Models and Algorithms for Information Diffusion

by

Jayesh Choudhari

Submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of

Ph.D. in Computer Science and Engineering

at the

INDIAN INSTITUTE OF TECHNOLOGY GANDHINAGAR

February 2020

Author.....
Department of Computer Science and Engineering
January 18, 2021

Certified by.....
Prof. Anirban Dasgupta
N. Rama Rao Chair Professor
Thesis Supervisor

To Prof. Anirban Dasgupta.
For his advice, his patience, and his faith.

To my family.
For their astounding love, patience, and support.

To my friends.
For motivating and supporting me throughout.

Models and Algorithms for Information Diffusion

by

Jayesh Choudhari

Submitted to the Department of Computer Science and Engineering
on August 04, 2020, in partial fulfillment of the
requirements for the degree of
Ph.D. in Computer Science and Engineering

Abstract

Social networks, whether real ones or online services, act as important platforms for the propagation of ideas, thoughts, innovation, rumors, infection, etc. The propagation of information from one user to others results in the formation of information cascades. These cascades can be thought of as conversations between the users that can be characterized by the topics being discussed, the time at which the conversation is going on, as well as the complex interactions (who responds to whom) between the users involved in the conversation. In this thesis, we first propose a series of generative models for this complex process of diffusion of information. The first model, Hidden Markov Hawkes Process (HMHP), incorporates the interaction between the users (influence of one user on another), users' topical preferences and, temporal patterns between posting and response times, as is the case with number of other models in existing literature; the novelty of HMHP is in incorporating topical interactions. We also propose a scalable algorithm based on Gibbs sampling that jointly infers the strength of interactions between the users, the path of diffusion of information, the topics of the posts, as well as the topical interactions. Experiments on synthetic data show that HMHP can recover user-user influence, parent identification, and topic identification more accurately as compared to the state-of-the-art baselines. More interestingly, HMHP finds insightful interactions between topics in real tweets which no existing model is able to do.

However, in HMHP, as well as in all the previous models in the literature, the time of the event is dependent only on the users (and their interaction), which is not the case in the real world. We propose the Dual Network Hawkes Process (DNHP) in which the event rates depend on the topic (topical-interaction) in addition to the user (user-interaction). In DNHP, in parallel to the user network, we model a network over the topics as well, so that the events propagate over a two dimensional space indexed by users and topics. This sets up an interaction between the two spaces, where the estimates of user- user weights depend on topic-topic weights, and vice versa. This resultant joint estimation of weights allows evidence to flow across weight variables. As compared to the HMHP, DNHP is not benefited from collapsed Gibbs sampling for the inference task. However, experiments on synthetic data show that DNHP outperforms state of the art models in terms of parent identification, and user-user weight estimation.

In addition to the proposed generative models, we also look at the information diffusion problem from an algorithmic perspective, specifically, in the context of parameterized algorithms. Note that it is not always the case that we want the information to propagate. For e.g., consider a piece of information which is a rumor or fake news, or from the view of the epidemiology, consider an infection or a disease. In these cases, the goal would be to prevent or at least curb the spread of the contagion. Thus, in contrast to the previously mentioned works, which are centered towards understanding the propagation of information and the factors responsible for the same, the goal here is to propose algorithms for preventing the spread of information. In this work, we focus on the firefighting problem which formalizes the question of designing inoculation strategies (termed as firefighters) against a contagion (termed as fire) that is spreading in the network. There could be different objectives in the firefighter problem, but we concentrate on the objective of saving a specific set of nodes (termed as a critical set) and prove some results for the same. First, we show that saving a critical set on trees is fixed-parameter tractable (FPT) when parameterized by the number of firefighters and propose an algorithm with time complexity $O^*(4^k)$. We also show that the problem is FPT on general graphs when parameterized by the number of firefighters. Additionally, we claim that there is no polynomial sized kernel on trees for saving a critical set on trees. Finally, we consider the spreading model of the firefighting problem, where the firefighters spread as well, similar to the fire. For this model, we show that the problem of saving a critical set parameterized by the number of firefighters is W[2]-hard (i.e. as hard as k-DOMINATING SET), which contrasts our FPT result for the previous non-spreading model.

Thesis Supervisor: *Prof. Anirban Dasgupta*
Title: *N. Rama Rao Chair Professor*

Acknowledgments

I have been privileged to work under the guidance of Prof. Anirban Dasgupta and Dr. Indrajit Bhattacharya who was my co-supervisor (unofficially). In addition to being the best researchers I have ever known, Prof. Anirban and Dr. Indrajit are the two kindest, patient, and the most supportive persons that I have had the pleasure of knowing and working with. Prof. Anirban has been a great teacher with an incredibly intuitive understanding of the theory of Computer Science and an impressive understanding of putting the theory into practice. I would like to thank him for his encouragements and his insights, for sharing with me the way of thinking over problems, his knowledge of a wide range of research, for his ability to form elegant questions and the way to approach towards the answer so that the answer emerges with equal elegance, and for his humility and patience, which enabled me to grow as a researcher. Dr. Indrajit has been a great co-supervisor and a mentor. He has an excellent way of looking at the problems from the lens of the knowledge that he has and connects them with ease. His explanation of the problem and his approach towards the solution is just superb. I want to thank him for all his time, for his ability to look at life's problems from the computational perspective, for his way of explaining things and approaching the problem. I would like to thank both Prof. Anirban and Dr. Indrajit for their passion for perfection that is demonstrated in all aspects of their research, and for their unwavering support and faith in me that pushes me to strive to be a better researcher.

I am grateful to Prof. Srikanta Bedathur, who was my host during my internship at IBM Research Labs, New Delhi, India and has been one of the collaborators for the modeling problems that are discussed in this thesis. During my time at IBM, he helped me develop an understanding of the cascade and virality problems that formed the base for the models that we developed for information diffusion presented in this thesis. I would like to thank him for his time and support during and after the internship and also for sharing his excellent knowledge about the tools and techniques related to handling the text-based data that was required in my thesis.

I am also grateful to Prof. Neeldhara Misra and Prof. Chetan Pahlajani for serving on my thesis committee. Their regular feedback has been instrumental in improving the quality of this thesis. Prof. Neeldhara has also been a collaborator on one of the important projects discussed in this thesis. I would like to thank her for helping me in developing a better understanding of the area of parameterized complexity and for motivating me to look at the machine learning related problems from the perspective of parameterized complexity. I also had an opportunity to work with Prof. Chetan on a separate problem related to modeling gene networks. This has helped me in developing a better understanding of the modeling problems in general.

This thesis work has been partially supported by the TCS Research Fellowship. Additionally, I was also provided travel grants by TCS, Microsoft, and ACM-India throughout the tenure of the thesis. I would like to extend my gratitude to all these organizations for their generous support.

A huge thanks to Prof. Manoj Gupta with whom I collaborated on core algorithmic problems. I would like to thank him for helping me in developing a better understanding of this area.

I would also like to thank Prof. Dinesh Garg, and Prof. Mayank Singh, for believing in me and having me as a TA for their respective courses. It was great to learn from them about formalizing and organizing the course-related work.

I would also like to thank all the anonymous reviewers of my publications for their valuable feedback.

A huge thanks to all my friends from IIT Kanpur for all their support and encouragement. Especially, I would like to thank Tejas Gandhi, Shahbaz, Hrishit, Nitesh, Himanshu, Vineet, Akshay, Vinay, Pandit, Lalit, Arjun, Atanu, Chiro, Ashu, Akhil, Ankit for motivating me to pursue a Ph.D. in Computer Science.

I would also like to thank Saransh, Gaurav, and GPS for believing in me and being an awesome family during my stay in Hyderabad. A great thanks to all my teammates Pavan, Arvind, Narayana, and Uday for helping me have a great learning experience at Deloitte and motivating me for doing better in life.

I am grateful to all my friends – Subhadeep, Rahul, Nitin, Vibhuti, Alok, Prathamesh, Sidhesh, Aparna, Sonali for their irrevocable support and help since the time I know them, and also for believing in me and motivating me to move forward relentlessly towards my goals.

A great thanks to all my friends – Ravi, Rajan, Samraj, Hasmukh, Haswani, Jenifer, Bhakti, Amit, Sunder, Ranjith, for their support and help whenever there was a need. And also for having faith in me and keeping me motivated throughout.

I would like to thank the institute, IIT Gandhinagar, and each and everyone at IIT Gandhinagar, who made each day better here for more than five years. Sudhakar was the first person I got in touch with and was also my room-mate for the initial days at IIT Gandhinagar. I am thankful to him all for his support, help, and all the discussions that we had since our first day at IIT Gandhinagar. I am thankful to all the members of the CS group – Supratim, Rachit, Ananya, Sudhakar, Mastan, Shivdutt, Chanda, Glint, all the new members for their help and support whenever needed. I am also thankful to the members of the Physics department – Yusuf, Soumen, Amit, Chakresh for their help and support throughout my Ph.D. at IIT Gandhinagar. I am also thankful to all the Masters and Bachelor students with whom I collaborated or was a TA for. It was great to learn from all of them. I am thankful to Ishita Doshi and Krishan Chugh, the Master's students with whom I collaborated closely. It was a great experience working with them and also to learn from them.

I am grateful to Supratim and Rachit, my labmates who work with Prof. Anirban, for collaborating with me and sharing their knowledge and experience, and for indefinite support and help throughout my Ph.D. years at IIT Gandhinagar.

I am grateful to Ajinkya, Chakresh, Dharmit, Prajakta, Rachit, Sardar (Bassu), for their indefinite and irrevocable support, help, and discussions throughout my stay at

IIT Gandhinagar. Special thanks to all the members of the “Bakait” group for adding fun to my life and making my Ph.D. years memorable at IIT Gandhinagar and for making me believe that “Yes, I can”.

Lastly, I am ever indebted to my family – my mother, my brother, my sister-in-law Priyanka, my wife Sonali, and my niece Pranusha :) – for having my back, being my unabashed champions, and creating in me a thirst for learning and education. They hold me to a high standard and help me achieve it. They have done so much for me during and before my Ph.D. and I know that they will continue to do so much for me after my Ph.D. that nothing I can say would sufficiently convey my love and appreciation for them. Special thanks to wife Sonali and her family for believing in me and supporting me in achieving my goals.

Contents

List of Figures

List of Tables

List of Symbols

Symbol	Description
N	Size of the corpus (in terms of words) i.e. number of words in all the documents of all the events
d_e	Document corresponding to topic e
$N_{e=1...\|E\|}$	number of words in d_e
$N_{e,w}$	Number of times word w appears in d_e
$\mathscr{V}$	Vocabulary and $\|\mathscr{V}\|$ is the size of the Vocabulary
$\mathcal{W}_{u=1...\|V\|,v=1...\|V\|}$	influence of node u on node v ($\|V\|$ is the number of nodes)
$\mathcal{W}$	$\|V\| \times \|V\|$ influence matrix
$\alpha_{w=1...\|\mathscr{V}\|}$	positive real, prior weight of word w in a topic; usually the same for all words
$\boldsymbol{\alpha}$	$\|\mathscr{V}\|$-dimension vector of positive reals, collection of all α_w values, viewed as a single vector (*parameter of the Dirichlet prior on the per-topic word distributions*)
$\beta_{k=1...K}$	positive real, prior influence of any topic i on topic k
$\boldsymbol{\beta}$	K-dimension vector of positive reals, collection of all β_k values, viewed as a single vector (*parameter of the Dirichlet prior on the topic-topic probability vector*)
$\gamma_{k=1...K}$	positive real, prior weight of topic k for a user u; usually the same for all topics; normally a number less than 1, e.g. 0.1, to prefer sparse topic distributions
$\boldsymbol{\gamma}$	K-dimension vector of positive reals, collection of all γ_k values, viewed as a single vector (*parameter of the Dirichlet prior on topic preference of each node/user*)
$z_{e=1...\|E\|}$	parent of event z_e
$\boldsymbol{z}$	$\|E\|$-dimension vector of event ids, i.e. collection of all z_e
$\zeta_{k=1...K,w=1...V}$	probability of word w occurring in topic k
$\boldsymbol{\zeta_k}$	distribution of words in topic k

Symbol	Description
$\boldsymbol{\zeta}$	matrix of dimension $K \times V$
$\mathcal{T}_{k=1\ldots K,k'=1\ldots K}$	probability of influence of topic k on other topic k' i.e. $P(\eta_e = k' \mid \eta_{z_e} = k)$
$\boldsymbol{\mathcal{T}_k}$	distribution of influence of topic k on other topics
$\boldsymbol{\mathcal{T}}$	topic-topic influence matrix of dimension $K \times K$
η_e	topic chosen for event e
$\boldsymbol{\eta}$	collection of all η_e ($\lvert E\rvert$-dimensional vector)
$\boldsymbol{\phi_v}$	distribution of preference over topics for user v
$\phi_{v=1\ldots V,k=1\ldots K}$	probability that user v chooses topic k i.e. $P(\eta_e = k \mid z_e = 0, c_e = v)$
$\boldsymbol{\Phi}$	$V \times K$ dimension user-topic preference matrix
z_e	Parent event for event e
$\boldsymbol{z}$	Collection of all z_e ($\lvert E\rvert$-dimensional)
$\mu_v(t)$	Base intensity for the Hawkes process at node v
$\boldsymbol{\mu}$	Collection of base intensities of all the nodes.
$E^{(C_e)}$	set of events those are child events of event e. i.e. $e' \in E \lvert z_{e'} = e$
$E^{([C_e])}$	$e \cup E^{(C_e)}$ (Closed set of child events of event e)
$E^{(\mathfrak{D}_e)}$	set of events those are descendants of event e
$E^{(\overline{\mathfrak{D}}_e)}$	$E \setminus \{e \cup E^{(\mathfrak{D}_e)}\}$
$\mathfrak{p}_e$	$\{(z_e, e)\}$ (edge between parent event i.e. z_e and event e)
C_e	$\{(e, e') \lvert e' \in E^{(C_e)}\}$ (set of all edges from event e to its child events)
$[C_e]$	$\{(e_1, e_2) \lvert e_1 \in \{z_e, e\}; e_2 \in E^{([C_e])}\}$
$\overline{C}_e$	$\{(e_1, e_2) \lvert e_1, e_2 \in E^{(\mathfrak{D}_e)}\}$ (set of edges among the descendants of event e)
$\mathfrak{D}_e$	$\{(e_1, e_2) \lvert e_1, e_2 \in E^{(\mathfrak{D}_e)}\}$
$\overline{\mathfrak{D}}_e$	$\{(e_1, e_2) \lvert e_1, e_2 \in E^{(\overline{\mathfrak{D}}_e)}\}$
$N_{l,l'} = \lvert\{\mathfrak{e}$	$(f, f') \in \mathfrak{E} \mid (\eta_f = l, \eta_{f'} = l')\}\rvert$
$N_{l,l'}^{(C_e)}$	$\lvert\{\mathfrak{e} = (e, e_2) \in C_e \mid (\eta_e = l, \eta_{e_2} = l')\}\rvert$
$N_{l,l'}^{(\overline{C}_e)}$	$\lvert\{\mathfrak{e} = (e_1, e_2) \in \mathfrak{D}_e \mid (\eta_{e_1} = l, \eta_{e_2} = l')\}\rvert$
$N_{l,l'}^{([C_e])}$	$\lvert\{\mathfrak{e} = (e_1, e_2) \in [C_e] \mid (\eta_{e_1} = l, \eta_{e_2} = l')\}\rvert$
$N_{l,l'}^{(\mathfrak{D}_e)}$	$\{\mathfrak{e} = (e_1, e_2) \in \mathfrak{D}_e \lvert (\eta_{e_1} = l, \eta_{e_2} = l')\}$

Symbol	Description
$N_{l,l'}^{(\overline{\mathfrak{D}}_e)}$	$\{(e_1, e_2) \in \overline{\mathfrak{D}}_e \vert (\eta_{e_1} = l, \eta_{e_2} = l')\}$
$\boldsymbol{N}_l^{(\mathcal{S})}$	$\vert\{(e_1, e_2) \in \mathcal{S} \vert (\eta_{e_1} = l, \eta_{e_2})\}\vert$, where $\mathcal{S}$ is some subset of edges
$\boldsymbol{N}_l^{(\neg\mathcal{S})}$	$\vert\{(e_1, e_2) \notin \mathcal{S} \vert (\eta_{e_1} = l, \eta_{e_2})\}\vert$, where $\mathcal{S}$ is some subset of edges

Chapter 1

Introduction

In the last two decades, there has been a lot of research in understanding the complex "interconnections" of modern society. The framework of *networks* has played a critical role in this research [30]. Primarily, networks can be defined as a set of objects along with *links* among some pairs of objects. Depending on the context, links can be defined using different forms of connections or relations. Because of the flexibility in the definition, networks are invoked in a wide range of domains such as epidemiology (social-sciences and diseases), financial trading (finance), mobility dynamics (social-sciences), protein-protein interaction network, gene regulatory network (biology), co-authorship and citation networks.

Online social networks can be thought of as analogous to the social networks formed by individuals in a society in the real world. Abstractly, these social networks are nothing but a collection of ties among individuals. Each node or user in a social network can be abstracted as an individual. And the ties or interaction between individuals can be thought of as relations, which are represented using edges between two users in the social network. Advent in technology and Internet has simplified distant communication & travel and digital interactions as well, and thus, these social networks among individuals are now not limited to the geographical boundaries of the society or individuals.

Information diffusion process or information propagation happens when a piece of information is shared or re-shared by a user and is consumed by other users in a network. That is when information flows from one user to another. This is also termed as information dissemination or information spread or information propagation. This sharing and re-sharing of information between users in social networks lead to a formation of cascades. Observe that, the formation of a cascade is because of the users taking decisions sequentially. That is, the later users in the cascade take a decision after observing the actions of the earlier users. A natural question to ask is, what are the factors that lead to these sequential decisions among the users? Brown and Reingen [9] show that the network topology or the structure of the network has a great impact on the extent and the patterns of information diffusion. Along with the network structure, the type or topic of information also has an impact on the dissemination of

information. For e.g., as shown in [78], repeated exposure of politically controversial topics have unusually large marginal effects on adoption, whereas, in case of conversational idioms, repeated exposures have a much less important marginal effect on the adoption. Weng et al. [90] show that it is not just that the spread of information is affected by the network structure; the communication activity among the users in the network also motivates the creation of links among the users in the network. That is, the flow of information among the users also shapes the structure of the network.

As people naturally choose to connect with those who share similar tastes, geography, ideologies, interests, or other aspects of life, social networks implicitly exhibit some community structure or groups within [70]. Generally, interconnections within a community are much more than the interconnections across the communities in social networks [69]. Because of the strong within-community connections, information circulates rapidly inside the community via the edges which are termed as "strong ties" (ties between the users with high emotional intensity and intimacy, frequency of communication, and reciprocal services), whereas, information rarely passes out of the communities via inter-community edges which are termed as "weak ties" (ties between the users with low emotional intensity and intimacy, frequency of communication, and reciprocal services) [40]. These weak ties also contribute significantly to the speed and the extent of information dissemination [40, 3].

1.1 Research Focus

Online social networking websites like Twitter, Facebook, Google Plus, Quora, Weibo, etc., act as platforms to facilitate interaction among the users and thus promote information sharing among these users. With the help of these platforms, billions of users daily share opinions, videos, photos, communicate ideas with their friend circle and followers around the world. With these interactions among the users, an exceptional amount of data is generated and this data can be used to study the mechanism of human communication and behavioral dynamics of the users in social networks [90]. Over time, with the development of technology, the way to interact has got simpler and the number of interactions has increased, but the complexity of understanding interactions has grown steadily as well [30]. Additionally, the set sources of high-quality information providers has grown and each such source might have different reliability, different intention, and wildly different perspective. Thus, understanding any piece of information requires understanding the way it is commended by and indicates other pieces of information within a large network of links [30].

In order to understand more about information propagation, the four components that play an important role are: the users generating the content, the type and topic of content, the network structure among the users, and actual the diffusion process [90]. The diffusion process factor, in particular, has received a lot of attention in the epidemiology domain. Starting with a simple diffusion model, i.e. SI

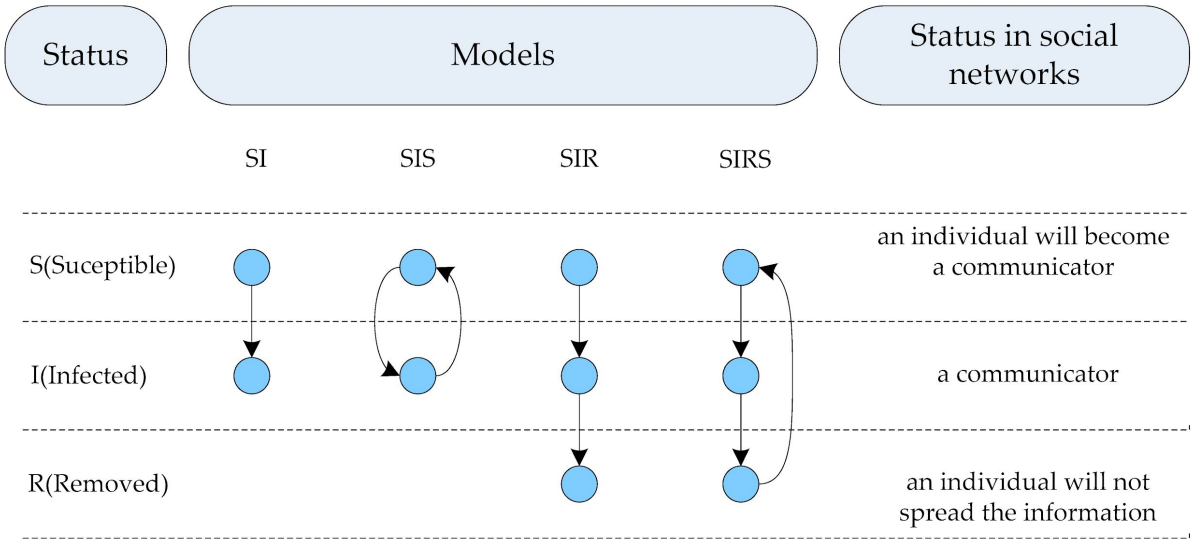


Figure 1-1: Comparison of four basic epidemic models (Image from: Li et al. [58])

(Susceptible-Infected) model [73], a number of followup models have been proposed: SIS (Susceptible, Infected, Susceptible) model [67, 68, 42], SIR (Susceptible, Infected, Removed/Recovered) [60], and the SIRS (Susceptible, Infected, Removed/Recovered, Susceptible) model [49]. Figure 1-1 shows the comparison of four basic epidemic models.

These epidemiology processes are basic and there have been improvements over these models as well [58]. But the contagion of information can be more complicated than the one in case of an epidemic because of the number of factors that play a critical role in the acceptance and transmission of information by a user [16]. Linear threshold (LT) model and the independent cascade (IC) model are two of the seminal models defined in the past that define a diffusion process and also consider the choices made by a user in the process. These models and their extensions are useful in understanding the problem of finding seed nodes in the network for maximizing the influence [51].

Generative models have been defined to model the process of information diffusion– these take into account various subsets of the possible factors responsible for the flow of information. These models are mostly based on modeling (marked) time-stamped events as per the theory of (marked) temporal point processes and Hawkes process to consider the excitation from the past events. The model defined by Simma and Jordan [84] provides the foundational framework for modeling time-stamped events. Linderman and Adams [59] and He et al. [48] utilize this framework to propose the Network Hawkes model and the Hawkes topic model (HTM) respectively. These models incorporate the network structure as well as user temporal dynamics. HTM

also successfully incorporates the content associated with the events in the model which is not the case for the Network Hawkes model.

Our research extends this thread by proposing novel generative models for a better understanding of the process of information diffusion. In addition to the factors considered by HTM, our models also incorporate interaction among the topics in the model. Table 1.1 gives a high-level overview of the models along with our models (HMHP and DNHP). We believe that our proposed generative models answer questions such as: 1) What are the factors that have an impact on the diffusion of information? 2) When and at what rate users generate content? 3) What are the topics of conversations among the users? 4) Why does a piece of information disseminate the way it does in the network? Additionally, this thesis also tries to understand the computational complexity of the information diffusion problem from the perspective of parameterized algorithms.

The following section gives an overview and also highlights the contributions of the thesis.

1.2 Thesis overview and contributions

The main interest of our research has been in understanding the factors that play an important role in the dissemination of information and thus model the process of information diffusion with the help of those factors. Additionally, our research also focuses on understanding the dynamics of processes like infection or virus propagation from the perspective of computational complexity by considering one of the simplest diffusion processes (the firefighting problem).

This dissertation is divided into two parts, where each part examines the information diffusion problem from a different aspect. The first part focuses on the modeling the process of information diffusion by proposing two models - Hidden Markov Hawkes Process (HMHP) (Chapter 3) and Dual Network Hawkes Process (DNHP) (Chapter 4). These models posit the generation of time-stamped text based cascades by users on a social networking platform, e.g. *Twitter*. The second part discusses the question of limiting the (mis-)information diffusion from the perspective of algorithms, specifically parameterized algorithms. This part proposes an algorithm for curbing the spread of infection considering a simpler and deterministic model for the spread of infection, formalized as the firefighting problem. Following is a brief overview of each of the parts along with the contributions.

1.2.1 Part 1 – Topical interactions: How information propagates over networks?

This part of the thesis highlights our generative models (HMHP and DNHP) for text-based cascades generated as a result of interaction among a set of users in an online

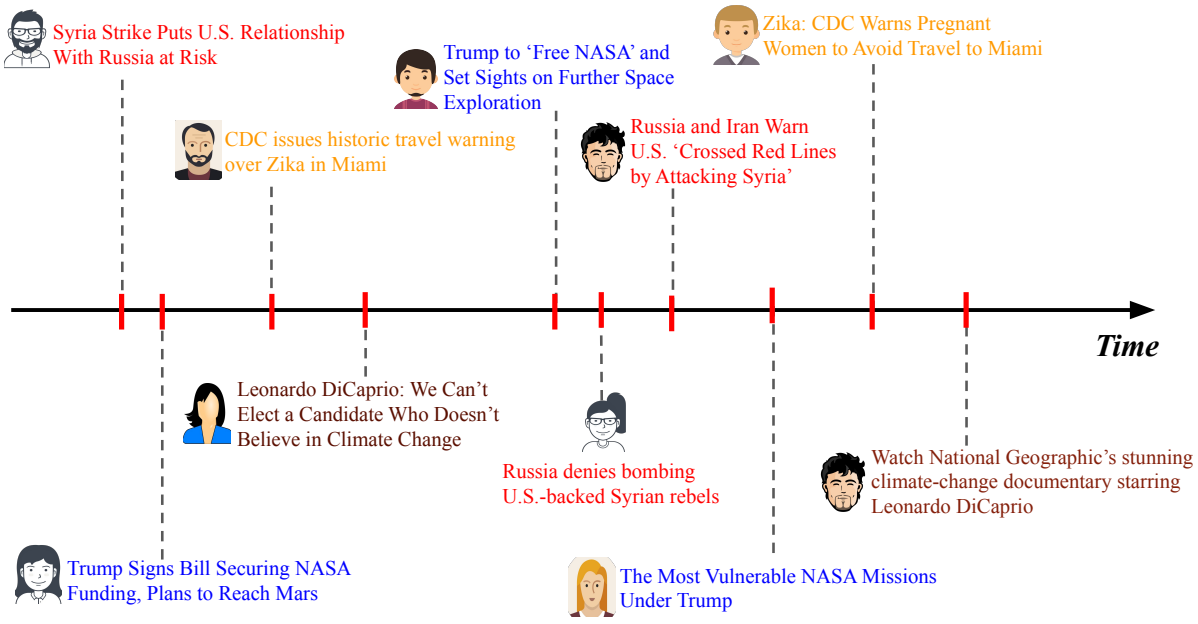


Figure 1-2: Example of time-series of tweets shown as conversations

social networking platform. As mentioned before, the process of dissemination of information from a user to another is dependent on a number of factors. Most of the time we can see only an activity or post posted by a user at a particular time. Figure 1-2 shows an example of time-stamped textual events. However, other relevant variables such as which *parent* influenced a user to generate this event, the topic of the parent event, and the user network influence strengths are mostly unobserved. And thus, to have a better understanding of the process of propagation of information these variables are to be inferred.

Step 1 – Models for information diffusion: (Chapters 3 and 4) We propose two models, viz. *Hidden Markov Hawkes Process* (*HMHP*) and *Dual Network Hawkes Process (DNHP)*, that posit the process of generation of the set of time-stamped events (tweets/documents) in a network of interacting users. HMHP and DNHP incorporate user-user interaction, user-topic patterns, rate of generation of events (tweets/documents), along with topic-topic interaction simultaneously. In contrast to other state-of-the-art models, HMHP and DNHP put forward the idea of *interacting topics* and also show its evidence with the help of experiments on real and synthetic datasets. Table 1.1 highlights various aspects captured by other the state-of-the-art models and our models.

Notice that, the time-series of events is nothing but number of conversations or cascades formed by interaction among the users in the network. Figure 1-3 gives a pictorial representation of the cascades over the time-series of posts. Each event e

User-N/W	Time-Series	Topic	Topic-N/W	Topic-Excitations	Models
×	√	√	×	√	Dir-Hawkes[28]
√	√	×	×	×	Simma-Jordan[84], Net-Inf[37], Net-Rate[38], MMHP[92], MMRATE [89], NHWKS[59]
√	√	√	×	×	HTM[48]
√	√	√	√	×	**HMHP**[17] (Chapter 3)
√	√	√	√	√	**DNHP** (Chapter 4)

Table 1.1: State-of-the-art Models.

here is a tuple denoted as $e = (t_e, c_e, z_e, \eta_e)$, where t_e is the time of the event, c_e is the user generating the event, z_e is the event that influenced this event i.e. the parent event, and η_e is the topic associated with this event. Additionally, there is a document or content associated with each event e and it is denoted by d_e. The variables z_e and η_e are latent for each event. The temporal component and the parent-child structure among the events are modeled using Hawkes process and the topical interactions are captured by combining the Hawkes process with Markov Chain over event topics over the parent-child structure among the events for the HMHP model. However, in the case of DNHP, the topical interactions are posited to have an effect on the intensity of the temporal Hawkes process defined for each user-topic pair.

Step 2 – Inference: *How can we efficiently infer the latent variables?* (Chapters 3 & 4) For each event, the topic associated with the event and parent event for the event is the latent variables which are to be inferred. Additionally, user-user network strengths parameters, topic-topic interaction parameters and the user base rate for each user to be inferred as well. It turns out that to infer these parameters it is sufficient to identify the cascades (as shown in Figure 1-4) from the time-series of events. Inferring the latent variables and the parameters results into understanding various aspects such as 1) user temporal dynamics (rate at which users generate content), 2) users topical preferences, 3) topics present in the conversations, 4) user-user influence, and 5) topic-topic interactions. Figure 1-5 gives a pictorial overview of the different aspects covered by both HMHP and DNHP.

In the case of DNHP, topic base rates are the additional set of parameters to be inferred. For both the models, we develop a Gibbs sampling based algorithm for the inference task. HMHP benefits from collapsed Gibbs sampling as the topic-topic interaction parameters can be integrated out. This is not the case for DNHP because of

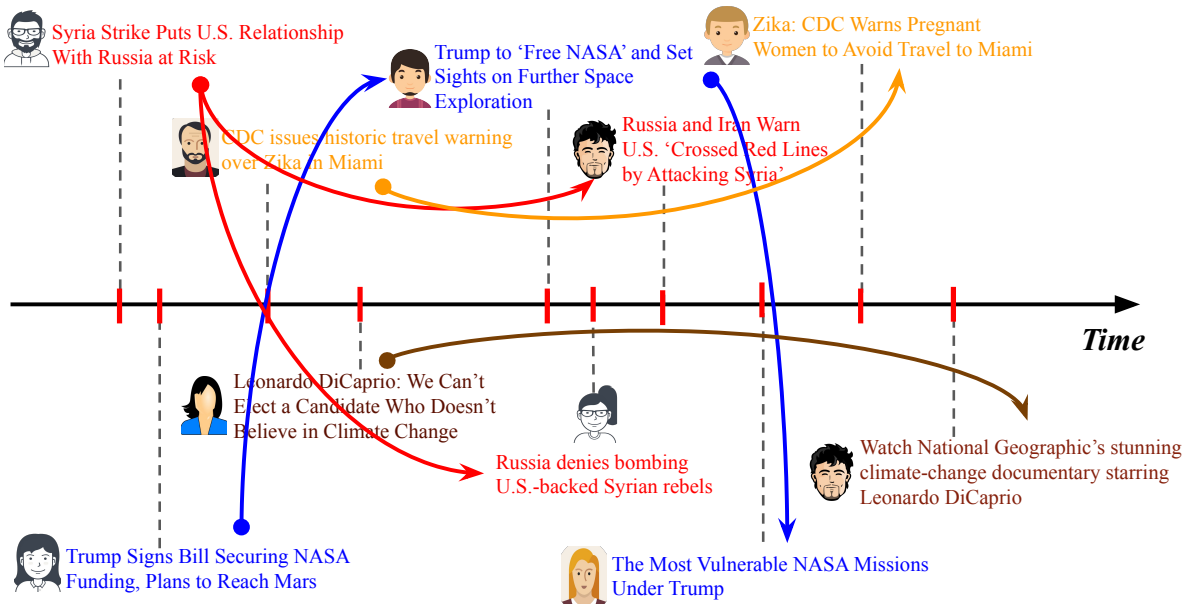


Figure 1-3: Example of time-series of tweets shown as conversations

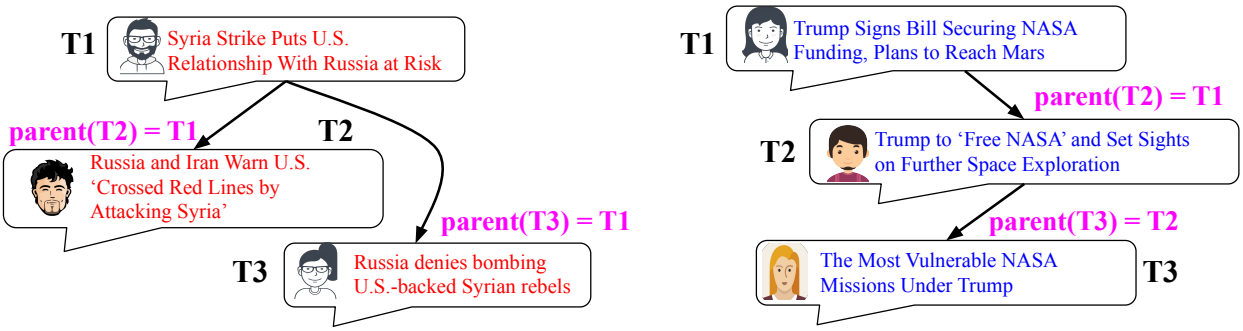
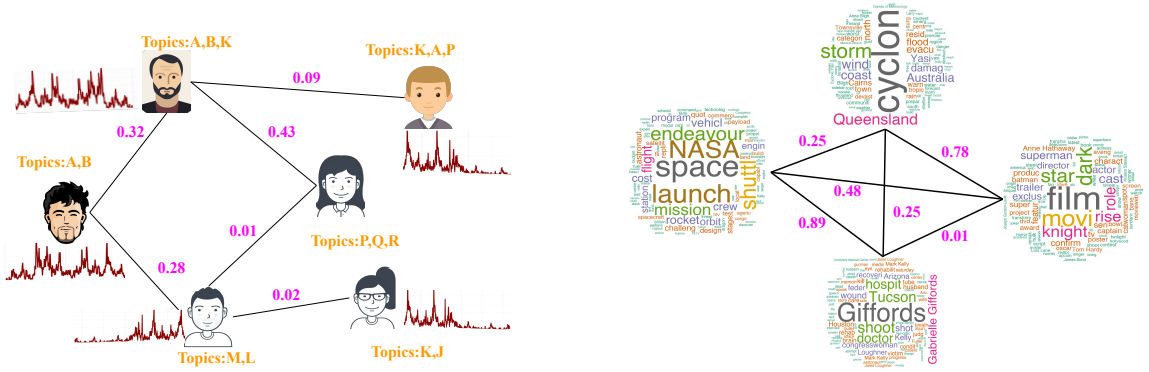


Figure 1-4: Example cascades or conversations



(a) User-User influence, User topic preferences, User temporal dynamics

(b) Topics and Topic-Topic interaction

Figure 1-5: Various inference tasks by the models HMHP and DNHP

the interconnection between the user-user and the topic-topic interaction parameters. Therefore, additional $O(K^2)$ parameters are to sampled in each iteration of DNHP, where K is the number of topics.

Contributions:

- To the best of our knowledge, our model (HMHP) is the first model that jointly models topical interactions along with user-user influence and user-topic patterns.
- Experiments on real and semi-synthetic data show that HMHP generalizes better and also recovers diffusion paths, network strengths, and topics more accurately as compared to other state-of-the-art models.
- More interestingly, HMHP also finds insightful cooperation among topics in real world tweet dataset, which, from what we know, other existing models could not. We believe that this could potentially lead to actionable insights enabling, e.g., user targeting for influence maximization. Figure 1-6 show some anecdotal results as a result of incorporating topical interactions in HMHP.
- Additionally, we developed another model, i.e. DNHP, in which the event rates are dependent not only on the user but also on the topic (or mark) associated with the event. This differs from all the state-of-the-art models for the text-based cascades which generate the topic or mark associated with the event independently post event time generation.
- As opposed to HMHP or other models that model text-based cascades, in DNHP events propagate over two dimensional space which is indexed by users and topics. This transaction over two dimensional space establishes a synergy between the two spaces. The topic-topic weights and the user-user weights are now

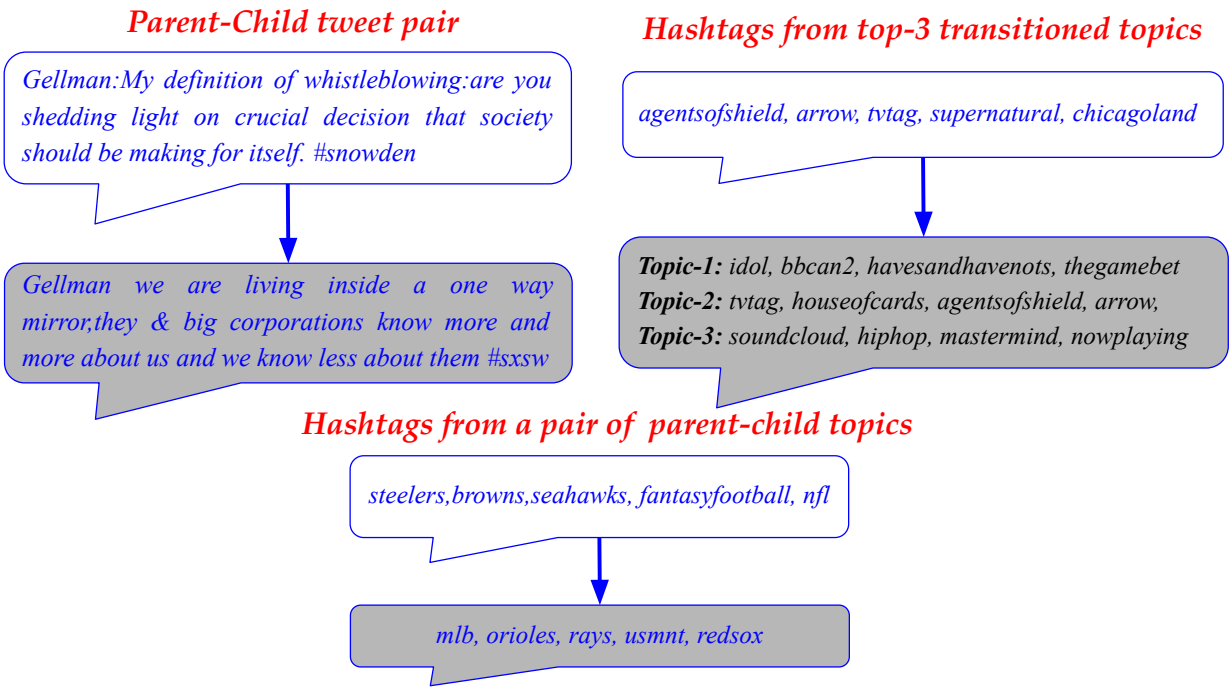


Figure 1-6: *Parent-Child tweet pair:* Both these tweets belong to different topics and interaction among these topics is found with high frequency. *Hashtags from top-3 transitioned topics:* Starting from a topic with hashtags related to TV shows mostly transitioned to topics with the hashtags related to entertainment broadly. *Hashtags from pair of parent-child topics:* Topic which has hashtag related to a sport (American Football) transitions to the topic which has hashtags related to some other sport (Baseball).

interdependent and the joint estimation allows evidence to flow across these variables.

- On semi-synthetic and real datasets, DNHP outperforms the other state-of-the-art models.

1.2.2 Part 2 – On the computational complexity of suppressing the spread of (mis-)information over networks

This part of the thesis discusses the information diffusion problem through the lens of parameterized algorithms. In this part of the thesis, the focus is on designing algorithms to curb the spread of mis-information or infection in the network. To define the problem formally here, we take the firefighting problem into consideration and propose results for the objective of saving a critical set from an infection (fire) in a network. The firefighting problem can be thought of as a game between two players playing alternately – fire and the firefighter. Here, fire is analogous to an infection or virus in a network and the firefighter is analogous to inoculation or

vaccination. The diffusion process in the firefighting problem is deterministic and is simpler as compared to the real world. At every even time-step, fire (infection) spreads deterministically to all its neighbors which are not saved by the firefighter. At every odd time step firefighter gets a chance to save a node or subset of nodes in the network. Once a node is saved, the fire cannot pass through this node.

Informally, in the area parameterized algorithms the goal is to design an algorithm (for a hard problem) that has time complexity of the form $f(k)O(n^c)$, where k is the parameter associated with the problem into consideration and c is some constant independent of k. Problems for which algorithms with the time complexity of the form $f(k)O(n^c)$ can be designed are termed as *fixed parameter tractable* problems. Here, for the firefighting problem with the objective of saving a critical set (SACS) with at most k firefighters (k is the parameter associated with the problem), it turns out that, the problem is fixed parameter tractable for general graphs. Moreover, when the graph is restricted to trees, we can design a faster algorithm with the help of *important separators* [19]. However, the important separators do not work in case of general graphs and so we use the notion of *tight separator sequence* [61] to propose the final algorithm in case of general graphs. (All the results and the required preliminaries associated with this part of the thesis are explained in Chapter 5.)

Contributions:

- With the help of a reduction, we show that saving a critical set problem is NP-complete even when the size of the critical set is 1.
- We develop a fixed parameter tractable algorithm for SACS problem on a graph.
- We also propose a faster algorithm for the SACS problem when the input graph is a tree.
- Moreover, we also show that it is not possible to preprocess the tree such that the size of the tree is polynomial in k and solve the SACS problem (*kernelization*), where, k is the parameter (number of allowed firefighters).
- Also, from what we know, our work is the first work that exploits the connection between the notion of *important* & *tight separators* and the firefighting problem.
- Additionally, for a slightly different and related model, that is the "Spreading model", defined in [1], we show solving the SACS problem is as hard as solving the DOMINATING SET problem.

PART - I
Models for Information Diffusion

Chapter 2

Preliminaries

This chapter gives an introduction to point processes and also discusses algorithms to simulate these processes. A point process can be thought of as a succession or series of points on a real number line. Different point processes are defined by the patterns of how these points are located on the real number line. We start with the definition of point processes first and then explain one of the simplest point processes, which is the Poisson process. A (homogeneous) Poisson process is characterized by an intensity function λ, which is a constant value. That is, in case of homogeneous Poisson process there is an assumption that the points are roughly evenly spread over any interval. Homogeneous Poisson process can also be extended to an inhomogeneous Poisson process by allowing the intensity to vary according to a deterministic function $\lambda(t)$. Poisson processes are used to model various temporal phenomena e.g. times of arrival of the buses at a bus stop, the phone calls at a help center, radioactive decay, customers arriving at a restaurant, events of traffic, activities of customers on a website, etc. Poisson processes are, however, not a good choice to model the settings where the occurrence of events in the future is dependent on the events in the past. For example, earthquakes, where there are aftershocks post the main event, stock prices, activity of a user on a social networking website, etc. To explain or model these processes, where the events are dependent on past events, the Hawkes process was developed. In this chapter after explaining the Poisson processes, we eventually move on to the Hawkes process.

2.1 Introduction

A point process is a random process realized by a sequence of random events $\{s_i\}$ where $s_i \in \mathbb{R}_+$, $i \in \mathbb{Z}^+$, and the events can be abstracted as points belonging to some set, such as location or time. A Point process is characterized with the help of an associated (conditional) intensity function, which quantifies the likelihood of Point processes turn out to be a perfect fit in number application domains such as finance, seismology, social media, disease modeling, etc. For example, in the financial domain,

the buying or selling of stocks over time in the stock market represents the events [2, 85, 54]. In the case of seismology, an earthquake represents an event and the event data primarily consists of the location of the earthquake [8, 88]. And, in case of online social media, the actions (for eg. liking a post, upvote, creating a new post, retweet, etc.) performed by the users over time represent the events [93, 48, 17].

Definition 1 (Point Process ([83] pg. 1)): *Let $(\Omega, \mathcal{F}, P)$ be a probability space. A point process on $\mathbf{R}^+$ is a sequence of non-negative random variables $\{T_k\}_{k=1,2,\ldots}$, such that for all k, $T_k \leq T_{k+1}$.*

T_k can be interpreted as the time of occurrence of the k^{th} event. Such processes are known as temporal point processes. The time between the consecutive events is termed as interarrival time. Also, when the points are distributed in a two dimensional space, the process is called a spatial point process. Indeed, the notions of interarrival times and counting processes that follow are naturally tailored to the temporal point process.

Definition 2 (Interarrival time ([80] p. 64)): *Consider a temporal point process $\{T_k\}_{k=1,2,\ldots}$ Define $W_k = T_k - T_{k-1}$ with the convention $T_0 = 0$. The sequence $\{W_k\}_{k=1,2,\ldots}$ is called the interarrival times.*

Definition 3 (Counting Process ([77])): *A counting process associated with the temporal point process $\{T_k\}_{k=1,2,\ldots}$ represented as N_t or $N(t)$ is a random function defined on time $t \geq 0$, taking values in $\mathbb{Z}^+$, that counts the number of occurrences up to and including time t, i.e.*

$$N(t) = \sum_{k=1,2,\ldots} \mathbb{1}_{\{T_k \leq t\}} \tag{2.1}$$

Definition 4 (Intensity of point process ([52])): *The intensity of a point process is defined as*

$$\lambda(t) = \lim_{h \to 0^+} \frac{P\{N(t, t+h] > 0\}}{h} \tag{2.2}$$

$N(t, t+h]$ indicate the count of the number of events in an interval of length h. $\lambda(t)$ intuitively measures the rate at which the points/events occur.

2.2 Poisson Process

One of the simplest families of point processes is the class of the Poisson process. In this section, we present the definition and some properties of the Poisson process.

Definition 5 (Poisson Process ([80] p. 314−315, Definition 5.3)): *The point process N is a (homogeneous) Poisson process with rate $\lambda, \lambda > 0$, if and only if, for all $t \geq 0$ and $h \to 0^+$*

(i) $N(0) = 0$

(ii) *The process has independent increments.*

(iii) $P\{N(t+h) - N(t) = 1\} = \lambda h + o(h)$

(iv) $P\{N(t+h) - N(t) \geq 2\} = o(h)$

where, $o(h)$ *is a quantity for small* h*, which is the lower order of* h*, i.e.*

$$\lim_{h \to 0^+} \frac{o(h)}{h} = 0$$

An equivalent definition of Poisson process in terms of the counting process is defined as follows:

Definition 6 ([80] p. 313, Definition 5.1): *The counting process* $\{N(t), t \geqslant 0\}$ *is said to be a homogeneous Poisson process having rate* $\lambda, \lambda > 0$*, if*

(i) $N(0) = 0$

(ii) *The process has independent increments.*

(iii) *The number of events in any interval of length* t *is Poisson distributed with mean* λt*. That is, for all* $s, t \geqslant 0$

$$P\{N(t+s) - N(s) = n\} = e^{-\lambda t} \frac{(\lambda t)^n}{n!}, \quad n = 0, 1, \ldots \tag{2.3}$$

Therefore, from condition (iii) it follows that Poisson process has stationary increments and also

$$E[N(t)] = \lambda t$$

Thus, λ is known as the rate of the process.

The following proposition shows an interesting property of the interarrival times in case of (homogenous) Poisson process.

