2009 Ninth International Conference on Intelligent Systems Design and Applications

A Preliminary Experience in Optimizing the Layout of Web Pages by Genetic Algorithms to Fit Mobile Devices

Luigi Troiano*, Gennaro Cirillo†, Roberto Armenise† and Cosimo Birtolo†

* *University of Sannio*
Department of Engineering
82100 Benevento, Italy
troiano@unisannio.it

† *Poste Italiane S.p.A. – TI - SSI*
Centro Ricerca e Sviluppo
80133 Napoli, Italy
{*ciril127,armenis5,birtoloc*}*@posteitaliane.it*

***Abstract*—Getting access to web content by mobile devices is becoming widespread. This poses the need of adapting content that have been designed for desktop application to being delivered on smaller displays. In this paper we investigate the application of a genetic algorithm as means for an automatic adaptation of existent pages to mobile device requirements, reporting preliminary results and outlining problems to be faced and solved in order to make this approach robust.**

I. Introduction

The diffusion of PDAs, cell phones, and many other mobile devices, along an increasing coverage of wireless networks, made browsing web pages on mobility a common means for accessing the Internet. However, small displays limit usability and accessibility to information. Users ought to scroll the screen in both vertical and horizontal directions to find the desired content, making information fruition and navigation a difficult and sometimes frustrating task. This is because pages are generally designed for being accessed by desktop devices with large screens. As the number of users is increasing, editors are more and more motivated to deliver content adapted to needs and constraints of mobile devices. So, it is becoming common to provide a reduced version of content suitable for being accessed by hand-held devices, generally referring to screen resolution of 240×320px. However the screen size, resolution and orientation is changing over the time, and many configurations (elder and newer) are available nowadays. There are different strategies to deal with mobile devices: (i) to deliver pages adapted to mobile requirements, (ii) to automatically fit pages in a standard structure as the single column layout, (iii) to keep the page unchanged. Each of these strategies has some limitations and drawback. For instance, keeping the page unchanged can make the page unsuitable to get accessed by smaller displays, whilst delivering specific content means to deal with many displays variants available today that could be better used individually and adapting to a specific layout means to renounce to part of information given by the page structure. Another option would consist in optimizing a page to fit specific displays and user requirements. Optimizing a page is challenging task as many constraints must be taken into consideration at a time. This task is delegated to human intelligence. Can an algorithm adapt web content as well and suggest new ways to organize a page? In this paper we attempt to provide a preliminary answer that question. Search based algorithms, meta-heuristics in particular, are a promising solution for supporting the task.

This paper investigates the application of a genetic algorithm to optimize web pages to get adapted to small displays, starting from those pages designed for desktop applications. The algorithm is aimed at manipulating off-line the structure of the page by evolving a set of parameters describing the DOM structure (e.g. horizontal and vertical scrolling, occupation ratio, maximum number of pages, etc.) in order to find the best content disposition and size and to fit the whole page to small screen of hand-held devices. The remainder of this paper is organized as follows: Section 2 provides a brief overview of layout design in web application and in mobile domain; Section 3 describes how the problem has been modeled and solved by a genetic algorithm; Section 4 presents preliminary experimental results; in Section 5 conclusions are drawn and future directions outlined.

II. Related works

Over the years we assisted to an increasing availability of screen size, from the the smallest 128×128 pixels to the largest 800×480 pixels, meaning that the largest screen is 23 times bigger than the smallest one. Even focusing on most popular screen sizes, availability increased from 2005 to today, as depicted in Fig.1, although 240×320 (QVGA) is the new baseline screen size.

978-0-7695-3872-3/09 $26.00 © 2009 IEEE
DOI 10.1109/ISDA.2009.53

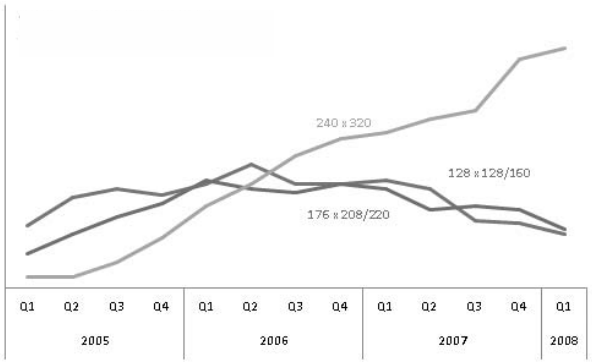

Figure 1. Popular screen size.

