GraphTiles: A Visual Interface for Supporting Imprecise Mobile Search

Figure 1. Comparing *GraphTiles* with IMDb's mobile website. (a) and (b): The MPM(Movie-person-movie) *QueryType*; (c) and (d): the PMP(Person-movie-person) *QueryType*.

ABSTRACT

Mobiles are generating a rapidly increasing proportion of search queries, ranging from specific fact-finding to unstructured exploration. Yet, search interfaces have not changed significantly to accommodate mobile constraints. Mobile search can be particularly challenging when traversing and learning about data relationships, such as those described in IMDb [2] and LinkedIn [3]. We examined the prevalence of these mobile search use cases in a two-week diary study. We hypothesized that the ability to view a link neighborhood around the search result could be quite helpful, particularly for more imprecise, open queries. For example, one user reported searching for the location where 'The sisterhood of the traveling pants' was filmed. Although their search was successful, it was difficult. A visual overview of such relationships could have been very helpful. To provide such overviews, we designed *GraphTiles*, a visual interface supporting mobile search. In an experimental evaluation, users found information more quickly with *GraphTiles* than with a standard mobile site.

Author Keywords

Mobile search, connected data, imprecise queries.

ACM Classification Keywords

H.5.m. Information Interfaces and Presentation (e.g. HCI): Miscellaneous

INTRODUCTION

Mobile search is becoming a prevalent everyday activity for millions of users to search, locate and discover information on-the-go. Cui and Roto [12] and Church and Oliver [11] found that mobile users effectively limit their expectations by posing either fairly specific *direct* queries or extremely broad ones. Fact-finding, where the user searches for a specific piece of information, and browsing more open subject-based exploration, are examples of these kinds of tasks.

However, mobile users may have difficulty articulating a precise query due to the lack of knowledge about the content and the limited query capability of the interface [16]. This can result in the users having to reformulate their queries several times on the basis of intermediate query results. For example, a user may have in mind a specific actor she wishes to find on her mobile device. Her approach to search is influenced by whether she remembers the name of the actor (in

which case she would directly search by name) or she might instead search by some combination of movie and actor to find whom she's looking for. This would lead to several iterations of search to finally find the answer. This"no man's land" of *imprecise* search that is neither very open nor very specific remains unsupported. Lee *et al.* [19] points out that these imprecise queries are more common than we might think. In a typical case, a user knows what they seek, but does not know the keyword that would retrieve it.

Leveraging the relationships between entities in search, whether searching for an actor in a graph of movies to cast in IMDb [2], songs to artists in Pandora [4] or a network of friends in Facebook [1], not only helps in structuring search results, but as a tool to guide users in reformulating queries that are typical of imprecise search. This approach not only reduces cognitive load through recognition, but allows users to navigate search results by attribute values rather than by keyword alone [15].

Mobile search interfaces offer little more than traditional desktop search, with most using simple adaptations of their web interfaces to meet the display characteristics of mobile handsets. Yet, the primary method of web navigation remains hyperlinks [8]. These are well-suited for depth-first traversal, where a user first selects a link on the current page, which in turn loads a new page. This process repeats until the user finds the needed information or the current search path is abandoned returning the user to the initial search. Some search engines offer partial solutions by showing autocompleted keyword sets in real time (e.g. Bing's search suggestions), or by displaying related facts from a database (e.g. Google's Knowledge Graph). Even so, users are often forced to perform several searches to help them find the appropriate keywords for their search.

In this paper, we present *GraphTiles*, a visual search interface for mobile devices, to help users perform imprecise queries. The interface leverages the neighborhood of entities and their relationships in a content graph, displaying an incomplete portion of a local link neighborhood: a thumbnail of the current page alone in the left column, some pages one link away in the middle column, and other pages two links away in the right column. To see the complete neighborhood, users can scroll the central and right columns vertically.

While this layout implies many links, it does not indicate exactly where the links are between the second and third columns. Users can reveal these locations by selecting a page thumbnail from these columns, triggering an *interactive reordering* that highlights pages linked to the selection and places them onscreen or nearly so. Users can restore the original (better known) order by deselecting the page.

