PostGraphe: a system for the generation of statistical graphics and text

Massimo Fasciano <fasciano@IRO.UMontreal.CA>
Guy Lapalme <lapalme@IRO.UMontreal.CA>
Département d'informatique et de recherche opérationnelle
Université de Montréal, CP 6128, Succ Centre-Ville
Montréal Québec Canada, H3C 3J7

April 21, 1996

Abstract

Graphics and text have to be well integrated in order to achieve their full potential. A picture shows but a text describes. In a statistical report, graphics show the data that is analyzed in the text. This paper describes a system, called PostGraphe, which generates a report integrating graphics and text from a single set of writer's intentions. The system is given the data in tabular form as might be found in a spreadsheet; also input is a declaration of the types of values in the columns of the table. The user chooses the intentions to be conveyed in the graphics (e.g. compare two variables, show the evolution of a set of variables ...) and the system generates a report in LaTeX with the appropriate PostScript graphic files.

1 Introduction

Graphics and text are very different media. Fortunately, when their integration is successful, they complement each other very well: a picture shows whereas a text describes. In this research, we are studying the interaction between the text of a statistical report and its figures. Reports are an organized synthesis of data that span a whole array of forms going from tables of numbers to a text summarizing the findings. Statistical reports are particularly interesting because the reader can easily be overwhelmed by the raw data. Without an appropriate preliminary statistical analysis to make the important points stand out and, without an efficient organization and presentation, the reader might be lost. In this paper, we present the important factors in the generation process as well as its important steps. We then give an overview of a statistical report generator called PostGraphe.

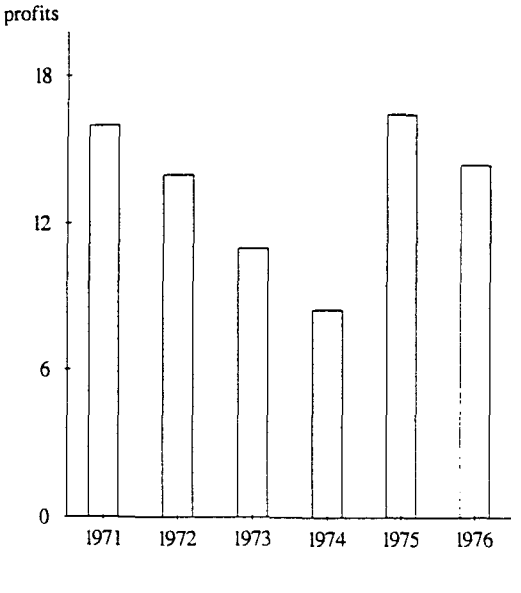
2 Important factors in the generation process

A number of factors have to be considered in order to produce a statistical report containing text and graphics. These factors include the writer's goals, the types and values of the variables to be presented, and the relations between these variables.

The writer's goals have a major role in the generation process. As we can see in figures 1 and 2, the same data can be expressed in very different ways according to the message the writer wishes to transmit. The example presents the same set of data — profits during the years 1971-1976 — according to two different perspectives which reflect the writer's goals or intentions. In figure 1, the goal is to present the evolution of the profits during the relevant time period. In figure 2, the message is totally different, and corresponds to a different goal: to compare the profits for the 6 years of the data set. Because of its tempo-

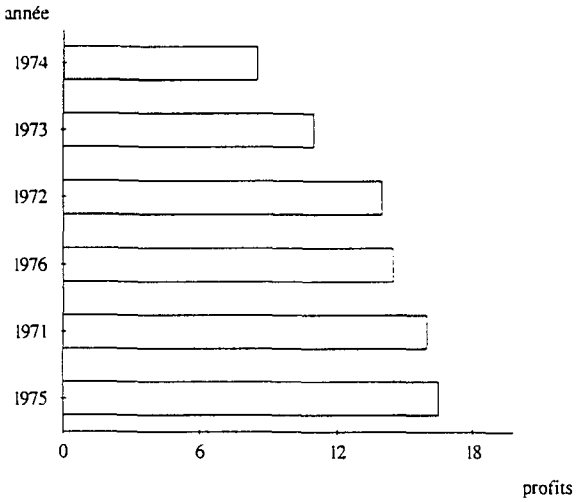
ral nature, the usual way of presenting this data is the message of evolution. The difference can be seen in the organization of the graphs and in the wording of the text. In figure 1, the evolution is emphasized by using the horizontal axis for the years [20, 3]. This is the accepted way of presenting temporal data. The years are sorted in ascending order, also to give the impression of evolution. The associated text describes the overall evolution and points out an interesting irregularity. On the other hand, the writer's intention for figure 2 is totally different. In order to show a comparison, a few structural changes have to be made. First of all, the years are presented on the vertical axis, thus eliminating the impression of evolution [20]. This change is important to the perception of the graph because it makes its message clearer by eliminating a false inference. Second, the years are treated as a nominal variable instead of an ordinal one, and thus sorted according to the profit values. This reordering has two positive effects: it further destroys the impression of evolution by making the years non-sequential and it allows a better comparison of the profits [9]. The text is also different from the one in figure 1: instead of describing how the profits evolved, it merely points out the best and the worst years for profits. This difference in perspective is important for a writer, especially when trying to convey more subjective messages [10].