Proposition 7. *([79] p. 64 Proposition 2.2.1) The interarrival times* $\{W_k\}_{k=1,2,\ldots}$ *of a (homogeneous) Poisson process with rate* $\lambda > 0$*, are independent and identically distributed exponential random variables having mean* $1/\lambda$*.*

The following theorem gives the joint density of n events that occur in a time interval $[0, T]$.

Theorem 8. *([15] Theorem 3.3) The joint density that exactly* n *events occur in the interval* $[0, T]$ *for a Poisson process with rate* λ *taking values* $0 < t_1 < t_2 < \cdots < t_n \leq T$ *is*

$$f_{T_1,\ldots,T_n}(t_1, \ldots, t_n \mid T_{n+1} > T) = \lambda^n e^{-\lambda T}$$

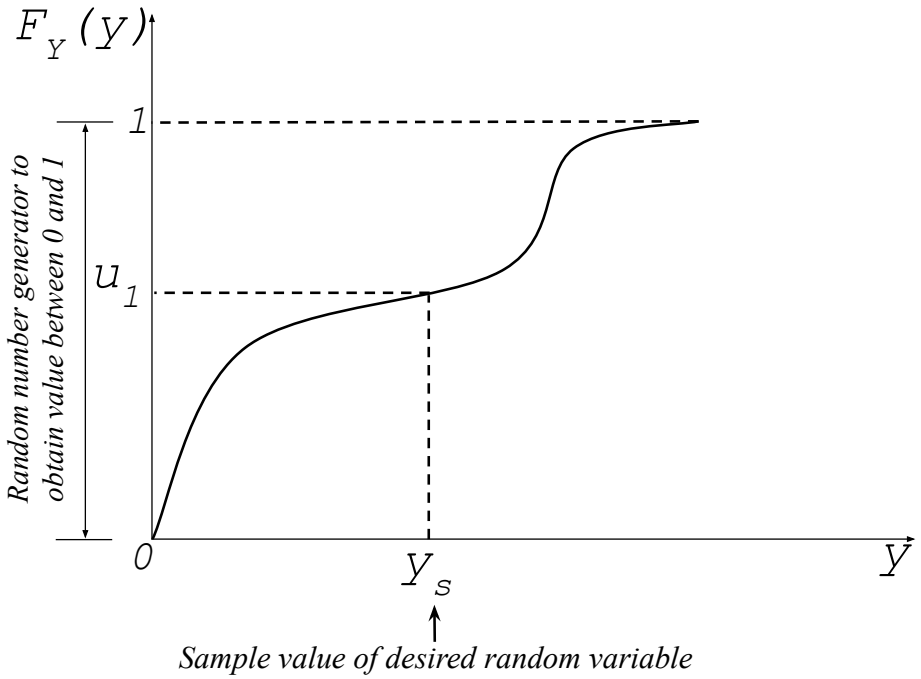


Figure 2-1: Illustration of inversion method

2.2.1 Simulation for homogeneous Poisson process

Here, we describe one of the methods to generate events as per a homogeneous Poisson process. The property that is exploited in generating events is that the interarrival times are independent and exponentially distributed with mean $1/\lambda$. Thus, to generate events as per a homogeneous Poisson process it suffices to generate independent exponentially distributed random variables. And one of the simplest methods to generate exponentially distributed points is the *inversion* method.

Inversion Method: This method of generating sample values can be applied in all cases where an explicit expression for the cumulative distribution function (CDF) of a random variable exists. The CDF for a random variable (say Y) $F_Y(y)$ is a non-decreasing function and is bounded between 0 and 1, i.e. $0 \leq F_Y(y) \leq 1$ for all y. Which says that there is a mapping between the CDF and a random variable between 0 and 1. Therefore, then by sampling a random number u_1 for a uniform random variable U and then taking an inverse image of u_1 on the abscissa a sample value y_s of a random variable Y can be generated. Figure 2-1 gives an illustration of this method. The reason that it works is because:

$$P(Y \leq y_s) = P(U \leq F_Y(y_s)) = F_Y(y_s)$$

Now, for exponential random variable X with a probability density function (pdf)

$f_X(t) = \lambda e^{-\lambda t}$ for $t \geq 0$, the CDF can be written as:

$$F_X(t) = \int_0^t \lambda e^{-\lambda \tau} d\tau = 1 - e^{-\lambda t} \qquad \text{for all } t \geq 0$$

Let u_1 be a sample from a uniformly distributed random variable U, then following the inversion method, we have:

$$u_1 = F_X(t) = 1 - e^{-\lambda t}$$

which implies that

$$t = \frac{-\ln(1-u_1)}{\lambda} = \frac{-\ln(u_2)}{\lambda}$$

where, $u_2 = (1 - u_1)$ is also a uniformly distributed random variable.

Thus, to get a sample from an exponentially distributed random variable with rate parameter λ, it is enough to sample a point $u \sim Uniform[0, 1]$, then w, such that $w = \frac{-\ln(u)}{\lambda}$ will be the required sample.

As the time between the events in case of homogeneous Poisson process is exponentially distributed, Algorithm 1 describes a pseudo-code to generate samples for homogeneous Poisson process in an interval $[0, T]$ as per the inversion method.

Algorithm 1: Simulating events using Homogeneous Poisson Process with rate λ in an interval $[0, T]$ ([15] Algorithm 1)

Data: λ, T
Result: Events with rate λ.
Initialize $n = 0$, $t_0 = 0$;
while *True* **do**
  Generate $u \sim uniform(0, 1)$;
  Let $w = -\ln u/\lambda$;
  Set $t_{n+1} = t_n + w$;
  if $t_{n+1} > T$ **then**
    return $\{t_k\}_{k=1,2,\ldots,n}$;
  else
    Set $n = n + 1$;
  end
end

We will now move on to an inhomogeneous Poisson process which is a bit more complex as compared to the homogeneous Poisson process.

2.3 Inhomogeneous Poisson Process

The properties of the inhomogeneous Poisson process are almost similar to that of the homogeneous Poisson process. In the case of an inhomogeneous Poisson process, the intensity λ varies as a function of t. Here is the formal definition for the inhomogeneous Poisson Process.

Definition 9 (Inhomogenous Poisson Process ([80] p. 339, Definition 5.4)): *The point process N is said to be an inhomogeneous Poisson process with intensity function $\lambda(t) \geq 0, t \geq 0$, if*

(i) $N(0) = 0$

(ii) *The process has independent increments.*

(iii) $P\{N(t+h) - N(t) = 1\} = \lambda(t)h + o(h)$, *as* $h \longrightarrow 0$

(iv) $P\{N(t+h) - N(t) \geq 2\} = o(h)$, *as* $h \longrightarrow 0$

As similar to that of the homogeneous Poisson process, the following proposition talks about the probability of a number of events in an interval for an inhomogeneous Poisson process.

Proposition 10 ([15] Proposition 4.1)**.** *Let the point process N be an inhomogeneous Poisson process with intensity function $\lambda(t)$, then for any $t \geq 0$, the random variable $N(t)$ follows a Poisson distribution with parameter $\int_0^t \lambda(s)ds$, i.e.*

$$P\{N(t) = n\} = \frac{e^{-\int_0^t \lambda(s)ds} \left(\int_0^t \lambda(s)ds\right)^n}{n!} \qquad n \in \mathbb{Z}^+ \tag{2.4}$$

2.3.1 Simulation for Inhomogeneous Poisson Process

Lewis and Shedler in 1979 [57] proposed that an inhomogeneous Poisson process can be simulated using a "thinning" algorithm, which can be thought of as similar to that of *rejection sampling*. Before going to the theorem (Theorem 11) that mathematically justifies the correctness of the thinning algorithm, we first give an informal overview of the algorithm and illustrate how it works.

Thinning Algorithm: This method can be used to generate samples for any random variable that has values within a finite range and has a bounded probability density function (pdf). This method is based on the geometrical interpretation of the pdf of the random variable. Consider a random variable X with a pdf $f_X(x)$, with the maximum value of the pdf denoted by C, and let X take values between A and B ($[A, B]$). Now to generate samples of X follow the steps as follows:

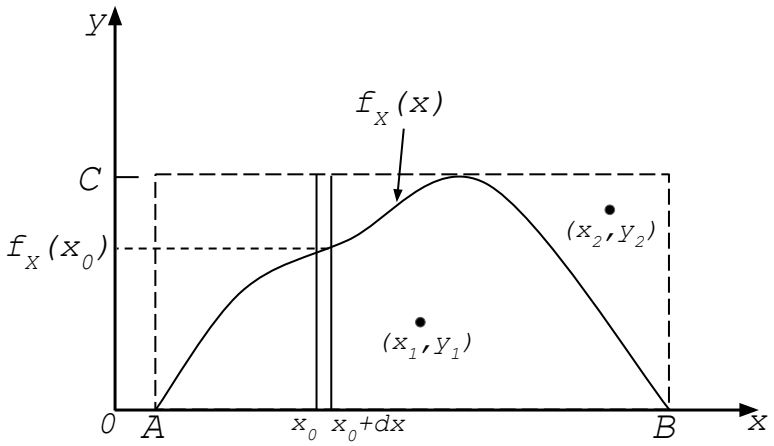


Figure 2-2: Illustration of Thinning method for generating samples from a random variable X

1. Generate two uniform random numbers u_1 and u_2 between 0 and 1. Let $x_0 = u_1 \times (B - A)$ and $y_0 = u_2 \times C$.
2. If the number y_0 is below the pdf at x_0 i.e. $y_0 \leq f_X(x_0)$, then accept x_0 as a sample for X, else, reject and continue the process from step 1.

Figure 2-2 shows a pictorial representation of this procedure. The reason that this procedure of sampling points works is: Observe that, the points generated according to step 1 of the above procedure are uniformly distributed over the region formed by the rectangle with area $(B-A)\times C$. Thus, for any point with the x-coordinate between x_0 and $x_0 + dx$ we can say that:

$$P(\text{acceptance of point } | x_0 \leq x \leq x_0 + dx) = \frac{f_X(x) \cdot dx}{C \cdot dx} = \frac{f_X(x)}{C}$$

Theorem 11 formally justifies the correctness of the "thinning" algorithm, where, first the points are generated according to homogeneous Poisson process with intensity $\bar{\lambda} \geq \lambda(t)$ and then are removed with probability $1 - \frac{\lambda(t)}{\bar{\lambda}}$, where $\lambda(t)$ is the intensity of the required inhomogeneous Poisson process.

Theorem 11 (Lewis and Shedler, 1979 [57]). *Consider a homogeneous Poisson process with intensity $\bar{\lambda}$. Suppose that $\bar{t}_1, \bar{t}_2, \ldots, \bar{t}_{\bar{N}(T)}$ are random variables representing event times from the homogeneous Poisson process with rate function $\bar{\lambda}$, and lying in the fixed interval $(0, t_0]$. Let $\lambda(t)$ be a rate function such that $0 \leq \lambda(t) \leq \bar{\lambda}$ for all $t \in [0, t_0]$. If the i^{th} event time T_i^* is independently deleted with probability $1 - \frac{\lambda(t)}{\bar{\lambda}}$ for $i = 1, 2, \ldots, n$, then the remaining event times form an inhomogeneous Poisson process with rate function $\lambda(t)$ in the interval $(0, T]$*

Algorithm 2 describes the pseudo-code for generating random variates from an inhomogeneous Poisson process which follows from Theorem 11.

Algorithm 2: Simulating events using Inhomogeneous Poisson Process with rate $\lambda(t)$ in an interval $[0, T]$ ([57] p.7, Algorithm 1)

Data: $\lambda(\cdot), T$
Result: Events with rate λ.
Initialize $n = m = 0, t_0 = s_0 = 0, \overline{\lambda} = \sup_{0 \le t \le T} \lambda(t)$;
while $s_m < T$ **do**
  Generate $u \sim uniform(0, 1)$;
  Let $w = -\ln u / \overline{\lambda}$;
  Set $s_{m+1} = s_m + w$;
  Generate $D \sim uniform(0, 1)$;
  if $D < \lambda(s_{m+1}) / \overline{\lambda}$ **then**
    $t_{n+1} = s_{m+1}$;
    $n = n + 1$;
  end
  $m = m + 1$;
end
if $t_{n+1} \ge T$ **then**
  return $\{t_k\}_{k=1,2,\ldots,n}$;
else
  return $\{t_k\}_{k=1,2,\ldots,n-1}$;
end

Notice that the only constraint on $\overline{\lambda}$ is $\lambda(t) \le \overline{\lambda}$ for all t in $[0, T]$. The higher the value of $\overline{\lambda}$ more is the number of events according to the homogeneous Poisson process with intensity $\overline{\lambda}$. But, observe that, the value of $\overline{\lambda}$ determines the probability of rejection of events from the final process with intensity $\lambda(t)$. A higher value of $\overline{\lambda}$ corresponds to a smaller probability of acceptance of points in the process with intensity $\lambda(t)$. Therefore, in practice, the value of $\overline{\lambda}$ should be as close to supremum/maximum of $\lambda(t)$ as possible for higher efficiency (lesser iterations in the while loop in Algorithm 2).

2.4 Hawkes Process

In the case of the Poisson process, let it be homogeneous or inhomogeneous, the events arrive independently of the events in the past. However, this assumption is not appropriate for all applications. In some settings, the rate at which events arrive is influenced by the number of events that have already happened in the past. For e.g., in the case of earthquakes, as mentioned before, the aftershocks are dependent on the main event. The processes where the rate of events depends on past events are termed as *self-exciting processes*. Hawkes process is one of the well-known self-exciting point processes. In this section, we first formally define the self-exciting process and

then move on to the Hawkes process.

Definition 12 (Self-exciting point process (Donald L. Snyder, 1991, p. 287 [86])): *A point process is called self-exciting if the intensity $\lambda(\cdot)$ depends not only on time t but also the entire past of the point process.*

The following definition describes the past or history of point process formally and mathematically.

Definition 13 (Natural filtration (Grandell, 1997, p. 51 [39])): *For any process $N(t)$, the natural filtration $F^N = \left(\mathcal{F}_t^N; t \geq 0\right)$ defined by*

$$\mathcal{F}_t^N = \sigma\{N(s); s \leq t\}$$

Here, $\mathcal{F}_t^N$ is the σ-algebra generated by N up to time t, and represents the internal history of N up to time t [15].

Definition 14 (Stochastic intensity function (Last and Brandt, 1995, p. 10 [])): *Let $N(t)$ be a point process with natural filtration $\mathcal{F}_t^N$. The process defined by*

$$\lambda\left(t|\mathcal{F}_{t^-}^N\right) = \lim_{h \to 0^+} \frac{P\left\{N(t+h) - N(t) > 0|\mathcal{F}_t^N\right\}}{h}$$

is called the stochastic intensity function of the point process.

2.4.1 Univariate Hawkes Process

Definition 15 (Hawkes Process (Hawkes, 1971, p. 84 [47])): *Let $N(t)$ be a univariate simple point process satisfying*

(i) $N(t) = 0$,

(ii) *where $\lambda(t)$ is a stochastic process given by*

$$\lambda(t) = \mu + \sum_{\{k:t_k<t\}} \alpha e^{-\beta(t-t_k)}$$

where $mu > 0$ and $0 < \alpha < \beta$

(iii) *$\lambda(t)$ is the stochastic intensity of the point process*

$$P\left\{N(t+h) - N(t) = 1|\mathcal{F}_{t-}^N\right\} = \lambda(t)h + o(h)$$

(iv) *The point process is orderly, i.e.*

$$P\left\{N(t+h) - N(t) \geq 2|\mathcal{F}_{t-}^N\right\} = o(h)$$

is called a univariate Hawkes process with exponential decay on $[0, \infty)$.

2.4.2 Simulating events using Hawkes Process

Observe that, to generate samples according to Hawkes process, a thinning algorithm similar to the one in case of inhomogeneous Poisson process can be used. Algorithm 3 describes the pseudo-code to generate events as per a Univariate Hawkes process with intensity given by $\lambda(t) = \mu + \sum_{\{k:t_k<t\}} \alpha e^{-\beta(t-t_k)}$. Algorithm 3 is similar to that of the algorithm 2, the difference is just in the intensity function. We still mention it here for completeness.

Algorithm 3: Simulating events using Univariate Hawkes Process with rate $\lambda(t)$ in an interval $[0, T]$ (*Ogata,* 1981*, Algorithm* 2, [71])

Data: $\lambda, T, \alpha, \beta$
Initialize $n = 0,\ t_0 = s = 0,\ \mathcal{T} = \varnothing$;
while $s < T$ **do**
  Let $\bar{\lambda} = \lambda(s^+) = \mu + \sum_{\{\tau\in\mathcal{T}\}} \alpha e^{-\beta(s-\tau)}$;
  Generate $u \sim uniform(0, 1)$;
  Let $w = -\ln u/\bar{\lambda}$;
  Set $s = s + w$;
  Generate $D \sim uniform(0, 1)$;
  if $D\bar{\lambda} < \lambda(s) = \mu + \sum_{\{\tau\in\mathcal{T}\}} \alpha e^{-\beta(s-\tau)}$ **then**
    $t_{n+1} = s$;
    $n = n + 1$;
  end
end
if $t_{n+1} \le T$ **then**
  return $\{t_k\}_{k=1,2,\ldots,n}$;
else
  return $\{t_k\}_{k=1,2,\ldots,n-1}$;
end

2.4.3 Multivariate Hawkes Process (MHP)

Multivariate Hawkes process (MHP) is an extension of the univariate Hawkes process to the multivariate case. In addition to the self-excitation feature, which is inherited from the univariate case, the Multivariate Hawkes process additional possesses the cross-excitation feature, which is between different dimensions.

Definition 16 (Multivariate Hawkes Process (MHP) (Hawkes, 1971**, p.** 86 **[**47**])):** *Let* $\boldsymbol{N}(t) = \left(N^1(t), N^2(t), \ldots, N^M(t)\right)$ *be a simple multivariate point process satisfying*

(i) $N(0) = 0$, *and for* $m = 1, 2, \ldots, M$

(ii) $\lambda^m(t)$ *is a stochastic process given by*

$$\lambda^m(t) = \mu_m + \sum_{n=1}^{M} \sum_{\{k:t_k^n<t\}} \alpha_{mn} e^{-\beta_{mn}\left(t-t_k^n\right)}$$

where $\mu_m > 0$, $\alpha_{mn} \geq 0$ *and* $\beta_{mn} \geq 0$ *for* $m, n = 1, 2, \ldots, M$

(iii) $\lambda^m(t)$, *independently for each* m, *is the stochastic intensity of the point process* $N^m(t)$

$$P\left\{N^m(t+h) - N^m(t) = 1 | \mathcal{F}_{t^-}^N\right\} = \lambda^m(t)h + o(h)$$

(iv) The point process is orderly, i.e.

$$P\left\{N(t+h) - N(t) \geq 2 | \mathcal{F}_{t^-}^N\right\} = o(h)$$

where $\mathcal{F}_t^N$ *is the natural filtration of the process, is called an M-variate Hawkes process with exponential decays on* $[0, \infty)$

2.4.4 Simulating events using MHP

Simulation of events as per the multivariate Hawkes process is similar to that of the univariate case (thinning algorithm), with an additional step that decides to which dimension an accepted point belongs. Proposition 17 justifies mathematically the correctness of the thinning based algorithm (Algorithm 4) for generating points according to Multivariate Hawkes process. Proposition 17 states that the accepted points must be distributed to each dimension with probability proportional to the intensity of the respective dimension.

Proposition 17 ((Ogata, 1981 p. 24, Proposition 1 [71])). *Let* $N(t) = \left(N^1(t), N^2(t), \ldots, N^M(t)\right)$ *be an Multivariate point process on an interval* $[0, T]$ *with stochastic intensities* $\lambda^{m*}(t) = \lambda^m\left(t|\mathcal{F}_{t^-}^N\right)$ *for* $m = 1, 2, \ldots, M$. *Suppose there is a one-dimensional* $\mathcal{F}_{t^-}^N$ *process* $\bar{\lambda}^*(t)$ *which is defined path-wisely satisfying*

$$\sum_{m=1}^{M} \lambda^{m*}(t) \leq \bar{\lambda}^*(t), \quad 0 < t \leq T$$

and set

$$\lambda^{0^*}(t) = \bar{\lambda}^*(t) - \textstyle\sum_{m=1}^{M} \lambda^{m^*}(t)$$

Let $\bar{t}_1, \ldots, \bar{t}_{N(T)} \in (0, T]$ *be the points of the process* $\bar{N}(t)$ *with stochastic intensity* $\bar{\lambda}^*(t)$. *For each of the points,* $\bar{t}_k$ *where* $k = 1, 2, \ldots, \bar{N}(T)$ *attach a mark* $m = 0, 1, \ldots, M$ *with probability* $\lambda^{m*}\left(\bar{t}_k\right) / \bar{\lambda}^*\left(\bar{t}_k\right)$, *respectively. Then the points with marks* $m = 1, 2, \ldots, M$, *provide a Multivariate point process with stochastic intensities* $\lambda^{m^*}(t)$

Algorithm 4: Simulating events using Multivariate Hawkes Process with rate $\lambda(t)$ in an interval $[0, T]$ ([71])

Data: $\boldsymbol{\mu}_{M\times 1}, \boldsymbol{\alpha}_{M\times M}, \boldsymbol{\beta}_{M\times M}, T$

Initialize $\mathcal{T}^1 = \cdots = \mathcal{T}^M = \varnothing, n^1 = \cdots = n^M = 0, s = 0$;

while $s < T$ **do**

  Let $\bar{\lambda} = \sum_{m=1}^{M} \lambda^m(s^+) = \sum_{m=1}^{M} \left(\mu_m + \sum_{n=1}^{M} \sum_{\tau\in\mathcal{T}^n} \alpha_{mn} e^{-\beta_{mn}(s-\tau)}\right)$;

  Generate $u \sim uniform(0, 1)$;

  Let $w = -\ln u/\bar{\lambda}$;

  Set $s = s + w$;

  Generate $D \sim uniform(0, 1)$;

  if $D\bar{\lambda} \leq \sum_{m=1}^{M} \lambda^m(s) = \sum_{m=1}^{M} \left(\mu_m + \sum_{n=1}^{M} \sum_{\tau\in\mathcal{T}^n} \alpha_{mn} e^{-\beta_{mn}(s-\tau)}\right)$ **then**

    $k = 1$;

    while $D\bar{\lambda} > \sum_{m=1}^{k} \lambda^m(s)$ **do**

      $k = k + 1$;

    end

    $n^k = n^k + 1$;

    $t^k_{n^k} = s$;

    $\mathcal{T}^k = \mathcal{T}^k \bigcup \{t^k_{n^k}\}$;

  end

end

if $t^k_{n^k} \leq T$ **then**

  return $\mathcal{T}^m$ for $m = 1, 2, \ldots, M$;

else

  return $\mathcal{T}^1, \ldots, \mathcal{T}^k \setminus \{t^k_{n^k}\}, \ldots, \mathcal{T}^M$;

end

Algorithm 4 gives the pseudo-code to simulate points as per the Multivariate Hawkes process. The following are the input parameters to the algorithm.

1. $\boldsymbol{\mu}$: A vector with elements μ_m for $m = 1, 2, \ldots, M$
2. $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$: $M \times M$ matrices that define the parameters α_{mn} and β_{mn} for $m, n = 1, 2, \ldots, M$, for every pair of dimensions
3. T: The maximum time till which the events are to be generated

Each set $\mathcal{T}^m$ for $m = 1, 2, \ldots, M$ represents the ordered set of points in the dimension m, and n^m for $m = 1, 2, \ldots, M$, counts the number of events in $\mathcal{T}^m$ (the dimension m).

For completeness, let us also add some basic information about Dirichlet distribution which we use widely in the processes we define.

Definition 18: *The Dirichlet distribution of order $K \geq 2$ with parameters $\alpha_1, \alpha_2, \ldots, \alpha_k > 0$ has a probability density function given by:*

$$f(\boldsymbol{x}|\boldsymbol{\alpha}) = \frac{1}{B(\boldsymbol{\alpha})} \prod_{i=1}^{K} x_i^{\alpha_i - 1} \tag{2.5}$$

where, $B(\boldsymbol{\alpha})$ is a normalizing constant given by:

$$B(\boldsymbol{\alpha}) = \frac{\prod_{i=1}^{K} \Gamma(\alpha_i)}{\Gamma\left(\sum_{i=1}^{K} \alpha_i\right)} \tag{2.6}$$

Equation 2.5 is a probability density function, therefore,

$$\begin{aligned} \int_{\boldsymbol{x}} \frac{1}{B(\boldsymbol{\alpha})} \prod_{i=1}^{K} x_i^{\alpha_i - 1} \, d\boldsymbol{x} &= 1 \\ \frac{1}{B(\boldsymbol{\alpha})} \int_{\boldsymbol{x}} \prod_{i=1}^{K} x_i^{\alpha_i - 1} \, d\boldsymbol{x} &= 1 \\ \int_{\boldsymbol{x}} \prod_{i=1}^{K} x_i^{\alpha_i - 1} \, d\boldsymbol{x} &= B(\boldsymbol{\alpha}) \end{aligned} \tag{2.7}$$

2.5 Bayesian Inference

Bayesian inference is a method that analyzes data to infer properties of the underlying probability distribution. It is mainly dependent on the Bayes' law that updates the distribution for a hypothesis as more and more evidence is seen. We may have a *prior* belief about some event, but, that might change (either becomes stronger or weaker) as more information about that event is brought to light. Bayesian inference is a mathematical tool that provides a formal way to update our prior beliefs to *posterior* beliefs as new evidence or data is exposed.

Let us first introduce Bayes' Law that form the basis for Bayesian inference.

Theorem 19 (Bayes' Law)**.** *Assume that $E_1, E_2, \ldots, E_n$ are mutually disjoint sets such that $\bigcup_{i=1}^{n} E_i = E$. Then,*

$$Pr(E_j|B) = \frac{Pr(E_j \cap B)}{P(B)} = \frac{Pr(B|E_j)Pr(E_j)}{\sum_{i=1}^{n} Pr(B|E_i)Pr(E_i)} \tag{2.8}$$

In machine learning literature, the Bayes' law is mostly represented in the following way:

$$Pr(\theta|\mathcal{D}) = \frac{Pr(\mathcal{D}|\theta)Pr(\theta)}{Pr(\mathcal{D})} \tag{2.9}$$

Where, θ is used for parameter, and $\mathcal{D}$ represents the data.

Consider a coin tossing example. Let's think about the question: "What is the probability of seeing 6 heads in 8 flips given that the coin is unbiased or fair (i.e. $Pr(Heads) = Pr(Tails) = 0.5$)?". In the Bayesian inference setting, we are interested in the alternate question. That is, we are interested in quantifying the *fairness* of the coin given that we have seen a particular number of heads and tails. Here is the formal way to put that up: "What is the probability of the coin being fair given that k number of heads are observed when the coin is flipped n number of times?" Figure 2-3 gives a view of the update of the posterior probability distribution as more and more data (coin flips) are observed. Before describing the Figure 2-3 in some more details, let us first describe the terminology that is used to express each of the probability terms in the Bayes' law (Eq. 2.9).

- $P(\theta)$ term is called as **prior**. This is the distribution for the parameter without seeing that data (i.e. apriori). And thus, is known as a *prior* distribution or *prior* belief about the parameter.
- $P(\theta|\mathcal{D})$ is termed as **posterior**. This term can be thought of as the updated belief about our parameter θ after observing the data $\mathcal{D}$. So let us say in the coin tossing example, you start with a uniform distribution over the parameter θ, which is the prior, and the posterior (updated belief) is the information about θ that we have after seeing 8 heads out of the 10 flips.
- $P(\mathcal{D}|\theta)$ is termed as **likelihood** of the data (given the parameter). This is the probability of the data $\mathcal{D}$ given that it was generated with parameter θ.
- $P(\mathcal{D})$ is called as the **evidence** term. This can also be written in the following way:

  $$Pr(\mathcal{D}) = \int_{\theta} Pr(\theta)Pr(\mathcal{D}|\theta)d\theta$$

  Thus, $Pr(\mathcal{D})$ is the weighted sum of $Pr(\mathcal{D}|\theta)$, weighted by the probability of the values that θ takes.

Figure 2-3 gives an idea of the update in the belief about the parameter θ as more and more data is exposed. In this case, the data observed is the number of heads after a particular number of flips. It starts with a prior belief that is the uniform distribution. That is, it says that the parameter $\theta = Pr(Heads)$ taking each value in the range $[0, 1]$ is equally likely. As the number of flips and correspondingly the number of heads increases, the belief about the parameter θ gets concentrated near the value $\theta = Pr(Heads) \approx 0.46$. Which gives an intuition about the coin being a fair coin.

2.5.1 Conjugate Priors

Notice that the posterior probability equation Eq. 2.9 itself requires computing an integral – the evidence term $P(\mathcal{D}) = \int Pr(\mathcal{D}|\theta)Pr(\theta)d\theta$. This is required to normalize

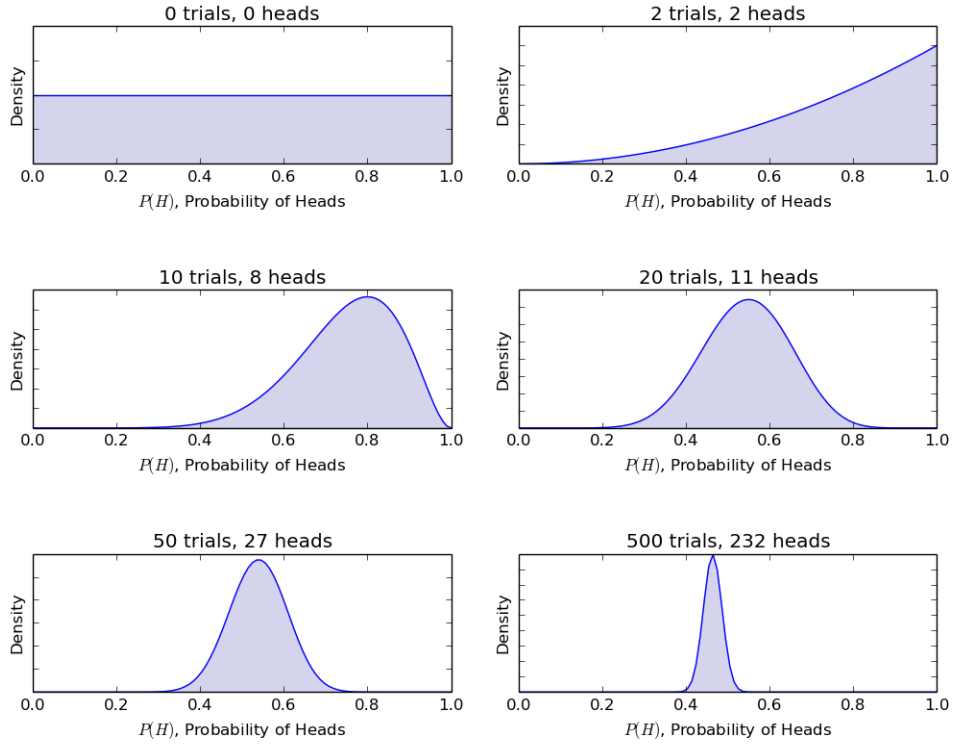


Figure 2-3: Example of Bayesian posterior update procedure [74]

the posterior so that it is a distribution. So the natural question to ask is about the tractability of this integral. And this exactly is the idea behind *conjugate priors*. That is, given a likelihood function $Pr(\mathcal{D}|\theta)$, choose a family of prior distributions such that the integrals as that of the evidence term $P(\mathcal{D})$ becomes tractable (for all the prior distributions in the family). Additionally, we can choose the family of prior distributions such that the posterior is also in the same family as that of the prior. Observe that, one can achieve the second goal (i.e. having the posterior in the same family as that of the prior) vacuously by defining the family of all the probability distributions. But this might not guarantee the tractability of the integral. However, on the other hand, one can achieve the goal of having tractable integrals by considering the family of distributions to be a distribution that is as simple as a constant, however, in that case, the posterior might not be in the same family as that of the prior [50].

Thus, conjugate priors have captivating computational properties and are used widely in practice. However, one should be careful while using conjugate priors. The use of conjugate priors implicitly has relatively solid assumptions and thus it is important to understand the influence of the prior on the posterior. If the posterior is not much influenced by the prior then one can move ahead with some confidence. However, if the posterior is highly influenced by the prior, then one must try to understand or consult a domain expert to understand the implications of the assumptions brought in by the use of the prior. The other option would be to use some other priors or collect

some more data and diminish the effect of the prior [50].

Let's try to understand more by considering an example. Going back to the coin tossing example, where we are interested in understanding how fair the coin is given k number of heads observed out of a n coin flips. The likelihood $Pr(\mathcal{D}|\theta)$ in the coin tossing example is nothing but the binomial distribution with parameters (k, n, θ). And it is of the form $Pr(\mathcal{D}|\theta) \propto \theta^x k(1-\theta)^{(n-k)}$. Observe that if the prior is of the similar form as that of the likelihood, i.e. if the prior $Pr(\theta)$ is proportional to the powers of θ and $(1-\theta)$, then the posterior $Pr(\theta|\mathcal{D})$ will also be distributed similarly as that of the prior, but, with different parameters. That is, if $Pr(\theta) \propto \theta^a(1-\theta)^{(b)}$, then the posterior $Pr(\theta|\mathcal{D}) \propto \theta^{a+k}(1-\theta)^{(b+n-k)}$. One of the distributions that could give a required form is the Beta distribution.

Beta Distribution: A random variable X with range $[0, 1]$ has a beta distribution with (hyper)parameters α ($\alpha > 0$) and β ($\beta > 0$) if the pdf is given as:

$$Pr(x|\alpha, \beta) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} x^{\alpha-1}(1-x)^{(\beta-1)} \tag{2.10}$$

Here, Γ is the *Gamma* function. The expectation of the Beta distributed random variable X turns out to be: $\mathbb{E}[X] = \frac{\alpha}{\alpha+\beta}$. Given this information about Beta distribution, if we now consider the prior $Pr(\theta)$ for the *fairness* parameter θ in the coin tossing example to be a Beta distribution with hyperparameters (α, β), then the posterior $Pr(\theta|\mathcal{D})$ will also turn out to be a Beta distribution with parameters $(\alpha + k, \beta + n - k)$. Here is the formal way to present that:

$$\begin{aligned} Pr(\theta|\mathcal{D}, \alpha, \beta) &\propto \theta^k(1-\theta)^{n-k}\theta^{\alpha-1}(1-\theta)^{\beta-1} \\ &= Beta(\alpha + k, \beta + n - k) \end{aligned} \tag{2.11}$$

We are ignoring some terms on the right hand side as those are independent of θ and thus can be considered as constant that do not effect the distribution. As the posterior turns out to be a Beta distribution on having a Beta prior for the Binomial likelihood, the Beta distribution is designated as emphconjugate prior for the binomial distribution. Another way to understand the benefit of conjugate priors is, observe equation Eq. 2.11, it indicates that the Bayesian update is now reduced to modifying the parameters (hyperparameters) of the prior distribution than computing the expensive integrals. In the coin tossing example, we mentioned using a uniform distribution as the prior belief for the parameter θ. However, the uniform distribution is a special case of Beta distribution, which is $Beta(1, 1)$.

2.6 Markov Chain Monte Carlo

Lets recall the equation for the posterior distribution (Eq. 2.9).

$$Pr(\theta|\mathcal{D}) = \frac{Pr(\mathcal{D}|\theta)Pr(\theta)}{Pr(\mathcal{D})} \tag{2.12}$$

The denominator here, which is termed as evidence, is also called as *total probability*. Total probability is the probability of observing the data under all the possible values of θ. Another way to represent the term $Pr(\mathcal{D})$ is:

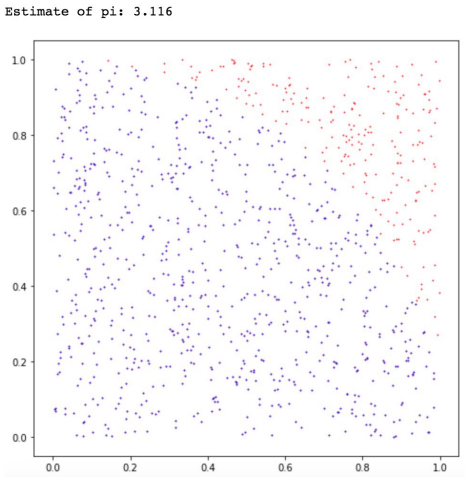
$$Pr(\mathcal{D}) = \int_{\theta} Pr(\theta)Pr(\mathcal{D}|\theta)d\theta \tag{2.13}$$

That is, it sums the probability of $\mathcal{D}$ over all the possible values of θ. Most of the times this integral does not have a closed form. And thus, we need an approximation. One of the methods that help in getting samples from unknown distribution is Markov Chain Monte Carlo (MCMC). MCMC generates samples such that the samples mimic the unknown distribution. MCMC methods help in estimating the distribution where it is impossible to obtain it analytically. One of the MCMC methods that are used in the subsequent chapters is Gibbs sampling. Here we would give an overview of the Gibbs sampling algorithm. But before going on to that, we will first give a very brief introduction to Monte Carlo simulations.

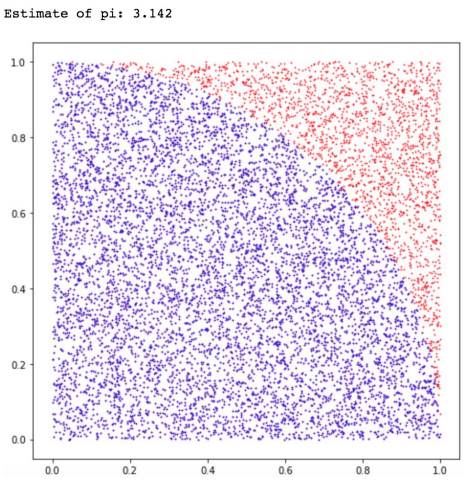
2.6.1 Monte Carlo (MC)

A collection of tools for estimating values through sampling is referred to as Monte Carlo (MC) methods. These methods are widely used in the areas of engineering and physical sciences [66]. These methods perform some computation on the random samples to provide an estimate of the required parameter where calculating it is too expensive or impossible directly. The basis of the MC methods is the Law of Large numbers which states (informally) that as the number of identically distributed randomly generated samples increase, the sample mean (average) of these samples approaches the theoretical mean. Let us look at the example of a very well known MC estimate which is the estimate of π (Pi).

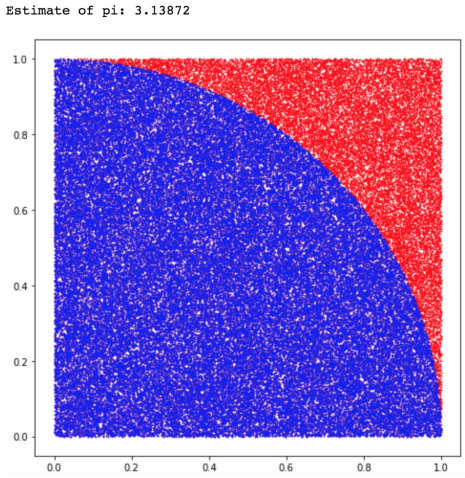
Generate random points within a box of length 1.0 unit and breadth 1.0 unit. Consider this as the portion of the first quadrant in the 2-D plane. Consider a circle of radius 1.0 unit at the center of the 2-D plane. Note that the first quarter of the circle is embedded in the box. Now, count the fraction of points that fall within an embedded circle. Also, notice that the probability of points falling inside the embedded portion of the circle is $\pi/4$. The fraction of points inside the portion of circle must be equal to this probability. This would give an estimate of π. Figure 2-4 gives a pictorial view of the idea that as the number of randomly sampled points increases, the value of π



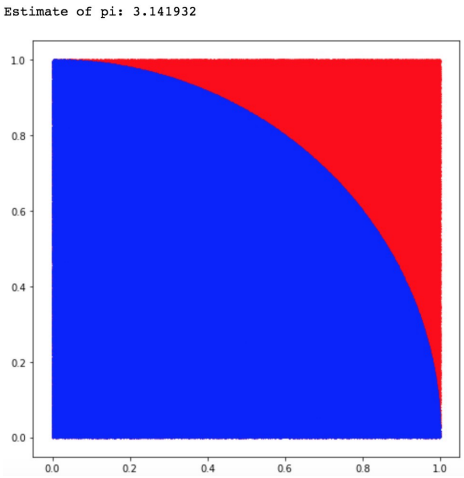
(a) $1K$ random points



(b) $10K$ random points



(c) $100K$ random points



(d) $1M$ random points

Figure 2-4: Plots for $1K$, $10K$, $100K$, and $1M$ random points [81]

would approach its original value.

Therefore, this example gives a good overview of the MC methods. However, in general, it is difficult to generate random samples that are from a required probability distribution. Even in the example of estimating the value of π the points were considered to be distributed uniformly. But most of the time even getting a uniform sample from the set of all possible values might not be easy. In such cases, the Markov Chain Monte Carlo (MCMC) method plays a crucial role.

MCMC method provides a way to sample points from a required probability distribution [66]. Often in the case of Bayesian inference where we need to sample from the posterior distribution, the MCMC method is suitable for this task. Here is the basic idea about an MCMC method. Define an ergodic Markov Chain for which the set of states is the sample space and the required probability distribution is the stationary distribution. Consider $X_0, X_1, \ldots, X_n$ as the states of the run of the Markov Chain.

After a sufficiently large number of steps, say r, the distribution of the state X_r will be close to the stationary distribution given that the Markov chain converges to a stationary distribution starting from any state X_0 after a large number of steps. Repeating this argument, one can take $X_r, X_{2r}, X_{3r}, \ldots$ as almost independent samples from the stationary distribution of the Markov chain [66]. Therefore, using this method we can sample values from the required posterior probability distribution. As mentioned above, here we give an overview of one of the MCMC methods, namely the Gibbs sampling, that helps us in obtaining samples from the posterior distribution.

2.6.2 Gibbs Sampling

The Gibbs sampling algorithm is a method in which a Markov chain is constructed by considering the probability of the current sample conditional on the prior sample. The basic idea behind Gibbs sampling is to sample each variable in turn conditioned on the values of all other variables. Here is a brief overview of the Gibbs sampling algorithm. Consider $X_1, X_2, \ldots, X_k$ as the random variables in the stochastic process. Start by initializing these variables to $x_1^{(0)}, x_2^{(0)}, \ldots, x_k^{(0)}$ (these are mostly samples from the prior distribution). In each iteration i, for each $j = 1, \ldots, k$ sample $x_j^{(i)}$ as:

$$x_j^{(i)} \sim Pr\left(X_j = x_j | X_1 = x_1^{(i)}, \ldots, X_{j-1} = x_{j-1}^{(i)}, X_{j+1} = x_{j+1}^{(i-1)}, \ldots, X_k = x_k^{(i-1)}\right) \quad (2.14)$$

Continue this process until *convergence*. Algorithm 5 gives a pseudo-code that explains how Gibbs sampling works.

Algorithm 5: Gibbs Sampling Algorithm

Let $i = 0$;
Initialize $x^{(i)} = \left(x_1^{(1)}, x_2^{(1)}, \ldots, x_k^{(i)}\right)$;
for $i = 0, 1, \ldots$ **do**
    $x_1^{(i)} \sim Pr\left(X_1 = x_1 | X_2 = x_2^{(i-1)}, \ldots, X_k = x_k^{(i-1)}\right)$;
    $x_2^{(i)} \sim Pr\left(X_2 = x_2 | X_1 = x_2^{(i)}, X_3 = x_3^{(i-1)}, \ldots, X_k = x_k^{(i-1)}\right)$;
    $\vdots$
    $x_k^{(i)} \sim Pr\left(X_k = x_k | X_1 = x_1^{(i)}, \ldots, X_{k-1} = x_{k-1}^{(i)}\right)$;
end

In Gibbs sampling, there is an assumption that we can compute the conditional distribution of variable conditioned on all other random variables and can sample from this distribution exactly. All the random variables might not always be dependent on all other variables. For example in graphical models, a random variable at a node is dependent only on the variables that are in the Markov blanket of that node. One of

the variants of the Gibbs sampling method is collapsed Gibbs sampling in which we marginalize as many variables as possible. The benefit of this is that it reduces the number of random variables to be sampled and thus the space of random variables.

In the subsequent chapters to obtain samples from the posterior distributions we compute the conditional distribution of the random variables and use the Gibbs sampling method.

Chapter 3

Discovering Topical Interactions in Text-Based Cascades[1]

3.1 Introduction

A popular area of recent research has been the study of information diffusion cascades, where information spreads over a social network when a 'parent' event from one infected node influences a 'child' event at neighboring node [28, 48, 93, 35, 44]. The action of propagating information between two neighboring nodes depends on various factors, such as the strength of influence between the nodes, the topical content of the parent event and the extent of the interest of the child node towards that topic. However, such explanatory variables and also the identity of influencing or parent event for any event, are typically unobserved and need to be inferred.

A recent body of work in this area has proposed increasingly sophisticated models for information cascades. To account for the temporal burstiness of events, Hawkes processes, which are self-exciting point processes, have been proposed to model their time stamps [76]. Also, influences travel more quickly over stronger social ties. This is modeled by the Network Hawkes process [59] by incorporating connection strengths between users into the intensity function of the Hawkes process. When events additionally have associated textual data, parent and child events in a cascade are typically similar in their topical content. The Hawkes Topic Model (HTM) [48] captures this by combining the Network Hawkes process with an LDA-like generative process for the textual content where the topic mixture for a child event is close to that for a parent event.

Despite its strengths, the HTM fails to capture one important aspect of the richness of information cascades. Typically, there are sequential patterns in the textual content of different events within a cascade. In terms of topical representation of the events, topics of parent and child events display repeating patterns. At the same time, these

[1]All the results in this chapter are based on the publication [17] at ICDM-2018

cannot be assigned to the same topic. Instead, our model assigns these to different topics and recognizes this to be a frequently appearing topic pair in other parent-child tweets. This serves as additional evidence for inferring a parent-child relationship between this tweet pair.

However, just like topic distributions, topic interaction patterns are also typically latent and need to be inferred. Interestingly, inferring topic interaction patterns, in turn, benefits from more accurate parent and topic assignment to events. This calls for joint inference of topic interaction patterns and the other latent variables.

We propose a generative model for textual cascades that captures topical interactions in addition to temporal burstiness, network strengths, and user topic preferences. Temporal and network burstiness is captured using a Hawkes process over the social network [59]. Topical interactions are modeled by combing this process with a Markov chain over event topics along a path of the diffusion cascade. Such topical Markov chains have appeared in the literature [41, 43, 4, 5, 27], but in the very different context of modeling word sequences in a document. Our model effectively integrates topical Markov chains with the Network Hawkes process and we call it the Hidden Markov Hawkes Process (HMHP).

We derive a simple and efficient collapsed Gibbs Sampling algorithm for inference using our model. This algorithm is significantly more scalable than the variational algorithm for the HTM and allows us to analyze large collections of real tweets. We validate using experiments that HMHP fits information cascades better compared to models that do not consider topical interactions. With the aid of semi-synthetic data, we show that modeling topical interactions indeed leads to more accurate identification of event parents and topics. More importantly, HMHP is able to identify interesting topical interactions in real tweets, which is beyond the capability of any existing information diffusion model.

3.2 Generative Model

We consider a set of nodes $V = \{1, \ldots, n\}$, representing content producers, and a set of directed edges E among them, representing the underlying graph using which information propagates. For every edge (u, v), $\mathcal{W}_{uv}$ denotes the weight of the edge, capturing the extent of influence between content producers or users u and v. For each event e, representing e.g. a social media post, t_e denotes its posting time, c_e the user who creates the post and d_e the document associated with the post. Event e could be *spontaneous*, or a *diffusion* event, meaning that it is triggered by some recent event by one of the followees of user c_e. The variable z_e denotes the unique parent event that triggered the creation of event e if it is a diffusion event. For spontaneous events, z_e is set to 0. The generated event e is then consumed by each of the followers of the user c_e, thereby triggering them, in turn, to create multiple events and producing an information cascade. Our generative model can be broken up into two distinct stages - generating the cascade events, and then the event documents.