CONTRIBUTIONS

The main contributions of this paper are:

- (vidya: I would downplay or omit this one)Examining the prevalence of mobile search use cases in a two-week diary study. By categorizing the different facets of the search behavior in the study, we identify problems that the participants encountered, particularly around imprecise, open queries.
- Implementing a system specifically designed to support open, imprecise and near miss mobile searches.
- Evaluation and findings of how well mobile users find information using GraphTiles vs. a standard mobile website, and implications for design of such systems to better support mobile search.

RELATED WORK

Mobile search applications often take advantage of user context such as location and time, to provide a localized experience. Lymberopoulos *et al.* apply a data-driven approach where local search model at different levels of location granularity (e.g. city, state, country) are combined together to improve click prediction accuracy in the search results [20]. *FindAll* is a local mobile search engine that lets users search and retrieve web pages, even in the absence of connectivity. The premise for their work is that mobile users often search for web pages that they have previously visited, known as re-finding. *FindAll* estimates the benefits of local search, by learning the re-finding behavior of users [7]. *Hapori*, a local mobile search tool, not only takes into account location in the search query but richer context such as the time, weather and the activity of the user [18]. Amini *et al.* present Trajectory-Aware Search (TAS) that predicts the user's destination based on location data from the current trip and shows search results near the predicted location [6]. SocialSearchBrowser incorporates social networking capabilities with key mobile contexts to improve the search and information discovery experience of mobile users [10].

GraphTiles is essentially a visual search interface of the local link graph. There has been little work specifically addressing mobile visualization [9], and even less work offering techniques specialized toward the visualization of graphs on mobile devices to help in imprecise search. Karstens [17] proposes node-link diagrams of hierarchies arranged around a rectangle to make efficient use of display space. He displayed nearly 1000 nodes, each represented with a very small circle. Hao and Zhang [14] propose a space-filling sunburst display of hierarchies. Their larger nodes are easier to interact with, but their graphs are much smaller. Pattath *et al.* [21] visualize general graphs numbering just a few dozen nodes using node-link diagrams. Finally, in work most closely related to our own, Da Lozzo *et al.* [13] use node-link diagrams centered around a specific node, again with very small nodes.

To recognize mobile constraints, we limit visualization to a graph neighborhood as do Da Lozzo *et al.*, but like Hao and Zhang, we display many links implicitly.

DIARY STUDY

In order to further understand how people perform search with mobile devices in everyday use, we ran a two-week diary study which collected individual search information. We provided each participant with a diary booklet to keep a history of their online searches. They were allowed to change keywords if necessary and redo the search. 32 people (21

college students, age 18–62, 17 male, 15 female) participated in the experiment. All had normal or corrected-normal vision. They were required to have a mobile device capable of search, and to be regular users of that functionality. 10 participants had iOS, 11 had Android, and one participant had a Windows phone. Others did not report their system.

These were the questions on each page of the diary that participants answered as soon as they performed a search.

1. Date
2. Time
3. Duration of search task
4. What app or website did you access
5. What were you searching for?
6. Did you find what you were searching for at all? YES/NO
7. If you did find your information, please continue by filling in the blanks with numbers: I performed _ searches to find my information. I followed _ links after leaving the search results page.
8. Rate the difficulty of finding your information from 1–5 with 5 being very difficult. Add text to explain if you like.

Results and Discussion

We collected 868 search entries with an average of 27 entries per person. The average time taken to record a search activity was 131.6 seconds ($median = 120, max = 1200$). 9% of searches (33 out of 868) failed, not providing users with the information they sought. About a third (279 out of 868 search entries) were difficult (opposed to 'easy' described below). 41.2% used Google app, 49.4% used a specific url, and the rest used other apps (i.e Amazon app). Participants performed an average of 1.2 searches ($median = 1, min = 1, max = 5$) and followed 2.5 links ($median = 2, min = 0, max = 39$) to find their information. Participants evaluated the task difficulty at an average of 1.9 ($median = 2, stdev = 1, min = 1, max = 5$) based on a Likert scale with 1 being very easy and 5 being very difficult.

