HELSINKI UNIVERSITY OF TECHNOLOGY
Faculty of Information Technology
Computer Science

Krista Lagus

Automated pagination of the generalized newspaper using simulated annealing

Master's thesis
Supervisor professor Markku Syrjänen
Instructors Lic.Tech. Ilkka Karanta and Ph.D. Juha Ylä-Jääski
Otaniemi, Finland 8.11.1995

Contents

1 Introduction **3**

1.1 Scope and contributions of the thesis 4

1.2 Organization of the thesis 4

2 Electronic publishing **6**

2.1 Visual publishing on the electronic media 6

2.2 Electronic newspapers—state of the art and future visions 7

2.3 Present tools and systems supporting electronic publishing and automatic pagination 8

2.3.1 Presentation of data 8

2.3.2 Tools for interactive layout design 11

2.3.3 Tools for automatic pagination 11

3 Automation of pagination **13**

3.1 Pagination 13

3.1.1 Page layout 14

3.2 Automatic pagination 16

3.2.1 Justification and objects of automatic pagination 16

3.2.2 Tasks 17

4 General layout problem—history and theoretical approaches **19**

4.1 Analysis of the layout problem 19

4.1.1 Type 1: Fixed size, fixed contents 19

4.1.2 Type 2: Varying size, fixed contents 19

4.1.3 Type 3: Fixed size, varying contents 20

4.2 Algorithmic approaches 21

4.2.1 Graph-theoretic approach 21

4.2.2 Knapsack-problems and bin-packing 22

4.2.3 Genetic algorithms 22

4.2.4 Simulated annealing 22

5 Case studies **27**

5.1 Problem description 27

5.2 Selected approaches 27

5.3 Fax newspaper prototype 28

5.3.1 VTT's V·I·P^E 28

5.3.2 Extensions to V·I·P^E 31

5.4 Simulated annealing, experiment 1: straightforward implementation 34

5.4.1 Selection of material—variable-base adder 35

5.4.2 Creating the layout by simulated annealing 37

5.4.3 Results 40

5.5 Simulated annealing, experiment 2: utilizing domain knowledge . 42

5.5.1 Results 46

5.6 Comparison of approaches 47

6 Conclusions and future directions **49**

6.1 Automatic pagination and electronic publishing 49

6.2 Simulated annealing 49

Acknowledgements

I would like to thank my instructor Ilkka Karanta for discussions and insightful comments about pagination in general and the theoretical foundations of this work. Warm thanks also to the other people at the Intelligent Pagination Group at Technical Research Centre of Finland (VTT), especially Teppo Veijonen, for valuable advise into the secrets of V·I·P^E and newspaper pagination, as well as many good questions and helpful comments about the whole. Special thanks to Janne Sinkkonen who had the patience to read through several drafts of this paper, and who offered an invaluable discussion partner especially for the sections concerning simulated annealing. Thanks also to the other people who read and commented on the work, Timo Honkela, my other instructor Juha Ylä-Jääski and my supervisor Markku Syrjänen.

The work described in this thesis was supported by Technical Research Centre of Finland (VTT).

1 Introduction

This thesis discusses automatic pagination applicable to a publication where a collection of pieces of information with varying sizes and shapes and without any strict sequential order is presented in page-like visual entities. The previous description portrays a *generalization of newspaper*. The medium of presentation may be paper, fax, TV or computer screen. The content may be derived or produced in a number of different ways, such as written by humans or compiled as a result of a database query or a World Wide Web search.

A pagination system where automatic pagination might be applied can be described as a publishing system for the generalized newspaper. It has the following properties:

- The material is derived from a large and gradually updated information storage.
- There is a need for selection of material: the system offers individualized views of the storage contents to the customer, according to order and possibly other information.
- Pagination is used to create the visual presentation of the publication. A generalized newspaper is formed, containing some or all of the material of interest.

Automatic pagination systems might be an answer to the paper publications struggling with high production costs and need for increased speed of production. In addition to traditional newspapers, automatic pagination might be applied to pagination of newspapers delivered by fax or in electronic newspapers, including ones that contain hypermedia properties, as well as in visualization of database queries and WWW search results. Furthermore, the pagination problem is studied here in such a general level that the methods are applicable to other similar problems as well.

Simulated annealing is a stochastic optimization method based on the statistical mechanics of cooling solids. It has previously been applied to large combinatorial problems, typically including some types of layout problems in VLSI design and cutting metals. It is thus a tempting method to apply to the layout problem within automatic pagination.

1.1 Scope and contributions of the thesis

In this work, automatic pagination and the methods for solving the layout problem are studied[1]. In particular, methods such as simulated annealing, which could be utilized in the automatic pagination of the generalized newspaper, are considered. Electronic publishing is also discussed briefly to illuminate where and how automatic pagination would be useful. In addition, an example application context, a newspaper delivered on fax, is described in detail. This context offers a problem definition and criteria for the comparison of different methods of automatic pagination.

The standards of pagination employed in comparison of the methods were mostly derived from goals of the traditional newspaper pagination. This choice of goals is justified for several reasons. The readers are used to the newspaper format, and even when viewing information on screen, they may find reading easier and more pleasant if there is a visual resemblance with the old medium. In addition, the readability of a newspaper layout has been developed through years, and has evolved into an easily readable format for presenting short pieces of information. Furthermore, the currently available computer screen resolution is inferior to the resolution of printed matter, and reading from screen is thus not nearly as pleasant as from paper. Thus it is probable that paper as a medium for reading will not disappear for a long time, although the delivery of information may increasingly take place over computer networks and other electronic transfer media.

Contributions of this thesis include the following:

- Assessment and slight improvement of V·I·P^E, an automatic pagination system with a heuristic graph-search algorithm.
- Addition of some user control to pagination of V·I·P^E.
- Design of an approximate best-first mechanism for selection of articles and article versions, represented with a variable-base adder.
- Implementation, analysis and comparison of two approaches for experimentally applying simulated annealing to the pagination of single page layouts.

1.2 Organization of the thesis

In chapter 2, publishing news-like information on the electronic media is discussed, with views on some of the currently available tools and methods.

[1] Only page layout desing is studied, neither article layout design nor inter-pagial relationships or constraints are considered in this work.

Chapter 3 discusses pagination and how to automate it. The principles of traditional manual pagination are described, and the aspects that an automated pagination system should take into account and tasks that should be performed are discussed in detail.

Chapter 4 puts the pagination problem within a larger theoretical context, and works as an introduction for the methods used in case studies of chapter 5.

Chapter 5 presents three case studies on tentative implementations attacking the problem of automatic pagination of a newspaper delivered on fax. One of the implementations utilizes heuristic graph search, while the two others are differing implementations of simulated annealing. These implementations are not full-fledged newspaper fax production systems but studies of the automatic pagination, aimed at illuminating the usability of each method in context of the given problem.

Chapter 6 presents the conclusions that were drawn from the whole work and especially the case studies. Some future directions are suggested both for implementation of simulated annealing in general as well as for the automatic pagination of a newspaper delivered on fax and other electronic publications.

2 Electronic publishing

While eletronic means of information transfer, manipulation and visualization have become more common, a growing interest has appeared in two overlapping directions: publishing on the electronic media (such as the Internet), and individually customized publishing. Both avail themselves to heavy use of electronic tools and automation of the publication process. Publishing for the electronic media as well as using electronic tools to design publications are the topics of this section.

2.1 Visual publishing on the electronic media

The following are examples of publication categories on the electronic media which, in addition of the textual content, utilize varying degrees of visual means to communicate with the reader/viewer:

- WWW,
- CD-rom products; books and hypermedia etc.,
- electronic newspapers, and
- TV.

Of these, TV is not further considered.

Publishing news-like information on the electronic media does not necessarily require lots of changes compared to publishing on the more traditional medium—paper. However, transferring paper publications to be viewed on e.g. computer screen may not be sense unless the differences of the media are taken into account.

At best, publishing for the electronic media may make heavy use of the potential of the medium. The publication may adapt cleverly to the different hardware and software environments such as varying screen sizes, fonts, colours, available fast and slow storage space and processing speed. Also, deriving content run-time from networks and databases is possible and advisable. The structure of the publication may be much more network-like—*hypermedic*—with much more references and possible reading paths than with paper publications. The most important benefits of electronic publishing are, however, the possibility to provide fresh contents and to take into account the individual needs of users by compiling a different publication for different users.

A perfect utopia about an electronic newspaper is one where the user describes his/her interest profile and an intelligent agent (or several of them) writes a fresh newspaper based on the personal interests of the reader.

In a more realistic vision, the content is not completely produced run-time, but compiled and laid out of pre-written articles and pre-designed advertisements and other components residing in a database. The articles may have several alternative versions that for instance have different lengths or include different amounts of details and background knowledge or different aspects of the topic. Some of these versions could be explicitly planned beforehand by the editor, others could be compiled from smaller parts.

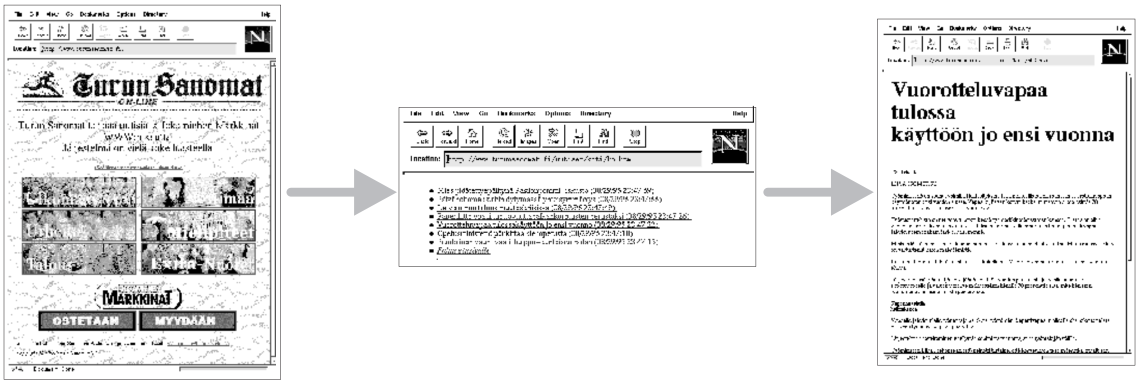
2.2 Electronic newspapers—state of the art and future visions

Many of the current electronic newspapers (e.g. OtaOnline[2] and Turun Sanomat[3]) in the Internet WWW seem to share similar design. There are a few "front pages" for which layouts are actually built by hand. From the front pages one can reach a list of clickable topics, and from each topic, article headings list etc. The hand-made front pages are either topic categories that remain constant over time (as in Turun Sanomat, see Fig. 1 a), or are built separately for every paper (as in different newspapers participating in OtaOnline). In each electronic newspaper, a page with a list of article headings (see Fig. 1 b) may be automatically produced, but the results are not very impressive. The page looks rather boring and does not encourage one to proceed to the articles themselves, because the headings are presented as a plain list and the long lines with no variation do not help the reader in picking out an interesting article. The dullness may be due to the fact that the available visual design tools for publishers have so far required a considerable amount of human labour, and it may not have been considered worthwhile to spend so much time on the visual design.

With automatic pagination the list of article headings could instead be a page laid out with article headings and captions, with for example headings of different size describing the length or priority of the article. One could then click a heading to reach the whole article. The page laid out with headings and captions could include advertisements as well.

Methods and tools such as SGML, HTML and graphical WWW-browsers have enabled a big step towards automatic production of the layout by providing the means to easily structure information. These tools are designed especially for

[2] `http://www.otaonline.fi`
[3] `http://www.turunsanomat.fi`



a. The front page b. Article titles c. An article

Figure 1: a) *The front page and b,c) samples of two subsequent layers of hierarchy of Turun Sanomat*

viewing the information with varying types of computers and different screen sizes and settings. The layouts are produced with a set of rules. The basic idea in these visualization tools is simple: the text and figures flow from left to right, from top to bottom into one wide column. Also reading is expected to take place in this order. Although this assumption is adequate for many publications, it is occasionally artificial and too restrictive, as often is the case with newspaper-like information.