3.2.1 Generating Cascade Events

We first generate (t_e, c_e, z_e) for all events using a multivariate Hawkes process (MHP) following existing models [84, 48, 59], where the users can mutually excite each other to produce events. For each node v, we define $\lambda_v(t)$, a rate at time t, as a superposition of the base intensity μ_v for user v, and the impulse responses for each event e_n that has happened at time t_n at a followee node c_n.

$$\lambda_v(t) = \mu_v(t) + \sum_{n=1}^{|\mathcal{H}_{t^-}|} h_{c_n,v}(t - t_n) \tag{3.1}$$

where $h_{c_n,v}(t - t_n)$ is the impulse response of user c_n on the user v, and $\mathcal{H}_{t^-}$ is the history of events upto time t. The impulse response can be decomposed as the product of the influence W_{uv} of user u on v, and a time-kernel as follows:

$$h_{u,v}(\Delta t) = \mathcal{W}_{u,v} f(\Delta t) \tag{3.2}$$

We use a simple exponential kernel $f(\Delta t) = exp(-\Delta t)$, as this is not the main thrust of our work. Following [84], we generate the events using a level-wise generation process. Level 0, denoted as Π_0, contains all the spontaneous events, generated using the base rates of the users. The events Π_l at level l, are generated according to the following non-homogenous Poisson process

$$\Pi_l \sim Poisson\left(\sum_{(t_n,c_n,z_n)\in\Pi_{l-1}} h_{c_n,\cdot}(t - t_n) \right) \tag{3.3}$$

3.2.2 Generating Event Documents

The document d_e for e is drawn using a topic model over a vocabulary of size $|\mathcal{V}|$. For spontaneous events, the topic choice η_e depends on the topic preference of the user c_e. We deviate from existing literature in modeling the topic of a diffusion event. Instead of being identical or 'close' to the topic of the triggering event [48], the diffusion event may be on a 'related' topic. We model this transition between related topics using a Markov Chain over topics, involving a topic transition matrix $\mathcal{T}$. This enables us to capture repeating patterns in topical transitions between parent and child events. Since the topical sequence is 'hidden' and observed only indirectly through the content of the events, we call our model the Hidden Markov Hawkes Process (HMHP). We have K topics, denoted $\{\boldsymbol{\zeta}_k\}$, assumed to be probability distributions over words (vocabulary with size $|\mathcal{V}|$), generated *iid* from a Dirichlet distribution. We also assume the existence of a topic-topic interaction matrix $\boldsymbol{\mathcal{T}}$, where $\boldsymbol{\mathcal{T}}_k$, again sampled from a Dirichlet distribution, denotes the distribution over 'child topics' for a 'parent topic' k. Since our data of interest is tweets (short documents), we model a document d_e as having a single topic η_e. To generate d_e, we first follow a Markov process for sampling

the topic η_e conditioned on the topic of its parent event η_{z_e}, followed by sampling the words *iid* according to the chosen topic. For spontaneous events, the topic for its document is sampled randomly from the preferred distribution over topics $\boldsymbol{\phi}_u$ for the corresponding user u.

An important consideration in the design of our model is the use of conjugate priors. As we will see in the Inference section such priors play a crucial role in the design of efficient and simple sampling-based inference algorithms. Models such as HTM [48], which have to sacrifice conjugacy to model data complexity, have to resort to more complex variational inference strategies.

We summarize the entire generative process below.

1. Generate (t_e, c_e, z_e) for all events according to the process described in previous subsection.
2. For each topic k: sample $\boldsymbol{\zeta}_k \sim Dir_{|\mathcal{V}|}(\boldsymbol{\alpha})$
3. For each topic k: sample $\boldsymbol{\mathcal{T}}_k \sim Dir_K(\boldsymbol{\beta})$
4. For each node v: sample $\boldsymbol{\phi}_v \sim Dir_K(\boldsymbol{\gamma})$
5. For each event e at node $c_e = v$:
   (a) i. **if** $z_e = 0$ (level 0 event):
   draw a topic $\eta_e \sim Discrete_K(\boldsymbol{\phi}_v)$
   ii. **else**:
   draw a topic $\eta_e \sim Discrete_K(\boldsymbol{\mathcal{T}}_{\eta_{z_e}})$
   (b) Sample document length $N_e \sim Poisson(\lambda)$
   (c) For $w = 1 \ldots N_e$: draw word $x_{e,w} \sim Discrete_{|\mathcal{V}|}(\boldsymbol{\zeta}_{\eta_e})$

The resultant joint likelihood can we written as follows:

$$
\begin{aligned}
P(E, \boldsymbol{\Phi}, \boldsymbol{\mathcal{T}}, \boldsymbol{\zeta}, \boldsymbol{\eta}, \boldsymbol{z} \mid \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\mathcal{W}}, \boldsymbol{\mu}) &= \prod_{v \in V} P(\boldsymbol{\phi}_v \mid \boldsymbol{\gamma}) \times \prod_{k=1}^{K} P(\boldsymbol{\zeta}_k \mid \boldsymbol{\alpha}) \times \prod_{k=1}^{K} P(\boldsymbol{\mathcal{T}}_k \mid \boldsymbol{\beta}) \\
&\times \prod_{v \in V} \left[exp\left(- \int_0^T \mu_v(\tau) d\tau \right) \prod_{e \in E} \mu_v(t_e)^{\delta_{c_e,v} \delta_{z_e,0}} \right] \\
&\times \prod_{e \in E} \prod_{v \in V} \left[exp\left(- \int_{t_e}^T h_{c_e,v}(\tau - t_e) d\tau \right) \prod_{e' \in E} h_{c_e,c_{e'}}(t_{e'} - t_e)^{\delta_{c_{e'},v} \delta_{z_{e'},e}} \right] \\
&\times \prod_{e \in E} \left\{ \left[\prod_{e' : t_{e'} < t_e} P(\eta_e | \boldsymbol{\mathcal{T}}_{\eta_{z_e}})^{\delta_{z_e,e'}} \right] P(\eta_e | \boldsymbol{\phi_v})^{\delta_{z_e,0}} \right\} \times \prod_{e \in E} \left[\prod_{w=1}^{N_e} P(x_{e,w} | \eta_e, \boldsymbol{\zeta}_{\eta_e}) \right] \qquad (3.4)
\end{aligned}
$$

Here, $\delta_{c_e,v}$ is an indicator for the event e being triggered at node v, $\delta_{z_{e'},e}$ is an indicator for event e being parent of event e', $\Delta\tau$ is $(\tau - t_e)$, $\Delta(t_{e'})$ is $(t_{e'} - t_e)$ and T is the time horizon.

3.3 Approximate Inference

The latent event variables are the topic η_e and the diffusion parent z_e.The process parameters to be estimated include the user-user influence values W_{uv}, the topic transition matrix $\mathcal{T}$, and the user-topic preferences.

By making use of the conjugate priors, we perform inference using collapsed Gibbs sampling. We integrate out the topic distributions over words $\boldsymbol{\zeta}$, the topic interaction distributions $\boldsymbol{\mathcal{T}}$ and the user-topic preference distributions $\boldsymbol{\Phi}$ as follows.

$$
\begin{aligned}
P(E, \boldsymbol{\eta}, \boldsymbol{z} \mid \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\mathcal{W}}, \boldsymbol{\mu}) &= \int_{\boldsymbol{\Phi}} \int_{\boldsymbol{\mathcal{T}}} \int_{\boldsymbol{\zeta}} P(E, \boldsymbol{\Phi}, \boldsymbol{\mathcal{T}}, \boldsymbol{\zeta}, \boldsymbol{\eta}, \boldsymbol{z} \mid \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\mathcal{W}}, \boldsymbol{\mu})\, d\boldsymbol{\mathcal{T}}\, d\boldsymbol{\Phi}\, d\boldsymbol{\zeta} \\
&= \int_{\boldsymbol{\zeta}} \prod_{k=1}^{K} P(\boldsymbol{\zeta}_k \mid \boldsymbol{\alpha}) \prod_{e \in E} \left[\prod_{i=1}^{N_e} P(x_{e,w_i} | \eta_e, \boldsymbol{\zeta}_{\eta_e}) \right] d\boldsymbol{\zeta} \\
&\times \int_{\boldsymbol{\Phi}} \prod_{v \in V} P(\boldsymbol{\phi}_v \mid \boldsymbol{\gamma}) \prod_{e \in E} P(\eta_e | \boldsymbol{\phi}_v)^{\delta_{z_e,0}}\, d\boldsymbol{\Phi} \\
&\times \int_{\boldsymbol{\mathcal{T}}} \prod_{k=1}^{K} P(\boldsymbol{\mathcal{T}}_k \mid \boldsymbol{\beta}) \prod_{e \in E} \left[\prod_{e' : t_{e'} < t_e} P(\eta_e | \boldsymbol{\mathcal{T}}_{\boldsymbol{\eta}_{\boldsymbol{z_e}}})^{\delta_{z_e,e'}} \right] d\boldsymbol{\mathcal{T}} \\
&\times \prod_{v \in V} \left[exp\left(- \int_0^T \mu_v(\tau) d\tau \right) \prod_{e \in E} \mu_v(t_e)^{\delta_{c_e,v} \delta_{z_e,0}} \right] \\
&\times \prod_{e \in E} \prod_{v \in V} \left[exp\left(- \int_{t_e}^T h_{c_e,v}(\tau - t_e) d\tau \right) \prod_{e' \in E} h_{c_e,c_{e'}}(t_{e'} - t_e)^{\delta_{c_{e'},v} \delta_{z_{e'},e}} \right] \qquad (3.5)
\end{aligned}
$$

We can treat each of $\boldsymbol{\zeta}, \boldsymbol{\mathcal{T}}$, and $\boldsymbol{\Phi}$ separately. So, we first focus only on $\boldsymbol{\zeta}$ part. Observe that all the $\boldsymbol{\zeta}s$ are independent of each other. And thus, we can leverage this information to integrate each of them independently. Let $\Psi_{k,r}$ denote the number of times a word r is assigned a topic k (or belongs to a document corresponding to event that is assigned topic k). Then,

$$
\begin{aligned}
&\int_{\boldsymbol{\zeta}} \prod_{k=1}^{K} P(\boldsymbol{\zeta}_k \mid \boldsymbol{\alpha}) \prod_{e \in E} \left[\prod_{w=1}^{N_e} P(x_{e,w} | \eta_e, \boldsymbol{\zeta}_{\eta_e}) \right] d\boldsymbol{\zeta} \\
&= \prod_{k=1}^{K} \int_{\boldsymbol{\zeta}_k} P(\boldsymbol{\zeta}_k \mid \boldsymbol{\alpha}) \prod_{e \in E} \left[\prod_{w=1}^{N_e} P(x_{e,w} | \eta_e, \boldsymbol{\zeta}_{\eta_e}) \right] d\boldsymbol{\zeta}_k \\
&= \prod_{k=1}^{K} \int_{\boldsymbol{\zeta}_k} \frac{1}{B(\boldsymbol{\alpha})} \prod_{r=1}^{|\mathcal{V}|} \zeta_{k,r}^{\alpha_r - 1} \prod_{r=1}^{|\mathcal{V}|} \zeta_{k,r}^{\Psi_{k,r}}\, d\boldsymbol{\zeta}_k = \prod_{k=1}^{K} \frac{1}{B(\boldsymbol{\alpha})} \int_{\boldsymbol{\zeta}_k} \prod_{r=1}^{|\mathcal{V}|} \zeta_{k,r}^{\alpha_r + \Psi_{k,r} - 1}\, d\boldsymbol{\zeta}_k
\end{aligned}
$$

$$= \prod_{k=1}^{K} \frac{B(\boldsymbol{\alpha} + \boldsymbol{\Psi}_k)}{B(\boldsymbol{\alpha})} \qquad \text{from Eq. 2.5, 2.6, 2.7} \tag{3.6}$$

Now we turn our attention to the $\boldsymbol{\Phi}$ part. Like $\boldsymbol{\zeta}$, all the $\boldsymbol{\phi}_v$s are independent. Let $\Omega_{v,k}$ be the number of events at v (without parent) that are assigned topic k. Then,

$$\begin{aligned}
&\int_{\boldsymbol{\Phi}} \prod_{v \in V} P(\boldsymbol{\phi}_v \mid \boldsymbol{\gamma}) \prod_{e \in E} P(\eta_e | \boldsymbol{\phi}_v)^{\delta_{z_e,0}} \, d\boldsymbol{\Phi} = \prod_{v \in V} \int_{\boldsymbol{\phi}_v} P(\boldsymbol{\phi}_v \mid \boldsymbol{\gamma}) \prod_{\substack{e \in E \\ c_e = v \\ z_e = 0}} P(\eta_e | \boldsymbol{\phi}_v) \, d\boldsymbol{\phi}_v \\
&= \prod_{v \in V} \int_{\boldsymbol{\phi}_v} \frac{1}{B(\boldsymbol{\gamma})} \prod_{k=1}^{K} \phi_{v,k}^{\gamma_v - 1} \prod_{\substack{e \in E \\ c_e = v \\ z_e = 0}} P(\eta_e | \boldsymbol{\phi}_v) \, d\boldsymbol{\phi}_v \\
&= \prod_{v \in V} \frac{1}{B(\boldsymbol{\gamma})} \int_{\boldsymbol{\phi}_v} \prod_{k=1}^{K} \phi_{v,k}^{\gamma_v - 1} \prod_{k=1}^{K} \phi_{v,k}^{\Omega_{v,k}} \, d\boldsymbol{\phi}_v = \prod_{v \in V} \frac{1}{B(\boldsymbol{\gamma})} \int_{\boldsymbol{\phi}_v} \prod_{k=1}^{K} \phi_{v,k}^{\gamma_v + \Omega_{v,k} - 1} \, d\boldsymbol{\phi}_v \\
&= \prod_{v \in V} \frac{B(\boldsymbol{\gamma} + \boldsymbol{\Omega}_v)}{B(\boldsymbol{\gamma})} \qquad \textit{from Eq. 2.5, 2.6, 2.7}
\end{aligned} \tag{3.7}$$

On similar lines, we can work with $\boldsymbol{\mathcal{T}}$ part. All the $\boldsymbol{\mathcal{T}_k}$s are independent. Let $\xi_{k,t}$ be the number of events that are assigned topic t and have parent event with topic k. Then,

$$\begin{aligned}
&\int_{\boldsymbol{\mathcal{T}}} \prod_{k=1}^{K} P(\boldsymbol{\mathcal{T}}_k \mid \boldsymbol{\beta}) \prod_{e \in E} \left[\prod_{e' : t_{e'} < t_e} P(\eta_e | \boldsymbol{\mathcal{T}}_{\eta_{z_e}})^{\delta_{z_e,e'}} \right] d\boldsymbol{\mathcal{T}} \\
&= \prod_{k=1}^{K} \int_{\boldsymbol{\mathcal{T}}_k} P(\boldsymbol{\mathcal{T}}_k \mid \boldsymbol{\beta}) \prod_{\substack{e \in E \\ \eta_{z_e} = k}} P(\eta_e | \boldsymbol{\mathcal{T}}_k) \, d\boldsymbol{\mathcal{T}}_k \\
&= \prod_{k=1}^{K} \int_{\boldsymbol{\mathcal{T}}_k} \frac{1}{B(\boldsymbol{\beta})} \prod_{t=1}^{K} \mathcal{T}_{k,t}^{\beta_t - 1} \prod_{\substack{e \in E \\ \eta_{z_e} = k}} P(\eta_e | \boldsymbol{\mathcal{T}}_k) \, d\boldsymbol{\mathcal{T}}_k \\
&= \prod_{k=1}^{K} \frac{1}{B(\boldsymbol{\beta})} \int_{\boldsymbol{\mathcal{T}}_k} \prod_{t=1}^{K} \mathcal{T}_{k,t}^{\beta_t - 1} \prod_{t=1}^{K} \mathcal{T}_{k,t}^{\xi_{k,t}} \, d\boldsymbol{\mathcal{T}}_k = \prod_{k=1}^{K} \frac{1}{B(\boldsymbol{\beta})} \int_{\boldsymbol{\mathcal{T}}_k} \prod_{t=1}^{K} \mathcal{T}_{k,t}^{\beta_t + \xi_{k,t} - 1} \, d\boldsymbol{\mathcal{T}}_k \\
&= \prod_{k=1}^{K} \frac{B(\boldsymbol{\beta} + \boldsymbol{\xi}_k)}{B(\boldsymbol{\beta})} \qquad \textit{from Eq. 2.5, 2.6, 2.7}
\end{aligned} \tag{3.8}$$

So now, Eq. 3.4 can be written as:

$$P(E, \boldsymbol{\eta}, \boldsymbol{z} \mid \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \mathcal{W}, \boldsymbol{\mu}) = \prod_{k=1}^{K} \frac{B(\boldsymbol{\alpha} + \boldsymbol{\Psi}_k)}{B(\boldsymbol{\alpha})} \times \prod_{v \in V} \frac{B(\boldsymbol{\gamma} + \boldsymbol{\Omega}_v)}{B(\boldsymbol{\gamma})} \times \prod_{k'=1}^{K} \frac{B(\boldsymbol{\beta} + \boldsymbol{\xi}_{k'})}{B(\boldsymbol{\beta})}$$
$$\times \prod_{v \in V} \left[exp\left(- \int_0^T \mu_v(\tau) d\tau \right) \prod_{e \in E} \mu_v(t_e)^{\delta_{c_e,v} \delta_{z_e,0}} \right]$$
$$\times \prod_{e \in E} \prod_{v \in V} \left[exp\left(- \int_{t_e}^T h_{c_e,v}(\tau - t_e) d\tau \right) \prod_{e' \in E} h_{c_e,c_{e'}}(t_{e'} - t_e)^{\delta_{c_{e'},v} \delta_{z_{e'},e}} \right] \quad (3.9)$$

The continuous valued latent variables that remain are the connection strengths $\mathcal{W}_{u,v}$. The resulting collapsed Gibbs sampling algorithm iteratively samples individual topic and parent assignments and the connection strengths from their conditional distributions given current assignments to all other variables until convergence. We next describe the conditional distributions for sampling the different variables in the algorithm. Algorithm 6 gives the pseudo-code to sample each of the parameters as per the Gibbs sampling procedure.

Algorithm 6: Gibbs Sampling Algorithm for HMHP Parameters

```
Initialize η_e, z_e for all events;
Initialize W_{u,v} for all u, v in the followers map;
for iter = 0; iter ! = maxIter; iter + + do
    for e ∈ allEvents do
        η_e ~ P(η_e = k | η_{¬e}, z, {X}, W, α, β, γ, μ) ;
    end
    for e ∈ allEvents do
        z_e ~ P(z_e = e'|E_t, z_{¬e}, η, α, β, γ, W, μ) ;
    end
    for (u, v) ∈ Edges do
        W_{u,v} ~ Gamma(N_uv + α', N_u + β') ;
    end
end
```

3.3.1 Topic assignment

The conditional probability for topic η_e of a diffusion event e ($z_e \neq 0$) being k is dependent on the (1) *text-content of the event*, (2) *topical interaction between the topic of the parent event and the current event*, and (3) *topical interaction between the current assigned topic of the event and all the child events of the current event*. So, the conditional probability of assigning topic k to this event e which has a parent event e' with topic

k' is given as:

$$P(\eta_e = k \mid \{x_{e\cdot}\}, \eta_{z_e} = k', \boldsymbol{\eta}_{\setminus z_e}, \{z_e\}) \propto \frac{\beta_k + N_{k',k}^{(\neg(z_e,e))}}{(\sum_l \beta_l) + \boldsymbol{N}_{k'}^{(\neg(z_e,e))}}$$
$$\times \frac{\prod_{l'=1}^{K} \prod_{i=0}^{N_{k,l'}^{(C_e)}-1} (\beta_{l'} + N_{k,l'}^{(\neg C_e)} + i)}{\prod_{i=0}^{\boldsymbol{N}_k^{(C_e)}-1} ((\sum_{l'} \beta_{l'}) + \boldsymbol{N}_k^{\neg C_e} + i)} \times \frac{\prod_{w \in d_e} \prod_{i=0}^{N_e^w - 1} (\alpha_w + \mathfrak{T}_{k,w}^{\neg e} + i)}{\prod_{i=0}^{N_e - 1} ((\sum_{w \in \mathscr{V}} \alpha_w) + \mathfrak{T}_k^{\neg e} + i)} \tag{3.10}$$

Here $N_{k',k}^{(C)}$ denotes the number of parent-child event pairs with topics k and k', $(\neg(z_e, e))$ denotes all edges excluding (z_e, e), C_e is the set of edges from event e to its child events ($\neg C_e$ being its complement), and $\mathfrak{T}_{k,w}^{\neg e}$ is the number of occurrences of word w under topic k in events other than e. Further, $\boldsymbol{N}_k^C = \sum_{k'} N_{k,k'}^C$, and N_e^w is count of w in d_e. The first term is the conditional probability of transitioning from parent topic k' to this event's topic k, the second term is that of transitioning from this topic k to each child event's topic l', and the third term is that of observing the words in the event document given topic k. It is important to observe how this conditional distribution pools together three different sources of evidence for an event's topic. Even when the document words do not provide sufficient evidence for the topic, the parent's topic and children's topic taken together can significantly reduce the uncertainty. The complete proof for the final conditional probability is presented in the Appendix in Section A.1.1.

For spontaneous events ($z_e = 0$), the conditional probability looks similar to Eqn. A.12. Only the first term changes to $\frac{\gamma_k + \mathfrak{U}_{v,k}^{(\neg e)}}{(\sum_k \gamma_k) + \mathfrak{U}_v^{(\neg e)}}$, where $\mathfrak{U}_{v,k}^{(\neg e)}$ is the number of events by user v with topic k, discounting event e. This captures the probability of node v picking topic k from its preferred topics.

3.3.2 Parent assignment

The conditional probability of event e' being the parent z_e of event e looks as follows:

$$P(z_e = e' | E_t, \boldsymbol{z}_{\neg e}, \boldsymbol{\mathcal{W}}) \propto \frac{(\beta_k + N_{k',k} - 1)}{((\sum_{k=1}^{K} \beta_k) + \boldsymbol{N}_{k'} - 1)} \times h_{u_{e'},u_e}(t_e - t_{e'}) \tag{3.11}$$

Here the first term is the transition probability from topic k' of the proposed parent event e' to this event's topic k. The second term is the probability of t_e being the time of this event e given the occurrence time $t_{e'}$ of the proposed parent event e'. As with topic identification, we see that evidence for the parent now comes from two different sources. When there is uncertainty about the parent based on the event time, common patterns of topic transitions between existing parent-child events helps in identifying the right parent.

The conditional probability of event e being a spontaneous event with no parent is given as:

$$P(z_e = 0|E_t, \boldsymbol{z}_{\neg e}, \boldsymbol{\mu}) \propto \frac{(\gamma_k + \mathfrak{U}_{u_e,k} - 1)}{((\sum_{k=1}^{K} \gamma_k) + \mathfrak{U}_{u_e} - 1)} \times \mu_{u_e}(t_e) \tag{3.12}$$

The complete proof for the final conditional probability is presented in the Appendix in Section A.1.2.

We limit the maximum number of parent candidates for an event to a parameter M (set to 100 in our experiments) and by limiting the maximum delay to D (set to 1 day).

3.3.3 Updating network strengths

For the network strength $\mathcal{W}_{u,v}$, using a Gamma prior $Gamma(\alpha, \beta)$, the posterior distribution can be approximated as follows:

$$P(\mathcal{W}_{u,v} = x \mid E_t^{(u,v)}, \boldsymbol{z}) \propto x^{\alpha_1} \exp(-x\beta_1) \tag{3.13}$$

where $\alpha_1 = (N_{u,v} + \alpha - 1)$ and $\beta_1 = (N_u + \frac{1}{\beta})^{-1}$, and $N_{u,v}$ is the number of parent-child events pairs between nodes u and v, and N_u is the number of events at node u. This is again a Gamma distribution $Gamma(\alpha_1, \beta_1)$. Instead of sampling W_{uv}, we set it to be the mean of the corresponding Gamma distribution. Note that in each iteration, we update W_{uv} only for edges that have at least one parent-child influence. The complete proof for final conditional probability is presented in the Appendix in Section A.1.3. Most edges have very few (typically just one) influence propagation events. This makes statistical estimation of their strengths infeasible. To get around this problem, we share parameters across edges. We group edges that have the same value for the tuple (out-degree(source), in-degree(destination)). The intuition is that the influence of the edge (u, v) is determined uniquely by the popularity (outdegree) of u and the number of different influencers (indegree) of v. We then pool data from all edges in a group and estimate a single connection strength for a group.

After convergence, the parameters $\zeta_{k,r}$ and $\mathcal{T}_{k,k'}$ that were integrated out are estimated using the samples.

3.4 Experiments and Results

In this section, we empirically validate the strengths of HMHP against competitive baselines over a large collection of real tweets as well as semi-synthetic data. We first discuss the baseline algorithms for comparison, the datasets on which we evaluate these algorithms, the tasks, and the evaluation measures, and finally the experimental results.

3.4.1 Evaluated Models

Recall that our model captures network structure, textual content, and timestamp of the posts/tweets, and identifies the topics and parents of the posts, in addition to reconstructing the network connection strengths. Considering this, we evaluate and compare performance for the following models:

1. HMHP: This is the full version of our model with Gibbs sampling-based inference. It performs all reconstruction tasks mentioned above jointly while assuming a single topic for a post and topical interaction patterns.

2. HTM: This is the state-of-the-art model closest to ours that addresses the same tasks. The key modeling differences are two-fold: HTM assumes a topical admixture for each document instead of a mixture, and secondly, it models parent and child events to be 'close' in terms of their topical admixture. As a result, it cannot capture any sequential pattern in the topical content of cascades.

3. HWK+Diag: This is a simplification of our model where the topic-topic interaction is restricted to be diagonal. In other words, each topic interacts only with itself. This model helps assess the value of topical interaction and serves as a crude approximation of HTM since the original did not scale for our larger datasets.

4. HWK×LDA: This is motivated by the Network Hawkes model [59], which jointly models the network strengths and the event timestamps, and therefore performs parent assignment and user-user influence estimation jointly. However, it does not model the textual content of the events. Therefore, we augment it with an independent LDA mixture model [75] (LDAMM) to model the textual content. Thus, comparison against this model helps in analyzing the importance of doing topic, parent and network strength estimation jointly. The network reconstruction component of HWK×LDA is similar to the Network Hawkes model [59].

3.4.2 Datasets

We perform experiments on the following datasets.

(1) `USPol` (US Politics Twitter Dataset): This dataset is a set of $\sim 370K$ tweets extracted for 151 users who are the US Congress Members[2]. The tweets were extracted in July 2018 using the *Twitter API*, and for each member, the API returned at most 3200 tweets. Each tweet in the dataset consists of a timestamp, the user-id, and the tweet-text(content). The total vocabulary size here is $\sim$ 33000 words after removing the less frequent words. We use this dataset to evaluate and compare the models based on their ability to fit the heldout test set (*Data fitting quality*).

[2]https://twitter.com/verified/lists/us-congress/members?lang=en

(2) `TweetMarApr14` (Twitter Mar-Apr 2014 Dataset): The `TweetMarApr14` dataset was created by crawling $7M$ seed nodes from Twitter for the months of March-April 2014. We restrict ourselves to $500K$ tweets corresponding to top 5K hashtags from the most prolific $1M$ users generated in a contiguous part of March 2014. For each tweet, we have the time stamp, creator-id, and tweet-text. We use this dataset for discovering and analyzing topical interactions.

For the datasets, `USPol`, and `TweetMarApr14`, the true user-user influence parameters, the topic associated with each tweet, the parent for each tweet, the topic-topic network, and the topic-topic network parameters are not known to us. Even if the retweet information is available, notice that the retweet would have the same topic as that of the tweet as the content is the same for both, the tweet and the retweet and, retweets form only a small fraction of the parent-child relations that we intend to infer.

(3) `SemiSynth` (Semi-Synthetic Dataset): Since we need gold-standard labels to evaluate the performance of the models, we additionally create a `SemiSynth` dataset using the generative process of our model while preserving statistics of the real data to the extent possible. From a sample of the `TweetMarApr14` data, we retain the underlying set of nodes and the follower graph. Then we estimate all the parameters of our model (the base rate per user, user-user influence matrix, the topic distributions, resulting topical interactions and the user-topic preferences) from `TweetMarApr14` data. The document lengths are randomly drawn from $Poisson(7)$ since 7 was the average length of the tweets in the `TweetMarApr14` dataset. We finally generate 5 different samples of $1M$ events using our generative model. All empirical evaluations are averaged over the 5 generated samples (together termed as `SemiSynth`). The details of the parameter estimations are as follows. We assign as the parent for a tweet e from user u the 'closest in time' preceding tweet from the followees of user u for the last 1 day. If this set is empty, then e is marked as a spontaneous event. Topics to every tweet are assigned by fitting a latent Dirichlet analysis based mixture model (LDAMM [75]) with 100 topics. From the parent and topic assignments, we get the network strengths, user-topic preferences, topic-topic interactions, and topic-word distributions. For W_{uv} estimation (as well as a subsequent generation) we applied the edge grouping described at the end of the inference section and then estimate the W_{uv} for an edge as the smoothed estimate for the group. Users who did not generate a tweet were assigned an 'average' base rate and topic preference.

Now we describe the experimental setup for each of the tasks and then the results.

3.4.3 Reconstruction Accuracy

We address three reconstruction tasks: *parent identification*, *network reconstruction*, and *topic identification*. Since the gold-standard values are not known for the `TweetMarApr14` data, we address these tasks only on the `SemiSynth` data. For parent identification (or cascade reconstruction), given a ranked list of predicted parents for

	HMHP	HWK+DIAG	HWK×LDA
ACCURACY	0.581	0.362	0.370
RECALL@1	0.595	0.373	0.380
RECALL@3	0.778	0.584	0.589
RECALL@5	0.838	0.674	0.678
RECALL@7	0.87	0.73	0.733

Table 3.1: Cascade Reconstruction: accuracy and the recall with top-K (1,3,5,7) candidate parents.

	HMHP	HWK+Diag	HWK×LDA
MEAN APE	0.448	0.565	0.552
MEDIAN APE	0.255	0.283	0.287
MEAN APE ($N_u \geq 100$)	0.398	0.520	0.496
MEDIAN APE ($N_u \geq 100$)	0.235	0.265	0.264

Table 3.2: Network reconstruction error (as fraction).

each event, we measure accuracy and recall of the parent prediction. The accuracy metric is calculated as the ratio of number events for which the parent is identified correctly to the total number of events. For network reconstruction, we measure the distance between estimated and true W_{uv} values. We report the error in terms of the median Average Percentage Error (APE), defined as $\sum_{u,v} \frac{|W_{uv}-\hat{W}_{uv}|}{W_{uv}}$, where, $\hat{W}_{uv}$ is the estimated W_{uv} value. For comparison with HTM, we use Total Absolute Error (TAE) (the sum of the absolute differences between the true and estimated W_{uv} values) which was used in the original paper. For topic identification, we take a (roughly balanced) sample of event pairs and then measure (using precision/recall/F1) whether the model accurately predicts the event pairs to be from the same or different topics.

The model that is closest to ours is HTM [48]. However, the HTM assigns each event to a topic distribution instead of a single topic. Also, the available HTM code cannot scale to the size of our `SemiSynth` dataset. Therefore, we compared our model with HTM separately. We first present comparisons with the other baselines on the `SemiSynth` dataset. All the presented numbers are averaged over five different semi-synthetic datasets generated independently.

Table 3.1 presents the accuracy and the recall@k values for the task of predicting parents, and thereby the cascades. For all the algorithms, the recall improves significantly by the time the top-3 candidates are considered. For recall@3, HMHP performs about 32% better than both the other baselines, whereas recall@1, as well as the accuracy

	HMHP	HWK+Diag	HWK×LDA
Precision	0.893	0.123	0.781
Recall	0.746	0.367	0.752
F1	0.811	0.18	0.765

Table 3.3: Event Topic Identification (P/R/F1)

of HMHP, are at least 57% better than the corresponding number of the baselines.

For the network reconstruction task, Table 3.2 describes the various APE values for models HMHP, HWK+Diag, and HWK×LDA. Given that in our simulation, event generation was truncated by limiting the number of events, the last level of events contributes to negative bias affecting the APE, since the algorithms are still trying to assign children to these events. We report both the mean and the median errors for this task. The mean error for HMHP is about 18% lower than other baselines, and the median error is about 10% lower. We also present the results for the set of nodes where the number of generated children (N_u) is at least 100 (this is an easier task). Here also the mean APE for HMHP is at least 20% lower.

Table 3.3 presents the result of the topic modeling task, measured by calculating the precision-recall over event pairs as defined earlier. The results show that HMHP performs much better than HWK+Diag and at least 5%-6% better than HWK×LDA. The above experiment confirms that joint inference of topics, parents, and network strengths by modeling topical interactions is more accurate than that ignoring topical interactions and also decoupled inference. While the trend is not surprising as the data was generated using our model, the numbers confirm that when topical interaction patterns are present in the data, our inference algorithm is able to detect and make use of these. Also, the improvement over the baselines for all tasks is both significant in magnitude as well as statistically, showing the importance of topical interactions and joint inference for reconstructions tasks.

Comparison with HTM

HTM uses a Hawkes process to model the event times, as well as a topic model that takes into account the dependence of child event topics on parent event topics. While there is no notion of topic-topic interaction in HTM, it is the closest baseline to our model. Therefore we do a more exhaustive comparison against HTM. Unfortunately, the available code that we had for HTM needed approximately half a day for the ArXiv dataset (~37000 events) and did not scale to our larger datasets. So we performed experiments on datasets very similar to the ones used in [48]. Also, a key difference between HTM and HMHP is that in HTM each document is a mixture of topics (admixture model), whereas in HMHP each document is generated with a single topic (mixture model). So, we compare HMHP and HTM with respect to the

Window Length	1000	2000	3000	4000	5000
HTM	2.811	1.982	1.464	1.292	1.351
HMHP	1.297	0.925	0.677	0.646	0.657

Table 3.4: Network Reconstruction Error as TAE (Average over 5 datasets for each observation window length)(Cycle Graph with 10 nodes)

Window Length	1000	2000	3000	4000	5000
HTM	0.681	0.687	0.712	0.716	0.708
HMHP	0.926	0.924	0.95	0.94	0.935

Table 3.5: Cascade Reconstruction Accuracy (Average over 5 datasets for each observation window length)(Cycle Graph with 10 nodes)

quality of network and cascade reconstruction capabilities only and not with respect to the topic identification task. To measure the quality of network reconstruction, the evaluation metric used is Total Absolute Error (TAE).

We experimented with synthetic data generated both by the HMHP and the HTM models. We first generate a collection of datasets assuming the HMHP generative model with two different diffusion networks used in the HTM paper [48].

Circular diffusion network: This dataset is similar to that mentioned in HTM paper [48] (Appendix B.1), but with $|V| = 10$ nodes and $k = 10$ topics. Topic word distribution vector for each topic, as well as the topic-topic interaction vectors, are drawn from $Dir(0.1)$ and $Dir(0.01)$ respectively. Data is generated for various observation window length ranging from 1000 to 5000. For the window length 1000, around 350 events get generated and for window length 5000 around 1900. Table 3.4 shows TAE in network reconstruction for HTM and HMHP. The TAE for HMHP is about half of the TAE of HTM. Table 3.5 presents the percentage of correctly identified parents for both the models HTM and HMHP. Here as well, HMHP has about 30% better parent identification capability as compared to that of HTM.

ArXiv Top-200 Authors Network: We also perform an experiment on the ArXiv citation network dataset. The ArXiv Top-200 authors network dataset is created using the ArXiv high-energy physics theory citation network data from SNAP[3]. We restrict

[3]http://snap.stanford.edu/data/cit-HepTh.html (we have obtained our version of the dataset from the authors of HTM [48])

	Network Reconstruction (TAE)	Cascade Reconstruction
HTM	38.812	0.698
HMHP	23.151	0.951

Table 3.6: Arxiv Top-200 authors Graph - TAE for Network Reconstruction and Accuracy for Cascade Reconstruction

Window Length	1000	2000	3000	4000	5000
HTM	3.167	2.377	2.014	1.964	1.519
HMHP	1.696	1.200	1.168	1.396	1.243

Table 3.7: Network Reconstruction TAE for Cycle (HTM generated events – short documents)

ourselves to a graph formed by the top 200 authors in terms of the number of publications. This graph is similar to the ArXiv Top-200 authors dataset mentioned in [48]. Topic distributions, topical interactions, and the user-topic preferences are sampled from $Dir(0.1)$, $Dir(0.01)$, and $Dir(0.01)$ respectively. Here as well we compare HTM and HMHP with respect to the quality of network and cascade reconstruction. Table 3.6 presents the mean of (five datasets) TAE and the parent prediction accuracy values for both HTM and HMHP models. Here as well, TAE for HMHP is about 40% lesser than that of HTM and parent prediction accuracy is at least 30% better.

Data Generation using HTM**:** In this experiment, we generate data according to HTM model and compare the performances of HTM and HMHP models. The document generated for each event is a short document with length sampled from $Poisson(10)$. Table 3.7 shows the average TAE in network reconstruction for both HTM and HMHP. Here as well the error in case of HMHP is half of the error in HTM. Similarly, Table 3.8 presents the average parent identification accuracy for both HTM and HMHP. The accuracy for HMHP is at least $14 - 15\%$ better than that of HTM.

In summary, we see that HMHP outperforms HTM quite significantly in parent identification and network strength reconstruction, even when the underlying generative model is that of the HTM. We believe there are two reasons for this. First, a mixture model is a more appropriate model for short documents and can be estimated with greater confidence. Secondly, the topic-topic interaction matrix provides crucial additional evidence for parent identification under uncertainty, and this, in turn, leads to more accurate network strength reconstruction. In our remaining experiments, we do not evaluate the HTM further since it does not scale for the datasets that we use.

Window Length	1000	2000	3000	4000	5000
HTM	0.575	0.588	0.61	0.618	0.628
HMHP	0.716	0.730	0.736	0.730	0.748

Table 3.8: Cascade Reconstruction Accuracy for Cycle (HTM generated events – short documents)

3.4.4 Data fitting quality

Here, we compare the performance of the models based on the goodness of fit of each model on the heldout set (`Test` set), i.e. Log-Likelihood ($\mathcal{LL}$) of the heldout set. We perform this task on the real dataset `USPol`. For each event e in the `Test` set the observables are the time t_e, the creator-id c_e, and the words/content w_e. Whereas, the parent z_e and the topic η_e for each event are latent.

Let $\mathcal{X}$ and $\mathcal{Y}$ denote the set of events in the `Train` and `Test` sets respectively. As per HMHP, the total log-likelihood of the test set $\mathcal{Y}$, say $\mathcal{LL}(HMHP)$ is given as:

$$\begin{aligned}\mathcal{LL}(HMHP) &= \log\left(\prod_{e\in\mathcal{Y}} P(t_e, c_e, w_e)\right) = \sum_{e\in\mathcal{Y}} \log(P(t_e, c_e, w_e)) \\ &= \sum_{e\in\mathcal{Y}} \sum_{\substack{e'\in E c_e\in\mathcal{N}(c_{e'}) \\ t_{e'}<t_e}} \sum_{\eta_{e'}} \sum_{\eta_e} \left(\exp(-W_{c_{e'},c_e}) W_{c_{e'},c_e} \times \exp(\Delta t_e) \times \mathcal{T}_{\eta_{e'},\eta_e} \times P(w_e|\eta_e)\right) \\ &\qquad + \sum_{\eta_e} \exp(\mu_{c_e} T)\mu_{c_e} \mathcal{U}_{c_e,\eta_e} \times P(w_e|\eta_e) \end{aligned} \tag{3.14}$$

Here, the summations over $e' \in E$, over $\eta_{e'}$, and over η_e are for the marginalization over the candidate set of parents, topic of parent event, and the topic of event e respectively. As mentioned before, $\mathcal{T}_{\eta_{e'},\eta_e}$ is probability of transition from topic $\eta_{e'}$ to topic η_e, and thus for a fixed $\eta_{e'}$, we have $\sum_k \mathcal{T}_{\eta_{e'},k} = 1$. Also, there is users topic preference probability $\mathcal{U}_{c_e,\eta_e}$, and for a fixed user c_e, $\sum_k \mathcal{U}_{c_e,k} = 1$.

Here we are mentioning the likelihood for HMHP only. However, the likelihood of other models has a similar form.

Notice that, if a candidate parent $z_e = e'$ for an event $e \in \mathcal{Y}$ is in the training set $\mathcal{X}$, i.e. if $e' \in \mathcal{X}$, then parent for e' is inferred at the time of the training. Whereas if the candidate parent e' is such that $e' \notin \mathcal{X}$, i.e. $e' \in \mathcal{Y}$, then the parent for e' cannot be inferred. Which means, there is a need to marginalize over candidate set of parents for e' as well. Doing this recursively, to get the exact $\mathcal{LL}$ of event $e \in \mathcal{Y}$, in the worst case, requires summing over all the possible paths to event e. Therefore, calculating $\mathcal{LL}$ for all the events $e \in \mathcal{Y}$ would mean iterating over all the possible set of cascades, which requires $O(n!)$ computation, to the best of our knowledge, where, $|\mathcal{Y}| = n$.

However, if there is a guarantee that for each event $e \in \mathcal{Y}$, each candidate parent event e' is in the train set $\mathcal{X}$, then calculating the likelihood does not require summing over all the possible cascades. For each event, the computation is of the order $O(d)$, where d is the maximum number of candidate parents for any event $e \in \mathcal{Y}$. Then the total computation complexity is $O(dn)$. Bringing in the additional latent variable, i.e. η_e, the topic of event e, the computational complexity is of the order $O(dnK)$, where K is the maximum number of topics. Note that, as the candidate parent e' is in the `Train` set $\mathcal{X}$, the topic $\eta_{e'}$ of the event is inferred at the time of training, thus, summing over $\eta_{e'}$ is not required. Therefore, our goal here is to create a `Test` set such that the parent event for each event is in the `Train` set and for the events in the `Train` set either there are no latent variables, or are inferred.

The semi-synthetic dataset `SynthUSPol` is generated according to DNHP and thus we know the original cascades formed by the events in the data. As the generation process follows a level-wise generation of events (as described in Section 4.2.1), the last level events are the leaf events of the cascades formed. We use this level information from the `SynthUSPol` dataset to classify the events into `Train` and `Test` sets.

If the true cascade structure information is available, we could have used the leaf events as `Test` set and the non-leaf events as `Train` set. But, for the real dataset `USPol`, the true cascade structure is unknown. Therefore, it is not possible to use a straightforward approach of splitting the dataset into `Train` and `Test` sets, such that the parents of the test events are in train (with already inferred latent variables). Instead, we resort to some heuristic to split the dataset into `Train` and `Test` such that the assumption is likely to hold. Note that as an event e goes farther in time from its parent event e', the probability of generation of event e by the parent event e' decreases exponentially, because of the exponential dependence on time. Keeping this in mind, we split the dataset into `Train` and `Test` such that the events in the `Test` set are not that far from their some potential parent events in the `Train` set. To create the test set, events are processed sequentially. Each event $e \in E$ is added to the `Test` set $\mathcal{Y}$ if and only if at most p_{test} fraction of its candidate parents are already in the `Test` set $\mathcal{Y}$. This ensures that $1 - p_{test}$ fraction of its candidate parents are still in the `Train` set $\mathcal{X}$. Note that increasing (decreasing) p_{test} results in increasing (decreasing) the test set size as well as decreasing (increasing) the train set size. To study the effects of increasing training data size without reducing the test size, we use an additional parameter $0 \leq p_{data} \leq 1$ to decide whether to include an event in our experiments at all. Specifically, we first randomly include each event in the dataset with probability p_{data}, and then the `Train` and `Test` split is performed.

So now, given the above mentioned construction of the `Train`-`Test` split, the $\mathcal{LL}(HMHP)$ is equivalent to that in Equation 3.14, without the summation over candidate parent's topic $\eta_{e'}$, which is as follows.

$$\mathcal{LL}(\mathcal{HMHP}) = \sum_{e \in \mathcal{Y}} \sum_{\substack{e' \in E \\ c_e \in \mathcal{N}(c_{e'}) \\ t_{e'} < t_e}} \sum_{\eta_e} \left(\exp(-W_{c_{e'}, c_e}) W_{c_{e'}, c_e} \times \exp(\Delta t_e) \mathcal{T}_{\eta_{e'}, \eta_e} \times P(w_e | \eta_e) \right)$$

#Topics = 25			
Train(Test)	HMHP	HWK+Diag	HWKxLDA
$114K(70K)$	**-6750709.066**	-7015212.403	-6757375.544
$177K(100K)$	**-8718791.534**	-9106946.866	-8734721.678
$240K(130K)$	**-10403676.26**	-10915951.09	-10423458.8
#Topics = 100			
Train(Test)	HMHP	HWK+Diag	HWKxLDA
$114K(70K)$	**-6807667.698**	-7151739.301	-6822410.294
$177K(100K)$	**-8785192.647**	-9299455.97	-8816274.845
$240K(130K)$	**-10477518.23**	-11133668.14	-10514576.24

Table 3.9: Average Log-Likelihood for Time + Content for Real dataset `USPol`. (The Train(Test) sizes mentioned are approximate.)

$$+ \sum_{\eta_e} \exp(\mu_{c_e} T) \mu_{c_e} \mathcal{U}_{c_e,\eta_e} \times P(w_e|\eta_e) \tag{3.15}$$

Also, for all the three models, the sum over the candidate set of parents for each event is restricted to the events that are not farther in time and are in the `Train` set $\mathcal{X}$.

Table 3.9 presents the $\mathcal{LL}$ of time and content for HMHP, HWK×LDA, and HWK+Diag on the `USPol` dataset with $K = 25$ and $K = 100$. The rows indicate the sizes of the `Train` and `Test` when the events are selected in the dataset with probability p_{data} set as 0.5, 0.7, and 1.0 (which is the complete dataset). Then the `Train-Test` split is performed with p_{test} set as 0.3, which indicates the maximum fraction of candidate parents for each event in the `Test` set.

Observe that as expected all the models, HMHP, HWK×LDA, and HWK+Diag, get better at estimating the $\mathcal{LL}$ with the increase in the size of the dataset across different values of p_{test}. Similarly, as with the synthetic data experiments, HMHP outperforms HWK×LDA and HWK+Diag in fitting the time stamps and the textual content. This says that HMHP i.e. a model that incorporates topical interactions models real cascades better than the two baseline models which do not completely incorporate topical interactions.

3.4.5 Discovery and Analysis of topic interactions

The final task is to discover statistically significant topical interactions from textual information cascades and investigate what actionable insights can be drawn from such topical patterns. We stress that this task can only be performed using HMHP, and as such, there is no baseline algorithm for this task. We perform this task on the `TweetMarApr14` dataset. The first set of results on the `SemiSynth` data demonstrates

Parent topic hashtags	Child topic hashtags
steelers, browns, seahawks, fantasyfootball, nfl	mlb, orioles, rays, usmnt, redsox
renewui, ableg, aca, stopcombatingme, obamacare	uniteblue, tx2014, saysomethingliberalin4words, tcot, gophatesvets
idf, rnb, bds, gaza, israel	russia,iran,syria,crimea,ukraine
thewalkingdead, theamericans, onceuponatime, houseofcards, tvtag	arrow, agentsofshield, truedetective, longislandmedium, tvtag
egypt,gaza,israel,syria,iran	kiev, putin, russia, crimea, ukraine
rterugby, rugby, sixnations, engvwal, 6nations	coys, arsenal, cfc, mufc, facup

Table 3.10: Sample hashtags from asymmetric topics pairs.

that when the data has realistic parameters but is generated from the model that we hypothesize, HMHP outperforms the baselines. The generalization experiments further confirm that even for real data HMHP performs better than the baselines, suggesting that real data indeed better matches our modeling assumptions about interacting topics. This lends credibility to the topical interactions discovered by HMHP model, even though there is no ground truth for these.

In order to demonstrate the usefulness of the topic-topic interaction matrix, we perform three types of analysis using this matrix. We first select a set of topic-topic relationships as anecdotes. Next, we demonstrate how we can get insights on topic drift in cascades using a hubs and authorities analysis on the topic-topic interaction matrix. Finally, we use a personalized PageRank based analysis to discover related topics for any given topic and discuss how this can lead to new strategies for user targeting and influence maximization problems for spreading conversations about any topic.

