ScoDA: Cooperative Content Adaptation Framework for Mobile Browsing

Ayush Dubey, Cornell Uniersity. Email: dubey@cs.cornell.edu
Pradipta De, SUNY Korea. Email: pradipta.de@sunykorea.ac.kr
Kuntal Dey, Sumit Mittal, Vikas Agarwal, Malolan Cheltur, Sougata Mukherjea, IBM Research India. Email: {kuntadey,sumittal,avikas,mchetlur,smukherj}@in.ibm.com

***Abstract*—Mobile browsing habits are characteristically different from browsing on traditional devices. Mobile users often look for information snippets instead of complete web pages. Also, mobile devices are often constrained in terms of resource availability, such as battery, data plan limits and network bandwidth. Under such constraints, partial-loading of a web page, by loading the most relevant content snippets early, can satisfy the user and consume less resources. Mobile content adaptation middleware has traditionally focused on user factors, such as user feedbacks and user context. We believe that the content creator, with complete knowledge of the importance of each item in the web page, is well-suited to guide the adaptation process. Combining user choice, and ratings assigned by the content creator to different web page elements (items), enables delivering the most relevant items in an ordered manner in response to a page request. We present ScoDA, a cooperative content adaptation middleware framework, under resource constraints. We evaluate the effectiveness of page loading using ScoDA, on simulated complex web pages as well as real web pages.**

I. Introduction

Mobile devices today generate a significant portion of the worldwide web traffic. Browsers continue to be one of the most popular applications on mobile devices. However, mobile browsing behavior is characteristically different from desktop browsing [12], [19], [21]. Mobile users tend to seek small snippets of easily consumable information, typically closely tied to the user's context.

Web pages have evolved in complexity with various embedded media content. Bloated web pages take longer to download on mobile, consuming more resources [23]. The mobile web tends to suffer from resource constraints, like data usage cap, low network bandwidth, limited battery power. To view a desired snippet on a resource-constrained mobile device, a user is often forced to wait for an entire page to load. On mobile web, where the average connection speed can vary from 324 Kbps to 7.8 Mbps [2], user experience varies significantly.

Although several companies design multiple web experiences by designing mobile compatible pages, and despite attempts of auto-generating pages adapted for mobile browsing [13], studies observe that over 50% of page views on smartphones are for non-optimized pages [25]. Ubiquitous connectivity leads to diverse and dynamic user contexts and varying content preference for web pages.

Web content adaptation has been an ongoing area of research to adapt to the device, network and browser variability. Adaptation to manage scarce resources, like limited data download plans, is addressed by [3], [6]. On use of context and user preference, Urica uses explicit user preference [17], while Mimosa automatically determines user context to adapt the content [15]. However, none of the web content adaptation approaches include the content creator, who is well-aware about the semantic importance of the content.

We propose ScoDA, a heuristic optimization framework for selecting the items to be displayed in a page, by factoring in the scores assigned by the user and content developer, under multiple resource constraints. ScoDA aims to maximize the selection of high rated items from both user and developer perspectives, using user and developer hints, under resource constraints. Items to deliver are selected from an ordered list, delivering more important items first. Using simulated and real web pages, we show the effectiveness of ScoDA in terms of (a) efficiency and (b) capability to deliver more relevant content compared to default browsing, under resource constraints. In summary, our contributions in this work are as follows.

1) We propose the ScoDA framework to jointly optimize the combination of developer and user preferences of items while loading web pages under resource constraints, and adapting towards different user contexts.
2) We jointly satisfy the user and developer at the earliest possible in the page load process, loading items with the highest combined developer and user preferences earliest, given resource constraints.
3) We empirically validate our proposition with experiments on multiple diverse real-life web pages and complex simulated web pages with thousands of items.

Our framework assumes the knowledge of user context to specify user preferences using existing solutions, and the ability to determine resource requirements for different contents. We refer to existing literature to address these issues [7], [9], [26], and note that these solutions can provide such input to our system.

II. Related Work

Web content adaptation has been a focus of research even before the demands of the mobile era. In the context of our work, we review the literature addressing web content adaptation in mobile and pervasive environments. A broad classification of web content adaptation is covered by Futzee and Abawajy [16]. The main body of work in mobile web

content adaptation can be split into device-centric and user-centric adaptations.

Device independent layout adaptation attempting to make web content suitable across diverse display sizes has been one of the earliest challenges [4], leading to commercial solutions like IBM Websphere Transcoding Publisher [11]. With emergence of smartphones, enabling touch based pan-zoom features, page loading without display adaptation is more acceptable. Though many popular web sites, like Facebook, Twitter, have customized mobile pages, a vast majority of the web pages in the Internet lack customization for layouts.