Furthermore, more versatile and visually interesting layouts can be built with these tools only by rather explicit hand-made design of the outlook, and the material to be put to the page must be rather precisely known when designing the outlook in order to ensure visually pleasing results. This type of design is comparable to the hand-made design of newspaper page layouts—one exception being that in newspapers the page height is fixed, whereas in WWW there exists no such concept or restriction as page height.

2.3 Present tools and systems supporting electronic publishing and automatic pagination

2.3.1 Presentation of data

An important consideration in electronic publishing is how the data to be published is viewed and presented at different stages of the process. Important stages may be identified at points where the data needs to be communicated from process or person to another. These presentations include the one used by the writer of an article, the one that is given to the pagination process and the one used to communicate the pagination results to visualization tools. Below some of the relevant views on the data on the different stages are identified for text, articles and page layout. The quality of a data representation scheme may be evaluated by assessing how well it is able to provide a representation for the different needs described below.

- Text may be viewed in the following ways:
  - In an initial view the thext is seen as *words composed of letters*. The writer typically sees the text this way.
  - In a more structured view words have styles, e.g. "emphasis", "strong". This may also be used by the writer.
  - A fully visual view of the text contains typography such as fonts, spacings, sizes, perhaps colour, special visual effects such as large capital letters starting a paragraph. It is used during the visual design of the publication, such as in article layouting.
- The different ways to view an article include the following:
  - An *abstract object* being of some topic, possibly with a content that is not exactly fixed. The view is used in deciding what to put on a page and into a publication.
  - Visual outline, inside which all the contents of the story reside, regardless of the content of the article. This view is relevant in pagination process.
  - Structured contents: headings, body text, figures, cross-references, appendices. These structures are relevant for creation of different versions of the same article, e.g. when deciding whether or not to include a picture.
  - Contents with exact looks in a fixed order: text chunks, figures etc. The view is relevant when building the final page according to decisions made during pagination.
- Page layout:
  - Page as departments and subdepartments such as classified advertisements, page head, editorial material of a certain topic etc. Relevant before and during selection of material.
  - A collection of more or less abstract objects that are supposed to be put on the page. Between selection of material and pagination.
  - Boxes of fixed dimensions on the page with certain locations, shapes etc. The result of pagination.
  - As a visual image on screen or paper. Relevant when the data is given to visualization tools such as printing programs.

Thus, a language for expressing the different representations is needed. Especially the representations before and after pagination are of interest in this work. One alternative for representing is to use a document formatting language such as PostSript, LaTeX or TeX to represent layouts. However, even more general approaches are emerging with the invent of SGML (Standard Generalized Markup Language) and HTML (HyperText Markup Language), a language defined with SGML especially for hypertext documents. Adobe PDF-format offers even better means for describing static page layouts, but unlike HTML, does not include any hypermedia capabilities. A new standard, MHEG, has been proposed for description of multimedia documents. It remains to be seen what effects for electronic publishing it will have and whether design tools will be implemented supporting it. Therefore, the currently most interesting standards—SGML, HTML and DSSSL—are discussed below.

SGML. Standard Generalized Markup Language is a metalanguage for describing markup languages enabling one to describe document structure apart from the data and its visualization (Sperberg-McQueen and Burnard, 1993; Goldfarb, 1990). From the point of view of electronic publishing, SGML may be used to structure the document during writing and before storing it to the article database. With SGML the writer is not forced to worry about the visual outlook, only the logical roles of parts in the article.

In editorial publishing, SGML offers a simple way for the writer to describe several views to the same document. The editor may specify different levels of depth of the material by tagging some parts as optional or alternative. Material tagged as background knowledge may be further categorized as much as is useful, for example according to interests or scientific fields. Thus, there is no need to write all versions of the article separately, but just to write production rules in SGML and to later read them with an SGML-parser when there is knowledge of the specific reading instance (time, reader information etc).

HTML and WWW. HTML, HyperText Markup Language, is a language described in SGML and designed for description of hypermedia documents. The WWW (World Wide Web) of Internet is a network of huge amount of documents (perhaps $2 \ldots 10 \times 10^6$) written in HTML (or some of its extensions) and linked to each other with HTML's hyperlink mechanism. These documents are situated on the information provider's *WWW-server*, and may be accessed and read by any of the numerous *WWW-browser clients*, usually residing at the reader's system.

With HTML, it is possible to write documents while specifying only semantic roles, *logical styles*, for the different parts of the document. In this case *WWW-browsers* have the freedom to visualize the given logical styles as they see fit.

Examples of logical styles are *level two heading, author, address* and *hyperlink*. HTML also enables one to describe some alternative information, e.g. pictures may have alternative brief textual descriptions, which are presented to users having no graphics viewing capabilities.

In addition, some WWW-browsers support non-standard extensions to HTML enabling the author to describe in more detail the physical outlook of the document on the reader's screen. These details include spaces between items, font sizes, colours etc. On the negative side, usage of these visual enhancements requires a lot of human design on top of providing the contents of the document.

DSSSL. Document Style Semantics and Specification Language (DSSSL) is described in Draft International Standard ISO/IEC 10179 (ISO, 1995). The draft DSSSL proposes a generalized transformation method between different types of document structures as well as enables semantic prosessing of documents.

DSSSL may be used with SGML but is not restricted to it. It, unlike SGML, will offer a way to describe the visual document layout produced by the pagination process, i.e. the placement of articles, sizes, fonts etc. This description in turn may be interpreted by future DSSSL-conforming visualizing tools or printing systems.

2.3.2 Tools for interactive layout design

There exist several tools to support manual design of pages, such as PageMaker, FrameMaker and the PDF-format of Adobe. In addition, there are tools and methods for publishing on the Internet, which require a lot of interactive design: HyperText document description language HTML used as the basis of World Wide Web (WWW) documents (see section 2.3.1 for further details of HTML and WWW), several different HTML-capable document browsers such as Netscape, Mosaic, Lynx and HotJava, editors supporting the writing of HTML-documents etc.

2.3.3 Tools for automatic pagination

There are not many tools for automatic pagination. However, in the past years some results have been obtained on this area at Technical Research Centre of Finland (VTT). The different versions of V·I·P (VTT's Intelligent Pagination) suit different kinds of pagination tasks: pagination of newspaper advertisement sections (see Ylä-Jääski and Ahonen, 1990), yellow page phone directories, prototypic pagination of editorial material of newspapers delivered by fax (V·I·P^E, see

Veijonen et al., 1994 and section 5.3). Of these, the yellow page application is in production use in Pindar Set (Great Britain), in Turun Sanomat (Finland), and the newspaper advertisement application is used in Dagens Nyheter (Sweden) and in Nederlandse Dagbladunie (Netherlands).

For automatic design of article layout, a prototype and method has been designed by (Kukkonen, 1992) and integrated into Grafimedia's Marita publishing system. According to (Kukkonen, 1992), given the width of the article as number of columns, the prototype is able to design an article layout that fills the usual design criteria of newspapers, and produces at least as good article layouts as professionals. The ability to design an article layout for the given textual content and article width forms a part of the basis for the automatic design of whole page layouts of editorial material.

3 Automation of pagination

3.1 Pagination

Pagination may be described as the process of producing good page layouts for given material. It is a clearly distinguishable task in the prepress phase of a newspaper (Ahonen and Viertiö, 1989), and its abstraction can be extended to the generalized newspaper as well. Newspaper pagination includes two main subtasks: (1) distributing given material onto a range of pages and (2) preparing the presentation of the material, i.e. the *page layout*.

Successful pagination requires a preprocess that maps the logical structure of a newspaper to the physical structure, which is defined by the printing press configuration or the restrictions of other visualization media. In newspaper pagination in particular, this translates to taking into account such things as what pages contain colour, which is the middle sheet of the newspaper and what constraints are imposed by the printing machines and process. The latter include such visual constraints as what is the smallest visible font size or the resolution of figures, and constraints adressing the content such as placing cut-out coupons on the edges of pages instead of on opposite sides of the same physical page, or distributing advertisements from competitors on different pages.

In pagination of the generalized newspaper there is often an abundance of material with different priorities and expiration times, instead of a strictly selected set of material that must be fitted into the publication. The material for a traditional newspaper may be extracted from a large article database. Many database queries as well as WWW searches now produce hits with methods such as scoring, giving only a priority value for the hits. In other cases the aim may be to display a fixed set of material while minimizing the total space. Therefore, the pagination task may be either one of the following:

- paginate all of the produced material while minimizing total space usage, or
- try to fill the given, fixed space with relevant material extracted from a large collection of material.

The first case arises in practice e.g. in pagination of telephone directories and newspaper advertisement sections. The second case is more common with pagination of editorial material. In both cases, devising page layouts is required. In other words, the general goal of pagination is *to form good page layouts out of some material,* where high quality is described in terms of constraints and/or optimization criteria. In this work both the above mentioned variations of the pagination problem are considered with emphasis on the general problem of producing good page layouts.

3.1.1 Page layout

Since the goal of pagination is to produce good page layouts, it is necessary to specify what the goodness of a page entails. Page layout provides a way to catch and direct the user's attention, control the order in which different parts are read and for how long time. It is also a way to visually group, divide and structure information and present it in an easily readable and aesthetically appealing way (Veijonen et al., 1994). In addition, all this must be carried out efficiently, i.e. with as little waste of resources as possible. Three aspects of producing good page layouts may thus be identified: saving space on the medium, cognitive considerations and aesthetic considerations.

Saving space on the medium is an important issue especially on paper or fax distribution, but also the computer screen space that can be used at any given time is very limited. Therefore, efficient usage of layout space is often important. For example, the yellow pages of Helsinki area contain about 1300 pages. If 1% of the space (13 pages per publication) may be saved by improved pagination methods, it amouts to saving millions of sheets of paper within Helsinki area. In some cases and especially in human-designed layouts empty space may also be used in meaningful ways such as to emphasize some area. However, since this kind of artistic or semantically inspired usage of empty space is rather delicate and requires artistic insight or understanding which is immensely difficult to build into any automatic system, it is not further considered here.

Cognitive considerations refer to those aspects of the publication that affect such processes as quickly finding the relevant information, (what you are looking for), grasping the status contrasts between items (advertisements vs. editorial material and scientific vs. speculative) or topic differences (weather, foreign news, editor's view etc) between items. These processes are partially affected by the same properties that affect the aesthetic value of the page. Cognitive considerations are relevant to three different interest groups interacting via the newspaper: the reader, the editor or producer of the newspaper and the possible advertiser. Table 1 depicts the goals and means to achieve the goal that these interest groups have.

Aesthetic considerations. One of the most difficult aspects in automatic pagination is how to make the page aesthetically appealing. Most of us, given a page

Interest group	Producer	Advertiser	Reader
Goal	Profit, quality, fast production, low cost	High impact	Informative value, pleasure, efficiency of reading
Means to achieve the goal	Replacement of some human work by automation	Placement of the ad on the same view with interesting information, maximizing saliency	Selective reading guided by the page layout

Table 1: *Different goals and means of the interest groups of a generalized newspaper.*

to look at, have an opinion of how good a page it is aesthetically. Nevertheless, it is much more difficult to describe what makes that page a good one. The good pages may look very different, have elements with different sizes and shapes in varying relations to each other. Typically the publishers want their publication to have an individual and consistent flavor, which is achieved with original but sophisticated design.

Aesthetic considerations play a major role when page layouts are designed by humans. This is also the most difficult aspect of the process to automate, because aesthetic appeal is not easily definable in terms of explicit rules. Nevertheless, to be able to take into account any aesthetic values in automatic pagination, it is necessary to have an idea of what makes a page aesthetically appealing.

