Mobile Information Retrieval with Search Results Clustering: Prototypes and Evaluations

Claudio Carpineto
Fondazione Ugo Bordoni, Rome, Italy. E-mail: carpinet@fub.it

Stefano Mizzaro
University of Udine, Udine, Italy. E-mail: mizzaro@dimi.uniud.it

Giovanni Romano
Fondazione Ugo Bordoni, Rome, Italy. E-mail: romano@fub.it

Matteo Snidero
University of Udine, Udine, Italy. E-mail: stereat@gmail.com

Web searches from mobile devices such as PDAs and cell phones are becoming increasingly popular. However, the traditional list-based search interface paradigm does not scale well to mobile devices due to their inherent limitations. In this article, we investigate the application of search results clustering, used with some success for desktop computer searches, to the mobile scenario. Building on CREDO (Conceptual Reorganization of Documents), a Web clustering engine based on concept lattices, we present its mobile versions Credino and *SmartCREDO*, for PDAs and cell phones, respectively. Next, we evaluate the retrieval performance of the three prototype systems. We measure the effectiveness of their clustered results compared to a ranked list of results on a subtopic retrieval task, by means of the device-independent notion of *subtopic reach time* together with a reusable test collection built from Wikipedia ambiguous entries. Then, we make a cross-comparison of methods (i.e., clustering and ranked list) and devices (i.e., desktop, PDA, and cell phone), using an interactive information-finding task performed by external participants. The main finding is that clustering engines are a viable complementary approach to plain search engines both for desktop and mobile searches especially, but not only, for multitopic informational queries.

Received September 4, 2008; revised December 19, 2008; accepted December 23, 2008

 DOI: 10.1002/asi.21036

Introduction

Paralleling the diffusion of high-performance mobile devices, and made possible in part by their technology, the willingness of mobile users to turn to their portable devices to find the answers they need has been increasing steadily. According to a 2008 report by The Nielsen Company, as many as 46.1 million users in the United States used mobile search functions in the third quarter of 2007 (http://www.nielsen.com/media/2008/pr_080116.html). Although the majority of users still rely on browsing listings maintained by mobile operators, the use of query-based search for accessing content not preclassified is growing fast, similar to the transition from directory services to search engines in the early Web. This interesting analogy has been investigated in a recent study by Church, Smyth, Cotter, and Bradley (2007).

Among the factors that still hamper a full escalation of query-based mobile search, a most critical one is probably represented by the inadequacy of the conventional, list-based presentation of results in the mobile scenario. The well-known limitations of this approach are in fact exacerbated by the characteristics of small mobile devices, such as their small screen, limited user-input functionalities, and high-cost connection. As the search interface of mobile terminals is designed almost the same as that of personal computers, mobile information retrieval by means of commercial search engines such as Google Mobile, Live Search Mobile, or Onesearch often results in a very tedious, long, and costly experience for users. The development

of more effective mobile search paradigms is thus a key open issue.

In this article,[1] we tackle the problem of mobile search using search results clustering, which consists of organizing the results obtained in response to a query into a hierarchy of labeled clusters that reflect the different components of the query topic. This approach has become rather popular for desktop searches,[2] attracting a substantial amount of research and commercial interest in the last years (for a recent survey, see Carpineto, Osinski, Romano, & Weiss, in press). Today, there are several Web clustering engines, the best known of which is probably Vivísimo, and the strengths and limitations of this technique are better understood. While a clustering engine may not be equally good for all types of Web queries, it supports some important information retrieval tasks where plain search engines typically fail.

The most striking feature of the cluster hierarchy is that it makes shortcuts to the items that relate to the same subtopic (or meaning, for an ambiguous query). If such items have been correctly placed within the same cluster and the user is able to choose the right path from the cluster labels, these items can be accessed in logarithmic rather than linear time. In addition, the cluster hierarchy helps filter irrelevant items and gives the user the possibility of quickly reviewing the content of search results without the need to download and scroll to subsequent pages. A further advantage is that it allows better topic understanding in unknown or dynamic domains by providing a high-level view of the whole query topic, including terms for query reformulation. The main advantages of clustering engines over plain search engines thus can be summarized as direct subtopic retrieval, reduction of information overlook (because shorter lists of snippets are presented to the user), and support for topic exploration.

It is arguable that the features of a clustering engine approach appear even more suitable for mobile information retrieval, where a minimization of user actions (e.g., scrolling and typing), device resources, and amount of data to be downloaded are primary concerns. Furthermore, such features seem to nicely comply with the usage patterns of mobile searchers, as observed in some recent studies (Kamvar & Baluja, 2006; MarketingVOX, 2005). Although there are signs that the behavior of mobile searchers is becoming more similar to that of desktop searchers (Kamvar & Baluja, 2007), the former users, compared with the latter, are still:

- more likely to enter shorter queries (The amount of effort required to enter a query into a mobile phone is more than double that required to type it on a keyboard.),
- less likely to scroll past the first few search results,
- less willing to click on search results,
- less likely to carry out multiple queries per session,
- willing to spend more time on the search results page before clicking their first link.

Despite such good potential, however, the application of search results clustering to small mobile devices has not received much attention to date in the literature. Even on the commercial side, the need for moving beyond the PC search-interface style has not been recognized until very recently, when the use of predefined categories in which to group results has started to surface in some system (e.g., Yahoo! Go; http://mobile.yahoo.com/go). The first major contribution of this article is to help fill this gap.

We build on CREDO (Conceptual Reorganization of Documents), a clustering engine based on the theory of concept lattices described in Carpineto and Romano (2004a, 2004b). CREDO was developed for a desktop computer and does not scale to a small mobile device. We study which requirements must be met to extend a desktop clustering engine to mobile devices, and then present Credino (which in Italian means "small CREDO") and SmartCREDO, two mobile versions of CREDO for PDAs and cell phones, respectively. They take the cluster hierarchy produced in response to a query by CREDO and display it on their respective device, handling the subsequent interaction with the user. To the best of our knowledge, these are the first mobile clustering engines available for testing on the Internet.

The next major contribution of this article is a comprehensive evaluation of the retrieval performance of desktop and mobile clustering engines. Our study is split in two parts. In the first part, we consider the theoretical retrieval performance of cluster hierarchies and ranked lists, regardless of the specific device used to display and interact with the results. We focus on subtopic retrieval performance and introduce an evaluation metric termed *subtopic reach time*, which is based on the number of information items that must be examined while browsing before retrieving a result relevant to a topic's subtopic. Using a set of queries labeled as *ambiguous* by Wikipedia, we find that for these queries, the subtopic reach time of clustering is clearly better than ranked list. An interesting by-product of this evaluation is a new test collection specifically designed for subtopic retrieval and made available on the Internet; it can be reused by the research community to test and compare existing and new Web clustering engines.

In the second part, we do not assume that there is a predefined model of access to information, as in the first experiment, and we explicitly consider not only the retrieval method (i.e., clustering and ranked list) but also the device (i.e., desktop, PDA, and cell phone). We use six systems (i.e., CREDO, Credino, SmartCREDO, and three versions of a plain search engine—one for each device) and make cross-comparisons through an experimental study in which external participants search a set of topics using the two methods on the three devices. We evaluate whether the retrieval performance decreases as the device becomes smaller (across

[1] A preliminary version of this work has appeared in the proceedings of ECIR 2006 (Carpineto, Della Pietra, Mizzaro, & Romano, 2006). We extend that previous work by presenting one more system (for mobile phones), a benchmark-based evaluation, and a complete user study.

[2] We use the term "desktop search" as opposed to "mobile search," with no reference to the systems that perform a search on the local computer such as Google desktop.

DOI: 10.1002/asi

retrieval methods) and whether it increases when passing from search engine to clustering engine (across devices). We formulate a number of detailed hypotheses and check if they hold. On the whole, the results of our second experiment have suggested that the retrieval effectiveness of search results clustering was usually better than that of ranked list, across multiple devices, and also for some nonstrict subtopic retrieval tasks. The benefits of clustering over ranked list were especially evident on the mobile phone. We also found that the retrieval effectiveness, whether of ranked list or clustering, in general decreases as the search device becomes smaller, although clustering may be occasionally better on smaller devices.

The remainder of this article is organized as follows. We begin by discussing the related work, namely desktop clustering engines other than CREDO, CREDO itself, and different nonconventional approaches to mobile information retrieval. We then present CREDO mobile variants, Credino and Smart-CREDO. Next, we turn to the experimental part. We first describe the "objective" evaluation, focused on a comparison between the subtopic reach time of the two retrieval methods, and then present the "subjective" evaluation, based on interactive search tasks performed using the method–device pairs. Finally, the article offers some conclusions and directions for future work.

Related Work

There are three main research lines related to this work. The first one refers to the numerous Web clustering engines for desktop searches that have been recently proposed; the second one is the CREDO clustering engine, the reference system used in this work; the last one is about alternative mobile information retrieval techniques. They are discussed, in turn, in the next subsections.

Clustering Engines

Over the last years, clustering engines have proved a viable complement to conventional search engines. Following the popularity of Vivísimo, several commercial systems perform Web-snippet clustering: Accumo, Fluster, Grokker, KartOO, Lingo3G, and Mooter, among others. This issue also has gained much attention in the academic research field, with several implemented prototypes available online besides CREDO, such as Lingo (Osinski, Stefanowski, & Weiss, 2004; Osinski & Weiss, 2005), SRC (Zeng, He, Chen, Ma, & Ma, 2004), SnakeT (Ferragina & Gulli, 2004, 2005), EigenCluster (Cheng, Vempala, Kannan, & Wang, 2005), and WhatsOnWeb (Di Giacomo, Didimo, Grilli, & Liotta, 2007). Even major search engines such as Google and Yahoo! have shown interest in this technology, and currently seem to have adopted some form of implicit clustering to increase the number of perspectives on their first result page.

The basic architecture of a Web clustering engine, common to most systems, consists of three components arranged in a pipeline: (a) acquisition and preprocessing of search results, (b) cluster construction and labeling, and (c) visualization of clustered results. The available systems mainly differ in the algorithm for clustering search results. Compared to traditional document clustering, it must fulfill a number of more stringent requirements raised by the nature of the application in which it is embedded, such as meaningful cluster labels, high computational efficiency, short input data description, unknown number of clusters, and overlapping clusters. Given the intended use of clustering engines, the most critical part is the quality of cluster labels, as opposed to optimizing only the clustering structure. Due to the large variety of search results clustering algorithms that have been developed, the currently available clustering engines usually output very different clusters and cluster labels for the same query.

There are two main approaches: *data-centric* and *description-centric* algorithms (for a more elaborate classification of search results clustering algorithms, see Carpineto et al., in press). In the *data-centric* group, we find conventional data clustering algorithms (e.g., hierarchical agglomerative, k-means, spectral) applied to search results and often slightly modified to produce a more comprehensible cluster description. Input texts are represented as bags of words, and cluster labels typically consist of single words recovered from the cluster representative, possibly expanded into phrases. This class contains both early and recent systems, such as Lassi (Maarek, Fagin, Ben-Shaul, & Pelleg, 2000), CIIRarchies (Lawrie & Croft, 2003; Lawrie, Croft, & Rosenberg, 2001), EigenCluster (Cheng et al., 2005), Armil (Geraci, Maggini, Pellegrini, & Sebastiani, 2007), with Scatter/Gather (Cutting, Pedersen, Karger, & Tukey, 1992) as remarkable predecessor.

When the quality of cluster labels is given priority over allocation of search results, we can speak of *description-centric* algorithms. In this latter class, cluster labels are generated first, usually by extracting short sequences of words (not necessarily contiguous) from the search results, and then a cluster hierarchy is built with or from such labels. This approach, pioneered by Suffix Tree Clustering (Zamir & Etzioni, 1998, 1999), has become mainstream in the last few years (e.g., Cheng et al., 2005; Di Giacomo et al., 2007; Ferragina & Gulli, 2004, 2005; Zeng et al., 2004). The systems in this class differ in the choice of the data source and method used to extract the textual features and in the algorithm for hierarchy construction.

