Vorabübertragung schwach strukturierter Informationen in ortsbasierten mobilen Systemen

Von der Fakultät Informatik, Elektrotechnik und Informationstechnik der Universität Stuttgart zur Erlangung der Würde eines Doktors der Naturwissenschaften (Dr. rer. nat.) genehmigte Abhandlung

vorgelegt von

Susanne Gudrun Bürklen

aus Merklingen

Hauptberichter:	Prof. Dr.-Ing. habil. Bernhard Mitschang
1. Mitberichter:	Prof. Dr. rer. nat. Dr. h. c. Kurt Rothermel
2. Mitberichter:	Prof. Dr.-Ing. Wolfgang Effelsberg

Tag der mündlichen Prüfung: 27.08.2007

Institut für Parallele und Verteilte Systeme (IPVS)
der Universität Stuttgart
2007

Meinen Söhnen Andreas und Thomas

Danksagung

Diese Dissertation ist am Institut für Parallele und Verteilte Systeme der Universität Stuttgart im Rahmen des Sonderforschungsbereichs 627 mit dem Titel „nexus - Umgebungsmodelle für mobile kontextbezogene Systeme" entstanden. Herzlichen Dank an Prof. Mitschang für die Übernahme des Hauptberichts sowie an Prof. Rothermel und Prof. Effelsberg für die Übernahme des Mitberichts. Sie haben mir einerseits den notwendigen Freiraum gelassen und andererseits in den verschiedenen Entstehungsphasen dieser Arbeit durch anregende und kritische Diskussionen wichtige inhaltliche Impulse gegeben.

Bei meinen Kollegen der Abteilung Verteilte Systeme und den Mitarbeitern des Sonderforschungsbereichs bedanke ich mich für die Zusammenarbeit in einer wirklich einmaligen Atmosphäre. Trotz ihrer eigenen zeitlichen Belastung nahmen sie sich stets Zeit für intensive und wertvolle Diskussionen zu meiner Arbeit. Herzlichen Dank auch an Ralph Lange, der die Arbeit Korrektur gelesen hat.

Natürlich wird eine wissenschaftliche Arbeit durch die Unterstützung von Studierenden wesentlich erleichtert. Hiermit bedanke ich mich bei Serena Fritsch, die bei der Implementierung der Simulationsumgebung geholfen hat und bei Timo Pfahl, der im Rahmen seiner Diplomarbeit an der Entwicklung des Verfahrens zur Clusterbildung beteiligt war.

Schließlich bedanke ich mich bei der Deutschen Forschungsgemeinschaft für die Ermöglichung meines Forschungsvorhabens.

Nicht zuletzt gilt mein ganz besonderer Dank meinen beiden Söhnen Andreas und Thomas für ihr Verständnis und ihre Geduld in den oft schwierigen Phasen.

Nachtrag Die in dieser Dissertation vorkommenden Bezeichnungen, Texte und Formulierungen sind lediglich aus praktischen Gründen in der männlichen Form verfasst, sie gelten für die Angehörigen beiderlei Geschlechts in gleicher Weise.

Inhaltsverzeichnis

Tabellenverzeichnis

Abbildungsverzeichnis

Kurzfassung

Die rasant fortschreitende Entwicklung der Mobilkommunikation in Verbindung mit immer leistungsfähigeren mobilen Endgeräten weckt den Wunsch, an jedem Ort und zu jeder Zeit auf entfernte Informationen zugreifen zu können. Drahtlose Weitverkehrsnetze wie die Mobilfunknetze der zweiten oder dritten Generation bieten zwar nahezu überall eine Netzverbindung, weisen jedoch negative Eigenschaften wie eine hohe Latenz, hohe monetäre Kosten und unzuverlässige Verbindungen auf, die teilweise zum entkoppelten Betrieb führen können. Eine hohe Latenz führt dazu, dass auf Grund der hieraus entstehenden langen Übertragungszeiten der Informationen der Energieverbrauch der Funkschnittstelle ansteigt, was in Anbetracht der geringen Energieressourcen mobiler Endgeräte nicht wünschenswert ist. Um diesen Nachteilen entgegen zu wirken, wurden zugriffsoptimierende Methoden wie beispielsweise Caching oder die Vorabübertragung entwickelt, die jedoch unterschiedlichen Zielsetzungen folgen.

In dieser Dissertation wird zur Optimierung des mobilen Informationszugriffs ein generisches Verfahren zur Vorabübertragung von beliebigen schwach strukturierten Informationen in ortsbasierten Anwendungen vorgestellt, das neben einer Verringerung der Latenz den entkoppelten Betrieb unterstützt. Für die Selektion der vorab zu übertragenden Informationen werden je nach Art der Informationen unterschiedliche Methoden angeboten. Als Basis wird eine Infrastruktur von so genannten Infostationen benötigt, an denen mobilen Benutzern mittels drahtloser lokaler Netze ein breitbandiger und kostengünstiger Zugriff auf Informationen ermöglicht wird. Sie sind in Gebieten verteilt, an denen sonst keine oder nur eine Kommunikation mit der maximalen Datenrate eines drahtlosen Weitverkehrsnetzes, wie beispielsweise GSM, GPRS oder UMTS möglich ist. Eine Infostation selektiert die in ihrem Dienstgebiet für einen Benutzer relevanten Informationen und überträgt sie vorab auf dessen mobiles Endgerät. Zukünftige Informationsanfragen können somit lokal aus dem Cache beantwortet werden, was jedoch eine gute Vorhersage voraussetzt.

Um eine hohe Relevanz der vorab geladenen Informationen zu erreichen, werden

als Selektionskriterium neben der Ortsabhängigkeit von Informationszugriffen auch Beziehungen zwischen den Informationen ausgewertet. Eine Infostation beobachtet das typische Zugriffsverhalten aller Benutzer, die sich in ihrem Dienstgebiet aufhalten und benutzt dieses Wissen zur Vorhersage der Informationen. Das Beobachten des kollektiven Zugriffsverhaltens hat den Vorteil, dass überwiegend diejenigen Informationen vorab geladen werden, die in einem Dienstgebiet zum aktuellen Zeitpunkt populär sind. Dieses Verfahren unterstützt somit auch solche Benutzer, die sich zum ersten Mal in diesem Gebiet aufhalten. Bisweilen können Informationen derart stark zusammenhängen, dass sie für einen Benutzer nur als Gruppe interessant sind und es keinen Sinn ergibt, einzelne Objekte dieser Gruppe isoliert von den anderen zu übertragen. In einem solchen Fall müssen Gruppen (Cluster) gebildet werden, die vollständig vorab übertragen werden. Des Weiteren resultiert die Auswertung des Zugriffsverhaltens einer Menge von Benutzern nicht immer in einer optimalen Entscheidung für einen individuellen Benutzer. Dieser Effekt kann jedoch verringert werden, wenn Nutzungsprofile in die Übertragungsentscheidung mit einbezogen werden.

Das generische Vorabübertragungsverfahren wurde für den mobilen Zugriff auf das Web spezialisiert und evaluiert. Zur systematischen Leistungsbewertung wurde ein Modell für das Navigationsverhalten von Benutzern im Web entwickelt und implementiert, das Sequenzen von synthetischen Zugriffen auf das Web erzeugt. Mit dem clusterbasierten Auswahlverfahren konnten Trefferraten erzielt werden, die andere Ansätze um mehr als das Dreifache übertreffen.

Schließlich ist der Energiebedarf der Funkschnittstelle ein nicht zu vernachlässigender Faktor für die Lebensdauer der Batterie, so dass die Zeit zum Senden und Empfangen von Daten möglichst gering gehalten werden sollte. In einem drahtlosen lokalen Netz steht eine um mindestens eine Größenordnung höhere Bandbreite als in drahtlosen Weitverkehrsnetzen zur Verfügung, wodurch die Übertragungszeit von Informationen deutlich verkürzt wird. Eine Analyse des Leistungsbedarfs von Funkschnittstellen beider Technologien hat gezeigt, dass durch den Einsatz des vorgestellten Verfahrens zur Vorabübertragung von Informationen in jedem Fall Energieeinsparungen möglich sind. Bei Kenntnis der zu erzielenden Trefferrate kann somit die Größe des Caches bestimmt werden, die den Energieverbrauch beim Laden der Informationen minimiert.

Abstract

Introduction

Looking at current trends in communication patterns, it seems clear that there is an increasing demand for information anytime and anywhere. Mobile users want to be able to access data independently of where they are and where this data is located. In particular, they want to have mobile access to the World Wide Web, which is the major source of information nowadays and the basis of many applications.

Wireless communication technology is a prerequisite for mobile data access. Wireless WANs, such as 2G and 3G cellular networks, provide wireless connectivity virtually everywhere. However, even 3G networks only provide moderate data rates up to a few Mbps, typically 300 to 600Kbps. Moreover, communication channels may be subject to substantial error rates, high latency and frequent disconnections. A complementary technology is based on wireless LAN (WLAN) hotspots, which are isolated "islands" of connectivity. Within a limited range, they provide comparatively cheap or sometimes even free high-speed Internet access and at least an order of magnitude higher data rates than wireless WANs.

The two technologies, WLAN hotspots and wireless WANs can be combined to optimize mobile data access. Mobile devices can use the available connectivity at a given hotspot to prefetch and locally cache data their users might access in future. This approach, called hoarding, aims at minimizing wireless WAN usage in order to decrease cost and access latency, in particular for larger data items. Moreover, it also supports disconnected operation since (a portion of) the data requests can be served from the local hoard cache and by this increases the robustness against disconnections.

In contrast to caching, the goal of hoarding is to fetch a data item into the cache *before* the first access occurs. Therefore, it is necessary to predict which data items will be needed next. This prediction, called hoarding decision, may be

done by the user himself, telling the system which items to hoard, or it may be done automatically by the system exploiting different types of information, such as user interests or location-dependency and syntactic/semantic relationships of data. In this dissertation, an automatic hoarding mechanism is presented that assumes requests to data items to be location-dependent. Moreover, the algorithm takes into account the semantic distance [58] of data items, which is derived from the users' navigational behavior in a semi-structured information space.

For the purpose of this dissertation, the term *infostation* [5, 104] is used to refer to hotspots that provide hoarding functionality. Infostations are responsible for selecting the data items to be downloaded into the hoard cache of mobile devices. In most hoarding algorithms described in the literature, the hoarding decision is based on information collected about the data access behavior of users. They mainly differ in the type of data they consider (e.g., unstructured or semi-structured) and the information they collect and exploit for the hoarding decision.

In unstructured information spaces, an infostation usually collects information about the popularity of individual data items. The items that seem to be most popular at a given location are then downloaded to the appropriate clients. In semi-structured information spaces, such as the World Wide Web, we can additionally analyze the behavior of users as they navigate through it. With this analysis we can leverage relationships among data items that can be used to determine their semantic distance. Such a relationship can be a (logical) link, which indicates that the linked items were accessed subsequently in a user session. In addition to the popularity of data items, an infostation can now take the popularity of links into account when making the hoarding decision.

The proposed hoarding scheme is generic with respect to the underlying semi-structured information space and has been specialized and evaluated for the Web. It takes into account how users navigate the Web. Moreover, it classifies pages into content and transit pages and uses this classification for clustering. A transit page is one that is (mainly) used to navigate to a content page, which is a page the user is actually interested in. The unit of information used for hoarding is a cluster of Web pages comprising at least one content page plus a sequence of transit pages. In addition, the algorithm computes the relevance for each cluster formed by taking into account its access probability and size. Although this hoarding algorithm has been designed for mobile Web access, it is generic enough to be also used for other semi-structured information spaces.

System Model and Assumptions

The system model comprises *infostations* and *mobile devices*. An infostation is a stationary server connected to the Internet and equipped with a WLAN access point. We use the term *transmission area* to refer to the transmission range of the access point and *hoarding area* to the geographic region, the infostation is responsible for. The hoarding area is much larger than the transmission area.

We assume that a mobile device, such as a PDA or smart phone, is equipped with WLAN capabilities and possibly with wireless WAN connectivity. Mobile devices can be in the following states: *connected*, *weakly connected*, or *disconnected*. A mobile device is connected while it is in the transmission area of an infostation. It is weakly connected if only wireless WAN connectivity is available and in the disconnected state if no connection to a network is possible.

A mobile device runs a browser allowing the user to navigate the Web. In addition, it maintains a *hoard cache*, which stores the Web pages downloaded from the most recently visited infostation. Furthermore, a mobile device maintains a *log file*, which is used to keep track of the locally requested Web pages. Whenever a page is requested, a record is added to the log file containing the URL and size of the page, as well as the time and location of the request. In order to determine the location of Web page requests, we assume that each mobile device can determine its position. The location information associated with a request should allow an infostation to determine in which hoarding area the request was issued. Therefore, GPS accuracy, for example, is fully sufficient. Even cell IDs can be used if a hoarding area is defined as a set of cells. Of course, in this case, cellular WAN connectivity is mandatory.

The proposed hoarding approach is based on the assumption that information access is location-dependent, i.e., the information a user is interested in depends on his current location. This assumption is valid for most applications taking into account the users' position, such as navigation systems, location-based services, or context-aware information systems. Location-based applications typically "focus" their information needs on a very limited fraction of the entire information space, where this fraction depends on the user's current location. To exploit this location-dependency for the hoarding decision, an infostation monitors the collective user behavior on a per-location basis and manages this knowledge using a so-called *information graph*. In other words, each infostation knows how the users residing in its hoarding area typically access the data. Of course, this knowledge

must be dynamically adapted since the access behavior will change over time. The advantage of monitoring the collective access behavior is that those data is selected for hoarding that currently is of interest at a particular location.

Further, hoarding also works for users visiting a location for the first time. However, the problem with considering collective behavior only is that the hoarding decision may not be optimal for an individual user. This effect diminishes if an infostation exploits user profiles including interests, such as in culture or sports, and monitors the access behavior on a per-location per-interest group basis. For each user profile, an information graph is managed.

Overview

When a user requests a page, the *hoard cache manager* checks whether or not this page is stored in the hoard cache. If the page is cached, it can immediately be returned to the user. In case of a hoard miss, the page may be accessed over the network, provided the device is (weakly) connected. In either case, a hoard miss is signaled to the mobile device.

When a mobile device moves into the transmission area of an infostation, it becomes connected and uploads its log file and hoard cache size to the corresponding infostation, which uses the log file to update the corresponding *information graph*. An information graph models the aggregated browsing behavior of all users of an interest group while they are in the hoarding area.

The *Graph Update Component* divides the uploaded log file into *sessions*, defined as a sequence of temporally related page requests. Since graph update is performed each time, a log file is uploaded, our approach permanently adapts to user access behavior.

The classification of pages into content and transit pages is the basis of our *clustering component*. The pages represented in the information graph are grouped into clusters including at least one content page plus a sequence of transit pages needed to navigate to the content pages.

The *hoard list generator* assigns a relevance value to each of the clusters that depends on the probability of its pages being accessed and its size. The final step involves the generation of an ordered list of Web pages, which is called a *hoard list*.

Due to performance reasons, clustering and hoard list generation is done only periodically, while graph update and hoard list download is performed whenever a log file is uploaded.

Graph Update

A user log file is composed of log entries which contain the URL of the page and its size, as well as the location and time of the request. We calculate the *visit period* of a page by subtracting the timestamps of two consecutive log entries and use this information to subdivide the log into so-called *sessions*. A session is a subset of consecutive log entries in a log file, where the visit period of each page falls below a threshold value.

We also use a page's visit period to detect whether it is a content page or a transit page. Content pages are assumed to be visited considerably longer than transit pages. Moreover, a log file usually contains more transit pages than content pages, since users tend to click through a series of transit pages to find the page they are interested in. Therefore, we decided to use the geometric mean as metric to filter out content pages.

As mentioned above, the (collective) access behavior is tracked on a per-location basis. Since a session models a sequence of logically related accesses, the location associated with the first entry of a session defines the location of the complete session.

An infostation maintains one *information graph* per user profile, that encodes the information received as part of the user log file into a weighted graph. The vertices represent the individual Web page requests and the edges contain information about the navigational browsing patterns of users.

More formally, an information graph IG is a tuple $IG = (V, E, f, w, B, z)$, where V is the set of vertices representing Web pages; E is the set of edges; f is a function that assigns a weight to each vertex and edge; w is the root vertex; $B \subseteq V$ is the set of session start vertices; and z is an accumulative vertex used for bookkeeping of sessions and user revisit patterns.

Additionally, there is extra information encoded in both vertices and edges. Each vertex $v \in V$ has attributes v.URL and v.size, that contain the URL and size of the page associated with it. Furthermore, a vertex also has counters v.request

and v.content that record, how many times a page has been requested and has been considered to be a content page, respectively. Analogously, each edge $e \in E$ has a counter e.traversal that keeps track of how many times its source and sink vertices have been requested consecutively over the lifetime of the information graph.

For each session $\mathcal{L}_s = \{l_1, \ldots, l_m\}$ identified during the log analysis step, the information graph assigned to the current user profile is updated using the following steps. For the ease of exposition, we assume that a log entry l_i has a corresponding vertex v_i in an information graph.

1. For each entry $l_i \in \mathcal{L}_s$, increment the $v_i.\mathit{request}$ counter. If l_i is a content page, increment $v_i.\mathit{content}$, too.
2. Increment the traversal counter of an edge e according to the following cases:
   a) If l_i is a session start page, update edge $e = (w, v_i)$.
   b) If l_i is a session end page, update edge $e = (v_i, z)$.
   c) For all other entries l_i with predecessor l_{i-1}, update $e = (v_{i-1}, v_i)$ if l_i appears in the session for the first time. Otherwise, update $e = (v_{i-1}, z)$.

Following this algorithm, the information graph only adapts very slowly to changing user behavior. In fact, the ability to adapt becomes even worse over time, since past and current page accesses are counted equally. Therefore, an *ageing mechanism* is integrated, which divides time into epochs. Whenever an epoch ends, for each of the aforementioned counters, a smoothed value is computed using an exponentially weighted moving average function.

Clustering and Hoard List Generation

At the end of an epoch, the infostation traverses the information graphs and builds a list of *information clusters* for each user profile, sorted by relevance to be used as hoard units.

A cluster is defined as a subset of the information graph that contains a sequence of vertices lying on a path from the root vertex w to one or more content vertices.

For each content vertex, a separate cluster is built and, hence, clusters might overlap.

The proposed clustering algorithm is based on a modified version of the Bounded Depth First Search (B-DFS) algorithm. It extends B-DFS in two aspects. First, vertices may be visited multiple times to potentially traverse all paths in the information graph. Secondly, the traversal is bounded in terms of the access probabilities associated with the individual paths.

In order to avoid cycles, each vertex maintains a mark which is set the first time, this vertex is visited in a root-to-leaf direction. The mark is removed during backtracking, which is initiated in three cases: (1) The probability of the path to the following vertex is smaller than a threshold value. (2) The accumulative vertex z is reached. In this case, a session ended or a user revisited pages, respectively. (3) A marked vertex is reached. Note that the *path probability threshold* is an important parameter for limiting the computation overhead.

At the end of this step, we have a set of clusters with statistical information such as their size and access probability. The *hoard list generator* sorts the clusters according to their *relevance*. Since the aim of hoarding is to maximize the number of cache hits when a mobile user requests Web pages and the hoard cache is limited, we consider a cluster's access probability per byte rather than per page when computing the relevance.

The final step involves the generation of the hoard list using the ranked clusters found by the previous algorithms and the hoard cache size reported by the client.

Modeling the Browsing Behavior in the Web

In the experiments, each infostation is assumed to be associated with a so-called location-based subspace. This is the portion of the Web that is relevant to users of location-based applications, while they are in an infostation's hoarding area. Such a location-based subspace (or subspace for short) includes Web pages, as well as links connecting them. Obviously, the number of pages in the associated subspace has a strong impact on the hit ratio. Therefore, subspaces differing in number of objects and average page sizes are considered.

Unfortunately, to my knowledge, a collection of user logs for location-based applications with statistical relevance is not publicly available. In principle, user logs

could be extracted from existing proxy logs. However, in order to systematically evaluate the proposed hoarding approach, it is imperative to experiment with logs that are associated with differently sized subspaces, differing in number of pages and average page sizes. Therefore, I decided to synthetically generate the subspaces and associated a large number of log files where each log models how a single user navigates the corresponding subspace. For this purpose, I created a tool that generates synthetic access sequences based on the *Web Browsing Model* that models the browsing behavior of users on the Web. This model that comprises two sub-models, the *Web Graph Model* and the *Access Behavior Model*. The former models information spaces with the bow-tie-structure found in the Web [14] and document sizes [33], while the latter models how users typically navigate the Web, taking into account the Web page popularity, the number of requests per session, the periods user visit Web pages, and the way users choose the next Web page. In the literature [13, 14, 23, 30, 64, 86], you can find substantial research that validates the applied sub-models and distribution functions by means of an extensive analysis of real-world log files. However, none of these papers have attempted to combine these partial models to provide an integrated Web browsing model, as described in this dissertation. Additionally, I could show using empirical data that both the probability of choosing some hyperlink from a given page and the probability of a user leaving a page without following a hyperlink is best characterized by a power-law. I calibrated the Web browsing model using the log files gathered from our computer center's proxy server in June 2004 and could show that the synthetic access sequences generated by my tool come very close to real-world logs [16]. Due to the self-similarity in the Web [7, 33], this model can be applied to the Web as a whole, as well as to a smaller fraction, such as a subspace.

Evaluation

From a user's perspective, the major performance parameter of a hoarding scheme is the so-called content hit ratio, which is the probability of a content page requested by a user being found in the hoard cache. In my experimental evaluations, I compared the performance of the proposed approach with that of other hoarding schemes, that do not make use of structural information and/or do not perform clustering of semantically related data items. The results show that the proposed hoarding algorithm more than triples the hoard cache content hit ratio compared to existing schemes.

Related Work

To overcome the shortcomings of wireless networks, such as low bandwidth, frequent disconnections, and high latency, various optimization techniques have have been developed for mobile data access.

Caching techniques cache data objects retrieved over the communication network to avoid communication for further accesses to this object (e.g., see [85]). Semantic caching strategies have been developed that exploit geographic locality when replacing objects, e.g., [87]. However, caching aims at optimizing the second and following acceses to an object, while hoarding tries to optimize the first-time access by prefetching the object. Some replacement strategies, such as LRU and LFU, can be applied in hoarding schemes to select the "hot items" to be prefetched (e.g., see [55, 58]).

Several prefetching techniques have been proposed for reducing the user-perceived latency to access Web pages (e.g., see [40], [101]). They predict the set of pages the user probably will access next based on the user's access history. In contrast to that, in the proposed hoarding scheme, the prediction is based on the access behavior of all users residing in a given geographic area.

The Coda file system [51] was one of the first systems that used the concept of hoarding to allow for disconnected operation. However, the hoarding decision requires user interaction. SEER [58] hoards files without user interaction. It also performs clustering of related files based on their semantic distance, where files referenced at the same time are assumed to be semantically related. In [56], the authors compared SEER to pure LRU hoarding without clustering. The results revealed that in most cases, LRU hoarding achieved the better results. The authors of [106] propose another automatic file hoarding scheme, where each file is assigned a priority based on access recency, access frequency and the time a file was opened. The file priority and other file-specific parameters determine which files are to be hoarded. All of the above file hoarding schemes are tailored to the characteristics of file access rather than Web access. Hence, they neither model the navigational access behavior nor the characteristics of Web pages (e.g., content or transit pages), which both are important for determining the semantic distance of Web pages. Moreover, these schemes make their hoarding decision on a per-user basis, while the proposed scheme does it on a per-location basis.

In the Map-on-the-Move project [104], drivers are provided with map information at different resolutions depending on the speed they travel through the

transmission areas of infostations. However, this approach is tailored to map data.

In [55], the authors propose a hoarding mechanism for location-dependent data, where the hoarding decision is based on how frequently a data item is accessed by all users at a given location. This LFU hoarding scheme was designed for unstructured information spaces and does not analyze the navigational access behavior to derive semantic relationships between data items.

The hoarding scheme presented in this dissertation provides for clustering. Several clustering methods have been also proposed in the field of information retrieval, which, however, do not aim at hoarding. Approaches such as [28, 68] thematically cluster documents based on common keywords. The authors of [9] propose a clustering method to group the pages in a result set of a search query based on other queries and their associated result sets. Nonetheless, these approaches do not build overlapping clusters.

Conclusion

In this dissertation, a generic hoarding approach for semi-structured information is introduced that has been specialized and evaluated for Web page requests. The approach is based on an infrastructure of infostations that provide comparatively cheap high-speed access to the Web. I have classified pages into content and transit pages and used this classification to guide the clustering algorithm. In order to maximize the content hit ratio, the clusters are sorted according to their relevance per byte rather than per page. I could show by means of experimental evaluation that the proposed hoarding scheme outperforms existing ones by a factor of more than three in terms of content hit ratio.

Another important parameter is energy consumption. An analysis referring to this shows that the proposed hoarding scheme can also be used to save energy, assuming current wireless LAN and WAN technologies. Knowing the content hit ratio that can be achieved for different hoard cache sizes, the hoard cache size for downloading Web pages can be determined, that is optimal in terms of energy savings.

1 Einführung

Ein Mobiltelefon wird heutzutage längst nicht mehr nur zum Telefonieren benutzt. Es hat sich zu einem Multifunktionsgerät entwickelt, das zusätzlich als Zugang zum World Wide Web (Web), elektronischer Kalender, tragbarer Rechner, Musikbox oder Fernseher verwendet werden kann. Mit der zunehmenden Verbreitung immer leistungsfähigerer mobiler Endgeräte und der rasanten Entwicklung im Telekommunikationssektor wächst auch das Bedürfnis, überall und zu jeder Zeit auf entfernte Informationen zugreifen zu können. Von großer Bedeutung ist in diesem Zusammenhang der mobile Zugriff auf das Web, das von immer mehr Benutzern als Hauptinformationsquelle benutzt wird und darüber hinaus den meisten Internet-Anwendungen als Grundlage dient.

Diese Entwicklungen forcierten neuartige Anwendungsfelder im Bereich mobiler Rechnerumgebungen (engl. *mobile computing*). Ein Beispiel hierfür sind ortsbasierte Systeme, die es ermöglichen, Benutzern ortsspezifische Informationen anzubieten. Dies können Navigationsanwendungen, kontextbezogene Systeme oder Anwendungen im Bereich des so genannten *Sentient Computing* sein, die auf Benutzeraktionen reagieren.

Solche Systeme müssen die zunehmende Mobilität ihrer Anwender und deren wachsendes Informationsbedürfnis zu jeder Zeit und an jedem Ort berücksichtigen. Hierdurch ergeben sich neue Herausforderungen im Bereich der mobilen Datenverwaltung (engl. *mobile data management*), von denen einige zentrale Problemstellungen in Abbildung 1.1 illustriert sind. Die *Software* muss *anpassungsfähig* sein: Sie muss mit unterschiedlichen drahtlosen Netztechnologien um-

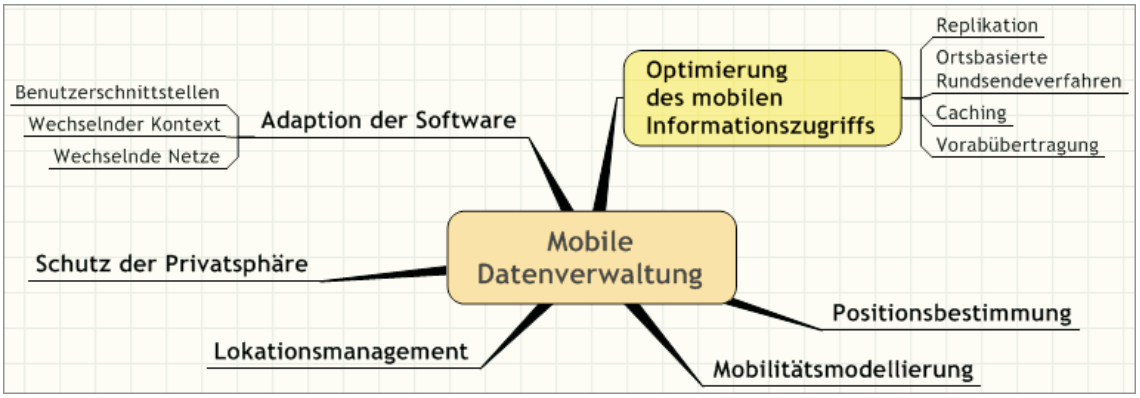


Abbildung 1.1: Problemstellungen in der mobilen Datenverwaltung

gehen und sich an verschiedene Benutzerschnittstellen und/oder wechselnden Kontext anpassen können. Der *Schutz der Privatsphäre* von Benutzern, d.h., die Gewährleistung der Vertraulichkeit personenbezogener Daten, ist eine wesentliche Bedingung für die Akzeptabilität mobiler Systeme. In ortsbasierten Systemen benötigen mobile Endgeräte zusätzlich *Positionsbestimmungssysteme* zur Erfassung der aktuellen Position, wie beispielsweise GPS (Global Positioning System) oder zukünftig Galileo. Die Verwaltung mobiler und statischer Objekte erfordert skalierbare *Lokationsdienste*. Für die Leistungsbewertung ortsbasierter Systeme werden realitätsnahe *Mobilitätsmodelle* benötigt.

In dieser Arbeit wird ausschließlich die *Optimierung des mobilen Informationszugriffs* betrachtet. Drahtlose Netze haben im Vergleich zu drahtgebundenen Netzen eine Reihe von Nachteilen, wie beispielsweise hohe Latenz, häufige Netztrennungen und vor allem auch die bislang noch sehr hohen monetären Gebühren der Betreiber für die Benutzer. Netztrennungen können zum einen dadurch entstehen, dass in einem bestimmten Gebiet keine drahtlose Kommunikation möglich ist. Zum andern kann eine Netztrennung auch vom Benutzer erzwungen werden, indem er die Funkschnittstelle seines Endgeräts aus Gründen der Energieeinsparung abschaltet, denn ein nicht zu vernachlässigender Faktor für die Lebensdauer der Batterie ist die relativ hohe Leistungsaufnahme der Funkschnittstelle. Da die Energieressourcen mobiler Endgeräte beschränkt sind, sollten die Zeiten für

das Senden und Empfangen von Daten möglichst gering gehalten werden. Zur Optimierung des mobilen Informationszugriffs können je nach Anwendungsfeld Verfahren wie Replikation, ortsbezogenes Rundsenden, Caching oder die Vorabübertragung eingesetzt werden. Die Wahl eines geeigneten Verfahrens hängt ab von der Klasse des zugrunde liegenden Systems (z.B verteiltes Dateisystem, ortsbasiertes System, o.ä.) und der Art des Informationsraums (unstrukturiert, schwach oder stark strukturiert). Eine Klassifikation des Informationsraums findet sich in Abschnitt 2.2.2. Stark strukturierte Informationen stehen durch eine genau definierte, explizite Struktur miteinander in Beziehung. Schwach strukturierte Informationen weisen eine unregelmäßige, implizite Struktur auf, mittels derer Beziehungen zwischen den Informationen abgeleitet werden können. In dieser Dissertation werden die Begriffe *Information* und *Daten* folgendermaßen definiert:

Definition 1 (Information, Daten) *Eine **Information** bildet den Inhalt einer Nachricht in textlicher, grafischer oder audiovisueller Form. **Daten** sind Nachrichten, also Folgen von Zeichen, die maschinell verarbeitet werden können.*

Diese Definitionen lehnen sich die entsprechenden Definitionen aus dem Brockhaus und dem Duden an.

In dieser Arbeit wird ein generisches Vorabübertragungsverfahren zur Optimierung des Zugriffs auf schwach strukturierte Informationen in mobilen ortsbasierten Systemen vorgestellt, dessen Ziel es ist, Benutzern einen möglichst schnellen Zugriff auf Informationen zu jeder Zeit und an jedem Ort zu ermöglichen. Das Verfahren wurde für den Zugriff auf Webseiten spezialisiert und evaluiert.

1.1 Motivation

Der mobile Zugriff auf Informationen wird durch die immer weiter fortschreitenden funkbasierten Kommunikationstechnologien ermöglicht.

Drahtlose Weitverkehrsnetze (engl. *Wireless Wide Area Network*, WWAN) wie zelluläre Netze der zweiten (2G) oder dritten (3G) Generation bieten nahezu überall eine Verbindung zum Internet. Allerdings stellen selbst 3G-Netze wie der heutige Mobilfunkstandard UMTS typischerweise nur einige (wenige) hundert Kbps Bandbreite zur Verfügung. Weitere Nachteile zellulärer Netze sind deren Fehleranfälligkeit, mögliche Netztrennungen und die immer noch hohen monetären Kosten der Betreiber für die Benutzer.

Als hierzu komplementäre Technologie bieten mit *drahtlosen lokalen Netzen* (engl. *Wireless Local Area Network*, WLAN) ausgestattete *Hotspots* mobilen Benutzern in einem begrenzten geographischen Gebiet einen schnellen und kostengünstigen oder sogar freien Zugriff auf Informationen. In lokalen Netzen werden durch die geringe Distanzüberbrückung Datenübertragungsraten zur Verfügung gestellt, die mindestens eine Größenordnung höher liegen als in zellulären Netzen.

Die Optimierung des mobilen Informationszugriffs wird durch die Verbindung dieser beiden Kommunikationstechnologien (zelluläre Netze und mit WLAN-Zugang ausgestattete Hotspots) ermöglicht. Der schnelle Zugriff an den Hotspots wird dazu genutzt, um diejenigen Informationen vorab in den lokalen Cache eines mobilen Endgeräts zu laden, auf die der Benutzer zukünftig mit hoher Wahrscheinlichkeit zugreifen wird. Das Ziel der Vorabübertragung ist also die Minimierung der Zahl der Informationszugriffe im WWAN, einhergehend mit einer gleichzeitigen Kostenreduktion und Verringerung der Latenz, also die vom Nutzer wahrgenommene Wartezeit von der Anforderung der Information bis zu deren Anzeige auf dem Endgerät. Diese Wartezeit kann auf Grund der oft großen Entfernungen zu einem (Web-)Server sowie häufiger Server-Engpässe trotz immer schnellerer Verbindungen im Sekunden- bis Minutenbereich liegen. Marshak und Levy werten in [69] die Zugriffszeiten auf die Informationen von einhun-

dert Web-Auftritten aus und ermitteln Latenzen von bis zu fünfzehn Sekunden bei mittlerer Server-Last. Dem gegenüber steht eine Latenz im Bereich von einigen Millisekunden, wenn eine Anfrage aus dem lokalen Cache beantwortet werden kann. Des Weiteren unterstützt die Vorabübertragung den entkoppelten Betrieb, wenn zumindest ein Teil der angeforderten (relevanten) Informationen im Cache gespeichert ist und somit die Robustheit des Informationszugriffs bezüglich Netztrennungen erhöht wird. Schließlich kann die Vorabübertragung auch dazu eingesetzt werden, um den Energieverbrauch zum Laden der Informationen zu minimieren. Dies ist eine wichtige Eigenschaft für mobile Endgeräte, denen typischerweise nur beschränkte Energieressourcen zur Verfügung stehen. Durch die in einem WLAN zur Verfügung stehende wesentlich höhere Bandbreite ist die Übertragungszeit bedeutend kürzer als in einem WWAN, was letztendlich zur Energieeinsparung führen kann, wenn der Cache überwiegend mit relevanten Informationen gefüllt wird.

Vorabübertragungsverfahren unterscheiden sich hauptsächlich in der Art, wie die Selektion der vorab zu ladenden Informationen vorgenommen wird. Diese Auswahl kann automatisch vom System oder durch Einbeziehung von Benutzerhinweisen getroffen werden. Die automatische Selektion ist jedoch immer dann besser, wenn ein zugrunde liegender Informationsraum wie beispielsweise das Web eine riesige, vom Benutzer nicht überschaubare Menge an Informationen bietet. Die Herausforderung bei der automatischen Selektion ist es, geeignete Kriterien für eine möglichst optimale Auswahl zu bestimmen. Diese wiederum sind abhängig von der Art des Informationsraums (unstrukturiert, strukturiert) und der Anwendung selbst. So sind beispielsweise die im Bereich verteilter Dateisysteme existierenden Vorabübertragungsverfahren, wie das im Projekt SEER [58] entwickelte, speziell auf die Eigenschaften von Dateien zugeschnitten und werten zur Bestimmung der vorab zu ladenden Dateien die Zugriffshistorie des aktuellen Benutzers aus. Ein weiteres Beispiel ist das in der Dissertation von Kubach [55] speziell für ortsbasierte Systeme konzipierte Vorabübertragungsverfahren, das unstrukturierte Informationsräume betrachtet.

Der wissenschaftliche Beitrag dieser Arbeit ist die Entwicklung eines *generischen Vorabübertragungsverfahrens* für beliebige Arten schwach strukturierter Informationsräume in ortsbasierten Anwendungen. Es stellt Auswahlverfahren mit und ohne Clusterbildung zur Verfügung, welche die semantische Nähe der Informationen berücksichtigen, die sich wiederum aus deren jeweils zu definierenden Beziehungen ableiten lässt. Da erwartungsgemäß nicht alle Benutzer ein ähnliches Zugriffsverhalten aufweisen, lässt sich durch die Berücksichtigung von Nutzungsprofilen die Relevanz der vorab übertragenen Informationen für einzelne Benutzergruppen erhöhen. Das Vorabübertragungsverfahren wird für den mobilen Zugriff auf das Web als Repräsentanten eines schwach strukturierten Informationsraums spezialisiert und evaluiert. Eine systematische Leistungsbewertung erfordert das Vorhandensein unterschiedlich dimensionierter Informationsräume und eine statistisch relevante Anzahl von Protokolldateien, die Benutzerzugriffe auf diese Informationsräume enthalten. Da solche Protokolldateien nicht öffentlich verfügbar sind, werden in dieser Dissertation mehrere Informationsräume und mit ihnen assoziierte Protokolldateien synthetisch erzeugt. Hierfür wird ein Modell für das Navigationsverhalten von Benutzern im Web entwickelt, mit dessen Implementierung Sequenzen von Zugriffen auf das Web generiert werden [16]. Mit dem in dieser Arbeit vorgestellten Vorabübertragungsverfahren konnten Trefferraten erzielt werden, die andere Ansätze um mehr als das Dreifache übertreffen. Die Ergebnisse wurden in [17], [18] und [19] vorgestellt.

Kommunikationstechnologien wie die Vorabübertragung von Informationen werfen grundsätzlich Fragestellungen bezüglich des Schutzes der Privatsphäre von Benutzern auf. Der Datenschutz kann durch die Sammlung und Aggregation personenbezogener Daten wie Identität, Ortsinformationen oder Profile durch einen Betreiber massiv bedroht werden. Hier spielt ähnlich wie im Bereich der Mobilfunknetze das Vertrauen in den Betreiber der Infostationen eine große Rolle. Daneben kann die Integration von Sicherheits- und Vertrauenskonzepten wie beispielsweise die Anonymisierung sicherlich dazu beitragen, die Akzeptabilität solcher Verfahren zu erhöhen. In dieser Dissertation liegt der Fokus jedoch auf der Optimierung der Selektion vorab zu ladender Informationen, Sicherheitsaspekte

sind nicht Gegenstand dieser Arbeit.

Eine Randbedingung bei der Entwicklung des Verfahrens war eine nahtlose Integration in die NEXUS-Plattform [44], die im Rahmen des Sonderforschungsbereichs 627 „NEXUS - Umgebungsmodelle für kontextbezogene Systeme" entwickelt wird. Durch die rasch fortschreitende Entwicklung im Bereich der Sensorik, der drahtlosen Kommunikationstechnologie und der Miniaturisierung von Computersystemen lässt sich absehen, dass zukünftig Kleinstcomputer in nahezu allen Bereichen des täglichen Lebens vorhanden sein werden. Diese Computersysteme werden mit Sensoren ihre Umgebung erfassen, auf das Web zugreifen und miteinander kommunizieren können. Durch die daraus entstehende Vielfalt von Anwendungen sind Benutzer darauf angewiesen, dass diese Programme ihre Anforderungen ohne aufwändige Konfiguration erfüllen, sich automatisch an die aktuelle Situation der Benutzer anpassen und ein auf die jeweilige Situation angepasstes Informationsangebot liefern. Ein Beispiel hierfür sind intelligente Umgebungen, die durch Sensorik und Aktorik mit dem Menschen interagieren und ihn so bei seinen täglichen Aufgaben unterstützen.

Die Grundlage kontextbezogener Anwendungen sind so genannte *Umgebungsmodelle*, die relevante Aspekte der physischen Welt modellieren. Diese Modelle beinhalten Abbilder von real existierenden Objekten und können zusätzlich mit digitalen Informationen angereichert werden. Sie beschreiben Beziehungen zwischen Objekten und beinhalten Kontextinformationen. Umgebungsmodelle können hoch dynamisch sein, insbesondere wenn sie mobile Objekte wie beispielsweise Personen oder Fahrzeuge enthalten, oder wenn die durch Sensoren erfassten Veränderungen der physischen Umgebung zeitnah in das Modell propagiert werden. Durch diese Dynamik entsteht ein immenser Aufwand für die Erzeugung und Verwaltung von Umgebungsmodellen. Damit sich die hohen Kosten auf möglichst viele Nutzer verteilen, wird eine gemeinsame Nutzung der Modelle zwingend erforderlich sein. Hieraus ergibt sich analog zum Web die Vision eines „World Wide Space", in den eine Vielzahl unterschiedlicher Umgebungsmodelle integriert werden können, die primär durch ihren räumlichen und zeitlichen Bezug verbunden

werden. Das zugrunde gelegte System ist offen, so dass jeder Anbieter sein Modell einbringen kann. Durch den Zusammenschluss der einzelnen lokalen Umgebungsmodelle in eine Föderation entsteht schließlich ein globales Umgebungsmodell, in dem Anwendungen in Raum und Zeit navigieren können. Bedingt durch die Offenheit des Systems ergeben sich Problemstellungen wie die der Sicherheit, des Vertrauens, der Datenqualität, der Konsistenz, der Fehlertoleranz, der Zuverlässigkeit und der Skalierbarkeit der zugrunde liegenden Systemmechanismen. Weiterhin müssen geeignete Bepreisungs- und Abrechnungsverfahren entwickelt werden, um die wirtschaftliche Überlebensfähigkeit der Modelle zu gewährleisten.

Bezüglich der Verwaltung, Präsentation und Nutzung von globalen Umgebungsmodellen ergeben sich unter anderem folgende Problemstellungen:

Kommunikation: Für einen geeigneten Zugriff mobiler Anwendungen auf die Modelldaten müssen neue Kommunikationskonzepte entworfen werden, welche den Datenzugriff unter Ausnutzung von Kontextinformation optimieren. So muss zum einen ein nahtloser Übergang zwischen heterogenen Netzen gewährleistet werden. Zum andern soll mit einem geeigneten Vorabübertragungsverfahren unter Einbeziehung von Kontextinformation das Problem von Netztrennungen und Verbindungen mit niedriger Bandbreite bzw. hohen Kosten gemildert werden.

Bei der *Erfassung von Kontextinformation* spielt deren Qualität eine große Rolle: Durch die Betrachtung der Qualität in mehreren Dimensionen wie Unschärfe, Ungenauigkeit oder Konsistenz sollen Anwendungen zukünftig entscheiden können, ob bereits vorhandene Informationen zur Bearbeitung ihrer Anforderungen ausreichen, oder ob zusätzliche Informationen erfasst werden müssen.

Präsentation von Kontextinformationen: Durch die kontextgesteuerte Visualisierung sollen den Anwendungen angepasste Sichtweisen auf komplexe Szenarien und Situationen ermöglicht werden. Ein weiteres Ziel ist es, auch die Qualität von Kontextinformationen geeignet zu visualisieren.

Entwurfsmethodik und Anwendungsunterstützung: Da sich kontextbezogene Anwendungen durch ihren hohen Grad an Adaptivität und die zum Zeitpunkt

der Entwicklung unbekannte Ausführungsumgebung wesentlich von existierenden Anwendungen unterscheiden, sind neue Architekturen von Anwendungen sowie begleitende Entwurfskonzepte zu entwickeln.

Anwendungen: Durch die Einbeziehung von Anwendern soll eine Rückkopplung an die Konzepte zur Verwaltung und Nutzung des Umgebungsmodells abgeleitet werden, wobei durch die unterschiedlichen Sichtweisen und speziellen Anforderungen interdisziplinäre Lösungsansätze und eine Konkretisierung der Vision eines „World Wide Space" zu erwarten sind.

Das vorgestellte Vorabübertragungsverfahren ist ein Kommunikationskonzept zur Optimierung des Zugriffs mobiler Anwendungen auf im NEXUS-Umgebungsmodell gespeicherte Informationen.

1.2 Übersicht

Die vorliegende Arbeit ist wie folgt strukturiert.

In Kapitel 2 wird die Arbeit in den Bereich der Optimierung des mobilen Informationszugriffs eingeordnet. Insbesondere wird der Unterschied zwischen der Vorabübertragung und den übrigen Optimierungsmethoden für den mobilen Informationszugriff herausgearbeitet. Im Anschluss daran wird das in dieser Arbeit vorgestellte Verfahren in die vorgeschlagene Taxonomie von Vorabübertragungsverfahren eingeordnet.

Kapitel 3 ist der eigentliche Kern dieser Arbeit. Zunächst wird der grundsätzliche Ablauf des Verfahrens geschildert und die sich hieraus ergebenden Problemstellungen diskutiert. Nach der Beschreibung des Systemmodells und der zugrunde liegenden Annahmen werden die generischen Konzepte vorgestellt, bevor deren Spezialisierungen für den Zugriff auf das Web detailliert beschrieben und bezüglich ihrer qualitativen und quantitativen Eigenschaften bewertet werden. Das Kapitel schließt mit der Konzeption einer möglichen Erweiterung und der Dis-

kussion verwandter Arbeiten im Bereich der Vorabübertragung.

In Kapitel 4 wird die zur Leistungsbewertung erforderliche Simulationsumgebung vorgestellt. Die systematische Leistungsbewertung des Vorabübertragungsverfahrens erfordert den Einsatz von synthetischen Zugriffen auf Informationen, die aus einer Vielzahl von unterschiedlich dimensionierten Informationsräumen stammen. Hierfür wird ein Modell für die Navigation von Benutzern im Web erstellt, das bekannte Ergebnisse aus der Literatur und eigene Forschungsergebnisse integriert. Die Kalibrierung erfolgt auf der Grundlage von realen Protokolldateien. Das Modell wurde implementiert und in die Simulationsumgebung integriert.

Die systematische Leistungsbewertung des Vorabübertragungsverfahrens erfolgt in Kapitel 5. Sie umfasst unter anderem das Verhalten des Verfahrens bezüglich unterschiedlich dimensionierter zugrunde liegender Informationsräume, den Vergleich mit anderen Ansätzen sowie den Vorschlag einer optimalen Parameterbelegung.

Die Analyse des Energiebedarfs der Funkschnittstellen mobiler Endgeräte zur Anforderung von Webseiten ist Gegenstand von Kapitel 6. Basierend auf den Werten zur Leistungsaufnahme aktueller Funkschnittstellen wird die für den Zugriff auf das Web mittels Vorabübertragung notwendige Energie berechnet. Diesem Wert wird der entsprechende Energiebedarf für den Zugriff ohne Vorabübertragung gegenübergestellt.

Die Implementierung und eine mögliche Integration in die neXus-Plattform wird in Kapitel 7 vorgestellt.

Schließlich werden in Kapitel 8 die wichtigsten Ergebnisse dieser Arbeit zusammengefasst, ergänzt durch einen Ausblick auf mögliche zukünftige Arbeiten.

2 Optimierung des mobilen Informationszugriffs

In diesem Kapitel werden Optimierungsverfahren für den mobilen Informationszugriff diskutiert. Je nach Anwendungsfeld können Verfahren wie *Replikation, ortsbezogenes Rundsenden, Caching* oder die *Vorabübertragung* eingesetzt werden. Insbesondere wird eine Taxonomie von Vorabübertragungsverfahren vorgestellt, in die schließlich das im Rahmen dieser Arbeit vorgeschlagene Verfahren eingeordnet wird.

2.1 Optimierungsverfahren

Um den Nachteilen drahtloser Kommunikationstechnologien entgegen zu wirken, wurde zur Optimierung des mobilen Informationszugriffs eine Reihe von Verfahren vorgeschlagen. Dies sind insbesondere die Replikation, ortsbezogene Rundsendeverfahren, Caching-Verfahren und schließlich die Vorabübertragung, bei der diejenigen Informationen vorab auf ein mobiles Endgerät geladen werden, auf die ein Benutzer zukünftig mit hoher Wahrscheinlichkeit zugreifen wird. In diesem Abschnitt wird diskutiert, wie sich die Vorabübertragung als Optimierungstechnik für den mobilen Informationszugriff von den übrigen Verfahren in diesem Bereich unterscheidet.

2.1.1 Replikation

Replikationsverfahren werden in verteilten Systemen überwiegend zur Reduzierung des Datenverkehrs, Erhöhung der Verfügbarkeit und der Zuverlässigkeit, sowie zur Lastbalancierung eingesetzt.

Ansätze zur partiellen Replikation, die den mobilen Informationszugriff unterstützen, findet man in *verteilten Informationssystemen* im Web. Acharya und Zdonik präsentieren in [2] eine dynamische, verteilte Replikationsstrategie zur Reduzierung des Datenverkehrs. Basierend auf der Auswertung von Zugriffsmustern für die einzelnen Informationen wird dynamisch die Anzahl der Kopien und deren Platzierung berechnet, um eine balancierte Verteilung der Informationen bezüglich der Lese- und Schreiboperationen zu gewährleisten. Bestavros schlägt in [10] eine anfragebasierte hierarchische Replikationsstrategie vor, mit der die Knoten des Internets in Cluster (von Clustern) aufgeteilt und die Informationen eines Anbieters automatisch und dynamisch auf solche Server verteilt werden, die näher beim Benutzer liegen. Der Grad der Verteilung hängt ab von der erwarteten Reduktion des Datenverkehrs und der Beliebtheit der Informationen, relativ gesehen zu allen Informationen im System.

Anbieter von Web-Inhalten (engl. *Web content providers*) verwenden zur Bereitstellung ihrer Informationen immer häufiger Systeme, die Replikationsverfahren mit einbeziehen. Ein Beispiel hierfür sind so genannte *content delivery networks*. In [92] geben Sivasubramanian et al. einen Überblick über einige dieser Systeme bezüglich folgender Kriterien: Bestimmung einer geeigneten Metrik, Initiierung der Anpassung des Verfahrens, Platzierung der Replikate, Einhalten der Konsistenzbedingungen und Weiterleitung der Benutzeranfragen zu den entsprechenden Servern.

Ein weiteres Anwendungsfeld sind *verteilte Anfragesysteme* (engl. *information retrieval systems*). In [66] stellen Lu und McKinley ein Verfahren zur partiellen Replikation vor, das die Lokalität von Anfragen zur Bildung und Suche von partiellen Kopien ausnutzt. Eine Kopie enthält hierbei eine Anfragelogik und

die dazu gehörende Teilmenge der Textdokumente und kann somit Anfragen beantworten, die zwar unterschiedlich sind, aber die gleiche oder sehr ähnliche Ergebnismengen liefern.

Tu et al. präsentieren in [100] einen transaktionsbasierten Ansatz für die partielle Replikation in *verteilten Datenbanken* für mobile Umgebungen. Dabei werden Datenobjekte, die innerhalb einer Transaktion angefordert werden, zusammen platziert.

Diskussion: Im Gegensatz zur Replikation steht bei Vorabübertragungsverfahren die Selektion von Informationen im Vordergrund. Ansätze, die zusätzlich Konsistenzbedingungen berücksichtigen müssen, verwenden in diesem Fall Replikationsstrategien. Ein Beispiel hierfür ist das von Satyanarayanan et al. in [90] vorgestellte Dateisystem CODA, das in Abschnitt 3.12.2 ausführlicher diskutiert wird.

2.1.2 Ortsbezogene Rundsendeverfahren

Das Ziel ortsbezogener Rundsendeverfahren ist es, Informationen effizient an mobile Benutzer zu verteilen.

Tan und Ooi klassifizieren in [98] Technologien zur Verteilung der Informationen bezüglich vier Dimensionen.

1. In der *informationsorientierten Dimension* werden die Verfahren dahingehend unterschieden, welche Art von Informationen sie übertragen. Die erste Klasse von Verfahren liefert nur solche Informationen aus, die unabhängig von ihrer aktuellen Anforderung rundgesendet werden, während in der zweiten Klasse nur aktuell angefragte Informationen ausgeliefert werden. Die dritte Klasse ist eine Kombination der beiden ersten Klassen.

2. Die *verfahrensorientierte Dimension* klassifiziert die Verfahren nach dem

verwendeten Übertragungsmechanismus. Hier wird in der ersten Unterdimension unterschieden, ob die Auslieferung vom Benutzer oder vom Server angestoßen wird. Die zweite Unterdimension teilt die Verfahren in zeitplangesteuerte und ereignisgesteuerte Verfahren auf. Bei zeitplangesteuerten Verfahren werden die Informationen nach einem vordefinierten Zeitplan verteilt (z.B. wöchentlich oder täglich), während bei ereignisgesteuerten Verfahren Informationen in Erwiderung auf Ereignisse gesendet werden, wie beispielsweise Aktualisierungen oder Anfragen. Die dritte Unterdimension schließlich ordnet die Verfahren nach der Art der Kommunikation, entweder 1-1 (*Unicast*) oder 1-N (*Broadcast*).

3. Die *organisationsorientierte Dimension* ordnet die Verfahren danach, wie die Informationen organisiert werden. Dies kann *ad hoc* geschehen, indem entweder Anfragen sofort beantwortet werden oder indem die Antworten gepuffert und dann zusammen in einem Block gesendet werden. Werden die Informationen in einer *festen Reihenfolge* gesendet, unterscheiden Tan und Ooi wiederum zwischen azyklischen und zyklischen Rundsendeprogrammen. Zyklische Rundsendeprogramme werden nochmals in flache und nicht-flache unterteilt. In flachen Rundsendeprogrammen sind alle Informationen innerhalb eines Rundsendezyklus genau einmal enthalten, während in nicht-flachen Rundsendeprogrammen häufig angeforderte Informationen in einem solchen Zyklus auch mehrfach vorkommen können. Azyklische oder randomisierte Rundsendeprogramme sind typischerweise nicht-flach und werden unter anderem mittels probabilistischer Verfahren erzeugt. Schließlich werden die Verfahren noch in indizierte und nicht-indizierte unterteilt. Bei indizierten Verfahren werden zusätzlich Metainformationen in Form eines Indexes mitgesendet. Ein Index beschreibt, welche Informationen wann innerhalb eines Zyklus gesendet werden. Hiermit können mobile Endgeräte gezielt auf die gewünschten Informationen zugreifen, was sich positiv auf den Energieverbrauch auswirkt.

4. Die *bandbreitenorientierte* Dimension schließlich ordnet die Verfahren da-

nach, ob die zur Verfügung stehende Bandbreite für die unterschiedlichen Informationstypen aus der ersten Dimension dynamisch oder statisch vergeben werden soll.

Acharya et al. stellen in [1] ein so genanntes *Datenkarussell* (eng. *broadcast disk*) vor. In diesem Verfahren werden zyklische Rundsendeprogramme erstellt und periodisch gesendet. Diese Übertragungsart erlaubt es mobilen Geräten, auf entfernte Informationen so zuzugreifen, als ob diese auf einer Festplatte gespeichert wären. Datenkarusselle sind nach der Klassifikation von Tan und Ooi zeitplangesteuerte und von Benutzern initiierte Verfahren.

An der Universität Lancaster (U.K.) wurde in Zusammenarbeit mit dem Touristeninformationszentrum ein *elektronischer Touristenführer (GUIDE)* erstellt. Die Resultate einer Feldstudie wurden von Cheverst et al. in [21] veröffentlicht. Als Kommunikationsinfrastruktur wurden in Lancaster an beliebten Touristenattraktionen WLAN-Funkzellen eingerichtet, deren zugeordnete Zell-Server einen Teil der insgesamt angebotenen Informationen im Cache speichern. Eine Auswahl hieraus wird mittels Rundsenden an die Benutzer verteilt, wobei ein flaches, periodisch gesendetes Rundsendeprogramm (Datenkarussell) eingesetzt wird, das einen Index enthält. Falls eine gewünschte Information im Rundsendeprogramm fehlt, kann sie explizit vom Benutzer angefordert werden. Auf Grundlage der Klassifizierung nach Tan und Ooi, handelt es sich um ein indiziertes, zeitplangesteuertes, vom Benutzer initiiertes Verfahren mit flachem Rundsendeprogramm.

Diskussion: Ortsbezogene Rundsendeverfahren zielen auf die Entwicklung von Mechanismen zur effizienten Datenübertragung ab, während bei Vorabübertragungsverfahren die Auswahl einer Menge von relevanten Informationen im Vordergrund steht. Rundsendeverfahren können jedoch zur Optimierung der Verteilung der Informationen zusätzlich in Vorabübertragungsverfahren eingesetzt werden.

2.1.3 Caching

Caching-Verfahren speichern Informationen in einem lokalen Cache, sobald sie zum ersten Mal angefordert wurden. Sie werden zur Reduzierung der Zugriffszeit oder Bandbreitenbelegung verwendet, wie beispielsweise im Bereich der Betriebssysteme, der Prozessorarchitektur, in der Datenbanktechnologie oder im Web. Das Ziel ist es, möglichst diejenigen Informationen zu speichern, auf die in Zukunft erneut zugegriffen wird. Da der zur Verfügung stehende Speicher begrenzt ist, besteht die Herausforderung darin, durch eine geeignete Ersetzungsstrategie diejenigen Informationen aus dem Cache zu entfernen, die wahrscheinlich nicht mehr referenziert werden. Der Ersetzungsprozess wird gestartet, sobald der Cache voll ist. Hierbei kann zeitliche und/oder räumliche (semantische) Lokalität zwischen den Informationsanforderungen ausgenutzt werden. Zeitliche Lokalität bezieht sich auf Zugriffe auf eine einzelne Information innerhalb einer gewissen Zeitspanne. Aus einer hohen zeitlichen Lokalität kann man dann schließen, dass diese Information wahrscheinlich zukünftig wieder referenziert wird. Ein Beispiel hierfür ist die Ersetzungsstrategie LRU (engl. *Least Recently Used*). Semantische Lokalität wird aus Beziehungen zwischen den Informationen abgeleitet, wie beispielsweise die Zugehörigkeit zu einem verwandten Thema. Sie weist darauf hin, dass der Zugriff auf eine Information vermutlich den Zugriff auf eine andere zur Folge hat. Ein Spezialfall hiervon ist die räumliche Lokalität, die sich auf ein geographisches Gebiet beziehen kann, aber auch beispielsweise auf den Speicherort in einer Datenbank.

Neben den Ersetzungsstrategien sind in Caching-Verfahren noch weitere Kriterien zu beachten. Ansätze wie beispielsweise [65, 103] beschäftigen sich mit der Frage, auf welchen Servern die Kopien gespeichert werden sollen. Kooperative Caching-Strategien wie zum Beispiel [102] haben zum Ziel, den Zustand von Caches gemeinsam zu nutzen und zu verwalten. Zur Konsistenzerhaltung von Cache-Inhalten schlagen beispielsweise Yu et al. in [105] eine skalierbare Architektur vor. Diese Kriterien sind für das vorgestellte Vorabübertragungsverfahren nicht relevant und werden deshalb hier nicht näher erläutert. Nachfolgend wer-

den einige der in der Literatur vorgestellten Ersetzungsstrategien diskutiert, die ähnlich wie bei der Vorabübertragung auf der Selektion von Informationen fokussieren.

Podlipnig und Böszörmenyi geben in [85] einen ausführlichen Überblick über *Ersetzungsstrategien für Web-Caches*, die sie in fünf Klassen einteilen: Neuheitsbasierte (engl. *recency-based*), häufigkeitsbasierte (engl. *frequency-based*), neuheits-/häufigkeitsbasierte, funktionsbasierte und randomisierte Verfahren. Neuheitsbasierte Verfahren nützen die zeitliche Lokalität zwischen Informationsanforderungen aus, während häufigkeitsbasierte Verfahren die Popularität von Informationen auswerten. Funktionsbasierte und randomisierte Verfahren kombinieren die Ansätze der ersten drei Klassen und beziehen zum Teil andere Faktoren wie die Latenz mit ein. Diese Ersetzungsstrategien nutzen darüber hinaus keine weiteren Beziehungen zwischen den Informationen aus.

Semantische Caches zeichnen sich dadurch aus, dass die Ergebnismengen einer Anfrage zusammen mit deren semantischer Bedeutung gespeichert werden. Somit können ähnliche Anfragen ganz oder zumindest teilweise aus dem Cache beantwortet werden. Die in diesen Verfahren verwendeten Ersetzungsstrategien nutzen semantische Lokalität zwischen den Anfragen aus, wie beispielsweise deren geographische Korrelation. Ren und Dunham stellen in [87] eine Ersetzungsstrategie für semantische Web-Caches in ortsbasierten Systemen vor, die auf der Annahme basiert, dass Zugriffsmuster von Benutzern mit deren Bewegung verknüpft sind. Es erfolgt also eine Zuordnung der Informationen zu geographischen Positionen. In die Entscheidung, welche Cache-Einträge zu entfernen sind, fließt vorhandenes Wissen über die aktuelle Position, Geschwindigkeit und Bewegungsrichtung der Benutzer mit ein. Als Ersetzungskandidaten kommen dann die Einträge in Frage, die am weitesten von der momentanen Position entfernt sind (*Furthest Away Replacement*).

Diskussion: Im Gegensatz zu Caching-Verfahren, bei denen nicht der erste Zugriff auf eine Information optimiert wird, sondern erst die nachfolgenden, wird

bei der Vorabübertragung versucht, diesen ersten Zugriff zu gewährleisten bzw. zu optimieren. Grundlage beider Verfahren ist die Selektion von Informationen, nur eben zu unterschiedlichen Zwecken. So werden Ersetzungsstrategien der Caching-Verfahren auch in einigen Vorabübertragungsverfahren zur Bestimmung der vorab zu übertragenden Informationen eingesetzt. Beispielsweise basiert im Projekt SEER [56] diese Auswahl auf der LRU Ersetzungsstrategie. Das in der Dissertation von Kubach vorgestellte Vorabübertragungsverfahren [54] setzt die LFU-Ersetzungsstrategie zusammen mit einer Alterungsfunktion ein. Wie jedoch in der in Kapitel 5 beschriebenen Leistungsbewertung gezeigt wird, sind diese Ersetzungsstrategien für den ortsbezogenen Zugriff auf schwach strukturierte Informationen weniger gut geeignet.

2.2 Taxonomie von Vorabübertragungsverfahren

Vorabübertragungsverfahren dienen dazu, Informationen auf ein mobiles Endgerät zu laden, bevor sie explizit angefordert werden. Abbildung 2.1 zeigt eine auf [54] basierende Taxonomie der Vorabübertragungsverfahren, in der die Eigenschaften des in dieser Arbeit vorgestellten Verfahrens grau unterlegt sind.

2.2.1 Verfahren

In der englischsprachigen Literatur wie beispielsweise in [62] werden zur Vorabübertragung von Informationen zwei *Verfahren* unterschieden: *Prefetching* und *Hoarding*. Prefetching-Verfahren zielen auf eine Reduzierung der Latenz ab, wobei aber immer eine Verbindung zu einem Netz angenommen wird. Eine ungeeignete Auswahl führt somit nicht automatisch dazu, dass eine angeforderte Information nicht angezeigt werden kann. Im Gegensatz hierzu unterstützen Hoarding-Verfahren den entkoppelten Betrieb und beugen somit eventuellen Netztrennungen vor. Eine detaillierte Abgrenzung dieser Verfahren ist in Kapi-

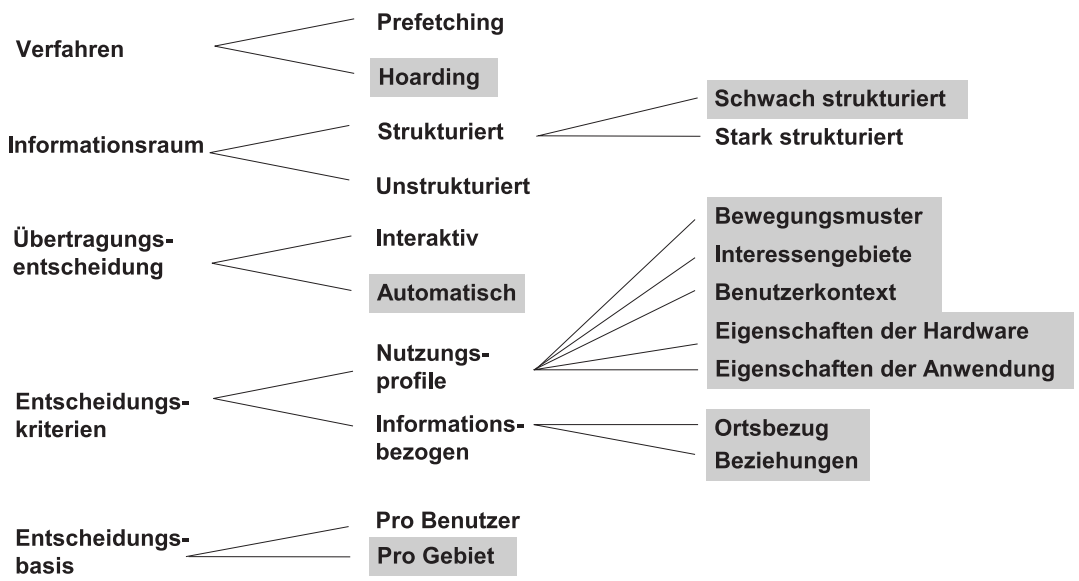


Abbildung 2.1: Taxonomie der Vorabübertragungsverfahren

tel 3.12 zu finden.

2.2.2 Informationsraum

Ein *Informationsraum* besteht aus einer Menge von Informationen und einer (auch leeren) Menge von Beziehungen zwischen diesen Informationen. Man unterscheidet unstrukturierte, schwach strukturierte und stark strukturierte Informationsräume.

Unstrukturierte Informationsräume: Die Informationen eines unstrukturierten Informationsraums stehen nicht miteinander in Beziehung und werden somit als isolierte Einheiten betrachtet. Sie weisen keinerlei Struktur auf, welche die für den entsprechenden Anwendungsfall notwendigen Informationen zur Ableitung von Beziehungen liefern könnte. Die Bedeutung unstrukturierter Informationen ist üblicherweise nur vom Menschen zu erfassen, d.h., eine automatische Interpre-

tation ist, wenn überhaupt, nur mit sehr viel Aufwand möglich. Ein Beispiel für unstrukturierte Informationen sind Prosatexte, die zwar eine gewisse Struktur aufweisen, wie beispielsweise Abschnitte oder Sätze, die Bedeutung des Inhalts kann jedoch nicht ohne weiteres von einer Anwendung erfasst werden. Im Fall der Vorabübertragung werden Informationen in unstrukturierten Informationsräumen als isolierte Einheiten betrachtet, weshalb in diesem Fall nur Aussagen über die Popularität der Informationen getroffen werden können, jedoch keine über das Zugriffsverhalten von Benutzern auf diese Informationen.

Stark strukturierte Informationsräume: Die Informationen eines stark strukturierten Informationsraums weisen eine von vornherein definierte, explizite Struktur auf. Diese Struktur und die Beziehungen zwischen den Informationen werden mit Hilfe eines formalen Datenmodells beschrieben. Eine wichtige Eigenschaft stark strukturierter Informationen ist es, dass sie vollständig sind in dem Sinn, dass keine der festgelegten Eigenschaften fehlt. Als Beispiel seien relationale Datenbanken angeführt. Im Fall der Vorabübertragung können mit Hilfe der Struktur der Informationen und ihrer Beziehungen Aussagen über das Zugriffsverhalten von Benutzern getroffen werden.

Schwach strukturierte Informationsräume: Die Informationen eines schwach strukturierten Informationsraums haben eine unregelmäßige, implizite Struktur, auf Grundlage derer Beziehungen zwischen den Informationen abgeleitet werden können. Im Vergleich zu stark strukturierten Informationsräumen, in denen die Struktur der Informationen fest vorgegeben ist, kann in schwach strukturierten Informationsräumen Strukturinformation nachträglich aus den Informationen selbst abgeleitet werden. Betrachten wir beispielsweise das Web als schwach strukturierten Informationsraum, so kann Strukturinformation durch die Auswertung der HTML-Tags einer Webseite gewonnen werden. So kann beispielsweise eine Beziehung zwischen denjenigen Webseiten bestehen, die durch Hyperlinks miteinander verknüpft sind. Stehen zusätzliche Informationen wie beispielsweise

der Zeitpunkt des Zugriffs zur Verfügung, kann die chronologische Aufrufreihenfolge als weitere Beziehung abgeleitet werden. Wenn wir nun die Prosatexte aus dem Beispiel für unstrukturierte Informationen nach bestimmten Kriterien klassifizieren, können wir eine Beziehung zwischen den Texten herstellen. In diesem Fall sind dann diese Texte als schwach strukturiert einzuordnen. Im Fall der Vorabübertragung können mit Hilfe der Struktur der Informationen, zusätzlich vorhandenen Informationen und den daraus abgeleiteten Beziehungen Aussagen über das Zugriffsverhalten von Benutzern getroffen werden.