Anecdotal parent-child topics

To identify interesting examples of topic-topic interaction, we first ensure that the topic pairs do not correspond to the same underlying topic that our model inappropriately partitioned into multiple finer topics. Such a phenomenon would cause a block-diagonal structure in the topic-topic interaction matrix. We first identify the most asymmetric topic pairs by sorting using $\mathcal{T}_{kk'} - \mathcal{T}_{k'k}$. Such pairs cannot conceivably come from topic splitting. To illustrate the topics, we find their top five hashtags. Table 3.10 shows examples of such topic pairs. In the first row, the parent and child hashtags are related to American football and baseball. These are different topics but indicate that tweets related to a particular sport (football) trigger tweets about some

Parent Tweet	Child Tweet
#MASalert Statement By Our Group CEO, Ahmad Jauhari Yahya on MH370 Incident. Released at 9.05 am/8 Mar 2014 MY LT	Missing #MalaysiaAirlines flight carrying 227 passengers (including 2 infants) of 13 nationalities and 12 crew members.
Investigators pursuing notion the 777 was diverted "with the intention of using it later for another purpose."	If #MH370 did fly for an additional 4 hrs as reported by WSJ, it could be anywhere in this circle.
Wanna be a part of GOTV in the first special election of year? Hop on our caller for Alex Sink in Florida: #FL13	Why #CPAC Isn't As Grassroots As You Think: #independents #moderates #tcot #gop #ccot #tlot #pjnet #uniteblue #p2
Certain people are ruining their reputations tonight-really sad! #Oscars	I should host the #Oscars just to shake things up - this is not good!
Dave Gallo and Bob Orr on the mysterious case of #MH370, Robert Wagner, and @JaneFonda tonight at 11pm ET on @PBS RT	@motherfuton: Best explanation I've read in regards to the missing Malaysia flight, #MH370.http://t.co/KdSgxUepfV
mt@conor64 Obama Is Complicit in Covering Up the Truth About #CIA Torture #p2 #tcot #topprogs #teaparty #uniteblue	The moral of 'Green Eggs and Ham' is lost on Ted Cruz and Sarah Palin. #UniteBlue and #TryNewThings
Gellman:My definition of whistle blowing:are you shedding light on crucial decision that society should be making for itself. #snowden	Gellman we are living inside a one way mirror,they & big corporations know more and more about us and we know less about them #sxsw

Table 3.11: Example Parent-Child Tweets

other sport (baseball). Alternatively, there can be users who participate in discussions related to both sports. Table 3.11 shows a few actual parent-child tweets from the selected asymmetric topic pairs. It is important to stress that these tweet pairs are not retweets. Also, though all these tweet pairs clearly on related topics, often (like the first pair about *MH370*) these do not share any hashtags or even other significant words, which some naive strategy might use to detect such relationships. Therefore, in both cases, we see evidence of interesting topic interactions that are discovered by our model.

Such significant topical interactions can then serve as input to various further tasks e.g. getting a global view of ongoing discussions on some events, possibly better estimations of which topics are going viral, etc. Indeed, we can argue that topics, rather than hashtags, represent the correct granularity for such analysis.

Source Topic hashtags	Hashtags from top-3 transitioned topics
{ukraine, crimea, russia, putin, syria}	{utpol, raisethewage, cdnpoli, obamacare, aca}, {irs, tcot, teaparty, gophatesvets, uniteblue}, {worldbookday, amwriting, books, litfestlive}
rhlaw, clinton, r4today, ecommerce, cadem14}	{irs, tcot, teaparty, gophatesvets, uniteblue}, {soundcloud, hiphop, mastermind, nowplaying, music}, {iahsbkb, nba, iubb, lakers, rockets}
{agentsofshield, arrow, tvtag, supernatural, chicagoland}	{idol, bbcan2, havesandhavenots, pll, thegamebet}, {tvtag, houseofcards, agentsofshield, arrow, theamericans}, {soundcloud, hiphop, mastermind, nowplaying, music},

Table 3.12: Example related topics using personalized pagerank. (After removing some top generic topics)

Modeling Topic Drifts via random walks

One of the prominent applications of information cascade modeling is in targeting– how to intelligently select a small set of triggers in the network that can encourage conversations/tweets about topics or hashtags that the advertiser is interested about. We claim that our model, by capturing the topic-topic interactions, provides alternate topics or hashtags for such advertisers to exploit. To demonstrate this, we choose a few of the topics and then run personalized pagerank [72] from these topic nodes on the Markov chain underlying the topic-topic interaction matrix. Table 3.12 shows the results of running personalized PageRank from example start topics. For each topic in the first column of Table 3.12, the second column contains the 3 topics (other than the topic itself) with the highest personalized PageRank. For the examples in the first 3 rows, most of the top 3 transitioned topics are the high authority score topics (more general ones). However, after removing some of these topics from the transition matrix, the top 3 transitioned topics (right column last 3 rows) have topics that are directly related to the start topic. For e.g. the start topic in the fourth row is about Russia, Syria, and Ukraine, and the transitioned topics have hashtags that are related to politics in the US. This gives a hint about the connection between the US, Russia, Syria, and Ukraine in Twitter conversations. Similarly, in the last row, the start topic is about TV shows, and the transitioned topics also contain hashtags which are about some other TV shows.

One interpretation of this is that the average conversation that starts with any topic from the first column, can be expected to drift to one of the topics in the corresponding row. While this knowledge can have many applications, it is particularly useful for advertisers who are interested in promoting specific keywords or hashtags.

To summarize, using multiple semi-synthetic datasets we show that the HMHP significantly outperforms state-of-the-art baselines including the HTM in cascade and

network reconstruction by virtue of modeling topical interactions and joint inference of parents, topics and network strengths. It also generalizes significantly better for real Twitter data, demonstrating that topical interactions is indeed a better model for the real cascade of tweets. The scalability of HMHP makes it amenable to real web-scale applications. Finally, we analyzed the topical interactions identified by HMHP to provide guidelines about how this could be used for influence maximization, as well as unearthing new insights about topic drift in cascades.

3.5 Related Work

There has been a lot of work on network reconstruction based on the observations of event times [36, 89]— these models often ignore the content information of the events. Our work shows that such content information, when present, can profitably be used for a better estimation of the network strengths. [4] studies a related model in which activation times and some side information (e.g. tweet content) about cascades are observed, but not the set of users.

The Dirichlet Hawkes Process (DHP) [28] and the Hawkes Topic Model (HTM) [48] both extend Hawkes Processes to model textual content associated with events. However, neither captures topical interactions. The DHP lacks the notion of a unique parent for any event and also any notion of users or networks. Our model is most similar to the HTM. But while the HTM forces parent and child events to be topically close, it does not capture any parent-child topical patterns beyond this. One consequence is that no two events are topically identical, while we can group events according to assigned topics and, additionally, parent-child relations according to topic pairs. HTM's document model is an admixture of topics, which is more powerful in general, but in the context of short documents such as tweets, this complexity is not as necessary, as we demonstrate in our experiments.

Various generalizations of Dirichlet Process such as Hierarchical Dirichlet process (HDP) [82] and variants of Chinese Restaurant Process (CRP) such as Nested CRP (nCRP) [87], the Chinese Restaurant Franchise Process (CRFP) [64] in combination with Hawkes processes have also been explored in literature. Seonwoo et al. [82] proposed Hierarchical Dirichlet Gaussian Marked Hawkes process (HD-GMHP) that reconstructs narratives and thread structure of discussions from the text data. HD-GMHP uses Hawkes process to model the temporal information of events, the textual information is modeled using Gaussian distribution (as similar to that in HTM), and the trigerring relationships are modeled using mixture of Gaussians.

In [87] the authors proposed a model named Nested Chinese Restaurant Process with Hawkes Gaussian process (nCRP-HGP) which combines the advantages of hierarchical nonparameteric Bayesian models (nCRP) with Hawkes processes to model the temporal and textual information of the events. nCRP-HGP incorporates the events in which multiple senders and receivers are involved, and exploits the properties of the senders and receivers for better modeling of the temporal information and textual content.

Mavraforakis et al. [64] developed Hierarchical Dirichlet Hawkes Processes (HDHP) model by replacing Chinese Restaurant Process (CRP) with a Chinese Restaurant Franchise Process (CRFP) which is capable of modeling streaming data from multiple users. The model uncovers the various latent learning patterns from the series of actions that users take on platforms like Stack Overflow and Wikipedia while gathering some new information on some topic.

All these models ([82, 64, 87]) are powerful in terms of modeling the textual content, but, either, as similar to DHP does not take into account the notion of users or network over users or as in HTM, forces parent-child text content to be topically close, which does not help in capturing topical relationships between parent-child topics.

In addition to the above mentioned models, models for sequence data have blended Markovian dynamics with topic models [41, 43, 27, 5]. This thread, is focused on modeling richer sequential structure within a single document, but has not looked at cascades of events with associated timestamps and networked users. Our contribution is in merging this line of research with the modeling of information cascades.

3.6 Conclusion

In summary, we propose a generative model for information diffusion cascades accounting for topical interactions, by coupling a Network Hawkes process with a Markov Chain over topics for diffusion paths. This enables us to fit real Twitter conversations better, and the use of topic interactions and collective inference using our model also leads to a more accurate reconstruction of network strengths, diffusion paths, and event topics. Using comparisons on a number of datasets, we show that HMHP outperforms the baselines on the standard metrics. On top of this, using HMHP, we are able to derive insights about topical interactions, which existing models cannot.

Chapter 4

Dual Network Hawkes Process

4.1 Introduction

In this chapter, we address the problem of modeling text-based information cascades, generated over a network of user nodes. There exists a number of models that uses a variety of non-homogeneous point-processes to model such cascades, with rate parameters capturing user-user influence [59], user-topic affinity [48], etc. Recent work [17] also incorporates topic-topic interactions in modeling the textual content. However, in these models, the content does not influence the response rate. As a result, two related and important questions are yet unexplored– (1) *how to incorporate the influence of topics on the event rate?* and (2) *how to decompose the overall responsiveness for a pair of users in terms of the responsiveness on different topics?* For e.g., during political elections, two politically inclined users are likely to interact faster on politics. However, their interaction rate in fashion related conversations can be quite different.

In this chapter, we address these two issues by extending the Network Hawkes Process [59] which executes over a one-dimensional network over users, to propose a *Dual-Network Hawkes Process* (DNHP) which executes over a two-dimensional network over user-topic pairs. Individual events now trigger for a user-topic pair. Each such event spawns a new Poisson process for every other user-topic pair in the neighborhood, whose rate is determined by the Once triggered, the cascade now proceeds via the Hawkes process that has as parameters the influence that one user-topic pair has on another. After the event generation step, the process becomes asymmetric for users and topics. While the users are directly observed, the topics are latent and only the textual content of the events, generated i.i.d. conditioned on the topics, are observed. The Network Hawkes Process is recovered as a special case of this model, (by disregarding the event content) when the number of topics is 1 (effectively observable).

We capture the influence between two user-topic pairs as a product of two different parameters, (a) the connection strength between the users, and (b) the connection strength between the topics. This decomposition leads to significant parameter shar-

ing between individual point processes. The rate for any two users u_1, u_2 and a specific pair of topics t_1, t_2 shares the topic-topic component of the parameter with the rate for two other users u_1', u_2' and the same pair of topics. This means that for predicting the reaction time for (user u_2, topic t_2) for (user u_1, topic t_1), we can use as evidence reaction times for other pairs of users on the same topic pair, *even when u_1 and u_2 have never interacted before on any topic.* Similar benefits are obtained for two different topic pairs and the same user pair. Thus in addition to being closer to the generation of real-life topical information cascades, the Dual-Network Hawkes Process promises significantly better generalization based on limited training data via parameter sharing.

Using the model, we address the task of recovering the user-user and topic-topic connection strengths (up to their product), along with recovering the latent topic and parent (or trigger) event for each event. For the user-user graph, we assume the structure to be known and the edge weights We consider the topics also to be connected as a weighted graph, where connection strengths between topics determine how quickly conversations transition between topics. Unlike the user graph, the structure of the topic graph is unknown and needs to be recovered along with the edge parameters. A significant challenge for parameter estimation is that the user-user and topic-topic weights are intrinsically coupled in our model and cannot be integrated out analytically. In fact, we show that the topic-topic weights and user-user weights are only identifiable as their product and not individually. We address the coupling issue by showing that the posterior distribution of the user-user (topic-topic) weights is conditionally Gamma distributed given the topic-topic (user-user) weights. Based on the conditional distributions, we propose a Gibbs sampling based inference algorithm for these tasks. In our inference algorithm, the update equations for the user-user and topic-topic weights become coupled, thereby allowing the flow of evidence between them.

Experiments over semi-synthetic and real-world data show that our model outperforms the state-of-the-art models in terms of topic identification, parent identification as well as user-user connection strength recovery. In addition, we also show that the generalization performance of our model is much better with respect to the baselines.

4.2 Dual-Network Hawkes Process (DNHP)

We consider text based cascades generated by a set of users $V = \{1, 2, \ldots, n\}$. There is a set of edges $\mathcal{E}$ among users V, over which these text-based cascades are formed or the information propagates. Each edge (u, v) is weighted as $\mathcal{W}_{u,v}$, which indicates the extent of the influence of user u on user v. We assume that the unweighted graph over the users is known or observed but the weights $\mathcal{W}_{u,v}$ are not. For each event e, where an event can represent a tweet or a social media post, the time at which the event was created t_e is observed along with the user c_e, who created the event, and the document d_e of the event. Each event has a latent variable η_e which represents the

topic of the event from which the document d_e is sampled using the topic-model over the vocabulary $\mathscr{V}$. Along with the latent topic η_e, there is one more latent variable z_e associated with each event. The latent variable z_e denotes the parent event of event e, i.e. the event that triggered event e. The events which are triggered by some other event are termed as *diffusion* events, and if an event is not triggered by any other event, then it is termed as *spontaneous* event. Notice that the diffusion events lead to the formation of cascades, and the spontaneous events represent the start of the cascade.

Motivated by the HMHP model, which posits the existence of the topic-topic interaction matrix, here as well we assume the existence of the topic-topic graph over the nodes $[K]$. The topic-topic graph as well as the topic-topic interaction strengths $\mathcal{T}_{k,k'}$ for every pair of topics $k, k' \in [K]$ are unobserved. The topic-topic interaction strength between a pair of topics represents how quickly the conversations transit over these topics.

Now, to model the response rate of an event as a combination of user and topic, we define a new graph $\mathcal{G}$ with $\mathcal{V}$ as a set of nodes, where each node of the graph is indexed by a user-topic pair $\mathbb{c} = (u, k)$. As opposed to the models HMHP [17], HTM [48], Network Hawkes [59], etc., where the Hawkes process is defined for each node of the user-user graph, here Hawkes process is defined for each node $\mathbb{c} = (u, k)$ of the graph $\mathcal{G}$. For every pair of nodes $((u, k), (v, k'))$ in the graph $\mathcal{G}$ an edge exists if the corresponding entry of the matrix $\mathcal{W} \otimes \mathcal{T}$, is non-zero, where $\otimes$ denotes Kronecker product. Therefore, now each event e which is generated by the Hawkes process at a node (u, k) of the graph $\mathcal{G}$ is consumed by the neighbors of the node (u, k) in the graph $\mathcal{G}$ and tries to trigger multiple events at each of the neighbors resulting into a formation of cascade. The complete generation process for the DNHP can be described in two phases, i.e. *modeling time and topic* and *modeling textual content*. Following is the description of each of the phases.

4.2.1 Modeling Time and Topic

Let $E = \{e\}$ be the set of all generated events. Each event e is defined using a tuple $e = (t_e, \mathbb{c}_e, z_e, d_e)$, where, t_e is the time at which event was generated, z_e indicates the unique parent event of e (i.e. the event that triggered event e), and d_e is the textual content associated with event e. Here, $\mathbb{c}_e = (c_e, \eta_e)$, where $c_e \in V$, is the user associated with the event e and $\eta_e \in [K]$ is the topic associated with the event. For any event e, all the events e' such that $t_{e'} < t_e$ and $c_{e'} \in \mathcal{N}(c_e)$, where $\mathcal{N}(c_e)$ denote the set of neighbors of c_e in the user-user graph, are the set of candidate parent event. In this phase, $(t_e, \mathbb{c}_e, z_e)$ for each event is generated using the Multivariate Hawkes Process (MHP) on graph $\mathcal{G}$ following the existing models ([84, 59, 48, 17]).

Let $\mathcal{H}_{t-}$ denote the set of all events generated prior to time t. Then, $\lambda_{\mathbb{v}}(t)$, the intensity function for $\mathbb{v}$ is given by the superposition of the base intensity of node $\mathbb{v} \in \mathcal{G}$ and the impulse response of the events $e \in \mathcal{H}_{t-}$ generated by the node $\mathbb{c}_e$ in the history

of events. $\lambda_{\mathbb{v}}(t) = \mu_{\mathbb{v}}(t) + \sum_{e \in \mathcal{H}_{t-}} h_{\mathbb{c}_e,\mathbb{v}}(t - t_e)$. Here, $\mu_{\mathbb{v}}(t)$ is the base intensity which is independent of the history of events, and $h_{\mathbb{c}_e,\mathbb{v}}$ captures the impulse response on node $\mathbb{v}$ because of an event at node $\mathbb{c}_e$. The base intensity for node $\mathbb{v} = (v, k)$ is defined as $\mu_{\mathbb{v}}(t) = \mu_v(t) \times \mu_k(t)$, where, $\mu_v(t)$ is base intensity associated with user v, and $\mu_k(t)$ is the base intensity for topic k. The impulse response $h_{\mathbb{u},\mathbb{v}}$ with nodes $\mathbb{u} = (u, k)$ and $\mathbb{v} = (v, k')$ can also be further decomposed into product of user-user influence term $\mathcal{W}_{u,v}$, topic-topic interaction term $\mathcal{T}_{k,k'}$ and a time-kernel term $f(\Delta t)$ as $h_{\mathbb{u},\mathbb{v}}(\Delta t) = \mathcal{W}_{u,v}\mathcal{T}_{k,k'}f(\Delta t)$.

To generate events with the intensity function $\lambda_{\mathbb{v}}(t)$, we follow the level-wise generation of events as described in [84]. Let, Π_0 be the level 0 events which are generated with the base intensity of the nodes $\mathbb{v} \in \mathcal{G}$, i.e. $\mu_{\mathbb{v}}(t)$. Then, the events at level $\ell > 0$ are generated as per the following non-homogeneous Poisson process: $\Pi_\ell \sim Poisson\left(\sum_{(t_e,\mathbb{c}_e,z_e)\in\Pi_{\ell-1}} h_{\mathbb{c}_e,*}(t, t_e)\right)$ This process can be described with the help of the pseudo-code as follows: for each event $e = (t_e, \mathbb{c}_e, z_e) \in \Pi_{\ell-1}$, for each neighbor $\mathbb{v} \in \mathcal{G}$ of node $\mathbb{c}_e$, generate events at node $\mathbb{v}$ according to non-homogenous Poisson process – $Poisson(h_{\mathbb{c}_e,\mathbb{v}}(t - t_e))$.

We would like to highlight to the reader that in this work we model the time-kernel term $f(\Delta t)$ using a simple exponential function i.e. $\exp(\Delta t)$. We understand that there has been a number of recent works that model the time-kernel efficiently ([48], [28], [59]), but the main aim of this work is not that.

4.2.2 Modeling Documents

Now, to generate the document d_e for each event e the only information required is the information about the topic η_e associated with the event, and the topic-word distribution for topic η_e. However, in the first phase of the generation of events, for each event, the topic is identified based on the $\mathbb{c}_e = (c_e, \eta_e)$ on which the event is generated. Thus, the for each event $e = (t_e, (c_e, \eta_e), z_e)$, the document is generated by sampling $Poisson(10)$ words from the topic-word distribution $\boldsymbol{\zeta}_{\eta_e}$.

The joint probability for the model is given as:

$$
\begin{aligned}
&P(E, \boldsymbol{\mathcal{W}}, \boldsymbol{\mathcal{T}}, \boldsymbol{\zeta}, \boldsymbol{\eta}, \boldsymbol{z} \mid \boldsymbol{\alpha}, \boldsymbol{\mu}, \alpha_1, \alpha_2, \beta_1, \beta_2) \\
&\qquad = \prod_{\mathbb{v}\in\mathcal{V}} \left[exp\left(- \int_0^T \mu_{\mathbb{v}}(\tau) d\tau \right) \prod_{e \in E} \mu_{\mathbb{v}}(t_e)^{\delta_{\mathbb{c}_e,\mathbb{v}}\delta_{z_e,0}} \right] \\
&\qquad \times \prod_{e\in E} \prod_{\mathbb{v}\in\mathcal{V}} \left[exp\left(- \int_{t_e}^T h_{\mathbb{c}_e,\mathbb{v}}(\Delta\tau) d\tau \right) \prod_{\substack{e' \in E \\ \mathbb{c}_{e'}=\mathbb{v} \\ z_{e'}=e}} h_{\mathbb{c}_e,\mathbb{c}_{e'}}(\Delta t_{e'}) \right] \\
&\qquad \times \prod_{k=1}^{K} P(\boldsymbol{\zeta}_k \mid \boldsymbol{\alpha}) \times \prod_{e\in E} \left[\prod_{w=1}^{N_e} P(x_{e,w} | c_{e_k}, \boldsymbol{\zeta}_{c_{e_k}}) \right]
\end{aligned}
$$

$$\times \prod_{(u,v)\in\mathcal{E}} P(\mathcal{W}_{u,v}|\alpha_1,\beta_1) \prod_{(k,k')\in([K],[K])} P(\mathcal{T}_{k,k'}|\alpha_2,\beta_2) \quad (4.1)$$

Here, the term on the first line describes the probability of generating spontaneous events. The term on the second line describes the probability of the formation of cascades. The first term on the third line, indicates the probability of sampling topic-word distribution $\boldsymbol{\zeta}_k$ for each topic k from a $Dirichlet(\boldsymbol{\alpha})$. And the last term there is the probability corresponding to generating document for each event. The terms on the last line correspond to the probability of sampling $\mathcal{W}_{u,v}$ (user-user influence) and $\mathcal{T}_{k,k'}$ (topic-topic interactions), which we consider as sampled from *Gamma* priors.

4.2.3 Identifiability in DNHP

We notice that user-user and topic-topic weights are not separately and individually identifiable in our model - i.e. the increased rate between two user-topic pairs can be explained either by increasing the user-user weight or the topic-topic weight. As a result, only the product of a user-user weight over topic-topic weights is identifiable, which is what we estimate in our experiments.

We formalize this observation in the following lemma.

Lemma 20. *Let $\theta > 0$ be any constant. For any set of events E and any set of parameter values of $\mathcal{W}, \mathcal{T}, \boldsymbol{\zeta}, \boldsymbol{\eta}, \boldsymbol{z}$, the probability $P(E \mid \mathcal{W}, \mathcal{T}, \boldsymbol{\zeta}, \boldsymbol{\eta}, \boldsymbol{z})$ is equal to the probability $P(E \mid \theta\mathcal{W}, \mathcal{T}/\theta, \boldsymbol{\zeta}, \boldsymbol{\eta}, \boldsymbol{z})$, i.e. where $\mathcal{W}$ and $\mathcal{T}$ have been replaced by $\theta\mathcal{W}$ and $\mathcal{T}/\theta$ respectively.*

The proof can be obtained by simply writing the actual expression for the probability and is hence omitted.

4.2.4 Stability of DNHP

One of the important properties of the Hawkes processes which makes it a perfect fit for the cascades of posting events is the *mutually exciting* property. Each event that has occurred in history adds a non-negative impulse response to the intensity function and thus increases the likelihood of the new events in the future. Because of this recurrent nature of the Hawkes processes, we need to ensure that the generative process does not lead to a generation of an infinite number of events.

Lemma 21. *To ensure that DNHP does not generate infinite number of events it is sufficient to ensure that $\lambda_{\max}(A \odot \mathcal{W}) < 1$ and $\lambda_{\max}(B \odot \mathcal{T}) < 1$, where, λ_{max} denotes highest eigenvalue, A and B are the adjacency matrices for the user-user and topic-topic graphs respectively, and $\mathcal{W}$ and $\mathcal{T}$ define the user-user influence and topic-topic interaction matrices respectively.*

Proof. As described in [21] and [59], for the system to not generate infinite number of events the condition that is to be satisfied is: $\lambda_{\max} = \max |eig(A \odot \mathcal{W})| < 1$ where, A is the adjacency matrix and $\mathcal{W}$ matrix specifies the weights over the edges of A and *eig* denotes eigenvalue. On the similar lines, for DNHP the condition becomes $\lambda_{\max} = \max |eig((A \odot \mathcal{W}) \otimes (B \odot \mathcal{T}))| < 1$ where, A and B are the adjacency matrices for the user-user and topic-topic graphs respectively, and $\mathcal{W}$ and $\mathcal{T}$ define the user-user influence and topic-topic interaction matrices respectively. In order to satisfy the above condition it is sufficient to ensure that $\lambda_{\max}(A \odot \mathcal{W}) < 1$ and $\lambda_{\max}(B \odot \mathcal{T}) < 1$, i.e. the maximum eigenvalue of each of the matrices is bounded by 1. However, in our case, the matrix A is fixed and the priors are used for the user-user influence values $\mathcal{W}_{u,v}$. Additionally, the graph B is assumed to be a complete graph and the priors are used for the topic-topic interactions weights. To favour conjugacy in the posterior of the $\mathcal{W}_{uv}$ and $\mathcal{T}_{k,k'}$, a *Gamma* prior is selected for both $\mathcal{W}_{uv}$ and $\mathcal{T}_{k,k'}$. Given that A is fixed we claim that $\lambda_{\max}(A \odot \mathcal{W}) \leq d_{\max}\mu^{\mathcal{W}}_{\max} + \sigma^{\mathcal{W}}_{\max}\sqrt{d_{\max}}$, where $d_{\max}$ is the maximum degree in A, and $\mu^{\mathcal{W}}_{\max}$ and $\sigma^{\mathcal{W}}_{\max}$ are the maximum mean and variance of the *Gamma* priors used for the user-user influence values $\mathcal{W}_{uv}$. Thus, to have the $\lambda_{\max}$ of $(A \odot \mathcal{W})$ to be bounded by 1, it is enough to choose appropriate $\mu_{\max}$ and $\sigma_{\max}$ such that $d_{\max}\mu_{\max} + \sigma_{\max}\sqrt{d_{\max}} < 1$.

Similarly, to ensure that the $\lambda_{\max}(B \odot \mathcal{T}) < 1$, notice that $\lambda_{\max}(B \odot \mathcal{T}) \leq K\mu^{\mathcal{T}}_{max} + \sigma^{\mathcal{T}}_{max}\sqrt{K}$, where $\mu^{\mathcal{T}}_{\max}$ and $\sigma^{\mathcal{T}}_{\max}$ are the maximum mean and variance of the *Gamma* priors used for the topic-topic interaction weights $\mathcal{T}_{k,k'}$ for all $k, k' \in [K]$. And thus, choosing a *Gamma* prior with appropriate values of $\mu^{\mathcal{T}}_{\max}$ and $\sigma^{\mathcal{T}}_{\max}$ should give the required condition. □

4.3 Approximate Posterior Inference

As similar to that of the HMHP model, the latent variables associated with each event in case of DNHP are the parent of an event (z_e), and the topic of the event (η_e). The variable z_e for each event e can be either 0 indicating a spontaneous event or some event e' in the history of events. Along with these latent variables, the user-user influence matrix $\mathcal{W}$, the topic-topic interaction matrix $\mathcal{T}$, user base rate for each user and topic base rate for each topic have to be estimated.

As the exact inference is intractable, we perform inference using the Gibbs sampling algorithm. As opposed to the HMHP model, here the topic-topic interaction strengths $\mathcal{T}_{k,k'}$ are tightly coupled with the user-user influence $\mathcal{W}_{u,v}$ terms in the likelihood and cannot be integrated out analytically. I.e. DNHP does not enjoy the conjugacy property, and thus the topic-topic interaction terms are to be sampled for each pair of topics. However, it turns out that the user-user influence $\mathcal{W}_{u,v}$ is *Gamma* distributed conditioned on topic-topic interactions and the topic-topic interaction $\mathcal{T}_{k,k'}$ is *Gamma* distributed conditioned on the user-user influence values. Thus, we use Gibbs sampling to iteratively sample the individual topic assignments (η_e), parent assignments (z_e), the user-user influence $\mathcal{W}_{u,v}$, the topic-topic interaction strengths $\mathcal{T}_{k,k'}$, user base

rate μ_v for each user and topic base rate μ_k for each topic from their conditional distributions given current assignments to all other variables until convergence. Following are the conditional distributions required for sampling the different latent variables.

Parent Assignment: The parameters on which the assignment of the parent event is dependent are: (1) *influence of the user $c_{e'}$ of candidate parent event e' on the user c_e of the current event e*, (2) *interaction between the topics $\eta_{e'}$ and η_e of the candidate parent event and current event respectively*, and (3) *the time of occurrence of candidate parent event and current event*. Therefore, once these parameters are fixed, then the conditional probability of event $e' = (t_{e'}, (c_{e'}, k'), z_{e'})$ being a parent of an event $e = (t_e, (c_e, k), z_e)$ is given as:

$$P(z_e = e'|\mathcal{W}, \mathcal{T}) \propto \exp(-\mathcal{W}_{c_{e'},c_e}\mathcal{T}_{\eta_{e'},\eta_e})\mathcal{W}_{c_{e'},c_e}\mathcal{T}_{\eta_{e'},\eta_e} f(\Delta t_{e'}) \tag{4.2}$$

Similarly, conditional probability of an event being spontaneous can be given as:

$$P(z_e = e|\mu_{c_e}, \mu_{\eta_e}, T) \propto exp(-T\mu_{c_e}\mu_{\eta_e})\mu_{c_e}\mu_{\eta_e} \tag{4.3}$$

Notice that the denominator or the total probability is same for all the candidate parents and thus we have the proportionality sign the above equations.

Topic Assignment: Notice that the parameters on which the assignment of the topic variable η_e for an event e is dependent on are: (1) *interaction between the topics $\eta_{e'}$ and η_e of the candidate parent event and current event respectively* (2) *influence of the user $c_{e'}$ of candidate parent event e' on the user c_e of the current event e*, (3) the content d_e of the current event e, (4) *interaction between the topics η_e of the current event and the topic η_f for all event f which are child events of the current event*, and (5) *influence of the user c_e of current event e on the users c_f where events $\{f\}$ are the child events of event e*. Therefore, once these parameters are fixed, the conditional probability of assigning a topic k to an event $e = (t_e, \mathbb{c}_e = (c_e, \eta_e), z_e)$ by user c_e when the parent event is $e' = (t_{e'}, \mathbb{c}_{e'} = (c_{e'}, \eta_{e'}), z_{e'})$ is given as:

$$\begin{aligned} P(\eta_e = k|\boldsymbol{z}, \mathcal{W}, \mathcal{T}, \boldsymbol{\alpha}) \propto & \frac{\prod_{w \in d_e} \prod_{i=0}^{N_{d_e}^{w}-1} \alpha_w + \mathfrak{T}_{k,w}^{\neg e} + i}{\prod_{i=0}^{N_{d_e}-1} \sum_{w \in \mathscr{V}} \alpha_w + \mathfrak{T}_{k}^{\neg e} + i} \\ & \times exp\left(-\mathcal{W}_{c_{e'},c_e}\mathcal{T}_{\boldsymbol{\eta_{e'},\eta_e}}\right)\mathcal{W}_{c_{e'},c_e}\mathcal{T}_{\boldsymbol{\eta_{e'},\eta_e}} f(\Delta t_e) \\ & \times \prod_{\substack{f \in E \\ z_f = e}} \left[exp\left(-\mathcal{W}_{c_e,c_f}\mathcal{T}_{\eta_e,\eta_f}\right)\mathcal{W}_{c_e,c_f}\mathcal{T}_{\eta_e,\eta_f} f(\Delta t_f)\right] \end{aligned} \tag{4.4}$$

Here, $\mathfrak{T}$ is the count matrix of dimension $K \times \mathscr{V}$ storing the count of each word for each topic. $\mathfrak{T}_{k,w}^{\neg e}$ indicates the count of word w in topic k excluding the counts from event e. The first term on the RHS is related to the content of event e, the second term is related to the influence of parent event on the current event, and the third term is related to the influence of current event on the child events f for generating event with topic η_f. When event e is a spontaneous event, the conditional probability has a

similar form with an exception to the second term, which is given as $exp(-\mu_v \mu_k T)\mu_v \mu_k$, indicating the probability of generation of an spontaneous event with topic k.

User-User Influence$(\mathcal{W}_{u,v})$: The conditional probability of influence of user u on v is dependent on the number of times the users u and v interacted. To be specific, it is dependent on the number of times a user u generated some event and tried to trigger event at user v (which is related to N_u – the number of events by user u) and the number of times user v got influenced and generated an event (which is $N_{u,v}$ – the number of interactions between u and v). So, the conditional probability for $\mathcal{W}_{u,v}$ can be given as:

$$
\begin{aligned}
P(\mathcal{W}_{u,v} \mid \mathcal{T}, \boldsymbol{z}) &\propto P(\mathcal{W}_{u,v}|\alpha_1, \beta_1) \cdot \prod_{\substack{e \in E \\ c_e = u}} \left[exp\left(- \int_{t_e}^{T} h_{\mathbb{u},\mathbb{v}}(\Delta\tau) d\tau \right) \prod_{\substack{e' \in E \\ c_{e'} = v}} h_{\mathbb{u},\mathbb{c}_{e'}}(\Delta t_{e'})^{\delta_{z_{e'},e}} \right] \\
&\propto P(\mathcal{W}_{u,v}|\alpha_1, \beta_1) \times \prod_{\substack{e \in E \\ c_e = u}} \left[exp\left(-\mathcal{W}_{u,v} \left(\sum_{k'} \mathcal{T}_{\eta_e, k'} \right) \right) \prod_{\substack{e' \in E \\ c_{e'} = v}} \mathcal{W}_{u,v} \mathcal{T}_{\eta_e, \eta_{e'}}^{\delta_{z_{e'},e}} \right] \\
&\propto P(\mathcal{W}_{u,v}|\alpha_1, \beta_1) \times \left[exp\left(-\mathcal{W}_{u,v} \sum_{k} \left(N_{u_k} \sum_{k'} \mathcal{T}_{k,k'} \right) \right) \mathcal{W}_{u,v}^{N_{u,v}} \prod_{k,k'} \mathcal{T}_{k,k'}^{N_{(u,k),(v,k')}} \right] \\
&\propto P(\mathcal{W}_{u,v}|\alpha_1, \beta_1) \times \left(\mathcal{W}_{u,v}^{N_{u,v}} exp\left(-\beta \mathcal{W}_{u,v} \right) C \right) \qquad (4.5)
\end{aligned}
$$

where, $\beta = \sum_k \left(N_{u,k} \sum_{k'} \mathcal{T}_{k,k'} \right)$ and $C = \prod_{k,k'} \mathcal{T}_{k,k'}^{N_{(u,k),(v,k')}}$ (term independent of $\mathcal{W}_{u,v}$). Therefore, $\mathcal{W}_{u,v}$ is $Gamma(\alpha^*, \beta^*)$ distributed with parameters $\alpha^* = N_{u,v} + \alpha_1$, and $\beta^* = \beta + \beta_1$ given that the prior is $Gamma$.

Topic-Topic Interaction$(\mathcal{T}_{k,k'})$: The conditional probability of interaction between topics k and k' is given as:

$$
P(\mathcal{T}_{k,k'}|\mathcal{W}, \boldsymbol{z}) \propto P(\mathcal{T}_{k,k'}|\alpha_2, \beta_2) \times \left(exp\left(-\beta \mathcal{T}_{k,k'} \right) \mathcal{T}_{k,k'}^{N_{k,k'}} C \right) \qquad (4.6)
$$

where, $\beta = \sum_u \left(N_{u,k} \sum_v \mathcal{W}_{u,v} \right)$ and $C = \prod_{u,v} \mathcal{W}_{u,v}^{N_{(u,k),(v,k')}}$ (term independent of $\mathcal{T}_{k,k'}$). Therefore, $\mathcal{T}_{k,k'}$ is $Gamma(\alpha^\star, \beta^\star)$ distributed with parameters $\alpha^\star = N_{k,k'} + \alpha_2$, and $\beta^\star = \sum_u \left(N_{u,k} \sum_v \mathcal{W}_{u,v} \right) + \beta_2$. Here, $N_{k,k'}$ is the number of times an event with topic k triggered an event with topic k'. The proof for this is completely similar to that of the user-user influence $\mathcal{W}_{u,v}$.

For K topics the additional number of parameters for DNHP as compared to HMHP is $O(K^2)$ (i.e. because of the matrix $\mathcal{T}$), which is a constant for a fixed K. However, the inference of these additional terms has another advantage. Note the interdependence between the user-user influence and the topic-topic interaction in the inference equations - one directly influences the other. Both are conditionally $Gamma$ distributed given the other. In fact, different user-user weights influence each other via topic-topic

weights. This results in a more efficient sharing of evidence across different user-user and topic-topic weights. In contrast, user-user and topic-topic weights in HMHP and user-user weights in the Network Hawkes model are conditionally independent given the event parents.

Base Rate Inference: The estimates for user base rate $\mu_u \; \forall u \in U$ and topic base rates $\mu_k \; \forall k \in K$ are given as:

$$\mu_u = \frac{N_u^{(spon)}}{T \sum\limits_{k \in K} \mu_k} \qquad \mu_k = \frac{N_k^{(spon)}}{T \sum\limits_{u \in V} \mu_u} \tag{4.7}$$

where, $N_u^{(spon)}$ is the number of spontaneous events by user u, $N_k^{(spon)}$ is the number of spontaneous events that have topic k, and T is total time for which the events are observed.

4.4 Experiments and Results

In this section we validate the strengths of DNHP empirically. Here we first describe the models with which we compare the performance of DNHP. We also then describe the datasets and define the tasks to evaluate the performances of the models. All the experiments are performed on a machine with 32 cores, 2.10GHz Intel(R) Xeon(R) E5-2620 CPU, and 256 GB RAM.

4.4.1 Models Evaluated

(1) HMHP: HMHP [17] is a model (described in the previous chapter) for generation of text-based cascades and incorporates user temporal dynamics, user-topic preferences, user-user influence, along with topical interaction in the model. As similar to the other models in the literature, for eg. [46], [84], [59] and [48], in HMHP as well the generation of time stamp of an event is independent of the topic associated with the event. That is the topics here do not influence the time of generation of an event. In HMHP, the timestamps for all the events are first generated based on Multivariate Hawkes Process (MHP) and then the topics are sampled for each event independently of the time. We evaluate and compare the performance of HMHP with DNHP based on the *generalization* and *reconstruction* performances.

(2) NHWKS: Network Hawkes [59] model jointly models event time stamps and the user-user network strengths. This model infers user-user influence and also the parent event for each event. As opposed to HMHP and DNHP, NHWKS does not model text content. Therefore, we evaluate and compare the performance of NHWKS to DNHP with respect to its ability to estimate the $\mathcal{LL}$ of time of the events in the test set.

(3) DNHP: This is our model (described in this chapter) with Gibbs sampling based inference algorithm. In this, the topic-topic graph is considered as a complete graph and the weights over all the edges are considered to have the same *Gamma* prior.

4.4.2 Datasets

We evaluate the performance of the above mentioned models on the following two datasets:

(1) `USPol` (US Politics Twitter Dataset): This dataset is a set of $\sim 370K$ tweets extracted for 151 users who are the US Congress Members[1]. The tweets were extracted in July 2018 using the *Twitter API*, and for each member, the API returned at most 3200 tweets. Each tweet in the dataset consists of a timestamp, the user-id, and the tweet-text(content). The total vocabulary size here is $\sim$ 33000 words after removing the less frequent words. The true user-user influence parameters, the topic associated with each tweet, the parent for each tweet, the topic-topic network, and the topic-topic network parameters are not known to us. Even if the retweet information is available, notice that the retweet would have the same topic as that of the tweet as the content is the same for both, the tweet and the retweet and, retweets form only a small fraction of the parent-child relations that we intend to infer.

(2) Semi-Synthetic Dataset (`SemiSynth`): As for the `USPol` dataset, the gold standard is not available, we generate the `SemiSynth` datasets using the `DNHP` generative model, and we do this for two different user-user graphs.

(a) `SynthUSPol`**:** Here the user-user graph is the same as the `USPol` dataset (i.e. graph with 151 users). The user-user influence and topic-topic interactions matrices are sampled using *Gamma* priors. For each edge $(u, v) \in \mathcal{E}$, $\mathcal{W}_{u,v} \sim Gamma(1.5, 0.2)$. Topic-Topic graph is sampled from $G(K, p)$ with $K = 25$, and $p = 0.3$. For all (k, k'), where $k, k' \in [K]$, $\mathcal{T}_{k,k'} \sim Gamma(2, 0.2)$. For all $u \in U$ and $k \in [K]$, $\mu_u = \mu_k = 0.003$. For all the users and all the topics, the base rate is set to 0.003. Here, $K = 25$, $|\mathcal{V}| = 10000$, and the topic-word distributions are sampled from $Dirichlet(0.01)$. For each event e, the document d_e consists of $Poisson(10)$ words sampled independently. Using this configuration we generate 3 different samples of $\sim 370K$ events each and the generalization performance using this dataset is averaged over these 3 samples (Table 4.3).

(b) `SynthArXiv`**:** Here the user-user graph is the network over top 50 authors in terms of the number of publications from ArXiv high-energy physics data from Stanford's SNAP[2] as described in the HTM paper [48]. The configuration for the generation of the data are similar as for `USPol-Synth` dataset, with an exception of $K = 10$, and $|\mathcal{V}| = 500$, and the topic-word distributions are sampled from $Dirichlet(0.01)$, and for each event e the document d_e consists of $Poisson(10)$ words sampled independently. Using this configuration we generate 5 different samples of $100K$ events each, and

[1] https://twitter.com/verified/lists/us-congress/members?lang=en
[2] snap.stanford.edu/data/cit-HepTh.html

the reconstruction performance using this dataset is averaged over these 5 samples (Tables 4.2, and 4.1).

4.4.3 Evaluation Tasks and Results

In this section we evaluate the models based on the (A) *Reconstruction*, and (B) *Generalization* performances. We now describe each of the tasks formally along with the respective evaluation metrics.

(A) ***Reconstruction Performance***: Considering the abilities of the models we consider three important tasks for the reconstruction, viz., *topic identification*, *parent identification*, and *network reconstruction*. DNHP and HMHP are evaluated on the basis of these tasks performed on the `SynthArXiv` dataset.

For the topic identification task, the evaluation measure used is the Mean Absolute Error (*MAE*) which is defined as follows. For each event $e \in E$, let η_e and $\hat{\eta}_e$ be the original and inferred topics respectively, associated with event e. Let $\pi(\eta_e)$ and $\pi(\hat{\eta}_e)$ be the distributions corresponding to the original topic η_e and inferred topic $\hat{\eta}_e$ respectively. Then, the total absolute error (TAE) is defined as $TAE = \sum_{e \in E} |\pi(\eta_e) - \pi(\hat{\eta}_e)|_1$ and the mean absolute error (MAE) is $MAE = TAE/|E|$, where $|E|$ is the total number of events.

For the parent identification task, the evaluation metrics used are accuracy and recall. Accuracy is defined as the percentage of events for which the correct parent is identified. And, given a ranked list of the predicted parents for each event, recall is calculated by considering the top 1, 3, 5 and 7 predicted parents.

For the network reconstruction task, the performance of the models is evaluated with respect to the reconstruction of the user-user influence matrix. However, notice that in the case of DNHP the influence of an event on another event is modeled as the function of the product of the user-user influence and the topic-topic interaction. And thus, in the case of DNHP, one can only preserve the product of user-user influence and topic-topic interaction but not user-user influence or topic-topic interaction independently. So, to evaluate the performance of the models we first define modified user-user influence as $\mathcal{W}^{mod}_{u,v} = \mathcal{W}_{u,v} \times \sum_{k,k'} \mathcal{T}_{k,k'}$ and then evaluate the performance of the models w.r.t. $\mathcal{W}^{mod}_{u,v}$. The evaluation metrics used are Mean Absolute Error (MAE) and Mean Relative Error (MRE) which are defined as average of $\sum_{u,v} |\mathcal{W}^{mod}_{u,v} - \hat{\mathcal{W}}^{mod}_{u,v}|$ and $\sum_{u,v} \frac{|\mathcal{W}^{mod}_{u,v} - \hat{\mathcal{W}}^{mod}_{u,v}|}{\mathcal{W}^{mod}_{u,v}}$ respectively. Here, $\hat{\mathcal{W}}^{mod}_{u,v}$ is the modified $\mathcal{W}_{u,v}$ value calculated using the estimated $\mathcal{W}_{u,v}$ ($\hat{\mathcal{W}}_{u,v}$) and estimated $\mathcal{T}_{k,k'}$ ($\hat{\mathcal{T}}_{k,k'}$). Also, note that for HMHP model the $\mathcal{W}^{mod}_{u,v} = \mathcal{W}_{u,v} \times K$, where, K is the total number of topics.

(B) ***Generalization Performance***: Here, we compare the performance of the models based on the goodness of fit of each model on the heldout set (`Test` set), i.e. Log-Likelihood ($\mathcal{LL}$) of the heldout set. We perform this task on the semi-synthetic dataset `SynthUSPol` and also on the real dataset `USPol`. For each event e in the `Test` set the

observables are the time t_e, the creator-id c_e, and the words/content w_e. Whereas, the parent z_e and the topic η_e for each event are latent.

Let $\mathcal{X}$ and $\mathcal{Y}$ denote the set of events in the `Train` and `Test` sets respectively. As per DNHP, the total log-likelihood of the test set $\mathcal{Y}$, say $\mathcal{LL}(DNHP)$ is given as:

$$\begin{aligned}\mathcal{LL}(DNHP) &= \log\left(\prod_{e\in\mathcal{Y}} P(t_e, c_e, w_e)\right) = \sum_{e\in\mathcal{Y}} \left(\log P(t_e, c_e, w_e)\right) \\ &= \sum_{e\in\mathcal{Y}} \sum_{\substack{e'\in E \\ c_e\in\mathcal{N}(c_{e'}) \\ t_{e'}<t_e}} \sum_{\eta_{e'}} \sum_{\eta_e} \left(\exp(-\mathcal{W}_{c_{e'},c_e}\mathcal{T}_{\eta_{e'},\eta_e}) \cdot \mathcal{W}_{c_{e'},c_e}\mathcal{T}_{\eta_{e'},\eta_e} \exp(\Delta t_e)\right) \times P(w_e|\eta_e) \\ &\qquad + \sum_{\eta_e} \exp(\mu_{c_e}\mu_{\eta_e}T)\mu_{c_e}\mu_{\eta_e} \times P(w_e|\eta_e) \end{aligned} \tag{4.8}$$

Here, the summations over $e' \in E$, over $\eta_{e'}$, and over η_e are for the marginalization over the candidate set of parents, topic of parent event, and topic of event e respectively.

Similarly, for HMHP it is given as:

$$\begin{aligned}\mathcal{LL}(HMHP) &= \log\left(\prod_{e\in\mathcal{Y}} P(t_e, c_e, w_e)\right) = \sum_{e\in\mathcal{Y}} \log(P(t_e, c_e, w_e)) \\ &= \sum_{e\in\mathcal{Y}} \sum_{\substack{e'\in E c_e\in\mathcal{N}(c_{e'}) \\ t_{e'}<t_e}} \sum_{\eta_{e'}} \sum_{\eta_e} \left(\exp(-\mathcal{W}_{c_{e'},c_e})\mathcal{W}_{c_{e'},c_e} \times \exp(\Delta t_e) \times \mathcal{T}_{\eta_{e'},\eta_e} \times P(w_e|\eta_e)\right) \\ &\qquad + \sum_{\eta_e} \exp(\mu_{c_e}T)\mu_{c_e}\mathcal{U}_{c_e,\eta_e} \times P(w_e|\eta_e) \end{aligned} \tag{4.9}$$

As opposed to DNHP, in HMHP, $\mathcal{T}_{\eta_{e'},\eta_e}$ is probability of transition from topic $\eta_{e'}$ to topic η_e, and thus for a fixed $\eta_{e'}$, we have $\sum_k \mathcal{T}_{\eta_{e'},k} = 1$. HMHP also has users topic preference probability $\mathcal{U}_{c_e,\eta_e}$, and for a fixed user c_e, $\sum_k \mathcal{U}_{c_e,k} = 1$.