Adaptation focused on optimizing resource usage, like energy and network have gained more attention in the mobile scenario. In order to operate optimally under metered data plans, Chava et al. proposes a scheme that adapts content based on residual data download capacity [6]. Techniques for energy usage has focused on offloading compute intensive tasks to the server. Amazon's Silk browser is a commercial implementation of the concept [22]. Diverse resources, like energy and network can be managed jointly as shown in the MaJaB system [14]. In addition to these, caching has been widely used to optimize resource usage in web content adaptation.

User-centric web content adaptation technique either puts the user explicitly in the adaptation loop, or attempts to infer a suitable selection based on user's context. [1] uses the context of the user to adapt multimedia content. MIMOSA is another context aware adaptation system which collects user contexts using the sensors on a device to drive the adaptation [15]. There are also other indirect ways of inferring user intent. Mohomed et al. present a technique where preference history of past users for a page is used to adapt the content for other users [18]. On the other hand, URICA proposes to give the control to the user to adapt content till she is satisfied [17]. It uses a feedback collection mechanism to better adapt content for the user.

Harumoto et al. [10] proposes an authoring tool for user and content author to specify their preference with respect to the page content. This technique complements our approach since collection of user preference and hints from authors in a systematic manner is essential to drive the decision process to select and order items from a page. This work also proposes a policy driven approach for adaptation using hints from user and developer. Unfortunately, it does not attempt to optimize the selection process. Our main contribution is to generate a selection of items that maximizes the combined satisfaction of the user and developer within resource constraints.

Table I summarizes the literature, along with our proposed framework ScoDA.

Literature	Resource -aware	User Preference	User Conext	Developer Hint
MIMOSA [15]	√	×	√	×
URICA [17]	×	√	×	×
Mohomed et al. [18]	×	√	×	×
MaJaB [14]	√	×	×	×
Harumoto [10]	×	√	×	√
ScoDA	√	√	√	√

TABLE I. COMPARISON OF MOBILE WEB CONTENT ADAPTATION LITERATURE

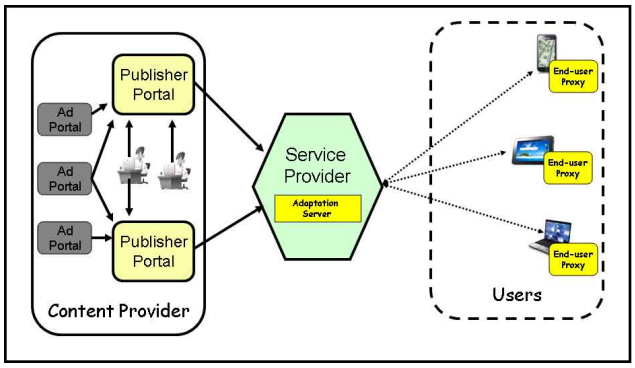


Fig. 1. Different parties involved in the content dissemination process in the mobile ecosystem.

III. CONTENT ADAPTATION MODEL

In this section, we present the system model for the content adaptation middleware. We introduce the different entities in the ecosystem, followed by an activity model of the system.

A user initiates a web page request from a browser application on a mobile device (e.g. smartphone or tablet PC). The URL request is routed through the infrastructure of a service provider to the publisher portal's web server. A publisher portal responds to the URL request with the complete content of a page. Fig. 1 shows the setup and the different entities in the process.

A requested HTML page contains blocks of text, along with resource links to fetch the other content embedded in the page. For example, when one visits a news site, first the index HTML page is downloaded. As the browser parses the HTML document, it triggers request for images, video, audio, and advertisements embedded in the page. Advertisements may be sourced from other advertisement portals, like Google AdSense. The request for the embedded resources is issued during the default HTML parsing order. Downloading the resources (like images, advertisements, video, audio) consume bulk of the bandwidth. In our design, these are the items which are sent to the client in an intelligent order, instead of following the HTML parse order.