2.2.3 Übertragungsentscheidung

Vorabübertragungsverfahren werden weiterhin dahingehend unterschieden, ob die *Entscheidung zur Vorabübertragung* automatisch erfolgt oder interaktiv vom Benutzer angestoßen und beeinflusst wird.

2.2.4 Entscheidungskriterien

Darüber hinaus werden Vorabübertragungsverfahren nach den *Kriterien* unterschieden, die für die Selektion der vorab zu übertragenden Informationen herangezogen werden.

Bei der *informationsbezogenen Selektion* spielt der *Ortsbezug* der Informationen vor allem in ortsbasierten Systemen eine wichtige Rolle. Dieser besagt, dass die Wahrscheinlichkeit, dass eine Information angefordert wird, vom Aufenthaltsort eines Benutzers abhängt. Folglich muss nicht der gesamte zugrunde liegende Informationsraum untersucht werden, da nur diejenigen Informationen betrachtet werden, die an einem bestimmten Ort relevant sind. Ein weiteres informationsbezogenes Kriterium ist die Eigenschaft des Informationsraums. In strukturierten Informationsräumen können zusätzlich *Beziehungen* zwischen den Informationen analysiert und ausgenutzt werden, aus denen dann deren semantische Di-

stanz [58] berechnet werden kann. Diese beschreibt das Maß für die Wahrscheinlichkeit, dass ein Zugriff auf eine Information den Zugriff auf die andere zur Folge hat.

Für Benutzergruppen mit ähnlichen Eigenschaften können zur Optimierung der Selektion *Nutzungsprofile* definiert werden. Wie in der Dissertation von Kubach [54] evaluiert wurde, kann die Berücksichtigung von *Bewegungsmustern* zu einer verbesserten Selektion der vorab zu übertragenden Informationen führen. So konnten die Trefferraten wesentlich erhöht werden, wenn die Route eines Benutzers bekannt war. *Interessengebiete* sind beispielsweise Vorlieben für Kultur- oder Sportinformationen, die entweder vom Benutzer angegeben werden oder durch eine Analyse des Inhalts der Informationen ermittelt werden muss. Steht ein Umgebungsmodell wie beispielsweise in NEXUS [44] zur Verfügung, kann der Kontext eines Benutzers daraus ermittelt und als Profilinformation verwendet werden. Beispielsweise könnte aus dem Kontext geschlossen werden, dass ein Benutzer zu Fuß in einem Einkaufszentrum unterwegs ist, woraus sich schließen lässt, dass er einen Einkaufsbummel macht. In diesem Fall könnten vorrangig Informationen über Sonderangebote oder besondere Aktionen für die Vorabübertragung selektiert werden. Eine *Hardware-Eigenschaft* ist zum Beispiel der Bildschirm eines mobilen Endgeräts. Diese Information kann zur Filterung der selektierten Informationen verwendet werden. So ergibt es beispielsweise wenig Sinn, hoch auflösende Grafiken auf einen PDA (*Personal Digital Assistant*) zu laden. Ebenso können abhängig von der *Eigenschaft der Anwendung* nur bestimmte Typen von Informationen selektiert werden. So würden beispielsweise auf Anforderung einer Navigationsanwendung vorrangig Kartendaten und aktuelle Verkehrssituationen selektiert werden.

2.2.5 Entscheidungsbasis

Schließlich werden Vorabübertragungsverfahren nach der grundsätzlichen *Entscheidungsbasis* für das Selektionsverfahren unterschieden. Bei einer *pro Benutzer*

getroffenen Auswahl wird ausschließlich das Zugriffsverhalten des momentanen Benutzers ausgewertet, um die Menge der vorab zu übertragenden Informationen zu bestimmen. Diese Auswahl ist sicherlich dann für den Benutzer befriedigend, wenn sein gegenwärtiges Zugriffsverhalten ähnlich dem vergangenen ist, also unabhängig vom aktuellen Aufenthaltsort. Im Vergleich hierzu wird bei einer *pro Gebiet* zu treffenden Auswahl das Zugriffsverhalten aller Benutzer bzw. Benutzergruppen in einem spezifizierten Gebiet analysiert. Die Selektion führt dann zu einem zufrieden stellenden Ergebnis, wenn das Zugriffsverhalten des aktuellen Benutzers ähnlich dem der Masse der in diesem Gebiet sich aufhaltenden Anwender bzw. dem der entsprechenden Benutzergruppen ist.

2.3 Einordnung des eigenen Verfahrens

In dieser Arbeit wird ein *automatisches Hoarding-Verfahren* vorgestellt, dessen zugrunde liegender Informationsraum *schwach strukturiert* ist. Es werden also implizit vorhandene Strukturen und zusätzlich vorhandenes Wissen ausgenutzt, um *Beziehungen* zwischen den Informationen abzuleiten und somit das Verfahren zu optimieren. Diese Beziehungen werden dann zusammen mit dem *Ortsbezug* der Informationen als Entscheidungskriterien verwendet. Hierbei wird angenommen, dass die Relevanz von Informationen vom jeweiligen Ort abhängig ist, woraus sich schließen lässt, dass das Zugriffsverhalten eines Benutzers von dessen Aufenthaltsort abhängt. Aus diesem Grund erfolgt die Selektion der vorab zu übertragenden Informationen *pro Gebiet*, d.h., zur Vorhersage der vorab zu ladenden Informationen wird das Zugriffsverhalten aller Benutzer bzw. Benutzergruppen in einem bestimmten Gebiet ausgewertet.

Nun haben nicht alle Benutzer das gleiche Zugriffsverhalten. Um die Selektion auch für unterschiedliche Zugriffsmuster zu optimieren, werden Nutzungsprofile als ein Entscheidungskriterium für die Selektion eingesetzt. Diese klassifizieren das Zugriffsverhalten auf der Grundlage von *Bewegungsmustern*, *Interessenge-*

bieten, *Benutzerkontext* sowie *Eigenschaften der Hardware und der Anwendung*. Das Profil eines Benutzers kann entweder von der Anwendung übermittelt werden oder auf Grundlage der Analyse des Zugriffsverhaltens dieses Benutzers abgeleitet werden. Solche Analysen wurden im Bereich des so genannten *Web usage mining* bereits nachhaltig erforscht (siehe u.a. [25, 35, 36, 50, 73, 74, 94]). Die Einordnung eines Benutzers in eine bestimmte Kategorie ist nicht Bestandteil dieser Dissertation, vielmehr wird angenommen, dass das Nutzungsprofil eines Benutzers bekannt ist. Ein Benutzer mit einem bestimmten Nutzungsprofil, der sich zum ersten Mal in einem Gebiet aufhält, bekommt somit eine Auswahl von denjenigen Informationen vorab auf sein Endgerät geladen, die von den meisten Benutzern der zugehörigen Benutzergruppe in diesem Gebiet angefordert werden.

Die Unterscheidung in Prefetching- und Hoarding-Verfahren wird nur noch in der Diskussion der verwandten Arbeiten in Kapitel 3.12 vorgenommen, um die Unterschiede zwischen beiden Verfahren deutlich zu machen. In den verbleibenden Kapiteln wird allgemein von *Vorabübertragungsverfahren* gesprochen.

3 Vorabübertragungsverfahren

Das nachfolgend vorgestellte Vorabübertragungsverfahren ist generisch hinsichtlich des zugrunde gelegten schwach strukturierten Informationsraums und wurde für den Zugriff auf das Web als konkreten Anwendungsfall spezialisiert und evaluiert.

In diesem Kapitel wird zunächst der grundsätzliche Ablauf des Verfahrens beschrieben und die sich hieraus ergebenden Problemstellungen diskutiert. Nach der Beschreibung des Systemmodells und der grundlegenden Annahmen folgt die Definition der generischen Konzepte, bevor Spezialisierungen für den Informationszugriff auf das Web detailliert beschrieben werden. Nach der Diskussion der qualitativen und quantitativen Eigenschaften des Verfahrens wird eine mögliche Erweiterung vorgeschlagen. Das Kapitel schließt mit der Diskussion verwandter Arbeiten und einer kurzen Zusammenfassung.

3.1 Übersicht über den Ablauf des Verfahrens

Wie bereits in Kapitel 1 erwähnt, wächst mit der zunehmenden Verbreitung immer leistungsfähigerer mobiler Endgeräte und der Entwicklung im Telekommunikationssektor auch das Bedürfnis, überall und zu jeder Zeit auf entfernte Informationen zugreifen zu können. Allein zelluläre Netze weisen jedoch erhebliche Nachteile auf wie mögliche Netztrennungen, hohe Latenz oder niedrige Bandbreiten, um nur einige zu nennen. Auf der anderen Seite gibt es bereits

an vielen Orten mit WLAN-Zugang ausgestattete Hotspots, an denen Benutzern ein schneller und oft sogar kostenfreier Zugang zum Internet zur Verfügung steht. Der schnelle Informationszugriff an den Hotspots kann nun dazu genutzt werden, um mit Hilfe eines geeigneten Vorabübertragungsverfahrens diejenigen Informationen in den lokalen Cache eines mobilen Endgeräts zu laden, auf die ein Benutzer zukünftig mit hoher Wahrscheinlichkeit zugreifen wird.

Im weiteren Verlauf dieser Arbeit wird ein Hotspot, der die Vorabübertragung von Informationen anbietet, als *Infostation* bezeichnet. Eine Infostation sammelt Daten hinsichtlich des Zugriffsverhaltens aller Benutzer, die sich in ihrem Dienstgebiet bewegen, und benutzt dieses Wissen zur Vorhersage, auf welche Informationen ein Benutzer in diesem Gebiet zukünftig mit hoher Wahrscheinlichkeit zugreifen wird. Zur Modellierung des Zugriffsverhaltens eines Benutzers werden aus diesem Grund alle Informationszugriffe des Benutzers protokolliert.

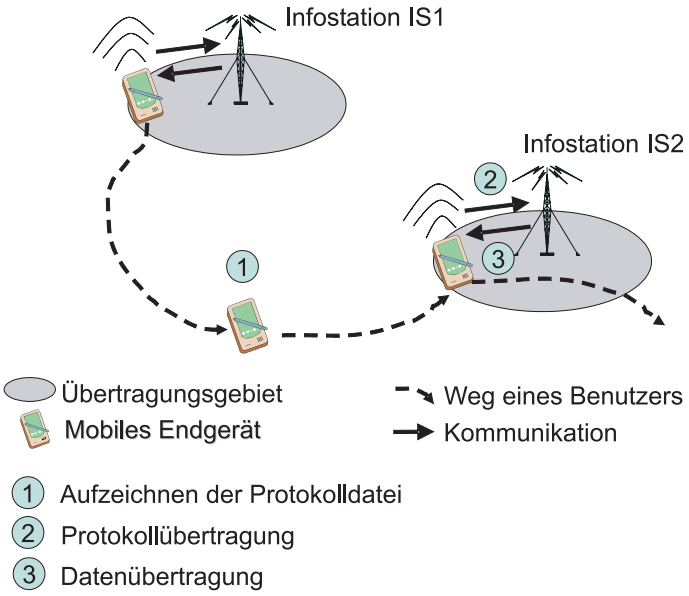


Abbildung 3.1: Ablauf der Vorabübertragung aus Sicht eines mobilen Benutzers

Aus Benutzersicht besteht der Ablauf des Vorabübertragungsverfahren aus den drei Phasen „Aufzeichnen der Protokolldatei", „Protokollübertragung" und „Informationsübertragung", die zyklisch verlaufen. Abbildung 3.1 zeigt diese bei-

spielhaft für einen Benutzer, der mit seinem mobilen Endgerät in einem Gebiet unterwegs ist, in dem zwei Infostationen *IS1* und *IS2* platziert wurden. Die Übertragungsgebiete der jeweiligen WLAN-Zugangspunkte (engl. *access point*) sind durch graue Ellipsen gekennzeichnet, die jeweiligen Phasen durch die eingekreisten Ziffern 1 bis 3. Wir nehmen nun an, dass an *IS1* Informationen vorab in den lokalen Cache des mobilen Endgeräts geladen wurden. Der Benutzer befindet sich auf dem Weg von *IS1* zu *IS2* in der Phase „Aufzeichnen der Protokolldatei", gekennzeichnet durch die eingekreiste Ziffer 1. Die drei Phasen verlaufen wie nachfolgend beschrieben.

1. In der Phase „Aufzeichnen der Protokolldatei" werden alle lokal ausgeführten Informationszugriffe des Benutzers mitsamt den zur Ableitung von Beziehungen erforderlichen Zusatzinformationen protokolliert. Dies erfolgt unabhängig davon, ob die Anfragen aus dem lokalen Cache des mobilen Endgeräts beantwortet werden konnten oder über die Verbindung zu einem Netz geladen wurden. Der Cache enthält die an der zuletzt besuchten Infostation (im Beispiel *IS1*) geladenen Informationen. Bei jeder Informationsanforderung des Benutzers wird geprüft, ob sich die betreffende Information im Cache befindet. Falls dies so ist, wird sie direkt angezeigt. Falls sich die Information nicht im Cache befindet, wird ein Cache-Fehler (engl. *cache miss*) gemeldet. Befindet sich der Benutzer in diesem Fall im Übertragungsgebiet des WLAN-Zugangspunkts einer Infostation, wird die Information über das WLAN geladen, außerhalb des Übertragungsgebiets kann bei Bedarf eine bestehende WWAN-Verbindung verwendet werden.

2. Betritt der Benutzer das Übertragungsgebiet des WLAN-Zugangspunkts einer Infostation (im Beispiel *IS2*), beginnt die Phase „Protokollübertragung", und die auf dem mobilen Endgerät erstellte Protokolldatei wird an die Infostation gesendet.

3. Ist die Übertragung der Protokolldatei abgeschlossen, beginnt die Phase „Informationsübertragung" und die Infostation lädt im Gegenzug diejenigen Informationen in den lokalen Cache des mobilen Endgeräts, die in

ihrem Dienstgebiet zum aktuellen Zeitpunkt relevant sind und auf die der Benutzer auf seinem weiteren Weg mit hoher Wahrscheinlichkeit zugreifen wird. Sobald die Übertragung beendet ist, endet diese Phase und der Zyklus beginnt erneut mit der Phase „Aufzeichnen der Protokolldatei".

Aus der Sicht einer Infostation verläuft das Vorabübertragungsverfahren wie folgt: Empfängt eine Infostation die Protokolldatei eines Benutzers, so analysiert sie diese, um damit Informationen hinsichtlich des Zugriffsverhaltens des entsprechenden Benutzers zu erhalten. Mit diesen Informationen aktualisiert sie ihr Wissen über das kollektive Zugriffsverhalten aller Benutzer bzw. Benutzergruppen, die sich in ihrem Dienstgebiet bewegen. Basierend auf diesem Wissen selektiert sie die in ihrem Dienstgebiet zum aktuellen Zeitpunkt relevanten Informationen und lädt diese in den Cache des Endgeräts des aktuellen Benutzers.

3.2 Problemstellungen

In diesem Abschnitt werden die Problemstellungen erläutert, die sich aus den einzelnen Schritten des vorgestellten Verfahrens ergeben.

Protokolldatei: Wie bereits erwähnt, sammelt eine Infostation Daten hinsichtlich des Zugriffsverhaltens von Benutzern. Hieraus ergibt sich die Problemstellung, das Zugriffsverhalten eines Benutzers in geeigneter Art und Weise zu modellieren. Da in schwach strukturierten Informationsräumen keine expliziten Beziehungen zwischen den Informationen definiert sind, mittels derer das Zugriffsverhalten beschrieben werden kann, müssen speziell für das Navigationsverhalten von Benutzern im Web entsprechende *Beziehungen* definiert und gegebenenfalls notwendige Zusatzinformationen bestimmt werden, die nicht aus der Struktur der Webseiten gewonnen werden können. Des Weiteren muss eine generische *Protokolldatei* spezifiziert werden, die alle für die Modellierung des Zugriffsverhaltens eines Benutzers notwendigen Angaben enthält. Diese ist schließlich für

Zugriffe auf das Web zu spezialisieren.

Bilden von Sitzungen: Da nicht unbedingt alle in einer Protokolldatei vermerkten Informationszugriffe miteinander in Beziehung stehen, ist es erforderlich, die Protokolldatei basierend auf den definierten Beziehungen in zusammengehörige Teile, so genannte *Sitzungen*, zu zerlegen. So definieren beispielsweise Huang et al. in [45] eine Sitzung als eine Gruppe von Zugriffen auf Webseiten, die ein Benutzer mit einer bestimmten Zielsetzung angefordert hat. Zu diesem Zweck ist ein geeignetes generisches *Sitzungskonzept* zu entwerfen, das schließlich für das Navigationsverhalten von Benutzern im Web spezialisiert wird.

Modellierung des Wissens über das kollektive Zugriffsverhalten von Benutzern bzw. Benutzergruppen: Das Beobachten des kollektiven Zugriffsverhaltens von Benutzern bzw. Benutzergruppen hat den Vorteil, dass überwiegend diejenigen Informationen vorab geladen werden, die im jeweiligen Gebiet zum jeweiligen Zeitpunkt populär sind. Zur Modellierung des Wissens über dieses kollektive Zugriffsverhalten ist eine *Datenstruktur* zu spezifizieren, mittels derer die Informationszugriffe der Benutzer bzw. Benutzergruppen effizient verwaltet und gemäß der definierten Beziehungen miteinander verknüpft werden können. Insbesondere ist darauf zu achten, dass das mit Hilfe von Sitzungen beschriebene Zugriffsverhalten abgebildet werden kann.

Weiterhin sind geeignete Funktionen zur Beschreibung einzelner Parameter des kollektiven Zugriffsverhaltens zu definieren, wie beispielsweise die Anzahl der Zugriffe auf eine bestimmte Information. Zur Messung der Stärke von Beziehungen sind geeignete Metriken zur Bestimmung der semantischen Nähe von Informationen zu erstellen. Schließlich ist ein Verfahren zu entwickeln, mit dessen Hilfe das erlangte Wissen dynamisch an sich änderndes Zugriffsverhalten angepasst werden kann.

Die generische Datenstruktur ist für Webseiten zu spezialisieren.

Clusterbildung Bisweilen können Informationen semantisch so stark zusammenhängen, dass sie für einen Benutzer nur als Gruppe interessant sind und es keinen Sinn ergibt, einzelne Objekte dieser Gruppe isoliert von den anderen zu übertragen. In diesem Fall müssen *Cluster* gebildet werden, die vollständig geladen werden.

Ein Beispiel aus dem Bereich des Webs soll dies verdeutlichen. Wie Cooley et al. in [25] und Pierrakos et al. in [83] feststellen, setzen Benutzer manche Webseiten als reines Navigationsmittel ein, um mit Hilfe der dort vorhandenen Hyperlinks die sie interessierenden Seiten zu finden. Die zur Navigation verwendeten Webseiten werden nachfolgend der besseren Lesbarkeit halber als *Transitseiten* und die Seiten mit dem interessierenden Inhalt als *Inhaltsseiten* bezeichnet. Wird nun bei der Vorabübertragung nicht die gesamte Folge von zusammengehörigen Transit- und Inhaltsseiten in den Cache geladen, sondern nur ein Teil davon, steht irgendwann im Verlauf der Navigation eine Webseite nicht zur Verfügung. Somit kann der Benutzer seine Suche zumindest auf diesem Pfad nicht abschließen, was für ihn natürlich nicht zufriedenstellend ist. Ein weiterer Nachteil ist, dass durch die umsonst übertragenen Seiten wertvolle Ressourcen wie beispielsweise Speicherplatz oder Energie verschwendet werden. Sinnvoller ist es also, entweder die komplette Folge von Webseiten zu laden oder gar keine Seite.

Für die Clusterbildung sind speziell für Webseiten geeignete *Beziehungen* zu definieren, sowie eine *Metrik* zur Messung der semantischen Nähe zu bestimmen.

Auswahl relevanter Informationen: Zunächst muss spezifiziert werden, nach welchen Leistungskriterien die Auswahl der vorab zu übertragenden Informationen speziell für Webseiten erfolgen soll. Zur Bestimmung der relevanten Informationen müssen effiziente *Auswahlverfahren* mit und ohne Clusterbildung zur Verfügung gestellt werden, die speziell auf die Vorabübertragung von Webseiten zugeschnitten sind. Insbesondere ist eine Metrik zur *Bestimmung der Relevanz* einer Webseite bzw. eines Clusters zu erstellen, die den Auswahlverfahren als Ordnungskriterium dient.

Nutzungsprofile: Sicherlich resultiert die Auswertung des Zugriffsverhaltens einer Menge von Benutzern nicht immer in einer optimalen Entscheidung für einen individuellen Benutzer. Dieser Effekt kann jedoch verringert werden, wenn Nutzungsprofile in die Entscheidung mit einbezogen werden, welche der Informationen vorab zu laden sind. Für diesen Fall muss ein geeignetes *Konzept für die Einbeziehung von Nutzungsprofilen* erstellt werden. Weiterhin muss die oben beschriebene generische Datenstruktur zur Modellierung des kollektiven Zugriffsverhaltens angepasst werden, um die Nutzungsprofile möglichst effizient in das Verfahren zu integrieren. Schließlich sind effiziente Verfahren zur Auswahl relevanter Informationen sowie eine Metrik zur Bestimmung der Relevanz einer Webseite bzw. eines Clusters für jedes Nutzungsprofil zu erstellen.

3.3 Systemmodell und Annahmen

Das in Abbildung 3.2 dargestellte Systemmodell, das dieser Arbeit als Grundlage dient, besteht aus drei Hauptkomponenten: dem Informationsraum, der Infostation und dem mobilen Endgerät.

Weiterhin wird angenommen, dass es Benutzergruppen mit ähnlichen Interessen gibt, die einem oder mehreren Nutzungsprofilen zugeordnet werden können.

3.3.1 Informationsraum

Dem vorgestellten Vorabübertragungsverfahren liegt ein schwach strukturierter Informationsraum zugrunde, dessen Eigenschaften in Abschnitt 2.2.2 beschrieben wurden. Solch ein Informationsraum, wie beispielsweise das Web, kann typischerweise sehr groß sein. Das Vorabübertragungsverfahren nutzt jedoch den Ortsbezug der Informationen aus, der besagt, dass die Wahrscheinlichkeit des Zugriffs auf eine Information vom jeweiligen Ort abhängt. Es handelt sich somit um ortsabhängige Informationszugriffe, die wie folgt definiert sind:

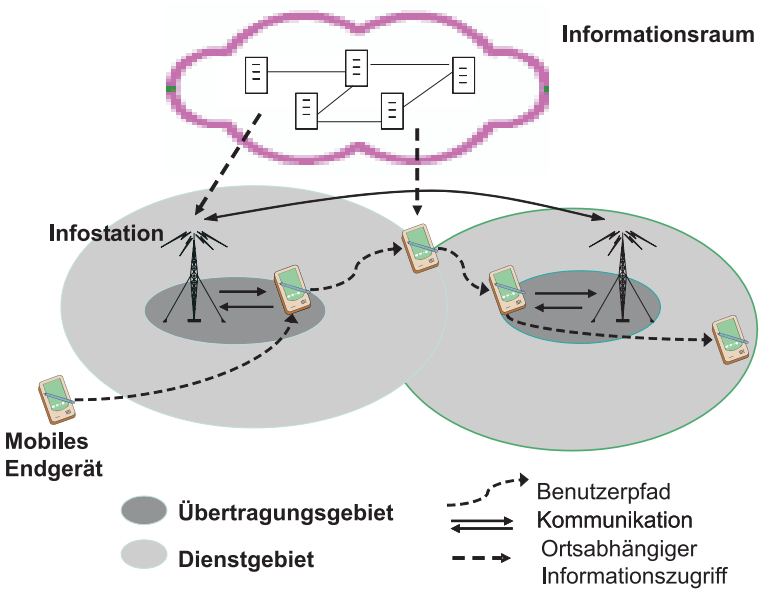


Abbildung 3.2: Systemmodell

Definition 2 (Ortsabhängiger Informationszugriff) *Ein Zugriff auf eine Information ist ein* ***ortsabhängiger Informationszugriff****, wenn die Wahrscheinlichkeit, dass ein Benutzer diese Information anfordert, von dessen Aufenthaltsort abhängt.*

Bei ortsabhängigen Informationszugriffen kann man folglich davon ausgehen, dass sich zum einen die Anfragen in einem geographisch begrenzten Gebiet auf eine stark begrenzte Teilmenge des Informationsraums konzentrieren. Zum andern werden Benutzer bzw. Benutzergruppen in diesem Gebiet ein ähnliches Informationsbedürfnis aufweisen.

Ein Beispiel hierfür ist ein elektronischer Web-basierter Touristenführer, der Benutzern Informationen beispielsweise über eine Stadt anbietet, die mit Hilfe eines Browsers angefordert werden können. Benutzer können in diesem Fall zwar potenziell auf jede beliebige Webseite zugreifen, es ist jedoch anzunehmen, dass überwiegend solche Informationen angefordert werden, die sich auf Informationen in ihrer Umgebung beziehen.

Je höher der Grad der Ortsabhängigkeit von Informationszugriffen ist, desto effektiver wird erwartungsgemäß das Vorabübertragungsverfahren arbeiten.

3.3.2 Infostation

Eine Infostation kann man sich vorstellen als eine Insel mit sehr guter drahtloser Netzanbindung in Gebieten mit ansonsten niedriger Bandbreite wie beispielsweise in zellulären Netzen. Das Konzept einer Infostation ist nicht neu. Bereits im Jahre 1996 wurde der Begriff *Infostation* im Rahmen des NIMBLE-Projekts der Rutgers-University von Thomasz Imielinski geprägt [5]. Infostationen wurden auch in den Vorabübertragungsverfahren der Projekte Map-on-the-Move [104] und GUIDE [21] eingesetzt.

Die in dieser Arbeit verwendete Infostations-Architektur und die Inter-Infostations-Kommunikation basieren auf dem in der Dissertation von Kubach [54] vorgestellten Konzept und werden hier deshalb nur kurz eingeführt. Der Schwerpunkt dieser Arbeit liegt auf dem Auswahlverfahren, mit dessen Hilfe die vorab zu übertragenden Informationen selektiert werden.

Eine Infostation ist ein stationärer Server, der den Zugang für ein WLAN zur Verfügung stellt. Dadurch sind mobile Endgeräte in der Lage, innerhalb des *Übertragungsgebiets* des Zugangspunkts (engl. *access point*) mit der Infostation zu kommunizieren (siehe Abbildung 3.2).

Eine Infostation sammelt Informationen hinsichtlich des Zugriffsverhaltens aller Benutzer, die in ihrem *Dienstgebiet* Informationen anfordern. Basierend hierauf selektiert sie die Informationen mit der höchsten Relevanz und lädt sie auf das mobile Endgerät eines Benutzers. Das Dienstgebiet ist ein der Infostation zugeordnetes geographisch begrenztes Gebiet, das wesentlich größer als das Übertragungsgebiet ist. Jeder Infostation ist genau ein Dienstgebiet zugeordnet.

Des Weiteren ist eine Infostation mit einem leistungsfähigen Zugang zum Internet ausgestattet. Diese Verbindung wird zum einen benötigt, um die Informationen

für die Vorabübertragung auf die Infostation zu laden. Zum andern kommunizieren Infostationen miteinander, um ihr Wissen über Informationsanforderungen auszutauschen. Die Kommunikationsarchitektur ist in Abbildung 3.3 veranschaulicht.

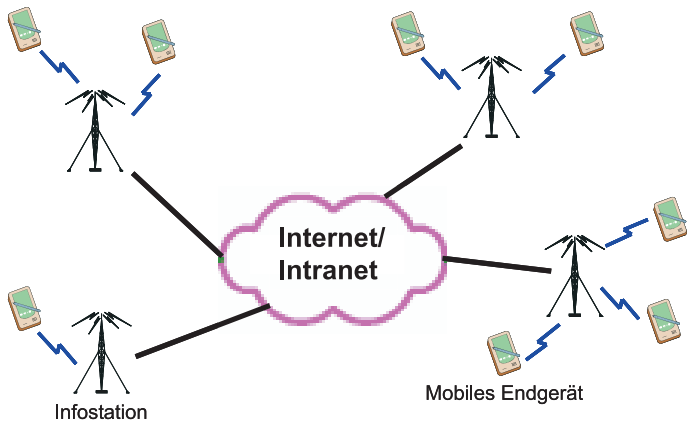


Abbildung 3.3: Kommunikationsinfrastruktur

Wie man in Abbildung 3.4 beispielhaft sehen kann, können mehrere Infostationen ein größeres Gebiet wie beispielsweise ein Stadtzentrum versorgen. Deren WLAN-Zugangspunkte sind in der Abbildung durch die Kreise um die Infostationen markiert. Sie sollten möglichst an solchen Orten platziert werden, die von Benutzern häufig besucht werden, so dass zum Erreichen einer Infostation keine Umwege in Kauf genommen werden müssen. In diesem Fall ist es sinnvoll, Besucher auf ihrem Weg zwischen den Übertragungsgebieten möglichst lückenlos mit Informationen zu versorgen. Aus diesem Grund sollte das Dienstgebiet einer Infostation mit dem Übertragungsgebiet mindestens einer benachbarten Infostation überlappen. Dies kann anschaulich mit Hilfe eines Beispiels begründet werden, in dem ein Benutzer zwischen den Infostation *IS1* und *IS2* unterwegs ist. In Abbildung 3.5(a) ist das Dienstgebiet *DG1* von *IS1* disjunkt zum Übertragungsgebiet von *IS2*. Verlässt der Benutzer *DG1* im Punkt *P*, so muss er den restlichen Weg zum Übertragungsgebiet von *IS2* zurücklegen, ohne die in Dienstgebiet *DG2* relevanten Informationen im Cache gespeichert zu haben. Überlappen sich jedoch

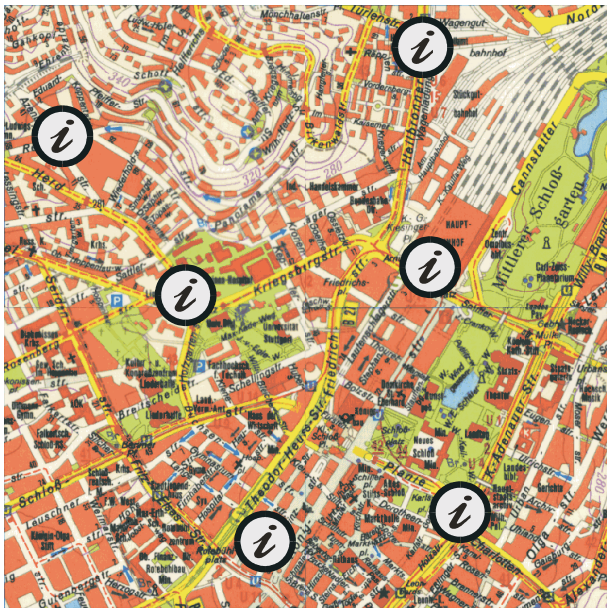
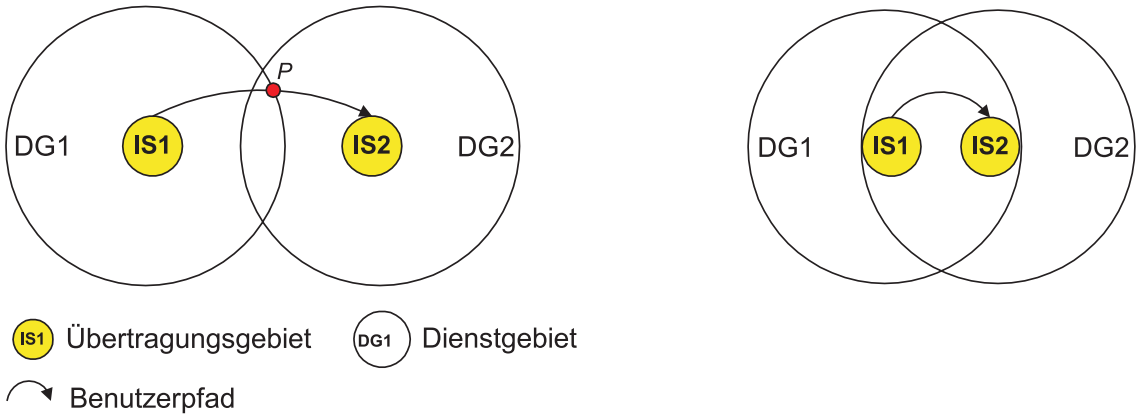


Abbildung 3.4: In einem Stadtzentrum verteilte Infostationen (Quelle: Kubach [55])

die beiden Gebiete wie in Abbildung 3.5(b), tritt dieser Fall nicht auf. Werden anschließend im Übertragungsgebiet von *IS2* die in *DG2* relevanten Informationen geladen, sollten bereits im Cache befindliche Informationen nicht nochmals geladen werden.

Es wird angenommen, dass der Zugriff auf die innerhalb eines Dienstgebiets angeforderten Informationen ortsabhängig ist. Infolgedessen werden sich die Anfragen im Dienstgebiet einer Infostation auf eine stark begrenzte Teilmenge des zugrunde liegenden Informationsraums konzentrieren. Diese Teilmenge kann sich über die Zeit ändern, da sie vom Zugriffsverhalten der Benutzer abhängt, die sich in diesem Dienstgebiet bewegen.

Zum Informationsaustausch zwischen Infostationen wird ein externer *Verzeichnisdienst* benötigt, der jede Infostation und ihr zugeordnetes Dienstgebiet kennt. Auf Anfrage liefert er zu einer gegebenen geographischen Position eine Liste mit allen zuständigen Infostationen.



(a) Zu Übertragungsgebieten disjunkte Dienstgebiete (b) Mit Übertragungsgebieten überlappende Dienstgebiete

Abbildung 3.5: Dienstgebiete benachbarter Infostationen

3.3.3 Mobile Endgeräte

Mobile Endgeräte, wie beispielsweise ein PDA oder ein Smartphone, sind mit einer WLAN-Schnittstelle ausgestattet. Eine zusätzliche Schnittstelle zur drahtlosen Kommunikation mit einem zellulären Netz ist optional. Mobile Endgeräte befinden sich in einem von drei Verbindungszuständen: *verbunden, schwach verbunden* und *nicht verbunden*. Ein mobiles Endgerät ist verbunden, wenn es sich im Übertragungsgebiet einer Infostation aufhält und Verbindung zum WLAN hat. Außerhalb des Übertragungsgebiets ist es schwach verbunden, wenn eine WWAN-Verbindung existiert. Es ist nicht verbunden, wenn kein Netz zur Verfügung steht.

Mobile Benutzer greifen über eine *Benutzeranwendung* wie beispielsweise einen Browser oder einen elektronischen Touristenführer auf Informationen zu. Diese Benutzeranwendung verwendet die *Cache-Verwaltung* des Vorabübertragungsverfahrens zur Optimierung des mobilen Informationszugriffs.

Ein mobiles Endgerät besitzt eine physische Uhr, die Uhren unterschiedlicher Endgeräte müssen nicht synchronisiert sein.

Neben dem ebenso von Infostationen benötigten *Verzeichnisdienst* stehen mobilen Endgeräten folgende externen Dienste zur Verfügung:

Positionsbestimmungssystem: Um Informationszugriffe einem bestimmten Dienstgebiet zuordnen zu können, muss jedes Endgerät in der Lage sein, seine aktuelle Position zu bestimmen. Dazu können unterschiedliche Dienste eingesetzt werden wie beispielsweise GPS, AGPS (*Assisted Global Positioning System*) oder zukünftig Galileo. Sind die Dienstgebiete als eine Menge von Funkzellen definiert, so genügen im Prinzip auch Zell-IDs zur Positionsbestimmung. In diesem Fall ist aber eine Schnittstelle zum WWAN zwingend erforderlich.

Ereignisdienst: Die Cache-Verwaltung muss benachrichtigt werden, wenn ein Benutzer mit dem mobilen Endgerät das Übertragungsgebiet einer Infostation betritt. Hierzu kann sich die Cache-Verwaltung beim Ereignisdienst für dieses Ereignis registrieren lassen. Kommt dann das Endgerät in das Übertragungsgebiet einer Infostation, wird die Cache-Verwaltung vom Ereignisdienst automatisch darüber benachrichtigt.

3.4 Generische Konzepte

In diesem Abschnitt werden die generischen Konzepte vorgestellt, die für alle Spezialisierungen des Verfahrens gültig sind:

- Die *Protokolldatei* modelliert das Zugriffsverhalten eines Benutzers in einem schwach strukturierten Informationsraum, das auf den zu bestimmenden Relationen basiert. Der generische Teil einer Protokolldatei enthält diejenigen Angaben, die für alle schwach strukturierten Informationsräume gültig sind. Zusätzliche Angaben, die zur Modellierung des Zugriffsverhaltens in einem bestimmten Informationsraum geeignet sind, müssen für die konkreten Anwendungen spezialisiert werden.

- Eine *Relation* beschreibt die Beziehungen zwischen den Zugriffen auf schwach strukturierte Informationen. Sie wird in diesem Abschnitt formal definiert und muss für konkrete Anwendungen abhängig vom zugrunde liegenden Informationsraum spezialisiert werden.
- Eine *Sitzung* ist eine Teilmenge der Protokolldatei, die miteinander in Beziehung stehende Einträge einer Protokolldatei enthält. Die in diesem Abschnitt vorgestellte Definition einer Sitzung gilt für alle zugrunde liegenden Informationsräume und muss nicht spezialisiert werden.
- Der *Informationsgraph* ist eine Datenstruktur, mit deren Hilfe das Wissen über das kollektive Zugriffsverhalten aller Benutzer im Dienstgebiet der Infostation modelliert wird. Der Informationsgraph muss für konkrete Anwendungen spezialisiert werden.
- Mit Hilfe von *Nutzungsprofilen* werden Informationsbedürfnisse von Benutzern klassifiziert. Basierend hierauf kann ein Benutzer einer Benutzergruppe zugeordnet werden. Das Konzept eines Nutzungsprofils gilt für alle Arten von schwach strukturierten Informationsräumen und Benutzeranwendungen.

Der Abschnitt endet mit einem Überblick über das generische Vorabübertragungsverfahren.

3.4.1 Protokolldatei

Mit Hilfe einer Protokolldatei wird das Zugriffsverhalten eines Benutzers in einem schwach strukturierten Informationsraum modelliert. Eine Protokolldatei ist eine Sequenz von Protokolleinträgen, wobei für jeden Zugriff auf eine Information ein Eintrag erstellt wird, der alle für die Modellierung notwendigen Angaben enthält. Nachfolgend werden *Kontrolleinträge* und *Dateneinträge* unterschieden. Kontrolleinträge werden dazu benutzt, um spezielle Hinweise der Benutzeranwendung über eingetretene Ereignisse an die Infostation zu übermitteln, wie

beispielsweise Angaben über das Nutzungsprofil eines Benutzers. Dateneinträge beschreiben die Zugriffe auf Informationen.

Jeder Protokolleintrag hat mindestens die folgenden vier Attribute:

l.Typ: Typ des Eintrags (Kontroll- oder Dateneintrag);

l.ID: Bezeichner des Eintrags, abhängig von dessen Typ;

l.Ort: Ort, an dem der Eintrag protokolliert wurde, wie beispielsweise die Koordinaten x und y eines geographischen Koordinatensystems;

l.Zeit: Zeitpunkt, zu dem der Eintrag protokolliert wurde.

Der Ort und der Zeitpunkt des Eintrags sind zusätzliche Informationen, die für die Ableitung von räumlichen bzw. zeitlichen Beziehungen zwischen den angeforderten Informationen notwendig sind.

Für **Kontrolleinträge** wird das Attribut **l.Typ = „Anw"** gesetzt. Nachfolgend werden drei Kontrolleinträge und die jeweils notwendigen Attribute spezifiziert:

1. Das Öffnen der Benutzeranwendung signalisiert, dass der Benutzer beginnt, Informationen anzufordern.

   l.ID = „öffne": beschreibt das Ereignis „Benutzeranwendung wurde geöffnet"

   l.Name: Name der Benutzeranwendung

   l.Größe: Maximale Größe des lokalen Caches in MByte

2. Das Schließen der Benutzeranwendung signalisiert, dass der Benutzer aufgehört hat, Informationen anzufordern.

   l.ID = „schließe": beschreibt das Ereignis „Benutzeranwendung wurde geschlossen"

   l.Name: Name der Benutzeranwendung

3. Die Benutzeranwendung legt den Profilmix fest, mit dem der Benutzer

momentan Informationen anfordert. Die Definition eines Profilmixes erfolgt in Abschnitt 3.4.5. Dieser Eintrag erfolgt immer dann, wenn entweder die Benutzeranwendung gestartet wurde oder der Benutzer das Profil ändert.

l.ID = „Profil": beschreibt das Ereignis „Profilmix wurde festgelegt"

l.Profil $= (n_1; P(n_1)), \ldots, (n_b; P(n_b))$**:** Neuer Profilmix, der sich aus unterschiedlichen Nutzungsprofilen n_i mit den zugehörigen Zugriffswahrscheinlichkeiten $P(n_i)$ zusammen setzen kann

Ein **Dateneintrag** hat folgende Attribute:

l.Typ = „Anf": Es handelt sich um einen Dateneintrag

l.ID: Global eindeutiger Bezeichner der Informationsanforderung, z.B. die URL einer Webseite;

l.Größe: Größe der Information in Byte;

Definition 3 (Protokolldatei) *Eine* ***Protokolldatei***

$$\mathcal{L} = \{l_1, \ldots, l_i, \ldots, l_k, \ldots l_n\}$$

ist eine total geordnete Menge von Protokolleinträgen mit der Ordnungsrelation

$$l_i < l_k \Leftrightarrow l_i.\texttt{Zeit} < l_k.\texttt{Zeit}$$

und den folgenden Funktionen:
$\texttt{min}(\mathcal{L})$ *und* $\texttt{max}(\mathcal{L})$ *stellen den Protokolleintrag mit dem minimalen bzw. maximalen Zeitpunkt dar.*
$\texttt{pre}(l_j, \mathcal{L})$ *und* $\texttt{suc}(l_j, \mathcal{L})$ *repräsentieren den unmittelbaren Vorgänger bzw. Nachfolger von* l_j*.*

Tabelle 3.1 zeigt den Teil einer Protokolldatei, die in dieser Arbeit als laufendes Beispiel verwendet wird. Die Webseiten wurden an zwei unterschiedlichen Orten

Tabelle 3.1: Beispiel-Protokolldatei

Typ	**ID**	**Ort**	**Zeit**	**Größe [Byte]**	**Name / Profil**
Anw	öffne	(5,0|7,0)	2006-06-20/11:59:50	20M	Browser XYZ
Anw	Profil	(5,0|7,0)	2006-06-20/11:59:51	-	$(n1; 0{,}7)$, $(n2; 0{,}3)$
Anf	*A*	(5,0|7,0)	2006-06-20/12:00:00	1512	-
Anf	*B*	(5,0|7,0)	2006-06-20/12:00:10	1432	-
Anf	*D*	(5,0|7,0)	2006-06-20/12:00:30	1510	-
Anf	*E*	(1,0|2,0)	2006-06-20/12:16:30	2312	-
Anf	*F*	(1,0|2,0)	2006-06-20/12:16:40	1460	-
Anf	*T*	(1,0|2,0)	2006-06-20/12:17:55	1643	-
Anf	*F*	(1,0|2,0)	2006-06-20/12:18:10	1856	-
Anf	*J*	(1,0|2,0)	2006-06-20/12:18:20	1746	-
Anf	*I*	(1,0|2,0)	2006-06-20/12:18:35	2684	-
Anw	schließe	(5,0|7,0)	2006-06-20/12:25:35	-	Browser XYZ

(5,0|7,0) und (1,0|2,0) aufgerufen, die als x- und y-Koordinaten eines Koordinatensystems angegeben sind. Die ersten beiden Einträge sind Kontrolleinträge, die am Ort mit den Koordinaten (5,0|7,0) am 20.6.2005 um 11:59:50 Uhr bzw. 11:59:51 Uhr protokolliert wurden. Sie signalisieren das Öffnen des Browsers XYZ und zeigen den Profilmix an, der zu 70% aus Nutzungsprofil *n1* und zu 30% aus Nutzungsprofil *n2* besteht. Im dritten Eintrag wurde am gleichen Ort die Webseite *A* angefordert, deren Größe 1512 Bytes beträgt.

3.4.2 Relationen zwischen Protokolleinträgen

Wie bereits in Abschnitt 2.2.2 erwähnt, nutzt das vorgestellte Verfahren Beziehungen zwischen schwach strukturierten Informationen zur Selektion der vorab zu übertragenden Informationen aus. Diese Beziehungen werden aus den Relationen zwischen Protokolleinträgen abgeleitet. Zwei Beispiele aus dem Bereich

des Webs sollen dies verdeutlichen.

(1) Eine zeitliche Relation $R_{\texttt{Zeit}}$ zwischen zwei direkt aufeinander folgenden Protokolleinträgen l_j und l_{j+1} besteht beispielsweise dann, wenn es sich bei beiden Einträgen um Dateneinträge handelt und die Zeitpunkte der Einträge nicht mehr als eine bestimmte Zeitspanne Δt auseinander liegen.

(2) Eine aus der Struktur von Webseiten abgeleitete Relation $R_{\texttt{Link}}$ zwischen zwei direkt aufeinander folgenden Protokolleinträgen l_j und l_{j+1} besteht beispielsweise dann, wenn es sich bei beiden Einträgen um Dateneinträge handelt und die angeforderten Webseiten durch einen Hyperlink miteinander verknüpft sind.

Die in dieser Arbeit verwendete Notation für eine Relation $R = \mathcal{L} \times \mathcal{L}$ zwischen zwei Einträgen l und l' einer Protokolldatei ist $l\,R\,l'$. Besteht eine solche Relation nicht, wird dies mit $\neg(l\,R\,l')$ gekennzeichnet.

Weiterhin können auch zusammengesetzte Relationen definiert werden. So könnte beispielsweise eine Relation zwischen zwei Protokolleinträgen nur dann bestehen, wenn beide oben angeführten Beispielrelationen $R_{\texttt{Link}}$ und $R_{\texttt{Zeit}}$ gelten. Allgemein wird eine Relation zwischen zwei Protokolleinträgen deshalb wie folgt definiert:

Definition 4 (Relation) *Seien R und R' Relationen zwischen zwei Protokolleinträgen. Dann gilt:*

$$
\begin{aligned}
R &\textit{ ist eine Relation}\\
\neg R &\textit{ ist eine Relation}\\
R \wedge R' &\textit{ ist eine Relation}\\
R \vee R' &\textit{ ist eine Relation}\\
\textit{nichts sonst} &\textit{ ist eine Relation}
\end{aligned}
$$

Definition 5 (Transitive Hülle einer Relation) *Sei $\mathcal{L}$ eine Protokolldatei und $\mathcal{R}$ die Menge aller hierauf definierten Relationen. Dann wird die* ***transitive***

Hülle R^+ *einer Relation* $R \in \mathcal{R}$ *rekursiv definiert als:*

$$
\begin{aligned}
R^1 &= R = \{(l, l') \mid l, l' \in \mathcal{L} \wedge lRl'\} \\
R^2 &= RR = \{(l, l') \mid l, l' \in \mathcal{L} \wedge \exists\, l'' \in \mathcal{L} : lRl'' \wedge l''Rl'\} \\
R^n &= R^{n-1}R = \{(l, l') \mid l, l' \in \mathcal{L} \wedge \exists\, l'' \in \mathcal{L} : lR^{n-1}l'' \wedge l''Rl'\} \\
R^+ &= \bigcup_{i \in \mathbb{N}} R^i
\end{aligned}
$$

3.4.3 Sitzung

Benutzer verfolgen in der Regel ein bestimmtes Ziel, wenn sie Informationen anfordern (siehe beispielsweise [20]). Ein Beispiel hierfür ist die Literaturrecherche im Web, deren Ziel es ist, Informationen über ein bestimmtes Thema zu finden. Ein weiteres Beispiel ist der bereits erwähnte elektronische Touristenführer, mit dessen Hilfe sich Benutzer beispielsweise Informationen über die Sehenswürdigkeit anzeigen lassen, vor der sie gerade stehen. In diesem Fall wäre es das Ziel, die gewünschten Informationen über die Sehenswürdigkeit zu erhalten.

Nun kann das Fortschreiben einer Protokolldatei über einen größeren Zeitraum hinweg verlaufen, ohne dass der Benutzer zusammenhängend Informationen anfordert. So könnte beispielsweise ein Tourist eine Stadtbesichtigung in mehreren Etappen durchführen. Zwischen diesen Etappen liegen größere Zeitabschnitte, während derer keine Informationen angefordert wurden, beispielsweise weil der Benutzer längere Zeit zu der nächsten Sehenswürdigkeit unterwegs war oder weil er die Besichtigung erst am folgenden Tag fortgesetzt hat. In solchen Fällen besteht die Protokolldatei aus Einträgen, die nicht alle unmittelbar zusammenhängen, sondern jeweils nur diejenigen Teile davon, die innerhalb einer Etappe angefordert wurden.

Zur Modellierung des Zugriffsverhaltens eines Benutzers müssen folglich diejenigen Protokolleinträge identifiziert werden, die miteinander in Beziehung stehen. Zu diesem Zweck wird das Konzept einer *Sitzung* (engl. *session*) eingeführt.

Bezug nehmend auf das oben angeführte Beispiel des elektronischen Touristenführers würden einer Sitzung diejenigen Protokolleinträge zugeordnet, die sich auf die während einer Etappe angeforderten Informationen beziehen. Als hierfür maßgebliche Relation zur Ableitung einer Sitzung eignet sich die in Abschnitt 3.4.2 beispielhaft angeführte zeitliche Relation $R_{\texttt{Zeit}}$, die zwischen zwei Protokolleinträgen dann besteht, wenn die Zeitpunkte der Einträge nicht mehr als eine bestimmte Zeitspanne Δt auseinander liegen.

Allgemein ist eine Sitzung als diejenige Teilmenge einer Protokolldatei definiert, deren Einträge gemäß einer durch Definition 4 beschriebenen Relation R miteinander in Beziehung stehen. Die in dieser Arbeit verwendete Notation für eine Sitzung ist $S(s, R)$, wobei s der Protokolleintrag ist, der die Sitzung startet und nachfolgend *Sitzungsstarteintrag* genannt wird. R stellt die der Sitzung zugrunde liegende Relation dar. Als Beispiele für solch eine Relation im Bereich des Webs seien die beiden in Abschnitt 3.4.2 beschriebenen Relationen $R_{\texttt{Zeit}}$ und $R_{\texttt{Link}}$ angeführt.

Definition 6 (Sitzung, Sitzungsstarteintrag) *Sei R die Relation, auf deren Grundlage die Sitzung bestimmt wird und R^+ deren transitive Hülle gemäß Definition 5. Sei weiterhin $\mathcal{L}$ eine Protokolldatei. Dann wird die Menge aller* ***Sitzungsstarteinträge*** *definiert als*

$$\mathcal{B} = \{s \in \mathcal{L} \mid (s = min(\mathcal{L}) \vee (\exists h \in \mathcal{L} : h = pre(s, \mathcal{L}) \wedge \neg(hRs)))\}$$

Eine ***Sitzung*** $S(s, R) \subseteq \mathcal{L}$ *ist die Teilmenge der Protokolldatei, für die gilt:*

$$\begin{aligned} &s \in \mathcal{B} \textit{ ist Sitzungstarteintrag der Sitzung} \\ &\quad R \textit{ ist die für die Sitzung maßgebliche Relation} \\ S(s, R) &= \{s\} \cup \{l \in \mathcal{L}_{req} \mid (s, l) \in (sR^+l)\} \end{aligned}$$

Abbildung 3.6 zeigt beispielhaft die Ableitung von Sitzungen aus einer Proto-

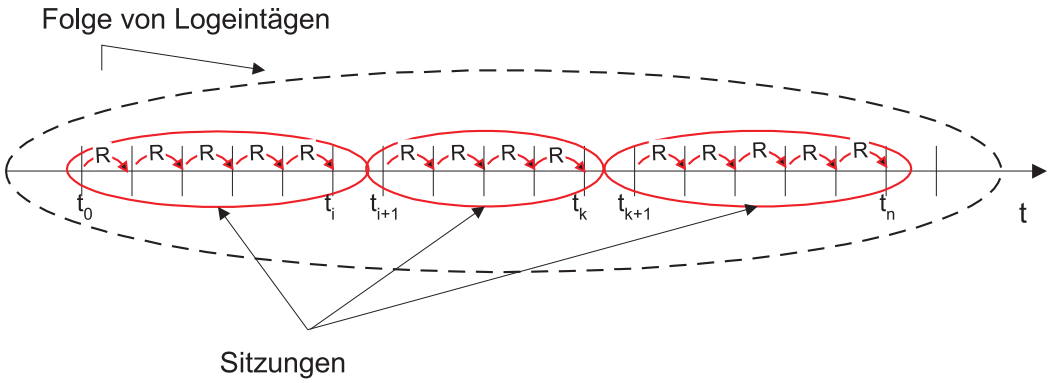


Abbildung 3.6: Aus einer Protokolldatei abgeleitete Sitzungen

kolldatei, die Einträge von Zeitpunkt t_0 bis t_n enthält. Für den Aufbau einer Sitzung wird die Relation R verwendet. Der besseren Lesbarkeit halber wird der zum Zeitpunkt t_i gehörende Logeintrag mit l_i bezeichnet. Wie der Abbildung zu entnehmen ist, startet l_0 die erste Sitzung. Die Einträge l_i und l_{i+1} sind nicht durch R verbunden, so dass Eintrag l_i die erste Sitzung beendet und l_{i+1} die zweite startet. Die restlichen Sitzungen $S(l_{i+1}, R)$ und $S(l_{k+1}, R)$ werden entsprechend aufgebaut.

Zur Modellierung des Zugriffsverhaltens von Benutzern ist es notwendig, nur solche Sitzungen zu betrachten, die vollständig sind in dem Sinn, dass der Benutzer sein Ziel im Rahmen der Informationssuche erreicht hat, so dass er diesbezüglich keine weiteren Informationen mehr anfordern wird. Handelt es sich beim letzten Eintrag in der Protokolldatei um einen Dateneintrag, so ist dies nicht ohne Weiters entscheidbar. Die entsprechende Sitzung kann folglich nicht als abgeschlossen bezeichnet werden. Handelt es sich in diesem Fall jedoch um einen der in Abschnitt 3.4.1 definierten Kontrolleinträge „Öffnen/Schließen der Anwendung", so kann man davon ausgehen, dass die Sitzung beendet wurde. Da die zugrunde liegende Relation vom Profil abhängen kann, gilt dies auch für den Kontrolleintrag „Änderung des Profils". Eine vollständige Sitzung wird wie folgt definiert:

Definition 7 (Vollständige Sitzung) *Sei $\mathcal{L}$ eine Protokolldatei, $S(s,R)$ eine daraus abgeleitete Sitzung und $l_n \in \mathcal{L}$ der die Sitzung $S(s,R)$ beendende Protokolleintrag. Dann ist $S(s,R)$ eine* ***vollständige Sitzung****, wenn gilt:*

$$(l_n \neq \texttt{max}(\mathcal{L}) \ \wedge \ \neg(l_n \, R \, suc(l_n, \mathcal{L}))) \vee$$
$$(l_n = \texttt{max}(\mathcal{L}) \ \wedge \ l_n.\texttt{Typ} = \text{„}Anw\text{”})$$

Tabelle 3.2: Auszug aus der Beispiel-Protokolldatei mit zwei abgeleiteten Sitzungen

Protokolldatei					**Information**	
Typ	**ID**	**Ort**	**Zeit**	**Größe [Byte]**	**Zeit-differenz [s]**	**Sitzung**
Anf	A	(5,0|7,0)	2006-06-20/12:00:00	1512	10	Sitzung 1
Anf	B	(5,0|7,0)	2006-06-20/12:00:10	1432	20	
Anf	D	(5,0|7,0)	2006-06-20/12:00:30	1510	960	
Anf	E	(1,0|2,0)	2006-06-20/12:16:30	2312	10	Sitzung 2
Anf	F	(1,0|2,0)	2006-06-20/12:16:40	1460	75	
Anf	T	(1,0|2,0)	2006-06-20/12:17:55	1643	15	
Anf	F	(1,0|2,0)	2006-06-20/12:18:10	1856	10	
Anf	J	(1,0|2,0)	2006-06-20/12:18:20	1746	15	
Anf	I	(1,0|2,0)	2006-06-20/12:18:35	2684	420	

Nachfolgend wird beispielhaft die Ableitung von Sitzungen aus der in Abschnitt 3.4.1 angeführten Beispiel-Protokolldatei aus dem Bereich des Webs beschrieben. In Tabelle 3.2 sind lediglich die darin enthaltenen Dateneinträge angeführt, der besseren Übersicht halber wurden die Kontrolleinträge weggelassen. Als Grundlage zur Bestimmung der Sitzungen wird die in Abschnitt 3.4.2 eingeführte zeitliche Relation $R_{\texttt{Zeit}}$ verwendet, wobei die Zeitspanne Δt auf 15 Minuten gesetzt wird. Mit Ausnahme der Dateneinträge für den Zugriff auf die

Webseiten D und E liegen die Zeitpunkte aller Dateneinträge um weniger als 15 Minuten auseinander. Da zudem der letzte Eintrag ein die Anwendung schließender Kontrolleintrag ist, erhalten wir somit die beiden vollständigen Sitzungen (A, B, D) und (E, F, T, F, J, I).

3.4.4 Informationsgraph

Als zentrale Datenstruktur zur Modellierung des Wissens über das kollektive Zugriffsverhalten aller im Dienstgebiet einer Infostation sich bewegenden Benutzer wird ein Graph verwendet, da diese Datenstruktur für die Modellierung von Informationen und ihren Beziehungen sehr gut geeignet ist. Solch ein Graph muss die folgenden Anforderungen erfüllen:

1. Es müssen vollständige Sitzungen abgebildet werden können. Der Graph muss folglich spezielle Knoten beinhalten, die Sitzungsstart und Sitzungsende einer einzelnen Sitzung, sowie den Zusammenhang aller modellierten Sitzungen repräsentieren. Weiterhin soll es möglich sein, innerhalb einer Sitzung mehrfach angeforderte Informationen nur einmal zu berücksichtigen, um diese nicht übermäßig stark zu gewichten.

2. Zur Beschreibung des kollektiven Zugriffsverhaltens und der Stärke der Beziehungen zwischen den Informationen müssen für eine konkrete Anwendung des Verfahrens geeignete Knoten- und Kantenattribute sowie passende Gewichtungsfunktionen definiert werden können.

3. Das Wissen über das kollektive Zugriffsverhalten muss dynamisch an sich änderndes Zugriffsverhalten angepasst werden können.

Basierend hierauf wird die in den empfangenen Protokolldateien enthaltene Information über das Zugriffsverhalten auf einen knoten- und kantengewichteten, gerichteten Graphen abgebildet, der unterschiedliche Arten von Knoten beinhaltet. Solch ein Graph wird nachfolgend *Informationsgraph* genannt.

Definition 8 (Informationsgraph) *Der **Informationsgraph** ist ein Tupel IG = (V, E, f, w, B, z), wobei gilt:*

IG.V: *Menge der Knoten, wobei jeder Knoten die in einem Protokolleintrag enthaltene Informationsanfrage repräsentiert;*

IG.E: $E \subseteq \{(u, v) \mid u, v \in V \wedge u \neq v\}$ *ist die Menge der Kanten, wobei eine Kante diejenige Beziehung zwischen den entsprechenden Anfragen darstellt, auf Grundlage derer eine Sitzung gebildet wird. Seien R die Relation für die Bildung einer Sitzung,* l_u *und* l_v *zwei Protokolleinträge sowie* $u \in V$ *und* $v \in V$ *zwei Knoten im Informationsgraphen, welche die Informationsanfragen aus* l_u *bzw.* l_v *repräsentieren. Dann ist* $e = (u, v) \in E$ *eine Kante im Informationsgraphen, wenn gilt:* $l_u \; R \; l_v$

IG.f: $E \rightarrow \mathbb{R}$ *und* $V \rightarrow \mathbb{R}$*: Funktion, die jeder Kante und jedem Knoten ein Gewicht zuordnet;*

IG.w: *Der* Wurzelknoten *hat keine eingehenden Kanten und stellt die Verbindung zwischen den einzelnen aus den Protokolldateien abgeleiteten Sitzungen dar. Da von ihm aus alle Knoten erreichbar sind, dient er als Einstiegspunkt für die Traversierung;*

IG.B: *Die Menge der* Sitzungsstartknoten *repräsentiert die aus den Protokolldateien extrahierten Sitzungsstarteinträge. Jeder Sitzungsstartknoten ist mit dem Wurzelknoten verbunden;*

IG.z: *Der* zentrale Knoten *repräsentiert sowohl das Ende von Sitzungen, als auch Zugriffsmuster bezüglich wiederholter Anforderungen derselben Information innerhalb einer Sitzung. Mit ihm werden diejenigen Knoten verbunden, deren zugeordnete Anfragen entweder eine Sitzung beenden oder deren entsprechende Anfragen innerhalb einer Sitzung bereits mehrfach gestellt wurden. Der zentrale Knoten hat keine ausgehenden Kanten.*

In einigen der in Kapitel 3.12 diskutierten Prefetching-Technologien werden zur Auswahl vorab zu übertragender Informationen so genannte Abhängigkeitsgra-

phen (engl. *dependency graphs*) verwaltet. Der hier verwendete Informationsgraph unterscheidet sich von diesen dahingehend, dass wesentlich mehr Information verwaltet wird als nur die Abhängigkeiten zwischen den Informationsanforderungen. Dies sind beispielsweise die speziellen Knoten zur Abbildung von Sitzungen.

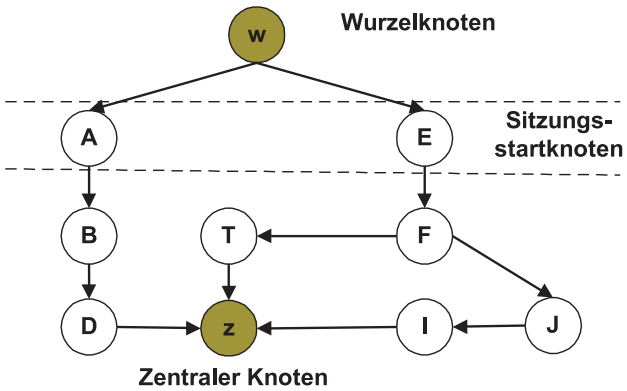


Abbildung 3.7: Beispiel eines Informationsgraphen

Abbildung 3.7 zeigt beispielhaft einen Informationsgraphen, der die beiden Sitzungen (A, B, D) und (E, F, T, F, J, I) modelliert, die in Abschnitt 3.4.3 aus der in Tabelle 3.2 eingeführten Beispiel-Protokolldatei abgeleitet wurden. Der Wurzelknoten w wird mit den beiden Sitzungsstartknoten A und E verbunden, die den Zugriff auf den jeweiligen Sitzungsstarteintrag repräsentieren. Jeweils zwei aufeinander folgende Informationsanforderungen innerhalb einer Sitzung werden durch eine Kante repräsentiert, was für die Sitzung (A, B, D) durch die Kanten (A, B) und (B, D) dargestellt wird. Das Ende der ersten Sitzung bei D wird durch die Kante von D zum zentralen Knoten z gekennzeichnet.

Betrachten wir nun in der zweiten Sitzung die Aufruffolge $F - T - F - J$. Die zweite Anforderung von F direkt nach T wird nicht berücksichtigt, statt dessen wird die Kante (T, z) eingefügt, die besagt, dass nach T eine Information wiederholt angefordert wurde. Der Benutzer setzt seine Informationsanforderung nach F mit J fort, was durch die Kante (F, J) beschrieben wird.

3.4.5 Konzept der Nutzungsprofile

Informationsbedürfnisse von Benutzern lassen sich nach diversen Kriterien klassifizieren (siehe auch Kapitel 2.2.4). Eine mögliche Unterteilung kann beispielsweise durch unterschiedliche Interessengebiete wie Kultur, Sport, Unterhaltung usw. erfolgen, die vom Benutzer angegeben werden. Um jedoch möglichst wenig Benutzerinteraktion zu verlangen, können zusätzliche Profilinformationen verwendet werden, die sich aus Protokolldateien, eventuell zur Verfügung stehenden Kontextinformationen wie beispielsweise aus dem NEXUS -Umgebungsmodell oder dem Typ der Benutzeranwendung ableiten lassen.

Zur Ableitung von Zugriffsmustern aus Protokolldateien finden sich in der Literatur viele Beiträge, wie beispielsweise [25,35,36,73,74,94] oder [50]. Hierzu werden Techniken aus dem Bereich des *Web usage mining* eingesetzt. Diese Art der Datenanalyse wird zur Personalisierung oder Evaluierung von Web-Auftritten (engl. *Websites*) oder e-Commerce-Anwendungen etc. angewandt, um beispielsweise einen Web-Auftritt benutzerfreundlicher zu gestalten und somit die Verweildauer der Besucher auf dem Web-Angebot zu erhöhen.

Kontextinformationen können beispielsweise die Altersgruppe, das Tätigkeitsmerkmal, die Tageszeit, die aktuelle Geschwindigkeit oder das verwendete Transportmittel sein. So wird ein Berufstätiger, der mit dem Zug zur Arbeit unterwegs ist, sehr wahrscheinlich andere Informationen abrufen als beispielsweise ein Tourist, der eine Stadt zu Fuß erkundet.

Schließlich können sich die angeforderten Informationen auch bezüglich der Benutzeranwendung unterscheiden. So werden beispielsweise von einem elektronischen Restaurantführer überwiegend Informationen über Restaurants abgerufen werden.

Ein Nutzungsprofil entspricht somit einem Informationskanal, der die auf das jeweilige Nutzungsprofil zugeschnittenen Informationen zur Verfügung stellt. Nun ist es möglich, dass Benutzer oder Anwendungen Informationen aus unterschied-

lichen Nutzungsprofilen benötigen. So kann beispielsweise ein Tourist, der zu Fuß unterwegs ist, Informationen über Sehenswürdigkeiten und Restaurants benötigen, aber auch Fahrpläne öffentlicher Verkehrsmittel oder Einkaufsmöglichkeiten. Aus diesem Grund können Benutzer mehreren Nutzungsprofilen zugeordnet werden, was im folgenden ein *Profilmix* genannt wird. Jedem darin enthaltenen Nutzungsprofil wird eine Wahrscheinlichkeit zugeordnet, mit der ein Benutzer die dem Profil entsprechenden Informationen anfordert.

Definition 9 (Profilmix) *Sei $\mathcal{N}$ die Menge aller der Infostation bekannten Nutzungsprofile und b ein Benutzer. Dann enthält ein* ***Profilmix*** *$\mathcal{N}_b \subseteq \mathcal{N}$ sämtliche Nutzungsprofile $n \in \mathcal{N}$, die b zugeordnet werden. Für alle enthaltenen Nutzungsprofile n_i mit $1 \leq i \leq |\mathcal{N}_b|$ gilt: $\sum_{n \in \mathcal{N}_b} P(n) = 1$.*

In dieser Arbeit liegt der Schwerpunkt auf der Integration der Nutzungsprofile in das Vorabübertragungsverfahren und nicht auf deren Ableitung. Die Zuordnung eines Benutzers zu einem Profilmix wird als bekannt voraus gesetzt, sie kann beispielsweise von der Benutzeranwendung an die Infostation übermittelt werden.

3.4.6 Übersicht über das Vorabübertragungsverfahren

Abbildung 3.8 veranschaulicht den prinzipiellen Ablauf des generischen Verfahrens.

Die in Abschnitt 3.3.3 eingeführten externen Dienste (Positionsbestimmungssystem, Verzeichnisdienst und Ereignisdienst) beeinflussen die Selektion relevanter Informationen nicht direkt und werden deshalb hier nicht näher beschrieben.

Benutzer navigieren mit Hilfe der *Benutzeranwendung* in einem schwach strukturierten Informationsraum. Die Benutzeranwendung gibt alle Informationszugriffe an die *Cache-Verwaltung* weiter. Wie bereits in Abschnitt 3.1 beschrieben, prüft

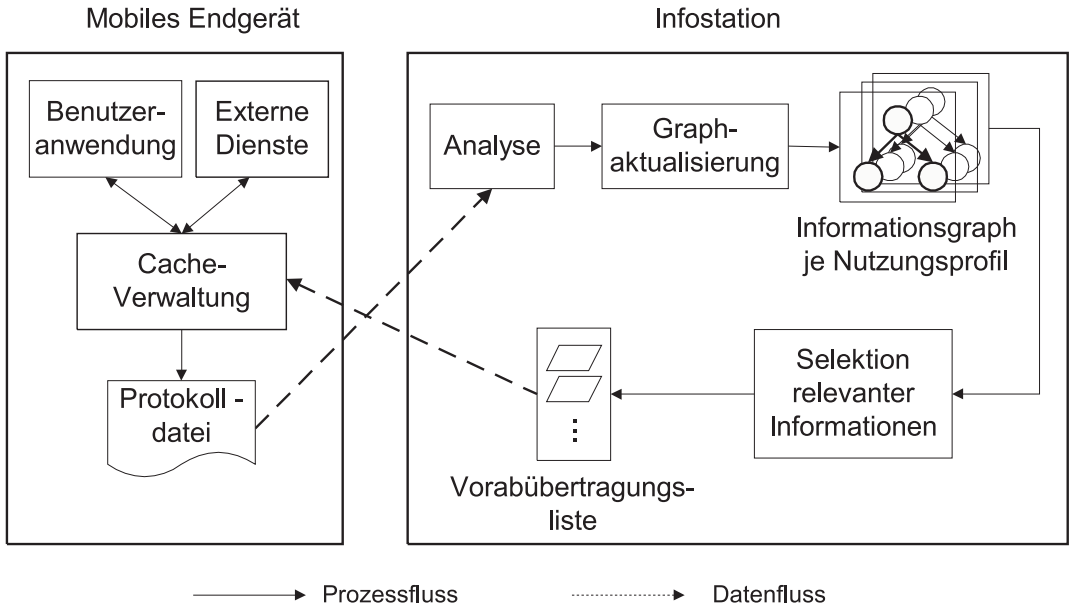


Abbildung 3.8: Übersicht über das Verfahren

die Cache-Verwaltung bei jeder Anforderung, ob sich die betreffende Information im Cache befindet.

