# HUMAN NAVIGATION OF INFORMATION NETWORKS

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Robert West

June 2016

# Abstract

Network navigation constitutes a fundamental human behavior: in order to make use of the information and resources around us, we constantly explore, disentangle, and browse networks such as the Web, social networks, academic paper collections, and encyclopedias, among others. Studying the navigation patterns humans employ is important because it lets us better understand how humans reason about complex networks and lets us build more intuitively navigable and human-friendly information systems.

In this dissertation, we study how humans navigate information networks by analyzing tens of thousands of navigation traces harvested from the human-computation game *Wikispeedia,* where participants are asked to navigate between two given Wikipedia articles in as few clicks as possible. We first shed light on human navigation strategies by describing the anatomy of typical human navigation traces. We then build on these results to develop models and tools for predicting the targets of human paths from only the first few clicks, learning to navigate automatically, and recommending the insertion of important missing hyperlinks. These are useful building blocks for designing more intuitively navigable information spaces and tools to help people find information.

The navigation traces collected through the Wikispeedia game have the unique property of being labeled with users' explicit navigation targets. In general, however, humans need not have a precise target in mind when navigating the Web. Records of such navigation traces are abundant in the logs kept by any web server software. We demonstrate the value of passively collected web server logs by presenting an algorithm that leverages such raw logs in order to improve website hyperlink structure. The resulting system is deployed on Wikipedia's full server logs at terabyte scale, producing links that are clicked 12 times as frequently as the average link added by human Wikipedia editors.

Dedicated to my wife Verena. Mnmnm.

# Acknowledgements

I am deeply grateful to my adviser Jure Leskovec for guiding me through my PhD. He has been an admirable mentor, sharp, creative, honest, and supportive. Our meetings have always been a dizzying flurry of ideas that lastingly strengthened my sense of orientation in the world of academia.—‘Super!’

I would also like to thank Dan Jurafsky, Chris Potts, Eric Horvitz, and Chris Ré, who have served on my thesis committee and have provided invaluable advice throughout my time at Stanford, especially as I was hunting for jobs.

Dan Jurafsky welcomed me to Stanford in my first quarter as a rotation mentor. His unique, enthusiastic advising style made work a breeze. I also learned from Dan how important it is to look for the broader story underneath the bare results, and that shorts and flip flops are the way to go.

Some of my most enjoyable collaborations were with Chris Potts, an eloquent, calm, and collegial mentor. Time after time have I been impressed by his poignant and convincing writing style and by his distinct skill to propose changes to the structure of a technical talk that seemed obvious after the fact, but that hadn’t been obvious at all before. I also have fond memories of working on our TACL paper together in Chris’s office—an effective way to write that I will certainly apply with my students in the future.

During two internships at Microsoft Research (and beyond), Eric Horvitz has been an ever-active quasar of ideas and good humor. I admire how he manages to maintain his deeply reflective attitude amidst all this momentum. Eric illuminated to me not only that wild-eyed ideas are worth pursuing, but also that carefully executing a single crisp idea pays off. Ironically, although one of our projects was health-related, Eric is the only person with whom I've ever smoked an entire cigar from start to finish (but I'm not sure if I should thank him for that).

The final year of my PhD was shaped by my collaboration with the Wikimedia Foundation. I am especially obliged to Leila Zia, who has initiated and steadily nurtured this fun and fruitful endeavor. I deeply appreciate Leila's gentle yet purposeful style, which makes every meeting start and end with a kind and honest smile, with many others in between. The support and friendship of Dario Taraborelli, Ellery Wulczyn, and many other Wikimedians also played a key role in making my Wikimedia Fellowship a success.

Some of the most productive months of my PhD were spent on summer internships, during which I was privileged to work with colleagues and mentors who influenced my research in important ways and who taught me lessons I couldn't have learned in academia. In particular, I would like to thank Ryen White from Microsoft, Evgeniy Gabrilovich and Kevin Murphy from Google, and Chato Castillo and Ingmar Weber from Yahoo.

I'm also grateful to all the other great people I've had the chance to work—and in the process become friends—with, in particular Ashwin Paranjape, Srijan Kumar, Cristian Danescu-Niculescu-Mizil, Aju Scaria, and Rose Philip.

I thank Alex Clemesha for granting access to the navigation traces collected through The Wiki Game, as well as all players of Wikispeedia and all contributors and readers of Wikipedia, without whom the present research would have been impossible.

Being a member of the Stanford Infolab showed me that a cohesive, friendly lab culture is key in academia. There have been so many kind Infolab members that I must apologize for not being able to name each of them separately. Suffice it to point to the roots of the forest and trust that my gratitude will percolate down from there: in lieu of all Infolab members, let me thank Hector Garcia-Molina, Jennifer Widom, Jeff Ullman, Gio Wiederhold, Rok Sosič, and Andreas Paepcke for the joy- and helpful Infolunch meetings every Friday, brimming with trip reports, international candy, clean as well as dirty jokes, and some of the most insightful tech-talk feedback out there. I also owe a special thank-you to our outstanding lab administrators Marianne Siroker and Yesenia Gallegos.

Academic progress hinges on the ability to pursue ideas freely. I therefore gratefully acknowledge the financial support from a Facebook Graduate Fellowship and a Hewlett-Packard Stanford Graduate Fellowship, which have afforded me that freedom.

Getting a PhD is tough work, and keeping the spirits high would be hard without the consolation of *non tibi hoc soli* and the support of peers who are going through the same ups and downs. More than the degree being concluded with this thesis, I value the dear friends I have made over the course of the past six years at Stanford: Thanks to Hristo 'The Wolf' Paskov, my companion from day zero beyond day two thousand at Stanford, a mathematical Conan, and one of the most generous and loyal people I have ever met. Thanks to Kelley Paskov, an angel of hospitality and the element of reason that has saved us from bears, starvation, and other stupidities. Thanks to Julian McAuley, a top human being and source of Delphic advice about research at the proper and meta levels, who not a single time said, 'I'm busy, come back later.' Thanks to Andrej Krevl, the best sysadmin in the world and a *condicio sine qua non* without whose advice and selfless help many of the results presented here would never have seen the light of day and whose honesty is bound to make him keep his promise to go on a bike ride with me before I leave California. Thanks to Ashton Anderson—roommate, labmate, courtmate, Seoulmate, fond of fonts and a boiling wort of words—, with whom I share so many interests and passions. And thanks to Jake Lussier, a rare combination of chaos and reason that is rare to find, especially around the Mission District.

Many others have contributed to making my Golden State years a blast. Special thanks to Ben Sanders, Anna Druet, Amit Levy, Elliott Engelmann, Amina Bayou, Greg Mesnil, Mònica Vilas, Enrique Fibla, Matthew and Joseph Williams, Kyle Knaggs, Chuck de Lannoy, Bobak Shahriari, Fabian Kaelin, and Ryan Faulkner.

A big shout goes to all my homies in Bavaria: Andy, Aussi, Colosso, Gbrl, Katha, Klausi, Kulzi, Schieder. Our friendship has remained so steadfast over the decades that no update is needed to the acknowledgements I included in my master's thesis six years ago.

More than to anyone else I need to send my gratitude and appreciation to my family. They are the sources, the hubs, and the targets of the paths on which I've been navigating life. A thousand thanks go to my parents Waltraut and Helmuth, my sister Tina, and my grandparents Agnetha and Heinrich Fleischer and Barbara and Johann-Anton West.

Finally, I dedicate this thesis to my beloved wife Verena, whom I will tell in person how much I love her.

# Contents

# List of Tables

# List of Figures

# Chapter 1

# Introduction

## 1.1 Motivation

There is more information in the world than a single human being could possibly make sense of, and it has become commonplace to remark that we live in an 'age of information overload' [30, 97]. As early as 1755, the encyclopedist Denis Diderot anticipated that, "[a]s long as the centuries continue to unfold, the number of books will grow continually, and one can predict that a time will come when it will be almost as difficult to learn anything from books as from the direct study of the whole universe." [24]

Two centuries later, Diderot's premonition had become a fact that threatened to seriously slow the progress of science. As remarked by Vannevar Bush in his seminal 1945 article *As We May Think,* "[t]here is a growing mountain of research. But there is increased evidence that we are being bogged down today as specialization extends." [13] In the same article, Bush sketched a remedy: the 'memex', a hypothetical information management device that would allow users to not only retrieve documents quickly, but to also easily link documents to each other, such that the subsequent retrieval of one document would also let the user effortlessly retrieve those documents linked to it previously. Decades later, the memex was to become a principal inspiration for early hypertext systems and ultimately the World Wide Web [9, 14, 29, 71].

The idea of interlinking documents was not new. Encyclopedic articles, scientific papers, and other types of document have always pointed to one another by means of references. Bush's seminal innovation was that links could be created *ad hoc,* by the reader rather than the writer. The early World Wide Web was still woven together by its writers, rather than its readers; only recently have we achieved Bush's vision of *ad-hoc* linking, through techniques such as social bookmarking [43], tagging [62], and wikis [54].

Information becomes useful to us only when we interact with it, when we disentangle and navigate the intricate networks in which distinct pieces of information relate to one another. Through the ability to fluidly associate items of information with each other and scaffold them into tall buildings of thought, we draw conclusions and are able to make sense of the world. Information becomes knowledge. As Bush put it, "[the human mind] operates by association. With one item in its grasp, it snaps instantly to the next that is suggested by the association of thoughts, in accordance with some intricate web of trails carried by the cells of the brain." [13] Consequently, he suggested to not only store and publish the pairwise links established by a user, but to store and publish entire trails emerging when users navigate the network formed by the links she or others have previously created: "Wholly new forms of encyclopedias will appear, ready made with a mesh of associative trails running through them, ready to be dropped into the memex and there amplified." [13]

In this regard, we still have not achieved Bush's vision yet: while there are projects such as Wikipedia, an open encyclopedia where any user can edit articles and connect related articles in a pairwise fashion, technology and science are still lagging behind with respect to creating, and reasoning about, the "mesh of associative trails running through them". Many important questions have yet to be answered. For instance, how do people interact with information networks such as the World Wide Web? What is the nature of the "associative trails" they embark on? By what strategies do they find information in those networks? How can we infer what users are looking for, and how can we help them find it? How can we optimally guide users through a network to impart on them a deep understanding of a topic? How should we optimally structure those "[w]holly new forms of encyclopedias" that Bush anticipated, and how can we extract a meaningful "mesh of associative trails" from them? How should those trails be gathered, curated, and presented, in order to render them maximally useful?

These are challenging questions, but answering them would yield significant payoffs: information and knowledge are some of our most valuable resources, and building more intuitive and human-friendly information systems will make those resources more easily accessible and will ultimately change our lives for the better.

The goal of this thesis is to make progress toward answering the above questions by

1. furthering our scientific understanding of how humans navigate information networks and
2. turning these insights into useful models and tools for facilitating and enhancing the navigability of those networks.

We shall proceed in a data-driven approach. With billions of users browsing the Web every day, and servers logging their every action, we have detailed records of human behavior in information networks. In particular, stringing records of individual page views together into records of entire trails, or traces, of page views presents the opportunity to study how humans navigate, and find their way through, networks.

The traces extracted from passively collected server logs may be triggered by any of a large number of information needs, and the logs typically do not explicitly specify the information need associated with each trace. This is true even for traces collected within a single website. For instance, users of an online shopping site may be looking for a specific product, or they may be casually browsing the virtual shelves out of boredom. Users of Wikipedia may be consulting the site in order to look up a definition, answer a homework question, decide whether to buy a certain product, settle an argument with a friend, deeply immerse themselves in a topic, or simply explore the encyclopedia by randomly drifting from page to page.

A coarse dichotomy lets us distinguish between two broad modes of navigation in information networks: targeted and exploratory [61]. In targeted navigation the user has a clear question or topic in mind, whereas in exploratory navigation the user's information need is less well defined, and may even change as the session progresses.

Unfortunately, determining the information need addressed by a given trace extracted from passively collected server logs, or even just determining whether it is an instance of targeted or exploratory navigation, is difficult and constitutes itself a challenging research

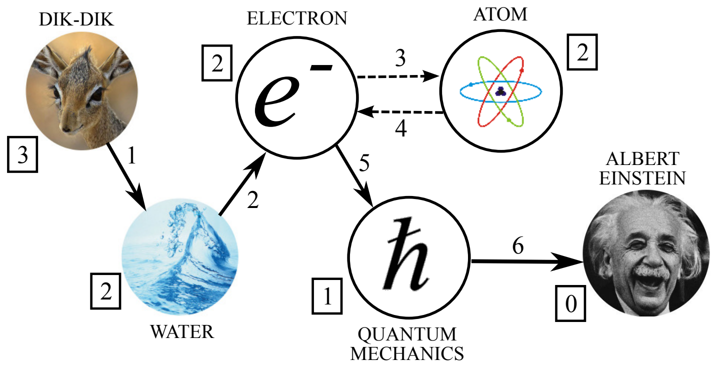


Figure 1.1: Wikispeedia example path between the concepts DIK-DIK and ALBERT EINSTEIN. Nodes represent Wikipedia articles and edges the hyperlinks clicked by the human. Edge labels indicate the order of clicks, the framed numbers the shortest-path length to the target. One of several optimal solutions would be ⟨DIK-DIK, WATER, GERMANY, ALBERT EINSTEIN⟩.

problem. This lack of ground-truth information complicates both the analysis of human information-seeking behavior (item 1 above), as well as the usefulness of traces for building models and tools (item 2).

To circumvent these challenges, prior work in human–computer interaction and information retrieval has tended to focus on small-scale studies and laboratory experiments, often of qualitative nature, where the user is given an explicit Web navigation task and is then observed solving the task [42, 70, 76, 86, 99]. But even in these controlled settings, it remains difficult to reason about certain aspects of human behavior within the network of webpages: There are no simple and concise ways of representing the content of general webpages and of measuring the level of relatedness of pairs of pages in a robust manner. Moreover, the aforementioned studies typically do not have access to the complete connectivity structure of the network being navigated; as a consequence, the full space of actions the users could have taken is unknown, which makes it hard to judge the optimality of their observed search strategies.

We alleviate these problems by leveraging navigation traces harvested from the human-computation game *Wikispeedia* [104], which we have designed and implemented in previous work [109]. Wikispeedia is played within Wikipedia and challenges participants to navigate from a given start article to a given target article by exclusively clicking Wikipedia's article-to-article links, using as few clicks as possible. To illustrate the dynamics of the Wikispeedia game, Fig. 1.1 gives the example of a human path between the start article DIK-DIK and the target ALBERT EINSTEIN.

This setup has the advantage of producing tens of thousands, rather than dozens or hundreds, of navigation traces that are explicitly labeled with the user's navigation target. Moreover, every Wikipedia page is about a clearly defined topic, which gives us a handle for reasoning about the semantic structure of the information space through which users are finding their way; *e.g.*, we may compute the relatedness of two pages by measuring their distance in Wikipedia's category hierarchy, and we may describe the content of a page in one concise phrase—its title. Finally, since the navigation environment is restricted to Wikipedia, we have access to the full connectivity structure of the underlying network.

In this thesis, we first harness the targeted navigation traces collected through Wikispeedia by conducting a detailed analysis of users' navigation strategies. We then strive to turn our insights into useful models and tools for predicting and supporting user behavior and improving the navigability of the underlying network. We argue that such tools for supporting navigation-based search should be rooted in empirically observed user behavior in order to be practically useful, just as the modern web search engines that support query-based search depend in critical ways on models learned from interaction logs [60].

## 1.2 Overview and summary of contributions

The purpose of this thesis is twofold. We strive to (1) deepen our scientific understanding of the patterns and strategies by which users navigate information networks and (2) build models and tools for supporting users as they navigate and for making the underlying networks themselves more navigable.

The overall structure of the thesis is as follows. We start by providing an overview of the most relevant areas of related work in Chapter 2. Then, in Chapter 3, we describe the datasets of navigation traces we use throughout this research. The main technical contributions are made in Chapters 4–7. In Chapter 4 we perform a detailed analysis of the targeted navigation traces collected through the human-computation game Wikispeedia. We then build on the results of this analysis to design models and algorithms for predicting the user's navigation target after observing only the first few clicks (Chapter 5), for navigating Wikipedia automatically (Chapter 6), and for improving the hyperlink structure of websites based on navigation traces collected actively via human-computation games and passively via raw web server logs (Chapter 7). These contributions are summarized in the following chapter-by-chapter outline.

### 1.2.1 Analysis of human navigation traces (Chapter 4)

*Originally published at the 21st International Conference on the World Wide Web [107].*

In order to design efficient and user-friendly information systems, it is important to understand how humans navigate and find the information they are looking for. To this end, we perform a large-scale study of human navigation of information networks in Chapter 4, leveraging the target-labeled navigation traces collected through the human-computation game Wikispeedia. What distinguishes our setup from other studies of human web-browsing behavior [42, 70, 76, 86, 99] is that in our case people navigate a graph of connections between concepts (*viz.*, Wikipedia articles) and that the exact goal of the navigation is known ahead of time.

We study tens of thousands of goal-directed human search paths and identify strategies people use when navigating information spaces. We observe that the solutions found by humans, while mostly very efficient, differ from shortest paths in characteristic ways. Our key findings can be summarized as follows:

1. *Humans manage to find short paths.* In Wikispeedia, participants only observe the links available from the current page and need to decide where to navigate next based solely on this local information. We find that, nevertheless, the solutions found by humans are not much longer than shortest paths on average (median 4 *vs.* 3 clicks).
2. *Making progress is easiest in the early and late navigation phases.* We find that the probability with which users make a click that decreases the shortest-path length to the target is highest for the first and last clicks along a trace, and lowest in between.

3. *Hubs are crucial in the early navigation phase.* The initial progress tends to be afforded by leaping to a 'hub' article of high out-degree. From there, the information seeker has many options to continue the search and continues via nodes of ever decreasing average out-degree.
4. *The conceptual distance to the target decreases steadily.* The articles through which users progress along their traces become ever more related conceptually to the target as the latter is approached.
5. *Users first make big conceptual leaps, followed by ever smaller steps.* Users start by making big leaps through concept space, with adjacent articles being rather unrelated conceptually (*e.g.*, when jumping to the hub); then, as they home in on the target, they straddle ever smaller conceptual gaps.
6. *Clicks are more predictable in the early and late navigation phases.* We measure the predictability of clicks information-theoretically and find that users' decisions are most predictable in the beginning and toward the end of traces. Users' decisions vary most in the middle of traces, when they are close to neither start nor target.
7. *Simple solutions are common, but less efficient.* We observe a trade-off between simplicity and efficiency: conceptually simple solutions are more common, but tend to be longer than more complex ones.

### 1.2.2 Predicting targets of human navigation traces (Chapter 5)

*Originally published at the 21st International Conference on the World Wide Web [107].*

In Chapter 5 we present our first application of the lessons learned in our analysis of human navigation traces. We show how information from a short prefix of the navigation path can be used to predict what target page the information seeker is looking for.

We formulate a Markov model of human browsing, where the probability of the next click $u'$ depends on the current page $u$ and the target page $t$ that the user is attempting to reach. This probability is parameterized as a function of features extracted from $u$, $u'$, and $t$; the features, in turn, are inspired by our empirical analysis of human navigation traces in Chapter 4. To learn good feature weights, we design a learning-to-rank machine learning model and an efficient parameter estimation algorithm. The model is subsequently trained and tested on datasets of real human navigation traces.

The experimental evaluation shows that our model can predict humans' intended targets with high accuracy. Notably, on the task of picking the correct one of two targets, our model achieves an accuracy of 80% when the first three clicks are observed, regardless of whether the entire trace is four, five, or six clicks long.

Several prior works have addressed the task of predicting users' next clicks, rather than their final targets [21, 22, 84, 87, 123]. Our performance on the target prediction task extends the prior literature by showing that features of the underlying path can provide useful information beyond simply predicting the next action of the user. We therefore think that results of our research can be incorporated into intelligent systems to facilitate human information browsing and navigation.

### 1.2.3 Automatic versus human navigation (Chapter 6)

*Originally published at the 6th International AAAI Conference on Weblogs and Social Media [106].*

People regularly face tasks that can be understood as navigation in information networks, where the goal is to find a path between two given nodes. In our human-computation game Wikispeedia, users solve such a task, but it is by far not the only scenario of this kind: during a literature review we follow bibliographical references in order to find particular papers relevant to our own research; we browse the virtual shelves of online stores by clicking from one product description to the next; and we explore social networking sites to find the name of that lovely person at yesterday's party.

In many such situations (*e.g.*, in Wikispeedia), the navigator only gets local access to the node currently under inspection and its immediate neighbors. Despite this lack of global information about the network, and despite the fact that real-world networks are typically very large, humans tend to be good at finding short paths (*cf.* Chapter 4).

One potential reason could be that humans possess vast amounts of background knowledge about the world, which they leverage to make good guesses about possible solutions.

In Chapter 6 we ask the question, Are human-like high-level reasoning skills really necessary for finding short paths? To answer this question, we design a number of navigation agents without such skills, which use only simple numerical features inspired by our analysis of human traces in Chapter 4.

The agents are based on depth-first search, with the additional property that the order in which the current node's successors are pushed on the stack is determined by an evaluation function that aims to capture the chances of making progress toward the target by continuing to the respective successor. This ensures that the most promising click will be greedily chosen next. We develop heuristic as well as machine-learned evaluation functions.

We evaluate the agents on the task of navigating Wikipedia. Since the Wikispeedia game provides us with large-scale human navigation data on this domain, we may readily compare our algorithms to humans. We observe that the agents find shorter paths than humans on average: on our test set, humans need about twice the optimal number of clicks on average, whereas our best automatic agent achieves a factor as low as 1.5. Also, the frequency with which human searches are over twice as long as shortest paths is 4 times as high as for automatic searches. Among all our agents, those based on machine learning fare best. We conclude that, perhaps surprisingly, no sophisticated background knowledge or high-level reasoning is required for navigating the complex Wikipedia network.

We envision that, in the future, automatic navigation agents may be combined with keyword-based search engines, which could give rise to integrated search-and-browse engines for supporting humans in finding desired information on the Web.

### 1.2.4 Improving website hyperlink structure (Chapter 7)

*Originally published at the 24th International Conference on the World Wide Web [108] and the 9th International ACM Conference on Web Search and Data Mining [78].*

Hyperlinks are an essential feature of the World Wide Web. Consider the example of online encyclopedias such as Wikipedia: an article can often only be understood in the context of related articles, and hyperlinks make it easy to explore this context. But important links are often missing, and several methods have been proposed to alleviate this problem by learning a linking model based on the structure of the existing links [31, 63, 64, 68, 110, 118].

In Chapter 7 we propose novel methods for identifying missing hyperlinks on websites. We build on the fact that the ultimate purpose of hyperlinks is to aid navigation. Rather than merely suggesting new links that are in tune with the structure of existing links, our methods leverage real human navigation traces with the goal of finding missing links that would immediately enhance the site's navigability.

We develop two such methods, one that is specifically designed for targeted navigation traces, such as those harvested from Wikispeedia (Sec. 7.2), and another that works on general navigation traces as may be extracted from the access logs collected passively by any web server software (Sec. 7.3). Both methods work by first mining the logs for missing-link candidates and then ranking these candidates by their expected future usefulness.

Given a set of useful link candidates, the task of incorporating them into the site can be expensive, since it typically involves humans editing pages. To explicitly take into account the cost of humans in the loop, we further define the combinatorial optimization problem of link placement under budget constraints and propose an efficient algorithm for solving the problem.

We demonstrate the effectiveness of our two approaches by evaluating them on Wikipedia. In an evaluation based on historical data we find that 60% (25%) of the 1,000 (10,000) link suggestions ranked highest by our algorithm of Sec. 7.3 are later on organically added by human Wikipedia editors independent of our research, which clearly shows that our recommendations are of high quality. Moreover, our top 1,000 (10,000) suggestions receive 95,000 (336,000) clicks in the month after being added—12 times as many as the average link added by human Wikipedia editors. Our system is being deployed on Wikipedia's full server logs at terabyte scale and feeds recommendations to a link-adding plugin for Wikipedia's visual editing tool [77].

Finally, since our second method is based exclusively on standard, passively collected web server logs, it may also be applied to other websites, as we show with the example of the biomedical research site Simtk.

# Chapter 2

# Background and related work

Network navigation is an important topic that has been investigated by researchers in a number of subfields of computer science and beyond. Our work has rich connections to prior research done in psychology, cognitive science, and computer science on information foraging (Sec. 2.1) and decentralized search in small-world networks (Sec. 2.2); in data mining, machine learning, and information retrieval on Markov modeling of website navigation (Sec. 2.3), search trail analysis (Sec. 2.4), and link recommendation and prediction (Sec. 2.5); and in human–computer interaction on games with a purpose (Sec. 2.6). This chapter reviews, and places our work in the context of, these lines of prior work.

## 2.1 Information foraging

Inspired by the nomenclature of ecology, where animals are classified as herbivores, carnivores, omnivores, *etc.*, humans have been termed 'informavores' [66], since, just as animals hunt for food, humans have developed mechanisms for hunting for information. Information foraging theory [32, 80, 81] posits that the principles that govern animal food-seeking behaviors also apply to human information-seeking behaviors. In particular, as animals follow scent signals in order to locate food, information foraging theory maintains that humans follow a notion of 'information scent' [15] in order to locate information. The theory further assumes that information seekers trade off the expected value of a piece information (as indicated by information scent) with the cost necessary to obtain the desired piece of information by interacting with the environment [79].

The notion of information scent has been implemented in navigation systems in order to improve the usability of user interfaces such as websites; *e.g.*, ‘ScentTrails’ [76] is a framework that blends query- and navigation-based search smoothly, inspired by the intuition that navigating becomes useful whenever keyword queries are infeasible or unsuccessful.

Observational and laboratory studies have conducted small-scale controlled experiments about users’ thought processes during web search by having them think aloud as they search [70], and about their interaction with information on the Web [86]. Related to these studies are descriptive models, such as orienteering [75] and berrypicking [8], to describe users’ information-seeking strategies. Systems like ‘ScentTrails’ [76] and guided tours [99] have been proposed to create annotations to indicate where other users have navigated in the past, all with the goal of helping people find information faster. Other assistive browsing tools include systems such as ‘WebWatcher’ [40], which highlights hyperlinks based on user goals inferred from an initial keyword query and subsequent clicks.

Our present work has deep connections to information foraging theory. For instance, in the game of Wikispeedia, players ‘hunt’ for a prescribed piece of information, while attempting to minimize the cost of interacting with the environment (by minimizing the number of clicks). Our analyses show that players approach the target by navigating through pages that become increasingly more related to the sought-after information, which may be interpreted as users following a notion of information scent. Our work extends prior work in information foraging theory in two important ways: First, our goal is not to formulate an analogy for human navigation, but rather to analyze it computationally using large-scale collections of real search traces (Chapter 4). Second, we go beyond merely analyzing traces, by leveraging the insights from our analyses of human navigation traces to design algorithms for the tasks of predicting a user’s navigation target, navigating toward a target automatically, and improving website navigability by inserting novel links (Chapters 5–7).

## 2.2 Decentralized search in small-world networks

Research on decentralized search is rooted in Milgram's seminal 1967 'small-world' experiment [65], whose goal it was to determine by how many degrees of separation people were apart in the human social network on average. In the experiment, letters were given to participants in various places in the United States. Participants were instructed to forward their letters to friends, who were to repeat the same process, such that the letters would eventually reach a predestined target person. The now famous result was that on average only six hops were required to route letters between distant members of the social network (hence the name 'small-world experiment')—a remarkable fact, considering that all participants only have knowledge about their local network neighborhoods.

The small-world experiment has spawned much subsequent research [2, 25, 36, 44, 52, 53, 57, 89, 98]. Watts and Strogatz [103] characterize the structure of small-world networks (such as the human social network) as having both short average path lengths and high local clustering. They also develop a model for generating networks with both of these properties.

Kleinberg [46, 47] investigates the algorithmic aspects of the small-world problem, asking the question what properties the network needs to have such that a simple, locally greedy routing algorithm manages to find short paths on average (defined as polylogarithmic in the number of nodes). Kleinberg assumes that nodes are arranged in a $k$-dimensional lattice with connections to all nearest neighbors, with some additional long-range connections sampled randomly from an underlying probability distribution. He shows that, if the greedy algorithm is to find short paths, then the probability of a long-range link spanning distance $d$ in the lattice must decay as $d^{-k}$. In other words, Kleinberg shows that an exact amount of homophily is both necessary and sufficient for short average paths to exist. Liben-Nowell *et al.* [57] extend this result to non-lattice networks.

The work presented in this thesis builds on the foundations laid by these prior works. The human-computation game we use for data collection may be considered an instance of decentralized search in a network, as players try to navigate between given start and target pages using only the local information provided on the current page (players see, and can follow, only hyperlinks of the current page). However, in the small-world experiment, search is in a sense even more decentralized, since each node—*i.e.*, human—on the path independently forwards the message and then forfeits control. While in our navigation game the information seeker also has only local knowledge about the unknown network, she stays in control all the way and can thus form more elaborate strategies than in the multi-person scenario.

Our present work also extends prior research in several other ways. First, the original small-world experiment was very small-scale: only 160 chains were started, and only 44 reached the target. Even a more recent, larger-scale repetition of the experiment, due to Dodds *et al.* [25], managed to collect only 384 completed chains. On the contrary, we work with tens of thousands of completed chains, which enables considerably more fine-grained analyses (Chapter 4). Second, we extend the scope to information networks, whereas prior work has primarily considered social networks, and show that the Wikipedia information network is navigable because it is connected by a nearly perfect mix of short- and long-range links (Sec. 4.2), in line with the conditions determined by Liben-Nowell *et al.* [57]. Finally, we investigate not only how local navigation algorithms can manage to find short paths (Chapter 6), but also how humans manage to do so (Chapter 4).

## 2.3 Markov modeling of website navigation

Navigation traces consist of sequences of webpage visits. A common approach to probabilistic sequence modeling is provided by Markov chain models, which assume that the probability of the page visited next depends only on a finite history of previously visited pages. For this reason, there is a long tradition of Markovian web navigation models [11, 21, 22, 27, 55, 87, 90, 123].

Chierichetti *et al.* [16] evaluate which Markov chain order (*i.e.*, history length) is best suited for predicting the user's next click, finding that human browsing is around 10% more predictable by Markov chains of an order larger than one. They also provide an algorithm for efficiently computing the variable-order Markov chain that best describes observed navigation traces (under the constraint that the number of probability distributions that may be kept in memory is fixed).

In terms of applications, Markov chains have primarily been used to predict users' next clicks [21, 22, 84, 87, 123]. Other applications include predicting whether users will give up in information network navigation [85] and computing the semantic relatedness of concepts [20, 91, 109]. We extend this body of work by devising Markovian methods for navigating automatically (Chapter 6), predicting the user's ultimate navigation target (Chapter 6), and recommending new links to add to websites (Chapter 7).

## 2.4 Search trail analysis

Another line of related work pertains to the analysis of so-called 'search trails'. This research, conducted primarily by the information retrieval community, studies the click paths on which users embark starting from search-engine result pages. Search trails can be used as endorsements to rank search results more effectively [10, 92], trail destination pages can themselves be used as search results [113], and the concept of teleportation can be used to navigate directly to the desired page [95].

The importance of studying users' navigation traces is emphasized by an influential study due to Teevan *et al.* [95], which shows that users frequently prefer click-based navigation to searching via keyword queries, even when they know their exact search target. One factor contributing to the attractiveness of navigating is that, even once an initial search query has been issued, further navigation starting on the search-engine result page is useful because it adds value to the content of the search results themselves, as established by White and Huang [113]. Downey *et al.* [26] investigate the benefits of navigating *vs.* querying further, finding that navigating is particularly useful when the information need is rare. Finally, White and Singla [114] explore how trail topology (such as star, tree, linear chain, *etc.*) varies between search scenarios with different information needs, such as informational *vs.* navigational.

A core difference between prior work and the research presented here is that we analyze tens of thousands of navigation traces that are each labeled with the exact target the user had in mind, whereas the datasets used in prior work have typically either been small [8, 75, 95] or not been labeled with the explicit navigation target [26, 113, 114].

## 2.5 Link recommendation and prediction

There is a rich literature on the problem of recommending new links, and predicting which links will form in the future, in networks. Unsupervised methods for link prediction in social networks were extensively evaluated by Liben-Nowell and Kleinberg [56], who found the Adamic–Adar measure [1], which is based on numbers of common neighbors, to perform best. More recently approaches based on network community detection [17, 38, 45] and random walks [7] were considered for predicting missing links. Supervised link prediction [58] was also studied by the relational-learning community [82, 94], but scalability remains a challenge with these approaches. For further background on the link prediction problem we refer the reader to the excellent survey by Liben-Nowell and Kleinberg [56].

While the above works focused mostly on the identification of missing links in social networks, there is also a rich line of work on the identification of missing links among Wikipedia articles [31, 33, 74, 110, 118] and on linking existing webpages to Wikipedia [63, 64, 68]. Generally these approaches focus on building models of Wikipedia's graph structure, while also performing keyword extraction and word-sense disambiguation.

Our work (Chapter 7) differs from the aforementioned methods in one important aspect: most prior methods (with the exception of Grecu's [33]) base their predictions on the static structure of the network, whereas we focus on how users interact with this static structure by browsing and searching on it. We show that signals extracted from usage logs are important indicators of the usefulness of a link. Moreover, web server access logs may serve as a sensor of real-world events and mirror users' information needs in real time, so by being based on such logs, our methods manage to make link recommendations that can improve website navigability in a timely manner, given current world events.

## 2.6 Games with a purpose

The bulk of analyses conducted in this thesis involve targeted navigation traces collected through the human-computation game Wikispeedia [104, 105, 109]. Such games, which are fun to play and at the same time solve a useful task or collect valuable information, have also been termed ‘games with a purpose’ [102]. They were popularized by von Ahn and colleagues, a seminal early example being the ‘ESP Game’ for labeling images [101].

Wikispeedia was originally designed in 2009 as a game with a purpose for computing the semantic relatedness between concepts [109]. Since then, several other games have been proposed for solving various other semantic tasks, such as extracting commonsense knowledge [39], improving search engines [59], and validating and extending semantic knowledge bases [41, 100], among others. For a survey on games for semantic knowledge acquisition, we refer the reader to Thaler *et al.* [96].

Further relevant work was done by Ageev *et al.* [5], who developed a human-computation game for collecting data in which users are asked to find the answers to as many factual questions (*e.g.*, ‘What is the highest peak in the Western Hemisphere?’) as possible within a given amount of time, using web search queries that may optionally be followed by click-based navigation. As in our navigation dataset collected through Wikispeedia, the goal is explicitly known here, but not in the form of a specific target page but rather in the form of a specific answer string.

# Chapter 3

# Obtaining datasets of navigation traces

The research presented in this thesis leverages datasets of human navigation traces. Such traces may be extracted from raw logs as collected by any standard web server software, such as Apache HTTP Server, Microsoft Internet Information Services, Nginx, Squid, Varnish, *etc.* Logs of this kind capture users' natural browsing behavior, and are therefore a valuable mirror of users' information-seeking patterns.

Navigation traces extracted from raw server logs cannot, however, tell us what exact information need, or target page, the user had in mind. In fact, it might not even be the case that the user had a clearly defined navigation target to begin with, *e.g.*, when engaging in exploratory search [61]. Hence, we cannot use passively collected server logs to analyze how users perform targeted navigation, but need to actively collect traces ourselves instead. We do so by framing the task of targeted navigation as a human-computation game.

In the remainder of this chapter, we first describe our approach for mining navigation traces from passively collected web server logs (Sec. 3.1) and then proceed to explaining how we use online games to actively collect traces with explicitly known navigation targets (Sec. 3.2). We also summarize some key properties of the datasets collected via the two approaches.

## 3.1 Obtaining navigation traces from raw web server logs

In this section we describe the structure and processing of web server logs. We also discuss the properties of the two websites whose logs we work with, Wikipedia and Simtk.