In order to enable reordering of the resource requests, all blocking calls to fetch the resources are converted to AJAX calls by the Service Provider. Thus the browser can initiate all calls to fetch the resources, but the requests are intercepted by the adaptation server hosted by the service provider. The browser only receives the layout resource (css files) which allows it to display the text, while the other resources are being fetched. The adaptation server computes the order of delivering each resource item based on the preferences for each item marked by the user and the content provider. The mechanism to assign preference scores are discussed in Section **??**. The resource items are delivered one by one, thus consuming the entire available bandwidth between between provider and user, instead of splitting the bandwidth across multiple threads which have requested resources. The browser displays each item as it is received at the user device. The flow of activity,

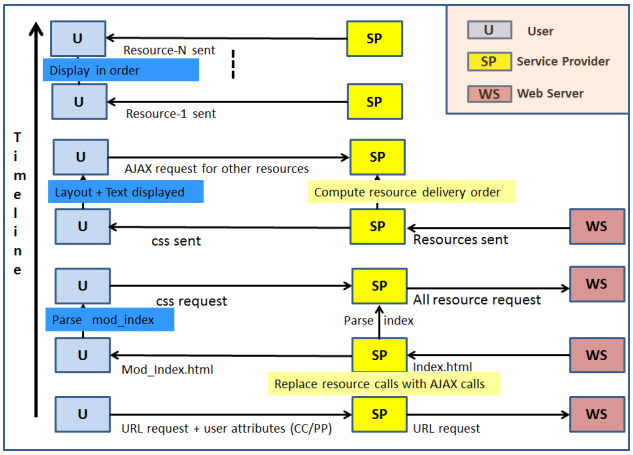


Fig. 2. Activity Diagram showing the adaptation process.

with actions performed at each entity, is shown in Fig. 2.

There are two choices with respect to the placement of the adaptation server: as a web proxy on the client device, or hosted by the service provider. In a client based proxy design, the client must download all the resources, thereby defeating the goal of saving download bandwidth. Therefore, the adaptation server is hosted by the service provider, which can download all the resources using faster backend connectivity. On the device end, we still require an end-user proxy. End-user proxy gathers device specific information, like battery status, number of active threads, and available working memory. The proxy attaches these information to the URL request sent to adaptation server to use during adaptation. Using the Composite Capabilities/Preferences Profile (CC/PP) model, the device information exchange between the end user and the adaptation server can be easily enabled [5]. The adaptation server mainly uses the information from the device to infer its resource limits.

IV. Problem Formulation

We now present the generalized formulation of the content selection problem, given user and developer scores for each item, and a budget on each resource type to load a page. In Section V, we shall present our proposition, ScoDA, and subsequently investigate its effectiveness empirically in our setting, comparing with the generalized formulation presented here.

Let there be m items in a page consisting of items, such as images, videos, audios, and advertisements. Let there be n resources for which the maximum usage budget is specified as R_j, where $j \in \{1, 2, \ldots, n\}$. Let r_{ij} be the resource j required for item i in the web page. Let X be the boolean vector which denotes a specific selection of items, $\{x_1, x_2, ..., x_m\}$ We need to find the selection X which maximizes a utility function, $\mathcal{F}$. Thus the optimization formulation is as follows.

$$arg\ max\ \mathcal{F}(X), \quad x_i \in \{0, 1\} \tag{1}$$

such that,

$$\forall j \in \{1, 2 \ldots, n\} \quad \sum_{i=1}^{m} x_i r_{ij} \leq R_j \tag{2}$$

Our goal is to maximize the user and developer satisfaction, based on their scores for individual items. Therefore the utility function, $\mathcal{F}$, is the sum of the combined scores of user and developer. Formally, $\mathcal{F}$ is represented as,

$$\mathcal{F} = \sum_{i=1}^{m} x_i z_i \tag{3}$$

where z_i represents a composite score (single value) per item by combining the user and developer's score for that item. Let us consider a vector, Z, of size m that contains the z_i values. The Z vector is computed as follows,

$$Z = f(\mathcal{D}, \mathcal{U}) \tag{4}$$

where $\mathcal{D}$ and $\mathcal{U}$ are vectors of size m representing developer and user scores for each item respectively.

The user does not know the page content at item level a priori, therefore, she provides only scores for each type of content. Let the types of items, i.e. image, video, audio, advertisement, in a web page be represented by $\mathcal{T}$. From the type level scores, $\mathcal{S}$, we derive the user score for each item, $\mathcal{U}$, using the following transformation.

$$u_i = s_j \mid \text{item } i \text{ is of item type } \ j \in \mathcal{T}$$

The publisher score vector, $\mathcal{D}$ is as follows:

$$\mathcal{D} = \{d_i\}_{1 \leq i \leq m}$$

wherein d_i gives the relative importance of item i from the point of view of the page developer.

