower values to nodes that are more interconnected. On the other hand, links were assigned weights in proportion to their utilisation, with two alternatives being considered, differing in whether only incoming, or both incoming and outgoing traffic is incorporated.

This cascade-based approach to disseminating messages containing information about topology changes for routing was experimentally evaluated, so as to assess its ability of minimising overhead, while maintaining routing tables up to date. It turned out that under both uniform and Poisson traffic patterns, the proposed approach was capable of almost perfectly routing packets, despite updating only a

small fraction of the network's nodes. Thus, the approach was able to target few but relevant nodes, promising to be vastly more scalable than the traditional approach of flooding the entire network with updates.

Fundamentally, the proposed approach forms the basis for a state synchronisation methodology, which is more generally applicable than routing, as the need for providing a consistent view among distributed systems arises in various networking (among other) domains, including Software-Defined Networking, cloud computing, and so on. Given the promising results acquired from the conducted experimentation, it may be worth further looking into and investigating under different contexts.

However, before such effort is undertaken, there is still much work to be done in the presented setting. To start with, different topologies may provide distinct challenges for the presented model to generalise, without introducing more sophisticated methods for assigning the required weights or requiring further parameters. For instance, a parameter could be used to alter the value of the adoption threshold depending on the topology, much like the value of $1/2$, which turned out to be effective in practice for the considered RGGs. To this end, perhaps different centrality metrics could be utilised for the adoption threshold, such as betweenness centrality, which captures a node's relevance in routing more directly than degree centrality, at the expense of being more costly to compute in a distributed setting. Last but not least, more diverse traffic patterns could motivate the investigation of different link weight assignments.

Bibliography

- [1] J. Kleinberg, “Cascading behavior in networks: Algorithmic and economic issues,” *Algorithmic game theory*, vol. 24, pp. 613–632, 2007.
- [2] T. C. Schelling, *Micromotives and macrobehavior*. WW Norton & Company, 2006.
- [3] M. Granovetter, “Threshold models of collective behavior,” *American journal of sociology*, vol. 83, no. 6, pp. 1420–1443, 1978.
- [4] I. Shah, S. Jan, I. Khan, and S. Qamar, “An overview of game theory and its applications in communication networks,” *International Journal of Multidisciplinary Sciences and Engineering*, vol. 3, no. 4, pp. 5–11, 2012.
- [5] A. V. Banerjee, “A simple model of herd behavior,” *The quarterly journal of economics*, vol. 107, no. 3, pp. 797–817, 1992.
- [6] I. Welch, “Sequential sales, learning, and cascades,” *The Journal of finance*, vol. 47, no. 2, pp. 695–732, 1992.
- [7] S. Bikhchandani, D. Hirshleifer, and I. Welch, “A theory of fads, fashion, custom, and cultural change as informational cascades,” *Journal of political Economy*, vol. 100, no. 5, pp. 992–1026, 1992.
- [8] A. Blum, E. Even-Dar, and K. Ligett, “Routing without regret: On convergence to nash equilibria of regret-minimizing algorithms in routing games,” in *Proceedings of the twenty-fifth annual ACM symposium on Principles of distributed computing*, pp. 45–52, 2006.
- [9] D. Fotakis, S. Kontogiannis, E. Koutsoupias, M. Mavronicolas, and P. Spirakis, “The structure and complexity of nash equilibria for a selfish routing game,” in

- International Colloquium on Automata, Languages, and Programming*, pp. 123–134, Springer, 2002.
- [10] A. A. Economides and J. A. Silvester, “Routing games,” tech. rep., USC Technical Report CENG 89, 1989.
- [11] R. J. La and V. Anantharam, “Optimal routing control: Repeated game approach,” *IEEE transactions on automatic control*, vol. 47, no. 3, pp. 437–450, 2002.
- [12] T. Roughgarden, “How unfair is optimal routing?,” in *Symposium on Discrete Algorithms: Proceedings of the thirteenth annual ACM-SIAM symposium on Discrete algorithms*, vol. 6, pp. 203–204, 2002.
- [13] A. Orda, R. Rom, and N. Shimkin, “Competitive routing in multiuser communication networks,” *IEEE/ACM Transactions on networking*, vol. 1, no. 5, pp. 510–521, 1993.
- [14] R. Banner and A. Orda, “Bottleneck routing games in communication networks,” *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 6, pp. 1173–1179, 2007.
- [15] Y.-C. Tseng, S.-Y. Ni, Y.-S. Chen, and J.-P. Sheu, “The broadcast storm problem in a mobile ad hoc network,” *Wireless networks*, vol. 8, no. 2, pp. 153–167, 2002.
- [16] Y.-C. Tseng, S.-Y. Ni, and E.-Y. Shih, “Adaptive approaches to relieving broadcast storms in a wireless multihop mobile ad hoc network,” *IEEE transactions on computers*, vol. 52, no. 5, pp. 545–557, 2003.
- [17] X. M. Zhang, E. B. Wang, J. J. Xia, and D. K. Sung, “A neighbor coverage-based probabilistic rebroadcast for reducing routing overhead in mobile ad hoc networks,” *IEEE transactions on mobile computing*, vol. 12, no. 3, pp. 424–433, 2012.
- [18] N. Wisitpongphan, O. K. Tonguz, J. S. Parikh, P. Mudalige, F. Bai, and V. Sadekar, “Broadcast storm mitigation techniques in vehicular ad hoc networks,” *IEEE Wireless Communications*, vol. 14, no. 6, pp. 84–94, 2007.

