RAIRO. INFORMATIQUE THÉORIQUE ET APPLICATIONS

H. MAURER
F. KAPPE

Theory as basis for advances in hypermedia

RAIRO. Informatique théorique et applications, tome 28, n° 3-4 (1994), p. 201-211

<http://www.numdam.org/item?id=ITA_1994__28_3-4_201_0>

Informatique théorique et Applications/Theoretical Informatics and Applications
(vol 28, n° 3-4, 1994, p 201 à 211)

THEORY AS BASIS FOR ADVANCES IN HYPERMEDIA

by H. MAURER ([1]) and F. KAPPE ([1])

I am dedicating this paper to my friend Karel Culik II on the occasion of his 60th birthday

Hermann MAURER

Abstract – *Hypermedia is currently one of the big buzz words in computer science, it is indeed sometimes ridiculed as an area with much too much "hype" Yet the basic idea to have large bodies of networked data of heterogeneous type (i e a mixture of data types such as text, pictures, videoclips, and other material) accessible by means of linking chunks of information together in an "associative fashion" will influence deeply how we live and work, and this statement is no exaggeration Yet much of current hypermedia work is severely limited due to the fact that the solution of some of the problems involved needs deep theoretical results We try to establish this fact by choosing two rather different problems, and the solution of one of them may well be obtained based on results of Karel Culik II*

1. INTRODUCTION

In addition to storing and manipulating numeric and textual data, computers have become increasingly capable of handling other types of information: drawings built from two-dimensional primitives, pictures as arrangements of "pixels" with growing resolution and going from black/white to systems involving millions of colours, digitised audio- and videoclips, three-dimensional models of "real" or "virtual" objects, etc.

Systems incorporating such a variety of heterogeneous types of information that use new paradigms for locating information by creating "links" from one object or "cluster of objects" to some other "document" have become valuable tools for presentation and teaching purposes, for creating new kinds of information systems and for novel ways of communication and collaboration.

([1]) Institute for Information Processing and Computer Supported New Media (IICM), Graz University of Technology, Schießstattg 4, 8010 Graz, Austria

Informatique théorique et Applications/Theoretical Informatics and Applications
0988-3754/94/03-04/$ 4 00/

Such systems are usually called hypertext of better hypermedia systems and have received wide-spread attention, starting in the mid-eighties. Rather than repeating arguments and explanations that have been given many times before we prefer to refer to some of the more readily accessible literature such as [16, 21] and the references therein, the survey paper [29], the books [3] and [25], the forthcoming book [4], and the materials on multimedia and related subject that have appeared as special issues of journals such as C.ACM 31, 7 (1988), C.ACM 33, 3 (1990), C.ACM 34, 4 (1991), C.ACM 37, 2 (1994), and J.MCA 14, 2 (1991). For readers interested in applications of hypermedia we finally want to refer to [18], and why (animated) pictures play such an important rule to [6], [15], and [19].

Despite the tremendous surge in activities in hypermedia, both on the research and project front, many of the most basic issues have not been solved in a way that can be considered truly satisfactory, so far. Digitized movies not only often still require sophisticated compression techniques and dedicated hardware such as most methods based on MPEG [17] but also need large amounts of storage (in the range of one GByte per hour of reasonable-quality movie); the situation with high quality 3D models (let alone animated ones, *see* [20]) is similar, if not worse; compression techniques for HiFi quality stereo-sound are as important as new database concepts and data models [22]. Even the most sophisticated data compression techniques have to be modified if one wants to have "reconfigurable" databases; and fast fuzzy-search techniques that allow the search for not exactly specified strings or even geometric objects are only now being developed, *see* e. g. [26]. Above are just a few of the many examples of fundamental research results necessary to support truly large and universal hypermedia applications.

This is also and particularly true of the area that is at the heart of most current-day hypermedia systems: large databases of high quality raster images. These are the two main problems we will deal with in this area to substantiate our claim that further developments in theory are necessary for successful hypermedia work. We will deal with two separate issues in the next sections. More specifically we will look at the problem of image compression in Section 2 and at the problem of picture retrieval in Section 3.