It has been estimated that 128×128, 128×160, 176×220 and 240×320 displays cover 78% of all devices in EMEA, US and CA, and 96% of all phones have a screen size ratio between 3:4 and 4:3. However still displays of mobile devices greatly differ in their characteristics.

With such a plethora of available displays differing in their dimensional characteristics, the task of adapting existing web content becomes challenging. Many different techniques have been developed over the time. Stormer [1] surveys different ways to adapt existing web pages to mobile devices, classifying them in three main categories: (i) Rewrite the page in a suitable way for mobile devices; (ii) Adapt the pages automatically choosing a different CSS; (iii) Use XML to transform the pages.

Ahmadi and Kong [2] propose a novel method to automatically adapt a desktop presentation to mobile, in order to overcome the frustrating task of users in scrolling the screen both vertically and horizontally. Their approach integrates structural analysis of the HTML source and visual layout detection to identify closely related content and then to generate an adapted layout.

An alternative approach is based on segmenting web pages, choosing informative sections and re-laying these sections into a compact form, as proposed by Sengamedu, Mehta, and Madaan [3]. In their work, they present a system that, leveraging structural, content and visual information, learns the page template, assigns relevance to each section, and scales them according to their score.

Lethonen et al. [4] introduce a proxy based platform able to render web pages and to extract only the necessary information to be sent to the mobile client. Novelty of this approach resides in resembling a view similar to that available by a desktop browser but smaller. Baluja [5] proposes to present the user a thumbnail image of the full web page, allowing the user to zoom into regions.

All solutions above and others (e.g. Olivera [6]) attempt to identify heuristics for solving the problem. More recently, another direction being investigated regards the page layout as an optimization problem aimed at maximizing usability criteria and user preferences, meeting as much as possible dimensional constraints.

Ahmad, Basir and Hassanein [7] pursue a different approach. They suggest fuzzy logic rules as means for trading-off between different, often conflictive, requirements and user preferences, still meeting the standard usability guidelines at the same time.

In another work, Ahmad et al. [8] regard the page layout as a bin packing problem. They address the problem by optimization strategies based on heuristic rules for placing content modules (shaped as rectangles) in sequence. The strategy outcome depends on the order by which modules are placed. In particular the strategy fitness is function of module position and dimension. A genetic algorithm is introduced in order to find the module sequence that maximize the strategy outcome.

Gajos, Weld and Wobbrock [9] solve the problem of finding an appropriate trade-off among device constraints and user preferences by adopting a decision-theoretic optimization.

Russo, Birtolo and Troiano [10] suggest a pure generative approach in layout design is able to face the problem. In our previous work [11], we successfully tested genetic algorithms in re-arranging the disposition of Web form fields on different pages according to some constraints. The paper is focused on a pre-defined page layout (i.e. vertical flow in mobile devices) which allows only to determine the sequence of fields per page. By this way horizontal scrolling is automatically avoided but we can not re-organize the whole page. In the proposed approach, instead, we consider a Web page containing images, text areas, buttons and we arrange elements in new and unforeseen layouts.

In [12] we investigated the application of genetic algorithms as a viable approach to optimize a menu layout in order to maximize accessibility and compliance to guidelines and user preferences.

III. Algorithm

The algorithm is inspired to the GA given by Goldberg [13] and its structure is outlined in Fig.2. The algorithm breeding is based on the following operations:

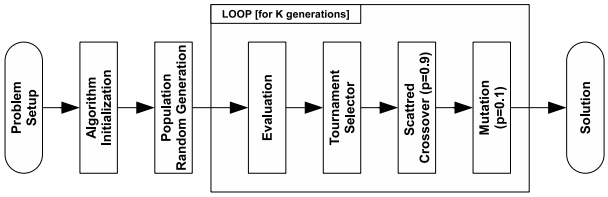

Figure 2. Algorithm structure.

- selection: Tournament Selector has been preferred in order to be less sensitive to the fitness scaling

- crossover: Scattered Crossover has been defined in order to maximize the population diversity during the evolution
- mutation: Simple Mutation, in order to change one gene randomly

In particular, the algorithm adopts elitism with a random selection of individual to be substituted by the best ones. Tournament is implemented by selecting the best individual after t pairwise comparisons, as described in [13].