In der in Abschnitt 3.4.1 definierten *Protokolldatei* vermerkt die Cache-Verwaltung alle lokal ausgeführten Informationsanforderungen. Bei jeder Informationsanforderung wird ein Eintrag hinzugefügt, der alle für die Modellierung des Zugriffsverhaltens des Benutzers notwendigen Angaben enthält.

Betritt ein Benutzer das Übertragungsgebiet einer Infostation, wird die Cache-Verwaltung vom Ereignisdienst darüber informiert. Daraufhin fordert sie vom Verzeichnisdienst die Adresse der für den aktuellen Ort zuständigen Infostation an. Das mobile Endgerät ändert seinen Verbindungsstatus zu „stark verbunden" und sendet die Protokolldatei an die entsprechende Infostation.

Empfängt eine Infostation eine Protokolldatei, ordnet sie den Benutzer auf der Grundlage von dessen Profilmix den entsprechenden Nutzungsprofilen zu. Die

Infostation *analysiert* die Protokolldatei und *aktualisiert* auf Grundlage der Analyseergebnisse die zu den Nutzungsprofilen gehörenden Informationsgraphen.

Für die *Selektion der für die Vorabübertragung relevanten Informationen* müssen Selektionsverfahren mit und ohne Clusterbildung sowie geeignete Bewertungskriterien zur Berechnung eines Relevanzwertes zur Verfügung gestellt werden. Die ausgewählten Informationen bzw. Cluster werden auf Grundlage der berechneten Relevanzwerte sortiert in eine Vorabübertragungsliste eingefügt. Die Informationen mit der höchsten Relevanz in dieser Liste werden schließlich, abhängig von der verfügbaren Größe des Caches, auf das mobile Endgerät des Benutzers übertragen.

Verlässt das mobile Endgerät das Übertragungsgebiet der Infostation, wechselt sein Status wieder zu „schwach verbunden" bzw. „nicht verbunden", falls kein Netz zur Verfügung steht.

Aus Performanzgründen kann der Selektionsprozess und die Erstellung der Vorabübertragungsliste periodisch durchgeführt werden, die Aktualisierung des Informationsgraphen wird jedoch jedesmal ausgeführt, wenn eine Protokolldatei an die Infostation gesendet wird. Somit passen sich die Informationsgraphen permanent an das Zugriffsverhalten der Benutzer an, während die Aktualität der Vorabübertragungsliste von der gewählten Zeitspanne zwischen den Selektionsprozessen abhängt.

3.5 Spezialisierung der generischen Konzepte für das Web

Bislang wurden die generischen Konzepte beschrieben, die unabhängig sind vom zugrunde liegenden Informationsraum und der Anwendung. Für eine konkrete Anwendung sind folgende Spezialisierungen erforderlich:

- Definieren von Relationen, die für die Ableitung von Sitzungen aus Protokolldateien maßgeblich sind. Hierfür muss gegebenenfalls die Spezifikation der Protokolldatei erweitert werden.
- Definieren von geeigneten Beziehungen und Metriken zur Bestimmung der semantischen Nähe von Informationen für die Selektionsverfahren mit Clusterbildung.
- Erweiterung der Definition des Informationsgraphen um die erforderlichen Knoten- und Kantenattribute sowie geeignete Gewichtungsfunktionen.

In diesem Abschnitt werden die Spezialisierungen der generischen Konzepte für den Zugriff auf das Web spezifiert.

3.5.1 Relationen zur Ableitung von Sitzungen

Wie in Abschnitt 3.4.3 erwähnt, modelliert eine Sitzung das Zugriffsverhalten eines Benutzers, der Webseiten anfordert, um damit ein bestimmtes Ziel zu erreichen. Im Web ist dieses Zugriffsverhalten dadurch charakterisiert, welche Webseiten ein Benutzer anfordert und wie groß die zeitlichen Abstände zwischen den einzelnen Zugriffen sind. In der Literatur, wie beispielsweise in [20, 42, 45], werden Sitzungen bezüglich des Zugriffs auf das Web überwiegend auf Grundlage der Besuchsdauer einer Webseite definiert. Überschreitet diese Besuchsdauer einen bestimmten Wert, wird davon ausgegangen, dass der Benutzer sein Ziel erreicht hat, womit die Sitzung endet. Dieser Ansatz zur Ableitung einer Sitzung aus einer Protokolldatei wird in dieser Arbeit übernommen. Relationen wie beispielsweise ein ähnlicher Inhalt oder die in Abschnitt 3.4.2 beschriebene Relation $R_{\texttt{Link}}$, mittels derer zwei durch einen Hyperlink verknüpfte Webseiten in Beziehung stehen, werden hier nicht berücksichtigt, da sie nur aufwändig durch Analyse jeder einzelnen Webseite abgeleitet werden können. Das Zugriffsverhalten wird, wie in [20, 42, 45] vorgeschlagen, durch die zeitliche Relation genügend genau modelliert.

Sei $\mathcal{L}$ eine Protokolldatei und $l_i, l_{i+1} \in \mathcal{L}$ zwei direkt aufeinander folgende Dateneinträge. Dann berechnet sich die *Besuchsdauer* einer Webseite aus der Differenz der Zeitstempel der Dateneinträge:

$$l_i.\texttt{dauer} = l_{i+1}.\texttt{Zeit} - l_i.\texttt{Zeit} \tag{3.1}$$

Darauf aufbauend wird die zeitliche Relation $R_{\texttt{Zeit}}$ wie folgt definiert.

Definition 10 (Zeitliche Relation $R_{\texttt{Zeit}}$) *Die* ***zeitliche Relation*** $R_{\texttt{Zeit}}$ *verbindet zwei Dateneinträge l_i und l_k miteinander, wenn beide zugehörigen Webseiten direkt nacheinander angefordert wurden und die Besuchsdauer auf der in l_i angeforderten Seite eine definierte Zeitspanne Δt nicht überschreitet.*

$$\begin{aligned} R_{\texttt{Zeit}} \quad = \{(l_i, l_k) \mid (l_k.\texttt{Typ} = \text{„}Anf\text{”} \wedge\ l_i.\texttt{Typ} = \text{„}Anf\text{”}) \\ \wedge\ l_i = \texttt{pre}(l_k, \mathcal{L}) \\ \wedge\ l_i.\texttt{dauer}) < \Delta t\} \end{aligned} \tag{3.2}$$

Gemäß den Empfehlungen von Arlitt et al. in [4] sind 15 Minuten ein typischer Wert für die Zeitspanne Δt.

3.5.2 Relationen für die Clusterbildung

Wie bereits in Abschnitt 3.2 erwähnt, setzen Benutzer manche Webseiten als reines Navigationsmittel ein, um mit Hilfe der dort vorhandenen Hyperlinks die sie interessierenden Seiten zu finden [25, 83]. Die zur Navigation verwendeten Webseiten werden nachfolgend als *Transitseiten* und die Seiten mit dem interessierenden Inhalt als *Inhaltsseiten* bezeichnet. Für die Klassifizierung von Webseiten wird die Besuchsdauer einer Webseite verwendet. Transitseiten haben die besondere Eigenschaft, dass sie im Verhältnis zu Inhaltsseiten nur kurz besucht

werden, während letztere länger betrachtet werden. Weiterhin wird es wesentlich mehr Transitseiten als Inhaltsseiten in einer Protokolldatei geben. Um die Inhaltsseiten aus der Protokolldatei herauszufiltern, wird das geometrische Mittel verwendet (siehe auch Anhang A.1). Es eignet hervorragend für Daten, die nicht normalverteilt sind, sondern denen eine schiefe Verteilung zugrunde liegt.

Definition 11 (Inhaltsseiten und Transitseiten) *Sei m_g das geometrische Mittel aller Besuchsdauern in einer Protokolldatei $\mathcal{L}$, $l.\texttt{dauer}$ die Besuchsdauer der Webseite, die im Protokolleintrag $l \in \mathcal{L}$ angefordert wurde und γ ein Wert im Intervall $[0, 1]$. Dann wird die im Protokolleintrag l angeforderte Webseite als* ***Inhaltsseite*** *klassifiziert, wenn gilt*

$$l.\texttt{dauer} \cdot \gamma > m_g$$

Andernfalls wird sie als ***Transitseite*** *eingestuft.*

Parameter γ bestimmt die Selektivität dieses Filters. Für $\gamma = 0$ wird keine Webseite als Inhaltsseite eingeordnet, denn in diesem Fall ist die Bedingung aus Definition 11 nie erfüllt. Ist $\gamma = \infty$, so wird jede Seite als Inhaltsseite betrachtet. Für $\gamma = 0{,}5$ muss die Besuchszeit einer Webseite mindestens doppelt so groß sein wie das geometrische Mittel, um als Inhaltsseite angesehen zu werden. Beim empirischen Vergleich mehrerer Parameterwerte mit zahlreichen Protokolleinträgen wurden für $\gamma = 0{,}75$ gute Resultate erzielt. Nun ist es für die letzte Seite, die innerhalb einer Sitzung besucht wurde, nicht möglich, die Besuchsdauer zu bestimmen, um diese Seite zu klassifizieren. Für diesen Fall wurde die Berechnung des Seitentyps (Transit- oder Inhaltsseite) modelliert. Gemäß den Resultaten der in dieser Arbeit durchgeführten Experimente folgt die Wahrscheinlichkeit einer Seite, als Inhaltsseite eingestuft zu werden, einer Lognormal-Verteilung mit dem Erwartungswert $\mu = 0{,}45$ und der Varianz $\sigma^2 = 0{,}22$.

3.5.3 Informationsgraph

Im Rahmen der Spezialisierung des Verfahrens für den Zugriff auf Webseiten müssen geeignete Knoten- und Kantenattribute sowie Gewichtungsfunktionen definiert werden.

Knoten: Ein Knoten im Informationsgraphen repräsentiert die Anforderung einer Webseite.

Definition 12 (Knotenattribute für Webseiten) *Sei l_v ein Dateneintrag und $v \in V$ ein Knoten im Informationsgraphen, der die in l_v enthaltene Anforderung einer Webseite repräsentiert. Dann besitzt v die folgenden* ***Attribute****:*

v.ID: *Global eindeutiger Bezeichner (URL) der angeforderten Webseite;*
v.größe = l_v.Größe: *Größe der Webseite in Byte;*
v.anfrageZähler: *Anfragezähler zählt, wie oft die Seite angefordert wurde;*
v.inhaltsZähler: *Wie bereits erwähnt, werden Webseiten nicht von allen Benutzern als gleich wichtig eingestuft: Was für den einen eine reine Transitseite ist, kann für einen anderen Benutzer eine Inhaltsseite sein. Der Inhaltszähler zählt, wie oft die Seite als Inhaltsseite angesehen wurde.*

Als Gewichtungsfunktion wird jedem Knoten $v \in V$ eine *Inhaltswahrscheinlichkeit* zugeordnet, die sich mittels Gleichung 3.3 berechnen lässt. Sie entspricht der Wahrscheinlichkeit, dass die entsprechende Seite für einen potenziellen Benutzer eine Inhaltsseite darstellt.

Definition 13 (Inhaltswahrscheinlichkeit) *Sei $v \in V$ ein Knoten eines Informationsgraphen. Dann wird die* ***Inhaltswahrscheinlichkeit*** *von v berechnet als*

$$P_{\texttt{Inhalt}}(v) = \frac{v.\texttt{inhaltsZähler}}{v.\texttt{anfrageZähler}} \tag{3.3}$$

In der Diplomarbeit von Pfahl [82] wurden umfangreiche Auswertungen bezüglich des Einflusses der Inhaltswahrscheinlichkeit auf die Relevanz von Clustern durchgeführt. So wurden unterschiedliche Werte für die minimale Inhaltswahrscheinlichkeit definiert und ausgewertet, die ein Knoten haben muss, um bei der Clusterbildung als Inhaltsknoten berücksichtigt zu werden. Die besten Resultate wurden erzielt, wenn mindestens ein Benutzer die Seite als Inhaltsseite betrachtet hat. Ein Knoten wird nachfolgend als *Inhaltsknoten* bezeichnet, wenn die entsprechende Webseite von mindestens einem Benutzer als Inhaltsseite angesehen wurde.

Definition 14 (Inhaltsknoten) *Ein Knoten $v \in V$ eines Informationsgraphen wird als **Inhaltsknoten** bezeichnet, wenn gilt:*

$$P_{\texttt{Inhalt}}(v) > 0$$

Kanten: Eine Kante im Informationsgraphen verbindet zwei Knoten, deren zugehörige Webseiten innerhalb einer Sitzung direkt nacheinander aufgerufen wurden.

Definition 15 (Kantenattribut für Webseiten) *Eine Kante $e \in E$ im Informationsgraphen hat folgendes **Attribut**:*

e.sequenzZähler: *Sequenzzähler zählt, wie oft die Kante über die Lebensdauer des Informationsgraphen hinweg aktualisiert wurde.*

Nachfolgend werden die Kanten- und Pfadwahrscheinlichkeit definiert, die Informationen über das kollektive Zugriffsverhalten liefern und somit der Berechnung eines Relevanzwertes als Grundlage dienen.

Definition 16 (Kantenwahrscheinlichkeit) *Sei $e = (u, v)$ eine Kante, welche die Knoten $u \in V$ und $v \in V$ verbindet. Sei weiterhin $E_{\texttt{out}(u)}$ die Menge aller*

von u ausgehenden Kanten. Dann bezeichnet die ***Kantenwahrscheinlichkeit*** *von e die Wahrscheinlichkeit, dass ein Benutzer die dem Zielknoten zugeordnete Informationsanfrage stellt, falls er zuletzt die dem Quellknoten entsprechende Anfrage gestellt hat. Sie wird gemäß Gleichung 3.4 berechnet als:*

$$P(e) = \frac{e.\texttt{sequenzZähler}}{\sum_{e' \in E_{\texttt{out}(u)}} e'.\texttt{sequenzZähler}} \tag{3.4}$$

Definition 17 (Pfadwahrscheinlichkeit) *Sei $v \in V$ ein Knoten im Informationsgraphen und $\texttt{Pfad}(w, v)$ ein Pfad von der Wurzel w zu v. Dann bezeichnet die* ***Pfadwahrscheinlichkeit*** *von v die Wahrscheinlichkeit, mit der ein Benutzer bzw. eine Benutzergruppe die entsprechende Webseite genau auf diesem Pfad anfordert. Sie wird berechnet als das Produkt der Kantenwahrscheinlichkeiten jeder Kante, die diesen Pfad bildet.*

$$P_{\texttt{Pfad}}(v) = \prod_{e \in \texttt{Pfad}(w,v)} P(e) \tag{3.5}$$

Abbildung 3.9 zeigt einen Beispiel-Informationsgraphen, der die definierten Konzepte darstellt. Die Kantenbeschriftungen repräsentieren deren Sequenzzähler.

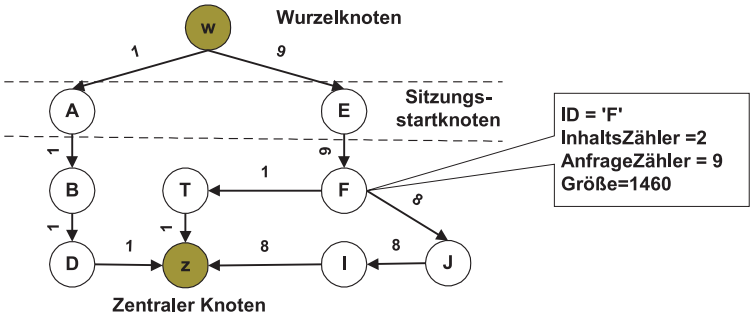


Abbildung 3.9: Beispiel-Informationsgraph

Der besseren Übersicht halber werden die Knotenattribute nur für Knoten F angezeigt, der die Anforderung der Webseite mit der URL F mit einer Größe

von 1460 Bytes darstellt. F wurde insgesamt neunmal angefordert und zweimal als Inhaltsseite betrachtet. Die Inhaltswahrscheinlichkeit von F ist demnach $P_{\texttt{Inhalt}}(F) = \frac{2}{9} = 0{,}\bar{2}$. Die Webseite F wurde demnach von zirka 22% der Benutzer als Inhaltsseite angesehen. Knoten F hat nur eine ausgehende Kante, folglich ist deren Kantenwahrscheinlichkeit $P(F, J) = 1$. Mit anderen Worten: Die Wahrscheinlichkeit, dass Benutzer nach F direkt J anfordern, ist 100%. Die Pfadwahrscheinlichkeit des Pfads vom Wurzelknoten über E und F zu J ist gemäß Gleichung 3.5 das Produkt der Kantenwahrscheinlichkeiten $P(w, E) \cdot P(E, F) \cdot P(F, J) = \frac{9}{9+1} \cdot 1 \cdot \frac{8}{8+1} = 0{,}8$. Sie besagt, dass eine Wahrscheinlichkeit von 80% besteht, dass ein Benutzer J genau auf dem Pfad (E, F, J) aufruft.

3.6 Analyse der Protokolldatei

Eine Infostation analysiert jede empfangene Protokolldatei, um daraus Sitzungen abzuleiten, diese einem Dienstgebiet zuzuordnen und Webseiten in Inhalts- oder Transitseiten zu klassifizieren. Die Sitzungen werden zur Aktualisierung des Informationsgraphen benötigt, der das Wissen über das kollektive Zugriffsverhalten aller Benutzer im Dienstgebiet der Infostation modelliert. Die Klassifizierung der Webseiten dient als Grundlage für die Clusterbildung.

3.6.1 Ableiten von Sitzungen

Wie bereits in Abschnitt 3.4.3 erwähnt, kann das Fortschreiben einer Protokolldatei über einen größeren Zeitraum hinweg verlaufen, ohne dass der Benutzer zusammenhängend Informationen anfordert. Zur Analyse der Zugriffsverhaltens eines Benutzers müssen folglich durch das Ableiten von Sitzungen diejenigen Protokolleinträge identifiziert werden, die miteinander in Beziehung stehen. Hierzu wird die in Definition 3.2 definierte zeitliche Relation $R_{\texttt{Zeit}}$ verwendet, die auf

der gemäß Gleichung 3.1 berechneten Besuchsdauer einer Seite basiert.

Da eine Sitzung eine Sequenz von miteinander in Beziehung stehenden Webseiten-Aufrufen ist, die logisch zusammen gehören, wird eine Sitzung nur einem einzigen Dienstgebiet zugeordnet, auch wenn innerhalb der Sitzung unterschiedliche Orte protokolliert wurden. Unter der Annahme, dass das Bedürfnis für die Anforderung einer Reihe von Webseiten an einem bestimmten Ort entstand, hat dieser einen signifikanten Einfluss auf die weitere Entwicklung der Sitzung. Eine Sitzung wird folglich demjenigen Dienstgebiet zugeordnet, das den Ort beinhaltet, an dem die erste Webseite der Sitzung aufgerufen wurde.

Es werden nur vollständige Sitzungen ausgewertet (siehe Definition 7), unvollständige Sitzungen verbleiben in der Protokolldatei. Die an die Infostation übertragenen vollständigen Sitzungen werden aus der Protokolldatei gelöscht.

Sitzungen, die in den Dienstgebieten weiterer Infostationen gültig sind, werden an diese weitergeleitet. Eine Infostation wertet nur Sitzungen aus, die in ihrem Dienstgebiet abgehalten wurden.

3.6.2 Klassifizierung der Webseiten

Weiterhin werden die in einer Sitzung angeforderten Webseiten in Transit- und Inhaltsseiten eingeteilt. Gemäß Definition 11 wird hierzu das geometrische Mittel der Besuchsdauern aller in der Protokolldatei angeforderten Webseiten verwendet. Eine Webseite wird somit als Inhaltsseite klassifiziert, wenn gilt: $l.\texttt{dauer} \cdot \gamma > m_g$, wobei für einen Wert von $\gamma = 0{,}75$ zufrieden stellende Resultate erzielt wurden.

Tabelle 3.3: Auszug aus der Beispiel-Protokolldatei mit abgeleiteter Information

Protokolldatei					**Information**	
Typ	**ID**	**Ort**	**Zeit**	**Größe [Byte]**	**Besuchs-dauer [s]**	**Sitzung**
Anf	*A*	(5,0|7,0)	2006-06-20/12:00:00	1512	10	Sitzung 1
Anf	*B*	(5,0|7,0)	2006-06-20/12:00:10	1432	20	
Anf	***D***	(5,0|7,0)	2006-06-20/12:00:30	1510	960	
Anf	*E*	(1,0|2,0)	2006-06-20/12:16:30	2312	10	Sitzung 2
Anf	***F***	(1,0|2,0)	2006-06-20/12:16:40	1460	75	
Anf	*T*	(1,0|2,0)	2006-06-20/12:17:55	1643	15	
Anf	*F*	(1,0|2,0)	2006-06-20/12:18:10	1856	10	
Anf	*J*	(1,0|2,0)	2006-06-20/12:18:20	1746	15	
Anf	***I***	(1,0|2,0)	2006-06-20/12:18:35	2684	420	

3.6.3 Beispiel

Tabelle 3.3 zeigt die Dateneinträge der in Abschnitt 3.4.1 eingeführten Beispiel-Protokolldatei, erweitert um die abgeleitete Besuchsdauer in Sekunden und die Aufteilung in zwei Sitzungen. Inhaltsseiten sind fett markiert und wurden folgendermaßen bestimmt: Das geometrische Mittel aller Besuchsdauern in dieser Protokolldatei sind 26 Sekunden, mit $\gamma = 0{,}75$ wird die Webseite F als Inhaltsseite eingestuft. Für D und I, die jeweils eine Sitzung beenden, wird mit Hilfe der in Abschnitt 3.5.2 ermittelten Lognormal-Verteilung ihr Status als Inhaltsseite berechnet. Die restlichen Webseiten werden als Transitseiten klassifiziert.

Gemäß der Beschreibung in Abschnitt 3.6.1 ist der Ort des ersten Eintrags einer Sitzung für die Zuordnung zu dem Dienstgebiet einer Infostation maßgeblich. Abbildung 3.10 illustriert beispielhaft die geometrische Zuordnung von Positionen zu den Dienstgebieten zweier Infostationen *I1* und *I2*. Deren zugeordnete Dienstgebiete sind durch die Kreise repräsentiert und mit *DG1* bzw. *DG2* bezeichnet. Empfängt beispielsweise die für *DG1* verantwortliche Infostation *I1* die Beispiel-Protokolldatei aus Tabelle 3.2, so aktualisiert sie ihren Informati-

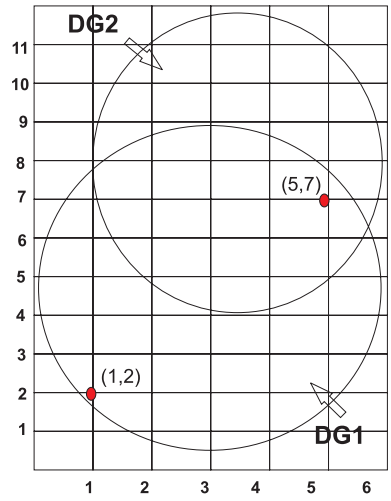


Abbildung 3.10: Einteilung der Dienstgebiete zweier Infostationen

onsgraphen mit beiden daraus abgeleiteten Sitzungen. Die Einträge aus *Sitzung 2* werden an die für *DG2* zuständige Infostation *I2* weitergeleitet.

3.7 Aktualisierung des Informationsgraphen

Wie bereits in Abschnitt 3.4.4 erwähnt, spiegelt der Informationsgraph das kollektive Zugriffsverhalten aller Benutzer im Dienstgebiet einer Infostation wider. Hierfür werden die mittels Analyse der empfangenen Protokolldateien abgeleiteten Sitzungen, die im Dienstgebiet der Infostation begonnen wurden, auf diesen Graphen abgebildet.

Für jede Sitzung $S(l_1, R) = \{l_1, \ldots, l_m\}$, die während der Analyse der Protokolldatei identifiziert und dem Dienstgebiet der Infostation zugeordnet wurde, wird zur Aktualisierung des Informationsgraphen der nachfolgend in Pseudocode dargestellte Algorithmus 1 ausgeführt. Hierbei bezeichnet P eine zu analysieren-

de Protokolldatei, S die aktuelle Sitzung, IG den Informationsgraphen, B die Menge der Sitzungsstartknoten und `URLs` die Menge der Bezeichner der Webseiten, die während der laufenden Aktualisierung bereits gesehen wurden. Der besseren Lesbarkeit halber wird nachfolgend die URL der in Protokolleintrag l_i angeforderten Webseite mit $l.\texttt{ID}$ bezeichnet.

In Zeilen 4 bis 8 wird der *erste Sitzungseintrag* l_1 bearbeitet: Nach dem Aktualisieren des entsprechenden Knotens v_1 wird dieser der Menge B der Sitzungsstarteinträge und seine URL der Menge `URLs` aller bereits gesehenen Webseiten hinzugefügt. Schließlich wird die Kante vom Wurzelknoten w zu v_1 aktualisiert, indem deren Sequenzzähler inkrementiert wird.

Für *alle folgenden Sitzungseinträge* l_i gilt dann:
Zeile 12: Falls die Webseite in der Sitzung bereits gesehen wurde, wird der Sequenzzähler der Kante von v_{i-1} zum zentralen Knoten z inkrementiert.
Zeilen 14-16: Falls nicht, wird Knoten v_i aktualisiert und der Sequenzzähler der Kante (v_{i-1}, v_i) inkrementiert.

Zeile 19: Für den letzten Eintrag l_m wird der Sequenzzähler der Kante von (v_m, z) inkrementiert.

Zeilen 21-29 (*Aktualisierung eines Knotens*): Falls für einen Eintrag l_i noch kein Knoten v_i existiert, setze $v_i.\texttt{ID} = l.\texttt{ID}$ und $v_i.\texttt{größe} = l.\texttt{Größe}$. Inkrementiere in jedem Fall $v_i.\texttt{anfrageZähler}$. Falls l_i eine Inhaltsseite ist, inkrementiere auch $v_i.\texttt{inhaltsZähler}$.

Zeilen 30-34(*Aktualisierung einer Kante*): Falls die Kante noch nicht existiert, wird sie erzeugt und der Sequenzzähler mit dem Wert 0 initialisiert. Anschließend wird der Sequenzzähler inkrementiert.

Abbildung 3.11 stellt den Informationsgraphen der Beispiel-Infostation *I1* dar, wie er zum Zeitpunkt vor dem Empfang der Beispiel-Protokolldatei aus Tabelle 3.3 ausgesehen haben könnte. Die Kantenbeschriftungen repräsentieren deren

Algorithm 1 Algorithmus zur Aktualisierung eines Informationsgraphen

```
 1: while Protokolldatei P enthält noch eine vollständige Sitzung do
 2:   S = P.nächsteSitzung();
 3:   URLs = ∅;
 4:   l_1 = S.ersterEintrag();
 5:   v_1 = IG.aktualisiereKnoten(l_1);
 6:   URLs = URLs ∪ {v_1.ID};
 7:   IG.B = IG.B ∪ {v_1}; // Sitzungsstarteinträge
 8:   IG.aktualisiereKante(w, v_1);
 9:   for i = 2 to m do
10:     l_i = S.nächsterEintrag();
11:     if l_i.ID ∈ URLs then // wurde Webseite in der Sitzung schon besucht?
12:       IG.aktualisiereKante(v_{i-1}, z);
13:     else
14:       v_i = IG.aktualisiereKnoten(l_i);
15:       URLs = URLs ∪ {v_1.ID};
16:       IG.aktualisiereKante(v_{i-1}, v_i);
17:     end if
18:   end for
19:   IG.aktualisiereKante(v_m, z); // Sitzungsende
20: end while

    function aktualisiereKnoten(Eintrag l): Knoten {
21: Knoten v = IG.gibKnoten(l.ID);
22: if Knoten existiert noch nicht then
23:   v = neuerKnoten(l.ID, l.Größe);
24: end if
25: v.anfrageZähler = v.anfrageZähler + 1;
26: if v ist Inhaltsseite then
27:   v.inhaltsZähler = v.inhaltsZähler + 1;
28: end if
29: return v;
    }

    function aktualisiereKante(Knoten u, Knoten v){
30: Kante e = IG.gibKante(u, v);
31: if Kante existiert noch nicht then
32:   e = neueKante(u, v);
33: end if
34: e.sequenzZähler = e.sequenzZähler + 1;
    }
```

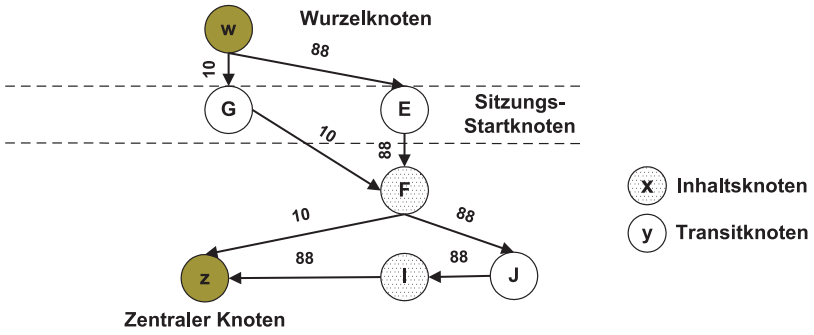


Abbildung 3.11: Informationsgraph vor der Aktualisierung

Sequenzzähler, die Knotenbeschriftung stellt die URL der angeforderten Webseite dar. Der Übersicht halber werden die restlichen Knotenattribute Größe, Anfrage- und Inhaltszähler nicht angezeigt.

Abbildung 3.12 zeigt den Informationsgraphen aus Abbildung 3.11 nach dessen Aktualisierung mit der Beispiel-Protokolldatei. Nachfolgend wird stellvertretend die Aktualisierung mit Sitzung 2 beschrieben, die aus der Aufruffolge $E - F - T - F - J - I$ besteht, wobei die Webseiten F und I als Inhaltsseiten klassifiziert wurden. Zuerst wird die Kante vom Wurzelknoten w zum Sitzungs-

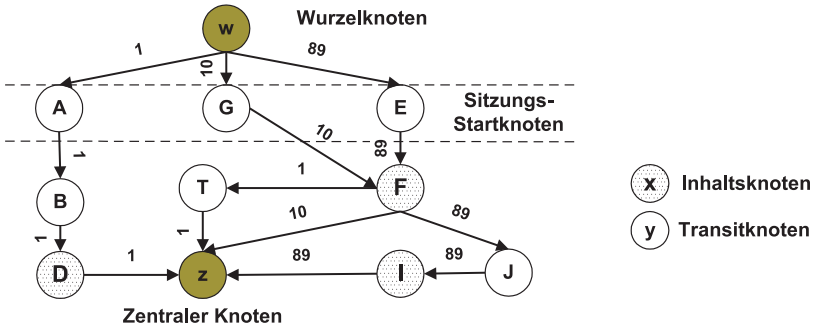


Abbildung 3.12: Informationsgraph nach der Aktualisierung

startknoten E aktualisiert, indem der Wert ihres Sequenzzählers von 88 auf 89 erhöht wird. Da Webseite E in dieser Sitzung eine Transitseite ist, wird lediglich der Anfragezähler von Knoten E inkrementiert. Im nächsten Schritt werden

Inhalts- und Anfragezähler von F inkrementiert, da F als Inhaltsseite angesehen wurde, und der Sequenzzähler von Kante (E, F) inkrementiert. Die Aktualisierung der Knotenattribute erfolgt immer nach diesem Schema und wird deshalb nachfolgend nicht mehr beschrieben. Anschließend wird eine neue Kante (F, T) eingefügt und ihr Sequenzzähler mit 1 initialisiert. Da nach T erneut F aufgerufen wurde, wird die Kante (T, z) eingefügt und initialisiert, damit F im Verlauf dieser Sitzung nicht übermäßig gewichtet wird. Nach F wird die Webseite J aufgerufen, wodurch die Kante (F, J) aktualisiert wird, indem ihr Sequenzzähler von 88 auf 89 erhöht wird. Dasselbe gilt für die Kante (J, I). Nachdem der Eintrag, der den Aufruf von I enthält, die Sitzung beendet, wird abschließend die Kante (I, z) aktualisiert.

Alterungsverfahren Verwendet man diesen Algorithmus zur Aktualisierung des Informationsgraphen für sich allein, passt sich der Informationsgraph nur sehr langsam an ein verändertes Zugriffsverhalten an. Schlimmer noch, wenn aktuelle und veraltete Zugriffe gleich stark berücksichtigt werden, kann diese Anpassungsfähigkeit mit der Zeit sogar abnehmen. Dies kann mit einem Beispiel veranschaulicht werden: Angenommen, eine Webseite, die aktuelle Informationen über die Fußball-Weltmeisterschaft (wie beispielsweise Spielzeiten und Paarbildungen) bietet, wird vor und während dieses Ereignisses recht häufig besucht. Nach der WM verlieren jedoch die Benutzer das Interesse an ihr und besuchen sie nur noch äußerst selten. Nun hat der Zugriffszähler dieser Webseite aber bereits einen sehr hohen Wert angenommen. Behielte er diesen Wert fortwährend, würde diese Seite noch lange Zeit als populär eingestuft werden, obwohl sie es in Wirklichkeit gar nicht mehr ist. Neue Webseiten dagegen hätten es dann sehr schwer, die Beliebtheit dieser Seite zu übertreffen. In diesem Fall kann der Informationsgraph seine Bestimmung, das Zugriffsverhalten zu modellieren, nicht mehr erfüllen.

Um diesem negativen Seiteneffekt entgegen zu wirken, wurde eine *Alterungsfunktion* `IG.`f eingeführt, welche die Zeit in Epochen einteilt und dafür sorgt, dass (bei

Bedarf) Werten mit zunehmendem Alter eine immer geringere Bedeutung beigemessen wird. Immer, wenn eine Epoche endet, wird für jeden der Knoten- und Kantenzähler ein geglätteter Wert berechnet. Hierzu wird eine zeitliche Glättungsfunktion verwendet, die auch in der Analyse von Zeitreihen Anwendung findet: der exponentiell gewichtete gleitende Mittelwert. Diese einfach exponentielle Glättungsfunktion wird auch in der Überlastkontrolle des Transmission Control Protocols (TCP) zur Glättung der Paketumlaufzeit (engl. *round trip time*) verwendet. Zu diesem Zweck wird für jeden der Zähler zusätzlich ein Hilfszähler eingeführt, der die Zahl der Zugriffe etc. in der aktuellen Epoche zählt. Sei $v \in V$, $e \in E$ und `zähler` einer der Zähler $v.$`anfrageZähler`, $v.$`inhaltsZähler` oder $e.$`sequenzZähler`. Sei weiterhin `hilfsZähler` der zugehörige Hilfszähler. Dann wird am Ende einer Epoche i, mit $i > 1$, der geglättete Zähler rekursiv berechnet als

$$\texttt{zähler}[i] = \delta \cdot \texttt{zähler}[i-1] + (1-\delta) \cdot \texttt{hilfsZähler}[i] \tag{3.6}$$

Danach wird der Hilfszähler wieder zurückgesetzt. Der Glättungsfaktor δ legt fest, wie stark der Hilfszähler in der Berechnung berücksichtigt werden soll, d.h., wie stark die in der aktuellen Epoche gezählten Zugriffe gewichtet werden. Je mehr sich δ dem Wert 0 nähert, um so weniger wird der geglättete Wert der vorigen Epochen berücksichtigt und um so höher wird der Hilfszähler der aktuellen Epoche gewertet. Diese Einstellung eignet sich besonders bei stark schwankendem Zugriffsverhalten. In den Experimenten, die im Laufe dieser Arbeit durchgeführt wurden, erwies sich $\delta = 0{,}5$ als passende Einstellung.

Schreibt man die Rekursion aus, so enthüllt sich der exponentielle Charakter der Funktion. Die Glättungsfunktion wird erst ab dem zweiten Intervall berechnet und mit $\texttt{zähler}[1] = \texttt{hilfsZähler}[1]$ initialisiert. Sei n die Anzahl der vergange-

nen Epochen und `hilfsZähler[j]` der Hilfszähler in Epoche j. Somit gilt dann:

$$
\begin{aligned}
\texttt{zähler}[2] &= \delta \cdot \texttt{zähler}[1] + (1-\delta) \cdot \texttt{hilfsZähler}[2] \\
\texttt{zähler}[3] &= \delta(\delta \cdot \texttt{zähler}[1] + (1-\delta) \cdot \texttt{hilfsZähler}[2]) + (1-\delta) \cdot \texttt{hilfsZähler}[3] \\
&= \delta^2 \cdot \texttt{zähler}[1] + \delta(1-\delta) \cdot \texttt{hilfsZähler}[2] + (1-\delta) \cdot \texttt{hilfsZähler}[3] \\
&\vdots \\
\texttt{zähler}[n] &= \delta^{n-1} \cdot \texttt{zähler}[1] + \delta^{n-2}(1-\delta) \cdot \texttt{hilfszähler}[2] + \cdots + \\
&\quad \delta(1-\delta) \cdot \texttt{hilfsZähler}[n-1] + (1-\delta) \cdot \texttt{hilfsZähler}[n]
\end{aligned}
$$

Setzt man nun $\texttt{zähler}[1] = \texttt{hilfsZähler}[1]$ ein, so ergibt sich

$$\texttt{zähler}[n] = \sum_{i=1}^{n} (\delta^{n-i} \cdot (1-\delta)^{i-1} \cdot \texttt{hilfsZähler}[i])$$

In der Implementierung des vorgestellten Vorabübertragungsverfahrens wird die Alterungsfunktion zusammen mit der Graphtraversierung durchgeführt, was eine zusätzliche Traversierung vermeidet. Überdies wird sie nur für diejenigen Kanten und Knoten ausgeführt, die vom Auswahlprozess betroffen sind.

3.8 Erstellung der Vorabübertragungsliste

Eine Vorabübertragungsliste enthält diejenigen Webseiten, die eine Infostation abhängig von der Größe des Caches vorab auf das mobile Endgerät eines Benutzers lädt. Nun wäre es sicherlich am einfachsten, wenn alle im Dienstgebiet der Infostation angefragten Webseiten in dieser Liste enthalten wären, denn in diesem Fall müsste die Infostation keine Webseiten selektieren. Diese Vorgehensweise ist jedoch nicht praktikabel, denn die Anzahl der im Dienstgebiet angeforderten Webseiten kann unter Umständen sehr groß sein, was im Gegensatz zu den beschränkten Speicher-Ressourcen mobiler Endgeräte steht. Ebenso steht für die Vorabübertragung nicht beliebig viel Zeit zur Verfügung. Schließlich führt eine

große zu ladende Datenmenge sehr schnell zu einem hohen Energieverbrauch, so dass möglichst nicht zu viele Informationen umsonst geladen werden sollten, auf die der Benutzer nie zugreifen wird.

Aus diesen Gründen muss eine Infostation die Menge der vorab zu ladenden Webseiten durch Verwendung eines geeigneten Filters begrenzen. Hierfür müssen Ordnungskriterien definiert werden, auf Grundlage derer eine Auswahl erfolgen kann. Die am höchsten bewerteten Webseiten werden dann in die Vorabübertragungsliste eingefügt. Dazu muss jedoch bekannt sein, nach welchem Kriterium diese Liste optimiert werden soll. Wie bereits in der Einführung in Kapitel 1 erwähnt, ist es das übergeordnete Ziel der Vorabübertragung, die Zahl der Zugriffe im WWAN zu minimieren, einhergehend mit einer gleichzeitigen Kostenreduktion für den Benutzer und Verringerung der Latenz.

Ein unmittelbar ersichtliches Ordnungskriterium ist es sicherlich, möglichst viele relevante Webseiten vorab zu laden. Eine Webseite ist relevant, wenn ein Benutzer sie mit hoher Wahrscheinlichkeit zukünftig anfordert. Eine hierzu passende Metrik zur Leistungsbewertung des Verfahrens ist die *Trefferrate*, die angibt, wie hoch der Anteil von Anfragen ist, die aus dem Cache beantwortet werden konnten. Nach dieser Metrik ist es besser, statt einer großen relevanten Webseite viele kleine relevante Seiten zu laden. Verwendet man dieses Ordnungskriterium, so kann das übergeordnete Ziel erreicht werden.

Ein weiteres Ordnungskriterium könnte beispielsweise sein, möglichst viele sehr große relevante Webseiten vorab zu laden, so dass im schwach verbundenen Modus (wenn also nur ein WWAN zur Verfügung steht) möglichst nur kleine Seiten über das WWAN nachzuladen sind. Dieses Kriterium hat jedoch den Nachteil, dass beispielsweise durch überlastete Server die Latenz bei Benutzung eines WWANs sehr hoch sein kann, obwohl es sich nur um kleine Webseiten handelt, die ansonsten schnell geladen werden. Auch können in diesem Fall bei Netztrennungen die kleinen Seiten nicht nachgeladen werden.

Aus diesen Gründen wird zur Auswahl von Webseiten für die Vorabübertragungs-

liste das Ordnungskriterium verwendet, möglichst viele relevante Webseiten zu laden. In den nachfolgend beschriebenen Auswahlverfahren wird infolgedessen für jede Webseite ein Relevanzwert definiert, auf dessen Basis die Seiten so geordnet werden, dass die Trefferrate für Benutzer möglichst hoch ist. Diese Auswahlverfahren werden unterschieden in Verfahren mit und ohne Clusterbildung.

3.8.1 Selektion der Webseiten ohne Clusterbildung

Für den Fall, dass von den angeforderten Webseiten jede für sich von Interesse ist, werden nachfolgend drei Auswahlverfahren ohne Clusterbildung vorgestellt. Die ersten beiden beziehen sich auf die Vorabübertragung von (Teil)-Sitzungen. Hier werden Webseiten unabhängig von ihrer Größe in der Reihenfolge übertragen, dass die von der Infostation abgeleiteten Sitzungen aller Benutzer möglichst zusammenhängend geladen werden. Das dritte Verfahren selektiert die vorab zu ladenden Webseiten abhängig von ihrer Zugriffswahrscheinlichkeit und Größe.

Selektion von (Teil-)Sitzungen

Wie bereits in Abschnitt 3.4.4 beschrieben, werden die Sitzungen aller Benutzer, die sich im Dienstgebiet der Infostation bewegen, zur Modellierung des kollektiven Zugriffsverhaltens auf den Informationsgraphen abgebildet. Die folgenden beiden Auswahlverfahren selektieren Webseiten für die Vorabübertragung derart, dass entweder möglichst viele, aber unter Umständen kurze Anfangsteile von beliebten Sitzungen oder möglichst viele beliebte Sitzungen vollständig geladen werden, wobei sich die Beliebtheit einer Sitzung in diesem Fall aus der Popularität ihres Sitzungsstartknotens ableiten lässt. Beide Ansätze wurden in [17] veröffentlicht.

Ein Beispiel soll dies verdeutlichen: Der Informationsgraph aus Abbildung 3.13 enthält die folgenden Sitzungen (A,B,D), (G,F), (G,F,T), (G,F,J,I), (E,F), (E,F,T) und (E,F,J,I). Im ersten Fall werden zuerst alle Anfangsteile der Sit-

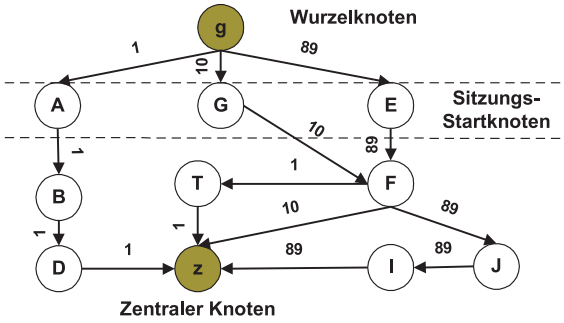


Abbildung 3.13: Im Informationsgraphen enthaltene Sitzungen

zungen mit der Länge 1 geladen, gefolgt von Anfangsteilen der Länge 2, 3 usw., bis die Liste gefüllt ist. Im Beispiel sind dies zuerst die Seiten *E, G, A* (Länge 1)), gefolgt von *F* und *B* (Länge 2), *T*, *J* und *D* (Länge 3) und schließlich *I* (Länge 4).

Im zweiten Fall würde zuerst die Sitzung (*E,F*), gefolgt von (*E,F,J,I*), (*E,F,T*), (*G,F*), (*G,F,J,I*), (*G,F,T*), und (*A,B,D*). Selbstverständlich werden Webseiten nicht doppelt eingefügt, deshalb ist die Reihenfolge in der Vorabübertragungsliste (*E,F,J,I,T,G,A,B,D*).

Beide Algorithmen werden nachfolgend detailliert beschrieben.

(1) Selektion möglichst vieler (Teil-)Sitzungen: Mit Hilfe dieses Auswahlverfahrens sollen möglichst viele der im Informationsgraphen repräsentierten Sitzungen zumindest teilweise vorab geladen werden. Zuerst sollen demnach alle in den Sitzungsstarteinträgen angeforderten Webseiten sortiert nach ihrer Popularität in die Vorabübertragungsliste eingefügt werden, gefolgt von den jeweils in den zweiten Einträgen angeforderten Webseiten. Dies wird so lange fortgesetzt, bis die Liste gefüllt ist. Es entspricht einer Sortierung, die gemäß eines Rundlaufverfahrens (engl. *Round-Robin*) erfolgt.

Als Auswahlverfahren bietet sich in diesem Fall die unter anderem in [26] beschriebene Breitensuche (engl. *Breadth First Search*, BFS) an, die jedoch leicht angepasst werden muss, indem die von einem Knoten $v \in V$ ausgehenden Kanten nach ihrer Kantenwahrscheinlichkeit sortiert werden, bevor die Nachbarn von v in die Warteschlange eingefügt werden. Passt eine Webseite auf Grund ihrer Größe nicht mehr in die Vorabübertragungsliste, so wird die Traversierung trotzdem fortgesetzt, da eventuell noch kleinere Webseiten folgen könnten. Die *modifizierte Breitensuche* wird in Pseudocode in Algorithmus 2 beschrieben, wobei folgende Bezeichner verwendet werden: `IG` ist ein Informationsgraph, $E_{\texttt{out}}(u)$ die Menge der von Knoten $u \in \texttt{IG}.V$ ausgehenden Kanten, `schlange` eine Warteschlange (engl. *queue*) und `Liste` eine Vorabübertragungsliste.

Algorithm 2 Modifizierte Breitensuche

```
1: Liste = (); // leere Liste
2: for all v ∈ IG.V do
3:    v.entferneMarkierung();
4: end for
5: E_out(IG.w).sortiere(); // sortiere alle ausgehenden Kanten des Wurzelknotens IG.w
6: for all e ∈ E_out(IG.w) do
7:    schlange.fügeHinzu(e.zielknoten);
8:    e.zielknoten.setzeMarkierung();
9: end for
10: while schlange enthält Einträge do
11:    u=schlange.Kopf;
12:    if Liste hat noch genügend freien Platz then
13:       Liste.fügeHinzu(u); // Eintrag in Vorabübertragungsliste
14:    end if
15:    E_out(u).sortiere();
16:    for all e ∈ E_out(u) do
17:       if u.istNichtMarkiert() then // Wurde Knoten schon besucht?
18:          schlange.fügeHinzu(e.zielknoten);
19:          e.zielknoten.setzeMarkierung();
20:       end if
21:    end for
22: end while
```

Algorithmus 2 arbeitet wie folgt:

Zeilen 1 bis 4: Die Vorabübertragungsliste wird initialisiert, die Markierung aller Knoten wird entfernt.

Zeilen 5 bis 9: Die vom Wurzelknoten ausgehenden Kanten werden nach ihrer Kantenwahrscheinlichkeit sortiert. Die zum Wurzelknoten benachbarten Sitzungsstartknoten werden in dieser Reihenfolge in die Warteschlange eingefügt und markiert.

Zeilen 10 bis 22: Diese Anweisungen werden ausgeführt, solange noch ein Knoten in der Warteschlange ist.

Zeilen 11 bis 14: Sobald ein Knoten u der Warteschlange entnommen wird, wird er in die Vorabübertragungsliste eingefügt.

Zeilen 15 bis 21: Die u ausgehenden Kanten werden nach ihrer Kantenwahrscheinlichkeit sortiert. Die benachbarten, noch nicht markierten Knoten werden in dieser Reihenfolge in die Warteschlange eingefügt und markiert.

(2) Selektion möglichst langer (Teil-)Sitzungen: Mit diesem Auswahlverfahren sollen überwiegend diejenigen Webseiten vorab geladen werden, die möglichst lange (Teil-)Sitzungen bilden, deren Länge frei wählbar ist. Die ermittelten Sitzungen sollen nach der Popularität der Sitzungen sortiert werden, die sich aus der Beliebtheit der die Sitzung beginnenden Webseite ergibt.

Als Auswahlverfahren bietet sich in diesem Fall die unter anderem in [26] beschriebene Tiefensuche (engl. *Depth First Search*, DFS) an, die jedoch leicht angepasst werden muss, indem die von einem Knoten $v \in V$ ausgehenden Kanten nach ihrer Kantenwahrscheinlichkeit sortiert werden, bevor die Nachbarn von v besucht werden. Passt eine Webseite auf Grund ihrer Größe nicht mehr in die Vorabübertragungsliste, so wird wie bei der modifizierten Breitensuche die Traversierung trotzdem fortgesetzt, da eventuell noch kleinere Webseiten folgen könnten. Die *modifizierte Tiefensuche* wird in Pseudocode in Algorithmus 3

beschrieben, wobei folgende Bezeichner verwendet werden: `IG` ist ein Informationsgraph, $E_{\texttt{out}}(u)$ die Menge der von Knoten $u \in \texttt{IG}.V$ ausgehenden Kanten und `Liste` eine Vorabübertragungsliste. Die maximale Länge der zu ladenden Sitzungen erfolgt durch eine Begrenzung der Suchtiefe.

Algorithm 3 Modifizierte Tiefensuche

```
 1: Liste = (); // leere Liste
 2: for all v ∈ IG.V do
 3:    v.entferneMarkierung();
 4: end for
 5: E_out(IG.w).sortiere(); // sortiere alle ausgehenden Kanten des Wurzelknotens IG.w
 6: for all e ∈ E_out(IG.w) do
 7:    u = e.zielknoten;
 8:    DFSNachfolger(u);
 9: end for

    function DFSNachfolger(Knoten u) {
10: if Suchtiefe erreicht then
11:    return // Rekursion wird abgebrochen
12: end if
13: u.setzeMarkierung();
14: if Liste hat noch genügend freien Platz then
15:    Liste.fügeHinzu(u); // Eintrag in Vorabübertragungsliste
16: end if
17: E_out(u).sortiere();
18: for all e ∈ E_out(u) do
19:    v = e.zielknoten;
20:    if v.istNichtMarkiert() then // Wurde Knoten schon besucht?
21:       DFSNachfolger(v);
22:    end if
23: end for
    }
```

Algorithmus 3 arbeitet wie folgt:

Zeilen 1 bis 4: Die Vorabübertragungsliste wird initialisiert, die Markierung aller Knoten wird entfernt.

Zeilen 5 bis 9: Die vom Wurzelknoten ausgehenden Kanten werden nach ihrer Kantenwahrscheinlichkeit sortiert. Die zum Wurzelknoten benachbarten Sitzungsstartknoten werden in dieser Reihenfolge besucht.

Zeilen 10 bis 12: Die Rekursion wird abgebrochen, sobald die maximale Suchtiefe erreicht wurde.

Zeilen 13 bis 16: Sobald ein Knoten u besucht wird, wird er markiert und in die Vorabübertragungsliste eingefügt.

Zeilen 17 bis 23: Die Funktion *DFSNachfolger* wird für die noch nicht markierten, nach der Kantenwahrscheinlichkeit sortierten Nachbarn von u rekursiv aufgerufen.

Eigenschaften: Die vorgestellten Auswahlverfahren bevorzugen Benutzer, die Webseiten aus populären Sitzungen anfordern, d.h, die Reihenfolge der Webseiten und die Popularität der die Sitzung startende Webseite spielen eine wesentliche Rolle. Unter der Annahme, dass das kollektive Zugriffsverhalten bezüglich der Länge von Sitzungen gemischt ist, eignet sich die modifizierte Breitensuche für Benutzer, die überwiegend Webseiten aus kurzen, populären Sitzungen anfordern, wohingegen die modifizierte Tiefensuche eher für Benutzer mit langen, populären Sitzungen passend ist. Die den Knoten entsprechenden Webseiten werden gemäß der Reihenfolge der Traversierung in die Vorabübertragungsliste eingefügt, die Größe der Webseiten wird demnach nicht berücksichtigt. Da die Sortierung hauptsächlich von der Popularität der Sitzungsstartknoten abhängt, werden eventuell vorhandene beliebtere Pfade zu einem Knoten nicht erkannt. Ein Vorteil ist jedoch die relativ geringe Komplexität, die in $\mathcal{O}(|V|+|E|\cdot\log|E|)$ liegt, wobei der Faktor $\log|E|$ durch die Sortierung der von einem Knoten ausgehenden Kanten entsteht. Dadurch geht leider die lineare Komplexität der zugrunde liegenden elementaren Traversierungsalgorithmen verloren.

Selektion von populären Webseiten

Ein Nachteil der modifizierten Tiefen- und Breitensuche liegt darin, dass die Seitengröße nicht in die Entscheidung mit einbezogen wird, was dazu führen kann, dass unter Umständen eine umfangreiche Webseite einen großen Teil des Caches belegt, der ansonsten mit mehreren kleineren Webseiten hätte gefüllt werden können. Weiterhin erfolgt die Sortierung der vorab zu übertragenden Seiten einzig und allein auf Grundlage der durch die Traversierung erkannten Sitzungen. Dieser Ansatz berücksichtigt jedoch nicht die Beliebtheit einzelner Webseiten, die mit Hilfe der höchsten Zugriffswahrscheinlichkeit der Webseite gemessen werden kann. Diese wiederum entspricht der Pfadwahrscheinlichkeit des zugeordneten Knotens für den beliebtesten Pfad, wobei ein Pfad umso beliebter ist, je höher die Pfadwahrscheinlichkeit des Endknotens ist.

Definition 18 (Zugriffswahrscheinlichkeit einer Webseite) *Die* ***Zugriffswahrscheinlichkeit einer Webseite*** *entspricht der Pfadwahrscheinlichkeit des entsprechenden Knotens im Informationsgraphen.*

Das nachfolgende Beispiel soll dies verdeutlichen.

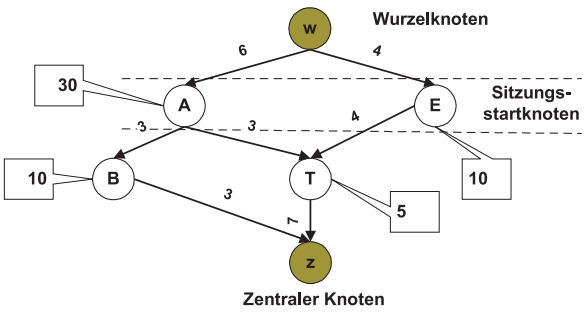


Abbildung 3.14: Mehrere Pfade für den Zugriff auf eine Webseite

Abbildung 3.14 zeigt einen Beispielgraphen, in dem die Kanten mit dem Sequenzzähler und die Knoten mit der URL (z.B. T) und der Größe der entsprechenden

Webseiten annotiert sind. In diesem Graphen führen zwei Pfade zum Knoten T. Betrachtet man den Pfad (A, T), so berechnet sich die Pfadwahrscheinlichkeit von T zu $P(T) = 0{,}3$, wohingegen die Pfadwahrscheinlichkeit von T über E den Wert $P(T) = 0{,}4$ ergibt. Die maximale Zugriffswahrscheinlichkeit für T beträgt somit 40%.

Bewertung der Knoten: Das Ziel des nachfolgend beschriebenen Auswahlverfahrens ist es nun, möglichst viele relevante Webseiten vorab zu übertragen, wobei die Relevanz auf Grundlage ihrer Zugriffswahrscheinlichkeit und Größe bestimmt wird. Die Qualität einer Vorabübertragungsliste ist dabei umso höher, je höher die Trefferrate ausfällt. Webseiten mit hoher Zugriffswahrscheinlichkeit verbessern definitiv die Qualität der Vorabübertragungsliste, belegen aber auch entsprechend ihrer Größe einen mehr oder minder großen Teil des zur Verfügung stehenden Speicherplatzes. Webseiten sollten folglich umso höher bewertet werden, je größer ihre Zugriffswahrscheinlichkeit ist. Zugleich sollten sie umso geringer bewertet werden, je größer sie sind, wodurch ein sparsamer Umgang mit dem zur Verfügung stehenden Speicherplatz gewährleistet wird.

Beim Vergleich zweier Webseiten ist demnach diejenige vorzuziehen, die für die Vorabübertragungsliste den größten Zuwachs an Qualität durch ihre Zugriffswahrscheinlichkeit pro Größeneinheit erbringt, es handelt sich demnach um eine *Relevanz pro Byte*. Nun stellt sich die Frage, ob und wie die beiden Faktoren Zugriffswahrscheinlichkeit und Seitengröße zu gewichten sind, um ein optimales Ergebnis zu erzielen.

Sei $v \in V$ ein der Webseite mit der URL $v.\texttt{ID}$ zugeordneter Knoten im Informationsgraphen mit Größe $v.\texttt{größe}$ und Pfadwahrscheinlichkeit $P_{\texttt{Pfad}}(v)$, die der Zugriffswahrscheinlichkeit der Webseite entspricht. Sei weiterhin $\alpha \geq 0$ ein Parameter zur Gewichtung der Pfadwahrscheinlichkeit. Dann wird die Relevanz pro Byte von v gemäß Gleichung 3.7 berechnet als:

$$R(v) = \frac{(P_{\texttt{Pfad}}(v))^{\alpha}}{v.\texttt{größe}} \tag{3.7}$$

Für $\alpha < 1$ wird die gewichtete Pfadwahrscheinlichkeit immer größer, womit der unmittelbare Eindruck entsteht, dass diese die Relevanz pro Byte immer mehr beeinflusst. Das Gegenteil ist jedoch der Fall, denn die gewichteten Pfadwahrscheinlichkeiten zweier zu vergleichender Knoten streben für $\alpha \to 0$ immer mehr gegen 1, wobei kleinere Pfadwahrscheinlichkeiten den gewichteten Wert wesentlich schneller gegen 1 gehen lassen als größere. Infolgedessen gleichen sie sich für sinkende Werte von α immer stärker an, womit im Extremfall für $\alpha = 0$ die Größe allein die Relevanz pro Byte bestimmt. Der Einfluss der Größe auf die Relevanz pro Byte ist demnach für $\alpha < 1$ höher als der Einfluss des Pfadgewichts. Im Gegenzug wird die Relevanz pro Byte für $\alpha > 1$ durch die Pfadwahrscheinlichkeit stärker beeinflusst als durch die Größe. In einfachen Worten bedeutet dies, dass beim Vergleich zweier Knoten niedrige Pfadwahrscheinlichkeiten mit steigendem α schneller gegen 0 gehen als höhere. Somit wird der Unterschied zwischen zwei Pfadwahrscheinlichkeiten mit steigendem α schnell größer, wodurch schließlich die Größe einer Webseite die Relevanz pro Byte immer weniger beeinflusst.

Das in Tabelle 3.4 angeführte Beispiel soll diese Eigenschaft verdeutlichen. Knoten A ist mit fünf Größeneinheiten nur halb so groß wie Knoten B, hat jedoch eine geringere Pfadwahrscheinlichkeit. Für $\alpha \leq 1$ wird auf Grund der geringeren Größe Knoten A vor B einsortiert. Diese Reihenfolge wird für $\alpha = 2$ umgekehrt, da nun die höhere Pfadwahrscheinlichkeit von B die Relevanz pro Byte wesentlich beeinflusst.

In der durchgeführten experimentellen Evaluation wurden die besten Trefferraten für $\alpha = 1$ ermittelt. Hieraus lässt sich schließen, dass zur Berechnung der Relevanz pro Byte beide Faktoren gleich wichtig sind.

Algorithmus: Da bei der modifizierten Tiefensuche jeder Knoten nur einmal besucht wird, ist es nicht gewährleistet, dass zur Berechnung der Relevanz pro Byte

Tabelle 3.4: Einfluss der Seitengröße auf die Relevanz pro Byte eines Knotens

	A	B
Größe	5	10
Pfadwahrscheinlichkeit	0,3	0,55
R(v) für $\alpha = 0{,}5$	0,110	0,074
R(v) für $\alpha = 1$	0,06	0,055
R(v) für $\alpha = 2$	0,018	0,030

der zugehörigen Seite die höchste Zugriffswahrscheinlichkeit verwendet wird. Um diese zu bestimmen, wird ein Auswahlverfahren benötigt, das alle möglichen Pfade im Informationsgraphen untersucht. Nun hat solch ein Algorithmus im schlimmsten Fall eine Zeitkomplexität, die in $\mathcal{O}(|V|!)$ liegt. Zur Einschränkung des Aufwands wird deshalb eine Heuristik benötigt, mit der nur für relevante Knoten die höchste Pfadwahrscheinlichkeit ermittelt wird. Ein Knoten wird als relevant betrachtet, wenn seine zugeordnete Webseite eine genügend hohe Zugriffswahrscheinlichkeit hat. Die nachfolgend vorgeschlagene *begrenzte Pfadsuche* basiert auf der rekursiven Tiefensuche und verwendet zur Bestimmung der höchsten Pfadwahrscheinlichkeit relevanter Knoten einen Schwellwert $\texttt{min}_{P(\texttt{Pfad})}$, der die *minimale Pfadwahrscheinlichkeit* darstellt, die ein Pfad haben muss, damit die Rekursion fortgeführt wird.

Da bei der begrenzten Pfadsuche Knoten mehrfach besucht werden können und ein Informationsgraph nicht zwingend azyklisch sein muss, tritt die Problematik einer eventuell auftretenden Endlosschleife bei der Traversierung auf. Aus diesem Grund wird jedem Knoten eine Markierung zugewiesen, die gesetzt wird, sobald der Knoten im rekursiven Abstieg expandiert wird. Sie wird zurückgesetzt, sobald das Rücksetzverfahren (engl. *backtracking*) beginnt.

Algorithmus 4 stellt die begrenzte Pfadsuche in Pseudocode dar. Es werden folgende Bezeichner verwendet: `IG` ist ein Informationsgraph, $u, v \in \texttt{IG}.V$ sind zwei Knoten im Informationsgraphen, $Adj(v)$ ist die Menge der Knoten, die adjazent zu v sind, $P(u, v)$ ist die Kantenwahrscheinlichkeit der Kante von u nach v, $\texttt{min}_{P(\texttt{Pfad})}$ ist ein Schwellwert für das minimale Pfadgewicht und `Liste`

ist die Vorabübertragungsliste. Damit der Algorithmus effizient arbeitet, ist eine Erweiterung der in Abschnitt 3.5.3 eingeführten Datenstruktur für einen Knoten notwendig. Ein Knoten hat somit folgende Attribute:

Definition 19 (Knotenattribute für Webseiten (begrenzte Pfadsuche))
Ein Knoten $v \in V$ im Informationsgraphen besitzt folgende ***Attribute****.*

v.ID: *Global eindeutiger Bezeichner der angeforderten Webseite;*
v.größe: *Größe der Webseite in Byte;*
v.anfrageZähler: *Anfragezähler;*
v.inhaltsZähler: *Inhaltszähler;*
v.pfadwahrscheinlichkeit: *höchste Pfadwahrscheinlichkeit.*

Der Algorithmus arbeitet wie folgt:

Zeilen 1 bis 5: Die Vorabübertragungsliste wird initialisiert und die Markierung jedes Knotens wird zurückgesetzt. Gleichzeitig wird die maximale Pfadwahrscheinlichkeit mit 0 initialisiert.

Zeilen 6 bis 10: Für jeden Sitzungsstartknoten wird die rekursive Funktion *DFS-Nachfolger* aufgerufen, falls das Kantengewicht größer als der Schwellwert ist.

Zeilen 11 bis 12: Die in der Vorabübertragungsliste enthaltenen Knoten werden gemäß der berechneten Relevanz pro Byte sortiert.

Zeile 13: Bei Aufruf der Funktion *DFSNachfolger* wird der Knoten markiert.

Zeilen 14 bis 16: Falls der besuchte Knoten noch nicht in der Vorabübertragungsliste enthalten ist, wird er eingetragen.

Zeilen 17 bis 21: Die Rekursion bricht ab und das Rücksetzverfahren (engl. *backtracking*) beginnt, sobald eine der folgenden Bedingungen erfüllt ist:

1. Ein markierter Knoten wurde erreicht. In diesem Fall wurde ein Zyklus erkannt.

Algorithm 4 Begrenzte Pfadsuche

```
1: Liste = (); // leere Vorabübertragungsliste
2: for all v ∈ IG.V do
3:    v.entferneMarkierung();
4:    v.pfadwahrscheinlichkeit = 0;
5: end for
6: for all n ∈ Adj(IG.w) do
7:    if P(x,n) > min_{P(Pfad)} then
8:       DFSNachfolger(n, P(IG.w,n));
9:    end if
10: end for
11: berechneRelevanzProByte(Liste);
12: Liste.sortiere();

    function DFSNachfolger(Knoten x, float pfadwahrscheinlichkeit){
13: x.markiere(); // zur Zyklenerkennung
14: if x ist nicht in Liste then
15:    Liste.fügeHinzu(x);
16: end if
17: for all n ∈ Adj(x) do // Rekursion: besuche alle Nachfolger
18:    if ¬n.istMarkiert() ∧ n ≠ z ∧ pfadwahrscheinlichkeit · P(x,n) >
       min_{P(Pfad)} then
19:       DFSNachfolger(n, pfadwahrscheinlichkeit · P(x,n));
20:    end if
21: end for
22: x.entferneMarkierung(); // Rücksetzverfahren: alle Nachfolger besucht
23: if x.pfadwahrscheinlichkeit < pfadwahrscheinlichkeit then // merke
    nur die höchste Pfadwahrscheinlichkeit
24:    x.pfadwahrscheinlichkeit = pfadwahrscheinlichkeit;
25:    Liste.ersetze(x);
26: end if
    }

    function berechneRelevanzProByte(Vorabübertragungsliste Liste){
27: for all v ∈ Liste do
28:    v.relevanz = v.pfadwahrscheinlichkeit / v.größe
29: end for
    }
```

2. Der nächste zu besuchende Knoten ist der zentrale Knoten z, der keine ausgehenden Kanten hat.

3. Die Pfadwahrscheinlichkeit des Pfads von der Wurzel zum nächsten zu besuchenden Knoten ist kleiner als der Schwellwert $\texttt{min}_{P(\texttt{Pfad})}$.

Zeilen 22 bis 26: Wenn das Rücksetzverfahren beginnt, wird die Knotenmarkierung wieder zurückgesetzt. Falls sich die maximale Pfadwahrscheinlichkeit des Knotens geändert hat, wird sie mit der aktuellen überschrieben und der Knoten wird in der Liste ersetzt.

Zeilen 27 bis 29: Für jeden Knoten in der Liste wird dessen Relevanz pro Byte berechnet.

Eigenschaften: Die begrenzte Pfadsuche eignet sich für Benutzer, die überwiegend populäre Webseiten mit einer hohen Zugriffswahrscheinlichkeit anfordern. Ein Nachteil des Verfahrens ist sicherlich die höhere Zeitkomplexität, die in Abschnitt 3.10.3 diskutiert wird. Durch die eingeführte Heuristik kann diese zwar verringert werden, wodurch jedoch nicht immer optimale Lösungen gefunden werden. Insbesondere hängt die Güte des Verfahrens wesentlich vom Schwellwert für die Pfadwahrscheinlichkeit ab. Ist er zu niedrig angesetzt, erhöht sich die Zeitkomplexität, wird er zu hoch gewählt, verringert sich die Qualität der Resultate. Weiterhin wird angenommen, dass von den angeforderten Webseiten jede für sich von Interesse ist, so dass keine Cluster gebildet werden müssen. Für den Fall der Clusterbildung muss die begrenzte Pfadsuche wie nachfolgend beschrieben erweitert werden.

3.8.2 Selektion der Webseiten mit Clusterbildung

In Abschnitt 3.2 wurde die Problemstellung diskutiert, dass Webseiten in manchen Fällen semantisch so stark zusammenhängen können, dass sie für einen Benutzer nur als Gruppe interessant sind. Die bislang vorgestellten Auswahl-

verfahren betrachten Webseiten als atomare Einheiten für die Vorabübertragung. Nachfolgend wird deshalb ein Auswahlverfahren zur Verfügung gestellt, auf Grundlage dessen Cluster gebildet werden können, die dann vollständig geladen werden.