If the communicative goals aren't well identified, it is very easy to convey the wrong impression to the reader. This problem is often complicated by the fact that a single graph or text can convey many messages at once, some more direct than others. For example, figures 3 and 4 show 2 graphs that share a subset of intentions. The main message is one of evolution in figure 3 graph and correlation in figure 4, but both graphs also transmit, with lower efficiency, the main message of the other graph. Correlation is perceptible in the line graph because the two sets of data can be followed together and evolution can be perceived in the point graph because significant year clusters are marked by different shapes. Thus, determining which types of graphs or text best satisfy single goals is not sufficient; one also has to



Globally, the profits have gone down despite a strong rise from 1974 to 1975.

Figure 1: single communicative goal: evolution



The profits were at their highest in 1975 and 1971. They were at their lowest in 1974, with about half their 1975 value.

Figure 2: single communicative goal: comparison

take into account the cumulative influence of the secondary messages conveyed by all parts of the report.

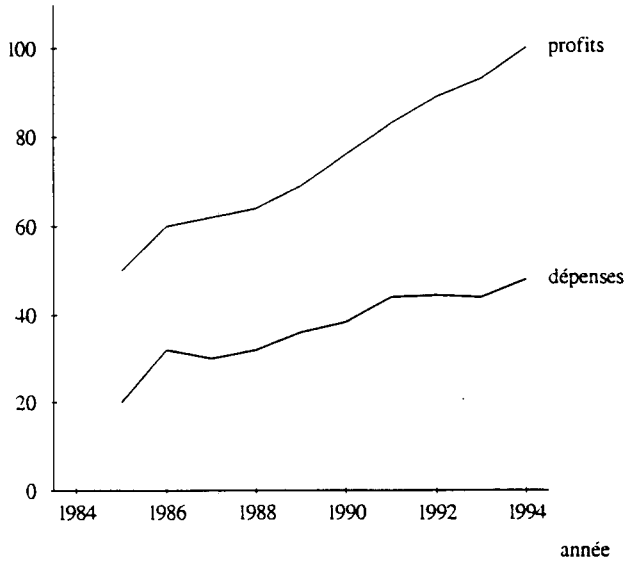


Figure 3: combined communicative goals: evolution and correlation

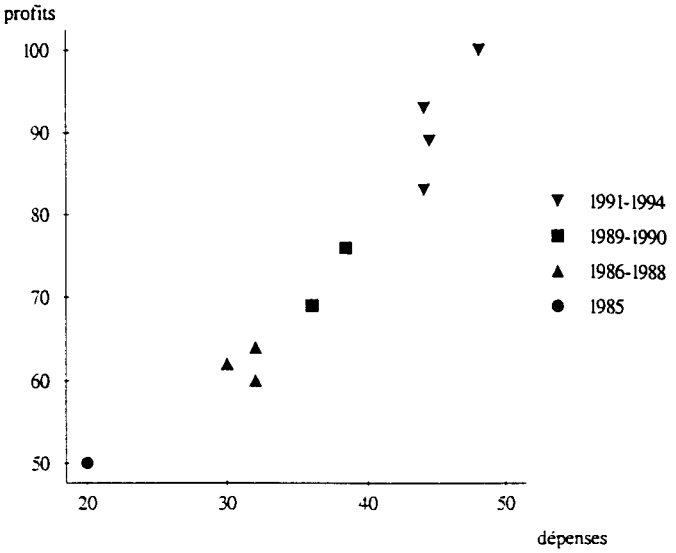


Figure 4: combined communicative goals: correlation and evolution

As might be expected, the types of variables give a lot of information about the structure of the elements of the report [2, 12, 13]. For example, although a continuous variable is better represented by a line graph, the nature of a discrete variable will become more apparent using a column graph. Graphics-only systems can get away with a simple type-system as presented in [12, 13]. This type system classifies the visual and organizational properties of data variables using such categories as *nominal*, *ordinal*, and *quantitative*. A more complex classification is helpful in general as it allows the classification of other useful properties, e.g. temporal, but in the case of text generation, it becomes necessary in order to express the units of the variables. For example, knowing that "May" and "July" are months allows a generator to produce temporal expressions such as "two months later" [11].

To further refine the selection process, we have to take into account not only the types, but also the specific values of the data samples. The number of values sometimes has a lot of influence on the choice of an expression schema. For example, a discrete variable with 200 values will often be treated as continuous, thus overriding the influence of its natural type. In other cases, the range of values has a strong influence. Indeed, as can be seen in figure 5, a seemingly good choice can be invalidated when the range of values is extreme.