### 3.1.1 From logs to trees

**Log format.** Web server log files contain one entry per HTTP request, specifying *inter alia:* timestamp, requested URL, referer URL, HTTP response status, user agent information, client IP address, and proxy server information via the HTTP X-Forwarded-For header. Since users are not uniquely identifiable by IP addresses (*e.g.*, several clients might reside behind the same proxy server, whose IP address would then be logged), we create an approximate user ID by computing an MD5 digest of the concatenation of IP address, proxy information, and user agent. Common bots and crawlers are discarded on the basis of the user agent string.

**From logs to trees.** Our goal is to analyze the traces users take on the hyperlink network of a given website. However, the logs do not represent these traces explicitly, so we first need to reconstruct them from the raw sequence of page requests.

We start by grouping requests by user ID and sorting them by time. If a page $t$ was requested by clicking a link on another page $s$, the URL of $s$ is recorded in the referer field of the request for $t$. The idea, then, is to reassemble the original traces from the sequence of page requests by joining requests on the URL and referer fields while preserving the temporal order. Since several clicks can be made from the same page, *e.g.*, by opening multiple browser tabs, the navigation traces thus extracted are generally trees.

While straightforward in principle, this method is unfortunately unable to reconstruct the original trees perfectly. Ambiguities arise when the page listed in the referer field of $t$ was visited several times previously by the same user. In this situation, linking $t$ to all its potential predecessors results in a directed acyclic graph (DAG) rather than a tree. Transforming such a DAG into a tree requires a heuristic approach. We proceed by attaching a request for $t$ with referer $s$ to the *most recent* request for $s$ by the same user.

If the referer field is empty or contains a URL from an external website, the request for $t$ becomes the root of a new tree.

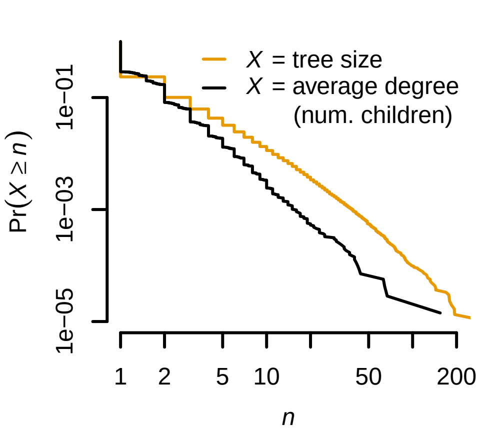
Not only is this greedy strategy intuitive, since it seems more likely that the user continued from the most recent event, rather than resumed an older one; it also comes with a global optimality guarantee. More precisely, we observe that in a DAG $G$ with edge weights $w$, attaching each internal node $v$ to its minimum-weight parent $\text{argmin}_u w_{uv}$ yields a minimum spanning tree of $G$. We follow this approach with time differences as edge weights. Hence, the trees produced by our heuristic are the best possible trees under the objective of minimizing the sum (or mean) of all time differences spanned by the referer edges.

**Mining search events.** Navigation traces are frequently interspersed with keyword searches that can provide important signals regarding the user's information need. In our analyses, we consider both site-internal search (*e.g.*, from the Wikipedia search box when using Wikipedia server logs) and site-external search from general engines such as Google, Yahoo, and Bing. Mining internal search is straightforward, since all search actions are usually fully represented in the logs. An external search from $s$ for $t$ is defined to have occurred if $t$ has a search engine as referer, if $s$ was the temporally closest previous page view by the user in the logs, and if $s$ occurred at most 5 minutes before $t$.

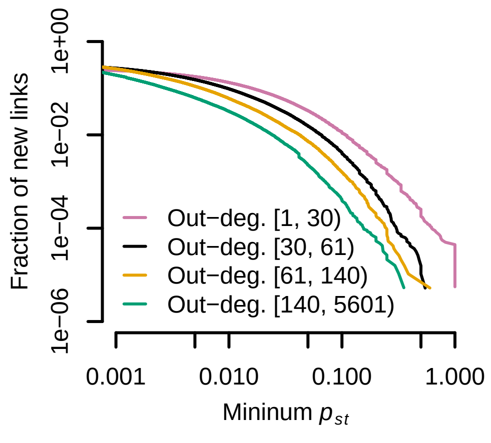
## 3.1.2 Wikipedia data

**Link graph.** Navigation traces are produced in the context of an underlying link graph. In order to analyze a user's browsing behavior, it is important to know which links were available to the user at navigation time. In the case of Wikipedia, the link graph is defined by nodes representing articles in the main namespace, and edges representing the hyperlinks used in the bodies of articles. Furthermore, we use the English Wikipedia's publicly available full revision history (spanning all undeleted revisions from 2001 through April 3, 2015) to determine when links were added or removed.

**Server logs.** We have access to Wikimedia's full server logs, containing all HTTP requests to Wikimedia projects. We consider only requests made to the desktop version of



(a) Tree properties



(b) New-link usage

Figure 3.1: Wikipedia dataset statistics. **(a)** CCDF of size and average degree of navigation trees. **(b)** CCDF of clickthrough rate for various source-page out-degrees. (Log–log scales.)

the English Wikipedia. The log files we analyze comprise three months of data, from January through March 2015. For each month we extracted around 3 billion navigation trees. Fig. 3.1(a) summarizes structural characteristics of trees by plotting the complementary cumulative distribution functions (CCDF) of the tree size (number of page views) and of the average degree (number of children) of non-leaf page views per tree. We observe that trees tend to be small: 77% of trees consist of a single page view; 13%, of two page views; and 10%, of three or more page views. Average degrees also follow a heavy-tailed distribution. Most trees are linear chains of all degree-one nodes, but page views with larger numbers of children are still quite frequent; *e.g.*, 6% (44 million per month) of all trees with at least two page views have an average degree of 3 or more.

### 3.1.3 Simtk data

Our second dataset of passively collected web server logs stems from Simtk.org, a website where biomedical researchers share code and information about their projects. We analyze logs from June 2013 through February 2015.

Contrasting Simtk and Wikipedia, we note that the Simtk dataset is orders of magnitude smaller than the Wikipedia dataset, in terms of both pages and page views, at hundreds, rather than millions, of pages, and tens of millions, rather than tens of billions, of page views. Simtk's hyperlink structure is also significantly less dense than Wikipedia's. On the one hand, this means that there is much improvement to be made by a method such as ours; on the other hand, it also means that the navigation traces mined from the logs are less rich than for Wikipedia. Therefore, instead of extracting trees, we extract *sessions,* defined here as sequences of page views with idle times of no more than one hour [34] between consecutive events.

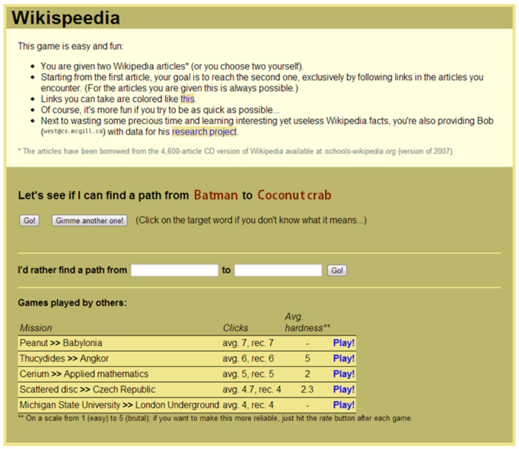
## 3.2 Collecting targeted navigation traces via human-computation games

Our datasets of targeted navigation traces were collected via a popular online game that is generically known as 'Wikiracing' [117]. Several websites offer versions of this game, such as *Wikispeedia* [104, 109] or *The Wiki Game* [18], but they all share the same general idea: a user is given two Wikipedia articles—a *start* and a *target*—and is asked to navigate from the start to the target by exclusively clicking hyperlinks contained in the visited pages. We also refer to start–target pairs as *missions*. In our experiments we use data from both Wikispeedia (Chapters 4–7) and The Wiki Game (Chapter 7). Here we describe the two games and provide more details about the datasets collected through them.

### 3.2.1 Wikispeedia

Wikispeedia is a single-player game. The user's task is to navigate from the start to the target in as few clicks as possible. Using the back button of the browser is permitted, and clicks that were undone by using the back button do not count toward the number of clicks needed. Once a mission is successfully completed, the user may enter her name into a high-score table associated with that mission, where users are ranked by number of clicks, with ties broken by time. (See Fig. 3.2 for screenshots.)

The game is played on a reduced, static snapshot of Wikipedia [115] containing 4,604 of the most important articles and about 120,000 links. Missions, *i.e.*, start–target pairs, are sampled uniformly at random from the strongly connected core component comprising



Wikispeedia

This game is easy and fun:

- You are given two Wikipedia articles* (or you choose two yourself).
- Starting from the first article, your goal is to reach the second one, exclusively by following links in the articles you encounter. (For the articles you are given this is always possible.)
- Links you can take are colored like this.
- Of course, it's more fun if you try to be as quick as possible...
- Next to wasting some precious time and learning interesting yet useless Wikipedia facts, you're also providing Bob (`west@cs.mcgill.ca`) with data for his research project.

* The articles have been borrowed from the 4,600-article CD version of Wikipedia available at *schools-wikipedia.org* (version of 2007).

**Let's see if I can find a path from Batman to Coconut crab**

(Click on the target word if you don't know what it means...)

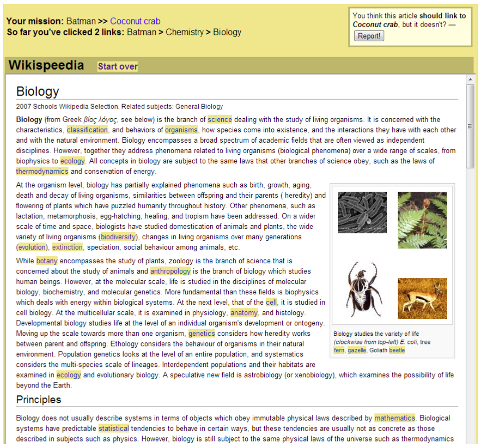
**I'd rather find a path from** ______ **to** ______

**Games played by others:**

| *Mission* | *Clicks* | *Avg. hardness*** | |
|---|---|---|---|
| Peanut **>>** Babylonia | avg. 7, rec. 7 | - | **Play!** |
| Thucydides **>>** Angkor | avg. 6, rec. 6 | 5 | **Play!** |
| Cerium **>>** Applied mathematics | avg. 5, rec. 5 | 2 | **Play!** |
| Scattered disc **>>** Czech Republic | avg. 4.7, rec. 4 | 2.3 | **Play!** |
| Michigan State University **>>** London Underground | avg. 4, rec. 4 | - | **Play!** |

** On a scale from 1 (easy) to 5 (brutal); if you want to make this more reliable, just hit the *rate* button after each game.

(a) Before navigation starts



**Your mission:** Batman **>>** Coconut crab
**So far you've clicked 2 links:** Batman **>** Chemistry **>** Biology

You think this article **should link to** ***Coconut crab***, but it doesn't? —

Wikispeedia

Biology

2007 Schools Wikipedia Selection. Related subjects: General Biology

**Biology** (from Greek *βίος λόγος*, see below) is the branch of science dealing with the study of living organisms. It is concerned with the characteristics, classification, and behaviors of organisms, how species come into existence, and the interactions they have with each other and with the natural environment. Biology encompasses a broad spectrum of academic fields that are often viewed as independent disciplines. However, together they address phenomena related to living organisms (biological phenomena) over a wide range of scales, from biophysics to ecology. All concepts in biology are subject to the same laws that other branches of science obey, such as the laws of thermodynamics and conservation of energy.

At the organism level, biology has partially explained phenomena such as birth, growth, aging, death and decay of living organisms, similarities between offspring and their parents ( heredity) and flowering of plants which have puzzled humanity throughout history. Other phenomena, such as lactation, metamorphosis, egg-hatching, healing, and tropism have been addressed. On a wider scale of time and space, biologists have studied domestication of animals and plants, the wide variety of living organisms (biodiversity), changes in living organisms over many generations (evolution), extinction, speciation, social behaviour among animals, etc.

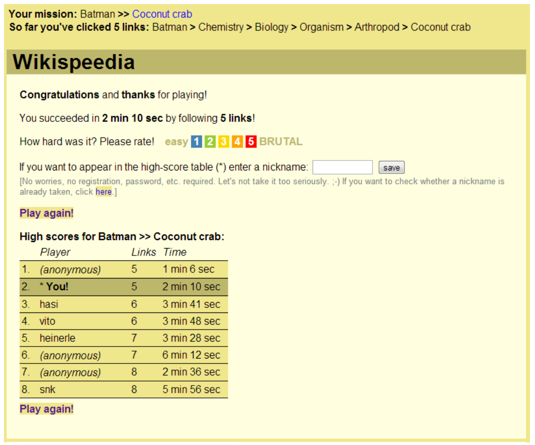
Biology studies the variety of life *(clockwise from top-left) E. coli*, tree fern, gazelle, Goliath beetle

While botany encompasses the study of plants, zoology is the branch of science that is concerned about the study of animals and anthropology is the branch of biology which studies human beings. However, at the molecular scale, life is studied in the disciplines of molecular biology, biochemistry, and molecular genetics. More fundamental than these fields is biophysics which deals with energy within biological systems. At the next level, that of the cell, it is studied in cell biology. At the multicellular scale, it is examined in physiology, anatomy, and histology. Developmental biology studies life at the level of an individual organism's development or ontogeny. Moving up the scale towards more than one organism, genetics considers how heredity works between parent and offspring. Ethology considers the behaviour of organisms in their natural environment. Population genetics looks at the level of an entire population, and systematics considers the multi-species scale of lineages. Interdependent populations and their habitats are examined in ecology and evolutionary biology. A speculative new field is astrobiology (or xenobiology), which examines the possibility of life beyond the Earth.

Principles

Biology does not usually describe systems in terms of objects which obey immutable physical laws described by mathematics. Biological systems have predictable statistical tendencies to behave in certain ways, but these tendencies are usually not as concrete as those described in subjects such as physics. However, biology is still subject to the same physical laws of the universe such as thermodynamics

(b) During navigation



**Your mission:** Batman **>>** Coconut crab
**So far you've clicked 5 links:** Batman **>** Chemistry **>** Biology **>** Organism **>** Arthropod **>** Coconut crab

Wikispeedia

**Congratulations** and **thanks** for playing!

You succeeded in **2 min 10 sec** by following **5 links**!

How hard was it? Please rate! easy 1 2 3 4 5 BRUTAL

If you want to appear in the high-score table (*) enter a nickname: ______

[No worries, no registration, password, etc. required. Let's not take it too seriously. ;-) If you want to check whether a nickname is already taken, click here.]

**Play again!**

**High scores for Batman >> Coconut crab:**

| | *Player* | *Links* | *Time* |
|---|---|---|---|
| 1. | *(anonymous)* | 5 | 1 min 6 sec |
| 2. | * **You!** | 5 | 2 min 10 sec |
| 3. | hasi | 6 | 3 min 41 sec |
| 4. | vito | 6 | 3 min 48 sec |
| 5. | heinerle | 7 | 3 min 28 sec |
| 6. | *(anonymous)* | 7 | 6 min 12 sec |
| 7. | *(anonymous)* | 8 | 2 min 36 sec |
| 8. | snk | 8 | 5 min 56 sec |

**Play again!**

(c) After successful navigation

Figure 3.2: Screenshots of the human-computation game Wikispeedia, which we use for data collection (Sec. 3.2.1).

4,051 articles (98% of all articles), such that every mission has a solution. Alternatively, users may specify their own mission or may choose from a list of missions played recently by other users. The mean shortest-path length between article pairs is as small as 3.2, with a median of 3, and a maximum of 9.

The dataset we work with is publicly available [105] and comprises about 51,000 paths collected from 2008 to 2014,[1] grouped into about 29,000 distinct missions, for an average of 1.8 paths per mission. The number of distinct targets is 3,326; *i.e.*, we have 15 paths per target on average, with a median of 10.

For an example of a trace collected via Wikispeedia, consider again Fig. 1.1, where the mission is to navigate from the start DIK-DIK to the target ALBERT EINSTEIN. With the third click, the information seeker navigates from ELECTRON to ATOM, but backs up after not finding the link to the target that she expected there. We call the sequence including ATOM and the *back-click* the *full path*, while referring to ⟨DIK-DIK, WATER, ELECTRON, QUANTUM MECHANICS, ALBERT EINSTEIN⟩ as the *effective path*. The shortest-path length from every article to the target is shown in squares in the picture. If a click decreases the shortest-path length, we call it *lucrative*.

### 3.2.2 The Wiki Game

Unlike Wikispeedia, The Wiki Game is a multi-player game. Users may choose from five challenges: 'least clicks' (minimize the number of clicks), 'speed race' (minimize time), 'five clicks [or fewer] to Jesus' (find JESUS in five or fewer clicks, minimizing time), 'no United States' (minimize time while avoiding the USA article), and 'six degrees of Wikipedia' (minimize time while finding a path of exactly six clicks). In each challenge, several players compete for the same mission simultaneously, navigating the full Wikipedia.

We pool the paths collected from all five challenges between 2009 and 2012, thus obtaining a dataset of about 974,000 paths grouped into about 364,000 distinct missions (start–target pairs); *i.e.*, there are 2.7 paths per mission on average. The number of distinct targets is 3,219, *i.e.*, we have 303 paths per target on average, with a median of 208. Targets with many paths are quite frequent; *e.g.*, there are 2,087 targets with at least 100, and 698

[1]Some chapters of this thesis use a slightly older, and thus smaller, dataset. Where this is the case, we explicitly note it.

targets with at least 500, paths. When using the dataset collected via The Wiki Game in Sec. 7.2, we focus on the 2,087 targets (65% of all targets) that have at least 100 paths.

# Chapter 4

# Analysis of human navigation traces

## 4.1 Introduction

As humans are navigating an information network, they use various strategies rooted in their commonsense and world knowledge (which lets them estimate, *e.g.*, how topically related two pages or concepts are), as well as expectations about the connectivity structure of the network itself. It is important to understand these strategies as well as the role of the interplay between the relatedness of pages and the underlying network structure, since this could give us important insights about the methods used by efficient information seekers. Understanding how users navigate sheds light on their ways of thinking about the world and lets us design information spaces that are better aligned with these ways of thinking.

From a practical viewpoint, there is useful information in the trail an information seeker has navigated so far, even before reaching the target. Such trails play an important role in the development of methods that can analyze the path taken so far and provide information seekers with navigational aids. One useful direction for this is in predicting what piece of information the information seeker is trying to locate. Another is in automatically detecting if the user has gotten lost [85]. Given that human navigation of information networks is so ubiquitous, a better understanding of the methods according to which humans find connecting paths would have applications in improving the design of information spaces [75], more intuitive and navigable link structures [37], and new intelligent information navigation systems [76].

These broad issues suggest a wide range of interesting open questions. In this chapter we take a step toward answering those questions by computationally analyzing how people navigate to specific target pages in the Wikipedia information network. As a tool we use the online human-computation game Wikispeedia (Sec. 3.2).[1] Since players have no knowledge of the global network structure, they must rely solely on the local information they see on each page—the outgoing links connecting the current article to its neighbors—and on their expectations about which articles are likely to be interlinked. In this respect, the task humans are trying to solve at each visited article is that of guessing which of the outgoing links to follow in order to eventually reach the target article.

What makes our study unique is that, for every instance we know the starting article and the given target article the user is trying to reach. Hence, we do not have to infer or guess the information need of the information seeker, but can base our methods on the ground truth instead.

To illustrate potential reasoning schemes and classes of strategies humans might use, consider again Fig. 1.1, which shows a human path between the start article DIK-DIK and the target ALBERT EINSTEIN. Note that, in this example, not every click is lucrative; rather, the information seeker makes progress by decreasing the shortest-path length to the target ALBERT EINSTEIN at first, but then orbits at a shortest-path length of 2, before finally gravitating towards the target with the choice of QUANTUM MECHANICS. We also emphasize the special role the article on WATER plays in the example. It connects to many parts of the network—hence we call it a *hub*—and marks the transition between getting away from the animal kingdom and homing in on the realm of physics.

**Chapter outline.** This chapter is structured as follows. First, we show in Sec. 4.2 that, despite the lack of global knowledge, humans are good at connecting the dots: the median human game path is only one click longer than the median optimal solution. We explain this effect by showing that certain properties of Wikipedia's hyperlink structure make it easily navigable. Then, in Sec. 4.3, we conduct a detailed analysis of human navigation strategies, showing that people commonly find their way by first locating a hub and by constantly decreasing the conceptual distance to the target thereafter. While approaching the target through a series of conceptually very related articles is safer and often humans'

[1]In this chapter we use a dataset of around 30,000 traces collected from 2008 to 2012.

preferred solution (*cf.* the example of Fig. 1.1), it is typically not the most efficient: we find that thinking 'out of the box' often allows information seekers to find shorter paths between concepts—at the risk of getting lost. A strategy that is both popular and often successful is to connect concepts in terms of their geographical commonalities. In the above example, ⟨DIK-DIK, AFRICA, EUROPE, GERMANY, ALBERT EINSTEIN⟩ would have been such a solution.

## 4.2 Efficiency of human search

The Wikipedia graph is an example of a 'small world' in which most pairs of nodes are connected by short chains, with a mean/median/max *shortest-path length* (SPL) across all pairs of 3.2/3/9. A natural first question to ask is, How good are humans at finding such short chains?

Fig. 4.1 gives a good impression of how the paths found by humans compare to optimal solutions (summary statistics of the distributions in the figure are provided in Table 4.1). The red line shows the distribution of human path lengths (where clicks that were later undone and back-clicks are counted as regular clicks), while effective paths were used for the blue line. For each human game we also computed an optimal solution, and the resulting path length distribution is plotted as a black line. We make three observations:

1. The variance in search time is much larger for human than for optimal solutions. While the distribution of optimal path length is tight around 3 clicks, the human distribution exposes a heavy tail.
2. Nonetheless, the effective paths found by humans (the blue line in Fig. 4.1) are typically not much longer than shortest paths. Both mode and median search times differ from optimal by just 1 click (3 *vs.* 4 clicks), mean search time by 2 clicks (2.9 *vs.* 4.9 clicks). (See Table 4.1.)
3. When considering full path length with undone and back-clicks (the red line in Fig. 4.1), the mode search time is still 4, and the mean and median search times are 1 click more than for effective paths (5 *vs.* 4, and 5.8 *vs.* 4.9 clicks). That is, humans click back on average once every other game.

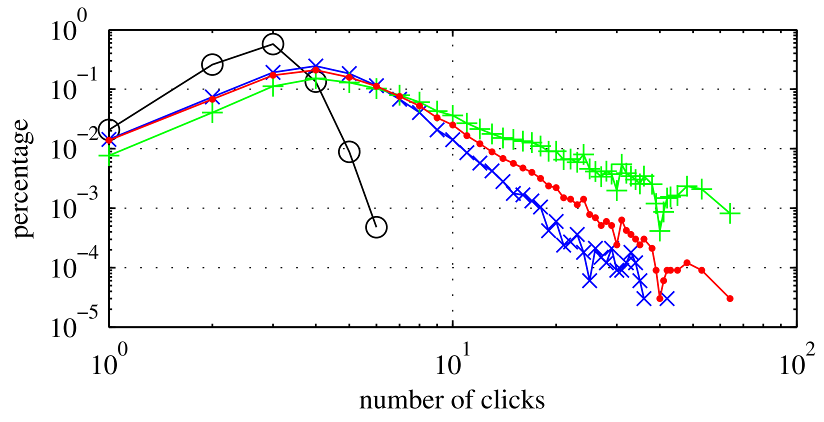


Figure 4.1: Distribution of Wikispeedia game length, according to different path-length metrics. **Black circles:** shortest possible paths. **Blue X's:** effective human paths (*i.e.*, ignoring back-clicks). **Red dots:** complete human paths (*i.e.*, including back-clicks). **Green plus signs:** complete human paths, corrected for drop-out rates.

| Path-length metric | Mode | Median | Mean |
|---:|:-:|:-:|:-:|
| Shortest possible paths | 3 | 3 | 2.9 |
| Human, effective | 4 | 4 | 4.9 |
| Human, incl. back-clicks | 4 | 5 | 5.8 |
| Human, drop-out–corrected | 4 | 6 | 8.9 |

Table 4.1: Summary statistics of the distributions of Fig. 4.1.

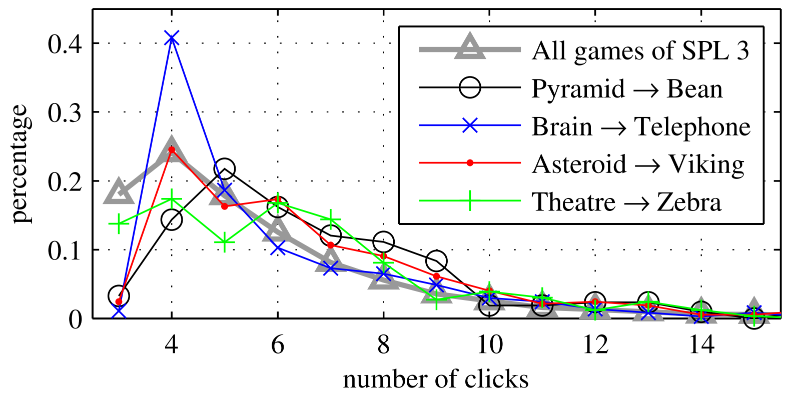


Figure 4.2: Distribution of game length for four specific missions with an optimal solution of 3 clicks. We recorded between 216 and 376 paths per mission. The gray curve shows the length distribution for all games with an optimal solution of 3 clicks.

Two questions arise: First, what is the reason for the large variance in human search time? Second, why is human search still so efficient on average?

The first question permits two potential answers. Either some missions are inherently harder than others, or some information seekers are better than others. Some missions have longer optimal solutions than others, so necessarily some games are inherently harder. However, even when restricting ourselves to missions of a fixed SPL, the numbers stay virtually unchanged (*e.g.*, for games with a SPL of 3 clicks, the mode/mean/median is 4/5/6.0, as opposed to 4/5/5.8 for all games). Of course, even among missions of a fixed SPL, some are harder for humans because the lucrative links might be less obvious. To control such effects, we posted four missions—all of SPL 3—on the game website with increased frequency. This allows us to find out how different humans perform on the exact same task. The search time distributions for the four frequent missions are plotted in Fig. 4.2. We see that for each separate mission there is considerable search time variance, but also that some missions allow for shorter games on average than others. This leads us to conclude that both hardness of mission and individual skill play a roll in explaining the large search time variance.

Regarding the second question, too,—Why is human search so efficient on average?—several answers are conceivable. One might argue that the efficiency of observed games is caused by a sampling bias. In studies that collect data from human volunteers, one

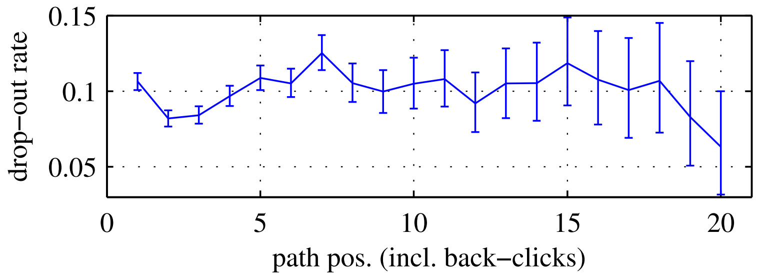


Figure 4.3: Drop-out rate in Wikispeedia as a function of path position (with 95% confidence intervals). At each step, players give up with a probability of around 10%.

always faces the problem of participants dropping out before finishing the task assigned. In our case, this might result in a bias towards observing shorter chains than what we would observe by forcing participants to finish all tasks, since the longer the game takes, the more likely the subject is to give up at some point. For instance, 54% of all games in our dataset were canceled before finishing. Fig. 4.3 shows that the drop-out rate $R_i$, *i.e.*, the probability of giving up at the $i$-th step,[2] is roughly constant at around 10%.

Using drop-out rates, we can correct for the aforementioned bias and compute an ideal search time histogram, for the hypothetical case that participants never give up [25]. The result is shown as the green line in Fig. 4.1. Although longer games are more frequent under the ideal than under the observed distribution, the distributions still look similar qualitatively, with mode 4 and a power-law–like tail. The median search time is only 1 click higher (6 *vs.* 5 clicks), and mean search time rises by 3 clicks (8.9 *vs.* 5.8 clicks). We conclude that the observed human efficiency in Wikispeedia play is not explicable by a sampling bias alone.

Instead, we conjecture that, even without knowing the set of all existing links, the Wikipedia graph is efficiently navigable for humans because they have an intuition about what links to expect. We shall see that the probability of two articles linking to each other is higher the more related they are, which can lead to efficient navigation even in the absence of global knowledge. In particular, Liben-Nowell *et al.* [57] have shown analytically that

[2]Technically, $R_i$ is defined as $c_i \Big/ \left( \sum_{j \geq i} c_j + \sum_{j>i} s_j \right)$, where $c_i$ is the number of games canceled at position $i$ and $s_i$ the number of games successfully finished at position $i$. That is, attrition rate captures how often players gave up at the respective path position, out of the times they had the option to give up there.

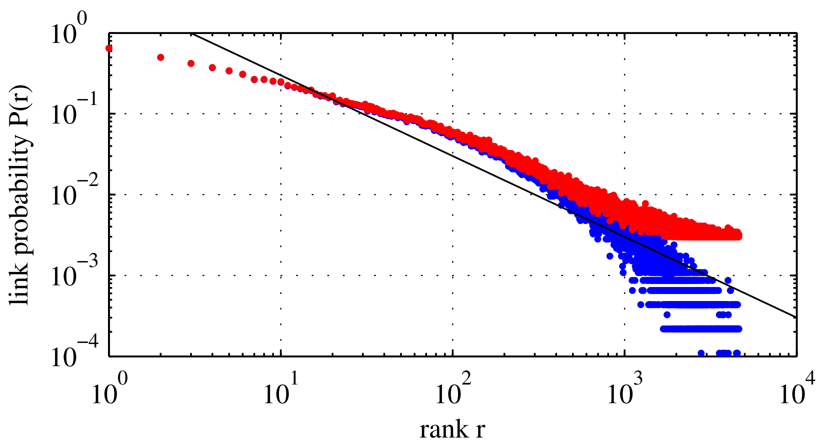


Figure 4.4: Link probability $P(r)$ as a function of rank $r$. Given $r$, consider all node pairs $(u,v)$ such that $v$ is the node that is $r$-th closest to $u$ among all nodes. Then $P(r)$ is defined as the fraction of these nodes for which $u$ links to $v$. **Blue:** $P(r)$. **Red:** $P(r)+\epsilon$, with $\epsilon = 0.005$. **Black:** ideal slope of $-1$ (not a fit; only for orientation).

short search times (technically defined as polylogarithmic in the number of nodes) can be expected under their model of 'rank-based friendship', *viz.*, if the probability of a node linking to its $r$-th closest fellow node decays as $1/r$. Intuitively, such a scenario is desirable because it constitutes an appropriate mix of many short- and a few long-range links. The latter are helpful for getting somewhat close to the target, while the former are necessary for fully reaching it.

We strive to investigate whether the Wikipedia graph satisfies rank-based friendship. Humans may tap into all their knowledge and reasoning skills during play, so it is hard to formalize their node distance measure. In the present analysis, we therefore coarsely approximate the human by a standard text-based distance measure and define the similarity of two articles as the cosine of their TF-IDF vectors [60] (and distance as one minus similarity). Fig. 4.4 plots the link probability $P(r)$ as a function of rank $r$. The black line was added to show an ideal slope of $-1$, as postulated by the rank-based friendship model. Note that, although $P(r)$ does not fully follow a power law, the overall slope of the curve comes close to $-1$, which leads us to conclude that Wikipedia is conducive to efficient navigation because its links represent an appropriate mix of long- and short-range connections across concept space.

Also note the red, upper curve in Fig. 4.4: after adding a small constant $\epsilon = 0.005$ to $P(r)$, the plot looks considerably more like the required power law. We take this as an indication that there is slight underlinking in the Wikipedia graph: if every node linked to even its furthest fellow nodes with a small background probability $\epsilon$, then Wikipedia could become even more easily navigable (at least under the TF-IDF distance measure).

## 4.3 Elements of human navigation

In the previous section we have argued that human search in the Wikipedia network is made possible by the statistical properties of its link structure. Next we turn our attention to a detailed analysis of how people actually exploit these properties.

### 4.3.1 Anatomy of typical paths

In our analysis we investigate how some key quantities of articles and clicks change as games progress from the start towards the target article. To facilitate the analysis, we restrict ourselves to all games whose start and target articles are optimally connected by exactly 3 clicks and consider only effective paths.

Fig. 4.5 contains a graphical summary of the findings we are about to discuss. Each subfigure tracks one quantity along game paths; each curve is computed from all games of the same effective path length, the leftmost curve representing games of length 8, the next one games of length 7, *etc.* (to avoid clutter, we consider only games of a maximum length of 8 clicks). The $x$-axes show the human-path distance, *i.e.*, the number of clicks to the target on the effective path (*i.e.*, paths may be thought of as running from left to right), while the $y$-axes represent the mean of the respective quantity over all games, alongside 95% confidence intervals. The bold gray curves plot the given quantity for the average optimal solution. To compute it, we found an optimal solution for every human game instance and averaged. We refer to the figure in row $r$ and column $c$ as plot $(r, c)$.