Wie bereits in Kapitel 3.5.2 beschrieben, können gemäß Definition 11 Webseiten in Inhalts- und Transitseiten eingeteilt werden. Das nachfolgend beschriebene und in [19] vorgestellte *clusterbasierte Auswahlverfahren* verwendet diese Klassifizierung für die Erzeugung von Clustern. Da eine Inhaltsseite auf mehreren Pfaden erreicht werden kann, können für eine solche Seite unter Umständen mehrere Cluster gebildet werden.

Als Ordnungskriterium für das Einfügen von Clustern in die Vorabübertragungsliste wird ein geeigneter Relevanzwert für Cluster definiert.

Taxonomie von Verfahren zur Clusterbildung: Vor der Beschreibung des clusterbasierten Auswahlverfahrens wird eine Taxonomie von Verfahren zur Clusterbildung erstellt, in die das Verfahren schließlich eingeordnet wird. In der

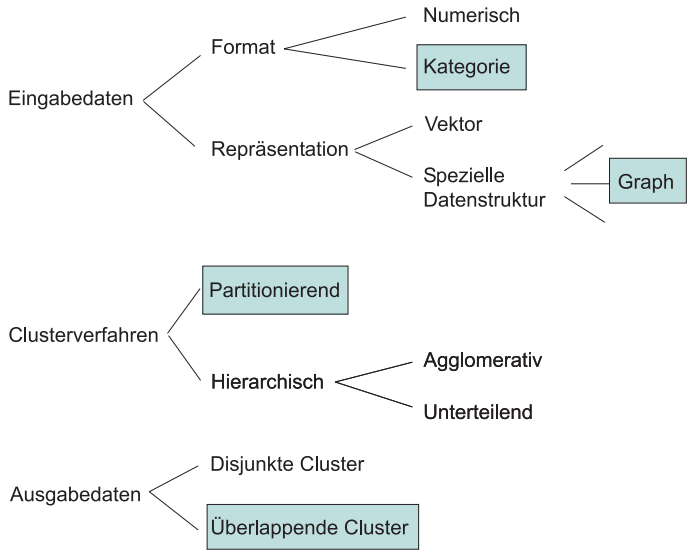


Abbildung 3.15: Klassifikation von Verfahren zur Clusterbildung

Literatur findet sich eine Vielzahl von Ansätzen zur Clusterbildung für die unterschiedlichsten Problemstellungen. Abbildung 3.15 zeigt die nachfolgend verwendete Taxonomie von Verfahren zur Clusterbildung, die auf den Vorschlägen von Jain et al. in [49] und [48] basiert.

Die Klassifizierung erfolgt auf Grundlage dreier Hauptkriterien:

Eingabedaten: Die Art der Daten ist ein wichtiger Faktor zur Auswahl eines Verfahrens zur Clusterbildung. So ist zur Berechnung einer Distanzfunktion das *Datenformat* entscheidend. Mit numerischen Werten können beispielsweise Euklidische Distanzen berechnet werden. Bei Werten, die einer Kategorie zugeordnet werden, müssen andere Bewertungskriterien bestimmt werden, wie zum Beispiel die semantische Distanz. Eine weitere wichtige Rolle spielt die *Repräsentation* der Daten bei der Clusterbildung. So können für Daten in Vektordarstellung andere Algorithmen eingesetzt werden als für Daten, die in einer Graphstruktur vorliegen.

Ausgabedaten: Verfahren zur Clusterbildung unterscheiden sich maßgeblich in der Art der erzeugten Cluster. In disjunkten Clustern werden im Gegensatz zu überlappenden Clustern die Elemente genau einem Cluster zugeordnet.

Clusterbildung: Die Algorithmen zur Clusterbildung werden in der Literatur häufig in hierarchische und partitionierende Verfahren unterteilt (siehe beispielsweise [48]).

Hierarchische Verfahren bilden eine geschachtelte Reihe von Clustern. Das Resultat eines solchen Verfahrens wird am besten durch ein Dendrogramm visualisiert. Der Cluster auf der obersten Ebene enthält alle Elemente, die Cluster auf Blattebene enthalten genau ein Element. Hierarchische Cluster können entweder agglomerativ (bottom-up) oder unterteilend (top-down) erstellt werden. Hierzu wird eine Ähnlichkeitsmatrix benötigt, die für jedes Paar den Abstand angibt. Je kleiner der für die Einteilung in Cluster maßgebliche Referenzabstand gewählt wird,

um so mehr Cluster entstehen. Mit hierarchischen Verfahren werden nur disjunkte Cluster erzeugt. Die Zeitkomplexität liegt beispielsweise beim Single-Link Verfahren in $\mathcal{O}(n^2 \log n)$, wobei n die Anzahl der Elemente ist.

Partionierende Verfahren unterteilen eine Menge von Elementen hingegen nur einmal in eine Menge von Clustern. Ein häufig verwendetes Verfahren ist der *k-means-Algorithmus*, bei dem zu Anfang die Anzahl k der zu bildenden Cluster festgelegt wird. Zunächst werden zufällig k Clusterzentren gewählt und jedes Element dem Cluster zugeordnet, zu dessen Clusterzentrum es die geringste Distanz hat. Nun werden die Clusterzentren neu berechnet und überprüft, ob die Elemente nicht doch näher bei einem anderen Clusterzentrum liegen. Diese Schritte werden so oft wiederholt, bis eine zufrieden stellende Einteilung erfolgt ist. Der Algorithmus muss nicht unbedingt konvergieren, denn im ungünstigsten Fall kann ein Cluster leer bleiben, wodurch sich das Clusterzentrum nicht mehr berechnen lässt. In diesem Fall muss der Algorithmus mit neu zu definierenden Clusterzentren neu gestartet werden. Partitionierende Verfahren können disjunkte und überlappende Cluster bilden. Graphentheoretische Verfahren zur Clusterbildung werden ebenfalls dieser Kategorie zugeordnet.

Das clusterbasierte Auswahlverfahren wird folgendermaßen in diese Taxonomie eingeordnet: Als Eingabedaten verwendet der Clusteralgorithmus Webseiten, die in zwei *Kategorien* eingeteilt sind. Die Zugriffsmuster bezüglich Transit- und zugehöriger Inhaltsseiten werden durch einen Pfad im *Informationsgraphen* repräsentiert. Nun kann eine Inhaltsseite auf mehreren Pfaden erreicht werden, womit sich die Forderung nach *überlappenden Clustern* als Ausgabe des Algorithmus ergibt. Da der Graph sehr groß werden kann, sollte der zeitliche Aufwand möglichst linear in der Anzahl der Knoten und Kanten sein. Hierarchische Verfahren kommen nicht in Frage, da sie nur disjunkte Cluster bilden. Es wird also ein *partitionierendes* Verfahren zur Clusterbildung eingesetzt.

Um einen Cluster im Informationsgraphen formal zu beschreiben, muss zunächst die semantische Nähe von zwei Knoten beschrieben werden.

Semantische Nähe von Knoten: Die semantische Nähe von Knoten im Informationsgraphen wird auf Grundlage der Klassifizierung von Webseiten in Transit- und Inhaltsseiten gemäß Definition 20 und dem Pfadgewicht der Knoten berechnet. Sie ist die Grundlage für die Erzeugung von Clustern, die, wie bereits erwähnt, aus mindestens einer Inhaltsseite und den dazugehörenden Transitseiten bestehen. Entsprechend dem Schwellwert $\texttt{min}_{P(\texttt{Pfad})}$, der bei der begrenzten Pfadsuche zur Bestimmung relevanter Knoten verwendet wird, ist es eine Grundvoraussetzung für die Modellierung der semantischen Nähe zwischen zwei Knoten x und y, dass das Pfadgewicht von y über x diesen Schwellwert $\texttt{min}_{P(\texttt{Pfad})}$ nicht unterschreitet. Dies bedeutet, dass die dem Knoten y zugeordnete Webseite eine genügend hohe Zugriffswahrscheinlichkeit haben muss, damit y semantisch nah zu x ist. Des Weiteren muss modelliert werden, dass eine Transitseite zu einer nachfolgenden Transit- oder Inhaltsseite eine stärkere semantische Nähe aufweist, als eine Inhaltsseite zu einer beliebigen nachfolgenden Seite. Im ersten Fall ist die Transitseite für die nachfolgende Seite unbedingt notwendig, da sie den Pfad zu einer Inhaltsseite bilden könnte. Im zweiten Fall kann für die Inhaltsseite ein eigener Cluster erzeugt werden. Sie könnte jedoch für eventuell nachfolgende Inhaltsseiten als Transitseite benötigt werden.

Definition 20 (Semantische Nähe) *Sei V die Knotenmenge und E die Kantenmenge des Informationsgraphen. Seien $x, y \in V$ und $y \neq x$. Seien weiterhin $\texttt{min}_{P(\texttt{Pfad})}$ ein minimales Pfadgewicht und $P(x, y)$ die Kantenwahrscheinlichkeit der Kante (x, y). Dann gilt für die* ***semantische Nähe*** $\texttt{semNähe}(x, y)$*:*

$$\texttt{semNähe}(x,y) = \begin{cases} 1 & \textit{wenn } P_{\texttt{Pfad}}(x) \cdot P(x,y) > \texttt{min}_{P(\texttt{Pfad})} \wedge P_{\texttt{Inhalt}}(x) = 0 \\ 0,5 & \textit{wenn } P_{\texttt{Pfad}}(x) \cdot P(x,y) > \texttt{min}_{P(\texttt{Pfad})} \wedge P_{\texttt{Inhalt}}(x) > 0 \\ 0 & \textit{wenn } P_{\texttt{Pfad}}(x) \cdot P(x,y) \leq \texttt{min}_{P(\texttt{Pfad})} \end{cases}$$

Da die semantische Nähe nicht exakt berechnet werden muss, ist diese einfache Art der Modellierung für das Verfahren zur Clusterbildung ausreichend.

Die beiden durch eine Kante $e = (x, y)$ verbundenen Knoten x und y sind *semantisch sehr nahe* (die semantische Nähe ist 1), wenn x ein Transitknoten ist und die Pfadwahrscheinlichkeit für den zu y fortgesetzten Pfad größer ist als der Schwellwert. In diesem Fall kann ein Cluster für x nicht erzeugt werden, da jeder Cluster mit einem Inhaltsknoten abschließen muss.

Die beiden oben angeführten Knoten x und y sind *semantisch mäßig nahe* (die semantische Nähe zwischen ihnen ist 0,5), wenn ein Cluster bei x endet, die Rekursion aber fortgesetzt wird, da die geforderte minimale Pfadwahrscheinlichkeit weiter zu y nicht unterschritten wird und eventuell noch weitere Inhaltsseiten folgen könnten.

Die Knoten x und y sind *semantisch nicht nah* (die semantische Nähe zwischen ihnen ist gleich 0), wenn die Pfadwahrscheinlichkeit kleiner als der Schwellwert ist. Dadurch ist y kein relevanter Knoten, wodurch gemäß der Forderung, dass nur relevante Knoten betrachtet werden, keine semantische Beziehung zwischen x und y besteht.

Aufbauend auf der semantischen Nähe kann nun die Struktur von Clustern definiert werden.

Clusterstruktur: Ein Cluster enthält eine Folge von Knoten, nachfolgend *Knotenliste* genannt, die aus mindestens einem Inhaltsknoten mitsamt den dazugehörenden Transitknoten besteht, die den Pfad von der Wurzel des Graphen zu dem Inhaltsknoten bilden. Der letzte Knoten der Folge muss ein Inhaltsknoten sein. Für jeden Inhaltsknoten wird ein eigener Cluster erstellt, wodurch sich Cluster überlappen können, wie zu Beginn dieses Abschnittes erwähnt wurde. In einem Informationsgraphen gibt es keine Cluster mit gleicher Knotenliste.

Definition 21 (Cluster) *Ein **Cluster** ist ein Tupel*

$$C = \langle L, P(\texttt{Pfad}), P(\texttt{Inhalt}), \texttt{größe} \rangle$$

wobei gilt:

$L = \langle v_1, \ldots v_m \rangle$*: Die Knotenliste L ist ein Tupel, bestehend aus den Knoten $v_1, \ldots, v_m \in V$, die den Pfad vom Sitzungsstartknoten v_1 zum für die Erzeugung des Clusters maßgeblichen Inhaltsknoten v_m bilden. Es existiert eine Reihenfolge, die den Pfad wiedergibt, wobei gilt:*

- *v_m ist der für die Erzeugung des Clusters maßgebliche Inhaltsknoten*
- *$P_{\texttt{Inhalt}}(v_m) > 0$ (siehe Definition 14 eines Inhaltsknotens). Da ein Cluster nur mit einem Inhaltsknoten enden kann und Inhaltsknoten unterschiedliche Inhaltswahrscheinlichkeiten haben können, symbolisiert die Inhaltswahrscheinlichkeit von v_m die Wahrscheinlichkeit, mit welcher der Cluster endet.*
- *$\forall i, 1 \leq i < m : semNähe(v_i, v_{i+1}) > 0$*

$P(\texttt{Pfad})$*: Die Pfadwahrscheinlichkeit des Clusters beschreibt die Wahrscheinlichkeit, dass die für die Erzeugung des Clusters verantwortliche Inhaltsseite von einem Benutzer bzw. einer Benutzergruppe auf dem durch L gebildeten Pfad aufgerufen wird. Sie wird mittels Gleichung 3.5 berechnet.*

$P(\texttt{Inhalt})$*: Die Inhaltswahrscheinlichkeit des Clusters definiert die Wahrscheinlichkeit, mit der ein potenzieller Benutzer die dem für die Clustererzeugung maßgeblichen Knoten zugeordnete Webseite als Inhaltsseite betrachtet:*

$$P(\texttt{Inhalt}) = P_{\texttt{Inhalt}}(v_m)$$

`größe`*: Die Größe des Clusters ist die aufsummierte Größe aller Webseiten, die*

von den in der Knotenliste enthaltenen Knoten repräsentiert werden, wobei gilt:

$$C.\texttt{größe} = \sum_{i=1}^{m} C.L.v_i.\texttt{größe}$$

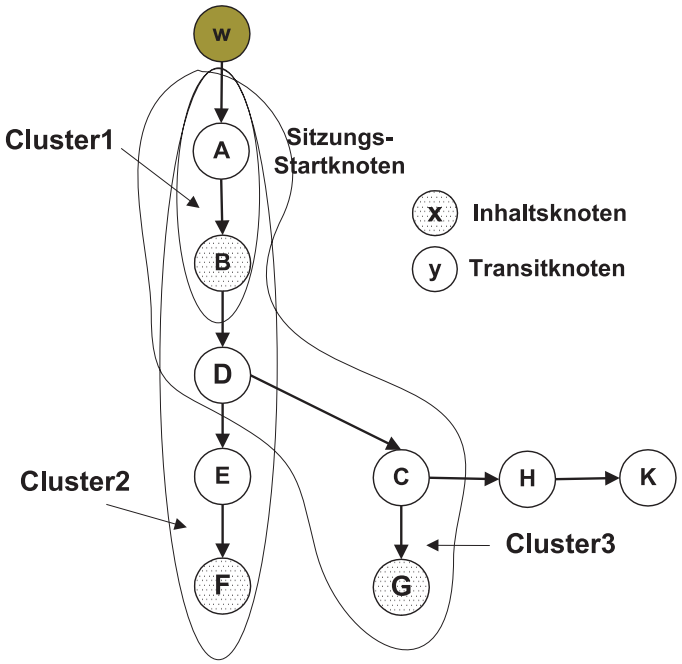


Abbildung 3.16: Ausschnitt eines Informationsgraphen mit Clustern

Abbildung 3.16 zeigt einen Ausschnitt eines Informationsgraphen, in dem die Inhaltsknoten schraffiert sind. In dieser Abbildung sind drei Cluster zu erkennen. `Cluster1` wird folgendermaßen definiert: $\texttt{Cluster1}.L = \langle A, B \rangle$, wobei B der für die Erzeugung des Clusters maßgebliche Knoten ist. Die Pfadwahrscheinlichkeit des Clusters wird berechnet als $\texttt{Cluster1}.P(\texttt{Pfad}) = P(w, A) \cdot P(A, B)$. Die Clustergröße ist $\texttt{Cluster1.größe} = A.\texttt{größe} + B.\texttt{größe}$. Analog hierzu werden `Cluster2` und `Cluster3` definiert, wobei `Cluster2` aus den Knoten *A, B, D, E, F* besteht und `Cluster3` *A, B, D, C, G* enthält.

Bewertung der Cluster: Da eine Inhaltsseite auf mehreren Pfaden erreicht werden kann, entstehen durch die Clusterbildung eine Reihe von nicht unbedingt disjunkten Clustern, die für die Erzeugung der Vorabübertragungsliste sortiert werden müssen. Diese Cluster unterscheiden sich in ihrer Pfad- und Inhaltswahrscheinlichkeit sowie ihrer Größe.

Bei der in Abschnitt 3.8.1 beschriebenen begrenzten Pfadsuche wurde die Relevanz einer Webseite über deren Zugriffswahrscheinlichkeit definiert. Bei der Clusterbildung muss nun zusätzlich dessen Inhaltswahrscheinlichkeit berücksichtigt werden. Die *Relevanz eines Clusters* stellt folglich die Wahrscheinlichkeit dar, mit der alle darin enthaltenen Webseiten in der durch die Knotenliste charakterisierten Reihenfolge angefordert werden und gleichzeitig die zuletzt angeforderte Webseite als Inhaltsseite eingestuft wird. Hieraus ergibt sich nun die Frage, ob und wie die beiden Faktoren Pfad- und Inhaltswahrscheinlichkeit zu gewichten sind. Hat ein Cluster eine hohe Pfadwahrscheinlichkeit, aber eine sehr geringe Inhaltswahrscheinlichkeit, wird die für dessen Erzeugung maßgebliche Inhaltsseite zwar von sehr vielen Benutzern aufgerufen, aber von den meisten als Transitseite betrachtet, womit das Clusterende an diesem Knoten eine geringe Wahrscheinlichkeit hat.

Entsprechend der begrenzten Pfadsuche wird schließlich auch beim clusterbasierten Auswahlverfahren die *Relevanz pro Byte* eines Clusters als Sortierkriterium verwendet, die wie folgt definiert ist:

Definition 22 (Relevanz und Relevanz pro Byte eines Clusters) *Sei C ein Cluster und v_m der für seine Erzeugung maßgebliche Inhaltsknoten. Seien weiterhin α und β mit $\alpha, \beta \geq 0$ zwei Parameter zur Gewichtung der beiden Relevanzfaktoren. Dann berechnet sich die* ***Relevanz*** *von C als*

$$r(C) = (C.P(\texttt{Pfad}))^{\alpha} \cdot (C.P(\texttt{Inhalt}))^{\beta}$$

Schließlich wird als Metrik für die Bewertung eines Clusters die ***Relevanz pro***

Byte *gemäß Gleichung 3.8 berechnet:*

$$R(C) = \frac{r(C)}{C.\texttt{größe}} = \frac{(C.P(\texttt{Pfad}))^{\alpha} \cdot (C.P(\texttt{Inhalt}))^{\beta}}{C.\texttt{größe}} \tag{3.8}$$

Wie in der Leistungsbewertung in Abschnitt 5.4.8 näher erläutert wird, sind beide Parameter α und β gleichermaßen wichtig, denn die besten Resultate werden für $\alpha = \beta = 1$ erzielt. Ist beispielsweise $\alpha = 1$ und $\beta < 1$, so beeinflusst die gewichtete Inhaltswahrscheinlichkeit die Relevanz weniger und stärkt somit den Einfluss der Pfadwahrscheinlichkeit, denn die gewichteten Inhaltswahrscheinlichkeiten beider Cluster streben für $\beta \to 0$ immer mehr gegen 1, wobei kleinere Inhaltswahrscheinlichkeiten den gewichteten Wert wesentlich schneller gegen 1 gehen lassen als größere. Infolgedessen gleichen sich die gewichteten Inhaltswahrscheinlichkeiten von Clustern für sinkende Werte von β immer stärker an, womit im Extremfall für $\beta = 0$ die Pfadwahrscheinlichkeit allein die Relevanz bestimmt.

Tabelle 3.5: Einfluss der Pfadwahrscheinlichkeit und der Inhaltswahrscheinlichkeit auf die Relevanz eines Clusters

	$C1$	$C2$
Pfadwahrscheinlichkeit	0,9	0,5
Inhaltswahrscheinlichkeit	0,5	1,0
Relevanz r(C) für $\alpha = 0{,}5, \beta = 1$	0,47	0,71
Relevanz r(C) für $\alpha = 1, \beta = 0{,}5$	0,64	0,5
Relevanz r(C) für $\alpha = \beta = 1$	0,45	0,5

Tabelle 3.5 zeigt die Auswirkungen unterschiedlicher Gewichtungen für zwei Beispielcluster $C1$ und $C2$. $C1$ hat eine hohe Pfadwahrscheinlichkeit von 0,9, seine Inhaltswahrscheinlichkeit ist jedoch nur 0,5. Im Gegensatz hierzu hat $C2$ eine Pfadwahrscheinlichkeit von 0,5 bei einer Inhaltswahrscheinlichkeit von 1,0. Für den Fall $\alpha = 0{,}5$ und $\beta = 1{,}0$ beeinflusst die Inhaltswahrscheinlichkeit die Relevanz stärker, wodurch die Relevanz von $C2$ höher ist als die von $C1$. Im Gegensatz hierzu ist für $\alpha = 1{,}0$ und $\beta = 0{,}5$ der Einfluss der Pfadwahrscheinlichkeit größer, wodurch der Relevanzwert von $C1$ höher ist als der von $C2$. Wie bei der begrenzten Pfadsuche beeinflussen α und β auch den Einfluss der Clustergröße

auf die Relevanz pro Byte. Dies wurde bereits in Abschnitt 3.8.1 beschrieben und wird deshalb an dieser Stelle nicht erneut diskutiert.

Clusterbildung: Nun steht mit der in Abschnitt 3.8.1 beschriebenen begrenzten Pfadsuche, die für die Berechnung der höchsten Pfadwahrscheinlichkeit relevanter Knoten sämtliche relevanten Pfade findet (d.h. Pfade, die nur relevante Knoten beinhalten), bereits eine Grundlage zur Berechnung von sich überlappenden Clustern zur Verfügung. Das clusterbasierte Auswahlverfahren erweitert die begrenzte Pfadsuche, indem es Inhaltsseiten und die dazu gehörenden Transitseiten in Clustern gruppiert. Durch die Begrenzung des Verfahrens mit Hilfe eines Schwellwertes für die Pfadwahrscheinlichkeit eines Knotens werden nur relevante Knoten betrachtet. Infolgedessen findet der Algorithmus nicht unbedingt alle möglichen Cluster, sondern nur die Teilmenge von Clustern, die relevante Knoten enthalten. Weitere Details der Implementierung können der Diplomarbeit von Pfahl [82] entnommen werden.

Damit der Algorithmus effizient arbeitet, ist eine Erweiterung der in Definition 12 (siehe Abschnitt 3.5.3) eingeführten Datenstruktur für einen Knoten notwendig. Ein Knoten hat somit folgende Attribute:

Definition 23 (Knotenattribute für Webseiten (Clusterbildung)) *Sei $v \in V$ ein Knoten im Informationsgraphen. Dann besitzt v die folgenden* ***Attribute****:*

v.ID: *Global eindeutiger Bezeichner der angeforderten Webseite;*
v.größe: *Größe der Webseite in Byte;*
v.anfrageZähler: *Anfragezähler;*
v.inhaltsZähler: *Inhaltszähler;*
v.clusterListe: *Liste aller Cluster, in denen v enthalten ist. Knoten können zwar in mehreren Clustern vorkommen, die zugeordneten Webseiten werden jedoch nur einmal in die Vorabübertragungsliste eingefügt. Diese Liste dient der einfachen Überprüfung, in welchen Clustern ein Knoten*

enthalten ist.

v.inListe: *Dieses Attribut wird auf wahr gesetzt, wenn die zugeordnete Webseite bereits in der Vorabübertragungsliste enthalten ist. Dies dient der einfachen Überprüfung, ob die einem Knoten gehörende Webseite bereits in der Vorabübertragungsliste enthalten ist.*

Algorithmus 5 beschreibt die Erzeugung von Clustern in Pseudocode, wobei folgende Bezeichner verwendet werden: `IG` ist ein Informationsgraph, $u, v \in \texttt{IG}.V$ sind zwei Knoten im Informationsgraphen, $Adj(v)$ ist die Menge der Knoten, die adjazent zu v sind, $P(u, v)$ ist die Kantenwahrscheinlichkeit der Kante von u nach v, $\texttt{min}_{P(\texttt{Pfad})}$ ist ein Schwellwert für die minimale Pfadwahrscheinlichkeit und `listeAllerCluster` ist eine Liste, die alle erstellten Cluster enthält.

Algorithmus 5 arbeitet wie folgt:

Zeilen 1 bis 6: Initialisierung

Zeilen 7 - 11: Für jeden Sitzungsstartknoten, dessen semantische Nähe zum Wurzelknoten größer als 0 ist, wird die rekursive Funktion *clusterNachfolger* aufgerufen.

Zeilen 12 bis 14: Jeder besuchte Knoten wird markiert und in die Knotenliste aufgenommen. Die Größe aller Webseiten auf dem Pfad wird um die Größe der Webseite des besuchten Knotens inkrementiert.

Zeilen 15 bis 19: Die Rekursion bricht ab und das Rücksetzverfahren beginnt, sobald eine der folgenden Bedingungen erfüllt ist:

1. Ein markierter Knoten wurde erreicht. In diesem Fall wurde ein Zyklus erkannt.
2. Der nächste zu besuchende Knoten ist der zentrale Knoten z, der keine ausgehenden Kanten hat.
3. Der Vorgängerknoten und der besuchte Knoten sind semantisch nicht nah.

Algorithm 5 Algorithmus zur Clusterbildung

```
1: listeAllerCluster = (); // leere Liste
2: for all v ∈ IG.V do
3:    v.entferneMarkierung();
4:    v.inListe = falsch;
5:    v.clusterListe = ();
6: end for
7: for all n ∈ Adj(IG.w) do
8:    if semNähe(w,n) > 0 then // siehe Definition 20
9:       clusterNachfolger(n, () , 0, P(w, n) ); // „()" ist leere Knotenliste
10:   end if
11: end for

    function clusterNachfolger(Knoten x, Knotenliste L, int größe, float
    pfadwahrscheinlichkeit){
12: x.markiere(); // zur Zyklenerkennung
13: L.fügeHinzu(x);
14: größe += x.größe();
15: for all n ∈ Adj(x) do // Rekursion: besuche alle Nachfolger
16:    if ¬n.istMarkiert() ∧ n ≠ z ∧ semNähe(x, n) > 0 then
17:       clusterNachfolger(n, L, größe, pfadwahrscheinlichkeit · P(x, n));
18:    end if
19: end for
20: x.entferneMarkierung(); // Rücksetzverfahren: alle Nachfolger besucht
21: if P_Inhalt(x) > 0 then // Inhaltsknoten
22:    erstelleCluster(x, L, größe, pfadwahrscheinlichkeit);
23: end if}

    function erstelleCluster(Knotenliste L, int größe, float
    pfadwahrscheinlichkeit){
24: Cluster C = new Cluster(L);
25: C.setzeGröße(größe);
26: C.setzePfadwahrscheinlichkeit(pfadwahrscheinlichkeit);
27: v_m = L.letzterEintrag();
28: C.setzeInhaltswahrscheinlichkeit(P_Inhalt(v_m));
29: v_m.clusterListe.fügeHinzu(C);
30: berechneRelevanzProByte(C); // gemäß Definition 22
31: listeAllerCluster.fügeHinzu[C);
    }
```

Zeile 20: Alle Nachfolger von x wurden betrachtet, der Algorithmus befindet sich jetzt beim Aufstieg aus der Rekursion. Die Markierung des Knotens wird entfernt, so dass dieser Knoten eventuell auf einem weiteren Pfad wieder besucht werden kann.

Zeilen 21 bis 23: Ist der besuchte Knoten ein Inhaltsknoten, so wird ein Cluster erzeugt.

Zeilen 24 bis 29: Die Funktion *erstelleCluster* erzeugt einen Cluster und setzt die entsprechenden Attribute.

Zeile 30 bis 31: Die Relevanz pro Byte des erstellten Clusters wird berechnet und der Cluster in die Liste aller Cluster aufgenommen.

Erstellen der Vorabübertragungsliste: Nachdem alle relevanten Cluster ermittelt wurden, müssen diese entsprechend ihrer Relevanz pro Byte sortiert werden, damit die zugeordneten Webseiten in die Vorabübertragungsliste aufgenommen werden können. Da ein Knoten in mehreren Clustern enthalten sein kann und eine Webseite nur einmal in die Vorabübertragungsliste aufgenommen wird, muss einfach geprüft werden können, ob die zugeordnete Webseite nicht bereits über einen höher bewerteten Cluster in die Vorabübertragungsliste aufgenommen wurde. Zu diesem Zweck wird das in Definition 23 spezifizierte Knotenattribut $v.\texttt{inListe}$ eingeführt. Bevor eine Webseite in die Vorabübertragungsliste eingefügt wird, wird geprüft, ob das Attribut des entsprechenden Knotens gesetzt ist. Falls nicht, wird die Seite eingefügt und das Attribut auf `wahr` gesetzt. Auf diese Weise wird eine Prüfung auf Enthaltensein einer Webseite in der Vorabübertragungsliste vermieden.

Da Cluster nicht disjunkt sind, genügt es nicht, zum Befüllen der Vorabübertragungsliste die Cluster genau einmal zu sortieren und die entsprechenden Webseiten dann der Reihe nach in die Vorabübertragungsliste einzufügen. Durch die Bewertung der Cluster mittels Relevanz pro Byte kann sich diese Reihenfolge ändern, wenn Teile eines Clusters bereits in der Vorabübertragungsliste sind,

Tabelle 3.6: Relevanz pro Byte vor und nach dem Einfügen eines Clusters in die Vorabübertragungsliste

	C1	**C2**	**C3**	**C4**
Vor dem Einfügen				
$r(Ci)$	0,5	0,45	0,4	0,2
Ci**.größe [KByte]**	15	23	18	12
$R(Ci)$	0,03333	0,0196	0,02222	0,01666
Nach dem Einfügen von $C1$				
$r(Ci)$	0,5	0,4	0,4	0,2
Ci**.größe [KByte]**	-	8	18	12
$R(Ci)$	-	0,05625	0,02222	0,01666

da in diesem Fall dessen Größe verringert wird. Dieses mehrmalige Bewerten und Sortieren von Clustern wird nachfolgend als *iteratives Sortieren* bezeichnet, bei dem nach jedem Einfügen eines Clusters in die Vorabübertragungsliste die Relevanz pro Byte für alle betroffenen Cluster neu berechnet wird.

Ein Beispiel soll dies veranschaulichen. Gegeben seien vier Cluster *C1, C2*, *C3* und *C4* mit den Knotenlisten $C1.L = \{A, B\}$, $C2.L = \{A, B, C, D\}$, $C3.L = \{E, F, G\}$, $C4.L = \{X, Y\}$. Sei weiterhin H eine Vorabübertragungsliste mit 35 KBytes Speicherplatz. Tabelle 3.6 beinhaltet die Bewertung der obigen Beispielcluster vor und nach dem erstmaligen Einfügen des am höchsten bewerteten Clusters in die Vorabübertragungsliste.

Würden die Cluster nur einmal sortiert und dann in dieser Reihenfolge in die Vorabübertragungsliste H übernommen, wären in $H = \langle A, B, E, F, G\}$ die Cluster *C1* und *C3* enthalten und in der Vorabübertragungsliste noch 2 KBytes frei, die nicht mehr genutzt werden können. Nun hat jedoch Cluster *C2* mit $r(C) = 0{,}45$ einen etwas höheren Relevanzwert als *C3* mit $r(C) = 0{,}4$, der nur wegen dessen geringerer Größe vor *C2* eingefügt wurde. Durch das Einfügen von *C1* ändert sich jedoch die Größe von *C2*, da die Seiten A und B nicht mehr berücksichtigt werden müssen, wodurch wiederum dessen Relevanz pro Byte steigt. Nach dem Einfügen von *C1* wird somit *C2* höher bewertet als *C3* und in die Vor-

abübertragungsliste eingefügt. Beim nächsten Durchlauf wird festgestellt, dass *C3* nicht mehr in die Vorabübertragungsliste passt, dafür aber *C4*. Die Vorabübertragungsliste $H = \{A, B, C, D, X, Y\}$ enthält schließlich drei Cluster und ist restlos gefüllt. Hierdurch steigt die Wahrscheinlichkeit, dass der Benutzer relevante Webseiten in seinem Cache vorfindet, denn zum einen wurden mehr Seiten geladen als im Fall des einmaligen Sortierens, zum anderen ist mit *C2* auch ein Cluster enthalten, dessen Relevanzwert höher ist als der des nicht eingefügten Clusters *C3*.

Algorithmus 6 beschreibt die Erzeugung von Clustern in Pseudocode, wobei folgende Bezeichner verwendet werden: `H` ist eine Vorabübertragungsliste, $v \in V$ ein Knoten im Informationsgraphen, c ein Cluster und `listeAllerCluster` eine Liste, die alle erstellten Cluster enthält.

Algorithm 6 Iteratives Sortieren

```
1:  while H.freierPlatz()> 0 und listeAllerCluster ≠ () do
2:    C_max = listeAllerCluster.sucheClusterMitHöchsterRelevanzProByte();
3:    if H.freierPlatz()≥ C_max.größe then // ist noch genügend Platz in H?
4:      for all v ∈ C_max.L do // Knoten in der Knotenliste des Clusters
5:        if v.inListe == falsch then
6:          H.fügeHinzu(v.ID); // Füge die dem Knoten entsprechende Web-
            seite in die Vorabübertragungsliste ein
7:          v.inListe = wahr;
8:          for all c ∈ v.clusterListe do
9:            c.größe = c.größe − v.größe
10:           berechneRelevanzProByte(c);
11:         end for
12:       end if
13:     end for
14:     listeAllerCluster.entferneCluster(C_max);
15:   end if
16: end while
```

Algorithmus 6 arbeitet wie folgt:

Zeilen 1 bis 16: Solange in der Vorabübertragungsliste noch freier Platz ist und

die Liste aller Cluster noch ein Cluster enthält, werden die folgenden Schritte ausgeführt:

Zeile 2: Das Cluster mit der höchsten Relevanz pro Byte wird bestimmt.

Zeilen 3 bis 15: Falls der am höchsten bewertete Cluster noch in die Vorabübertragungsliste passt, werden die folgenden Schritte ausgeführt:

Zeilen 4 bis 13: Für alle in der Knotenliste des Clusters enthaltenen Knoten werden die folgenden Schritte ausgeführt:

Zeilen 5 bis 7: Falls die dem Knoten zugeordnete Webseite noch nicht in der Vorabübertragungsliste ist, wird diese eingefügt und das Knotenattribut `inListe` auf *wahr* gesetzt.

Zeilen 8 bis 11: Anschließend wird für jeden Cluster in der Clusterliste des Knotens die Größe angepasst und die Relevanz pro Byte neu berechnet.

Zeile 14: Der am höchsten bewertete Cluster wird aus der Liste aller Cluster entfernt.

3.9 Nutzungsprofile

Im vorgestellten Vorabübertragungsverfahren wird die Entscheidung, welche Informationen in die Vorabübertragungsliste aufgenommen werden sollen, pro Gebiet getroffen. Als Konsequenz hiervon wird für den Aufbau der Vorabübertragungsliste das Zugriffsverhalten aller Benutzer analysiert, die sich im Dienstgebiet einer Infostation aufhalten. Weisen all diese Benutzer ein vergleichbares Zugriffsverhalten auf, können durchaus zufriedenstellende Ergebnisse bezüglich der Trefferrate erzielt werden, wie in Kapitel 5.4 zu sehen sein wird. Dies ist jedoch nicht der Fall, wenn sich das Zugriffsverhalten von Benutzergruppen maßgeblich unterscheidet. Aus diesem Grund werden nachfolgend Nutzungsprofile zur Optimierung der Selektion bei unterschiedlichem Zugriffsverhalten berücksichtigt.

3.9.1 Modellierung des Wissens über das Zugriffsverhalten von Benutzergruppen

Wie bereits in Abschnitt 3.3.2 beschrieben, modelliert eine Infostation das Wissen über das Zugriffsverhalten aller Benutzer in ihrem Dienstgebiet mit Hilfe eines Informationsgraphen. Zur Integration von Nutzungsprofilen in diese Modellierung bieten sich zwei Möglichkeiten: (1) Ein Multigraph, der mehrere Kanten zwischen zwei Knoten zulässt und somit alle Nutzungsprofile verwaltet und (2) ein Informationsgraph pro Nutzungsprofil. Nachfolgend werden beide Vorgehensweisen einander kurz gegenübergestellt.

Multigraph: Die Clusterbildung kann bei Verwendung eines Multigraphen auf zwei Arten erfolgen. Bei der einmaligen Traversierung werden die Cluster während der Traversierung durch Verknüpfung der entsprechenden Kanten erzeugt, bei der mehrmaligen Traversierung wird der Multigraph für jedes im Profilmix enthaltene Nutzungsprofil einmal traversiert.

Bei einem Multigraphen werden nachfolgend diejenigen Kanten, die zwischen denselben Knoten verlaufen, eine identische Orientierung besitzen und dem gleichen Nutzungsprofil im aktuellen Profilmix zugeordnet sind, als *Einzelkanten* bezeichnet.

Wird der Multigraph nur ein einziges Mal traversiert, müssen die zusammengehörenden Einzelkanten zu einer *Gesamtkante* zusammengefasst werden. Deren Kantenwahrscheinlichkeit wird dann als Summe über die Produkte der Kantenwahrscheinlichkeiten der zusammengehörenden Einzelkanten mit dem prozentualen Anteil des entsprechenden Nutzungsprofils wie folgt berechnet. Seien $x, y \in V$ zwei Knoten, $\mathcal{N}_b$ ein Profilmix, $n_i \in \mathcal{N}_b$ ein im Profilmix enthaltenes Nutzungsprofil und $e_i = (x, y)_i$ diejenige Einzelkante zwischen x und y, die dem Nutzungsprofil n_i zugeordnet ist. Dann wird die Kantenwahrscheinlichkeit der Gesamtkante $e = (x, y)$ berechnet als $P(e) = \sum_{i=1}^{|\mathcal{N}_b|} P(e_i) \cdot P(n_i)$. Die Pfadwahrscheinlichkeit eines Clusters berechnet sich schließlich aus dem Produkt der

Kantenwahrscheinlichkeiten der Gesamtkanten.

Das in Abbildung 3.17 dargestellte Beispiel soll dies verdeutlichen. Der Profilmix des Benutzers enthält die beiden Profile *p1* und *p2*. Der prozentuale Anteil von *p1* am Profilmix ist $P(p1) = 0{,}9$, der entsprechende Anteil für *p2* ist $P(p2) = 0{,}1$. Die als durchgezogene Linien markierten Kanten betreffen Pro-

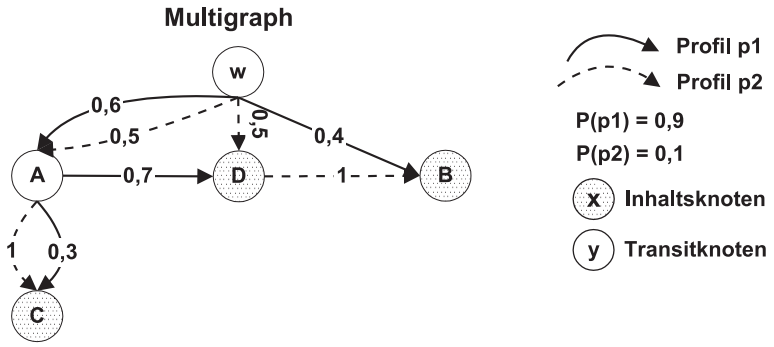


Abbildung 3.17: Integration von Nutzungsprofilen mit Hilfe eines Multigraphen

fil *p1*, die gestrichelten Kanten Profil *p2*. Die Kantenbeschriftungen zeigen die Kantenwahrscheinlichkeiten für die entsprechenden Profile. Zwischen den Knoten A und C gibt es zwei zusammengehörende Einzelkanten. Die Pfadwahrscheinlichkeit von Cluster C_1, das die Knoten A und C enthält, berechnet sich dann beispielsweise zu $C_1.P(\texttt{Pfad}) = (0{,}6 \cdot 0{,}9 + 0{,}5 \cdot 0{,}1) \cdot (0{,}3 \cdot 0{,}9 + 1 \cdot 0{,}1) = 0{,}1625$.

In diesem Beispiel kann man erkennen, dass durch den Einsatz von Multigraphen Cluster gebildet werden können, die so in den einzelnen Profilen gar nicht erzeugt worden wären. So kann beispielsweise Cluster (A, D, B) gebildet werden, das weder in *p1* noch in *p2* vorkommt.

Die einmalige Traversierung kommt jedoch nicht in Frage, denn die Pfadwahrscheinlichkeit des Clusters entspricht nicht der ursprünglichen Definition der Pfadwahrscheinlichkeit eines Clusters, wie sie in Abschnitt 3.8.2 definiert ist. Gemäß Definition 21 beschreibt die Pfadwahrscheinlichkeit eines Clusters die Wahrscheinlichkeit, dass die für die Erzeugung des Clusters verantwortliche In-

haltsseite *von einem Benutzer bzw. einer Benutzergruppe* auf dem durch die Knotenliste gebildeten Pfad aufgerufen wird. Dies kann bei der einmaligen Traversierung jedoch nicht gewährleistet werden, denn in diesem Fall wird die Wahrscheinlichkeit berechnet, mit der ein Benutzer mit einem bestimmten Profilmix diese Folge von Webseiten anfordert. Dies resultiert daraus, dass die prozentualen Anteile der Nutzungsprofile am Profilmix bereits bei der Berechnung der Kantenwahrscheinlichkeit der Gesamtkante berücksichtigt werden müssen.

Aus diesem Grund muss der Multigraph für die Selektion relevanter Cluster $|\mathcal{N}_b|$-mal traversiert werden (einmal für jedes im Profilmix enthaltene Nutzungsprofil). Dadurch können Cluster mit gleicher Knotenliste existieren, deren Pfadwahrscheinlichkeit die Wahrscheinlichkeit beschreibt, dass die für dessen Erzeugung maßgebliche Inhaltsseite von der Benutzergruppe mit dem entsprechenden Nutzungsprofil auf dem gegebenen Pfad angefordert wurde. Bezogen auf das obige Beispiel ergeben sich zwei Cluster, die jeweils die Knoten A und C beinhalten. Zusätzlich zur Relevanz pro Byte eines jeden Clusters muss nun noch zur Bewertung der Cluster die Wahrscheinlichkeit des zugehörigen Nutzungsprofils berücksichtigt werden.

Der Speicheraufwand für einen Multigraphen liegt in $\mathcal{O}(|V| + |\mathcal{N}_b| \cdot |E|)$, wobei V die Menge der Knoten ist, deren zugehörige Webseiten im Dienstgebiet der Infostation angefordert werden. Im schlechtesten Fall sind jeder Knoten und jede Kante in jedem Nutzungsprofil enthalten, was für die Kanten den Faktor $|\mathcal{N}_b| \cdot |E|$ ergibt.

Einzelgraphen: Für die Selektion relevanter Cluster muss jeder Einzelgraph traversiert werden. Analog zu der $|\mathcal{N}_b|$-maligen Traversierung eines Multigraphen können Cluster mit gleicher Knotenliste entstehen. Abbildung 3.18 zeigt entsprechend die Modellierung des Zugriffsverhaltens aus dem obigen Beispiel aus Abbildung 3.17 mit Hilfe von Einzelgraphen.

Bei der Verwendung von Einzelgraphen kann eine Webseite von mehreren Knoten (aus unterschiedlichen Einzelgraphen) repräsentiert werden. Bezogen auf das

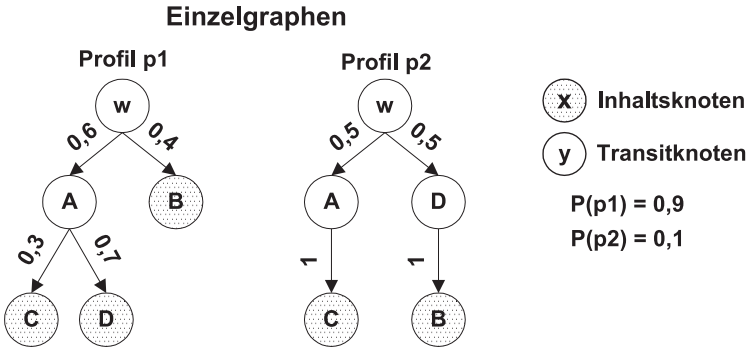


Abbildung 3.18: Integration von Nutzungsprofilen mit Hilfe von Einzelgraphen

Beispiel gilt dies für alle darin enthaltenen Webseiten. In diesem Fall muss beim Einfügen einer Webseite in die Vorabübertragungsliste geprüft werden, ob sie nicht schon durch einen anderen am Profilmix beteiligten Knoten darin enthalten ist. Zur einfachen Überprüfung kann eine Hashtabelle geführt werden, die zu jeder Webseite die Knoten speichert, von denen sie repräsentiert wird. Wird nun eine Webseite in die Vorabübertragungsliste eingefügt, so wird die Markierung aller entsprechenden Knoten in den Einzelgraphen $\texttt{IG}_n.v.\texttt{inListe} = \mathit{wahr}$ gesetzt, wobei $\texttt{IG}_n$ den Informationsgraphen des Profils n darstellt. Der Speicheraufwand liegt in $\mathcal{O}(|\mathcal{N}_\lfloor| \cdot (|V| + |E|))$, da im schlechtesten Fall jeder Knoten in jedem Einzelgraphen enthalten ist.

Der besseren Übersichtlichkeit wegen wird in dieser Arbeit für jedes Nutzungsprofil $n \in \mathcal{N}_\lfloor$ ein separater Informationsgraph verwaltet. Hierbei erfolgt eine eindeutige Zuordnung eines Nutzungsprofils auf einen Informationsgraphen.

3.9.2 Aktualisierung der Einzelgraphen

Aus dem ermittelten Profilmix $\mathcal{N}_b$ eines Benutzers b werden zunächst für alle darin enthaltenen Nutzungsprofile n die zugehörigen Informationsgraphen $\texttt{IG}_n$ bestimmt. Anschließend werden im Rahmen der Aktualisierung der Einzelgra-

phen sämtliche in der Protokolldatei enthaltenen Anforderungen von Webseiten den jeweiligen Informationsgraphen zugewiesen. Da eine einzelne Anforderung nachträglich nicht mehr einem bestimmten Nutzungsprofil zugeordnet werden kann, erfolgt diese Zuordnung gemäß dem prozentualen Anteil des entsprechenden Nutzungsprofils im Profilmix.

Sei $S(s, R) = \{l_1, \ldots, l_m\}$ eine Sitzung, die während der Analyse der von Benutzer b übermittelten Protokolldatei identifiziert wurde und $P(n)$ der Anteil des Nutzungsprofils n im Profilmix $\mathcal{N}_b$. Dann werden bei der Graphaktualisierung die als Knoten- und Kantenattribute verwendeten Zähler gemäß Gleichung 3.9 erhöht.

$$\texttt{hilfsZähler} = \texttt{hilfsZähler} + P(n) \tag{3.9}$$

Für die Aktualisierung aller am Profilmix beteiligten Informationsgraphen $\texttt{IG}_n, \forall n \in \mathcal{N}_b$, wird der in Abschnitt 3.7 beschriebene Algorithmus 1 ausgeführt.

3.9.3 Erzeugung der Vorabübertragungsliste

Die Traversierung eines Einzelgraphen inklusive Clusterbildung erfolgt nach dem in Abschnitt 3.8.2 beschriebenen Algorithmus 5 immer dann, wenn eine Epoche endet. Um einen Cluster eindeutig dem Informationsgraphen $\texttt{IG}_n$ zuordnen zu können, durch dessen Traversierung er erstellt wurde, muss die Definition eines Clusters erweitert werden zu $C = \langle L, P(\texttt{Pfad}), P(\texttt{Inhalt}), \texttt{größe}, n\rangle$, wobei n das Nutzungsprofil kennzeichnet, das $\texttt{IG}_n$ zugeordnet ist. Nach der Traversierung aller Einzelgraphen liegt als Resultat eine Menge von Clustern vor, deren Knotenlisten im Gegensatz zur Clusterbildung ohne Nutzungsprofile nicht mehr unbedingt unterschiedlich sein müssen. Dies kann mit Hilfe des Beispiels aus Abbildung 3.18 veranschaulicht werden, in dem der Cluster (A, C) in beiden Einzelgraphen erzeugt wird. Entsprechend der Gesamtkante, die bei der einmaligen Traversierung eines Multigraphen berechnet wird, werden die Cluster mit gleicher Knotenliste, nachfolgend *Teilcluster* genannt, zu einem *Gesamtcluster*

zusammengefasst.

Definition 24 (Gesamtcluster, Teilcluster) ***Teilcluster(L)*** *ist die Menge der Cluster mit gleicher Knotenliste L, die aus der Traversierung der Einzelgraphen entstanden sind.* ***Gesamtcluster(L)*** *ist der Cluster, der die gleiche Knotenliste wie die in Teilcluster(L) enthaltenen Cluster besitzt und dessen Pfad- und Inhaltswahrscheinlichkeit aus den entsprechenden Wahrscheinlichkeiten der Teilcluster(L) wie folgt berechnet werden.*

Seien $\mathcal{N}_b$ der Profilmix des Benutzers b, $n \in \mathcal{N}_b$ ein Nutzungsprofil im Profilmix und Teilcluster(L) $= \{C_1, \ldots C_j\}$, mit $j \leq |\mathcal{N}_b|$. Sei weiterhin $P(n)$ der Anteil des Nutzungsprofils n im Profilmix, wobei gilt: $\sum_{n \in \mathcal{N}_b} P(n) = 1$. Dann berechnen sich die Pfad- und Inhaltswahrscheinlichkeit des Gesamtclusters(L) als Summe der Produkte aus den Pfad- bzw. Inhaltswahrscheinlichkeiten der Teilcluster(L) mit der Wahrscheinlichkeit des zugehörigen Nutzungsprofils wie folgt:

$$C_{\texttt{ges}}(L).P(\texttt{Pfad}) = \sum_{n \in \mathcal{N}_b} (C_n.P(\texttt{Pfad}) \cdot P(n)) \tag{3.10}$$

$$C_{\texttt{ges}}(L).P(\texttt{Inhalt}) = \sum_{n \in \mathcal{N}_b} (C_n.P(\texttt{Inhalt}) \cdot P(n)) \tag{3.11}$$

Schließlich wird die Relevanz pro Byte von $C_{\texttt{ges}}(L)$ folgendermaßen berechnet:

$$R(C_{\texttt{ges}}(L)) = \frac{(C_{\texttt{ges}}(L).P(\texttt{Pfad}))^{\alpha} \cdot (C_{\texttt{ges}}(L).P(\texttt{Inhalt}))^{\beta}}{C_{\texttt{ges}}(L).\texttt{größe}} \tag{3.12}$$

Zum effizienten Auffinden von Teilclustern wird von jedem Cluster C ein Hashwert aus der Knotenliste berechnet, der als Clusterattribut $C.\texttt{hash}$ gespeichert wird und zur Prüfung auf Gleichheit verwendet wird. Der Algorithmus zur Erzeugung der Gesamtcluster wird nachfolgend in Pseudocode beschrieben, wobei `listeAllerCluster` die Liste aller erzeugten Teilcluster, $P(C.n)$ der prozentuale Anteil des Nutzungsprofils n am Profilmix von Cluster C und

`listeGesamtcluster` die Liste der erzeugten Gesamtcluster darstellt.

Algorithm 7 Erzeugung von Gesamtclustern

```
 1: listeGesamtcluster = ();
 2: listeAllerCluster.sortiere();
 3: C_ref = listeAllerCluster.erstesElement();
 4: C = listeAllerCluster.nächstesElement();
 5: while listeAllerCluster ist noch nicht abgearbeitet do
 6:    summePfadW = C_ref.P(Pfad) * P(C.n);
 7:    summeInhaltW = C_ref.P(Inhalt) * P(C.n);
 8:    while C.hash == C_ref.hash do
 9:       summePfadW+ = C.P(Pfad) * P(C.n);
10:       SummeInhaltW+ = C.P(Inhalt) * P(C.n);
11:       C = listeAllerCluster.nächstesElement();
12:    end while
13:    C_Ges = ⟨C_ref.L, summePfadW, summeInhaltW, C_ref.größe, 0⟩
14:    listeGesamtcluster.fügeHinzu(C_Ges);
15:    C_ref = C;
16: end while
```

Der Algorithmus arbeitet wie folgt:

Zeile 2: Die Liste aller Teilcluster wird nach dem aus der Knotenliste der Cluster berechneten Hashwert sortiert, um die zusammengehörenden Teilcluster einfach zu ermitteln.

Zeilen 3 bis 4: C_{ref} stellt den Referenzcluster dar, mit dem die nachfolgenden verglichen werden.

Zeile 5: Die Liste aller Teilcluster wird vollständig durchsucht:

Zeilen 6 und 7: Die Pfad- und Inhaltswahrscheinlichkeit des neuen Gesamtclusters werden mit den Werten des Referenzclusters initialisiert.

Zeilen 8 bis 11: Solange die Knotenlisten des aktuellen Clusters und des Referenzclusters übereinstimmen, wird die Pfad- und Inhaltswahrscheinlichkeit des Gesamtclusters aktualisiert.

Zeilen 13 bis 15: Der Gesamtcluster wird erzeugt und in die Liste aller Gesamtcluster eingefügt.

Die Vorabübertragungsliste wird abschließend mittels des in Abschnitt 3.8.2 beschriebenen iterativen Sortierens der Liste aller Gesamtcluster gebildet. Da bei der Einbeziehung von Profilen in das Auswahlverfahren der Profilmix eines Benutzers zur Berechnung der Relevanz eines Clusters verwendet wird, kann die Sortierung nicht mehr nur einmal pro Epoche erfolgen, wenn eine für den aktuellen Benutzer optimale Trefferrate erzielt werden soll. In diesem Fall wird die Clusterliste für jeden Benutzer sortiert.

3.10 Qualitative und quantitative Eigenschaften

Im folgenden Abschnitt werden zunächst die qualitativen Eigenschaften des vorgestellten Verfahrens diskutiert, insbesondere dessen Adaptivität und Skalierbarkeit. Anschließend wird die Komplexität des Verfahrens als quantitative Eigenschaft ermittelt.

3.10.1 Adaptivität hinsichtlich Informationsbedarf und Informationsraum

Das Vorabübertragungsverfahren passt sich an Änderungen des Informationsbedarfs von Benutzern an, wobei der Grad der Anpassung mit Hilfe unterschiedlicher Parameter eingestellt werden kann. Des Weiteren kann das generische Verfahren für beliebige schwach strukturierte Informationsräume spezialisiert werden.

Informationsbedarf: Der Informationsgraph modelliert das kollektive Zugriffsverhalten aller Benutzer, die sich im Dienstgebiet der Infostation aufhalten. Die

Aktualisierung des Graphen erfolgt immer dann, wenn eine Protokolldatei an die Infostation übertragen wird. Hierdurch wird gewährleistet, dass das Wissen der Infostation über das Zugriffsverhalten ständig aktuell ist. Durch die zusätzliche Integration einer Alterungsfunktion wird die dynamische Anpassung des Graphen an Veränderungen im Zugriffsverhalten der Benutzer ermöglicht. Diese erhalten also primär diejenigen Informationen auf ihr Endgerät, die zur Zeit der Vorabübertragung an einem bestimmten Ort bevorzugt angefordert werden. Die in dieser Arbeit vorgeschlagene Alterungsfunktion *exponentiell gewichteter gleitender Mittelwert* kann mit Hilfe des Glättungsfaktors an das Zugriffsverhalten angepasst werden, indem die älteren Informationen über das Zugriffsverhalten mehr oder weniger stark gewichtet werden. Nun kann sich das Zugriffsverhalten auch saisonal ändern. Werden beispielsweise in einer Stadt regelmäßig tagsüber überwiegend Informationen zu touristischen Attraktionen abgerufen und abends überwiegend Informationen über kulturelle Veranstaltungen, so kann dies mit Hilfe der in Anhang A.2 diskutierten Glättungsfunktion *doppelt exponentiell gewichteter gleitender Mittelwert* modelliert werden. Diese Glättungsfunktion wird analog zum exponentiell gewichteten gleitenden Mittelwert berechnet, wobei noch eine *Saisonkomponente* hinzukommt, die jedoch hier nicht weiter erläutert wird.

Informationsraum: Das vorgestellte generische Verfahren kann durch eine geeignete Spezialisierung an unterschiedlichste Arten von schwach strukturierten Informationen angepasst werden. Für einen konkreten Anwendungsfall müssen Beziehungen zwischen den Informationen definiert werden, die für die Ableitung von Sitzungen maßgeblich sind, hierfür müssen die Einträge in der Protokolldatei gegebenenfalls erweitert werden. Falls es Informationen gibt, die semantisch so stark zusammen gehören, dass sie nur als vollständige Gruppe interessant sind, müssen Beziehungen und Metriken zur Bestimmung der semantischen Nähe von Informationen spezifiziert werden. Darauf aufbauend müssen die Attribute der Knoten und Kanten des Informationsgraphen an die definierten Beziehungen angepasst, sowie ein geeignetes Verfahren zur Aktualisierung des Informations-

graphen zur Verfügung gestellt werden. Schließlich muss die Selektion vorab zu übertragender Informationen durch geeignete Auswahlverfahren mit und ohne Clusterbildung an den konkreten Anwendungsfall angepasst werden.

3.10.2 Skalierbarkeit

Die Skalierbarkeit des Verfahrens wird von zwei Faktoren bestimmt: (1) Die Größe des Informationsgraphen hat vor allem Auswirkungen auf den Aufwand der Selektionsverfahren und (2), die Anzahl von Benutzern im Dienstgebiet einer Infostation beeinflusst die im Übertragungsgebiet zur Verfügung stehende Bandbreite und die Last einer Infostation hinsichtlich der Aktualisierung des Graphen.

Größe des Informationsgraphen: Wie bereits in Abschnitt 3.4.4 erwähnt, repräsentiert jeder Knoten im Informationsgraphen eine Informationsanforderung. Bei wachsendem oder sich ständig änderndem Informationsbedürfnis der Benutzer kann die Anzahl der Knoten und Kanten sehr schnell ansteigen. Zur Begrenzung des Speicherbedarfs für die Informationsgraphen und des Aufwands für deren Verwaltung können diejenigen Knoten und Kanten entfernt werden, die eine bestimmte Zeit lang nicht aktualisiert wurden. Der zeitliche Aufwand für die Selektion kann durch eine geeignete Wahl des Schwellwertes begrenzt werden, da die Auswahlverfahren mit und ohne Clusterbildung nur relevante Knoten bzw. Cluster betrachten (siehe Abschnitte 3.8.1 und 3.8.2).

Anzahl der Benutzer: In dieser Arbeit wurde für die Kommunikation zwischen Infostation und mobilen Endgeräten ein pull-basierter Ansatz verfolgt. Das heißt, die vorab zu übertragenden Informationen werden mittels einer Punkt-zu-Punkt Verbindung individuell für jedes Endgerät von den Infostationen herunter geladen. Dieser Ansatz skaliert jedoch nicht mit wachsender Anzahl der von einer Infostation zu bedienenden mobilen Endgeräte. Zur Optimierung der Kommu-

nikation bietet sich in diesem Fall ein push-basierter Ansatz an, bei dem eine Infostation die vorab zu ladenden Informationen periodisch rundsendet. Die mobilen Endgeräte filtern dann die für sie relevanten Informationen heraus und speichern diese im lokalen Cache (siehe auch Kapitel 8). Außerdem kann die Bandbreite durch die Verteilung mehrerer Zugangspunkte (eng. *access points*) erhöht werden.

Ein weiteres Problem mit einer hohen Ankunftsrate von Protokolldateien ergibt sich dadurch, dass die Informationsgraphen jedesmal aktualisiert werden, sobald eine Protokolldatei empfangen wird. Dies kann zur Überlastung einer Infostation führen, wenn gleichzeitig die Selektionsprozesse in kleineren Zeitabständen ausgeführt werden. Dieser Fall könnte durch die Verteilung der Prozesse (Aktualisierung und Selektion) auf mehrere Rechner verhindert werden.

3.10.3 Komplexität

Nachfolgend wird die Zeitkomplexität der Algorithmen ermittelt, die für die einzelnen Schritte des Vorabübertragungsverfahrens notwendig sind. Dies sind insbesondere die Aktualisierung des Informationsgraphen und die Selektion vorab zu ladender Webseiten mit Clusterbildung.

Aktualisierung des Informationsgraphen

Sei $\mathcal{N}$ die Menge der Nutzungsprofile und L eine Protokolldatei mit einer durchschnittlichen Anzahl von $|L|$ Einträgen. Seien weiterhin $|V|$ die Menge der Knoten und $|E|$ die Menge der Kanten eines Informationsgraphen. Jedes Laden einer Protokolldatei auf eine Infostation bedingt die Aktualisierung von $|\mathcal{N}|$ Informationsgraphen. Für jeden Eintrag $l \in \mathcal{L}$ muss die Knoten- und Kantenliste des Informationsgraphen durchsucht werden. Diese Suche kann in linearer Zeit erfolgen, falls die Listen unsortiert sind. Bei Verwendung einer günstigen Hashfunktion kann die Suche jedoch in konstanter Zeit erfolgen, im schlechtesten Fall

Tabelle 3.7: Maximale Anzahl von Clustern in Abhängigkeit von der Länge ihrer Knotenliste

Länge der Knotenliste eines Clusters	Anzahl der Cluster
$\|V\|$	$\|V\| * (\|V\| - 1) * ... * 3 * 2 * 1 = \frac{\|V\|!}{(\|V\|-\|V\|)!} = \frac{\|V\|!}{0!}$
$\|V\| - 1$	$\|V\| * (\|V\| - 1) * ... * 3 * 2 = \frac{\|V\|!}{(\|V\|-(\|V\|-1))!} = \frac{\|V\|!}{1!}$
$\|V\| - 2$	$\|V\| * (\|V\| - 1) * ... * 3 = \frac{\|V\|!}{(\|V\|-(\|V\|-2))!} = \frac{\|V\|!}{2!}$
$\vdots$	
2	$\|V\| * (\|V\| - 1) = \frac{\|V\|!}{(\|V\|-2)!}$
1	$\|V\| = \frac{\|V\|!}{(\|V\|-1)!}$

ist sie wieder linear. Dann liegt die Zeitkomplexität der Graphaktualisierung im besten Fall in $\mathcal{O}(|\mathcal{N}| \cdot |\mathcal{L}|)$, im schlechtesten Fall in $\mathcal{O}(|\mathcal{N}| \cdot |\mathcal{L}| \cdot (|V| + |E|))$.

Auswahlverfahren

Die Verwendung der begrenzten Pfadsuche für die Clusterbildung liefert im schlechtesten Fall (wenn es sich um einen stark zusammenhängenden Graphen handelt und kein Schwellwert verwendet wird) $|V|!$ Pfade vom Wurzelknoten w zum zentralen Knoten z mit der Länge $|V|$. Handelt es sich bei allen Knoten um Inhaltsknoten, wird die Anzahl an erzeugten Clustern wie folgt berechnet. Tabelle 3.7 zeigt für jede mögliche Pfadlänge die maximale Anzahl an gebildeten Clustern. Die Summe aller Cluster beträgt somit

$$\texttt{sumCluster} = |V|! * \sum_{i=0}^{|V|-1} \frac{1}{i!}$$

Der Wert der Summe konvergiert für $|V| \rightarrow \infty$ gegen die Eulersche Zahl ($e \approx 2,718$) (siehe auch [71]). Es werden im schlechtesten Fall also ungefähr $|V|!$ Pfade und $e * |V|!$ Cluster gebildet.

Um diesen Aufwand zu verringern, wurde ein Schwellwert für die Pfadwahrscheinlichkeit $\texttt{min}_{P(\texttt{Pfad})} \geq 0$ eingeführt. Ist die Pfadwahrscheinlichkeit zum näch-

sten zu besuchenden Knoten kleiner als dieser Wert, so bricht die Rekursion ab. Sei n_{ges} die Anzahl aller Pfade vom Wurzelknoten w zum zentralen Knoten z. Dann ist die Summe der Pfadwahrscheinlichkeiten all dieser Pfade gleich eins:

$$\sum_{i=1}^{n_{ges}} P(\texttt{Pfad}_i(z)) = 1 \tag{3.13}$$

Dies lässt sich einfach zeigen: Beim Informationsgraphen handelt es sich um die Modellierung des Zugriffsverhaltens von Benutzern. Der zentrale Knoten „sammelt" alle endenden Zugriffe, sei es das Ende einer Sitzung oder das wiederholte Besuchen eines Knotens. Er ist also der einzige Knoten ohne ausgehende Kanten und kann zudem von jedem Knoten aus erreicht werden. Da es sich bei den Kantenwahrscheinlichkeiten um relative Häufigkeiten handelt, ergibt sich die Summe aller von einem Knoten ausgehenden Kanten zu eins. Schließlich enthält ein Pfad keine Zyklen, welche die Pfadwahrscheinlichkeit verfälschen könnten.

Sei n die Anzahl aller gebildeten Pfade mit einer Pfadwahrscheinlichkeit größer als der Schwellwert und $\bar{p}(\texttt{Pfad}(v))$ ihre durchschnittliche Pfadwahrscheinlichkeit. Dann folgt aus Gleichung 3.13

$$n \cdot \bar{p}(\texttt{Pfad}(v)) \leq 1$$

Somit gilt für die Anzahl dieser Pfade:

$$n \leq \frac{1}{\bar{p}(\texttt{Pfad}(v))} \tag{3.14}$$

Da die Menge der möglichen Pfade mit Hilfe des Schwellwertes begrenzt ist, hat jeder gebildete Pfad von w bis zum Knoten v, an dem die Rekursion endet, die Wahrscheinlichkeit $P(\texttt{Pfad}(v)) \geq \texttt{min}_{P(\texttt{Pfad})}$. Setzt man dies in Gleichung 3.14 ein, kann die Anzahl der Pfade mit einer Pfadwahrscheinlichkeit größer als dem

Schwellwert gemäß Gleichung 3.15 berechnet werden als

$$n \leq \frac{1}{\texttt{min}_{P(\texttt{Pfad})}} \tag{3.15}$$

Somit berechnet sich die Zahl der rekursiven Aufrufe der Methode *clusterNachfolger* eines Knotens (siehe Algorithmus 5) zu maximal $\frac{1}{\texttt{min}_{P(\texttt{Pfad})}}$. Jeder dieser Pfade hat maximal die Länge $|V|$, da keine Zyklen enthalten sind. Gleichzeitig wird jede Kante in den Pfaden einmal untersucht, womit die zeitliche Komplexität der Clusterbildung in $\mathcal{O}(\frac{1}{\texttt{min}_{P(\texttt{Pfad})}} \cdot (|V| + |E|))$ liegt.

Sortierverfahren

Zur Optimierung des Aufwands für das iterative Sortieren müssen sowohl das Auffinden der neu zu bewertenden Cluster mitsamt den dazu gehörigen Löschoperationen, als auch die Sortierung der Cluster effizient implementiert werden, wie nachfolgend erläutert wird.

Auffinden neu zu bewertender Cluster inklusive Löschoperationen: Sei $\mathcal{C}$ die Menge der Cluster. Wenn ein Knoten in die Vorabübertragungsliste aufgenommen wird, muss er aus jedem Cluster, der ihn enthält, entfernt werden. Um diese effizient zu finden, führt jeder Knoten eine Liste aller Cluster, in denen er enthalten ist. Somit können beim Einfügen eines Knotens in die Vorabübertragungsliste die neu zu bewertenden Cluster mit jeweils konstantem Aufwand bestimmt werden. Für die Löschoperationen tritt der schlechteste Fall dann ein, wenn es sich um einen stark zusammenhängenden Graphen handelt, bei dem jeder Knoten auch gleichzeitig ein Sitzungsstartknoten ist. Dann ist ein Knoten in jedem Cluster vertreten, womit $|\mathcal{C}|$ Löschoperationen notwendig werden. Das Auffinden aller beteiligten Cluster beim Einfügen eines Knotens in die Vorabübertragungsliste und die zugehörigen Löschoperationen liegen demnach in $\mathcal{O}(|\mathcal{C}|)$.

Sortieren der Cluster: In die Vorabübertragungsliste wird in jedem Schritt jeweils der am höchsten bewertete Cluster eingefügt, die Reihenfolge der restlichen Cluster spielt keine Rolle. In diesem Fall bietet sich als Datenstruktur die in [26] beschriebene, auf einer Heapstruktur aufbauende Prioritätswarteschlange an, die unter anderem im Bereich des Job-Scheduling in Betriebssystemen oder in Simulationssystemen eingesetzt wird. Die zum Sortieren notwendigen Operationen sind *löschen, upHeap* und *downHeap*. Letztere lassen einen Cluster im Heap auf- bzw. absteigen, um die Heap-Eigenschaft nach dem Entfernen des am höchsten bewerteten Clusters wieder herzustellen. Gemäß [26] hat der erstmalige Aufbau des Heaps eine Zeitkomplexität von $\mathcal{O}(|\mathcal{C}|)$.