These factors influence the structure and contents of a statistical report and have to be looked at simultaneously in order to be effective. Many systems based on APT [12, 13] use types to determine structure, but specific values are often overlooked and the simultaneous use of types and goals is rare. To further illustrate the importance of simultaneous application of these factors, let's look at figure 5 again. In this graph the small values are not readable because of the scale. In general, this is considered a problem and can be corrected by using a different scale (logarithmic or split). However, if the intention of the writer is to illustrate the enormous difference between company D and the others, the graph is very efficient as it is.

Our research extends the work of Bertin [2] and

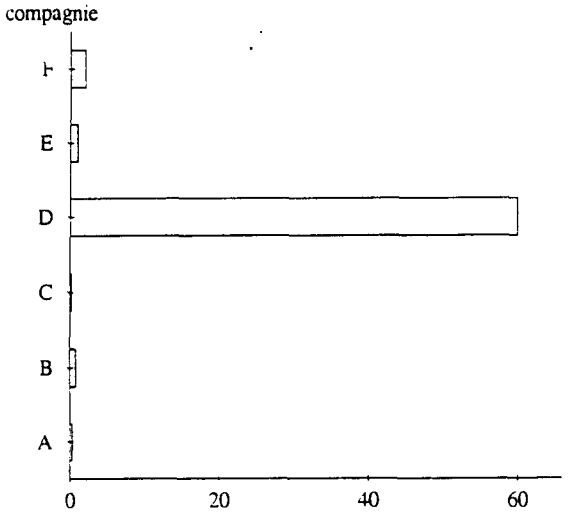


Figure 5: Extreme ranges cause low readability

MacKinlay [12, 13] on the types and organization of variables, the work of Zelazny on messages and goals [20] and integrates it with other theories on the use of tables [19, 9] and graphs [5, 6, 8, 16, 17, 18].

3 A report generator: the PostGraphe system

Our prototype, the PostGraphe system is a compromise between keeping the implementation simple and obtaining satisfactory results. After examining a number of reports, we noticed that text and graphics were often used together to transmit the same message. Since one of our goals was the study of the integration of text and graphics, we decided to always generate a text/graphics pair for every message.

Unfortunately, we could not simplify the realization level. We would have prefered to use a readily available graphical tool for realization and spend more time on higher-level aspects such as the medium selection. A few attempts were made using tools such as X-Lisp-Stat for point and line graphs and LaTeX for tables. Unfortunately, too many high-level choices depend on simple low-level details such as the number of available colors or the positioning of textual labels in a graph. By designing our own graphical realizer in Prolog, the same language as the rest of the system, we were able to precisely integrate it in the decision process, thus allowing more accurate heuristics and a backtracking approach for more complex cases.

As for the text realization tool, we chose to adapt a systemic-based text generator called PréTexte [11]. This system was well-suited to our needs for two reasons: first, it was developed in Prolog, making it easy to integrate into PostGraphe. Second, it specializes in the generation of temporal expressions. Since evolution is one of the most frequent goals in a statistical report, the temporal knowledge built into PréTexte proved very useful.