2. IMAGE COMPRESSION

It can often be heard that image compression will be a non-issue in the future, because of the rapid increase of (disk) storage available to modern computers, the ever-decreasing cost per byte, and the availability of networks

that allow sharing of disk space. However, from practical experience with large-scale hypermedia systems we contend that this is not true at all for the following reasons:

• Modern computers can display rather high quality images (typically about 1 million pixels, each of them in 24-bit color, *i. e.* a screenful is worth 3 MB), and therefore users demand this high quality, while a few years before they would be more than satisfied with much less. Even worse, high-quality colour slides are said to have a resolution of over 100 lines per mm. To capture a colour slide of the size of 24×36 mm at full quality 8,64 million pixels, each pixel with one of 2^{24} colours have to be stored, amounting to almost 25 MByte of data. Thus, only some 40 high-quality images can be held on one GByte of storage (if no data compression is used).

• The amount of information (about 600 MB) that can be held on a CD-ROM – the most popular distribution medium for hypermedia information – does not increase (at least in the foreseeable future), so we can store only 24 full-quality color slides or 200 screen-size images on one CD, without using compression. Also, the reading speed of CD-ROMs is slow (150 KB/s on standards drives, better drives offer 300 KB/s). Even a 300 KB/s a full screen takes 10 seconds to read, and a slide even about 1.4 minutes, which is clearly undesirable for interactive applications.

• Most depressingly, the network speeds do not even remotely increase as fast as disk capacity, bytes of RAM per dollar, or CPU power. The very same 10 MBits/s Ethernet has been dominating the local-area network market for the last decade, and chances are that it will do so for a number of years more. Even if it will eventually be replaced by a, say, 100 Mbits/s medium, this increase by a factor of 10 over 10 years looks rather slim compared to the 100-fold increase of typical disk space, and the almost 1000-fold boost in CPU power available on microcomputers during the same time. Furthermore, the gain in network speed will be more than neutralized by the fact that the number of computers on the network will also have increased, the network is used by more distributed applications, and that new applications consume more bandwidth (e. g. because they transfer more multimedia data than the old applications).

Let us examine the last point more closely: Under real-word conditions, an Ethernet provides only about 100 KB/s, which means that a full screen takes about 30 seconds and a high-quality slide more than 4 minutes to transmit. It is not necessary to point out that in wide-area networks like Internet the situation gets even worse. Under these circumstances, compression clearly

is an issue. Even if the images are archived completely uncompressed on a server disk, it may be faster (depending on CPU power and compression method) to compress them at the server end, transmit the compressed image over the network, and decompress then at the user's workstation (note that for many compression methods the 3 operations can be performed in parallel!) than to transmit the uncompressed image. Clearly, there is a tradeoff between the necessary transmission time and the compression/decompression time. Because CPU power grows much faster than network speed, if makes a lot of sense to look at computationally more expensive compression algorithms. This is why we think that high-performance compression algorithms are an issue even (and even more!) in the future.

We will discuss a number of such techniques in this Section, culminating in work done by Culik II and co-authors. However, before doing so, let us mention in passing that for certain images (composed of simple geometric objects) the technique of "vectorisation" (of storing the image as a set of graphical primitives with certain attributes) can provide the highest compression factors [14]; simpler pictures may be encoded even using formal languages as has been attempted in e. g. [12, 13, 24, 23].

Let us now turn, however, to image compression techniques as such, *i. e.* to algorithms that compress a raster image while preserving all information ("lossless" techniques) or essential information ("lossy" techniques).

2.1. Lossless compression techniques

Lossless compression/decompression methods are able to reproduce every bit of the original input. The algorithms used are fast, simple, and in widespread use. However, the compression rate that can be achieved is significantly worse than with lossy methods.

For example, the FAX G3 standard for black and white (1 bit/pixel) images uses a combination of run-length and huffman encoding. The huffman table is optimized for images with small black run-lengths and large white run-lengths (which is a typical FAX), and fails (*i. e.* it needs more than the original 1 bit/pixel) when applied to dithered raster images with both small black and white run-lengths.