- [19] A. Qayyum, L. Viennot, and A. Laouiti, "Multipoint relaying for flooding broadcast messages in mobile wireless networks," in *Proceedings of the 35th annual Hawaii international conference on system sciences*, pp. 3866–3875, IEEE, 2002.
- [20] H. Liu, B. Zhang, H. T. Mouftah, X. Shen, and J. Ma, "Opportunistic routing for wireless ad hoc and sensor networks: Present and future directions," *IEEE Communications Magazine*, vol. 47, no. 12, pp. 103–109, 2009.
- [21] G. G. Chen, J. W. Branch, and B. K. Szymanski, "Self-selective routing for wireless ad hoc networks," in *WiMob'2005), IEEE International Conference on Wireless And Mobile Computing, Networking And Communications, 2005.*, vol. 3, pp. 57–64, IEEE, 2005.
- [22] M. Meisel, V. Pappas, and L. Zhang, "Listen first, broadcast later: Topology-agnostic forwarding under high dynamics," in *Annual conference of international technology alliance in network and information science*, p. 8, 2010.
- [23] P. Domingos, "Mining social networks for viral marketing," *IEEE Intelligent Systems*, vol. 20, no. 1, pp. 80–82, 2005.
- [24] J. Leskovec, M. McGlohon, C. Faloutsos, N. Glance, and M. Hurst, "Cascading behavior in large blog graphs: Patterns and a model," *Society of Applied and Industrial Mathematics: Data Mining*, pp. 551–556, 2007.
- [25] P. S. Dodds and D. J. Watts, "Universal behavior in a generalized model of contagion," *Physical review letters*, vol. 92, no. 21, p. 218701, 2004.
- [26] Z. Han, D. Niyato, W. Saad, T. Başar, and A. Hjørungnes, *Game theory in wireless and communication networks: theory, models, and applications*. Cambridge university press, 2012.
- [27] M. O. Jackson and Y. Zenou, "Games on networks," in *Handbook of game theory with economic applications*, vol. 4, pp. 95–163, Elsevier, 2015.
- [28] D. Kempe, J. Kleinberg, and É. Tardos, "Maximizing the spread of influence through a social network," in *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 137–146, 2003.

- [29] M. Fazli, M. Ghodsi, J. Habibi, P. Jalaly, V. Mirrokni, and S. Sadeghian, “On non-progressive spread of influence through social networks,” *Theoretical Computer Science*, vol. 550, pp. 36–50, 2014.
- [30] S. Morris, “Contagion,” *The Review of Economic Studies*, vol. 67, no. 1, pp. 57–78, 2000.
- [31] L. L. Peterson and B. S. Davie, *Computer networks: a systems approach*. Elsevier, 2007.
- [32] J. Diaz, D. Mitsche, and X. Perez, “Dynamic random geometric graphs,” *arXiv preprint cs/0702074*, 2007.

Appendices

Appendix I

Glossary

Distributed system	A system with multiple components located on different machines that communicate and coordinate actions in order to appear as a single coherent system to the end-user.
Information diffusion	The process by which information is spread from one node to another through interactions.
Information dissemination	The process by which information is broadcasted to the nodes without necessarily awaiting direct feedback.
Game theory	The study of interactive decision-making, where the outcome for each participant or “player” depends on the actions of all.
Routing update packet	A packet sent to neighbour routers that provides updates concerning all nodes connected to that particular network. Routing update packets are supported by routing protocols like RIP and OSPF.
Network cascade	A model through which individuals are connected in networks to each other then they can influence each other’s behaviour and decisions.
Adoption threshold	The portion of adjacent nodes that is required to adopt a certain behaviour, for given node to also adopt it.

Link weight	The weight a given node assigns to its link with another node represents the importance of this link to the node. This importance can be determined by different metrics, depending on the needs of the model.
Converted node	A node that has accepted the new behaviour diffused by a network cascade model.
Routing protocols	Routing protocols are mechanisms by which routing information is exchanged between routers so that routing decisions can be made.
Distance vector	Routing algorithm that relies on transmitting the routing nodes' entire routing table' to directly connected neighbors
Link state	Routing algorithm that relies on transmitting the routing nodes' information about directly connected links to all the routers in the network.

Appendix II

Notations

G	Graph representing the network
V	Vertices, or nodes of the graph
E	Edges, or links of the graph
S	Contagious set in a cascade model
q	Adoption threshold for a cascade model behaviour
δ_u	Number of nodes adjacent to node $u \in N$
b_{uv}	Weight of the link between nodes u and v
d_{uv}	Distance between nodes u and v
p_{uv}	Routing path between nodes u and v
τ	Traffic matrix
l_{uv}	Load of the link between nodes u and v
D_u	Degree centrality of node u

Appendix III

Abbreviations

MANET	Mobile Ad Hoc Network
RIP	Routing Information Protocol
IP	Internet Protocol
OSPF	Open Shortest Path First
TTL	Time to Live
LSP	Link-State Packet
RGG	Random Geometric Network

Index

Adoption threshold, 11, 29
Distance-vector routing, 15
Game theory, 2, 9
Information cascade, 3

Information diffusion, 1
Link weight, 14, 29
Link-state routing, 19
Routing, 14