In order to understand the complexity of computing this likelihood, let us first consider, for simplicity, that only parent z_e is latent for each event $e \in \mathcal{Y}$. In that case the likelihood function requires marginalization only over the set of candidate parents for each event $e \in \mathcal{Y}$.

$$\begin{aligned}\mathcal{LL}(\mathcal{Y}) &= \sum_{e\in\mathcal{Y}} \sum_{\substack{e'\in E \\ c_e\in\mathcal{N}(c_{e'}) \\ t_{e'}<t_e}} \left(\exp(-\mathcal{W}_{c_{e'},c_e}\mathcal{T}_{\eta_{e'},\eta_e}) \times \mathcal{W}_{c_{e'},c_e}\mathcal{T}_{\eta_{e'},\eta_e} \exp(\Delta t_e)\right) \times P(w_e|\eta_e) \\ &\qquad + \exp(\mu_{c_e}\mu_{\eta_e}T)\mu_{c_e}\mu_{\eta_e} \times P(w_e|\eta_e) \end{aligned} \tag{4.10}$$

Here we are mentioning the likelihood for DNHP only. However, the likelihood in the case of HMHP is of a similar form. Notice that, if a candidate parent $z_e = e'$ for an event $e \in \mathcal{Y}$ is in the training set $\mathcal{X}$, i.e. if $e' \in \mathcal{X}$, then parent for e' is inferred

at the time of the training. Whereas if the candidate parent e' is such that $e' \notin \mathcal{X}$, i.e. $e' \in \mathcal{Y}$, then the parent for e' cannot be inferred. Which means, there is a need to marginalize over candidate set of parents for e' as well. Doing this recursively, to get the exact $\mathcal{LL}$ of event $e \in \mathcal{Y}$, in the worst case, requires summing over all the possible paths to event e. Therefore, calculating $\mathcal{LL}$ for all the events $e \in \mathcal{Y}$ would mean iterating over all the possible set of cascades, which requires $O(n!)$ computation, to the best of our knowledge, where, $|\mathcal{Y}| = n$.

However, if there is a guarantee that for each event $e \in \mathcal{Y}$, each candidate parent event e' is in the train set $\mathcal{X}$, then calculating the likelihood does not require summing over all the possible cascades. For each event, the computation is of the order $O(d)$, where d is the maximum number of candidate parents for any event $e \in \mathcal{Y}$. Then the total computation complexity is $O(dn)$. Bringing in the additional latent variable, i.e. η_e, the topic of event e, the computational complexity is of the order $O(dnK)$, where K is the maximum number of topics. Note that, as the candidate parent e' is in the `Train` set $\mathcal{X}$, the topic $\eta_{e'}$ of the event is inferred at the time of training, thus, summing over $\eta_{e'}$ is not required. Therefore, our goal here is to create a `Test` set such that the parent event for each event is in the `Train` set and for the events in the `Train` set either there are no latent variables, or are inferred.

The semi-synthetic dataset `SynthUSPol` is generated according to DNHP and thus we know the original cascades formed by the events in the data. As the generation process follows a level-wise generation of events (as described in Section 4.2.1), the last level events are the leaf events of the cascades formed. We use this level information from the `SynthUSPol` dataset to classify the events into `Train` and `Test` sets.

For the real dataset `USPol`, the true cascade structure is unknown. Therefore, it is not possible to use the earlier straightforward approach of splitting the dataset into `Train` and `Test` sets, such that the parents of the test events are in train (with already inferred latent variables). Instead, we resort to some heuristic to split the dataset into `Train` and `Test` such that the assumption is likely to hold. Note that as an event e goes farther in time from its parent event e', the probability of generation of event e by the parent event e' decreases exponentially, because of the exponential dependence on time. Keeping this in mind, we split the dataset into `Train` and `Test` such that the events in the `Test` set are not that far from their some potential parent events in the `Train` set. To create the test set, events are processed sequentially. Each event $e \in E$ is added to the `Test` set $\mathcal{Y}$ if and only if at most p_{test} fraction of its candidate parents are already in the `Test` set $\mathcal{Y}$. This ensures that $1 - p_{test}$ fraction of its candidate parents are still in the `Train` set $\mathcal{X}$. Note that increasing (decreasing) p_{test} results in increasing (decreasing) the test set size as well as decreasing (increasing) the train set size. To study the effects of increasing training data size without reducing the test size, we use an additional parameter $0 \leq p_{data} \leq 1$ to decide whether to include an event in our experiments at all. Specifically, we first randomly include each event in the dataset with probability p_{data}, and then the `Train` and `Test` split is performed.

So now, given the above mentioned construction of the `Train`-`Test` split, the $\mathcal{LL}(DNHP)$ and $\mathcal{LL}(HMHP)$ is equivalent to that in Equations 4.8 and 4.9 respectively, without

the summation over candidate parent's topic $\eta_{e'}$, which is as follows.

$$\begin{aligned}\mathcal{LL}(\mathcal{DNHP}) &= \sum_{e\in\mathcal{Y}} \sum_{\substack{e'\in E \\ c_e\in\mathcal{N}(c_{e'}) \\ t_{e'}<t_e}} \sum_{\eta_e} \left(\exp(-\mathcal{W}_{c_{e'},c_e}\mathcal{T}_{\eta_{e'},\eta_e})\times \mathcal{W}_{c_{e'},c_e}\mathcal{T}_{\eta_{e'},\eta_e}\exp(\Delta t_e)\times P(w_e|\eta_e)\right) \\ &\quad + \sum_{\eta_e} \exp(\mu_{c_e}\mu_{\eta_e}T)\mu_{c_e}\mu_{\eta_e}\times P(w_e|\eta_e) \end{aligned} \tag{4.11}$$

$$\begin{aligned}\mathcal{LL}(\mathcal{HMHP}) &= \sum_{e\in\mathcal{Y}} \sum_{\substack{e'\in E \\ c_e\in\mathcal{N}(c_{e'}) \\ t_{e'}<t_e}} \sum_{\eta_e} \left(\exp(-\mathcal{W}_{c_{e'},c_e})\mathcal{W}_{c_{e'},c_e}\times\exp(\Delta t_e)\mathcal{T}_{\eta_{e'},\eta_e}\times P(w_e|\eta_e)\right) \\ &\quad + \sum_{\eta_e} \exp(\mu_{c_e}T)\mu_{c_e}\mathcal{U}_{c_e,\eta_e}\times P(w_e|\eta_e) \end{aligned} \tag{4.12}$$

Also, for both the models, the sum over a candidate set of parents for each event is restricted to the events that are not farther in time and are in the `Train` set $\mathcal{X}$.

4.4.4 Results

Reconstruction Performance:

This exercise is mostly to verify our Gibbs sampling based inference algorithm. And the tasks here are addressed on the `SynthArXiv` dataset. Each result presented here is an average over 5 samples of the generated dataset.

Parent Identification: Table 4.1 presents the results for this task. For both the models, the recall improves significantly as we consider more candidates predicted parents. The accuracy for the `DNHP` is $\sim$ 60% better than that of the HMHP model. Even for recall @ 3, `DNHP` performs at least $\sim$ 50% better than that of the HMHP model.

Network Reconstruction: Table 4.1 highlights the results for this task. When all the edges are considered the *MAE* and *MRE* for `DNHP` is $\sim$ 59% and $\sim$ 40% respectively lesser than the HMHP model. However, on considering the edges with more than 100 transactions, the error in the case of `DNHP` improves further as compared to HMHP.

Topic Identification: The results for this task are displayed in Table 4.2. Both the models perform almost equally. The reason for this could be that there are only 10 topics in the dataset, and because of the way the parameters are set, the topics are likely near-orthogonal and hence the topic identification task gets easier.

In summary, `DNHP` outperforms the HMHP model with respect to the reconstruction performance for the synthetic data.

Parent Identification		
	DNHP	**HMHP**
Accuracy	**0.45**	0.28
Recall@1	**0.52**	0.29
Recall@3	**0.73**	0.46
Recall@5	**0.81**	0.56
Recall@7	**0.85**	0.63
Network Reconstruction		
	DNHP	**HMHP**
MAE	**0.95**	2.37
MAE ($N_{uv} > 100$)	**1.13**	3.17
MRE	**0.39**	0.69
MRE ($N_{uv} > 100$)	**0.28**	0.66

Table 4.1: Performance for Parent Identification and Network Reconstruction on Semi-Synthetic Dataset

MAE	**DNHP**	**HMHP**
AVG(SD)	0.0074(0.013)	0.0074(0.011)

Table 4.2: Text modeling performance on Semi-Synthetic Dataset

Generalization Performance

The generalization performance for the models is evaluated on the basis of their ability to estimate the heldout $\mathcal{LL}$. This task is addressed on both semi-synthetic (`SynthUSPol`) and the real (`USPol`) dataset.

`SynthUSPol` **Dataset:** Table 4.3 presents the heldout $\mathcal{LL}$ of time and content by the models `DNHP` and HMHP, and $\mathcal{LL}$ of time by the models `DNHP` and `NHWKS` for the `SynthUSPol` dataset. The results are averaged over 3 independent samples each of size $\sim 370K$ (which is roughly the size of the real dataset `USPol`). The size of the `Train` set upto 3^{rd} last level and upto 2^{nd} last level is $\sim 170K$ and $\sim 340K$ respectively. The first row indicates the $\mathcal{LL}$ when 2^{nd} last level events are used as a `Test` set and all the events above that level are in `Train` set. Similarly, the second row highlights the results when the events at the last level (truncated level) are used as `Test` set. Observe that as more training is provided to both the models the $\mathcal{LL}$ estimation gets better. Overall, `DNHP` outperforms `NHWKS` in explaining the timestamps by benefiting from estimating topic-topic parameters given the text, and in turn, using those to better estimate the user-user parameters. In the same way, by better estimating both of these parameters using coupled updates, `DNHP` outperforms HMHP in explaining

Average Log-Likelihood of Time & Content		
Test On	**DNHP**	**HMHP**
2^{nd} Last Level	**-58.66**	-59.31
Last Level	**-57.6**	-58.48
Average Log-Likelihood of Time		
Test On	**DNHP**	**NHWKS**
2^{nd} Last Level	**-2.56**	-2.76
Last Level	**-2.32**	-2.48

Table 4.3: Average $\mathcal{LL}$ for Semi-Synthetic data

the time stamps and the textual content together.

`USPol` **Dataset:** Tables 4.4 and 4.5 presents the $\mathcal{LL}$ of time and content for `DNHP` and HMHP and $\mathcal{LL}$ of time for `DNHP` and `NHWKS` on the `USPol` dataset with $K = 25$ and $K = 100$. The rows indicate the sizes of the `Train` and `Test` when the events are selected in the dataset with probability p_{data} set as 0.5, 0.7, and 1.0 (which is the complete dataset). Then the `Train-Test` split is performed with p_{test} set as 0.3 *and* 0.5, which indicates the maximum fraction of candidate parents for each event in the `Test` set.

Observe that as expected all the models, `DNHP`, HMHP, and `NHWKS`, get better at estimating the $\mathcal{LL}$ with the increase in the size of the dataset across different values of p_{test}. Similarly, as with the synthetic data experiments, `DNHP` outperforms `NHWKS` in fitting the timestamps and HMHP in fitting the time stamps and the textual content. However, in the synthetic data experiments, the data had been generated using `DNHP` and therefore the results were perhaps to be expected. Similar trends on real data confirm that `DNHP` indeed models real cascades better than the two baseline models.

Equally significantly, the gap between `DNHP` and the two baselines HMHP and `NHWKS` is larger when dataset size (and as a result, the training size) is smaller. This agrees with our understanding of parameter sharing leading to better generalization given limited volumes of training data. This demonstrates that `DNHP` has already learned the parameters efficiently with the smaller dataset size, using the flow of evidence between the parameters in the update equations.

4.5 Related Work

Recently, there has been a spate of research work in inferring the information diffusion network. The network reconstruction task can be based on just the event times ([37], [38], [92], [36], [89], [59]), where the content of the events is not considered. Dirichlet Hawkes Process (DHP) [28] is one of the models that use the content and

p = 0.3				
	#Topics = 25		#Topics = 100	
Train(Test)	DNHP	HMHP	DNHP	HMHP
$114K(70K)$	**-80.51**	-95.71	**-81.91**	-96.51
$177K(100K)$	**-78.09**	-86.66	**-79.22**	-87.32
$240K(130K)$	**-76.57**	-80.14	**-77.34**	-80.71
p = 0.5				
	#Topics = 25		#Topics = 100	
Train(Test)	DNHP	HMHP	DNHP	HMHP
$86K(98K)$	**-81.39**	-95.42	**-83.12**	-96.15
$133K(144K)$	**-79.05**	-87.01	**-70.49**	-87.78
$179K(190K)$	**-77.44**	-81.30	**-78.57**	-81.96

Table 4.4: Average Log-Likelihood for Time + Content for Real dataset USPol. (The Train(Test) sizes mentioned are approximate.)

p = 0.3				
	#Topics = 25		#Topics = 100	
Train(Test)	DNHP	NHWKS	DNHP	NHWKS
$114K(70K)$	**-8.74**	-24.46	**-8.04**	-24.46
$177K(100K)$	**-7.45**	-16.32	**-7.09**	-16.32
$240K(130K)$	**-6.56**	-10.27	**-6.37**	-10.27
p = 0.5				
	#Topics = 25		#Topics = 100	
Train(Test)	DNHP	NHWKS	DNHP	NHWKS
$86K(98K)$	**-8.97**	-23.57	**-8.08**	-23.57
$133K(144K)$	**-7.74**	-16.02	**-7.23**	-16.02
$179K(190K)$	**-6.81**	-10.90	**-6.55**	-10.90

Table 4.5: Average Log-Likelihood for Time for Real dataset USPol. (The Train(Test) sizes mentioned are approximate)

time information, but the tasks performed are not related to network inference or cascade reconstruction. As similar to our model the DHP as well is a mixture model and assigns a single topic to each event, but it does not have any notion of parent event topical interactions. The recent models such as HTM [48], and HMHP [17] show that using the content information can be profitable and can give better estimates for the network inference tasks as well the cascade inference task. HMHP model is the closest model to our model, which considers topical interactions as well. However, both HMHP and HTM [48] consider the topics as marks that are generated independently and are not generated along with the time.

In addition to the above mentioned models various generalizations of Dirichlet Process such as Hierarchical Dirichlet process (HDP) [82] and variants of Chinese Restaurant Process (CRP) such as Nested CRP (nCRP) [87], the Chinese Restaurant Franchise Process (CRFP) [64] in combination with Hawkes processes have also been explored in literature. Seonwoo et al. [82] proposed Hierarchical Dirichlet Gaussian Marked Hawkes process (HD-GMHP) that reconstructs narratives and thread structure of discussions from the text data. HD-GMHP uses Hawkes process to model the temporal information of events, the textual information is modeled using Gaussian distribution (as similar to that in HTM), and the trigerring relationships are modeled using mixture of Gaussians.

In [87] the authors proposed a model named Nested Chinese Restaurant Process with Hawkes Gaussian process (nCRP-HGP) which combines the advantages of hierarchical nonparameteric Bayesian models (nCRP) with Hawkes processes to model the temporal and textual information of the events. nCRP-HGP incorporates the events in which multiple senders and receivers are involved, and exploits the properties of the senders and receivers for better modeling of the temporal information and textual content.

Mavraforakis et al. [64] developed Hierarchical Dirichlet Hawkes Processes (HDHP) model by replacing Chinese Restaurant Process (CRP) with a Chinese Restaurant Franchise Process (CRFP) which is capable of modeling streaming data from multiple users. The model uncovers the various latent learning patterns from the series of actions that users take on platforms like Stack Overflow and Wikipedia while gathering some new information on some topic.

All these models ([82, 64, 87]) are powerful in terms of modeling the textual content, but, either, as similar to DHP, does not take into account the notion of users or network over users, or, as in HTM, forces parent-child text content to be topically close, which does not help in capturing topical relationships between parent-child topics. Also, these models do not consider the interaction between the the user-user and the topic-topic networks, which play an important role in better modeling of events.

On the other hand, there have also been efforts in using Recurrent Neural Networks (RNN) to model the intensity of point processes [29, 91, 65].

4.6 Conclusions

In this chapter, we present a new Dual-Network Hawkes process that treats the users and topics symmetrically in modeling the event generation. In the future, we wish to incorporate more dimensions and study the effect of the inter-dimensional flow of evidence in handling data sparsity. A concrete outcome will be to incorporate structured prior over a latent topic graph or, in general, a structure over marks, and improve the existing knowledge-base (e.g. DBpedia) from this cascade evidence.

PART - II
Parameterized Complexity of Information Diffusion

Chapter 5

Parameterized Complexity of Information Diffusion[1]

5.1 Introduction

In this chapter, we try to understand the diffusion of information in social networks from a perspective of algorithms, and specifically, from the perspective of *parameterized algorithms*. However, before going into the details of the *parameterized algorithms* let us first give an overview of the algorithmic problem that we try to solve here. The diffusion of information problem from the algorithmic perspective has been studied before. For example, consider the problem of selecting the most influential nodes in the network. Here, the aim is to find the initial seed nodes or "influential members" of the network whose adoption of an idea or the product would trigger a large cascade of adoption of idea or product and maximize the adoption in the network. In the past decade or two, there has been a spate of work to understand the effects of diffusion processes like "word-of-mouth" and "viral marketing" in the success of new products or ideas or innovation [23, 51, 56, 55, 14, 10]. The two most prevalent models that are considered to formalize the diffusion processes like "word-of-mouth" and "viral marketing" are the Independent Cascade (IC) model and the Linear Threshold (LT) model. In the IC model, given that a node u has adopted an idea (or activated), each of its neighbor v adopts the same idea (or gets activated) with some probability p_{uv}. And, in case of the LT model there is a weight associated w_{uv} with each pair of neighbors (u, v), such that $\forall v \sum_u w_{u,v} \leq 1$. For each node v, there is threshold $\theta_v \in [0, 1]$ associated with it. A node v adopts an idea or gets activated only if $\sum_{\text{active } u} w_{uv} \geq \theta_v$. With these diffusion models, there has been significant research in providing faster algorithms and algorithms with provable guarantees for the influence maximization problem.

However, in this chapter, our goal is somewhat opposite. Note that it's not always the case that we want the information to spread in the network. Consider, for example,

[1]All the results in this chapter are based on the publication [18] at ICALP-2017

an epidemic or an infection, or some rumor or fake news floating in the network. In all of these cases, the goal is to curb the spread. Thus, in contrast to all the previous discussions or the generative models where we tried to understand the factors responsible for the spread of information in networks, here we are interested in designing algorithms that deal with the suppression of information. As similar to the work on finding the "influential members", in parallel there has been significant research in the development of algorithms for the faster detection of an outbreak or to minimize the population affected by the infection [56, 10]. In this work, we focus on the Firefighting problem ([11, 6, 34, 32, 20, 45, 31]) where the aim is to design an algorithm to place firefighters (inoculation) on the nodes in the network such that particular set of nodes in the graph are saved from an infection or are protected from fire. Let us describe the Firefighting problem formally.

The Firefighting problem formalizes the question of designing inoculation strategies against a contagion that is spreading through a given network. The goal is to come up with a strategy for placing firefighters on nodes in order to intercept the spread of fire. More precisely, firefighting can be thought of like a turn-based game between two players, traditionally the fire and the firefighter, played on a graph G with a source vertex s. The game proceeds as follows:

- At time step 0, fire breaks out at the vertex s. A vertex on fire is said to be *burned*.
- At every odd time step $i \in \{1, 3, 5, \ldots\}$, when it is the turn of the firefighter, a firefighter is placed at a vertex v that is not already on fire. Such a vertex is permanently *protected*.
- At every even time step $j \in \{2, 4, 6, \ldots\}$, the fire spreads in the natural way: every vertex adjacent to a vertex on fire is burned (unless it was protected).

The game stops when the fire cannot spread any more. A vertex is said to be *saved* if there is a protected vertex on every path from s to v.

This said it is not hard to notice that the fire in the firefighting problem played on the graph is analogous to an infection or an epidemic or a rumor in a network of users. Understanding this parallel between the infection and the fire, hereafter, we would use the terms contagion, infection, fire, epidemic, or rumor interchangeably. Also, we would like to bring to the notice of the reader that, in case of the Firefighting problem explained above, the model for the spread of the fire from a node to its neighbor is simpler as compared to the IC and LT models. I.e. if a node is burnt, the fire spreads to all its neighbor deterministically, unless protected by a firefighter; which is not the case for the IC and the LT models. Needless to say, the complexity of the algorithm depends on the model of the spread of the fire or an infection chosen, in this work, we focus on the simpler spreading model first.

However, irrespective of what the spreading model is, the natural algorithmic question associated with the firefighting problem is to find a strategy that optimizes some

desirable criteria, for instance, maximizing the number of saved vertices [11], minimizing the number of rounds, the number of firefighters per round [12], or the number of burned vertices [31, 11], and so on. These questions are well-studied in the literature, and while most variants are NP-hard, approximation and parameterized algorithms have been proposed for various scenarios. An excellent survey by Finbow and MacGillivray [32] on the Firefighting problem highlights various results and questions related to the Firefighting problem.

As mentioned above, we are addressing the Firefighting problem from the perspective of *parameterized algorithms*. We now give an overview of *parameterized algorithms*.

5.2 Overview of Parameterized Algorithms

Let us try to understand more about *parameterized algorithms* with the help of a very well known Vertex Cover problem. Recall that for an undirected graph G with the vertex set V and the set of edges E, such that $|V| = n$ and $|E| = m$, a vertex cover is a subset S of vertices (say $|S| = k$) such that for every edge (u, v) in the graph, either u or v is present in the set S. And the Vertex Cover problem is to find the minimum size vertex cover. Unfortunately, the Vertex Cover problem is a classical NP-Complete problem. One of the algorithms to solve this problem is by trying all the possibilities i.e. all the possible subsets, which is $O(2^n)$. Luckily, if there is a budget on the size of the vertex cover i.e. $|S| \leq k$ is not large, say $k \leq 10$, then the time complexity of the naive algorithm would be $O(n^k)$, which is a polynomial in the input size. However, there is a smarter algorithm that uses *bounded search tree technique*, which has a time complexity of $O(2^k(n + m))$. This algorithm works based on the following simple observation, at least one of the two nodes of an edge must be selected. Thus, as long as there is an edge, say (u, v), which is not covered, the algorithm proceeds as follows. Move u to the set of accepted nodes, remove the edges covered by u and then run the algorithm recursively to check if the remaining edges can be covered by at most $k - 1$ nodes. If this succeeds we have a solution. If it fails, move v to the set of accepted nodes, remove the edges covered by v and then run the algorithm recursively to check if the remaining edges can be covered by at most $k - 1$ nodes. Therefore, at each step, the algorithm checks if there are uncovered edges, if so, it makes two recursive calls. Note that, as the depth of this recursive tree can be at most k, there are at most 2^k recursive calls. Also, each call can be implemented in linear time $O(n + m)$.

Observe that, if we consider k as a constant, both the algorithms are polynomial time. But notice that, for a fixed value of k, *bounded search tree* based algorithm is significantly better than the naive algorithm in terms of the running time complexity. In the case of the naive algorithm, for a fixed value of k, the exponent of the polynomial depends on the value of k, whereas, for a fixed value of k, the *bounded search tree* based algorithm is a linear time algorithm. In $O(2^k(n + m))$ algorithm, the running time is exponential in k, but depends only linearly on the input size $(n + m)$. I.e. here the combinatorial explosion is limited to the parameter k of the problem. And this is what

the final objective of parameterized algorithms is – to design algorithms that have the running time complexity of the form $f(k) \cdot n^c$, for a constant c that is independent of both n and k. Such algorithms, with running time $f(k) \cdot n^c$, for a parameter k and a constant c that is independent of both n and k are termed as *fixed-parameter tractable* algorithms or FPT algorithms. And typically the goal here is to make both, the factor $f(k)$, and the constant c as small as possible. In parameterized algorithms, the term "parameterized" is attributed to the *relevant secondary measurement* k that captures some aspect of the input instance. The aspect could be related to the size of the solution, or it could be a number that describes how "structured" the input instance is.

Note that, there can be multiple choices of parameters for the same problem. Whether the problem has a FPT algorithm depends a lot on the choice of the parameter. In Section 5.6 we will present a negative example where a particular choice of parameter does not give a FPT algorithm. We would now move towards laying foundations for *parameterized complexity* that helps us to understand the influence of different parameters on the complexity of the problem. We start with the formal definition of what a parameterized problem is and also explain the complexity classes for the parameterized problems.

Definition 22 (Parameterized problem [19]): *A* parameterized problem *is a language* $L \subseteq \Sigma^* \times \mathbb{N}$*, where* Σ *is a fixed, finite alphabet. For an instance* $(x, k) \in \Sigma^* \times \mathbb{N}$*,* k *is called the parameter.*

Let us go back to the Vertex Cover problem and try to see it from the lens of parameterized problem. An instance of the Vertex Cover problem parameterized by the size of the solution is a pair (G, k), where G is an undirected graph encoded as a string over Σ, and k is an integer denoting the size of the vertex cover. The pair (G, k) belongs to the Vertex Cover parameterized language if and only if G correctly encodes the undirected graph and G contains a vertex cover of size k. There is a bit abuse of notation here, we are using G to denote both the graph as well as the encoded string. The size of an instance of a parameterized problem (x, k) is given as $|x| + k$. Here, the convention is that the parameter k is encoded in unary. Given the formal definition of a parameterized problem, it's now time to formally define the *fixed-parameter tractable* problem.

Definition 23 (Fixed-Parameter Tractable (FPT) [19]): *A* parameterized problem $L \subseteq \Sigma^* \times \mathbb{N}$ *is called fixed-parameter tractable (*FPT*) if there exists an algorithm* $\mathcal{A}$ *(called a fixed-parameter algorithm), a computable function* $f : \mathbb{N} \to \mathbb{N}$*, and a constant* c *such that, given* $(x, k) \subseteq \Sigma^* \times \mathbb{N}$*, the algorithm* $\mathcal{A}$ *correctly decides whether* $(x, k) \in L$ *in time bounded by* $f(k) \cdot |(x, k)|^c$*. The complexity class containing all fixed-parameter tractable problems is called* FPT*. And the time complexity of the form* $f(k) \cdot |(x, k)|^c$ *is also known as fpt-time.*

Recall that, for the Vertex Cover problem we also mentioned a naive algorithm with the time complexity $O(n^k)$. Such algorithms with the time complexity of the form

$f(k) \cdot n^{g(k)}$ are termed as *slice-wise polynomial (XP)* algorithms. Let us define this formally for completeness.

Definition 24 (Slice-wise Polynomial (XP) [19]): *A* parameterized problem $L \subseteq \Sigma^* \times \mathbb{N}$ *is called slice-wise polynomial (XP) if there exists an algorithm* $\mathcal{A}$ *and two computable functions* $f, g : \mathbb{N} \to \mathbb{N}$*, such that, given* $(x, k) \subseteq \Sigma^* \times \mathbb{N}$*, the algorithm* $\mathcal{A}$ *correctly decides whether* $(x, k) \in L$ *in time bounded by* $f(k) \cdot |(x, k)|^{g(k)}$*. The complexity class containing all slice-wise polynomial problems is called XP.*

Also, we would like to add a point that the parameter k in the definitions of *parameterized problem*, FPT, and XP can be generalized to a vector of d parameters (d non-negative integers) for some fixed constant d. And accordingly, the functions f and g in the definitions of FPT and XP would depend on all these parameters.

Observe that, given a parameterized problem L, there are two different optimization problems that a designer has to work on essentially while designing FPT algorithms. As the complexity should be of the form $f(k) \cdot n^c$, the designer can:

1. aim to design an algorithm where the constant c is as small as possible. I.e. a designer can try to optimize the *polynomial factor* in the running time; or
2. aim to design an algorithm such that the function f which is dependent on the parameter k grows as slowly as possible. I.e. a designer can try to optimize the *parametric dependence* of the running time.

Given this understanding of FPT algorithms, we highlight a couple of definitions that show an analogy between parameterized complexity and the classical complexity classes.

Definition 25 (para-K (Definition 3 [33])): *Let* K *be a classical complexity class. Then, the parameterized complexity class para-*K *is defined as the class of all parameterized problems* $L \subseteq \Sigma^* \times \mathbb{N}$ *for which there exist a computable function* $f : \mathbb{N} \to \Sigma^*$ *and a problem* $L' \in K$ *such that for all instances* $(x, k) \in \Sigma^* \times \mathbb{N}$ *of* L *we have that* $(x, k) \in L$ *if and only if* $(x, f(k)) \in L'$ *(consider that* $(x, f(k))$ *is encoded in* Σ^**).*

Intuitively, para-K consists of all problems that are in K after a pre-computation that involves only the parameter [33]. One of the common examples is the complexity class para-NP. This class consists of all the parameterized problems that are solvable by a non-deterministic algorithm in FPT-time [22].

Definition 26 (FPT-reduction): *Let* $P \subseteq \Sigma^* \times \mathbb{N}$ *and* $P' \subseteq \Sigma'^* \times \mathbb{N}$ *be parameterized problems. A para-PTIME-reduction, or* FPT*-reduction, from* P *to* P' *is a mapping* $R : \Sigma^* \times \mathbb{N} \to \Sigma'^* \times \mathbb{N}$ *such that:*

1. *For all* $(x, k) \in \Sigma^* \times \mathbb{N}$ *we have* $(x, k) \in P \iff R(x, k) \in P'$.

2. *There exists a computable function* $g : \mathbb{N} \longrightarrow \mathbb{N}$ *such that for all* $(x, k) \in \Sigma^* \times \mathbb{N}$, *say with* $R(x, k) = (x', k')$, *we have* $k' \leq g(k)$.

3. *There exists a computable function* $f : \mathbb{N} \longrightarrow \mathbb{N}$ *and a constant* $c \in \mathbb{N}$ *such that* R *is computable in time* $f(k) \cdot n^c$.

If there is an FPT-reduction from P to P' then it is written $P \leq_{FPT} P'$

Again, as similar in the classical complexity theory, a natural question to ask is what are the complete problems for classes para-K? One of the guesses could be to apply some suitable parameterization to the complete problems for K would be complete for para-K. However, it turns out that, essentially there is no need for any parameterization – i.e. the trivial parameterization of a complete problem for K is complete for para-K. Therefore, one can say that the parameterized complexity classes para-K have complete problems that are essentially unparameterized [33].

This argument has been formalized in the following proposition given by Flum and Grohe [33].

Lemma 27. *(Proposition 14 [33]) Let* K *be a robust complexity class, and let* X *be complete for* K *under PTIME-reductions. Then* $X \times \{0\}$ *(*trivial parameterization of X*) is complete for para-*K *under* FPT*-reductions.*

An example of a para-NP-complete problem is SAT(constant) which is a parameterized version of the SAT problem, where the parameter is a fixed constant, i.e. $SAT(constant) = \{(x, 1) : x \in SAT\}$. It turns out that, for any classical complexity class K and the K-complete problem X, the problem $X(constant) = \{(x, 1) : x \in X\}$ is para-K complete [22].

Let us now turn to the definition of parameterized firefighting problem along with our proposed FPT algorithm.

Saving a Critical Set. The game proceeds as described earlier: we are given a graph G with a vertex $s \in V(G)$. To begin with, the fire breaks out at s and vertex s is burning. At each step $t \geq 1$, first the firefighter protects one vertex not yet on fire - this vertex remains permanently protected - and the fire then spreads from burning vertices to all unprotected neighbors of these vertices. The process stops when the fire cannot spread any more. In the definitions that follow, we formally define the notion of a firefighting strategy.

Definition 28 (Firefighting Strategy): *A* k*-step firefighting strategy is defined as a function* $\mathfrak{h} : [k] \longrightarrow V(G)$. *Such a strategy is said to be valid in* G *with respect to* s *if, for all* $i \in [k]$, *when the fire breaks out in* s *and firefighters are placed according to* $\mathfrak{h}$ *for all time steps up to* $2(i-1)-1$, *the vertex* $\mathfrak{h}(i)$ *is not burning at time step* $2i-1$, *and the fire cannot spread anymore after time-step* $2k$. *If* G *and* s *are clear from the context, we simply say that* $\mathfrak{h}$ *is a valid strategy.*

Definition 29 (Saving C): *For a vertex s and a subset $C \subseteq V(G)\setminus\{s\}$, a k-step firefighting strategy $\mathfrak{h}$ is said to save C if $\mathfrak{h}$ is a valid strategy and $\cup_{i=1}^{k}\mathfrak{h}(i)$ is a $\{s\}$-C separator in G, in other words, there is no path from s to any vertex in C if firefighters are placed according to $\mathfrak{h}$.*

We are now ready to define the parameterized problem that is the focus of this work.

Saving A Critical Set (SACS)	Parameter: k
Input:	An undirected n-vertex graph G, a vertex s, a subset $C \subseteq V(G)\setminus\{s\}$, and an integer k.
Question:	Is there a valid k-step strategy that saves C when a fire breaks out at s?

As a starting point, we first show that the SACS problem is NP-complete even when the critical set has only one vertex.

5.2.1 SACS is para-NP-complete

Theorem 30 ([Choudhari et al. [18]). *] SACS is* NP-complete *even when the critical set has one vertex.*

Proof. Let (G, k) be an instance of k-*CLIQUE* problem. We construct a graph G' as follows. For each edge $(u, v) \in E(G)$ create a node s_{uv}. We denote this set of nodes by E. For each node $v \in V(G)$ create a node $v \in G'$ and denote this set of nodes by V. Add two nodes s and t, where s is the node on which the fire starts and $t \in C$ (CRITICAL SET), the node to be saved. Connect t to all the nodes in set E. Connect s to each node $v \in V$ by a path of length k. Let us refer to these nodes, which are on the paths from s to V, as V_{-1}. Create $(m - \binom{k}{2} - 1)$ copies of set V. Denote these copies as $V_1, V_2, ..., V_{m-\binom{k}{2}-1}$. Let $V_m = \cup_{i=1}^{m-\binom{k}{2}-1} V_i$ and V be denoted by V_0. Add an edge $(v_{i,j}, v_{i,j+1})$ for $v_i \in V_j$ and $v_i \in V_{j+1}$ for all $0 \leq j < (m - \binom{k}{2} - 1)$. For each edge $e = (u, v)$ in $E(G)$ add an edge $(u, s_{u,v})$ and $(v, s_{u,v})$ where $u, v \in V_{m-\binom{k}{2}-1}$ and $s_{u,v} \in E$. Now set $k' = k + m - \binom{k}{2}$.

Lemma 31. *At most k firefighters can be placed on the nodes in set $(V_{-1} \cup V_0)$ in G'.*

Proof. As per the construction of G', each node $v \in V_0$ is connected to s with a path of length k and each node $u \in V_{-1}$ is at a distance of length less than k. Thus, there are at most k time steps at which firefighters can be employed on the nodes in $V_{-1} \cup V_0$. □

Lemma 32. *Protecting less than k nodes from the set $H_k = (V_{-1} \cup V_0 \cup V_m)$ in G' is not a successful strategy to save t.*

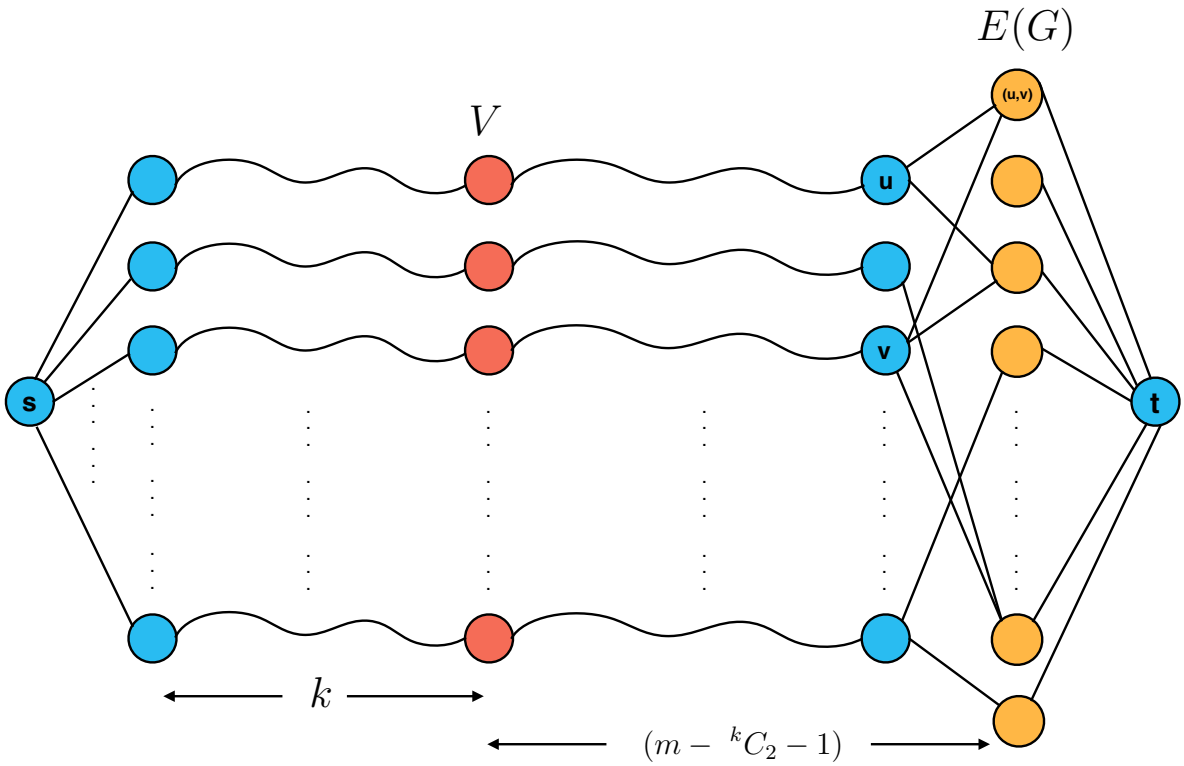


Figure 5-1: Saving a critical set is NP-complete

Proof. To save t, all the nodes in E ($|E| = m$) must either be protected or saved. Suppose we protect $\gamma < k$ nodes in set H_k. Observe that, protecting γ nodes in H_k can save at most $\binom{\gamma}{2}$ nodes in E. Therefore, in order to save t we need to protect/save remaining $m - \binom{\gamma}{2}$ nodes in E. Given the constraint on the budget as well as the number of time instances we are left with, we can protect $m - \binom{k}{2} + (k - \gamma)$ more nodes to save t. If $\alpha = m - \binom{k}{2} + (k - \gamma)$ and $\beta = m - \binom{\gamma}{2}$ then, the claim is $\beta - \alpha > 0$.

$$
\begin{aligned}
\beta - \alpha &= m - \binom{\gamma}{2} - \left(m - \binom{k}{2} + (k - \gamma) \right) \\
&= \binom{k}{2} - \binom{\gamma}{2} + (\gamma - k) \\
&= \frac{k(k-1)}{2} - \frac{\gamma(\gamma-1)}{2} + (\gamma - k) \\
&= \frac{k^2 - k - \gamma^2 + \gamma}{2} + (\gamma - k) \\
&= \frac{(k+\gamma)(k-\gamma) - (k-\gamma)}{2} + (\gamma - k) \\
&= (k - \gamma)\left[\frac{k+l}{2} - \frac{1}{2} - 1 \right] > 0
\end{aligned}
\tag{5.1}
$$

Therefore, it is not possible to save t if we protect $\gamma < k$ nodes in H_k, as this requires $m - \binom{\gamma}{2}$ more nodes to be protected, which is not feasible. □

We claim that SAVING A CRITICAL SET with $|C| = 1$ on $(G', k', s, C = t)$ is a YES-instance if and only if $k - CLIQUE$ is an YES-instance in (G, k).

Suppose G has a k-CLIQUE denoted as K. Then the firefighting strategy is to protect the vertices $v_i \in V_0$ corresponding to the vertices $v_i \in K$. Protecting these k vertices guarantees to save $\binom{k}{2}$ vertices in the set E. Also, from Lemma 31 it follows that protecting these k vertices is valid with respect to time constraints. The remaining $m - \binom{k}{2}$ in E can be protected at each allowed time step after placing the k firefighters in V_0. This as well is valid with respect to the time constraints as the set of nodes E are at a distance $m - \binom{k}{2}$ from the set of nodes V_0.

Suppose that G' has a valid firefighting strategy $S = \{u_1, u_2, \ldots, u_{k+m-\binom{k}{2}}\}$ with $k + m - \binom{k}{2}$ firefighters. If a firefighter is placed on any node $u \in V_{-1}$ or on any node $w \in V_m$, then it is equivalent to placing a firefighter on the node $v_i \in V_0$ to which the nodes u and w have a path. Therefore, in the firefighting strategy S if there is a firefighter on node $u \in V_{-1}$ then it is pushed to $v_i \in V_0$ such that $u - v_i$ is a path, and if there is a firefighter on node $w \in V_m$, then it is pulled back to $v_j \in V_0$ such that $w - v_j$ is a path. Also, it follows from Lemma 32 that for a successful firefighting strategy, k firefighters must be placed in $H_k = (V_{-1} \cup V_0 \cup V_m)$. Therefore, these k nodes in V_0 on which the firefighters are placed (by pushing from $u \in V_{-1}$ or pulling from $w \in V_m$), must form a clique in G with the $\binom{k}{2}$ edges corresponding to the nodes saved in the set E. If these k nodes do not form a clique, then with the remaining $m - \binom{k}{2}$ nodes it won't be possible to save/cover all the vertices in E. □

Thus, Theorem 30 and the Lemma 27 implies that *SACS is para*-NP-complete *when parameterized by the size of the critical set*.

Before going to the FPT algorithm we highlight some terms and definitions in the next section. These terms and definitions along with some lemmas lay foundations for the final FPT algorithm for the problem.

5.3 Preliminaries

Most of our notation is standard. We use $[k]$ to denote the set $\{1, 2, \ldots, k\}$, and we use $[k]_O$ and $[k]_{\mathcal{E}}$, respectively, to denote the odd and even numbers in the set $[k]$.

Definition 33 (Reachable Sets): *Let $G = (V, E)$ be an undirected graph, let $X \subseteq V$ and $S \subseteq V \setminus X$. We denote by $R_G(X, S)$ the set of vertices of G reachable from X in $G \setminus S$ and by $NR_G(X, S)$ the set of vertices of G not reachable from X in $G \setminus S$. We drop the subscript G if it is clear from the context.*

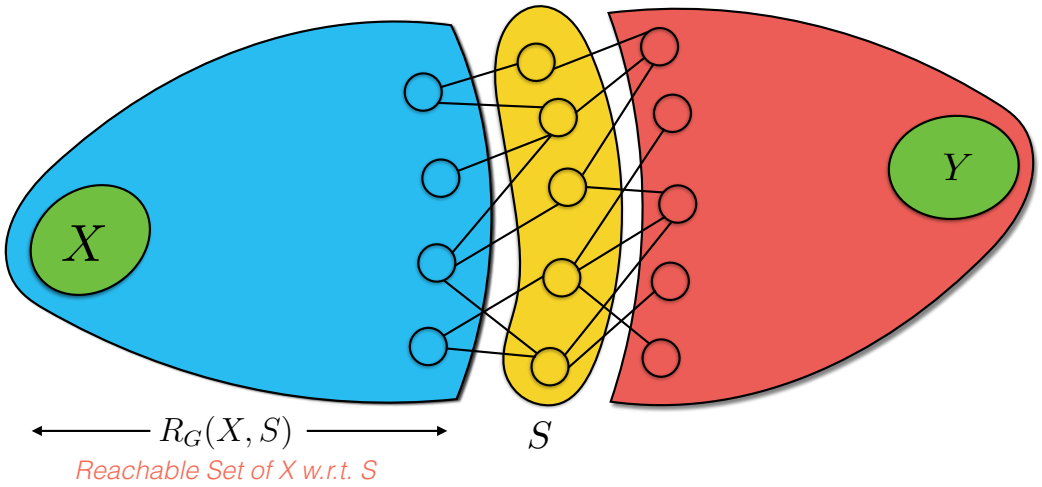


Figure 5-2: Separator

We now turn to the notion of a X-Y separator and what it means for one separator to cover another.

Definition 34 (Covering by Separators): *Let $G = (V, E)$ be an undirected graph and let $X, Y \subset V$ be two disjoint vertex sets. A subset $S \subseteq V \setminus (X \cup Y)$ is called an $X - Y$ separator in G if $R_G(X, S) \cap Y = \emptyset$, or in other words, there is no path from X to Y in the graph $G \setminus S$. We denote by $\lambda_G(X, Y)$ the size of the smallest $X - Y$ separator in G. An $X - Y$ separator S_1 is said to cover an $X - Y$ separator S with respect to X if $R(X, S_1) \supset R(X, S)$. If the set X is clear from the context, we just say that S_1 covers S. An $X - Y$ separator is said to be inclusionwise minimal if none of its proper subsets is an $X - Y$ separator.*

If $X = \{x\}$ is a singleton, then we abuse notation and refer to a $x - Y$ separator rather than a $\{x\} - Y$ separator. A separator S_1 dominates S if it covers S and is not larger than S in size:

Definition 35 (Dominating Separators [19]): *Let $G = (V, E)$ be an undirected graph and let $X, Y \subset V$ be two disjoint vertex sets. An $X - Y$ separator S_1 is said to dominate an $X - Y$ separator S with respect to X if $|S_1| \leq |S|$ and S_1 covers S with respect to X. If the set X is clear from the context, we just say that S_1 dominates S.*

We finally arrive at the notion of important separators, which are those that are not dominated by any other separator.

Definition 36 (Important Separators [19]): *Let $G = (V, E)$ be an undirected graph, $X, Y \subset V$ be disjoint vertex sets and $S \subseteq V \setminus (X \cup Y)$ be an $X - Y$ separator in G. We say that S is an important $X - Y$ separator if it is inclusionwise minimal and there does not exist another $X - Y$ separator S_1 such that S_1 dominates S with respect to X.*

It is well known that the important separator of smallest size is, in fact, unique:

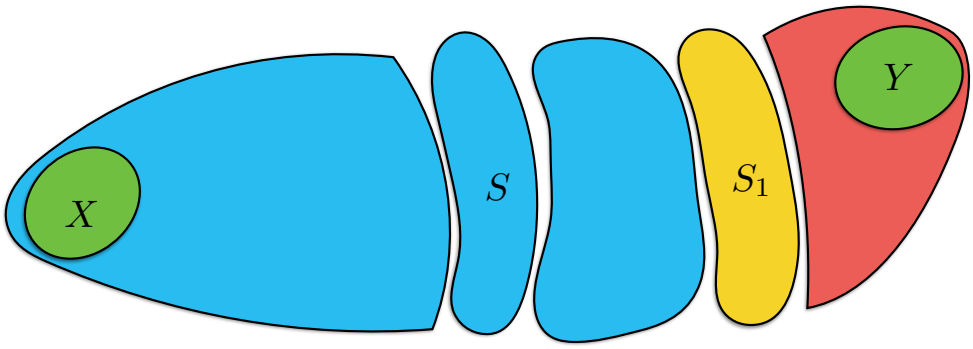


Figure 5-3: Dominating Separator

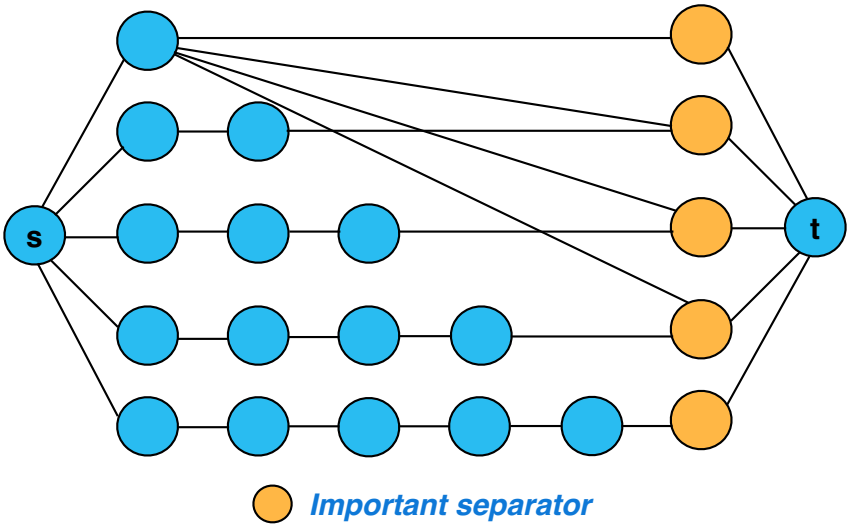


Figure 5-4: Important Separator

Lemma 37. *[63] Let $G = (V, E)$ be an undirected graph, $X, Y \subset V$ be disjoint vertex sets. There exists a unique important X-Y separator $S^\star$ of size $\lambda_G(X, Y)$ and it can be computed in polynomial time.*

It is useful to know that the number of important separators is bounded as a FPT function of the size of the important separators.

