Computers & Industrial Engineering 115 (2018) 331–341

Contents lists available at ScienceDirect

Computers & Industrial Engineering

journal homepage: www.elsevier.com/locate/caie

Algorithms for the bin packing problem with overlapping items

Aristide Grange[1], Imed Kacem*, Sébastien Martin

LCOMS EA7306, Université de Lorraine, Metz, France

ARTICLE INFO

Keywords:
Pagination
Bin packing
Virtual-machine packing
Integer linear programming
Heuristics
Genetic algorithms

ABSTRACT

We study an extension of the bin packing problem, where packing together two or more items may make them occupy less volume than the sum of their individual sizes. To achieve this property, an item is defined as a finite set of symbols from a given alphabet. Unlike the items of Bin Packing, two such sets can share zero, one or more symbols. The problem was first introduced by Sindelar et al. (2011) under the name of VM Packing with the addition of hierarchical sharing constraints making it suitable for virtual machine colocation. Without these constraints, we prefer the more general name of Pagination. After formulating it as an integer linear program, we try to approximate its solutions with several families of algorithms: from straightforward adaptations of classical Bin Packing heuristics, to dedicated algorithms (greedy and non-greedy), to standard and grouping genetic algorithms. All of them are studied first theoretically, then experimentally on an extensive random test set. Based upon these data, we propose a predictive measure of the statistical difficulty of a given instance, and finally recommend which algorithm should be used in which case, depending on either time constraints or quality requirements.

1. Introduction

Over the last decade, the *book* (as a physical object) has given ground to the multiplication of screens of all sizes. However, the *page* arguably remains the fundamental visual unit for presenting data, with degrees of dynamism varying from video to static images, from infinite scrolling (e.g., Windows Mobile interface) to semi-permanent display without energy consumption (e.g., electronic paper). Pagination is to information what Bin Packing is to matter. Both ask how to distribute a given set of items into the fewest number of fixed-size containers. But where Bin Packing generally handles concrete, distinct, one-piece objects, Pagination processes abstract groups of data: as soon as some data are shared by two groups packed in the same container, there is no need to repeat it twice.

1.1. Practical applications

As an introductory example, consider the following problem. A publisher offers a collection of audio CDs for language learning. Say that a typical CD consists of 100 short texts read by a native speaker; for each of them, a bilingual vocabulary of about 20 terms has to be printed out on the CD booklet. How best to do this? The most expansive option, both financially and environmentally, would require the impression of a 100-page booklet, i.e., with one page per audio text. But now suppose that each page can accommodate up to fifty terms. If all individual vocabularies are collated into one single glossary, no more than $100 \times 20/50 = 40$ pages are needed. This is the cheapest option, but the least convenient, since it forces the consumer to constantly leaf through the booklet while listening to a given text. To minimize cost without sacrificing usability, the publisher will be better off to pack into each page as many individual vocabularies as possible. If there were no common term between any two vocabularies, this problem would be Bin Packing; but obviously, most of the time, the vocabulary of a given text partially overlaps with several others: this is what we propose to call **the pagination problem**, in short Pagination. In this example, it takes advantage of the fact that the more terms are shared by two vocabularies, the cheaper to pack them together; as an added benefit, a good pagination will tend to group on the same page the audio texts dealing with the same topic.

Coincidentally, it was in this context of linguistics that we first stumbled across Pagination. At that time, we needed to display selected clusters of morphologically related Chinese characters on a pocket-sized screen. A full description of our initial purpose would be beyond the scope of this paper, and ultimately unnecessary, since the problem is in fact perfectly general. It only differs from Bin Packing by the nature of the items involved: instead of being atomic, each such item is a combination of elements, which themselves have two fundamental properties: first, they are all the same size (relatively to the bin capacity);

* Corresponding author.
E-mail addresses: aristide.grange@univ-lorraine.fr (A. Grange), imed.kacem@univ-lorraine.fr (I. Kacem), sebastien.martin@univ-lorraine.fr (S. Martin).
[1] Principal corresponding author.

https://doi.org/10.1016/j.cie.2017.10.015
Received 7 May 2017; Received in revised form 5 September 2017; Accepted 16 October 2017
Available online 11 November 2017