CREDO

CREDO is based on formal concept analysis, which combines a strong mathematical background (Ganter & Wille, 1999) with a set of efficient manipulation algorithms (Carpineto & Romano, 2004a). Formal concept analysis has been applied in several fields, including those related to information science (for a recent review, see Priss, 2006). As the algorithm employed by CREDO has been fully presented elsewhere (Carpineto & Romano, 2004a, 2004b), here we mainly provide an operational description of the system and then focus on its use for information retrieval.

DOI: 10.1002/asi

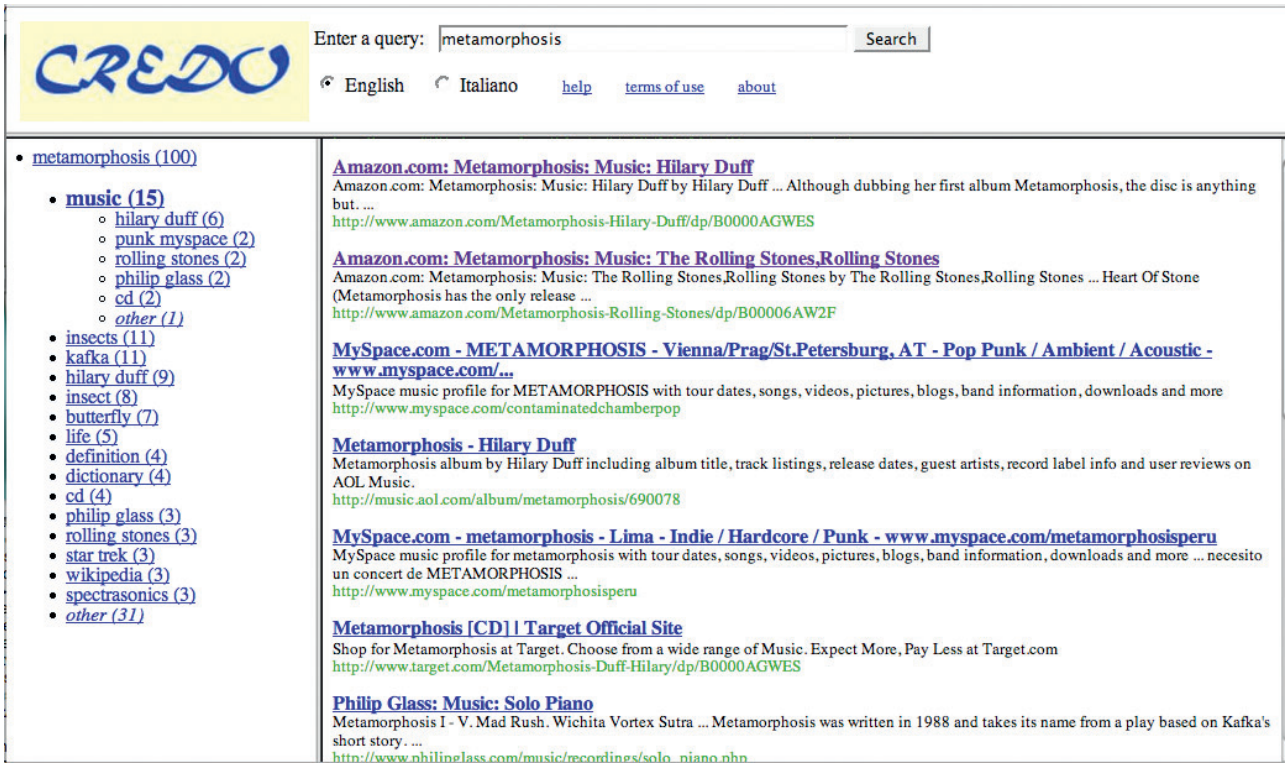

FIG. 1. Results of CREDO for the query "metamorphosis," with selection of cluster "music."

CREDO forwards a user query to an external Web search engine and collects the first 100 results. Each result is then indexed by single terms extracted from its title and snippet (up to simple text normalization). The next step is the construction of a hierarchical clustering structure, termed *CREDO hierarchy*.

The clusters built by CREDO can be seen as formal concepts (Ganter & Wille, 1999). Each concept is formed by a subset of terms (the concept intent) and a subset of search results (the concept extent). The intent and extent of any concept are such that the intent contains *all* terms shared by the search results in the extent, and the extent contains *all* search results that share the terms in the intent. A formal concept is therefore a *closed set* on both its dimension. In addition, the concepts can be ordered by the standard set inclusion relation applied to the intent and extent that form each concept, yielding a subconcept/superconcept relation.

In practice, CREDO starts from the cluster placed at the hierarchy root, which is usually described by the query terms and covers all retrieved results, and then iteratively builds two lower levels of the hierarchy. Each level contains the most general of the concepts that are theoretically more specific than the concepts in the preceding level, according to the definition of formal concepts. To increase the utility of the clustering process for subtopic retrieval, the first level is generated using only the terms contained in the title of search results, and the second level using both the title and the snippet.

The CREDO hierarchy is then visualized using a simple folder tree layout. The system initially shows the hierarchy root and the first level of the hierarchy. The user can click on each cluster to see the results associated with it and expand its subclusters (if any). All the documents of one cluster that are not covered by its "children" are grouped in a dummy cluster named "other."

To illustrate, Figure 1 shows the clustered results returned by CREDO for the query "metamorphosis," with subsequent selection of cluster "music" on the part of the user (as of January 2008). Like many queries on the Web, "metamorphosis" has multiple meanings with several high-scoring hits each: the changes in the life cycle of insects, Franz Kafka's novella, Hilary Duff's album, the Rolling Stones' compilation, Philip Glass's composition, the Star Trek episode, and so on. These different subjects are well represented in Figure 1, with the musical meanings appropriately grouped in a "music" cluster. By contrast, if we submit the query "metamorphosis" to Google or Yahoo!, we see that the first result pages are only about the biological process and the Kafka's novella, with the musical meanings of metamorphosis being completely missing in the first three pages (as of January 2008). This problem is being recognized by major search engines, which interweave documents related to different topics by some form of *implicit clustering*; however, certain topics will be inevitably hidden from the user due to the limited capacity of a single results page.

CREDO does not neatly fit in either of the two classes discussed earlier. Similar to data-centric algorithms, it uses strict single-word indexing. Its monothetic clusters are mostly described by a single word, but they also can accommodate labels with multiple contiguous words, reflecting the causal

DOI: 10.1002/asi

(or deterministic) associations between words in the given query context. For instance, for the query "metamorphosis" (see Figure 1), CREDO returns some multiple-word concepts such as "hilary duff" and "star trek," consistent with the fact that in the limited context represented by the results of "metamorphosis," "hilary" always co-occurs with "duff" and "star" with "trek."

In CREDO, cluster labeling is integrated with cluster formation by definition because a concept intent is univocally determined by a concept extent, and vice versa. Thus, CREDO builds the cluster structure and cluster descriptions at once. By contrast, these two operations are usually treated separately. The disadvantage of the common approach is that there may be a mismatch between the criterion used to find a common description and that used to group the search results, thus increasing the chance that the contents will not correspond to the labels (or vice versa).

Another interesting feature of CREDO, which is uncommon in other systems, is that the same result can be assigned to multiple (i.e., overlapping) clusters. For instance, both the clusters "butterfly" and "life" in Figure 1 contained a common search result, whose title was "Metamorphosis—Life Cycle of the Butterfly." Incidentally, this is the reason why the sum of the search results covered by the children of the top element of the hierarchy in Figure 1 is greater than 100 (i.e., 124). Furthermore, CREDO allows the user to reach the same cluster through multiple paths because the cluster structure is a graph rather than a tree. For instance, in the metamorphosis example, the secondary topic described by "music hilary duff" can be found by choosing "music," and then "hilary duff" (as in Figure 1), or by looking at "hilary duff," and then "music." Neither path is better than the other, but one may better fit a particular user's paradigm or need. This feature may be especially useful when a result can be categorized along orthogonal axes (e.g., functional, geographic, descriptive); however, this does not come without drawbacks because the presence of repeated labels along different paths may clutter the tree-based layout of CREDO.

Although a quantitative comparison between CREDO and other clustering engines is outside the scope of this article, note that such a study has been conducted elsewhere (Carpineto et al., in press). Of the five systems tested, CREDO achieved the second-best subtopic retrieval performance. This result supports the choice of CREDO as a reference system for testing the merits of clustering over plain search in the experiment presented here.

Computational efficiency is in principle another weak point of a concept-based approach to search results clustering because the asymptotic worst-case time complexity of the construction algorithm grows quadratically with the number of search results or index terms. Fortunately, this is not so much an issue in practice because the input to the clustering algorithm usually consists of a few hundred short snippets; in fact, the time spent on cluster construction is typically a small fraction of that required to acquire and preprocess search results (for a detailed discussion of the efficiency of Web clustering engines, see Carpineto et al., in press).

CREDO runs on a Web server as a Common Gateway Interface application implemented in *Shark*, a Lisp-like language being developed by the third author of this article that is automatically translated into ANSI C and then compiled by GNU Compiler Collection. Recently, the system has been partially re-implemented to collect the search results via Yahoo! APIs and to optimize the construction of the cluster hierarchy. In the current version, the time necessary for the latter operation is about $\frac{1}{10}$ of a second. CREDO is available for testing at http://credo.fub.it/

Mobile Information Retrieval

Although mobile information retrieval is quite a recent research topic, some systems are available to the general public. The mobile versions of Google, Yahoo!, and Microsoft search engines have been available online for a few years, as mentioned earlier, and a steadily increasing number of general-purpose search engines provide a version specifically designed for mobile devices. Likewise, a more specific search engine such as PubMed has provided since 2003 two mobile versions for PDAs: MEDLINE Database on Tap (MDoT; http://mdot.nlm.nih.gov/proj/mdot/mdot.php), a stand-alone application running on Palm OS and Pocket PC mobile devices, and PubMed for Handhelds (PM4HH; http://pubmedhh.nlm.nih.gov/nlm/), a Web application explicitly designed for mobile-device browsers.

Mobile information retrieval already has some space in the scientific literature, from different perspectives. From a human–computer interaction perspective, several issues have been discussed, and some suggestions for effective mobile information retrieval can be drawn. For example, data visualization on small screens is discussed in Noirhomme-Fraiture, Randolet, Chittaro, and Custinne (2005), in which it was suggested to avoid horizontal scrolling and page navigation, and to exploit clustering techniques to increase the amount of information shown at the same time. Indeed, an important feature of clustering is that it does not need graphical effects to show a large amount of information in a condensed form, as noted by S. Jones, Jones, and Deo (2004), who discussed how to replace conventional surrogates with sets of keyphrases automatically extracted from document text. M. Jones, Buchanan, and Mohd-Nasir (1999) also presented WebTwig, a prototype search interface for handheld browsers that provides to the user a hierarchical outline view of a Web site. Another relevant study by the same research group is M. Jones, Buchanan, and Thimbleby (2002), in which the danger of a complete failure when searching with mobile devices is discussed and some general guidelines are proposed. Query suggestion seems to have a good potential as well: As reported in Kamvar and Baluja (2008), users' difficulty in entering query terms on their mobile phones is relieved by term suggestion, and users seem to be more satisfied and effective when using the term-suggestion feature.