**Making progress is easiest far from and close to the target.** Plot $(1, 1)$ shows how the shortest-path length (SPL) to the target changes as a function of human-path distance. Necessarily, the shorter the game, the steeper the curve. Additionally, all curves share a

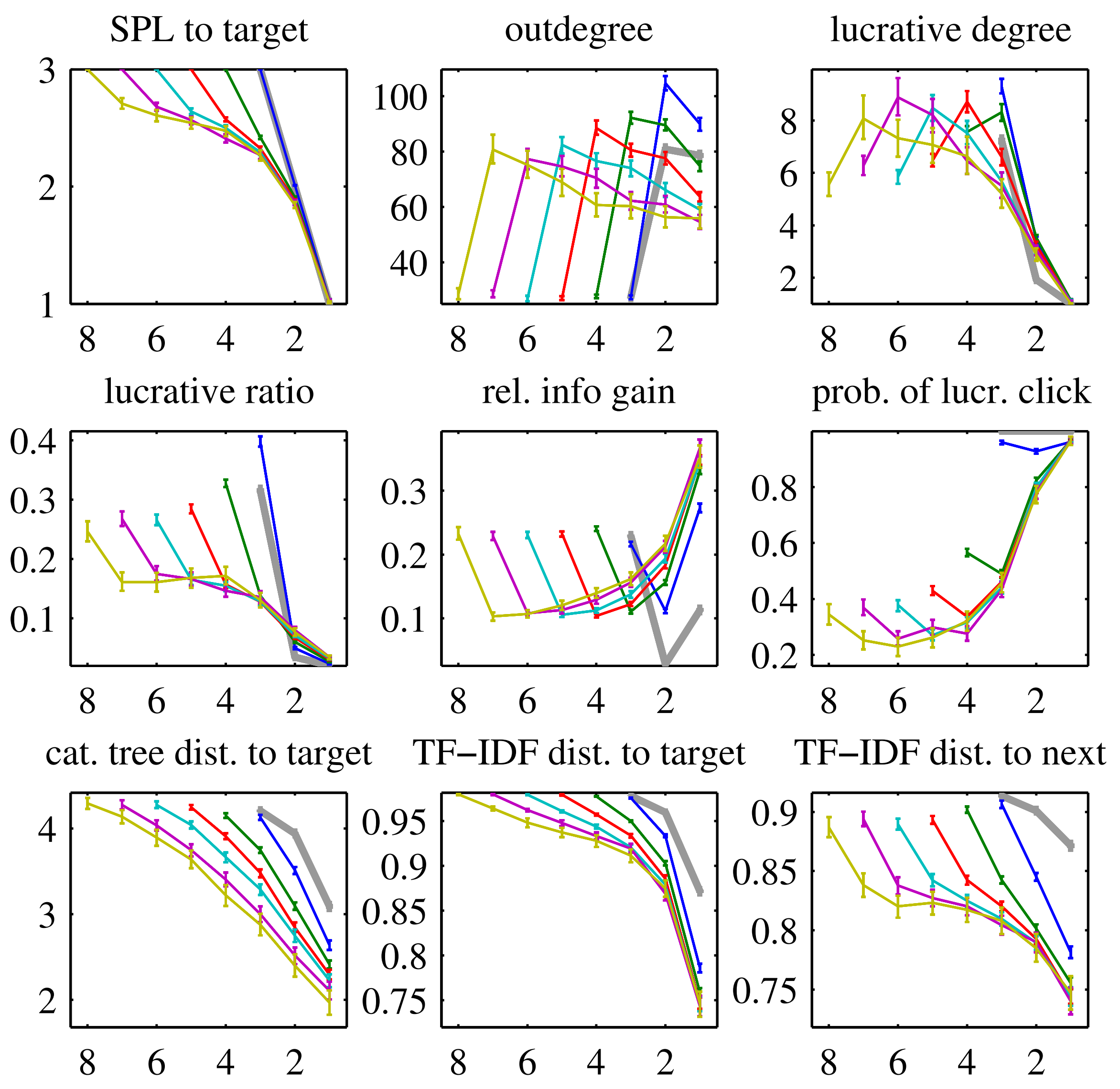


Figure 4.5: The evolution of article properties along Wikispeedia paths, for games of optimal length 3. Only games of between 3 and 8 clicks are shown. Each colored line represents games of the same length. The *x*-axis shows the distance-to-go to the target, the *y*-axis the average value of the respective property (with 95% confidence intervals). The bold gray curves are computed based on optimal solutions for the considered human paths.

typical anatomy: with the first click, the information seeker gets significantly closer to the target on average, then the curve flattens out and becomes steeper again towards the endgame. In short games, the players blasts straight through to the target, making progress with nearly every step, while in long games the player goes through a phase of inefficient circling around the target before finally gravitating towards it. Another perspective of the same phenomenon is afforded by plot $(2,3)$, which shows the fraction of times humans picked a lucrative link, *i.e.*, one that led them closer to the goal in terms of SPL. We observe a down–up pattern in the curves: information seekers are more likely to make progress with the first click than with the second. Later on, in the endgame, clicks become again ever more likely to be lucrative. In long games, the phases of progress in the opening and endgame are separated by a phase of stagnation where the probability of picking a good link stays roughly constant, a manifestation of the circling effect described above.[3]

**Hubs are crucial in the opening.** The initial progress with the first click is afforded by leaping to a 'hub' article, *i.e.*, a high-degree node that is easily reachable from all over the graph and that has connections to many regions of it. This makes sense intuitively, since a good hub gives the information seeker more options to continue the search, and is demonstrated by plots $(1,2)$, $(1,3)$, and $(2,1)$. While the start article has an average degree of only about 30 (plot $(1,2)$), the first click leads to an article with an average degree of between 80 and 100. After the sudden degree increase with the first click, the quantity decreases slowly as the target is approached.

Note that the shorter the game, the higher the degree of the hub (and of any given position, for that matter). This could mean (1) that better information seekers pick better hubs, or (2) that some missions are easier because the start articles have links to better hubs. While the availability of good hubs certainly helps, Fig. 4.6 demonstrates that the first alternative plays a role as well. We plot the ratio $\deg(u_2)/\deg(u_2^*)$ of the degree of the second article and that of the highest-degree neighbor of the start article, averaged over all games of the respective length. The quantity decreases with increasing game length,

[3]The fact that the probability is not 100% even when humans achieve the optimal path length (the blue curve) is due to the fact that players might have later undone clicks taken from articles along the effective path by means of the browser's back button, such that they may have taken suboptimal links while still achieving the optimal effective path length.

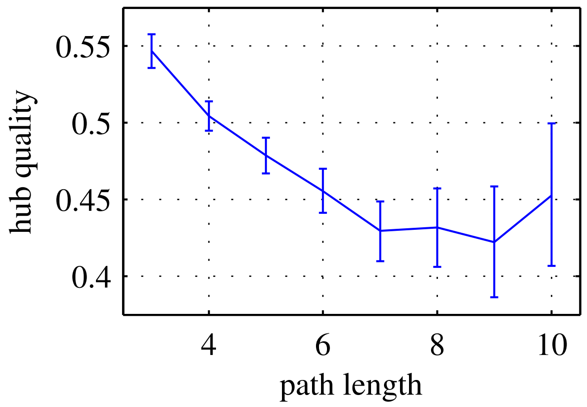


Figure 4.6: Hub quality as a function of search time (with 95% confidence intervals). Hub quality is defined as the degree of the second article, divided by the degree of the maximum-degree neighbor of the start article.

implying that better information seekers tend to start games with relatively higher-degree hubs.

Let the term ‘lucrative degree’ stand for the number of outgoing links that decrease the SPL to the target. Plot $(1,3)$ shows that, just like the plain degree, the lucrative degree, too, increases significantly with the first click—the hub article typically offers more lucrative options than the start article. Also, the mean lucrative degree then decreases as the games continue (necessarily, since there are more articles far from the target than close to it). We do not see a correlation between the hub’s lucrative degree and game length. However, the start article itself has higher lucrative degree for very short games than for longer ones, an indicator that some games are inherently easier than others, even if the optimal number of clicks is held fixed. This certainly is a factor in the aforementioned negative correlation between search time and hub degree: if there are many good hubs it is easier to find one of them.

An interesting additional insight is afforded by looking at how the average of the ratio of lucrative degree and degree changes during games (plot $(2,1)$). The resulting quantity, which we call ‘lucrative ratio’, corresponds to the probability of getting closer to the target when randomly choosing an outgoing link. While both degree and lucrative degree achieve their maximum with the second article, their ratio drops drastically between the first and second articles. From this we conclude that the second article is a true hub, in that it does not only have many outlinks leading closer to the target, but has even more that lead further away from it, *i.e.*, that it has connections into many different regions of the graph.

**Conceptual distance to the target decreases steadily.** Plots $(3,1)$ and $(3,2)$ show that articles get ever more related textually to the target as the latter is approached (in other words, textual distance decreases). We verify this using two distinct measures of conceptual relatedness, (1) the cosine of the TF-IDF vectors of the two respective articles, as in Section 4.2, and (2) the number of edges that have to be traversed in the category tree that comes with our Wikipedia version, in order to reach one article from the other ('category tree distance'). The fact that the conceptual distance to the target decreases strictly along paths corroborates our conjecture from Section 4.2 that humans approximately perform a decentralized search using a distance measure between concepts. Also, note that the very intuition that the distance between concepts along the path and targets decreases was the original *raison d'être* of the game of Wikispeedia [109].

**Big leaps first, followed by smaller steps.** While plots $(3,1)$ and $(3,2)$ track the textual distance between the current article and the target, plot $(3,3)$ does so for the distance between the current and the next articles. This 'textual step size' is monotonically decreasing: first, information seekers make big leaps, with adjacent articles being rather unrelated (*e.g.*, when jumping to the hub); then, as they home in on the target, they straddle ever smaller 'gaps'. This progression is possible because Wikipedia's link structure trades off long- versus short-range connections in a favorable manner, as laid out in our discussion of rank-based friendship in Section 4.2. We also see the aforementioned circling effect for long games again: between the initial getting-away and the final homing-in, both the textual distance to the target and the textual step size stagnate, as the player stumbles around on the graph.

**Clicks are most predictable far from and close to the target.** Finally, consider plot $(2,2)$, which attempts to capture the agreement between different humans. Consider a target article $t$. For each article $u$, we define a click probability distribution over $u$'s outlinks, which counts for each outlink how often it was taken when humans were searching for the target $t$ (with add-0.1 smoothing, to mitigate the effect of zero counts). The entropy of this distribution provides us with a measure of how predictable human clicks are, lower entropy meaning higher predictability. We let the term 'information gain' refer to the difference between the prior entropy of the uniform click distribution before observing any clicks and the posterior entropy given all game data. It measures how much more predictable clicks at a given article $u$ are after seeing the game data than before. 'Relative information gain' is the ratio of information gain and prior entropy, or in other words, the percentage-wise decrease in uncertainty afforded by observing the game data. This quantity exposes a characteristic pattern, as shown in plot $(2,2)$. The relative information gain at the start article is typically around 23% on average and much lower (around 10%) for the following article. The leap to the hub is much more predictable than the ways in which people continue from there. (This is compounded by the fact that, given a start article, not all humans choose the same hub, such that for each hub we have fewer samples than for the start article and the respective click distribution stays more uniform, resulting in higher posterior entropy and thus lower information gain.) As information seekers approach the target, their behavior becomes again more coherent and predictable, with information gain increasing.

**Comparison of human with shortest paths.** To conclude our discussion of typical human search paths, we compare them to the optimal solutions found by a shortest-path algorithm (the bold gray curve in each plot). Most of the curves are qualitatively similar to those for human paths, which follows from the structural constraints imposed by the link graph. However, there are quantitative differences with respect to all quantities we investigate. For instance, for shortest paths, too, the average degree goes up with the first click, but this is purely statistically so because the shortest-path finder is more likely to pick high–betweenness-centrality nodes, which in turn tend to have high degree; note that nonetheless the hub has about 20 fewer outlinks than for optimal humans. The lucrative degree of the hub is about 3.5 for optimal solutions found by humans, while it is and only 2 for solutions found by the shortest-path finder.

The relative information gain is nearly zero for the second article, much smaller than the 10% typical for humans. The reason is that shortest paths are often entirely different from human paths, such that the second article itself is often one that humans never picked. Since the information gain is computed solely based on human paths, the entropy at the second article stays very uniform (*i.e.*, information gain close to zero).

The curves for TF-IDF similarity to the target and to the next article are qualitatively similar to those for human paths, in that the distance values decrease as games progress. This is due to the fact that closeness in the Wikipedia graph is correlated with textual similarity (Fig. 4.4). Therefore, as the graph distance to the target decreases, so does the textual distance (plots $(3,1)$ and $(3,2)$). Note, however, that the decrease is much more pronounced for human paths: humans explicitly navigate according to the content of articles, while the shortest-path finder does so only because it is implicitly constrained by the statistical properties of the hyperlink graph.

### 4.3.2 Trade-off between similarity and degree

Given the findings of the previous section, degree and similarity seem to be the most important factors in human navigation of Wikipedia. We hypothesize that humans navigate more strongly according to degree in the early game phase, when finding a good hub is important, and more strongly according to textual similarity later on, in the homing-in phase. The goal of this section is to test this hypothesis.

We conduct the following experiment to gauge the trade-off between similarity and degree. Consider only the games with an optimal solution of 3 clicks. Then divide the set of all human trajectories into subsets according to the number of clicks taken by the player (the maximum length we consider is 8).[4] Each of these subsets is divided into balanced training (70%) and test (30%) sets. For each training set and each path position in the training set, we train a logistic regression classifier, using two features (and a constant bias term), representing degree and similarity to the target, respectively. The positive examples consist of all human clicks contained in the respective training set. The negative examples have to be contrived (since we have no ground truth of clicks a human will never make). We do so by randomly (with replacement) sampling clicks that were never observed, until there are as many negative as there are positive examples. Once the classifiers for all combinations of path length and path position have been trained, we inspect the resulting feature weights to infer how important each feature is in humans' click choice at each position.

[4]We use complete paths including back-clicks. However, while back-clicks themselves are neglected, we do consider the forward clicks that the player undoes later on.

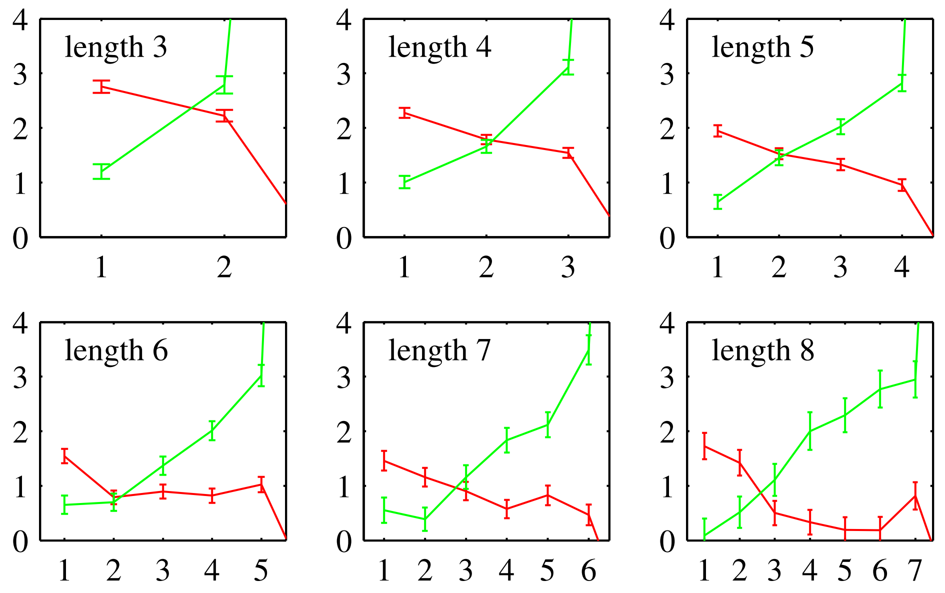


Figure 4.7: Logistic regression weights for classifying human *vs.* non-human clicks (with standard errors). **Green:** textual similarity. **Red:** degree. There is one plot per human path length; the *x*-axes show path positions, the *y*-axes weights.

Before presenting the results, we add some notes about the two features. When regression is used for the purpose of feature analysis, it is important to have uncorrelated features. The natural choice for similarity would be the TF-IDF cosine that we have also used in previous sections of this paper. However, this similarity measure is highly correlated with degree: the higher a node's degree, the higher its average TF-IDF similarity with all other articles. This happens because high-degree articles are typically long, and long articles are more likely to have some text overlap with the target article. (The effect is noticeable even in the face of the length-normalization implicit in cosine similarity.) On the contrary, no such correlation with degree is exhibited by the category tree distance. We therefore adopt the latter to quantify similarity in our regression analysis. To be able to compare the weights for features that can take on very different values, we also have to normalize. We do so by adopting a rank-based approach. Consider an article $u$ and a given feature. The neighbors of $u$ get values from the interval $[0,1]$, such that the highest-ranking neighbor, according to the feature, gets value 1, and the lowest-ranking neighbor value 0.

Fig. 4.7 plots the resulting weights for the two features. There is one plot for each game length between 3 and 8. The *x*-axes show path positions, and each data point represents one feature weight. The red curves are the degree and the green ones the similarity weights. The weight of the bias term was omitted from the plots, since it is not informative. Note that, for visibility's sake, we do not show the weights for the last click. There, similarity becomes a nearly perfect indicator for the target article, since the target has maximum similarity with itself, so the similarity feature gets a very large weight, and the interesting part of the plots would get squished and hard to read.

Interpreting the plots, our expectation is confirmed. Both features obtain positive weights everywhere, which means that both high degree and high similarity with the target are characteristics of the click choices made by humans. More interesting, as hypothesized, degree dominates in the beginning of games, but as games progress, similarity becomes ever more important, superseding degree starting with the second or third click. Furthermore, similarity starts dominating earlier in more efficient games.

We emphasize that the purpose of this experiment is an analysis of the fitted feature weights, not maximizing the accuracy of the classifiers. Still, to justify our conclusions, we need to show that the classifiers perform better than chance (50%) on a statistically significant level. Evaluating the classifiers on the held-out test set, we find that this is the case. Accuracy is similar for all game lengths. It drops from around 90% for the first path position to about 65% for the second and then stays in the regime of between 55% and 65%. When maximum accuracy is the goal, more powerful features, such as TF-IDF cosine, perform better, but as mentioned earlier, feature correlation does not permit us to use this feature in our analysis.

## 4.3.3 Endgame strategies

The main finding of the previous section is that in the opening of games it is common to navigate through hubs. Next we take a closer look at the strategies players adopt in endgames, in order to home in on the target.

In the present analysis, we define an *endgame* as the last 3 articles (*i.e.*, 2 clicks) of a path. To make sure the endgames we analyze do not contain artifacts from the game

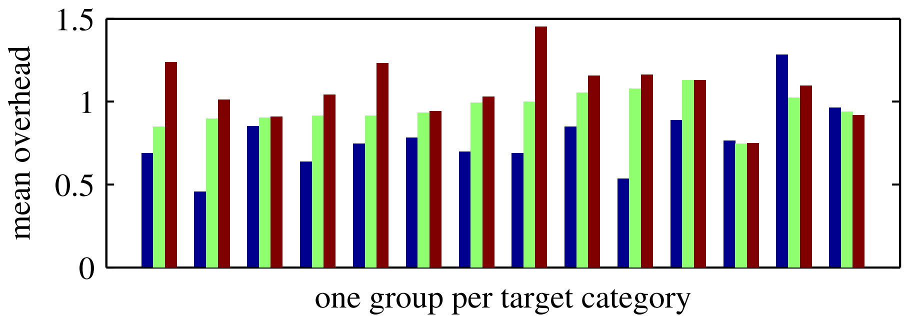


Figure 4.8: Overhead with respect to optimal solutions, for **(red)** single-category and **(blue)** most popular multi-category strategies, with one group per target category. The **(green)** middle bars show means over all games of the respective target category. From left to right: PEOPLE, MUSIC, IT, LANGUAGE AND LITERATURE, HISTORY, SCIENCE, RELIGION, DESIGN AND TECHNOLOGY, CITIZENSHIP, ART, BUSINESS STUDIES, MATHEMATICS, EVERYDAY LIFE, GEOGRAPHY.

openings, we consider only games of a full length of at least 5 articles (*i.e.*, 4 clicks). We also neglect all games above the length threshold of 20 articles. The *endgame strategy* corresponding to an endgame $\langle u_{n-2}, u_{n-1}, u_n \rangle$ is defined as $\langle C(u_{n-2}), C(u_{n-1}), C(u_n) \rangle$, where $C(u)$ is $u$'s top-level category in the hierarchy that comes with our Wikipedia version. For instance, the full category of DIK-DIK is SCIENCE/BIOLOGY/MAMMALS, and $C(\text{DIK-DIK}) = \text{SCIENCE}$. All 14 values $C$ can take on are listed in the caption of Fig. 4.8.

We divide the set of all Wikispeedia games into subsets according to target categories, such that all games with target articles from the same category are placed in the same subset. For each target category, we observe between 29 and 104 distinct strategies, out of the possible $14^2 = 196$. For all target categories, the distribution over strategies is highly non-uniform, with most games following one of only a few top strategies. As a consequence, for each target category, the top 10 strategies typically cover between 60% and 90% of all games. Furthermore, the distributions over articles within each category are also non-uniform; *e.g.*, 14% of all instances of GEOGRAPHY are UNITED STATES and 6.1% UNITED KINGDOM.

In 12 out of the 14 target categories, the most popular strategy is the one that consists of the target category only, which we call the 'simple' strategy: people tend to approach the target through articles from the same category as the target. In the remaining two categories, the simple strategy has very high rank, too: it is second most frequent when the target is from DESIGN AND TECHNOLOGY, and fourth most frequent when it is from PEOPLE. In the former case, the more frequent strategy is $\langle$GEOGRAPHY, GEOGRAPHY, DESIGN AND TECHNOLOGY$\rangle$; in the latter case, the three more frequent strategies are $\langle$GEOGRAPHY, GEOGRAPHY, PEOPLE$\rangle$, $\langle$GEOGRAPHY, CITIZENSHIP, PEOPLE$\rangle$, and $\langle$GEOGRAPHY, HISTORY, PEOPLE$\rangle$.

In these examples, GEOGRAPHY seems to play a prominent role. And indeed this is a general property of human paths. To demonstrate this, we count, for each category $c$, how often articles from it appear in endgames in which the target is *not* also of category $c$. We find that GEOGRAPHY accounts for 20% of articles in endgames of which GEOGRAPHY is not the target. The next most common categories according to this metric are SCIENCE (7.5%) and HISTORY (5.1%). One might argue that certain categories are *a priori* more likely, since they contain more articles. We can correct for this bias by considering the ratio of the above-introduced frequency and the *a-priori* category frequency, *i.e.*, the number of articles in the category, divided by the overall number of articles. In the resulting ranking, GEOGRAPHY is still top, now followed by CITIZENSHIP (mostly about politics and culture) and RELIGION. This finding might imply that humans organize their knowledge strongly according to geographical lines, and that they often associate concepts with their countries of origin.

Previously, we saw that simple single-category endgames are typically most popular with players. Next, we investigate how efficient they are compared to other, more complex strategies. Let $l$ be the length (number of clicks) of a human game, and $l^*$ the number of clicks in an optimal solution. We define the *overhead* of a game as $(l - l^*)/l^*$, *i.e.*, the percentage of the optimal solution length that the information seeker needed extra. For each endgame strategy, we compute the mean overhead over all games of that strategy. As a baseline, we consider the mean overhead across all games of the given target category. We find that the overhead of the simple strategy is on average (over the 14 target categories) 12% higher than that of the mean game; *i.e.*, games using the simple strategy are typically worse than average. Now consider, instead of the simple strategy, the most frequent multi-category strategy. Averaged across all target categories, its overhead is 18% smaller than that of the mean game; *i.e.*, games using the most frequent multi-category strategy are typically considerably better than average. Fig. 4.8 shows that this is not only true on average but for nearly every target category taken by itself. One of only three categories for which the simple strategy is, on the contrary, most efficient is GEOGRAPHY (the rightmost group in the bar chart). This is in tune with our previous findings: since GEOGRAPHY plays a prominent role even when it is not the target, it makes sense that it allows for efficient paths when it is.

Our interpretation of these findings is that information seekers face a trade-off between efficiency and simplicity. Whilst reaching the target through very related articles from the same category is conceptually simple, the steps taken this way can be small and prolong the game. It often pays off to think out of the box—or to think in geographic terms.

## 4.4 Conclusions

Finding paths connecting different concepts—like linking causes to effects—is a task humans have been performing for millennia. In this chapter, we have analyzed human behavior on this task through the lens of navigation traces collected through the human-computation game Wikispeedia. We study thousands of goal-directed human search paths and identify aggregate strategies people use when navigating information spaces.

As information spaces become more complex, it is increasingly important to understand how users navigate them and to assist users in locating the desired information. In this light, the present chapter provides a basis for the rest of this thesis, by providing the fundamental insights on which we build when designing methods for predicting a user's navigation target (Chapter 5), for navigating automatically (Chapter 6), and for improving the hyperlink structure of websites (Chapter 7).

# Chapter 5

# Predicting targets of human navigation traces

## 5.1 Introduction

In the previous chapter, we have conducted an in-depth analysis of how people navigate Wikipedia towards a given target article. Our next goal is to apply the lessons learned in designing a learning algorithm for predicting an information seeker's target, given only a prefix of a few clicks. Our method explicitly takes the characteristic features of human search into account and is trained on real human trajectories.

A method such as the one we propose here would be a useful component in intelligent browsing interfaces that would allow users to be more effective and efficient at finding the information they are seeking. For instance, maintaining a high-quality estimate of the user's navigation target would allow us to propose useful shortcuts to the user, highlight certain information, or suggest other relevant content.

**Chapter outline.** We proceed as follows in this chapter. Sec. 5.2 introduces the Markov model we use for probabilistically reasoning about human navigation traces. We then propose two models for estimating the click probabilities used in the Markov model: a binomial logistic regression model (Sec. 5.3) and a multinomial learning-to-rank model (Sec. 5.4). In Sec. 5.5 we introduce the features used for estimating the click probabilities, and we finally perform an evaluation by comparing our various models in Sec. 5.6.

## 5.2 Human Markov model

We cast our task as a ranking problem: given the observed path prefix $q$, rank all articles $t$ according to how plausible they are as targets of the current search. At the heart of our approach is a Markov model of human search, the parameters $\Theta$ of which are learned. To make the prediction, we order candidate targets $t$ according to a ranking function $g(t|q;\Theta)$, defined as the likelihood of $t$ given the prefix $q$, *i.e.*, as $P(q|t;\Theta)$.

Let $q = \langle u_1, ..., u_k \rangle$ be a prefix of $k-1$ clicks. Given target $t$ and model parameters $\Theta$, the probability of seeing $q$ is obtained by multiplying the local click probabilities, and we aim to find the most likely target, *i.e.*,

$$\operatorname*{argmax}_t P(q|t;\Theta) = \operatorname*{argmax}_t P(u_1) \prod_{i=1}^{k-1} P(u_{i+1}|u_i,t;\Theta), \tag{5.1}$$

where $P(u_1) = 1/N$ is constant, with $N$ the number of articles (since start articles are picked randomly). Note that we will work with the *prefix log-likelihood* $L(t|q;\Theta) := \log P(q|t;\Theta)$ instead.

In our analysis of humans, we saw that people trade off features differently at different steps. We mimic this by learning a separate set of weights for each step, such that $\Theta = (\boldsymbol{\theta}_1, ..., \boldsymbol{\theta}_{k-1})$ is in fact a collection of weight vectors, with $\boldsymbol{\theta}_i$ being the weights for step $i$.

We next propose and test two alternative models of click probability $P(u_{i+1}|u_i,t;\Theta)$, each with its own model-fitting algorithm.

## 5.3 Binomial logistic model

The first, simpler model is similar in spirit to the regression of Section 4.3.2 (but using stronger features), where we fit a model to predict whether humans would pick a given link. The model specifies, for any given click triple $(u_i, u_{i+1}, t)$ separately, the probability that a human would choose it.[1] Formally, we define the *binomial logistic model* as

$$P(u_{i+1}|u_i,t;\Theta) = \frac{\sigma(\boldsymbol{\theta}_i^\top \mathbf{f}(u_i,u_{i+1},t))}{\sum_{v\in\Gamma(u_i)} \sigma(\boldsymbol{\theta}_i^\top \mathbf{f}(u_i,v,t))}, \tag{5.2}$$

where $\sigma(x) = (1+e^{-x})^{-1}$ is the logistic function; $\mathbf{f}(u_i,u_{i+1},t)$ is a feature vector for the click from $u_i$ to $u_{i+1}$ given target $t$; and $\Gamma(u_i)$ the set of $u_i$'s neighbors. The model parameters $\Theta$ are fitted as in standard logistic regression using gradient descent.

## 5.4 Learning-to-rank model

Since the task is to rank target candidates, we also explore a different setup in which we fit $\Theta$ explicitly to optimize a ranking objective that we refer to as *cumulative reciprocal rank*. This metric is defined as $\ell(r) := \sum_{j=1}^{r} 1/j$, where $r$ is the rank of the true target [112]. Minimizing this objective implies ranking the true target as high as possible; additionally, cumulative reciprocal rank has the desirable property of putting more emphasis on the top of the ranking (*e.g.*, $\ell(20)-\ell(1) \gg \ell(120)-\ell(101)$), and is therefore a sensible choice for evaluating rankings.

Notice that, unlike assumed by the simplistic binomial logistic model, humans really face a multinomial choice at each step. Therefore, we now represent click probabilities in a *multinomial logistic model:*

$$P(u_{i+1}|u_i,t;\Theta) = \frac{\exp(\boldsymbol{\theta}_i^\top \mathbf{f}(u_i,u_{i+1},t))}{\sum_{v\in\Gamma(u_i)} \exp(\boldsymbol{\theta}_i^\top \mathbf{f}(u_i,v,t))}. \tag{5.3}$$

Another advantage of this framework is that we may use other features in addition to the likelihood. Some important factors depend on the entire prefix and cannot be encoded naturally into the Markov model (*e.g.*, How often did the player not take a direct link to the target although this was possible?), which considers only local clicks. Thus, in the learning-to-rank setup, our final ranking function $g$ consists of a linear combination of prefix log-likelihood and those additional *prefix-global features.* As our prediction, we

[1]One might be tempted to phrase the problem as multinomial logistic regression instead, but this is not possible, since the degree of $u_i$ and hence the number of classes is variable.

select the target $t$ that maximizes $g$, *i.e.*,

$$\operatorname*{argmax}_t g(t|q;\Theta,\boldsymbol{\beta}) = \operatorname*{argmax}_t \beta_1 F_1(q,t) + \ldots + \beta_m F_m(q,t), \tag{5.4}$$

where $F_1 = L(t|q;\Theta)$, and $F_2,\ldots,F_m$ are the prefix-global features, the details of which are provided below (note that they do not depend on $\Theta$).

We fit $\Theta$ and $\boldsymbol{\beta}$ using an approach inspired by a method proposed recently by Weston *et al.* [112]. The algorithm minimizes cumulative reciprocal rank via stochastic gradient descent, using a novel sampling trick to speed up learning. In our case, this is necessary since we would otherwise have to iterate over all target candidates (*i.e.*, all articles) for every training example. We refer the reader to Weston *et al.*'s paper regarding the details of the framework and restrict ourselves to highlighting how we adapt their algorithm:

The learning algorithm requires computing the derivative of $g$ with respect to $\Theta$ and $\boldsymbol{\beta}$. We have

$$\frac{\partial}{\partial \beta_j} g(t|q;\Theta,\boldsymbol{\beta}) = F_j(q,t) \tag{5.5}$$

and

$$\begin{aligned}
\frac{\partial}{\partial \boldsymbol{\theta}_i} g(t|q;\Theta,\boldsymbol{\beta}) &= \beta_1 \frac{\partial}{\partial \boldsymbol{\theta}_i} L(t|q;\Theta) && (5.6)\\
&= \beta_1 \left[ \mathbf{f}(u_i,u_{i+1},t) - \sum_{v\in\Gamma(u_i)} P(v|u_i,t;\Theta)\, \mathbf{f}(u_i,v,t) \right]. && (5.7)
\end{aligned}$$

That is, the likelihood gradient with respect to the weights for prefix position $i$ is equal to the difference of the feature vector of click $i$ and the expected feature vector under the current weights $\Theta$.

## 5.5 Features for learning

So far, we have only described the abstract framework of our algorithms but have not yet discussed the concrete features we use. As mentioned, our choice of features is inspired by the results of our study of human behavior: if we design features that capture the characteristics of human paths, the algorithms can learn weights to predict targets under which the seen prefix resembles human behavior most. Specifically, these are the entries of the likelihood feature vector $\mathbf{f}(u_i, u_{i+1}, t)$ (recall that we learn a separate weight vector for each $i$):

1. TF-IDF$(u_{i+1}, t)$: as we have seen, articles become ever more related to the target as human games proceed (Fig. 4.5 $(3,2)$);
2. TF-IDF$(u_i, u_{i+1})$: 'textual step size' is large at first and becomes ever smaller (Fig. 4.5 $(3,3)$);
3. deg$(u_{i+1})$: humans commonly navigate through hubs in the beginning, so we expect the weight for this feature to be large for early and small for later steps $i$;
4. deg$(u_i) \times$ deg$(u_{i+1})$: if $u_i$ already is a hub, there is no more need for the player to search for another one, which can be captured by this interaction feature;
5. TF-IDF$(u_i, t) \times$ deg$(u_{i+1})$: if the player is already close to $t$ textually, finding a hub is less important;
6. TF-IDF$(u_i, t) \times$ TF-IDF$(u_i, u_{i+1})$: close to $t$, the step size is typically also smaller;
7. the indicator SPL$(u_{i+1}, t) >$ SPL$(u_i, t)$: if a click increases the shortest-path length to $t$, then it is less likely to be chosen by a player;
8. $|$TF-IDF$(u_i, t) -$ TF-IDF$(u_{i+1}, t)|_+$: a click is also less likely if it decreases the textual similarity to $t$. (Here $|x|_+ := \max\{x, 0\}$ is the hinge-loss function.)

In the learning-to-rank approach, we additionally have the following prefix-global features (Eq. 5.4):

1. Number of times the player could have taken a direct link to $t$ yet did not;
2. Number of clicks through which the shortest-path length to $t$ increased;
3. Sum (over all clicks) of decreases in textual similarity to $t$.

The last two prefix-global features are similar to likelihood features 7 and 8, but here they can modify the ranking function explicitly rather than merely via the likelihood term. We expected the first prefix-global feature to receive a large negative weight, guided by the intuition that humans would always go directly to the target as soon as this is possible.

However, a weight of nearly zero is learned for this feature, which indicates that information seekers often miss the best links because they do not expect them and hence do not notice them in the often long article text. This emphasizes the usefulness of the task of target prediction: if the algorithm can infer the intended target, it could highlight the most useful links so they become more salient to the user.

## 5.6 Evaluation

For training our models, we use prefixes of 3 clicks, taken from human games of at least 4 clicks.[2] The weights converge quickly for the multinomial ranking model, after seeing a few thousand examples, which takes only some minutes.

We also compare our algorithms to a baseline that simply predicts as the target the article with the largest TF-IDF similarity to the last article of the prefix.

We use a test set not seen during training to evaluate the algorithms on two tasks: (1) given a prefix $q = \langle u_1, ..., u_k \rangle$ and a choice of two targets, pick the true target $t$; the false target is picked randomly from the set of articles that ever occurred as targets and have the same shortest-path length from $u_k$ as $t$; (2) given $q$, rank the set of *all* articles such that the true target is ranked high. The second task is much harder but also more useful than the first; *e.g.*, if the method is to be implemented for an intelligent browsing tool, reasonable targets must be picked from all candidates, not just from a set of two.

In the first task, we use accuracy as a metric; in the second, we measure ranking loss according to cumulative reciprocal rank, the objective we also use for training. While this captures ranking quality objectively, it might be overly strict; *e.g.*, if $t =$ WINE, then predicting BEER is much better than, say, GASOLINE. We account for this by measuring 'sibling precision@$m$', which is the same as precision@$m$, with the difference that not only $t$ but all articles from the same category as $t$ are counted as relevant (we use the leaves of the hierarchy of our Wikipedia version as categories).

We vary two parameters of the test prefixes: $k$, the number of articles in the prefix; and $n$, the length of the entire human path. The results are summarized in Fig. 5.1 and 5.2. In all plots, the bold solid line represents the multinomial ranking (MR) model, the thin

[2] In this chapter we use a dataset of around 30,000 traces collected from 2008 to 2012.

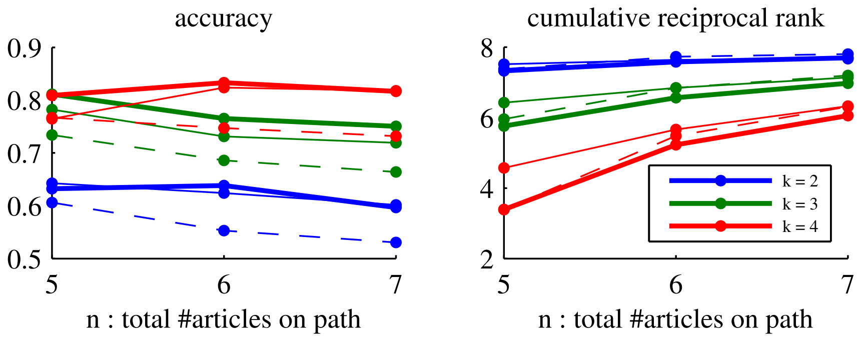


Figure 5.1: Performance of our target prediction algorithms, for varying prefix lengths *k* (indicated by color). **Left:** accuracy (higher is better). **Right:** cumulative reciprocal rank (lower is better). **Bold solid:** multinomial ranking model. **Thin solid:** binomial logistic regression. **Dashed:** TF-IDF baseline.