Scattered Crossover is described in Fig.3. First the number of genes to exchange is chosen, then which genes to use is decided, the two parent chromosomes are mixed and offsprings generated.

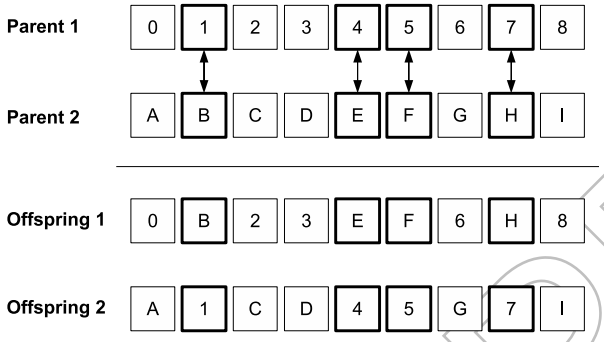

Figure 3. Scattered crossover.

We preferred scattered crossover to more usual single-point and two-point crossover, as it performs better in mixing parameters as required by the layout optimization, thus enabling the genetic algorithm to converge faster and produce better solution.

A. Chromosome structure and mapping

We assume that web page is structured in a hierarchy of elements. Depending on what kind a page element is, different optimization parameters are made available. More specifically, elements we are interested to optimize are:

1) *Container*: holds simpler elements or other containers. This element gives the possibility to decide the disposition of the inner elements, vertically piled or horizontally aligned. The container size depends on the size of its content. *Parameters: orientation and page break (only for the top level container).*
2) *Text*: is a paragraph, an hypertext or a text-area. A text is characterized by width and font size; height is determined according to these two properties. It is also possible to choose a detailed or a summarized version of the content. *Parameters: width resize ratio, font resize ratio, summarized (only for paragraphs and hypertexts).*
3) *Graphics*: an image, an icon or an image button. This element is only characterized by a resize ratio able to keep the aspect ratio. *Parameters: resize ratio, if zero the graphics is removed from the page.*
4) *Textual Button*: a simple labeled button. The algorithm can arbitrarily change width or height of the button, ensuring a correct visualization of the label inside. *Parameters: width resize ratio and height resize ratio.*

The whole page is modeled as a container. An additional parameter (i.e. page break) allows to split the page in several pages. This makes possible to dispose the content in more pages if necessary.

In order to reduce the search space, each parameter can assume a discrete number of possible values. Values are indexed. For instance, a resize ratio can assume values such in $\{20\%, 50\%, 100\%, 120\%\}$. Each of these values is indexed, so ratio = 0 means 20%. Boolean parameters are coded as 0 for false and 1 for true.

In order to keep the page aspect consistent, elements can be grouped in classes (i.e. categories). Each category refers to specific parameters, so that optimization is able to control the rendering of several elements at a time.

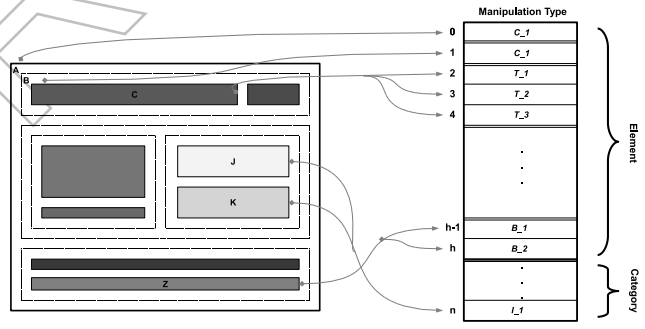

Figure 4. Mapping of page layout to chromosome

The page structure is accessible by its Document Object Model (DOM). At beginning, the algorithm inspects the page DOM and identifies the elements to be optimized. According to resulting list and to elements categories, a chromosome structure is properly instanced. Individuals represent a particular set of parameter values for rendering the page in a modified layout.

As means of making the solution independent on a specific device, element dimensions are expressed as relative ratios. So, starting from the original screen size (in pixel), the algorithm determines the absolute dimensions of elements, also adding a border around them for a better page rendering, and ultimately the actual page size.