Five main principles are identified in (Rehe, 1985): organization, simplicity, contrast, balance and harmony or unity. Some similar discussion is carried out also in (Henning et al., 1988). The questions for automating the pagination are, how to evaluate the fulfillment of these principles on existing pages and how to build pages that conform to these princples. Below are some practical suggestions for the design of page layouts:

- Contrast: There should be both large and small articles on a page, as well as headings of different sizes. One large photo and strong heading on one page is enough (Rehe, 1985). There should be at least one strong element on a page.
- Balance: There should be balance between the strong elements and different areas of the page, so that no area is misrepresented. Absolute balance should also be avoided (Rehe, 1985). A suitable balance may be achieved by positioning pictures apart from each other and trying to avoid producing completely symmetrical layouts.

- Harmony or unity: On each page, the designer should control the number and distances of the imaginary lines that form on the page as continuations of article borders. Too many separate lines should be avoided, since this produces easily too much variation and too little harmony. On the other hand, too few lines on the page often signify a dull page. Also the distances between the lines should be controlled. Lines that are too close to each other are visually rather disturbing.

3.2 Automatic pagination

3.2.1 Justification and objects of automatic pagination

On-demand publishing is used to provide readers with fresh information, as well as to customize publications for different user profiles. When customized on-demand publishing is taken to the extreme, each customer receives an individually designed edition of the publication. In practice, this degree of customization can only be achieved when the publication process is nearly automatic. Currently, the part of the process that most readily may be automated is the pagination. As a multitude of tools and standards for publishing on the electronic media have emerged in recent years, automatic pagination has become an interesting option for a variety of situations.

Automatic pagination may be useful anywhere where pagination is considered useful but too expensive or too slow by traditional means. The presence of the following factors might therefore suggest using an application carrying out automatic pagination:

Limited production resources. Availability of some kind of automatic pagination might enable small publishers or even individual people to produce high-quality publications. This may be the case for example with newspapers delivered by fax to a small group of users, where the contents are restricted excerpts from traditional newspapers. The situation also arises in the Internet, where anyone who wishes is able to become an information provider with very small investment.

Need for fast delivery but some visuality. Sometimes there is a need for fast delivery of information to the customers, and in the extreme case even a "quick and dirty" interactive pagination would be too slow. Despite that pagination could then be skipped altogether and the items could just be put on page one after another, visuality may nevertheless be considered important.

Customer-dependent content. Publications where the information content heavily varies for different users or user groups, must accordingly be visualized individually. This situation may arrive with customized newspaper, database searches etc.

Variable viewing dimensions. In some publications, the information content presented may be identical for everybody, but the layout requirements (such as page size) vary between users. This situation arises often with publishing on the electronic media, where screen sizes may vary considerably. And not only the screen sizes: the user may wish to view several different publications or documents simultaneously on the same screen. A publication with dimensions that can be adapted at time of view would be useful. Also the medium of the publication may vary: sometimes the user may prefer to read from the screen, at other times from a print.

3.2.2 Tasks

The production of a page layout consists of two main tasks: Selecting the material to be paginated, and designing a page layout for the material. These tasks may be carried out sequentially, or they may be mixed by letting the optimization of the visual outlook affect the selection of material. For example, filler material may be used to fill spots that would otherwise remain empty, or an article version may be replaced with another if that eventually seems to fit better to the page. These tasks are affected by the different goals that can be associated with pagination (see Table 1 in section 3.1). The balance between partially conflicting goals such as minimization of empty space and maximization of high-priority material on page must be decided by the publisher.

Selection of material. The content of the generalized publication may be obtained from a large collection of material, which may be viewed as a kind of database. The database might, for example, consist of all the abstracts of scientific articles obtainable from the Internet via WWW searches (see section 2.3.1) or all the Usenet News articles of the group *sci.med,* or all the stories written by the editors of the electronic newspaper that one orders.

From the database, one probably would want to restrict the amount of material eventually viewed. This might be preferred in one of several ways of classifying or scoring text chunks based on rules, keywords, hierarchy of subject classes, different properties of the pieces of material such as how recent they are, who wrote them, subject etc. Data mining techniques or methods such as classification by artificial neural networks algorithms might also be used for this task. There exist learning algorithms that recognize the reader's reading pattern from previous experience and match it against yet unseen material (see for example the adaptive scoring of Ding Gnus by Ingebritsen, 1994). This kind of categorization might also be carried out by associative neural networks such as SOM (Kohonen, 1995).

Pagination. The material must be assigned to individual pages, and each page layout needs to be designed. The problems associated with this task are discussed in detail in the next section, starting from general principles.

4 General layout problem—history and theoretical approaches

In this section, different kinds of layout problems are presented, and several algorithms traditionally used to solve them are described.

4.1 Analysis of the layout problem

Viewed abstractly, producing a layout requires assigning a given stock of items to given space. Historically, layout research has been carried out in at least two research fields, namely operations research and artificial intelligence. The layout problems studied in these fields may roughly be divided into two categories: the minimization of empty space within a given, fixed area, and the minimization of area used for a given, fixed set of items.

4.1.1 Type 1: Fixed size, fixed contents

In certain layout problems the aim is to fit the given contents to the given, fixed area, or conclude that this is impossible. These may be formulated as quadratic assignment problems (QAP), i.e. assigning N items to N locations[4]. The precise formulation depends on the problem. Unfortunately, the QAP is NP-complete (well-known, see e.g. Kouvelis et al., 1992), and does not suit well to 2-dimensional or 3-dimensional problems with varying-sized elements, because the available positions depend on what elements have already been inserted.

4.1.2 Type 2: Varying size, fixed contents

In some problems, all the given material must be laid out, simultaneously minimizing the total space or transportation of resources between items, or while some additional constraints between the items are obeyed.

In this set of problems, active research can be found for instance from operations research. The problems vary from VLSI chip layout design to plant machinery layout design and to deciding the placement of interacting departments and facilities in an office building. Further taxonomy for machine layout problems has

[4]The layout problem may be described as a QAP if there is a one-to-one relation between layouts and permutations of articles. A prerequisite to this approach is thus to design a procedure to translate a given permutation into a layout and prove that all interesting layouts are produced in this way.

been described in (Heragu, 1992). Also some problems mostly studied in artificial intelligence may be viewed as falling into this category: in pagination of yellow page telephone directories or newspaper advertisements all the items must be placed into the publication whose length is not exactly fixed in advance.

A number of optimal algorithms have been developed to solve the layout problem. However, due to the NP-completeness of QAP, these algorithms cannot be used to solve large-scale layout problems, such as problems with fifteen or more items (Heragu and Alfa, 1992). Therefore, also different types of suboptimal algorithms have been designed, falling into the following classes:

- graph-theoretic algorithms,
- simulated annealing (several approaches, see eg. Heragu and Alfa, 19922), and
- genetic algorithms.

4.1.3 Type 3: Fixed size, varying contents

In some problems the space to be filled is fixed and must be used as efficiently as possible by items from the given stock. Application areas falling under this category include cutting required shapes and sizes from cow skin, textiles or metals and filling a series of trucks with material to be transported. Sometimes additional constraints are defined e.g. for the the cutting equipment, such as that the machine only can make "slice cuts" (i.e. cut a piece with a straight line from side to side with no turns or corners on the way). Another example of such constraints is found for instance from filling a truck with given material having handling orders, with constraints like some items may not be placed under other items.

The goal is thus to fill the given space with items selected from the stock and to minimize wasted space. It is not, however, necessary to place all the items from the stock to the layout.

Problems of this kind can be formulated as some variant of the knapsack problem (fill a sack as full as possible from the given stock, or fill the given stock to as few sacks as possible). In this group of problems the sacks have fixed sizes, but the items to be fitted into each sack may vary. The problem of filling any single page of a newspaper falls into this category.

4.2 Algorithmic approaches

Some of the algorithmic approaches mentioned above will be described here, including those that were experimentally applied and studied in practice in this work (see next section).

4.2.1 Graph-theoretic approach

One way to solve a problem is to describe it in terms of a graph with nodes as states in the problem-solving process and edges as transitions between states, i.e. steps to take in search of a good solution. There exists a multitude of different graph-searching techniques from elementary brute-force methods such as depth-first and breadth-first to rather exclusive ones such as tabu search. We will discuss only one of these here, the *heuristic graph search*.

Heuristic graph searches. Heuristics are rules of thumb that guide the search of good solutions to a problem. In graph searches heuristics are used to select the most promising-looking branch of the search tree or graph. These branches may then be examined first, before others, or even to cut out parts of the search tree altogether.

Good heuristics can reduce the time required to find a good-enough solution, compared to brute-force methods. However, with bad heuristics, finding the solutions may take very long. If the heuristics are bad enough, they may guide to bad solutions instead of good ones. Thus, the heuristic algorithm may perform even worse on the average in terms of time used to find an acceptable solution than a brute-force search without heuristics. Even worse, the acceptable solutions may not be found at all, if the heuristics are used to cut out branches of the search tree without examining them.

The problem with heuristics is how to decide half-way what would be an appropriate next action, i.e. how to design heuristic rules that lead to good solutions instead of bad ones. The following are only a few examples of possible heuristics that could be used to guide a search for good layout configurations:

- Try to insert the articles in order of decreasing size.
- Insert an article to the smallest empty region where it fits.
- If there is a large empty region on the page, first try to insert the largest article fitting in that region, and fill the remaining space with small articles.

- When inserting into any empty region: try to find combinations of articles that fill the region. Prefer combinations where the article sizes vary (alternative: prefer combinations that leave rather one big gap than many small gaps on the page).

4.2.2 Knapsack-problems and bin-packing

The third type of layout problems (fixed free area, variable content) may be approached as bin-packing problems. The basic bin-packing problem, as described in (Hofri, 1987), consists of an unlimited (or large) number of bins of equal size and of a number of items of variable sizes. The task is to fit the items into the bins so that a minimum number of bins is used and the capacity of no bin is exeeded. Some pagination problems fit this formulation approximately: Each page is represented as a bin, and the given material is represented as the items to be packed.

There are many algorithms for solving bin-packing problems in one dimension, although the problem is NP-hard in the strong sense. However, as the dimension grows the problem quickly becomes very complex. Even the 2-dimensional case is rather difficult to solve. Furthermore, many pagination problems incolce different kinds of constraints (see section 3.1 for examples in newspaper pagination) that would need to be integrated into the pure bin-packing algorithms.

4.2.3 Genetic algorithms

Genetic algorithms (GA) have been applied to latter two of the above described three problem types. The major difficulties, as reported by (Hwang et al., 1994) have been encountered in developing a representation of the bin-packing problem that is suitable for a GA and is not limited to cuttings representable by slicing trees. The same authors also reported using GA's successfully for improvement of the solutions found by heuristic algorithms, and suggested combining simulated annealing with genetic algorithms.

4.2.4 Simulated annealing

Simulated annealing (SA) is a stochastic optimization method that may be used to minimize a given cost function (energy), given a possibly combinatorial system with many degrees of freedom. The algorithm has been described in numerous publications, including the seminal paper (Kirkpatrick et al., 1983), and a book (Aarts and Korst, 1989) containing a thorough analysis of the theory and its relation to Markov Theory, and applications. A concise overview of the SA may be found e.g. in (Haykin, 1994). When not otherwise stated, the following description of SA is in accordance with (Haykin, 1994).

The idea for SA is drawn from statistical physics, especially from the theory of cooling solids. According to the theory, if a substance is cooled slowly enough from a sufficiently high starting temperature (melting point), it will arrange itself into a low-energy lattice structure. If the cooling is too rapid, the system will freeze into a non-optimal state, with "errors", i.e. unsymmetries, in the lattice structure.

From metaphor to algorithm. The annealing metaphor can be changed into an optimization algorithm as follows:

- Cool temperature of the system slowly.
- At each temperature,
  1. suggest a small random change to the system state,
  2. calculate the energy change associated with the state change, and
  3. choose whether to accept the change:
     - Always accept changes for the better.
     - Accept deteriorating changes "often" in high temperatures, and "rarely" in low temperatures.

In this algorithm, "temperature" translates to a probabilistic parameter defining how often deteriorating changes are accepted, and "cooling" into reducing the probability. A solution to the problem corresponds to the state of a physical system, and the cost of the solution to the energy of the state. Thus, optimal solutions (and faultless crystals) are represented by the global minima of the cost or energy landscape (see Fig. 2).