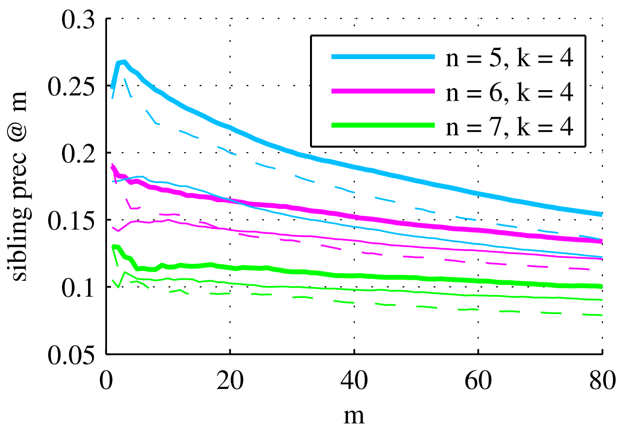


Figure 5.2: Sibling precision of our target prediction algorithms. **Bold solid:** multinomial ranking model. **Thin solid:** binomial logistic regression. **Dashed:** TF-IDF baseline.

solid line the binomial logistic regression (BLR) model, and the dashed line the TF-IDF baseline. First note that MR is at least as good as, and often better than, both BLR and TF-IDF according to every metric. Now consider Fig. 5.1. As expected, our methods work better when prefixes are longer (*cf.* the order of the bold curves) and when full paths are shorter (*cf.* the slopes of the curves). Notably, on the task of picking the correct one of two targets, MR achieves an accuracy of 80% when 3 clicks are seen, regardless of whether the entire game is 4, 5, or 6 clicks long. Interestingly, while BLR has higher accuracy on the binary task, the simple TF-IDF baseline achieves better ranking performance. We take this as an indicator that MR combines the better properties of both.

Finally, consider Fig. 5.2, which shows sibling precision@$m$. For the sake of brevity, we display only the case $k = 4$, but in relative terms the results are the same for all prefix lengths. The precision@30 of MR is 20% for $n = 5$, which means that 6 of the top 30 targets are of the same category as the true target, when we see 3 clicks and the full game has 1 more click. Even when there are 2 (3) more clicks, we still see 5 (3) top-ranked articles that are very close to the true target (for comparison, in a random ranking, precision is only 1% on average). This property of the ranking algorithm is desirable, since in a real-world application making a close enough guess might often be nearly as good as predicting the exact target.

## 5.7 Conclusions

There are many potential use cases of target-prediction methods such as the one proposed here. For instance, an intelligent browsing interface could use the algorithm for tracking the user's goal and adapt accordingly, *e.g.*, by suggesting useful shortcuts, thus making human search more efficient.

Here we evaluate our method only in the context of Wikipedia, but we believe it is general enough to extend to other search scenarios, if appropriate features are used.

Not all instances of human navigation are directed toward a specific target; oftentimes users browse networks in an undirected fashion, with the goal of exploring content. Therefore, a promising direction of future research would be to devise methods for determining if a user has a target in mind in the first place. Such a method would serve as a filter before applying the algorithm proposed in this chapter.

Furthermore, when humans search for information, they mostly use a search engine in combination with click-based navigation. Since users formulate their search queries in order to explicitly express their information needs, we expect it to be a powerful extension to our method to also incorporate features derived from search queries.

# Chapter 6

# Automatic versus human navigation

## 6.1 Introduction

Human navigation as analyzed in Chapter 4 can be phrased as search in a graph, where the user gradually moves across edges from a start to a target node. However, identifying short paths by means of, say, Dijkstra's algorithm is impossible, since this would require knowledge of the full graph, whereas typically users have only local access to the network, seeing only the current and the previously visited nodes, as well as their direct neighbors. Other than that, neither the—typically large—set of all existing nodes (often with complex content) nor the links between them are known ahead of time. Therefore, it can be easy to get lost. Nevertheless, people are rather good at finding short paths between pairs of nodes (Sec. 4.2).

One potential explanation is that people are good at 'connecting the dots' because they have common sense and a large body of background knowledge for reasoning about the world, such that their search can be guided by intuitive expectations about the unknown link structure of the network. For instance, one might think that, in order to find a path from the Wikipedia article about MOZART to that about TERMINATOR, one must know that Mozart is from the same country as the person enacting the Terminator, such that trying to find a path like ⟨MOZART, AUSTRIA, SCHWARZENEGGER, TERMINATOR⟩ would make sense.

But how hard is navigating content-rich information networks really? Does the task require high-level reasoning skills and common sense—which computers do not have yet, and which is a very ambitious goal? Or can it be formalized in a straightforward way such that even a simple computer program can solve it? In this chapter we attempt to shed light on this question by designing a number of automatic agents without human-like knowledge and reasoning skills, which use only simple numerical features. We evaluate these agents in the controlled example task of Wikipedia navigation. We choose Wikipedia as the validation domain for two reasons. First, it contains rich knowledge of the kind people are good at reasoning about. Second, since Wikipedia navigation is the task that also underlies the human-computation game Wikispeedia, we may readily compare the performance of our agents to that of humans.[1]

When designing the search algorithm and the features available to the agents, we have striven to allow for as fair a comparison between humans and automatic agents as possible. Analyzing the results, we observe that even simple algorithms can find shorter paths on average than humans. We shall see that, in our test set, humans need about twice the optimal number of clicks on average, whereas our best automatic agent achieves a factor as low as 1.5. Also, the frequency with which human searches are over twice as long as shortest paths is 4 times as high as for automatic searches. Hence we conclude that, maybe surprisingly, very basic numerical features suffice for navigating a complex network such as Wikipedia.

On a more faceted level, human and automatic search are qualitatively different. Our evaluation lends support to the conclusion that, while the agents are more efficient on average, humans are less likely to get totally lost during search. Humans typically form robust high-level plans with backup options, something our automatic agents cannot do. Instead, the agents compensate for this lack of smartness with increased thoroughness: since they cannot know what to expect, they always have to inspect all options available, thereby missing out on fewer immediate opportunities than humans, who, focused on executing a premeditated plan, may overlook shortcuts.

While we consider these findings interesting in their own right, our algorithms may also prove useful in the context of social and information networks from a practical perspective.

[1] In this chapter we use a dataset of around 30,000 traces collected from 2008 to 2012.

For instance, 'social browsing' [51] is an emerging paradigm. Also, it is estimated that 99.8% of the Web's content, the so-called 'deep Web', is hidden 'behind the query forms of searchable databases' [35] and therefore cannot be systematically indexed by general search engines, making it necessary to navigate by clicking from page to page. Even on the 'surface Web', there is little use in today's search engines whenever we cannot formulate a concise query describing what we are looking for; and even if a decent query was entered, the results often only serve as the starting point for further, click-based navigation. Furthermore, sometimes the entire path that connects two pages rather than a single result page matters; *e.g.*, the answer to a question might be spread over a series of Wikipedia articles. Finally, some information resources, such as peer-to-peer networks, are deliberately designed in a decentralized manner, such that global search is impossible. In all these scenarios, navigation-based search algorithms may complement the use of query-based search engines. By analyzing the learned importance weights attributed by our algorithms to different features, we can quantify what information matters most for solving search tasks, which may in turn help us design tools for the support of human browsing.

**Chapter outline.** We proceed as follows. First, we introduce and motivate the abstract search algorithm (Sec. 6.2), before describing concrete implementations of it, which differ only in the way they score the neighbors for choosing the next click (Sec. 6.3). Thereafter, we compare our algorithms among each other and to humans, and analyze the relative importance of features (Sec. 6.4). We conclude by discussing our findings and their implications (Sec. 6.5).

## 6.2 The abstract search algorithm

The task we investigate is how to navigate from a *start* to a *target* node by traversing edges of a given graph. Since our goal is to design algorithms whose performance we can compare to that of humans in a fair manner, it is important that we subject the automatic agents to the same restrictions a human navigator faces.

First, it is a requirement that in deciding where to go next, our agents use only local node features independent of global information about the network. Only the nodes visited so far, their immediate neighbors, and the target may play a role in picking the next node.

**Algorithm 1** Basic navigation algorithm

```
Input: start node $s$; target node $t$; evaluation function $V$
Output: sequence of nodes visited
 1: push $s$ onto initially empty stack $\mathcal{S}$
 2: while stack $\mathcal{S}$ is not empty do
 3:     pop node $u$ from stack $\mathcal{S}$ and remember it as visited
 4:     if $u = t$ then
 5:         return sequence of nodes visited
 6:     else if $u$ has not been visited before then
 7:         for all neighbors $u'$ of $u$, in order of increasing value according to $V(u'|u,t)$ do
 8:             push $u'$ onto stack $\mathcal{S}$
 9:             if $u' = t$ then
10:                 break // make sure t is picked in next iteration
11:             end if
12:         end for
13:     end if
14: end while
15: return 'no path between $s$ and $t$ exists'
```

Second, typical web-browsing software also imposes restrictions. The user can only follow links from the current page or return to the immediate predecessor by clicking the back button. Jumping between two unconnected pages is not possible, even if they were both visited separately before.

These constraints give rise to a simple navigation algorithm. As input, it takes the start node $s$, the target node $t$, and an evaluation function $V$. The latter is used to score each neighbor $u'$ of the current node $u$ and, due to the locality constraint, is a function of $u$, $u'$, and $t$ only. We write the value of $u$'s neighbor $u'$ as $V(u'|u,t)$. Ideally, $V$ should capture the likelihood of $u'$ being closer to the target than $u$ is. At each step, the algorithm chooses from those neighbors of the current node $u$ that have not previously been visited, picking the candidate $u'$ with the highest value $V(u'|u,t)$. If all neighbors of $u$ were previously visited, or if it has none, we backtrack, which is the analogue of a human's clicking the back button. In the worst case, the algorithm has to explore all nodes reachable from $s$ in order to find $t$. Pseudocode for the basic navigation procedure is listed in Algorithm 1.

Algorithm 1 may be considered an abstract meta-algorithm, since the order in which the neighbors of the current node are visited is determined by an arbitrary evaluation function

$V$. The concrete choice of $V$ is, of course, crucial for the performance of the algorithm. We experiment with several choices of $V$, which we describe next.

## 6.3 Implementations of the search algorithm

For the evaluation function $V$, we experiment with simple heuristics, as well as a set of machine learning methods.

### 6.3.1 Heuristic agents

The most basic agents navigate according to a single, simple criterion. In particular, degree-based [3] and similarity-based [46] methods have been proposed.

**Degree-based navigation (DBN).** For the degree-based agent, we define $V(u'|u,t) := \deg(u')$ (the outdegree of $u'$). Note that neither the current node $u$ nor the target $t$ are considered. This agent is based on the intuition that nodes with many neighbors are a good choice because they are more likely to contain a direct link to $t$, or at least offer many ways of continuing the search.

**Similarity-based navigation (SBN).** For the similarity-based agent, we let $V(u'|u,t) :=$ TF-IDF$(u',t)$, which is defined as the standard TF-IDF cosine of the textual contents of the candidate $u'$ and the target $t$. The choice of similarity as a feature is based on the notion of *homophily,* a property shared by many real-world networks: the more similar two nodes are, the more likely they are to link to each other.

**Expected-value navigation (EVN).** Şimşek and Jensen [88] combine degree and similarity in a simple probabilistic model, giving rise to 'expected-value navigation'. They make the assumption that each endpoint $w$ of a node $v$'s outlinks is sampled independently of the others with probability $q_{vw}$, which is a function of the similarity of $v$ and $w$. The value of a candidate $u'$ is then defined as the probability that at least one outlink of $u'$ ends in the target $t$, *i.e.*,

$$V(u'|u,t) := 1 - (1 - q_{u't})^{\deg(u')}. \tag{6.1}$$

Following Şimşek and Jensen [88], we assign all pairs of nodes $(v,w)$ to discrete bins based on TF-IDF$(v,w)$, and estimate $q_{vw}$ as the fraction of pairs in the respective bin that are direct neighbors.

### 6.3.2 Machine learning agents

Next, we introduce machine learning agents that all combine features linearly in a weighted sum but differ in how the weights are learned. Thus, they are more adaptable to the navigation domain at hand than the hard-coded heuristics.

The agents learn a separate weight vector (and thereby a separate evaluation function $V_i$) for each path position $i$; *i.e.*, the $i$-th iteration of the while loop of Algorithm 1 uses $V_i$ to evaluate neighbors.

**Supervised learning.** The first class of learning agents we discuss is based on supervised learning. In this scenario, the evaluation function may be seen as a classifier measuring the confidence that a given click is a 'good' choice. Training is done for each path position independently and is straightforward: sample pairs $(u,t)$; for each pair, choose a number of 'good' and an equal number of 'bad' neighbors of $u$; fit the weights. The way in which we define 'good' and 'bad' determines the ultimate behavior of the agent, and we experiment with two options: In the first approach, $u$'s neighbor $u'$ is defined as 'good' if it decreases the shortest-path length to $t$ (we call this 'lucrative *vs.* non-lucrative').[2] Our second approach is applicable when human search paths are available. Then we may define $u'$ as 'good' if the link from $u$ to $u'$ was ever clicked by a human under target $t$, and as 'bad' otherwise. This will make the agent behave similarly to human players ('human *vs.* non-human').

For weight-fitting, we experiment with two methods. First, *logistic regression* (LR) is a standard choice for binary classification. Second, a slightly more sophisticated classifier arises from the observation that, at each step, Algorithm 1 ever only picks the neighbor with the highest value. So choosing a neighbor may be considered a ranking problem, where

[2] Recall that the agents are allowed only local access to the network at navigation time, whereas shortest-path length, needed to assemble this type of training set, is a global notion. However, this information is needed only at training, not at navigation time, and in many scenarios this might be legitimate, since the agent might get full access to a small training portion of the entire network.

we require that the top-ranked candidate be good. In other words, whereas LR maximizes accuracy, we in fact want to maximize precision at $k$ with $k = 1$. A host of learning methods are available for learning to rank. In particular, we experiment with SVM-MAP [121], a *ranking support vector machine* that chooses the weights that maximize the mean average precision (MAP) of the induced ranking.

**Reinforcement learning (RL).** Supervised techniques require labeled ground-truth data. Either we need global network information (the shortest-path lengths between all node pairs) or large amounts of human click behavior data. Also, these algorithms work offline: first, all the training data are processed for learning weights; then, the algorithm is run with fixed weights.

A more natural paradigm for teaching an agent how to behave in a given environment can be found in reinforcement learning (RL) [93]. Here, the shortest-path lengths of node pairs need not be known ahead of time. The agent starts navigating with random weights, and only once it happens to find the target does it learn through positive feedback that the recent choices were good and adapts its weights to make similar choices more likely in the future. Therefore, the RL agent works online—it acts and learns at the same time.

In an RL setting, an agent moves from state to state by performing actions, at each step receiving a numerical reward (or punishment) indicating whether the current state is desirable (or not). In network navigation, states are nodes and actions are link clicks. The reward $r(u)$ received for reaching state $u$ is positive if $u = t$ and negative otherwise. Consider a path $\langle s = u_1, ..., u_n = t \rangle$ produced by running Algorithm 1. The cumulative reward obtained from $u_i$ onward is defined as $R(u_i) := \sum_{k \geq i} r(u_k)$. That is, the shorter the path, the higher the cumulative reward (since all summation factors but the last are negative), and reaching the target $t$ from $u_i$ as quickly as possible is equivalent to maximizing the cumulative reward $R(u_i)$. Let $R^*(\cdot)$ be the $R(\cdot)$ achieved by an optimal agent. Then, if we manage, for all $k$, to find weights $\mathbf{w}_k$ such that $V_k(u_k|u_{k-1}, t) = \mathbf{w}_k^\top \mathbf{f}(u_k|u_{k-1}, t)$ is a good approximation of $R^*(u_k)$, then choosing the neighbor $u_{i+1}$ of the current $u_i$ that maximizes $V_{i+1}(u_{i+1}|u_i, t)$ will tend to result in short solution paths (where $\mathbf{f}(u_{i+1}|u_i, t)$ is the feature vector of $u_{i+1}$, given the current node $u_i$ and the target $t$).

To learn the weights, a temporal difference (TD) update is performed in each iteration of the while loop, after pushing all neighbors of the current node onto the stack (*i.e.*, between lines 12 and 13 of Algorithm 1):

$$\mathbf{w}_i \leftarrow \mathbf{w}_i + \alpha \left[ r(u_i) + V_{i+1}(u_{i+1}|u_i,t) - V_i(u_i|u_{i-1},t) \right] \mathbf{f}(u_i|u_{i-1},t), \tag{6.2}$$

where $\alpha$ is a learning rate parameter, $u_i$ the current node, and $u_{i+1}$ the node on top of the stack, *i.e.*, the one that will be visited next. This update is also used in the so-called SARSA algorithm [83]. Instead of deriving it, we explain it in intuitive terms. Recall that we strive to achieve $V_k(u_k|u_{k-1},t) \approx R^*(u_k)$, for all $k$. In equation (6.2), $V_i(u_i|u_{i-1},t)$ is our estimate of $R^*(u_i)$ before receiving the reward $r(u_i)$, and $r(u_i) + V_{i+1}(u_{i+1}|u_i,t)$ is our estimate for it afterwards, so the update adjusts the weights in a gradient-descent way, making $V_i(u_i|u_{i-1},t)$ match the true $R^*(u_i)$ more closely.

To learn the weights, we have the agent solve a number of training navigation tasks, where start and target nodes are picked randomly. In the first task, the agent starts with random weights. Subsequently, task $j+1$ starts with the weights resulting from task $j$. During testing, weights are fixed and not updated any more.

## 6.4 Empirical evaluation

The above descriptions of our agents are abstract in the sense that we have not specified which features we consider. The appropriateness of features obviously depends on the specific network to be navigated. We now describe our evaluation network, list and analyze the features used, and compare the agents among each other as well as to humans.

### 6.4.1 Experimental setup

**The Wikipedia network.** We evaluate our navigational agents on the Wikipedia hyperlink network also used in the Wikispeedia game (Sec. 3.2), under the task of finding short paths between given pairs of Wikipedia articles. Wikipedia is an attractive validation domain mainly for two reasons. First, there is large-scale human navigational data for the Wikipedia network, collected through Wikispeedia, where the task formulation is the same as for our agents: find as short a path as possible from a given $s$ to a given $t$, only by clicking links (or the back button). Second, Wikipedia contains a wealth of rich knowledge of the kind that humans can easily deal with: they may use all their world knowledge and reasoning skills to find clever paths between the start and target articles; recall the example task 'MOZART to TERMINATOR' from the introduction. Since our agents do not possess such sophisticated, structured knowledge, the comparison sheds light on the question whether navigating rich knowledge resources is possible without human-like knowledge.

**Features.** In our experiments, we consider the following features (alongside a constant bias term):

1. $\deg(u')$: degree of the candidate $u'$, as in DBN;
2. $\deg(u)$: degree of the current node $u$;
3. TF-IDF$(u',t)$: the TF-IDF cosine of $u'$ and $t$, as in SBN, is important because of homophily;
4. TF-IDF$(u,t)$: the similarity between $u$ and $t$;
5. TF-IDF$(u,u')$: if this feature gets a large weight, then the next node is preferred to be similar to the current one; otherwise, clicks between unrelated nodes are preferred; *e.g.*, a small similarity between neighbors might be good in the beginning of paths, for quickly getting away from $s$, a large one towards the end, for homing in on $t$;
6. linkcos$(\cdot,\cdot)$: the equivalent of features 3–5, but with TF-IDF cosine replaced by *link cosine,* a similarity measure based on the outgoing links of the nodes under consideration: represent a node as the sparse vector of its outgoing links, where the non-zero entries are IDF-weighted; similarity is then defined as the cosine of such vectors [67]; the purpose of including this in addition to TF-IDF is to have redundancy: when one fails, the other might still work;
7. taxdist$(u',t)$: the Wikipedia version we use contains a tree-like hierarchy of categories, the leaves of which are populated by the articles; the 'taxonomic distance' between $u'$ and $t$ is defined as the number of edges in the tree that separate the two.

Features 1–6 are considered in logarithmic form, since the distributions of their values are approximately log-normal.

We emphasize that the content of the neighbor $u'$ is not available to the agents explicitly; *e.g.*, while the content of $u'$ is necessary to compute TF-IDF$(u',t)$ (feature 3), the agents know only this resulting number, not the content of $u'$ itself. We argue that this way the agents do not have an unfair advantage over humans, who also only get to see the textual content of an article once they visit it. Also, the agents know the degree of the neighbor $u'$ (feature 1), while humans do not. However, humans can probably roughly anticipate the degrees of articles based on intuitions about the importance of the corresponding concepts (more important concepts typically have higher degree), and we think that this difference in available information does not crucially affect the fairness of our comparison either.

**Training set creation for supervised learning.** Recall from the section about supervised learning that we consider classifiers for distinguishing 'lucrative *vs.* non-lucrative' and 'human *vs.* non-human' clicks, respectively. Whereas for 'human *vs.* non-human' the examples are necessarily created from human paths, this need not be the case for 'lucrative *vs.* non-lucrative'. As an alternative we explore using the transitions made on random shortest paths as positive examples, with negative examples subsampled randomly.

**Evaluation methodology.** As the basic notion of performance, we consider the number of nodes visited until the target is found, which we call *game length* or *search time* (we emphasize that search time is not measured in seconds). It is important to also count nodes from which the agent or human backtracked, since otherwise it would be trivial to always find an optimal solution simply by running breadth-first search.

We count a path as 'long' if its length is more than twice that of an optimal solution. This gives us an idea how often agents and humans do reasonably well.

We report the following quantities:

1. The *percentage of long paths,* which counts how often the agent took over twice the optimal number of steps.
2. The *overhead factor* of games, defined as $\ell/\ell^*$ for a game of actual length $\ell$ and optimal length $\ell^*$.

In our online game, humans often give up on missions; at each step, there is a probability of approximately 10% of this happening, such that only 46% of all games begun are finished. This might happen, *e.g.*, because the player gets frustrated, so oftentimes paths that would be long are not even recorded, resulting in an overestimation of performance. To be fair and give the automatic agents the chance to drop out as well, we introduce a small probability $\rho$ of giving up at each step. The missions given up are discarded, which lets us compute less noisy averages that are not skewed by the essentially failed searches in which nearly the entire graph must be explored before finding $t$ (only a small fraction of searches, *cf.* Fig. 6.1).

**Test sets.** We run our agents on two test sets. For the *random test set,* we sampled 100,000 $(s,t)$-pairs uniformly at random from the strongly connected core component. This large test set allows for tight performance estimates. We choose $\rho = .005$, which implies that a very long search is given up after an expected 200 steps. The probability of giving up within the first 1,000 clicks is 99%, that of not giving up within the first 10 clicks is 95%, such that we exclude the essentially failed searches without giving up too many searches too early.

The *human test set* contains 1,200 missions that were also played by humans, such that we can compare the agents' to human performance. Over 30,000 human paths have been collected through our online game, but we consider only the missions played at least 4 times (median 5 times), such that the estimate of human performance can be made more reliable by taking the median human search time for each mission. Since the agents find short paths for virtually all human missions, no drop-out option is required, and we set $\rho = 0$.

### 6.4.2 Results

Fig. 6.1 summarizes the performance of humans (blue), the learning (yellow), and the heuristic agents (red), in terms of the mean overhead factor and the percentage of searches longer than twice the optimum (*i.e.*, lower means better). Error bars are small and therefore not shown.

**Mean overhead factor and percentage long.** First, the learned agents (yellow) perform better than the heuristic ones (red). In particular, RL (B) and SVM-MAP (C) are best. What is more surprising is that, with the exception of the poorly performing DBN algorithm, even the simple agents do fairly well. In fact, most agents achieve lower mean overhead factors than humans: while humans (blue) take on average twice the optimal time, our best agents need only about 1.5 times the optimum on the same test set (upper left).

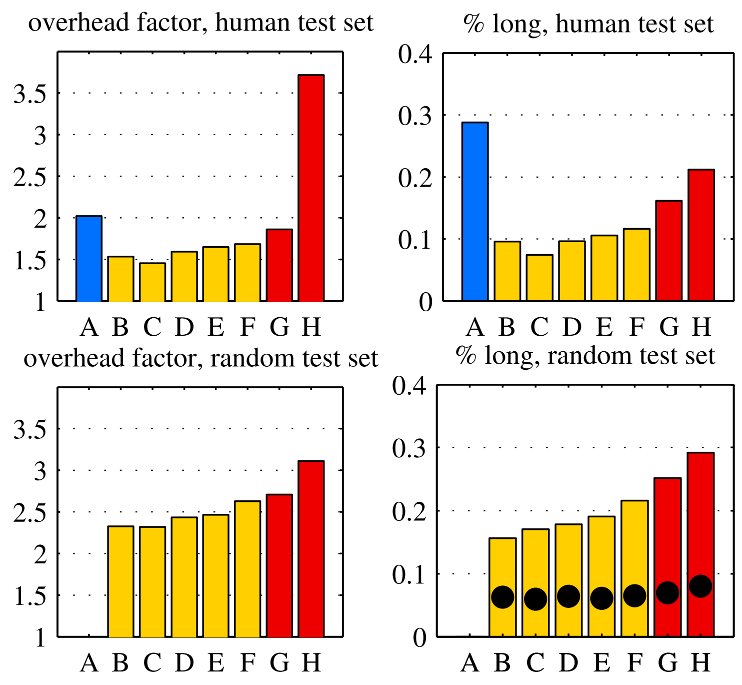


Figure 6.1: Results of the task of navigating Wikipedia automatically, with comparison to humans. **Left:** mean overhead factor (ratio of actual to optimal path length); **right:** percentage of long paths (*i.e.*, of more than twice the optimal length); **top:** 1,200 missions played by at least 4 humans each; **bottom:** 100,000 random missions; **blue (A):** humans; **yellow (B–F):** learning agents; **red (G–H):** heuristic agents; **black dots:** percentage of searches given up; **(A)** humans, **(B)** RL, **(C)** SVM-MAP, **(D)** LR (lucrative *vs.* non-lucrative, human paths), **(E)** LR (lucrative *vs.* non-lucrative, opt. paths), **(F)** LR (human *vs.* non-human), **(G)** SBN, **(H)** EVN. Not shown: DBN (overhead factors: 20 [top row] and 51 [bottom row]; percentages long: 0.81 and 0.86; percentage given up: 0.47).

Whereas 29% of human paths require more than twice the optimal number of steps, this is the case for only about 7% of the searches performed by the best agent (upper right). We also note that missions accomplished by humans appear to be easier: all agents perform better on these missions (top row) than on random ones (bottom row). The reason is probably that humans choose not to play the really hard missions in the first place, or start but do not finish them.

In Fig. 6.2 we take a closer look at how humans differ from agents. We plot only RL (blue) as a proxy for all learning agents, since they show rather similar behavior.

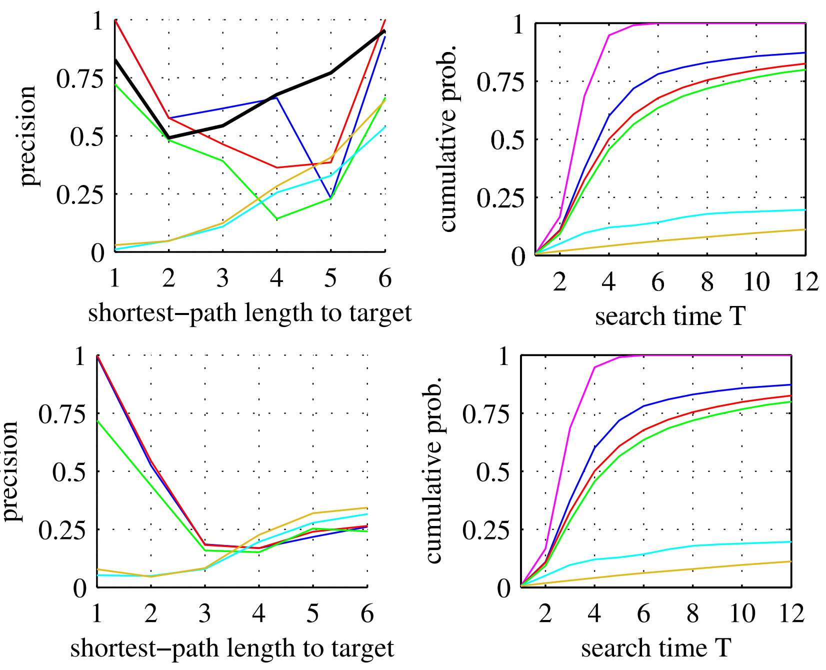


Figure 6.2: Results of the task of navigating Wikipedia automatically. **Left:** Precision (probability of top-ranked neighbor being closer to target than current node); **right:** CDF of search time; **top:** 1,200 missions played by at least 4 humans each; **bottom:** 100,000 random missions; **magenta:** optimal; **blue:** RL; **red:** SBN; **green:** EVN; **cyan:** DBN; **ocher:** random; **black:** human (dashed: hypothetical case of humans never giving up). (Color order corresponds to top-down order in lower right plot.)

**Precision.** The left column of Fig. 6.2 shows the precision of humans and of the agents' evaluation functions $V$, as a function of shortest-path length (SPL) to $t$, where precision is defined as the probability that the top-ranking neighbor decreases the SPL to $t$. Again, the top row shows the human, the bottom row the random test set (the estimates for the random test set are much less noisy because it is 100 times the size). The basic shape of curves is identical for most agents: precision is higher far away from and close to the target, and lower in between. Again, the only exception is DBN (cyan), whose mean precision is as bad as that of a randomly acting agent (ocher); since many targets are not reachable directly from high-degree nodes, DBN is not able to find these targets quickly. The learned agent's precision (blue) exceeds that of humans (*i.e.*, the agent makes good choices more reliably) when the SPL to $t$ is 1, 2, or 3. We note that 69% of all connected pairs have a SPL of at most 3, so one may say this agent makes better picks than humans most of the time.

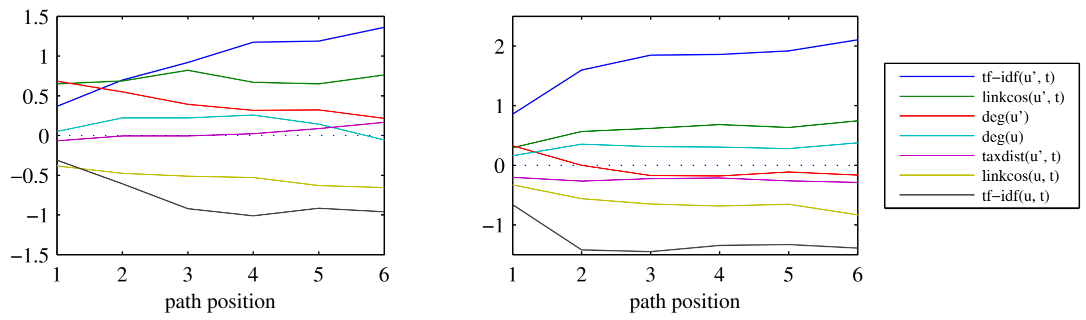


Figure 6.3: Weights learned by logistic regression for navigating Wikipedia automatically; one set of weights per path position. **Left:** classifier 'lucrative *vs.* non-lucrative'; **right:** 'human *vs.* non-human' (*cf.* section about supervised learning); TF-IDF$(u,u')$ and linkcos$(u,u')$ are neglected because their weights are close to zero everywhere. (Colors are ordered top-down at $x=3$ in left plot.)

**Cumulative search time distribution.** The right column of Fig. 6.2 shows the cumulative distribution function (CDF) of search time, *i.e.*, the fraction of searches of length at most $T$. One curve lying entirely above the others may be seen as the respective agent being better all around. The top curve (magenta) constitutes the upper bound of a hypothetical agent that always finds the shortest path; the bottom curve (ocher) is the lower bound set by a randomly acting agent. In between, the ranking is led by the learning agents (blue), SBN (red), EVN (green), and finally, far off, DBN (cyan). We see two reasons for the unexpected fact that plain SBN does better than EVN. On the one hand, even the TF-IDF cosine alone is a powerful feature because of Wikipedia's rich textual content and its homophily-based link structure. On the other hand, EVN includes degree information in a fixed way, which does not work well in the case of Wikipedia, possibly because the probabilistic model it adopts (Eq. 6.1) is not true: link endpoints are not picked independently, *e.g.*, no two links can have the same endpoints in our graph representation of Wikipedia (although a page may contain several anchors linking to the same neighbor, in the graph these are represented as a single edge). Nonetheless, the learners' increased performance proves that taking degree into account can help if done in a more flexible way.

**Comparison to humans.** Now consider the CDF of search time on the human test set (top right). Virtually all test missions are optimally solvable with at most 4 clicks (magenta). The learning agent manages to solve 75% of them with 4 clicks or fewer, while humans achieve only 30%. However, the agents sometimes fail entirely; when this happens, nearly all of the network has to be searched before $t$ is found. This is why the 100% mark is approached only slowly in the CDF curve. Humans, on the contrary, incur fewer complete failures, manifest in the fact that the human CDF (solid black) approaches the 100% mark faster. One reason for this is, of course, that humans give up rather than visit thousands of nodes. To account for this, we compute an ideal curve for the hypothetical case that humans are forced to finish every mission, based on measured drop-out rates [25]. We find that even then, the human curve (dashed black) starts to overtake the others around $T = 9$. A potential explanation of this effect could be that humans, while on average further away from optimal than the automatic agents, are less prone to get totally lost.

### 6.4.3 Feature analysis

In the previous section we found that combining several features linearly with learned weights yields better performance than the heuristic baselines. An analysis of the resulting weights can tell us about the relative importance of the respective features and lets us draw conclusions about what information should be taken into account by real search applications based on our algorithms.

Fig. 6.3 plots the weights learned by the logistic regression agent as functions of path position (feature values were normalized to mean 0 and variance 1, such that the weights are comparable). We show both types of classifiers (*cf.* section about supervised learning): ‘lucrative *vs.* non-lucrative’ (left) and ‘human *vs.* non-human’ (right). The weights are similar in both cases. Features capturing the similarity between $u'$ and $t$ (blue TF-IDF cosine, green link cosine) dominate on nearly all positions, and even more so as paths progress. This is in tune with the good performance of the SBN agent based on TF-IDF alone.

The weight of the candidate’s degree $\deg(u')$ (red) decreases in importance along paths. In the ‘lucrative *vs.* non-lucrative’ classifier, it has the strongest weight first but is then overtaken by similarity; in the ‘human *vs.* non-human’ classifier, it starts positive but becomes negative after two clicks. While navigating through high-degree hubs is very common in the beginning of paths, people seem to actively avoid it later on. Once searchers are close to $t$, they often reach it by navigating through specific articles similar to $t$, which often have low degree. This causes the purely degree-based agent (DBN) to perform so badly and contributes to the lesser performance of EVN: while high-degree nodes are not necessarily desirable, both heuristics always prefer them.

We remark that the basic structure 'degree to get away from the start, similarity to home in on the target' is in tune with our analysis of human Wikipedia navigation in Sec. 4.3.

Note that, once the weights have been learned, features not depending on the candidate $u'$ play no role. However, including them during learning may still affect the weights, since the quality of $u'$ may be caused by factors independent of $u'$ itself, and only by including these factors can this be accounted for; *e.g.*, TF-IDF$(u,t)$ and linkcos$(u,t)$ get large negative weights, as *any* $u'$ linked from a node $u$ with small similarity to $t$ is itself likely to be closer to $t$ than $u$ is.

## 6.5 Conclusions

This chapter investigates the question whether structured knowledge and high-level reasoning are necessary for navigating a rich information network such as Wikipedia—a legitimate question, given that connections are oftentimes latent, *i.e.*, not manifest in plain content words. Recall the introductory example mission from MOZART to TERMINATOR with the solution path ⟨MOZART, AUSTRIA, SCHWARZENEGGER, TERMINATOR⟩. The word overlap of the articles on AUSTRIA and TERMINATOR is small, so the plain TF-IDF cosine might not fire in this case. While this problem can probably in many cases be mitigated via generalization techniques such as Latent Semantic Analysis [50], this is not even necessary: as shown, even simple agents with bare-bones word-count knowledge and no generalization, let alone reasoning skills, can outperform humans.

Due to the verbose nature of Wikipedia articles, cases as above, in which a semantic connection is not realized verbally, might be rare. Also, even if the agent misses the AUSTRIA opportunity, any other neighbor of MOZART that increases the similarity to TERMINATOR is likely to be a good choice, too. This follows from the homophily present in Wikipedia: as we have seen in Sec. 4.2, Wikipedia links are distributed in a way that theoretically allows similarity-based navigation to find short paths.