The mapping between page structure and chromosome is depicted in Fig.4

B. Fitness Function

The fitness function for an individual x is aimed at assessing a resulting layout according to the following requirements:

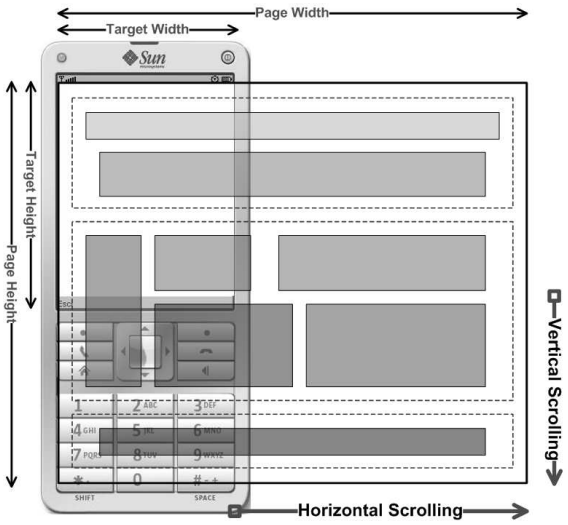

Figure 5. Scrolling

- scrolling should be limited in both vertical and horizontal direction ("*scrolling constraint*", see Fig.5);
- a page should occupy as much as possible the space made available on the display ("*area occupation constraint*", see Fig.6);
- elements should be arranged in order to minimize the empty space among them and should be minimized as this result in wasting available space ("*area waste constraint*", see Fig.6);
- the number of pages, in which the original document has to be split, should not be over a certain limit ("*number of page constraint*");
- a minimum width for text rendering should be met as much as possible ("*text-width constraint*").

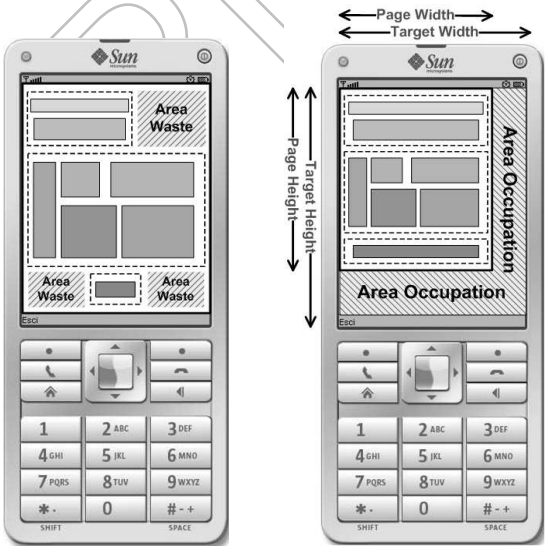

Figure 6. Area waste and occupation

Fitness is a function (to maximize) assuming values in the unit interval $[0, 1]$ and defined as

$$fitness(x) = C(x)^{\frac{1}{m}} \cdot T(x) \tag{1}$$

where $C(x) \in [0, 1]$, defined by Eq.2, is a convex combination of page requirements, m is a score related to the number of pages (see Eq.3), while $T(x)$ is a score given by meeting text-width constraints. As $C(x), m \in [0, 1]$, power $1/m$ makes $C(x)$ smaller by deacreasing m, so that minimum is 0 when the number of pages exceeds the maximum of allowed pages $N_{\max}$.[1] When the number of pages is below $N_{\min}$, therefore fully acceptable, $m = 1$ and $C(x)$ is left unchanged. The product of $C(x)$ and $T(x)$ requires both contributions to be maximized.

$$C(x) = \frac{w_1 \cdot A_1(x) + w_2 \cdot A_2(x) + w_3 \cdot A_3(x)}{w_1 + w_2 + w_3} \tag{2}$$

where $A_1(x)$ is an index of how the scrolling constraint is met, whilst $A_2(x)$ is referred to the occupation constraint (area maximization of device display) and $A_3(x)$ to the area waste constraint. These three factor are opportunely traded off by weights w_i with $i = 1..3$

The page limit is defined fuzzy, in order to avoid a sharp border between what is acceptable and what is not. So the maximum acceptable number of pages is $N_{\max}$, whilst the ideal number of pages should be not over $N_{\min}$. Therefore

$$m = \begin{cases} 1 & n \leq N_{\min} \\ 1 - \frac{n - N_{\min}}{N_{\max} - N_{\min}} & N_{\min} < n < N_{\max} \\ 0 & n \geq N_{\max} \end{cases} \tag{3}$$

where $m = 1$ means ideal, and $m = 0$ unacceptable.