The changes are accepted according to the following function:

$$P(accept) = \begin{cases} 1 & \text{if } \Delta E < 0, \\ \exp(-\frac{\Delta E}{T}) & \text{if } \Delta E \geq 0, \end{cases} \tag{1}$$

where T is analogous to temperature and E refers to energy of the state. After enough rounds in a temperature, the system will reach a *thermal equilibrium* in which the probability of being in a state depends on the energy of the state, i.e. $P(state)$ is a function of $E(state)$, the Boltzmann distribution, for the given temperature T. When the system has reached the thermal equilibrium, the temperature may be decreased.

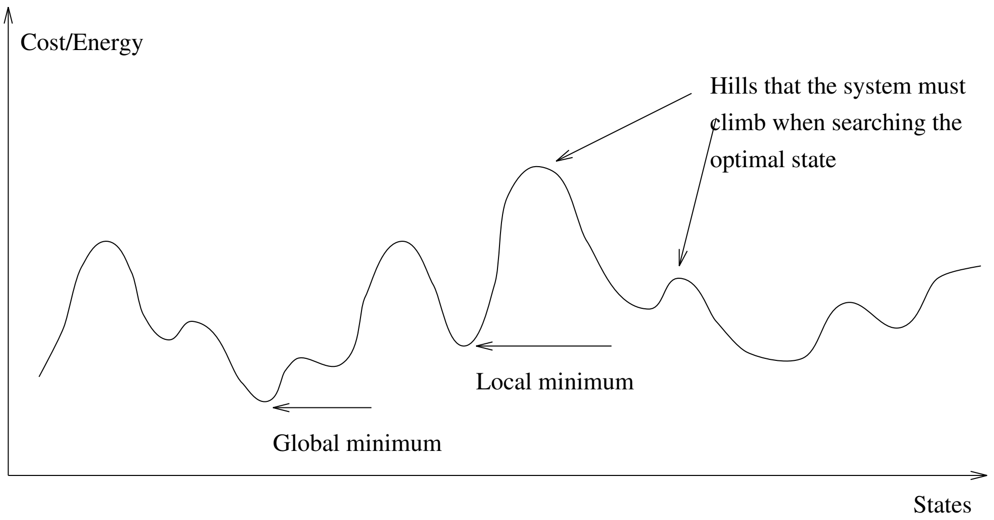


Figure 2: *Energy (vertical axis) defined for a one-dimensional state space.*

Thus, at high temperatures (i.e. in the beginning of the simulation), changes are mostly accepted regardless of whether they are improvements or not. As the temperature slowly drops, the big up-hill changes are increasingly rejected. Finally, in zero temperature SA behaves like a normal gradient-descent search technique where only downhill changes (i.e. immediate improvement in the system state, measured by the energy function) are accepted.

To understand why this method works, the following intuitive explanation may help: When temperature is high, there is enough thermal noise in the system to climb over high hills of the energy landscape (see again Fig. 2). With low temperatures, the thermal noise is lesser and high hills are rarely climbed over. Finally, with zero temperature, the state has no "internal energy" to climb over high hills and only downhill changes are possible. At this point the system soon freezes into one state or oscillates between states with equally low energy.

Practical implementation. The following steps must be taken when implementing this algorithm for a given optimization problem (Haykin, 1994):

1. Define the problem concisely, including the *state space* (i.e. states that the system is allowed to go into during the simulation) and associate a cost with each state. The cost function is used to express the constraints and goals of the problem.

2. Define state transitions, and a way to locally calculate the energy change for any transition.

3. Decide on a cooling schedule.

To sensibly apply SA, the state space must be defined so that there is some kind of topology, and so that the connectivity between states, defined by the state transitions, is not complete. It is evident why the connectivity should not be complete: with complete connectivity there is a path of length one from any node to any other node. In this situation full search performs better than SA. This fact becomes evident even without a mathematical proof from the following analogy: The state space is represented by an apple basket and each state by an apple. With each step, one apple is selected from the basket. In deterministic, full search, the apple is taken out from the basket and measured, and not returned. The best apple is always kept aside. Each apple is encountered exactly once. In SA, the apple is returned to the basket after measurement, and may subsequently be retrieved and measured many times, because of the symmetry of the states. SA has no memory of what apples (states) were already measured.

The need for "sufficient" connectivity is also intuitively rather clear: if the connectivity is minimal, in the worst case the state space is a linked list of states. Again, deterministic search, i.e. going through the states in the list order, performs clearly better than SA. This is because for SA to work according to the theory, each temperature should be kept long enough so that an equal probability distribution of being in any state is achieved in that temperature. With a list, approaching the equal probability distribution is extremely slow.

SA apparently works best in a situation where there are areas of the state space with several "rather good" solutions rather close to each other (i.e. with topology). This way, by sampling the space the system may proceed to areas with more good samples (this is achieved by approximately reaching the thermal equilibrium) and look at them closer, i.e. in a lower temperature.

SA differs from gradient-descent optimization procedures in that it does not necessarily get stuck in a local minimum. The SA method, if applied by the theory, i.e. if annealing is performed slowly enough, is guaranteed to find the globally optimal solution. According to Geman and Geman (1984) as cited in (Haykin, 1994), with sufficiently large T_0, if the following applies for every k,

$$T_k \geq \frac{T_0}{log\,(1+k)}, \tag{2}$$

the system will converge to miminum energy configuration with probability one. Unfortunately, this cooling schedule is too slow to be used in any practical system. However, practical approximations of the schedule often produce near-optimal solutions to the given problem, especially when there are many such solutions.

Applications of simulated annealing to layout generation. Simulated annealing has previously been applied to minimization of layout area and wire length on VLSI layout design (Sugai and Hirata, 1991), and to machine layout problems in manufacturing where transportation of production resources between different machines must be minimized (Kouvelis et al., 1992).

5 Case studies

The following case studies provide a basis for comparison of three specific approaches used to automatically solve the layout problem of the generalized newspaper (falling under layout problem type 3, see 4.1.3). The practical and concrete example problem used was the pagination of a newspaper delivered on fax.

An instance of the graph-theoretic approach was mainly studied within a previously existing system, VTT's V·I·P^E, with some modifications added by the author of this work. Simulated annealing was implemented twice with different problem formulations. The first was a rather straightforward implementation with intuitive formulations of state space and state transitions. In the second experiment, better knowledge of the application domain and SA in general had been acquired, and the problem formulation respectively improved. In the latter two studies only the pagination module was implemented, since the purpose was to study different methods for automatic pagination, not to produce a full system for production of newspaper delivered on fax.

5.1 Problem description

The pagination problem was defined as follows:

1. All produced layouts must be a) as full as possible, and b) filled with as high-priority material as possible.
2. There is more available material than what eventually ends up on the layout (i.e. selection of material is necessary).

Thus, the two variations of this pagination problem can be described in terms of knapsack problems as follows:

- The number of bins (pages), is fixed, but there is an abundance of items (story material) to choose from. The goal is to minimize empty space and maximize priority value of the material within the bins.
- The items are fixed and any number of bins can be used. The goal is to minimize number of bins (pages) required for the whole material.

5.2 Selected approaches

The heuristic graph search was chosen for study and improvement, because a mostly working prototype was already implemented in V·I·P^E. Thus it offered a basis for comparison with the other two examples. The approach of the prototype was also somewhat improved.

Simulated annealing (SA) was selected as an alternative approach to pagination. SA has been successful in other kinds of constraint satisfaction problems where a cost function must be minimized. SA was also tempting, because the heuristic approach seemed to have some serious limitations. Specifically the difficulty of guiding the search and designing and implementing new and better heuristics was considered difficult. In SA the goals may be coded as objectives in the cost function, and the balance between different goals is easily controllable.

Genetic algorithms (GA) were also considered. In literature discussing the differences between SA and GA it has been suggested that it would be wise to first try SA, and only if it doesn't give satisfactory results, the more complex GA should be applied. Thus, only SA was implemented in this work.

During the work, several different SA-implementations were attempted and discarded since they were too slow. Here two cases will be described, the first of which did not prove fast enough for practical use, while the second succeeded well. In addition to being experiments of pagination of editorial material, these cases demonstrate the difficulties of SA implementation, and hopefully offer insights on how to represent a problem successfully for SA.

5.3 Fax newspaper prototype

(Veijonen et al., 1994) describe the prototype V·I·P^E designed at VTT for automatic page layout of electronic news distribution. Pagination methods applied were derived from advertisement pagination system previously designed at VTT (Ylä-Jääski and Ahonen, 1990).

5.3.1 VTT's V·I·P^E

The initial goal of V·I·P^E pagination had been defined as pagination of articles with all kinds of sizes and rectangular shapes. However, the requirement of any article sizes had been relaxed and problem space considerably discretized to reduce it. Instead of allowing any article sizes starting from any location on the page, a sparse grid (typically 5×5 grid for an A4 page) had been defined and article sizes were accordingly modularized. Division into columns was natural, since in a typical newspaper the text usually resides in columns of equal width. The horizontal divisions, usually into 4-6 horizontal sections, were artificial (see Fig. 3). They had been adopted not only to discretize the search space but also to ensure tranquility of the layouts. The resulting pages indeed were tranquil,

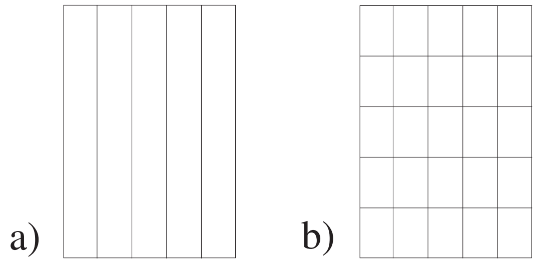


Figure 3: *a) A layout with five columns. b) A modularized layout with 5x5 modules.*

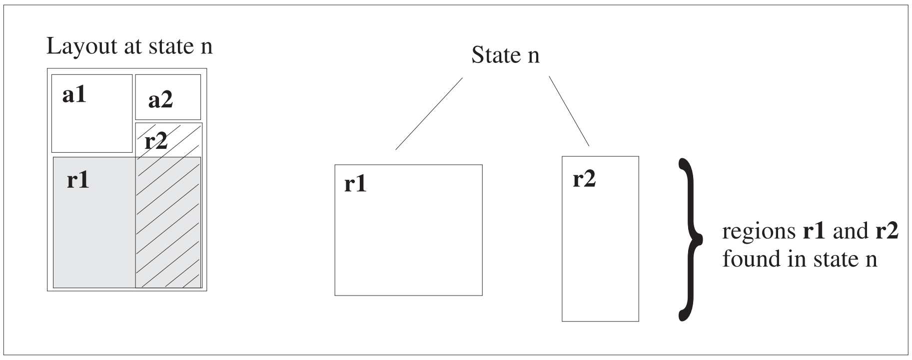


Figure 4: *a) An intermediate page layout of four columns, with two articles* a1 *and* a2 *already on the page. Two maximal regions,* r1 *and* r2 *can be identified on the layout, based on the article end lines. b) The same situation represented by a graph.*

often even dull, and occasionally resulted in quite a lot of wasted space.

The contents of each article were first paginated into several different modularized shapes—*article forms*—by rough article layouting. All these alternative forms were then passed on to pagination, which selected at most one of them to a page. Due to the modularization, the form often contained also empty space. The grid thus had three effects: it reduced state space, ensured that the pages would not look too irregular and at the same time compromized some empty space.