A second perspective is taken in the studies that have tried to understand the specific characteristics of information needs

DOI: 10.1002/asi

for mobile devices, and how these are different from classical needs of desktop search users. Kamvar and Baluja (2006) studied the query log of the two versions of Google search engine for PDAs (http://www.google.com/pda/) and cell phones (http://www.google.com/xhtml/). They found some differences between mobile and desktop users, and between PDA and cell phone users as well. Church et al. (2007) did not take into account the logs of a single search engine only but exploit the logs of major European mobile operators, which include queries submitted to more than 30 different mobile search engines. They found that although still much less frequent than mobile browsing, mobile search is a varied, interesting, and increasing activity. On the basis of the experimental evidence, several design guidelines for mobile search services also have been proposed. More recently, Sohn, Li, Griswold, and Hollan (2008) followed a different and complementary approach: Their 2-week diary study involved 20 participants and aimed to capture the different types of information needs that real users have when on the move. Results have shown a wealth of situations and needs, and of different user strategies to fulfill them. Some implications for the design of mobile technologies are derived; for example, since several needs were postponed or dismissed, delayed search seems important (as already proposed by M. Jones, Jain, Buchanan, & Marsden, 2003).

A third perspective concerns summarization, obviously a very important issue for mobile devices where efficient screen usage is paramount. Buyukkokten, Garcia-Molina, and Paepcke (2001) reported a user study aimed at understanding the effectiveness of five different methods for summarizing parts of Web pages on mobile devices. In a similar study, Otterbacher, Radev, and Kareem (2006) studied the effectiveness of hierarchical summarization, measured both as user performance in accomplishing a task and as bandwidth consumption. Sweeney and Crestani (2006) sought the optimal summary length on three different devices: a smartphone, a PDA, and a laptop. The results of their user study, perhaps unexpectedly, do not show a correlation between screen size and summary length.

A complementary approach to summarization was taken by Church and Smyth (2007), who studied content enrichment; that is, how to enrich the description of a short Web page (e.g., a page designed for a mobile device), with the aim of a more effective indexing.

The aforementioned studies have agreed on some features (as discussed earlier): Mobile users tend to enter fewer and shorter queries, are less willing to click on search results, tend not to scroll past the first few search results, are more reluctant to click on hyperlinks, and require compact data-visualization strategies. The clustering approach that we propose in this article takes into account in a natural way all these issues.

CREDO Goes Mobile

We developed two versions of CREDO for mobile devices: Credino, for PDAs, and SmartCREDO, for cell phones. We refer to both of them as *Mobile CREDO*. They are described in this section, starting from common requirements and architecture.

General Requirements

Using a clustering engine approach for mobile search poses additional requirements compared to those which must be met in a desktop search. There are two main reasons why CREDO, like other clustering engines optimized for desktop searches, cannot be used on a mobile device.

The first general requirement concerns usability and accessibility issues. Simply reproducing the frame-based CREDO interface would lead to a virtually unusable interface on a small screen, with an unacceptable amount of scrolling. In particular, users would be required to scroll horizontally, which is a very tedious way to see an entire CREDO screen.

A second constraint comes from bandwidth considerations. As the current implementation of CREDO is based on computing the whole hierarchy and sending out at once the results of all possible cluster selections, it is suitable for medium or broadband Internet connections (An average CREDO page like the one shown in Figure 1 is about 150,000 bytes.) By contrast, mobile device connection to the Web often has a low bandwidth (like General Packet Radio Service) and is not free, the billing depending on the amount of data transmitted. Therefore, we chose for the mobile version of CREDO an architecture that minimizes the amount of data transmitted to the mobile device (In the mobile versions, the category pages weigh about 10,000 bytes.)

Overall Architecture

The bandwidth constraint suggests to rely on an intermediate server (henceforth CREDOMobile server) to minimize both the amount of data sent to the mobile device and the load on the CREDO server (see Figure 2). The CREDOMobile server connects on one side to a user's mobile device and on the other side to (a slightly modified version of) the CREDO search engine.

The CREDOMobile server receives a user's commands (i.e., query execution, cluster expansion, cluster contraction, visualization of the content of a cluster, visualization of a Web page) as HTTP requests from a mobile device. When the user submits a query, CREDOMobile server forwards the command, appropriately formatted into another HTTP request, to the CREDO server, which processes it and returns the result as an XML data structure. We purposefully developed an XML version of CREDO output (rather

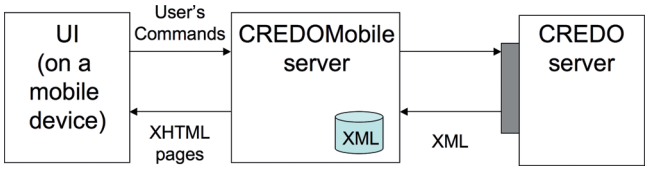

FIG. 2. Overall CREDOMobile architecture.

DOI: 10.1002/asi

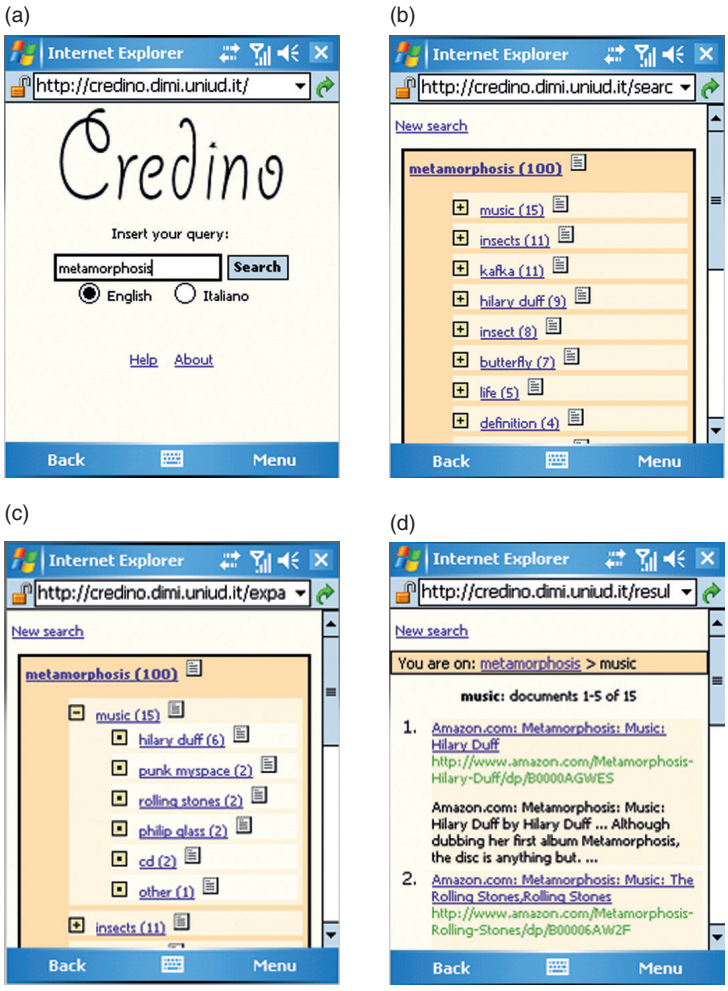

FIG. 3. Credino home page with the query "metamorphosis" (a); Credino clusters for "metamorphosis" (b) and (c), with the cluster "music" expanded; snippets of the documents associated with the path: metamorphosis > music (d).

than relying on the HTML plus Javascript output produced by the standard version of CREDO) to facilitate subsequent processing by CREDOMobile. The result of a query, consisting of clusters and documents, is then locally stored by CREDOMobile. This enables CREDOMobile to execute the other user commands without connecting again to CREDO: After a query, CREDOMobile can handle all subsequent user actions until the next query is issued or the visualization of a Web page is requested. CREDOMobile answers are returned to the Web browser on the mobile device as XHTML pages. CREDOMobile and CREDO servers could run on the same computer, but in general, they can be different (as in our current implementation).

In the next two sections, we present the two user interfaces of CREDOMobile, for PDAs (i.e., Credino) and cell phones (i.e., SmartCREDO), respectively.

Credino

Figure 3a shows the Credino home page with the query "metamorphosis." Figures 3b and 3c show the first-level clusters displayed by Credino in response to the query "metamorphosis." Note that the query box is at the bottom of the page to save space for the first results. The user can expand a cluster into its subclusters by clicking on the "+" icon. In Figure 3c, the cluster "music" is expanded into "hilary duff," "punk myspace," and so on.

The user also may see the snippets of the documents contained in a cluster by clicking on the cluster name or on the icon on the right. Figure 3d shows the snippets of the documents associated with the selected subcluster. The figure shows the main difference between Credino and CREDO: The document snippets and the categories are not shown together due to space limitations. To provide information to users as to where they are located within the hierarchy, we use the breadcrumb trail metaphor (i.e., a sequence of clusters from the root to the current page). Path breadcrumb trails are dynamically displayed during the interaction between the user and the system, as shown in Figure 3d for the path "metamorphosis" > "music."

Credino is implemented as PHP scripts plus domxml PHP module for XML parsing on a Linux server running

DOI: 10.1002/asi

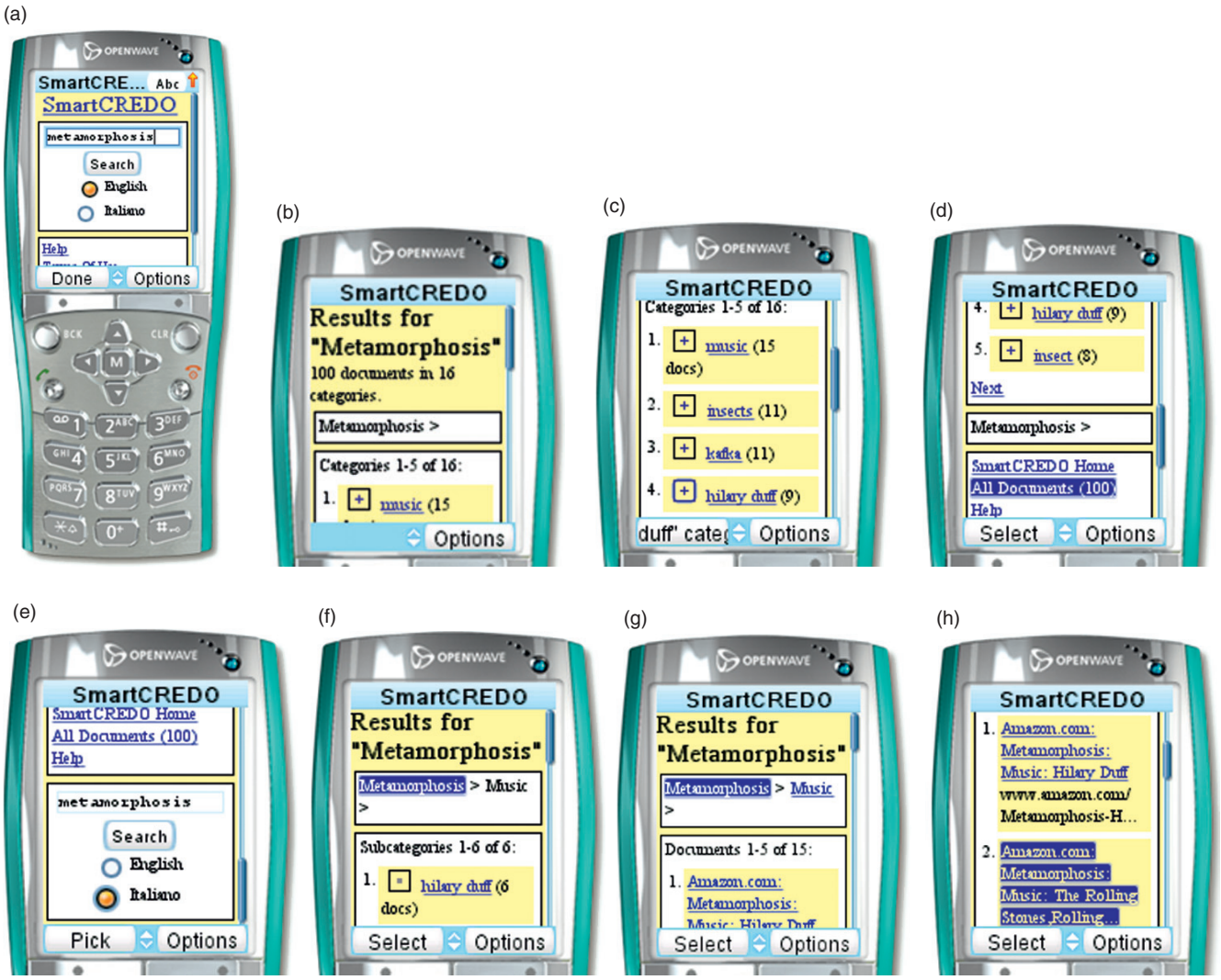

FIG. 4. The SmartCREDO home page, as shown by the Openwave simulator, with the query "metamorphosis" (a); main categories (b), (c), (d), and (e); subcategories in an expanded category (f); and documents in the subcategory "metamorphosis" > "music" (g) and (h).