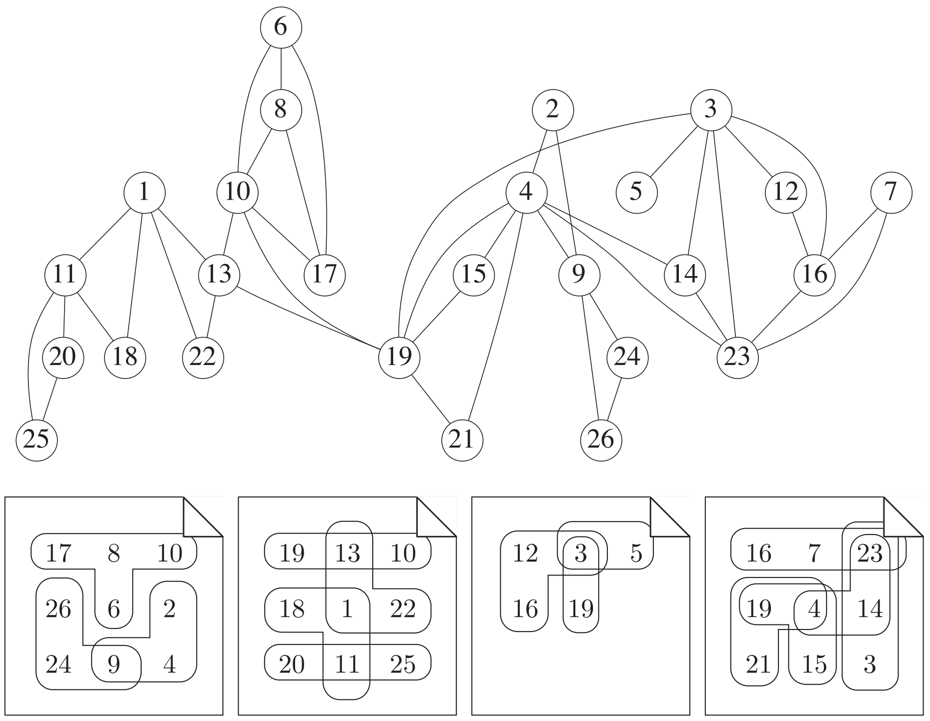

Fig. 1. Application of Pagination to the visualization of the cliques of a given graph. The cliques are split over a small number of pages (here, 4) with a given capacity (here, at most 9 vertices). To allow each clique to be entirely contained in a single page, some vertices may be repeated on different pages (here, 19 on 3 pages, and 3, 4, 10, 16 on 2 pages). The individual page layouts are not relevant.

and second, their combination is precisely what conveys the information we care about.

For instance, the members of a social network may interest us only to the extent that they are part of one or several friendship circles. Such groups of mutual friends are nothing more than the so-called cliques of a graph (Fig. 1, upper), but the cliques are notoriously difficult to extract visually. Visualizing them as separated sets of vertices is more effective, although quite redundant. A better compromise between compactness and clarity is attained by paginating these sets as in Fig. 1 (lower part). Note that, although no group is scattered across several pages, finding all the friends of a given person may require the consultation of several pages.

1.2. Definition and complexity

Our problem is not just about visualization. It extends to any need of segmentation of partially redundant data (see Section 1.3.1 for an application to virtual machine colocation). Let us define it in the most general way:

Definition 1. Pagination can be expressed as the following decision problem:

- *Input:* a finite collection $\mathcal{T}$ of nonempty finite sets (the **tiles**[2]) of **symbols**, an integer $C > 0$ (the **capacity**) and an integer $n > 0$ (the **number of pages**).
- *Question:* does there exist an n-way partition (or **pagination**) $\mathcal{P}$ of $\mathcal{T}$ such that, for any tile set (or **page**[3]) p of $\mathcal{P}$, $|\cup_{t\in p} t| \leqslant C$?

Example 1. On the left of Fig. 2, $\mathcal{T}$ = {{a,b,c,d,e},{d,e,f},{e,f,g},{h,i,j,k}} denotes the tiles to be distributed on pages (with respect to a given capacity of 7). On the right, we present 4 possible such paginations of this set. For easier reading, in the remainder of this paper, any set of symbols (especially, a tile) defined by extension (for instance, {w,o,r,d}) will be represented as a gray block: word. Moreover, in any collection of symbol sets (especially $\mathcal{T}$, or a page), the separating commas will be omitted: {May June July}.

Proposition 1. Pagination *is NP-complete in the strong sense.*

Proof. Any given pagination $\mathcal{P}$ can be verified in polynomial time. Furthermore, Bin Packing is a special case of Pagination with no shared symbol. Hence, the latter is at least as difficult as the former, which is strongly NP-complete. □

In the rest of this paper, we focus on the associated NP-hard optimization problem (i.e., where the aim is to minimize the number of pages).

1.3. Related works

Bin Packing and its numerous variants are among the most studied optimization problems, and, as such, regularly subjected to comprehensive surveys (Coffman, Garey, & Johnson, 1997, chap. 2; Coffman et al., 2013; Johnson, 1973; Martello & Toth, 1990). The purpose of the present section is far more limited: to extract from this vast literature the few problems which exhibit the distinguishing feature of Pagination, i.e., which deal with objects able to overlap in a non-additive fashion. For instance, despite its similar name, we will *not* discuss Newspaper Pagination, a Bin Packing variant studied in Lagus, Karanta, and Ylä-Jääski (1996) for the placement of (obviously non overlapping) blocks of text on the columns of a generalized newspaper.

1.3.1. Virtual machine allocation

Pagination was first introduced in 2011 by Sindelar, Sitaraman, and Shenoy (2011) in the context of virtual machine (VM) colocation. A VM can be seen as a set of memory pages: although most of them belong exclusively to one machine, some pages are identical across several machines. This happens more often as the configurations are close in terms of platform, system version, software libraries, or installed applications. If these VM run on the same physical server, their common memory pages can actually be pooled in order to spare resources.

The authors study two related problems. The first one, VM Maximization, is defined as follows: being given a collection of VM (our

[2] The fact that Pagination generalizes Bin Packing has its counterpart in our terminology: the *items* become the *tiles*, since they can overlap like the tiles on a roof.

[3] Likewise, the move from concrete to abstract is reflected by the choice of the term *page* instead of *bin*.