The pagination engine had been rather straightforwardly borrowed from a version of V·I·P paginating newspaper display advertisements. The pagination method might be described as "heuristic best-first forward search utilizing branch-and-bound technique". The algorithm proceeds by searching through a graph which is expanded at time of need to selected directions. In the graph, leafs represent solutions proposed by the algorithm, i.e. finished layouts where nothing else can be added. Inner nodes represent states between insertions of new articles. The empty space on a page is represented by *empty regions* which are maximal rectangular areas of empty space, calculated from the end lines of articles on the page (see Fig. 4). In the state graph, transition from a father to a child carries out the following operations:

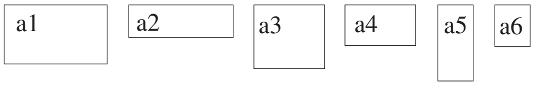


Figure 5: *A sample selection of articles ready to be paginated onto the layout depicted in Fig. 4. The articles have been ordered with a heuristic rule: first into descending order by width, and next the articles with the same number of columns into descending order by height.*

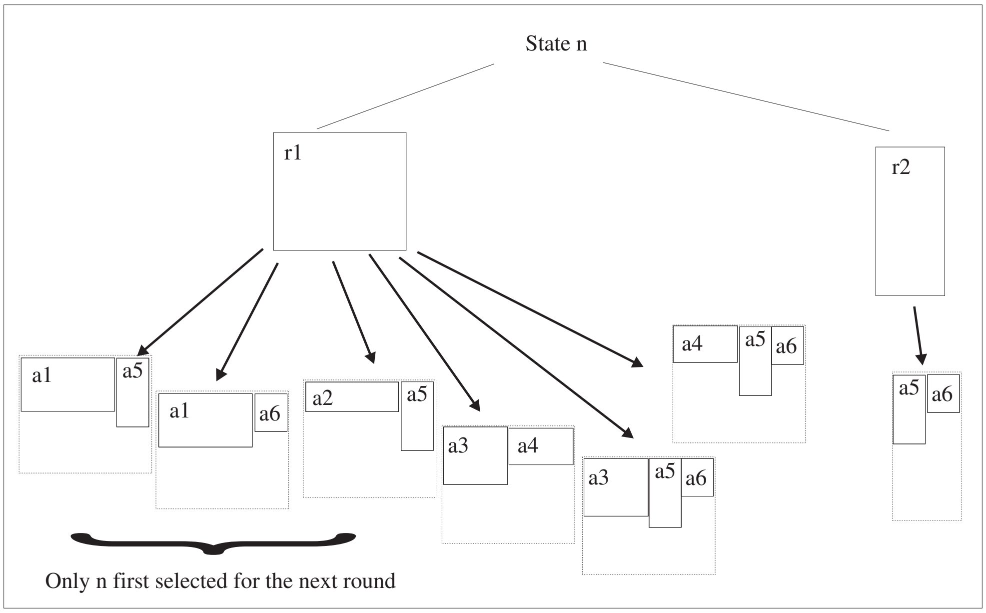


Figure 6: *The region-box combinations created in state n. To cut down the search tree, an upper limit has been defined for the amount of article combinations to create. The branching is done mainly in the beginning, and at later stages only one branch is typically created and proceeded.*

1. Out of all the regions on page, select N "best" regions where to paginate (N is a small integer, perhaps 1).

2. For each region: out of the available article shapes (for example, see fig. 5), calculate all the possible combinations of article boxes that would be equal or below the available region's width (Fig. 6). Select N "best" combinations (with N being again a small integer, even 1).

Next, a number up to a dynamically decided upper limit of region-box- combinations will be expanded to create new nodes of the search graph by inserting the box combinations to the regions (see Fig. 6). Each node produced by this method is expanded further until no more articles can be added on the page. Finally, the best page according to given criteria is selected.

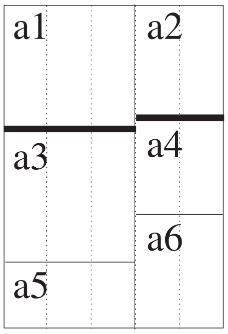


Figure 7: *An example of a page where two horizontal lines (the thick ones) defined by article borders are disturbingly close to each other.*

The heuristics of the method reside in the ordering of empty regions and article combinations supposedly into best-first order. Heuristics often help, but since available material greatly affects the task, it is possible that also good paths are cut out by the control mechanisms that reduce the state space by cutting the number of paths examined. When there is little material to start with, the loss of paths may result in pages with too much empty space, even when an acceptable solution exists. Developing a good set of heuristic rules that works well with different kinds of input data is a tedious and time-consuming task.

5.3.2 Extensions to V·I·P^E

The V·I·P^E prototype described in 5.3.1 was extended and developed further:

- Pagination of blocks with modularized article sizes was replaced with pagination of variable-sized blocks.
- Some user control was added to make it possible to adjust the direction and results of the pagination.

Pagination of variable-sized blocks. In this experiment, the modularized article forms and grid were replaced with variable article sizes and free location on the page. After the modifications, wasted space on the page decreased. Unfortunately, pages sometimes became uglier, because now the horizontal lines on the page could start very close to each other (see Fig. 7 for an example).

Therefore, to restore some of the lost tranquility of the pages, the positions of horizontal article borders needed to be controlled in another way. This was carried out by implementing a strict rule in the selection of possible article locations (regions), so that solutions with borders starting too close to each other were abandoned.

Adding user control to the pagination results. It is an interesting question how much control the future reader wants to have on the content, and on the visual presentation, or whether he/she prefers to only receive a pre-designed thought-out publication with no individual control on its design. It is probable that there exist readers at both ends of the spectrum. Furthermore, different kinds of publications or situations where information must be presented visually, may call for different amounts of control over the results. The following discussion on user control of the results is applicable both to a situation where the pagination results are controlled by the publisher, and to a situation when the pagination takes place individually for each user, guided by his/her choices.

Giving user more control over the results of the pagination is a two-edged sword. On one hand, the user often wants to control the results (if he/she cares enough to spend time and think about it; sometimes the default result is fine). On the other hand, if the user is given several switches to turn, he/she expects the system to change its behavior in certain ways in return. Actually, he/she has some representation of what the range of generally possible results of the pagination process might be, and would like the system to change within this range, depending in some logical way on the choices he/she makes. It is also obvious that no combination of user choices should result in malfunctioning or total failure.

All this, however, is not easily obtainable in a complex system depending heavily on the given data. To fully attain the goals, the following would be needed: 1. A behavioral model of the input-output-relation of the process itself with any given parameter combination and the usual data, and 2. A mental model of the expectations of the user.

The overall goal is to translate the mental model and expectations of the user into effects on the system behavior via given input lines. This may be achieved in one of three ways: either by the designer's thorough understanding of both the user and the system in advance, or by the user learning the system behavior and its response to different parametric alterations, or by the system learning about the user's mental model and adjusting to it.

The last case is the most interesting, since it is both user-friendly and does not depend as much on the system designer's guesses about the user's mental models. However, it was beyond the scope of this work to try to define a proper mental model of the user, or to design a learning control interface. However, some control capabilities that seemed useful were selected and implemented into

V·I·P^E. The emphasis was on studying the existing pagination system and on enabling control in the first place by finding out how and where to guide its rather implicit pagination heuristics.

Implemented controls. Three controls were implemented with which the user was able to guide the selection of the best layout. This kind of controls might be used in a traditional newspaper by the publisher to give an individual flavor to the publicatioon. Giving the user means to decide on strict rules about the pagination results would not have been wise, since sometimes no allowed solution might be found. Thus, the controls were of the nature "weigh this optimization goal more than that" or "completely disregard this goal". The user's wishes were taken into account when selecting between a group of rather good layouts. The implemented controls let the user specify values for the following parameters:

- **The number of horizontal and vertical lines.** This constraint affects the tranquility and variation of the page. Two articles that start at the same row form a single horizontal line with their upper boundaries. The user may suggest upper and lower limits for the numbers of horizontal and vertical lines on the page.
- **The importance of minimizing empty space vs. maximizing article priorities defined by the editors.** The user has some control over whether it is more important to e.g. minimize space or to pay attention to the article priorities which have been defined by the editor or publisher of the information. The results of different selections of importance of priorities can be seen in Fig. 8, where the same material has been paginated twice with different user control selections. With the page on the left, the user decided that priorities are not very important, but minimization of space is. The opposite (i.e. priorities are important) was selected next, giving the page on the right, which displays more wasted space but a better priority value of the content than the left one.
- **The cost of deviating from the desired number of articles on a page.** The user may specify the desired number of articles on a page, and a cost is associated for deviations from this number. The cost grows in square of the deviation from the preferred number of articles, i.e. $w(x-m)^2$, where w is the weight defined by the user, m is the user-defined desired number and x the actual number of articles on the page. The control does not guarantee reaching the desired number.

The effects of these selections are most profound when the user only wants one of them to work. In the test runs, if all goals were simultaneously attempted,

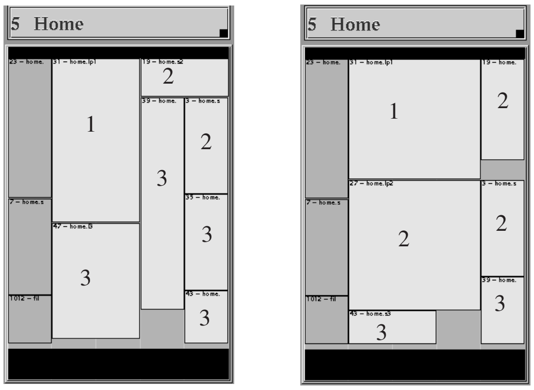


Figure 8: *Example of effects of user control on whether to weigh minimization of empty space (on the left) or maximization of priority value (on the right). Priorities are marked with numbers 1–3 in the article boxes, highest priority being 1.*

mediocre results were achieved. This follows a general rule of thumb in optimization: one has to choose what is important and what is not.

5.4 Simulated annealing, experiment 1: straightforward implementation

The theory of SA leaves plenty of implementation details unspecified, since due to time constraints SA can never be applied in the form for which the theory guarantees optimal behavior. The numerous experiments with SA have however produced useful implementation knowledge. Hints exist for choosing several control parameters affecting the algorithm (Aarts and Korst, 1989). Still, some of these hints are rather arbitrary, as are the guidelines on how to choose the state space and the allowed transitions. Success in formulating the problem to best suit to SA requires some experience and understanding of the algorithm.

The following problem formulation is a final stage in a series of tests with slightly different state spaces and transitions as well as values for the control parameters. The two tasks of pagination—selection of material and creation of the layout—were separated in this approach. First, it was attempted to select "good" initial material consisting of boxes with fixed size, shape and contents (such as advertisements or editorial articles already laid out to some shape) and then annealing was used to find a layout where these boxes would not overlap. Naturally, in this scheme a solution may not be even reachable with the selected boxes. Regardless, this choice was made in order to reduce the state space for annealing, since the annealing approach selected turned out to be so slow[5].

5.4.1 Selection of material—variable-base adder

The first task was to select those article forms[6] that would participate in the annealing of the page layout. A simple but effective scheme was needed for generation and selection of suitable article combinations, since it definitely was not desirable to go through the time-consuming annealing process with article combinations that were known to be unfitting or otherwise poor. A *generate-and-test* approach was needed. That is, we wanted to

1. *generate* prospective combinations of article versions in some kind of best-first order, and
2. *test* these combinations to outrule some combinations that were known not to produce an acceptable solution to the problem.

Generation of prospective combinations. For the generation of prospective solutions, a method following the metaphor of an adder for numbers with variable base was devised. Within this metaphor, each combination of digits (i.e. a number like 5366) represented one combination of article versions (version 5 of article 1, version 3 of article 2, version 6 of articles 3 and 4).

A binary number system could therefore represent articles where each has two versions. The decimal number system accordingly would represent articles with ten different versions each. However, since the available articles had different number of versions (minimun two, with one actual version and the other meaning that the article was not included on the page at all), the base of the system needed to vary according to the number of versions that each article had, i.e. the base was individual for each digit position. See Table 2 for an example. By going through each number representable with this number system, all the possible article version combinations are produced. This may be carried out by starting with number zero and incrementing it one at a time. Therefore, this generation method was called a *variable-base adder*.

[5]In some test runs, selection of articles was performed during annealing by adding and deleting articles from the page, but these approaches failed to produce results at all in any sensible annealing times, and were thus discarded to reduce the problem complexity.