We looked for duplicate themes from collected diary entries and did not start with a set of fixed categories. We adopted most of the categories from [22] and added new categories such as '*how-to*', '*unit conversion*', '*definition*', and '*name*'. There are 17 categories based on the diary entries. The *how-to* category includes information about practical advice and detailed instruction of an activity. The *unit conversion* includes kilometer to miles conversion, fahrenheit to celsius, and exchange rate. The *definition* includes any need of terminology definition and detailed explanation. The *name* includes search of a certain person or movie title. The largest category of collected entries was *trivia* (51%). They are the random thoughts from the participants such as "story of shutter island". The second highest was *shopping* (13%), followed by *point of interest* (10%) and *definition* (6%).

We find several interesting discoveries from our study. People use applications more than websites for certain activities. On an average, 18.5% (160 out of 868) of the searches were using applications than web portals. The most popular category searched using an application was *shopping* (47%). Surprisingly, participants frequently knew the exact keyword to put in as a search phrase. People use mobile apps for regular, standard search activities (i.e. weather, shopping, location, names, exchange rate, bus route, definition). For less frequent, more complicated searches, they will try it on a web portal but easily give up because of the screen size restriction and choose another method (i.e. go to their desktop, ask friends).

We divided searches into three categories beside success and fail. By collecting the number of keyword changes, the number of links, the time needed to finish a search, and the perceived difficulty rating, we measured the *difficulty* level (easy/hard) of each search query. We define 'easy' queries to be 'at most one query made' and 'perceived difficulty rating 3 or lower' and 'open query or taken time lesser than 3 minutes'. Otherwise, a query is difficult.

We define open queries to be 'two or more searches performed' or 'information found does not fit into one typical webpage'. Open queries include broader subject-based explorations. Finally, we divided successful searches into hits or misses to reflect how hard users had to work to find their desired information. When users only had to follow one link, the search was a hit.

Figure 2 categorizes search type of diary entries by success/failure, easy/difficult, open/specific queries, and hit or miss. Overall, the large blue section shows that people are mostly successful in finding information with their mobile device (though they may never attempt many challenging searches). About a third of searches are still difficult, and over half of difficult searches are open.

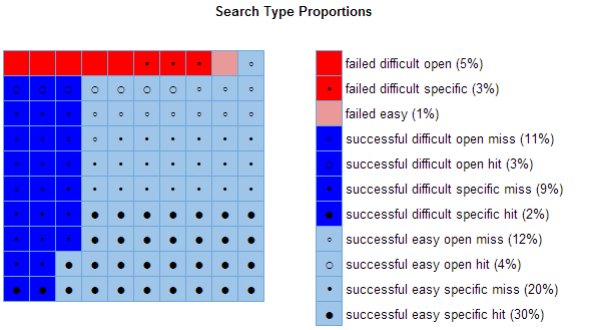


Figure 2. Search type by success/failure, easy/difficult, open/specific, and hit/miss. Blue: success, red: failure, saturated color: easy, bright color: difficult, empty dot: open, filled dot: specific, smaller dot: miss, bigger dot: hit

Although search was usually successful, it was difficult about a third of the time, especially when search was more open and exploratory. Even among easy searches, over half were near misses requiring several clicks to find the needed information. In sum, we believe the large majority of searches could have benefited from a tool that helped users navigate through the information neighborhoods typical of open and near miss searches.

THE GRAPHTILES INTERFACE

We designed *GraphTiles* to help mobile search users handle three challenging search types: open, exploratory search, where information foraging behavior is opportunistic and direction is weak at best; ill-formed search, where direction is known but difficult to articulate; and near miss searches, where the direction can be articulated, but the search engine leaves users some distance away from desired information. All of these search types would benefit from an interface that displays some of the local information neighborhood, allowing users to approach their information more directly. We designed *GraphTiles* to be such an interface.

With *GraphTiles* (Figure 1 (a),(c)), we assume that users will employ search to find a locality of concern around a central node (e.g. for IMDb, "near John Wayne"). As discussed above, position in the layout reflects link distance from the center. When necessary, users can drag a non-central node to the left to change the central node.We display links largely implicitly: every node in the middle column has an implied link to the central node, and every node in the right column is reachable from the middle column. To represent links between the middle and right columns we support both explicit link display and interactive reordering around a selected link. Explicit links appear only when both linked nodes are currently displayed. Reordering has the added benefit of accelerating access to off-screen nodes.