Many image formats (e. g., "GIF" and "TIFF") use some variation of the algorithm first presented by Lempel, Ziv, and Welch [32] and since then known as the LZW algorithm. This method uses a dictionary to compress repeated occurrences of byte patterns in the original image. The dictionary is adaptive, *i. e.* it is generated by the encoder depending on its input. The

important trick of the LZW method is that the encoder needs not to transmit the dictionary to the decoder (or save it together with the compressed result), as it can be reproduced from the encoder output. The LZW method is fast and simple to implement ([5] present a chip capable of compressing 20 MB/s), and can be used for any kind of data (text, programs, etc.). The compression rate varies with the king of input: computer-generated images of line drawings are compressed very well, while LZW applied to photographs usually achieves not more than 50% compression.

In general, lossy techniques are preferred for photograph-like raster images because they offer a much higher compression without visual effect. There are, however, at least three situations where lossy algorithms can not be used to store images (and hence lossless techniques remain important even for photograph):

• When the image is intended to be post-processed using some image processing tools (e. g. satellite images) in order to extract certain features. These toolds could easily generate misleading data based on artifacts of a lossy compression algorithm (e. g. the block structure of JPEG, *see* below). This is a pity because satellite images tend to be extremely large.

• When the image is expected to be manipulated interactively (enhanced, modified) and the manipulated version stored again, because every decompression/compression cycle further reduces image quality.

• Some medical applications (e. g. archiving of digital X-ray images) demand lossless compression because of legal problems: Otherwise, the implementors could easily face a situation whre they would have to prove in court that, say, some fine but important detail in an X-ray image has not been considered expendable by a lossy compression algorithm, or that some other detail is not an artifact of the particular algorithm involved. This would give them a difficult time, because the original image (uncompressed) image is usually no longer available.

2.2. JPEG

JPEG (for "Joint Photographic Experts Group") is an informal name for ISO international standard number 10918 ("Digital Compression and Coding of Continuous-tone Still Images"). It is the only ISO standard for image compression so far, and it defines a number of compression and encoding options, including lossless techniques [31]. The most important of them (and usually the only one implemented) is the "baseline sequential codec"

which is a lossy compression scheme based on the discrete cosine transform (DCT).

Compressing a grayscale image with the baseline sequential codec basically involves three steps:

1. The image is transformed from brightness values into a two-dimensional frequency spectrum ("spatial frequencies") using the forward discrete cosine transform (similar to a discrete fourier transform). Because it would be too time-consuming to do this on the whole image, the image is broken into 8×8 blocks before, and these blocks are processed independently. This step yields one so-called DC coefficient (*i. e.* the average brightness of the whole 8×8 blocks) and 63 so-called AC coefficients which resemble the two-dimensional frequency spectrum.

2. This spectrum is then quantized (*i. e.* divided an rounded) using a variable quantization table. This is the only step in the JPEG compression that really loses information. The idea is that most of the energy is usually contained in the lower frequencies, while a small amount of higher-frequency signal is invisible. Therefore, the low-frequency coefficients are divided by small numbers (around 16), while the high-frequency coefficients are divided by 100 and more; and always rounded to the nearest integer. The quantization table can be modified to obtain different compression rates and qualities (higher numbers in the quantization table give better compression but less quality).

3. The resulting stream of small integers is then arranged in zig-zag order, runlength-encoded and then huffman-encoded (no information loss in the step).

Decompression applies the reverse steps in the oppositve order. Compressing color images involves some more steps. The compression method is symmetrical, *i. e.* the amount of computation needed for compression and decompression is about the same. Compression chips have been developed (most notably C-Cube's CL550 [7]) that are fast enough to compress/decompress video frames in real times (*i. e.* 25 or 30 per second). This solution is suboptimal, however, as it does not take into account the similarities between successive video frames (like MPEG [17] does).

At data rates higher than about 1.2 bits/pixel, JPEG-compressed color images are usually indistinguishable from the original (24 bits/pixel). However, below about 0.5 bits/pixel (*i. e.* a 50-times compression) the 8×8 block structure becomes cleary visible.