the Apache Web server. It adheres to the XHTML 1.0 and CSS standards: The user interface can be visualized on any browser conforming to the W3C standards. Credino is available at http://credino.dimi.uniud.it/

SmartCREDO

Credino is tailored to the screen of a PDA, but most mobile phones on the market today have much smaller screens. Indeed, the Credino user interface is inadequate for mobile phone small screens. Therefore, on the basis of Credino, we have developed a second version of Mobile CREDO, named *SmartCREDO*, featuring a user interface tailored to even smaller screens like those of smartphones and cell phones.

Figure 4 shows SmartCREDO pages after the query "metamorphosis." The system presented in figure is the last version, developed after some heuristic evaluation sessions on previous prototypes. There are some small differences with Credino, to show less information and to adapt to the phone interface and controls. The main difference between SmartCREDO and Credino is that categories and subcategories are not shown together: After category expansion, subcategory visualization in SmartCREDO takes place on a new page (see Figure 4f).

SmartCREDO is implemented as a J2EE Web application. Like Credino, its user interface can be visualized on any browser conforming to the W3C standards. SmartCREDO is available at http://smartcredo.dimi.uniud.it/

Objective Evaluation of CREDO Hierarchy

In this section, we present an evaluation of the subtopic retrieval performance of the CREDO hierarchy compared to a ranked list of results. The focus is on the structure, content, and labels of the generated clusters, independently of their display and usage on a specific device. We first review current evaluation methods, then describe the AMBIENT test collection for subtopic retrieval, introduce the subtopic reach time metric, and present the experimental results.

DOI: 10.1002/asi

Review of Existing Approaches to Evaluating Clustering Versus Ranked List

The typical approach consists of first linearizing the clustered representation by making some usage assumption, and then comparing the retrieval effectiveness of the found linear representation with that of the ranked list using standard information retrieval methods. One of the earliest and simplest techniques is to assume that the user can choose the cluster(s) with the highest density of relevant documents and to consider only the documents contained in it ranked in order of relevance (Hearst & Pedersen, 1996; Schütze & Silverstein, 1997; Zamir & Etzioni, 1998). In practice, however, the user may easily fail to choose the optimal cluster(s).

A more analytic approach (Kummamuru, Lotlikar, Roy, Singal, & Krishnapuram, 2004) is based on the *reach time*; that is, a model of the time taken to locate a relevant document in the hierarchy. It is assumed that starting from the root, a user chooses one node at each level of the hierarchy by looking at the descriptions of that level, and iteratively drills down the hierarchy until a leaf node that contains the relevant document is reached. At this point, the documents under the leaf node are sequentially inspected to retrieve the relevant document.

More precisely, let s be the branching factor, d the number of levels that must be inspected, and p_{ic} the position of the ith relevant document in the leaf node. The reach time for the ith document is: $rt_{clustering} = s \cdot d + p_{i,c}$. Clearly, the corresponding reach time for the ranked list is: $rt_{rankedlist} = p_{i,r}$ (where $p_{i,r}$ is the position of the ith relevant document in the ranked list). The reach times are then averaged over the set of relevant documents.

The reach time seems to model the user behavior more accurately, but it relies on the assumption that a cluster label will allow the user to choose the right cluster (i.e., the cluster in which the sought document is contained), which is not always the case. This aspect cannot be easily accounted for. Some work has been done to evaluate the quality of cluster labels, for instance, by computing their informativeness with respect to the clusters content (Lawrie et al., 2001), or by measuring the precision of the list of labels associated with the clusters, which requires prior manual assessment of the relevance of labels to topics (Spiliopoulou, Schaal, Müller, & Brunzel, 2005; Zeng et al., 2004). However, the quality of labels has been usually considered as a stand-alone issue, without considering how it affects the overall system's retrieval performance.

Note that in all approaches discussed so far, it is assumed that the user is interested in retrieving all documents relevant to the query as a whole, although this may not be the most typical usage scenario for a Web clustering engine. In fact, the relative performance of clustering engines and plain search engines may vary strongly depending on the search task being considered. Of special interest is the case of subtopic retrieval, in which we want to find documents that cover many different subtopics of a query topic. This task seems very suitable for clustering engines because the subtopics should match the high-level clusters, but it is more difficult to evaluate because the utility of a result is dependent on the results that already have been seen.

A similar subtopic-retrieval perspective has been adopted for evaluating the performance of ranked lists of results in a few recent works (e.g., Chen & Karger, 2006; Hersh & Over, 1999; Zhai, Cohen, & Lafferty, 2003). It is based on *instance recall at rank n*, defined as the number of unique subtopics covered by the first n results, divided by the total number of subtopics. For our purposes, the most interesting feature of this metric is that multiple documents on the same topic count as one instance; however, it focuses on pure recall (i.e., it does not consider the order in which the first n documents are retrieved) and cannot be applied to clustered results.

The lack of task-specific measures for comparing clustering to ranked list is compounded by the paucity of suitable test collections. Most evaluation studies have relied on a small set of ambiguous queries chosen anecdotally, and they do not give access to the results and their relevance judgments. One notable exception is the test collection made available in Ferragina and Gulli (2005), in which, however, the relevance of documents was still evaluated with respect to the query as a whole, as for a traditional information retrieval task. To our knowledge, the only test collection that is tailored to query aspects is that formed by the TREC-6, -7, and -8 interactive track runs; however, its focus is on the *instances* of a given topic rather than on the topic's subtopics. These concepts are quite distinct.

Overall, the experimental findings reported in the literature are in general favorable to clustering engines, suggesting that they may be more effective than plain search engines. However, caution is warranted because these results may be biased by unrealistic usage assumptions or unrepresentative search tasks. To date, the hypothesis that clustering engines are superior to plain search engines has not been convincingly demonstrated, not even for restricted types of queries (e.g., ambiguous informational queries).

In the next sections, we present a new test collection for subtopic retrieval and a new effectiveness metric for clustered results; together, they address some of the limitations of current evaluation methodologies.

The AMBIENT Test Collection for Subtopic Retrieval

In this section, we present the AMBIENT (AMBIguous ENTries) test collection. In summary, it consists of 44 topics, each with a set of subtopics and a list of 100 ranked documents. The 100 documents associated with each topic were collected from a Web search engine (i.e., Yahoo!) by submitting the topic name as a query and subsequently were manually annotated with subtopic relevance judgments.

The intended use of AMBIENT is to support a focused comparison between the subtopic retrieval effectiveness of a ranked list of results and that of the same list postprocessed by search results clustering. In practice, the search results clustering system to be evaluated takes as input the

DOI: 10.1002/asi

100 documents associated with each topic and clusters them; the clustered results are then compared to the original ranked list using the relevance judgments.

We now describe in detail how the collection was built and its main features. The topics were selected from the list of ambiguous Wikipedia entries; that is, those with "disambiguation" in the title (see http://en.wikipedia.org/wiki/Wikipedia:Links_to_%28disambiguation%29_pages). To ensure that the test was significant, we required that each topic had at least five subtopics. Additionally, given the constraint on the number of results per topic (One hundred is the typical number of search results processed by a Web clustering engine.), we considered only the ambiguous Wikipedia topics with fewer than 35 subtopics. A further constraint was to require that there were at least two Wikipedia subtopics in the first 10 results returned by the search engine to make sure that there was a significant intersection between the Wikipedia subtopics and the subtopics retrieved by the search engine. The complete set of topics, finalized as of September 2007, is shown in the first column of Table 1.

The topics are mixed in content. There are both broad (e.g., cube, globe) and specific (e.g., labyrinth, indigo) words, person (e.g., Landau, Oppenheim) and location (e.g., Bronx, La Plata) names, scientific (fahrenheit, monte carlo) and artistic (magic mountain, excalibur) subjects, and so on. There are 34 topics described by a single word, 7 by two words, and 3 by three words. The number of Wikipedia subtopics per topic is shown in the second column of Table 1; its value ranges from 6 to 37, with an average of 17.95. Almost all topics have at least one popular meaning, and usually more.

The documents associated with each topic were collected using Yahoo! API, with the "English language" search option (as of September 2007). Each document was represented by its URL, title, and snippet. The next step was to assign the relevance judgments. For each topic and for each result, the relevance of the result to each of the Wikipedia topic's subtopics was assessed. Only the information made available in the results pages was used, without downloading the full documents. This operation was performed by the authors of this article with the support of a Web-accessible evaluation interface developed on purpose.

We now provide some statistics obtained by analyzing the subtopic relevance judgments. The distribution of retrieved subtopics over the set of topics is shown in the third column of Table 1. Of the total 790 Wikipedia subtopics, only 345 were present in the search results, with an average of 7.93 subtopics per topic. In the fourth column of Table 1, we show the number of relevant results retrieved for each topic (i.e., which were relevant to at least one of its Wikipedia subtopics). On average, 51.29 of the 100 results returned by Yahoo! were found to be relevant. The average number of relevant results per retrieved subtopic is 6.467, with a minimum of 1 and a maximum of 76. The number of search results relevant to at least one subtopic is 2,224, of the total 4,400 search results. The average number of subtopics per relevant search result is 1.014 (Thus, each relevant search result is typically assigned to only one subtopic.), with a minimum of 1 and a maximum of 3. If we also consider the multiple assignments, the total number of relevant search results slightly grows from 2,224 to 2,257.

These statistics tell us that less than half of the Wikipedia topics were retrieved and that roughly half of the search results were not about some Wikipedia topic. For instance, considering the "metamorphosis" topic, only 7 of its 16 Wikipedia subtopics were present in the first 100 results returned by Yahoo!. The converse also is true. There are many retrieved noticeable subtopics that are not present in the Wikipedia list. Fortunately, the mismatch between the Wikipedia subtopics and the subtopics retrieved by the search

TABLE 1. Main features of the AMBIENT test collection.