After obtaining the selection of items X, the problem also requires fixing an order of transmission. Let

$$I = \{i_1, i_2, \ldots i_l\}$$

be the set of items selected, as defined by X. The loading order of the items in I is decided by specifying a permutation "π". The result of the content adaptation, i.e. the final ordered set of items, is thus

$$\{i_{\pi(1)}, i_{\pi(2)}, \ldots i_{\pi(l)}\}$$

The choice of π could depend on the composite score vector Z and/or the resource usage matrix R.

A. Complexity

In order to understand the complexity of the problem, we map it to Multi-dimensional 0-1 Knapsack Problem (MKP). MKP can be formulated as,

$$\text{maximize} \sum_{j=1}^{m} c_j y_j,$$

subject to

$$\sum_{j=1}^{m} a_{ij} y_j \leq b_i \quad i = \{1, \ldots, n\}$$

$$y = 0 \text{ or } 1 \qquad j = \{1, \ldots, m\}$$

As is well known, this problem is NP-complete [24]. To show that our problem is also NP-complete, we can map this generic formulation of the multidimensional 0-1 knapsack problem to a particular instance of our problem as follows:

.

- $c_j \rightarrow z_i$
- $a_{ij} \rightarrow r_{ij}$
- $b_j \rightarrow R_j$
- $y_i \rightarrow x_i$

$\Big\} \forall i = \{1, \ldots, m\}, \forall j = \{1, \ldots, n\}$

This shows that the selection of items in the content adaptation problem is at least as hard as MKP.

V. Score Driven Content Adaptation Technique

In this section, we present different approaches to solve the item set selection problem. Since we have formulated the selection of items as a linear optimization problem, we can use a standard solver, like CPLEX [8] to obtain a solution. However, a standard solver is not guaranteed to generate a solution in bounded time. In our design, the selection must be performed on each URL request, therefore, fast computation of the item set is essential.

Since the item set selection problem maps to the multi-dimensional 0-1 knapsack problem, we explore fast heuristics, known to be applicable for solving the multi-dimensional 0-1 knapsack problem, as a solution approach in our setting [20]. In order to maximize the value of selection with minimum resource penalty, it is ideal to pick items in the order of descending value density, since it will pack a larger item at a lower resource cost. Value density for the selection problem can be represented as,

$$\frac{z_i}{\sqrt{\sum_{i=1}^{n} r_{ij}}} \tag{5}$$

A. Score Driven Cooperative Content Adaptation (ScoDA)

We observe that the choice of value density function as shown in Eqn. 5 may prefer an item, that has low score with low resource requirement, over an item with high score but relatively higher resource requirement. The scenario is akin to loading a thumbnail picture before loading the main image in a page, although the main image is rated with a high score compared to the thumbnail. While this may apply in other settings, clearly the generic heuristic solution for multi-dimensional 0-1 knapsack problem does not provide an obvious way to take the domain information such as user and developer intent and resource constraints into account. This has been further substantiated with our experiments in Section VI. This motivates for a different heuristic which will deeply ingrain the domain knowledge and domain constraints into account while solving this problem.

Algorithm 1 ScoDA Heuristic

```
R = {r_ij} ← item-resource usage matrix
for i = 1 → n do
    R_i ← resource limit for i^th resource
end for
U ← user context, D ← developer preferences
Z ← f(D,U)
Z_sort ← sortAndBreakTies(Z)

for k = 1 → m do
    i ← indexOf(Z_sort[k])
    for j = 1 → n do
        if r_ij > R_j then
            continue
        end if
    end for
    X_i ← true {X is the decision vector}
    for j = 1 → n do
        R_j ← R_j − r_ij
    end for
end for
Y = orderedItemList(X, Z_sort, R)

Output: Y
```

With this background, we propose the ScoDA heuristic. This heuristic is designed to select rated items high in a combined manner by the content developer and the user, prior to loading lower rated items, while respecting the resource constraints, with an objective of satisfying both the parties (developer and user) at the earliest possible during the process of loading items on the user's device. In ScoDA, we sort the items based on the composite score, as defined by Eqn. 4, and check for the resource constraint satisfaction as part of the sorting process. In order to compute the composite score, we provide for a function to combine the user and developer score, as follows:

$$Z = f(D, U) \tag{6}$$

An appropriate choice of the function ensures that items rated high by both parties will always be selected first, and so on. The heuristic for computing the item set is presented in Algorithm 1.

For experimental purposes, we have used:

$$Z = f(D, U) = D. * U \tag{7}$$

Other choices for composing the function f includes harmonic or arithmetic mean, LP norm based functions and Lagrange multipliers. With a focus on highlighting the efficacy of the heuristic under an obvious choice, we have chosen the geometric mean to define f. Benefits derived from other choices of the function remains to be explored.