We considered a circular (or rectangular) layout to make better use of the blank space in the left column, with a scroll around the central node rather than along it, but discarded it so that we could provide a glimpse of a larger two-link neighborhood. A circular layout with a two-link neighborhood would require much smaller nodes (difficult to touch with a finger tip), and would fit poorly in rectangular mobile displays.

IMDb is largely a bipartite graph, with movies linking to crew members and vice versa. *GraphTiles* quite appropriately exploits this in its interface, putting movies in one column, crew in the next, and assuming that there are no links between nodes in the same column. When graphs are not bipartite, representing within-column links explicitly is awkward at best. We generalize *GraphTiles* to all graphs by relying on interactive reordering alone, and demonstrate this with a second database, the Seattle Band Map.

EXPERIMENT: COMPARISON TO IMDB'S MOBILE SITE

As a summative evaluation, we compared *GraphTiles* to IMDb's mobile site. In *GraphTiles*, we used both interactive reordering and the best explicit link representation, —connecting nodes with lines —a result from our other experiment (below).

Our experiment focused on helping users answer ill-formed queries. In the context of IMDb, users often want to recommend a movie to a friend, but cannot remember the name of that movie, nor the name of any actors in that movie. They do however know that one of the actors in the movie was also in a movie they know. They move from movie to actor to movie. Movie buffs also often try their hand at casting future films. They cannot remember the actor's name, nor the name of the movie they remember them from. But, they do remember the name of a second actor in that movie. Thus, they move from actor to movie to actor. We focused on answering imprecise queries of the same type. (We assumed that IMDb's mobile web app is a better solution for more precise queries such as "The movies that John Wayne has acted in").

Figure 1 shows a comparison of the visuals used in *GraphTiles* and IMDb's mobile website to answer the movie-person-movie(MPM) and person-movie-person(PMP) *QueryTypes*. We expected that *GraphTiles* would allow users to find answers to imprecise queries more quickly than IMDb's web app.

Methods

We compared *GraphTiles* to IMDb's mobile site in an experiment with twenty participants, all of them employees at a large corporate research center. Each participant performed 120 information seeking tasks, using the same graph neighborhoods we used in our first experiment.

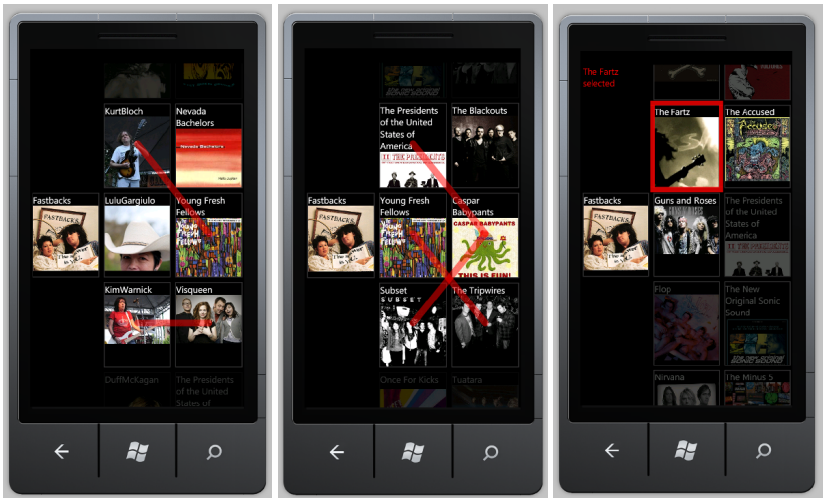
We used a fully crossed within subjects 2×2 design. As participants performed the tasks, we systematically altered two variables. *Interface*, or the tool used to access the IMDb information, had two levels: *GraphTiles* and the IMDb web app. *QueryType* had two levels: a movie-person-movie (MPM) query or a person-movie-person (PMP) query. If *QueryType* was MPM, we asked participants to find the person who worked in two given movies. In this case, the central node at the left of the visualization was always a movie. If *QueryType* was PMP, we asked participants to find the movie on which two given people collaborated. In this case, the central node at the left of the visualization was always a person. To answer the question, participants used a phone to scroll in the right column to find the second person's node, and then scroll in the middle column to find the movie connecting the two people, and select it.