We will now describe the major steps followed by the system in the generation of a report.

The input of PostGraphe, consists of 3 special annotations followed by the raw data. These annotations indicate the types of the variables, how to determine the relational keys for the data and a series of predicates describing the writer's intentions. The justification for these annotations and their Prolog syntax are presented in detail in [10]. See figure 6 for an example of their use.

3.1 Types

The type system's role is to associate to every variable of the input a set of properties and a unit. The properties are organised as a multiple inheritance graph divided into a number of sub-graphs, each corresponding to a specific feature [10]. The most important sub-graphs describe the following features: organization (nominal, ordinal, quantitative, ...) [2, 12, 13], domain (enumeration, range, ...), temporal (month, year, ...), format (integer, real, ...), mesurements (distance, duration, ...), and specific objects (countries, ...). The properties have a variable number of parameters which can be used to further specify their function. For example, for an enumerated type (domain sub-graph), a single parameter specifies

the list of values for the enumeration.

In the input, the main type (or class) of each variable is specified, as well as a list of auxiliary types. The auxiliary properties override the ones that are inherited from the class, thus allowing the tailoring of built-in types in the input. Also, a number of automatic type definitions are added according to the nature of the data (integers, labels, ...).

Units are organized in a parallel inheritance graph. The inheritance mecanism is much simpler than the one used for types. A unit can be associated with every type (e.g. percentage $\mapsto$ %). If a unit cannot be found using single inheritance, the name of the type is used as a unit. This process is described in more detail in [10].

3.2 Relational keys

Relational keys are similar to the notion of the same name in relational databases [7] and help determine which variables depend on which others. They are also used for ordering variables in some graphics so that the more important ones (usually the keys) are given the more visible positions.

One of the design goals of PostGraphe was to be able to function as a front-end to a spreadsheet. It was thus important to keep the data as close as possible to a format compatible with that type of software. Although a representation at the level of an entity relationship diagram would have been quite useful, especially for long reports and global relationships between sets of data, we chose to limit the input to a table-like structure which is easily obtainable from a spreadsheet. Consequently, PostGraphe must be able to automatically compute the relational keys it needs from the data.

Sometimes, automatic calculation of keys can give strange results which do not fit with the semantics of the variables. For example, a variable such as *profits* can wind up as a key if its values are all different but it is rarely desirable to express a set of variables such as *years* and *company names* as a function of *profits*. It is usually the other way around.

To solve this problem, 2 optional informations are specified in the input: a list of variables that can be used as keys and a list of variables that cannot be used as keys. This method is easy to implement in a spreadsheet, and some control is maintained without having to abandon automatic calculation of keys (useful for large partially unknown data sets).

3.3 Writer's intentions and planning

The writers' intentions describe what to say and up to a certain point, how to say it. Intentions are constraints on the expressivity of the chosen text and graphics. PostGraphe tries to find the smallest set of schemas that covers the writer's intentions.

The following basic intentions are covered in our model: the *presentation* of a variable, the *comparison* of variables or sets of variables, the *evolution* of a variable along another one, the *correlation* of variables and the *distribution* of a variable over another one. Some of these intentions are further divided into more specific subtypes.

The study of intentions is a major topic of our research. More details about the organization of our goal system can be found in [10].

PostGraphe uses the same planning mechanism to generate text and graphics. The planner uses the types and values of the data as well as the relational keys but it is mainly goal-driven. It builds on the ideas of Mackinlay [12, 13] but extends them in important ways.

MacKinlay's algorithm, as used in APT, takes as input a set of typed variables and determines the most efficient graphical encoding (position, length, color, ...) for each of them. There are many ways of expressing each variable and the system tries to find a way of expressing them all graphically in the same figure, if possible, or in a set of related figures. APT works by allocating the best possible graphical encoding to each variable and then checking if the result is feasible. If it is not, it backtracks on the last allocation and tries the next best encoding for it. The feasability of a set of choices depends on the output medium (2D vs 3D, color vs greyscale). Since the variables are

allocated sequentially, their ordering is important and determines which variables will get the best encodings in problem situations. The algorithm doesn't try to maximize the overall efficiency of a result but assumes that important variables are listed first and gives them the best encodings.

