Eric Sanlaville
Professor of computer sciences
Normandy University, Le Havre, France
eric.sanlaville@univ-lehavre.fr

September the 16th, 2017

Reviewer's opinion on Ph.D. dissertation authored by

Jakub Marszałkowski

entitled:

Combinatorial optimization problems in internet applications

Problem and its impact

The PHD dissertation considers three combinatorial optimization problems from the web management. The three parts present original scientific results that can be applied in web engineering. As such, the presented work builds bridges between practical issues from web engineering and the classical approaches of combinatorial optimization, through large efforts on modeling, designing and testing ad-hoc methods. Hence it opens new perspectives for practical applications of OC. The third part, that considers the issue of building sprites for web page downloading, is for me the one that brings the most significant and complete results.

Contribution

The thesis is organized in one introduction, three main parts, each part dealing with a specific problem, and one general conclusion. The three parts are independent, even if some methods may be used in both second and third parts.

The introduction section mainly points out the new and growing importance of optimization problems in web engineering. Indeed, few works propose a formal model for these problems. It also provides the common features of the three considered problems. Mainly, the theoretical background is the theory of two-dimensional packing. Furthermore, as the three problems deal with web page design, a key issue is the temporal dimension: the time a user waits for the page to download is a major issue. Hence a small algorithm complexity is mandatory. Even if the presentation of complexity theory and of general solving approaches is a bit clumsy and the structure of the chapter sometimes confusing , the chapter presents clearly the goals of the thesis.

Section 2 considers the problem of building a layout of a web page to optimize the placement of ads. It is easy to understand the economic importance of this question, as most web site profitability depends on ads. The page is limited in width but not in depth. the ads have all a rectangle form. The goal is to divide vertically the page in columns so that these rectangles will best fit the column widths. However, the precise optimization model is not that easy to grasp. There are two main objectives, first to maximize the flexibility, second to minimize the wasted space. The first objective allows to place a maximum number of different size rectangles. When the page layout is designed, the ads that will be placed are not known, only the possible sizes are known. Jakub Marszałkowski proposes two antagonistic criteria to maximize that number but also to maximize the possibilities to fit one given ad size in the columns. A third criterion is added to limit the wasted space as wasted space amount is an indicator of layout quality.

Hence the problem is modeled by a three-criterion linear optimization model. A large number of constraints are added. The author proposes a solving in several steps, based on partial enumeration first of ad size configurations (in one column), then of column widths. The method can keep all solutions of the Pareto front, but is exponential.

The generation of the constraints and of the performance criteria sometimes lack of justifications, but the provided example, and the benchmark presentation, help. Note that no real state of the art on strip packing is provided, the text heavily relies on one reference to build the model and the algorithm (Wang 1983). It seems to me that in order to increase the speed of the algorithm, approximate algorithms limiting the number of configurations (removing configurations close to the ones that are kept) could be used, instead of limiting the characteristics of the configuration as it is done here. The computational experiments are done on benchmarks issued from real ad providers, and the results are positive, as the computation times remain acceptable and the number of proposed alternative solutions suitable. The experiments provide useful informations for the different data sets. Still, the comparison with alternative methods would have been interesting. I also regret that the observed frequency of the different ad sizes was not taken into account here. In the conclusion of the chapter, some perspectives are given, both for applications on different domains (e.g., port logistics) and for extensions to other web page layout optimization problems. More explanations would have been appreciated. This chapter is very rich as it proposes a complete study (background, model, algorithm, tests) of a complex problem, however, the reader might be a bit frustrated by some lacks regarding state of the art and alternative methods.