In addition to displaying link lines, *GraphTiles* here implemented interactive reordering, which highlights nodes' links to a selected node and moves them onscreen. Every participant performed 30 trials with each of the $2 \times 2 = 4$ experimental treatments. We grouped trials by *Interface* into two blocks of 60 trials each. Thus participants performed all trials with the current Interface before moving on to the next. To combat the effects of fatigue and learning, we used complete counterbalancing across participants: half of them performed the *GraphTiles* block first, the other half the web app block first. Within each of these blocks, we randomly ordered the levels of *QueryType*. We randomized the order of graph neighborhoods without replacement, so that each participant saw each neighborhood exactly once.

Apparatus

We implemented our interface on three Samsung SGH-i917 phones running Windows Phone 7.5, with an AMOLED display and a full capacitive touch screen. The monitor used to display questions was a 1920×1200 pixel Dell 24". Participants interacted with the visualization on a phone by scrolling with a swipe gesture or selecting nodes with a long tap.

We obtained our IMDb graph neighborhoods using the official IMDb API, obtaining a large cross section of its database



(a) Band and artist relationship

(b) Band relationship

(c) Interactive reordering with dimmed image when not related.

Figure 3. Applying *GraphTiles* to Seattle's music band data.

(approximately 3GB in size). We then randomly selected 60 nodes within the IMDb graph describing well known actors (supporting PMP queries), and 60 nodes describing well known movies (supporting MPM queries). We then sampled the two-link neighborhood around each actor (PMP) node by adding the top movies linked to it as indicated by IMDb's own API call; and then for each of those top movies, adding its top actors, again as indicated by IMDb's API call. We created two-link neighborhoods around movie (MPM) nodes similarly. The number of top movies returned by IMDb's API was generally much lower than the number of top actors.

Results and discussion

All participants performed all trials correctly, so we report only completion times here. We tested significance using a two-factor repeated measures ANOVA. Only the two single variable effects were significant; they did not interact.

When using *GraphTiles*, participants were significantly ($F(1, 19) = 2291.833$, $p < 0.001$) faster than when using the IMDb web app. Average completion time with *GraphTiles* was 18.2s (SD 5.27), while with IMDb web app, it was 31.5s (SD 5.26).

Although its effect was significant ($F(1, 19) = 11.27$, $p < 0.005$), *QueryType*'s effect was not meaningful. The difference in completion times when participants looked for movies rather than persons was 0.6s: (25.0s for movies, 24.4s for persons).

Results in fact exceeded our expectations, with *GraphTiles* users almost twice as fast as IMDb web app users. *GraphTiles* was designed for imprecise queries; IMDb probably was not. What remains to be seen is whether or not a single interface can support both precise and imprecise queries well.

GRAPHTILES IN NON-BIPARTITE GRAPHS

While real and practical, the IMDb graph is bipartite: nodes contain either people (e.g. actors) or movies. *GraphTiles* quite appropriately exploits this structure, placing people and movies in different columns. However if *GraphTiles* is to find use with more general applications, it must be tested with non-bipartite graphs.

With this goal in mind, we used *GraphTiles* to the Seattle Band Map [5]. In this database, music bands from the Pacific Northwest are linked if they share band members or have collaborated with one another. By preprocessing the database, we could create a bipartite graph of musicians and bands (Figure 3(a)), but that is not our purpose here.

Figure 3(b) shows a non-bipartite band-band layout using lines to represent links. The challenge here is representing links that start and end within the same *GraphTiles* column, which do not exist in bipartite graphs. Lines and most of the other explicit link representations we discussed perform poorly in such cases, since they are only displayed when both endpoints are onscreen, which will happen only rarely within the same column.

We believe interactive reordering is the best solution to this problem. In Figure 3(c), the user selects the band 'The Fartz', bringing all related bands onscreen or nearly so. Unrelated bands are dimmed out in the interface to further accentuate band connections.