In the proposed heuristic, we sort the values in the Z vector in descending order so that the highest rated items are picked first for selection. Note that multiple items may end up with

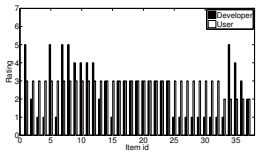


Fig. 3. NEWS site: Scores assigned to individual items by user and developer.

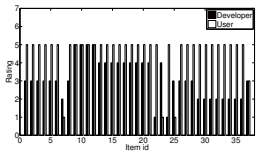


Fig. 4. BLOG site: Scores assigned to individual items by user and developer.

same score. Therefore, we break the tie by comparing the resource usage. Smaller resource usage is given preference. However, it is possible to use other policies to break the tie. For instance, one can use the user score, or developer score for the item; items which are related to the already selected item can also be preferred. Function `sortAndBreakTies` at Line 7 in the algorithm performs this operation. The output of this goes into identifying the load order, which is performed by the function `orderedItemList` at Line 20.

VI. Experimental Evaluation

We evaluate the effectiveness of the proposed adaptation technique under resource constrained environments, using multiple real web pages as well as simulated web pages with thousands of items.

The end-user proxy runs on an Android based smartphone. A URL request triggered from Android browser is redirected to the adaptation server, along with device parameters and user preference values. The requested web page is annotated with developer scores for individual items. Once the adaptation server selects the items, these are streamed back to the end-user proxy in the order specified by ScoDA scheme.

For simulation, we have used the CPLEX solver [8] to generate solutions for different item set sizes. We also run the simulation to select the item set for the same problem set using the chosen heuristic for MKP, and ScoDA adaptation scheme.

A. Adaptation Effectiveness on Web Page

We present our findings here with two real web pages: a page from a news site and from a blog site. The news site contains multiple images, with few large images as part of the main content, and a number of thumbnails. There are also some advertisements, one of which is from the news site, and the rest sourced from third-party providers. We mimic the developer rating by assigning high score to the content images, and low score to the thumbnails. The advertisement from the site is rated high compared to the other ads. We approached an independent user, who assigned scores of 3 and 2 to image and ad types respectively, as she *perceived* reasonable in her setting.

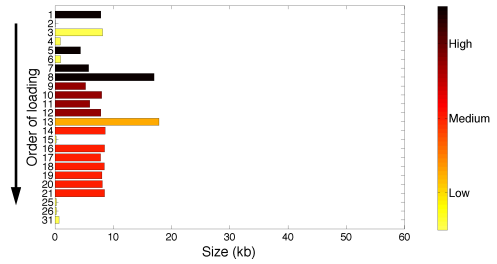
The blog site contains images, videos and ads. The developer score for some images and ads were set to be high. The independent user assigned scores of 5, 3 and 1 to image, video and ad types, respectively.

Fig. 3 and Fig. 4 show the user and developer scores for all the items in the news and blog pages. Fig. 5 shows three views of the web page from the news site. Fig. 5(a) is the page view with no resource constraint when the entire page is loaded; Fig. 5(b) shows the page loaded according to the default HTML parsing order under the resource constraint, while Fig. 5(c) shows the same page adapted using ScoDA. For this experiment, the resource constraint is set in terms of number of bytes that can be downloaded for the page. The cap is set to 40% and 28% of the total page size for the news and blog site respectively. The advertisement, marked in red, is rated high by developer. Hence it is selected during adaptation along with other high rated images. The thumbnails, marked in blue, are dropped in order to accommodate the advertisement. In the no-adaptation setting, the thumbnails are loaded and the advertisement is dropped.

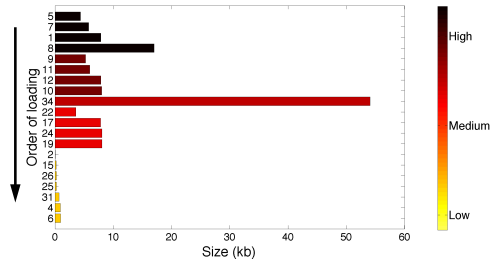
The order in which the items are loaded in the two settings are shown in Fig 6. The length of each bar indicates the size of the item, and the color bar indicates the importance of an item (dark means higher rating) based on user and developer rating. The items are loaded from top to bottom in the graphs. Out of 37 items, ScoDA picked 20 items, compared to 24 items that were loaded by default browser without adaptation under the given resource constraint. However, Fig. 6(b) shows that the items were loaded in the order of their importance in ScoDA, whereas in case of default browsing, relatively less important items were loaded before loading more important items, as shown in Fig. 6(a). This demonstrates that under resource constrained environment ScoDA is more effective in delivering as per user preference, while satisfying the developer.