Degree information can further help to increase performance, but care must be taken how it is incorporated. When a search begins, it is often beneficial to visit a hub with many links to diverse parts of the graph, but since many targets are reachable only via topic-specific articles of low degree, navigating according to degree only is bound to fail. Our feature analysis shows that degree should be weighted less strongly (and similarity more strongly) on later path positions.

What adds to making the automatic agents better than humans is that the latter often miss good opportunities. Even when the target is only one hop away, they pick the respective link with a probability of only 83% (*cf.* precision in Fig. 6.2). A possible explanation of this effect would be that humans typically have commonsense expectations about what links may exist and, based thereupon, form high-level plans regarding the route to take before even making the first click. Following through on their premeditated plans, subjects might often just skim pages for links they already expect to exist, thereby not taking notice of shortcuts hidden in the abundant textual article contents. For instance, in the above example, there might for some reason be a link from AUSTRIA directly to TERMINATOR, but as the searcher has set her mind on the more reasonable chain through SCHWARZENEGGER, she may not even become aware of that link's existence. We also hypothesize that humans' deeper understanding of the world is the reason why their searches fail completely (*i.e.*, take an extremely long time) less often: instead of exact, narrow plans, they sketch out rough, high-level strategies with backup options that are robust to contingencies. Analyzing subjects' strategy-making behavior to decide to what extent these hypotheses are true remains an interesting avenue of future research.

Our evaluation is restricted to Wikipedia, since one of the principal goals of this paper is to compare automatic to human navigation and since, to the best of our knowledge, Wikipedia is the only network for which large-scale targeted human navigation traces are available. (Vast numbers of click trails are also collected, *e.g.*, by browser toolbars, but most such trails are not targeted, and even for those that are, the target is not explicitly known.) We expect that our algorithms are equally applicable to other information networks with rich content, *e.g.*, social, citation, or peer-to-peer networks, or even the 'deep

Web'. While further experimentation is required to confirm this intuition, we anticipate that our algorithms could be used to design search interfaces that combine the strengths of humans and computers in a symbiosis of intuition and precision.

We deem our reinforcement learning approach particularly attractive, as it reasons directly about actions. Thus, if it is to be used to navigate more general networks than Wikipedia, it is possible to include additional actions other than link clicks, *e.g.*, submitting a form, posting a question to a Q&A site, requesting help through a crowdsourcing service, or querying the user for feedback. Extending our algorithm this way is certainly not trivial but might be a promising step towards creating smarter programs with the ability of exploring and acting on the Web more autonomously.

# Chapter 7

# Improving website hyperlink structure

## 7.1 Introduction

Websites are networks of interlinked pages, and a good website makes it easy and intuitive for users to navigate its content. One way of making navigation intuitive and content discoverable is to provide carefully placed hyperlinks. Consider for instance Wikipedia. The links that connect articles are essential for presenting concepts in their appropriate context, for letting users find the information they are looking for, and for providing an element of serendipity by giving users the chance to explore new topics they happen to come across without intentionally searching for them.

Links are also important from an information management perspective. They are among the most prominent features used by search engines, for crawling, indexing, and ranking. When building a web index, anchor texts serve as informative descriptors of the target page they point to, oftentimes more so than page titles and content [60]. Moreover, in search-result ranking, several standard methods, such as PageRank [12] and HITS [48], rely on the web graph induced by the hyperlinks.

And finally, links are important from a content provider perspective, since they make content discoverable to users and search engines. If a document has no incoming links, it cannot be accessed by a browsing user, nor can it be crawled and indexed by a search engine.

Unfortunately, maintaining website navigability can be difficult and cumbersome, especially if the site is large and changes frequently [23, 73]. For example, about 1,000 new pages are created on Wikipedia every day [122]. As a result, there is an urgent need to keep the site's link structure up to date, which in return requires much manual effort and does not scale as the website grows, even with thousands of volunteer editors as in the case of Wikipedia. Neither is the problem limited to Wikipedia; on the contrary, it may be even more pressing on other sites, which are often maintained by a single webmaster. Thus, it is important to provide automatic methods to help maintain and improve website hyperlink structure.

Considering the importance of links, it is not surprising that the problems of finding missing links and predicting which links will form in the future have been studied previously [56, 68, 74, 110] (*cf.* Sec. 2.5). Prior methods typically cast improving website hyperlink structure simply as an instance of a link prediction problem in a given network; *i.e.*, they extrapolate information from existing links in order to predict the missing ones. These approaches are often based on the intuition that two pages should be connected if they have many neighbors in common, a notion that may be quantified by the plain number of common neighbors, the Jaccard coefficient of sets of neighbors, and other variations [1, 56].

However, simply considering the existence of a link on a website is not all that matters. Networks in general, and hyperlink graphs in particular, often have traffic flowing over their links, and a link is of little use if it is never traversed. For example, in the English Wikipedia, of all the 800,000 links added to the site in February 2015, the majority (66%) were not clicked even a single time in March 2015, and among the rest, most links were clicked only very rarely. For instance, only 1% of links added in February were used more than 100 times in all of March (Fig. 7.1). Consequently, even if we could perfectly mimic Wikipedia editors (who would *a priori* seem to provide a reasonable gold standard for links to add), the suggested links would likely have little impact on website navigation. Thus, what is needed are automatic approaches to improving website hyperlink structure that account for humans' usage of hyperlinks.

Web server logs contain strong signals with regard to which existent hyperlinks are useful: from a user interface perspective, if a link is traversed often by humans then it is useful, and if it is never traversed by humans then it is redundant (although from a search

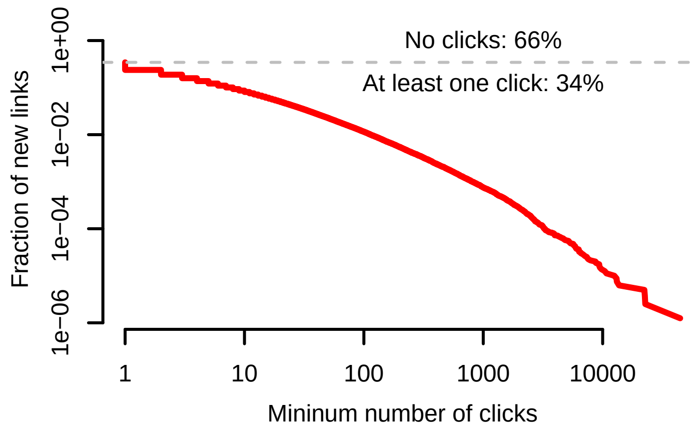


Figure 7.1: Complementary cumulative distribution function of the number of clicks received in March 2015 by links introduced in the English Wikipedia in February 2015.

engine's perspective it could still be useful for indexing and ranking). Hence we expect such logs to also contain clues into whether an as yet non-existent link *should* be there or not. For instance, if we often observe users going through page $s$ and ending up in page $t$, although $s$ does not directly link to $t$, then it might be a good idea to introduce a 'shortcut' link from $s$ to $t$.

As an analogy, consider the task of improving a road network. A civil engineer would not just look at the existing road segments and try to infer which road segments to build next. Rather, she would take into account how heavily each road segment is used and would then decide where it would make sense to add a shortcut, an extra lane, or a traffic light. We argue that similarly we should consider how the Web's hyperlink structure is used and decide on that basis what hyperlinks to add next. The *raison d'être* of hyperlinks is to enable navigation, so by creating hyperlinks that aid navigation, we are optimizing the right objective.

**Pull *vs.* push approaches to link recommendation.** We distinguish between two broad classes of link recommendation methods, which we term *'pull'* and *'push'*. A pull approach receives as input a specific article for which links are to be recommended; in other words, it pulls links from the recommendation engine for its present input. A push approach, on the contrary, processes a large dataset (such as web server logs or the current snapshot of the link network) in bulk and returns all links that it considers worthwhile

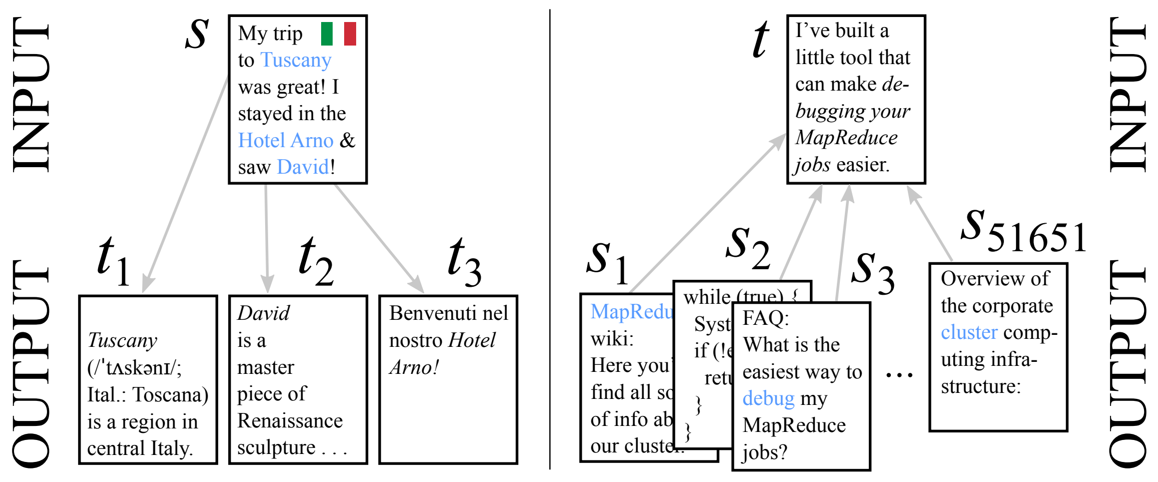

INPUT
OUTPUT
s
My trip to Tuscany was great! I stayed in the Hotel Arno & saw David!
t1
Tuscany (/ˈtʌskənɪ/; Ital.: Toscana) is a region in central Italy.
t2
David is a master piece of Renaissance sculpture . . .
t3
Benvenuti nel nostro Hotel Arno!
t
I've built a little tool that can make debugging your MapReduce jobs easier.
s1
MapRedu wiki: Here you find all so of info ab our cluste
s2
while (true) { Syst if (!e retu } }
s3
FAQ: What is the easiest way to debug my MapReduce jobs?
. . .
s51651
Overview of the corporate cluster computing infrastructure:
INPUT
OUTPUT
(a) Target prediction (b) Source prediction

Figure 7.2: Source *vs.* target prediction. In scenario **(a)** a source *s* is given, and the aim is to find mentions of relevant concepts in the source page and link them to appropriate targets. Here the set of candidate anchors is limited to the source document. In scenario **(b)** a target article *t* is given, and the aim is to identify sources that contain relevant mentions of *t* and could benefit from linking to it. Here every document mentioning the target is a potential source.

adding to the network, alongside scores that capture the quality of the individual links (in other words, it pushes an entire batch of new links onto the network).

**Source *vs.* target prediction.** Among the pull approaches, we further distinguish between those addressing the *target prediction* task and those addressing the *source prediction* task (Fig. 7.2):

- In the *target prediction* task (Fig. 7.2(a)), a source document $s$ is given, and the goal is to find mentions of relevant concepts in $s$ and link them to appropriate targets $t$.
- In the *source prediction* task (Fig. 7.2(b)), a target document $t$ is given, and the goal is to identify sources $s$ that contain relevant mentions of $t$ and would benefit from referencing $t$.

To illustrate these two abstract tasks, consider the following concrete scenarios (schematized in Fig. 7.2).

*Target prediction:* In a typical target prediction scenario, a traveler might write a blog post about his recent trip to Tuscany, in which he mentions several places, foods, landmarks, and historical persons. To provide more context, he wants to link these mentions to external resources, such as Wikipedia articles, online recipes, or hotel websites. Manually identifying the relevant concepts and linking them to the most appropriate target pages can be a tedious process [23, 73].

*Source prediction:* A typical source prediction scenario might involve a software engineer in a large company who has just finished a piece of code that could increase the productivity of many colleagues. She also created a documentation and tutorial page, but for it to be visible, she needs to link to it from other pages that colleagues interested in her code are likely to visit, such as company-internal wikis, Q&A fora, *etc.* Along the same lines, consider a Wikipedia editor who has just written a new article. The article is of little use if it is not reachable from other articles, so the editor wants to plant links into other articles to point to the new article.

Identifying appropriate sources for a given target is even more difficult than identifying appropriate targets for a given source: In target prediction, the set of candidate anchors is limited and can be identified by inspecting the source document. In source prediction, on the contrary, the set of candidate sources is practically unbounded, as any page on the Web is a potential candidate to link to the target.

**Existing approaches.** Previous work on recommending Wikipedia links has proposed mainly pull methods, in particular for the target prediction problem; *i.e.*, they annotate a given source document with links to external resources, primarily Wikipedia articles. One class of techniques can process arbitrary plain-text documents. Here the text of the input document is combined with background knowledge from Wikipedia's textual content as well as graph structure to predict outgoing links [63, 64, 68]. A second group of approaches takes Wikipedia articles as input and uses the already existing links (and possibly the text content) to predict further outgoing links for the input article, *e.g.*, based on adjacency-matrix factorization [110, 111], information retrieval techniques [31, 118], or machine learning [74].

With regard to the source prediction problem, a very simple approach would be to first collect all anchor texts in the document collection that frequently link to the target $t$ and to then link all as yet unlinked occurrences of these anchor texts to $t$ as well. Unfortunately, this method is too simplistic and suffers from some major drawbacks: first, a phrase might not be link-worthy in every context (*e.g.*, 'flower' is a good link anchor in the context of botany, but not in that of a wedding, where the concept of a flower needs no further explanation); second, the disambiguation problem is not addressed (*e.g.*, 'Florence' should link to FIRENZE in most contexts, but to FLORENCE, ALABAMA in the context of Lauderdale County, Alabama). Hence more sophisticated algorithms are required for solving the source prediction task.

Independent of whether the pull or push paradigm, and whether the source or target prediction task is considered, what is largely missing from the picture is the realization that, beyond text content and graph structure, there are additional sources of data that could be utilized in order to detect missing links more accurately. In particular, one such source of data that has remained mostly unexplored is human navigational traces on websites. Such traces are captured by the usage logs recorded on the server side by many websites, and the question arises: How can human click trails be harnessed in order to detect missing links?

**Chapter outline.** In this chapter, we use navigation traces in order to address the task of recommending useful new links to add to a website. We propose a pull approach that addresses the source prediction problem, as well as a push approach.

Our pull approach (Sec. 7.2) leverages targeted navigation traces collected through human-computation games such as Wikispeedia and The Wiki Game. In this setup, the website maintainer would specify a target page $t$ for which new incoming links should be recommended, which would trigger a number of game instances that all have $t$ as the target; link recommendations would be computed based on the games thus collected.

Our push approach (Sec. 7.3) leverages raw page request logs collected through standard web server software. It processes the logs in batch mode and returns a large, ranked list of links that would be well-suited for making the website more easily navigable. In this setup, traces need not be labeled with explicit navigation targets. This makes the method well-suited for websites beyond Wikipedia, for which targeted navigation traces (such as those collected through Wikispeedia) are typically not available.

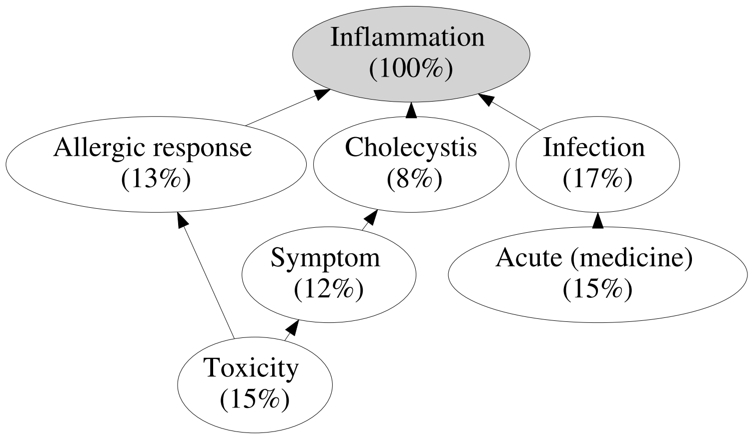


Figure 7.3: The final portions of several navigation paths with the same target $t =$ INFLAMMATION. The unfilled nodes are Wikipedia articles that appeared on paths to $t$. The number in each node indicates the percentage of paths with target $t$ that passed through that node.

## 7.2 Improving hyperlink structure using targeted navigation traces

In this section we propose a method for addressing the source prediction problem by leveraging targeted navigation traces to discover missing links; *i.e.*, given a target page, we find good sources to link to the target. We demonstrate the effectiveness of our approach using Wikipedia. We choose Wikipedia as our proof-of-concept domain because high-quality navigation traces labeled with the user's exact target are available for it, collected via human-computation games such as Wikispeedia or The Wiki Game (Sec. 3.2).

We reason as follows. If a page $s$ is traversed by many users in search of a target $t$, then this is an indicator that users expect the link from $s$ to $t$ to exist. So if $s$ does not link to $t$ yet (or not any more, for that matter), but contains a phrase that could be used as an anchor for $t$, then we should consider the link $(s,t)$ for addition.

As a concrete example, consider Fig. 7.3. The figure summarizes several navigational paths, all with the target $t =$ INFLAMMATION. Paths progress from bottom to top, and only the last few clicks are shown per path. Each node $s$ also contains the fraction of all paths with target INFLAMMATION that passed through $s$. For instance, we see that 17%

of times INFLAMMATION was reached from INFECTION and 13% of times it was reached from ALLERGIC RESPONSE. A considerable fraction of paths (15%) passed through ACUTE (MEDICINE), which does not link to $t$, although it mentions $t$ several times and could clearly benefit from a link to it.

The central part of our approach is that we mine many link candidates $(s,t)$ from a large number of navigation traces for each target $t$ and then rank these candidates by relevance.

After providing a detailed account of our method (Sec. 7.2.1), we perform a set of experiments (Sec. 7.2.4–7.2.6) using automatically (and thus only approximately) defined ground-truth missing links, as well as an evaluation involving human raters. In our automatically defined ground truth, we consider as positive examples of missing links such links that existed for a substantial amount of time but are missing from the latest Wikipedia snapshot. In our evaluation by humans, raters labeled the identified missing links as relevant or not. Experiments show that restricting the candidate set to pairs observed in paths and then ranking those candidates using a simple heuristic performs better than applying more sophisticated ranking methods to the set of all possible candidates (*i.e.*, including those not observed in paths). The reason why simple ranking methods suffice is that the 'heavy lifting' is done by the users before ranking, by using vast amounts of world knowledge to select the pages that are best-suited to link to the target.

### 7.2.1 Method for link recommendation

We address the source prediction task of Fig. 7.2(b): given a target page $t$, we suggest a good list of source pages $s$ that should link to $t$. Our method for source prediction consists of four steps, illustrated in Fig. 7.4:

**Step 1: Collect paths.** We start by collecting navigation paths with the given target $t$, up to time $T$.

**Step 2: Generate pairs.** For each path $p = \langle p_0, \dots, p_n = t \rangle$, the initial set of candidates is $\{(p_i,t) \,:\, 0 < i < n\}$, *i.e.*, every direct link from any page along the path to the target $t$ is initially a candidate. (The start page $p_0$ is exempt, since it is selected randomly and is therefore unlikely to be a useful candidate.) There are in general many paths for the same

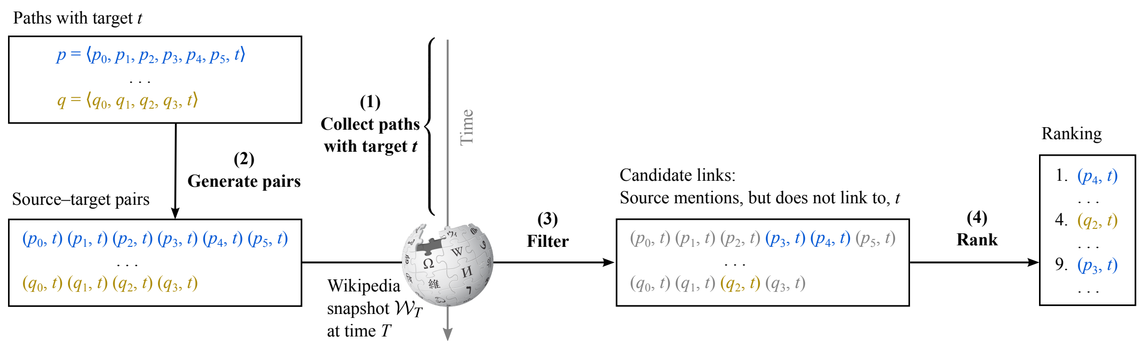

Paths with target $t$
$p = \langle p_0, p_1, p_2, p_3, p_4, p_5, t\rangle$
...
$q = \langle q_0, q_1, q_2, q_3, t\rangle$
(1) Collect paths with target $t$
Time
(2) Generate pairs
Source–target pairs
$(p_0, t)$ $(p_1, t)$ $(p_2, t)$ $(p_3, t)$ $(p_4, t)$ $(p_5, t)$
...
$(q_0, t)$ $(q_1, t)$ $(q_2, t)$ $(q_3, t)$
Wikipedia snapshot $\mathcal{W}_T$ at time $T$
(3) Filter
Candidate links: Source mentions, but does not link to, $t$
$(p_0, t)$ $(p_1, t)$ $(p_2, t)$ $(p_3, t)$ $(p_4, t)$ $(p_5, t)$
...
$(q_0, t)$ $(q_1, t)$ $(q_2, t)$ $(q_3, t)$
(4) Rank
Ranking
1. $(p_4, t)$
...
4. $(q_2, t)$
...
9. $(p_3, t)$
...

Figure 7.4: Overview of our approach for mining missing hyperlinks to a given target $t$ from human navigation traces. **(1)** Collect paths with target $t$ up to time $T$, and capture the reference Wikipedia snapshot $\mathcal{W}_T$ at time $T$. **(2)** Generate source–target pairs. **(3)** Filter the pairs based on $\mathcal{W}_T$: a pair $(s,t)$ becomes a candidate if $s$ mentions, but does not link to, $t$ in $\mathcal{W}_T$; also exclude $(s,t)$ if $s$ tends to appear in the second half of paths with target $t$. **(4)** Rank the candidate links.

target $t$ (upper left box in Fig. 7.4), so we take the union of the candidate sets resulting from all these paths (lower left box) as the initial candidate set for $t$.

**Step 3: Filter.** Next, we filter this initial set using the Wikipedia version $\mathcal{W}_T$ at time $T$, which serves as our *reference snapshot*. A link $(s,t)$, where $s \in \{p_1, \dots, p_{n-1}\}$, can be suggested only if it does not already exist in $\mathcal{W}_T$. Further, the source $s$ should contain a phrase that could serve as the anchor for a link to $t$; in other words, $s$ should mention $t$ in $\mathcal{W}_T$.

To detect pages that mention the target $t$, we construct the set $\mathcal{A}_t$ of all phrases that serve as anchor texts for $t$ across all articles in the reference Wikipedia snapshot[1] and subsequently define that $s$ mentions $t$ if it contains any phrase from $\mathcal{A}_t$.

In Chapter 4 we saw that navigation traces tend to consist of a ‘getting-away-from-the-start’ phase, in which the user attempts to reach a hub article that is well-connected in the network of pages, and a ‘homing-in-on-the-target’ phase, in which the user actively seeks out pages related to $t$. Guided by this finding, we apply one additional filtering step and include in our final candidate set (the box labeled ‘candidate links’ in Fig. 7.4) only

[1]In practice, we exclude (1) phrases that rarely (less than 6.5% of all cases [68]) serve as link anchors for any target, which excludes, *e.g.*, ‘A’ as an anchor for AMPERE, and (2) anchor texts for which $t$ is seldom (less than 1% of all cases) the target, which excludes, *e.g.*, ‘Florence’ as an anchor for FLORENCE, ALABAMA.

sources that tend to appear in the second half of paths with target $t$. More precisely, we first define the *relative path position* of $p_i$ along the path $p = \langle p_0, \dots, p_n = t \rangle$ to be $i/n$, and then discard the pair $(s,t)$ if the relative path position of $s$ on paths with target $t$ is less than or equal to 0.5 on average.

**Step 4: Rank.** Source candidate selection yields an unordered set of candidates for each target $t$. The goal of the next (and final) step in our pipeline is to turn this set into a meaningful ranking (step 4 in Fig. 7.4). Since the source prediction task (Fig. 7.2(b)) asks for sources for a given target $t$, we produce a separate ranking for each $t$. Several ranking methods are conceivable:

1. **Ranking by relatedness.** It seems reasonable to rank source candidates $s$ by their relatedness to $t$, since clearly a link is more relevant between articles with topical connections.[2] Since we deal with Wikipedia as our dataset, we choose relatedness measures based on Wikipedia (see below).

2. **Ranking by path frequency.** Navigation traces provide us with statistics about how frequently a source $s$ was traversed by users searching for target $t$. Based on this, we compute the *path frequency* of $s$ given target $t$, defined as the fraction of paths that passed through $s$, out of all the paths with target $t$. Intuitively, pages $s$ that were traversed more frequently on paths to $t$ should be better sources for links to $t$.

We experiment with two relatedness measures for case 1 above. The first is due to Milne and Witten [67] and is based on the inlink sets $\mathcal{S}$ and $\mathcal{T}$ of $s$ and $t$, respectively. It calculates the distance between $s$ and $t$ as the negative log probability of seeing a link from $\mathcal{S} \cap \mathcal{T}$ when randomly sampling a link from the larger one of the sets $\mathcal{S}$ and $\mathcal{T}$ (normalized to approximately lie between 0 and 1), and the relatedness as one minus that distance:

$$\mathrm{MW}(s,t) = 1 - \frac{\log(\max\{|\mathcal{S}|,|\mathcal{T}|\}) - \log(|\mathcal{S} \cap \mathcal{T}|)}{\log(N) - \log(\min\{|\mathcal{S}|,|\mathcal{T}|\})}, \tag{7.1}$$

where $N$ is the total number of Wikipedia articles.

[2] According to the Wikipedia linking guidelines [116], links should correspond to 'relevant connections to the subject of another article that will help readers understand the article more fully.'

The second relatedness measure is due to West *et al.* [110] and works by finding a low-rank approximation of Wikipedia's adjacency matrix via the singular-value decomposition (SVD). The pair $(s,t)$ corresponds to an entry $A[s,t]$ in the adjacency matrix $A$ and to an entry $A_k[s,t]$ in the rank-$k$ approximation $A_k$ obtained from $A$ via SVD. If $A[s,t] = 0$ and $A_k[s,t] \gg 0$ then $s$ does not link to $t$ yet but is a good candidate. Therefore we define the SVD-based relatedness as

$$\mathrm{SVD}(s,t) = A_k[s,t] - A[s,t]. \tag{7.2}$$

In our experiments on The Wiki Game, we use the reduced rank $k = 1{,}000$. Since the adjacency matrix is much smaller for Wikispeedia (Sec. 3.2), we use the smaller value of $k = 256$ there.

### 7.2.2 Exploratory analysis of link candidates

Having introduced our source prediction method, we now explore the dataset of human navigation traces collected through The Wiki Game to build intuitions on strengths and potential weaknesses of our approach. We use the Wikipedia version as of $T =$ 2014-01-02 as our reference snapshot $\mathcal{W}_T$ in all experiments.

**Number of pages on a path mentioning the target.** We count for each path $p = \langle p_0, \dots, p_n = t \rangle$ how often the target $t$ is mentioned across all visited nodes $p_1, \dots, p_{n-1}$ (excluding the randomly selected start page $p_0$) and find that, on average, $t$ is mentioned on 1.7 pages per path. Since $p_{n-1}$ contains a link to $t$, it is very likely to also mention $t$ (for our definition of a mention, *cf.* Sec. 7.2.1), which means that, on average, each path contains 0.7 additional pages that mention $t$.

Now consider the subset of visited pages that mention $t$. Out of these, 73% contain a link to $t$ in the reference Wikipedia snapshot $\mathcal{W}_T$. The remaining 27%, which do not link to $t$ in $\mathcal{W}_T$, are potentially good candidate sources to link to $t$, since these pages were actively chosen by the user while searching for $t$.

**Properties of pages along paths.** We also investigate which parts of a path carry most value for source prediction. Consider Fig. 7.5(a), which aggregates all paths and shows for each part of the path how likely the pages in that part are to mention $t$. In order to be able to aggregate paths of variable length, we adopt the notion of relative path position (Sec. 7.2.1). Fig. 7.5(a) uniformly buckets the range $[0,1]$ into five intervals and plots the average for each interval. We only include paths of at least five clicks, such that each path contributes to each bucket, and the page $p_{n-1}$ just before the target always falls into the last bucket.

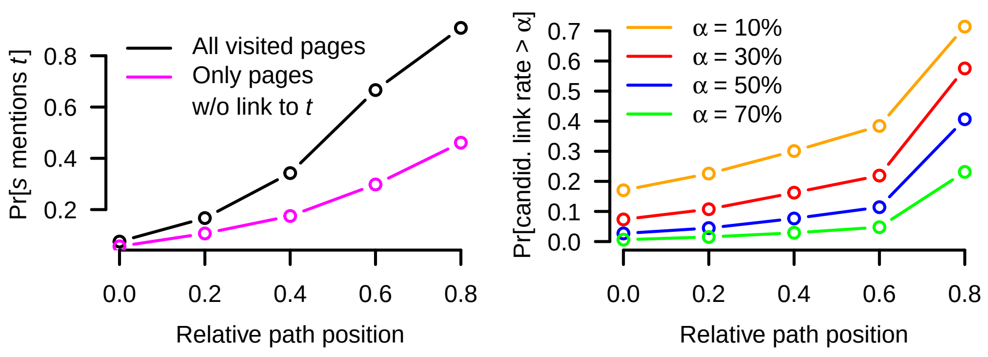
We see that target mentions become more frequent as paths progress (the black curve in Fig. 7.5(a)): two-thirds of pages with relative path positions in the interval $[0.6,0.8)$ mention the target, while at positions in $[0.8,1.0)$ nearly all pages (91%) do. We are particularly interested in mentions that are not accompanied by a link to $t$ (the magenta curve in Fig. 7.5(a)), since these are our source candidates. The figure tells us that candidates are more likely to appear towards the end of paths: at relative path positions in the interval $[0.6,0.8)$, 30% of pages without a link to $t$ mention $t$, and for the interval $[0.8,1.0)$, the fraction is as high as 46%.

We note that these curves are in tune with the results from Sec. 4.3.1, where we have shown that humans tend to follow a 'semantic gradient' during information network navigation, passing through articles that get ever more related to the target. In this light it makes a lot of sense that the rate of target mentions should increase as paths progress.

### 7.2.3 Obtaining ground truth based on Wikipedia evolution

In order to form intuitions about how meaningful our suggestions are, we would ideally like to evaluate for each relative path position how good the source candidates at that position are. However, ground-truth data is hard to come by; in order to make strong claims, we need to ask humans how good our predictions are. We do so later on (Sec. 7.2.5.2), but since obtaining human ratings is expensive and time-consuming, we preliminarily adopt a notion of ground truth that is approximate and biased, but nevertheless allows us to gain some initial insights. In this subsection, we define this approximate ground truth and analyze our navigation traces in terms of it.

We obtain a weak notion of ground truth from the evolution of Wikipedia's graph structure as follows. First we define the *link rate* of $(s,t)$ as the fraction of time $s$ contained a



(a) Fraction of target mentions (b) Fraction of good candidates

Figure 7.5: Exploratory analysis of link candidates. **(a)** Fraction of sources $s$ mentioning target $t$, as a function of the relative path position of $s$; the magenta curve is conditioned on $s$ not linking to $t$. **(b)** Fraction of candidates that are positive according to the automatically generated ground truth, as a function of relative path position, for several link-rate thresholds $\alpha$ (Sec. 7.2.3); source candidates mention, but do not link to, the target and are considered positive if their link rate is greater than $\alpha$.

link to $t$ since $s$ was created. (We compute these values based on Wikipedia's complete edit history.) Then we choose a *link-rate threshold* $\alpha \in [0\%, 100\%]$ and label a candidate link $(s,t)$ as positive if its link rate is greater than $\alpha$. Candidates with a positive label correspond to links that existed for a substantial amount of time, but got deleted before the reference Wikipedia snapshot $\mathcal{W}_T$ (*cf.* step 3 of Sec. 7.2.1). That is, such links could have been valuable for navigation, yet were removed at some point in time, so we argue that reintroducing them is likely to improve Wikipedia.

Consider a candidate $(s,t)$ labeled as positive according to the above definition. The link $(s,t)$ may (case 1), or may not (case 2), have existed during the game from which it was mined. Further, if it existed during the game, it may (case 1a), or may not (case 1b), have been clicked by the user. These three cases correspond to the following scenarios. If the link $(s,t)$ existed during the game and was clicked by the user (case 1a), but has been deleted since (as required for $(s,t)$ to be a candidate), then it is probably a good idea to suggest it for reintroduction. If the link existed during the game, but was not clicked by the user (case 1b), this means that she did not see it in her rush to reach $t$ as fast as possible (or else she would have clicked on it to immediately win the game); so either the user found another promising way to continue the search before seeing the link to $t$, or the link was too hard to find in the text of $s$, which is a signal that we should reintroduce that link and make it more obvious. Finally, if $(s,t)$ did not exist during the game (case 2) then the user could not possibly have taken it, although she might have intended to do so (since she actively navigated to $s$ while searching for $t$); in this case, too, $(s,t)$ might be a good link suggestion.

We refer to our automatically obtained labels as 'weak' because, by definition, they contain many *false negatives*. Wikipedia is an evolving organism, and an important part of our task is to suggest links which never existed. However, by the above link-rate threshold criterion, these links will be counted as negative examples. For an example of such a false negative, consider again Fig. 7.3, where the article on ACUTE (MEDICINE) should clearly link to INFLAMMATION, as it explains a concept critical to understanding the term ACUTE as used in medicine, but the link from ACUTE (MEDICINE) to INFLAMMATION is labeled as negative by the automatic ground truth, since ACUTE (MEDICINE) has never linked to INFLAMMATION in Wikipedia's history. In other words, the automatically obtained ground truth has high precision, but low recall of truly positive examples. Nonetheless, this weak ground truth is useful during development because it provides us with many labeled examples for free and allows for relative comparisons between different methods.

Fig. 7.5(b) captures this approximate notion of candidate quality, again broken up by relative path position. The graph shows that the fraction of positives obtained from the automatically obtained ground truth becomes higher for pages appearing later on in paths. We conclude that not only are mentions at later positions more frequent (Fig. 7.5(a)), but that they also correspond to better link anchors. (We try several values for the link-rate threshold $\alpha$, but the same trend holds for all thresholds.) This provides additional justification for our decision to include in our set of source candidates only sources that tend to appear in the second half of navigation traces with the given target (Sec. 7.2.1).

### 7.2.4 Evaluation methodology

In our experiments we compare five methods: Given a target $t$, we can either consider as source candidates the set of *all* articles that mention $t$ but do not link to it (across our entire reference Wikipedia snapshot $\mathcal{W}_T$); or we can subselect candidate sources based on whether we observe them on navigation paths. Further, we consider two relatedness measures for ranking (‘MW’ and ‘SVD’; *cf.* Sec. 7.2.1). This yields four combinations of candidate selection methods (‘none’ and ‘path-based’) and relatedness measures (‘MW’ and ‘SVD’). The fifth method requires no external relatedness measure but simply ranks candidates with respect to their frequency among paths with target $t$ (Sec. 7.2.1).

To sum up, we consider the following five methods for predicting missing links to a given target page $t$:

- **No selection, rank by MW:** Use all candidate sources and rank them based on the MW method (Eq. 7.1).
- **No selection, rank by SVD:** Use all candidate sources and rank them based on the SVD method (Eq. 7.2).
- **Path-based selection, rank by MW:** Only use candidates appearing in navigational traces and rank them based on the MW method.
- **Path-based selection, rank by SVD:** Only use candidates appearing in navigational traces and rank them based on the SVD method.
- **Path-based selection, rank by frequency:** Only use candidates appearing in navigational traces and rank them based on the frequency with which they appear in paths (*cf.* Sec. 7.2.1 for our definition of *path frequency*).