This method has a few shortcomings: it is based on a very limited set of types (quantitative, ordinal, nominal), it works on individual variables instead of global relations and it is not easily applicable to text. Working with individual variables is an interesting approach to the problem of graphics generation as it allows the system to reason on the low level components of graphics and it makes it more efficient. On the other hand, it creates 2 major problems: it is ambiguous at the realization phase and it ignores inter-variable phenomena. The ambiguity stems from the fact that a number of structurally different graphs can express the same variables using the same encodings. For example, line, bar, column and point graphs can all be used to present 2 variables using positional encoding. However, there are important differences between these 4 graphs. The lines in a line graph, the rectangles and their orientation in bar and column graphs all play an important role in the perception of the data. These differences play a major role in the expression of inter-variable phenomena such as comparison and correlation. For example, correlation is better preceived on a point graph than on a line graph.

PostGraphe does not use a list of variables as its main input. Instead, it uses a set of inter-variable or intra-variable goals. The result of our planning algorithm is a schema for each group of compatible goals. These schemas are used for text as well as graphics. No ordering of goals or variables is assumed because all choices are weighted and a global quality function allow the system to maximize the overall efficiency of each graph. By default, the system assumes that all user goals are equivalent but the user can choose to change their relative weights in the input to assure that some of them are better expressed by the system. This maximization complicates the exploration of the solutions as it becomes impossible to return the first feasible solution. Theoretically, one should look at all possible groups of goals to see if they can coexist in the same graph and evaluate how efficient each group is both globally and in regards to constraints placed on individual goals by the user. This is obviously impossible as it leads to massively exponential behaviour. Heuristics are used by PostGraphe to trim the number of solutions down to a usable level.

The user has the option of manually limiting the scope of the grouping process by building sets of related goals. The system will respect these boundaries and never try to group goals from different sets. The normal algorithm is applied to goals inside each set. If only a single set if goals is specified, the system does all the work of grouping and ordering the information. This manual partitioning of goals is useful to organize goals according to themes (e.g. a set of goals to present the data, a set of goals to illustrate a trend, ...).

Inside a set of goals, the planning process is divided in 4 steps: we first find the intentions that are "compatible" so that each schema takes into account as many intentions as possible while keeping each one "readable". The compatibility of intentions is determined using simple heuristics.

Then we check if each group is feasible and determine the best schema to express it. This step is based on a lookup table, much like MacKinlay's algorithm [12, 13] which uses an association between the type of a variable and the most efficient graphical methods to express it. Our table is goal-oriented instead of type-oriented: it associates each possible user goal with the schemas that can express it. The table entries are weighted, and the result of this phase is a list of candidates sorted from the most to the least efficient for the current goals.

The next step is the low-level generation of graphic primitives and text. It can be determined at this stage that a figure cannot be generated because of physical reasons: it is too big to fit, not enough grey levels are available, ... This low level work is quite involved because it has to take into account the 2-D constraints and the limitations of the media. For this we had to develop a Postscript generation system in Prolog in or-

der to determine the exact position of each element (character, line, axis, etc...) of a generated graph. If a candidate is rejected, the next one on the sorted list is tried. The surface text generation is handled by a modified version of PréTexte [11].

Finally, a post-optimization phase eliminates redundancies which can occur because the heuristics sometimes miss a compatible grouping of intentions.

An important aspect of PostGraphe is that it uses no high-level reasoning on intentions. Instead, all of its knowledge is encoded in the links and weights of the table, which was first created using a set of graphical rules and conventions. This approach is more similar to neural nets than MacKinlay's graphical language. The advantage of such an approach is that the table could be automatically modified by the system in response to user satisfaction or dissatisfaction with a result. The obvious problem, as with neural nets, is that the system's knowledge is not easily expressible as a set of human-readable rules.