Der Prozess des iterativen Sortierens verläuft wie folgt: Das Entfernen des Clusters an der Spitze des Heaps bewirkt, dass der letzte Cluster an die Spitze gesetzt und mittels *downHeap* neu einsortiert wird. Diese Operation hat einen Aufwand von $\mathcal{O}(\log|\mathcal{C}|)$. Darauf hin steigen diejenigen Cluster nach oben, deren Bewertung sich geändert hat. Auch diese Operation liegt in $\mathcal{O}(\log|\mathcal{C}|)$. Sei $\mathcal{C}_{\texttt{liste}}$ die Menge der Cluster, die in die Vorabübertragungsliste aufgenommen werden können und $\mathcal{C}_{\texttt{neu}}$ die Menge der Cluster, deren Bewertung sich nach dem Einfügen eines Knotens in die Vorabübertragungsliste ändert. Dann liegt die gesamte Zeitkomplexität in

$$\mathcal{O}(|\mathcal{C}_{\texttt{liste}}| \cdot log\ |\mathcal{C}| \cdot |\mathcal{C}_{\texttt{neu}}| \cdot \log\ |\mathcal{C}|) = \mathcal{O}(|\mathcal{C}_{\texttt{liste}}| \cdot |\mathcal{C}_{\texttt{neu}}| \cdot \log^2\ |\mathcal{C}|)$$

Im schlechtesten Fall können alle Cluster in die Vorabübertragungsliste eingefügt werden und es müssen beim Einfügen eines Knotens immer alle Cluster neu bewertet werden, was zu einem mehr als quadratischen Aufwand führen würde. In der in Kapitel 5.4.8 diskutierten Leistungsbewertung ist dieser schlechteste Fall jedoch nicht aufgetreten.

3.11 Mögliche Erweiterung: Einbeziehung von Wissen über zukünftige Benutzerbewegungen

In diesem Abschnitt wird eine mögliche Erweiterung des Vorabübertragungsverfahrens vorgeschlagen. Diese hat zum Ziel, die Trefferrate nochmals zu verbessern, indem zusätzliches Wissen über zukünftige Aufenthalte von Benutzern berücksichtigt wird.

Das generische Vorabübertragungsverfahren zieht Wissen über zukünftige Aufenthalte von Benutzern nicht in Betracht. Kubach konnte jedoch in der erweiterten Version des in seiner Dissertation [54] vorgestellten Vorabübertragungsverfahrens zeigen, dass die Trefferraten deutlich ansteigen, wenn eine Infostation den Weg eines Benutzers durch ihr Dienstgebiet kennt. In dieser erweiterten Version wird das Dienstgebiet einer Infostation in sich nicht überlappende Zonen aufgeteilt. Die Infostation verwaltet für jede dieser Zonen eine Liste mit den Zugriffswahrscheinlichkeiten der dort angefragten Informationen. Für die Selektion der vorab zu ladenden Informationen werden nun nicht mehr alle im Dienstgebiet angeforderten Informationen herangezogen, sondern nur noch diejenigen, die in den Zonen angefragt werden, durch die sich der Benutzer mit einer gewissen Wahrscheinlichkeit bewegen wird.

Zur Modellierung der Benutzerbewegungen innerhalb der Zonen des Dienstgebiets werden so genannte *interne* und *externe Besuchswahrscheinlichkeitskarten* (BWK) verwaltet. Diese Karten enthalten für jede Zone die Wahrscheinlichkeit, mit der ein Benutzer diese besucht. Interne BWKs werden von der Infostation durch die Beobachtung der Bewegungsmuster von Benutzern geführt, externe BWKs werden von den einzelnen Benutzern zur Verfügung gestellt. So kennt beispielsweise eine Navigationsanwendung den genauen Weg eines Benutzers und teilt diesen der Cache-Verwaltung mit. Auf Basis dieser Informationen wird dann jeder auf diesem Weg liegenden Zone eine Besuchswahrscheinlichkeit von 1 zugeordnet, alle anderen Zonen erhalten eine Besuchswahrscheinlichkeit von 0. Aus

den internen und externen BWKs wird schließlich eine endgültige BWK erzeugt. Die genaue Beschreibung dieses Verfahrens ist der Dissertation von Kubach [54] zu entnehmen, in dieser Arbeit wird davon ausgegangen, dass eine solche endgültige BWK existiert.

Eine hierfür notwendige Erweiterung des eigenen Vorabübertragungsverfahrens wird nachfolgend vorgeschlagen. Die für eine Evaluierung notwendige Integration eines Mobilitätsmodells in die Simulationsumgebung wird in Kapitel 4.4 diskutiert.

Aktualisierung der Graphen: Analog zur Einbeziehung von Nutzungsprofilen wird jeder Zone ein Informationsgraph zugeordnet, der das Zugriffsverhalten aller Benutzer in der entsprechenden Zone modelliert. Da Zonen sich nicht überlappen und eine Sitzung aus logisch zusammenhängenden Anforderungen von Webseiten besteht, wird eine Sitzung genau einem Zonengraphen zugeordnet. Diese Zuordnung erfolgt mit Hilfe des Orts, an dem der Sitzungsstarteintrag (siehe Definition 6) erzeugt wurde. Die Aktualisierung eines Zonengraphen erfolgt wie in Abschnitt 3.7 beschrieben. Eine spezielle Anpassung des Verfahrens wie bei der Einbeziehung von Nutzungsprofilen ist nicht notwendig, denn die Zuordnung einer Anforderung zu einem Zonengraphen ist zu hundert Prozent möglich: Sämtliche Knoten- und Kantenzähler werden um den Wert eins inkrementiert.

Selektion mit Clusterbildung Die Traversierung eines Zonengraphen inklusive Clusterbildung unterscheidet sich nicht von der in den vorigen Abschnitten beschriebenen Traversierung des Informationsgraphen. Sie erfolgt nach dem in Abschnitt 3.8.2 beschriebenen Algorithmus 5 zur Clusterbildung.

In die Bewertung und Sortierung der Cluster fließen nun zusätzlich die Besuchswahrscheinlichkeiten der einzelnen Zonen mit ein. Entsprechend der Integration von Nutzungsprofilen in die Entscheidung, welche Informationen vorab zu laden sind, müssen auch in diesem Fall Gesamtcluster gemäß Definition 24 gebildet

werden, da eine Knotenliste in mehreren Teilclustern enthalten sein kann, die wiederum durch die Traversierung unterschiedlicher Zonengraphen entstanden sind. Die Berechnung der Pfad- und Inhaltswahrscheinlichkeit, sowie der Relevanz pro Byte eines Gesamtclusters erfolgt wie nachfolgend beschrieben.

Seien $\mathcal{Z}$ die Menge der Zonen, $z_i \in \mathcal{Z}$ eine Zone und $P(z_i)$ die Wahrscheinlichkeit, dass ein Benutzer Zone z_i besucht. Sei weiterhin $C_{\texttt{ges}}$ ein Gesamtcluster, erzeugt aus den Teilclustern C_1 bis C_n, mit $n = |\mathcal{Z}|$, die auch leer sein können, falls z_i nicht besucht wird. Dann berechnet sich die **Pfadwahrscheinlichkeit des Gesamtclusters** $C_{\texttt{ges}}$ wie folgt:

$C_i.P(\texttt{Pfad}) \cdot P(z_i)$ ist die Wahrscheinlichkeit, dass die in C_{ges} enthaltenen Informationen in Zone z_i angefordert werden. Aus dem Produkt der negierten Wahrscheinlichkeiten eines Zugriffs auf die in C_{ges} enthaltenen Informationen für alle Zonen im Dienstgebiet wird dann die Wahrscheinlichkeit, dass C_{ges} in keiner Zone angefordert wird, berechnet als

$$\overline{C_{ges}.P(\texttt{Pfad})} = \prod_{i=1}^{|\mathcal{Z}|}(1 - C_i.P(\texttt{Pfad}) \cdot P(z_i))$$

Durch erneute Negierung wird schließlich die Wahrscheinlichkeit, dass C_{ges} in mindestens einer Zone angefordert wird , wie folgt berechnet:

$$C_{\texttt{ges}}.P(\texttt{Pfad}) = 1 - \prod_{i=1}^{|\mathcal{Z}|}(1 - C_i.P(\texttt{Pfad}) \cdot P(z_i)) \tag{3.16}$$

Die **Inhaltswahrscheinlichkeit des Gesamtclusters** beschreibt die Wahrscheinlichkeit, dass die für die Erzeugung eines Clusters maßgebliche Inhaltsseite in mindestens einer Zone als Inhaltsseite betrachtet wird. Sie wird analog zur Bestimmung der Pfadwahrscheinlichkeit des Gesamtclusters berechnet als:

$$C_{\texttt{ges}}.P(\texttt{Inhalt}) = 1 - \prod_{i=1}^{|\mathcal{Z}|}(1 - C_i.P(\texttt{Inhalt}) \cdot P(z_i)) \tag{3.17}$$

Die **Relevanz pro Byte eines Gesamtclusters** $C_{\texttt{ges}}$ wird schließlich mittels Gleichung 3.12 berechnet als

$$R(C_{\texttt{ges}}) = \frac{(C_{\texttt{ges}}.P(\texttt{Pfad}))^{\alpha} \cdot (C_{\texttt{ges}}.P(\texttt{Inhalt}))^{\beta}}{C_{\texttt{ges}}.\texttt{größe}}$$

Schließlich wird die Vorabübertragungsliste mit Hilfe des in Abschnitt 3.8.2 beschriebenen Algorithmus 6 zum **iterativen Sortieren** der Cluster erstellt.

3.12 Verwandte Arbeiten

Nachfolgend werden verwandte Arbeiten im Bereich der Vorabübertragung von Informationen sowie Verfahren zur Clusterbildung diskutiert. Vorabübertragungsverfahren sind entsprechend der Klassifikation in Abbildung 2.1 aus Kapitel 2 Prefetching-Verfahren, sowie Hoarding-Verfahren für Dateisysteme und Informationssysteme. In allen Verfahren werden Informationen vorausschauend übertragen, noch bevor sie angefordert werden.

3.12.1 Prefetching

Das Konzept des Prefetchings wurde bereits Mitte der 60er Jahre zur Beschleunigung der Prozessor-Pipeline entwickelt. Mittels Sprungvorhersagen wird hier versucht, die nächsten Befehle und die zugehörigen Operanden in die Pipeline zu laden.

Prefetching-Verfahren nutzen zur Vorhersage die zeitliche Lokalität zwischen den Anfragen aus. Basierend auf der Auswertung der aktuellen Anfrage und der Zugriffshistorie wird berechnet, welche Dateien bzw. Webseiten voraussichtlich unmittelbar danach angefordert werden.

Griffioen und Appleton stellen in [40] ein Prefetching-Verfahren vor, das auf

Basis der Zugriffshistorie einen Wahrscheinlichkeitsgraphen konstruiert, dessen Knoten die Dateien repräsentieren. Die Kanten stellen deren Beziehungen untereinander dar. Dabei stehen zwei Dateien nur dann miteinander in Verbindung, wenn zwischen dem Öffnen beider Dateien nicht mehr als eine bestimmte Anzahl anderer Dateien geöffnet wurde. In diesem Fall wurden sie „kurz" nacheinander referenziert. Als Kantengewicht fungiert ein Zähler für die Anzahl, wie oft die beiden entsprechenden Dateien kurz hintereinander aufgerufen wurden. Seien A und B zwei Knoten, die durch eine Kante (A,B) verbunden sind. Dann wird die Wahrscheinlichkeit, dass Datei B direkt nach Datei A aufgerufen wird, als das Verhältnis des Kantengewichts von (A,B) und der Summe der Kantengewichte aller Kanten, die von A ausgehen, berechnet.

Im Bereich des Webs stellen Tuah et al. in [101] ein Prefetching-Verfahren mit integrierter Cache-Verwaltung vor. Aus allen Anfragen wird ein Zugriffsgraph konstruiert, auf dessen Basis dann die Auswahl der vorab zu ladenden Webseiten getroffen wird. Ein Knoten stellt eine komplette Webseite inklusive aller eingebetteten Dokumente wie beispielsweise Multimediadaten dar. Jeder Knoten besitzt als Attribut einen Zähler für die Anzahl der Seitenaufrufe. Eine Kante besagt, dass die entsprechenden Webseiten direkt nacheinander aufgerufen wurden. Das Kantengewicht wird durch einen entsprechenden Zähler repräsentiert. Seien A und B wiederum zwei Knoten, die durch eine Kante (A,B) verbunden sind. Dann wird die Wahrscheinlichkeit, dass B nach A aufgerufen wird, als das Verhältnis des Kantengewichts zum Wert des Zählers von A berechnet. Als Metrik für Prefetching und Cache-Verwaltung wird statt der Trefferrate die Verbesserung der Antwortzeit (engl. *access improvement*) verwendet, die aus den berechneten Zugriffswahrscheinlichkeiten und der erwarteten Zugriffszeit berechnet wird.

Basierend auf dem Verfahren von Griffioen und Appleton konstruieren Padmanabhan und Mogul in [80] einen Abhängigkeitsgraphen aus dem Zugriffsverhalten aller Benutzer. Jeder Knoten verwaltet einen Zähler für die Anzahl der Seitenaufrufe. Die Wahrscheinlichkeit, dass bei einer Kante (A,B) B nach A an-

gefordert wird, berechnet sich aus dem Verhältnis des Kantengewichts von (A,B) und dem Wert des Zählers von A.

Eine völlig andere Technologie zur Vorhersage, welche Webseiten als nächstes angefordert werden, stellen Nanapoulos et al. in [77] vor. Die Autoren beschreiben Prefetching-Algorithmen mittels einer Markov-Methode n-ter Ordnung. Sei $S = \langle s_1, \ldots, s_n \rangle$ eine Sequenz von Webseiten-Anforderungen. Dann werden mit Hilfe dieser Methode bedingte Wahrscheinlichkeiten zwischen den Anforderungen von Webseiten $P(s_{n+1}, \ldots, s_{n+m} | s_1, \ldots, s_n)$ und Assoziationsregeln der Form $s_1, \ldots, s_n \Rightarrow s_{n+1}, \ldots, s_{n+m}$ berechnet, mit $m, n \geq 1$. Der linke Teil der Regel wird „Kopf" genannt, der rechte „Rest". Die Regel wird dann ausgeführt, wenn der Kopf eine vom Benutzer getätigte Aufrufsequenz enthält. Die im Rest dieser Regel enthaltenen Seiten werden dann auf das Endgerät übertragen.

Zhang et al. präsentieren in [107] ein im Mozilla-Browser eingesetztes Prefetching-Verfahren. Die vorausschauend zu ladenden Seiten werden durch Auswertung sowohl der Zugriffshistorie als auch des Inhalts der zuletzt besuchten Seiten berechnet.

Diskussion: Die in Prefetching-Verfahren eingesetzten Techniken kommen für den Einsatz in Vorabübertragungsverfahren für ortsbasierte Anwendungen nur bedingt in Frage, da die Auswahl vorab zu ladender Informationen für einen Benutzer abhängig von der aktuell angeforderten Information getroffen wird. Wie in der Leistungsbewertung des vorgestellten Verfahrens in Kapitel 5 gezeigt wird, werden bessere Resultate erzielt, wenn die Entscheidung, welche Informationen geladen werden sollen, pro Gebiet getroffen wird. Schließlich unterscheidet sich der in dieser Arbeit verwendete Informationsgraph von den in den Prefetching-Verfahren verwendeten Zugriffs- oder Abhängigkeitsgraphen dadurch, dass zum einen durch die speziellen Knoten wie den zentralen Knoten oder den Wurzelknoten das Navigationsverhalten abgebildet werden kann und zum andern im Informationsgraphen zusätzliche Merkmale wie beispielsweise die Inhaltswahrscheinlichkeit eines Knotens verwaltet werden.

3.12.2 Hoarding in Dateisystemen

Das Coda-Dateisystem (siehe [51,89,90]) wurde von Satyanarayanan et al. Ende der 80er Jahre entwickelt und war eines der ersten Systeme, das ein Hoarding-Verfahren zur Unterstützung des entkoppelten Betriebs einsetzte. Der Algorithmus verwendet *implizite und explizite Informationsquellen* zur Berechnung von Prioritäten, auf deren Basis dann die Entscheidung getroffen wird, welche Dateien vorab zu laden sind. Implizite Informationen werden aus der Zugriffshistorie ähnlich wie bei der LRU-Ersetzungsstrategie in Caching-Verfahren gewonnen. Als explizite Informationsquelle wird pro Benutzer eine so genannte *Hoard-Datenbank* eingesetzt, die vom Benutzer entweder interaktiv oder mittels einer Skriptsprache (so genannte *Hoard-Profile*), geändert werden kann. Coda verlangt Benutzerinteraktion zur Selektion der Dateien, anders als das vorgestellte automatische Vorabübertragungsverfahren.

Tait et al. entwickelten mit SPY UTILITY ein *intelligentes Hoarding-System für Dateien in mobilen Umgebungen*, das vom Benutzer nur noch die Eingabe so genannter Buchstützen (engl. *bookends*) verlangt, die als zeitliche Begrenzung für Benutzeraktivitäten verstanden werden können, während derer das Zugriffsverhalten beobachtet wird. Diese Benutzeraktivitäten werden vom System aufgezeichnet und analysiert, wobei so genannte Arbeitsmengen (engl. *working sets*) gebildet werden. Das System identifiziert *Projekte* mittels eines top-down Ansatzes durch Generierung eines Baums, dessen Knoten Dateizugriffe repräsentieren (Programme und Daten). Es werden jedoch keine Beziehungen zwischen den durch die Wurzel dargestellten top-level Programmen einzelner Bäume hergestellt. Die Ergebnisse werden angezeigt und können interaktiv vom Benutzer nach seinen individuellen Bedürfnissen angepasst werden. Bei diesem Verfahren handelt sich um ein teil-automatisches Hoarding-Verfahren.

Im Projekt SEER (siehe [56, 58]) wurde von Kuenning et al. ein vollautomatisches Hoarding-Verfahren für Dateien entwickelt. Die Selektion der zu hortenden Dateien basiert auf der Analyse des Zugriffsverhaltens. Hierfür wird als Maß die

semantische Distanz eingeführt, wobei zwei Dateien miteinander in Beziehung stehen, wenn sie zur gleichen Zeit geöffnet waren. Aufbauend auf dieser semantischen Distanz werden mittels eines hierarchischen agglomerativen Verfahrens zur Clusterbildung einzelne Dateien zu Projekten gruppiert, die dann entweder vollständig oder gar nicht übertragen werden. Hierfür muss zunächst eine Distanzmatrix erstellt werden, welche die semantische Distanz für alle Paare von Dateien enthält. Um den hierfür notwendigen quadratischen Aufwand zu vermeiden, wird eine Heuristik eingesetzt, die jeweils nur die n nächsten Nachbarn betrachtet. Bei agglomerativen Verfahren zur Clusterbildung wird zunächst jede Datei als ein Cluster betrachtet. In jeder Runde werden anschließend ähnliche Cluster vereinigt, bis keine nennenswerten Änderungen mehr durchgeführt werden. Dieser Algorithmus wird in zwei Phasen aufgeteilt. (1) Für alle in Frage kommenden Paare (a, b) mit mindestens k_n gemeinsamen nächsten Nachbarn gilt: Vereinige die Cluster, die entweder a oder b enthalten. (2) Für alle Paare (a, b), die mindestens k_{min}, aber weniger als k_n gemeinsame nächste Nachbarn haben, gilt: Füge a allen Clustern hinzu, die b enthalten, und umgekehrt. Die Entscheidung, welche Projekte übertragen werden, wird mittels der LRU-Ersetzungsstrategie und anschließendem Filtern auf Grundlage von dateispezifischen Parametern getroffen. In [56] vergleichen Kuenning et al. SEER mit reinem LRU-basiertem Hoarding ohne Clusterbildung und stellen fest, dass in den meisten Fällen die LRU-basierte Lösung bessere Ergebnisse mit wesentlich weniger Aufwand als die Clusterbildung liefert.

Zhang et al. stellen in [106] eine Architektur für einen mobilen Dateidienst in ubiquitären Umgebungen vor. Pro Benutzer wird jeder Datei eine Priorität zugewiesen, die sich aus der Zugriffshäufigkeit, -neuheit und dem Zeitpunkt, an dem die Datei geöffnet wurde, berechnet. Diese Priorität wird gemeinsam mit zusätzlichen dateispezifischen Eigenschaften zur Berechnung der zu hortenden Dateien verwendet.

Diskussion: Jedes dieser Verfahren ist auf die speziellen Eigenschaften von Dateisystemen zugeschnitten, die nicht ohne Weiteres auf Informationssysteme übertragen werden können. Infolgedessen können mit ihnen weder das Zugriffsverhalten von Benutzern im Web, noch die speziellen Eigenschaften von Webseiten wie beispielsweise Inhalts- und Transitseiten modelliert werden, was jedoch ein wichtiger Bestandteil dieser Arbeit ist. Des Weiteren spielt bei diesen Verfahren die temporale Lokalität zwischen den Dateizugriffen eine wichtige Rolle. Das vorgestellte Verfahren ist jedoch für ortsbasierte Systeme konzipiert unter der Annahme, dass Benutzer in einem bestimmten Gebiet ein ähnliches Zugriffsverhalten aufweisen. Hierfür wurden sowohl das kollektive Zugriffsverhalten als auch die speziellen Eigenschaften von Webseiten modelliert. Wie in der Leistungsbewertung in Kapitel 5 gezeigt wird, eignet sich für ortsbezogene Anfragen die räumliche Lokalität zwischen Anfragen eher als die zeitliche.

3.12.3 Hoarding in Informationssystemen

Das von Ye et al. in [104] vorgestellte Projekt Map-on-the-Move versorgt Benutzer, die auf einem Highway unterwegs sind, mit Kartendaten in unterschiedlichen Auflösungen, deren Detailgrad von der aktuellen Geschwindigkeit abhängt. Als Kommunikationsarchitektur werden Infostationen verwendet, die vorbeifahrenden Anwendern die entsprechenden Kartendaten automatisch übertragen. Dieses Verfahren ist speziell auf Kartendaten zugeschnitten. Die Auswahl der Informationen ist in diesem Fall sehr einfach, da das System die genaue Route des Fahrers kennt und somit feststeht, welche Informationen an welchen Infostationen benötigt werden.

Kooperative Hoarding-Verfahren für mobile Ad-hoc-Netze, wie sie beispielsweise von Lai et al. in [61] vorgestellt werden, verbinden die Vorteile von Hoarding und kooperativen Caching-Verfahren, basierend auf dem Mobilitätsverhalten von Benutzergruppen. Im entkoppelten Betrieb haben mobile Benutzer somit zusätzlich die Möglichkeit, nicht im eigenen lokalen Cache gespeicherte Informationen

im Ad-hoc-Modus von benachbarten Knoten zu erhalten. Lai et al. schlagen hierfür zwei Ansätze vor. Mit dem Algorithmus *Greedy Global Hoard* wählen Klienten die zu hortenden Informationen basierend auf der eigenen Zugriffswahrscheinlichkeit, dem Inhalt des Caches der benachbarten Knoten, sowie der Verbindungswahrscheinlichkeit und Anzahl der Hops zu diesen. Mit *Cooperative Access Probability-based Hoarding* wird mit Hilfe einer globalen Kostenfunktion der beste Speicherort für die unterschiedlichen Informationen bestimmt. Die Berechnungen werden von einem leistungsfähigen Knoten durchgeführt, während die Klienten verbunden sind. Diese Ansätze setzen voraus, dass die mobilen Endgeräte außerhalb des Übertragungsgebiets von Hotspots im Ad-hoc-Modus miteinander kommunizieren können. Der Fokus liegt bei diesen Verfahren jedoch auf der verteilten Verwaltung der Caches und nicht auf der Selektion der Informationen, wie das in dieser Arbeit der Fall ist. Es ist jedoch vorstellbar, eine verteilte Cache-Verwaltung zu integrieren.

In der Dissertation von Kubach [54] wird ein Hoarding-Verfahren für unstrukturierte Informationen in ortsbasierten Anwendungen vorgestellt. Als Kommunikationsinfrastruktur werden wie in dem hier beschriebenen Verfahren Infostationen eingesetzt. Die Entscheidung, welche Informationen zu horten sind, basiert hauptsächlich darauf, wie häufig auf diese von allen Benutzern innerhalb eines bestimmten Gebiets zugegriffen wurde. Wie bereits in Abschnitt 3.11 erwähnt, wird zur Verbesserung der Ergebnisse in der erweiterten Version zusätzliches Wissen über den zukünftigen Aufenthaltsort eines Benutzers ausgenutzt. Schließlich kann die Selektion durch Einbeziehen von Informationskanälen verfeinert werden. Dieses auf der LFU-Ersetzungsstrategie basierende Verfahren bezieht sich auf unstrukturierte Informationen, es werden also keinerlei Beziehungen zwischen den Informationen ausgenutzt. Hierdurch wird angenommen, dass die Zugriffe auf die Informationen unabhängig von einander erfolgen, während in dieser Arbeit das Navigationsverhalten von Benutzern modelliert wird, um daraus Beziehungen zwischen schwach strukturierten Informationen abzuleiten. In der Leistungsbewertung in Kapitel 5 wird gezeigt, dass sich die Berücksichtigung von solchen Beziehungen positiv auf die Trefferrate auswirkt. In Abschnitt 3.11 wurde eine

Erweiterung des vorgestellten Verfahrens um die Einbeziehung von Wissen über zukünftige Aufenthaltsorte in den Selektionsprozess vorgeschlagen.

3.12.4 Clusterbildung im Bereich der Informationsgewinnung im Web

In dieser Arbeit wird ein Verfahren zur Clusterbildung eingesetzt. Viele Ansätze im Bereich der Informationsgewinnung (engl. *information retrieval*) bilden Cluster, um beispielsweise Suchanfragen zu verbessern oder deren Ergebnisse übersichtlicher präsentieren zu können.

Bereits 1989 setzten Crouch et al. in [28] zur Gruppierung semantisch ähnlicher Dokumente ein agglomeratives, hierarchisches Verfahren zur Clusterbildung ein. Der Fokus liegt auf einer schnellen Bearbeitungszeit von Benutzeranfragen und nicht in einer effizienten Erstellung der Cluster, weshalb die quadratische zeitliche Komplexität des Verfahrens wohl keine Rolle spielt. Die Dokumente werden auf Grund ihrer Verknüpfung mittels Hyperlinks und des Inhalts in Gruppen zusammengefasst. Hierarchische Verfahren bilden allerdings nur disjunkte Cluster, weshalb sie in dieser Arbeit nicht angewandt werden können.

López et al. stellen in [68] einen Ansatz vor, der mit Hilfe von Clustern die Resultate einer Suchanfrage übersichtlicher präsentieren soll. Das Hauptaugenmerk liegt auf der Darstellung der Ergebnisse und nicht auf der Clusterbildung, die mit dem *k-means-* und dem *bisecting k-means-Algorithmus* erzeugt werden. Diese partitionierenden Verfahren erzeugen allerdings keine überlappenden Cluster und haben zudem quadratischen Aufwand. Das Gruppieren der Dokumente in Cluster erfolgt nach inhaltlichen Kriterien.

Leuski vergleicht in [63] zur Verbesserung der Ergebnisse von Suchanfragen sechs hierarchische Verfahren zur Clusterbildung für die Gruppierung von Dokumenten. Die Berechnung der Ähnlichkeit von Dokumenten geschieht auf Basis eines Vektorraummodells. Die Dimension des Vektorraums wird durch die Anzahl der

Wörter des verwendeten Vokabulars festgelegt, jedes Dokument wird durch einen Vektor repräsentiert. Die Ähnlichkeit zwischen den Dokumenten hängt unter anderem davon ab, wie oft ein Wort des Vokabulars im Dokument vorkommt und wie viele Wörter das Dokument enthält. Der quadratische Aufwand der hierarchischen Verfahren ist durch die relativ kleine Zahl an Dokumenten gerechtfertigt.

Beeferman und Berger diskutieren in [9] einen Ansatz zur Clusterbildung, bei dem ähnliche Anfragen und ähnliche Bezeichner von Webseiten (URLs) gruppiert werden. Durch Analyse von Protokolldateien einer Suchmaschine wird ein bipartiter Graph erstellt, der auf der einen Seite die Anfragen enthält und auf der anderen die URLs. Die Elemente der Mengen mit ähnlichen Anfragen repräsentieren unterschiedliche Ausdrucksformen für ein bestimmtes Informationsbedürfnis. Umgekehrt stellen die in einem Cluster gruppierten Webseiten Ergebnisse für ähnlichen Anfragen dar. Die Ähnlichkeit zwischen Anfragen bzw. Webseiten wird iterativ berechnet als der Quotient der Größe der Schnittmenge der unmittelbaren Nachbarknoten und der Größe der entsprechenden Vereinigungsmenge. Das Verfahren hat linearen Aufwand in der Anzahl von Knoten, bildet jedoch nur disjunkte Cluster.

3.13 Zusammenfassung

In diesem Kapitel wurde ein generisches Verfahren zur Vorabübertragung von beliebigen schwach strukturierten Informationen für ortsbasierte Dienste beschrieben, das Nutzungsprofile unterstützt. Insbesondere wurden die generischen Konzepte vorgestellt und die für den Zugriff auf Webseiten als eine konkrete Anwendung notwendigen Spezialisierungen beschrieben.

Als zentrale Datenstruktur für die Modellierung des Wissens einer Infostation über das Zugriffsverhalten aller Benutzer, die sich in ihrem Dienstgebiet aufhalten, wird ein Informationsgraph verwendet, der das kollektive Zugriffsverhalten aller Benutzer widerspiegelt. Das Verfahren passt sich mit Hilfe einer Alterungs-

funktion automatisch an sich änderndes Informationsbedürfnis der Benutzer an, wodurch Benutzer stets diejenigen Informationen erhalten, die zum aktuellen Zeitpunkt im Dienstgebiet der Infostation populär sind.

Für die Selektion vorab zu übertragender Webseiten werden je nach Anwendungsfall vier Auswahlverfahren zur Verfügung gestellt, davon eines mit Clusterbildung für den Fall, dass Webseiten semantisch so stark zusammenhängen, dass sie für einen Benutzer nur als Gruppe von Interesse sind. Für die Entscheidung, welche Webseiten vorab geladen werden sollen, wird deren semantische Nähe berechnet, die in den Auswahlprozess integriert wird. Da der Cache eines mobilen Endgeräts nur begrenzt Speicherplatz zur Verfügung hat, müssen die selektierten Webseiten nach einem Ordnungskriterium sortiert werden. Hierfür wird bei zwei der Verfahren die Relevanz pro Byte verwendet, die neben der Zugriffswahrscheinlichkeit auch die Größe der Seiten berücksichtigt. Der zeitliche Aufwand für die Clusterbildung und das zugehörige Sortieren der Cluster zur Erstellung der Vorabübertragungsliste wird durch die Einführung eines Schwellwertes begrenzt, der die Zahl der zu bildenden Cluster einschränkt. Da die Clusterbildung überdies nur periodisch zu bestimmten Zeiten durchgeführt wird und es sich bei Infostationen um leistungsfähige Rechner handelt, lohnt sich der im Gegensatz zu anderen Vorabübertragungsverfahren höhere Aufwand durch das Erzielen höherer Trefferraten, wie in der Leistungsbewertung in Kapitel 5 gezeigt wird.

Für den Fall, dass es Benutzergruppen mit unterschiedlichem Informationsbedürfnis gibt, werden Nutzungsprofile in die Selektion relevanter Informationen integriert.

Schließlich wurde eine mögliche Erweiterung vorgeschlagen, die das Wissen über zukünftige Benutzerbewegungen in den Auswahlprozess integriert.

4 Modellierung des Navigationsverhaltens im World Wide Web

Das vorgestellte generische Vorabübertragungsverfahren wurde für den Zugriff auf Webseiten spezialisiert und evaluiert. Um eine systematische Leistungsbewertung gewährleisten zu können, wurden die zugrunde liegenden Informationsräume und die zugehörigen Zugriffe synthetisch generiert. Hierzu ist ein Modell des Navigationsverhaltens von Benutzern im Web erforderlich, dessen Implementierung in eine Simulationsumgebung integriert werden muss. In diesem Kapitel werden das Web-Navigationsmodell sowie dessen Implementierung und Integration in die Simulationsumgebung beschrieben.

Das Web-Navigationsmodell wurde in Java 5.0 implementiert und liefert als Ausgabe eine Reihe von Protokolldateien für unterschiedlich dimensionierte Informationsräume und Nutzungsprofile, die dem Vorabübertragungsverfahren als Eingabe dienen. In den nachfolgenden Abschnitten wird zunächst ein Überblick über das Web-Navigationsmodell gegeben, gefolgt von einer detaillierten Beschreibung der Teilmodelle, der Implementierung sowie der Integration in eine Simulationsumgebung. Nach dem Vorschlag einer möglichen Erweiterung werden abschließend die diesem Thema verwandten Arbeiten diskutiert.

4.1 Problemstellung

Heutzutage bauen zahlreiche Anwendungen auf dem Web auf, was einen standardisierten Zugriff auf diese Dienste erst ermöglicht. Beispiele hierfür sind ortsbasierte Systeme, elektronischer Handel, elektronisch unterstütztes Lernen oder auch Online-Spiele mit verteiltem Zustand. Ein kritischer Faktor für solche Anwendungen ist die zugrunde liegende Web-Technologie wie beispielsweise die optimale Organisation eines Web-Angebots, die schnelle Erzeugung dynamischer Webseiten, der effiziente Zugriff auf (entfernte) Datenbanken, die Minimierung der Zugriffszeit von Webseiten oder Optimierungsmethoden für den Zugriff auf das Web, wie beispielsweise Caching- oder Vorabübertragungsverfahren.

Zur Evaluierung solcher Technologien werden als Eingabe typischerweise sehr viele Sequenzen von Webseiten-Anforderungen benötigt. Diese können auf unterschiedliche Arten erzeugt werden. Für die *empirische Evaluierung* einer Web-Technologie werden reelle Daten ausgewertet, die entweder durch spezielle Messungen oder aus der Analyse der Protokolldateien geeigneter (Proxy-)Server erzeugt wurden. Da zur Auswertung eine statistisch relevante Anzahl von Protokolleinträgen benötigt wird, kommen eigene Messungen nur in Frage, wenn das System bereits existiert und entsprechend lange genug läuft. Fremde Server-Protokolldateien müssen vor der Auswertung bereinigt werden, da sie oftmals Duplikate oder unbrauchbare Einträge enthalten, häufig aber auch unvollständig sein können.

Ein großer Nachteil der empirischen Evaluierung ist es jedoch, dass der zugrunde liegende Informationsraum nicht variiert werden kann. Optimierungsverfahren in ortsbasierten Diensten, wie das vorgestellte Vorabübertragungsverfahren, benötigen für eine systematische Evaluierung jedoch eine Vielzahl von Protokolldateien, die mit einer Reihe von unterschiedlich dimensionierten Informationsräumen assoziiert sind. Diese Informationsräume unterscheiden sich in der Anzahl der darin enthaltenen Informationsobjekte sowie deren durchschnittlicher Größe. Soll zusätzlich das Verhalten des Verfahrens für unterschiedliche Interessengruppen

evaluiert werden, sollten solche Protokolldateien von Benutzern mit unterschiedlichen Nutzungsprofilen erzeugt worden sein. Leider stehen reale Protokolldateien mit statistischer Relevanz, die all diese Bedingungen erfüllen, nicht öffentlich zur Verfügung.

Aus diesen Gründen wurde in dieser Dissertation eine Reihe von unterschiedlichen Informationsräumen synthetisch erzeugt, die jeweils mit einer Vielzahl von Protokolldateien assoziiert wurden. Diese wiederum wurden für mehrere Nutzungsprofile generiert. Hierfür wurde ein Modell für das Navigationsverhalten von Benutzern im Web (*Web-Navigationsmodell*) erstellt und in [16] veröffentlicht, das aus zwei Teilmodellen besteht, dem *Webgraph-Modell* und dem *Zugriffsmodell*.

Das Webgraph-Modell stellt geeignete Verteilungsfunktionen für die Größen von Webseiten, sowie für die Ein- und Ausgangsgrade der Knoten eines Webgraphen zur Verfügung, die für die typische Struktur des Webs in Form einer Fliege (engl. *bow-tie*) verantwortlich sind. Das Zugriffsmodell bildet das eigentliche Navigationsverhalten der Benutzer im Web ab und beinhaltet die Beliebtheit von Webseiten, die Anzahl der Webseiten-Anforderungen in einer Sitzung, die Zeit, wie lange sich ein Benutzer eine Webseite anschaut sowie die Art der Webseiten-Anforderung.

4.2 Web-Navigationsmodell

Das Navigationsverhalten von Benutzern im Web wird hauptsächlich von zwei Faktoren beeinflusst: Der *Struktur des Webs* und dem *Zugriffsverhalten* von Benutzern. Das Web besteht aus einer riesigen Anzahl von Webseiten, die mittels Hyperlinks miteinander verknüpft sind. Aufbauend auf dieser Konnektivität kann nun ein Benutzer einem Hyperlink folgen oder Webseiten direkt aufrufen, entweder mittels Eintippen einer URL in die Adresszeile des Browsers oder Verwendung von dessen Lesezeichenfunktion (engl. *bookmark*). Das Zugriffsver-

halten bestimmt, wie viele und welche Webseiten ein Benutzer anfordert und in welcher Reihenfolge, und wie lange er sich die Webseiten ansieht. Dieses Zugriffsverhalten gilt nach den Untersuchungen von Halvey et al. in [41] sowohl für Benutzer mit leistungsstarken Geräten wie beispielsweise Laptops, als auch für mobile Benutzer mit verhältnismäßig kleinen Endgeräten wie Mobiltelefone oder PDAs. Abbildung 4.1 illustriert das aus den Teilmodellen Webgraph-

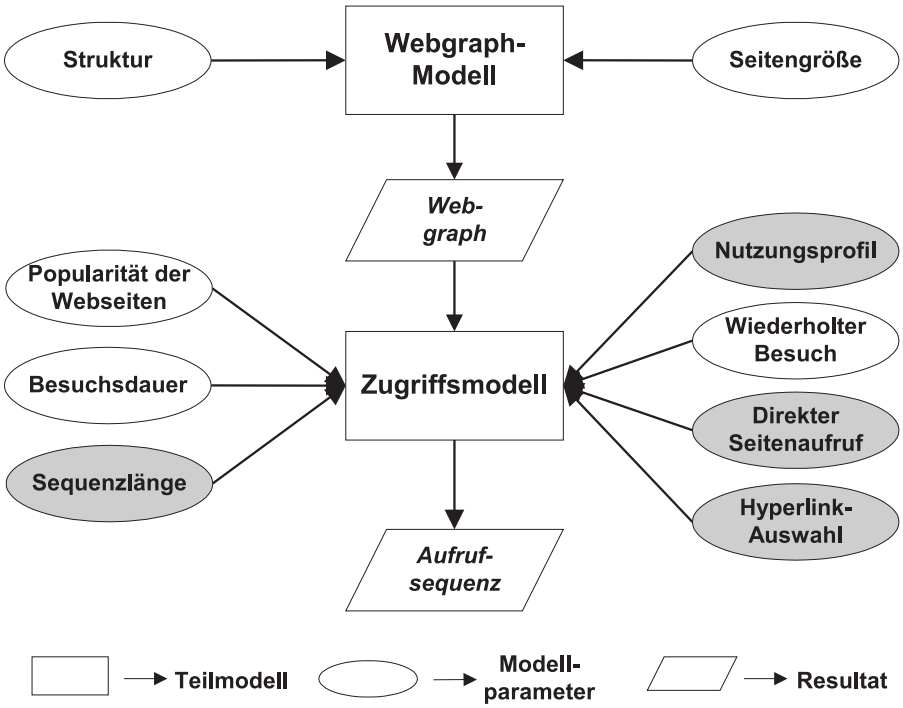


Abbildung 4.1: Web-Navigationsmodell

Modell und Zugriffsmodell bestehende Web-Navigationsmodell. Die Teilmodelle sind durch Rechtecke, die hierfür benötigten Modellparameter durch Ellipsen und das Resultat als Parallelogramm gekennzeichnet. Für die grau unterlegten Modellparameter werden in dieser Arbeit neue Forschungsergebnisse vorgestellt. Die Modellierungsansätze für die nicht markierten Parameter basieren auf in der Literatur vorhandenen Forschungsergebnissen und wurden durch eigene Messungen validiert (siehe Abschnitte 4.2.1 und 4.2.2).

Das Webgraph-Modell mit den beiden Modellparametern *Struktur* und *Seitengröße* liefert Informationen über die Struktur des Webs, die durch die Hyperlinks zwischen den Webseiten bestimmt wird, sowie die Größe von Webseiten. Eine wichtige Erkenntnis der Erforschung des Webs ist dessen Eigenschaft als inhomogenes skalenfreies (engl. *scale-free*) Netzwerk. Im Gegensatz zur ursprünglichen Annahme, dass das Web ein homogenes Netzwerk mit exponentiellem Charakter darstellt, in dem jeder Knoten annähernd gleich viele Verbindungen hat, weist es Selbstähnlichkeit auf (siehe auch Anhang A.3): einige hochgradig verlinkte Webseiten sind mit den restlichen Seiten verbunden und bilden somit den Kern des Webs. Mit Hilfe des Webgraph-Modells kann schließlich ein synthetischer Webgraph generiert werden.

Das Zugriffsmodell liefert Informationen darüber, wie Benutzer im Web navigieren. Es benötigt als Eingabe einen Webgraphen, dessen Traversierung schließlich die Navigation virtueller Benutzer simuliert. Dies erklärt die Zweiteilung des Web-Navigationsmodells, denn für das Zugriffsmodell spielt es keine Rolle, ob es sich um einen synthetischen Graphen handelt, oder ob der Graph ein Abbild (eines Teils) des Webs ist. Die Zugriffsmuster der virtuellen Benutzer werden mittels folgender Modellparameter charakterisiert: Die *Popularität der Webseiten* beeinflusst die Auswahl einer bestimmten Seite aus der gesamten Menge von Webseiten. Die *Besuchsdauer* liefert Informationen darüber, wie lange sich ein Besucher die gewählte Webseite ansieht. Beispielsweise tendieren manche Benutzer dazu, den Inhalt einer Webseite nur zu überfliegen, während ihn andere genauer lesen. Die *Sequenzlänge* bestimmt die Anzahl der Webseiten-Anforderungen innerhalb einer Sitzung. Das *Nutzungsprofil* bestimmt, welcher Bereich des Webgraphen innerhalb eines bestimmten Nutzungsprofils traversiert wird und welcher Profilmix schließlich einem Benutzer zugeordnet wird. Der Modellparameter *Wiederholter Besuch* bildet die mehrmalige Anforderung einer Webseite innerhalb einer Sitzung ab. Dieses häufig beobachtete Verhalten von Benutzern wird beispielsweise durch Klicken auf den Zurück-Schalter (engl. *back button*) des Web-Browsers hervorgerufen. Mittels des Modellparameters *Direkter Seitenaufruf* wird bestimmt, ob ein Benutzer die nächste Webseite direkt auf-

ruft oder einem Hyperlink folgt. Soll einem Hyperlink gefolgt werden, wird mit Hilfe des Modellparameters *Hyperlink-Auswahl* festgelegt, welcher Link auf der aktuellen Webseite angeklickt werden soll.

In den letzten Jahren beschäftigte sich die Forschung intensiv mit dem Web und dem Zugriffsverhalten von Benutzern. Mit Ausnahme der Modellparameter *Hyperlink-Auswahl*, *Direkter Seitenaufruf* und *Nutzungsprofil* finden sich in der Literatur umfassend validierte Modellierungsansätze und Verteilungsfunktionen für die verbleibenden Modellparameter, auf denen das Web-Navigationsmodell aufbaut. Zu dessen Kalibrierung wurde die anonymisierte Protokolldatei vom Juni 2004 des Proxy-Servers des Rechenzentrums der Universität Stuttgart analysiert und ausgewertet. Deren Einträge wurden auf Grundlage der Webseiten-Anforderungen von wissenschaftlichem Personal der Universität und Studierenden erstellt. Nach Bereinigung der Protokolldatei konnten 619 verschiedene Benutzer mit insgesamt ungefähr 1,5 Millionen Anfragen extrahiert werden, die wiederum in 17.609 Sitzungen aufgeteilt wurden. Die Ableitung der Sitzungen aus den Protokolleinträgen erfolgte nach dem in Kapitel 3.6 beschriebenen Algorithmus zur Analyse einer Protokolldatei. Die durchgeführten Auswertungen bestätigten zum einen die Gültigkeit der in der Literatur ermittelten Verteilungen. Zum anderen konnten die bislang fehlenden Verteilungen für die Modellparameter *Hyperlink-Auswahl* und *Direkter Seitenaufruf* zur Verfügung gestellt werden. Als weiterer Beitrag zur Forschung wurde das in [64] von Levene et al. beschriebene Modell zur Bestimmung der *Sequenzlänge* erweitert und eine Modellierung von *Nutzungsprofilen* vorgeschlagen.

Die Implementierung des Web-Navigationsmodells basiert einzig und allein auf der Verwendung der zugrunde liegenden Verteilungen mitsamt ihrer Kenngrößen, die in der Literatur und durch eigene Messungen validiert wurden. Man kann davon ausgehen, dass die hiermit erstellten synthetischen Informationszugriffe real erzeugten Protokolldateien sehr nahe kommen.

4.2.1 Webgraph-Modell

Das Web wird typischerweise als direkter Graph modelliert, dessen Knoten einzelne Webseiten und dessen Kanten die Hyperlinks zwischen den Webseiten darstellen. Zur Modellierung der topologischen Eigenschaften des Webgraphen müssen folgende Modellparameter berücksichtigt werden:

- Die *typische Struktur* des Webs wird durch geeignete Verteilungen der Eingangs- und Ausgangsgrade von Knoten beschrieben, die wiederum die Verknüpfung zwischen den repräsentierten Webseiten charakterisieren. Diese Struktur ist maßgeblich für die selbstähnliche Eigenschaft des Webs verantwortlich.
- Die *Größe der Webseiten* ist vor allem für Anwendungen wie Caching- und Vorabübertragungsverfahren wichtig, in denen die Auswahl der Webseiten von der Größe des Caches abhängt.

Modellierung der Struktur des Webs Kumar et al. präsentieren in [60] die Auswertungen eines 1997 von der Alexa Inc. durchgeführten Web-Crawls. Für den Eingangs- und Ausgangsgrad der Knoten wurden Zipf-ähnliche Verteilungsfunktionen nach dem Potenzgesetz (siehe auch Abschnitt A.3) festgestellt. Basierend auf diesen Beobachtungen führten Broder et al. im Mai 1999 und Oktober 1999 drei Alta Vista Web-Crawls durch und stellten im Jahr 2000 in [14] die Ergebnisse ihrer umfangreichen Analysen der topologischen Eigenschaften der durch die jeweiligen Crawls erzeugten Webgraphen vor. Diese Ergebnisse bestätigen die von Kumar et al. ermittelten endlastigen Verteilungsfunktionen der Eingangs- und Ausgangsgrade der Knoten, die schließlich zu der für den Webgraphen typischen so genannten „Bow-tie-Struktur" führen, die in Abbildung 4.2 dargestellt ist.

Danach besteht der Webgraph aus fünf Komponenten, deren Größen ebenfalls nach dem Potenzgesetz verteilt sind. Der *Kern* ist eine stark zusammenhängende

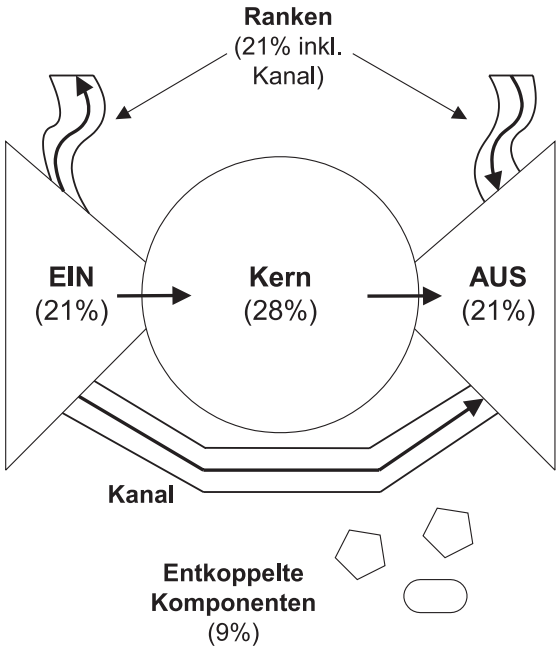


Abbildung 4.2: Struktur des Webgraphen nach Broder et al. [14]

Komponente, in der jede Webseite von jeder anderen Webseite aus erreicht werden kann. Die meisten Webseiten aus der *EIN*-Komponente können alle Seiten des Kerns erreichen, was aber umgekehrt nicht gilt. Im Gegenzug können die Webseiten des Kerns alle Seiten der *AUS*-Komponente erreichen, was wiederum umgekehrt nicht möglich ist. *Ranken* und *Kanäle* bilden zusammen die vierte Komponente. Die Webseiten in den Ranken können nur entweder von Seiten in der EIN-Komponente erreicht werden bzw. nur Seiten der AUS-Komponente erreichen. Kanäle verbinden Webseiten der EIN-Komponente direkt mit der AUS-Komponente. Die Webseiten in den *Entkoppelten Komponenten* haben nur eine Verbindung innerhalb der einzelnen Teilkomponenten.

Während Broder et al. die Struktur des gesamten Webs untersuchten, analysieren Dill et al. in [33] die topologischen Eigenschaften einzelner Teile des Webs, die in einer bestimmten Beziehung zueinander stehen. Sie gruppieren das Web in so genannte thematisch vereinheitlichte Cluster, die nach folgenden Kriteri-

en ausgewählt werden: zufällig selektierte Web-Auftritte, einzelne Intranets und Webseiten mit gleichem Inhalt oder geographischer Lage. Die Resultate zeigen, dass die einzelnen Cluster ebenfalls die Bow-tie-Struktur aufweisen. Die Cluster unterscheiden sich lediglich in den Exponenten der Zipf-ähnlichen Verteilungen der Ein- und Ausgangsgrade der Knoten sowie der Größe der Komponenten.

Bharat et al. untersuchen in [11] das Web auf der Basis von Web-Auftritten statt einzelner Webseiten und erzeugen einen so genannten Host-Graphen, dessen Knoten die einzelnen Hosts repräsentieren. Eine Kante zwischen zwei Knoten sagt aus, dass mindestens eine vom zugeordneten Quell-Host angebotene Webseite mit mindestens einer vom Ziel-Host angebotenen Seite durch einen Hyperlink verbunden ist. Sie führten im Oktober 1999, August 2000, und Juni 2001 drei Experimente durch, wobei sie die Datenmenge auf die Hosts beschränkten, die von einem in der Kern-Komponente des Webs angesiedelten Referenz-Host aus erreicht werden konnten. Die Ergebnisse zeigen, dass die Ein- und Ausgangsgrade der Knoten ebenfalls einer Zipf-ähnlichen Verteilung folgen.

Donato et al. untersuchten im Jahr 2005 in [31] sowohl die Struktur des Webgraphen als auch die Struktur der einzelnen Komponenten. Ihre Ergebnisse bestätigen die Zipf-ähnlichen Verteilungen der Ein- und Ausgangsgrade der Knoten. Diese gelten demnach sowohl für den gesamten Webgraphen als auch innerhalb der einzelnen Komponenten. Eine Untersuchung der inneren Struktur der Komponenten ergab jedoch, dass nicht alle Einzelkomponenten die Bow-tie-Struktur aufweisen. So fehlt beispielsweise bei der EIN- und AUS-Komponente jeweils ein großer Kern. Demzufolge ist nach diesen Untersuchungen zwar das Web selbstähnlich, jedoch nicht der Webgraph selbst. Dies spricht jedoch nicht gegen die Ergebnisse von Dill et al. bezüglich der thematisch vereinheitlichten Cluster, denn die darin enthaltenen Webseiten müssen nicht zwangsläufig nur aus einer einzelnen Komponente des Webgraphen stammen.

Auf Grund dieser umfangreichen Analysen ist die Annahme gerechtfertigt, dass die Bow-tie-Struktur für das Web allgemein gültig ist. Durch die Selbstähnlichkeit des Webs gilt diese Struktur sowohl für das gesamte Web, als auch für

einzelne Teile des Webs wie beispielsweise thematisch vereinheitlichte Cluster oder Web-Hosts. Die Zipf-ähnlichen Verteilungsfunktionen für die Ein- und Ausgangsgrade der Knoten werden folglich in das Webgraph-Modell aufgenommen.

Modellierung der Größe von Webseiten Entsprechend den Untersuchungen von Barford et al. [8], Crovella et al. [29] und Czyczowicz et al. [30] ist die Verteilung der Größen von Webseiten insgesamt ebenfalls endlastig, jedoch aus zwei einzelnen Verteilungen zusammengesetzt. Der vordere Teil bis zu einem bestimmten *Übergangswert* ist lognormal-verteilt, während das Ende einer Pareto-Verteilung folgt. Während Barford et al. einen Übergangswert von ungefähr einhundertdreißig KBytes ermittelten, lag dieser in den beiden anderen Analysen bei ca. zehn KBytes. In [86] stellen Reed und Jorgensen eine Doppel-Pareto-Verteilung mit einem Übergangswert von ca. zehn KBytes fest. Überraschenderweise liefert das Resultat einer Untersuchung von Fetterly et al. eine Gaußsche Verteilung der Größen von Webseiten. Abbildung 4.3 zeigt als Resultat eigener Messungen mittels der in Abschnitt 4.2 angeführten Protokolldatei die komplementäre kumulative Verteilungsfunktion der Größen von Webseiten. Auf der logarithmisch skalierten x-Achse ist die Größe von Webseiten aufgetragen, die logarithmisch skalierte y-Achse zeigt die Wahrscheinlichkeit $P(X > x)$. Dieses Ergebnis bestätigt die Doppel-Pareto-Verteilung mit Exponenten $\alpha_1 = 0{,}3$ und $\alpha_2 = 1{,}1$ und einem Übergangswert von zehn KBytes, die auch von Reed und Jorgensen festgestellt wurde.

Da diese Verteilung auf den neuesten Daten beruht und durch eigene Messungen bestätigt wurde, wird diese in das Webgraph-Modell aufgenommen.

4.2.2 Zugriffsmodell

Die Modellierung des Zugriffsverhaltens von Benutzern des Webs wirft eine Reihe von Fragen auf: Welche Seiten werden wie aufgerufen und in welcher Reihenfolge? Wie lange verweilt ein Benutzer auf einer Seite? Welchem Hyperlink auf der

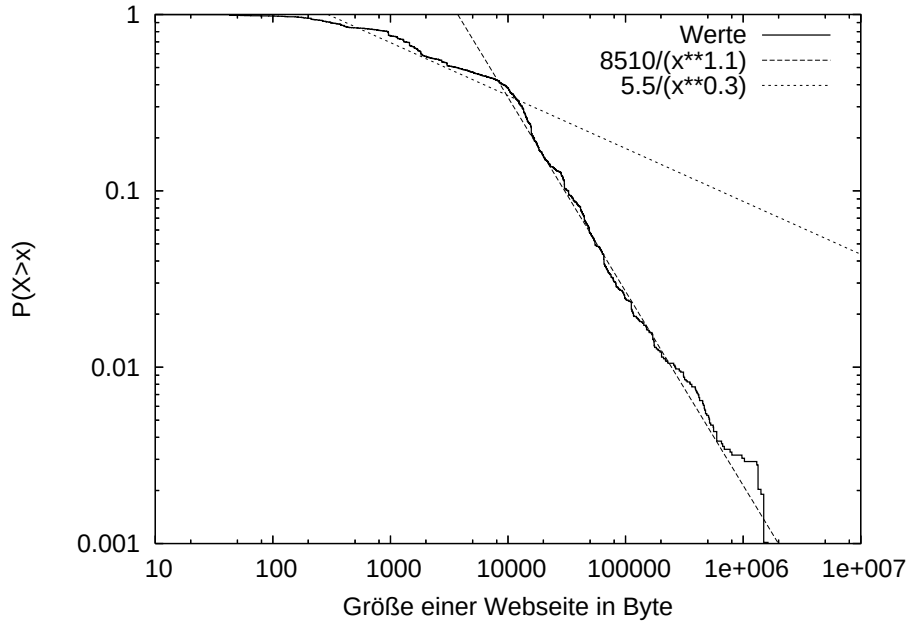


Abbildung 4.3: Verteilung der Größen von Webseiten

Webseite folgt ein Benutzer? Wie viele Webseiten werden insgesamt innerhalb einer Sitzung aufgerufen? Das Zugriffsmodell soll helfen, auf diese Fragen eine Antwort zu finden und benötigt hierfür die folgenden Modellparameter:

Direkter Seitenaufruf: bestimmt, ob ein Benutzer die nächste Webseite direkt aufruft oder einem Hyperlink folgt;

Hyperlink-Auswahl: legt fest, welchem Link auf der aktuellen Webseite gefolgt wird;

Sequenzlänge: bestimmt die Anzahl der Webseiten-Anforderungen innerhalb einer Sitzung;

Wiederholter Besuch: bildet die mehrmalige Anforderung einer Webseite innerhalb einer Sitzung ab;

Popularität der Webseiten: bestimmt, welche Webseite aufgerufen wird;

Besuchsdauer: liefert Informationen darüber, wie lange ein Besucher auf der

gewählten Webseite verweilt.

Profil: ordnet einem Benutzer einen Profilmix zu;

Nachfolgend werden diese Modellparameter detailliert vorgestellt.

Um die ersten beiden Modellparameter bestimmen zu können, wurde eine statische Analyse der rund 48000 Webseiten aus der oben angeführten Protokolldatei durchgeführt. Eine Seite gilt als direkt angefordert, wenn die direkt davor angeforderte Webseite einen entsprechenden Hyperlink enthält. Die Wahrscheinlichkeit, dass ein Benutzer eine Webseite direkt aufruft, ohne einem Link zu folgen, wird im Folgenden *Sprungwahrscheinlichkeit* genannt. Sei n_{direkt} die Anzahl der direkten Anfragen einer Webseite und n_{link} entsprechend die Anzahl der Anfragen einer Webseite durch Folgen eines Hyperlinks. Für jede Webseite wird dann die Sprungwahrscheinlichkeit $P(\texttt{Sprung})$ berechnet als

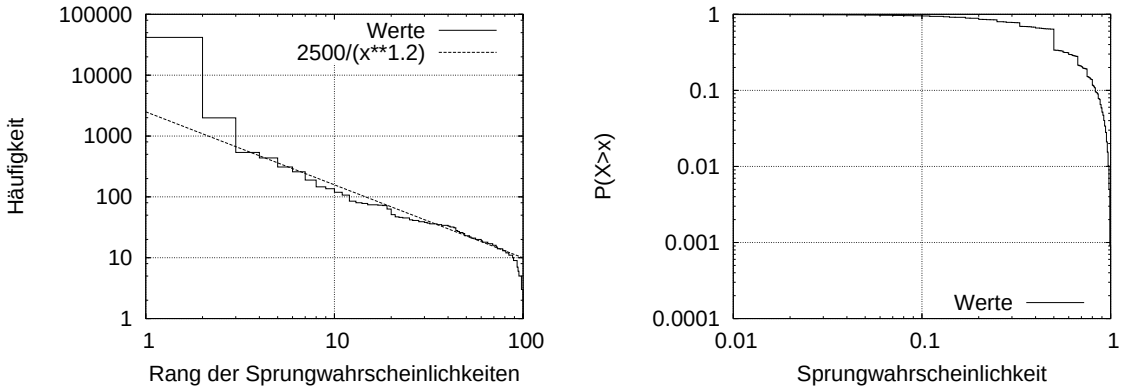
$$P(\texttt{Sprung}) = \frac{\lceil \frac{100 \cdot n_{direkt}}{n_{direkt}+n_{link}} \rceil}{100}$$

Um die Position eines angeforderten Links auf einer Webseite bestimmen zu können, wurden aus dem HTML-Dokument der Webseite alle <A HREF>-Elemente extrahiert und in der Reihenfolge ihres Auftretens in einer Liste gespeichert.

Modellierung des direkten Seitenaufrufs Die Sprungwahrscheinlichkeit wurde nach bestem Wissen bislang noch nicht untersucht. So setzen sie beispielsweise Page et al. in [81] und Henzinger et al. in [43] im *Random-Walk-Modell*, das in Abschnitt 4.5 diskutiert wird, auf einen festen Wert d. Mit der Wahrscheinlichkeit $1 - d$ wird einem Link gefolgt.

Unsere Untersuchungen ergaben jedoch, dass die Sprungwahrscheinlichkeit nicht konstant ist, sondern dass ihre Häufigkeitsverteilung Zipf-ähnlich ist mit einem Exponenten $\alpha = 1{,}2$. Hierzu wurden die Häufigkeiten der Sprungwahrscheinlichkeiten berechnet und anschließend sortiert, wobei die am häufigsten aufgetretene

Sprungwahrscheinlichkeit an erster Stelle steht. Nach den durchgeführten Messungen waren die zehn am häufigsten aufgetretenen Sprungwahrscheinlichkeiten 100%, 50%, 33%, 67% und 25%, 75%, 20%, 80%, 17%, 40%.



(a) Häufigkeitsverteilung der Sprungwahrscheinlichkeiten

(b) Verteilungsfunktion der Sprungwahrscheinlichkeiten

Abbildung 4.4: Eigenschaften der Sprungwahrscheinlichkeiten

Abbildung 4.4(a) zeigt die Häufigkeitsverteilung der Sprungwahrscheinlichkeiten. Auf der logarithmisch skalierten x-Achse sind die Sprungwahrscheinlichkeiten aufgetragen, sortiert nach der Häufigkeit ihres Auftretens. Die logarithmisch skalierte y-Achse zeigt die Häufigkeit selbst an. In dieser Abbildung ist gut zu erkennen, dass es sich bei Rang 1 mit 100% Sprungwahrscheinlichkeit um einen Ausreißer handelt. Dies ergibt sich aus der Tatsache, dass von den ca. siebzehntausend Sitzungen ungefähr achttausend nur bis zu maximal zehn Einträge hatten, davon allein viertausend nur einen oder zwei. In Abbildung 4.4(b) ist als Ergänzung die komplementäre kumulative Verteilungsfunktion der Sprungwahrscheinlichkeiten dargestellt. Um den Verlauf der Kurve für Werte zwischen 1% und 99% genauer darstellen zu können, wurde der Ausreißer 100% in dieser Abbildung nicht berücksichtigt. Mit Berücksichtigung dieses Wertes hätte sich die Kurve an die Achsen geschmiegt. Die Sprungwahrscheinlichkeiten an zweiter bis fünfter Position in der nach ihrer Häufigkeit sortierten Liste können mit Hilfe der Sprünge in der Verteilungsfunktion an den entsprechenden Stellen identifiziert

werden.

Aus der Häufigkeitsverteilung kann man zwar nicht direkt eine Sprungwahrscheinlichkeit ableiten, da innerhalb der Liste der Sprungwahrscheinlichkeiten keine Regelmäßigkeit zu erkennen ist. Sie zeigt jedoch deutlich, dass Benutzer eines bestimmten Typs ein ähnliches Zugriffsverhalten aufweisen. In diesem Fall handelt es sich bei den Benutzern um wissenschaftliches Personal und Studierende, die vermutlich oftmals die Lesezeichen des Browsers benutzen. Die Sortierung der Sprungwahrscheinlichkeiten kann jederzeit an beobachtetes oder erwartetes Benutzerverhalten angepasst werden. Die Wahrscheinlichkeit, dass ein Benutzer eine Webseite direkt aufruft, wird dann auf Grundlage dieser Liste mittels der Zipf-ähnlichen Verteilung berechnet.

Modellierung der Hyperlink-Auswahl Dieser Modellparameter beschreibt das Verhalten von Benutzern des Webs, wenn sie einen bestimmten Hyperlink auf einer Webseite auswählen, dem sie dann folgen. Nielsen beschreibt in [79], dass Benutzer überwiegend Hyperlinks folgen, die am Anfang einer Seite stehen, weil in diesem Fall der Fensterinhalt des Browsers nicht verschoben werden muss. Das Ergebnis der in dieser Arbeit durchgeführten Analysen bestätigt diese Vermutung. Abbildung 4.5 zeigt die komplementäre kumulative Verteilungsfunktion der Position eines ausgewählten Hyperlinks in der nach Auftreten sortierten Liste aller Links des entsprechenden HTML-Dokuments. Auf der logarithmisch skalierten x-Achse ist die Position eines Links in der Liste aufgetragen. Die logarithmisch skalierte y-Achse zeigt die Wahrscheinlichkeit $P(X > x)$. Die Hyperlink-Auswahl weist das Verhalten einer Doppel-Pareto-Verteilung mit Position 12 als Übergangswert und den Exponenten $\alpha_1 = 0{,}1$ und $\alpha_2 = 2{,}4$ auf. Für Hyperlinks auf Positionen kleiner als der Übergangswert ist die Auswahl nahezu gleichverteilt. Dies rührt vermutlich daher, dass für die Auswahl der ersten zwölf Hyperlinks in der Liste der Inhalt des Browser-Fensters üblicherweise nicht verschoben werden muss.

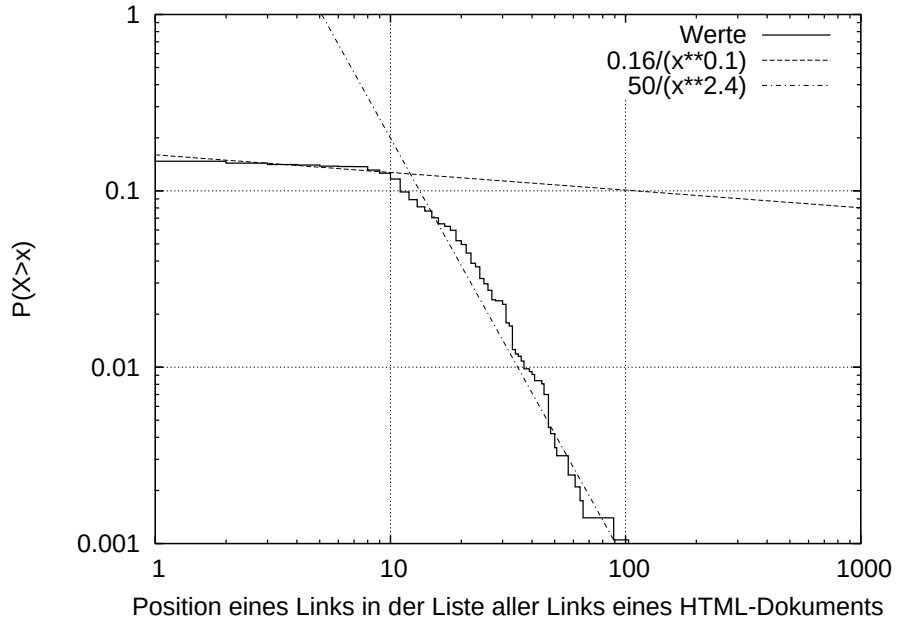


Abbildung 4.5: Verteilung der Positionen von angeklickten Hyperlinks

Modellierung der Sequenzlänge Die Sequenzlänge ist definiert als die Anzahl von Webseiten, die ein Benutzer innerhalb einer Sitzung anfordert. Huberman et al. untersuchen in [46], wie viele Webseiten ein Benutzer innerhalb eines Web-Auftritts anfordert. Die Annahme hierbei ist, dass Benutzer an jeder angeforderten Seite ein gewisses Interesse haben. Fällt dieses Interesse unter einen bestimmten Schwellwert, so beenden sie eine Sitzung. Untersuchungen der Protokolldateien von America Online (AOL) und Xerox im Jahr 1997 zeigen, dass die Verteilung der Sequenzlänge einem Potenzgesetz folgt, in diesem Fall einer inversen Gaußschen Verteilung, was sie das „Gesetz des Browsings" (engl. *law of surfing*) nannten. Basierend hierauf untersuchen Adar und Huberman in [3], ob das Navigationsverhalten von der Information selbst abhängt. Analysen der Protokolldateien von Excite und einem größeren anonymen Web-Portal ergaben, dass die Verteilungen der Sequenzlänge in Portalen mit verschiedenen Informationsangeboten dem obigen Gesetz des Browsings folgen, jedoch unterschiedliche Parameter aufweisen. Levene et al. modellieren in [64] die Sequenzlänge

als eine absorbierende Markov-Kette und zeigen, dass die Sequenzlänge einer Zipf-Verteilung folgt. Für die Evaluierung ihres Modells werteten sie die Protokolldateien der Universitäten Washington (1997) und Berkeley (1999) aus. Die Resultate bestätigen ihre Annahme, für den Zipf-Exponenten wurden Werte im Bereich zwischen 0,6 und 1,0 ermittelt.