Lemma 38. *[13] Let $G = (V, E)$ be an undirected graph, $X, Y \subset V$ be disjoint vertex sets of G. For every $k \geq 0$ there are at most 4^k important $X - Y$ separators of size at most k. Furthermore, there is an algorithm that runs in time $O(4^k k(m + n))$ which enumerates all such important X-Y separators, where $n = |V|$ and $m = |E|$.*

Definition 39 (Important Separators of Order i [61]): *Let $G = (V, E)$ be a graph and let $X, Y \subseteq V$ be disjoint vertex sets. We define an important X, Y-separator of order i, S^i, to be the unique smallest important $X - S^{i-1}$-separator in G, where $S^0 = Y$.*

Definition 40 (Tight Separator Sequences [61]): *Let $G = (V, E)$ be a graph and let $X, Y \subseteq V$. Let $\ell \geq 1$ be such that there is no important $X - Y$ separator of order $\ell + 1$. We define a tight $X - Y$ separator sequence I to be a set $I = \{S^i | 1 \leq i \leq \ell\}$, where S^i is an important $X - Y$ separator of order i.*

It turns out that tight separator sequences are unique, and can also be computed in polynomial time [61].

Lemma 41 ([61])**.** *Let $G = (V, E)$ be a graph and let $X, Y \subseteq V$. A tight $X - Y$ separator sequence is unique and can be computed in polynomial time.*

However, the definition and structural lemmas regarding tight separator sequences used in this paper are closer to that from [61]. Since there are minor modifications in the definition as compared to the one in [61], we give the requisite proofs for the sake of completeness.

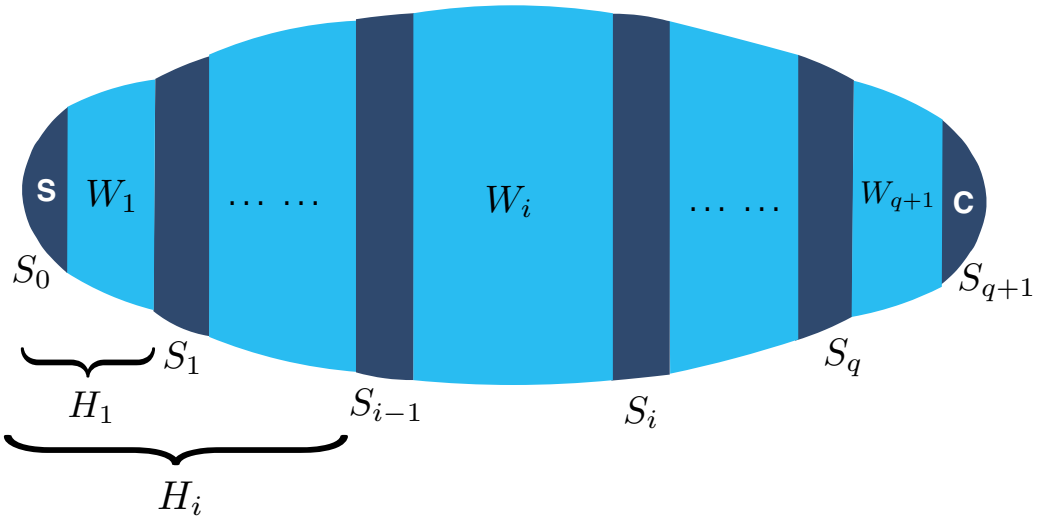


Figure 5-5: Tight Separator Sequence

Definition 42: *Let X and Y be two subsets of $V(G)$ and let $k \in \mathbb{N}$. A* tight (X,Y)-reachability sequence *of order k is an ordered collection $\mathcal{H} = \{H_0, H_1, \ldots, H_q\}$ of sets in $V(G)$ satisfying the following properties:*

- *$X \subseteq H_i \subseteq V(G) \setminus N[Y]$ for any $0 \leq i \leq q$;*
- *$X = H_0 \subset H_1 \subset H_2 \subset \cdots \subset H_q$;*
- *for every $0 \leq i \leq q$, H_i is reachable from X in $G[H_i]$ and every vertex in $N(H_i)$ can reach Y in $G - H_i$*
  (implying that $N(H_i)$ is a minimal (X,Y)-separator in G);
- *$|N(H_i)| \leq k$ for every $1 \leq i \leq q$;*

- $N(H_i) \cap N(H_j) = \emptyset$ *for all* $1 \leq i, j \leq q$ *and* $i \neq j$;
- *For any* $0 \leq i \leq q-1$, *there is no (X,Y)-separator* S *of size at most* k *where* $S \subseteq H_{i+1} \setminus N[H_i]$ *or* $S \cap N[H_q] = \emptyset$ *or* $S \subseteq H_1$.

We let $S_i = N(H_i)$, *for* $1 \leq i \leq q$, $S_{q+1} = Y$, *and* $\mathcal{S} = \{S_0, S_1, \ldots, S_q, S_{q+1}\}$. *We call* $\mathcal{S}$ *a* tight (X,Y)-separator sequence *of order* k.

Lemma 43. *([Choudhari et al. [18]], see for example [62]) There is an algorithm that, given an n-vertex m-edge graph* G, *subsets* $X, Y \in V(G)$ *and an integer* k, *runs in time* $O(kmn^2)$ *and either correctly concludes that there is no (X,Y)-separator of size at most* k *in* G *or returns the sets* $H_0, H_1, H_2 \setminus H_1, \ldots, H_q \setminus H_{q-1}$ *corresponding to a tight (X,Y)-reachability sequence* $\mathcal{H} = \{H_0, H_1, \ldots, H_q\}$ *of order* k.

Proof. The algorithm begins by checking whether there is a X-Y separator of size at most k. If there is no such separator, then it simply outputs the same. Otherwise, it uses the algorithm of Lemma 38 to compute an arbitrary important X-Y separator S of size at most k such that there is no X-Y separator of size at most k that covers S.

Although the algorithm of Lemma 38 requires time $O(4^k k(m+n))$ to enumerate *all* important X-Y separators of size at most k, *one* important separator of the kind described in the previous paragraph can in fact be computed in time $O(kmn)$ by the same algorithm.

If there is no X-S separator of size at most k, we stop and return the set $R(X,S)$ as the only set in a tight (X,Y)-reachability sequence. Otherwise, we recursively compute a tight (X,S)-reachability sequence $\mathcal{P} = \{P_0, \ldots, P_r\}$ of order k and define $\mathcal{Q} = \{P_0, \ldots, P_r, R(X,S)\}$ as a tight (X,Y)-reachability sequence of order k. It is straightforward to see that all the properties required of a tight (X,Y)-reachability sequence are satisfied. Finally, since the time required in each step of the recursion is $O(kmn)$ and the number of recursions is bounded by n, the number of vertices, the claimed running time follows. □

5.4 The FPT Algorithm for SACS

5.4.1 Overview of the Algorithm

Here we first give a brief overview of the FPT algorithm for Saving A Critical Set. The starting point for our FPT algorithm is the fact that every solution to an instance (G, s, C, k) of SACS is in fact a s-C separator of size at most k. Although the number of such separators may be exponential in the size of the graph, it is a well-known fact that the number of *important* separators is bounded by $4^k n^{O(1)}$ [13]. For several problems, one is able to prove that there exists a solution that is, in fact, an important separator. In such a situation, an FPT algorithm is immediate by guessing the important separator.

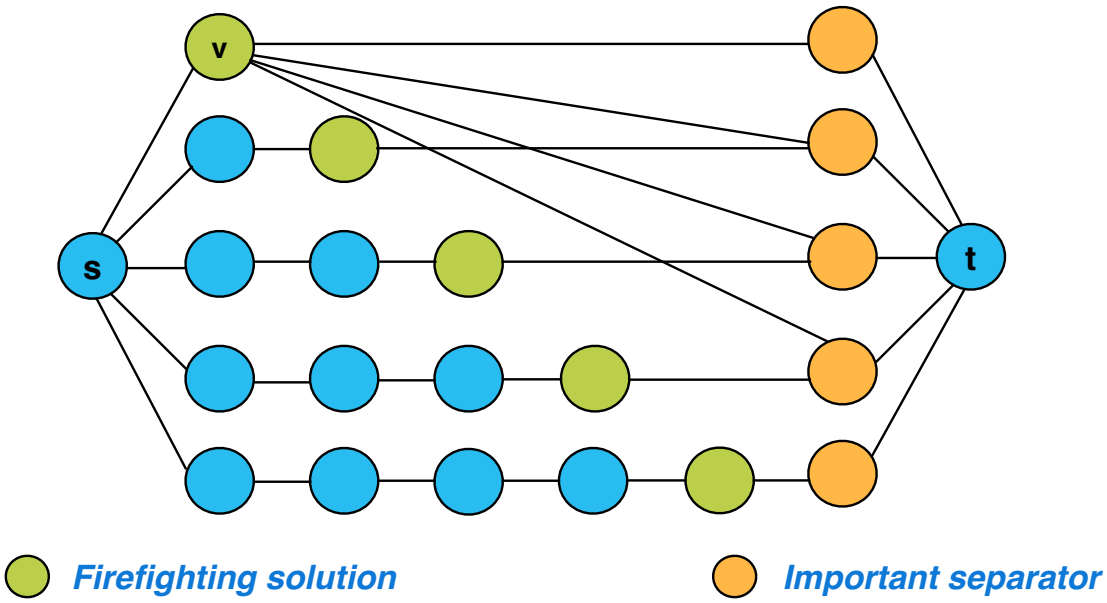


Figure 5-6: Important Separators do not work

In the SACS problem, unfortunately, there are instances where none of the solutions are important separators. However, this approach turns out to be feasible if we restrict our attention to trees, leading to improved running times. This is described in greater detail in Section 5.4.3. Further, in Section 5.5.2, we also show that we do not expect SACS to admit a polynomial kernel under standard complexity-theoretic assumptions. We establish this by a cross-composition from SACS itself, using the standard binary tree approach, similar to [6]. We describe our FPT algorithm for general graphs in Section 5.4.2. This is an elegant recursive procedure that operates over tight separator sequences, exploiting the fact that a solution can never be contained entirely in the region "between two consecutive separators". Although the natural choice of measure is the solution size, it turns out that the solution size by itself cannot be guaranteed to drop in the recursive instances that we generate. Therefore, we need to define an appropriate generalized instance and work with a more delicate measure.

5.4.2 The FPT Algorithm

Towards the FPT algorithm for SACS, we first define a generalized firefighting problem as follows. In this problem, in addition to (G, s, C, k), we are also given the following:

- $P \uplus Q \subseteq [2k]_O$, a set of *available time steps*,
- $Y \subset V(G)$, a subset of *predetermined firefighter locations*, and
- a bijection $\gamma : Q \to Y$, a *partial strategy* for Q.

The goal here is to find a valid partial k-step firefighting strategy over $(P \cup Q)$ that is consistent with γ on Q and saves C when the fire breaks out at s. We assume that no firefighters are placed during the time steps $[2k]_O \setminus (P \cup Q)$. For completeness, we formally define the notion of a valid partial firefighting strategy over a set.

Definition 44 (Partial Firefighting Strategy): *A partial k-step firefighting strategy on $X \subseteq [2k]_O$ is defined as a function $\mathfrak{h} : X \to V(G)$. Such a strategy is said to be valid in G with respect to s if, for all $i \in X$, when the fire breaks out in s and firefighters are placed according to $\mathfrak{h}$ for all time steps upto $[i-1]_O \cap X$, the vertex $\mathfrak{h}(i)$ is not burning at time step i. If G and s are clear from the context, we simply say that $\mathfrak{h}$ is a valid strategy over X.*

What it means for a partial strategy to save C is also analogous to what it means for a strategy to save C. The only difference here is that we save C despite not placing any firefighters during the time steps j for $j \in [2k]_O \setminus X$.

Definition 45 (Saving C with a Partial Strategy): *For a vertex s and a subset $C \subseteq V(G) \setminus \{s\}$, a partial firefighting strategy $\mathfrak{h}$ over X is said to save C if $\mathfrak{h}$ is a valid strategy and $\cup_{i \in X} \mathfrak{h}(i)$ is a $s - C$ separator in G, in other words, there is no path involving only burning vertices from s to any vertex in C if the fire starts at s and firefighters are placed according to $\mathfrak{h}$.*

The intuition for considering this generalized problem is the following: when we recurse, we break the instance G into two parts, say subgraphs G' and H. An optimal strategy for G employs some firefighters in H at some time steps X, and the remaining firefighters in G' at time steps $[2k]_O \setminus X$. When we recurse, we would, therefore, like to achieve two things:

- Capture the interactions between G' and H when we recursively solve H, so that a partial solution that we obtain from the recursion aligns with the larger graph, and
- Constrain the solution for the instance H to only use time steps in X, "allowing" firefighters to work in G' for the remaining time steps.

The constrained time steps in our generalized problem cater to the second objective, and the predetermined firefighter locations partially cater to the first. We now formally define the generalized problem.

Saving A Critical Set With Restrictions (SACS-R)	Parameter: k
Input:	An undirected n-vertex graph G, vertices s and g, a subset $C \subseteq V(G) \setminus \{s\}$, a subset $P \uplus Q \subseteq [2k]_O$, $Y \subset V(G)$ (such that $\lvert Y \rvert = \lvert Q \rvert$, $2k-1 \in Q$ and $g \in Y$), a bijeciton $\gamma : Q \to Y$ such that $\gamma(2k-1) = g$, and an integer k.
Question:	Is there a valid partial k-step strategy over $P \cup Q$ that is consistent with γ on Q and that saves C when a fire breaks out at s?

We use p and q to denote $|P|$ and $|Q|$, respectively. Note that we can solve an SACS instance (G, s, C, k) by adding an isolated vertex g and solving the SACS-R instance $(G, s, C, 2k+2, g, P, Q, Y, \gamma)$, where $P = [2k]_O$, $Q = \{2k+1\}$, $Y = \{g\}$ and $\gamma(2k+1) = g$.

Therefore, it suffices to describe an algorithm that solves SACS-R.

The role of the vertex g is mostly technical and will be clear in due course.

We now describe our algorithm for solving an instance $\mathcal{I} := (G, s, C, k, g, P, Q, Y, \gamma)$ of SACS-R. Throughout this discussion, for the convenience of analysis of YES instances, let $\mathfrak{h}$ be an arbitrary but fixed valid partial firefighting strategy in G over $P \cup Q$, consistent with γ on Q, that saves C. Our algorithm is recursive and works with pieces of the graph based on a tight $s - C$-separator sequence of separators of size at most $|P|$ in $G \setminus Y$. We describe the algorithm in three parts: the pre-processing phase, the generation of the recursive instances, and the merging of the recursively obtained solutions.

Phase 0 — Preprocessing. Observe that we have the following easy base cases:

- If $G \setminus Y$ has no $s - C$ separators of size at most p, then the algorithm returns NO.
- If $p = 0$, then we have a YES-instance if, and only if, s is separated from C in $G \setminus Y$ and $\mathfrak{h} := \gamma$ is a valid partial firefighting strategy over Q. In this case, the algorithm outputs YES or NO as appropriate.
- If $p > 0$ and s is already separated from C in $G \setminus Y$, then we return YES, since any arbitrary partial strategy over $P \cup Q$ that is consistent with γ on Q is a witness solution.

If we have a non-trivial instance, then our algorithm proceeds as follows. To begin with, we compute a tight $s - C$ separator sequence of order p in $G \setminus Y$. Recalling the notation of Definition 42 , we use $S_0, \ldots, S_{q+1}$ to denote the separators in this sequence, with S_0 being the set $\{s\}$ and $S_{q+1} = C$. We also use $W_0, W_1, \ldots, W_q, W_{q+1}$ to denote the reachability regions between consecutive separators. More precisely, if $\mathcal{H}$ is the tight $s - C$ reachability sequence associated with $\mathcal{S}$, then we have:

$$W_i := H_i \setminus N[H_{i-1}] \text{ for } 1 \leq i \leq q,$$

while W_{q+1} is defined as $G \setminus (N[H_q] \cup C)$. We will also frequently employ the following notation:

$$\mathcal{S} = \bigcup_{i=1}^{q} S_i \text{ and } \mathcal{W} = \bigcup_{i=1}^{q+1} W_i. \tag{5.2}$$

This is a slight abuse of notation since $\mathcal{S}$ is also used to denote the sequence $S_0, \ldots, S_{q+1}$, but the meaning of $\mathcal{S}$ will typically be clear from the context.

We first observe that if $q > k$, the separator S_q can be used to define a valid partial firefighting strategy. The intuition for this is the following: since every vertex in S_q is at a distance of at least k from s, we may place firefighters on vertices in S_q in any order during the available time steps. Since $|S_q| \leq p$ and S_q is a $s - C$ separator, this is a valid solution. Thus, we have shown the following:

Lemma 46. *If G admits a tight $s - C$ separator sequence of order q in $G \setminus Y$ where $q > k$, then $\mathcal{I}$ is a YES-instance.*

Therefore, we return YES if $q > k$ and assume that $q \leq k$ whenever the algorithm proceeds to the next phase.
This concludes the pre-processing stage.

Phase 1 — Recursion. Our first step here is to guess a partition of the set of available time steps, P, into $2q + 1$ parts, denoted by $A_0, \ldots, A_q, A_{q+1}$ and $B_1, \ldots, B_{q+1}$. The partition of the time steps represents how a solution might distribute the timings of its firefighting strategy among the sets in $\mathcal{S}$ and $\mathcal{W}$. The set A_i denotes our guess of $\cup_{v \in S_i} \mathfrak{h}^{-1}(v)$ and B_j denotes our guess of $\cup_{v \in W_j} \mathfrak{h}^{-1}(v)$. Note that the number of such partitions is $(2q + 1)^p \leq (2k + 1)^k$. We define $g_0(k) := (2k + 1)^k$. We also use $\mathcal{T}_1(P)$ to denote the partition $A_0, \ldots, A_q$ and $\mathcal{T}_2(P)$ to denote $B_0, \ldots, B_{q+1}$.

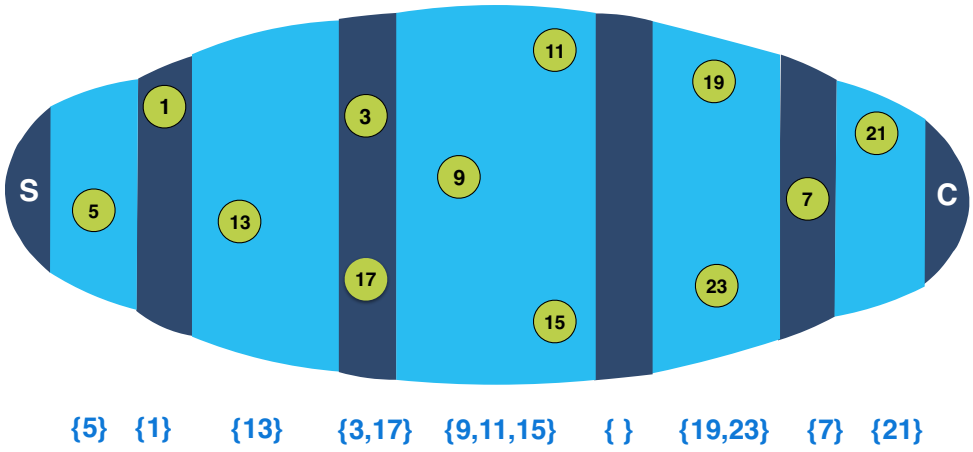


Figure 5-7: Guess the partition

We say that the partition $(\mathcal{T}_1(P), \mathcal{T}_2(P))$ is non-trivial if none of the B_i's are such that $B_i = P$. Our algorithm only considers non-trivial partitions — the reason this is sufficient follows from the way tight separator sequences are designed, and this will be made more explicit in Lemma 48 in due course.

Next, we would like to guess the behavior of a partial strategy over P restricted to $\mathcal{S}$. Informally, we do this by associating a signature with the strategy $\mathfrak{h}$, which is is a labeling of the vertex set with labels corresponding to the status of a vertex in the firefighting game when it is played out according to $\mathfrak{h}$. Every vertex is labeled as either a vertex that had a firefighter placed on it, a burned vertex, or a saved vertex. The

labels also carry information about the earliest times at which the vertices attained these statuses. More formally, we have the following definition.

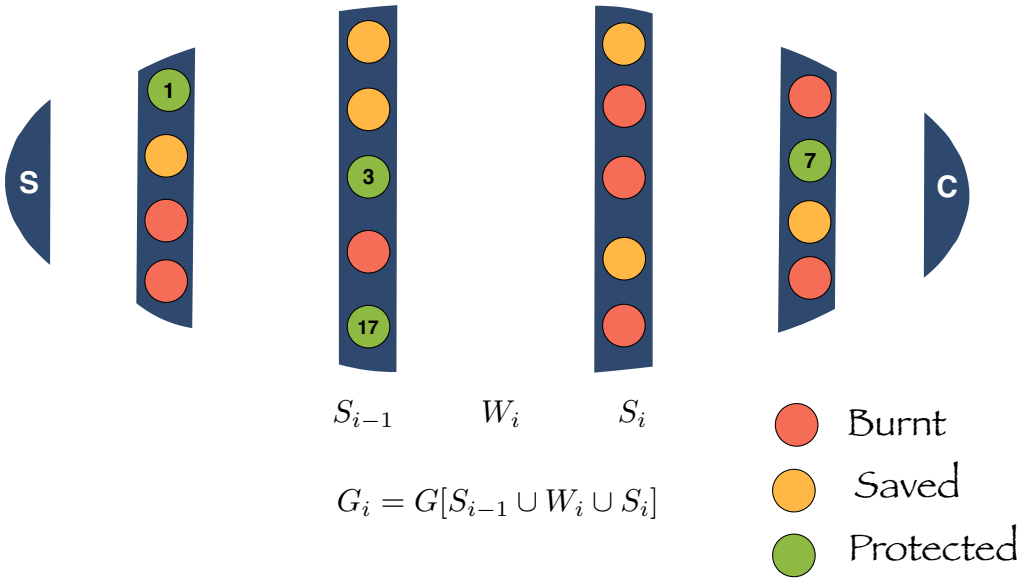


Figure 5-8: Possible Labeling

Definition 47: *Let $\mathfrak{h}$ be a valid k-step firefighting strategy (or a partial strategy over X). The signature of $\mathfrak{h}$ is defined as a labeling $\mathfrak{L}_{\mathfrak{h}}$ of the vertex set with labels from the set:*

$$\mathcal{L} = (\{\mathfrak{f}\} \times X) \cup (\{\mathfrak{b}\} \times [2k]_{\mathcal{E}}) \cup \{\mathfrak{p}\},$$

where:

$$\mathfrak{L}_{\mathfrak{h}}(v) = \begin{cases} (\mathfrak{f}, t) & \text{if } \mathfrak{h}(t) = v, \\ (\mathfrak{b}, t) & \text{if } t \text{ is the earliest time step at which } v \text{ burns}, \\ \mathfrak{p} & \text{if } v \text{ is not reachable from } s \text{ in } G \setminus (\{\mathfrak{h}(i) \mid i \in [2k]_O\}) \end{cases}$$

We use array-style notation to refer to the components of $\mathfrak{L}(v)$, for instance, if $\mathfrak{L}(v) = (\mathfrak{b}, t)$, then $\mathfrak{L}(v)[0] = \mathfrak{b}$ and $\mathfrak{L}(v)[1] = t$. The algorithm begins by guessing the restriction of $\mathfrak{L}_{\mathfrak{h}}$ on $\mathcal{S}$, that is, it loops over all possible labellings:

$$\mathfrak{T} : \mathcal{S} \to (\{\mathfrak{f}\} \times P) \cup (\{\mathfrak{b}\} \times [2k]_{\mathcal{E}}) \cup \{\mathfrak{p}\}.$$

The labeling $\mathfrak{T}$ is called legitimate if, for any $u \neq v$, whenever $\mathfrak{T}(u)[0] = \mathfrak{T}(v)[0] = \mathfrak{f}$, we have $\mathfrak{T}(u)[1] \neq \mathfrak{T}(v)[1]$. We say that a labeling $\mathfrak{T}$ over $\mathcal{S}$ is compatible with $\mathcal{T}_1(P) = (A_0, \ldots, A_q)$ if we have:

- for all $0 \leq i \leq r$, if $v \in S_i$ and $\mathfrak{h}(v)[0] = \mathfrak{f}$, then $\mathfrak{h}(v)[1] \in A_i$.
- for all $0 \leq i \leq r$, if $t \in A_i$, there exists a vertex $v \in S_i$ such that $\mathfrak{h}^{-1}(\mathfrak{f}, t) = v$.

The algorithm considers only legitimate labelings compatible with the current choice of $\mathcal{T}_1(P)$. By Lemma 46, we know that any tight $s-C$ separator sequence considered by the algorithm at this stage has at most k separators of size at most p each. Therefore, we have that the number of labelings considered by the algorithm is bounded by $g_1(k) := (p+k+1)^{(kp)} \leq (3k)^{O(k^2)} \leq k^{O(k^2)}$.

We are now ready to split the graph into $q+1$ recursive instances. For $1 \leq i \leq q+1$, let us define $G_i = G[S_{i-1} \cup W_i \cup S_i \cup Y]$. Also, let $\mathfrak{T}_i := \mathfrak{T} \mid_{V(G_i) \cap \mathcal{S}}$. Notice that when using G_i's in recursion, we need to ensure that the independently obtained solutions are compatible with each other on the non-overlapping regions, and consistent on the common parts. We force consistency by carrying forward the information in the signature of $\mathfrak{h}$ using appropriate gadgets, and the compatibility among the W_i's is a result of the partitioning of the time steps.

Fix a partition of the available time steps P into $\mathcal{T}_1(P)$ and $\mathcal{T}_2(P)$, a compatible labeling $\mathfrak{T}$ and $1 \leq i \leq q+1$. We will now define the SACS-R instance $\mathcal{I}\langle i, \mathcal{T}_1(P), \mathcal{T}_2(P), \mathfrak{T}_i \rangle$. Recall that $\mathcal{I} = (G, s, C, k, g, P, Q, Y, \gamma)$. To begin with, we have the following:

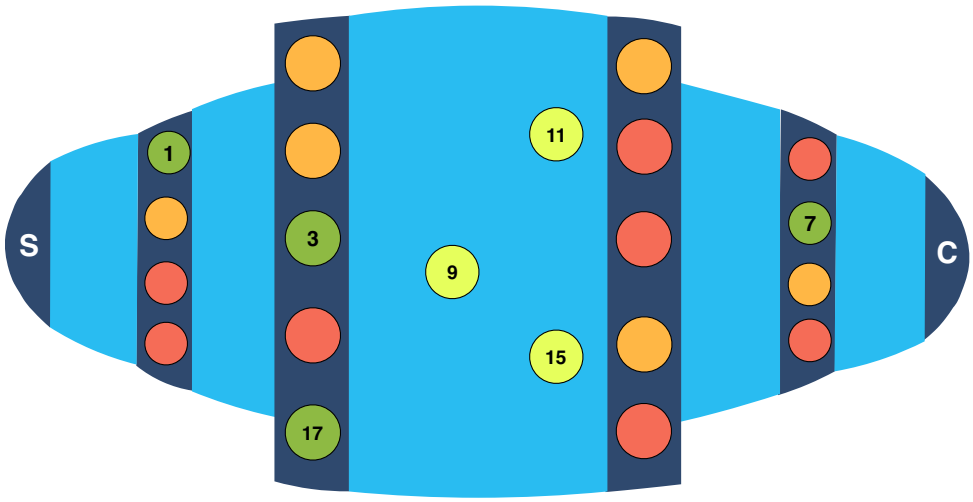
- Let $X_i = A_{i-1} \cup A_i$ and let $P_i = B_i$.
- Let $Q_i := X_i \cup Q$ and $Y_i := Y \cup X_i$. We define γ_i as follows:

$$\gamma_i(t) = \begin{cases} \gamma(t) & \text{if } t \in Q, \\ v & \text{if } t \in X_i \text{ and } \mathfrak{T}_i(v) = (\mathfrak{f}, t) \end{cases}$$

Note that γ_i is well-defined because the labeling was legitimate and compatible with $\mathcal{T}_1(P)$. We define H_i to be the graph $\chi(G_i, \mathfrak{T}_i)$, which is described below.

- To begin with, $V(H_i) = V(G_i) \cup \{s^\star, t^\star\}$
- Let $v \in V(G_i)$ be such that $\mathfrak{T}_i(v)[0] = \mathfrak{b}$. Use ℓ to denote $\mathfrak{T}_i(v)[1]/2$. Now, we do the following:
  - Add $k+1$ internally vertex disjoint paths from $s^\star$ to v of length $\ell+1$, in other words, these paths have $\ell-1$ internal vertices.
  - Add $k+1$ internally vertex disjoint paths from v to g of length $k-\ell-1$.
- Let $v \in V(G_i)$ be such that $\mathfrak{T}_i(v) = \mathfrak{p}$. Add an edge from v to $t^\star$.
- We also make $k+1$ copies of the vertices $t^\star$ and all vertices that are labeled either burned or saved. This ensures that no firefighters are placed on these vertices.

For $1 \leq i \leq q+1$, the instance $\mathcal{I}\langle i, \mathcal{T}_1(P), \mathcal{T}_2(P), \mathfrak{T}_i \rangle$ is now defined as $(\chi(G_i, \mathfrak{T}_i), s^\star, C = \{t^\star\}, k, g, P_i, Q_i, Y_i, \gamma_i)$.



S_{i-1} W_i S_i

$$G_i = G[S_{i-1} \cup W_i \cup S_i]$$

Figure 5-9: Solving the border problem

Phase 2 — Merging. Our final output is quite straightforward to describe once we have the $\mathfrak{h}[\mathfrak{T}_i, i]$'s. Consider a fixed partition of the available time steps P into $\mathcal{T}_1(P)$ and $\mathcal{T}_2(P)$, and a labeling $\mathfrak{T}$ of $\mathcal{S}$ compatible with $\mathcal{T}_1(P)$. If all of the $(q+1)$ instances $\mathcal{I}\langle i, \mathcal{T}_1(P), \mathcal{T}_2(P), \mathfrak{T}_i\rangle$, $1 \le i \le q+1$ return YES, then we also return YES, and we return NO otherwise. Indeed, in the former case, let $\mathfrak{h}[i, \mathcal{T}_1(P), \mathcal{T}_2(P), \mathfrak{T}]$ denote a valid partial firefighting strategy for the instance $\mathcal{I}\langle i, \mathcal{T}_1(P), \mathcal{T}_2(P), \mathfrak{T}_i\rangle$. We will show that $\mathfrak{h}^\star$, described as follows, is a valid partial firefighting strategy that saves C.

- For the time steps in Q, we employ firefighters according to γ.
- For the time steps in $\mathcal{T}_1(P)$, we employ firefighters according to $\mathfrak{T}$. This is a well-defined strategy since $\mathfrak{T}$ is a compatible labeling.
- For all remaining time steps, i.e, those in $\mathcal{T}_2(P) = \{B_1, \ldots, B_{q+1}\}$, we follow the strategy given by $\mathfrak{h}[i, \mathcal{T}_1(P), \mathcal{T}_2(P), \mathfrak{T}]$.

It is easily checked that the strategy described above agrees with $\mathfrak{h}[i, \mathcal{T}_1(P), \mathcal{T}_2(P), \mathfrak{T}]$ for all i. Also, the strategy is well-defined, since $\mathcal{T}_1(P)$ and $\mathcal{T}_2(P)$ form a partition of the available time steps. Next, we will demonstrate that $\mathfrak{h}^\star$ is indeed a valid strategy that saves C, and also analyze the running time of the algorithm.

Correctness of the Algorithm

We first show that the quantity p always decreases when we recurse.

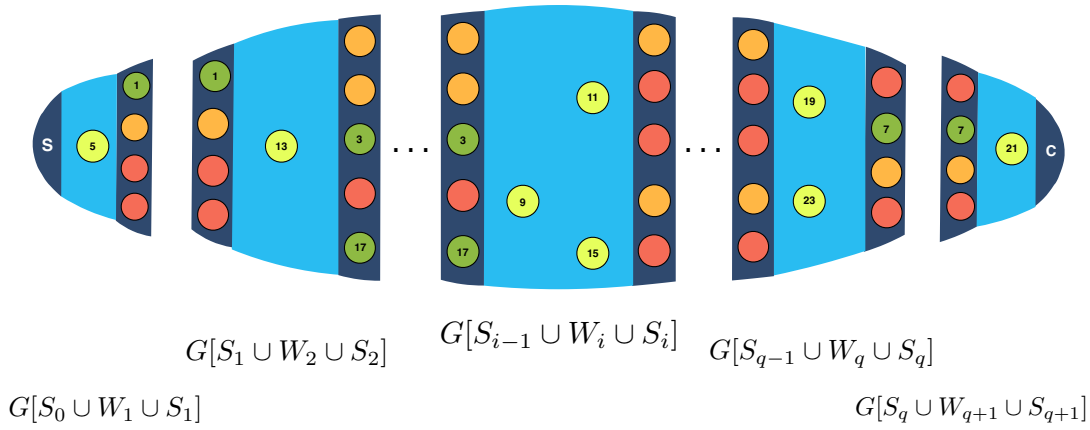


Figure 5-10: Solving Recursively

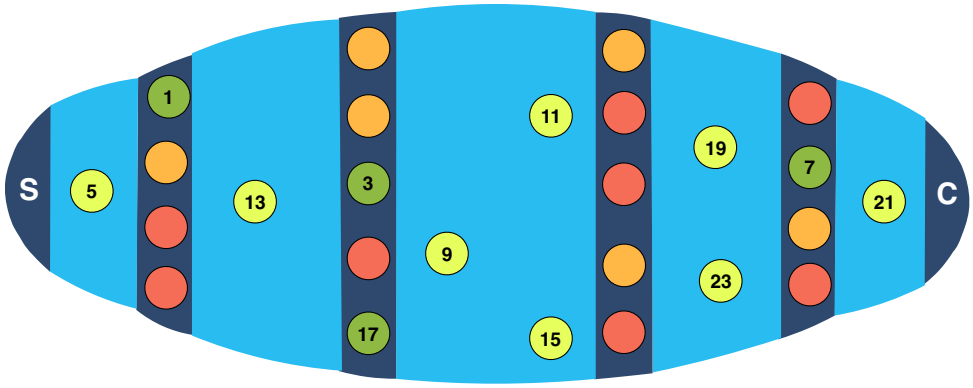


Figure 5-11: Merging Solutions

Lemma 48 (Choudhari et al. [18]). *Let $\mathcal{I}' := \mathcal{I}\langle i, \mathcal{T}_1(P), \mathcal{T}_2(P), \mathfrak{T}_i\rangle = (\chi(G_i, \mathfrak{T}_i), s^\star, C = \{t^\star\}, k, g, P_i, Q_i, Y_i, \gamma_i)$ be an arbitrary but fixed instance constructed by Solve-SACS-R($\mathcal{I}$). Then, $|P_i| < |P|$.*

Proof. The claim follows from the fact that $P_i = B_i$, and $(\mathcal{T}_1(P), \mathcal{T}_2(P))$ is a non-trivial partition of P. □

Lemma 49 (Choudhari et al. [18]). *If $\mathcal{I}$ is a YES instance of SACS-R, then our algorithm returns YES.*

Proof. The statement follows by induction on p, which allows us to assume the correctness of the output of the recursive calls. The correctness of the base cases is easily checked. If $\mathcal{I}$ is a YES instance admits a solution $\mathfrak{h}$, then it induces a partition $\mathcal{T}_1(P), \mathcal{T}_2(P)$ of P. We argue that this is always a non-trivial partition and is therefore considered by the algorithm. Indeed, suppose not. This would imply that $\mathfrak{h}$ places all its firefighters in W_i for some $1 \leq i \leq q+1$. However, this implies that $D := \cup_{t \in P_i} \mathfrak{h}(t) \subseteq W_i$ is a $s - C$ separator in $G \setminus Y$ that is entirely contained in W_i, which contradicts the definition of a tight $s - C$ separator sequence.

Algorithm 7: Solve-SACS-R($\mathcal{I}$)

Input: An instance $(G, s, C, k, g, P, Q, Y, \gamma)$, $p := |P|$
Result: YES if $\mathcal{I}$ is a YES-instance of SACS-R, and NO otherwise.

```
1  if p = 0 and s and C are in different components of G \ Y then return YES;
2  else return NO;
3  if p > 0 and s and C are in different components of G \ Y then return YES;
4  if there is no s − C separator of size at most p then return NO;
5  Compute a tight s − C separator sequence S of order p.
6  if the number of separators in S is greater than k then return YES;
7  else
8      for a non-trivial partition T_1(P), T_2(P) of P into 2q + 1 parts do
9          for a labeling 𝔗 compatible with T_1(P) do
10             if ⋀_{i=1}^{q+1} (Solve-SACS-R(I⟨i, T_1(P), T_2(P), 𝔗_i⟩)) then return YES;
11     return NO
```

Now, define the labeling $\mathfrak{T}$ by projecting the signature of $\mathfrak{h}$ on $\mathcal{S}$. Clearly, this labeling is compatible with $\mathcal{T}_1(P)$ (by definition). We claim that all the instances $\mathcal{I}\langle i, \mathcal{T}_1(P), \mathcal{T}_2(P), \mathfrak{T}_i\rangle$ are YES instances. Indeed, it is easy to check that the projection of $\mathfrak{h}$ on the subgraph G_i is a valid solution for the instance $\mathcal{I}\langle i, \mathcal{T}_1(P), \mathcal{T}_2(P), \mathfrak{T}_i\rangle$. It follows that the algorithm returns YES since these $(q+1)$ instances return YES by the induction hypothesis. □

For the remainder of our discussion on correctness, our goal will be to prove the following reverse claim.

Lemma 50 (Choudhari et al. [18])**.** *If the output of the algorithm is YES, then $\mathcal{I}$ is a YES instance of SACS-R.*

This lemma is shown by establishing that $\mathfrak{h}^\star$ is indeed a valid solution for SACS. We arrive at this conclusion by a sequence of simple claims. We fix the accepting path in the algorithm, that is, an appropriate partition of P and the labeling $\mathfrak{T}$ on $\mathcal{S}$ that triggered YES output. Our first claim says that the recursive instances respect the behavior dictated by the labeling that they were based on.

Lemma 51 (Choudhari et al. [18])**.** *Let $1 \leq i \leq q+1$ and $v \in \mathcal{S} \cap H_i$. If $\mathfrak{T}_i(v) = (\mathfrak{b}, t)$, then in any partial strategy employed on the instance $\mathcal{I}\langle i, \mathcal{T}_1(P), \mathcal{T}_2(P), \mathfrak{T}_i\rangle$ that saves the critical set, the vertex v burns exactly at time step t.*

Proof. Let $\mathfrak{h}'$ be an arbitrary but fixed valid partial strategy for $\mathcal{I}\langle i, \mathcal{T}_1(P), \mathcal{T}_2(P), \mathfrak{T}_i\rangle$ that saves the critical set. Assume that v burns with respect to $\mathfrak{h}'$. Let t' be the earliest time step at which v burns with respect to $\mathfrak{h}'$. The graph $\chi(G_i, \mathfrak{T}_i)$ contains $(k+1)$ vertex-disjoint paths from the source to v of length $t/2$, which ensures that $t' \leq t$. However, if $t' < t$, then the vertex g catches fire at time step $2k-1$ because of the

$(k+1)$ vertex-disjoint paths of length $k - t/2 - 1$ that are present from v to g. This implies that $t^\star$ cannot be saved if v burns earlier than t. The other case is that v does not burn with respect to $\mathfrak{h}'$. The only way for this to happen is if a firefighter is placed on v, however, since the instance has $k+1$ copies of v, we have that at least one copy of v burns, and the claim follows. □

Our next claim is that vertices that are labeled saved never burn with respect to $\mathfrak{h}^\star$.

Lemma 52 (Choudhari et al. [18]). *For any $v \in \mathcal{S}$, if $\mathfrak{T}(v) = \mathfrak{p}$, then v does not burn with respect to $\mathfrak{h}^\star$.*

Proof. We prove this by contradiction. Let P be a path from s to v where all vertices on P are burning. Let v' be the first vertex on this path which is such that $\mathfrak{T}(v') = \mathfrak{p}$, and let u be the last vertex on this path which is before v' and that intersects a separator. Observe that u is well-defined unless $v' \in S_1$, which is a special case that we will deal with separately. Note that u must be a vertex that is labeled as burned (by the choice of v'). Therefore, the path from u to v is present in a recursive instance, where we know that v' is adjacent to a critical vertex, which contradicts the fact that we defined $\mathfrak{h}^\star$ based on valid strategies that save critical sets in the recursive instances. If $v' \in S_1$ and u is not well-defined, then observe that there is a direct path from s to v' in the first recursive instance, and the same argument applies. □

We finally show, over the next two claims, that the function $\mathfrak{h}^\star$ is a valid strategy that saves the critical set.

Lemma 53 (Choudhari et al. [18]). *The function $\mathfrak{h}^\star$ is a valid partial strategy over $(P \cup Q)$ for the instance $\mathcal{I}$.*

Proof. We prove this by contradiction, and we also assume that Lemma 50 holds for all recursive instances (by induction on p). If $\mathfrak{h}^\star$ is not a valid strategy, then there exists some $1 \leq i \leq q+1$ for which there is a vertex $v \in W_i$ and a time step $t \in B_i$ such that $\mathfrak{h}^\star(t) = v$ and v is burning at time step t. Consider the path P from s to v. Let u be the last vertex on the path P that intersects $S_{i-1} \cup S_i$. Observe that u is a vertex that is either labeled by $(\mathfrak{b}, t)$ or $\mathfrak{p}$. This leads to the following two scenarios:

- In the first case, the part of the path from u to v is present in the instance $\mathcal{I}\langle i, \boldsymbol{\mathcal{T}}_1(P), \boldsymbol{\mathcal{T}}_2(P), \mathfrak{T}_i\rangle$, because of the agreement of $\mathfrak{h}^\star$ and $\mathfrak{h}[i, \boldsymbol{\mathcal{T}}_1(P), \boldsymbol{\mathcal{T}}_2(P), \mathfrak{T}]$ and Lemma 51. Therefore, $\mathfrak{h}[i, \boldsymbol{\mathcal{T}}_1(P), \boldsymbol{\mathcal{T}}_2(P), \mathfrak{T}]$ was not a valid firefighting strategy.
- The second situation implies that a vertex labeled protected is burning in $\mathcal{I}$ when $\mathfrak{h}^\star$ is employed, which contradicts Lemma 52.

□

Lemma 54 (Choudhari et al. [18])**.** *The partial strategy* $\mathfrak{h}^\star$ *saves the set* C *in the instance* $\mathcal{I}$*.*

Proof. (Sketch.) The proof of this is similar to the proof of Lemma 53. Any burning path P from s to any vertex in C must intersect S_q. We let v be the last vertex from S_q on P, and observe that v must have a $(\mathfrak{b}, t)$. Therefore, the path from v to the vertex in C exists in $\mathcal{I}\langle q+1, \boldsymbol{\mathcal{T}}_1(P), \boldsymbol{\mathcal{T}}_2(P), \mathfrak{T}_i\rangle$, and also burns in the same fashion, since $\mathfrak{h}^\star$ agrees with $\mathfrak{h}[q+1, \boldsymbol{\mathcal{T}}_1(P), \boldsymbol{\mathcal{T}}_2(P), \mathfrak{T}]$. Again, this contradicts the accuracy of the algorithm on the recursive instance. □

The proof of Lemma 50 now follows from Lemmas 53 and 51.

Running Time Analysis

Subsequently, it branches into at most $r \leq f(k).(q+1)$ instances This is given by $g_0(k)g_1(k)$, recall that this is bounded by $k^{O(k^2)}$.

We show that our algorithm runs in time $f(k)^{p^2}O(n^2m)$, where $f(k) = k^{O(k)}$. Indeed, observe that the running time of the algorithm is governed by the following recurrence:

$$T(n, m, k, p) \leq O(n^2mp) + (p+k+1)^{kp} \sum_{i=1}^{q+1} T(n_i, m_i, k, p_i)$$

where the term $(p+k+1)^{kp}$ denotes an upper bound on the product of the total number of non-trivial partitions of P into $(2q+1)$ parts and the total number of legitimate labelings of $\mathcal{S}$ compatible with $\boldsymbol{\mathcal{T}}_1(P)$.

The first term accounts for checking the base cases and the running time of computing a tight separator sequence of order p (see Lemma 43). Since p is always at most k, we rewrite the recurrence as

$$T(n, m, k, p) \leq O(n^2mp) + f(k)^p \sum_{i=1}^{q+1} T(n_i, m_i, k, p_i),$$

where $f(k) = k^{O(k)}$. Furthermore, observe that $\sum_{i=1}^{q+1} n_i \leq n + pk$. The first inequality follows from the fact that every vertex in $\mathcal{S}$ appears in at most two recursive instances. The first occurrence is counted in n, while all the second occurrences combined amount to at most pk.

Finally, recall that $p_i < p$ by Lemma 48 and $p \leq k$ by definition (since $P \subseteq [2k]_O$). Therefore, the depth of the recursion is bounded by p, and the time spent at each level of recursion is proportional to $k^{O(kp)}n^2m$. This implies the claimed running time.

Based on the analysis above and Lemmas 49 and 50, we have thus demonstrated the main result of this section.

Theorem 55 (Choudhari et al. [18]). *SACS is* FPT *and has an algorithm with running time $f(k)O(nm^2)$, where $f(k) = k^{O(k^3)}$.*

5.4.3 A Faster Algorithm For Trees

In this section, we consider the setting when the input graph G is a tree. WLOG, we consider the vertex s to be the root of the tree. We first state an easy claim that shows that WLOG, we can consider the critical set to be the leaves. The proof of the following lemma follows from the fact that the firefighting solution has to be a $s - C$ separator.

Lemma 56 (Choudhari et al. [18]). *When the input graph G is a tree, if there exists a solution to SACS, there exists a solution such that all firefighter locations are on nodes that are on some path from s to C.*

Given the above claim, our algorithm to construct a firefighting solution is the following– exhaustively search all the important $s - C$ separators that are of size k. For each vertex v in a separator Y, we place firefighters on Y in the increasing order of distance from s and check whether this is a valid solution.

The following lemma claims that if there exists a firefighting solution, the above algorithm will return one.

Lemma 57 (Choudhari et al. [18]). *Solving the SACS problem for input graphs that are trees takes time $O^*(4^k)$.*

Proof. Using Lemma 56, it is enough to consider the subtree T that contains nodes only on $s - C$ paths. The critical set C is then a subset of the leaves of T. Suppose $Y \subset V(T)$ contains the locations for a solution to the firefighter problem. WLOG, Y is a minimal $s - C$ separator. Consider I which is an important separator that dominates Y. Clearly, $|I| \leq |Y| \leq k$, and I is also a minimal separator.

For each $x \in I$, define S_x to be the set of $y \in Y$ such that y lies on the (unique) $s - x$ path. Note that each $y \in Y$ must belong to some $s - C$ path, and all $s - C$ paths have some node $x \in I$. Furthermore, $R(s, Y) \subseteq R(s, I)$, which means that each $y \in Y$ lies on some $s - I$ path. It follows that $Y \subseteq \cup_{x \in I} S_x$. Finally, by the minimality of Y, each S_x satisfies $|S_x| \leq 1$, since otherwise, we could remove one of the nodes of S_x from Y. Hence,

$$|Y| \leq |\cup_{x \in I} S_x| \leq \sum_{x \in I} |S_x| \leq |I| \leq |Y|$$

Thus $|\cup_{x \in I} S_x| = \sum_{x \in I} |S_x|$, and it follows that $S_x \cap S_u = \emptyset$ for all $x \neq u$. From $\sum_{x \in I} |S_x| = |I|$ and $|S_x| \leq 1$ for each x it also follows that for each x, $|S_x| = 1$.

We then design a firefighting solution using I in the following manner. For each node, $x \in I$, the firefighter in location x is placed whenever the original solution (using Y) placed a firefighter on the unique node in S_x. Since the node in S_x is closer to s than x, the location x is still available at this step. Hence this is a valid strategy. Note that if there is any valid placement ordering for I, then the placement ordering according to increasing distance from s is also valid. The claim then follows by noting that enumerating all the important separators of size at most k takes time $O^*(4^k)$. □

5.5 Kernelization

Recall the instance of the Vertex Cover problem (G, k) that was used as an example to explain FPT algorithms. We will now show that with the help of some simple observations it is possible to reduce the size of the instance without affecting the final solution of the problem. So here is the first reduction rule:

Reduction Rule 1: If the graph G contains a vertex v such that, the degree of v is 0, then remove that vertex from the graph and consider the new instance of the problem, which is $(G \setminus \{v\}, k)$.

Notice that, removing the degree 0 vertices from the graph does not affect the final vertex cover solution. The next reduction rule is based on the observation that if the problem instance (G, k) contains a vertex v of degree greater k, then v must be the part of the vertex cover solution.

Reduction Rule 2: If the graph G contains a vertex v of degree at least $k + 1$, then remove v from the graph along with its incident edges and reduce the parameter k to $k - 1$. The updated instance of the problem is $(G \setminus \{v\}, k - 1)$.