EXPERIMENT: COMPARISON OF EXPLICIT LINK REPRESENTATIONS

We considered how to display links explicitly on the mobile screen. It might be tempting simply to draw lines between linked nodes (Figure 4c), but *GraphTiles* has unique characteristics that could make this solution untenable. As users scroll, nodes appear and disappear, meaning that linking lines do as well. Scrolling also causes the lines to move when they

are onscreen, occluding a variety of other nodes and dynamically relocating link crossings (making a well-known drawback of link lines still worse). All of this dynamic behavior does not exist in most graph visualizations and could be quite disorienting during mobile search. .

In creating alternative designs for displaying explicit links, we were (loosely) inspired by the grouping principles of Gestalt psychology [23]. The *proximity* principle places nearby items in the same group. Because we could not use proximity alone to display complex many-to-many relationships, we approximated proximity with an iconic representation of the neighboring column (Figure 4(d)). Rectangles in the representation indicate the presence of links to the node in the same position in the neighboring column. *Similarity* groups items that have similar properties such as color or texture (Figure 4(b) and 4(e)). Here, nodes containing the outline color or a thumbnail of a neighboring node are linked to that node. Like link lines (which use the Gestalt principle of *connectedness*), all of these representations must dynamically change as the user scrolls and nodes move, but the changes are much more restrained.

Methods

Using IMDb as a testbed, we compared connectedness-inspired lines to our alternative designs in a controlled experiment, and included a text-based link display (Figure 4(a)) as a control condition. In this condition, nodes with the same text were linked. Note that because we were testing only explicit link representations, we did not enable interactive reordering in this experiment.

We expected that connectedness-, color- and texture-inspired links would perform better than text-based or proximity-inspired links. Because of the unique dynamic qualities of the *GraphTiles* visualization, we did not attempt to predict which link representation would be best.

Design

We used a fully crossed within subjects $5 \times 2 \times 2$ design. Link *Depiction* had five levels: text-based as well as proximity-, color-, texture-, and connectedness-inspired representations. *QueryType*, or the type of question asked, had two levels: a movie-person-movie (MPM) query or a person-movie-person (PMP) query. *Size*, or the rough size of the surrounding graph neighborhood, had two levels: small or below median, and large or above median.

Participants and procedure

We had ten participants, all university students with normal or corrected-to-normal vision. We obtained informed consent from the participants, and asked them to read the instructions for the experiment. We then familiarized them with the task and link depictions using 10 training datasets, one for each combination of link *Depiction* and *QueryType*. Participants were free to ask verbal questions during training.

Participants then each performed 120 information seeking tasks, each using a different graph neighborhood in the IMDb database, with median size of 115 nodes. On average, they completed all their tasks in one hour. Every participant performed six trials with each of the $5 \times 2 \times 2 = 20$ experimental treatments. We formed five blocks of 24 trials each, each block corresponding to one *Depiction*. Thus participants performed all trials with the current *Depiction* before moving on to the next. To combat the effects of fatigue and learning, we sampled all the orderings of *Depiction* using a 5×5 Latin Square. Within each of these *Depiction* blocks, we formed two 12-trial *QueryType* blocks. Half of the participants performed MPM questions first, half performed PMP questions first. Within each *QueryType* block, participants performed 6 trials with small neighborhoods and 6 with large neighborhoods. We randomized the order of these trials. To avoid any confound between treatments and graph neighborhoods, we randomized the match of graph to treatment. Each participant saw each neighborhood only once.

For each task, participants answered a question displayed on a nearby monitor. As for the *QueryType*, we used the same method as the previous summative experiment. We recorded the time to complete each trial, and whether or not the participant performed the trial correctly. Participants were paid $10 for their effort.