3.4 An automatically generated report

In this section, we present a simple example of input and output from the PostGraphe system. The Prolog input can be seen in figure 6; lines starting with % are comments. The output was generated by the system, but the information was manually re-ordered and formatted in order to better satisfy the space requirements of this article. In particular, the graphs are presented at roughly 60% of their actual size and the structure of the report was flattened by removing section titles. The captions of the figures were translated from the French output of PostGraphe, but the internal labels and the text produced by PréTexte (figure 11) were left in French. The captions show the name of the schema and the intentions used to generate each figure, with a quality factor (0-100) for each intention.

```
data(% names of the variables
     [annee,compagnie,profits,depenses],
     % types of the variables
     % (/ with aux. properties)
     [annee,
      etiquette,
      dollar/[pluriel(profit)],
      dollar/[pluriel(depense)]],
     % variables that can be part
     % of a relational key
     [annee,compagnie],
     % variables that can't be part
     % of a relational key
     [profits,depenses],
     % writer's intentions
     [% section 1
      [presentation(annee),
       presentation(compagnie),
       presentation(profits),
       presentation(depenses)],
      % section 2
      [comparaison([reduce(moyenne,profits)],
                   [compagnie])>90,
       comparaison([reduce(moyenne,profits),
                    reduce(moyenne,depenses)],
                   [compagnie]),
       correlation(profits,depenses),
       repartition(reduce(moyenne,depenses),
                   [compagnie]),
       proportion([compagnie]),
       evolution(reduce(moyenne,profits),
                 annee)*2]],
     % the data
     [[1987,'A',30,80],
      [1988,'A',35,90],
      [1989,'A',40,110],
      [1990,'A',35,110],
      [1991,'A',30,100],
      ...
      [1990,'E',85,90],
      [1991,'E',40,36],
      [1992,'E',120,105]]).
```

Figure 6: The input as a Prolog term

annee	1987	1988	1989	1990	1991	1992	1987	1988	1989	1990	1991	1992
compagnie	depenses	depenses	depenses	depenses	depenses	depenses	profits	profits	profits	profits	profits	profits
A	80	90	110	110	100	120	30	35	40	35	30	40
B	250	275	250	280	290	300	160	165	140	155	160	160
C	97	120	140	170	190	230	50	55	60	95	100	110
D	120	120	125	160	170	170	60	65	60	75	80	70
E	27	60	70	90	36	105	10	40	62	85	40	120

Figure 7: [Schema: *tableau1*]. presentation of the variables: years (100), companies (100). spending (100) and profits (100).

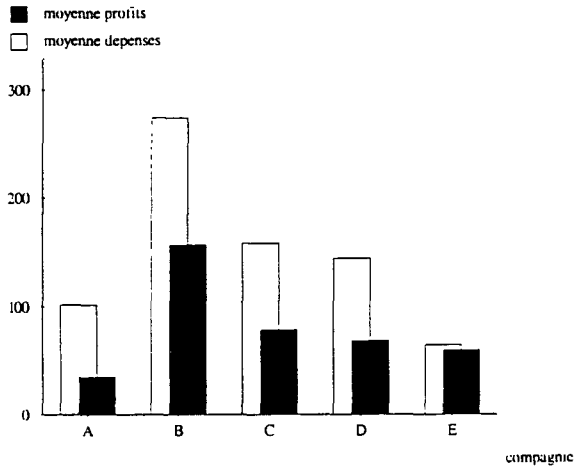


Figure 8: [Schema: *colonnes3*]. comparison of the profit average and the spending average of the companies (80).

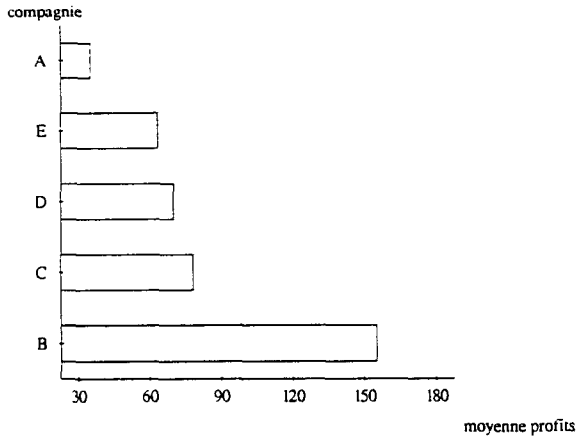


Figure 9: [Schema: *barres1*]. comparison of the profits average between companies (100).