[6]An *article form* refers to a version of an article that has a fixed size, shape and contents. One article may have many versions of the contents, and each content version still further versions of its shape. In the context of pagination, two of these three concepts are relevant: the article with neither fixed content nor fixed shape, and the fixed box with fixed contents. In the following discussions, both "article version" and "article form" are used to refer to the latter concept.

articles and their versions				
version number	article 1	article 2	article 3	article 4
version 1	size = 6	size = 9	size = 3	size = 7
version 2	size = 3	size = 4	size = 0	size = 2
version 3	size = 0	size = 2		size = 1
version 4		size = 0		size = 0
version 5				
Number of article versions	3	4	2	4
Base of this digit position	3	4	2	4

Table 2: *Four articles with varying number of versions and their correspondence to a number system with individually defined base for each digit position. Each article is represented in one digit position. The base in this position is the number of versions the article has. The versions of the article correspond to digits from zero to base* $-$ 1. *In this table, a version is characterized by its size. Zero size means that the article does not appear on the page at all.*

The best-first order of prospective solutions was achieved by assigning the articles to digit positions in the order of their priority, and arranging the article versions in the order of size from large to small. This ordering tends to maximize the amount of any higher prority material on page before descending to material of lower priorities, and produces an order roughly from best to worst in terms of the total priority of the material on the page, when the priority of a page is defined as follows:

$$\text{Priority of page} = \sum_i \text{area}(i) \times \text{priority}(i), \tag{3}$$

where i goes through the article versions on the page.

In fact, this scheme also describes a depth-first tree-traversing algorithm. The depth-first search tree produced by the above described ordering of the articles described in Table 2 is shown in Fig. 9.

Testing and discarding unsuitable combinations. After defining the best-first generation mechanism for article selections, a method was needed for discarding selections that were known not to produce an acceptable result. The most important restriction was that the material has to fit on the page. Two other restrictions were adopted as well; one stating the minimum allowed area that is occupied by the material on page, and another stating the minimum acceptable

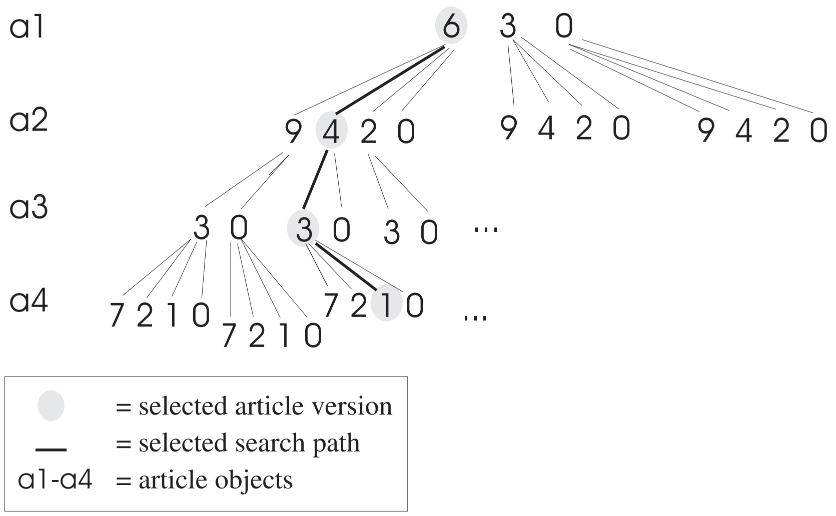


Figure 9: *A search tree representing the order in which the solutions are produced for the articles described in Table 2. The nodes of the tree contain the required space of that article version. Zero space means that this article does not appear on the page at all. Each solution is represented by a path from root to one of the leaf nodes.*

priority value for the whole page.

Ordering of the material by area and priority caused the clearly unsuitable solutions to appear neatly within the same branches of the search tree. This meant that when one unsuitable solution was found, it was unnecessary to examine that branch further. An example of cutting a branch due to exceeding the space on a page is shown in Fig. 10. To be exact, since the ordering produced is not strictly best-first but only close to it, some branches with possibly suitable solutions may also be cut. However, considering the reasons behind the best-first ordering, it is enough to produce a sufficiently large amount of possible solutions that pass the test.

Yet another integrity check was used to reject unsuitable combinations. Two articles that were wider than half of the page width, could not fit side to side on the page. Therefore, they should be stacked on top of each other, and thus their summed height should not exceed the page height. After this rule was added to the rejection criteria, the approach actually resembled some more developed bin-packing algorithms, although the problem they are used to solve is different.

5.4.2 Creating the layout by simulated annealing

Formulation of the problem at hand into a form suitable for the application of SA may be approached either by first determining some naturally appealing transi-

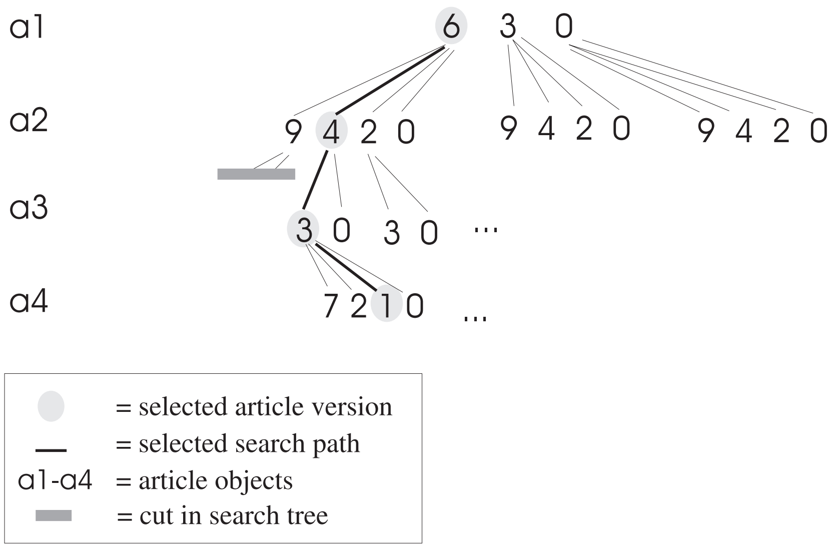


Figure 10: *The previously shown search tree, now with a cut. The size of the page was defined as 16 units, whereas the lower limit for the area of the material was set into 14 units.*

tions, or by defining the states first and then determining a group of appropriate transitions. Naturally, the transitions should only lead to states that were defined as part of the state space, and reach all of them. In addition, since at each temperature the system should reach the thermal equilibrium in a reasonable time, it is suggested that there should not be too long minimum paths between pairs of states in the state–transition graph (noted e.g. in Aarts and Korst, 1989). The longer the minimum paths, the longer time (i.e. longer Markov chains) it takes to reach the thermal equilibrium. However, no quantitative advice is given in literature on how to decide what constitutes a "too long minimum path between two states".

State space. The state space should naturally include at least all the states representing acceptable solutions. These are layouts with no articles crossing page borders or overlapping with each other. As these states cannot be characterized before solving the problem, also other kinds of states had to be included in the state space. Practice has also shown that it is useful to have intermediate states representing pseudo-solutions (Aarts and Korst, 1989). Still, no advice is given on *how* to extend the state space, i.e.how to decide which conditions of the optimal solution should be relaxed.

It was decided to include extra states by allowing article overlappings during annealing, and thus the objective of no overlappings was coded into the cost funtion. Crossings of page borders, on the other hand, were not allowed even during the annealing. This kind of approach was also suggested by Aarts and Korst (1989), pp. 86–88, for a rather similar problem.

State transitions. The state transitions were chosen so that an article might at one transition move 1–30 rows within the same column or, alternatively move one column to either direction on the same row, with equal probabilities. Thus there were 32 possible transitions from any state.

Annealing schedule. Annealing schedule was chosen so that in the initial temperature almost all attempts (over 90%) were accepted. The end temperature was chosen so that less than 10% of the lowest-energy changes were still accepted.

The temperature T was decreased according to the exponential scheme $T(t+1) = cT(t)$, where t and $t+1$ are consecutive rounds in the annealing process. The parameter c affecting the speed of annealing was chosen make the process go from start temperature to the end temperature in 500 000 rounds. It was possible to calculate these temperatures beforehand from the types of possible state transitions.

Cost function. Since in this experiment the article forms, priorities and contents were decided already during selection of the material, only the positions of the articles could contribute to the cost of the page. Avoiding overlaps in the final results was the most important goal to reach for. The total energy of the page therefore consisted of calculating the overlappings:

$$E = \sum_{i,j,i \neq j} \text{overlap area}(i,j), \tag{4}$$

where i and j represent articles on the page. The energy change associated with a transition thus becomes

$$\Delta E = \sum_{j,j \neq new} \text{overlap area}(new,j) - \sum_{j,j \neq old} \text{overlap area}(old,j), \tag{5}$$

where *new* is the new article, i.e. the article in the new position, and *old* respectively the old position before suggested change.

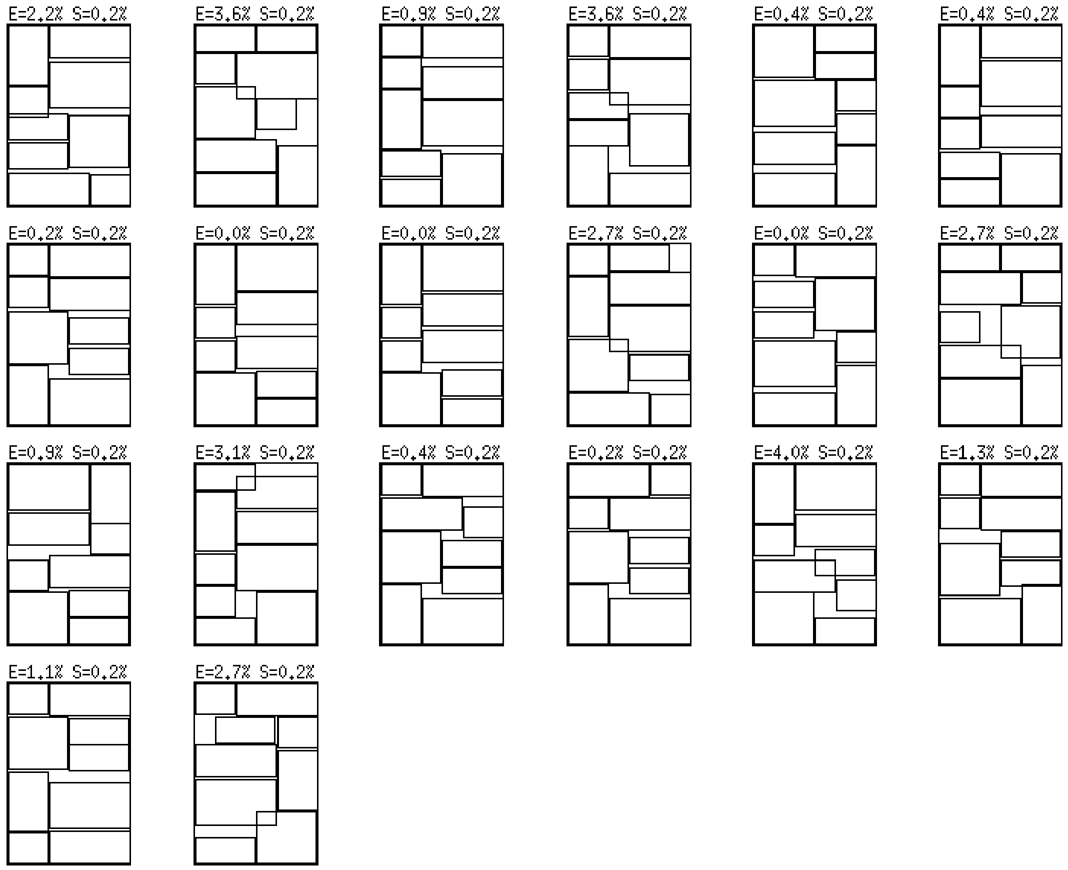


Figure 11: *Annealing results of one article configuration on 20 rounds. The configurations with only slight overlap may be considered acceptable, as they may be adjusted to fit to the page with fine-tuning techniques.*

5.4.3 Results

In some cases the optimal solution was visited upon during annealing, but in such a high temperature that the system continues to look for better solutions. As a minor optimization, the best solution found so far was saved and in the end compared with the final solution — the better of the two was taken as the output of the annealing.