**Ground truth.** We perform a twofold evaluation, one based on the automatically obtained and approximate labels defined in Sec. 7.2.3 (we use the link-rate threshold $\alpha = 30\%$ throughout), the other based on labels obtained from human raters. For our human evaluation, we select a subset of targets, predict sources for them using the methods that performed best during the development phase on the automatic ground truth, and ask raters on Amazon Mechanical Turk [6] to label the top predictions. Here we get rid of the shortcomings of the automatic ground truth, on which we cannot obtain absolute performance numbers (mainly due to the high false-negative rate; Sec. 7.2.3), but have less data to work with.

We perform an evaluation by humans only on the predictions obtained on data from The Wiki Game.

**Evaluation metric.** As our evaluation metric, we use precision at $k$ for $k = 1, \dots, K$. We first calculate the $K$ precision values for each target separately and then compute the aggregate value for each $k$ by averaging over all targets. This means we can only include targets for which our methods find at least $K$ source candidates (which naturally shrinks the set of test targets). We use $K = 10$ for The Wiki Game, which defines our evaluation set of 699 targets. Since the Wikispeedia dataset contains fewer paths, we are less restrictive here and choose $K = 5$, obtaining an evaluation set of 181 targets.

We evaluate our approach on two datasets, Wikispeedia (*cf.* Sec. 3.2.1) and The Wiki Game (*cf.* Sec. 3.2.2). Comparing the two datasets, we notice that The Wiki Game, on the one hand, has the advantage of more data, in particular more paths per target. Wikispeedia, on the other hand, has the advantage of using a static Wikipedia snapshot (while The Wiki Game fetches pages from live Wikipedia on the fly and caches them for some time [19]), which allows for a different kind of evaluation, and of being publicly available, which makes our experiments reproducible. The bulk of our experiments is performed in Sec. 7.2.5 on data from The Wiki Game. Subsequently, Sec. 7.2.6 completes the evaluation by demonstrating that our algorithm works equally well on Wikispeedia.

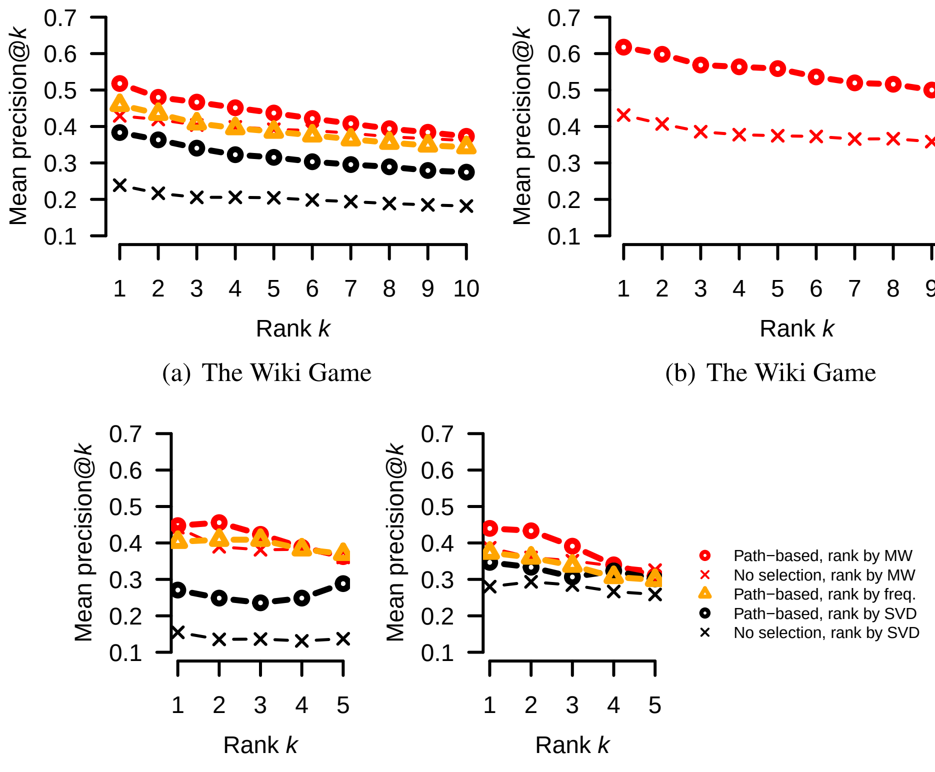
### 7.2.5 Evaluation on The Wiki Game

We start by evaluating our algorithm on data from The Wiki Game, first based on the automatically obtained ground truth, then by asking human raters.

#### 7.2.5.1 Evaluation using automatically obtained ground truth

The precision at $k$ curves for all five methods as evaluated on The Wiki Game are displayed in Fig. 7.12, and their performance is summarized in terms of the area under the precision at $k$ curve in Table 7.1.

Overall, we achieve good performance, especially given that our ground truth is of high precision but low recall, with many false negatives. Even though the precision at $k$ lies in the range between 0.4 and 0.5 (for path-based candidate selection and MW ranking), manual error inspection revealed that most suggested links make sense and are truly missing, and that, in fact, the Wikipedia community has simply never included these links into



(a) The Wiki Game (b) The Wiki Game

(c) Wikispeedia (d) Wikispeedia

Figure 7.6: Performance in terms of precision at *k* for different source selection and ranking methods on our two datasets. Bold lines represent path-based candidate selection. **(a)** The Wiki Game, automatically obtained ground truth (Sec. 7.2.5.1). **(b)** The Wiki Game, human ground truth (Sec. 7.2.5.2); only the MW ranking method was used in the human evaluation. Note that performance as evaluated by humans exceeds the estimate from the automated evaluation (Fig. 7.6(a)), *i.e.*, the latter underestimates the actual quality of suggested links. **(c)** Wikispeedia, automatically obtained ground truth, same evaluation methodology as applied to The Wiki Game ('standard evaluation'; Sec. 7.2.6). **(d)** Wikispeedia, automatically obtained ground truth, stricter evaluation (Sec. 7.2.6).

| Candidate selection | Rank by MW | Rank by SVD | Rank by path freq. |
|---:|---:|---:|---:|
| None | 39% | 20% | N/A |
| Path-based | 43% | 32% | 39% |

Table 7.1: Area under the precision at $k$ curve for no candidate selection *vs.* path-based candidate selection for all ranking measures (The Wiki Game; Fig. 7.6(a)). Note that path frequency is only applicable for path-based candidate selection.

the Wikipedia graph so that they could have made their way into our ground truth (*e.g.*, $(\text{ACUTE (MEDICINE)}, \text{INFLAMMATION})$ in Fig. 7.3).

Comparing the different methods, we observe that path-based candidate selection performs better than doing no subselection for both relatedness measures used in ranking. Path-based selection improves performance by a particularly large margin for the SVD-based ranking method, which has much lower precision at $k$ than the other methods. This establishes the fact that there is a lot of value in path-based candidate selection especially when the ranking measure does not excel by itself.

The margin between path-based selection and no selection is larger for smaller $k$, which means that considering navigational paths is particularly useful for predicting the top link sources.

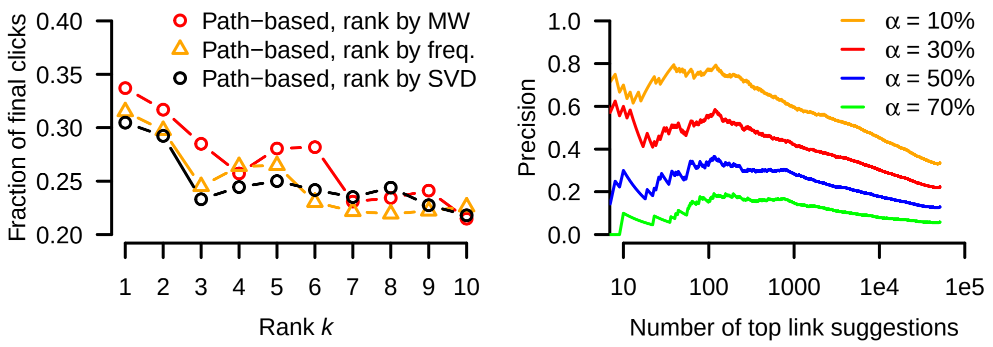
Note that both relatedness measures (MW and SVD) use the high-quality link structure of the Wikipedia page graph (Sec. 7.2.1). If we wanted to generalize our approach to domains beyond Wikipedia, we can easily imagine scenarios where no such high-quality relatedness measures are readily available (*e.g.*, when pages are not as topically coherent as Wikipedia articles, or pages have scarce content and are poorly interlinked). With such situations in mind, it is encouraging to see that our fifth measure ('rank by frequency' in combination with path-based candidate selection; the yellow curve in Fig. 7.6(a)) performs quite competitively. Recall that that ranking method does not rely on any external relatedness measure but simply ranks source candidates with respect to the frequency with which they appeared on paths with target $t$. This is an important observation because it means our method has the potential to generalize well to use cases where a good relatedness measure is not readily available.

**Reintroduction of valuable but deleted links.** By construction, the last click on a path always leads into the target. The fact that a user looked for, found, and clicked on this link is a very strong signal that the link is useful for navigation. Removing such links from Wikipedia is particularly harmful from a user-interface perspective, and it is desirable that a source prediction method suggest them for reintroduction. To see if our path-based candidate selection method meets this desideratum, Fig. 7.7(a) plots, for each rank *k*, the fraction of predicted links that were also the last link on the paths they were mined from. We observe that, while most suggested links were not clicked by humans (most likely because they were not present in the version of the Wikipedia page used by The Wiki Game), a substantial fraction (between 20% and 35%) correspond to links that existed at game time and were chosen by the user but do not exist in the reference snapshot any more. We conclude that our top suggestions are often links that were taken by the user as the last click to the target but have since been removed, and thus our method rightfully reintroduces such links back into Wikipedia.

**Total volume of added links.** So far we have conducted a per-target evaluation, by first computing precision at *k* values for each target and then averaging over all targets for each rank *k*. But it is also interesting to consider the total number of links we can suggest at a given precision level, across all targets, since this gives us an idea of the potential number of improvements we could make to Wikipedia by deploying our system. The results of this evaluation are presented in Fig. 7.7(b), which shows that we can make 1,000 link suggestions at a precision of 42%, and 10,000 suggestions at a precision of 30% (ranking candidates by frequency and assuming the same link-rate threshold $\alpha = 30\%$ used in our automated evaluation, corresponding to the red curve in Fig. 7.7(b)).

#### 7.2.5.2 Evaluation by human raters

Wikipedia is a continuously evolving entity. Although the link history, on the basis of which we defined our automatic ground truth, captures this evolution, it can only tell us which links are positive examples (because they persisted throughout a long period of time). However, there are many links that should be, but have never been, added to Wikipedia, and if our method suggests such a link, then the previous evaluation would count it as a bad suggestion. Therefore the above notion of ground truth suffers from false negatives. (At the end of this section, we confirm the prevalence of false negatives *post hoc,* after having



(a) Fraction of final clicks (b) Volume *vs.* precision

Figure 7.7: Results of link recommendation. **(a)** Fraction of clicks that were the final click along their respective path, as a function of rank. If a suggestion corresponds to the final click along a path, the suggested link must have existed at game time, so a useful link is effectively reintroduced into Wikipedia by that suggestion. **(b)** Precision as a function of the total number of top link suggestions made by our method across all targets, where suggestions are ranked by frequency (Sec. 7.2.1). Positive suggestions are those whose link rate lies above the respective link-rate threshold $\alpha$ (Sec. 7.2.3).

collected ground-truth labels from humans.) To combat this problem, we perform a more accurate evaluation by human raters in this section. Having done so, we can also confirm the prevalence of false negatives *post hoc* (see the end of this subsection).

**Methods compared via human evaluation.** In our human evaluation, we compare two of the top-performing methods: (1) path-based candidate selection with MW ranking and (2) no candidate selection with MW ranking. By using the same ranking method and only switching whether path-based candidate selection was performed, we can gauge the impact of the latter on performance.

**Target sampling.** In order to select targets on which to evaluate the predictions of the two methods, we stratify the base set of 699 targets by the number of paths observed per target and select ten targets from each decile, for a total of 100 test targets. The rationale behind stratification is that we want to avoid being biased towards targets for which the path-based candidate selection method can make a particularly large number of good predictions (because more data are available for those targets).

**Obtaining ratings through Amazon Mechanical Turk.** We use Amazon Mechanical Turk [6] for recruiting human raters. As in the automatic evaluation, our goal is to assess the precision at $k$, where $k = 1, \dots, 10$, for the two compared methods. In each rating task, the human evaluator was presented with a target $t$ and a set of 14 candidate sources and was asked to indicate which of the candidates should contain a link to the target article. There were no constraints on the number of source articles the rater could choose. The set of 14 candidate sources comprised the following entries:

1. Five predictions from each of the two compared methods (either suggestions 1 through 5 or suggestions 6 through 10 from each method).
2. Two control sources, sampled randomly from the set of all Wikipedia articles that link to the target $t$.
3. Two control non-sources, sampled randomly from the set of all Wikipedia articles and hence highly unlikely to link to $t$.

In cases where the two methods agreed on a suggestion, that suggestion was included only once in the set of source candidates, thereby making the presented list shorter than the maximum of 14 items. Also, to prevent any ordering bias, we shuffled the order of sources in the presented list. The task description is reproduced *verbatim* in Appendix A.

We paid 5¢ per task, and each task was presented to ten different workers. We consider a source to be a positive example if over half of the ten raters labeled it as such.

Fig. 7.6(b) presents the results. We observe that path-based candidate selection followed by MW ranking outperforms MW ranking on the set of all candidates by a large margin. Table 7.2, which summarizes the performance of both compared methods on the human-labeled ground truth (again as the area under the precision at $k$ curve) and compares it to the performance obtained on the automatically labeled ground truth, shows that the area under the precision at $k$ curves for path-based candidate selection increases by 12%, compared to the automatically obtained ground truth. On the other hand, when doing no candidate selection, the area under the curve decreases by 1%.

The reasons for the increased performance on the human-labeled ground truth are twofold. First, the automatically obtained ground truth uses only a historical notion of

| Candidate selection | Automatic ground truth | Human-labeled ground truth |
|---:|---:|---:|
| None | 39% | 38% |
| Path-based | 43% | 55% |

Table 7.2: Area under the precision at *k* curve for MW ranking, comparing the automated (Fig. 7.6(a)) and human (Fig. 7.6(b)) evaluations of our method run on data from The Wiki Game.

correctness in which many actually positive examples are mislabeled as negative. Second, MW relatedness alone, without performing path-based candidate selection, might not capture the notion of human-intuition–based similarity well. Path-based selection, on the contrary, captures exactly that quality by design, and it is thus not surprising that it prevails on a human-labeled ground truth by such a large margin.

Out of the controls that represent randomly selected sources already linking to the target page (item 2 in the above list of source-candidate types presented to raters), only 9% are labeled as positive by more than five of the ten raters, a value much lower than even our precision at 10 of about 50%. This tells us that the links we suggest are better than the average pre-existing link to the target.

Finally, out of the random control non-sources (item 3 in the above list), only one pair (GEOGRAPHY OF KOREA to SOUTH KOREA) was rated positive by more than five of the ten raters (we happened to sample a connected pair here). This statistic confirms that human labeling was not random.

**False negatives in the automated ground truth.** Now that we have human-labeled data, we can quantify the prevalence of false negatives in the automatically constructed ground truth. For this purpose, consider Fig. 7.8, which shows a histogram of the average human labels for the candidates that were labeled as negative according to the automatic ground truth. Here, 'average human label' refers to the average of the binary labels obtained from the ten human raters for each candidate. We see that a large fraction of the examples labeled as negative according to the automated ground truth are in fact positive examples according to the more reliable human ground truth.

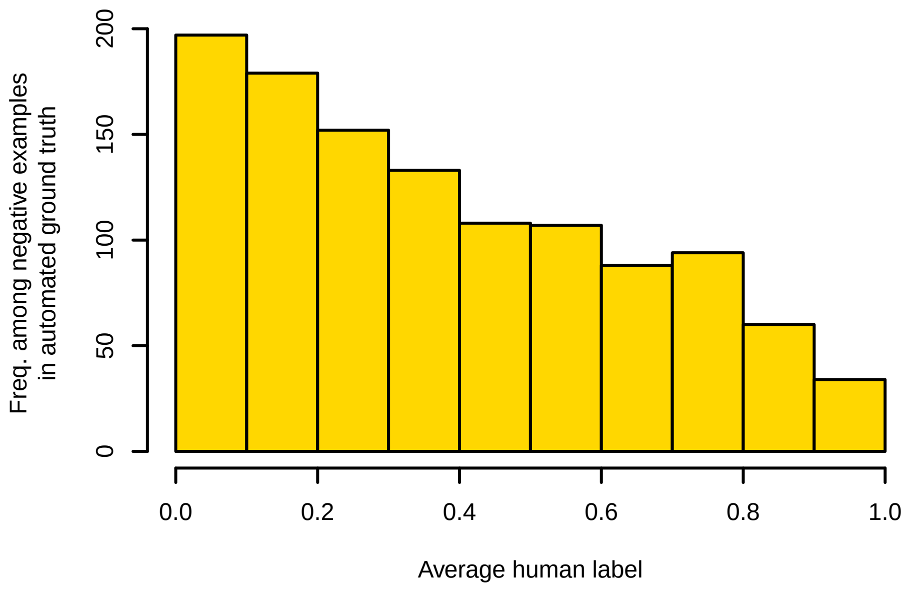


Figure 7.8: Histogram of average human labels for examples labeled as negative by the automatically obtained ground truth, highlighting the prevalence of false negatives in the latter.

### 7.2.6 Evaluation on Wikispeedia

To conclude the evaluation, we present results on the second dataset of navigation paths, collected via Wikispeedia. Recall from Sec. 7.2.4 that, while The Wiki Game has the advantage of more data, it also has a slight drawback: it does not use a static Wikipedia snapshot but rather fetches articles from live Wikipedia on the fly and caches them for some time [19], which means that we do not know the exact version of the article the user saw at game time; the versions used in different games may be different from each other and from the reference snapshot $\mathcal{W}_T$. Hence, our evaluation on The Wiki Game could not account for what a source article $s$ looked like at game time. Instead, we allowed for suggestion all links $(s,t)$ not present in the reference snapshot $\mathcal{W}_T$, regardless of whether they existed during the respective game, and our automated evaluation counted a suggestion $(s,t)$ as positive if the link was present for a substantial fraction of the entire lifetime of $s$. We call this the *standard evaluation.*

Wikispeedia, on the contrary, uses a static Wikipedia snapshot $\mathcal{W}$ [115], so we know exactly which links existed during the game. In the notation of Sec. 7.2.1 and Fig. 7.4, the snapshot $\mathcal{W}_T$ is replaced by $\mathcal{W}$, which is identical to the snapshot used in all games. This in turn allows for a stricter evaluation methodology: By allowing for suggestion only those links that were not present in $\mathcal{W}$, we permit only links that did not exist during the game and that the user could thus not possibly have clicked. Further, we count a suggestion as positive only if it has been present in the live Wikipedia for a substantial amount of time *after* the date of the static snapshot $\mathcal{W}$. If a suggested link did not exist during the game, but was added afterwards, this is an even stronger signal that the suggestion is good. Hence we call this the *stricter evaluation*.[3]

Now, if we can show that the stricter evaluation yields similar results to the standard evaluation on Wikispeedia, then we may argue by analogy that the stricter evaluation would likely give similar results on The Wiki Game, too, if such an evaluation were possible on that dataset.

The results are displayed in Fig. 7.6(c) and Fig. 7.6(d). As for The Wiki Game, we use the link-rate threshold (Sec. 7.2.3) $\alpha = 30\%$ for deciding if a suggestion is positive, and as described in the beginning of Sec. 7.2.4, we consider targets for which our algorithm can make at least $K = 5$ suggestions and show the average precision at $k$ for $k = 1, \dots, 5$.

Our first observation is that the standard-evaluation results are similar for Wikispeedia (Fig. 7.6(c)) and The Wiki Game (Fig. 7.12). In particular, the orderings of methods by performance are identical. (The results are somewhat less clean for Wikispeedia, due to the smaller amount of data.) Further, the outcome of the stricter evaluation (Fig. 7.6(d)) is similar to that of the standard evaluation (Fig. 7.6(c)), the main difference being that SVD ranking performs better under the stricter evaluation.

We therefore have reason to believe that the performance would also remain high on The Wiki Game under the stricter evaluation if this kind of evaluation were possible on that dataset, which corroborates the result that our algorithm finds good new links.

[3]When applying the standard evaluation to Wikispeedia, we use the same reference snapshot $\mathcal{W}_T$ also used for The Wiki Game in order to decide whether a source mentions, or links to, a target. Under the stricter evaluation, the static snapshot $\mathcal{W}$ is used for this purpose.

### 7.2.7 Discussion

This paper introduces an effective method for the source prediction problem (Fig. 7.2(b)), in which a target page $t$ is given, and the task is to find and rank sources $s$ that should link to $t$. Prior work (*cf.* Sec. 7.1) has primarily addressed the complementary target prediction problem (Fig. 7.2(a)), where $s$ is given and $t$ to be found. We consider source prediction more challenging than target prediction, since in the latter the set of link candidates is immediately given by the phrases contained in the source page $s$, whereas, in the former, every page could potentially be a source, so the set of source candidates must first be retrieved in a candidate selection step.

**Computational feasibility.** To illustrate this point, we briefly report on an experiment we had initially planned on doing. We intended to compare the performance of our method to the link predictions made by Milne and Witten's [68] machine-learned target prediction algorithm, but this was computationally infeasible: In order to use this target prediction method in a source prediction setting, we first had to find all articles $s$ mentioning $t$ (this required a full scan of a 44GB Wikipedia dump). Next, we intended to annotate each source $s$ with outgoing links and then rank $s$ according to the score it gives to $t$. However, each annotation takes on the order of several seconds [69], and nearly every article mentions at least one of the targets we want to evaluate, so we would have had to annotate essentially all of Wikipedia, which would have taken several million seconds, or several thousands of hours. One reason for the computational complexity of Milne and Witten's algorithm is that they (as well as other target prediction methods [63, 64, 118]) tend to spend significant effort on mention disambiguation.

On the contrary, in our approach we neither have to scan Wikipedia for articles that mention $t$, nor do we need to do any sophisticated disambiguation or ranking. We simply use as source candidates all pages seen in our navigation traces, look for mentions only in this small subset of all Wikipedia pages, and rank according to a simple precomputed metric or simple frequency counts. This is possible because the brunt of the computational effort is done by humans: since they actively seek out pages that are likely to link to the target, these pages tend to already be good source candidates, and issues such as disambiguation are much less critical.

**Applications beyond Wikipedia.** Now we address the question if and how our technique could apply beyond the realm of Wikipedia. We envision two ways forward.

The first approach is to use passively rather than actively collected log data for source candidate selection and ranking. We persue this idea in the next section (Sec. 7.3), where we devise a method that is similar to the one presented above but works on raw web server logs, rather than targeted navigation traces collected through a game. As part of our evaluation, we shall show that our method works on websites beyond the special case of Wikipedia (Sec. 7.3.4.2).

The second idea would be to gamify arbitrary websites. One could imagine a framework, *e.g.*, written in JavaScript, that would wrap the website of interest, recruit players, and ask them to navigate to the targets we are interested in linking to. This would require adding at least some initial links pointing to $t$ manually, such that $t$ is reachable by navigating. Furthermore, our method for finding valid anchors for the target, which is currently based on anchor-text/target-page pairs mined from Wikipedia (Sec. 7.2.1), would need to be adapted to the new domain. Possibilities would include the use of prevalent phrases from the target's title and content as anchor texts, or, akin to our current method, the use of anchor texts that are already being used in other pages to refer to the target.

On Wikipedia, the linking guidelines are explicitly stated [116], so links are fairly consistent. Further, each page is typically about a single, well-defined topic. These are among the reasons why machine-learning methods can infer powerful models for linking to Wikipedia articles. Websites other than Wikipedia are less likely to have the above properties, so it will be more difficult for statistical models to predict meaningful links. We expect methods for mining missing links directly from navigational traces to suffer less from this problem, since they do not take the detour through modeling the static structure of the link graph, but instead directly optimize navigability as the objective.

What we find especially promising in this light is a result from Fig. 7.6, namely that our method does not crucially rely on any measure of relatedness between pages: ranking our source candidates simply by the frequency with which they occurred in navigational traces for the given target (the yellow curves of Fig. 7.6) constitutes a competitive method. We believe that this makes our approach a strong candidate for the source prediction task on websites other than Wikipedia, where a notion of relatedness between pages might be much harder to obtain.

## 7.3 Improving hyperlink structure using raw server logs

The approach presented in the previous section leverages data collected through the human-computation game Wikispeedia in order to recommend new 'shortcut' links from pages along the path to the target page. Although this approach is simple and works well, it is limited by the requirement that the user's navigation target be explicitly known. Eliciting this information is practically difficult and, worse, may even be fundamentally impossible: often users do not have any target in mind, *e.g.*, during exploratory search [61] or when browsing the Web for fun or distraction. To alleviate this problem, we now develop a method that does not rely on a known navigation target, but rather works on passively collected server logs.

Our approach is language- and website-independent and is based on using access logs as collected by practically all web servers. Such logs record users' natural browsing behavior, and we show that server logs contain strong signals not only about which existing links are presently used, but also about how heavily currently nonexistent links would be used if they were to be added to the site. We build models of human browsing behavior, anchored in empirical evidence from log data, that allow us to predict the usage of any potentially introduced hyperlink $(s,t)$ in terms of its *clickthrough rate, i.e.*, the probability of being chosen by users visiting page $s$. The models are based on the following intuition: Consider a page $s$ not linked to another page $t$. A large fraction of users visiting $s$ and later on, via an indirect path, $t$, suggests that they might have taken a direct shortcut from $s$ to $t$ had the shortcut existed. Note that we do not require here that $t$ be the ultimate target of the user's navigation session; in fact, we do not even require that the session have a concrete target. All that we require is that $t$ appear on many paths that have previously passed through $s$.

Missing-link prediction systems can suggest a large number of links to be inserted into a website. But before links can be incorporated, they typically need to be examined by a human, for several reasons. First, not all sites provide an interface that lets content be changed programmatically. Second, even when such an interface is available, determining where in the source page to place a suggested link may be hard, since it requires finding, or introducing, a phrase that may serve as anchor text [68]. And third, even when identifying anchor text automatically is possible, human verification and approval might still be required in order to ensure that the recommended link is indeed correct and appropriate. Since the addition of links by humans is a slow and expensive process, presenting the complete ranking of all suggested links to editors is unlikely to be useful. Instead, a practical system should choose a well composed subset of candidate links of manageable size.

Hence, an important aspect of our solution is that we only suggest a small number of the top $K$ most useful links, where $K$ is a 'link budget' constraint. Consider a naïve approach that first ranks all candidate links by their probability of being used and then simply returns the top $K$. An extreme outcome would be for all $K$ links to be placed in a single page $s$. This is problematic because it seems *a priori* more desirable to connect a large number of pages reasonably well with other pages than to connect a single page extremely well. This intuition is confirmed by the empirical observation that the expected number of clicks users take from a given page $s$ is limited and not very sensitive to the addition of new links out of $s$. Thus we conclude that inserting yet another out-link to $s$ after some good ones have already been added to it achieves less benefit than adding an out-link to some other page $s'$ first. This diminishing-returns property, besides accurately mirroring human behavior, leads to an optimization procedure that does not over-optimize on single source pages but instead spreads good links across many pages.

We demonstrate and evaluate our approach on two very different websites: the full English Wikipedia and Simtk, a small community of biomedical researchers. For both websites we utilize complete web server logs collected over a period of several months and demonstrate that our models of human browsing behavior are accurate and lead to the discovery of highly relevant and useful hyperlinks. Having full access to the logs of Wikipedia, one of the highest-traffic websites in the world, is particularly beneficial, as it allows us to perform a unique analysis of human web-browsing behavior. Our results on Simtk show that our method is general and applies well beyond Wikipedia.

In the remainder of this section, we proceed as follows. First, we introduce the link placement problem under budget constraints, and design objective functions anchored in empirically observed human behavior and exposing an intuitive diminishing-returns property; we also provide an efficient algorithm for optimizing these objectives (Sec. 7.3.1). Second, we identify signals in the logs that can indicate the usage of a future link before it is introduced, and we operationalize these signals in simple but effective methods for finding promising new links and estimating their future clickthrough rates (Sec. 7.3.2). Third, we characterize the usage and impact of newly added links by analyzing human navigation traces mined from Wikipedia's server logs (Sec. 7.3.3). Finally, we evaluate our method in detail on Wikipedia and Simtk (Sec. 7.3.4).

### 7.3.1 The link placement problem

First we introduce the *link placement problem under budget constraints*. In this context, the budget refers to a maximum allowable number of links that may be suggested by an algorithm. As mentioned in the introduction, realistic systems frequently require a human in the loop for verifying that the links proposed by the algorithm are indeed of value and for inserting them. As we need to avoid overwhelming the human editors with more suggestions than they can respond to, we need to carefully select our suggestions.

Formally, the task is to select a set $A$ of suggested links of a given maximum size $K$, where the quality of the chosen set $A$ is determined by an *objective function* $f$:

$$\text{maximize} \quad f(A) \tag{7.3}$$

$$\text{subject to} \quad |A| \leq K. \tag{7.4}$$

In principle, any set function $f : 2^V \to \mathbb{R}$ may be plugged in (where $V$ is the set of pages on the site, and $2^V$ is the set of all subsets of $V$), but the practical usefulness of our system relies vitally on a sensible definition of $f$. In particular, $f$ should be based on a reasonable model of human browsing behavior and should score link sets for which the model predicts many clicks above link sets for which it predicts fewer clicks.

In the following we introduce two simplified models of human browsing behavior and propose three objective functions $f$ based on these models. Then, we show that two of these objectives have the useful property of diminishing returns and discuss the implications thereof. Finally, we show that all three objectives can be maximized exactly by a simple and efficient greedy algorithm.

#### 7.3.1.1 Browsing models and objective functions

We first define notation, then formulate two web-browsing models, and finally design three objective functions based on the models.

**Notation.** We model hyperlink networks as directed graphs $G = (V, E)$, with *pages* as nodes and *hyperlinks* as edges. We refer to the endpoints of a link $(s,t)$ as *source* and *target,* respectively. Unlinked pairs of nodes are considered potential *link candidates*. We denote the overall number of views of a page $s$ by $N_s$. When users are in $s$, they have a choice to either stop or follow out-links of $s$. To simplify notation, we model stopping as a link click to a special sink page $\varnothing \notin E$ that is taken to be linked from every other page. Further, $N_{st}$ counts the transitions from $s$ to $t$ via direct clicks of the link $(s,t)$. Finally, a central quantity in this research is the *clickthrough rate* $p_{st}$ of a link $(s,t)$. It measures the fraction of times users click to page $t$, given that they are currently on page $s$:

$$p_{st} = N_{st}/N_s. \tag{7.5}$$

The *stopping probability* is given by $p_{s\varnothing}$. Note that, since in reality users may click several links from the same source page, $p_{st}$ is generally not a distribution over $t$ given $s$.

Next we define two models of human web-browsing behavior.

**Multi-tab browsing model.** This model captures a scenario where the user may follow several links from the same page $s$, *e.g.*, by opening them in multiple browser tabs. We make the simplifying assumption that, given $s$, the user decides for each link $(s,t)$ independently (with probability $p_{st}$) if he wants to follow it.

**Single-tab browsing model.** This is a Markov chain model where the user is assumed to follow exactly one link from each visited page $s$. This corresponds to a user who never opens more than a single tab, as might be common on mobile devices such as phones. The probability of choosing the link $(s,t)$ is proportional to $p_{st}$.

Based on these browsing models, we now define objective functions that model the value provided by the newly inserted links $A$. Note that the objectives reason about the clickthrough rates $p_{st}$ of links $A$ that do not exist yet, so in practice $p_{st}$ is not directly available but must be estimated from data. We address this task in Sec. 7.3.2.

**Link-centric multi-tab objective.** To capture the value of the newly inserted links $A$, this first objective assumes the multi-tab browsing model. The objective computes the expected number of new-link clicks from page $s$ as $\sum_{t:(s,t)\in A} p_{st}$ and aggregates this quantity across all pages $s$:

$$f_1(A) = \sum_s w_s \sum_{t:(s,t)\in A} p_{st}. \tag{7.6}$$

If we choose $w_s = N_s$ then $f_1(A)$ measures the expected number of clicks received by all new links $A$ jointly when $s$ is weighted according to its empirical page-view count. In practice, however, page-view counts tend to follow heavy-tailed distributions such as power laws [4], so a few top pages would attract a disproportional amount of weight. We mitigate this problem by using $w_s = \log N_s$, *i.e.*, by weighting $s$ by the order of magnitude of its raw count.

**Page-centric multi-tab objective.** Our second objective also assumes the multi-tab browsing model. However, instead of measuring the total expected number of clicks on all new links $A$ (as done by $f_1$), this objective measures the expected number of source pages on which at least one new link is clicked (hence, we term this objective *page-centric,* whereas $f_1$ is link-centric):

$$f_2(A) = \sum_s w_s \left( 1 - \prod_{t:(s,t)\in A} 1 - p_{st} \right). \tag{7.7}$$

Here, the product specifies the probability of no new link being clicked from $s$, and one minus that quantity yields the probability of clicking at least one new link from $s$. As before, we use $w_s = \log N_s$.

**Single-tab objective.** Unlike $f_1$ and $f_2$, our third objective is based on the single-tab browsing model. It captures the number of page views upon which the one chosen link is one of the new links $A$:

$$f_3(A) = \sum_s w_s \frac{\sum_{t:(s,t)\in A} p_{st}}{\sum_{t:(s,t)\in A} p_{st} + \sum_{t:(s,t)\in E} p_{st}}. \tag{7.8}$$

The purpose of the denominator is to renormalize the independent probabilities of all links—old and new—to sum to 1, so $p_{st}$ becomes a distribution over $t$ given $s$. Again, we use $w_s = \log N_s$.

#### 7.3.1.2 Diminishing returns

Next, we note that all of the above objective functions are monotone, and that $f_2$ and $f_3$ have a useful diminishing-returns property.

Monotonicity means that adding one more new link to the solution $A$ will never decrease the objective value. It holds for all three objectives:

$$f(A \cup \{(s,t)\}) - f(A) \geq 0. \tag{7.9}$$

The difference $f(A \cup \{(s,t)\}) - f(A)$ is called the *marginal gain* of $(s,t)$ with respect to $A$.

The next observation is that, under the link-centric multi-tab objective $f_1$, marginal gains never change: regardless of which links are already present in the network, we will always have

$$f_1(A \cup \{(s,t)\}) - f_1(A) = w_s p_{st}. \tag{7.10}$$

This means that $f_1$ assumes that the solution quality can be increased indefinitely by adding more and more links to the same source page $s$. However, we will see later (Sec. 7.3.3.2) that this may not be the case in practice: adding a large number of links to the same source page $s$ does not automatically have a large effect on the total number of clicks made from $s$.

This discrepancy with reality is mitigated by the more complex objectives $f_2$ and $f_3$. In the page-centric multi-tab objective $f_2$, further increasing the probability of at least one new link being taken becomes ever more difficult as more links are added. This way, $f_2$ discourages adding too many links to the same source page.

The same is true for the single-tab objective $f_3$: since the click probabilities of all links from $s$—old and new—are required to sum to 1 here, it becomes ever harder for a link to obtain a fixed portion of the total probability mass of 1 as the page is becoming increasingly crowded with strong competitors.