The third chapter deals with the construction of tag clouds: from a given set of tags, i.e. words or groups of words, how to organize them as a cloud so that the resulting cloud is both aesthetically satisfying and highlights the main features. The main difficulty here is to take into account the aesthetic dimension of the solution evaluation. The chapter proposes a presentation of the problem and of the existing methods, a formulation as an optimization problem, some methods to solve it and some experiments. An important point in this study is the fact that tag characteristics (police, size, orientation) are fixed. Hence the problem may be seen as a packing of rectangles problem, but the optimization criterion is to be defined. If the order of the tags is fixed, the problem is highly related to "how to break lines" in a text, thoroughly studied by Knuth et al for Latex. If not, how to compute that order is a mandatory first step. The performance criterion, as for texts, is usually related to the empty spaces (at the end of lines, between lines), but also to the tonal distribution (from typography : balancing color or levels of gray in the space). In the literature, simple greedy heuristics, genetic algorithm, but also geometric, Dynamic Programming and Divide and Conquer methods are used. The chapter analyses the specific uses of tag clouds on the web. It concludes that the problem indeed consists in packing rectangles into a large rectangle of fixed width but variable height (as the columns of previous chapter). Moreover, pre-ordering the tags (e.g. by alphabetical order) results in badly looking clouds and packing based on the shape of the tags should really be used . For the practical use of the methods to design, it should be executed on the client side (as the result highly depends on the client browser) and be fast enough on any client machine.

Jakub Marszałkowski proposes a mathematical formulation, with a unique objective based on the tonal distribution : the tonality should be distributed most evenly among the "lines" (called shelves in the text) of the cloud. In fact this is a classification model, a class corresponding to a shelf. The complete formulation as an Integer program is not provided. The objective function is parameterized by a factor k that in fact corresponds to the choice of one norm L_i. The final choice is k=1/2, so that the function is not linear, but this choice is not really justified and a linear function so that the average value of the tonal weight is minimized would have been also appropriate.
A Branch and Bound method is implemented, even if its complexity should not allow to use it in practice. However, it is basically an enumeration method, without any real bounding scheme (branching is done on the allocation of one tag). In order to use an exact method for comparing with heuristic found solutions of instances of larger sizes, a more sophisticated method is necessary (B&B or Dynamic programming). The proposed practical methods are greedy algorithms with different rules and a TABU search. The results show that the greedy solutions are very close to the optimal solution when it is known. Hence logically the TABU search provides little improvement. In fact, the improvement comes from a decreasing in the number of shelves, but the greedy algorithms may easily be used on several target values of this number.

The presentation of the problem, the previous works from the literature, and the specifics of tag cloud on the web, are clear and interesting. The modeling phase is incomplete. The tested methods are rather elementary, and a larger effort on methodologies would have been appreciated. However, they are tested on real instances of reasonable size and the results seem good, even surprisingly good. An analysis of this would have been useful, for instance the choice of the objective function form might have an influence. Finally, the perspectives of this work are mainly, I think, in the extension to the choice of tag characteristics according to designer preferences (importance of the tags, semantic links between them).

Chapter four is the longest and the most complete. It deals with another problem with real significance, as it tries to minimize the downloading time of a web page by reorganizing the page elements into blocks (sprites) that may be routed in parallel. The final objective is to propose a tool for doing this in practice. Hence the emphasis is put on simple but efficient and fast methods, and on tests in real conditions. To present the problem in all its dimensions, Jakub Marszałkowski had to draw knowledge from many different fields, mainly 2D packing again, image compression, communication on the web models. A solution is a partitioning of the units of the initial web page (tiles) into a set of sprites (the number of sprites may vary). Ultimately, the quality of a solution is expressed by the downloading time, hence it is heavily related to the size (in bytes) of each sprite. Minimizing the size of each sprite implies to minimize the empty space, but not only: the geometric arrangement of the tiles in the sprite may allow, or not, optimizations during the compression phase. Note also that using as many sprites as the number of tiles minimizes the wasted space, but there are limits to the number of sprites that may be sent in parallel.

Jakub Marszałkowski first presents the different issues, about the packing, the compression. He shows that the problem is NP-Hard even for simple subcases. A model for the communication performance is then given. The author is right when he states that there are too many uncertainties (from the server side, the client side, the network itself) to come up with a really accurate model. Hence a simple model taking into account informations about latency, throughput, and maximum allowed concurrency (number of channels opened by the browser to connect to the web server), is designed to be used during the solution building phase. Some preliminary tests are done for a better understanding of the correlation between packing types and compression, and to set experimentally the communication parameter models. Note however that the high variability of bandwidth for instance questions the choice of fixed values, and this choice is not well justified in the text. The resulting values are rather small, the increase of bandwidth remaining smaller that a factor 2 even for 8 channels. A random model would certainly be more accurate.