Fig. 11 depicts 20 annealing runs made with one article selection. In many of the runs, a fully acceptable solution (i.e. one with zero overlappings) was found. Many runs also produced results that would not be useful pagination results even with article layout methods.

Another test run was made with 18 different article configurations and is shown in Fig. 12. In most cases, a solution with only slight overlap was found, but there

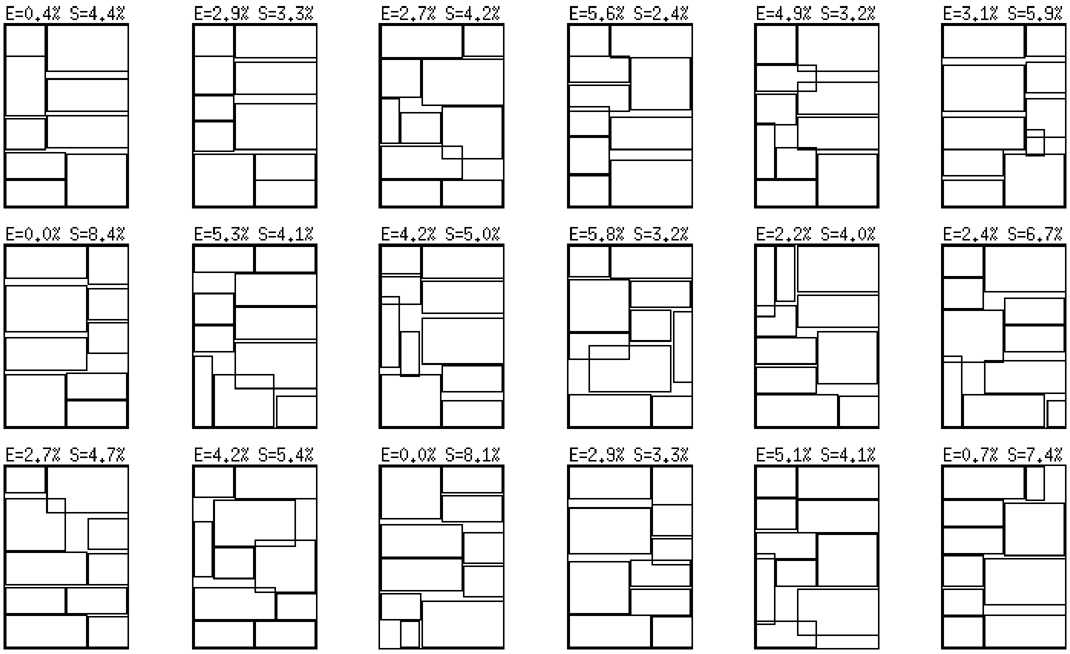


Figure 12: *Annealing results for 18 different article configurations. Each article selection was produced with the variable-base adder mechanism. Annealing was then applied to the selection. E is the amount of overlap, which was used as the energy of the page. S is the percentage of the extra space on the page that would be there if the annealing of this selection was perfect. The total value of the page is $S + E$.*

were very few results totally without overlap. It should be noted that in many of the examples a completely overlap-free solution may not even exist, despite the attempts to minimize this possibility during article selection.

It is worth noting that slight overlaps in the final result are acceptable if fine-tuning methods (such as described by Kukkonen, 1992) are available for final adjustments. With these methods small overlappings as well as small empty spaces may be deleted by re-arranging the preliminary article layouts. In both Fig. 12 and Fig. 11 the layouts with less than 2% of overlap seem rather acceptable, and as this small difference amounts to about one text line of overlap, it may be taken care of by other means. Thus, from the results with article selection in Fig. 11, it can be seen that 6 out of 20 runs produced an acceptable solution.

Lessons learned from the approach. Although this method is not a good choice for exactly this problem, some useful components and insights were definitely gained. In retrospect, if selection of article forms is carried out independently of forming the layout, pure bin-packing methods may perform better than simulated annealing. At least it is obvious that some mutually overlapping work was now carried out by the selection and the layout creation processes, since already in the selection it was noticed that articles wider than half the page could not fit side by side. However, it was not obvious how to code this understanding to SA without changing the annealing into something else or disturbing its own internal mechanisms too much.

The problem of the approach may be crystallized as follows: reaching thermal equilibrium takes too long a time, i.e. the probability distribution spreads slowly in terms of steps taken. The three factors affecting the propagation of the probability distribution are the size and connectivity properties of the state space, and the energy jumps between states.

It seems that the major problem with the state space formulation used in SA1 was that individual state transitions were *too local* compared to the size of the state space. It would have been better either to allow longer jumps on one move, so that the probability of going over an energy hill would have been higher. It might have been wise to allow moving an article from side to side, or even exchanging two articles instead of moving just one. The same effect would have been achieved by discretizing the state space further[7].

Admittedly, the previous tentative analysis of the dynamics of the state space is rather intuitive, and an exact mathemathical formulation and analysis as well as practical tests would be required to confirm the analysis.

5.5 Simulated annealing, experiment 2: utilizing domain knowledge

In the second experiment, the motivation was to solve the problem of fax newspaper better than in the first experiment. The problem of the previous approach was that finding a good layout might take several annealing rounds, and this was considered far too slow for the fax newspaper application considered. Thus it was deemed essential to perform some of the selection of material also within the algorithm building the layout. The ensuing increase in problem size had to be countered by redefining the problem formulation, especially the state space. To counter the increase in problem size and to ensure that overlaps would not occur in the resulting layouts, overlapping during annealing was not allowed at all.

[7] On the layout there are a maximum of $cols \times rows$ different starting locations for any article. For a selection with n articles there exist therefore $n^{cols \times rows}$ different possible layouts, if states with overlappings are included. In our typical layout these values were $n = 10$, $rows = 150$ and $cols = 6$ and so the size was about 10^{900} states. Even a lower limit for the size, taking into consideration that page borders cannot be crossed, would be at least in the order of 10^{150}.

State transitions. By analyzing the problem, there appeared to be four natural kinds of state transitions. The following table contains each transition and its approximate contribution to the cost made up of empty space:

Transition	Contribution to wasted space cost
1. Add article	Large decrease
2. Delete article	Large increase
3. Move article	No change
4. Replace article	Small increase or decrease

The transitions move and replace are actually combinations of add and delete. The problem with move is that although it does contribute to the value of the page layout in the long term, no cost change can easily be associated with the change when minimizing only empty space or maximizing priority value of material on a page. All moves would then have zero effect on the cost and would therefore be accepted at any time of the annealing, and the system could for a long time alternate between two equivalent states. In fact, this happened with the previous experiment, SA1.

Delete was also considered problematic, because it introduces such a large addition to the page cost states, and actually is just a leading step in an actual change. This means that if transitions delete and add are used, the actually interesting transitions necessarily have a high mountain peak to climb over on the delete transition. Therefore, at lower temperatures no deletions would be accepted. This dilemma led to consideration of the transition replace, and the following idea emerged: add was combined with replace to produce the transition replace if overlap, which works as follows:

1. Randomly select an article not yet on the page.
2. Randomly select a "corner" of empty space for it.
3. Collect overlapping articles for deletion.
4. Accept (add article and delete colliding ones) or reject change on the basis of the SA test, which depends on current temperature and cost of the total change.

In each transition, one article is added to the page and zero or more articles are removed. See Fig. 13 for an example of a transition where a replace occurs due to a collision.

In the selection of the suggested position, the V·I·P empty regions -mechanism was used (see Fig. 14). In the region formulation, the empty space on a page is represented as a group of rectangular, potentially overlapping regions, defined by the borders of existing articles on the page. Each region is maximal, i.e. it

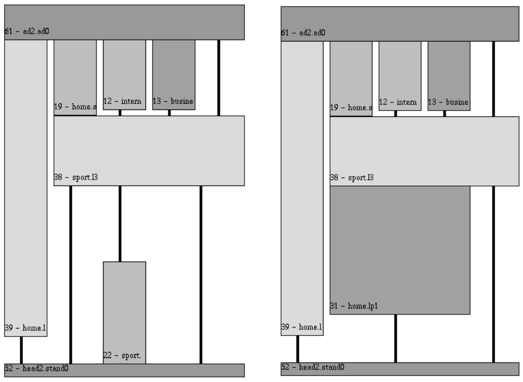


Figure 13: *A* replace-if-overlap *transition during the annealing for a 5-column page, with snapshots of the page before and after the transition. The shaded boxes are articles, the shades of gray showing the priority values (dark has high priority). The thick black lines stand for gaps, i.e. contiguous empty spaces where articles could be added. On the left is the situation before an accepted transition. On the right side is the same page after adding article 31 into the upper left corner of an empty region. The addition resulted in an overlap with article 22, which was therefore thrown out.*

could not be extended horizontally or vertically. All empty space on the page is covered by at least one region.

Annealing schedule. In the average about 1500 rounds (i.e. suggested moves) were used in annealing, with a final period of 30 rounds in zero temperature to further ensure being in the bottom of a minimum.

Cost function. Since the goal was to maximize the amount of high-priority material on the page and minimize empty space, both of these factors were used

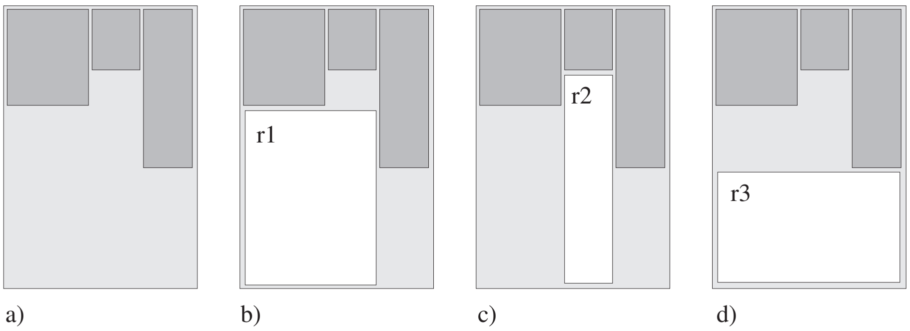


Figure 14: *a) A sample page layout in the middle of pagination, with three articles placed. b), c) and d) The three maximal empty regions, r1, r2 and r3 that can be found from the layout.*

in the cost function. The energy of a whole page was therefore calculated with the following equation:

$$E = U(A - \sum_i \mathrm{area}(i)) + V \sum_i \mathrm{priority}(i) \times \mathrm{area}(i), \tag{6}$$

where i denotes an article, A is the total available space on page, and U and V are weights used to set the balance between minimizing empty space and maximizing priority value, respectively. The first term of the sum calculates the empty space on a page, the second producing the priority value of the page, 1 being the best priority and 10 the worst. Thus also the second term must be minimized to maximize the actual value of the page. Therefore, the energy change pertaining to a single transition is given by

$$\begin{aligned} \Delta E = \; & U(\textstyle\sum_{old} \mathrm{area}(old) - \mathrm{area}(new)) + \\ & V(\textstyle\sum_{old} \mathrm{priority}(old) \times \mathrm{area}(old) - \mathrm{priority}(new) \times \mathrm{area}(new)), \end{aligned} \tag{7}$$

where *old* and *new* denote the outgoing and incoming articles, respectively. Since minimization of empty space was considered important, it was weighted relatively high in the cost function.

Analysis of the approach. As mentioned before, the layout problem is NP-complete. To gain a more exact idea about the problem complexity, some analysis of the size of the state space was carried out.