Results

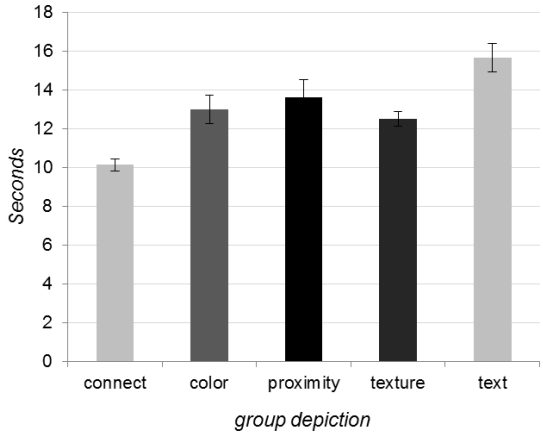


Figure 5. Average task completion times per depiction for the first experiment.

All participants completed all trials correctly, so we report only on completion time here. On completion times, we performed a single, three-factor repeated measures ANOVA. All single variable effects were significant.

The connectedness-inspired link Depiction indeed supported the fastest information seeking performance ($F(4, 36) = 4.942, p < 0.005$). Average completion times in seconds for each Depiction were: connectedness 10.1s, texture 12.5s, color 13.0s, proximity 13.6s, and text 15.7s. We show the same times in Figure 5, along with standard error. Despite their drawbacks, link lines also have strengths: they are familiar to most viewers; and they are simple, introducing only one primitive per link, while other representations require changes at both linked nodes.

Figure 4. Different explicit link representations for *GraphTiles*. (a) *text:* nodes with the same name are linked. (b) *color:* nodes with the same color are linked. (c) *connectedness:* nodes with lines between them are linked. (d) *proximity:* nodes at/containing the same vertical position are linked. (e) *texture:* nodes with/containing the same image are linked.

Participants were much faster when asked to locate a person (the MPM *QueryType*) than when asked to locate a movie (PMP) ($F(1,9) = 43.869, p < 0.001$). Average completion times for person queries were 10.5s, and for movies 15.5s. This is likely an effect of graph size rather than some more subtle task difference. Recall that IMDb's API returned many more top actors working on a movie than top movies in which an actor worked. This meant that PMP neighborhoods contained many more nodes than MPM neighborhoods.

Participants were faster when working with small graph Sizes than with large graph Sizes ($F(1,9) = 83.911, p < 0.001$). Average completion times for small graphs were 11.7s, while for large graphs they were 14.2s.

The only significant interaction occurred between the *QueryType* and *Size* variables ($F(1,9) = 25.824, p = 0.001$). When participants were asked to find movies in PMP neighborhoods, increasing graph Size had a large effect on completion times (13.4s vs. 17.6s). When they were asked to find persons in MPM neighborhoods, *Size*'s effect was minor (10.1s vs. 10.9s). In PMP neighborhoods, graphs were larger, so increasing *Size* had a larger effect.

Readers may wonder why average times in this experiment with *GraphTiles* were lesser than they were in our first experiment. One cause may be the increased practice with *GraphTiles* (10 training datasets) in this experiment.

Discussion

Results largely matched our expectations, with text-based and proximity-inspired links performing worst, texture- and color- inspired link *Depictions* performing better, and connectedness-inspired link lines performing best. However, users were only about 20% faster with link lines than with texture-inspired links containing thumbnails.

CONCLUSION AND FUTURE WORK

As mobile devices become the dominant form of computing, mobile search will become increasingly important. In this paper we described *GraphTiles*, a new search interface specifically designed to support open, imprecise, and near miss mobile queries. In an experimental evaluation, accessing the IMDb graph with *GraphTiles* was nearly twice as fast as with the existing IMDb mobile web app.

GraphTiles could use design improvements to maintain visual continuity. When users change the central node, they can quickly become disoriented. Future experiments might study how well *GraphTiles* supports *both* precise and imprecise queries, open and near miss search, as well as non-bipartite graphs. Our current implementation is quite visual. *GraphTiles* will need improvement for largely textual search domains.

We also plan to evaluate *GraphTiles* on other devices such as tablets, where we might display larger neighborhoods. The comparative merits of each of our explicit link display techniques might be different when many more links must be displayed at the same time.

REFERENCES

1. Facebook. `http://wwwfacebook.com`.
2. Internet movie database (imdb). `http://www.imdb.com`.
3. Linkedin. `http://www.linkedin.com`.
4. Pandora. `http://www.pandora.com`.
5. The Seattle Band Map. `http://www.seattlebandmap.com`.
6. Amini, S., Brush, A., Krumm, J., Teevan, J., and Karlson, A. Trajectory-aware mobile search. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '12, ACM (New York, NY, USA, 2012), 2561–2564.