Topic no.	Topic description	No. of Wikipedia subtopics	No. of retrieved subtopics	No. of relevant results
1	Aida	31	11	61
2	B-52	6	3	75
3	Beagle	11	4	86
4	Bronx	10	4	81
5	Cain	22	8	38
6	Camel	13	6	70
7	Coral Sea	6	4	45
8	Cube	26	10	48
9	Eos	21	8	65
10	Excalibur	26	8	32
11	Fahrenheit	13	8	72
12	Globe	18	11	53
13	Hornet	23	9	45
14	Indigo	28	15	40
15	Iwo Jima	10	7	90
16	Jaguar	22	6	80
17	La Plata	12	7	68
18	Labyrinth	16	6	26
19	Landau	37	15	40
20	Life on Mars	7	4	84
21	Locust	15	7	50
22	Magic Mountain	10	7	41
23	Matador	22	8	38
24	Metamorphosis	17	7	57
25	Minotaur	7	7	51
26	Mira	22	13	38
27	Mirage	31	9	34
28	Monte Carlo	10	5	72
29	Oppenheim	13	9	41
30	Out of Control	14	8	18
31	Pelican	24	7	63
32	Purple Haze	11	8	27
33	Raam	8	5	58
34	Rhea	19	6	52
35	Scorpion	32	12	44
36	The Little Mermaid	18	7	49
37	Tortuga	10	6	29
38	Urania	14	7	45
39	Wink	17	10	46
40	Xanadu	21	14	50
41	Zebra	29	10	71
42	Zenith	21	6	30
43	Zodiac	22	7	20
44	Zombie	25	10	34

DOI: 10.1002/asi

engine does not affect the scope of our evaluation, provided that their intersection contains a significant number of elements. In fact, we are not interested in measuring the ability of a search engine to retrieve all possible subtopics of a query; our goal is to evaluate whether postretrieval clustering is better than ranked list presentation for accessing at least *some* subtopics contained in the search results; that is, those retrieved by the search engine *and* listed in the Wikipedia entry corresponding to the topic.

The AMBIENT test collection is available at http://credo.fub.it/ambient/ It contains the topics along with their subtopics, the search results obtained in response to the topics, and the subtopic relevance judgments. The search results are necessary for cross-system comparison because the output returned by Web search engines changes frequently. To our knowledge, this is the first test collection built analyzing the subtopic (rather than the topic) relevance of Web search results, thus helping fill a gap between the recent growth of Web clustering engines and the lack of appropriate tools for their evaluation.

Subtopic Reach Time: A Novel Evaluation Measure

In this section, we introduce the subtopic reach time (SRT) metric. It is computed in two distinct ways depending on whether we consider ranked list or clustering.

- *Ranked list.* The SRT of a topic is defined as the mean, averaged over the n retrieved topics' subtopics, of the position (p) of the first result relevant to each retrieved subtopic in the ranked list associated with the topic. A retrieved subtopic is a subtopic for which at least one relevant result has been retrieved. Mathematically,

$$SRT_{list} = \frac{\sum_{i=1}^{n} \min_j(p_{i,j})}{n},$$

  where the index j refers to the results relevant to subtopic i.
- *Clusters.* Similar to the reach time metric, we explicitly take into account both the cluster labels that must be scanned and the snippets that must be read. The SRT of a topic is defined as the mean, averaged over the n retrieved topics' subtopics, of the smallest of the reach times associated with each subtopic's relevant results. The reach time of a result is given by the position of the cluster in which the result is contained (c) plus the position of the result in the cluster (r):

$$SRT_{cluster} = \frac{\sum_{i=1}^{n} \min_j(c_{i,j} + r_{i,j})}{n}.$$

  The position of clusters is given by their top-down order on the screen; if the system returns a cluster hierarchy, as with CREDO, only the first partition of the hierarchy is considered. If the same result is contained in more than one cluster, the smallest of the reach times associated with each cluster is selected.

To illustrate, consider again the clustered results of CREDO for the query "metamorphosis" in Figure 1. The subtopic reach time of the "rolling stones" subtopic is the reach time of the second result displayed in Figure 1. Its value is equal to 3, given by the position of cluster "music" in the list of cluster (i.e., 1) plus the position of the result in that cluster (i.e., 2). Note that the other results relevant to the subtopic have higher reach times, including those contained in the very specific "rolling stones" cluster at the first level of the CREDO hierarchy. Clearly, for a topic with n subtopics, the minimum SRT is $\frac{n+1}{2}$ in the case of ranked list and $\frac{n+3}{2}$ in the case of clustered results. The two values are different because the theoretically minimum cognitive effort involved with browsing clustered results is inherently greater than ranked list. Using clustered results, the user always will have to scan both the list of clusters and the list of results within a selected cluster whereas with a ranked list, it may be sufficient to inspect just the first n results. Of course, the latter situation is very unlikely in practice.

Also note that the minimum SRT depends on the number of subtopics. Topics with more subtopics have an inherently higher minimum SRT because the system will have to give access to more distinct pieces of information. This is perfectly reasonable, but it may result in an excessive penalization of the topics with few subtopics when the variability in the number of subtopics in the test collection is high. One drastic solution to make subtopic reach times more comparable across topics is to normalize the individual values over the number of subtopics.

In the definition of SRT, the shortest path to the subtopic's relevant results is computed automatically. The SRT value is thus an upper bound on the performance to be expected when the clusters are manually selected by the user. In addition, as the user has to make the right choice twice (once for the cluster and once for the item), such an upper bound is probably further from the really achievable effectiveness in the case of clustered search than in the case of ranked search (in which only one right choice for the item is required).

In a more realistic evaluation, the cluster selection should be based on the relevance of the cluster labels to the subtopic of interest. This can be modeled in the following way. For each subtopic, the set of cluster labels returned by the clustering engine are first manually ranked in order of relevance to the subtopic; then, the SRT is computed by visiting the clusters in the same order as their ranked labels until one relevant result is found. As an illustration of this label-driven method, consider again the "rolling stones" example in Figure 1. The most relevant label would be "rolling stones," thus implying that the correspondent cluster would be considered first in the computation of SRT. In this case, we would obtain a (nonoptimal) SRT value equal to 13; that is, 12 (the position of cluster "rolling stones") + 1 (assuming that the first result contained in it is relevant to the "rolling stones" subtopic).

This approach has the advantage that the label quality would be fully integrated in the task performance. Its main shortcoming is that it assumes that the user keeps selecting clusters even when they do not have relevant labels anymore, which may not reflect the typical behavior of a clustering engine user. In our experiment, we have used a simplified method that will be described later.

DOI: 10.1002/asi

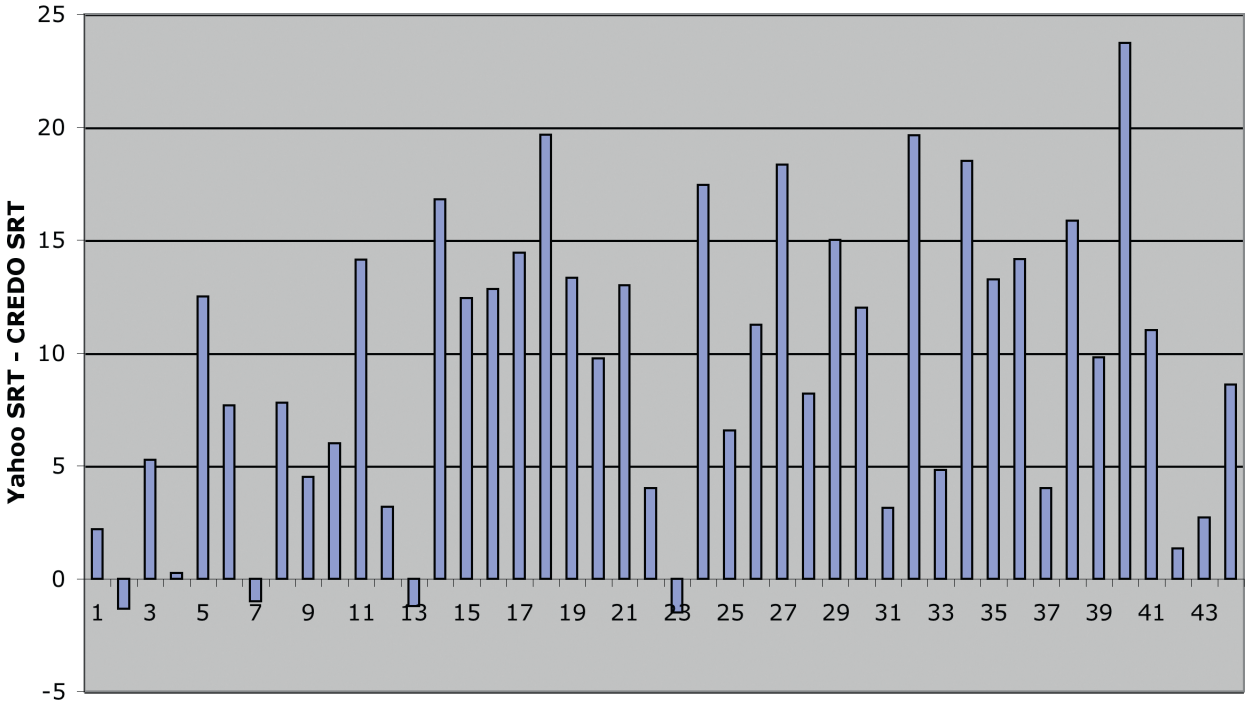

FIG. 5. Difference between Yahoo! SRT and CREDO SRT on the AMBIENT individual topics (in the same order as in Table 1).

Before concluding, we would like to emphasize that the subtopic reach time can be used not only for comparing clustering to ranked list but also for comparing alternative methods for search results clustering. Note, however, that SRT only makes sense to compare systems which act on exactly the same set of retrieved documents, and eventually present them all to the user; it is not adequate when comparing systems that retrieve different documents. Another shortcoming of SRT is that all subtopics are seen as equally important, which may not work well if some subtopic is much more popular than the others.

Results

For each topic, we computed the Yahoo! SRT and CREDO SRT values. They were given, respectively, by the SRT_{list} equation applied to the ranked list of results returned by Yahoo!, and by the $SRT_{cluster}$ equation applied to the CREDO hierarchy built from Yahoo! results. In Figure 5, we depict the difference between Yahoo! SRT and CREDO SRT on the individual topics. When the bar is above (below) the x axis, it means that Yahoo! took longer (shorter) than did CREDO. Figure 5 clearly shows that CREDO outperformed Yahoo!; it achieved better performance than Yahoo! for as many as 40 topics, with slightly worse values in the remaining 4.

The average SRT value computed over the set of topics was 22.4668 for Yahoo! and 13.0603 for CREDO, with a decrease in time when passing from Yahoo! to CREDO equal to 58.13%. The difference was statistically significant, based on a two-tailed paired t test ($p < .001$). We also computed a normalized version of SRT, by dividing the value associated with each topic by the number of a topic's retrieved subtopics. We observed a very similar behavior. The normalized SRT values for Yahoo! and CREDO were 2.9607 and 1.7549, respectively, with a gain of 59.27%.

As stated in the preceding section, the computation of $SRT_{cluster}$ reflects a best-case scenario in which the cluster with the shortest subtopic reach time is always selected. To address this limitation, we performed another series of experiments. First, for each topic and for each of its retrieved subtopics, we manually chose the cluster (among those displayed by CREDO) whose label most closely matched the corresponding Wikipedia subtopic formulation. We then checked whether the chosen cluster indeed contained at least one document relevant to the subtopic with which it was associated. The answer was positive for 330 of the total 349 subtopics. This finding is encouraging because it tells us that by selecting the cluster with the most appropriate label, we will almost always find some relevant document; however, as we do not know the cluster position and the position of the relevant document within the cluster, the SRT value might still be negatively affected.

To gain more insight into this aspect, we computed, for each of these 330 subtopics, a modified version of SRT in which the cluster with the best semantic label was chosen rather than the optimal cluster. We then computed the average SRT value over the set of topics, using the optimal SRT values for the remaining 19 subtopics. We obtained an average SRT value of 13.8585, which is very close to the optimal average SRT value of 13.0603, thus implying that the clusters with a manually selected label yield similar retrieval performance to optimal clusters. On the whole, these results substantially

DOI: 10.1002/asi

confirm the superiority of CREDO over Yahoo! even when the assumption of optimal cluster selection is relaxed using a more natural criterion.