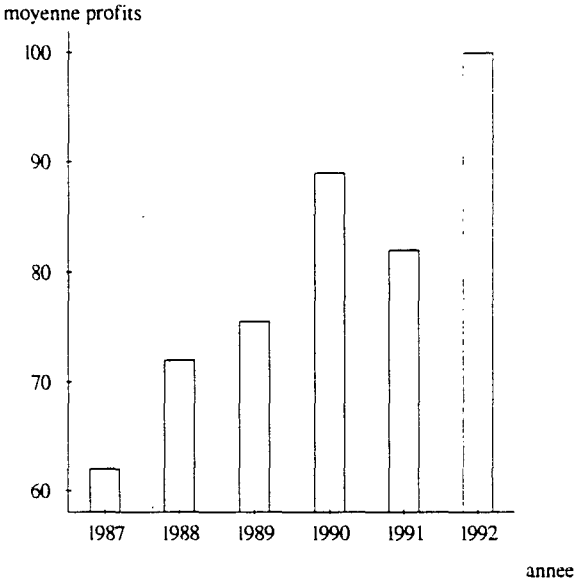


Figure 10: [Schema: *colonnes1*]. evolution of the profit average along the years (94).

De 1987 à 1990 la moyenne des profits a augmenté de 62$ à 89$.

Pendant 1 année elle a diminué de 7$.

Jusqu'en 1992 elle a augmenté de 82$ à 100$.

Figure 11: [Schema: *evolution1*]. evolution of the profit average along the years (99).

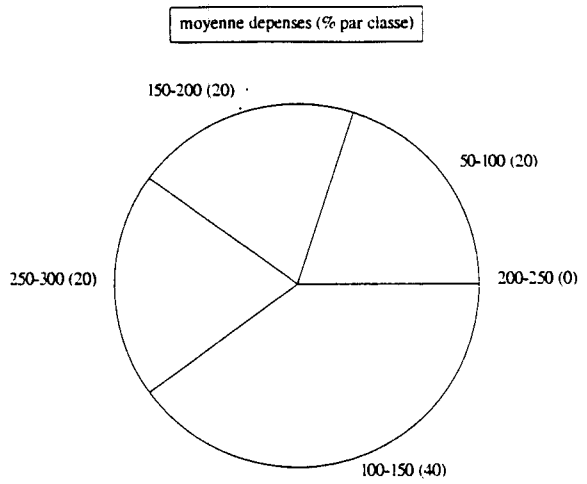


Figure 12: [Schema: *tarte3*]. proportion of companies (100) in the distribution of the spending average of the companies (90).

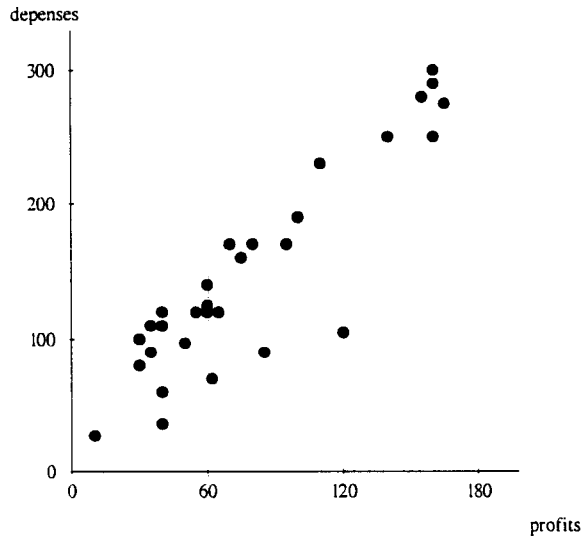


Figure 13: [Schema: *points1*]. correlation between profits and spending (100).

4 Conclusion

The focus of our research is the integrated generation of text and graphics in statistical reports. In order to achieve our objectives, we have considered the writer's goals, the types and values of the variables to be presented, and the relations between these variables.

As we have shown in this paper, all of these factors must be taken into account simultaneously in order to produce an efficient report. Some research has focused on these problems separately, such as MacKinlay's APT system [12, 13] which focuses mainly on type-based graph generation, Zelazny's work on graphs and messages [20], and Casner's study of tasks/goals [4]. Good design rules also have to be considered when choosing and generating tables [19, 9] and graphs [6, 8, 16, 17, 18, 5], as these have a direct influence on the reader's perception of a report. Thus, one has to consider the writer's goals, the data itself and the reader's interpretation.