2.3 IFS

The use of iterated function systems (IFS) for image compression has first been described by Barnsley in [1]. Without going into the details (the interested reader is referred to the more recent book [2]), it is based on the theorem ("collage theorem") that every image can be composed from a number of fractals that in turn can be generated from a set of affine transformations (iterated functions). The idea is that the coefficients for these functions can be stored in much more compact form than the original image, and that the image can be reproduced (with some loss) from just these coefficients.

The techniques works extremely well for fractal-like pictures (e. g. ferns, trees, clouds, etc.) where compression rates of up to 10,000 have been reported, and compression rates of about 100 seem realistic for usual photographs images. A nice feature of the method is that the image can be reproduced at arbitrary resolution, *i. e.* zooming in does not give the "blocky" structure seen in, e. g., JPEG compressed images. Instead, such an image looks "painted".

2.4 WFA

It was Karel Culik II who (with co-workers) introduced the use of weighted finite automata (WFA) for describing texture (grey-tone or color) images [8]. While the WFA methods is somewhat similar to IFS (the idea is that the image-generating automaton is small and can be stored much more efficiently than the image), the generative power of nondeterministic WFA is much higher than that of IFS. A fast decoding algorithm exists (including fast zooming-in without block structure), and, most important, the WFA seems to be the first "fractal type" method for which there is a simple and fast automatic encoding method [10, 9], *i. e.* an algorithm that is able to find a suitable, small automaton that generates (an approximation of) a given image.

Both compression rates and image quality are very high for both grey-tone and color, especially when the WFA is combined with the discrete wavelet transform [30] (this combined encoding method can be viewed as a sophisticated quantization of wavelet coefficients): The most recent work [11] shows good-quality compressed images of the well-known "Lenna" test image that need only 0.027 bits/pixel (*i. e.* a compression ratio of 300 compared to 8 bits/pixel and 900 compared to the original 24 bits/pixel image)!

The WFA method seems to be the best image compression method currently available (and we are quite convinced that Karel Culik II will try to improve it even more). The implications for distributing hypermedia technology are enormous:

- For example, using WFA with 900-times compression a single 600 MB CD-ROM has room for almost 22,000 full-resolution color slides (3,600 × 2,400 pixels each), and even 180,000 full-screen images (of 1 K^2 pixels each)!
- As a full-screen image of 1 million pixels, compressed with 0.027 bits/pixel needs only 27,000 bits (3,375 bytes), it can be transferred in only 4 ms on a local-arae Ethernet, and still in less than half a second on a 64 kbits/s ISDN line. This means that sending images over wide-area networks becomes feasible even in the interactive applications.
- Assuming that high-speed decompression software or hardware becomes available, it may even be possible to use WFA to transmit high-quality video over ordinary (64 kbits/s) phone lines (what is now possible using MPEG at this speed is definitely not high quality).

We hope that this section has shown how theoretical results like the WFA can influence the development in more application-oriented fields like hypermedia, and eventually influence the may we will live and work in the future.

3. PICTURE RETRIEVAL

When using a large database of pictures it is clearly necessary to locate pictures when desired.

Very often, this is done by using keywords or descriptors attached to the pictures and employing ordinary database techniques and queries.

In many situations it would be desirable, however, to locate pictures by their contents ("all pictures with an island with palm trees and no people") rather than their descriptors.

This, however, clearly leads to very complex issues of feature extraction, feature categorisation, and feature comparison. To explain how far away we are presently from being able to manage queries such as the one mentioned above about islands with palm trees efficiently let us look at some very simple instances and observe the difficulties inherent in them.

Image processing has developed to an extent that simple feature extraction is indeed possible, *i. e.* the detection of lines, rectangles, circles, simple

polygons and such is possible; however, the categorisation of such simple geometric objects into notions such as "islands", "palm trees", "people", etc. *i. e.* the understanding of pictures, is quite elusive at this point in time, and no significant progress is foreseeable.