More precisely, all three objective functions $f$ have the *submodularity* property:

$$f(B \cup \{(s,t)\}) - f(B) \leq f(A \cup \{(s,t)\}) - f(A), \quad \text{for } A \subseteq B, \tag{7.11}$$

but while it holds with equality for $f_1$ (Eq. 7.10), this is not true for $f_2$ and $f_3$. When marginal gains decrease as the solution grows, we speak of *diminishing returns.*

To see the practical benefits of diminishing returns, consider two source pages $s$ and $u$, with $s$ being vastly more popular than $u$, *i.e.*, $w_s \gg w_u$. Without diminishing returns (under objective $f_1$), the marginal gains of candidates from $s$ will nearly always dominate those of candidates from $u$ (*i.e.*, $w_s p_{st} \gg w_u p_{uv}$ for most $(s,t)$ and $(u,v)$), so $u$ would not even be considered before $s$ has been fully saturated with links. With diminishing returns, on the contrary, $s$ would gradually become a less preferred source for new links. In other words, there is a trade-off between the popularity of $s$ on the one hand (if $s$ is frequently visited then links that are added to it get more exposure and are therefore more frequently used), and its 'crowdedness' with good links, on the other.

As a side note we observe that, when measuring 'crowdedness', $f_2$ only considers the newly added links $A$, whereas $f_3$ also takes the pre-existing links $E$ into account. In our evaluation (Sec. 7.3.4.1), we will see that the latter amplifies the effect of diminishing returns.

#### 7.3.1.3 Algorithm for maximizing the objectives

We now show how to efficiently maximize the three objective functions. We first state two key observations that hold for all three objectives and then describe an algorithm that builds on these observations to find a globally optimal solution:

1. For a single source page $s$, the optimal solution is given by the top $K$ pairs $(s,t)$ with respect to $p_{st}$.

**Algorithm 2** Greedy marginal-gain link placement

```
Input: Hyperlink graph $G = (V, E)$; source-page weights $w_s$; clickthrough rates $p_{st}$;
    budget $K$; objective $f$
Output: Set $A$ of links that maximizes $f(A)$ subject to $|A| \leq K$
1:  $Q \leftarrow$ new priority queue
2:  for $s \in V$ do
3:      $\Sigma_E \leftarrow \sum_{(s,t) \in E} p_{st}$   // sum of $p_{st}$ values of old links from s
4:      $\Sigma \leftarrow 0$   // sum of new $p_{st}$ values seen so far for s
5:      $\Pi \leftarrow 1$   // product of new $(1 - p_{st})$ values seen so far for s
6:      $C \leftarrow 0$   // current objective value for s
7:      for $t \in V$ in decreasing order of $p_{st}$ do
8:          $\Sigma \leftarrow \Sigma + p_{st}$
9:          $\Pi \leftarrow \Pi \cdot (1 - p_{st})$
10:         if $f = f_1$ then $C' \leftarrow w_s \cdot \Sigma$
11:         else if $f = f_2$ then $C' \leftarrow w_s \cdot (1 - \Pi)$
12:         else if $f = f_3$ then $C' \leftarrow w_s \cdot \Sigma / (\Sigma + \Sigma_E)$
13:         Insert $(s, t)$ into $Q$ with the marginal gain $C' - C$ as value
14:         $C \leftarrow C'$
15:     end for
16: end for
17: $A \leftarrow$ top $K$ elements of $Q$
```

2. Sources are 'independent': the contribution of links from $s$ to the objective does not change when adding links to sources other than $s$. This follows from Eq. 7.6–7.8, by noting that only links from $s$ appear in the expressions inside the outer sums.

These observations imply that the following simple, greedy algorithm always produces an optimal solution[4] (pseudocode is listed in Algorithm 2). The algorithm processes the data source by source. For a given source $s$, we first obtain the optimal solution for $s$ alone by sorting all $(s,t)$ with respect to $p_{st}$ (line 7; *cf.* observation 1). Next, we iterate over the sorted list and compute the marginal gains of all candidates $(s,t)$ from $s$ (lines 7–15). As marginal gains are computed, they are stored in a global priority queue (line 13); once

[4] For maximizing general submodular functions (Eq. 7.11), a greedy algorithm gives a $(1 - 1/e)$-approximation [72]. But in our case, observation 2 makes greedy optimal: our problem is equivalent to finding a maximum-weight basis of a uniform matroid of rank $K$ with marginal gains as weights; the greedy algorithm is guaranteed to find an optimal solution in this setting [28].

computed, they need not be updated any more (*cf.* observation 2). Finally, we return the top $K$ from the priority queue (line 17).

### 7.3.2 Estimating clickthrough rates

In our above exposition of the link placement problem, we assumed we are given a clickthrough rate $p_{st}$ for each link candidate $(s,t)$. However, in practice these values are undefined before the link is introduced. Therefore, we need to estimate what the clickthrough rate for each nonexistent link would be in the hypothetical case that the link were to be inserted into the site.

Here we propose four ways in which historical web server logs can be used for estimating $p_{st}$ for a nonexistent link $(s,t)$. Later (Sec. 7.3.4.1) we evaluate the resulting predictors empirically.

**Method 1: Search proportion.** What we require are indicators of users' need to transition from $s$ to $t$ before the direct link $(s,t)$ exists. Many websites provide a search box on every page, and thus one way of reaching $t$ from $s$ without taking a direct click is to 'teleport' into $t$ by using the search box. Therefore, if many users search for $t$ from $s$ we may interpret this as signaling the need to go from $s$ to $t$, a need that would also be met by a direct link. Based on this intuition, we estimate $p_{st}$ as the *search proportion* of $(s,t)$, defined as the number of times $t$ was reached from $s$ via search, divided by the total number $N_s$ of visits to $s$.

**Method 2: Path proportion.** Besides search, another way of reaching $t$ from $s$ without a direct link is by navigating on an indirect path. Thus, we may estimate $p_{st}$ as the *path proportion* of $(s,t)$, defined as $N^*_{st}/N_s$, where $N^*_{st}$ is the observed number of navigation paths from $s$ to $t$. In other words, path proportion measures the average number of paths to $t$, conditioned on first seeing $s$.

**Method 3: Path-and-search proportion.** Last, we may also combine the above two metrics. We may interpret search queries as a special type of page view, and the paths from $s$ to $t$ via search may then be considered indirect paths, too. Summing search proportion and path proportion, then, gives rise to yet another predictor of $p_{st}$. We refer to this joint measure as the *path-and-search proportion*.

**Algorithm 3** Power iteration for estimating path proportions from pairwise transitions probabilities

**Input:** Target node $t$, pairwise transition matrix $P$ with $P_{st} = p_{st}$
**Output:** Vector $Q_t$ of estimated path proportions for all source pages when $t$ is target page

```
1: $Q_t \leftarrow (0,\dots,0)^T$
2: $Q_{tt} \leftarrow 1$
3: $Q'_t \leftarrow (0,\dots,0)^T$   // used for storing the previous value of $Q_t$
4: while $\|Q_t - Q'_t\| > \varepsilon$ do
5:     $Q'_t \leftarrow Q_t$
6:     $Q_t \leftarrow PQ_t$      // recursive case of Eq. 7.12
7:     $Q_{tt} \leftarrow 1$        // base case of Eq. 7.12
8: end while
```

**Method 4: Random-walk model.** The measures introduced above require access to rather granular data: in order to mine keyword searches and indirect navigation paths, one needs access to complete server logs. Processing large log datasets may, however, be computationally difficult. Further, complete logs often contain personally identifiable information, and privacy concerns may thus make it difficult for researchers and analysts to obtain unrestricted log access. For these reasons, it is desirable to develop clickthrough rate estimation methods that manage to make reasonable predictions based on more restricted data. For instance, although the Wikipedia log data used in this paper is not publicly available, a powerful dataset derived from it, the matrix of pairwise transition counts between all English Wikipedia articles, was recently published by the Wikimedia Foundation [119]. We now describe an algorithm for estimating path proportions solely based on the *pairwise* transition counts for those links that already exist.

As defined above, the path proportion measures the expected number of paths to $t$ on a navigation trace starting from $s$; we denote this quantity as $Q_{st}$ here. For a random walker navigating the network according to the empirically measured transitions probabilities, the following recursive equation holds:

$$Q_{st} = \begin{cases} 1 & \text{if } s = t, \\ \sum_u p_{su} Q_{ut} & \text{otherwise.} \end{cases} \tag{7.12}$$

The base case states that the expected number of paths to $t$ from $t$ itself is 1 (*i.e.*, the random walker is assumed to terminate a path as soon as $t$ is reached). The recursive case defines that, if the random walker has not reached $t$ yet, he might do so later on, when continuing the walk according to the empirical clickthrough rates.

**Solving the random-walk equation.** The random-walk equation (Eq. 7.12) can be solved by power iteration. Pseudocode is listed as Algorithm 3. Let $Q_t$ be the vector containing as its elements the estimates $Q_{st}$ for all pages $s$ and the fixed target $t$. Initially, $Q_t$ has entries of zero for all pages with the exception of $Q_{tt} = 1$, as per the base case of Eq. 7.12 (lines 1–2). We then keep multiplying $Q_t$ with the matrix of pairwise transition probabilities, as per the recursive case of Eq. 7.12 (line 6), resetting $Q_{tt}$ to 1 after every step (line 7). Since this reset step depends on the target $t$, the algorithm needs to be run separately for each $t$ we are interested in.

The algorithm is guaranteed to converge to a fixed point under the condition that $\sum_s p_{st} < 1$ for all $t$ (proof omitted for conciseness), which is empirically the case in our data.

## 7.3.3 Evaluation: effects of new links

The goal of this section is to investigate the effects of adding new links and assess some of the assumptions we made when formulating the link placement problem (Sec. 7.3.1). In particular, we answer the following questions: First, how much click volume do new links receive? Second, are several new links placed in the same source page independent of each other, or do they compete for clicks?

To answer these questions, we consider the around 800,000 links $(s,t)$ added to the English Wikipedia in February 2015 and study all traces involving $s$ in the months of January and March 2015.

### 7.3.3.1 Usage of new links

In the introduction we have already alluded to the fact that new links tend to be rarely used, to the extent that 66% of the links added in February were not used even a single time in March, and only 1% were used more than 100 times (Fig. 7.1).

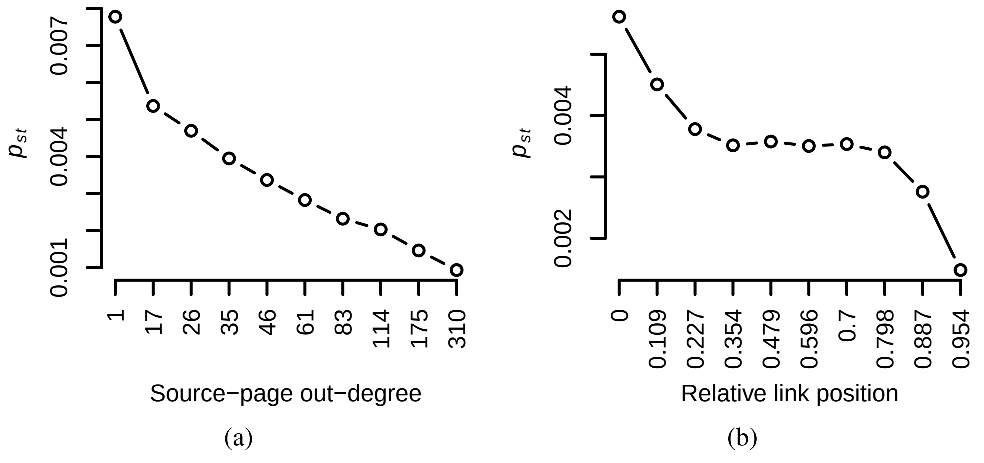


Figure 7.9: Clickthrough rate as function of **(a)** source-page out-degree and **(b)** relative position of link in wiki markup of source page (0 is top; 1 is bottom). Lower bin boundaries on $x$-axis.

While Fig. 7.1 plots absolute numbers, Fig. 3.1(b) shows the complementary cumulative distribution function (CCDF) of the clickthrough rate $p_{st}$, stratified by the out-degree of the source page $s$. We observe that, depending on source-page out-degree, between 3% and 13% of new links achieve over 1% clickthrough, with higher values for lower-degree source pages. This finding is confirmed by Fig. 7.9(a), which plots the clickthrough rate $p_{st}$ against source-page out-degree and shows a strong negative trend.

Finally, Fig. 7.9(b) confirms the fact that the popularity of a link is also correlated with its position in the source page [49], with links appearing close to the top of the page achieving a clickthrough rate about 1.6 times as high as that of links appearing in the center, and about 3.7 times as high as that of links appearing at the bottom.

### 7.3.3.2 Competition between links

Traces are generally trees, not chains (Sec. 3.1.1). While in chains, $p_{st}$ would form a distribution over $t$ given $s$, *i.e.*, $\sum_t p_{st} = 1$, this is not necessary in trees, where several next pages may be opened from the same page $s$, such that, in the most extreme case (when all $p_{st} = 1$), $\sum_t p_{st}$ may equal the out-degree of $s$.

In the multi-tab browsing model with its independence assumption (Sec. 7.3.1.1), we would see no competition between links; the larger the out-degree of $s$, the larger the expected number of clicks from $s$ for fixed $p_{st}$. In its pure form, this model seems unlikely to be true, since it would imply a reader scanning the entire page, evaluating every link option separately, and choosing to click it with its probability $p_{st}$. In a less extreme form, however, it is well conceivable that adding many good links to a page $s$ might significantly increase the number of links a given user chooses to follow from $s$.

In Sec. 7.3.3.1 we already saw that links from pages of higher out-degree tend to have lower individual clickthrough rates, which may serve as a first sign of competition, or interaction, between links. As our link placement method is allowed to propose only a limited number of links, the question of interaction between links is of crucial importance. Next we investigate this question more carefully.

First we define the *navigational degree* $d_s$ of $s$ to be the total number of transitions out of $s$, divided by the total number of times $s$ was seen as an internal node of a tree, *i.e.*, without stopping there:

$$d_s = \frac{\sum_{t \neq \varnothing} N_{st}}{N_s - N_{s\varnothing}}. \tag{7.13}$$

In other words, the navigational degree of $s$ is simply the average number of transitions users make out of $s$, given that they do not stop in $s$. We also define the *structural degree,* or *out-degree,* of $s$ to be the number of pages $s$ links to.

Next, we examine the relation of the structural degree of $s$ with (1) the probability of stopping at $s$ and (2) the navigational degree of $s$, across a large and diverse set of pages $s$.

Fig. 7.10(a), which has been computed from the transition counts for 300,000 articles in January 2015, shows that stopping is less likely on pages of larger structural degree, with a median stopping probability of 89% for pages with less than 10 out-links, and 76% for pages with at least 288 out-links. Additionally, given that users do not stop in $s$, they make more clicks on average when $s$ offers more links to follow (median 1.00 for less than 10 out-links *vs.* 1.38 for 288 or more out-links; Fig. 7.10(b)).

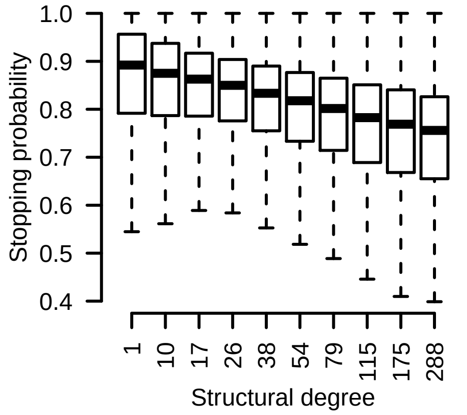
These effects could be explained by two hypotheses. It is possible that (i) simply adding more links to a page also makes it more likely that more links are taken; or (ii) structural degree may be correlated with latent factors such as 'interestingness' or 'topical complexity': a more complex topic $s$ will have more connections (*i.e.*, links) to other topics that might be relevant for understanding $s$; this would lead to more clicks from $s$ to those topics, but not simply because more link options are present but because of inherent properties of the topic of $s$.

To decide which hypothesis is true, we need to control for these inherent properties of $s$. We do so by tracking the same $s$ through time and observing whether changes in structural degree are correlated with changes in navigational degree for fixed $s$ as well. In particular, we consider two snapshots of Wikipedia, one from January 1, 2015, and the other from March 1, 2015. We take these snapshots as the approximate states of Wikipedia in the months of January and March, respectively. Then, we compute structural and navigational degrees, as well as stopping probabilities, based exclusively on the links present in these snapshots, obtaining two separate values for each quantity, one for January and one for March. For a fixed page $s$, the difference between the March and January values now captures the effect of adding or removing links from $s$ on the stopping probability and the navigational degree of $s$.

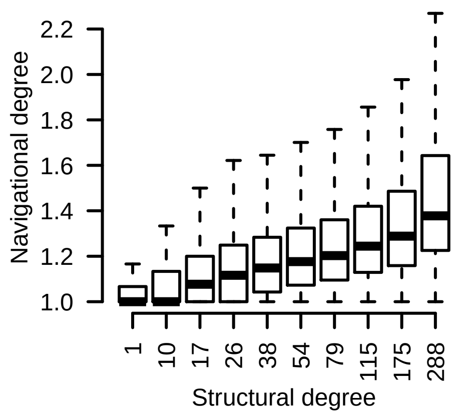
Fig. 7.10(c) and 7.10(d) show that this effect is minuscule, thus lending support to hypothesis ii from above: as structural degree grows or shrinks, both stopping probability (Fig. 7.10(c)) and navigational degree (Fig. 7.10(d)) vary only slightly, even for drastic changes in link structure; *e.g.*, when 100 links or more are added, the median relative increase in navigational degree is still only 0.3%, and when 100 links are deleted, the median relative decrease is only 0.1%.

This is not to say that adding links has no effect at all, but this effect stems from extreme, rather than typical, values, as indicated by the upward (downward) shift of interquartile ranges (Fig. 7.10(d)) as links are added (removed).

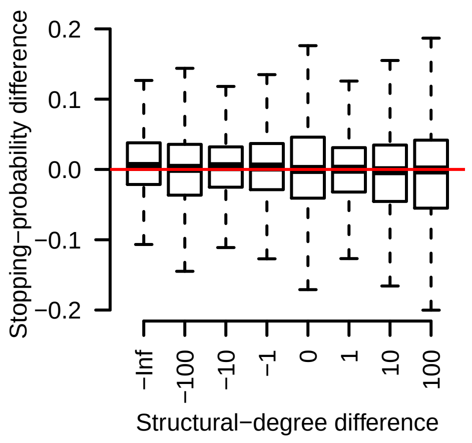
In a nutshell, simply adding more links does not increase the overall number of clicks taken from a page. Instead, links compete with each other for user attention. This observation has important implications for user modeling and hence for budget-constrained link placement algorithms: one should avoid spending too much of one's budget on the same source page, since this will not increase click volume indefinitely; rather, one should spread high-clickthrough links across many different source pages.



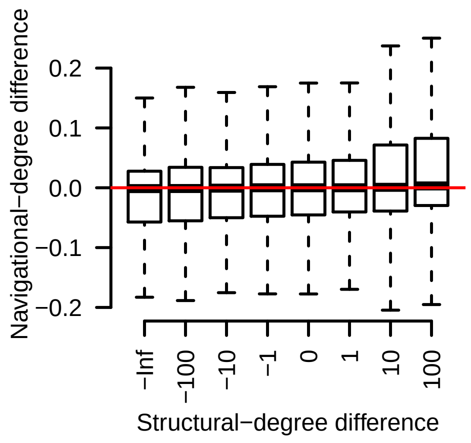
(a)



(b)



(c)



(d)

Figure 7.10: Across different source pages, structural degree is **(a)** negatively correlated with stopping probability, and **(b)** positively correlated with navigational degree. When fixing the source page, however, structural degree has only a small effect on **(c)** stopping probability and **(d)** navigational degree. The *y*-axes in (c) and (d) are relative with respect to the values from January. Lower bin boundaries on *x*-axes. Boxes show quartiles; whiskers show the full range without outliers.

These findings justify, *post hoc,* the diminishing-returns properties of the objective functions $f_2$ and $f_3$ (Sec. 7.3.1.2).

## 7.3.4 Evaluation: link placement

Next we demonstrate the universality of our approach by evaluating it on two very different websites, Wikipedia and Simtk.

### 7.3.4.1 Wikipedia

The analysis of results for Wikipedia is divided into three parts. First, we show that the estimation methods from Sec. 7.3.2 are suited for predicting the clickthrough rates of new links. Since the evaluation set of added links is known only after the fact, a practically useful system must have the additional ability to identify such links on its own before they are introduced. Therefore, we then check whether a large predicted $p_{st}$ value indeed also means that the link is worthy of addition. Last, we investigate the behavior of our algorithm for link placement under budget constraints.

We also developed a graphical user interface that makes it easy to add missing links to Wikipedia: users are shown our suggestions and accept or decline them with the simple click of a button [77].

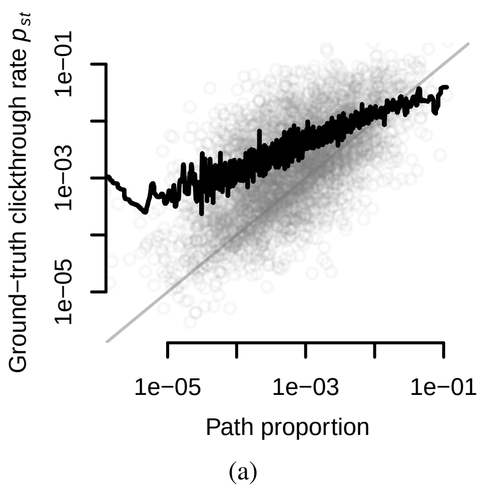
**Estimating clickthrough rates**

As described in Sec. 7.3.3, we have identified 800,000 links that were added to the English Wikipedia in February 2015. Here we evaluate how well we can predict the clickthrough rates $p_{st}$ of these links in March 2015 from log data of January 2015, *i.e.*, before the link was added. In particular, we compare five different methods (Sec. 7.3.2):

1. search proportion,
2. path proportion,
3. the combined path-and-search proportion,
4. random walks, and
5. a mean baseline.

| | Mean absolute error | Pearson corr. | Spearman corr. |
|---|---|---|---|
| Path proportion | **0.0057** (±0.0003) | 0.58 (±0.12) | **0.64** (±0.01) |
| Search proportion | 0.0070 (±0.0004) | 0.49 (±0.22) | 0.17 (±0.02) |
| Path-and-search proportion | **0.0057** (±0.0003) | **0.61** (±0.13) | **0.64** (±0.01) |
| Random walks | 0.0060 (±0.0004) | 0.53 (±0.13) | 0.59 (±0.02) |
| Mean baseline | 0.0072 (±0.0004) | 0.20 (±0.06) | 0.34 (±0.02) |

Table 7.3: Comparison of clickthrough rate estimation methods on Wikipedia, with bootstrapped 95% confidence intervals.



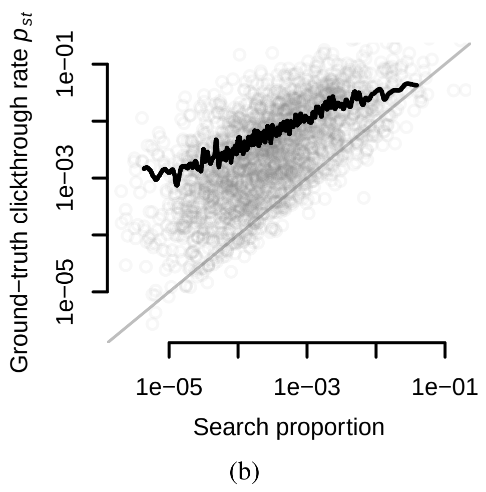


Figure 7.11: Results of clickthrough prediction on Wikipedia. **(a)** Path proportion *vs.* clickthrough rate. **(b)** Search proportion *vs.* clickthrough rate (log–log; black lines correspond to gray dots kernel-smoothed on linear scales).

The mean baseline makes the same prediction for all candidates originating in the same source page $s$, namely the average clickthrough rate of all out-links of $s$ that already exist.

Table 7.3 evaluates the results using three different metrics: mean absolute error, Pearson correlation, and Spearman rank correlation. The table shows that across all metrics path proportion and path-and-search proportion perform best. Further, it is encouraging to see that the random-walk–based predictor, which only requires the pairwise transition matrix, is not lagging behind by much. Also, recall from Fig. 7.9(a) that $p_{st}$ is to a large part determined by source-page out-degree. This is why the mean baseline performs quite well (mean absolute error of 0.72%, *vs.* 0.57% achieved by the best-performing method) even though it predicts the same value for all links from the same page (but see the next section for when the baseline fails). For a graphical perspective on the relation between predicted and ground-truth values, we refer to Fig. 7.11(a) and 7.11(b).

### Predicting link addition

Our models for clickthrough prediction reason about the hypothetical scenario that a link $(s,t)$ is added to the site. In the previous subsection we showed that the models manage to predict the future usage of a link fairly well for the subset of links that were actually added. But, as stated above, this set is not known when the predictors are deployed in practice.

Intuitively, a large predicted clickthrough rate $p_{st}$ should also imply that $(s,t)$ is a useful link and should thus be added. Here we test whether this is actually the case by evaluating whether our high-ranking predictions using the January data correspond to links that were added to Wikipedia in February.

For this task, we need a ground-truth set of positive and negative examples. We require a non-negligible number of positive examples for which our methods can potentially predict a high $p_{st}$ value. Since our methods are based mainly on path and search counts, we therefore consider all links added in February with at least 10 indirect paths or searches in January; call this set $L$. Since in reality most page pairs are never linked, we also need to add a large number of negative examples to the test set. We do so by including all out-links of the sources, and in-links of the targets, appearing in $L$. Using this approach, we obtain a set of about 38 million link candidates, of which 9,000 are positive examples. The aforementioned threshold criterion is met by 7,000 positive, and 104,000 negative, examples. Therefore, while there is a fair number of positive examples with many indirect paths and searches, they are vastly outnumbered by negative examples with that same property; this ensures that retrieving the positive examples remains challenging for our methods (*e.g.*, returning all links meeting the threshold criterion in random order yields a precision of only 6%).

Given this labeled dataset, we evaluate the precision at $k$ of the candidate ranking induced by the predicted $p_{st}$ values. The results, plotted in Fig. 7.12, indicate that our methods are well suited for predicting link addition. Precision stays high for a large set of top predictions: of the top 10,000 predictions induced by the path-proportion and random-walk measures, about 25% are positive examples; among the top 2,000 predictions, as many as

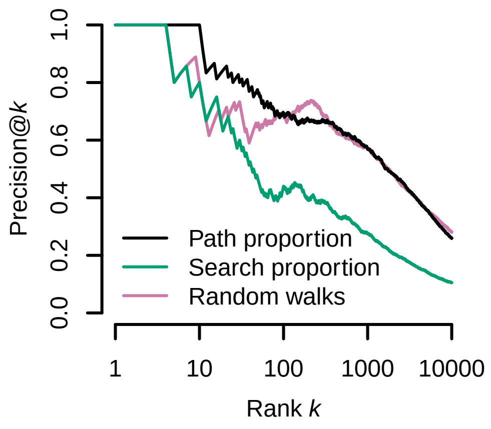


Figure 7.12: Precision at $k$ on link prediction task (Sec. 7.3.4.1; logarithmic $x$-axis); path-and-search proportion nearly identical to path proportion and thus not shown.

50% are positive. As before, random walks perform nearly as well as the more data-intensive path proportion, and better than search proportion.

Note that the mean baseline, which performed surprisingly well on the clickthrough rate prediction task (Sec. 7.3.4.1), does not appear in the plot. The reason is that it fails entirely on this task, producing only two relevant links among its top 10,000 predictions. We conclude that it is easy to give a good $p_{st}$ estimate when it is known which links will be added. However, predicting whether a link should be added in the first place is much harder.

## Link placement under budget constraints

The above evaluations concerned the quality of predicted $p_{st}$ values. Now we rely on those values being accurate and combine them in our budget-constrained link placement algorithm (Sec. 7.3.1.3). We use the $p_{st}$ values from the path-proportion method.

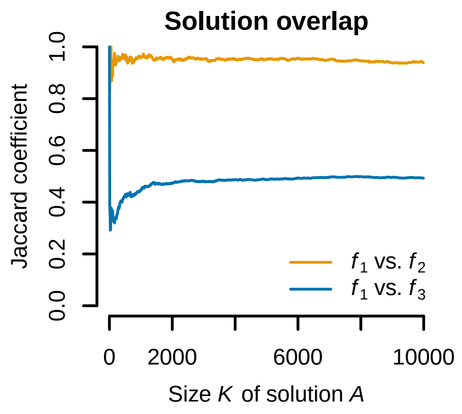
First, we consider how similar the solutions produced by the different objective functions are. For this purpose, Fig. 7.13(a)) plots the Jaccard coefficients between the solutions of $f_1$ and $f_2$ (orange) and between the solutions of $f_1$ and $f_3$ (blue) as functions of the solution size $K$. We observe that $f_1$ and $f_2$ produce nearly identical solutions. The reason is that clickthrough rates $p_{st}$ are generally very small, which entails that Eq. 7.6 and 7.7 become very similar (formally, this can be shown by a Taylor expansion around 0). Objective $f_3$, on the contrary, yields rather different solutions; it tends to favor source pages with fewer pre-existing high-clickthrough links (due to the second term in the denominator of Eq. 7.8; Fig. 7.13(b)) and attempts to spread links more evenly over all source pages (Fig. 7.13(c)). In other words, $f_3$ offers a stronger diminishing-returns effect: the marginal value of more links decays faster when some good links have already been added to the same source page.

Next, we aim to assess the impact of our suggestions on real Wikipedia users. Recall that we suggest links based on server logs from January 2015. We quantify the impact of our suggestions in terms of the total number of clicks they received in March 2015 (contributed by links added by editors after January). Fig. 7.13(d), which plots the total March click volume as a function of the solution size $K$, shows that (according to $f_3$) the top 1,000 links received 95,000 clicks in total in March 2015, and the top 10,000 received 336,000 clicks. Recalling that two-thirds of all links added by humans receive no click at all (Fig. 7.1), this clearly demonstrates the importance of the links we suggest. Fig. 7.13(e) displays the average number of clicks per suggested link as a function of $K$. The decreasing shape of the curves implies that higher-ranked suggestions received more clicks, as desired. Finally, comparing the three objectives, we observe that $f_3$ fares best on the Wikipedia dataset: its suggestions attract most clicks (Fig. 7.13(d) and 7.13(e)), while also being spread more evenly across source pages (Fig. 7.13(c)).

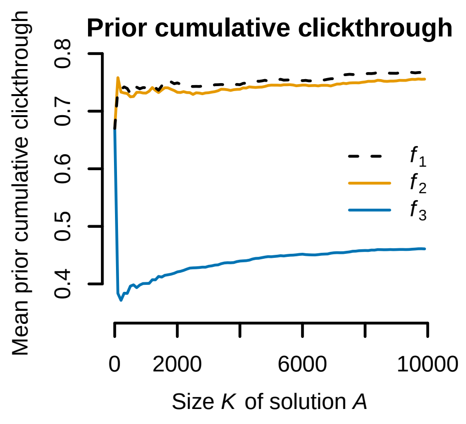
For comparison, the around 800,000 links added by independent human editors organically during the month of February 2015 received only 8 clicks during March 2015 on average. Our top 1,000 recommendations, on the contrary, received 95 clicks on average during March 2015; *i.e.*, the top 1,000 links recommended by our method are clicked about 12 times as frequently as the average human-added link.

These results show that the links we suggest fill a real user need. Our system recommends useful links before they would normally be added by editors, and it recommends additional links that we may assume would also be clicked frequently if they were added.

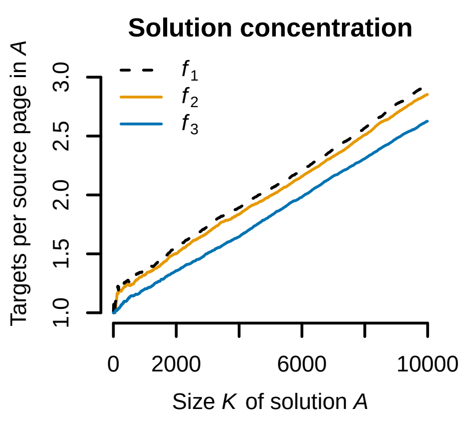
Table 7.4 illustrates our results by listing the top link suggestions.



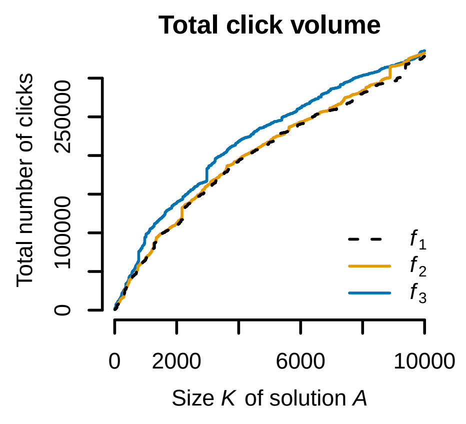
(a)



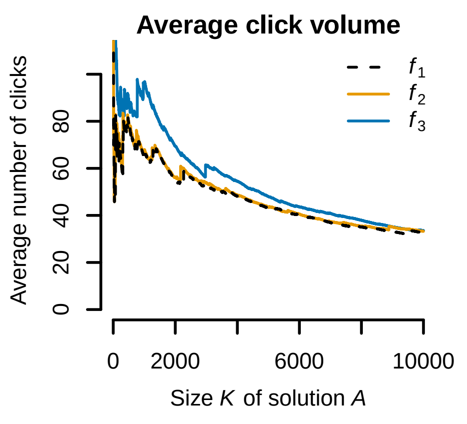
(b)



(c)



(d)



(e)

Figure 7.13: Results of budget-constrained link placement for objectives of Eq. 7.6–7.8; ‘prior cumulative clickthrough’ refers to second term in denominator of Eq. 7.8.

(a) Page-centric multi-tab objective $f_2$

| | Source | Target |
|---|---|---|
| * | ITunes Originals – Red Hot Chili Peppers | Road Trippin' Through Time |
| * | Internat. Handball Federation | 2015 World Men's Handball Champ. |
| | Baby, It's OK! | Olivia (singer) |
| | Payback (2014) | Royal Rumble (2015) |
| * | Nordend | Category:Mountains of the Alps |
| * | Gran Hotel | Grand Hotel (TV series) |
| * | Edmund Ironside | Edward the Elder |
| * | Jacob Aaron Estes | The Details (film) |
| | Confed. of African Football | 2015 Africa Cup of Nations |
| | Live Rare Remix Box | Road Trippin' Through Time |

(b) Single-tab objective $f_3$

| | Source | Target |
|---|---|---|
| * | ITunes Originals – Red Hot Chili Peppers | Road Trippin' Through Time |
| * | Gran Hotel | Grand Hotel (TV series) |
| * | Tithe: A Modern Faerie Tale | Ironside: A Modern Faery's Tale |
| * | Nordend | Category:Mountains of the Alps |
| * | Jacob Aaron Estes | The Details (film) |
| | Blame It on the Night | Blame (Calvin Harris song) |
| * | Internat. Handball Federation | 2015 World Men's Handball Champ. |
| * | The Choice (novel) | Benjamin Walker (actor) |
| | Payback (2014) | Royal Rumble (2015) |
| * | Just Dave Van Ronk | No Dirty Names |

Table 7.4: Top 10 link suggestions of Algorithm 2 using objectives **(a)** $f_2$ and **(b)** $f_3$. Clickthrough rates $p_{st}$ estimated via path proportion (Sec. 7.3.2). Objective $f_1$ (not shown) yields same result as $f_2$ but includes an additional link for source page *Baby, It's OK!*, demonstrating effect of diminishing returns on $f_2$. Asterisks mark links added by editors after prediction time, independently of our predictions.

| | Mean absolute err. | Pearson corr. | Spearman corr. |
|---|---|---|---|
| Path proportion | 0.020 ($\pm$0.003) | **0.41** ($\pm$0.10) | **0.50** ($\pm$0.09) |
| Mean baseline | **0.013** ($\pm$0.003) | $-$0.01 ($\pm$0.11) | $-$0.27 ($\pm$0.10) |

Table 7.5: Performance of path-proportion clickthrough estimation on Simtk, with bootstrapped 95% confidence intervals.