We performed the same evaluation for the page from the blog site. In this case, we set the resource cap to 28% of the total page size. Fig. 6(c) shows that default browser without adaptation fails to load any of the important items, whereas adaptation using ScoDA loads the important items, as shown in Fig. 6(d).

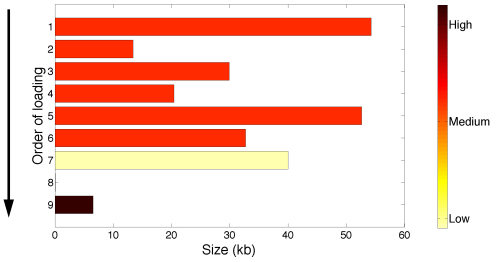
An important observation by comparing the results from the two pages is that if a developer writes the page with the default load order in perspective, then the default setting performs well in loading the important items. For the news site under consideration, the developer had ensured that the main image loads first by invoking it early. However, for the blog site, the importance of items were not considered while designing. The ScoDA framework makes it easier to set the relative importance, and suitably adapts the page such that the important items start loading early. A comparison of Fig. 6(b) and Fig. 6(d) indicates that a large number of important items loaded much earlier compared to the default browsing behavior in case of the not-so-well-designed blog, but the reordering of loading of items was comparatively lesser in case of the well-designed news site. This shows the positive impact of ScoDA while loading pages with less comprehensive design, such as the blog site used in our experiments. In other words, ScoDA makes it easier for webpage developers to focus on developing their core content, rather than forcing them to focus too much on creating layouts that would ensure improved loading order (an aspect that had, in all likelihood, been a significant focus and investment in designing the exemplified news website in the first place, and ironically, could still be significantly improved by ScoDA as indicated by Fig. 6).



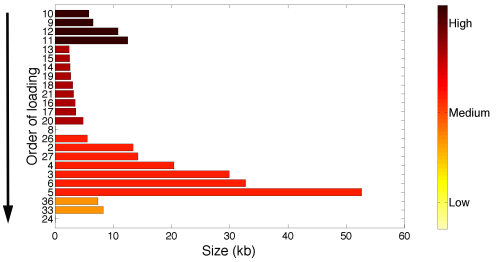
(a) News site: Items loaded without ScoDA



(b) News site: Items selected and loaded using ScoDA



(c) Blog site: Items loaded without ScoDA



(d) Blog site: Items selected and loaded using ScoDA

Fig. 6. Comparison of web page items selected and the loading order with and without using ScoDA adaptation.

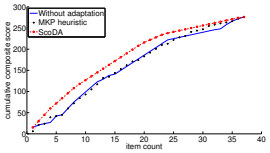


Fig. 7. News page: Comparison of increase in user and developer satisfaction with gradual item loading

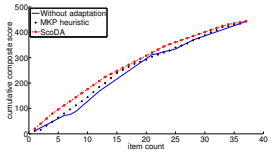


Fig. 8. Blog page: Comparison of increase in user and developer satisfaction with gradual item loading

We evaluate the satisfaction (for user and developer) by measuring the growth in composite score (c.f. Eqn. 7) as items are loaded for the web pages. Fig. 7 and Fig. 8 show that for both the sites ScoDA increases the satisfaction metric with loading of each item. MKP heuristic, which selects and orders items based on value density (c.f. Eqn. 5), often selects items which may be low in importance, but are also smaller in size.

B. Analysis of Item Ordering

We analyze the effect of ordering the items once the items are selected. Without adaptation, the loading order is determined by the HTML parsing order. For MKP heuristic and ScoDA, loading order is determined by sorting (descending) the items on the basis of the composite score of each item. For MKP heuristic composite score is the value density, given by Eqn. 5; for ScoDA it is given by Eqn. 7. Fig. 9 shows temporal progression of loading of different items from the news site web page. Due to space constraints, we skip the corresponding graph for blog site web page. Fig. 9(a) shows the total number of items loaded by ScoDA is lower than the other two methods. ScoDA picks score-5 items to load first, as shown in Fig. 9(b). Since these items are larger in size for this page, hence ScoDA lags behind in terms of number of items loaded. Fig. 9(c) shows that in the other approaches, score-1 items are loaded much before ScoDA. Note that a score-5 item is much more relevant to the user than the other items, hence ScoDA delivers contents to the user in order of higher relevance.