Observe that, if the vertex v with degree at least $k + 1$ is not a part of the vertex cover solution, then all the neighbors of v must be in the solution. This would make the solution size to be at least $k + 1$ which is not desirable given the parameter k.

The reduction rules 1 and 2 lead to the following lemma:

Lemma 58 ([19])**.** *If (G, k) is a Yes-instance and none of the reduction rules 1 and 2 are applicable to G, then the number of vertices $|V(G)|$ in G is at most $k^2 + k$, and the number of edges $|E(G)|$ in G is at most k^2.*

Therefore, now we can conclude that, if (G, k) with $k > 0$ is an input instance such that reduction rules 1 and 2 are not applicable, and if G has more than $k^2 + k$ vertices or more than k^2 edges, then the instance (G, k) is a No-instance of the Vertex Cover problem.

In the above exercise of applying reduction rules to the instance of the problem, notice that the aim was to reduce the size of the instance of the problem by doing some

preprocessing. Also, notice that all the reduction rules can be applied in polynomial-time. And this is what the aim behind *kernelization* is, i.e. to reduce the problem instance efficiently. Mostly, while dealing with NP-hard problems preprocessing(data reduction or kernelization) is the first step towards solving the problem. In the preprocessing subroutine, the aim is to solve the "easy parts" of the problem instance efficiently and then work on the shrunken or reduced part, which is the main core of the problem. The advantage of the shrunken part is that it being smaller one can think of running slower algorithms as well.

In the language of parameterized complexity, *kernelization* formalizes the definition of useful preprocessing. One can think that in *kernelization* the demand is shrunk the large instances with a smaller parameter, while the instances which are smaller as compared to the parameter should not be preprocessed further.

Let us now move on to the formal definition for *kernelization* or what does it mean to have a *kernel* for a problem instance.

Definition 59 (Data reduction rule (Section 2.1 [19])): *A data reduction rule, or simply, reduction rule, for a parameterized problem Q is a function $\Phi : \Sigma^* \times \mathbb{N} \to \Sigma^* \times \mathbb{N}$ that maps an instance (I, k) of Q to an equivalent instance (I', k') of Q such that Φ is computable in time polynomial in $|I|$ and k.*

Two instances of Q are said to be equivalent if $(I, k) \in Q$ if and only if $(I', k') \in Q$; this property of the reduction rule Φ, that it translates an instance to an equivalent one, is sometimes referred to as the *safeness or soundness* of the reduction rule [19].

Definition 60 (Kernelization, kernel (Definition 2.1 [19])): *A kernelization algorithm, or simply a kernel, for a parameterized problem Q is an algorithm $\mathcal{A}$ that, given an instance (I, k) of Q, works in polynomial time and returns an equivalent instance (I', k') of Q. Moreover, we require that $size_{\mathcal{A}}(k) \leq g(k)$ for some computable function $g : \mathbb{N} \to \mathbb{N}$.*

Here, $size_{\mathcal{A}}(k)$ is the *output size* of the preprocessing algorithm $\mathcal{A}$, which is a function $size_{\mathcal{A}}(k) : \mathbb{N} \to \mathbb{N} \cup \{\infty\}$ defined as:

$$size_{\mathcal{A}}(k) = sup\{|I'| + k' : (I', k') = \mathcal{A}(I, k), I \in \Sigma^*\}$$

This says that the *Kernelization algorithms* are nothing but preprocessing algorithms that create an instance with the size that is finite and bounded by a computable function of the parameter. Also, observe that, if there is a kernelization algorithm for a problem for which there is an algorithm that decides whether (I, k) is a YES-instance, then the problem is FPT because the size of the reduced instance of I is just a function of k which is independent of the input size n. However, there is a surprising result that says that the converse is also true.

Lemma 61 (Lemma 2.2 [19])**.** *If a parameterized problem Q is* FPT *then it admits a kernelization algorithm.*

Lemma 61 implies that a decidable problem admits a kernel if and only if it is *fixed-parameter tractable*. Therefore, *kernelization* can be thought of as another way of defining fixed-parameter tractability. Also, observe that, the VERTEX COVER problem presented as an example at the start of this Section 5.5 is an evidence of Lemma 61. That is, the (G, k) instance of the VERTEX COVER problem is FPT and admits a *kernel* with $O(k^2)$ vertices and $O(k^2)$ edges.

5.5.1 Compositionality

For algorithmic problems, usually, we are more interested in the running time of the algorithm. However, in case of *kernelization* the focus is on the size of the output ($\mathcal{A}(I, k)$). Needless to say, the running time of the *kernelization* algorithm is important, but, in theory, it is required to be polynomial. Kernelization algorithms that give good guarantees on the output size (say, polynomial) are considered to be interesting and the designers aim for that mostly. Thus, a natural question to ask is, under what assumptions and what conditions can we rule out the possibility of the existence of a polynomial kernel for a particular FPT problem? And one of the ways to answer this question is via the framework of *compositionality*. This framework made it possible to systematically classify the problems depending on whether they admit polynomial kernels or not [19]. Very briefly the framework can be described as follows:

- Consider multiple instances of a problem L (say, $x_1, x_2, \ldots, x_t$), where, L is a unparameterized version of some parameterized problem Q
- Compute a single instance $(x*, k*)$ of Q in time polynomial in $\sum_{i=1}^{t} |x_i|$
- Try to show that $(x^*, k^*) \in Q$ if and only if $x_i \in L$ for some $1 \leq i \leq t$

We now present the formal definition of *cross-composition* and use it to show that the Firefighting problem – SAVING A CRITICAL SET does not admit polynomial kernel when restricted to trees. Before that, we first define *polynomial equivalence relations*, which would be used in the *cross-composition* framework.

Definition 62 (Polynomial equivalence relation [7]): *An equivalence relation $\mathcal{R}$ on Σ^*, where Σ is a finite alphabet, is called a* polynomial equivalence relation *if the following holds:*

1. *equivalence of any $x, y \in \Sigma^*$ can be checked in time polynomial in $|x| + |y|$, there is an algorithm that given two strings $x, y \in \Sigma^*$ decides whether x and y belong to the same equivalence class in $(|x| + |y|)^{O(1)}$ time, and*
2. *any finite set $S \subseteq \Sigma^*$ has at most $(\max_{x \in S} |x|)^{O(1)}$ equivalence classes.*

Definition 63 (Cross-composition [7]): *Let $L \subseteq \Sigma^*$ and let $Q \subseteq \Sigma^* \times \mathbb{N}$ be a parameterized problem. We say that L* cross-composes *into Q if there is a polynomial*

equivalence relation $\mathcal{R}$ and an algorithm which, given t strings $x_1, x_2, \ldots, x_t$ belonging to the same equivalence class of $\mathcal{R}$, computes an instance $(x^*, k^*) \in \Sigma^* \times \mathbb{N}$ in time polynomial in $\sum_{i=1}^{t} |x_i|$ such that:

1. *$(x^*, k^*) \in Q \Leftrightarrow x_i \in L$ for some $1 \leq i \leq t$, and*
2. *k^* is bounded by a polynomial in $(\max_{1 \leq i \leq t} |x_i| + \log t)$.*

The following theorem allows us to rule out the existence of a polynomial kernel for a parameterized problem.

Theorem 64 ([7]). *If an* NP-hard *problem $L \subseteq \Sigma^*$ has a cross-composition into the parameterized problem Q and Q has a polynomial kernel then NP $\subseteq$ coNP/poly.*

5.5.2 No Polynomial Kernel, Even on Trees

Given that there is an FPT algorithm for SACS when restricted to trees, here we show that SACS on trees has no polynomial kernel using the framework of *compositionality*. The proof technique used here is on the similar lines of the proof showing no polynomial kernel for *SAVING ALL BUT k-VERTICES* by Bazgan et. al.[6].

Theorem 65 (Choudhari et al. [18]). *SACS when restricted to trees does not admit polynomial kernel, unless NP $\subseteq$ coNP/poly.*

To prove this theorem, we will use the Definitions (62, 63) mentioned in section 5.3. We use Theorem 64, for which we consider SACS on trees as analogous to language L, which is shown to be NP-complete when the critical set C is the set of all leaves [53]. First we give a lemma that we will be using in the proof.

Lemma 66 (Choudhari et al. [18]). *For a given instance of SACS (T, r, C, k), where T is a full binary tree with root r, height h, and $k = h$, if more than one vertex is protected at a depth $d \leq h$, then more than one leaf burns.*

Proof. Consider a case when more than one firefighter is placed at a depth $d \leq h$. It is easy to see that at most 1 firefighter can be placed at depth 1. Therefore, more than 1 firefighter will always be at depth $d \geq 2$. Also, at any depth $d \geq 1$, there are 2^d nodes, which is strictly greater than the number of odd time steps at which the firefighters are allowed to be placed. This says that all the nodes at a particular depth $d \geq 1$ cannot be protected by firefighters. Therefore, at any depth $d \geq 2$, there are at least two subtrees that are unprotected and can be burnt. And thus, given the constraint that at most one firefighter can be placed at any allowed time step, there is at least one subtree that is burnt and can never be completely protected/saved by the firefighters.

However, one of the strategies to let only one leaf burn is to fix a path from the root to a leaf and keep protecting the siblings of the nodes on that path. This will be clearer as we go ahead in the proof of the current theorem. □

Lemma 67 (Choudhari et al. [18])**.** *The unparameterized version of SACS restricted to trees cross composes to SACS restricted to trees when parameterized by the number of firefighters.*

Proof. We take an appropriate equivalence relation $\mathcal{R}$ such that $\mathcal{R}$ puts all the malformed instances into one class and all the well-formed instances are grouped into equivalence classes according to the number of vertices of the tree, and number of firefighters (parameter k) required to save the critical set C. We assume that we are given a sequence of t instances $(T_i, s_i, C_i, k)_{i=1}^{t}$ of the unparameterized version of SACS restricted to trees, each rooted at s_i. Note that each of the t instances belongs to the same equivalence class i.e. for all the instances k is the same. Also, consider that $t = 2^h$, for some $h \geq 1$, else, we duplicate some instances and the duplication at most doubles the number of instances.

Using these t instances, we create a new full binary tree T' as follows. Let T' be rooted at s', and h be the height of T' ($2^h = t$). For each leaf $i \in \{1, \ldots, t\}$, replace the i^{th} leaf by the instance (T_i, s_i, C_i, k) and now, set $k' = k + h = k + \log t$. Observe that, for T' the set of all leaves is the union of the leaves of the instances T_i i.e. $\cup_{i=1}^{t} C_i$.

To prove the correctness, we show that the tree T' formed by the composition of t instances saves all the leaves with k' firefighters if and only if there exists at least one instance T_i for $i \in \{1, 2, \ldots, t\}$, that saves its critical set C_i (i.e. the set of all leaves) with k firefighters.

Suppose, T' has a successful firefighting strategy. Then, from lemma 66, it follows that the firefighting strategy that saves all but one root s_i of T_i is the one that protects exactly one vertex at each depth of the tree. This costs exactly $\log(t) = h$ firefighters. Thus, after $2h$ time steps, there is exactly one vertex s_i (the root of instance T_i) which is on fire. And the critical set C_i for the instance (T_i, s_i, C_i, k) is saved using k firefighters.

Now suppose, there is an instance (T_i, s_i, C_i, k) that has a successful firefighting strategy with k firefighters. Thus, the leaves C_i of this instance are saved by the firefighting strategy with k firefighters. Now, the goal is to save all other leaves $(\cup_{j=1}^{t} C_j) \setminus C_i$. T' being a tree, there is an unique path from the root s to the node s_i (i.e. the root of the instance T_i). Denote the path as $P = (s, v_1, v_2, \ldots, v_{\log(t)-1}, s_i)$. Note that each node v_j is at depth j and s_i is at depth $\log(t)$. Let $u_1, u_2, \ldots, u_{\log(t)-1}, s_j$ be the siblings of the nodes $v_1, v_2, \ldots, v_{\log(t)-1}, s_i$ on the path P respectively. Now, the firefighting strategy that protects the sibling u_j of node v_j at time step $2j - 1$ and sibling s_j of the node s_i at time step $2h - 1 = 2\log(t) - 1$ saves all other leaves of T'. □

5.6 Fixed Parameter Intractability

A natural question to ask about the FPT algorithms is: *Can we always have a* FPT *algorithm for a problem for any choice of the parameter?* And the answer is "No".

Let us now give an example of a parameterized problem that is unlikely to be *fixed-parameter tractable*. Consider the Vertex Coloring problem, where, given a graph G and an integer k the aim is to answer whether G has proper k-coloring. A k-coloring is said to be proper if for every edge (u, v), both u and v are colored differently and the total number of colors used is at most k. But, it is well-known that Vertex Coloring is a NP-complete for $k \geq 3$. Which means that it is NP-complete even for a small value of k (say, k = 5). Therefore, if we expect that Vertex Coloring, say for $k = 5$, has a FPT algorithm with the time complexity of the form $f(k) \cdot n^c$, then this would imply $P = NP$. Note that, expecting even a XP algorithm, where the running time is of the form $f(k) \cdot n^{g(k)}$ would again imply $P = NP$.

Therefore, from the Vertex Coloring problem it is clear that there are parameterized problems that do not admit FPT algorithms. But the question is can we explain precisely why were we not able to give an efficient algorithm even when the number of colors is small. In complexity theory, if we are unable to design a polynomial-time algorithm for a problem then it is quite likely it is NP-hard. So, on similar lines, can we use the theory of NP-hardness to show negative evidence for the FPT algorithm. If we are unable to get an algorithm for a parameterized problem with the time complexity of the form $f(k) \cdot n^c$, is it because the problem is NP-hard for some fixed constant value of the parameter k, say $k = 5$ or $k = 100$? Let us look at another example – the Clique problem.

In the Clique problem, given a graph G, the goal is to find a set of k vertices that form a *clique*, that is, a set of k vertices such that there is an edge between every pair of them. One of the easy $O(n^k)$ algorithms is to check for a clique on at least k vertices in $O(k^2)$ time, for every $\binom{n}{k}$ subset of vertices of size k. This algorithm has a time complexity of the form $f(k) \cdot n^{g(k)}$ and is an XP algorithm. Running this algorithm for say $n = 10000$ and $k = 5$ is kind of hopeless. Till now there is no evidence of a FPT algorithm for this problem. Also, it is clear from the $O(n^k)$ XP algorithm that the *k-clique* problem is not NP-hard for a fixed value of k. Because, if the Clique problem had been NP-hard for a fixed value of k, then the $O(n^k)$ algorithm, which is polynomial-time algorithm would imply $P = NP$. Thus, this says that using the theory of NP-hardness in this way is not sufficient to rule out the possibility of a FPT algorithm for a Clique problem. That is, the problem being NP-hard is not enough to explain the existence of an XP algorithm but no FPT algorithm.

As the theory of NP-hardness lacks in differentiating between the problems with time complexity $f(k) \cdot n^{g(k)}$ and $f(k) \cdot n^c$ the hierarchy of classes of parameterized problems, FPT $\subseteq W[1] \subseteq W[2] \ldots W[SAT] \subseteq W[P] \subseteq XP$, i.e. the *W-hierarchy* was introduced by Downey and Fellows [26, 24, 25]. As compared to the aspect of NP-completeness where there are a huge number of equally hard problems that are reducible to each other, in the case of parameterized problems the situation is different. Even the basic parameterized problems such as Clique (W[1]) and Dominating set (W[2]) seem to occupy a different level of hardness and thus the notion of *hierarchy* was proposed. This theory of *W-hierarchy* helps us to prove that the problem is not FPT even if it is polynomial-time solvable for a fixed value of the parameter k, under certain complexity

assumptions, and also helps to categorize the problem in the hierarchy depending on its hardness. As similar to the notion of *polynomial-time reduction* which is used in the proofs of NP-hardness, there is an analogous notion of reduction for parameterized problems.

Definition 68 (Parameterized Reduction [19]): *Let $A, B \subseteq \Sigma^* \times \mathbb{N}$ be two parameterized problems. A parameterized reduction from A to B is an algorithm that, given an instance (x, k) of A, outputs an instance (x', k') of B such that*

1. *(x, k) is a yes-instance of A if and only if (x', k') is a yes-instance of B,*
2. *$k' \leq g(k)$ for some computable function g, and*
3. *the running time is $f(k) \cdot |x|^{O(1)}$ for some computable function f.*

Given this definition, the following theorems about the parameterized reduction from one problem to another are analogous to that of polynomial-time reduction in case of NP-hardness.

Theorem 69 ([19])**.** *If there is a parameterized reduction from A to B and B is* FPT*, then A is* FPT *as well.*

Theorem 70 ([19])**.** *If there are parameterized reductions from A to B and from B to C, then there is a parameterized reduction from A to C.*

To have a better understanding of parameterized reductions let us describe some examples here.

- **Independent Set and Clique**: Observe that, an independent set in a graph G is a clique in the complementary graph $\overline{G}$. Therefore, the size of the maximum independent set in the graph is the same as the size of the maximum clique in the complementary graph. Note that, given an instance (G, k) of Independent set, creating an instance $(\overline{G}, k)$ of Clique is polynomial-time reduction. Moreover, this is a parameterized reduction, where $g(k) = k$ is a trivial bound on the new parameter as $k' = k$. Also, a similar parameterized reduction works in other direction as well, i.e. from Clique to Independent Set. Therefore, both these problems are equally hard and we can say that one of them is FPT if and only if the other is.
- **Independent Set and Vertex Cover**: There is a nice connection between the independent set in the graph and the vertex cover. A very well known fact is a graph G with n vertices has an independent set of size k if and only if it has a vertex cover of size $n - k$. Thus, (G, k) is a Yes-instance of Independent set if and only if $(G, n - k)$ is a Yes-instance of Vertex Cover. But notice that, this reduction is a polynomial-time reduction but not a parameterized reduction. Here, the parameter (k') for the Vertex Cover problem is $n - k$

which is not just a function of k and cannot be bounded by some function $g(k)$ given the dependence on n. Also, recall that the VERTEX COVER problem is FPT when parameterized by the size of the vertex cover. Thus, the existence of a parameterized reduction from INDEPENDENT SET to VERTEX COVER would imply (Theorem 69) that INDEPENDENT SET is FPT as well, which is not true.

Given this understanding of the parameterized intractability, we present an intractability result for a variant of the Firefighting problem. In this problem, which we call as *the Spreading model*, as similar to the fire, the firefighter "spreads" as well. That is, at every even time step, if a firefighter is present at some node v, then it extends to all neighbors of v. In the next section, we describe the problem more formally and present the intractability result.

5.6.1 The Spreading Model

The spreading model for firefighters was defined by Anshelevich et al. [1] as "Spreading Vaccination Model". In contrast to the firefighting game described in Section 5.2, in the spreading model, the firefighters (vaccination) also spread at even time steps as similar to that of the fire. That is, at any even time step if there is a firefighter at node v_i, then the firefighter extends (vaccination spreads) to all the neighbors of v_i which are not already on fire or are not already protected by a firefighter. Consider a node v_i which is not already protected or burning at a time step $2j$. If u_i and w_i are neighbors of v_i, such that, u_i was already burning at time step $2j-1$ and w_i was protected at time step $2j-1$, then at time step $2j$, v_i is protected. That is, in the spreading model the firefighters dominate or win over the fire. For the spreading model, the firefighting game can be defined formally as follows:

- At time step 0, fire breaks out at the vertex s. A vertex on fire is said to be *burned*.
- At every odd time step $i \in \{1, 3, 5, \ldots\}$, when it is the turn of the firefighter, a firefighter is placed at a vertex v that is not already on fire. Such a vertex is permanently *protected*.
- At every even time step $j \in \{2, 4, 6, \ldots\}$, first the firefighter extends to every adjacent vertex to a vertex protected by a firefighter (unless it was already protected or burned), then the fire spreads to every vertex adjacent to a vertex on fire (unless it was already protected or burned). Needless to say, the vertices protected at even time steps are also permanently *protected*.

In the following theorem, we show that in spite of the spreading power that the firefighters have, SACS is hard.

Theorem 71 (Choudhari et al. [18]). *In the spreading model, SACS is as hard as k-DOMINATING SET.*

Proof. Let (G, k) be an instance of k-Dominating Set problem. We construct a graph G' as follows. Add 2 copies $V^{(1)}$ and $V^{(2)}$ of the nodes $V(G)$ in G', i.e. for each node $v_i \in V(G)$ add nodes $v_i^{(1)}$ and $v_i^{(2)}$. Add a vertex s, the vertex from where the fire breaks out. For each vertex $v_i^{(1)} \in V^{(1)}$, add a path of length k from s (i.e. a path from s to $v_i^{(1)}$ would be like $(s, u_{i_1}, \ldots, u_{i_{k-1}}, v_i^{(1)})$). Similarly, for each edge $(v_i, v_j) \in E(G)$, add a path of length k from $v_i^{(1)}$ to $v_j^{(2)}$, and a path of length $2k$ from $v_i^{(2)}$ to $v_j^{(1)}$ in G'. Also, for each $v_i \in V(G)$ add a path of length $2k$ from $v_i^{(1)}$ to $v_i^{(2)}$. Let $C = V^{(2)}$ be the critical set and $k' = k$.

We claim that, SACS on $(G', s, k', C = V^{(2)})$ is a Yes-instance if and only if k-Dominating Set is a Yes instance on (G, k).

Suppose that, G has a k-dominating set K. Then the strategy that saves the critical set C is the one that protects the vertices $v_i^{(1)} \in V^{(1)}$ corresponding to the vertices v_i in the dominating set K. The nodes $v_i \in K$ being the dominating set, the corresponding nodes $v_i^{(1)} \in V^{(1)}$ dominate all the nodes $v_j^{(2)} \in V^{(2)}$. And thus, the firefighters on the nodes $v_i^{(1)} \in V^{(1)}$ corresponding to the dominating set K, eventually extend firefighters to all the nodes in $V^{(2)}$ before fire reaches to any node $v_j^{(2)} \in V^{(2)}$. Also, protecting the vertices $v_i^{(1)} \in V^{(1)}$ corresponding to $v_i \in K$ is a valid firefighting strategy as per the Definition 28.

Suppose that $\mathfrak{h} : [k] \to V(G')$ is an optimal firefighting strategy for $(G', s, k', C = V^{(2)})$. Let $S = \{u_1, \ldots, u_k\}$ be the set of vertices which are protected by the firefighting strategy, i.e. $\mathfrak{h}(i) = u_i$ for $i \in [k]$. Note that, as per the definition of the firefighting game, i^{th} firefighter is placed at a time step $2i - 1$. Lets denote the paths from the nodes in $V^{(1)}$ to the nodes in $V^{(2)}$ as P_q for $q \in [1, 2m + n]$. Observe that, an optimal firefighting strategy would not have (1) two distinct firefighters u_i and u_j for $i \neq j$ on the same path P_q and (2) two distinct firefighters u_i and u_j for $i \neq j$ on the paths P_{q_1} and P_{q_2} that are incident on the same node $v_i^{(1)} \in V^{(1)}$. As for both the conditions, it is better to have one firefighter on the node $v_i^{(1)} \in V^{(1)}$ on which the paths incident. Hence, we can assume that at most one firefighter is placed on any path P_q and at most one firefighter is placed on the paths incident on vertex $v_i^{(1)}$. Moreover, if a firefighter is placed at any node p_i on a path P_q between $v_i^{(1)} \in V^{(1)}$ and $v_j^{(2)} \in V^{(2)}$, then, in $2k$ time steps it can extend protections to none other than $v_i^{(1)} \in V^{(1)}$, $v_j^{(2)} \in V^{(2)}$, and the intermediate nodes on the path P_q. It can be said that any node on a path P_q has exactly one vertex $v_i^{(1)} \in V^{(1)}$ within in a diameter of $2k$. Thus, placing a firefighter on any path P_q is equivalent to placing a firefighter on the vertex $v_i^{(1)}$ on which the path P_q is incident. Therefore, in the firefighting strategy S, if u_i is a node on a path P_q which is between $v_x^{(1)} \in V^{(1)}$ and $v_y^{(2)} \in V^{(2)}$, we push back the firefighter to $v_x^{(1)} \in V^{(1)}$. As the fire reaches $V^{(1)}$ in $2k$ time steps, there can be at most k nodes $v_i^1 \in V^{(1)}$ on which the firefighters are placed at the allowed time steps. Therefore, there is at least

one vertex $v_i^{(1)} \in V^{(1)}$ that is burnt. And thus, if the firefighters are not placed on the vertices $v_i^{(1)} \in V^{(1)}$ that form a dominating set in G, then there is at least one path to a node $v_j^{(2)} \in V^{(2)}$ at which the firefighters cannot extend protection. □

Chapter 6

Conclusions and Future Work

With the goal of understanding information diffusion in social networks, we have proposed two generative models, viz. *Hidden Markov Hawkes Process* (HMHP) and *Dual Network Hawkes Process* DNHP that model cascades of information diffusion. Both the models account for topical interactions along with user-user influence and user-topic patterns. As opposed to HMHP and other state-of-the-art models, in DNHP, the rate of generation of events by a user is dependent not only on the user but also on the topic or the mark associated with the event. In DNHP the topical interactions and the user-user influence weights are coupled and the joint estimation of these parameters enables the flow of evidence across these parameters. In case of HMHP, empirical results show that the use of topical interactions and collective inference leads to a more accurate reconstruction of network strengths, diffusion paths and event topics on the semi-synthetic datasets; it also fits the real *Twitter* conversations better (in terms of test likelihood) as compared to other state-of-the-art models.

For DNHP, modeling over the dual space of users and topics along with the topical interactions and joint inference play a crucial role in outperforming the state-of-the-art models in terms of reconstruction and the generalization performances.

This thesis also makes an effort to develop algorithmic approaches to moderate the flow of (mis-)information in social networks; we have presented one such algorithm and characterized its complexity in the parameterized complexity setting. This problem was formalized as the Firefighting problem, where the spread of fire on the graphs is analogous to that of the spread of an infection in the networks. However, in the Firefighter problem, the flow of fire or infection is considered to be deterministic and thus the fire flows to all the unprotected neighbors of the node in the next time step once the node is on fire. Keeping this setting in mind, we defined the problem *Saving a Critical Set* (SACS) where the goal is to protect a set of nodes from the fire with the help of at most k firefighters; k i.e. the maximum number of firefighters is the parameter of the problem. We showed that the problem is para-NP-complete even when the size of the critical set is 1. We proposed a faster algorithm with time complexity $O^*(4^k)$ for SACS when the problem is restricted to trees. However, it turns out that SACS does not admit a polynomial kernel for trees. Additionally, we

present a FPT algorithm for SACS on general graphs with running time complexity $f(k) \cdot O(n^2 m)$ where, $f(k) = k^{O(polyk)}$.

6.1 Future Work

In the case of both HMHP and DNHP, the number of distinct topics that is used to fit the data is considered to be fixed. However, in the real world, the number of topics increases as more and more data is seen. Therefore, incorporating an infinite number of topics (Bayesian Nonparametric) in the models is one of the potential future directions. Currently, the held-out log-likelihood calculation for both HMHP and DNHP is based on a heuristic. Calculating the exact log-likelihood is hard to the best of our knowledge. Surprisingly, despite a wealth of literature on cascade modeling, this question of calculation of log-likelihood for cascade models has not been studied carefully. Thus, designing an efficient algorithm for the calculation of log-likelihood exactly or approximately with theoretical guarantees is one of the potential future questions that we wish to answer.

Additionally, the prior over the topical interaction network is considered to be a complete graph both in HMHP and DNHP. Currently, we have no knowledge of appropriate priors that can incorporate structure over the topics. However, we believe that incorporating structural knowledge about topics (or marks) can help in improving the existing knowledge-bases (e.g. DBpedia) using the cascade evidence from the models. Furthermore, we intend to study the question of whether inferring topical interactions in a text even-stream corpus, using DBPedia as a prior can, in turn, suggests missing edges in DBPedia by identifying topics that frequently influence one another.

In the case of the Firefighter problem, in all studied literature, the spread of fire is considered to be deterministic. However, we believe that for the model to be made more realistic and interesting it will be appropriate to study the variant of the problem that considers a probabilistic spread of fire. Furthermore, right now the graph on which the Firefighting problem is played is considered to be a static graph. We think that it will also be interesting to define and consider the Firefighting problem when the edges in the graph are added or deleted over time.

Furthermore, it seems possible to further study the Firefighter question in the Bayesian information propagation setting that is used in HMHP and DNHP. We could perhaps model the “fire” propagation as depending on user interests and influence, and “placing a firefighter” can correspond to exposing the user to alternate points of view (i.e. changing her vector of topic affinities), thereby moderating her propensity of “catching fire”. It will be interesting to see whether the algorithmic machinery developed in the firefighting literature can be adapted to be used in such settings.

Appendix A

Hidden Markov Hawkes Process (HMHP)

A.1 Conditional Probabilities for Gibbs Sampling

A.1.1 Topic Assignment Conditional Probability

The probability of assigning a topic k to an event e conditional on all other parameters is:

$$\begin{aligned} &P(\eta_e = k \mid \boldsymbol{\eta}_{\neg e}, \boldsymbol{z}, W, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\mathcal{W}}, \boldsymbol{\mu}) \\ &\qquad = P(\eta_e = k \mid \boldsymbol{w}_e, \boldsymbol{\eta}_{\overline{\mathfrak{D}}_e}, \boldsymbol{\eta}_{\mathfrak{D}_e}, \boldsymbol{z}, W_{\overline{\mathfrak{D}}_e}, W_{\mathfrak{D}_e}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}) \end{aligned} \tag{A.1}$$

Where, $\boldsymbol{w}_e$ is the vector of all words that belong to document of event e, $W_{\overline{\mathfrak{D}}_e}$ and $W_{\mathfrak{D}_e}$ are the words belonging to the documents those are not descendants of event e and the words that are in the documents corresponding to the descendants of event e. $\boldsymbol{\eta}_s$ are defined on the similar lines. The influence matrix $\boldsymbol{\mathcal{W}}$ and the base intensities $\boldsymbol{\mu}$ do not play any role in topic assignment and hence can be removed from the conditioning part. Using Bayesian rule, the above equation can be written as:

$$\begin{aligned} &P(\eta_e = k \mid \boldsymbol{w}_e, \boldsymbol{\eta}_{\overline{\mathfrak{D}}_e}, \boldsymbol{\eta}_{\mathfrak{D}_e}, \boldsymbol{z}, W_{\overline{\mathfrak{D}}_e}, W_{\mathfrak{D}_e}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}) \\ &\qquad \propto P(\boldsymbol{\eta}_{\overline{\mathfrak{D}}_e}, \eta_e = k, \boldsymbol{\eta}_{\mathfrak{D}_e}, W_{\overline{\mathfrak{D}}_e}, \boldsymbol{w}_e, W_{\mathfrak{D}_e} | \boldsymbol{z}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}) \\ &\qquad = P(W_{\overline{\mathfrak{D}}_e}, \boldsymbol{w}_e, W_{\mathfrak{D}_e} | \boldsymbol{\eta}, \boldsymbol{z}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}) \times P(\boldsymbol{\eta}_{\overline{\mathfrak{D}}_e}, \eta_e = k, \boldsymbol{\eta}_{\mathfrak{D}_e} | \boldsymbol{z}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}) \end{aligned} \tag{A.2}$$

Now, considering $z_e = e'$, $\eta_{e'} = k'$ and solving each of terms from the above equation

individually (ignoring the terms related to γ for simplicity):

$$P(\boldsymbol{\eta}_{\overline{\mathfrak{D}}_e \setminus \{e'\}}, \eta_{e'} = k', \eta_e = k, \boldsymbol{\eta}_{\mathfrak{D}_e} \mid \boldsymbol{z}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}) = \int_{\boldsymbol{\mathcal{T}}} P(\boldsymbol{\mathcal{T}}, \boldsymbol{\eta}_{\overline{\mathfrak{D}}_e}, \eta_e = k, \boldsymbol{\eta}_{\mathfrak{D}_e} | \boldsymbol{z}) d\boldsymbol{\mathcal{T}}$$

$$= \int_{\boldsymbol{\mathcal{T}}} P(\boldsymbol{\mathcal{T}}; \boldsymbol{\beta}) \cdot P(\boldsymbol{\eta}_{\overline{\mathfrak{D}}_e} | \boldsymbol{z}, \boldsymbol{\mathcal{T}}) \cdot P(\eta_e = k | \boldsymbol{\eta}_{\overline{\mathfrak{D}}_e}, \boldsymbol{z}, \boldsymbol{\mathcal{T}}) \cdot P(\boldsymbol{\eta}_{\mathfrak{D}_e} | \boldsymbol{\mathcal{T}}, \boldsymbol{\eta}_{\overline{\mathfrak{D}}_e}, \eta_e = k, \boldsymbol{z}) d\boldsymbol{\mathcal{T}}$$

$$= \int_{\boldsymbol{\mathcal{T}}} \prod_{l=1}^{K} P(\boldsymbol{\mathcal{T}}_l; \boldsymbol{\beta}) \times \prod_{l=1}^{K} \prod_{l'=1}^{K} \mathcal{T}_{l,l'}^{N_{l,l'}^{(\overline{\mathfrak{D}}_e)}} \times \mathcal{T}_{k',k}^{(\mathfrak{p}_e)} \times \prod_{m'=1}^{K} \mathcal{T}_{k,m'}^{N_{k,m'}^{(C_e)}} \times \prod_{l=1}^{K} \prod_{l'=1}^{K} \mathcal{T}_{l,l'}^{N_{l,l'}^{(\overline{C}_e)}} d\boldsymbol{\mathcal{T}}$$

The fourth term here corresponds to the events those are child events of event e.

$$= \int_{\boldsymbol{\mathcal{T}}} \left(\prod_{l=1}^{K} \frac{1}{B(\boldsymbol{\beta})} \prod_{l'=1}^{K} \mathcal{T}_{l,l'}^{\beta_{l'}-1} \right) \left(\prod_{\substack{l=1 \\ l \neq k}}^{K} \prod_{l'=1}^{K} \mathcal{T}_{l,l'}^{N_{l,l'}^{(\neg[C_e])}} \right) \mathcal{T}_{k',k}^{(\mathfrak{p}_e)} \left(\prod_{m'=1}^{K} \mathcal{T}_{k,m'}^{N_{k,m'}^{(\overline{\mathfrak{D}}_e)} + N_{k,m'}^{(C_e)} + N_{k,m'}^{(\overline{C}_e)}} \right) d\boldsymbol{\mathcal{T}} \tag{A.3}$$

In the second equality of the equation above $\mathcal{T}_{k',k}^{(\mathfrak{p}_e)}$ indicate the probability of event with topic k' (parent event of event e) influencing event with topic k (event e) i.e. probability over the edge (z_e, e); and the count of the edges in the set $\mathfrak{D}_e$ (descendants of event e) are split into two disjoint sets: 1) the count of edges over child events of event e and 2) non-child events of event e. In the third equality the terms from $\overline{\mathfrak{D}}_e$ and $\overline{C}_e$ are combined to give a term $(\neg[C_e])$ (i.e. the term $\mathcal{T}_{l,l'}^{N_{l,l'}^{(\neg[C_e])}}$).
Therefore,

$$P(\boldsymbol{\eta}_{\overline{\mathfrak{D}}_e \setminus \{e'\}}, \eta_{e'} = k', \eta_e = k, \boldsymbol{\eta}_{\mathfrak{D}_e} | \boldsymbol{z}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}) =$$

$$\int_{\boldsymbol{\mathcal{T}}} \left(\prod_{l=1}^{K} \frac{1}{B(\boldsymbol{\beta})} \prod_{l'=1}^{K} \mathcal{T}_{l,l'}^{\beta_{l'}-1} \right) \cdot \left(\prod_{\substack{l=1 \\ l \neq k}}^{K} \prod_{l'=1}^{K} \mathcal{T}_{l,l'}^{N_{l,l'}^{(\neg[C_e])}} \right) \cdot \mathcal{T}_{k',k}^{(\mathfrak{p}_e)} \cdot \left(\prod_{m'=1}^{K} \mathcal{T}_{k,m'}^{N_{k,m'}^{(\overline{\mathfrak{D}}_e)} + N_{k,m'}^{(C_e)} + N_{k,m'}^{(\overline{C}_e)}} \right) d\boldsymbol{\mathcal{T}}$$

$$= \left(\prod_{\substack{l=1 \\ l \neq \{k,k'\}}}^{K} \int_{\boldsymbol{\mathcal{T}}_l} \frac{1}{B(\boldsymbol{\beta})} \prod_{l'=1}^{K} \mathcal{T}_{l,l'}^{\beta_{l'}-1+N_{l,l'}} d\boldsymbol{\mathcal{T}}_l \right)$$

$$\times \int_{\boldsymbol{\mathcal{T}}_{k'}} \left(\frac{1}{B(\boldsymbol{\beta})} \prod_{l'=1}^{K} \mathcal{T}_{k',l'}^{\beta_{k'}-1+N_{k',l'}^{(\neg[C_e])}} \right) \times \mathcal{T}_{k',k} \, d\boldsymbol{\mathcal{T}}_{k'}$$

$$\times \int_{\boldsymbol{\mathcal{T}}_k} \frac{1}{B(\boldsymbol{\beta})} \prod_{m'=1}^{K} \mathcal{T}_{k,m'}^{\beta_{m'}-1+N_{k,m'}^{(\overline{\mathfrak{D}}_e)} + N_{k,m'}^{(C_e)} + N_{k,m'}^{(\overline{C}_e)}} d\boldsymbol{\mathcal{T}}_k$$

$$
\begin{aligned}
&\propto \int_{\mathcal{T}_{k'}} \left(\frac{1}{B(\boldsymbol{\beta})} \prod_{l'=1}^{K} \mathcal{T}_{k',l'}^{\beta_{k'}-1+N_{k',l'}^{(\neg[C_e])}} \right) \times \mathcal{T}_{k',k}\, d\boldsymbol{\mathcal{T}}_{k'} \\
&\qquad \times \int_{\boldsymbol{\mathcal{T}}_k} \frac{1}{B(\boldsymbol{\beta})} \prod_{m'=1}^{K} \mathcal{T}_{k,m'}^{\beta_{m'}-1+N_{k,m'}^{(\overline{\mathfrak{D}}_e)}+N_{k,m'}^{(C_e)}+N_{k,m'}^{(\overline{C}_e)}} d\boldsymbol{\mathcal{T}}_k \\
&= \frac{B(\boldsymbol{\beta}+\boldsymbol{N}_{k'}^{(\neg[C_e])}+1)}{B(\boldsymbol{\beta})} \times \frac{B(\boldsymbol{\beta}+\boldsymbol{N}_{k}^{(\overline{\mathfrak{D}}_e)}+\boldsymbol{N}_{k}^{(\overline{C}_e)}+\boldsymbol{N}_{k}^{(C_e)})}{B(\boldsymbol{\beta})}
\end{aligned} \tag{A.4}
$$

The terms related to the topics other than k and k' cancel out because of the denominator. The denominator for the equation A.2 is similar to that of the numerator with the topic of event e excluded, and thus, can be written as:

$$
\begin{aligned}
&P(\boldsymbol{\eta}_{\overline{\mathfrak{D}}_e \setminus \{e'\}}, \eta_{e'}=k', \eta_e=k, \boldsymbol{\eta}_{\mathfrak{D}_e} | \boldsymbol{z}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}) \\
&\propto \frac{\dfrac{B(\boldsymbol{\beta}+\boldsymbol{N}_{k'}^{(\neg[C_e])}+1)}{B(\boldsymbol{\beta})}}{\dfrac{B(\boldsymbol{\beta}+\boldsymbol{N}_{k'}^{(\neg[C_e])})}{B(\boldsymbol{\beta})}} \times \frac{\dfrac{B(\boldsymbol{\beta}+\boldsymbol{N}_{k}^{(\overline{\mathfrak{D}}_e)}+\boldsymbol{N}_{k}^{(\overline{C}_e)}+\boldsymbol{N}_{k}^{(C_e)})}{B(\boldsymbol{\beta})}}{\dfrac{B(\boldsymbol{\beta}+\boldsymbol{N}_{k}^{(\overline{\mathfrak{D}}_e)}+\boldsymbol{N}_{k}^{(\overline{C}_e)})}{B(\boldsymbol{\beta})}} \\
&= \frac{\dfrac{\prod_{\substack{l=1 \\ l\neq k}}^{K} \Gamma(\beta_l + N_{k',l}^{(\neg[C_e])}) \times \Gamma(\beta_k + N_{k',k}^{(\neg[C_e])}+1)}{\Gamma((\sum_l \beta_l)+\boldsymbol{N}_{k'}^{(\neg[C_e])}+1)}}{\dfrac{\prod_{\substack{l=1 \\ l\neq k}}^{K} \Gamma(\beta_l + N_{k',l}^{(\neg[C_e])}) \times \Gamma(\beta_k + N_{k',k}^{(\neg[C_e])})}{\Gamma((\sum_l \beta_l)+\boldsymbol{N}_{k'}^{(\neg[C_e])})}} \cdot \frac{\dfrac{\prod_{l'=1}^{K} \Gamma(\beta_{l'} + N_{k,l'}^{(\overline{\mathfrak{D}}_e)} + N_{k,l'}^{(\overline{C}_e)} + N_{k,l'}^{(C_e)})}{\Gamma((\sum_{l'} \beta'_l)+\boldsymbol{N}_{k}^{(\overline{\mathfrak{D}}_e)}+\boldsymbol{N}_{k}^{(\overline{C}_e)}+\boldsymbol{N}_{k}^{(C_e)})}}{\dfrac{\prod_{l'=1}^{K} \Gamma(\beta_{l'} + N_{k,l'}^{(\overline{\mathfrak{D}}_e)} + N_{k,l'}^{(\overline{C}_e)})}{\Gamma((\sum_{l'} \beta'_l)+\boldsymbol{N}_{k}^{(\overline{\mathfrak{D}}_e)}+\boldsymbol{N}_{k}^{(\overline{C}_e)})}} \\
&= \frac{\dfrac{\Gamma(\beta_k + N_{k',k}^{(\neg[C_e])}+1)}{\Gamma((\sum_l \beta_l)+\boldsymbol{N}_{k'}^{(\neg[C_e])}+1)}}{\dfrac{\Gamma(\beta_k + N_{k',k}^{(\neg[C_e])})}{\Gamma((\sum_l \beta_l)+\boldsymbol{N}_{k'}^{(\neg[C_e])})}} \times \frac{\prod_{l'=1}^{K} \prod_{i=1}^{N_{k,l'}^{(C_e)}} (\beta_{l'} + N_{k,l'}^{(\overline{\mathfrak{D}}_e)} + N_{k,l'}^{(\overline{C}_e)} + N_{k,l'}^{(C_e)} - i)}{\prod_{i=1}^{\boldsymbol{N}_k^{(C_e)}} ((\sum_{l'} \beta'_l)+\boldsymbol{N}_{k}^{(\overline{\mathfrak{D}}_e)}+\boldsymbol{N}_{k}^{(\overline{C}_e)}+\boldsymbol{N}_{k}^{(C_e)} - i)} \\
&= \frac{(\beta_k + N_{k',k}^{(\neg[C_e])})}{((\sum_l \beta_l)+\boldsymbol{N}_{k'}^{(\neg[C_e])})} \times \frac{\prod_{l'=1}^{K} \prod_{i=1}^{N_{k,l'}^{(C_e)}} (\beta_{l'} + N_{k,l'}^{(\overline{\mathfrak{D}}_e)} + N_{k,l'}^{(\overline{C}_e)} + N_{k,l'}^{(C_e)} - i)}{\prod_{i=1}^{\boldsymbol{N}_k^{(C_e)}} ((\sum_{l'} \beta'_l)+\boldsymbol{N}_{k}^{(\overline{\mathfrak{D}}_e)}+\boldsymbol{N}_{k}^{(\overline{C}_e)}+\boldsymbol{N}_{k}^{(C_e)} - i)}
\end{aligned} \tag{A.5}
$$

When $k' \neq k$ i.e. parent event topic is not same as the event topic k (currently that we are trying to assign), then $N_{k',k}^{(\neg[C_e])} = N_{k',k}^{(\neg \mathfrak{p}_e)} = N_{k',k} - 1$. Similarly, $\boldsymbol{N}_{k'}^{(\neg[C_e])} = \boldsymbol{N}_{k'}^{(\neg \mathfrak{p}_e)} = \boldsymbol{N}_{k'} - 1$.

Therefore, when $k' \neq k$

$$P(\boldsymbol{\eta}_{\overline{\mathfrak{D}}_e \setminus \{e'\}}, \eta_{e'} = k', \eta_e = k, \boldsymbol{\eta}_{\mathfrak{D}_e} | \boldsymbol{z}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma})$$
$$\propto \frac{(\beta_k + N_{k',k} - 1)}{((\sum_l \beta_l) + \boldsymbol{N}_{k'} - 1)} \times \frac{\prod_{l'=1}^{K} \prod_{i=1}^{N_{k,l'}^{(C_e)}} (\beta_{l'} + N_{k,l'}^{(\overline{\mathfrak{D}}_e)} + N_{k,l'}^{(\overline{C}_e)} + N_{k,l'}^{(C_e)} - i)}{\prod_{i=1}^{\boldsymbol{N}_k^{(C_e)}} ((\sum_{l'} \beta'_l) + \boldsymbol{N}_k^{(\overline{\mathfrak{D}}_e)} + \boldsymbol{N}_k^{(\overline{C}_e)} + \boldsymbol{N}_k^{(C_e)} - i)} \tag{A.6}$$

When $k' = k$ i.e. parent events topic is same as the topic of the current event, the term $\mathcal{T}_{k',k}^{(\mathfrak{p}_e)}$ in Equations A.3 and A.4 will be replaced with $\mathcal{T}_{k,k}^{(\mathfrak{p}_e)}$. And thus the term related to the integration $\boldsymbol{\mathcal{T}}_{k'}$ has no change and can be canceled out from the denominator. Thus, the only term that remains is the term related to the integration $\boldsymbol{\mathcal{T}}_k$ along with an additional term $\mathcal{T}_{k,k}^{(\mathfrak{p}_e)}$. Thus Equation A.4 can be written as:

$$P(\boldsymbol{\eta}_{\overline{\mathfrak{D}}_e \setminus \{e'\}}, \eta_{e'} = k, \eta_e = k, \boldsymbol{\eta}_{\mathfrak{D}_e} | \boldsymbol{z}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma})$$
$$= \frac{B(\boldsymbol{\beta} + \boldsymbol{N}_k^{(\overline{\mathfrak{D}}_e)} + \boldsymbol{N}_k^{(\overline{C}_e)} + \boldsymbol{N}_k^{(C_e)} + \boldsymbol{N}_k^{(\mathfrak{p}_e)})}{B(\boldsymbol{\beta})} = \frac{B(\boldsymbol{\beta} + \boldsymbol{N}_k^{(\overline{\mathfrak{D}}_e)} + \boldsymbol{N}_k^{(\overline{C}_e)} + \boldsymbol{N}_k^{(C_e)} + 1)}{B(\boldsymbol{\beta})} \tag{A.7}$$

Solving further, we can write the probability as:

$$P(\boldsymbol{\eta}_{\overline{\mathfrak{D}}_e \setminus \{e'\}}, \eta_{e'} = k, \eta_e = k, \boldsymbol{\eta}_{\mathfrak{D}_e} | \boldsymbol{z}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma})$$
$$\propto \frac{\prod_{\substack{l'=1 \\ l' \neq k}}^{K} \prod_{i=1}^{N_{k,l'}^{(C_e)}} (\beta_{l'} + N_{k,l'}^{(\overline{\mathfrak{D}}_e)} + N_{k,l'}^{(\overline{C}_e)} + N_{k,l'}^{(C_e)} - i) \times \prod_{i=1}^{N_{k,k}^{(C_e)}+1} (\beta_k + N_{k,k}^{(\overline{\mathfrak{D}}_e)} + N_{k,k}^{(\overline{C}_e)} + N_{k,k}^{(C_e)} - i)}{\prod_{i=1}^{\boldsymbol{N}_k^{(C_e)}+1} ((\sum_{l'} \beta'_l) + \boldsymbol{N}_k^{(\overline{\mathfrak{D}}_e)} + \boldsymbol{N}_k^{(\overline{C}_e)} + \boldsymbol{N}_k^{(C_e)} - i)} \tag{A.8}$$

Note that, for $l' \neq k$, $N_{k,l'}^{(C_e)} = N_{k,l'}^{([C_e])}$, and for $l' = k$, $N_{k,l'}^{(C_e)} + 1 = N_{k,l'}^{([C_e])}$. Therefore, the above equation can be written as:

$$P(\boldsymbol{\eta}_{\overline{\mathfrak{D}}_e \setminus \{e'\}}, \eta_{e'} = k, \eta_e = k, \boldsymbol{\eta}_{\mathfrak{D}_e} | \boldsymbol{z}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma})$$
$$\propto \frac{\prod_{l'=1}^{K} \prod_{i=1}^{N_{k,l'}^{[C_e]}} (\beta_{l'} + N_{k,l'}^{(\overline{\mathfrak{D}}_e)} + N_{k,l'}^{(\overline{C}_e)} + N_{k,l'}^{[C_e]} - i)}{\prod_{i=1}^{\boldsymbol{N}_k^{[C_e]}} ((\sum_{l'} \beta'_l) + \boldsymbol{N}_k^{(\overline{\mathfrak{D}}_e)} + \boldsymbol{N}_k^{(\overline{C}_e)} + \boldsymbol{N}_k^{[C_e]} - i)} \tag{A.9}$$

Note that in the above calculation we have ignored the terms related to $\boldsymbol{\gamma}$ i.e. the terms related to the matrix $\mathfrak{U}$ (user-topic count matrix). However, for the case when event is having a parent event the terms containing to $\boldsymbol{\gamma}$ or $\mathfrak{U}$ would not play any role. For the other case when the event e has no parent, it could be solved by symmetry.