Thus, content queries will have to be restricted to queries such as "find all pictures with at least 30% blue, at least 10% yellow, at least 5% green and a bright circle surrounded by some blue area": such query might be an attempt to locate an island (yellow) with blue water, and some palm trees and a sun (bright circle) in a blue sky.

More realistically, in a CAD database a query for "pictures with two interlocking cogwheels, one rectangle below the cogwheels and at least 3 circles positioned to the right of the rectangle" might find us a collection of objects containing the one we are really interested in.

Query languages for such situations have been discussed in detail in [28]. Searches of this kind can usually only be carried out by examining image after image.

To speed up the searching process it is advisable to introduced simple "signatures" for pictures that permit the exclusion of many objects in the database by just comparing query and signature.

Typical signatures can be vectors of integers indicating how many objects of a certain kind are present in the picture: such signatures ignore positional and size information completely, yet can hopefully be computed and searched fairly fast.

A typical signature is thus a vector of positive integers $(a_1, a_2, \ldots, a_m)$ where each a_i indicates that the object of type i is present a_i times. A query could be translated into a "search vectors" of pairs $q = ((d_1, b_2), (d_2, b_2), \ldots, (d_m, b_m))$ where d_i indicates whether the object of type i is to be present exactly b_i times $(d_i = 1)$, at lest b_i times $(d_i = 2)$, at most b_i times $(d_i = 3)$ or whether the presence/absence of objects of type i is irrelevant ($d_i = 0$, indicating a "don't care" situation).

Searching the database according to a specific query q implies now

(*a*) computing the search vector corresponding to q;

(*b*) looking through the once-and-for-all computed signatures one by one;

(*c*) examining only those pictures whose signatures "match" with the search vector.

As described, the search effort grows linearly with the size n of the database because of (*b*). It is, thus, natural to look for a data structure for

the signature that would allow sub-linear search times. It is interesting and depressing to note that no reasonable sublinear algorithm is known to date for this problem unless $d_i = 1$ for all i in which case an arbitrary sorting criterion and a binary search provide logarithmic performance. The use of trees or high-dimensional Voronoi diagrams yields techniques that work in $\log^m(n)$ time which, for large m is only a theoretical improvement and simple heuristic techniques provide somewhat better average performance but no serious computational evaluation is known to the authors. The most promising approaches seem to be based on Geometric Hashing and variations thereof, *see* [27].

Essentially, however, even the most basic subproblem occurring in the picture retrieval area remains open, leaving only brute-force approaches for contents directed searches.

Thus, despite all theoretical results that are available to handle large hypermedia systems more efficiently much important basic research remains to be done. We trust that Karel Culik II, to whom this paper is dedicated, will continue his important contributions in the future ahead.

REFERENCES

1. M. F. Barnsley, *Fractals Everywhere*, Academic Press, New York, 1988.
2. M. F. Barnsley and L. P. Hurd, *Fractal Image Compression*, AK Peters, Ltd., Wellesley, MA, 1993.
3. E. Berk and J. Delvin (editors), *Hypertext/Hypermedia Handbook. Software Engineering Series*, McGraw-Hill, New York, 1991.
4. J. Buford-Koegel (editor), *Multimedia Systems*, ACM Press, 1994.
5. S. Bunton and G. Boriello, Practical Dictionary Management for Hardware Data Compression, *Communications of the ACM,* 1992, *35*, 1, pp. 95-104.
6. P. Carlson and H. Maurer, Computer Visualization, a Missing Organ and a Cyber-Equivalency, *Collegiate Microcomputer*, 1992, *10*, 2, pp. 110-116.
7. D. Clark, C-Cube's JPEG Image Compression: Where's It Headed., *Advanced Imaging*, 1990, pp. 50-71.
8. K. Culik II and J. Kari, Image Compression using Weighted Finite Automata, *Computers & Graphics*, 1993, *17*, 3, pp. 305-313.
9. K. Culik II and J. Kari, Image-data Compression Using Edge-optimizing Algorithm for WFA Inference, *Journal of Information Processing and Management* (to appear).
10. K. Culik II and J. Kari, Inference Algorithms for WFA and Image Compression, in Y. Fisher (editor), *Fractal Image, Encoding and Compression,* Springer-Verlag (to appear).
11. K. Culik II and J. Kari, *A Recursive Algorithm for Image Compression with Finite Automata*, submitted to DDC'94.
12. K. Culik II and H. Maurer, Linearizing Selector-Graphs and Applications theoreof, *Angewandte Informatik*, 1977, *9*, pp. 386-394.