Finally, $T(x)$ takes into the account quality of text rendering, as

$$T(x) = 1 - \frac{v_i(x)}{v_i^*(x)} \tag{4}$$

with $v_i(x)$ giving the number of text violations of x, and $v_i^*(x)$ the maximum number of possible violations.

IV. Experimental Results

Experimentation is aimed at verifying if the fitness function is a good predictor of page rendering quality, how the algorithm is able to fit more complex pages, and how evolution is affected by the parameters of genetic algorithm and fitness function.

We run the algorithm several times with different parameter settings in order to study quantitatively the algorithm convergence.

In our experimentation we considered three different web pages with different complexity (see Fig.7):

1) A simple Web Form Page (Login Page)

[1]The value $C(x)^{\frac{1}{m}} = 0$ is obtained as limit by $\frac{1}{m} \to +\infty$.

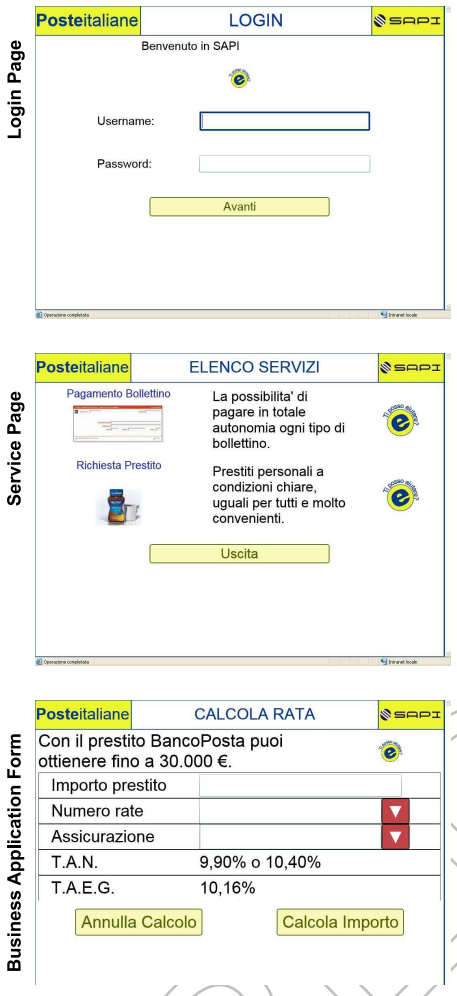

Figure 7. Initial Web Pages

2) Page for choosing a customer service (Service Page)
3) A complex Web Form Page (Business Application Form)

Initial page layouts are used to populate the algorithm at the beginning along a set of random solutions. The ratio between initial clones and random solutions is named the *randomization rate*.

We repeated 5 runs for different problem configurations.[2] First of all, we investigated about the fitness trend at varying the population size (200, 500, 1000 individuals) as depicted in Fig.8 and Fig.9. By plotting the arithmetic mean of best fitness, we can verify the convergence for both the Login Page and the Service Page.

By increasing the page complexity we notice some issues regarding the convergence due to longer chromosomes, from 20 genes for Login Page to 34 genes for Business Application Form (Fig.10).

[2]Each run required between 16.5 minutes and 3.88 hours on Pentium IV 2GB RAM.