Abbildung 4.6 zeigt die komplementäre kumulative Verteilungsfunktion der Sequenzlänge als das Resultat der eigenen Messungen. Auf der logarithmisch skalierten x-Achse ist die Sequenzlänge aufgetragen, die logarithmisch skalierte y-Achse zeigt die Wahrscheinlichkeit $P(X > x)$. Die Wahrscheinlichkeit für die Sequenzlänge folgt einer Doppel-Pareto-Verteilung mit einem Übergangswert von 100 Webseiten-Anforderungen und den Exponenten $\alpha_1 = 0{,}3$ und $\alpha_2 = 1{,}5$. Wie man deutlich sehen kann, sinkt die Wahrscheinlichkeit, dass ein Benutzer mehr als 100 Seiten innerhalb einer Sitzung anfordert, deutlich schneller als für kürzere Sitzungen. Dieser Übergangswert ist nur unwesentlich kleiner als die maximale Sequenzlänge, die von Levene et al. ermittelt wurde, der Exponent für Werte kleiner 100 liegt jedoch unter dem in Levene's Analysen ermittelten Wertebereich. Auf Grund dieser Analysen konnte in dieser Arbeit das Modell von Levene et al. erweitert werden, indem eine Verteilungsfunktion für Sequenzlängen größer als dem Übergangswert ermittelt wurde.

Eine weitere geeignete Modellierung der Sequenzlänge für ortsbezogene Anfragen, die beispielsweise in elektronischen Touristenführern vorkommen, ist eine Poisson-Verteilung mit einem Durchschnittswert λ. Die in [21] veröffentlichten Resultate des GUIDE-Projekts, das an der Universität Lancaster von Cheverst et al. durchgeführt wurde, zeigen eine durchschnittliche Informationsanforderung von ca. fünfundzwanzig Objekten pro Sehenswürdigkeit. In dem in dieser Arbeit vorgestellten Web-Navigationsmodell werden beide Modellierungen (Doppel-Pareto-Verteilung und Poisson-Verteilung) integriert.

Modellierung des wiederholten Besuchs Der wiederholte Besuch von Webseiten ist ein wichtiger Modellparameter für das in dieser Arbeit vorgestellte Web-

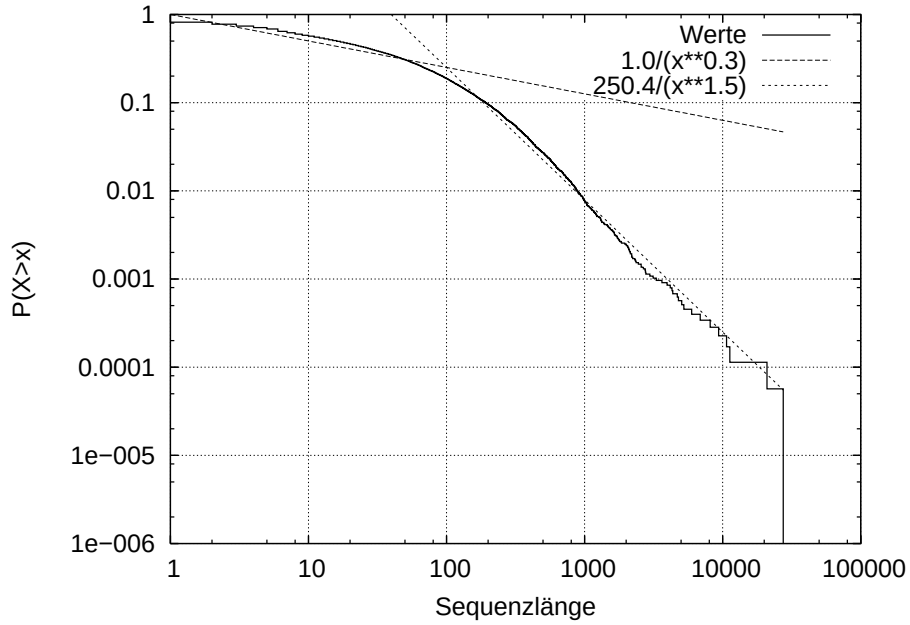


Abbildung 4.6: Verteilung der Sequenzlänge

Navigationsmodell, da die Tendenz, Webseiten innerhalb einer Sitzung mehrfach zu besuchen, sehr hoch ist: Gemäß den Resultaten der Untersuchungen von Tauscher und Greenberg im Jahr 1997 in [99] werden 58% der Webseiten innerhalb einer Sitzung mehrfach besucht. Cockburn et al. ermitteln in [23] sogar Raten zwischen 61% und 92%, mit einem Durchschnittswert von 81%.

Die Modellierung dieses Verhaltens beinhaltet zwei Verteilungen:

1. Die Aktion des wiederholten Besuchs, d.h., wann und wie oft werden bereits besuchte Seiten angefordert, und

2. welche der bereits besuchten Seiten wird aufgerufen.

Zu (1): Bezüglich der ersten Verteilung wurden in dieser Arbeit die Zugriffsmuster der in der oben erwähnten Protokolldatei ermittelten Benutzer analysiert: Die Wahrscheinlichkeit des wiederholten Besuchs einer Webseite als Benutzeraktion wird durch eine Lognormal-Verteilung mit dem Erwartungswert $\mu \approx 0,54$

und der Standardabweichung $\sigma \approx 0,23$ beschrieben. Abbildung 4.7 zeigt die komplementäre kumulative Verteilungsfunktion für den wiederholten Besuch als Aktion. Die logarithmisch skalierte x-Achse zeigt die Wahrscheinlichkeit, dass ein Benutzer eine Seite wiederholt besucht, auf der logarithmisch skalierten y-Achse ist die Wahrscheinlichkeit $P(X > x)$ aufgetragen.

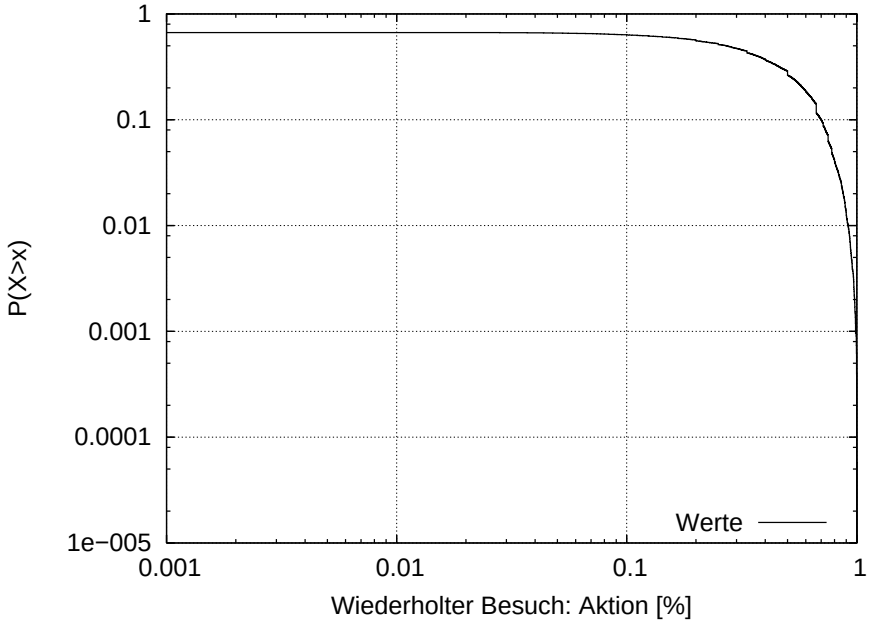


Abbildung 4.7: Verteilung des wiederholten Besuchs

Zu (2): Für die Auswahl einer Seite werden alle bereits besuchten Webseiten in einem Stapel (engl. *stack*) gespeichert. Mit Hilfe einer geeigneten Verteilung wird dann die Tiefe im Stapel (engl. *stack distance*) berechnet, an der sich die wiederholt zu besuchende Seite befindet. In [8] zeigen Barford und Crovella, dass diese Verteilung einer Lognormal-Verteilung folgt mit Erwartungswert $\mu = 1{,}5$ und Standardabweichung $\sigma = 0{,}8$. Demnach springen Benutzer beim wiederholten Besuch typischerweise eine bis zwei Seiten zurück. Dieser Modellierungsansatz wird in das Web-Navigationsmodell übernommen.

Modellierung der Popularität von Webseiten Die Wahrscheinlichkeit, dass eine bestimmte Webseite aufgerufen wird, hängt direkt mit ihrer Popularität zusammen. Nach den Resultaten zahlreicher Analysen in der Literatur wie beispielsweise in [12, 13, 30, 39, 76] folgt die Popularität einer Zipf-Verteilung mit einem Exponenten $alpha = 0{,}8$. In Abbildung 4.8 ist die Häufigkeitsverteilung der Popularität dargestellt, wie sie in den eigenen Analysen festgestellt wurde. Die logarithmisch skalierte x-Achse zeigt die nach Popularität sortierten Webseiten, auf der logarithmisch skalierten y-Achse ist die Häufigkeit aufgetragen, mit der eine Webseite angefordert wurde. Die eigenen Analysen bestätigen übereinstimmend die in der Literatur ermittelte Zipf-Verteilung.

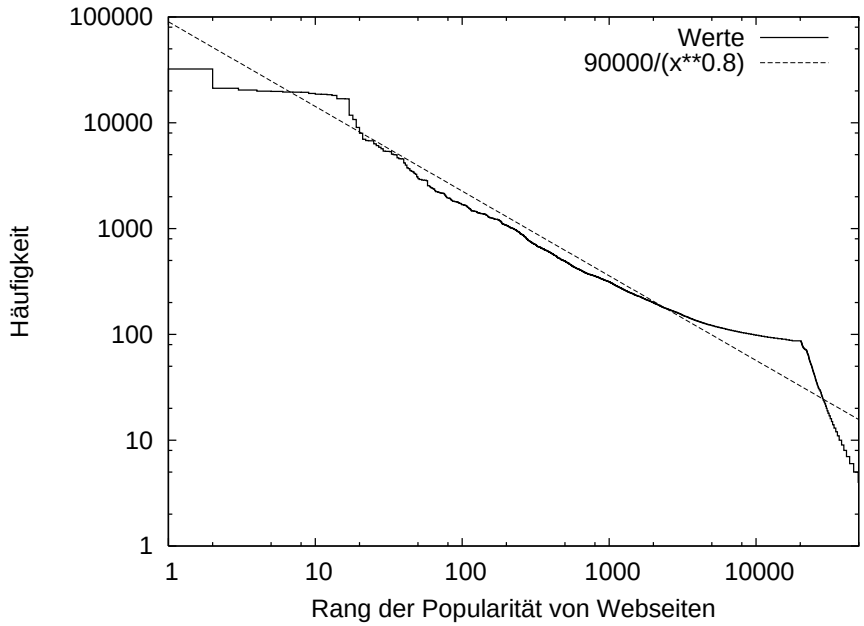


Abbildung 4.8: Verteilung der Popularität von Webseiten

Modellierung der Besuchsdauer einer Webseite Die Besuchsdauer einer Webseite beschreibt, wie lange sich ein Benutzer eine Webseite ansieht und wird als Indikator für das Interesse eines Benutzers an der Webseite verwendet. Crovella und Bestavros [29] sowie Barford und Crovella [8] untersuchten die

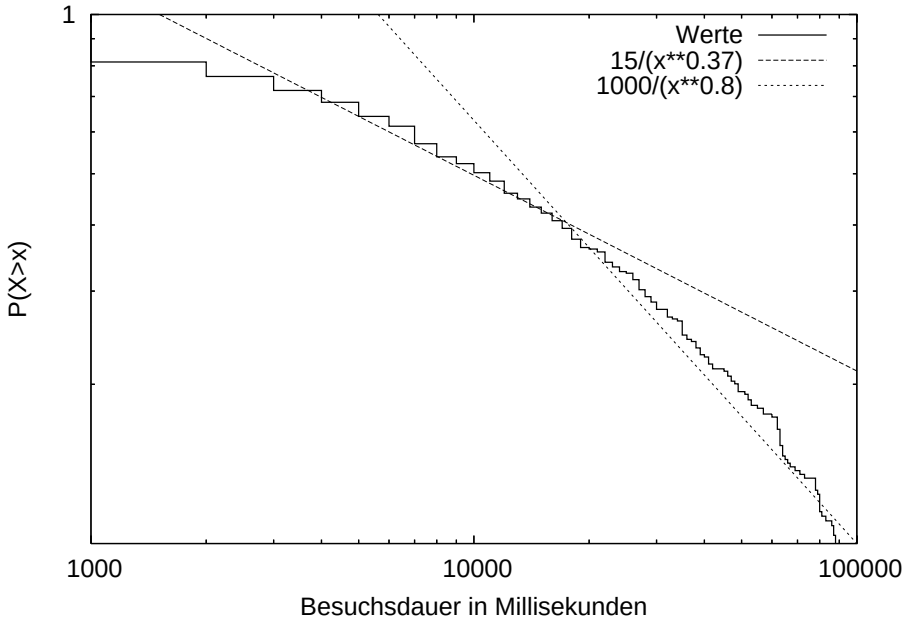


Abbildung 4.9: Verteilung der Besuchsdauer von Webseiten

so genannten OFF-Zeiten, während derer keine Datenübertragung statt findet. Dabei unterscheiden sie zwischen den aktiven und inaktiven OFF-Zeiten. Aktive OFF-Zeiten sind solche, während denen die in die Webseite eingebetteten Dokumente geladen werden. Inaktive OFF-Zeiten sind Übertragungspausen größer als dreißig Sekunden, während denen der Benutzer keine Eingaben tätigt. Die Analysen von Barford und Crovella zeigen, dass die Wahrscheinlichkeit für eine inaktive OFF-Zeit einer Pareto-Verteilung mit einem Exponenten $\alpha = 1{,}5$ folgt. In den im Rahmen dieser Arbeit durchgeführten Messungen, deren Ergebnisse in Abbildung 4.9 dargestellt sind, wird die Besuchsdauer durch Subtraktion der Zeitstempel zweier aufeinander folgenden Protokolleinträge berechnet. Die Analysen zeigen, dass die Besuchsdauer einer Doppel-Pareto-Verteilung mit einem Übergangswert von ca. 12 Sekunden und den beiden Exponenten α_1=$0,37$ und α_2=$0,8$ folgt, die schließlich in das Web-Navigationsmodell übernommen wird.

Modellierung von Profilen Wie bereits in Abschnitt 3.4.5 beschrieben, wird ein Nutzungsprofil einem Informationskanal zugeordnet, der die auf dieses Profil zugeschnittenen Informationen zur Verfügung stellt. Zur Modellierung des Informationszugriffs mit unterschiedlichen Nutzungsprofilen muss zuerst einmalig der Informationsraum auf die unterschiedlichen Profile aufgeteilt werden. Anschließend wird dann für einen (virtuellen) Benutzer mit Hilfe dieses Modells ein Profilmix berechnet. Nach bestem Wissen wurde der Informationszugriff mit unterschiedlichen Profilen noch nicht untersucht, es stehen auch keine derartigen Protokolldateien öffentlich zur Verfügung. Die in Abschnitt 4.2 erwähnte Protokolldatei kann ebenfalls nicht zu diesem Zweck verwendet werden, da sich nicht feststellen lässt, ob die anonymisierten Benutzer unterschiedliche Profile nutzten. Nachfolgend wird deshalb eine Modellierung des Zugriffsverhaltens mit Profilen vorgeschlagen, die auf eigenen Annahmen beruht.

Die *Zuordnung der Webseiten zu Profilen* erfolgt gleichverteilt: Jedem Nutzungsprofil wird ein gleich großer, disjunkter Teil des Informationsraums zugeteilt, aus dem im Fall des direkten Seitenaufrufs eine Webseite entsprechend ihrer Popularität gewählt wird. Die Gleichverteilung erfolgt unter der Annahme, dass sich das Informationsbedürfnis unterschiedlicher Benutzergruppen zwar wesentlich unterscheidet, es jedoch keine bevorzugten Profile gibt, mit denen mehr Webseiten angefordert werden als mit anderen. Dieses auf Profilen beruhende Informationsbedürfnis zeigt sich vor allem bei direkt angeforderten Webseiten, da ein Benutzer diese bewusst auswählt. Das Folgen eines Hyperlinks wird von den Profilen nicht direkt beeinflusst. Hierdurch wird modelliert, dass nicht der gesamte Informationsraum strikt unterteilt ist, sondern dass es durchaus auch Webseiten gibt, die von allgemeinem Interesse sind.

Bei der *Berechnung eines Profilmixes* für einen Benutzer wird zunächst die Anzahl der im Profilmix enthaltenen Profile gemäß einer Zipf-Verteilung berechnet, wobei kleinere Zahlen häufiger vorkommen sollen. Wie sich durch die vorangegangenen Modellierungen gezeigt hat, eignen sich endlastige Verteilungen hervorragend zur Modellierung des Zugriffsverhaltens von Benutzern im Web. Die

Zipf-Verteilung wurde gewählt, da es sich hier um eine Häufigkeitsverteilung handelt. Die Anzahl der Profile im Profilmix werden aufsteigend sortiert unter der Annahme, dass die meisten Benutzer nur wenige Nutzungsprofile gleichzeitig für die Navigation im Web verwenden, während nur eine relativ geringe Anzahl eine größere Anzahl von Profilen gleichzeitig benutzt. Unter der Annahme, dass es keine bevorzugten Profile gibt, werden die im Profilmix enthaltenen Profile mitsamt der Wahrscheinlichkeit ihres Auftretens gleichverteilt ausgewählt.

Zusammenfassung der Modellparameter Die in das Web-Navigationsmodell integrierten Verteilungen für die einzelnen Modellparameter mitsamt der Belegung ihrer Kenngrößen ist in Tabelle 4.1 zusammenfassend dargestellt.

Tabelle 4.1: Modellparameter

Modellparameter	**Verteilung**	**Kenngröße**	**Wert**
Hyperlink-Auswahl	Doppel-Pareto	Exponent (< 10)	0,1
		Exponent (> 10)	2,4
Direkter Seitenaufruf	Zipf-ähnlich	Exponent	1,2
Wiederholter Besuch (Aktion)	Lognormal	Erwartungswert μ	0,54
		Standardabw. σ	0,23
Wiederholter Besuch (Seite)	Lognormal	Erwartungswert μ	1,5
		Standardabw. σ	0,8
Sequenzlänge (1)	Poisson	Durchschnitt	25
Sequenzlänge (2)	Doppel-Pareto	Exponent (< 110)	0,3
		Exponent (> 110)	1,5
Popularität der Webseiten	Zipf-ähnlich	Exponent	0,8
Besuchsdauer	Doppel-Pareto	Exponent (> 12)	0.8
		Exponent (< 12)	0,37
Profil (Anzahl)	Zipf	Exponent	0,8
Profil (Wahrsch.)	Gleichvert.	—	—
Profil (Aufteilung der Seiten)	Gleichvert.	—	—
Eingangsgrad	Zipf-ähnlich	Exponent	2,1
Ausgangsgrad	Zipf-ähnlich	Exponent	2,72
SCC	Fest	–	0,28
IN	Fest	–	0,21
OUT	Fest	–	0,21
Ranken	Fest	–	0,21
Entkoppelt	Fest	–	0,09
Seitengröße	Doppel-Pareto	Exponent (< 10)	0,3
		Exponent (> 10)	1,1

4.3 Integration des Web-Navigationsmodells in die Simulationsumgebung

Das Web-Navigationsmodell wurde in Java 5.0 implementiert und in die Simulationsumgebung integriert. Der Anfragengenerator „User Centric Walk" (UCW) ist die Kernkomponente und beinhaltet die Erzeugung von Protokolleinträgen sowie die automatische Generierung eines Webgraphen.

In diesem Abschnitt wird zunächst mit Hilfe eines Ablaufdiagramms die Vorgehensweise zur Erzeugung von synthetischen Protokolldateien geschildert. Daran anschließend wird die Architektur der Simulationsumgebung vorgestellt, gefolgt von einer Beschreibung der Schnittstellen.

4.3.1 Ablaufdiagramm

Das *Ablaufdiagramm* des Anfragengenerators UCW ist in Abbildung 4.10 dargestellt. Rauten kennzeichnen die Berechnung einer Wahrscheinlichkeit, wobei der nächste Schritt vom ermittelten Ergebnis abhängt. Ellipsen beschreiben die Berechnung von Wahrscheinlichkeiten ohne nachfolgende Entscheidungsfindung, der nächste Schritt ist also vorgegeben. Das Parallelogramm schließlich dokumentiert das Zusammensetzen des Protokolleintrags und dessen Eintrag in die Protokolldatei.

Zu Beginn werden ein Profilmix und die Anzahl von Einträgen (Sequenzlänge) der Protokolldatei berechnet. Die Anforderung der ersten Webseite in der Protokolldatei erfolgt durch direkten Seitenaufruf. Die Auswahl der direkt anzufordernden Webseite erfolgt auf Grundlage der Verteilungsfunktion für die Popularität von Webseiten. Falls der entsprechende Knoten noch nicht im Webgraphen enthalten ist, wird zunächst mit Hilfe des Webgraph-Modells die Komponente im Webgraphen ermittelt (Kern, EIN, AUS, ...). Anschließend wird der Knoten mit der berechneten Seitengröße sowie den ermittelten Ein- und Ausgangsgraden

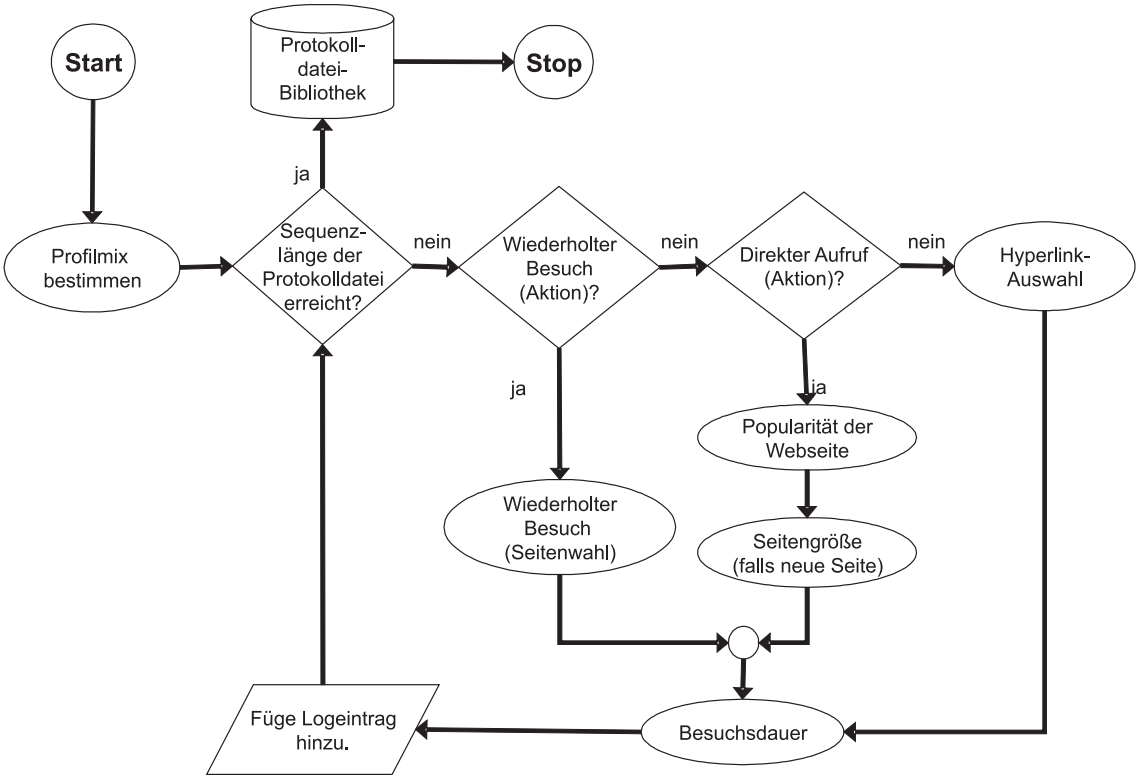


Abbildung 4.10: Ablaufdiagramm des Anfragengenerators UCW

in den Webgraphen eingefügt. Anschließend wird eine Besuchsdauer berechnet, der Protokolleintrag zusammengesetzt und in die Protokolldatei eingetragen.

Solange die erforderliche Anzahl von Einträgen noch nicht generiert wurde, werden die folgenden Schritte ausgeführt.

Für jeden weiteren Protokolleintrag wird anschließend zunächst die Wahrscheinlichkeit berechnet, mit der ein Benutzer eine bereits besuchte Seite anfordert. Falls ja, wird die wiederholt zu besuchende Webseite ermittelt. Falls nein, wird die Wahrscheinlichkeit für einen direkten Seitenaufruf berechnet. Soll die nächste Seite direkt angefordert werden, wird eine neue Webseite mit Hilfe der Verteilungsfunktion für die Popularität von Webseiten ausgewählt. Existiert der entsprechende Knoten noch nicht, wird er wie oben beschrieben erzeugt und in

den Webgraphen eingefügt. Falls einem Hyperlink gefolgt werden soll, wird aus den ausgehenden Kanten des aktuellen Knotens mittels der Verteilungsfunktion für die Hyperlink-Auswahl die entsprechende (ausgehende) Kante gewählt. Falls noch kein Knoten an diese Kante gebunden ist, wird unter Berücksichtigung der Popularität von Webseiten und der Konnektivitätsbedingungen der Webgraph-Komponenten ein Knoten mit einer freien eingehenden Kante bestimmt. Befindet sich beispielsweise der aktuelle Knoten in der AUS-Komponente, so kann die zu bestimmende Webseite nicht Element der IN-Komponente sein. Wurde die nächste anzufordernde Webseite ermittelt, wird eine Besuchsdauer berechnet, der Protokolleintrag zusammengesetzt und in die Protokolldatei eingetragen.

Sobald die erforderliche Anzahl von Einträgen erstellt wurde, wird die Protokolldatei mit einem Bezeichner versehen und in eine Bibliothek geschrieben. Diese stellt die Protokolldateien Anwendungen wie beispielsweise der Cache-Verwaltung zur Verfügung.

4.3.2 Architektur

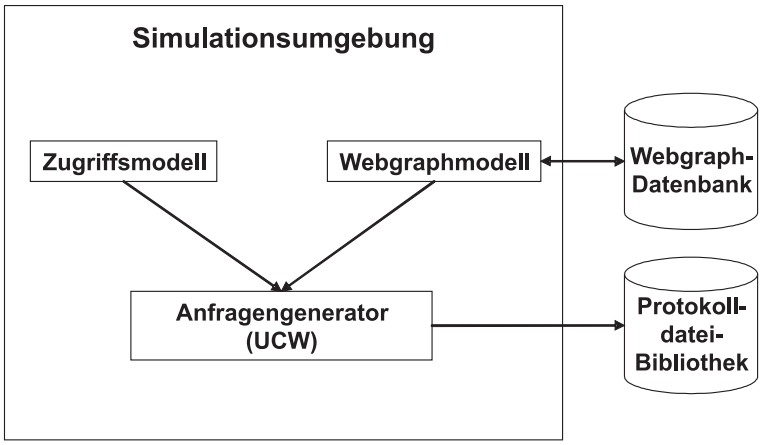


Abbildung 4.11: Architektur der Simulationsumgebung

Abbildung 4.11 beschreibt die Architektur der Simulationsumgebung für das vorgestellte Vorabübertragungsverfahren, mit deren Hilfe synthetische Protokoll-

dateien erzeugt werden, die zur Leistungsbewertung des Verfahrens verwendet werden können.

Der Anfragengenerator UCW ist die Hauptkomponente der Simulationsumgebung. Er ist eine Implementierung des Web-Navigationsmodells und erzeugt mit Hilfe des Zugriffsmodells und des Webgraph-Modells eine Sequenz von Webseiten-Anforderungen für eine Protokolldatei. Die erstellten Protokolldateien werden in eine Protokolldatei-Bibliothek geschrieben.

Das *Webgraph-Modell* verwaltet die Informationen über Knoten und Kanten des Webgraphen in einer Webgraph-Datenbank. Ein zugrunde liegender Webgraph wird bei Bedarf mit jedem Programmlauf erweitert.

Das *Zugriffsmodell* liefert die in Abschnitt 4.2.2 beschriebenen Verteilungsfunktionen für die Auswahl von Webseiten oder deren Besuchsdauer.

4.3.3 Beschreibung der Schnittstellen

Nachfolgend werden die Schnittstellen der einzelnen Teilmodelle definiert.

Anfragengenerator

Der Anfragengenerator UCW ist der Kern der Simulationsumgebung und stellt die folgende Schnittstelle nach außen zur Verfügung:

erzeugeSequenz(*String* ID, *String* vert, *Vector* seiten, *Vector* sprung):

In dieser Methode wird eine Sequenz von Protokolleinträgen erstellt und die generierte Protokolldatei in einer Bibliothek gespeichert. *ID* ist ein Bezeichner für die Protokolldatei, mit Hilfe dessen sie später wieder aus der Bibliothek gelesen werden kann. Das Attribut *vert* bestimmt, welche der beiden Verteilungen für die Sequenzlänge verwendet werden sollen. *seiten* ist ein Feld mit den Bezeichnern aller Webseiten eines

Informationsraums, sortiert nach Popularität, und *sprung* ist ein Feld, das die nach Popularität sortierten Sprungwahrscheinlichkeiten enthält.

UCW wird immer dann gestartet, wenn eine oder mehrere Protokolldateien erzeugt werden sollen.

Webgraph-Modell

Das Webgraph-Modell verwaltet den Webgraphen und stellt die hierfür notwendigen Verteilungsfunktionen zur Verfügung. Der Webgraph wird in einer Webgraph-Datenbank verwaltet. Das Webgraph-Modell wird von UCW auf Grundlage der nachfolgend beschriebenen Schnittstellen verwendet.

***String* webKomponente():** Diese Methode gibt die Komponente innerhalb des Webgraphen zurück (Kern, EIN, AUS, Ranken, entkoppelte Komponente).

***double* seitengröße(*double* alpha1, *double* alpha2, *int* min, *int* max):** Diese Methode gibt die Größe einer Webseite zurück. *alpha1* ist der Exponent der Doppel-Pareto-Verteilung für Webseiten kleiner als zehn KBytes, *alpha2* ist der Exponent für Webseiten größer als zehn KBytes, *min* und *max* sind die minimale bzw. maximale Seitengröße.

***double* eingangsgrad(*double* alpha):** Diese Methode berechnet den Eingangsgrad eines Knotens. *alpha* ist der Exponent der Zipf-ähnlichen Verteilung.

***double* ausgangsgrad(*double* alpha):** Diese Methode berechnet den Ausgangsgrad eines Knotens. *alpha* ist der Exponent der Zipf-ähnlichen Verteilung.

Zugriffsmodell

Das Zugriffsmodell stellt die für die Modellierung des Zugriffsverhaltens erforderlichen Verteilungen zur Verfügung. Diese können von UCW mit den nachfolgend beschriebenen Schnittstellen angefordert werden.

***Vector* profilmix():** Diese Methode liefert einen Profilmix.

***int* sequenzlänge(*double* alpha1, *double* alpha2, *int* min):** Mit Hilfe dieser Methode wird die Anzahl der Protokolleinträge gemäß der Doppel-Pareto-Verteilung berechnet. *alpha1* und *alpha2* sind die Exponenten der Doppel-Pareto-Verteilung für Werte kleiner bzw. größer als der Übergangswert, *min* ist eine minimale Sequenzlänge.

***int* sequenzlängePoisson(*int* lambda):** Diese Methode gibt die Anzahl der Protokolleinträge zurück, die gemäß einer Poisson-Verteilung mit dem Durchschnittswert *lambda* berechnet werden.

***int* popularität(*double* alpha):** Diese Methode berechnet den Rang einer Webseite in der nach Popularität sortierten Liste. *alpha* ist der Exponent der Zipf-Verteilung.

***double* aktionWiederholterBesuch(*double* mu, *double* sigma):** Diese Methode berechnet die Wahrscheinlichkeit, dass eine bereits besuchte Seite angefordert wird. *mu* ist der Erwartungswert, *sigma* die Standardabweichung der Lognormal-Verteilung.

***int* tiefeWiederholterBesuch(*double* mu, *double* sigma):** Diese Methode berechnet die Tiefe im Stapel bereits besuchter Webseiten, um die wiederholt zu besuchende Seite zu bestimmen. *mu* ist der Erwartungswert, *sigma* die Standardabweichung der Lognormal-Verteilung.

***double* sprungwahrscheinlichkeit(*double* alpha):** Diese Methode berechnet den Rang der Wahrscheinlichkeit, dass die nächste Seite direkt angefordert wird. Der Rang bezieht sich auf die Position der Sprungwahrscheinlichkeit in der entsprechenden, nach Häufigkeit sortierten Liste.

alpha ist der Exponent der Zipf-Verteilung.

***int* besuchsdauer(*double* alpha1, *double* alpha2, *int* min):** Diese Methode berechnet die Besuchsdauer in Sekunden einer Webseite. *alpha1* und *alpha2* sind die Exponenten der Doppel-Pareto-Verteilung für Werte kleiner bzw. größer als der Übergangswert. *min* ist die minimale Besuchsdauer.

4.4 Mögliche Erweiterung

Wie in Abschnitt 3.4.1 beschrieben, enthalten Einträge in Protokolldateien den Ort der Informationsanforderung. Um diesen Ort möglichst realistisch zu bestimmen, wird die *Integration eines Mobilitätsmodells* in die Simulationsumgebung notwendig, das als Ausgabe Bewegungstrajektorien liefert, also Orte, an denen sich Benutzer zu bestimmten Zeitpunkten aufhalten. Ein solches Modell, das Benutzerbewegungen in einem bestimmten geographischen Gebiet sehr detailliert modelliert, wurde in der Abteilung Verteilte Systeme des Instituts für Parallele und Verteile Systeme der Universität Stuttgart von Stepanov et al. entwickelt [95–97]. Die Integration eines Mobilitätsmodells ist zwingend erforderlich, wenn das in Kapitel 3 beschriebene Verfahren um die Einbeziehung von Wissen über zukünftige Aufenthalte von Benutzern erweitert wird. In dieser in Abschnitt 3.11 beschriebenen Erweiterung wurde zur Modellierung der Bewegungsmuster von Benutzern das Dienstgebiet in Zonen aufgeteilt. Die mittels des Mobilitätsmodells erzeugten Aufenthaltsorte werden den zu besuchenden Zonen zugeordnet.

Die Modellierung des ortsabhängigen Informationszugriffs kann analog zu der bei der Modellierung von Profilen angewandten Methode erfolgen, die in Abschnitt 4.2.2 beschrieben worden ist. Hierfür ist eine (einmalige) Zuordnung von Teilen des Informationsraums zu den einzelnen Zonen erforderlich. Jeder Zone im Dienstgebiet wird ein Teil der Webseiten des gesamten zugrunde liegenden In-

formationsraums zugeordnet, die von Benutzern direkt aufgerufen werden. Diese Teile können disjunkt sein oder sich überlappen. Für nicht disjunkte Teile muss für jede Webseite festgelegt werden, wie vielen Zonen sie zugeordnet wird. Hierfür bietet sich die Zipf-Verteilung unter der Annahme an, dass für den ortsbezogenen Informationszugriff zwar die meisten Webseiten nur in einigen wenigen Zonen angefordert werden, es aber durchaus Webseiten geben kann, die in vielen Zonen beliebt sind. Anschließend müssen für jede Webseite, die von Benutzern direkt aufgerufen werden, die zugeordneten Zonen gleichverteilt ermittelt werden. Entsprechend der Modellierung des Informationszugriffs mit Nutzungsprofilen ist das Folgen eines Hyperlinks auf einer Seite unabhängig vom aktuellen Ort.

4.5 Verwandte Arbeiten

In der Literatur wurden mehrere Ansätze zur Modellierung des Navigationsverhaltens von Benutzern des Webs diskutiert. Die meisten von ihnen modellieren die Navigation innerhalb eines Web-Auftritts, während sich nur wenige mit der Navigation im gesamten Web beschäftigen.

In transaktionalen Vergleichstests für Anwendungen im Bereich des elektronischen Handels (TPC-W benchmarks) werden Zugriffsmuster von Benutzern mit Hilfe eines Graphen modelliert (engl. *customer behavior model graph*, CBMG), wie unter anderem von Menascé in [70] und von Dodge et al. in [34] beschrieben. Dieser Graph zeigt, wie Benutzer innerhalb des Web-Auftritts eines solchen Unternehmens navigieren. Knoten repräsentieren die durch das System vorgegebenen Zustände wie beispielsweise *browsen*, *in den Warenkorb legen*, *bezahlen* oder ähnliches. Kanten repräsentieren die Wahrscheinlichkeiten der Zustandsübergänge. Basierend auf der Traversierung dieses Graphen werden dann Sequenzen von Benutzeranfragen konstruiert. Krishnamurthy und Rolia modellieren in [53] das Navigationsverhalten innerhalb eines Web-Auftritts mit Hilfe eines so genannten URL-Graphen, dessen Knoten die Webseiten und dessen Kanten die Hyperlinks

zwischen den Seiten repräsentieren. Hierbei unterscheiden die Autoren zwischen Navigations-URLs und Transaktions-URLs, wobei letztere besonders die wichtigen Einkaufsoperationen strapazieren. Ein Lastgenerator erzeugt eine Reihe von Benutzeranfragen unter der Annahme, dass nach einer Transaktions-URL eine bestimmte Anzahl von Navigations-URLs angefordert werden. Die Auswahl eines Hyperlinks auf einer Seite erfolgt gleichverteilt. Das Ziel dieser Ansätze ist die Optimierung von Web-Auftritten im Bereich des elektronischen Handels, während das in dieser Arbeit vorgestellte Web-Navigationsmodell das gesamte Web betrachtet und die einzelnen Zugriffsmuster sehr viel detaillierter modelliert.

In [84] stellen Pitkow et al. einen Modellierungsansatz vor, der auf vorangegangenen Webseiten-Anforderungen eines Benutzers basiert. Auf der Basis von Markov-Modellen k-ter Ordnung definieren sie Algorithmen zur Mustererkennung (eng. *pattern extraction*) und zum Mustervergleich (engl. *pattern matching*), die auf der Auswertung von Teilsequenzen beruhen. Bei Markov-Modellen k-ter Ordnung wird der neue Zustand auf der Basis der letzten k Zustände berechnet. Im Gegensatz zu dem hier vorgestellten Navigationsmodell hängt die Qualität der Resultate direkt von den zur Verfügung gestellten Trainingsdaten ab. Weiterhin werden nur solche Aufrufsequenzen erzeugt, die Muster enthalten, die bereits in den Trainingsdaten enthalten sind.

Das *Random-Walk-Modell* wird unter anderem im *PageRank*-Algorithmus von Page et al. verwendet, der in [81] vorgestellt wird und Seiten mittels syntaktischer Analysen zur Optimierung von Suchmaschinen bewertet. Die grundlegende Idee dieses Modell ist, dass ein Benutzer mit einer bestimmten Wahrscheinlichkeit d eine beliebige Seite anfordert und mit der Wahrscheinlichkeit $1 - d$ einem Hyperlink folgt. Die Auswahl des Links auf einer Seite sowie die Auswahl einer Seite beim direkten Seitenaufruf erfolgen gleichverteilt.

Ein Ansatz, der zusätzlich die Wahrscheinlichkeit des wiederholten Besuchs und der Hyperlink-Auswahl mit einbezieht, wird von Diligenti et al. in [32] vorgestellt. Die Autoren entwickelten ein Rahmenwerk zur Modellierung des Navigationsverhaltens von Benutzern im Web, genannt *Web Page Scoring Systems* (WPSS).

Das Modell basiert auf der Annahme, dass es vier Aktionen gibt, die ein Benutzer ausführen kann: einem Hyperlink folgen, eine Seite direkt aufrufen, eine Seite wiederholt besuchen und auf der letzten Seite bleiben. Die Wahrscheinlichkeit, wie sich ein Benutzer von einer Seite zu einer anderen bewegt, wird mit Hilfe dreier Matrizen berechnet, welche die jeweiligen Wahrscheinlichkeiten für einen Zustandsübergang für die oben genannten ersten drei Aktionen beinhalten. Während in dem in dieser Arbeit vorgestellten Web-Navigationsmodell jede beliebige bislang besuchte Webseite wiederholt angefordert werden kann, beschränkt sich WPSS auf die unmittelbar zuvor aufgerufene Seite. Obwohl WPSS und das Web-Navigationsmodell ähnlich aufgebaut sind, verfolgen beide Ansätze unterschiedliche Ziele. WPSS ist für Verfahren optimiert, die ein Ranking für Webseiten durchführen, während das Web-Navigationsmodell auf die Erstellung von Aufrufsequenzen für die Evaluierung Web-basierter Verfahren wie dem vorgestellten Vorabübertragungsverfahren abzielt. Hierzu werden zusätzliche Modellparameter wie die Besuchsdauer oder Größe einer Webseite benötigt. Schließlich wird in dieser Arbeit eine konkrete Implementierung des Modells vorgestellt und nicht nur ein Rahmenwerk, mit der zusätzlich noch ein Webgraph in Realzeit generiert werden kann.

Abbildung 4.12 fasst die unterschiedlichen, in diesem Abschnitt diskutierten Modellierungsansätze zusammen. Die ersten beiden Verfahren werden zur Optimierung von Web-Auftritten verwendet, die nächsten beiden zur Optimierung von Verfahren zur Bewertung von Suchmaschinen. das letzte, grau unterlegte Verfahren stellt den Anfragengenerator UCW dar, der in dieser Arbeit entwickelt wurde. Ein Häkchen bedeutet, dass die entsprechende Funktionalität von einem Verfahren angeboten wird, während ein Kreuz bedeutet, dass eine entsprechende Verteilungsfunktion nicht berücksichtigt wird.

	Informations-raum	#An-fragen	Web-seiten-größe	Besuchs-dauer	Popularität der Seiten	Direkter Aufruf	Link-auswahl	Wieder-holter Besuch: Aktion	Wieder-holter Besuch: Ziel
CBMG	Webauftritt	✓	✗	✓	✗	✗	✓	✗	✗
URL-Graph	Webauftritt	✓	✓	✓	✗	✗	✓	✗	✗
Page-Rank	Ges. Web	✗	✗	✗	✗	✓	✓	✗	✗
WPSS	Ges. Web	✓	✗	✗	✗	✓	✓	✓	✗
UCW	Ges. Web	✓	✓	✓	✓	✓	✓	✓	✓

CMBG, URL-Graph: Optimieren Webauftritte (elektronischer Handel)
Page-Rank, WPSS: Optimieren Algorithmen zur Bewertung von Suchergebnissen
UCW: Anfragengenerator (z.B. zur Bewertung von Vorabübertragungsverfahren)

Abbildung 4.12: Charakteristika der Modellierungsansätze für den Zugriff auf das Web

4.6 Zusammenfassung

In diesem Abschnitt wurde das Web-Navigationsmodell vorgestellt, das überwiegend auf bekannten und evaluierten Modellen und Verteilungen basiert. Es stellt zwei weitere, nach bestem Wissen noch nicht bekannte Verteilungsfunktionen zur Verfügung: für den direkten Seitenaufruf und die Auswahl eines Hyperlinks auf einer Webseite. Die in der Literatur bekannten Verteilungen wurden durch eigene umfangreiche Messungen bestätigt und zum Teil erweitert, wie beispielsweise die Anzahl der Webseiten-Anforderungen innerhalb einer Sitzung oder die Besuchsdauer einer Webseite. Die eigenen Messungen wurden zur Kalibrierung des Modells verwendet. Schließlich wurde ein Modell für die Zuordnung eines Profilmixes vorgeschlagen. Als Erweiterung wurde die Integration eines Mobilitätsmodells und die damit verbundene Zuordnung von Teilen des Informationsraums auf die einzelnen Zonen im Dienstgebiet skizziert.

5 Simulative Leistungsbewertung

In diesem Kapitel wird zunächst die Problemstellung erörtert, weshalb die Leistungsbewertung simulativ erfolgt. Anschließend werden die Eigenschaften eines dem Verfahren zugrunde liegenden Informationsraums beschrieben, welche die Leistung des Vorabübertragungsverfahrens wesentlich beeinflussen. Nachdem die Methodik der Leistungsbewertung aufgezeigt wurde, schließt das Kapitel mit der Diskussion einiger repräsentativer Ergebnisse der Leistungsbewertung des Vorabübertragungsverfahrens. Diese wurden auf Grundlage umfangreicher Simulationen erzielt, die mit der in Abschnitt 4.3 vorgestellten Simulationsumgebung durchgeführt wurden.

5.1 Problemstellung

Das übergeordnete Ziel der Leistungsbewertung war eine systematische Evaluierung des für den Zugriff auf Webseiten spezialisierten Vorabübertragungsverfahrens. Die erwartete Trefferrate hängt offensichtlich von der Anzahl der in einem Informationsraum enthaltenen Webseiten ab: Je kleiner diese Zahl ist, desto bessere Trefferraten sind zu erwarten. In diesem Fall ist es interessant zu wissen, wie sich die Trefferrate mit steigender Anzahl von Webseiten verhält. Nicht so offensichtlich ist jedoch, welchen Einfluss die durchschnittliche Größe der in einem Informationsraum enthaltenen Webseiten auf das Ergebnis hat. Infolgedessen sollte das Verfahren für unterschiedlich dimensionierte Informationsräume evaluiert werden. Um ein statistisch relevantes Ergebnis zu erzielen, sollte weiterhin

eine Vielzahl von Protokolldateien zur Verfügung stehen, die mit diesen unterschiedlich dimensionierten Informationsräumen assoziiert sind. Schließlich sollten zusätzliche Protokolldateien verwendet werden, die mit unterschiedlichen Nutzungsprofilen erzeugt wurden. Aus diesen Gründen wurde die Leistungsbewertung simulativ durchgeführt. Hierzu wurden die Informationsräume und die zugehörigen Protokolldateien synthetisch mit dem Anfragengenerator UCW (User Centric Walk) erzeugt, einer Implementierung des in Kapitel 4 beschriebenen Web-Navigationsmodells.

Weitere Ziele waren der Vergleich mit anderen Vorabübertragungsverfahren sowie die Untersuchung, welche Trefferraten durch das Einbeziehen von Profilen zu erwarten sind. Darüber hinaus war es interessant, zu sehen, wie schnell das Verfahren lernen kann, das heißt, wie viele Protokolldateien erforderlich sind, um akzeptable Trefferraten zu erzielen. Schließlich wurde das Verfahren mit verschiedenen Parameterbelegungen evaluiert, um die Reaktion auf unterschiedliche Randbedingungen zu testen.

Selbst in ortsbasierten Anwendungen ist es möglich, dass ein Teil der Zugriffe nicht ortsbezogen ist. In der Evaluierung wurde dieser Anteil jedoch nicht berücksichtigt, da solche Anfragen typischerweise vom Cache nicht beantwortet werden können und somit die Trefferrate verringern. Zur Behandlung dieses Anteils können Caching-Verfahren eingesetzt werden, die jedoch nicht Thema dieser Arbeit sind. Es ist leicht einzusehen, dass die mit dem Verfahren erzielten Trefferraten um so besser sein werden, je höher der Anteil an ortsbezogenen Zugriffen ist. Deshalb beziehen sich die in diesem Kapitel beschriebenen Ergebnisse auch nur auf den ortsbezogenen Anteil der Zugriffe.

5.2 Eigenschaften eines Informationsraums

Wie bereits in Abschnitt 3.3.1 erwähnt wurde, kann der dem Verfahren zugrunde liegende Informationsraum, wie beispielsweise das Web, typischerweise sehr groß

sein. Dadurch, dass das vorgestellte Verfahren den Ortsbezug der Informationen ausnutzt, kann man davon ausgehen, dass sich die Anfragen im Dienstgebiet auf einen kleinen Teil des Informationsraums beschränken werden. Diese Teilmenge wird nachfolgend *ortsbezogener Teilraum* genannt und wie folgt definiert:

Definition 25 (Ortsbezogener Teilraum) *Ein Informationsraum ist ein* ***ortsbezogener Teilraum****, wenn die Zugriffe auf die enthaltenen Informationen gemäß Definition 2 ortsbezogen sind, d.h., wenn die Wahrscheinlichkeit, dass ein Benutzer diese Informationen anfordert, von dessen Aufenthaltsort abhängt.*

Ein ortsbezogener Teilraum, nachfolgend kurz *Teilraum* genannt, kann dynamisch bezüglich der darin enthaltenen Informationen sein: Es können neue Informationen hinzukommen und solche wegfallen, die nicht mehr angefordert werden. Die Informationen eines Teilraums, deren Zugriffe mit Hilfe von Protokolldateien an die Infostation übermittelt wurden, werden durch die Knoten des Informationsgraphen repräsentiert.

Die Qualität eines Vorabübertragungsverfahrens hängt offensichtlich davon ab, wie die Zugriffe auf die Informationen des Teilraums verteilt sind. Eine Konzentration der Zugriffe liegt dann vor, wenn diese Verteilung ungleich ist, d.h. ein großer Teil der Zugriffe bezieht sich auf einen geringen Teil der Informationen. Konzentrieren sich nun sehr viele Anforderungen auf einen kleinen Teil der Informationen, können durch deren Übertragung viele Anfragen aus dem lokalen Cache beantwortet werden. Man spricht in diesem Fall von der *Fokussierung der Zugriffe*, auch *Disparität* oder *Ungleichheit* genannt. Zur Messung von Konzentration werden zwei Arten unterschieden: die absolute und die relative Konzentration. Eine *hohe absolute Konzentration* liegt dann vor, wenn sich die Zugriffe nur auf eine geringe (absolute) Anzahl von Informationen verteilen und die restlichen Informationen nicht angefordert werden. Sie wird vor allem in Bereichen verwendet, in denen festgestellt werden soll, ob es Marktführer gibt, die beispielsweise den Umsatz unter sich aufteilen. Verteilen sich die Zugriffe jedoch auf sehr viele Informationen, wie das bei einem Teilraum typischerweise

der Fall ist, wird der Herfindahlindex H als Maß für die absolute Konzentration eher niedrig ausfallen, da dieser die Summe der quadrierten prozentualen Anteile der Zugriffe auf jede einzelne Information darstellt. (siehe Anhang A.1). Aus diesem Grund eignet sich die relative Konzentration besser zur Messung der Fokussierung von Zugriffen auf Informationen. Eine *hohe relative Konzentration* liegt vor, wenn sich ein großer Teil der Zugriffe auf einen kleinen prozentualen Anteil der Informationen konzentriert (und nicht auf eine kleine Anzahl). Die relative Konzentration wird mit Hilfe der Lorenzkurve beschrieben, welche die Abweichung gegenüber der Gleichverteilung angibt. Der Gini-Koeffizient G ist deren begleitende Maßzahl, wobei gilt: $0 \leq G \leq 1$ (siehe auch Anhang A.1). Je größer der Gini-Koeffizient ist, desto höher ist die relative Konzentration, und desto besser werden erwartungsgemäß die mit diesem Vorabübertragungsverfahren erzielbaren Resultate sein.

Definition 26 (Fokussierung von Informationszugriffen) *Sei $\mathcal{IR}$ ein Teilraum und A ein geographisches Gebiet. Sei weiterhin $f : \mathcal{IR} \rightarrow \mathbb{N}$ eine Häufigkeitsverteilung, die jedem Element des Teilraums eine absolute Häufigkeit zuordnet, mit der es in A referenziert wird. Dann ist die* ***Fokussierung der Informationszugriffe*** *in $\mathcal{IR}$ der Gini-Koeffizient von f.*

5.3 Methodik

Um die systematische Evaluierung des für den Zugriff auf Webseiten spezialisierten Vorabübertragungsverfahrens zu ermöglichen, wurden mit Hilfe des in Kapitel 4 beschriebenen Anfragengenerators UCW unterschiedliche Teilräume erzeugt (siehe auch Definition 25), die dem Dienstgebiet einer Infostation zugeordnet wurden. Ein solcher Teilraum stellt den Teil des Webs dar, der für Benutzer ortsbasierter Systeme von Interesse ist, während sie sich im Dienstgebiet der Infostation aufhalten. Nachfolgend werden die für die Leistungsbewertung verwendeten Metriken sowie der Simulationsaufbau beschrieben. Der Abschnitt

schließt mit der Beschreibung der Eigenschaften der simulierten Teilräume.

5.3.1 Metriken

Aus Benutzersicht ist das wichtigste Maß zur Leistungsbewertung eines Vorabübertragungsverfahrens zweifellos die Trefferrate, die aussagt, wie gut der Cache gefüllt ist. Des Weiteren soll eine genügend hohe Trefferrate mit möglichst wenig zusätzlichem Kommunikationsaufwand erzielt werden. Ein Verfahren ist also umso besser, je weniger Webseiten vorab übertragen werden müssen, um eine bestimmte Trefferrate zu erzielen.

Analog zum vorgestellten Vorabübertragungsverfahren, das eine Menge von Webseiten vorab in den lokalen Cache lädt, wird im Bereich der Informationssuche (engl. *information retrieval*) eine Menge von relevanten Dokumenten als Ergebnis einer Suchanfrage geliefert. Zur Bewertung der Qualität einer Suchstrategie werden als Evaluierungsmaße die Präzision (engl. *precision*) und Vollständigkeit (engl. *recall*) der gelieferten Ergebnismenge eingesetzt, die in [6] und [37] beschrieben sind. Sei H die Menge der auf Grundlage einer zu evaluierenden Suchstrategie ermittelten relevanten Dokumente zu einer Anfrage und L die Menge aller bezüglich dieser Anfrage relevanten Dokumente. Dann wird die Präzision berechnet als $\frac{|H \cap L|}{|H|}$ und die Vollständigkeit als $\frac{|H \cap L|}{|L|}$.

Übertragen auf die Bewertung der Qualität von Vorabübertragungsverfahren entspricht L der Menge von Webseiten, die von einem Benutzer angefragt werden und H der Menge der gehorteten Webseiten. Die Trefferrate entspricht dann der Vollständigkeit der Ergebnismenge und wird nachfolgend als *allgemeine Trefferrate* bezeichnet. Wie bereits in Abschnitt 3.2 erwähnt, sind für einen Benutzer im Web nicht alle Webseiten gleich wichtig, weshalb sie in Inhalts- und Transitseiten unterteilt werden. Um diese Unterscheidung auch in der Leistungsbewertung zu berücksichtigen, werden zwei zusätzliche Metriken zur Verfügung gestellt. Die mittels Gleichung 5.2 berechnete *Inhaltstrefferrate* wertet alle Inhaltsseiten, die aus dem Cache geladen werden konnten, als erfolgreich gehortet, Transitseiten

werden nicht berücksichtigt. Nun werden mit dieser Metrik sicherlich auch solche Inhaltsseiten als Treffer gerechnet, deren zugehörige Transitseiten im Cache fehlen. Aus diesem Grund wird zusätzlich die gemäß Gleichung 5.3 ermittelte Pfadtrefferrate eingeführt, die eine Inhaltsseite nur dann als Treffer bewertet, wenn auch alle zugehörigen Transitseiten gehortet wurden. Diese beiden Metriken bilden eine obere und untere Schranke für die tatsächlich zu erwartende Inhaltstrefferrate, denn es besteht immerhin die Möglichkeit, dass Benutzer bei Nichtauffinden einer Transitseite im Cache die gewünschte Inhaltsseite schließlich auf einem anderen Pfad finden können.

Sei L eine Protokolldatei, $L_{\text{Inhalt}} \subseteq L$ die Menge der in der Protokolldatei L angeforderten Inhaltsseiten und $L_{\text{Transit}}(l) \subseteq L$ die Menge der Transitseiten, die zu einer angeforderten Inhaltsseite $l \in L_{\text{Inhalt}}$ gehören. Aus Gründen einer einfachen Darstellung wird die im Protokolleintrag l angeforderte Webseite abgekürzt mit $l.\texttt{ID}$ bezeichnet. Sei weiterhin H die Menge aller Seiten im Cache. Dann werden die Trefferraten folgendermaßen definiert:

$$\texttt{AllgemeineTrefferrate} = \frac{|H \cap L|}{|L|} \tag{5.1}$$

$$\texttt{Inhaltstrefferrate} = \frac{|H \cap L_{\texttt{Inhalt}}|}{|L_{\texttt{Inhalt}}|} \tag{5.2}$$

$$\texttt{Pfadtrefferrate} = \frac{|\{l \in L_{\texttt{Inhalt}} | \forall l' \in L_{\text{Transit}}(l) : l'.\texttt{ID} \in H\}|}{|L_{\texttt{Inhalt}}|} \tag{5.3}$$

In den folgenden Auswertungen werden die Inhalts- und die Pfadtrefferraten untersucht, die allgemeine Trefferrate wird in Kapitel 6 bei der Analyse des Energiebedarfs mobiler Endgeräte verwendet.

5.3.2 Simulationsaufbau

Teilräume können sich in der Anzahl der angeforderten Webseiten sowie deren durchschnittlicher Größe unterscheiden. So wurden beispielsweise im elektro-

nischen Touristenführer, der im Rahmen des GUIDE-Projekts der Universität Lancaster [21] entwickelt und in einer Feldstudie evaluiert wurde, insgesamt 500 Informationen angeboten. Basierend hierauf wurden in dieser Arbeit Teilräume mit 500, 1000, 2000, 3000 und 10000 Webseiten erstellt.

Webseiten enthalten längst nicht mehr nur Text, sondern sind mit Bildern, Musikdateien oder Videoclips angereichert. Um die Empfindlichkeit des Verfahrens bezüglich der Größe von Webseiten zu untersuchen, wurde die durchschnittliche Seitengröße nach diesen Gesichtspunkten variiert. Reyes et al. [88] sowie auch Mah [67] ermittelten für Webseiten, die lediglich Text und Bilder enthalten, durchschnittliche Größen zwischen 20 KBytes und 40 KBytes. Für die Leistungsbewertung des Verfahrens im Hinblick auf unterschiedliche durchschnittliche Seitengrößen wird nachfolgend angenommen, dass Musik- und Videodateien mit einer durchschnittlichen Dateigröße von jeweils 1 MByte einen Anteil von zehn bzw. zwanzig Prozent auf einer Webseite haben. Die synthetischen Teilräume enthalten infolgedessen Webseiten mit durchschnittlichen Seitengrößen von 20 KBytes (nur Text und Bilder), 110 KBytes (zehn Prozent Multimediadaten) und 230 KBytes(zwanzig Prozent Multimediadaten). Das vorgestellte Vorabübertragungsverfahren wurde somit für insgesamt fünfzehn unterschiedliche Teilräume evaluiert. Jedem dieser Teilräume wurden 5000 Protokolldateien zugeordnet, von denen jede das Zugriffsverhalten eines Benutzers im zugehörigen Teilraum modelliert.

Zur Evaluierung des Verfahrens wurde aus dieser Menge zufällig eine Teilmenge von 1000 so genannten *Evaluierungs-Protokolldateien* selektiert. Für jeden der 15 zugeordneten Teilräume wurden für alle Evaluierungs-Protokolldateien Auswertungen durchgeführt und die erzielten Trefferraten gemittelt. In den nachfolgenden Abbildungen stellt also jeder Kurvenpunkt den Durchschnitt von 1000 Auswertungen dar. Die restlichen 4000 so genannten *Aktualisierungs-Protokolldateien* wurden zur Aktualisierung des Informationsgraphen verwendet. Zur Analyse des Lernverhaltens des Vorabübertragungsverfahrens wurden für jeden Teilraum zufällig jeweils 150, 300, 500, 1000, 2000, 3000 und 4000

Aktualisierungs-Protokolldateien verwendet.

Um die Auswirkung der Sequenzlänge, also der Anzahl der Benutzeranfragen pro Protokolldatei, auf das Vorabübertragungsverfahren zu analysieren, wurden zwei unterschiedliche Verteilungen für deren Berechnung eingesetzt, die in Abschnitt 4.2.2 beschrieben sind. (1) Um eine homogene Sequenzlänge zu evaluieren, wurden basierend auf den Ergebnissen der Studie zum GUIDE-Projekt [21] durchschnittlich 25 Benutzeranfragen pro Sehenswürdigkeit angenommen. Unter der Annahme, dass ein Benutzer vier solcher Orte während eines Trips besucht, wurde die Anzahl der Anfragen auf durchschnittlich 100 gesetzt und mittels einer Poisson-Verteilung variiert. Für die 15 Teilräume wurden somit 75000 Protokolldateien generiert. (2) Zum Zweiten wurde untersucht, wie das Verfahren auf inhomogene Sequenzlängen reagiert, also sehr viele sehr kurze und wenige sehr lange. Aus diesem Grund wurden zusätzlich Protokolldateien erstellt, deren Sequenzlänge mit Hilfe einer Doppel-Pareto-Verteilung bestimmt wurde. Diese Protokolldateien wurden mit dem Teilraum assoziiert, der 10000 Webseiten mit einer durchschnittlichen Seitengröße von 20 KBytes enthält. Mit dieser Verteilung wurden durchschnittlich 23 Einträge generiert, also ähnlich der durchschnittlichen Anzahl im ersten Fall. Damit die Anzahl der mit beiden Ansätzen insgesamt erstellten Protokolleinträge vergleichbar ist, wurden mit Hilfe der Doppel-Pareto-Verteilung 20000 Protokolldateien erstellt. Von diesen wurden zufällig 15000 als Aktualisierungs-Protokolldateien gewählt, die restlichen 5000 wurden als Evaluierungs-Protokolldateien verwendet.

Zur Evaluierung der Integration von Profilen in das Vorabübertragungsverfahren wurden weitere 15000 Protokolldateien erzeugt. Da ein Benutzer mehreren Nutzungsprofilen zugeordnet werden kann, wurden diese Protokolldateien mit Hilfe unterschiedlicher Profilmixe erstellt und dem Informationsraum zugeordnet, der 10000 Webseiten mit einer durchschnittlichen Seitengröße von 20 KBytes enthält. Die Länge dieser Protokolldateien ist Poisson-verteilt mit einem Durchschnittswert von 100 Anfragen. Insgesamt wurden fünf Profile definiert, denen jeweils ein disjunkter Teil des Informationsraums zugeordnet wurde, der für die Auswahl

einer Webseite beim direkten Seitenaufruf berücksichtigt wird. Die Leistungsbewertung wurde für drei unterschiedliche Profilmixe durchgeführt: Ein Profilmix kann aus einem, zwei oder drei Profilen zusammengestellt sein. Für jeden dieser Fälle wurden 5000 Protokolldateien erstellt.

Die Klassifizierung der Webseiten in Inhalts- und Transitseiten im Verlauf der Analyse der Protokolldatei erfolgt gemäß Definition 11 auf Grundlage der Besuchsdauer einer Seite, wobei der Parameter γ die Sensitivität des Verfahrens steuert. Beim empirischen Vergleich mehrerer Parameterwerte mit den erzeugten Protokolleinträgen wurden für einen Wert von $\gamma = 0{,}75$ gute Resultate erzielt, so dass dieser in den folgenden Evaluierungen zur Klassifikation der Webseiten verwendet wird. Um das von Cooley et al. in [25] und Pierrakos et al. in [83] festgestellte Zugriffsverhalten von Benutzern bezüglich Inhalts- und Transitseiten zu modellieren, wurden Sprungwahrscheinlichkeiten für den direkten Seitenaufruf zwischen zehn und zwanzig Prozent definiert.

Die beiden Exponenten α und β zur Gewichtung der einzelnen Faktoren für die Berechnung der Relevanz pro Byte eines Clusters werden auf Grundlage der in Tabelle 5.1 aufgeführten Wertebereiche variiert. Hierzu werden sämtliche Parameter-Kombinationen ausgewertet, um daraus die optimale zu bestimmen. Der Schwellwert für die minimale Pfadwahrscheinlichkeit wird zwischen 10^{-3} und 10^{-7} variiert.

Schließlich wird das Vorabübertragungsverfahren für unterschiedlich große Caches evaluiert. Unterschiedliche durchschnittliche Seitengrößen führen jedoch zu unterschiedlich großen Caches, weshalb der besseren Übersichtlichkeit wegen in den Abbildungen nicht die absolute Cache-Größe in MByte angegeben ist. Statt dessen wird die relative Cache-Größe verwendet, die das Verhältnis der Cache-Größe zum Anfragevolumen darstellt. Das Vorabübertragungsverfahren wird für relative Cache-Größen zwischen 0,1 und 10 evaluiert. Eine relative Cache-Größe von 10 bedeutet demnach, dass die Cache-Größe dem zehnfachen Anfragevolumen entspricht.

Tabelle 5.1: Parameter für die Evaluation

Parameter	**Wertebereich**
#Webseiten des Teilraums	500; 1000; 2000; 3000; 10000
Durchschnittliche Seitengröße [KByte]	20, 110, 230
#Aktualisierung-Protokolldateien	150; 300; 500; 1000; 2000; 3000; 4000 (; 15000)
#Evaluierungs-Protokolldateien	1000 (; 5000)
α und β zur Berechnung der Relevanz pro Byte	0; 0,25; 0,5; 0,75; 1, 2, 5, 10, 20
Minimales Pfadgewicht	$10^{-7} \ldots 10^{-3}$
Relative Cache-Größe	0,1; 0,25; 0,75; 1;2 ... 10

In Tabelle 5.1 sind alle Parameter übersichtlich zusammengefasst.

5.3.3 Eigenschaften der simulierten Teilräume

Wie in Abschnitt 4.2.2 beschrieben wurde, folgt die Wahrscheinlichkeitsverteilung der Popularität einer Webseite (und damit ihre Zugriffswahrscheinlichkeit) einer Zipf-Verteilung mit Parameter $\alpha = 0{,}8$ für den Exponenten. Wie die hierdurch entstehende Fokussierung der Zugriffe (siehe Definition 26) von der Anzahl der Webseiten eines Teilraums abhängt, ist in Tabelle 5.2 zusammengefasst: Mit wachsender Anzahl von Webseiten steigt auch die Fokussierung leicht an. Abbildung 5.1 zeigt die entsprechenden Lorenzkurven.

Tabelle 5.2: Fokussierung von Zugriffen in Abhängigkeit von der Größe der Teilräume

Anzahl Webseiten	**Fokussierung**
500	0.554
1000	0.572
2000	0.587
3000	0.595
10000	0.612

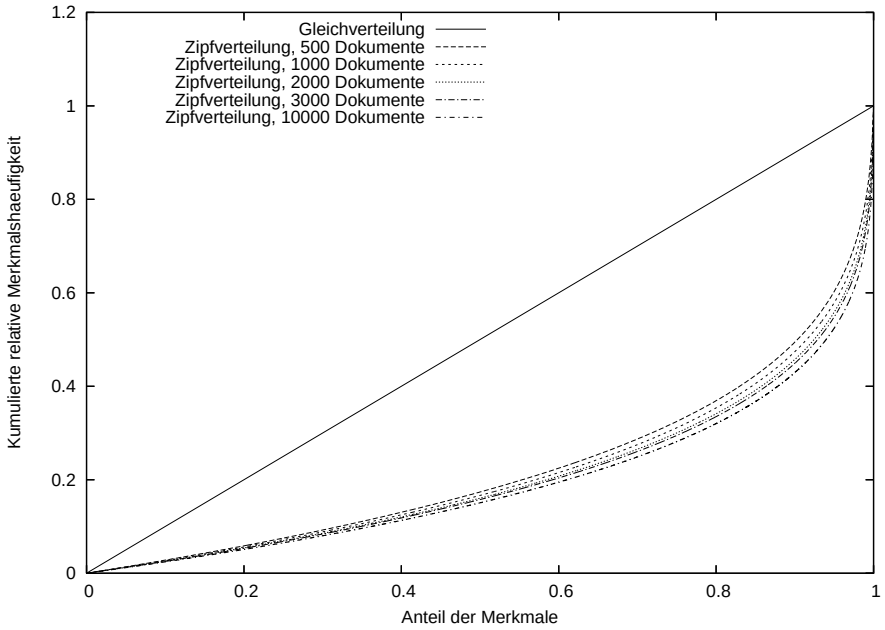


Abbildung 5.1: Lorenzkurven für Informationsräume mit unterschiedlicher Anzahl von Webseiten

5.4 Diskussion der Ergebnisse

In den folgenden Abschnitten werden einige repräsentative Ergebnisse der umfangreichen Auswertungen diskutiert. Die charakteristischen Merkmale der in diesem Abschnitt verglichenen Vorabübertragungsverfahren sind in Tabelle 5.3 zusammengefasst.

Bei den Abbildungen in den folgenden Abschnitten zeigt jeweils die x-Achse die relative Cache-Größe und die y-Achse die Inhalts- bzw. Pfadtrefferraten.