13. K. CULIK II and H. MAURER, String Representation of Graphs, *Intern. J. Computer Math.*, 1978, *6*, pp. 273-301.
14. ISO, *Information Processing Systems – Computer Graphics – Metafile for the Storage and Transfer of Picture Description Information* (CGM), ISO IS 8632. ISO, 1987.
15. D. JONASSEN, R. GOLDMAN-SEGAL and H. MAURER, *DynamIcons as Dynamic Graphic Interfaces: Interpreting the Meaning of a Visual Representation*, IIG Report 376, IIG, Graz University of Technology, Austria, 1993 (submitted for publication to An International Journal of Human Computer Interaction).
16. F. KAPPE, H. MAURER and N. SHERBAKOV, Hyper-G – A Universal Hypermedia System, *Journal of Educational Multimedia and Hypermedia*, 1993, *2*, 1, pp. 39-66.
17. D. LE GALL, MPEG: A Video Compression Standard for Multimedia Applications, *Communications of the ACM*, 1991, *34*, 4, pp. 46-63.
18. J. LENNON and H. MAURER, *Applications of Hypermedia in Universities*, Computer Science Report 78, University of Auckland, September 1993.
19. J. LENNON and H. MAURER, *MUSLI – A Multi-Sensory Interface Language*, IIG Report 375, IIG, Graz, University of Technology, Austria, 1993, to appear in Proc. ED-MEDIA 94, Vancouver, 1994.
20. N MAGNENAT-THALMANN and D. THALMANN (editors), *Computer Animation'92*, Springer Pub. Co., 1992.
21. H. MAURER, Why Hypermedia Systems are Important, In *Proc. ICCAL'92*, LNCS 602, Springer Pub. Co., 1992, pp. 1-15.
22. H. MAURER, A. PHILPOTT and N. SHERBAKOV, *Hypermedia Systems without Links*, Computer Science Report 75, University of Auckland, 1993. Accepted for publication in J.MCA.
23. H. MAURER, G. ROZENBERG and E. WELZL, Chain Code Picture Languages, In *LNCS 153*, Springer Pub. Co., 1983, pp. 232-244.
24. H. MAURER, G. ROZENBERG and E. WELZL, Picture Description Languages, *Information and Control*, 1982, *54*, pp. 155-185.
25. J. NIELSEN, *Hypertext & Hypermedia*, Academic Press, San Diego, CA, 1990.
26. I. RIGONTOS and A. CALIFONA, *dFLASH: A Distributed Fast Wok-Up Algorithm for String Homology*, Technical Report, IBM T. J. Watson Research Center, Yorktown Heights, 1993.
27. I. RIGONTOS and R. HUMMEL, A Bayesian Approach to Model Matching with Geometric Hashing, 1991, (submitted to: Computer Vision, Graphics and image Processing: Image Understanding).
28. J. STOGERER, *Suchen und Ersetzen in Bilddatenbeständen*, PhD thesis, Graz University of Technology, 1990.
29. I. TOMEK, S. KHAN, T. MULDNER, M. NASSAR, G. NOVAK and P. PROSZYNSKI, Hypermedia – Introduction and Survey, *Journal of Microcomputer Applications*, 1991, *14*, 2, pp. 63-100.
30. R. A. D. VORE, B. JAWERTH and B. J. LUCIER, Image Compression through Wavelet Transform Encoding, *IEEE Transactions on Information Theory*, 1992, *38*, pp. 719-746.
31. G. K. WALLACE, The JPEG Still Picture Compression Standard, *Communications of the ACM*, 1991, *34*, 4, pp. 30-44.
32. T. A. WELCH, A Technique for High Performance Data Compression, *IEEE Computer*, 1984, *17*, 6, pp. 8-19.