We also observe that under resource constraints ScoDA performs better in loading important items. We see that at any moment of time (e.g. the vertical green line at t=4s in Fig. 9(b) and Fig. 9(c)), ScoDA has loaded more of higher rated items. In this particular case, we loaded a score-5 ad by forgoing a few score-1 thumbnails. Even though this advertisement was large in size, but it was important for developer revenue.

C. Performance Analysis of Item Selection

We use simulation to study the performance of adaptation for pages with large number of items of different types, with different ratings. We vary the item count per page, and apportion the items into different types (image, video, advertisements). Developer score for items and user score of item types are chosen within a scale of 1 to 5, with scores distributed normally with mean 3. The item sizes are proportional to the score, but normally distributed to introduce varying sizes for items with the same score.

Fig. 10 and Fig. 11 show the percentage of different scored items selected with respect to user and developer's scores, respectively. In comparison to other techniques, ScoDA picks higher score items more effectively. Note that the total number of items selected by ScoDA can be lower than the other techniques since resource usage of the higher rated items can be higher.

In terms of compute time, on a workstation with $2GHz$ CPU, ScoDA, MKP heuristic and CPLEX generated selection from an item set of size 1000 in 0.35ms, 1.75ms and 1315ms respectively (average of 50 runs). For different item set sizes, ScoDA reached above 97% of optimal composite score computed by CPLEX, and outperformed MKP heuristic.

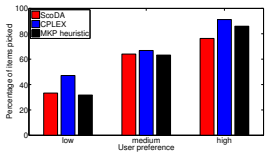


Fig. 10. Percentage of items selected from *user* assigned score category

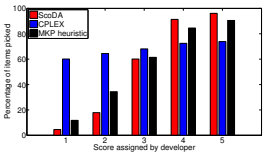


Fig. 11. Percentage of items selected from *developer* assigned score category

VII. Conclusion

User experience on the mobile web is affected by resource constraints like device diversity, network bandwidth and resource limitations. Growing complexity of web pages adversely affects page load time in a mobile browser. Dynamic web content adaptation, guided by user preference in different contexts, helps in improving user experience on mobile web. Involving content creator in the adaptation process can further improve user experience. ScoDA combines user preference to item types, and importance assigned by content creator to items, under resource constraints, to select and generate a web page item load order. Experimental results on large simulated web pages with thousands of items, and real web pages such as news and blog pages, show the efficacy of ScoDA. In an era of device diversity, user preference and context heterogeneity, and growing web page complexity, applying ScoDA is likely to have significant benefits.

As part of future work, we plan to conduct a user satisfaction survey of web pages adapted using ScoDA framework. Further, we plan to explore the behavior of ScoDA in settings where the resource constraints vary at the time of actual download, such as variation in bandwidth leading to variable download time, or unpredictable battery drain due to processes running in parallel on the end device. Exploring the impact of different architectural settings of placing ScoDA, such as on the server, on the client and on an intermediate proxy, is another potentially interesting area to study in future. Another potential avenue requiring further work is a detailed exploration of alternative functions to combine the user and developer ratings, and assess their correlation with type of content and user context.

VIII. Acknowledgement

This research was supported by the MSIP(Ministry of Science, ICT and Future Planning), Korea, under the *IT Consilience Creative Program* (NIPA-2013-H0203-13-1001) supervised by the NIPA (National IT Industry Promotion Agency).

References

[1] V. Adzic, H. Kalva, and B. Furht, "A survey of multimedia content adaptation for mobile devices," *Multimedia Tools Appl.*, vol. 51, no. 1, 2011.

[2] "Akamai q3 2012 report summary," http://www.akamai.com/dl/akamai/akamai_soti_q312_exec_summary.pdf.

[3] M. Butkiewicz, Z. Wu, S. Li, P. Murali, V. Hristidis, H. V. Madhyastha, and V. Sekar, "Enabling the transition to the mobile web with websieve," in *HotMobile*, 2013.

[4] M. Butler, F. Giannetti, R. B. Gimson, and T. Wiley, "Device independence and the web," *IEEE Internet Computing*, vol. 6, no. 5, pp. 81–86, 2002.

[5] "Introduction to composite capabilities / preferences profile (cc/pp)," http://www.webstandards.org/learn/articles/askw3c/feb2004/.

[6] S. Chava, R. Ennaji, J. Chen, and L. Subramanian, "Cost-aware mobile web browsing," *IEEE Pervasive Computing*, vol. 99, 2012.

[7] J. Chon and H. Cha, "Lifemap: A smartphone-based context provider for location-based services," *IEEE Pervasive Computing*, vol. 10, no. 2, Apr. 2011.