5.4.1 Vergleich unterschiedlicher Informationsräume

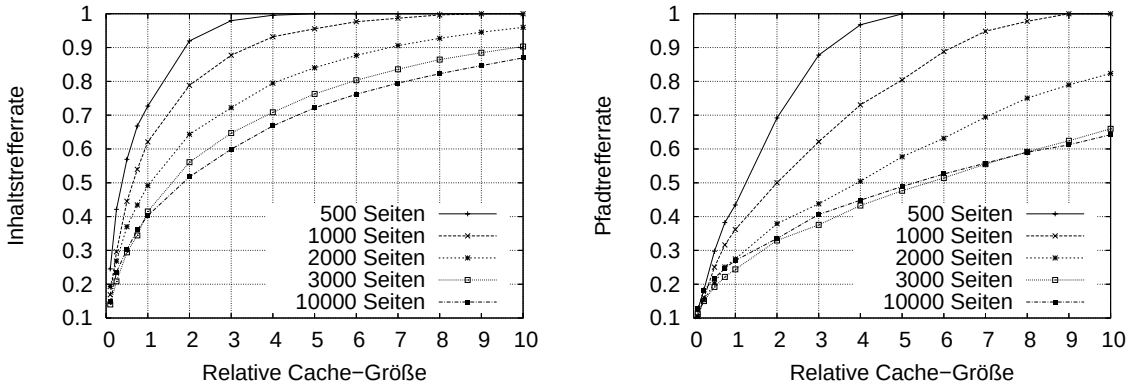
In den folgenden beiden Auswertungen werden die Inhalts- und Pfadtrefferraten diskutiert, die mit dem clusterbasierten Verfahren (CBH) für die in Ab-

Tabelle 5.3: Merkmale der zu vergleichenden Vorabübertragungsverfahren

Kürzel	**Beschreibung**	**Sortierkriterium**
CBH	Selektion mit Clusterbildung (engl. ***C**luster-**B**ased **H**oarding*)	Relevanz pro Byte eines Clusters, abhängig von Größe, Pfad- und Inhaltswahrscheinlichkeit
BPS	Begrenzte Pfadsuche, keine Clusterbildung	Relevanz pro Byte einer Webseite, abhängig von Größe und Pfadwahrscheinlichkeit
BFS	Modifizierte Breitensuche: Auswahl der nächsten Kante abhängig von der Kantenwahrscheinlichkeit, keine Clusterbildung	Reihenfolge der Traversierung
DFS	Modifizierte Tiefensuche: Auswahl der nächsten Kante abhängig von der Kantenwahrscheinlichkeit, keine Clusterbildung	Reihenfolge der Traversierung
LFU	Infostations-basierte Vorabübertragung (Kubach [55]) für unstrukturierte Informationen, keine Clusterbildung	primär Zugriffswahrscheinlichkeit, sekundär Größe
LRU	Vorabübertragung in Dateisystemen (SEER), keine Clusterbildung	Gemäß Ersetzungsstrategie least recently used (LRU)

schnitt 5.3.2 beschriebenen Teilräume erzielt wurden, die sich wiederum in der Anzahl der Webseiten und deren durchschnittlicher Größe unterscheiden. Zum Aufbau des Informationsgraphen wurden 4000 Protokolldateien mit jeweils durchschnittlich 100 Einträgen (Poisson-verteilt) verwendet.

Zunächst wird untersucht, wie sich die Anzahl der Webseiten eines Teilraums bei konstanter durchschnittlicher Größe der Webseiten auf die unterschiedlichen Trefferraten auswirkt. Abbildung 5.2(a) zeigt die Inhaltstrefferraten für Teilräume mit 500, 1000, 2000, 3000 und 10000 Webseiten mit einer durchschnittlichen Seitengröße von 20 KBytes. In Abbildung 5.2(b) ist entsprechend die Pfadtrefferrate dargestellt. Vergleiche mit den durchschnittlichen Größen 110



(a) Inhaltstrefferrate (b) Pfadtrefferrate

Abbildung 5.2: Trefferraten für Teilräume mit unterschiedlicher Anzahl Webseiten (durchschnittliche Seitengröße von 20 KBytes)

und 230 KBytes zeigen ein ähnliches Verhalten, weshalb hier nur die Ergebnisse für 20 KBytes diskutiert werden. Obwohl erwartungsgemäß die Trefferrate mit steigender Anzahl von Webseiten sinkt, wird für den 10000 Seiten umfassenden Teilraum, von dem durchschnittlich 100 Webseiten pro Protokolldatei angefordert werden, dennoch eine Inhaltstrefferrate von 80,9% erzielt. Betrachten wir nun den Anteil der Webseiten eines Teilraums, der gehortet werden muss, um eine bestimmte Inhaltstrefferrate zu erzielen. Je niedriger dieser Teil ist, desto weniger Seiten müssen übertragen werden. In Abbildung 5.3 sind diese Anteile beispielhaft für CBH und LFU (siehe Tabelle 5.3) für eine anvisierte Trefferrate von 50% dargestellt. Bei CBH sinkt der Anteil zu hortender Webseiten an der Gesamtgröße des Teilraums mit steigender Anzahl an Webseiten. Dies ist beim 10000 Webseiten Teilraum am deutlichsten zu erkennen. Hierdurch erzielt CBH wesentlich bessere Inhaltstrefferraten für große Informationsräume als andere Verfahren wie beispielsweise LFU, die diese Eigenschaft nicht aufweisen. Bei der Pfadtrefferrate tritt diese Charakteristik noch deutlicher zu Tage, denn für die Teilräume mit 10000 und 3000 Webseiten sind die mit CBH erzielten Pfadtrefferraten nahezu gleich (siehe Abbildung 5.2(b)).

Die nächste Auswertung bezieht sich auf die Auswirkung unterschiedlicher

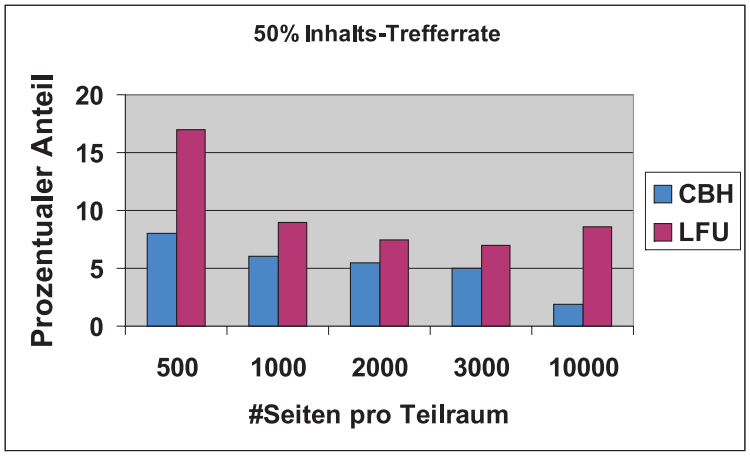
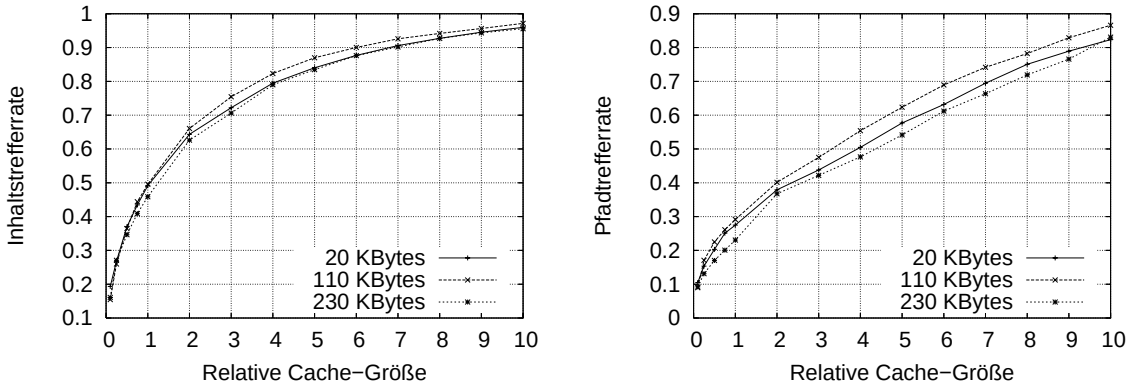


Abbildung 5.3: Anteil zu hortender Webseiten pro Teilraum für 50% Inhaltstrefferrate

durchschnittlicher Webseiten-Größen bei gleich bleibender Teilraumgröße auf die Inhalts- und Pfadtrefferraten. Abbildung 5.4(a) zeigt die für durchschnittliche Seitengrößen von 20, 110 und 230 KBytes erzielten Inhaltstrefferraten für einen Teilraum mit 2000 Webseiten. In Abbildung 5.4(b) sind entsprechend die Pfadtrefferraten dargestellt. Die erzielten Ergebnisse zeigen, dass die Inhaltstrefferrate unabhängig von der durchschnittlichen Webseiten-Größe ist, während die Pfadtrefferrate etwas empfindlicher auf unterschiedliche Durchschnittsgrößen reagiert. Die Ergebnisse der Auswertungen für die Teilräume mit 500, 1000, 3000 und 10000 Seiten zeigen die gleiche Charakteristik, die aus der Einbeziehung der Seitengröße in die Berechnung der Relevanzwerte von Clustern resultiert.

In den nachfolgenden Auswertungen wird nur noch die Inhaltstrefferrate diskutiert. Wie bereits in Abschnitt 5.3.1 erwähnt, werden mit dieser Metrik alle in einer Protokolldatei angeforderten Inhaltsseiten, die aus dem Cache geladen werden konnten, als Treffer gezählt, auch wenn nicht alle zugehörigen Transitseiten gehortet wurden. Die Inhaltstrefferrate bildet somit eine obere Schranke für die tatsächlich zu erwartende Inhaltstrefferrate.



(a) Inhaltstrefferrate (b) Pfadtrefferrate

Abbildung 5.4: Trefferraten für Teilräume mit 2000 Webseiten unterschiedlicher durchschnittlicher Größe

5.4.2 Einfluss der Sequenzlänge

In der nachfolgenden Auswertung wird die Leistung von CBH untersucht, wenn die Anzahl der Einträge in einer Protokolldatei mittels unterschiedlicher Modelle berechnet wurde. Wie bereits in Abschnitt 5.3.2 beschrieben, wurde diese Anzahl zum einen mittels einer Doppel-Pareto-Verteilung ermittelt. Die insgesamt 20000 Protokolldateien hatten durchschnittlich dreiundzwanzig Einträge, wobei sich der Durchschnittswert durch die gewählten Parameter der Verteilungsfunktion ergab. Zum Zweiten wurde die Sequenzlänge nach dem Beispiel des elektronischen Touristenführers im GUIDE-Projekt [21] gemäß einer Poisson-Verteilung mit einem Durchschnittswert von 100 Einträgen berechnet. Nach diesem Modell wurden dann 5000 Protokolldateien generiert, damit die Gesamtzahl an Protokolleinträgen in beiden Fällen vergleichbar war.

Zur Graph-Aktualisierung wurden 4000 bzw. 15000 der Protokolldateien verwendet. Mit den restlichen 1000 bzw. 5000 Protokolldateien wurde CBH evaluiert.

Abbildung 5.5 stellt die Inhaltstrefferraten dar, die mit CBH im Teilraum mit 10000 Seiten und durchschnittlicher Größe von 20 KBytes erzielt werden, wenn

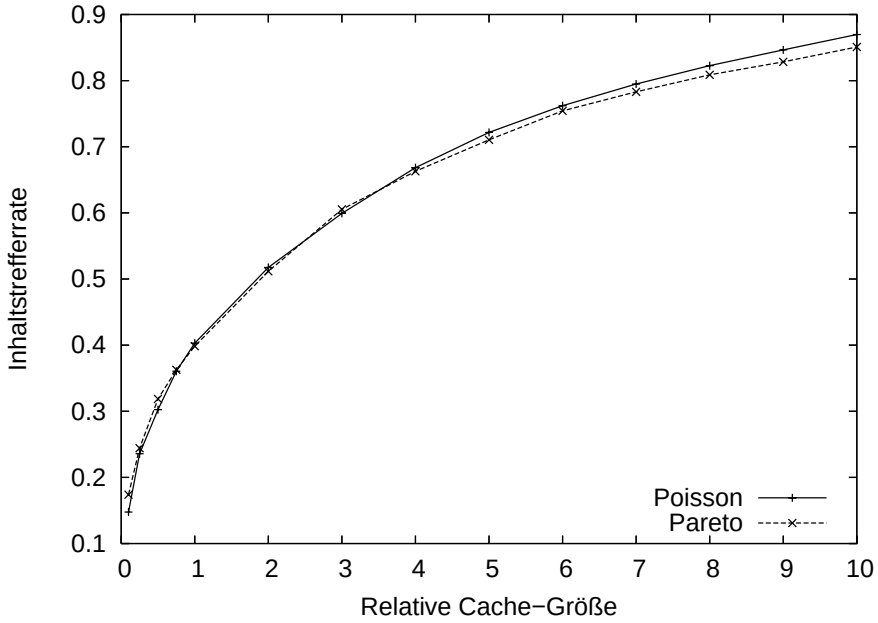


Abbildung 5.5: Einfluss der Sequenzlänge auf die Inhaltstrefferrate für den Teilraum mit 10000 Seiten und durchschnittlicher Seitengröße von 20 KBytes

die Anzahl der Protokolleinträge entweder mit einer Poisson- oder einer Pareto-Verteilung erzeugt wurde. Die Anzahl der Einträge hat so gut wie keine Auswirkungen auf die Inhaltstrefferrate, denn eine Protokolldatei wird zur Analyse in Sitzungen unterteilt, die wiederum auf Grundlage der Besuchsdauer einer Seite gebildet werden. Erst bei größeren Caches werden mit der Poisson-Verteilung um $2,5\%$ höhere Trefferraten erzielt.

5.4.3 Einfluss der maximalen Besuchsdauer einer Webseite zur Bestimmung einer Sitzung

Abbildung 5.6 stellt die Inhaltstrefferraten dar, die mit CBH erzielt werden, wenn die maximale Besuchsdauer einer Webseite für die Zuordnung zu einer Sitzung von 15 Minuten auf 60 Minuten erhöht wird. Für diese Auswertung wur-

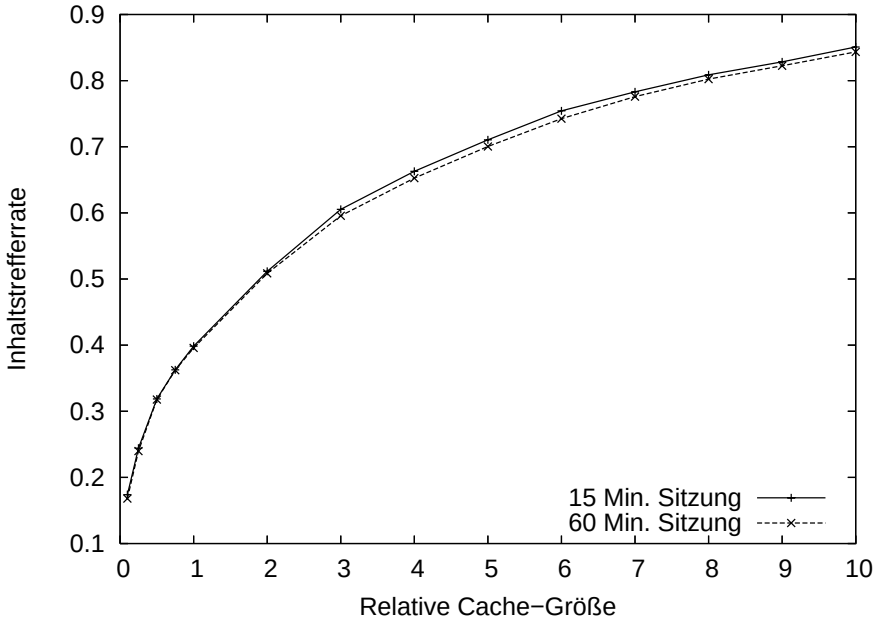


Abbildung 5.6: Einfluss der Dauer einer Sitzung auf die Inhaltstrefferrate für den Teilraum mit 10000 Seiten und durchschnittlicher Seitengröße von 20 KBytes

den die Protokolldateien verwendet, deren Anzahl an Einträgen Pareto-verteilt berechnet wurde, denn die aus diesen Protokolldateien abgeleiteten Sitzungen unterscheiden sich deutlicher von einander als diejenigen, deren Anzahl an Einträgen mittels einer Poisson-Verteilung erzeugt wurde. Die maximale Besuchsdauer wird zur Aktualisierung des Informationsgraphen benötigt und bestimmt, wie viele und welche Webseiten-Anforderungen einer Sitzung zugeordnet werden. Die Kurven in Abbildung 5.6 sind entsprechend mit „15 Min. Sitzung" bzw. „60 Min. Sitzung" gekennzeichnet. Die Dauer einer Sitzung hat offensichtlich einen vernachlässigbar kleinen Einfluss auf die erzielbaren Trefferraten, denn die Trefferraten unterscheiden sich nur um zirka ein Prozent. Mit kürzeren Sitzungen erzielt das Verfahren leicht bessere Inhaltstrefferraten.

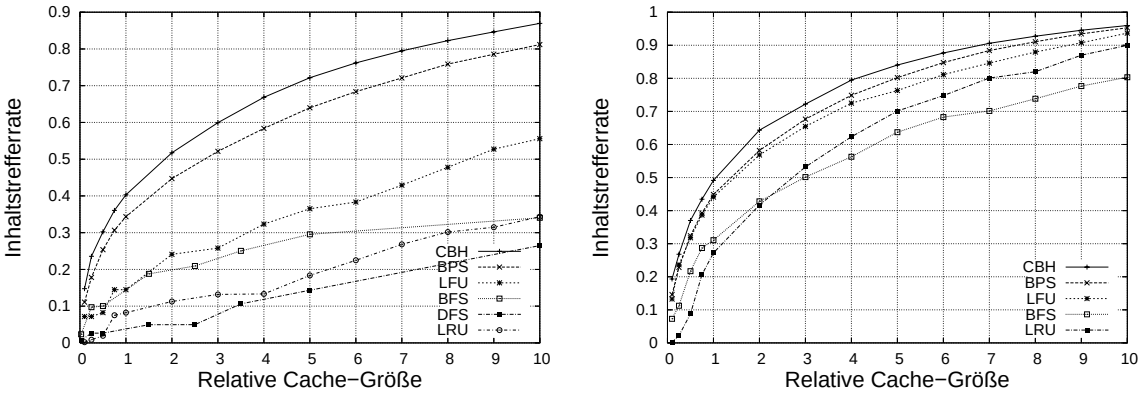
5.4.4 Vergleich unterschiedlicher Vorabübertragungsverfahren

In diesem Abschnitt werden zum einen die in dieser Arbeit vorgestellten Auswahlverfahren einander gegenüber gestellt. Dies sind insbesondere die Selektion mit Clusterbildung (CBH), die Selektion mittels begrenzter Pfadsuche (BPS) sowie die modifizierte Tiefen- und Breitensuche (DFS,BFS). Zum andern wird CBH mit zwei Vorabübertragungsverfahren aus den verwandten Arbeiten verglichen, deren Selektionsmechanismen auf im Caching verwendeten Ersetzungsstrategien (LFU, LRU) beruhen.

In Tabelle 5.3 sind die zum Vergleich herangezogenen Verfahren zusammengefasst. Die beiden in Caching-Verfahren angewendeten Ersetzungsstrategien LFU und LRU werden auch in anderen Vorabübertragungsverfahren eingesetzt. So verwenden Kuenning et al. in SEER [57] die LRU-Technik, um die Dateien für die Vorabübertragung auszuwählen. Das Vorabübertragungsverfahren von Kubach und Rothermel [55] verwendet die LFU-Ersetzungsstrategie, berücksichtigt jedoch nicht die Größe der Webseiten, und wurde deshalb zu Evaluationszwecken leicht modifiziert: Bei Webseiten mit gleicher Zugriffswahrscheinlichkeit wird die Größe als Entscheidungshilfe benutzt, welche dieser Seiten zuerst geladen werden soll.

Die beiden Abbildungen 5.7(a) und 5.7(b) zeigen die Inhaltstrefferraten für die Informationsräume mit 10000 bzw. 2000 Webseiten, deren Webseiten jeweils eine Durchschnittsgröße von 20 KBytes haben. CBH übertrifft die anderen Verfahren in allen Fällen. Der besseren Übersichtlichkeit halber wird das mit DFS erzielte Resultat nur in Abbildung 5.7(a) dargestellt, da DFS in allen Teilräumen wesentlich schlechtere Trefferraten als BFS erzielt.

Verglichen mit BPS, das in allen Teilräumen das zweitbeste Ergebnis liefert, erzielt CBH ab einer relativen Cache-Größe von 2 Verbesserungen um durchweg zirka acht Prozent. Es sei nochmals darauf hingewiesen, dass bei der Berechnung der Relevanz pro Byte bei CBH im Gegensatz zu BPS die Inhaltswahrscheinlichkeit einer Webseite mit berücksichtigt wird.



(a) 10000 Webseiten (b) 2000 Webseiten

Abbildung 5.7: Vergleich unterschiedlicher Vorabübertragungsverfahren für Teilräume mit 2000 und 10000 Seiten (durchschnittliche Größe von 20 KBytes)

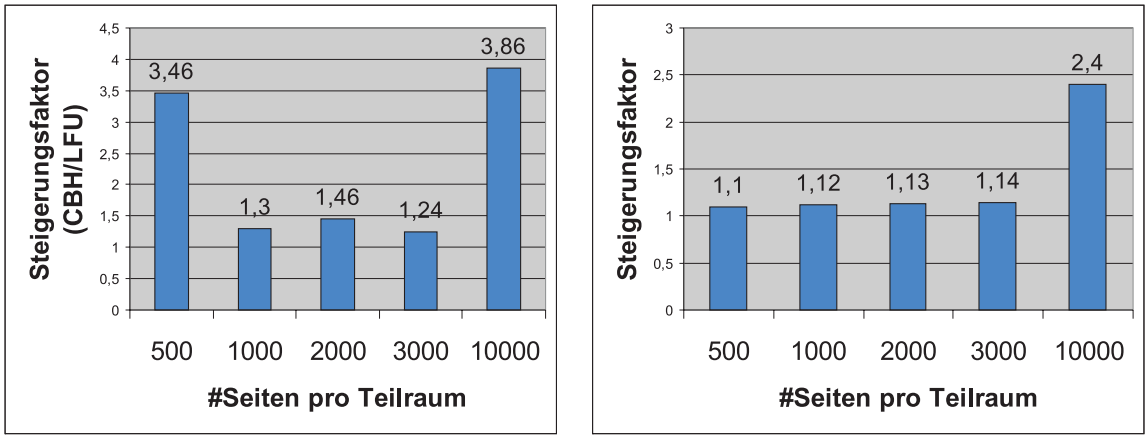
Man kann deutlich erkennen, dass mit CBH, BPS und LFU durchweg bessere Inhaltstrefferraten erzielt werden als mit dem LRU-basierten Verfahren. Letzteres nutzt zeitliche Beziehungen zwischen Dateizugriffen zur Berechnung der semantischen Distanz aus. Es trifft die Entscheidung, welche Dateien zu horten sind, für jeden Benutzer auf Grundlage des Alters des Dateizugriffs. Im Gegensatz hierzu wird bei den zuerst genannten Verfahren diese Entscheidung durch die Ausnützung von Wissen über das kollektive Zugriffsverhalten in einem Gebiet getroffen, was in ortsbasierten Systemen deutlich bessere Trefferraten erzielen lässt.

Weiterhin übertreffen CBH und BPS das von Kubach entwickelte LFU-basierte Verfahren, da mit den aus der Navigation im Web entstehenden Zugriffsmustern Beziehungen zwischen den Webseiten ausgenutzt werden können. Im Teilraum mit 2000 Seiten ist CBH um bis zu Faktor 1,46 besser als das LFU-basierte Verfahren, das die drittbesten Trefferraten liefert. Im Teilraum mit 10000 Seiten beträgt dieser Faktor sogar 3,86 gegenüber LFU, das wiederum drittbestes Verfahren ist.

Diese Auswertungen wurden ebenso für die restlichen Teilräume durchgeführt. Die erzielten Ergebnisse zeigen, dass der Steigerungsfaktor, um den CBH das jeweils drittplatzierte Verfahren (LFU) übertrifft, ab einer relativen Cache-Größe von 2 mit zunehmender Teilraumgröße steigt. Für relative Cache-Größen kleiner als 2 schwankt dieser Faktor beträchtlich zwischen 1,24 für den Teilraum mit 3000 Seiten und 3,86 für 10000 Seiten. Die Steigerungsfaktoren für alle Teilräume sind in den Abbildungen 5.8(b) und 5.8(a) für relative Cache-Größen kleiner als 2 bzw. größer oder gleich 2 grafisch dargestellt. Die unregelmäßigen Steigerungsfaktoren für relative Cache-Größen kleiner als 2 rühren daher, dass bei LFU als Primär-Sortierkriterium die Zugriffswahrscheinlichkeit verwendet wird. Dies kann dazu führen, dass die Trefferrate bei der nächst größeren relativen Cache-Größe gleich bleibt, wie beispielsweise in Abbildung 5.7(a) bei den Übergängen von 0,1 zu 0,25 und von 0,75 zu 1 zu sehen ist. Bei CBH hingegen ist das Sortierkriterium die Relevanz pro Byte, wodurch die Trefferrate beständig ansteigt. Wie Abbildung 5.8(a) zu entnehmen ist, führt dies bei kleinen Caches zu schwankenden Steigerungsfaktoren. Ab einer relativen Cache-Größe von 2 steigt die Cache-Größe ständig um zwei MBytes, so dass dieser Aspekt nicht mehr so sehr ins Gewicht fällt und der Steigerungsfaktor nur noch zunimmt.

Ähnliche Steigerungsfaktoren ergeben sich durch den Vergleich von CBH und BFS, das nur für sehr kleine Caches etwas besser abschneidet als LFU. Dieses Ergebnis ist besonders interessant, da es indirekt die Bewertung der Qualität des Web-Navigationsmodells erlaubt. In [17] wurden BFS und LFU miteinander verglichen, wobei jedoch keine synthetischen Daten zugrunde lagen, sondern die Proxy-Protokolldatei eines anonymen Unternehmens. Der Teilraum fasste ungefähr 10000 Webseiten. Die mit BFS und LFU erzielten Trefferraten sind ähnlich denen aus dieser Evaluierung.

LRU erzielt zwar bessere Trefferraten als DFS, schneidet jedoch im 10000 Seiten fassenden Teilraum schlechter ab als BFS. In kleineren Teilräumen ist LRU nur für größere Caches besser als BFS. Verglichen mit LFU erzielt LRU durchweg schlechtere Trefferraten. Der Grund dafür ist, dass für ortsbasierte Anwendungen



(a) Relative Cache-Größe kleiner als 2

(b) Relative Cache-Größe größer oder gleich 2

Abbildung 5.8: Steigerungsfaktoren von CBH gegenüber LFU für die Inhaltstrefferraten

diejenigen Verfahren besser geeignet sind, welche die Auswahl pro Gebiet treffen (LFU) und nicht pro Benutzer (LRU).

5.4.5 Einbeziehen von Profilen

Abbildung 5.9 stellt das Verhalten von CBH mit integrierten Profilen dar, wenn Benutzer mit unterschiedlichen Profilmixen Webseiten eines Teilraums anfordern. In der Grafik sind die Kurven für unterschiedliche Profilmixe gekennzeichnet mit „1 Profil", „Max. 2 Profile" und „Max. 3 Profile". Die Inhaltstrefferraten werden mit steigender Anzahl von Profilen in einem Profilmix schlechter. So sinkt die Trefferrate beispielsweise um bis zu zehn Prozent, wenn statt einem Profil in einem Profilmix maximal zwei Profile enthalten sein dürfen. Lässt man maximal drei Profile zu, sinkt die Trefferrate nur noch um bis zu drei Prozent gegenüber maximal zwei erlaubten Profilen. Der Grund hierfür ist, dass der Infostation nur der Profilmix bekannt ist, jedoch nicht, auf welches der darin enthaltenen Profile sich ein Protokolleintrag bezieht. Somit können die zugeordneten Informationsgraphen nicht optimal aktualisiert werden. Im Gegensatz hierzu hat dies bei nur

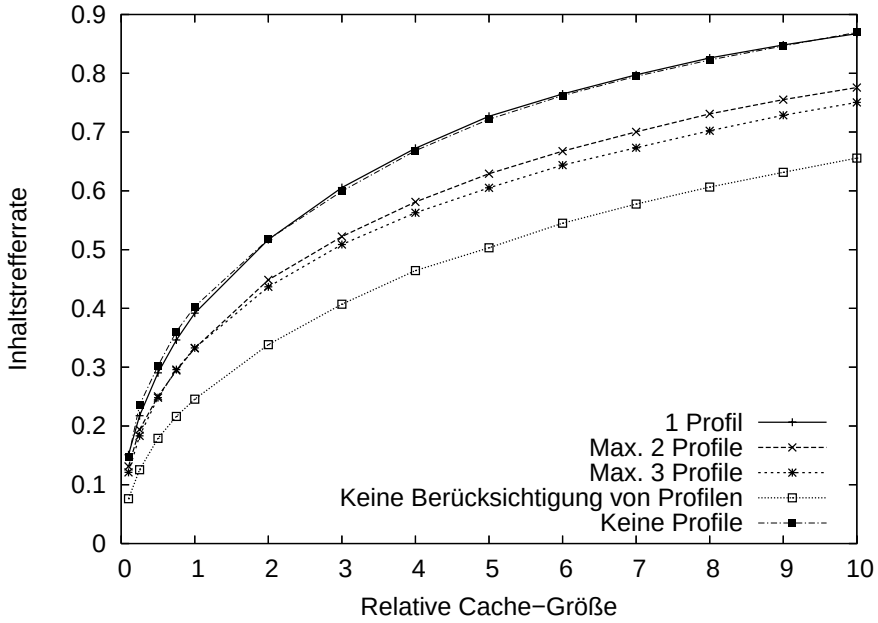


Abbildung 5.9: Inhaltstrefferraten für unterschiedliche Profilmixe

einem Nutzungsprofil im Profilmix keine Auswirkungen.

Die mit „Keine Berücksichtigung von Profilen" bezeichnete Kurve zeigt zum Vergleich die Inhaltstrefferraten für den Fall, dass die Webseiten zwar mit unterschiedlichen Profilmixen angefordert wurden, die Profile jedoch nicht in das Vorabübertragungsverfahren mit einbezogen wurden. Der Einfachheit halber wird nur eine solche Kurve gezeigt, denn die Ergebnisse für die Auswertungen für unterschiedliche Maximalwerte erlaubter Profile sind sich sehr ähnlich. Die erzielten Trefferraten ohne Berücksichtigung der Profile sind im Vergleich zu einem erlaubten Profil pro Mix um mehr als zwanzig Prozent schlechter. Dieses Verhalten basiert auf dem inhomogenen Zugriffsverhalten, das durch die unterschiedlichen Nutzungsprofile entsteht, das jedoch durch deren Berücksichtigung im Verfahren ausgeglichen werden kann.

Die letzte Kurve in Abbildung 5.9, die mit „Keine Profile" beschriftet ist, zeigt zum Vergleich die Inhaltstrefferraten für ein homogenes Zugriffsverhalten ohne

Profile, das in den vorigen Auswertungen diskutiert wurde. Sie ist fast identisch mit der durch „1 Profil" beschriebenen Kurve. Eigentlich könnte man erwarten, dass die Ergebnisse deutlich besser werden, wenn nur ein Teil des Teilraums für den direkten Seitenaufruf verwendet wird und sich somit die Größe des Teilraums entsprechend verringern sollte. Durch die Hyperlink-Struktur des Webs und damit des Teilraums wird jedoch nicht nur die einem Profil zugeordnete Teilmenge von Webseiten für den direkten Seitenaufruf angefordert, sondern ein wesentlich größerer Teil. Dadurch unterscheidet sich das Zugriffsverhalten mit Profilen nicht mehr so sehr von dem ohne Profile.

5.4.6 Lernverhalten von CBH

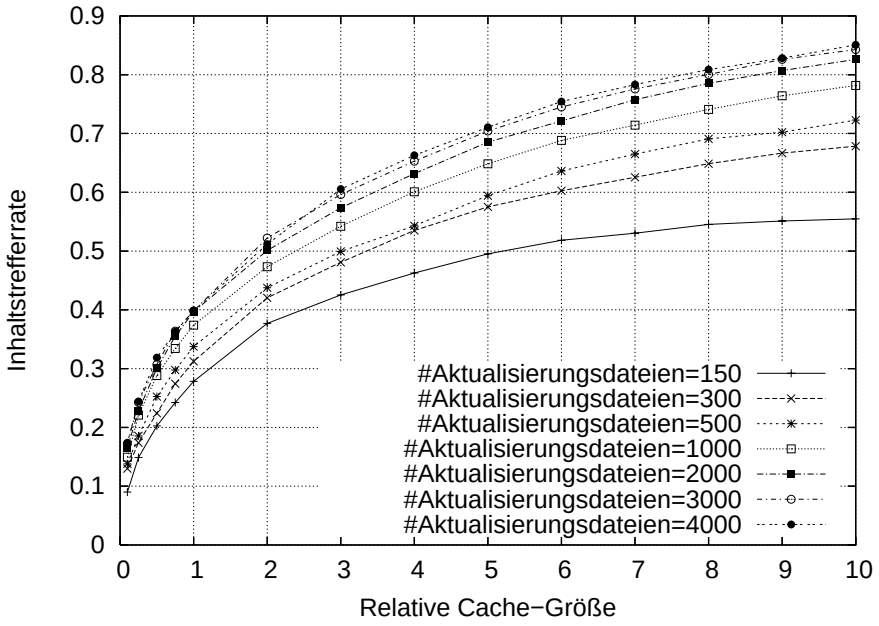


Abbildung 5.10: Inhaltstrefferraten in Abhängigkeit von der Zahl der Aktualisierungs-Protokolldateien

Nachfolgend wird angenommen, dass der Informationsgraph einer Infostation initial nur aus dem Wurzelknoten und dem zentralen Knoten besteht. In der näch-

sten Auswertung wird untersucht, wie schnell eine Infostation das Zugriffsverhalten der Benutzer lernt, die sich in ihrem Dienstgebiet aufhalten. Hierfür wird eine zwischen 150 und 4000 variierende Anzahl von Aktualisierungs-Protokolldateien verwendet (siehe Tabelle 5.1), um den Informationsgraphen zu aktualisieren. Abbildung 5.10 zeigt die erzielten Inhaltstrefferraten für den Teilraum mit 10000 Webseiten mit durchschnittlicher Größe von 20 KBytes.

Die mit 150 und 4000 Aktualisierungs-Protokolldateien erzielten Trefferraten weichen um bis zu dreißig Prozent voneinander ab. Trotzdem wird bereits mit 150 Protokolldateien eine Inhaltstrefferrate von über fünfzig Prozent erzielt. Ab 2000 Aktualisierungs-Protokolldateien unterscheiden sich die Resultate nur noch marginal.

5.4.7 Laufzeitanalyse für die Erstellung der Vorabübertragungsliste

Abbildung 5.11 zeigt die gemittelten Laufzeiten für die Erstellung der Vorabübertragungsliste in Abhängigkeit von der Anzahl gebildeter Cluster. Für das iterative Sortieren wird als Datenstruktur die Prioritätswarteschlange verwendet. Der zugrunde liegende Teilraum enthält 10000 Webseiten mit einer durchschnittlichen Seitengröße von 20 KBytes. Das Sortierverfahren an sich beeinflusst die Trefferrate nicht, weshalb letztere hier nicht angezeigt wird.

Der schlechteste Fall einer mehr als quadratischen Zeitkomplexität ist nicht eingetreten. Wie aus Tabelle 5.4 ersichtlich wird, führt eine Verdoppelung der Anzahl von Clustern, die in den Zeilen 2 und 3, sowie 4 und 5 zu erkennen ist, nur zu einer logarithmisch ansteigenden Laufzeit. Für die größte relative Cache-Größe von zehn werden ohne Einbeziehung von Profilen ca. 14600 Cluster gebildet. Der hierfür notwendige Sortier- und Einfügeprozess benötigt nur wenig mehr als eine Sekunde.

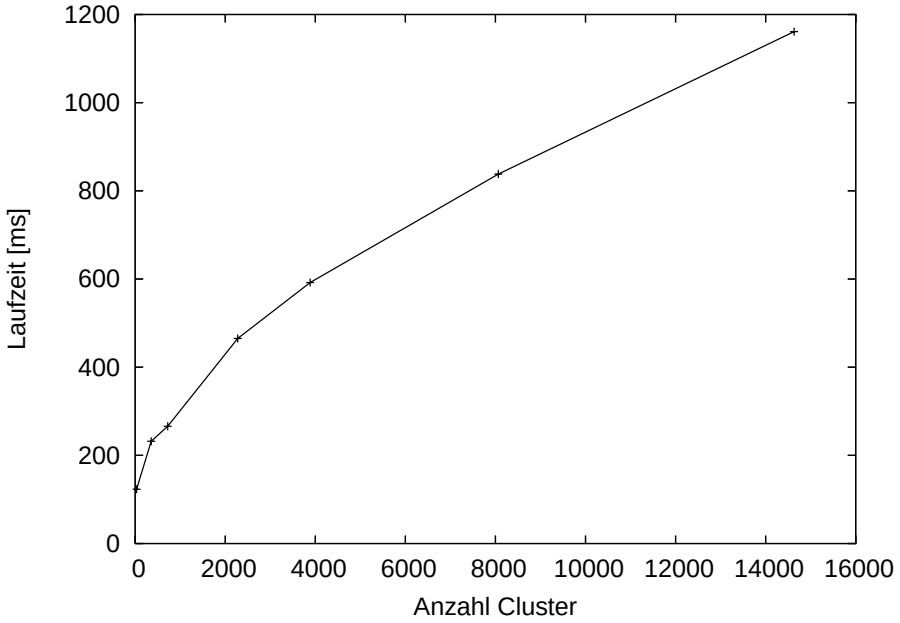


Abbildung 5.11: Laufzeiten für die Erstellung von Vorabübertragungslisten

5.4.8 Optimale Parameterbelegung

Nachfolgend wird der Einfluss des minimalen Pfadgewichts, sowie der Faktoren zur Berechnung der Relevanz pro Byte eines Clusters auf die Leistung des Verfahrens untersucht.

Einfluss des minimalen Pfadgewichts Wie bereits in Kapitel 3.8.1 angesprochen wurde, ist das minimale Pfadgewicht bei der begrenzten Pfadsuche, auf der CBH basiert, ein wichtiger Faktor zur Begrenzung des Aufwands. Dieser im Intervall [0,1] liegende Parameter bestimmt das minimale Pfadgewicht, das ein vom Wurzelknoten zum nächsten zu besuchenden Knoten verlaufender Pfad haben muss, damit der rekursive Abstieg fortgesetzt wird. Je größer das minimale Pfadgewicht ist, desto geringer ist zwar der zeitliche Aufwand, dafür werden jedoch weniger Cluster gebildet, was eventuell zu sinkenden Trefferraten führt. In den nächsten beiden Auswertungen wird nun dessen Einfluss sowohl auf die Inhalts-

Anzahl der Cluster	Gemittelte Laufzeit (in Millisekunden)
31	123
356	232
723	266
2277	465
3884	592
8064	838
14632	1161

Tabelle 5.4: Gemittelte Laufzeiten des iterativen Sortierens in Abhängigkeit von der Zahl der Cluster

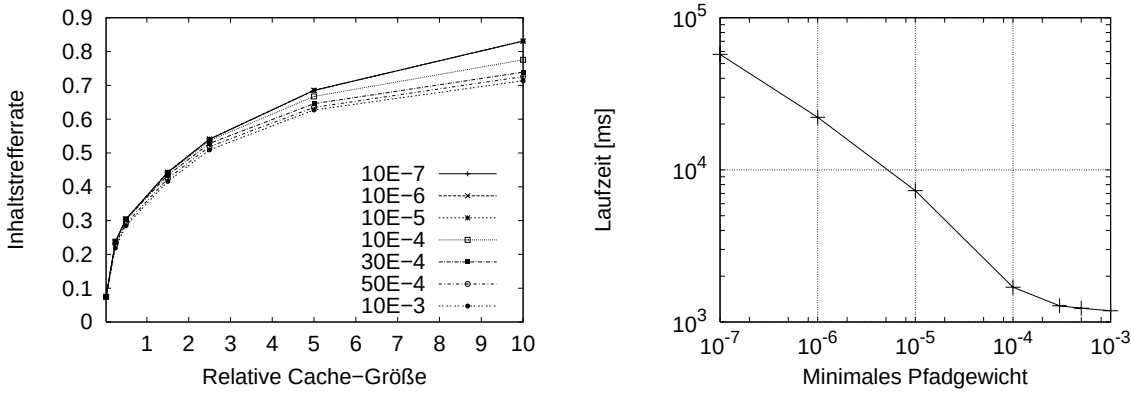
trefferraten als auch auf das Laufzeitverhalten des Verfahrens untersucht. Der zugrunde liegende Informationsraum enthält 10000 Webseiten mit einer durchschnittlichen Seitengröße von 20 KBytes. Wie Abbildung 5.12(a) zu entnehmen ist, unterscheiden sich die Trefferraten für Parameter zwischen 10^{-7} und 10^{-5} kaum, während sie für 10^{-4} bzw. 10^{-3} bereits um 5,5% bzw. 12% sinken.

Die Laufzeit des Verfahrens für die unterschiedlichen Werte für das minimale Pfadgewicht ist in Abbildung 5.12(b) dargestellt, wobei zu beachten ist, dass die Koordinatenachsen eine logarithmische Skala haben. Die Laufzeit verringert sich erheblich von knapp sechzig Sekunden für ein minimales Pfadgewicht von 10^{-7} auf ca. sieben Sekunden für einen Wert von 10^{-5}, wobei die Trefferraten nahezu gleich sind.

Einfluss der Faktoren zur Berechnung der Relevanz pro Byte eines Clusters
Die Relevanz pro Byte wird mittels der Pfad- und Inhaltswahrscheinlichkeit eines Clusters, sowie dessen Größe berechnet.

$$R(C) = \frac{(C.p(\texttt{Pfad}))^{\alpha} \cdot (C.p(\texttt{Inhalt}))^{\beta}}{C.\texttt{größe}}$$

Die beiden Exponenten α und β dienen zur Gewichtung der einzelnen Faktoren. Je kleiner ein Exponent ist, desto stärker wird der andere Faktor gewichtet.

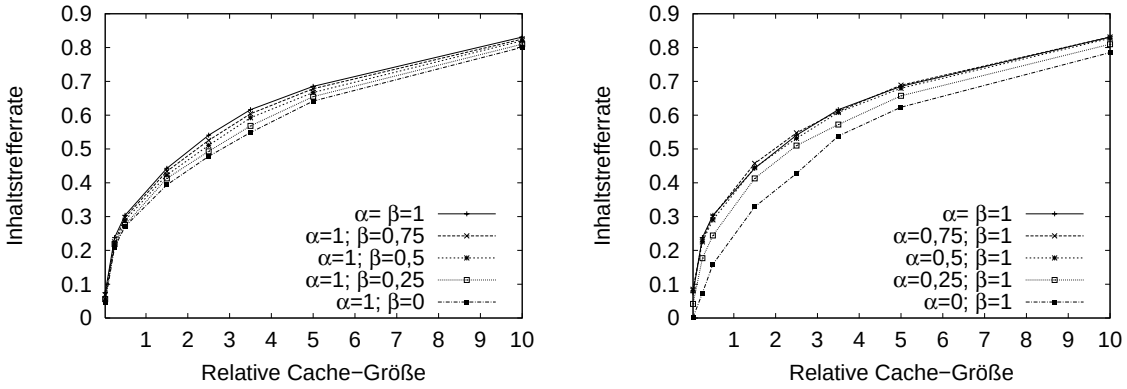


(a) Inhaltstrefferraten (b) Laufzeiten

Abbildung 5.12: Laufzeitverhalten und Inhaltstrefferraten bei variierendem minimalem Pfadgewicht

Zunächst wird das optimale Verhältnis zwischen α und β gesucht. In Abbildung 5.13(a) wird α konstant auf 1 gesetzt, während β zwischen 0,25 und 1 variiert wird. In Abbildung 5.13(b) wird umgekehrt α variiert und β konstant auf 1 gesetzt. Die Bedeutung der Inhaltswahrscheinlichkeit ist in Abbildung 5.13(a) sehr gut zu erkennen, in der α konstant auf 1 gesetzt ist. Die besten Ergebnisse werden für $\beta = 1$ erzielt. Das schlechteste Ergebnis wird erzielt, wenn die Inhaltswahrscheinlichkeit gar nicht berücksichtigt wird. Diese Bedeutung wird in Abbildung 5.13(b) noch unterstrichen, denn mit $\alpha = 0{,}75$ wird für kleinere Caches eine leicht höhere Inhaltstrefferrate erzielt, bei größeren Caches gilt dies für $\alpha = 1$ und $\alpha = 0{,}5$. Bei der Pfadtrefferrate, die hier nicht abgebildet ist, werden in allen Fällen die besten Ergebnisse für $\alpha = \beta = 1$ erzielt. Hieraus lässt sich schließen, dass beide Faktoren für die Relevanz unbedingt erforderlich sind und die besten Resultate erzielt werden, wenn beide gleich stark gewichtet werden.

In der nächsten Auswertung wird die Bedeutung der Clustergröße für die Relevanz pro Byte untersucht. Hierzu werden α und β gleich gesetzt und ihr Wert zwischen 0 und 20 variiert. Für Werte kleiner als 1 wird die Clustergröße stärker gewichtet, für Werte größer als 1 dagegen schwächer. Abbildung 5.14 zeigt, dass wiederum mit $\alpha = \beta = 1$ die besten Inhaltstrefferraten erzielt werden. Mit



(a) $\alpha = 1$ (b) $\beta = 1$

Abbildung 5.13: Einfluss der Inhalts- und Pfadwahrscheinlichkeit auf die Inhaltstrefferraten

steigendem Exponenten größer als 1 verschlechtern sich die Trefferraten, da die Größe immer weniger berücksichtigt wird. Umgekehrt gilt für Exponenten zwischen 0 und 1, dass mit sinkenden Exponenten die Trefferraten ebenso fallen, da nun die Größe überbewertet wird. Für Caches bis zu einer Größe von zirka dem dreifachen des Anfragevolumens sind die Inhaltstrefferraten am schlechtesten für $\alpha = \beta = 0$, da in diesem Fall nur die Größe der Cluster mit in die Berechnung der Relevanz mit einbezogen wird. Für größere Caches werden die schlechtesten Ergebnisse für Werte von $\alpha = \beta > 10$ erzielt, da nun die Clustergröße nicht genügend berücksichtigt wird.

5.4.9 Diskussion

Der Vergleich der ausgewerteten Vorabübertragungsverfahren in unterschiedlichen Teilräumen macht deutlich, dass mit Hilfe der Selektion mit Clusterbildung in kleinen Teilräumen zwar die besten Trefferraten erzielt werden, sich dieses Ergebnis jedoch mit einer deutlich höheren zeitlichen Komplexität gegenüber dem LFU-basierten Ansatz erkauft wird. Die jeweiligen Trefferraten differieren um bis zu zirka acht Prozent. Bei großen Teilräumen lohnt sich dieser Aufwand jedoch

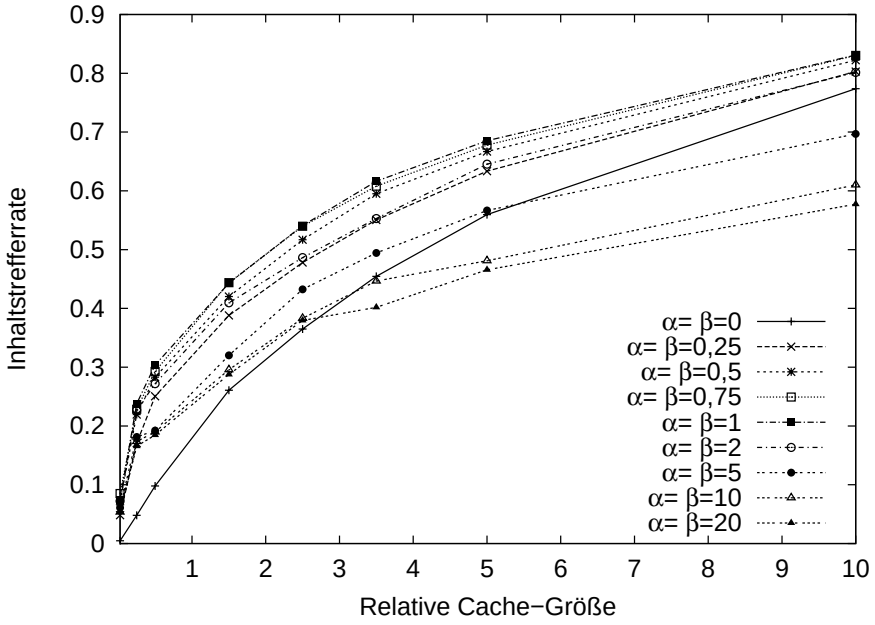


Abbildung 5.14: Einfluss der Clustergröße auf die Inhaltstrefferraten

allemal, denn in diesem Fall sind die Trefferraten um bis zu 3,86 mal höher als das LFU-basierte Verfahren.

Die zeitliche Komplexität des Verfahrens kann durch eine geeignete Wahl des minimalen Pfadgewichts für die Clusterbildung so reduziert werden, dass trotz einer kurzen Laufzeit noch eine genügend hohe Inhaltstrefferrate erzielt werden kann. Der schlechteste Fall eines mehr als quadratischen Aufwands zum Sortieren der Cluster ist nicht aufgetreten, so dass sich auch in großen Informationsräumen mit 10000 Webseiten moderate Laufzeiten ergeben.

Die Vergleiche mit unterschiedlichen Kombinationen der Exponenten für die Berechnung der Relevanz pro Byte eines Clusters haben gezeigt, dass alle drei Faktoren Inhalts-, Pfadwahrscheinlichkeit und die Clustergröße gleich wichtig sind. Die optimale Belegung ergibt sich zu $\alpha = \beta = 1$.

Die Integration der Profile in die Entscheidung, welche Webseiten vorab zu laden sind, hat den Nachteil, dass mehr Cluster gebildet werden und deren Sortierung

für jeden Benutzer durchgeführt werden muss. Die Traversierung der Informationsgraphen mit Clusterbildung kann weiterhin periodisch erfolgen. Die erzielte Steigerung der Trefferraten spricht jedoch bei inhomogenem Zugriffsverhalten für die Integration von Profilen. Zur Verbesserung der Performanz könnten beispielsweise periodisch Vorabübertragungslisten erstellt werden, die populäre Profilmixe darstellen. Aus diesen wird dann diejenige gewählt, die dem aktuellen Profilmix am ehesten entspricht.

Schließlich lernt CBH im Teilraum mit 10000 Webseiten sehr schnell. Bereits mit 150 Aktualisierungs-Protokolldateien wird eine Inhaltstrefferrate über fünfzig Prozent erreicht, wofür beispielsweise mit dem LFU-basierten Verfahren 4000 Protokolldateien benötigt werden.

Die Erweiterung des Verfahren um eine Einbeziehung von Wissen über zukünftige Benutzerbewegungen hat in dem in der Dissertation von Kubach vorgestellten Vorabübertragungsverfahren eine deutliche Steigerung der Trefferraten gegenüber dem Basisverfahren ermöglicht. Aus diesem Grund ist zu erwarten, dass mit der in Abschnitt 3.11 diskutierten diesbezüglichen Erweiterung des Vorabübertragungsverfahrens die Trefferraten nochmals verbessert werden können.

6 Analyse des Energiebedarfs mobiler Endgeräte beim mobilen Informationszugriff

Die Batterie zählt bei mobilen Endgeräten zu den knappen Ressourcen. Optimierungsstrategien für den mobilen Informationszugriff müssen deshalb den Energiebedarf berücksichtigen, um die Akzeptabilität für deren Einsatz zu erhöhen. In diesem Kapitel wird ein Modell des Energiebedarfs von mobilen Endgeräten für das Laden von Webseiten mit und ohne Vorabübertragungsverfahren vorgestellt. Darauf aufbauend wird der Energiebedarf der Funkschnittstellen mobiler Endgeräte analysiert.

6.1 Grundlagen

Die Funkschnittstelle eines mobilen Endgeräts kann sich in einem von drei Zuständen befinden, die in diesem Kapitel für die Analyse des Energiebedarfs betrachtet werden. (1) Im Schlummermodus (S) ist der größte Teil der Schaltkreise abgeschaltet, (2) im Empfangsmodus (RX) empfängt das Gerät Daten und (3) im Übertragungsmodus (TX) werden Daten gesendet. Die WLAN-Funkschnittstelle kann sich zusätzlich noch im Bereitschaftsmodus befinden, in dem das Gerät zwar den Kanal abhört, jedoch keine Daten weiterleitet. Dieser Zustand wird jedoch im nachfolgend vorgestellten Modell des Energiebedarfs nicht berücksichtigt, da

bei der Vorabübertragung die WLAN-Funkschnittstelle nur zur Übertragung der Protokolldatei und der zu hortenden Informationen eingeschaltet werden muss und in der restlichen Zeit im Schlummermodus bleiben kann. In [91] wird gezeigt, dass für Datenmengen bis zu einem Kilobit der Energiebedarf für das Anschalten der Funkschnittstelle größer ist als der Verbrauch für das tatsächliche Senden. Bei größeren Datenmengen fällt der Energiebedarf für das Einschalten jedoch nicht mehr ins Gewicht, weshalb dieser im Folgenden ignoriert wird.

Die Funkschnittstellen mobiler Endgeräte für WWAN- und WLAN-Technologien unterscheiden sich im Energiebedarf nicht nur beim Senden und Empfangen, sondern auch im Schlummermodus. Der Energiebedarf für das Senden oder Empfangen von Daten wird deshalb als die Differenz zwischen dem tatsächlichen Verbrauch während der Sende- bzw. Empfangszeit und der Energie, den die Funkschnittstelle während dieser Zeit im Schlummermodus verbraucht hätte, mittels Gleichung 6.1 berechnet.

$$\hat{E}_{\mathrm{TX/RX}} = E_{\mathrm{TX/RX}} - E_{\mathrm{S}} \tag{6.1}$$

$E_{\mathrm{TX/RX}}$ stellt den tatsächlichen Energiebedarf für das Senden bzw. Empfangen dar und E_{S} den Energiebedarf im Schlummermodus während desselben Zeitintervalls.

Der jeweilige Energiebedarf berechnet sich als

$$E_{\mathrm{TX/RX/S}} = P_{\mathrm{TX/RX/S}} \cdot T_{\mathrm{TX/RX}} \tag{6.2}$$

wobei $P_{\mathrm{TX/RX/S}}$ die Verlustleistung der Funkschnittstelle in den unterschiedlichen Zuständen ist. Das Zeitintervall, in dem s Bytes mit einer verfügbaren Bandbreite B gesendet oder empfangen werden, wird berechnet als:

$$T_{\mathrm{TX/RX}} = s \cdot \frac{8}{B} \tag{6.3}$$

In den folgenden Abschnitten wird der Energiebedarf für die Anforderung von

Webseiten mit und ohne Vorabübertragung berechnet. Dabei werden die folgenden Bezeichner verwendet:

$P_{\text{WWAN,RX}}$: Leistung der WWAN-Funkschnittstelle im Empfangsmodus

$P_{\text{WWAN,TX}}$: Leistung der WWAN-Funkschnittstelle im Übertragungsmodus

$P_{\text{WWAN,S}}$: Leistung der WWAN-Funkschnittstelle im Schlummermodus

$P_{\text{WLAN,RX}}$: Leistung der WLAN-Funkschnittstelle im Empfangsmodus

$P_{\text{WLAN,TX}}$: Leistung der WLAN-Funkschnittstelle im Übertragungsmodus

$P_{\text{WLAN,S}}$: Leistung der WLAN-Funkschnittstelle im Schlummermodus

B_{WLAN}: Bandbreite bei der Nutzung der WLAN-Technologie

$B_{\text{WWAN,TX}}$: Bandbreite für das Senden bei der Nutzung der WWAN-Technologie

$B_{\text{WWAN,RX}}$: Bandbreite für das Empfangen bei der Nutzung der WWAN-Technologie

s_{GET}: Durchschnittliche Größe der HTTP-GET-Anforderung

s_{eintrag}: Durchschnittliche Größe des Eintrags einer Protokolldatei

s_{seite}: Durchschnittliche Größe einer Webseite (inklusive der eingebetteten Dokumente)

$n_{\text{eingebettet}}$: Durchschnittliche Anzahl eingebetteter Dokumente in einer Webseite, wie beispielsweise Multimedia-Dokumente. Diese Zahl spielt eine wichtige Rolle bei der Berechnung des Energiebedarfs, denn das HTTP-Protokoll verlangt, dass jedes eingebettete Dokument separat mittels einer HTTP-GET-Anforderung geladen werden muss.

6.2 Energiebedarf mit Vorabübertragung

Abbildung 6.1 zeigt nochmals die drei durch eingekreiste Ziffern gekennzeichneten Phasen des Vorabübertragungsverfahrens aus Sicht eines mobilen Benutzers.

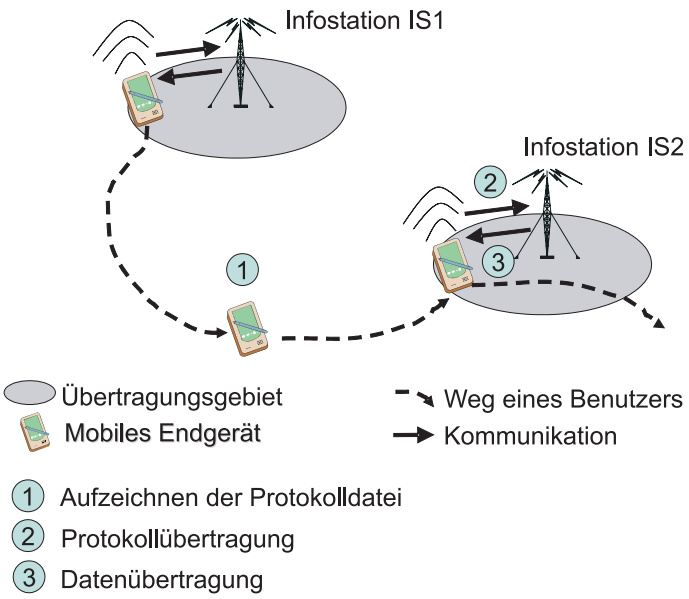


Abbildung 6.1: Ablauf der Vorabübertragung aus Sicht eines mobilen Benutzers

In diesem Abschnitt werden die Phasen hinsichtlich des Energiebedarfs der Funkschnittstelle beschrieben. Für die Vorabübertragung von Webseiten wird in den einzelnen Phasen folgendes Protokoll verwendet:

Phase 1: Jede angeforderte Webseite, die nicht im Cache gefunden wird, muss mittels WWAN-Technologie angefordert werden. Hierzu sind folgende Schritte notwendig:

1. Sende jeweils eine HTTP-GET-Anforderung mittels WWAN-Technologie für die Anforderung der Seite und für jedes eingebettete Dokument in der Seite.
2. Empfange für jede HTTP-GET-Anforderung die HTTP-Antwort mittels WWAN-Technologie.

Phase 2: Sende die Protokolldatei beim Betreten des Übertragungsgebiets einer Infostation mittels WLAN-Technologie an die Infostation.

Phase 3: Empfange die Vorabübertragungsliste mittels WLAN-Technologie.

Sendeenergie Da jede nicht im Cache enthaltene Webseite mittels WWAN-Technologie angefordert werden muss, müssen zur Berechnung der Trefferrate alle gehorteten Webseiten berücksichtigt werden, d.h., es wird nicht zwischen Transit- und Inhaltsseiten unterschieden. Sei L die Menge der in einer Protokolldatei angeforderten Webseiten und H die Menge aller Seiten im Cache. Dann ist die allgemeine Trefferrate $h = \frac{|H \cap L|}{|L|}$ (siehe auch Gleichung 5.1 in Abschnitt 5.3.1).

Sei $n = |L|$ die Anzahl der angeforderten Webseiten und h die mit dem Vorabübertragungsverfahren erzielte allgemeine Trefferrate. Dann werden die Sendezeiten für den WLAN- und WWAN-Anteil nach Gleichung 6.3 wie folgt berechnet, wobei $T_{\text{WLAN,TX}}$ die Zeit zum Senden der Protokolldatei darstellt und $T_{\text{WWAN,TX}}$ entsprechend die Zeit zum Senden der HTTP-GET-Anforderungen. Sei

$$T_{\text{WLAN,TX}} = n \cdot s_{\text{eintrag}} \cdot \frac{8}{B_{\text{WLAN}}} \tag{6.4}$$

$$T_{\text{WWAN,TX}} = (1-h) \cdot n \cdot (n_{\text{eingebettet}} + 1) \cdot s_{\text{GET}} \cdot \frac{8}{B_{\text{WWAN,TX}}} \tag{6.5}$$

Die Faktoren zur Berechnung der Zeit zum Senden der HTTP-GET-Anforderungen in Gleichung 6.5 setzen sich wie folgt zusammen: $(1-h) \cdot n$ ist die Anzahl der Webseiten, die nicht im Cache gefunden wurden, während $(n_{\text{eingebettet}} + 1)$ die Zahl der für eine Webseite notwendigen HTTP-GET-Anforderungen darstellt. Der Energiebedarf für das Senden beträgt schließlich nach den Gleichungen 6.1 und 6.2

$$\hat{E}_{\text{TX}} = (P_{\text{WLAN,TX}} - P_{\text{WLAN,S}}) \cdot T_{\text{WLAN,TX}} + (P_{\text{WWAN,TX}} - P_{\text{WWAN,S}}) \cdot T_{\text{WWAN,TX}}$$

Empfangsenergie Sei $m \cdot n \cdot s_{\text{seite}}$ die Cache-Größe als m-faches des Anfragevolumens $n \cdot s_{\text{seite}}$ und h die mit dem Vorabübertragungsverfahren für diese Cache-Größe erzielte allgemeine Trefferrate. Dann berechnen sich die Zeiten für den Empfang der Webseiten nach Gleichung 6.3 wie folgt, wobei $T_{\text{WLAN,RX}}$ die Zeit zum Empfang der Vorabübertragungsliste darstellt und $T_{\text{WWAN,RX}}$ entsprechend die Zeit zum Empfangen der gesamten Webseite. Der Einfachheit halber

wurde der Empfang der Webseite über das WWAN nicht in den Empfang der eingebetteten Dokumente zerlegt, da in der Gesamtgröße der Webseite die Größe dieser Dokumente bereits enthalten ist.

$$T_{\mathrm{WLAN,RX}} = m \cdot n \cdot s_{\mathrm{seite}} \cdot \frac{8}{B_{\mathrm{WLAN}}} \tag{6.6}$$

$$T_{\mathrm{WWAN,RX}} = (1-h) \cdot n \cdot s_{\mathrm{seite}} \cdot \frac{8}{B_{\mathrm{WWAN,RX}}} \tag{6.7}$$

Der Energiebedarf für das Empfangen beträgt schließlich nach den Gleichungen 6.1 und 6.2

$$\hat{E}_{\mathrm{RX}} = (P_{\mathrm{WLAN,RX}} - P_{\mathrm{WLAN,S}}) \cdot T_{\mathrm{WLAN,RX}} + (P_{\mathrm{WWAN,RX}} - P_{\mathrm{WWAN,S}}) \cdot T_{\mathrm{WWAN,RX}}$$

6.3 Energiebedarf ohne Vorabübertragung

Um eine Webseite über ein WWAN wie GSM (2G-Technologie) oder UMTS (3G-Technologie) anzufordern, wird folgendes Protokoll verwendet:

Jede Webseite muss mittels WWAN-Technologie angefordert werden. Hierzu sind folgende Schritte notwendig:

1. Sende jeweils eine HTTP-GET-Anforderung mittels WWAN-Technologie für die Anforderung der Seite und für jedes eingebettete Dokument in der Seite.
2. Empfange für jede HTTP-GET-Anforderung die HTTP-Antwort mittels WWAN-Technologie.

Sendeenergie Sei n die Anzahl angeforderter Webseiten. Dann berechnet sich die Zeit zum Senden der erforderlichen HTTP-GET-Anforderungen nach Glei-

chung 6.3 wie folgt:

$$T_{\mathrm{WWAN,TX}} = n \cdot (n_{\mathrm{eingebettet}} + 1) \cdot s_{\mathrm{GET}} \cdot \frac{8}{B_{\mathrm{WWAN,TX}}}$$

Der Energiebedarf nach den Gleichungen 6.1 und 6.2 zum Senden der HTTP-GET-Anforderungen berechnet sich dann zu

$$\hat{E}_{\mathrm{TX}} = (P_{\mathrm{WWAN,TX}} - P_{\mathrm{WWAN,S}}) \cdot T_{\mathrm{WWAN,TX}}$$

Empfangsenergie Die Zeit zum Empfangen der HTTP-Antworten berechnet sich nach Gleichung 6.3 zu

$$T_{\mathrm{WWAN,RX}} = n \cdot s_{\mathrm{seite}} \cdot \frac{8}{B_{\mathrm{WWAN,RX}}}$$

Schließlich wird die Energie zum Empfangen der Webseiten nach den Gleichungen 6.1 und 6.2 berechnet als

$$\hat{E}_{\mathrm{RX}} = (P_{\mathrm{WWAN,RX}} - P_{\mathrm{WWAN,S}}) \cdot T_{\mathrm{WWAN,RX}}$$

6.4 Analyse des Energiebedarfs

Im folgenden Abschnitt werden die der Berechnung des Energiebedarfs zugrunde liegenden Parameterwerte beschrieben und anschließend die Ergebnisse der Berechnung vorgestellt.

6.4.1 Parameterbelegung

Der Energiebedarf der Funkschnittstelle eines mobilen Endgeräts hängt ab von deren Leistungsaufnahme, der zur Verfügung stehenden Bandbreite, der durch-

schnittlichen Anzahl der in einer Webseite eingebetteten Dokumente, der Durchschnittsgröße der Webseiten sowie der Anzahl der durchschnittlich angefragten Seiten. Zur Berechnung des Energiebedarfs werden den folgenden Parametern Werte zugeordnet, die aus Forschungsergebnissen in der Literatur stammen.

Leistung der Funkschnittstelle: Für das WLAN wurden Werte aus den Datenblättern der Orinoco 11b client PC-Karte [27] entnommen. Für zelluläre Netze mit 3G-Technologie waren dies entsprechend die Daten der WaveLinx GPC-6210 Karte [47].

Bandbreite: Skold et al. ermitteln in [93] für UMTS eine Bandbreite von 300 Kbps zum Empfangen und 64 Kbps zum Senden von Daten. In [24] werden von der Atheros Communications Inc. unter anderem die zur Verfügung stehenden Bandbreiten für unterschiedliche WLAN-Karten gemessen und beispielsweise für 802.11g eine durchschnittliche Bandbreite von 20 Mbps ermittelt. In dieser Analyse werden jedoch nur 3 Mbps angenommen, was der zehnfachen Bandbreite zum Empfangen mittels WWAN-Technologie entspricht. Der kleinere Wert wurde unter der Annahme gewählt, dass sich an einer Infostation mehr als ein Benutzer gleichzeitig aufhalten wird und somit die maximale Bandbreite höchstwahrscheinlich nicht zur Verfügung steht.

Durchschnittliche Anzahl eingebetteter Dokumente: In der Literatur, wie beispielsweise in [22, 67, 75], variiert diese Anzahl zwischen 4 und 14. Basierend hierauf werden Werte von 4, 9 und 14 angenommen.

Durchschnittliche Größe einer HTTP-GET-Anforderung:
Entsprechend den Ergebnissen von Mah in [67] wird diese Größe zu 320 Bytes gewählt.

Tabelle 6.1: Parameter des Energiebedarfs

Parameter	**Wertebereich**
$P_{\text{WWAN,RX}}$[W]	0,5
$P_{\text{WWAN,TX}}$[W]	2,8
$P_{\text{WWAN,S}}$[W]	0,1
$P_{\text{WLAN,RX}}$[W]	0,9
$P_{\text{WLAN,TX}}$[W]	1,4
$P_{\text{WLAN,S}}$[W]	0,05
$B_{\text{WWAN,RX}}$[Kbps]	300
$B_{\text{WWAN,TX}}$[Kbps]	64
B_{WLAN}[Kbps]	3000
s_{seite}[KByte]	20
s_{eintrag}[KByte]	0,32
s_{GET}[KByte]	0,32
$n_{\text{eingebettet}}$	4;9;14
n	100
$m \cdot n \cdot s_{\text{seite}}$	0,2 ... 20

Die durchschnittliche Anzahl n von Webseiten-Anforderungen wird entsprechend der Leistungsbewertung in Kapitel 5 auf 100 gesetzt, die Größe des Caches ($m \cdot n \cdot s_{\text{seite}}$) variiert zwischen 200 KBytes und 20 MBytes. In Tabelle 6.1 sind diese Parameterbelegungen zusammengefasst.

6.4.2 Diskussion

In den folgenden Auswertungen wird der Energiebedarf der Funkschnittstelle eines mobilen Endgeräts analysiert, wenn das vorgestellte Vorabübertragungsverfahren mit Clusterbildung (CBH) eingesetzt wird. Berechnungsgrundlage sind die mittels CBH erzielten allgemeinen Trefferraten (berechnet nach Gleichung 5.1 in Kapitel 5) für Teilräume mit 10000 bzw. 1000 Seiten bei einer durchschnittlichen Seitengröße von 20 KBytes. Diesem Wert wird der entsprechende Energiebedarf für eine reine Nutzung der WWAN-Technologie ohne Vorabübertragung gegenübergestellt.