Subjective Evaluation of CREDO, Credino, and SmartCREDO

The presented objective evaluation measured the effectiveness of search results clustering in a theoretical manner by modeling the user behavior and without considering the effect of the search device being used. As the focus of this article is on mobile information retrieval, it is interesting to understand if and how much our approach is effective when the user is explicitly involved in the search process and he or she makes use of a mobile device. To this aim, we designed another, complementary evaluation: A comprehensive user experiment aimed to evaluate how the choice of retrieval method (clustering or plain search) and device (desktop computer, PDA, or mobile phone) affects retrieval performance. The principal goals of this experiment are (a) to compare clustering and ranked list on a retrieval task across multiple search devices, and (b) to check if the effectiveness of retrieval (whether of clustering or ranked list) decreases as smaller devices are used. The latter hypothesis is often assumed with little evidence as support.

In the next section, we describe the experimental setup, then present the results.

Experimental Setup

The description is broken down into four parts: which retrieval systems were used, who did the search and which information tasks were performed, how retrieval performance was measured, and which specific hypotheses were tested.

Systems. We used six systems in the experiment: three clustering engines and three search engines (i.e., for each device, we had a specific pair of clustering/search engines). To minimize the effect of different interfaces on performance, the search engines used in the experiment were obtained from the corresponding clustering engines by switching off the clustering module. The complete list of systems follows.

- CREDO (*Desktop-Clustering*).
- Credino (*PDA-Clustering*).
- SmartCREDO (*Mobphone-Clustering*).
- A desktop search engine obtained from CREDO (*Desktop-Search*).
- A PDA search engine obtained from Credino (*PDA-Search*).
- A mobile phone search engine obtained from SmartCREDO (*Mobphone-Search*).

The mobile browsers and devices used in our experiment are the Pocket Internet Explorer Web browser running on a Windows Mobile 2003 iPAQ PocketPC 5450 (240×320 pixel screen resolution) and the built-in browser running on LG U8380 (176×220 pixel), for PDA and mobile phone, respectively.

Participants and tasks. We tested 72 participants in the experiment. They were computer science students or young faculty members at the University of Udine, with good experience in computers, mobile devices, and search engine usage. As no participant was aware of the CREDO, Credino, and SmartCREDO systems before the experiment, and more than 90% of the participants were not users of any clustering engine, they were trained: Participants were shown the system usage by the experimenter, and they performed a pilot task.

We used the four following tasks, which represent various types of Web searches (e.g., navigational, transactional, and informational) and are characterized by different levels of term ambiguity and difficulty.

T1. "Your task is to find the Web site of the worldwide institution regulating the chess game."
T2. "Imagine you are a tourist going to visit Potenza.[3] You are interested in finding information about available accommodations, in particular you want to book a hotel room online."
T3. "You have to find a friend of yours which is on holidays in South Italy. You cannot reach him by phone, and you only know that he is in a place called 'Canasta' (and you do not know if it is a hotel, camping, village, etc.)."
T4. "Imagine that you have to find information concerning some athletes in the Boxing world. You are looking for an English-language web site that would allow you to search by name, weight, nationality, etc."

Experiment design. In our experimental setting, there are two independent variables (device and method) and one dependent variable (the retrieval performance on the set of tasks). The 72 participants were randomly split into three groups with 24 people each, with each group being assigned to one device only (i.e., desktop computer, PDA, or mobile phone), and each participant in each group used the two methods (clustering and search) to perform all four finding tasks on the group's device. To minimize learning effects, each participant performed each task once, using one method for the first half of the tasks and the other method for the second half of the tasks. Furthermore, we varied the order of the two methods over the tasks set.

Performance measures. It is well known that evaluating the effectiveness of interactive IR systems is a difficult task, for which there are no standard metrics available (Berenci, Carpineto, Giannini, & Mizzaro, 2000). We took into account the two main aspects of the overall retrieval performance, namely, *success* and *speed*. *Success* represents the degree to which the goal has been accomplished and was measured using a 4-point rating scale: 0 (*unsuccessful*), 1/2 (*partially successful*), 3/4 (*almost successful*), and 1 (*fully successful*). The success was judged by the experimenter, who scored the results; most of the participants received either a 0 or a 1. *Speed* was computed as an inverse function of the time

[3]Potenza is an Italian city.

DOI: 10.1002/asi

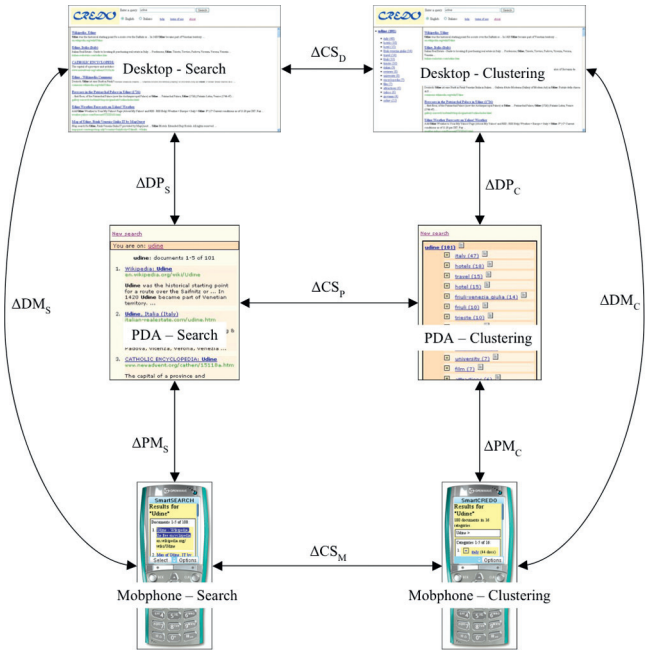

FIG. 6. The six systems tested in the experiment, and the nine "Δs."

taken to complete the task, normalized by a task's maximum admissible execution time. A single numeric effectiveness value was then used by taking their product; that is,

$$Performance = Success \times Speed \quad (1)$$

This measure applies to all retrieval methods and devices. In the rest of the article, we represent the differences in performance (as given by Equation 1) by a ΔXY_Z notation, where X and Y are the initials of the pair of methods (devices) being compared and where Z is the initial of the device (method) being used. For instance, ΔCS_D is the difference between Clustering and plain Search on Desktop computer, and ΔDP_S is the difference between Desktop computer and PDA on plain Search. The complete list of initials is: C (clustering), S (plain search), D (desktop computer), P (PDA), and M (mobile phone). Figure 6 shows the scenario.

Hypotheses. We tested the following hypotheses.

H1: For some information tasks, clustering is more effective than plain search for all search devices.
H2: For some information tasks, plain search is more effective than clustering for all search devices.
H3: The retrieval effectiveness of clustering and plain search on average decreases as smaller devices are used. This amounts to testing the three subhypotheses that desktop has better performance than PDA, desktop than mobile phone, and PDA than mobile phone (averaged over the information tasks).

Results

Table 2 shows the values of *Performance* (Equation 1) obtained for each task by each retrieval method and device, normalized from 0 to 1 and averaged over the subgroup of participants who performed the relative tasks. ΔCS values also are shown. Figure 7 shows graphically the performance values and their differences.

TABLE 2. Median performance by method and device on individual tasks.

Tasks	Devices	Search	Clustering	ΔCS
	Desktop	0.51	0.45306	−0.05694
T1	PDA	0.40667	0.515	0.10833
	Mobile	0.055	0.32167	0.26667
	Desktop	0.34604	0.44417	0.09813
T2	PDA	0.38584	0.4175	0.03166
	Mobile	0.29688	0.48417	0.18729
	Desktop	0.31427	0.45136	0.13709
T3	PDA	0.3325	0.37407	0.04157
	Mobile	0.26979	0.44333	0.17354
	Desktop	0.35602	0.36459	0.00857
T4	PDA	0.14602	0.10282	−0.0432
	Mobile	0	0	0
	Desktop	0.40688	0.45223	0.04535
Grouped	PDA	0.3649	0.42042	0.05552
	Mobile	0.26875	0.44333	0.17458

Another representation of the data is shown in Figure 8, which adds to Figure 6 the performance differences charts for all nine Δs. The three charts in the middle column show the Δs between clustering and search (across devices), and correspond to the chart in Figure 7c; the other six charts show the Δs between pairs of devices (across methods).

The participants were in general able to successfully complete their tasks except for T4, where a substantial number of failures was observed ($success = 0$) and lower values of performance were thus achieved.

Let us first consider the comparison between clustering and search. Considering the individual tasks, the clustering engine outperformed the search engine, with a few exceptions (T1 on desktop, and T4 on PDA and mobile). Considering the four tasks together, the clustering engine approach always performed better than did the search engine approach on all devices (see the rightmost bars in Figure 7c and the lower bar in the three middle charts of Figure 8).

These results confirm the superiority of clustering over plain search, but they also raise the question of why some tasks did not profit as much from our approach. The performance of clustering on T4 was in general low because unlike the first three tasks, the fourth featured a rather bad categorization by CREDO (the relevant results are "hidden" in some nonobviously relevant category name.) and a good performance by classical search (the relevant site is ranked in 5th position.) Note also that cell phone users not only failed when using clustering but they also were unable to take advantage of the theoretically good document ranking provided by plain search (see the third row of T4 in Table 2).

The observed behavior for T1, where clustering did very well on PDA and cell phone and badly on desktop, is more difficult to explain. Although it is fairly reasonable that plain search can perform better than can clustering (especially

DOI: 10.1002/asi

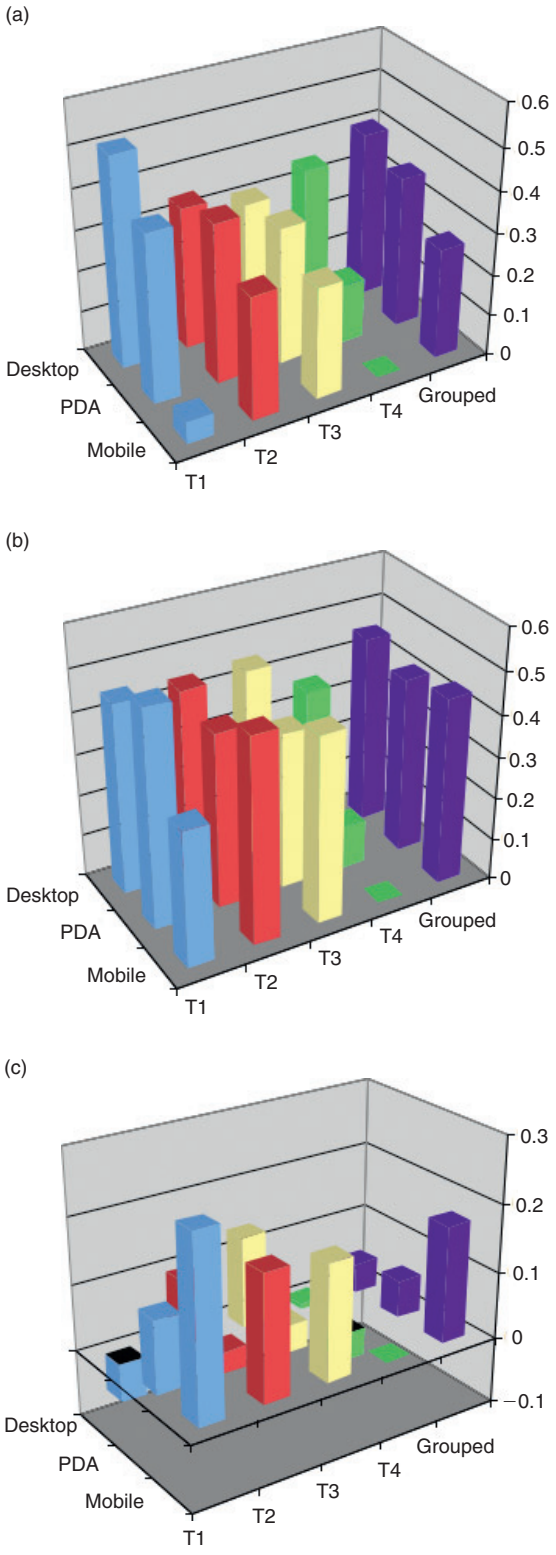

FIG. 7. Median performance for search (a), clustering (b), and ΔCS (c).

when the hits are relevant and the clusters do not match the user's goal well), the results for T1 seem somewhat surprising because the best retrieval method markedly changed depending on the device being used.