7. Balasubramanian, A., Balasubramanian, N., Huston, S. J., Metzler, D., and Wetherall, D. J. Findall: A local search engine for mobile phones. In *Proceedings of the 8th International Conference on Emerging Networking Experiments and Technologies*, CoNEXT '12, ACM (New York, NY, USA, 2012), 277–288.
8. Catledge, L. D., and Pitkow, J. E. Characterizing browsing strategies in the world-wide web. In *Proceedings of the Third International World-Wide Web conference on Technology, tools and applications*, Elsevier North-Holland, Inc. (New York, NY, USA, 1995), 1065–1073.
9. Chittaro, L. Visualizing information on mobile devices. *Computer 39*, 3 (2006), 40–45.
10. Church, K., Neumann, J., Cherubini, M., and Oliver, N. Socialsearchbrowser: A novel mobile search and information discovery tool. In *Proceedings of the 15th International Conference on Intelligent User Interfaces*, IUI '10, ACM (New York, NY, USA, 2010), 101–110.
11. Church, K., and Oliver, N. Understanding mobile web and mobile search use in today's dynamic mobile landscape. In *Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services*, MobileHCI '11, ACM (New York, NY, USA, 2011), 67–76.
12. Cui, Y., and Roto, V. How people use the web on mobile devices. In *Proceedings of the 17th international conference on World Wide Web*, WWW '08, ACM (New York, NY, USA, 2008), 905–914.
13. Da Lozzo, G., Di Battista, G., and Ingrassia, F. Drawing graphs on a smartphone. 153–164. 10.1007/978-3-642-18469-7-14.
14. Hao, J., and Zhang, K. A mobile interface for hierarchical information visualization and navigation. In *Consumer Electronics, 2007. ISCE 2007. IEEE International Symposium on*, IEEE (2007), 1–7.
15. Hearst, M., Elliott, A., English, J., Sinha, R., Swearingen, K., and Yee, K.-P. Finding the flow in web site search. *Commun. ACM 45*, 9 (Sept. 2002), 42–49.
16. Kamvar, M., Kellar, M., Patel, R., and Xu, Y. Computers and iphones and mobile phones, oh my!: A logs-based comparison of search users on different devices. In *Proceedings of the 18th International Conference on World Wide Web*, WWW '09, ACM (New York, NY, USA, 2009), 801–810.
17. Karstens, B. Visualization of complex structures on mobile handhelds. In *In Proceedings of International Workshop on Mobile Computing* (2003), 17–18.
18. Lane, N. D., Lymberopoulos, D., Zhao, F., and Campbell, A. T. Hapori: Context-based local search for mobile phones using community behavioral modeling and similarity. In *Proceedings of the 12th ACM International Conference on Ubiquitous Computing*, Ubicomp '10, ACM (New York, NY, USA, 2010), 109–118.
19. Lee, U., Kang, H., Yi, E., Yi, M., and Kantola, J. Understanding mobile q&a usage: an exploratory study. In *SIGCHI '12: ACM SIGCHI Conference on Human Factors in Computing Systems* (2012), 3215–3224.
20. Lymberopoulos, D., Zhao, P., Konig, C., Berberich, K., and Liu, J. Location-aware click prediction in mobile local search. In *Proceedings of the 20th ACM International Conference on Information and Knowledge Management*, CIKM '11, ACM (New York, NY, USA, 2011), 413–422.
21. Pattath, A., Ebert, D. S., May, R. A., Collins, T. F., and Pike, W. Real-time scalable visual analysis on mobile devices. R. Creutzburg and J. H. Takala, Eds., vol. 6821, SPIE (2008), 682102.
22. Sohn, T., Li, K. A., Griswold, W. G., and Hollan, J. D. A diary study of mobile information needs. *In Proceedings of CHI'08, ACM* (2008), 433–442.
23. Wertheimer, M. Laws of organization in perceptual forms. *Psychologische Forschung 4* (1923), 301–350.