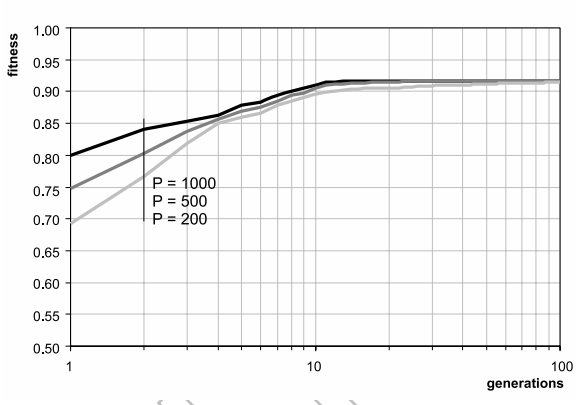

Figure 8. Average fitness behavior for Login Page by varying the population size

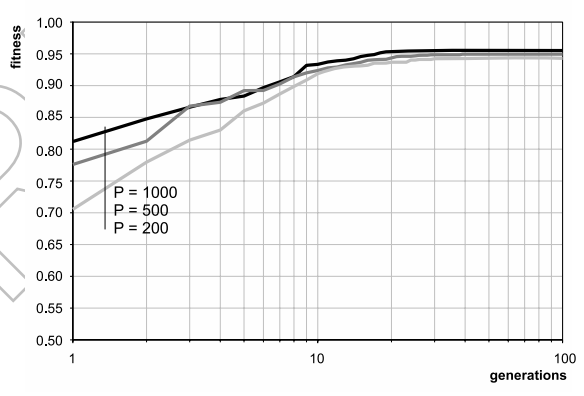

Figure 9. Average fitness behavior for Service Page by varying the population size

Algorithm convergence is confirmed when we modify the randomization rate of the initial population (see Fig.11) although searching an optimal solution for the Service Page (Fig.12) seems to better perform when rate is 1.0 (meaning a completely random initial population).

In Fig.13 and in Fig.14, we reported the average fitness behavior for different configurations of fitness parameters w_i (Eq.2). We select three different fitness parameter settings:

$$\begin{aligned} W1 &: (w_1, w_2, w_3) = (0.5, 0.25, 0.25) \\ W2 &: (w_1, w_2, w_3) = (0.25, 0.5, 0.25) \\ W3 &: (w_1, w_2, w_3) = (0.25, 0.25, 0.5) \end{aligned} \quad (5)$$

We can conclude that analysis is independent on these factors and the average fitness behavior is related to the complexity of the target page. We can notice that the algorithm reached high values of fitness in common Web interface, although the behavior differs qualitatively according to the problem settings.

As shown in Fig.15 we observe an optimal solution found for Service Page. In the figure we compare an individual

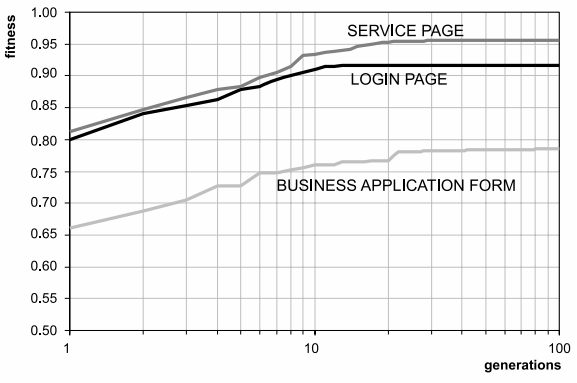

Figure 10. Average fitness behavior.

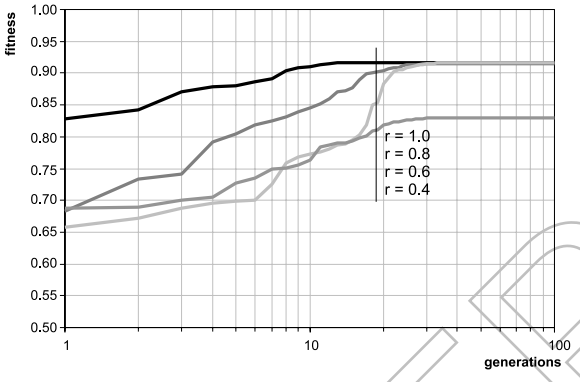

Figure 11. Average fitness behavior for Login Page by varying randomization factor.

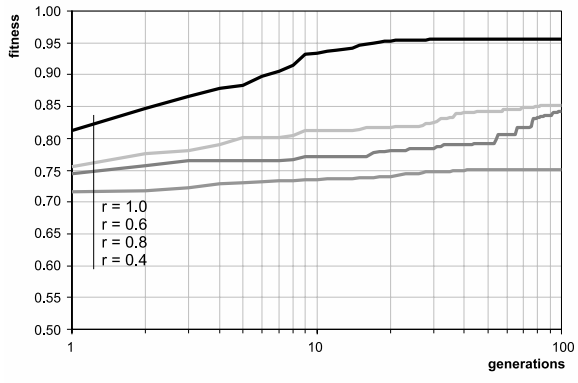

Figure 12. Average fitness behavior for Service Page by varying randomization factor.