Upon closer inspection of T1 ("Your task is to find the Web site of the worldwide institution regulating the chess game."), we found that most of the users were not aware of the Fédération Internationale des Échecs (FIDE); they also were not told that it is a federation rather than an organization. The cluster hierarchy proposed by the system was pretty good since the fourth cluster "world" contained the subcluster "federation," which in turn contained the FIDE Web site. This explains why clustering performed well on mobile devices. On the desktop computer, however, clustering was clearly worse than was plain search. We found that the participants were able to detect relevant snippets on the results page returned by the search method and to come up with effective query-refinement strategies (which account for the excellent performance of plain search) whereas on the PDA device, the screen-size limitations and the typing constraints might have prevented them from doing so in a fast and effective manner. Thus, the results obtained for this topic seem to suggest that mobile clustering may be useful even when the hits returned by plain search are relevant.

We now turn to the statistical significance of the results. We first ran the Shapiro–Wilk normality test, obtaining normality for some data only. We therefore adopted nonparametric tests.

We consider H1 first. Although it does not hold for all combined tasks, it can be shown to hold by considering the first three tasks grouped together. The results are the following.

- ΔCS_D: On the desktop, clustering is more effective than is search, and the difference is statistically significant according to the Wilcoxon nonparametric test ($p < .05$).
- ΔCS_P: On the PDA, clustering is more effective than is search, and the difference is strongly statistically significant according to the Wilcoxon nonparametric test ($p < .01$).
- ΔCS_{DM}: On the mobile phone, clustering is more effective than is search, and the difference is strongly statistically significant according to the Wilcoxon nonparametric test ($p < .01$).

Some of the differences are statistically significant, according to the Wilcoxon test, even on individual tasks: T1 on mobile ($p < .05$); T2 on mobile ($p < .01$) and desktop ($p < .05$); and T3 on mobile ($p < .05$) and desktop ($p < .01$). If we consider all four tasks grouped together (i.e., including T4), clustering is still significantly better than is search on the two mobile devices ($p < .05$ on the PDA, $p < .01$ on the mobile phone).

H2 is not supported by the data. The Wilcoxon test applied to T1 on the desktop and to T4 on the PDA (i.e., where plain search was better than was clustering) showed that the difference is not significant. We do not have any evidence that the search is more effective than is clustering even on single task–device combinations.

Let us now consider the comparison between pairs of different devices for both methods (i.e., the remaining six Δs). Figure 8 shows that in most cases, desktop outperformed both PDA and mobile phone, and PDA outperformed mobile phone: Most of the bars in the six charts point upwards (i.e., they are above zero). However, there were several exceptions: PDA-Search outperformed Desktop-Search on T2 and T3;

DOI: 10.1002/asi

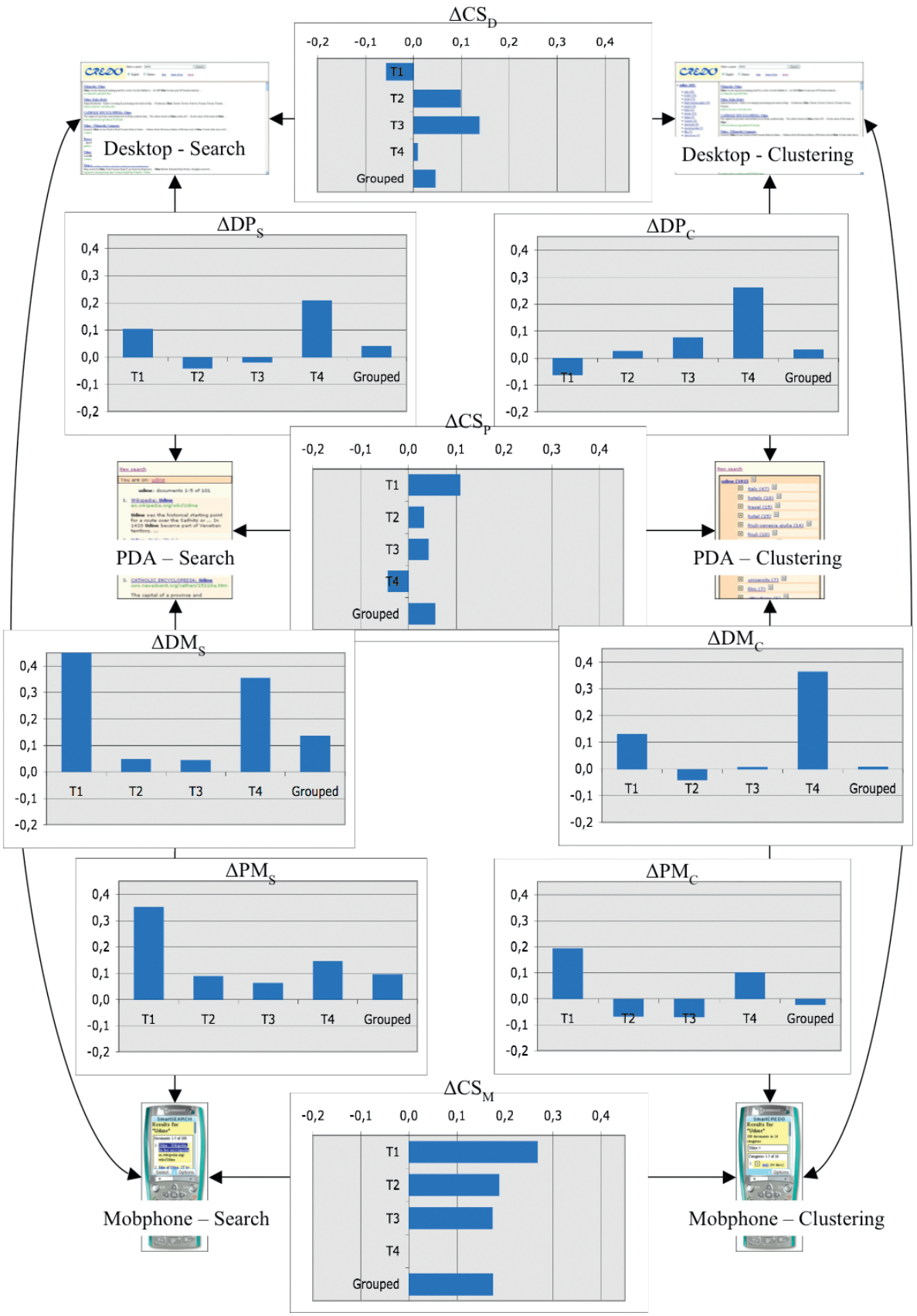

FIG. 8. The nine Δs.

DOI: 10.1002/asi

TABLE 3. The ANOVA results.

Set	Tasks	Kruskal–Wallis χ^2	p
Search	T1	13.68260	.00107**
	T2	2.38360	.30370†
	T3	1.98480	.37070†
	T4	5.96590	.05064†
	Grouped	22.10790	.00002**
Clustering	T1	3.38260	.18430†
	T2	0.45200	.79770†
	T3	2.97980	.22540†
	T4	1.49150	.47440†
	Grouped	3.01250	.22170†
Combined	T1	14.39860	.00075**
	T2	1.54350	.46220†
	T3	2.28780	.31860†
	T4	7.42010	.02448*
	Grouped	21.85380	.00002**

**$p < .01$. *$p < .05$. †not significant.

PDA-Clustering outperformed Desktop-Clustering on T1; Mobile-Clustering outperformed PDA-Clustering on T2 and T3 and on the four aggregated tasks; and Mobile-Clustering outperformed Desktop-Clustering on T2. In addition, differences over devices seem more manifest for the search approach, which presents only two exceptions on individual tasks, whereas the clustering approach presents four exceptions on individual tasks and one exception even on the four grouped tasks.

To verify H3, we grouped the data in three sets: the Search set (with the search performance data), the Clustering set (with the clustering performance data), and the Combined set (with both the search and clustering performance data), and we further divided each set into three groups: Desktop, PDA, and Mobphone. We then performed a variance analysis, by means of the Kruskal–Wallis test, on the three groups in each set.

The results, shown in Table 3, demonstrate that the differences between devices on the four grouped tasks are significant in the Search and Combined sets whereas they are not significant in the Clustering set. A significance analysis on all pairs of devices by means of the Wilcoxon test substantially confirmed these results: When considering the four tasks grouped together, significant differences are obtained only for Search ($p < .01$ for all three comparisons) and Combined ($p < .01$ for Mobile-PDA and Mobile-Desktop; $p < .05$ for PDA-Desktop) sets, and only sometimes on individual tasks. There was no significance in the Clustering set. In other words, H3 is confirmed if we consider plain search alone, and if we group together search and clustering data. We do not have significance on the clustering data alone.

To conclude, note that the tasks being performed implied some form of goal-directed search rather than standard information retrieval. Kamvar and Baluja (2006) observed that there currently is little exploration in mobile search, probably due to the time it takes for the user to move beyond the single-query mode. Our findings suggest that the adoption of search results clustering may make mobile, goal-directed searches more feasible and faster, thus helping to increase the range of information needs which users can satisfy through a mobile device.

Method Parameters and Generality of Results

The CREDO server requires the user to specify the language in which his or her query will be forwarded to the external search engine, with the option of choosing between English or Italian. In the current implementation, the choice of language does not affect the preprocessing of search results prior to clustering because the tokenization and stop-wording steps are the same while stemming is not performed in either case. In general, however, these operations will depend on the language. Apart from the language option, there are two main parameters involved in our method: the number of search results and the maximum number of top-level clusters. They are briefly discussed next.

The CREDO server collects the first 100 search results retrieved by the search engine. This is a good compromise between having few results (which can be better examined sequentially and are possibly too homogeneous in content to give rise to useful subgroups) and many results (which are possibly not relevant and cannot be processed with reasonable response times).

The number of top-level clusters generated by CREDO and its mobile versions varies with the query. As clusters containing very few search results are allowed in principle, this number may become relatively large for some queries even if we consider only the words in the title. In this case, it seems convenient to restrict the set of clusters generated by the system because the user would have to scan a long cluster list and to discriminate between clusters with similar content. On the other hand, choosing a very small number of admissible clusters at the top level also has shortcomings because the user may get many heterogeneous results mixed together in one large cluster. Considering that we process only the first 100 search results, we decided to limit the number of clusters to 15 (excluding the "other" cluster). In this way, it is possible to discriminate between a sufficiently large number of distinct and relatively numerous topics.

It also is useful to look at the scope of our findings. In both experiments, we considered only Web searches, although mobile users are often presented with the option of searching several information repositories, including local and image collections and the mobile Web (e.g., WML pages). Search results clustering can be still applied, in principle, but its utility may be lower because the results of searches on the latter repositories are usually tailored for presentation on mobile devices. We not only considered Web data but also focused on ambiguous informational queries on such data (at least in the first experiment). However, this is probably not a fundamental restriction because informational queries account for 80% of Web queries (Jansen, Booth, & Spink, 2008), and virtually any query expressed by one word has multiple aspects (or components) on the Web. Finally, note that in the second experiment we used only four tasks, although they were

DOI: 10.1002/asi

designed to be representative of typical Web search types. In this way, we have been able to observe the behavior of a comparatively large set of participants, but it did not allow us to investigate in depth the differences between tasks. More generally, this limitation calls into question the generalizability of our findings to other types of search tasks that may be specific to the mobile scenario. This is left for future work.