#### 7.3.4.2 Simtk

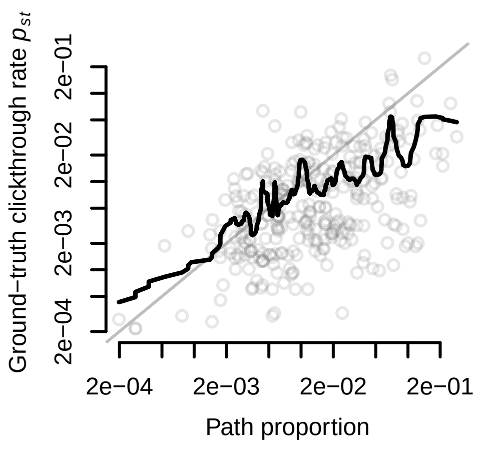
It is important to show that our method is general, as it relies on no features specific to Wikipedia. Therefore, we conclude our evaluation with a brief discussion of the results obtained on the second, smaller dataset from Simtk.

For evaluating our approach, we need to have a ground-truth set of new links alongside their addition dates. Unlike for Wikipedia, no complete revision history is available for Simtk, but we can nonetheless assemble a ground truth by exploiting a specific event in the site's history: after 6 months of log data, a sidebar with links to related pages was added to all pages. These links were determined by a recommender system once and did not change for 6 months. Our task is to predict these links' clickthrough rates $p_{st}$ for the 6 months after, based on log data from the 6 months before.

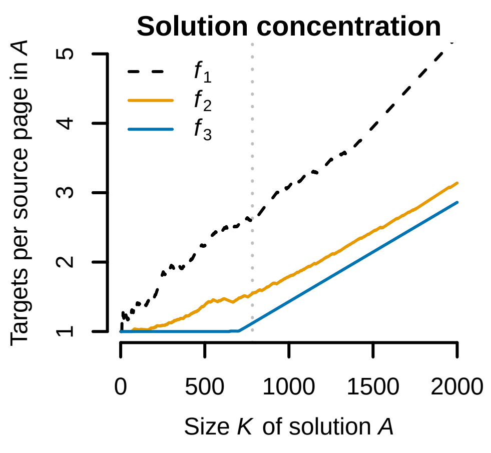
Since we consider navigation sessions (Sec. 3.1.3) rather than navigation trees in the case of Simtk, path proportions (Sec. 7.3.2) were computed by tallying up how often $s$ occurred before $t$ in the same session, rather than on a path in the same tree.

The results on the $p_{st}$ prediction task are summarized in Table 7.5; we compare path proportion to the mean baseline (*cf.* Sec. 7.3.4.1). At first glance, when considering only mean absolute error, it might seem as though the former were outperformed by the latter. But upon closer inspection, we realize that this is simply because the baseline predicts very small values throughout, which tend to be close to the ground-truth values, but cannot discriminate between good and bad candidates, as signified by the negative correlation coefficients. Path proportion, on the contrary, does reasonably well, at Pearson (Spearman) correlation coefficients of 0.41 (0.50). This correlation can be observed graphically in Fig. 7.14(a).

Last, we analyze the solutions of our budget-constrained link placement algorithm. Our first observation is that the two multi-tab objectives $f_1$ and $f_2$ differ much more on Simtk



(a)



(b)

Figure 7.14: Link placement results on Simtk. **(a)** Path proportion *vs.* clickthrough rate (black line corresponds to gray dots kernel-smoothed on linear scales). **(b)** Solution concentration of budget-constrained link placement; dotted vertical line: number of distinct pages.

than on Wikipedia (with Jaccard coefficients between 40% and 60%, compared to over 90%), which shows that the page-centric multi-tab objective $f_2$ is not redundant but adds a distinct option to the repertoire.

We also observe that the value of the single-tab objective $f_3$ saturates after inserting around as many links as there are pages (750). The reason is that only very few links were available and clicked before the related-link sidebar was added, so the second term in the denominator of Eq. 7.8 is typically much smaller than the first term and the numerator; with the first link added to $s$, the ratio in Eq. 7.8 becomes close to 1, and the contribution of $s$ to $f_3$ close to its possible maximum $w_s$. Thereafter, the objective experiences extreme diminishing returns for $s$. Fig. 7.14(b) confirms that $f_3$ starts by inserting one link into each of the around 750 pages.

We conclude that with few pre-existing links, the page-centric multi-tab objective $f_2$ might be better suited than the single-tab objective $f_3$, as it provides a more balanced trade-off between the number of links per source page and predicted clickthrough rates.

### 7.3.5 Discussion

As mentioned in Sec. 7.1, most prior link recommendation methods base their calculations on the static structure of the network, whereas we focus on how users interact with this static structure by browsing and searching on it. Since web server logs capture user needs and behavior in real time, a clear advantage of our log-based approach is the timeliness of the results it produces. This is illustrated by Table 7.4, where most suggestions refer to current events, films, music, books, *etc.*

Our random-walk model (Sec. 7.3.2) builds on a long tradition of Markovian web-navigation models (see Sec. 2.3 for an overview). Our model might be improved by using higher-order Markov chains, based on work by Chierichetti *et al.* [16], who found that human browsing is around 10% more predictable when considering longer contexts.

Finally, our method is universal in the sense that it uses only data logged by any commodity web server software, without requiring access to other data, such as the content of pages in the network. We demonstrate this generality by applying our method on two rather different websites without modification. While this is a strength of our approach, it would nonetheless be interesting to explore if the accuracy of our link clickthrough model could be improved by using machine learning to combine our current estimators, based on path counts, search counts, and random walks, with domain-specific elements such as the textual or hypertextual content of pages.

## 7.4 Conclusions

In this chapter we have studied the problem of identifying missing links on websites. We build on the fact that the ultimate purpose of Wikipedia links is to aid navigation. Our methods harness human navigation traces and find missing links that would immediately enhance Wikipedia's navigability. For this purpose, our algorithms first mine the click traces to detect a set of missing-link candidates and then rank these candidates by importance. We have developed methods that can deal with both targeted navigation traces (Sec. 7.2) and raw access logs collected by any standard web server software (Sec. 7.3).

In the present work we assume that frequent indirect paths between two pages indicate that a direct link would be useful. While this is confirmed by the data (Fig. 7.11(a)), research has shown that users sometimes deliberately choose indirect routes [95], which may offer additional value [113]. Detecting automatically in the logs when this is the case would be an interesting research problem.

A further route forward would be to extend our method to other types of networks that people navigate. For instance, citation networks, knowledge graphs, and some online social networks could all be amenable to our approach.

Finally, it would be worthwhile to explore how our methods could be extended to not only identify links inside a given website but to also links between different sites.

# Chapter 8

# Conclusions

## 8.1 Summary of contributions

There is no such thing as an isolated piece of knowledge. Bits of information are interconnected in giant networks, and we are daily facing the complex task of navigating and finding paths through such networks. The goal of this thesis has been to extend our understanding of how humans tackle this task and how we can turn the insights thus gained into practical models and tools for supporting human network navigation and for making our information networks more navigable.

Browsing the Web is an important example of network navigation, but by far not the only one: we follow leads in citation networks to find work that is related to our own research; we search social networks for people we know or people who might be able to provide help; when we reason about, or try to find explanations for, the phenomena around us, we are implicitly disentangling a network of relations between concepts, with the goal of finding a path of connections between the 'cause' and the 'effect'; and we constantly look up things in cross-referenced dictionaries and encyclopedias, be it in the form of books or online resources such as Wikipedia.

This last example is particularly interesting, since Wikipedia is not just a regular website but a rich network representing human knowledge as well as the connections between single pieces of knowledge, by means of hyperlinks. This distinguishes Wikipedia browsing from navigation on typical web resources. By observing humans finding their way between articles in Wikipedia, we are watching them navigate a large information network, using their mental maps of relationships in order to find the paths that connect concepts.

For these reasons, we chose Wikipedia navigation as the focus of this thesis. We leveraged data harvested through the human-computation game Wikispeedia [109], where participants solve the task of navigating between two Wikipedia articles by exclusively clicking article-to-article links, thereby describing trajectories on the underlying hyperlink network. The traces thus collected—over 50,000 in number—constitute a unique resource, as they are labeled with the exact navigation target the users attempted to reach.

In our in-depth analysis (Chapter 4), we observed that, although humans navigate based on local information only, without access to the complete adjacency structure of the underlying link graph, they are good at finding short paths between start and target nodes, needing in the median only one click more than the respective optimal solutions (4 *vs.* 3 clicks). We then continued to characterizing the navigation strategies that make humans so efficient and identified a set of reoccurring navigation patterns. For instance, users tend to be guided by expectations about the graph structure in the early navigation phase, jumping from the start article to a hub article of high out-degree. From there on, they tend to be guided by the conceptual, or semantic, relatedness of articles, homing in on the target node by navigating through nodes that become gradually more related to the target conceptually. In the process, they take large strides through concept space initially, but continually decrease their conceptual step size thereafter.

We then applied these lessons in models and tools that leverage features inspired by our analysis and that have the ability to support and facilitate human navigation. We addressed the tasks of predicting the user's navigation target after observing only a few initial clicks (Chapter 5), navigating toward target articles in Wikipedia automatically (Chapter 6), and improving website hyperlink structure by mining logs of navigation traces (Chapter 7).

The results are promising: our target prediction method achieves an accuracy of 80% on the task of guessing the correct one of two candidate targets; our automatic navigation algorithms can locate target articles in Wikipedia faster than humans on average; and the top 1,000 Wikipedia links suggested by our log-based link recommendation algorithm are clicked 12 times as frequently as the average human-added Wikipedia link.

## 8.2 Future directions

The results presented in this thesis point to several interesting future directions, some of which we shall sketch here, thus concluding the thesis.

### 8.2.1 Network structure versus navigability

An important way of extending our analysis of human navigation strategies would be to further our understanding of how the structure of the underlying network affects the efficiency with which users are able to locate their targets.

We saw that Wikipedia's hyperlink graph is structured in a way that is conducive to making local, greedy navigation strategies successful (Sec. 4.2), through its nearly perfect mix of short- and long-range semantic connections. In the future, it would be interesting to better understand exactly how crucial this semantic structure is for enabling efficient targeted navigation. For instance, we could design an experiment in which we add or remove certain links in a controlled fashion and where we subsequently observe how this affects human search performance.

Another approach would be to proceed observationally, by comparing how well participants manage to navigate to prescribed targets in different language editions of Wikipedia, which tend to differ not only with respect to the presentation of content, but also with respect to the connectivity of the hyperlink network.

A third approach would be to employ simulations, by training an automatic navigation agent to mimic human behavior (as in Chapter 6) and running it on different versions of the Wikipedia hyperlink network. By using different snapshots from Wikipedia's 15-year-long history, we could this way also answer the interesting question whether Wikipedia has become more or less navigable over time.

### 8.2.2 Navigation versus keyword search

The two principal modes of finding information on the Web are click-based navigation and query-based keyword search using search engines. This thesis focuses on the former, but incorporating the use of query-based search engines into our analysis would open up many intriguing avenues for future research. For instance, how do users mix navigating with query-based search? What are the tradeoffs? What is the optimal switching policy? Answering these questions could ultimately lead to hybrid search-and-browse engines to support human information seeking more naturally.

### 8.2.3 Further models and tools

Beyond the models and tools developed in Chapters 5–7, many more applications of navigation trace data are conceivable.

In Chapter 7 we have introduced algorithms for augmenting a website by inserting additional hyperlinks, with the goal of making the site more navigable. A proposed link from page $s$ to page $t$ can be inserted immediately only if $s$ already contains a phrase that could serve as an anchor text for a link to $t$. For instance, a link to $t =$ STANFORD UNIVERSITY could only be inserted into a page $s$ if the article text of $s$ contains a phrase such as 'Stanford', 'Stanford University', 'Leland Stanford Jr. University', *etc.* Imagine, however, that our algorithm suggests to add a link from $s =$ CONDOLEEZZA RICE, the former U.S. Secretary of State, to $t =$ STANFORD UNIVERSITY. Given that Rice teaches at Stanford, this would clearly be a useful link, although in several language editions of Wikipedia (*e.g.*, in Catalan) the article on Condoleezza Rice contains no mention of Stanford University. Here, our algorithm has in effect suggested a topic, rather than merely a link, that should be added to the article on Condoleezza Rice. Determining where in the text of $s$ a mention of $t$ should be inserted puts a high cognitive load on the human contributor aiming to make the edit, since it requires scanning the entire text of $s$. Thus, it will be interesting to devise methods that can leverage click logs to automatically determine the most appropriate positions for inserting a mention of $t$ in $s$.

There are also interesting challenges for structured machine learning. One intriguing question is whether, and how, we could use navigation traces in order to embed networks into metric spaces, *e.g.*, under the objective that empirically observed paths should correspond to shortest paths in the learned metric space. Such embeddings could be of help in visualizing and indexing networks, which would in turn make them better searchable by humans.

As yet another avenue for future research, we mention the problem of finding and filling knowledge gaps in online encyclopedias. Recent work has addressed this issue from a recommender system angle [120], and it would be interesting to incorporate features derived from navigation traces into such systems. For instance, assume that we frequently observe the sequence $\langle a,b,c,d\rangle$ in the server logs of a language edition $L_1$ of Wikipedia. If in the logs of another language edition $L_2$ we frequently see the sequence $\langle a,b,c\rangle$, but language $L_2$ has no article on $d$, then this indicates that an important knowledge gap could be plugged by creating an article on $d$ in language $L_2$.

### 8.2.4 Navigation in the wild

The bulk of this thesis has considered targeted navigation traces, *i.e.*, traces during which the user intended to reach a specific target node. In the latter part of Chapter 7 (Sec. 7.3), we have abandoned this requirement by working with navigation traces extracted from passively collected web server logs, which may capture any of a wide variety of browsing modes. For instance, some Wikipedia users have specific targets (such as definitions or concise facts) in mind, others want to take a deep dive into a topic, and yet others are aimlessly exploring the encyclopedia by drifting from page to page.

In future work, we will aim to better understand the full spectrum of user intentions on Wikipedia—an important question, given that Wikipedia is among the world's top ten most popular websites. We have started by designing and deploying a survey on live Wikipedia to elicit readers' information needs. The responses thus collected can be associated with those readers' navigation traces extracted from the server logs, which will ultimately lead to a better understanding of how different information needs and browsing modes manifest themselves in trace data, and to statistical models that will allow us to predict a user's information need after seeing only their first few actions on the site. This, in turn, would let us adaptively change the user interface on the fly, with the goal of catering to the user's information need more effectively and efficiently.

We have demonstrated the applicability of some of our techniques to websites beyond Wikipedia; *e.g.*, we have applied our link recommendation method to Simtk.org

(Sec. 7.3.4.2). Future research should investigate to what extent users' navigation behaviors differ across different types of websites. Moreover, as mentioned in the introduction to this conclusion, network navigation is not constrained to websites, but encompasses citation networks, social networks, semantic networks, and many other types. Therefore, a fertile avenue of future research would be to broaden the scope and aim to understand the commonalities and differences of these different kinds of network with respect to the human navigation behaviors to which they give rise, and to build models and tools to facilitate these behaviors.

---

Turning the trace this thesis has taken into a cycle, we revisit Vannevar Bush's 1945 paper *As We May Think*. There, musing about the bright future that would be brought about by intuitively navigable information systems, Bush prophesied that our societies would see the advent of "a new profession of trail blazers, those who find delight in the task of establishing useful trails through the enormous mass of the common record. The inheritance from the master becomes, not only his additions to the world's record, but for his disciples the entire scaffolding by which they were erected." [13]

Let us thus aspire to equip those coming trail blazers with tools and torches to illuminate their paths. And let us strive to create algorithmic trail blazers that can collect, digest, curate, and present in meaningful ways the navigation traces that people leave behind in the logs of servers by the billion every day, so we may all benefit from the paths trodden by each other.

# Appendix A

# Human-rater instructions on Amazon Mechanical Turk

When using Amazon Mechanical Turk to evaluate (Sec. 7.2.5.2) the link recommendation made by our method of Sec. 7.2, the following task description was given to human raters:

*"Here's the deal! Our good friend Wikipedia is having self-doubts and wants you to help improve its links.*

*You are given a Wikipedia article (referred to as the target) and a list of other Wikipedia articles (referred to as source articles). You have to tell Wikipedia if the source article should contain a link to the target. And of course, if you are unsure of what the source or target article means, you can always click on the article name to open it in a new tab.*

*But remember that Wikipedia is a sensitive fellow and will be mad if you don't play by the rules: There should be a link from the source to the target if and only if (1) the target article has some relevant information about the source article and could help readers understand the source article more fully, or (2) the target article describes a proper name which is likely to be unfamiliar to readers."*

# Bibliography

[1] L. Adamic and E. Adar. Friends and neighbors on the Web. *Social Networks*, 25(3):211–230, 2003.

[2] L. Adamic and E. Adar. How to search a social network. *Social Networks*, 27(3):187–203, 2005.

[3] L. A. Adamic, R. M. Lukose, A. R. Puniyani, and B. A. Huberman. Search in power-law networks. *Physical Review E*, 64(4):046135, 2001.

[4] E. Adar, J. Teevan, and S. T. Dumais. Large scale analysis of Web revisitation patterns. In *Proceedings of the Annual SIGCHI Conference on Human Factors in Computing Systems*, 2008.

[5] M. Ageev, Q. Guo, D. Lagun, and E. Agichtein. Find it if you can: A game for modeling different types of Web search success using interaction data. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2011.

[6] Amazon. Mechanical Turk. Website, 2016. `http://www.mturk.com` (accessed Apr. 29, 2016).

[7] L. Backstrom and J. Leskovec. Supervised random walks: Predicting and recommending links in social networks. In *Proceedings of the International ACM Conference on Web Search and Data Mining*, 2011.

[8] M. J. Bates. The design of browsing and berrypicking techniques for the online search interface. *Online Review*, 13(5):407–424, 1989.

[9] T. J. Berners-Lee. Information management: A proposal. Technical report, CERN, 1989.

[10] M. Bilenko and R. W. White. Mining the search trails of surfing crowds: Identifying relevant websites from user activity. In *Proceedings of the International Conference on the World Wide Web*, 2008.

[11] J. Borges and M. Levene. Evaluating variable-length Markov chain models for analysis of user Web navigation sessions. *IEEE Transactions on Knowledge and Data Engineering*, 19(4):441–452, 2007.

[12] S. Brin and L. Page. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems*, 30(1):107–117, 1998.

[13] V. Bush. As we may think. *The Atlantic Monthly*, 176(1):101–108, 1945.

[14] R. Cailliau and H. Ashman. Hypertext in the web: A history. *ACM Computing Surveys*, 31(4), 1999.

[15] E. H. Chi, P. Pirolli, K. Chen, and J. Pitkow. Using information scent to model user information needs and actions and the Web. In *Proceedings of the Annual SIGCHI Conference on Human Factors in Computing Systems*, 2001.

[16] F. Chierichetti, R. Kumar, P. Raghavan, and T. Sarlos. Are Web users really Markovian? In *Proceedings of the International Conference on the World Wide Web*, 2012.

[17] A. Clauset, C. Moore, and M. E. J. Newman. Hierarchical structure and the prediction of missing links in networks. *Nature*, 453(7191):98–101, 2008.

[18] A. Clemesha. The Wiki Game. Website, 2009. `http://www.thewikigame.com` (accessed Nov. 10, 2014).

[19] A. Clemesha. Personal communication, Jan. 14, 2014.

[20] A. Dallmann, T. Niebler, F. Lemmerich, and A. Hotho. Extracting semantics from random walks on Wikipedia: Comparing learning and counting methods. In *Proceedings of the Wiki Workshop at ICWSM*, 2016.

[21] B. D. Davison. Learning Web request patterns. In *Web Dynamics*. Springer, 2004.

[22] M. Deshpande and G. Karypis. Selective Markov models for predicting Web page accesses. *ACM Transactions on Internet Technology*, 4(2):163–184, 2004.

[23] P. Devanbu, Y.-F. Chen, E. Gansner, H. Müller, and J. Martin. CHIME: Customizable hyperlink insertion and maintenance engine for software engineering environments. In *Proceedings of the International Conference on Software Engineering*, 1999.

[24] D. Diderot. The definition of an encyclopedia. In J. Boyer and K. Baker, editors, *Readings in Western Civilization, Volume 7: The Old Regime and the French Revolution*. University of Chicago Press, 1987.

[25] P. S. Dodds, R. Muhamad, and D. J. Watts. An experimental study of search in global social networks. *Science*, 301(5634):827–829, 2003.

[26] D. Downey, S. Dumais, D. Liebling, and E. Horvitz. Understanding the relationship between searchers' queries and information goals. In *Proceedings of the ACM Conference on Information and Knowledge Management*, 2008.

[27] D. Downey, S. T. Dumais, and E. Horvitz. Models of searching and browsing: Languages, studies, and application. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 2007.

[28] J. Edmonds. Matroids and the greedy algorithm. *Mathematical Programming*, 1(1):127–136, 1971.

[29] D. C. Engelbart. Augmenting human intellect: A conceptual framework. Technical report, Stanford Research Institute, 1962.

[30] M. J. Eppler and J. Mengis. The concept of information overload: A review of literature from organization science, accounting, marketing, MIS, and related disciplines. *The Information Society*, 20(5):325–344, 2004.

[31] S. Fissaha Adafre and M. de Rijke. Discovering missing links in Wikipedia. In *Proceedings of the International KDD Workshop on Link Discovery*, 2005.

[32] W.-T. Fu and P. Pirolli. SNIF-ACT: A cognitive model of user navigation on the World Wide Web. *Human–Computer Interaction*, 22(4):355–412, 2007.

[33] M. Grecu. Navigability in information networks. Master's thesis, ETH Zürich, 2014.

[34] A. Halfaker, O. Keyes, D. Kluver, J. Thebault-Spieker, T. Nguyen, K. Shores, A. Uduwage, and M. Warncke-Wang. User session identification based on strong regularities in inter-activity time. In *Proceedings of the International Conference on the World Wide Web*, 2015.

[35] B. He, M. Patel, Z. Zhang, and K. C.-C. Chang. Accessing the deep Web. *Communications of the ACM*, 50(5):94–101, 2007.

[36] D. Helic, M. Strohmaier, M. Granitzer, and R. Scherer. Models of human navigation in information networks based on decentralized search. In *ACM Conference on Hypertext and Social Media*, 2013.

[37] D. Helic, M. Strohmaier, C. Trattner, M. Muhr, and K. Lerman. Pragmatic evaluation of folksonomies. In *Proceedings of the International Conference on the World Wide Web*, 2011.

[38] K. Henderson and T. Eliassi-Rad. Applying latent Dirichlet allocation to group discovery in large graphs. In *Proceedings of the ACM SIGAPP Symposium on Applied Computing*, 2009.

[39] A. Herdagdelen and M. Baroni. The concept game: Better commonsense knowledge extraction by combining text mining and a game with a purpose. In *Proceedings of the AAAI Fall Symposium on Commonsense Knowledge*, 2010.

[40] T. Joachims, D. Freitag, and T. Mitchell. WebWatcher: A tour guide for the World Wide Web. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 1997.

[41] D. Jurgens and R. Navigli. It's all fun and games until someone annotates: Video games with a purpose for linguistic annotation. *Transactions of the Association for Computational Linguistics*, 2:449–464, 2014.

[42] I. Kaur and A. J. Hornof. A comparison of LSA, WordNet and PMI-IR for predicting user click behavior. In *Proceedings of the Annual SIGCHI Conference on Human Factors in Computing Systems*, 2005.

[43] R. M. Keller, S. R. Wolfe, J. R. Chen, J. L. Rabinowitz, and N. Mathe. A bookmarking service for organizing and sharing URLs. In *Proceedings of the International Conference on the World Wide Web*, 1997.

[44] P. Killworth, C. McCarty, H. Bernard, and M. House. The accuracy of small world chains in social networks. *Social Networks*, 28(1):85–96, 2006.

[45] M. Kim and J. Leskovec. The network completion problem: Inferring missing nodes and edges in networks. In *Proceedings of the SIAM International Conference on Data Mining*, 2011.

[46] J. Kleinberg. Navigation in a small world. *Nature*, 406(6798):845–845, 2000.

[47] J. Kleinberg. The small-world phenomenon: An algorithmic perspective. In *Proceedings of the Annual ACM Symposium on Theory of Computing*, 2000.

[48] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.

[49] D. Lamprecht, D. Helic, and M. Strohmaier. Quo vadis? On the effects of Wikipedia's policies on navigation. In *Proceedings of the ICWSM Workshop on Wikipedia as a Social Pedia: Research Challenges and Opportunities*, 2015.

[50] T. Landauer and S. T. Dumais. A solution to Plato's problem: The Latent Semantic Analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review*, 104(2):211–240, 1997.

[51] K. Lerman and L. A. Jones. Social browsing on Flickr. In *Proceedings of the International Conference on Weblogs and Social Media*, 2007.

[52] J. Leskovec and E. Horvitz. Planetary-scale views on a large instant-messaging network. In *Proceedings of the International Conference on the World Wide Web*, 2008.

[53] J. Leskovec and E. Horvitz. Geospatial structure of a planetary-scale social network. *IEEE Transactions on Computational Social Systems*, 1(3):156–163, 2014.

[54] B. Leuf and W. Cunningham. *The Wiki Way: Quick Collaboration on the Web*. Addison–Wesley, 2001.

[55] Z. Li and J. Tian. Testing the suitability of Markov chains as Web usage models. In *Proceedings of the Annual International Computer Software and Applications Conference*, 2003.

[56] D. Liben-Nowell and J. Kleinberg. The link-prediction problem for social networks. *Journal of the Association for Information Science and Technology*, 58(7):1019–1031, 2007.

[57] D. Liben-Nowell, J. Novak, R. Kumar, P. Raghavan, and A. Tomkins. Geographic routing in social networks. *Proceedings of the National Academy of Sciences*, 102(33):11623–11628, 2005.

[58] R. N. Lichtenwalter, J. T. Lussier, and N. V. Chawla. New perspectives and methods in link prediction. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2010.

[59] H. Ma, R. Chandrasekar, C. Quirk, and A. Gupta. Page hunt: Improving search engines using human computation games. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2009.

[60] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[61] G. Marchionini. Exploratory search: From finding to understanding. *Communications of the ACM*, 49(4):41–46, 2006.

[62] C. C. Marshal and A. J. B. Brush. Exploring the relationship between personal and public annotations. In *Proceedings of the Joint ACM/IEEE Conference on Digital Libraries*, 2004.

[63] E. Meij, W. Weerkamp, and M. de Rijke. Adding semantics to microblog posts. In *Proceedings of the International ACM Conference on Web Search and Data Mining*, 2012.

[64] R. Mihalcea and A. Csomai. Wikify! Linking documents to encyclopedic knowledge. In *Proceedings of the ACM Conference on Information and Knowledge Management*, 2007.

[65] S. Milgram. The small world problem. *Psychology Today*, 2(1):60–67, 1967.

[66] G. A. Miller. Informavores. In *The study of information: Interdisciplinary messages*, pages 111–113. Wiley–Interscience, 1984.

[67] D. Milne and I. H. Witten. An effective, low-cost measure of semantic relatedness obtained from Wikipedia links. In *Proceedings of the AAAI Workshop on Wikipedia and Artificial Intelligence*, 2008.

[68] D. Milne and I. H. Witten. Learning to link with Wikipedia. In *Proceedings of the ACM Conference on Information and Knowledge Management*, 2008.

[69] D. Milne and I. H. Witten. An open-source toolkit for mining Wikipedia. *Artificial Intelligence*, 194:222–239, 2013.

[70] J. Muramatsu and W. Pratt. Transparent queries: investigation users' mental models of search engines. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2001.

[71] T. H. Nelson. Complex information processing: A file structure for the complex, the changing and the indeterminate. In *Proceedings of the ACM National Conference*, 1965.

[72] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. An analysis of approximations for maximizing submodular set functions – I. *Mathematical Programming*, 14(1):265–294, 1978.

[73] C. Nentwich, L. Capra, W. Emmerich, and A. Finkelstein. Xlinkit: A consistency checking and smart link generation service. *ACM Transactions on Internet Technology*, 2(2):151–185, 2002.

[74] T. Noraset, C. Bhagavatula, and D. Downey. Adding high-precision links to Wikipedia. In *Proceedings of the Conference on Empirical Methods on Natural Language Processing*, 2014.

[75] V. L. O'Day and R. Jeffries. Orienteering in an information landscape: how information seekers get from here to there. In *Proceeding of the Annual SIGCHI Conference on Human Factors in Computing Systems*, 1993.

[76] C. Olston and E. H. Chi. ScentTrails: Integrating browsing and searching on the Web. *ACM Transactions on Computer–Human Interaction*, 10(3):177–197, 2003.

[77] A. Paranjape, R. West, and L. Zia. Project website, 2015. `https://meta.wikimedia.org/wiki/Research:Improving_link_coverage` (accessed Dec. 10, 2015).

[78] A. Paranjape, R. West, L. Zia, and J. Leskovec. Improving website hyperlink structure using server logs. *Proceedings of the International ACM Conference on Web Search and Data Mining*, 2016.

[79] P. Pirolli. Rational analyses of information foraging on the Web. *Cognitive Science*, 29(3):343–373, 2005.

[80] P. Pirolli. *Information foraging theory: Adaptive interaction with information*. Oxford University Press, 2007.

[81] P. Pirolli and S. K. Card. Information foraging. *Psychological Review*, 106(4):643–675, 1999.

[82] A. Popescul, R. Popescul, and L. H. Ungar. Statistical relational learning for link prediction. In *Proceedings of the IJCAI Workshop on Learning Statistical Models from Relational Data*, 2003.

[83] G. A. Rummery and M. Niranjan. On-line Q-learning using connectionist systems. Technical report, University of Cambridge, 1994.

[84] R. R. Sarukkai. Link prediction and path analysis using Markov chains. *Computer Networks*, 33(1):377–386, 2000.

[85] A. T. Scaria, R. M. Philip, R. West, and J. Leskovec. The last click: Why users give up information network navigation. In *Proceedings of the International ACM Conference on Web Search and Data Mining*, 2014.

[86] A. J. Sellen, R. Murphy, and K. L. Shaw. How knowledge workers use the web. In *Proceedings of the Annual SIGCHI Conference on Human Factors in Computing Systems*, 2002.

[87] R. Sen and M. H. Hansen. Predicting Web users' next access based on log data. *Journal of Computational and Graphical Statistics*, 12(1):143–155, 2003.

[88] Ö. Şimşek and D. Jensen. Decentralized search in networks using homophily and degree disparity. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 2005.

[89] Ö. Şimşek and D. Jensen. Navigating networks by using homophily and degree. *Proceedings of the National Academy of Sciences*, 105(35):12758–12762, 2008.

[90] P. Singer, D. Helic, A. Hotho, and M. Strohmaier. HypTrails: A Bayesian approach for comparing hypotheses about human trails on the Web. In *Proceedings of the International Conference on the World Wide Web*, 2015.

[91] P. Singer, T. Niebler, M. Strohmaier, and A. Hotho. Computing semantic relatedness from human navigational paths on Wikipedia. In *Companion to the International Conference on the World Wide Web*, 2013.

[92] A. Singla, R. White, and J. Huang. Studying trailfinding algorithms for enhanced web search. In *Proceeding of the International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2010.

[93] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. Cambridge University Press, 1998.

[94] B. Taskar, M. F. Wong, P. Abbeel, and D. Koller. Link prediction in relational data. In *Proceedings of the Annual Conference on Neural Information Processing Systems*, 2003.

[95] J. Teevan, C. Alvarado, M. S. Ackerman, and D. R. Karger. The perfect search engine is not enough: a study of orienteering behavior in directed search. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2004.

[96] S. Thaler, K. Siorpaes, E. Simperl, and C. Hofer. A survey on games for knowledge acquisition. Technical report, STI Innsbruck, 2011.

[97] A. Toffler. *Future Shock*. Random House, 1970.

[98] C. Trattner, D. Helic, P. Singer, and M. Strohmaier. Exploring the differences and similarities between hierarchical decentralized search and human navigation in information networks. In *Proceedings of the International Conference on Knowledge Technologies and Data-driven Business*, 2012.

[99] R. H. Trigg. Guided tours and tabletops: Tools for communicating in a hypertext environment. *ACM Transactions on Information Systems*, 6(4):398–414, 1988.

[100] D. Vannella, D. Jurgens, D. Scarfini, D. Toscani, and R. Navigli. Validating and extending semantic knowledge bases using video games with a purpose. In *Proceedings of the Annual Meeting for the Association for Computational Linguistics*, 2014.

[101] L. von Ahn and L. Dabbish. Labeling images with a computer game. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2004.

[102] L. von Ahn and L. Dabbish. Designing games with a purpose. *Communications of the ACM*, 51(8):58–67, 2008.

[103] D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393(6684):440–442, 1998.

[104] R. West. Wikispeedia. Website, 2009. `http://www.wikispeedia.net` (accessed Apr. 29, 2016).

[105] R. West. Wikispeedia project website, 2014. `http://snap.stanford.edu/data/wikispeedia.html` (accessed Apr. 29, 2016).

[106] R. West and J. Leskovec. Automatic versus human navigation in information networks. In *Proceedings of the International Conference on Weblogs and Social Media*, 2012.

[107] R. West and J. Leskovec. Human wayfinding in information networks. In *Proceedings of the International Conference on the World Wide Web*, 2012.

[108] R. West, A. Paranjape, and J. Leskovec. Mining missing hyperlinks from human navigation traces: A case study of Wikipedia. In *Proceedings of the International Conference on the World Wide Web*, 2015.

[109] R. West, J. Pineau, and D. Precup. Wikispeedia: An online game for inferring semantic distances between concepts. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 2009.

[110] R. West, D. Precup, and J. Pineau. Completing Wikipedia's hyperlink structure through dimensionality reduction. In *Proceedings of the ACM Conference on Information and Knowledge Management*, 2009.

[111] R. West, D. Precup, and J. Pineau. Automatically suggesting topics for augmenting text documents. In *Proceedings of the ACM Conference on Information and Knowledge Management*, 2010.

[112] J. Weston, S. Bengio, and N. Usunier. Large scale image annotation: Learning to rank with joint word-image embeddings. *Machine Learning*, 81(1):21–35, 2010.

[113] R. W. White and J. Huang. Assessing the scenic route: Measuring the value of search trails in Web logs. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2010.

[114] R. W. White and A. Singla. Finding our way on the Web: Exploring the role of waypoints in search interaction. In *Proceedings of the International Conference on the World Wide Web*, 2011.

[115] Wikipedia. 2007 Wikipedia Selection for schools. Website, 2007. `http://schools-wikipedia.org` (accessed Aug. 3, 2008).

[116] Wikipedia. Wikipedia:Linking. Website, 2010. `http://en.wikipedia.org/w/index.php?title=Wikipedia:Linking&oldid=342061829` (accessed Feb. 5, 2010).

[117] Wikipedia. Wikiracing. Website, 2014. `http://en.wikipedia.org/w/index.php?title=Wikiracing&oldid=630702489` (accessed Nov. 10, 2014).

[118] F. Wu and D. S. Weld. Autonomously semantifying Wikipedia. In *Proceedings of the ACM Conference on Information and Knowledge Management*, 2007.

[119] E. Wulczyn and D. Taraborelli. Wikipedia Clickstream. Website, 2015. `http://dx.doi.org/10.6084/m9.figshare.1305770` (accessed July 16, 2015).

[120] E. Wulczyn, R. West, L. Zia, and J. Leskovec. Growing Wikipedia across languages via recommendation. In *Proceedings of the International Conference on the World Wide Web*, 2016.

[121] Y. Yue, T. Finley, F. Radlinski, and T. Joachims. A support vector method for optimizing average precision. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2007.

[122] E. Zachte. Wikipedia statistics. Website, 2016. `https://stats.wikimedia.org/EN/TablesArticlesNewPerDay.htm` (accessed Apr. 24, 2016).

[123] I. Zukerman, D. W. Albrecht, and A. E. Nicholson. Predicting users' requests on the WWW. In *Proceedings of the International Conference on User Modeling*, 1999.