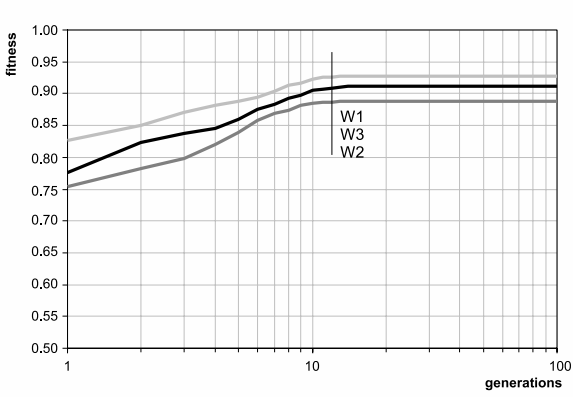

Figure 13. Average fitness behavior for Login Page by varying fitness parameters.

which is not optimal (at left) with the best solution (at right) found for the same page after 100 generations. We observe that the worst individual with fitness value equal to 0.74693 has some issues such as empty space in the page and scrolling in horizontal direction. The best individual ($fitness = 0.9626$) is composed by 2 pages, area occupation is optimized and there is no scrolling.

Fig.16 and Fig.17 show the results of the optimization process for Login Page and Business Application Form in relation to the fitness value obtained.

We can notice how the fitness function can effectively measure the result quality in terms of aesthetics and usability. Indeed, high values of fitness match with better pages having content enclosed within the display size. On opposite, low levels of fitness entail pages badly fitted to the displays, thus requiring to scroll the content.

V. Conclusions and future work

In this paper we investigated the use of a pure generative approach in adapting the layout of web pages to reduced screen size on mobile devices. In particular we tested a simple genetic algorithm as means for searching a set of transformations able to fit a page to a mobile device assuming a set of user constraints. The preliminary results are twofold. On one side, they prove that this approach is promising and able to ensure the optimization of the pages in accordance with aesthetics and usability requirements. On the other side, the solution is not able to scale an increasing page complexity. In our opinion this is due to the conflictive constraints able to trap the algorithm in local optima. A solution to overcome the current limitations is to treat the problem as multi-objective optimization, does avoiding the fusion of objectives in one single fitness value. In brief, although generative approach is feasible, there is still work to do before page layouts will be automatically discovered and adapted.

Acknowledgment

This work was partially supported by MIUR Project Automatic System For The Visually Impaired (SAPI), n.1642-2006.

References

[1] H. Stormer, "Exploring solutions for a mobileweb," in *Proc. of CEC/EEE*. IEEE Computer Society, 2006, p. 75.

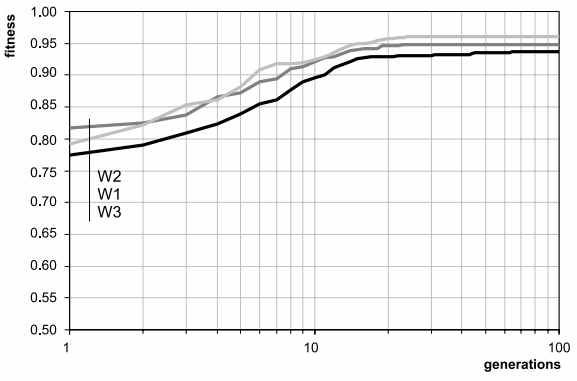

Figure 14. Average fitness behavior for Service Page by varying fitness parameters.

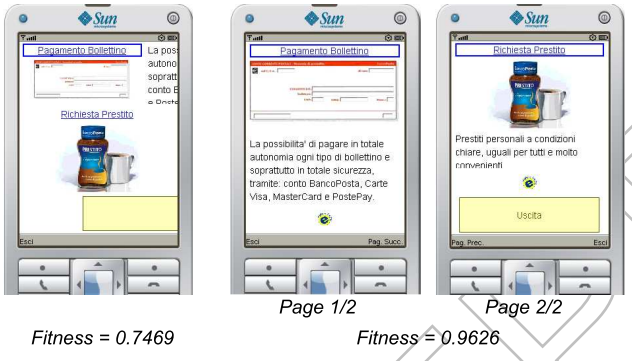

Figure 15. Optimizing Service Page.