The calculation of the number of possible article combinations on the page may be divided into two parts: selection of articles for the page, and selection of the exact version, article form, for each article on the page. For the sake of simplicity it may be assumed that each article has the same amount of versions. Thus the number of possible combinations for a fixed number of articles on the page is:

$$C(m, n, N) = m^n \binom{N}{n}, \tag{8}$$

where m stands for the number of article versions per article, n is the number of articles selected on the page and N is the total number of articles to choose from. Since the number of articles on the page may vary depending on the available material, the total number of combinations is given by summing the counts produced by each number of articles that may reside on the page, and the equation thus becomes

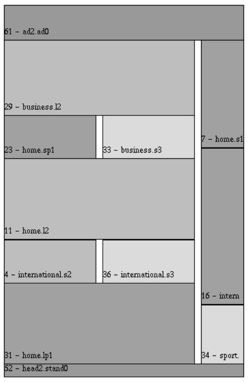


Figure 15: *A good result especially in terms of wasted space.*

$$C(m, n, N) = \sum_{n=n_{min}}^{n_{max}} m^n \begin{pmatrix} N \\ n \end{pmatrix}. \tag{9}$$

As noted before, this describes only the number of different possible article combinations. The total number of states would be the number of article combinations times the number of possible layouts per combination. Calculating all the different possible layouts that an article combination produces is practically impossible, as it depends heavily on the available data. In addition, most the available state space may actually never be traversed, and it is virtually impossible to find out the actual probability distribution of visiting different areas of the state space. This analysis of the size of the state space was nevertheless provided, since it may be illuminating in comparison with the approach of SA1.

5.5.1 Results

The implementation was tested with the same material as in the previous two experiments. Fig. 15 depicts a full page obtained in one run of the algorithm, a result of quality which never occurred when experimenting with the SA1 or V·I·P^E methods with the same material.

The results obtained with SA2 were compared to those obtained with pure gradient-descent algorithm. Pure gradient-descent equals to running SA with temperature zero, i.e. accepting only transitions leading downward. As may be seen from Fig. 16, even applying gradient-descent in this problem formulation produces results that are not completely unacceptable. However, SA on the right of Fig. 16 performs clearly better. This comparison suggests that while SA performs very well with these few goals, after adding goals it still would perform rather well, whereas gradient-descent would probably do poorly.

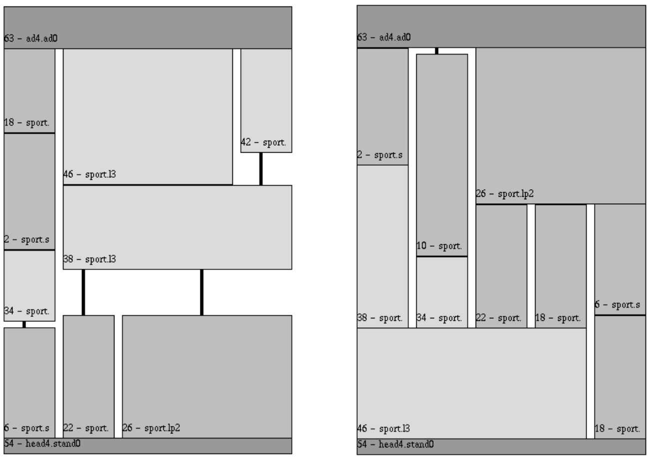


Figure 16: *Two sample layouts produced out of the same material a) with pure gradient-descent and b) with SA.*

5.6 Comparison of approaches

The case studies lead to the following characterizations of each approach:

V·I·P^E. The heuristically guided and bounded search of V·I·P^E is a rather good method, although it has its shortcomings. It is fast enough for the problem definition of the case studies: pagination of a newspaper delivered on fax. However, controlling or changing the heuristics proved rather difficult within the design of V·I·P^E. The design of a system guided with heuristic rules that can be easily replaced or continuously adjusted is not a simple task, although it would be a useful feature for an automatic pagination application. If the search heuristics can be clearly separated from the implementation, this approach might be advantageous.

SA1. In the first SA experiment, the definition of the state space resulted in awkwardly long annealing times for a fax newspaper. Still, considering the size of the state space, the annealing times were understandable. If the problem had been to paginate blocks of fixed size and shape, this approach would have been acceptable. After all, the produced layouts were quite good. In this experiment the selection of material and generation of layout were separated, which may be a desirable characteristic in some problem instances.

SA2. In the second application of SA, the design of the state space and transitions was more suitable to the given problem, as described in 5.1. The transition replace if overlap achieved in reducing the local "raggedness" (discontinuity of ΔE) of the search space, which enabled considerable improvements of state in the lower temperatures and even in a frozen state.

As compared with SA1, the state space was extended to cover a larger total colume of the solution space, but simultaneously decreased by cutting out a large amount of intermediate states that could not contain solutions—the states with overlapping articles. With a careful selection of the state transitions, a state space with suitable connectivity and shape of the cost function was created. Compared with SA1, *the frequency of good solutions was also increased.* Given this it is not surprising that the results of SA2 were most promising. The approach has the advantage of being fast (as had the V·I·P^E experiment), and it enables continuous continuous control of the objectives of the search via the cost function.

Despite the considerable improvement in speed of the pagination, the SA2 implementation probably would not be quick enough for pagination to take place as part of a user interface of an electronic newspaper, which is generated and viewed interactively. However, since the speed of SA2 was comparable to that of V·I·P^E, also SA2 can be used in a newspaper delivered on fax or some other application without dritical efficiency requirements paginating into fixed space from large storage.

It must be emphasized that both SA1 and SA2 were implemented to solve the general layout problem of type 3 (see section 4.1.3), for building the layout with a fixed area and varying material. V·I·P^E, on the other hand, was a prototype specifically designed for pagination of editorial material.

6 Conclusions and future directions

In this work, we have studied automatic pagination in context of electronic publishing and pagination of editorial material. In particular, two implementations of simulated annealing and one based on heuristic search were examined for automatic pagination of editorial material of a newspaper delivered on fax. Based on the experiments and their analysis, some guidelines were found on how to approach this kind of a pagination problem, and how to apply simulated annealing to the layout problem.

6.1 Automatic pagination and electronic publishing

An automatic pagination method for editorial material clearly seems to be necessary for on-line publishing where interactive pagination is too slow or expensive. The desired quality of the results naturally determines how heavy an approach should be used. If the minimization of empty space is not as important as just roughly deciding which articles to put on which page, and if a rough layout is sufficient, a simple heuristic method is probably the wisest choice. The same holds if the pagination must be carried out run-time as part of a user interface on computer screen. In fax newspapers, where the criterion of empty space minimization is stricter and more time may be spent per page, the simulated annealing approach of the last case study, SA2, might be used, enhanced with some of the advanced features of the V·I·P^E. The future versions of SA2 might include at least controlling the distances between horizontal article borders and implementation of the user controllike implemented in V·I·P^E. The implementation of such goals and constraints in the cost function is straightforward, and the balance between different goals is easily adjustable, unlike in the heuristic search approach, where each heuristic must be implemented and carefully tested to ensure that goal states can still be found.

6.2 Simulated annealing

SA is not a magic answer to problems although it may seem such, and although it is often presented with a touch of mysterious glory. Since SA has no memory, and since not even a nearly full search can be performed during the algorithm, there must exist another way within its operation to help it find its way. This is what the design of the state space and transitions achieves, and with which SA rises and falls. In some problems the original intuitive problem formulation is naturally suitable for SA, but in others the problem should be translated into a suitable representation, requiring good understanding of both the implementation area and SA itself.

Although the theory of simulated annealing does not pose any constraints or restrictions on the shape of the state space, in practice some state spaces do behave much better than others during annealing. For annealing to function properly, either the system must reach thermal equilibrium at each temperature and the temperatures must be lowered slowly enough, as described by the theory of SA, or the energy function of the state space must be locally rather smooth. The latter condition depends on the problem formulation, in particular on the states and the transitions. The first condition can almost never be satisfied adequately, and therefore the problem representation is essential. To be able to transform any problem into one suitable for SA, a thorough understanding of the application area is important. The importance of the appropriate problem formulation does not seem to be adequately stressed in the literature on application of SA.

In future, theoretical work is needed on SA to define in exact mathematical terms what properties does a successful implementation of SA have. So far the situation has been somewhat amusing: theoretical grounds that determine the success have indeed been defined but the conditions are impossible to fulfill in practice due to time constraints. Therefore there is no general awareness of why the current implementations actually do work despite that they do not follow the theoretical limits.

Empirical guidelines (if not even mathemathical analysis) could perhaps be defined for the relationship between annealing times and the ratio of search space vs. the longest transition paths required to cover it. The optimal value necessarily resides between 1:N and N:N but it remains to be seen whether an optimum independent of any particular implementation exists.

If criteria for a successful state space connectivity can be found, also a method for constructing such state spaces would be useful. Similar methods guiding the selection of good annealing schedules have already been studied both empirically and theoretically. Now the same remains to be done for the design of the state space and the selection of transitions.

References

Aarts, E. and Korst, J. (1989). *Simulated Annealing and Boltzmann Machines.* John Wiley & Sons.

Ahonen, H. and Viertiö, P. (1989). Knowledge-based pagination. In Jaakkola, H. and Linnainmaa, S., editors, *Scandinavian Conference on Artificial Intelligence*, pages 607–611, Netherlands. IOS.

Goldfarb, C. F. (1990). *The SGML Handbook.* Oxford University Press, New York.

Haykin, S. (1994). *Neural Networks.* Macmillan College Publishing Company, Inc, New York.

Henning, C., Ojala, E., and Pellinen, J. (1988). *Sanomalehden ulkoasuopas.* Sanomalehtien Liitto, Forssan Kirjapaino Oy, 2. edition.

Heragu, S. S. (1992). Recent models and techniques for solving the layout problem. *European Journal of Operational Research*, 57:136–144.

Heragu, S. S. and Alfa, A. S. (1992). Experimental analysis of simulated annealing based algorithms for the layout problem. *European Journal of Operational Research*, 57:190–202.

Hofri, M. (1987). *Probabilistic Analysis of Algorithms.* Springer-Verlag, New York.

Hwang, S.-M., Kao, C.-Y., and Horng, J.-T. (1994). On solving rectangle bin packing problems using genetic algorithms. In *Proceedings of the 1994 IEEE International Conference on Systems, Man and Cybernetics*, volume 2, pages 1583–1590. IEEE Press, Piscataway, NJ, USA.

Ingebritsen, L. M. (1994). (Ding) Gnus. http://www.ifi.uio.no/ larsi/ding.html.

ISO (1995). DSSSL draft international standard. International Standards Organization, IEC DIS 10179.2.

Kirkpatrick, S., Gelatt, C., Jr., and Vecci, M. (1983). Optimization by simulated annealing. *Science*, 220:671–680.

Kohonen, T. (1995). *Self-Organizing Maps.* Springer-Verlag.

Kouvelis, P., Chiang, W.-C., and Fitzsimmons, J. (1992). Simulated annealing for machine layout problems in the presence of zoning constraints. *European Journal of Operational Research*, 57:190–202.

Kukkonen, S. (1992). Sanomalehtiartikkelin automaattinen taitto. Master's thesis, Helsinki University of Technology, Espoo, Finland.

Rehe, R. F. (1985). *Typography and design for newspapers*. IFRA, Germany.

Sperberg-McQueen, C. M. and Burnard, L., editors (1993). *A Gentle Introduction to SGML in Guidelines for Electronic Text Encoding and Interchange*. ISO, Chicago.

Sugai, Y. and Hirata, H. (1991). Hierarchical algorithm for a partition problem using simulated annealing: application to placement in VLSI layout. *International Journal of Systems Science*, 22:2471–2487.

Veijonen, T., Ahonen, H., Leikola, V., Niku-Paavola, J., and Ylä-Jääski, J. (1994). Automatic page layout for electronic news distribution. *Proc. IASTED/ISMM International Conference of Distributed Multimedia systems and Applications*.

Ylä-Jääski, J. and Ahonen, H. (1990). Knowledge-based newspaper pagination. *Applications of Artificial Intelligence VIII*, 1293:767–774.