Now, solving for $P(W_{\overline{\mathfrak{D}}_e}, \boldsymbol{w}_e, W_{\mathfrak{D}_e} | \boldsymbol{\eta}, \boldsymbol{z}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma})$. Let W be the list of all the words in all the documents (i.e. $W = \boldsymbol{w}_e \cup W_{\mathfrak{D}_e} \cup W_{\overline{\mathfrak{D}}_e}$), $\mathfrak{W}$ is the vocabulary set and $\mathfrak{T}$ is the

topic-word count matrix.

$$
\begin{aligned}
P(W_{\overline{\mathfrak{D}}_e}, \boldsymbol{w}_e, W_{\mathfrak{D}_e} | \boldsymbol{\eta}, \boldsymbol{z}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}) &= P(W \mid \boldsymbol{\eta}, \boldsymbol{\alpha}, rest) = \int_{\boldsymbol{\zeta}} P(W|\boldsymbol{\eta}, \boldsymbol{\zeta}) P(\boldsymbol{\zeta}|\boldsymbol{\alpha}) d\boldsymbol{\zeta} \\
&= \int_{\boldsymbol{\zeta}} \prod_{i=1}^{|W|} \zeta_{k_i, w_i} \prod_{k=1}^{K} P(\boldsymbol{\zeta}_k|\boldsymbol{\alpha}) d\boldsymbol{\zeta} = \int_{\boldsymbol{\zeta}} \prod_{k=1}^{K} \prod_{w \in \mathfrak{W}} \zeta_{k,w}^{\mathfrak{T}_{k,w}} \prod_{k=1}^{K} \frac{1}{B(\boldsymbol{\alpha})} \prod_{w \in \mathfrak{W}} \zeta_{k,w}^{\alpha_w - 1} d\boldsymbol{\zeta} \\
&= \int_{\boldsymbol{\zeta}} \prod_{k=1}^{K} \frac{1}{B(\boldsymbol{\alpha})} \prod_{w \in \mathfrak{W}} \zeta_{k,w}^{\alpha_w + \mathfrak{T}_{k,w} - 1} d\boldsymbol{\zeta} = \prod_{k=1}^{K} \frac{1}{B(\boldsymbol{\alpha})} \int_{\boldsymbol{\zeta}_k} \prod_{w \in \mathfrak{W}} \zeta_{k,w}^{\alpha_w + \mathfrak{T}_{k,w} - 1} d\boldsymbol{\zeta}_k \\
&= \prod_{k=1}^{K} \frac{B(\boldsymbol{\alpha} + \boldsymbol{\mathfrak{T}}_k)}{B(\boldsymbol{\alpha})}
\end{aligned} \tag{A.10}
$$

If we had been writing the complete term in the above equation along with the marginalized denominator (from eq. A.2) then most of the terms would cancel out and the only terms that remain would be the terms related to the words in the document of event e (as the only difference in the likelihood and the marginalized likelihood is the count of words in document of event e). And thus the above equation can be simplified as follows:

$$
P(W_{\overline{\mathfrak{D}}_e}, \boldsymbol{w}_e, W_{\mathfrak{D}_e} \mid \eta_e = k, \boldsymbol{\eta}_{\mathfrak{D}_e}, \boldsymbol{\eta}_{\overline{\mathfrak{D}}_e}, \boldsymbol{\alpha}, rest) \propto \frac{\prod_{w \in d_e} \prod_{i=1}^{N_{d_e}^w - 1} (\alpha_w + \mathfrak{T}_{k,w}^{\neg e} + i)}{\prod_{i=1}^{N_{d_e} - 1} ((\sum_{w \in V} \alpha_w) + \mathfrak{T}_k^{\neg e} + i)} \tag{A.11}
$$

where, N_{d_e} is the number of words in the document corresponding to event e and $N_{d_e}^w$ is the number of times word w appears in document corresponding to event e. Note that, $N_{d_e} = \mathfrak{T}_k^e$, and $N_{d_e}^w = \mathfrak{T}_{k,w}^e$. Therefore, the update rule for topic assignment can be written using Equations A.6, A.9, A.10 and A.11 as follows:

if $z_e \neq 0$:

1. $\eta_{z_e} = k'$, and $k' \neq k$:

$$
\begin{aligned}
P(\eta_e = k \mid \boldsymbol{w}_e, \eta_{z_e} = k', \boldsymbol{\eta}_{\overline{\mathfrak{D}}_e \setminus z_e}, \boldsymbol{\eta}_{\mathfrak{D}_e}, \boldsymbol{z}, W_{\overline{\mathfrak{D}}_e}, W_{\mathfrak{D}_e}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}) &\propto \frac{(\beta_k + N_{k',k} - 1)}{((\sum_l \beta_l) + \boldsymbol{N}_{k'} - 1)} \\
\times \frac{\prod_{l'=1}^{K} \prod_{i=1}^{N_{k,l'}^{(C_e)}} (\beta_{l'} + N_{k,l'}^{(\overline{\mathfrak{D}}_e)} + N_{k,l'}^{(\overline{C}_e)} + N_{k,l'}^{(C_e)} - i)}{\prod_{i=1}^{\boldsymbol{N}_k^{(C_e)}} ((\sum_{l'} \beta_{l'}) + \boldsymbol{N}_k^{(\overline{\mathfrak{D}}_e)} + \boldsymbol{N}_k^{(\overline{C}_e)} + \boldsymbol{N}_k^{(C_e)} - i)} \\
\times \frac{\prod_{w \in d_e} \prod_{i=1}^{N_{d_e}^w - 1} (\alpha_w + \mathfrak{T}_{k,w}^{\neg e} + i)}{\prod_{i=1}^{N_{d_e} - 1} ((\sum_{w \in V} \alpha_w) + \mathfrak{T}_k^{\neg e} + i)}
\end{aligned}
$$

$$
\begin{aligned}
&= \frac{(\beta_k + N_{k',k}^{(\neg \mathfrak{p}_e)})}{((\sum_l \beta_l) + \boldsymbol{N}_{k'}^{(\neg \mathfrak{p}_e)})} \times \frac{\prod_{l'=1}^{K} \prod_{i=0}^{N_{k,l'}^{(C_e)}-1} (\beta_{l'} + N_{k,l'}^{(\overline{\mathfrak{D}}_e)} + N_{k,l'}^{(\overline{C}_e)} + i)}{\prod_{i=0}^{\boldsymbol{N}_k^{(C_e)}-1} ((\sum_{l'} \beta_{l'}) + \boldsymbol{N}_k^{(\overline{\mathfrak{D}}_e)} + \boldsymbol{N}_k^{(\overline{C}_e)} + i)} \\
&\qquad \times \frac{\prod_{w \in d_e} \prod_{i=0}^{N_{d_e}^{w}-1} (\alpha_w + \mathfrak{T}_{k,w}^{\neg e} + i)}{\prod_{i=0}^{N_{d_e}-1} ((\sum_{w \in V} \alpha_w) + \mathfrak{T}_k^{\neg e} + i)} \\
&= \frac{(\beta_k + N_{k',k}^{(\neg \mathfrak{p}_e)})}{((\sum_l \beta_l) + \boldsymbol{N}_{k'}^{(\neg \mathfrak{p}_e)})} \times \frac{\prod_{l'=1}^{K} \prod_{i=0}^{N_{k,l'}^{(C_e)}-1} (\beta_{l'} + N_{k,l'}^{(\neg C_e)} + i)}{\prod_{i=0}^{\boldsymbol{N}_k^{(C_e)}-1} ((\sum_{l'} \beta_{l'}) + \boldsymbol{N}_k^{\neg C_e} + i)} \\
&\qquad \times \frac{\prod_{w \in d_e} \prod_{i=0}^{N_{d_e}^{w}-1} (\alpha_w + \mathfrak{T}_{k,w}^{\neg e} + i)}{\prod_{i=0}^{N_{d_e}-1} ((\sum_{w \in V} \alpha_w) + \mathfrak{T}_k^{\neg e} + i)}
\end{aligned} \tag{A.12}
$$

2. $\eta_{z_e} = k'$, and $k' = k$:

$$
\begin{aligned}
&P(\eta_e = k \mid \boldsymbol{w}_e, \eta_{z_e} = k, \boldsymbol{\eta}_{\overline{\mathfrak{D}}_e \setminus z_e}, \boldsymbol{\eta}_{\mathfrak{D}_e}, \boldsymbol{z}, W_{\overline{\mathfrak{D}}_e}, W_{\mathfrak{D}_e}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}) \propto \\
&\frac{\prod_{\substack{l'=1 \\ l' \neq k}}^{K} \prod_{i=1}^{N_{k,l'}^{(C_e)}} (\beta_{l'} + N_{k,l'}^{(\overline{\mathfrak{D}}_e)} + N_{k,l'}^{(\overline{C}_e)} + N_{k,l'}^{(C_e)} - i) \times \prod_{i=1}^{N_{k,k}^{(C_e)}+1} (\beta_k + N_{k,k}^{(\overline{\mathfrak{D}}_e)} + N_{k,k}^{(\overline{C}_e)} + N_{k,k}^{(C_e)} - i)}{\prod_{i=1}^{\boldsymbol{N}_k^{(C_e)}+1} ((\sum_{l'} \beta'_l) + \boldsymbol{N}_k^{(\overline{\mathfrak{D}}_e)} + \boldsymbol{N}_k^{(\overline{C}_e)} + \boldsymbol{N}_k^{(C_e)} - i)} \\
&\qquad \times \frac{\prod_{w \in d_e} \prod_{i=1}^{N_{d_e}^{w}-1} (\alpha_w + \mathfrak{T}_{k,w}^{\neg e} + i)}{\prod_{i=1}^{N_{d_e}-1} ((\sum_{w \in V} \alpha_w) + \mathfrak{T}_k^{\neg e} + i)} \\
&= \frac{\prod_{\substack{l'=1 \\ l' \neq k}}^{K} \prod_{i=0}^{N_{k,l'}^{(C_e)}-1} (\beta_{l'} + N_{k,l'}^{(\overline{\mathfrak{D}}_e)} + N_{k,l'}^{(\overline{C}_e)} + i) \times \prod_{i=0}^{N_{k,k}^{(C_e)}} (\beta_k + N_{k,k}^{(\overline{\mathfrak{D}}_e)} + N_{k,k}^{(\overline{C}_e)} + i)}{\prod_{i=0}^{\boldsymbol{N}_k^{(C_e)}} ((\sum_{l'} \beta'_l) + \boldsymbol{N}_k^{(\overline{\mathfrak{D}}_e)} + \boldsymbol{N}_k^{(\overline{C}_e)} + i)} \\
&\qquad \times \frac{\prod_{w \in d_e} \prod_{i=0}^{N_{d_e}^{w}-1} (\alpha_w + \mathfrak{T}_{k,w}^{\neg e} + i)}{\prod_{i=0}^{N_{d_e}-1} ((\sum_{w \in V} \alpha_w) + \mathfrak{T}_k^{\neg e} + i)} \\
&= \frac{\prod_{\substack{l'=1 \\ l' \neq k}}^{K} \prod_{i=0}^{N_{k,l'}^{(C_e)}-1} (\beta_{l'} + N_{k,l'}^{(\neg [C_e])} + i) \times \prod_{i=0}^{N_{k,k}^{(C_e)}} (\beta_k + N_{k,k}^{(\neg [C_e])} + i)}{\prod_{i=0}^{\boldsymbol{N}_k^{(C_e)}} ((\sum_{l'} \beta'_l) + \boldsymbol{N}_k^{(\neg [C_e])} + i)} \\
&\qquad \times \frac{\prod_{w \in d_e} \prod_{i=0}^{N_{d_e}^{w}-1} (\alpha_w + \mathfrak{T}_{k,w}^{\neg e} + i)}{\prod_{i=0}^{N_{d_e}-1} ((\sum_{w \in V} \alpha_w) + \mathfrak{T}_k^{\neg e} + i)}
\end{aligned} \tag{A.13}
$$

if $z_e = 0$:

$$
\begin{aligned}
&P(\eta_e = k \mid \boldsymbol{w}_e, \boldsymbol{\eta}_{\overline{\mathfrak{D}}_e}, \boldsymbol{\eta}_{\mathfrak{D}_e}, \boldsymbol{z}, W_{\overline{\mathfrak{D}}_e}, W_{\mathfrak{D}_e}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}) \propto \\
&\frac{\gamma_k + \mathfrak{U}_{v,k} - 1}{(\sum_k \gamma_k) + \mathfrak{U}_v - 1} \times \frac{\prod_{l'=1}^{K} \prod_{i=1}^{N_{k,l'}^{(C_e)}} (\beta_{l'} + N_{k,l'}^{(\overline{\mathfrak{D}}_e)} + N_{k,l'}^{(\overline{C}_e)} + N_{k,l'}^{(C_e)} - i)}{\prod_{i=1}^{N_k^{(C_e)}} ((\sum_{l'} \beta'_l) + \boldsymbol{N}_k^{(\overline{\mathfrak{D}}_e)} + \boldsymbol{N}_k^{(\overline{C}_e)} + \boldsymbol{N}_k^{(C_e)} - i)} \\
&\qquad \times \frac{\prod_{w \in d_e} \prod_{i=1}^{N_{d_e}^{w}-1} (\alpha_w + \mathfrak{T}_{k,w}^{\neg e} + i)}{\prod_{i=1}^{N_{d_e}-1} ((\sum_{w \in V} \alpha_w) + \mathfrak{T}_k^{\neg e} + i)} \\
&= \frac{\gamma_k + \mathfrak{U}_{v,k}^{(\neg e)}}{(\sum_k \gamma_k) + \mathfrak{U}_v^{(\neg e)}} \times \frac{\prod_{l'=1}^{K} \prod_{i=0}^{N_{k,l'}^{(C_e)}-1} (\beta_{l'} + N_{k,l'}^{(\neg C_e)} + i)}{\prod_{i=0}^{N_k^{(C_e)}-1} ((\sum_{l'} \beta_{l'}) + \boldsymbol{N}_k^{\neg C_e} + i)} \\
&\qquad \times \frac{\prod_{w \in d_e} \prod_{i=0}^{N_{d_e}^{w}-1} (\alpha_w + \mathfrak{T}_{k,w}^{\neg e} + i)}{\prod_{i=0}^{N_{d_e}-1} ((\sum_{w \in V} \alpha_w) + \mathfrak{T}_k^{\neg e} + i)}
\end{aligned} \tag{A.14}
$$

A.1.2 Parent Assignment Conditional Probability

$$
\begin{aligned}
P(z_e = e' | E_t, \boldsymbol{z}_{\neg e}, \boldsymbol{\eta}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\mathcal{W}}, \boldsymbol{\mu}) &= P(z_e = e' | E_t, \boldsymbol{\eta}, \boldsymbol{z}_{<e}, \boldsymbol{z}_{\mathfrak{D}_e}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\mathcal{W}}, \boldsymbol{\mu}) \\
&\propto P(\boldsymbol{z}_{<e}, z_e = e', \boldsymbol{z}_{\mathfrak{D}_e}, \boldsymbol{\eta} | E_t, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\mathcal{W}}, \boldsymbol{\mu})
\end{aligned} \tag{A.15}
$$

$$
\begin{aligned}
P(\boldsymbol{z}_{<e}, z_e = e', \boldsymbol{z}_{\mathfrak{D}_e}, \boldsymbol{\eta} | E_t, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\mathcal{W}}, \boldsymbol{\mu}) &= P(\boldsymbol{z}_{\overline{\mathfrak{D}}_e} | E_t, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\mathcal{W}}, \boldsymbol{\mu}) \\
&\times P(z_e = e' | \boldsymbol{z}_{\overline{\mathfrak{D}}_e}, E_t, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\mathcal{W}}, \boldsymbol{\mu}) \times P(\boldsymbol{z}_{\mathfrak{D}_e} | \boldsymbol{z}_{\overline{\mathfrak{D}}_e}, z_e = e', E_t, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\mathcal{W}}, \boldsymbol{\mu}) \\
&\times P(\boldsymbol{\eta} | \boldsymbol{z}, E_t, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\mathcal{W}}, \boldsymbol{\mu})
\end{aligned} \tag{A.16}
$$

Let us solve the terms related to the $P(\boldsymbol{z}_s)$ and the terms related to the $P(\boldsymbol{\eta}_s)$ separately.

Solving for the $\boldsymbol{\eta}_s$ first. We would integrate out the $\boldsymbol{\mathcal{T}}$ and split the $\boldsymbol{\eta}_s$ so that we can take into account the probability related to η_e given its parent z_e distinctly, as that would be one of the terms that would change with change in parent assignment.

$$
\begin{aligned}
P(\boldsymbol{\eta} \mid \boldsymbol{z}, E_t, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\mathcal{W}}, \boldsymbol{\mu}) &= P(\boldsymbol{\eta}_{\overline{\mathfrak{D}}_e} | \boldsymbol{z}, E_t, \boldsymbol{\beta}, \boldsymbol{\mathcal{W}}, \boldsymbol{\mu}) \times P(\eta_e = k | \boldsymbol{\eta}_{\overline{\mathfrak{D}}_e}, \boldsymbol{z}, E_t, \boldsymbol{\beta}, \boldsymbol{\mathcal{W}}, \boldsymbol{\mu}) \\
&\times P(\boldsymbol{\eta}_{\mathfrak{D}_e} | \boldsymbol{\eta}_{\overline{\mathfrak{D}}_e}, \eta_e = k, \boldsymbol{z}, E_t, \boldsymbol{\beta}, \boldsymbol{\mathcal{W}}, \boldsymbol{\mu}) \\
&= \int_{\boldsymbol{\mathcal{T}}} P(\boldsymbol{\mathcal{T}}; \boldsymbol{\beta}) P(\boldsymbol{\eta}_{\overline{\mathfrak{D}}_e} | \boldsymbol{z}, E_t, \boldsymbol{\mathcal{T}}, \boldsymbol{\mathcal{W}}, \boldsymbol{\mu}) \times P(\eta_e = k | \boldsymbol{\eta}_{\overline{\mathfrak{D}}_e}, \boldsymbol{z}, E_t, \boldsymbol{\mathcal{T}}, \boldsymbol{\mathcal{W}}, \boldsymbol{\mu}) \\
&\quad \times P(\boldsymbol{\eta}_{\mathfrak{D}_e} | \boldsymbol{\eta}_{\overline{\mathfrak{D}}_e}, \eta_e = k, \boldsymbol{z}, E_t, \boldsymbol{\mathcal{T}}, \boldsymbol{\mathcal{W}}, \boldsymbol{\mu}) d\boldsymbol{\mathcal{T}}
\end{aligned}
$$

$$
\begin{aligned}
&= \int_{\mathcal{T}} \left(\prod_{l=1}^{K} P(\mathcal{T}_k; \boldsymbol{\beta}) \right) P(\boldsymbol{\eta}_{<e} | \boldsymbol{z}, E_t, \mathcal{T}, \mathcal{W}, \boldsymbol{\mu}) \times P(\eta_e = k | \boldsymbol{\eta}_{\overline{\mathfrak{D}}_e}, \boldsymbol{z}, E_t, \mathcal{T}, \mathcal{W}, \boldsymbol{\mu}) \\
&\qquad \times P(\boldsymbol{\eta}_{\mathfrak{D}_e} | \boldsymbol{\eta}_{\overline{\mathfrak{D}}_e}, \eta_e = k, \boldsymbol{z}, E_t, \mathcal{T}, \mathcal{W}, \boldsymbol{\mu}) d\mathcal{T} \\
&= \int_{\mathcal{T}} \left(\prod_{l=1}^{K} \frac{1}{B(\boldsymbol{\beta})} \prod_{l'=1}^{K} \mathcal{T}_{l,l'}^{\beta_{l'}-1} \right) \times \left(\prod_{l=1}^{K} \prod_{l'=1}^{K} \mathcal{T}_{l,l'}^{N_{l,l'}^{(\overline{\mathfrak{D}}_e)}} \right) \times \mathcal{T}_{k',k}^{(\mathfrak{p}_e)} \times \left(\prod_{l=1}^{K} \prod_{l'=1}^{K} \mathcal{T}_{l,l'}^{N_{l,l'}^{(\mathfrak{D}_e)}} \right) d\mathcal{T} \\
&= \left(\prod_{\substack{l=1 \\ l \neq k'}}^{K} \frac{1}{B(\boldsymbol{\beta})} \int_{\mathcal{T}_l} \prod_{l'=1}^{K} \mathcal{T}_{l,l'}^{\beta_{l'}-1+N_{l,l'}^{(\overline{\mathfrak{D}}_e)}+N_{l,l'}^{(\mathfrak{D}_e)}} d\mathcal{T}_l \right) \\
&\qquad \times \left(\int_{\mathcal{T}_{k'}} \frac{1}{B(\boldsymbol{\beta})} \prod_{m'=1}^{K} \mathcal{T}_{k',m'}^{\beta_{l'}-1+N_{k',m'}^{(\overline{\mathfrak{D}}_e)}+N_{k',m'}^{(\mathfrak{D}_e)}} d\mathcal{T}'_{\boldsymbol{k}} \right) \times \mathcal{T}_{k',k}^{(\mathfrak{p}_e)} \\
&= \left(\prod_{\substack{l=1 \\ l \neq k'}}^{K} \frac{1}{B(\boldsymbol{\beta})} \int_{\mathcal{T}_l} \prod_{l'=1}^{K} \mathcal{T}_{l,l'}^{\beta_{l'}-1+N_{l,l'}} d\mathcal{T}_l \right) \\
&\qquad \times \left(\int_{\mathcal{T}_{k'}} \frac{1}{B(\boldsymbol{\beta})} \prod_{m'=1}^{K} \mathcal{T}_{k',m'}^{\beta_{l'}-1+N_{k',m'}^{(\neq e)}} d\mathcal{T}'_{\boldsymbol{k}} \right) \times \mathcal{T}_{k',k}^{(\mathfrak{p}_e)} \\
&\propto \left(\int_{\mathcal{T}_{k'}} \prod_{m'=1}^{K} \mathcal{T}_{k',m'}^{\beta_{l'}-1+N_{k',m'}^{(\neq e)}} d\mathcal{T}'_{\boldsymbol{k}} \right) \mathcal{T}_{k',k}^{(\mathfrak{p}_e)} \quad = \quad \frac{B(\boldsymbol{\beta} + \boldsymbol{N}_{k'}^{(\neg \mathfrak{p}_e)} + 1)}{B(\boldsymbol{\beta})}
\end{aligned} \tag{A.17}
$$

Here, $N_{k'}^{(\neg \mathfrak{p}_e)}$ indicates the number of times the topic-topic combination (k', m), for $m \in [1, K]$ was found without considering the parent assignment of event e i.e. without considering the count over the edge (z_e, e).

Solving for the $\boldsymbol{z}_s$ now. For any $e_i \in E$ at node u_{e_i} at time t_{e_i}, neighborhood set given by $\mathcal{N}(u_{e_i})$ and the parents for all the non-descendant events of e_i given, $P(z_{e_i} = f' | z_{\overline{\mathfrak{D}}_{e_i}}, E_t, \mathcal{W}, \boldsymbol{\mu})$, where $f' \in \mathcal{N}(u_{e_i})$ can be written as:

$$
\begin{aligned}
&P(z_{e_i} = f' | z_{\overline{\mathfrak{D}}_{e_i}}, E_t, \mathcal{W}, \boldsymbol{\mu}) \\
&= \prod_{\substack{f \in E \\ t_f \leq t_{e_i} \\ u_f \in \mathcal{N}(u_{e_i})}} \exp\left(- \int_{t_f}^{T} h_{u_f, u_{e_i}}(\tau - t_f) d\tau \right) h_{u_{f'}, u_{e_i}}(t_{e_i} - t_{f'})
\end{aligned} \tag{A.18}
$$

Therefore, the probability for $\boldsymbol{z}_s$ can be written as:

$$
\begin{aligned}
P(\boldsymbol{z}_{\overline{\mathfrak{D}}_e} | E_t, \mathcal{W}, \boldsymbol{\mu}) \times P(z_e = e' | \boldsymbol{z}_{\overline{\mathfrak{D}}_e}, E_t, \mathcal{W}, \boldsymbol{\mu}) \times P(\boldsymbol{z}_{\mathfrak{D}_e} | \boldsymbol{z}_{\overline{\mathfrak{D}}_e}, z_e = e', E_t, \mathcal{W}, \boldsymbol{\mu}) \\
= P(z_1 | E_t, \mathcal{W}, \boldsymbol{\mu}) \times P(z_2 | \boldsymbol{z}_{<e_1}, E_t, \mathcal{W}, \boldsymbol{\mu}) \times \quad \ldots \\
\times P(z_e = e' | \boldsymbol{z}_{\overline{\mathfrak{D}}_e}, E_t, \mathcal{W}, \boldsymbol{\mu}) \times P(z_{(e+1)} = e' | \boldsymbol{z}_{\overline{\mathfrak{D}}_{(e+1)}}, E_t, \mathcal{W}, \boldsymbol{\mu}) \times
\end{aligned}
$$

$$
\begin{aligned}
&\dots \qquad \times P(z_n = e' | \boldsymbol{z}_{\overline{\mathfrak{D}}_n}, E_t, \boldsymbol{\mathcal{W}}, \boldsymbol{\mu}) \propto P(z_e = e' | \boldsymbol{z}_{\overline{\mathfrak{D}}_e}, E_t, \boldsymbol{\mathcal{W}}, \boldsymbol{\mu}) \\
&= \prod_{\substack{f \in E \\ t_f \le t_e \\ u_f \in \mathcal{N}(u_e)}} \exp\left(- \int_{t_f}^{T} h_{u_f, u_e}(\tau - t_f) d\tau \right) h_{u_{e'}, u_e}(t_e - t_{e'}) \\
&\propto h_{u_{e'}, u_e}(t_e - t_{e'})
\end{aligned} \tag{A.19}
$$

Note that the proportionality sign in equation A.15 is because of neglecting the denominator, which has the similar form as that of the numerator except that the parent assignment of event e is not taken into consideration. And thus the proportionality in the second step of above equation is because all the other terms are similar in the denominator and thus can be canceled out. Also, the proportionality in the last or fourth step is because the exponential term is same for all the values of parent assignment and thus can be neglected.

Thus, the update for parent assignment of an event e can be given using equations A.19 and A.17 as follows:

$$
P(z_e = e' | E_t, \boldsymbol{z}_{\neg e}, \boldsymbol{\eta}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\mathcal{W}}, \boldsymbol{\mu}) \propto \frac{B(\boldsymbol{\beta} + \boldsymbol{N}_{k'}^{(\neg \mathfrak{p}_e)} + 1)}{B(\boldsymbol{\beta})} \times h_{u_{e'}, u_e}(t_e - t_{e'}) \tag{A.20}
$$

Note that, $N_{k',k} = N_{k',k}^{(\neg \mathfrak{p}_e)} + 1$ and similarly, $\boldsymbol{N}_{k'} = \boldsymbol{N}_{k'}^{(\neg \mathfrak{p}_e)} + 1$.

Therefore, considering the denominator with exception to parent assignment of event e, the above equation can be simplified further as:

$$
P(z_e = e' | E_t, \boldsymbol{z}_{\neg e}, \boldsymbol{\eta}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\mathcal{W}}, \boldsymbol{\mu}) \propto \frac{(\beta_k + N_{k',k} - 1)}{((\sum_{k=1}^{K} \beta_k) + \boldsymbol{N}_{k'} - 1)} \times h_{u_{e'}, u_e}(t_e - t_{e'}) \tag{A.21}
$$

Working on similar lines, the probability of assigning no parent to the event e which was triggered at node u_e at time t_e, can be given as:

$$
P(z_e = 0 | E_t, \boldsymbol{z}_{\neg e}, \boldsymbol{\eta}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\mathcal{W}}, \boldsymbol{\mu}) \propto \frac{(\gamma_k + \mathfrak{U}_{u_e,k} - 1)}{((\sum_{k=1}^{K} \gamma_k) + \mathfrak{U}_{u_e} - 1)} \times \mu_{u_e}(t_e) \tag{A.22}
$$

A.1.3 Conditional Probability for User-User Influence

$$
P(\mathcal{W}_{u,v} = x \mid \boldsymbol{\mathcal{W}}_{\neg u,v}, E_t, \boldsymbol{z}, \boldsymbol{\eta}, \boldsymbol{W}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\mu}) = P(\mathcal{W}_{u,v} = x \mid E_t^{(u,v)}, \boldsymbol{z}) \tag{A.23}
$$

where, $E_t^{(u,v)}$ denotes the timestamps of the events those are at u and have triggered events at node v. The simplification in the above equation is because $W_{u,v}$ is dependent only on the parent assignments, and the time at which the events are generated (specifically, $E_t^{(u,v)}$).

Using the Bayes rule we can write it as:

$$P(\mathcal{W}_{u,v} = x \mid E_t^{(u,v)}, \boldsymbol{z}) \propto P(\mathcal{W}_{u,v} = x \mid \boldsymbol{z}) \times P(E_t^{(u,v)} \mid \mathcal{W}_{u,v} = x, \boldsymbol{z}) \tag{A.24}$$

Where, the first term on the RHS is the conditional prior and the second term is the conditional likelihood. The likelihood term can be written as:

$$\begin{aligned} P(E_t^{(u,v)} \mid \mathcal{W}_{u,v} = x, \boldsymbol{z}) &= \prod_{\substack{e \in E \\ c_e = u}} \left[exp\left(- \int_{t_e}^{T} h_{c_e,v}(\tau - t_e) dt \right) \prod_{\substack{e' \in E \\ c_{e'} = v \\ z_{e'} = e}} h_{c_e,c_{e'}}(t_{e'} - t_e) \right] \\ &= \prod_{\substack{e \in E \\ c_e = u}} \left[exp\left(- \int_{t_e}^{T} \mathcal{W}_{c_e,v} f_{c_e,v}(\tau - t_e) dt \right) \prod_{\substack{e' \in E \\ c_{e'} = v \\ z_{e'} = e}} \mathcal{W}_{c_e,c_{e'}} f_{c_e,c_{e'}}(t_{e'} - t_e) \right] \end{aligned} \tag{A.25}$$

Thus, the conditional in Equation A.24 can be written as:

$$\begin{aligned} P(\mathcal{W}_{u,v} = x \mid E_t^{(u,v)}, \boldsymbol{z}) &\propto P(\mathcal{W}_{u,v} = x \mid \boldsymbol{z}) \\ &\times \prod_{\substack{e \in E \\ c_e = u}} \left[exp\left(- \int_{t_e}^{T} \mathcal{W}_{c_e,v} f_{c_e,v}(\tau - t_e) dt \right) \prod_{\substack{e' \in E \\ c_{e'} = v \\ z_{e'} = e}} \mathcal{W}_{c_e,c_{e'}} f_{c_e,c_{e'}}(t_{e'} - t_e) \right] \end{aligned} \tag{A.26}$$

Using Gamma prior with parameters (α_1, β_1) for $P(\mathcal{W}_{u,v} = x \mid \boldsymbol{z})$, then the previous equation can be approximately written as:

$$P(\mathcal{W}_{u,v} = x \mid E_t^{(u,v)}, \boldsymbol{z}) \propto \quad x^{N_{u,v} + \alpha_1 - 1} \times \exp\left(-x \left(N_u + \frac{1}{\beta_1} \right) \right) \tag{A.27}$$

Which is again a Gamma distribution with parameters $\alpha_1^{'} = N_{u,v} + \alpha_1 - 1$ and $\beta_1^{'} = \frac{1}{\left(N_u + \frac{1}{\beta_1} \right)}$

Bibliography

[1] Elliot Anshelevich, Deeparnab Chakrabarty, Ameya Hate, and Chaitanya Swamy. Approximation algorithms for the firefighter problem: Cuts over time and submodularity. In *International Symposium on Algorithms and Computation*, pages 974–983. Springer, 2009.

[2] Emmanuel Bacry, Iacopo Mastromatteo, and Jean-François Muzy. Hawkes processes in finance. *Market Microstructure and Liquidity*, 1(01):1550005, 2015.

[3] Eytan Bakshy, Itamar Rosenn, Cameron Marlow, and Lada Adamic. The role of social networks in information diffusion. In *Proceedings of the 21st international conference on World Wide Web*, pages 519–528, 2012.

[4] N. Barbieri, G. Manco, and E. Ritacco. Survival factorization on diffusion networks. In *ECML-PKDD*, 2017.

[5] N. Barbieri, G. Manco, E. Ritacco, M. Carnuccio, and A. Bevacqua. Probabilistic topic models for sequence data. *Mach. Learn.*, 93(1):5–29, 2013.

[6] Cristina Bazgan, Morgan Chopin, Marek Cygan, Michael R Fellows, Fedor V Fomin, and Erik Jan van Leeuwen. Parameterized complexity of firefighting. *Journal of Computer and System Sciences*, 80(7):1285–1297, 2014.

[7] Hans L. Bodlaender, Bart M. P. Jansen, and Stefan Kratsch. Kernelization lower bounds by cross-composition. *SIAM J. Discrete Math.*, 28(1):277–305, 2014.

[8] Andrew Bray and Frederic Paik Schoenberg. Assessment of point process models for earthquake forecasting. *Statistical science*, pages 510–520, 2013.

[9] Jacqueline Johnson Brown and Peter H Reingen. Social ties and word-of-mouth referral behavior. *Journal of Consumer research*, 14(3):350–362, 1987.

[10] Ceren Budak, Divyakant Agrawal, and Amr El Abbadi. Limiting the spread of misinformation in social networks. In *Proceedings of the 20th international conference on World wide web*, pages 665–674, 2011.

[11] Leizhen Cai, Elad Verbin, and Lin Yang. Firefighting on trees:$(1 - 1/e)$–approximation, fixed parameter tractability and a subexponential algorithm. In *International Symposium on Algorithms and Computation*, pages 258–269. Springer, 2008.

[12] Parinya Chalermsook and Julia Chuzhoy. Resource minimization for fire containment. In *Proceedings of the twenty-first annual ACM-SIAM symposium on Discrete Algorithms*, pages 1334–1349. Society for Industrial and Applied Mathematics, 2010.

[13] Jianer Chen, Yang Liu, and Songjian Lu. An improved parameterized algorithm for the minimum node multiway cut problem. *Algorithmica*, 55(1):1–13, 2009.

[14] Wei Chen, Yajun Wang, and Siyu Yang. Efficient influence maximization in social networks. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 199–208, 2009.

[15] Yuanda Chen. Thinning algorithms for simulating point processes. 2016. https://www.math.fsu.edu/~ychen/research/Thinning%20algorithm.pdf (Online; accessed 31-Dec-2019).

[16] Zhuo Chen. An agent-based model for information diffusion over online social networks. *Papers in Applied Geography*, 5(1-2):77–97, 2019.

[17] J. Choudhari, A. Dasgupta, I. Bhattacharya, and S. Bedathur. Discovering topical interactions in text-based cascades using hidden markov hawkes processes. In *ICDM*, 2018.

[18] Jayesh Choudhari, Anirban Dasgupta, Neeldhara Misra, and M. S. Ramanujan. Saving critical nodes with firefighters is FPT. In Ioannis Chatzigiannakis, Piotr Indyk, Fabian Kuhn, and Anca Muscholl, editors, *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland*, volume 80 of *LIPIcs*, pages 135:1–135:13. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2017.

[19] Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015.

[20] Marek Cygan, Fedor V Fomin, and Erik Jan van Leeuwen. Parameterized Complexity of Firefighting Revisited. In *Parameterized and exact computation*, pages 13–26. Springer, Heidelberg, Berlin, Heidelberg, 2012.

[21] D. J. Daley and D. Vere-Jones. *An introduction to the theory of point processes. Vol. I*. Probability and its Applications (New York). Springer-Verlag, New York, second edition, 2003. Elementary theory and methods.

[22] Ronald de Haan. *Parameterized Complexity in the Polynomial Hierarchy: Extending Parameterized Complexity Theory to Higher Levels of the Hierarchy*. Springer Nature, 2020.

[23] Pedro Domingos and Matt Richardson. Mining the network value of customers. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 57–66, 2001.

[24] Rod G Downey and Michael R Fellows. Fixed-parameter tractability and completeness i: Basic results. *SIAM Journal on Computing*, 24(4):873–921, 1995.

[25] Rod G Downey and Michael R Fellows. Fixed-parameter tractability and completeness ii: On completeness for w[1]. *Theoretical Computer Science*, 141(1-2):109–131, 1995.

[26] Rodney G Downey and Michael R Fellows. Fixed-parameter tractability and completeness. *Congressus Numerantium*, pages 161–161, 1992.

[27] L. Du, W. Buntine, H. Jin, and C. Chen. Sequential latent dirichlet allocation. *Knowledge and Information Systems*, 31(3):475–503, 2012.

[28] N. Du, M. Farajtabar, A. Ahmed, A. Smola, and L. Song. Dirichlet-hawkes processes with applications to clustering continuous-time document streams. In *SIGKDD*, 2015.

[29] Nan Du, Hanjun Dai, Rakshit Trivedi, Utkarsh Upadhyay, Manuel Gomez-Rodriguez, and Le Song. Recurrent marked temporal point processes: Embedding event history to vector. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1555–1564, 2016.

[30] David Easley, Jon Kleinberg, et al. *Networks, crowds, and markets*, volume 8. Cambridge university press Cambridge, 2010.

[31] S Finbow, B Hartnell, Q Li, and K Schmeisser. On minimizing the effects of fire or a virus on a network. *Journal of Combinatorial Mathematics and Combinatorial Computing*, 33:311–322, 2000.

[32] Stephen Finbow and Gary MacGillivray. The firefighter problem: a survey of results, directions and questions. *The Australasian Journal of Combinatorics*, 43:57–77, 2009.

[33] Jörg Flum and Martin Grohe. Describing parameterized complexity classes. *Information and Computation*, 187(2):291–319, 2003.

[34] Fedor V Fomin, Pinar Heggernes, and Erik Jan van Leeuwen. Making life easier for firefighters. In *Fun with algorithms*, pages 177–188. Springer, Heidelberg, Berlin, Heidelberg, 2012.

[35] S. Gao, J. Ma, and Z. Chen. Modeling and predicting retweeting dynamics on microblogging platforms. In *WSDM*, 2015.

[36] M. Gomez-Rodriguez, J. Leskovec, D. Balduzzi, and B. Schölkopf. Uncovering the structure and temporal dynamics of information propagation. *Network Science*, 2(1):26–65, 2014.

[37] Manuel Gomez-Rodriguez, Jure Leskovec, and Andreas Krause. Inferring networks of diffusion and influence. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 5(4):1–37, 2012.

[38] Manuel Gomez-Rodriguez, Jure Leskovec, and Bernhard Schölkopf. Modeling information propagation with survival theory. In *International Conference on Machine Learning*, pages 666–674, 2013.

[39] Jan Grandell. *Mixed poisson processes*, volume 77. CRC Press, 1997.

[40] Mark S Granovetter. The strength of weak ties. In *Social networks*, pages 347–367. Elsevier, 1977.

[41] T. Griffiths, M. Steyvers, D. Blei, and J. Tenenbaum. Integrating topics and syntax. In *NIPS*, 2005.

[42] Thilo Gross, Carlos J Dommar D'Lima, and Bernd Blasius. Epidemic dynamics on an adaptive network. *Physical review letters*, 96(20):208701, 2006.

[43] A. Gruber, Y. Weiss, and M. Rosen-Zvi. Hidden topic markov models. In *AISTATS*, 2007.

[44] Adrien Guille, Hakim Hacid, Cecile Favre, and Djamel A Zighed. Information diffusion in online social networks: A survey. *ACM Sigmod Record*, 42(2):17–28, 2013.

[45] Bert Hartnell. Firefighter! an application of domination. In *25th Manitoba Conference on Combinatorial Mathematics and Computing*, 1995.

[46] A. Hawkes. Spectra of some self-exciting and mutually exciting point processes. *Biometrika*, 58(1), 1971.

[47] Alan G Hawkes. Spectra of some self-exciting and mutually exciting point processes. *Biometrika*, 58(1):83–90, 1971.

[48] X. He, T. Rekatsinas, J. Foulds, L. Getoor, and Y. Liu. Hawkestopic: A joint model for network inference and topic modeling from text-based cascades. In *ICML*, 2015.

[49] Yu Jin, Wendi Wang, and Shiwu Xiao. An sirs model with a nonlinear incidence rate. *Chaos, Solitons & Fractals*, 34(5):1482–1497, 2007.

[50] Michael Jordan. The exponential family: Conjugate priors. 2010. https://people.eecs.berkeley.edu/~jordan/courses/260-spring10/other-readings/chapter9.pdf (Online; accessed 15-Jan-2020).

[51] David Kempe, Jon Kleinberg, and Éva 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*, pages 137–146, 2003.

[52] AY Khintchine. Mathematical methods in the theory of queueing, griffin's statistical monographs and courses. *by MG Kendall, Hafner Publishing Co., NY*, 1960.

[53] Andrew King and Gary MacGillivray. The firefighter problem for cubic graphs. *Discrete Mathematics*, 310(3):614–621, 2010.

[54] Baron Law and Frederi Viens. *Hawkes Processes and Their Applications to High-Frequency Data Modeling*, pages 183–219. 04 2016.

[55] Jure Leskovec, Lada A Adamic, and Bernardo A Huberman. The dynamics of viral marketing. *ACM Transactions on the Web (TWEB)*, 1(1):5–es, 2007.

[56] Jure Leskovec, Andreas Krause, Carlos Guestrin, Christos Faloutsos, Jeanne VanBriesen, and Natalie Glance. Cost-effective outbreak detection in networks. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 420–429, 2007.

[57] PA W Lewis and Gerald S Shedler. Simulation of nonhomogeneous poisson processes by thinning. *Naval research logistics quarterly*, 26(3):403–413, 1979.

[58] Mei Li, Xiang Wang, Kai Gao, and Shanshan Zhang. A survey on information diffusion in online social networks: Models and methods. *Information*, 8(4):118, 2017.

[59] S. Linderman and R. Adams. Discovering latent network structure in point process data. In *ICML*, 2014.

[60] D Liu, EW Yan, and M Song. Microblog information diffusion: Simulation based on sir model. *J. Beijing Univ. Posts Telecommun*, 16:28–33, 2014.

[61] Daniel Lokshtanov and M. S. Ramanujan. Parameterized tractability of multiway cut with parity constraints. In *Automata, Languages, and Programming - 39th International Colloquium, ICALP 2012, Warwick, UK, July 9-13, 2012, Proceedings, Part I*, pages 750–761, 2012.

[62] Daniel Lokshtanov, M. S. Ramanujan, and Saket Saurabh. A linear time parameterized algorithm for node unique label cover. *CoRR*, abs/1604.08764, 2016.

[63] Dániel Marx. Parameterized graph separation problems. *Theoretical Computer Science*, 351(3):394–406, 2006.

[64] Charalampos Mavroforakis, Isabel Valera, and Manuel Gomez-Rodriguez. Modeling the dynamics of learning activity on the web. In *Proceedings of the 26th International Conference on World Wide Web*, pages 1421–1430, 2017.

[65] Hongyuan Mei and Jason M Eisner. The neural hawkes process: A neurally self-modulating multivariate point process. In *Advances in Neural Information Processing Systems*, pages 6754–6764, 2017.

[66] Michael Mitzenmacher and Eli Upfal. *Probability and computing: randomization and probabilistic techniques in algorithms and data analysis*. Cambridge university press, 2017.

[67] Mark EJ Newman. The structure and function of complex networks. *SIAM review*, 45(2):167–256, 2003.

[68] Mark EJ Newman. Threshold effects for two pathogens spreading on a network. *Physical review letters*, 95(10):108701, 2005.

[69] Mark EJ Newman. Modularity and community structure in networks. *Proceedings of the national academy of sciences*, 103(23):8577–8582, 2006.

[70] Mark EJ Newman and Juyong Park. Why social networks are different from other types of networks. *Physical review E*, 68(3):036122, 2003.

[71] Yosihiko Ogata. On lewis' simulation method for point processes. *IEEE Transactions on Information Theory*, 27(1):23–31, 1981.

[72] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. 1998.

[73] Romualdo Pastor-Satorras and Alessandro Vespignani. Epidemic spreading in scale-free networks. *Physical review letters*, 86(14):3200, 2001.

[74] Quantstart. Bayesian statistics: A beginner's guide, 2019. https://www.quantstart.com/articles/Bayesian-Statistics-A-Beginners-Guide/ (Online; accessed 16-Jan-2020).

[75] L. Rigouste. *Inference and evaluation of the multinomial mixture model for unsupervised text clustering*. PhD thesis, Télécom ParisTech, 2006.

[76] M. Rizoiu, Y. Lee, S. Mishra, and L. Xie. A tutorial on hawkes processes for events in social media. In *arXiv*, 2017.

[77] Marian-Andrei Rizoiu, Young Lee, Swapnil Mishra, and Lexing Xie. A tutorial on hawkes processes for events in social media. *arXiv preprint arXiv:1708.06401*, 2017.

[78] Daniel M Romero, Brendan Meeder, and Jon Kleinberg. Differences in the mechanics of information diffusion across topics: idioms, political hashtags, and complex contagion on twitter. In *Proceedings of the 20th international conference on World wide web*, pages 695–704, 2011.

[79] Sheldon M Ross. *Stochastic processes*, volume 2. Wiley, 1996.

[80] Sheldon M Ross. Introduction to probability models, tenth edition. 2009.

[81] Towards Data Science. An overview of monte carlo methods, 2019. https://towardsdatascience.com/an-overview-of-monte-carlo-methods-675384eb1694/ (Online; accessed 18-Feb-2020).

[82] Yeon Seonwoo, Alice Oh, and Sungjoon Park. Hierarchical dirichlet gaussian marked hawkes process for narrative reconstruction in continuous time domain. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3316–3325, 2018.

[83] K Sigman. Stationary marked point processes: An intuitive approach. 1995.

[84] Aleksandr Simma and Michael I Jordan. Modeling events with cascades of poisson processes. In *Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence*, pages 546–555, 2010.

[85] Graham Simon. *Hawkes Processes in Finance: A Review with Simulations*. PhD thesis, University of Oregon, 2016.

[86] Donald L Snyder and Michael I Miller. *Random point processes in time and space*. Springer Science & Business Media, 2012.

[87] Xi Tan, Vinayak Rao, and Jennifer Neville. Nested crp with hawkes-gaussian processes. In *International Conference on Artificial Intelligence and Statistics*, pages 1289–1298, 2018.

[88] David Vere-Jones. Some models and procedures for space-time point processes. *Environmental and Ecological Statistics*, 16(2):173–195, 2009.

[89] S. Wang, X. Hu, P. Yu, and Z. Li. Mmrate: Inferring multi-aspect diffusion networks with multi-pattern cascades. In *SIGKDD*, 2014.

[90] Lilian Weng, Jacob Ratkiewicz, Nicola Perra, Bruno Gonçalves, Carlos Castillo, Francesco Bonchi, Rossano Schifanella, Filippo Menczer, and Alessandro Flammini. The role of information diffusion in the evolution of social networks. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 356–364, 2013.

[91] Shuai Xiao, Junchi Yan, Xiaokang Yang, Hongyuan Zha, and Stephen M Chu. Modeling the intensity function of point process via recurrent neural networks. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[92] Shuang-Hong Yang and Hongyuan Zha. Mixture of mutually exciting processes for viral diffusion. In *International Conference on Machine Learning*, pages 1–9, 2013.

[93] Q. Zhao, M. Erdogdu, H. He, A. Rajaraman, and J. Leskovec. Seismic: A self-exciting point process model for predicting tweet popularity. In *KDD*, 2015.