Conclusions

We have shown that mobile search results clustering is both feasible and effective. In particular, our results support the view that mobile clustering engines can be faster and more accurate than the corresponding mobile search engines, especially for subtopic retrieval tasks. We also found that although mobile retrieval becomes, in general, less effective as the search device gets smaller, the adoption of clustering may help expand the usage patterns beyond mere informational search while mobile.

This research can be extended in several ways. The proposed mobile clustering engines can be technically improved both on the CREDO server side (e.g., by investigating more effective techniques for concept-based clustering such as phrase-based indexing of search results and nondeterministic formation of concepts) and on the mobile device client (e.g., by re-implementing the clustering interface under more powerful mobile platforms that have recently emerged, such as Google's Android and iPhone SDK). In addition, it would be useful to collect more evidence about the retrieval performance of search results clustering versus ranked list by experimenting with a larger and diversified set of search tasks—and this can be done in a more effective and reliable way as soon as more data about real user needs, such as those mentioned earlier, are available. Finally, we plan to compare the cluster hierarchy produced by CREDO to those of different search results clustering systems using the AMBIENT dataset made available in this article, and to evaluate Credino and SmartCREDO against alternative mobile clustering engines (as soon as they become available) and/or other nonconventional mobile IR techniques.

In conclusion, we recall the URLs of the three prototypes and the test collection:

CREDO: http://credo.fub.it/
Credino: http://credino.dimi.uniud.it/
SmartCREDO: http://smartcredo.dimi.uniud.it/
AMBIENT dataset: http://credo.fub.it/ambient/

Acknowledgments

We thank Matt Jones, Stanislaw Osinski, Barry Smyth, Dawid Weiss, and the anonymous referees for many useful comments on earlier versions of this article. We also are indebted to Andrea Della Pietra, Luca Di Gaspero, and Annalisa Filardo for their help with preparation of the experiments.

References

Berenci, E., Carpineto, C., Giannini, V., & Mizzaro, S. (2000). Effectiveness of keyword-based display and selection of retrieval results for interactive searches. International Journal on Digital Libraries, 3(3), 249–260.

Buyukkokten, O., Garcia-Molina, H., & Paepcke, A. (2001). Seeing the whole in parts: Text summarization for web browsing on handheld devices. Paper presented at the 10th International WWW Conference (WWW10), Hong Kong. (pp. 652–662). Retrieved February 2, 2008, from http://infolab.stanford.edu/~orkut/papers/pb5.pdf

Carpineto, C., Della Pietra, A., Mizzaro, S., & Romano, G. (2006). Mobile clustering engine. In Proceedings of the 28th European Conference on Information Retrieval (pp. 155–166). Berlin, Germany: Springer.

Carpineto, C., Osinski, S., Romano, G., & Weiss, D. (in press). A survey of Web clustering engines. To appear in ACM Computing Survey.

Carpineto, C., & Romano, G. (2004a). Concept data analysis—Theory and applications. Chichester, United Kingdom: Wiley.

Carpineto, C., & Romano, G. (2004b). Exploiting the potential of concept lattices for information retrieval with CREDO. Journal of Universal Computer Science, 10(8), 985–1013.

Chen, H., & Karger, D.R. (2006). Less is more: Probabilistic models for retrieving fewer relevant documents. In Proceedings of the 29th annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Seattle, WA (pp. 429–436). New York: ACM Press.

Cheng, D., Vempala, S., Kannan, R., & Wang, G. (2005). A divide-and-merge methodology for clustering. In C. Li (Ed.), Proceedings of the 24th ACM Symposium on Principles of Database Systems (pp. 196–205), Baltimore. New York: ACM Press.

Church, K., & Smyth, B. (2007). Mobile content enrichment. In Proceedings of the 12th International Conference on Intelligent User Interfaces (pp. 112–121). New York: ACM Press.

Church, K., Smyth, B., Cotter, P., & Bradley, K. (2007). Mobile information access: A study of emerging search behavior on the mobile internet. ACM Transactions on the Web, 1(1), 1–38.

Cutting, D.R., Pedersen, J.O., Karger, D., & Tukey, J.W. (1992). Scatter/gather: A cluster-based approach to browsing large document collections. In Proceedings of the 15th annual International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 318–329), Copenhagen, Denmark. New York: ACM Press.

Di Giacomo, E., Didimo, W., Grilli, L., & Liotta, G. (2007). Graph visualization techniques for Web clustering engines. IEEE Transactions on Visualization and Computer Graphics, 13(2), 294–304.

Ferragina, P., & Gulli, A. (2004). The anatomy of SnakeT: A hierarchical clustering engine for Web-page snippets. In Proceedings of the 8th European Conference on Principles and Practice of Knowledge Discovery in Databases (pp. 506–508). New York: Springer-Verlag, New York, USA.

Ferragina, P., & Gulli, A. (2005). A personalized search engine based on web-snippet hierarchical clustering. In Proceedings of the 14th World Wide Web Conference (pp. 801–810), New York: ACM Press.

Ganter, B., & Wille, R. (1999). Formal concept analysis: Mathematical foundations. Berlin: Springer-Verlag.

Geraci, F., Maggini, M., Pellegrini, M., & Sebastiani, F. (2007). Cluster generation and cluster labelling for Web snippets: A fast and accurate hierarchical solution. Internet Mathematics, 3(4), 413–444.

Hearst, M.A., & Pedersen, J.O. (1996). Reexamining the cluster hypothesis: Scatter/gather on retrieval results. In Proceedings of the 19th ACM International Conference on Research and Development in Information Retrieval (pp. 76–84). New York: ACM Press.

Hersh, W.R., & Over, P. (1999). TREC-8 Interactive Track Report. In E.M. Vorhees & D.K. Harman (Eds.), Proceedings of the Eighth Text REtrieval Conference (TREC-8) (pp. 57–64). Gaithersburg, MD: National Institute of Standards and Technology (NIST).

Jansen, B.J., Booth, D.L., & Spink, A. (2008). Determining the informational, navigational, and transactional intent of Web queries. Information Processing and Management, 44(3), 1251–1266.

DOI: 10.1002/asi

Jones, M., Buchanan, G., & Mohd-Nasir, N. (1999). Evaluation of WebTwig—A site outliner for handheld web access. In Proceedings of the International Symposium on Handheld and Ubiquitous Computing, Karlsrhue, Germany. Lecture Notes in Computer Science, 1707, 343–345. Berlin: Springer-Verlag.

Jones, M., Buchanan, G., & Thimbleby, H. (2002). Sorting out searching on small screen devices. In Proceedings of the 4th International Symposium on Mobile Human–Computer Interaction (pp. 81–94).

Jones, M., Jain, P., Buchanan, G., & Marsden, G. (2003). Using a mobile device to vary the pace of search. In Proceedings of Human–Computer Interaction with Mobile Devices and Services (Vol. 2795, pp. 390–394), Udine, Italy.

Jones, S., Jones, M., & Deo, S. (2004). Using keyphrases as search result surrogates on small screen devices. International Journal of Personal and Ubiquitous Computing, 8(1), 55–68.

Kamvar, M., & Baluja, S. (2006). A large scale study of wireless search behavior: Google mobile search. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (pp. 701–709). New York: ACM Press.

Kamvar, M., & Baluja, S. (2007). Deciphering trends in mobile search. IEEE Computer, 40(8), 58–62.

Kamvar, M., & Baluja, S. (2008). Query suggestions for mobile search: Understanding usage patterns. In Proceedings of CHI 2008 (pp. 1013–1016), Florence, Italy.

Kummamuru, K., Lotlikar, R., Roy, S., Singal, K., & Krishnapuram, R. (2004). A hierarchical monothetic document clustering algorithm for summarization and browsing search results. In Proceedings of the 13th International Conference on World Wide Web (pp. 658–665). New York: ACM Press.

Lawrie, D.J., Croft, B.W., & Rosenberg, A. (2001). Finding topic words for hierarchical summarization. In Proceedings of the 24th International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 349–357), New Orleans. New York: ACM Press.

Lawrie, D.J., & Croft, W.B. (2003). Generating hierarchical summaries for web searches. In Proceedings of the 26th annual International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 457–458).

Maarek, Y.S., Fagin, R., Ben-Shaul, I.Z., & Pelleg, D. (2000). Ephemeral document clustering for Web applications (Tech. Rep. No. RJ 10186). San Jose, CA: IBM Research.

MarketingVOX (2005). Mobile search growth has implications for marketers. Retreived February 2, 2009, from http://www.marketingvox.com/archives/2005/07/28/mobile_search_growth_has_implications_for_marketers/

Noirhomme-Fraiture, M., Randolet, F., Chittaro, L., & Custinne, G. (2005). Data visualizations on small and very small screens. In Proceedings of Applied Stochastic Models and Data Analysis. Available at: http://asmda2005.enst-bretagne.fr/

Osinski, S., Stefanowski, J., & Weiss, D. (2004). Lingo: Search results clustering algorithm based on singular value decomposition. In M.A. Kłopotek, S.T. Wierzdnón, & K. Trojanonski (Eds.), Proceedings of the International Intelligent Information Processing and Web Mining Conference (pp. 359–368). Berlin, Germany: Springer.

Osinski, S., & Weiss, D. (2005). A concept-driven algorithm for clustering search results. IEEE Intelligent Systems, 20(3), 48–54.

Otterbacher, J., Radev, D., & Kareem, O. (2006). News to go: Hierarchical text summarization for mobile devices. In Proceedings of the 29th annual International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 589–596). New York: ACM Press.

Priss, U. (2006). Formal concept analysis in information science. Annual Review of Information Science and Technology, 40, 521–543.

Schütze, H., & Silverstein, C. (1997). Projections for efficient document clustering. In Proceedings of the 20th ACM International Conference on Research and Development in Information Retrieval (pp. 74–81). New York: ACM Press.

Sohn, T., Li, K.A., Griswold, W.G., & Hollan, J.D. (2008). A diary study of mobile information needs. In Proceedings of CHI 2008 (pp. 433–442), Florence, Italy.

Spiliopoulou, M., Schaal, M., Müller, R.M., & Brunzel, M. (2005). Evaluation of ontology enhancement tools. In M. Ackermann, B. Berendt, M. Grobelnik, & D. Mladenic (Eds.), Proceedings of the Semantics, Web and Mining, Joint International Workshops, EWMF 2005 and KDO 2005 (pp. 132–146). Berlin, Germany: Springer.

Sweeney, S., & Crestani, F. (2006). Effective search results summary size and device screen size: Is there a relationship? Information Processing and Management, 42, 1056–1074.

Zamir, O., & Etzioni, O. (1998). Web document clustering: A feasibility demonstration. In Proceedings of the 21st International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 46–54), Melbourne, Australia. New York: ACM Press.

Zamir, O., & Etzioni, O. (1999). Grouper: A dynamic clustering interface to Web search results. Computer Networks, 31(11–16), 1361–1374.

Zeng, H.-J., He, Q.-C., Chen, Z., Ma, W.-Y., & Ma, J. (2004). Learning to cluster Web search results. In Proceedings of the 27th ACM International Conference on Research and Development in Information Retrieval (pp. 210–217), Sheffield, United Kingdom. New York: ACM Press.

Zhai, C., Cohen, W.W., & Lafferty, J. (2003). Beyond independent relevance: Methods and evaluation metrics for subtopic retrieval. In Proceedings of the 26th International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 10–17), Toronto. New York: ACM Press.

DOI: 10.1002/asi