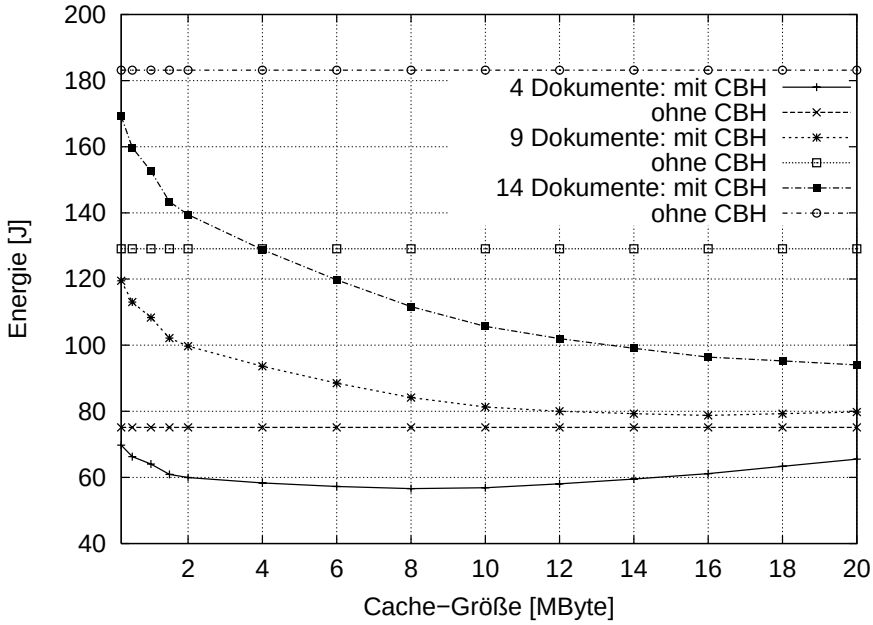


Abbildung 6.2: Energiebedarf mit und ohne Vorabübertragung für mehrere eingebettete Dokumente (Teilraum mit 10000 Seiten (20KBytes))

In Abbildung 6.2 ist der Energiebedarf abhängig von der Cache-Größe für jeweils 4, 9 und 14 eingebettete Dokumente dargestellt. Auf der x-Achse ist die absolute Cache-Größe in MByte aufgetragen, die im Intervall zwischen 0,2 MBytes und 20 MBytes liegt. Die y-Achse zeigt den Energieverbrauch in Joule. Die Kurven sind mit „4 Dokumente mit CBH" bzw. „ohne CBH" für die jeweilige Anzahl eingebetteter Dokumente gekennzeichnet. Die erzielten allgemeinen Trefferraten beziehen sich auf den Informationsraum mit 10000 Webseiten.

Abbildung 6.3 zeigt entsprechend den Energiebedarf für die allgemeinen Trefferraten, die für den Informationsraum mit 1000 Webseiten erzielt werden.

Aus den Auswertungen ergibt sich, dass durch Einsatz von CBH in allen Fällen Energieeinsparungen möglich sind. Abhängig von der Zahl eingebetteter Dokumente sinkt der mit CBH benötigte Energiebedarf bis zu einer bestimmten Cache-Größe auf einen minimalen Wert ab und steigt dann wieder mit größerem

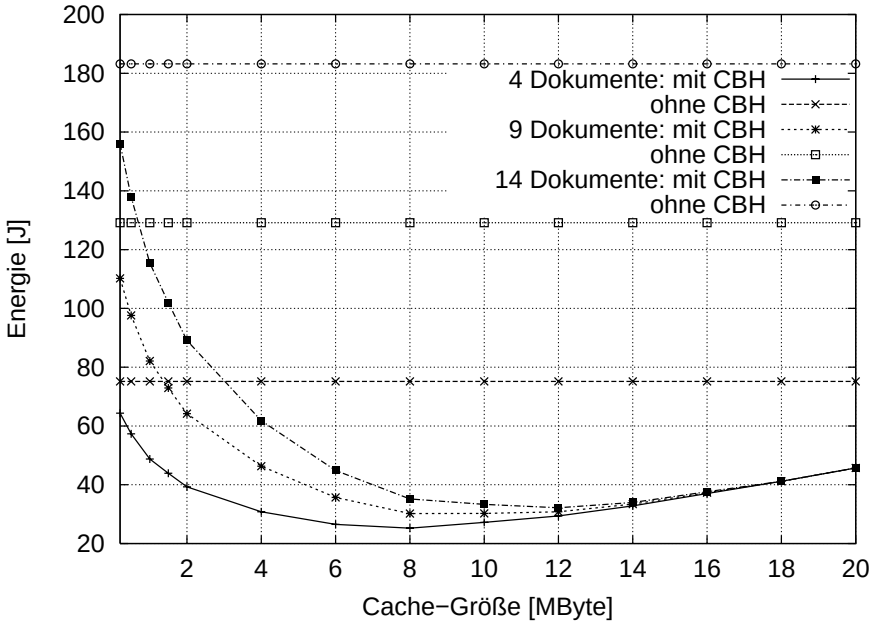


Abbildung 6.3: Energiebedarf mit und ohne Vorabübertragung für mehrere eingebettete Dokumente (Teilraum mit 1000 Seiten (20KBytes))

Cache. Beispielsweise ist der Energiebedarf bei durchschnittlich vier eingebetteten Dokumenten im 10000 Seiten fassenden Teilraum bei einer Cache-Größe von 8 MBytes minimal. Die Ursache hierfür liegt darin, dass in dem Maße, wie die Trefferrate steigt, zwar die Anzahl zusätzlicher Anfragen über das WWAN verringert wird, gleichzeitig jedoch durch die steigende Cache-Größe mehr Informationen vorab übertragen werden. Um diesen Effekt zu untersuchen, wird in den nächsten Abbildungen der gesamte Energiebedarf in Sende- und Empfangsenergie aufgegliedert.

Abbildung 6.4 stellt den Energiebedarf für das Senden und Empfangen von Webseiten mit durchschnittlich vier eingebetteten Dokumenten dar, aufgeschlüsselt nach den Anteilen von Sende- und Empfangsenergie. Die x-Achse zeigt die absolute Cache-Größe in MByte, auf der y-Achse ist der Energieverbrauch in Joule aufgetragen. Das Balkendiagramm beschreibt die Sende- und Empfangsenergie unter Einsatz von CBH im Teilraum mit 10000 Webseiten und ist gekennzeich-

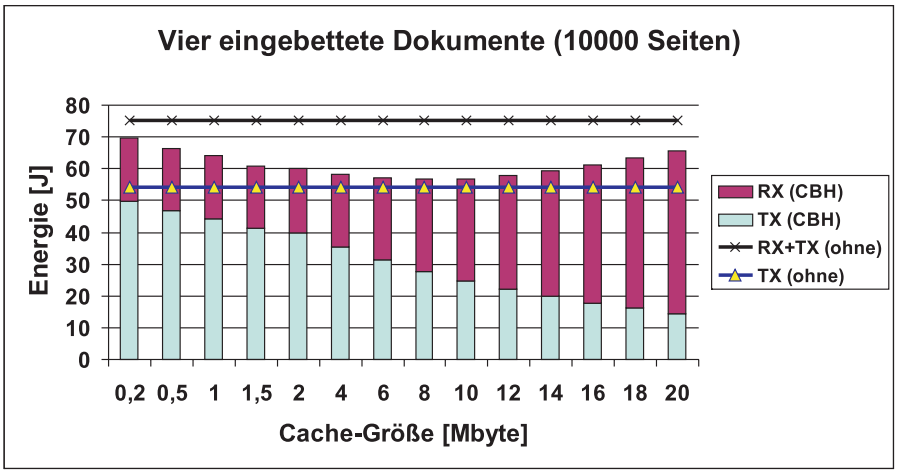


Abbildung 6.4: Energiebedarf aufgeschlüsselt nach Sende- und Empfangsenergie (10000 Seiten, durchschnittlich vier eingebettete Dokumente)

net durch „TX (CBH)" bzw. „RX (CBH)". Die beiden waagrechten Linien stellen die Sende- und Gesamtenergie für das Browsen im Web ohne Vorabübertragung dar (wenn also nur die WWAN-Technologie eingesetzt wird) und sind bezeichnet mit „TX (ohne)" bzw. „RX+TX (ohne)". Im Falle der Vorabübertragung liegt der Energiebedarf der Funkschnittstellen für WLAN und WWAN in allen Fällen unter dem Energiebedarf der Funkschnittstelle für das WWAN, wenn keine Vorabübertragung verwendet wird: Bei der Übertragung von Daten über die WWAN-Funkschnittstelle wird der Energiebedarf überwiegend von der Sendeenergie verursacht. Dies macht sich auch bei der Vorabübertragung bemerkbar, denn bei kleinen Cache-Größen und damit geringen allgemeinen Trefferraten, werden die meisten Webseiten über die WWAN-Schnittstelle übertragen. Mit wachsender Cache-Größe steigt zwar erwartungsgemäß die Empfangsenergie, die Sendeenergie sinkt jedoch in gleichem Maße, wie die allgemeine Trefferrate ansteigt. Der Energiebedarf ist minimal für eine Cache-Größe von 8 MBytes. Für größere Caches steigt zwar die allgemeine Trefferrate an, das Verhältnis der Anzahl der Webseiten, die zum Anstieg der Trefferraten beitragen, zur Anzahl der insgesamt übertragenen Seiten wird jedoch immer kleiner. Grob gesagt, werden in diesem Fall immer mehr Webseiten umsonst übertragen. Für Caches bis

20 MBytes ist jedoch für Webseiten mit durchschnittlich vier eingebetteten Dokumenten die hohe Empfangsenergie durch die Einsparungen beim Senden gerechtfertigt, denn der gesamte Energiebedarf liegt immer noch unter dem ohne Vorabübertragung.

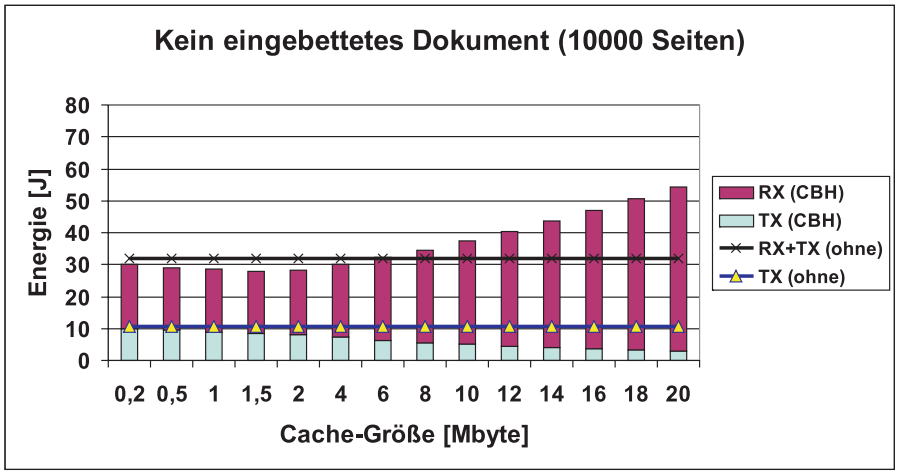


Abbildung 6.5: Energiebedarf aufgeschlüsselt nach Sende- und Empfangsenergie (10000 Seiten, kein eingebettetes Dokument)

Analog hierzu zeigt Abbildung 6.5 die gleiche Aufschlüsselung in Sende- und Empfangsenergie, nur wird in diesem Fall eine Webseite betrachtet, die nur aus einem einzigen Dokument besteht. Hier kann man deutlich erkennen, dass nun in beiden Fällen (mit und ohne CBH) die Empfangsenergie überwiegend für den gesamten Energiebedarf verantwortlich ist. Ab einer Cache-Größe von 10 MBytes übersteigt die Empfangsenergie mit CBH den gesamten Energiebedarf ohne CBH. Für Cache-Größen bis 6 MBytes ist jedoch der gesamte Energiebedarf mit CBH geringer als der ohne CBH. Die maximale Einsparung von Energie wird bei einer Cache-Größe von 1,5 MBytes erreicht.

6.5 Zusammenfassung

Das ursprüngliche Ziel der Vorabübertragung von Webseiten als Optimierungstechnik für den mobilen Informationszugriff war es, den Nachteilen drahtloser Weitverkehrsnetze, wie beispielsweise eine hohe Latenz, entgegen zu wirken. Auf den ersten Blick könnte man vermuten, dass durch das Laden von Webseiten, die nie angefordert werden, die erzielten Vorteile des Verfahrens durch einen erhöhten Energieverbrauch des mobilen Endgeräts erkauft werden. Aus den Analyseergebnissen wird jedoch deutlich, dass durch die Reduzierung der Zugriffe im WWAN, die durch eine hohe Trefferrate erreicht wird, die Einsparung von Energie ermöglicht wird.

Der Energiebedarf einer Funkschnittstelle steigt proportional mit der zum Senden und Empfangen der Webseiten erforderlichen Übertragungszeit. Diese fällt durch die höhere Bandbreite im WLAN wesentlich geringer aus als im WWAN, was sich durch einen geringeren Energiebedarf der WLAN-Funkschnittstelle gegenüber der WWAN-Funkschnittstelle bemerkbar macht. Dies wird besonders beim Senden der Webseiten-Anforderungen deutlich, denn hier ist der Unterschied zwischen den zur Verfügung stehenden Bandbreiten wesentlich größer als beim Empfangen der Daten. Aus diesem Grund ist die Energieeinsparung auch umso größer, je höher die der Anteil an eingebetteten Dokumenten in einer Webseite ist. In dem Maße, wie die Trefferrate steigt, sinkt zwar die Übertragungszeit für das Senden der Anforderungen, da in diesem Fall mehr Webseiten mit Hilfe der WLAN-Technologie gesendet werden. Gleichzeitig steigt jedoch die zur Erzielung einer höheren Trefferrate notwendige Anzahl vorab zu ladender Webseiten, wodurch wiederum die Zeit zum Empfangen erhöht wird. Aus diesem Grund wird die Energieeinsparung umso größer sein, je höher die erzielten Trefferraten bereits für geringe Cache-Größen sind.

7 Implementierung und Integration in die NEXUS-Plattform

Im Rahmen dieser Arbeit wurde ein generisches Vorabübertragungsverfahren entwickelt und in Java 5.0 implementiert, das die Aktualisierung eines Informationsgraphen sowie die Erzeugung einer Vorabübertragungsliste auf unterschiedliche Arten ermöglicht. Der Fokus dieser Arbeit liegt auf der Leistungsbewertung des Verfahrens und nicht auf der Entwicklung eines für den realen Einsatz konzipierten Systems, weshalb lediglich ein Prototyp mit der notwendigen Kernfunktionalität erstellt wurde (engl. *Proof-of-Concept*). Eine für ein real einsetzbares System erforderliche Architektur und Definition der Schnittstellen wurde in der Dissertation von Kubach [54] vorgeschlagen. Die Implementierung des entwickelten Verfahrens kann in diese Architektur integriert werden. Eine wichtige Randbedingung für die Implementierung war eine nahtlose Integration in die NEXUS-Plattform [44].

7.1 Architektur

Das System zur Vorabübertragung besteht aus Infostationen und mobilen Endgeräten. In dem während dieser Arbeit erstellten Prototypen erfolgt die Kommunikation zwischen mobilem Endgerät und Infostation über eine Datenbankschnittstelle: Die Protokolldateien wurden mit dem Anfragengenerator UCW erzeugt (siehe Abschnitt 4.3) und in einer Datenbank gespeichert, von der die Infostati-

on jeweils eine Protokolldatei liest. Aus diesem Grund wird in dieser Arbeit die Beschreibung der Architektur auf die Infostation beschränkt. Gemäß dem in Abschnitt 3.3 beschriebenen Systemmodell werden zusätzlich ein Verzeichnisdienst, ein Positionsbestimmungssystem und ein Ereignisdienst als externe Dienste benötigt. Diese Dienste sind jedoch zur Leistungsbewertung des vorgestellten Verfahrens nicht notwendig und wurden in der Dissertation von Kubach [54] bereits spezifiziert, weshalb sie in dieser Arbeit nicht berücksichtigt werden.

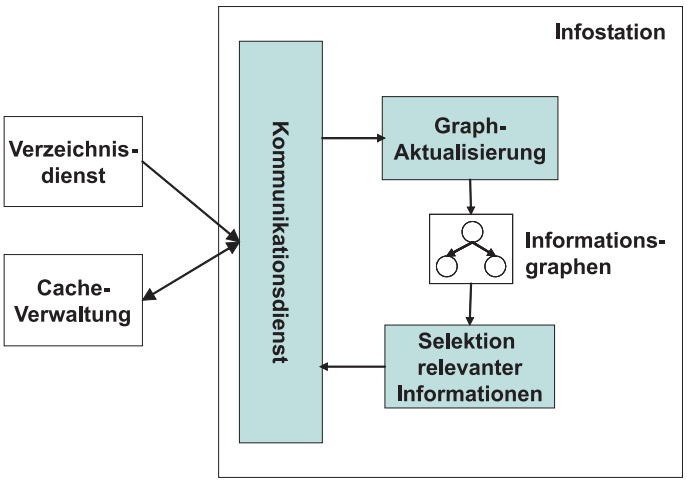


Abbildung 7.1: Architektur der Infostation

Abbildung 7.1 zeigt die Architektur der Infostation. Der *Kommunikationsdienst* empfängt von der Cache-Verwaltung des mobilen Endgeräts eine Protokolldatei und sendet auf Anfrage die aktuelle Vorabübertragungsliste zurück. Der *Verzeichnisdienst* gibt auf Anfrage eine Liste aller Infostationen zurück, deren Dienstgebiet eine bestimmte Position beinhalten.

Die Komponente zur *Graphaktualisierung* analysiert die Protokolldatei und aktualisiert damit die Informationsgraphen, die den Nutzungsprofilen im Profilmix zugeordnet sind. Mit Hilfe der Komponente für die *Selektion relevanter Informationen* wird eine Vorabübertragungsliste erstellt.

Das vorgestellte Verfahren ist generisch, das heißt, es unterstützt prinzipiell mehrere Verfahren zur Graphaktualisierung auf Grundlage unterschiedlich definierter

Sitzungen oder Alterungsfunktionen, sowie unterschiedliche Selektionsverfahren. Aus diesem Grund müssen Klassen dynamisch instantiiert werden können, da erst zur Laufzeit bekannt ist, um welche Klasse es sich handeln soll. Hierzu wird das von Gamma et al in [38] vorgestellte Entwurfsmuster *Fabrikmethode* (engl. *factory method design pattern*) eingesetzt. Es gehört zur Gruppe der Erzeugungsmuster, die eine Abstraktion von der Erzeugung von Objekten bieten. Mit Hilfe der Fabrikmethode wird die Objekterzeugung an spezielle *Fabrikobjekte* delegiert, die auch *virtuelle Konstruktoren* genannt werden. Die Information, welche Instanz einer Klasse erzeugt werden soll, wird zur Laufzeit aus einer Konfigurationsdatei gelesen.

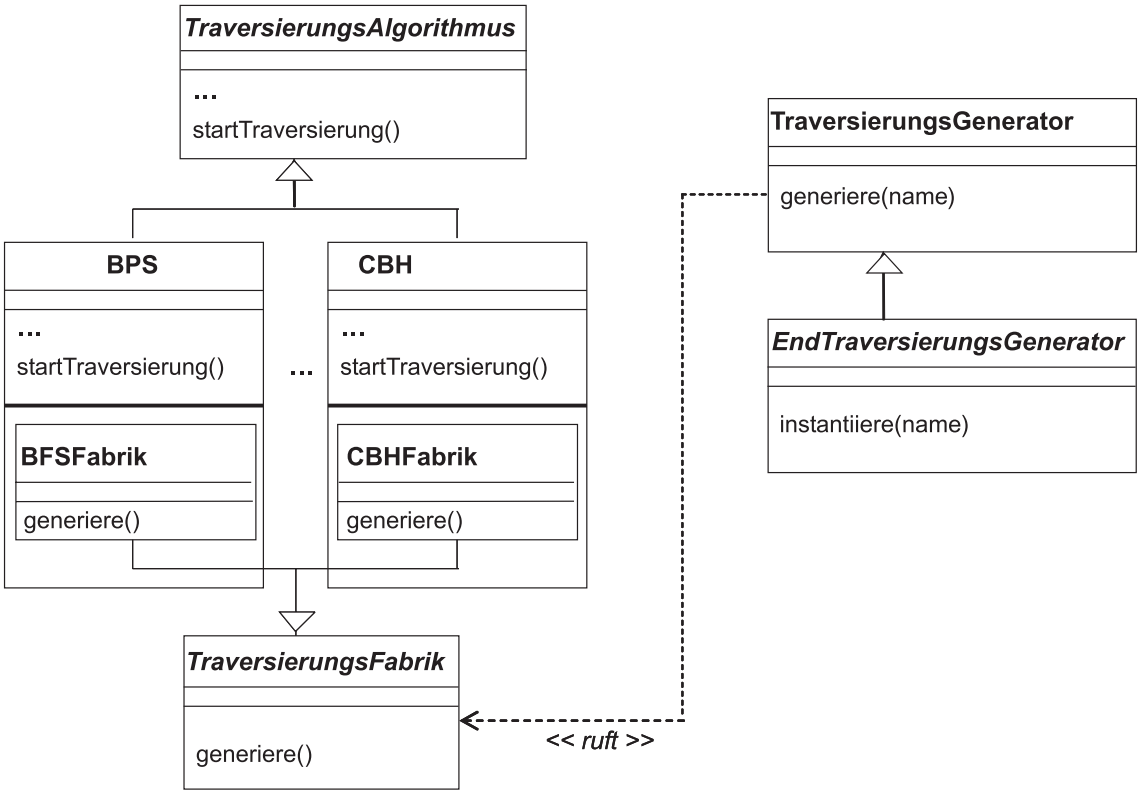


Abbildung 7.2: Traversierungsalgorithmen, modelliert nach dem Entwurfsmuster *Fabrikmethode*

Die Umsetzung dieses Entwurfsmusters wird beispielhaft in Abbildung 7.2 für das

dynamische Laden eines Algorithmus zur Traversierung eines Informationsgraphen beschrieben. Die abstrakte Klasse *TraversierungsAlgorithmus* hat für jeden implementierten Algorithmus eine Unterklasse, in der Abbildung sind beispielhaft *BPS* und *CBH* dargestellt. Diese besitzen jeweils eine innere Fabrik-Klasse *BPSFabrik* bzw. *CBHFabrik*, die wiederum Unterklassen der abstrakten Klasse *TraversierungsFabrik* sind. Beim Aufruf der Methode *generiere()* einer inneren Fabrik-Klasse wird eine Instanz der äußeren Klasse erzeugt und zurückgegeben.

Die abstrakte Klasse *EndTraversierungsGenerator* fungiert als virtueller Konstruktor für die zu instantiierende Klasse, deren Name erst zur Laufzeit bekannt ist, und wird von der Klasse *TraversierungsGenerator* implementiert. Letztere verwaltet eine Hashtabelle, die Instanzen aller ihr bislang bekannten Fabrik-Klassen enthält. Ist eine gewünschte Instanz *neueFabrik* noch nicht in der Hashtabelle enthalten, wird mit Hilfe der Methode *Java.lang.Class.forName(neueFabrik)* diese Fabrik-Klasse instantiiert und die Instanz in die Hashtabelle eingetragen. Zuletzt wird die Methode *generiere* der gewünschten Fabrik-Klasse aufgerufen, die als Ergebnis die Instanz der Klasse des gewünschten Algorithmus für die Traversierung zurückgibt.

Nachfolgend werden die Schnittstellen definiert, die der Kommunikationsdienst einer Infostation nach außen anbieten muss.

upload(*Protokolldatei* log): Wird von der Cache-Verwaltung aufgerufen, um eine Protokolldatei auf die Infostation zu laden.

***Vorabübertragungsliste* download():** Wird von der Cache-Verwaltung aufgerufen und lädt die aktuelle Vorabübertragungsliste auf das mobile Endgerät.

7.2 Realisierung

Die Initialisierung der Infostation erfolgt mittels einer Konfigurationsdatei, die unter anderem folgende Informationen beinhaltet:

- Bezeichner der Aktualisierungs- und Evaluierungs-Protokolldateien
- Zeitspanne einer Epoche
- die einer Sitzung zugrunde liegende Relation (Graphaktualisierung)
- Alterungsfunktion (Graphaktualisierung)
- Traversierungsalgorithmus (Selektionsverfahren)

Schließlich sind noch die für die verwendeten Verfahren notwendigen Parameterwerte zu setzen.

Beim erstmaligen Start einer Infostation bestehen die Informationsgraphen jeweils nur aus dem Wurzelknoten und dem zentralen Knoten. Jede Ankunft einer Protokolldatei führt dann zur Aktualisierung der Graphen. Sobald eine Epoche endet, werden die Graphen traversiert, die daraus entstehende Cluster- oder Knotenliste sortiert und daraus die Vorabübertragungsliste gebildet. Diese Funktionalität wurde mit Hilfe von Threads realisiert.

7.2.1 Schnittstellen zu neXus

Im Projekt neXus werden Umgebungsmodelle unterschiedlichster Anbieter verwaltet. Diese Modelle enthalten Abbilder von Objekten der realen Welt wie beispielsweise Gebäude oder Straßen, aber auch mobile Objekte wie Personen oder Fahrzeuge. Weiterhin können die Modelle mit digitalen Informationen angereichert sein, die beispielsweise aus dem Web oder aus digitalen Bibliotheken stammen. Als Anfragesprache für diese Modelle wird die *Augmented World Query Language* (AWQL) verwendet, als Beschreibungssprache die *Augmented*

World Modeling Language (AWML), die in [78] dokumentiert sind. Die *Föderation* stellt eine einheitliche Schnittstelle für die Kommunikation zwischen einer NEXUS-Anwendung und den einzelnen Umgebungsmodellen zur Verfügung, so dass diese die Sicht auf ein globales Umgebungsmodell erhalten.

Für die Einbindung in NEXUS muss auf dem mobilen Endgerät die Cache-Verwaltung so angepasst werden, dass sie die AWQL-Anfragen der NEXUS-Anwendung interpretieren kann und umgekehrt Antworten auf diese Anfragen in AWML an die Anwendung weitergeben kann.

8 Zusammenfassung und Ausblick

In dieser Dissertation wurde zum Zweck der Optimierung des mobilen Informationszugriffs ein generisches Verfahren zur Vorabübertragung von beliebigen schwach strukturierten Informationen in ortsbasierten Anwendungen vorgestellt. Als Basis wird eine Infrastruktur von so genannten Infostationen benötigt, die mobilen Benutzern mittels drahtloser lokaler Netze einen breitbandigen und kostengünstigen Zugriff auf Informationen ermöglichen. Sie sind in Gebieten verteilt, in denen sonst keine oder nur eine Kommunikation mit der maximalen Datenrate eines Mobilfunknetzes wie beispielsweise GSM, GPRS oder UMTS möglich ist. Da für den Betrieb einer Infostation lediglich Standardkomponenten benötigt werden, können Informationsanbieter einfach und kostengünstig eine solche Infrastruktur aufbauen.

Eine Infostation selektiert die für einen Benutzer relevanten Informationen und überträgt sie vorab auf das mobile Endgerät. Um eine möglichst hohe Relevanz der vorab geladenen Informationen zu erzielen, wird als Selektionskriterium neben deren Ortsbezug auch die semantische Nähe von Informationen berücksichtigt. Letztere ist ein Maß für die Wahrscheinlichkeit, dass ein Zugriff auf die eine Information den Zugriff auf die andere zur Folge hat. Hierfür modelliert eine Infostation das Wissen über das Zugriffsverhalten aller Benutzer, die sich in ihrem Dienstgebiet aufhalten, mit Hilfe eines Informationsgraphen. Für die Selektion der vorab zu ladenden Informationen werden Verfahren mit und ohne Clusterbildung zur Verfügung gestellt. Bei dem Verfahren mit Clusterbildung stellt ein erzeugter Cluster die Vorabübertragungseinheit dar. Da die Auswertung des Zugriffsverhaltens aller Benutzer im Dienstgebiet einer Infostation nicht in jedem

Fall in einer optimalen Entscheidung für einen individuellen Benutzer resultieren muss, werden Nutzungsprofile mit in die Entscheidung einbezogen, welche Informationen vorab zu laden sind.

Das generische Verfahren wurde für Webseiten als schwach strukturierte Informationen spezialisiert und evaluiert. Die Clusterbildung basiert in diesem Fall auf der Klassifikation der Seiten in so genannte Inhalts- und Transitseiten, wobei eine Transitseite zur Navigation verwendet wird, um eine Inhaltsseite zu finden, also eine Seite, an der ein Benutzer tatsächlich interessiert ist.

Die systematische Evaluierung des Verfahrens erfordert eine statistisch relevante Anzahl von Protokolldateien, die unterschiedlich dimensionierten Informationsräumen zugeordnet sind. Aus diesem Grund wurde in dieser Dissertation eine Reihe unterschiedlicher Informationsräume synthetisch erzeugt, die jeweils mit einer Vielzahl von Protokolldateien assoziiert wurden. Letztere wiederum wurden jeweils für mehrere Nutzungsprofile erzeugt. Hierfür wurde ein Modell für das Navigationsverhalten von Benutzern im Web erstellt, das aus zwei Teilmodellen besteht, dem *Webgraph-Modell* und dem *Zugriffsmodell*. Das Webgraph-Modell stellt geeignete Verteilungsfunktionen für die Größen von Webseiten, sowie für die Ein- und Ausgangsgrade der Knoten eines Webgraphen zur Verfügung, die zur typischen Struktur des Webs in Form einer Fliege (engl. *bow-tie*) führen. Das Zugriffsmodell bildet das eigentliche Navigationsverhalten der Benutzer im Web ab und beinhaltet die Popularität von Webseiten, die Anzahl der Zugriffe auf Webseiten innerhalb einer Sitzung, die Zeit, wie lange ein Benutzer auf einer Webseite verweilt, sowie die Art der Webseiten-Anforderung. Mit Hilfe des Anfragengenerators als Implementierung des Web-Navigationsmodells können Informationsräume und damit assoziierte Protokolldateien für unterschiedliche Nutzungsprofile erzeugt werden, die beispielsweise dem Vorabübertragungsverfahren als Eingabe dienen.

Die erzielten Resultate zeigen, dass die Selektion mit Clusterbildung andere Verfahren, die keine Beziehungen zwischen den Webseiten und/oder keine Clusterbildung durchführen, um mehr als das Dreifache in der Trefferrate übertrifft.

Schließlich verspricht die vorgeschlagene Erweiterung des Vorabübertragungsverfahrens um die Integration von Wissen über zukünftige Benutzerbewegungen, die mit der Clusterbildung erzielten sehr guten Ergebnisse nochmals zu verbessern. Für eine vollständige Leistungsbewertung dieses erweiterten Ansatzes ist dessen Implementierung erforderlich.

Selbst wenn zukünftig in Weitverkehrsnetzen höhere Datenraten erzielt werden, so werden lokale Netze doch immer durch die geringe Distanzüberbrückung eine höhere Bandbreite bieten. Somit lohnt sich die Vorabübertragung in zweierlei Hinsicht. Zum einen ermöglicht sie die Einsparung von Energie, was in Anbetracht der knappen Energieressourcen mobiler Endgeräte ein nicht zu vernachlässigender Faktor ist. Zum anderen wird die Latenz für den Zugriff auf die im Cache gespeicherten Informationen nahezu auf Null reduziert. Dies wirkt sich vor allem bei Informationen mit einem hohen Anteil an Multimediadaten oder bei 3D-Modellen positiv aus, deren Größe sehr schnell ansteigen kann. Der Einsatz eines Vorabübertragungsverfahrens in ortsbasierten Systemen wird also stets von Vorteil sein.

Nachfolgend werden einige Erweiterungsmöglichkeiten vorgeschlagen.

Das vorgestellte Verfahren kann auch zur *Vorabübertragung von stark strukturierten (räumlichen) Informationen* eingesetzt werden. Hier ergeben sich vielfältige neue Einsatzmöglichkeiten, insbesondere im Bereich der ortsbasierten oder allgemein kontextbezogenen Systeme, die auf einem räumlichen Umgebungsmodell wie in NEXUS aufbauen und deren Relevanz ständig zunimmt. So wurde beispielsweise im Bereich der kontextbezogenen Informationssysteme am European Media Lab (EML) in Zusammenarbeit mit der Universität Heidelberg ein intelligenter elektronischer Touristenführer namens „Deep Map" entwickelt und bereits prototypisch eingesetzt. Die für die Vorabübertragung in stark strukturierten Informationsräumen notwendigen Erweiterungen werden zur Zeit in einer Diplomarbeit untersucht, für eine abschließende Bewertung des Ansatzes ist eine vollständige Realisierung des Verfahrens erforderlich. Zu dessen systematischer Evaluierung werden ähnlich wie bei schwach strukturierten Informationen

unterschiedlich dimensionierte Informationsräume benötigt, die wiederum mit einer statistisch relevanten Anzahl von Protokolldateien assoziiert werden müssen. Hierfür ist die Modellierung des Zugriffsverhalten von Benutzern in stark strukturierten Informationsräumen erforderlich.

Zur *Optimierung der Kommunikation* bietet sich ein in Abschnitt 2.1.2 beschriebenes ortsbasiertes Rundsendeverfahren an, bei dem die Struktur des Rundsendeprogramms die Reihenfolge und Häufigkeit der zu sendenden Informationen festlegt. Zu dessen Optimierung können die ermittelten Relationen zwischen den Informationen, Profilinformationen und Bewegungsmuster herangezogen werden. Dies stellt vor allem in stark strukturierten Informationsräumen neue Herausforderungen. So könnten beispielsweise bei räumlichen Modellen, die in unterschiedlichen Detaillierungsgraden vorliegen, gröbere Modellinformationen häufiger gesendet werden als solche mit einem höheren Detaillierungsgrad. Aus Gründen der Einsparung von Energie empfiehlt sich die Integration von Indexinformationen in den Rundsendezyklus. Hierdurch müssen mobile Endgeräte nicht mehr einen vollständigen Rundsendezyklus lesen, um die relevanten Informationen herauszufiltern. Ein Endgerät schaltet sich erst dann in einen Rundsendezyklus ein, wenn die benötigte Information übertragen wird, in der restlichen Zeit verharrt es im Schlummermodus. Der push-basierte Ansatz ist offensichtlich dann vorteilhaft, wenn sich die Informationsbedürfnisse der Klienten weitgehend überlappen und viele mobile Endgeräte bedient werden müssen. Im anderen Fall ist der in dieser Arbeit beschriebene pull-basierte Ansatz vorzuziehen. Zur Steigerung der Effizienz können beide Ansätze abhängig vom Überlappungsgrad des Informationsbedarfs und der Anzahl von Endgeräten kombiniert werden. Da beide Kriterien dynamischer Natur sind, kann man davon ausgehen, dass mit einem adaptiven hybriden Verfahren die höchste Effizienz zu erreichen ist.

A Mathematische Grundlagen

A.1 Beschreibende Statistik

Im Folgenden werden die in dieser Arbeit verwendeten Lokations- und Konzentrationsmaße kurz erläutert.

Das **geometrische Mittel** wird verwendet, wenn die Datenmasse nicht normalverteilt ist, also nach einer Seite hin sehr große Ausreißer möglich sind. Ein weiterer Einsatzbereich ist die Berechnung der relativen Veränderung der Merkmalsausprägungen wie beispielsweise durchschnittliche Wachstumsraten. Das geometrische Mittel wird mittels Gleichung A.1 berechnet:

$$G = \sqrt[n]{x(1) \cdot x(2) \cdots x(n)} \tag{A.1}$$

Konzentrationsmaße dienen zur Charakterisierung der Verteilung einer Merkmalssumme S auf die einzelnen Merkmalsträger. Die Merkmalssumme wird mittels Gleichung A.2 berechnet.

$$S = \sum_{i=1}^{n} x(i) \tag{A.2}$$

Eine Konzentration liegt vor, wenn diese Verteilung ungleich ist, d.h. ein großer Teil der Merkmalssumme häuft sich bei einem geringen Teil der Merkmalsträger. Dabei werden zwei Arten der Messung von Konzentration unterschieden:

Absolute Konzentration: Ein großer Teil der Merkmalssumme entfällt auf eine *kleine Zahl* von Merkmalsträgern. Die Maßzahl für die absolute Konzentration ist der *Herfindahlindex*, der mittels Gleichung A.3 berechnet wird, wobei S die Merkmalssumme aus Gleichung A.2 ist.

$$H = \sum_{i=1}^{n} \left(\frac{x_i}{S}\right)^2 \tag{A.3}$$

Bei völliger Gleichverteilung der Anteile ist der Herfindahlindex $H = \frac{1}{n}$, bei Vorliegen eines Monopols $(x_1 = 1, x_2 \ldots x_n = 0)$ ist $H = 1$.

Relative Konzentration: Ein großer Teil der Merkmalssumme entfällt auf einen *kleinen prozentualen Anteil* von Merkmalsträgern, sie wird deshalb auch Disparität genannt. Ein Beispiel hierfür ist die so genannte 80-20-Regel nach Pareto: 80% des Gesamtwerts einer Menge wird von 20% der Elemente erreicht. Das graphische Modell für die relative Konzentration ist die *Lorenzkurve*, mit der die Abweichung einer gegebenen Verteilung von der Gleichverteilung dargestellt wird. Abbildung A.1 illustriert die Lorenzkurve für das obige Beispiel der 80/20-Regel. Die x-Achse repräsentiert die relative Summenhäufigkeit $F_i = \frac{i}{n}$, wobei n die Anzahl der Merkmalsträger ist. Die y-Achse stellt die kumulierten Anteile an der Merkmalssumme dar: $P_i = \frac{\sum_{j=1}^{i} x_j}{S}$ mit $P_0 = 0$. Der *Gini-Koeffizient* G ist deren begleitende Maßzahl und wird mittels Gleichung A.4 berechnet. Die relative Konzentration ist maximal für $G = 1$ und minimal für $G = 0$.

$$\begin{aligned} G &= \frac{\text{Fläche zwischen Diagonale und Lorenzkurve}}{\text{Fläche unter der Diagonalen}} \\ &= 1 - \frac{1}{n} \sum_{i=1}^{n} (P_i + P_{i-1}) \end{aligned} \tag{A.4}$$

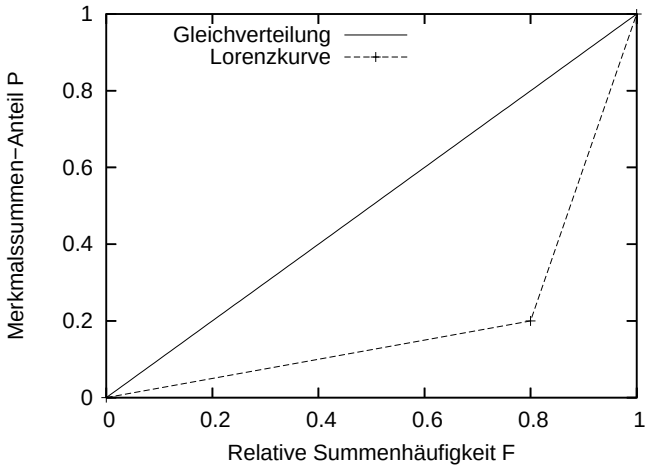


Abbildung A.1: Lorenzkurve für die 80-20-Regel

A.2 Zeitreihenanalyse

Eine Zeitreihe $(x_1, x_2, \ldots, x_n)$ beschreibt die Zuordnung von Daten zu Zeitintervallen, wobei die Zeitintervalle meist konstant sind. Eine Aufgabe der Zeitreihenanalyse ist die Erstellung von Prognosen. Dabei wird untersucht, ob die aufeinander folgenden Werte einer Zeitreihe bestimmten Gesetzmäßigkeiten unterliegen. Ein Beispiel hierfür ist die Erstellung einer Prognose für Börsenkurse.

Zur Bestimmung der Gesetzmäßigkeiten muss ein stochastisches Modell erstellt und die Parameter dieses Modells mittels einer Glättungsfunktion bereinigt werden. In [15] werden Glättungsfunktionen vorgestellt, von denen an dieser Stelle drei näher erläutert werden.

Exponentiell gewichteter gleitender Mittelwert $S(n)$: Mit dieser Glättungsfunktion werden die Historienwerte folgendermaßen gemittelt: je älter die Werte sind, um so weniger Gewicht wird ihnen für die Mittelung zugeteilt. Sei $0 \leq \alpha \leq 1$ ein Glättungsfaktor. Dann wird der exponentiell gewichtete gleitende Mittelwert mittels Gleichung A.5 berechnet, wobei $S(n-1)$ den gewichteten Mittelwert zum vorigen Zeit-

punkt darstellt.

$$S(n) = \alpha x_n + (1 - \alpha)S(n - 1) \tag{A.5}$$

Löst man die Rekursion auf, tritt der exponentielle Charakter zum Vorschein: $S(n) = \alpha x_n + \alpha(1-\alpha)x_{n-1} + \alpha(1-\alpha)^2 x_{n-2} + \cdots + \alpha(1-\alpha)^n x_1$.

Doppelt exponentiell gewichteter gleitender Mittelwert $D(n)$: Diese Glättungsfunktion wird analog zum exponentiell gewichteten gleitenden Mittelwert berechnet, wobei noch eine *Saisonkomponente* hinzukommt. Sei $0 \leq \gamma \leq 1$ ein Glättungsfaktor für den saisonalen Trend. Dann wird der doppelt exponentiell gewichtete gleitende Mittelwert mittels Gleichung A.6 berechnet, wobei $B(n)$ und $B(n-1)$ die Saisonkomponenten zum aktuellen Zeitpunkt bzw. zum vorigen Zeitpunkt darstellen.

$$D(n) = \alpha x_n + (1 - \alpha)(D(n - 1) + B(n) \tag{A.6}$$

$$B(n) = \gamma(D(n) - D(n - 1)) + (1 - \gamma)B(n - 1) \tag{A.7}$$

A.3 Selbstähnlichkeit

Ein System wird selbstähnlich genannt, wenn es in jeder beliebigen Größenordnung ein ähnliches Verhalten aufweist. Dies ist der Fall, wenn die Komponenten solcher Systeme eng miteinander interagieren und somit die Änderung eines kleinen Anteils der Komponenten unter Umständen eine große Auswirkung auf die Gesamtheit der Komponenten haben kann. Diese Tatsache unterscheidet sie von zufälligen Systemen, bei denen sich die Komponenten zufallsgesteuert und nicht abhängig von einander verändern.

Selbstähnlichkeit ist in vielen Systemen zu finden, die Spanne reicht von biologischen über soziale Netze [7, 59], bis hin zum Web [7, 33]. Im Web gibt es relativ wenige hochgradig verlinkte Webseiten, die den Kern des Webs bilden [14].

Grund hierfür ist, dass Betreiber von Webseiten ihre Seiten eher mit bekannten Webseiten verlinken als mit unbekannten. Dieses Verhalten trifft auch auf zusammenhängende Teilbereiche des Webs zu, wie beispielsweise einzelne Web-Auftritte oder so genannte Web-Communities.

A.4 Potenzgesetz und endlastige Verteilungen

Das Potenzgesetz (engl. *power law*) beschreibt die Skaleninvarianz von selbstähnlichen Systemen als polynomielle Abhängigkeiten, wobei die Häufigkeit f, mit der eine Beobachtungsgröße x auftritt, invers proportional zu einer Potenz α eben dieser Größe ist: $f(x) \sim x^{-\alpha}, \alpha > 0$.

Die Verteilungen der Wahrscheinlichkeiten von Beobachtungsgrößen, die dem Potenzgesetz folgen, werden *endlastige Verteilungen* genannt (engl. *heavy-tailed distributions*). Grob gesagt liegt bei diesen Verteilungen der größte Anteil der Wahrscheinlichkeitsmaße (Wahrscheinlichkeitsbelegung) im hinteren Teil der Verteilung. In einer doppel-logarithmischen Grafik, in der sowohl die x-Achse, als auch die y-Achse logarithmisch skaliert ist, wird eine endlastige Verteilung als Gerade dargestellt. Bei nicht logarithmischen Achsen schmiegt sich diese Kurve an die Achsen an. Es gibt also sehr viele Objekte mit einem sehr niedrigen Wert und sehr wenige Objekte mit einem sehr hohen Wert. Als Beispiel für endlastige Verteilungen seien die Zipf-Verteilung, die Pareto-Verteilung und die Lognormal-Verteilung angeführt.

Zipf-Verteilung: Das Zipfsche Gesetz, benannt nach dem Harvard-Linguistik-Professor George Kingsley Zipf (1902-1950), besagt, dass die Häufigkeit $f(i)$ des Auftretens einer in eine Rangfolge gestellten Beobachtungsgröße umgekehrt proportional zu deren Rang ist: $f(i) \sim i^{-\alpha}$, mit $\alpha = 1$. Die Wahrscheinlichkeit P(X=i) wird mit Hilfe der Riemannschen Zeta-Funktion $\zeta(\alpha) = \sum_{n=1}^{\infty} n^{-\alpha}$ berechnet. Eine Verteilung heißt *Zipf-ähnlich*, wenn der Exponent größer als eins ist.

Die **Pareto-Verteilung** wurde nach Vilfredo Pareto (1848-1923) benannt, der Professor für politische Ökonomie an der Universität von Lausanne war und die Verteilung der Einkommen untersuchte. Nach ihm wurde das 80/20-Prinzip benannt, das vor allem im Bereich des Managements angewandt wird. Richard Koch beschreibt in [52], dass beispielsweise 80% des Ertrags von 20% des Aufwands herrühren. Die Pareto-Verteilung wird bezüglich der komplementären kumulativen Verteilungsfunktion $P(X > x) = \left(\frac{x_{\min}}{x}\right)^{-\alpha}$ angegeben.

Die **Doppel-Pareto-Verteilung** folgt ebenfalls dem Potenzgesetz. Während die Pareto-Verteilung in einer doppel-logarithmischen Darstellung als eine gerade Linie dargestellt ist, besteht die Doppel-Pareto-Verteilung aus zwei Linien, die sich in einem Übergangswert schneiden.

Lognormal-Verteilungen gehören streng genommen nicht zu den endlastigen Verteilungen. Eine Zufallsvariable X ist lognormal-verteilt, wenn die Zufallsvariable $Y = \ln(X)$ einer Gaußschen Normalverteilung folgt. Während endlastige Verteilungen in einer log-log-Darstellung eine Gerade bilden, stellen Lognormal-Verteilungen eine Parabel dar, wobei nur der hintere Teil eine Gerade bildet. Sie zeigen somit ein den endlastigen Verteilungen ähnliches Verhalten [72]. Die Lognormal-Verteilung wird bezüglich der komplementären kumulativen Verteilungsfunktion $P(X > x)$ angegeben.

$$P(X > x) = \int_{z=x}^{\infty} \frac{1}{\sigma z \sqrt{2\pi}} e^{-(\ln z - \sigma)^2 / 2\sigma^2} \, \mathrm{dz}$$

Literaturverzeichnis

[1] Swarup Acharya, Rafael Alonso, Michael Franklin, and Stanley Zdonik. Broadcast disks: data management for asymmetric communication environments. In *SIGMOD '95: Proceedings of the 1995 ACM SIGMOD international conference on Management of data*, pages 199–210, New York, NY, USA, 1995. ACM Press.

[2] Swarup Acharya and Stanley B. Zdonik. An efficient scheme for dynamic data replication. Technical report, Providence, RI, USA, 1993.

[3] Eytan Adar and Bernardo A. Huberman. The economics of surfing. *WWW9 Posters Proceedings of the 9th Intl. WWW Conference (Amsterdam)*, 2000.

[4] Martin Arlitt, Diwakar Krishnamurthy, and Jerry Rolia. Characterizing the scalability of a large web-based shopping system. *ACM Transactions on Internet Technology*, 1(1):44–69, 2001.

[5] D. Badrinath, T. Imielinski, R. Frenkiel, and D. Goodman. Nimble: Many-time, many-where communication support for information systems in highly mobile and wireless environments. http://www.cs.rutgers.edu/dataman/nimble/, 1996.

[6] Ricardo Baeza-Yates and Berthier Ribeiro-Neto. *Modern Information Retrieval*. Addison Wesley, 1999.

[7] Albert-László Barabási and Eric Bonabeau. Scale-free networks. *Scientific*

American, May 2003.

[8] Paul Barford and Mark Crovella. Generating representative web workloads for network and server performance evaluation. In *Measurement and Modeling of Computer Systems*, pages 151–160, 1998.

[9] Doug Beeferman and Adam Berger. Agglomerative clustering of a search engine query log. In *KDD '00: Proceedings of the 6th ACM SIGKDD International Conference on Knowledge discovery and data mining*, pages 407–416, New York, NY, USA, 2000. ACM Press.

[10] A. Bestavros. Demand-based document dissemination to reduce traffic and balance load in distributed information systems. In *SPDP '95: Proceedings of the 7th IEEE Symposium on Parallel and Distributeed Processing*, page 338, Washington, DC, USA, 1995. IEEE Computer Society.

[11] Krishna Bharat, Bay-Wei Chang, Monika Henzinger, and Matthias Ruhl. Who links to whom: Mining linkage between web sites. In *Proceedings of IEEE International Conference on Data Mining (ICDM '01), San Jose, California, November 2001.*, 2001.

[12] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker. The implications of zipf's law for web caching, 1998.

[13] Lee Breslau, Pei Cao, Li Fan, Graham Phillips, and Scott Shenker. Web caching and zipf-like distributions: Evidence and implications. In *INFOCOM (1)*, pages 126–134, 1999.

[14] A. Broder, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, and J. Wiener. Graph structure in the web. *Computer Networks*, 33(1-6):309–320, 2000.

[15] Robert Goodell Brown. *Smoothing, Forecasting, and Prediction of Discrete Time Series*. Prentice Hall, 1963.

[16] Susanne Bürklen, Pedro José Marrón, Serena Fritsch, and Kurt Rother-

mel. User centric walk: An integrated approach for modeling the browsing behavior of users on the web. In *Proceedings of the 38th IEEE Annual Simulation Symposium (ANSS'05), San Diego, CA, USA*, 2005.

[17] Susanne Bürklen, Pedro José Marrón, and Kurt Rothermel. An enhanced hoarding approach based on graph analysis. In *Proceedings of the IEEE International Conference on Mobile Data Management (MDM 2004), Berkeley, CA, USA*, pages 358–369, 2004.

[18] Susanne Bürklen, Pedro José Marrón, and Kurt Rothermel. Proactive hoarding in location-based systems. In *Proc. of 2nd Workshop on Context Awareness for Proactive Systems (CAPS 2006), Kassel, Germany*, 2006.

[19] Susanne Bürklen, Pedro José Marrón, Kurt Rothermel, and Timo Pfahl. Hoarding location-based data using clustering. In *MobiWac '06: Proceedings of the international workshop on Mobility management and wireless access*, pages 164–171, New York, NY, USA, October 2006. ACM Press.

[20] Lara D. Catledge and James E. Pitkow. Characterizing browsing strategies in the World-Wide Web. *Computer Networks and ISDN Systems*, 27(6):1065–1073, 1995.

[21] K. Cheverst, N. Davies, K. Mitchell, A. Friday, and C. Efstratiou. Developing a context-aware electronic tourist guide: Some issues and experiences. In *Proceedings of the SIGCHI Conference on Human factors in Computing systems (CHI 2000)*, 2000.

[22] Hyoung-Kee Choi and John O. Limb. A behavioral model of web traffic. In *ICNP '99: Proceedings of the Seventh Annual International Conference on Network Protocols*, page 327, Washington, DC, USA, 1999. IEEE Computer Society.

[23] Andy Cockburn, Saul Greenberg, Steve Jones, Bruce McKenzie, and Michael Moyle. Improving web page revisitation: Analysis, design and evaluation. *IT&Sociecty*, 1(3):159–183, 2003.

[24] Atheros Communications. Power consumption and energy efficiency comparisons of wlan products. http://www.atheros.com/pt/papers.html.

[25] R. Cooley, B. Mobasher, and J. Srivastava. Grouping web page references into transactions for mining world wide web browsing patterns. In *KDEX '97: Proceedings of the 1997 IEEE Knowledge and Data Engineering Exchange Workshop*, page 2, Washington, DC, USA, 1997. IEEE Computer Society.

[26] Thomas H. Cormen, Charles E. Leiserson, and Ronald R. Riverst. *Introduction to Algorithms*. McGraw-Hill, 19947.

[27] Proxim Corporation. Orinoco 11b client pc card. http://www.proxim.com/learn/library/datasheets/11bpccard_A4.pdf.

[28] D. B. Crouch, C. J. Crouch, and G. Andreas. The use of cluster hierarchies in hypertext information retrieval. In *HYPERTEXT '89: Proceedings of the second Annual ACM Conference on Hypertext*, pages 225–237, New York, NY, USA, 1989. ACM Press.

[29] Mark E. Crovella and Azer Bestavros. Self-similarity in world wide web traffic: evidence and possible causes. *IEEE/ACM Transactions on Networking (TON)*, 5(6):835–846, 1997.

[30] J. Czyczowicz, E. Kranakis, D. Krizanc, A. Pelc, and M. Martin. Enhancing hyperlink structure for improving web performance. *Journal of Web Engineering*, 1(2):93–127, 2003.

[31] Debora Donato and Stefano Leonardi and Stefano Millozzi and Panayiotis Tsaparas. Mining the inner structure of the Web. In *Proc. of International Worshop on the Web and Databases (WebDB'05*, pages 145–150, June 2005.

[32] Michelangelo Diligenti, Marco Gori, and Marco Maggini. Web page scoring systems for horizontal and vertical search. In *Proceedings of the eleventh*

international conference on World Wide Web, pages 508–516. ACM Press, 2002.

[33] Stephen Dill, S. Ravi Kumar, Kevin S. McCurley, Sridhar Rajagopalan, D. Sivakumar, and Andrew Tomkins. Self-similarity in the web. In *The VLDB Journal*, pages 69–78, 2001.

[34] Ronald C. Dodge, Daniel A. Menascé, and Daniel Barbará. Testing e-commerce site scalability with tpc-w. In *Proc. 2001 Computer Measurement Group Conference, Anaheim, CA, December 2-7, 2001*, 2001.

[35] Magdalini Eirinaki and Michalis Vazirgiannis. Web mining for web personalization. *ACM Trans. Inter. Tech.*, 3(1):1–27, 2003.

[36] Federico Michele Facca and Pier Luca Lanzi. Mining interesting knowledge from weblogs: A survey. *Data & Knowledge Engineering*.

[37] Reginald Ferber. *Information Retrieval - Suchmodelle und Data-Mining-Verfahren für Textsammlungen und das Web*. 2003.

[38] Erich Gamma, Richard Helm, Ralph E. Johnson, and John Vlissides. *Entwurfsmuster. Elemente wiederverwendbarer objektorientierter Software*. Addison Wesley, 2004.

[39] Steven Glassman. A caching relay for the World Wide Web. *Computer Networks and ISDN Systems*, 27(2):165–173, 1994.

[40] Jim Griffioen and Randy Appleton. Reducing file system latency using a predictive approach. In *USENIX Summer*, pages 197–207, 1994.

[41] Martin Halvey, Mark T. Keane, and Barry Smyth. Mobile web surfing is the same as web surfing. *Commun. ACM*, 49(3):76–81, 2006.

[42] D. He and A. Göker. Detecting session boundaries from web user logs. In *Proceedings of the BCS-IRSG 22nd Annual Colloquium on Information Retrieval Research*, 2000.

[43] Monika R. Henzinger, Allan Heydon, Michael Mitzenmacher, and Marc Najork. Measuring index quality using random walks on the web. *Computer Networks: The International Journal of Computer and Telecommunications Networking*, 31(11-16):1291–1303, 1999.

[44] F. Hohl, U. Kubach, A. Leonhardi, K. Rothermel, and M. Schwehm. Next century challenges: Nexus – an open global infrastructure for spatial-aware applications. In T. Imielinski and M. Steenstrup, editors, *Proceedings of the Fifth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom 1999)*, pages 249–255, Seattle, Washington, USA, August 1999.

[45] Xiangji Huang, Fuchun Peng, Aijun An, and Dale Schuurmans. Dynamic web log session identification with statistical language models. *J. Am. Soc. Inf. Sci. Technol.*, 55(14):1290–1303, 2004.

[46] Bernardo A. Huberman, Peter L. T. Pirolli, James E. Pitkow, and Rajan M. Lukose. Strong regularities in World Wide Web surfing. *Science*, 280(5360):95–97, 1998.

[47] Wavelinx Inc. Wavelinx pcmcia card gpc-6210 (1x ev-do + cdma2000). http://www.wavelinx.co.kr/korean/products_content.asp?goods_no=38.

[48] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM Comput. Surv.*, 31(3):264–323, 1999.

[49] Anil K. Jain, Alexander Topchy, Martin H. C. Law, and Joachim M. Buhmann. Landscape of clustering algorithmss. In *Proc. of the 17th Int. Conference on Pattern Recognition (ICPR'04)*, pages 260–263, 2004.

[50] Xin Jin, Yanzan Zhou, and Bamshad Mobasher. Web usage mining based on probabilistic latent semantic analysis. In *KDD '04: Proceedings of the tenth ACM SIGKDD International Conference on Knowledge discovery and data mining*, pages 197–205, New York, NY, USA, 2004. ACM Press.

[51] James J. Kistler and M. Satyanarayanan. Disconnected operation in the coda file system. *ACM Transactions on Computer Systems (TOCS)*, 10(1):3–25, 1992.

[52] Richard Koch. *The 80/20 principle. The secret of achieving more with less.* 1997.

[53] D. Krishnamurthy and J. Rolia. The internet vs e-commerce servers: when will server performance matter? In *CASCON '98: Proceedings of the 1998 conference of the Centre for Advanced Studies on Collaborative research*, page 14. IBM Press, 1998.

[54] Uwe Kubach. *Vorabuebertragung ortsbezogener Informationen zur Unterstützung mobiler Systeme.* PhD thesis, IPVS, Universität Stuttgart, 2002.

[55] Uwe Kubach and Kurt Rothermel. Exploiting location information for infostation-based hoarding. In *Proc. of the 7th Annual International Conference on Mobile Computing and Networking (MobiCom 2001)*, pages 15–27. ACM Press, 2001.

[56] Geoffrey H. Kuenning, Wilkie Ma, Peter Reiher, and Gerald J. Popek. Simplifying automated hoarding methods. In *Proceedings of the 5th ACM International workshop on Modeling analysis and simulation of wireless and Mobile systems*, pages 15–21. ACM Press, 2002.

[57] Geoffrey H. Kuenning, Wilkie Ma, Peter Reiher, and Gerald J. Popek. Simplifying automated hoarding methods. In *Proceedings of the 5th ACM International workshop on Modeling analysis and simulation of wireless and Mobile systems*, pages 15–21. ACM Press, 2002.

[58] Geoffrey H. Kuenning and Gerald J. Popek. Automated hoarding for mobile computers. In *Proceedings of the sixteenth ACM Symposium on Operating Systems Principles*, pages 264–275. ACM Press, 1997.

[59] Ravi Kumar, Prabhakar Raghavan, Sridhar Rajagopalan, and Andrew

Tomkins. The web and social networks. *Computer*, 35(11):32–36, 2002.

[60] S. Ravi Kumar, Prabhakar Raghavan, Sridhar Rajagopalan, and Andrew Tomkins. Extracting large-scale knowledge bases from the web. In *The VLDB Journal*, pages 639–650, 1999.

[61] Kwong Lai and Zahir Tari and Peter Bertok. Supporting Disconnected Operations Through Cooperative Hoarding. In *Proc. of International Conference on Computer Communications and Networks (ICCCN)*, October 2005.

[62] Hui Lei and Dan Duchamp. An analytical approach to file prefetching. In *1997 USENIX Annual Technical Conference*, Anaheim, California, USA, 1997.

[63] Anton Leuski. Evaluating document clustering for interactive information retrieval. In *CIKM '01: Proceedings of the tenth international conference on Information and knowledge management*, pages 33–40, New York, NY, USA, 2001. ACM Press.

[64] Mark Levene, Jose Borges, and George Loizou. Zipf's law for web surfers. *Knowledge and Information Systems*, 3(1):120–129, 2001.

[65] Keqiu Li, Hong Shen, Francis Y. L. Chin, and Si Qing Zheng. Optimal methods for coordinated enroute web caching for tree networks. *ACM Transactions on Internet Technology*, 5(3):480–507, 2005.

[66] Zhihong Lu and Kathryn S. McKinley. Partial collection replication versus caching for information retrieval systems. In *SIGIR '00: Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 248–255, New York, NY, USA, 2000. ACM Press.

[67] Bruce A. Mah. An empirical model of http network traffic. In *INFOCOM '97: Proceedings of the INFOCOM '97. Sixteenth Annual Joint Conference*

of the IEEE Computer and Communications Societies. Driving the Information Revolution, page 592, Washington, DC, USA, 1997. IEEE Computer Society.

[68] M. J. Mana-López, M. De Buenaga, and J. M. Gómez-Hidalgo. Multidocument summarization: An added value to clustering in interactive retrieval. *ACM Transactions on Information Systems*, 22(2):215–241, 2004.

[69] Marik Marshak and Hanoch Levy. Evaluating web user perceived latency using server side measurements. *Computer Communications*, 26(8):872–887, May 2003.

[70] Daniel A. Menascé. Tpc-w: A benchmark for e-commerce. *IEEE Internet Computing*, 6(3):83–87, 2002.

[71] Gerhard Merziger and Thomas Wirth. *Repetitorium der höheren Mathematik*. Binomi, fourth edition, 1999.

[72] Michael Mitzenmacher. A brief history of generative models for power law and lognormal distributions. Preprint (EECS, Harvard Univ), 2002.

[73] B. Mobasher, H. Dai, and M. Tao. Discovery and evaluation of aggregate usage profiles for web personalization. *Data Mining and Knowledge Discovery*, 6:61–82, 2002.

[74] B. Mobasher, N. Jain, E. Han, and J. Srivastava. Web mining: Pattern discovery from world wide web transactions, 1996.

[75] Brian H. Murray. Sizing the internet (cyveillance white paper). http://www.cyveillance.com/web/downloads/Sizing_the_Internet.pdf.

[76] J Liu N Zhong and Y Yao (eds). *Agent-Based Characterization of Web Regularities*, chapter 2, pages 19–36. Springer Verlag, 2003.

[77] Alexandros Nanopoulos, Dimitrios Katsaros, and Yannis Manolopoulos. A data mining algorithm for generalized web prefetching. *IEEE Transactions*

on Knowledge and Data Engineering, 15(5):1155–1169, 2003.

[78] Daniela Nicklas, Matthias Großmann, Thomas Schwarz, and Bernhard Mitschang. A model-based, open architecture for mobile, spatially aware applications. In *SSTD '01: Proceedings of the 7th International Symposium on Spatial and Temporal Databases*, pages 117–135, Berlin, Heidelberg, New York, July 2001. Springer-Verlag.

[79] Jakob Nielsen. Top ten mistakes: Revisited ten years later. http://www.useit.com/alertbox/990502.html, 1999.

[80] Venkata N. Padmanabhan and Jeffrey C. Mogul. Using predictive prefetching to improve World-Wide Web latency. In *Proceedings of the ACM SIGCOMM '96 Conference*, Stanford University, CA, 1996.

[81] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project, 1998.

[82] Timo Pfahl. *Diplomarbeit Nr. 2359: Entwicklung von Verfahren zur Cluster-Bildung in einem Informationsgraphen für die Vorabübertragung vonWebseiten*. PhD thesis, IPVS, Universität Stuttgart, 2005.

[83] Dimitrios Pierrakos, Georgios Paliouras, Christos Papatheodorou, and Constantine D. Spyropoulos. Web usage mining as a tool for personalization: A survey. *User Modeling and User-Adapted Interaction*, 13(4):311–372, 2003.

[84] James E. Pitkow and Peter Pirolli. Mining longest repeating subsequences to predict world wide web surfing. In *USENIX Symposium on Internet Technologies and Systems*, 1999.

[85] Stefan Podlipnig and Laszlo Böszörmenyi. A survey of web cache replacement strategies. *ACM Comput. Surv.*, 35(4):374–398, 2003.

[86] William J. Reed and Murray Jorgensen. The double pareto-lognormal

distribution - a new parametric model for size distributions. In *Proceedings of the Int. Conf. on Distribution Theory, Order Statistics and Inference*, 2004.

[87] Qun Ren and Margaret H. Dunham. Using semantic caching to manage location dependent data in mobile computing. In *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking (MobiCom 2000)*, pages 210–221. ACM Press, 2000.

[88] A. Reyes-Lecuona, E. González-Parada, E. Casilari, and A. Díaz-Estrella. A page-oriented www traffic model for wireless system simulations. In *Proceddings of the 16th International Teletraffic Congress (ITC'16), Edinburgh, UK*, pages 1271–1280, 1999.

[89] M. Satyanarayanan, James J. Kistler, Lily B. Mummert, Maria R. Ebling, Puneet Kumar, and Qi Lu. Experience with disconnected operation in a mobile environment. In USENIX, editor, *Proceedings of the USENIX Mobile and Location-Independent Computing Symposium: August 2–3, 1993, Cambridge, Massachusetts, USA*, pages 11–28, Berkeley, CA, USA, August 1993. USENIX.

[90] M. Satyanarayanan, J.J. Kistler, P. Kumar, M.E. Okasaki, E.H. Siegel, and D.C. Steere. Coda: A highly available file system for a distributed workstation environment. *IEEE Transactions on Computers*, 39(4):447–459, 1990.

[91] Eugene Shih, Seong-Hwan Cho, Nathan Ickes, Rex Min, Amit Sinha, Alice Wang, and Anantha Chandrakasan. Physical layer driven protocol and algorithm design for energy-efficient wireless sensor networks. In *MobiCom '01: Proceedings of the 7th annual international conference on Mobile computing and networking*, pages 272–287, New York, NY, USA, 2001. ACM Press.

[92] Swaminathan Sivasubramanian, Michal Szymaniak, Guillaume Pierre, and

Maarten van Steen. Replication for web hosting systems. *ACM Comput. Surv.*, 36(3):291–334, 2004.

[93] Johan Skold, Magnus Lundeval, Stefan Parkvall, and Magnus Sundelin. Broadband data performance of third-generation mobile systems. *Ericsson Review*, 1, 2005.

[94] Jaideep Srivastava, Robert Cooley, Mukund Deshpande, and Pang-Ning Tan. Web usage mining: discovery and applications of usage patterns from web data. *SIGKDD Explor. Newsl.*, 1(2):12–23, 2000.

[95] Illya Stepanov, Jörg Hähner, Christian Becker, Jing Tian, and Kurt Rothermel. *A Meta-Model and Framework for User Mobility in Mobile Networks*. In *Proceedings of the 11th International Conference on Networking 2003 (ICON 2003), Sydney, Australia, September 28 - October 1, 2003*, pages 231–238. Universität Stuttgart, Fakultät Informatik, Elektrotechnik und Informationstechnik, IEEE, September 2003. ISBN 0-7803-7788-5.

[96] Illya Stepanov, Pedro José Marrón, Serena Fritsch, and Kurt Rothermel. Mobility modeling of outdoor scenarios for manets. In *Proceedings of the 38th IEEE Annual Simulation Symposium (ANSS'05), San Diego, CA, USA*, 2005.

[97] Illya Stepanov and Kurt Rothermel. *Simulating Mobile Ad-Hoc Networks in City Scenarios*. In T. Braun, G. Carle, S. Fahmy, and Y. Koucheryavy (Eds.), editors, *Proceedings of the 4th International Conference on Wired/Wireless Internet Communications (WWIC 2006), Bern, Switzerland, May 2006*, pages 1–12. Universität Stuttgart, Fakultät Informatik, Elektrotechnik und Informationstechnik, Springer, Mai 2006. ISBN 3-540-34023-8.

[98] K. Tan and B. Ooi. *Data Dissemination in Wireless Computing Environments*. Kluwer Academic Publishers, 2000.

[99] Linda Tauscher and Saul Greenberg. How people revisit web pages: Empirical findings and implications for the design of history systems. *Inter-*

national Journal of Human Computer Studies, 47(1):97–137, 1997.

[100] Manghui Tu, Peng Li, and I-Ling Yen. Transaction based dynamic partial replication in mobile environments. In *IPDPS '04: Proc. of 18th Intl. Parallel and Distributed Processing Symposium*, 2004.

[101] N. J. Tuah, M. J. Kumar, and S. Venkatesh. Performance modelling of speculative prefetching for compound requests in low bandwidth networks. In *WOWMOM '00: Proceedings of the 3rd ACM international workshop on Wireless mobile multimedia*, pages 83–92, New York, NY, USA, 2000. ACM Press.

[102] Alec Wolman, M. Voelker, Nitin Sharma, Neal Cardwell, Anna Karlin, and Henry M. Levy. On the scale and performance of cooperative web proxy caching. In *SOSP '99: Proceedings of the 17th ACM Symposium on Operating Systems Principles*, pages 16–31, New York, NY, USA, 1999. ACM Press.

[103] Jiong Yang, Wei Wang, and Richard Muntz. Collaborative web caching based on proxy affinities. In *SIGMETRICS '00: Proceedings of the 2000 ACM SIGMETRICS Int. Conf. on Measurement and Modeling of Computer Systems*, pages 78–89, New York, NY, USA, 2000. ACM Press.

[104] Tao Ye, H.-Arno Jacobsen, and Randy Katz. Mobile awareness in a wide area wireless network of info-stations. In *Proceedings of the fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom 1998)*, pages 109–120. ACM Press, 1998.

[105] Haobo Yu, Lee Breslau, and Scott Shenker. A scalable web cache consistency architecture. In *SIGCOMM '99: Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, pages 163–174, New York, NY, USA, 1999. ACM Press.

[106] Jinsuo Zhang, Abdelsalam (Sumi) Helal, and Joachim Hammer. Ubidata: ubiquitous mobile file service. In *Proceedings of the 2003 ACM Symposium*

on Applied Computing, pages 893–900. ACM Press, 2003.

[107] Wei Zhang, David B. Lewanda, and Christopher D. Jannek. Personalized web prefetching in mozilla. Technical Report LU-CSE-03-006, 2003.