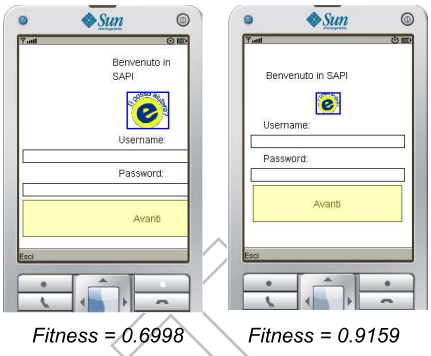

Figure 16. Optimizing Login Page.

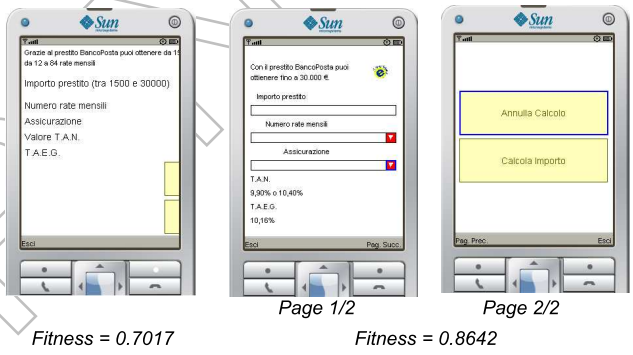

Figure 17. Optimizing Business Application Form.

[2] H. Ahmadi and J. Kong, "Efficient web browsing on small screens," in *AVI '08: Proc. of the working conference on Advanced visual interfaces*. New York, NY, USA: ACM, 2008, pp. 23–30.

[3] S. H. Sengamedu, R. R. Mehta, and A. Madaan, "Web page layout optimization using section importance," in *Proc. of the 17th International World Wide Web Conference*. WWW2008, 2008.

[4] T. Lehtonen, S. Benamar, V. Laamanen, I. Luoma, O. Ruotsalainen, J. Salonen, and T. MikkonenP, "Towards user-friendly mobile browsing," in *AAA-IDEA '06: Proc. of the 2nd Int. Workshop on Advanced Architectures and Algorithms for Internet Delivery and Applications*. New York, NY, USA: ACM, 2006, p. 6.

[5] S. Baluja, "Browsing on small screens: Recasting web-page segmentation into an efficient machine learning framework," in *WWW '06: Proc. of the 15th Int. Conf. on World Wide Web*. New York, NY, USA: ACM, 2006, pp. 33–42.

[6] J. a. B. S. de Oliveira, "Two algorithms for automatic document page layout," in *DocEng '08: Proc. of the 8th ACM symposium on Document engineering*. New York, NY, USA: ACM, 2008, pp. 141–149.

[7] A. R. Ahmad, O. A. Basir, and K. Hassanein, "Fuzzy inferencing in the web page layout design," in *WSMAI*, 2003, pp. 33–41.

[8] A. Ahmad, O. Basir, K. Hassanein, and M. H. Imam, "Improved placement algorithm for layout optimization," in *The 2nd International Industrial Engineering Conference (IIEC2004)*. IIEC press, 2004.

[9] K. Z. Gajos, D. S. Weld, and J. O. Wobbrock, "Decision-theoretic user interface generation," in *AAAI*, D. Fox and C. P. Gomes, Eds. AAAI Press, 2008, pp. 1532–1536.

[10] G. Russo, C. Birtolo, and L. Troiano, "Generative ui design in sapi project," in *CHI '08: extended abstracts on Human Factors in Computing Systems*. New York, NY, USA: ACM, 2008, pp. 3627–3632.

[11] L. Troiano, C. Birtolo, R. Armenise, and G. Cirillo, "Web form page in mobile device: Optimization of layout with a simple genetic algorithm," in *Proc. of 11th Int. Conf. on Enterprise Information Systems*. ICEIS, 2009.

[12] ——, "Optimization of menu layouts by means of genetic algorithms," in *EvoCOP*, ser. Lecture Notes in Computer Science, J. I. van Hemert and C. Cotta, Eds., vol. 4972. Springer, 2008, pp. 242–253.

[13] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison Wesley, 1989.