Other related works include Mittal et al.'s extension [14] to the SAGE system [15] which uses text to explain the structure of graphs and charts – unlike our system, which uses it to present and explain the data itself – and WIP [1], a well-known multimedia generator which has the same goals as our system, but works on a different type of data (structured representations vs tables of numbers). WIP is more concerned with content and media selection according to the user's goals, whereas with PostGraphe, the content is almost directly determined by the writer's intentions, but the structure is totally flexible, as the system must build its internal representations and output from raw data.

Acknowledgements

This work was made possible in part through a grant by the Canadian Natural Sciences and Engineering Research Council (NSERC). Scholarships were also given by NSERC and the Fonds pour la formation de chercheurs et l'aide à la recherche (FCAR). We wish to thank Michel Gagnon for his help in adapting Prétexte. We also wish to thank the reviewers for their very helpful comments.

References

[1] E. André, W. Finkler, W. Graf, T. Rist, A. Schauder, and W. Wahlster. WIP: the automatic synthesis of multimodal presentations. In M. Maybury, editor, *Intelligent*

Multimedia Interfaces. AIII Press, Cambridge, MA, 1993.

[2] Jacques Bertin. *Semiology of Graphics*. The University of Wisconsin Press, 1983. Translated by William J. Berg.

[3] Jacques Bojin and Marcel Dunand. *Documents et exposés efficaces: messages, structure du raisonnement, illustrations graphiques*. Éditions d'Organisation, Paris, 1982.

[4] S. M. Casner. A Task-analytic Approach to the Automated Design of Graphic Presentations. *ACM Transactions on Graphics*, 10(2):111–151, April 1991.

[5] W. S. Cleveland and R. McGill. Graphical Perception: Theory, Experimentation, and Application to the Developments of Graphical Methods. *Journal of the American Statistical Association*, 79(387):531–554, September 1984.

[6] H. M. Culbertson and R. D. Powers. A study of graph comprehension difficulties. *AV Comm. Review*, 2(7):97–100, 1959.

[7] C. J. Date. *An Introduction to Database Systems*, volume I. Addison-Wesley, 4 edition, 1988.

[8] G. DeSanctis and S. L. Jarvenpaa. An investigation of the "tables versus graphs" controversy in a learning environment. In L. Gallegos, R. Welke, and J. Wetherbe, editors, *Proceedings of the 6^{th} international conference on information systems*, pages 134–144, December 1985.

[9] A.S.C. Ehrenberg. Rudiments of Numeracy. *Journal of the Royal Statistical Society*, 140(Part 3):277–297, 1977.

[10] Massimo Fasciano. *Génération intégrée de textes et de graphiques statistiques*. PhD thesis, Université de Montréal, 1996.

[11] M. Gagnon and G. Lapalme. From conceptual time to linguistic time. *Computational Linguistics*, 22(1):91–127, January 1996.

[12] Jock D. Mackinlay. *Automatic Design of Graphical Presentations*. PhD thesis, Computer Science Department, Stanford University, 1986.

[13] Jock D. Mackinlay. Automating the Design of Graphical Presentations of Relational Information. *ACM Transactions on Graphics*, 5(2):110–141, April 1986.

[14] V. Mittal, S. Roth, J. Moore, J. Mattis, and G. Carenini. Generating Explanatory Captions for Information Graphics. In *Proceedings of the 14^{th} International Joint conference on Artificial Intelligence (IJCAI-95)*, volume 2, pages 1276–1283, Montréal, Canada, August 1995.

[15] Steven F. Roth, Joe Mattis, and Xavier Mesnard. Graphics and natural language as components of automatic explanation. In Joseph W. Sullivan and Sherman W. Tyler, editors, *Intelligent User Interfaces*, Frontier Series, chapter 10. ACM Press, 1991.

[16] Howard. Schutz. An Evaluation of Formats for Graphic Trend Displays — Experiment II. *Human Factors*, 3:99–107, 1961.

[17] Edward R. Tufte. *The Visual Display of Quantitative Information*. Graphics Press, 1983.

[18] Edward R. Tufte. *Envisioning Information*. Graphics Press, 1990.

[19] P. Wright. The comprehension of tabulated information: some similarities between reading prose and reading tables. *NSPI Journal, XIX*, 8:25–29, 1980.

[20] Gene Zelazny. *Dites-le avec des graphiques*. InterÉditions, 1989.