[8] "Ibm ilog cplex optimizer," http://www-01.ibm.com/software/integration/optimization/cplex-optimizer/.

[9] M. Dong and L. Zhong, "Self-constructive high-rate system energy modeling for battery-powered mobile systems," in *Mobisys*, 2011.

[10] K. Harumoto, T. Nakano, S. Fukumura, S. Shimojo, and S. Nishio, "Effective web browsing through content delivery adaptation," *ACM Trans. Internet Technol.*, vol. 5, no. 4, Nov. 2005.

[11] "Ibm websphere transcoding publisher," http://www.redbooks.ibm.com/abstracts/sg246233.html?Open.

[12] A. Kaikkonen, "Full or tailored mobile web- where and how do people browse on their mobiles?" in *Mobility*, 2008.

[13] A. Koehl and H. Wang, "m.site: efficient content adaptation for mobile devices," in *Middleware*, 2012.

[14] D. Li and M. Anand, "Majab: improving resource management for web-based applications on mobile devices," in *Mobisys*, 2009.

[15] D. Malandrino, F. Mazzoni, D. Riboni, C. Bettini, M. Colajanni, and V. Scarano, "Mimosa: context-aware adaptation for ubiquitous web access," *Personal and Ubiquitous Computing*, vol. 14, no. 4, pp. 301–320, 2010.

[16] M. F. Md Fudzee and J. Abawajy, "A classification for content adaptation system," in *IIWas*, 2008.

[17] I. Mohomed, J. C. Cai, and E. de Lara, "Urica: Usage-aware interactive content adaptation for mobile devices." in *EuroSys*, 2006.

[18] I. Mohomed, A. Scannell, N. Bila, J. Zhang, and E. de Lara, "Correlation-based content adaptation for mobile web browsing," in *Middleware*, 2007.

[19] I. Papapanagiotou, E. M. Nahum, and V. Pappas, "Smartphones vs. laptops: comparing web browsing behavior and the implications for caching," in *SIGMetrics*, 2012.

[20] R. Parra-Hernandez and N. Dimopoulos, "Heuristic approaches for solving the multidimensional knapsack problem (mkp)," *WSEAS Transactions on Systems*, vol. 1, no. 2, pp. 248–253, 2002.

[21] S. Shrestha, "Mobile web browsing: usability study," in *Mobility*, 2007.

[22] "Amazon silk browser," http://amazonsilk.wordpress.com/2011/09/28/introducing-amazon-silk/.

[23] N. Thiagarajan, G. Aggarwal, A. Nicoara, D. Boneh, and J. P. Singh, "Who killed my battery?: analyzing mobile browser energy consumption," in *WWW*, 2012.

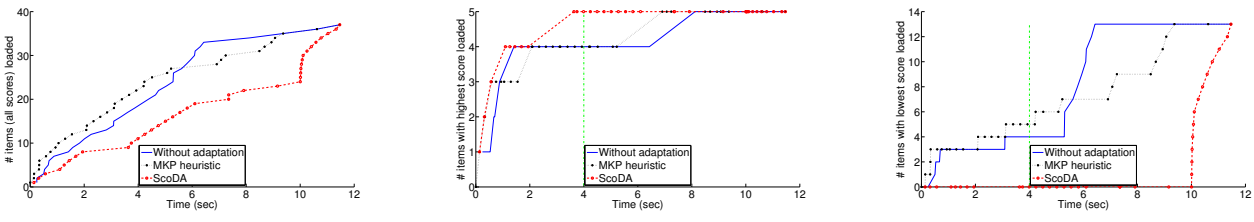
[24] A. Volgenant and J. A. Zoon, "An improved heuristic for multidimensional 0-1 knapsack problems," *J Oper Res Soc*, vol. 41, no. 10, pp. 963–970, 1990.

[25] Z. Wang, F. X. Lin, L. Zhong, and M. Chishtie, "Why are web browsers slow on smartphones?" in *HotMobile*, 2011.

[26] S. Zhang, P. McCullagh, C. Nugent, and H. Zheng, "Activity monitoring using a smart phone's accelerometer with hierarchical classification," in *Proceedings of the 2010 Sixth International Conference on Intelligent Environments*, 2010.

(a) Complete Web Page (b) Without ScoDA, with constraints (c) Using ScoDA

Fig. 5. Screenshots of a news site showing the result of adaptation when using the HTML loading order, and ScoDA based adaptation.



(a) For all the items (b) Only items of score 5 (c) Only items of score 1

Fig. 9. Comparison of loading of items of different scores over time for the page from news site.