The chapter then presents existing tools. It is a bit surprising that these tools are so few, and so limited in their possibilities. By contrast, the proposed tool Spritepack appears to be more sophisticated and versatile. It constructs a solution through a sequence of treatments : tile classification according to colors used and transparency (for grouping so as to maximize compression performances), packing inside of each sprite, a second merging step between existing sprites, final compression stage using post processing optimization. The rectangle packing is done using a greedy method, using a set of packing rules seen in previous parts. The merging also uses a greedy method, merging two sprites into one and keeping the best compression result as an evaluation for the final sprite. The output is the set of sprites with globally the best compression performance.

Spritepack execution time is close to a few minutes, but most of the time is used at the merging stage (needing compression). The performance increasing with regard to the sending of tiles one by one is significant. It is then compared with other available tools on real instances (Spritepack is then parameterized to produce one final sprite), first using the optimization function from communication model, second recording the real communication time. In both cases Spritepack outperforms other existing tools.

Hence this chapter provides a strong contribution to sprite building, and a complete tool that has been experimentally tested on real instances. Even as the global problem is NP-hard (the presented proofs are somewhat imprecise but the result is OK), the reader may regret that for each stage, very simple heuristics are used; this is due to computing time constraints. Still, even if the complete tool has been experimentally

tested, one may regret, from an optimization point of view, the lack of tests at each stage against other possible methods. A study on the respective contributions of each stage of the tool to the overall performance, and hence on the interest of increasing the optimization effort, is a perspective of this work.

The conclusion chapter mainly summarizes the contributions and gives some perspectives that are not developed.

Global advice on the contributions

The theme of this work is very large, as it considers three different problems. The reader may regret that some aspects are not treated more deeply. Indeed, the choice of the optimization methods for each part are not sufficiently justified. The state of the art on the proposed methods is sometimes a bit too fast, especially as these methods come essentially from well identified optimization domains,especially two dimensional packing. More space might have been allocated to clear presentation of the major concepts and definitions used. The methods are not the most efficient, which is justified by the need for rapid solving. Second, one may regret the lack of validation by comparing the results obtained by the chose nmethod and those of (theoretically) more efficient ones. There are still a large number of typos in the text and some lacks of definitions. Still, the text remains quite readable and clear in its objectives and results.

This works builds a bridge between the web engineering and combinatorial optimization, which is a major contribution by itself. The author shows very well that most existing methods from web engineering do not consider at all the optimization issues, and that it may produce far from efficient solutions. This is specially obvious for the third problem, where the sequence of methods proposed clearly outperforms existing proposals. Note however that building an optimization function may not be direct and unquestionable, especially when aesthetic considerations must be added. Moreover, the author displays an impressive knowledge of these practical issues about web design, but also very good skills on the implementation and the tests on practical instances. The third part of the thesis manuscript constitutes the major achievement, as Jakub Marszałkowski is able to present a complete study on the sprite packing, from a clear presentation of the practical problem to the design and implementation of a software ready for use, without forgetting method analyses and complexity issues.

Conclusion

Jakub Marszałkowski presents an impressive original contribution on the combinatorial optimization issues of web engineering, that can be considered as a pioneering work. It has been published in several conferences and two good journals from both domains (EJOR and ACM transactions on the web). Jakub Marszałkowski deserves without any doubt the title of doctor in computer sciences.

Taking into account what I have presented above and the requirements imposed by Article 13 of the Act of 14 March 2003 of the polish parliament on the academic degrees and the Academic Title (with amendments), my evaluation of the dissertation according to the three basic criteria is the following:

A. The dissertation presents an original solution to a scientific problem: **Definitely Yes**.

B. The candidate has eneral theoretical knowledge and understanding of the discipline of **Computing**, and particularly the area of **Combinatorial Optimization** : **Rather Yes**.

C. The dissertation supports the claim that the candidate is able to conduct scientific work :**Definitely Yes**.