Université de Montréal

Génération intégrée de textes et de graphiques statistiques

par
Massimo Fasciano

Département d'informatique et de recherche opérationnelle
Faculté des arts et des sciences

Thèse présentée à la Faculté des études supérieures
en vue de l'obtention du grade de
Philosophiæ Doctor (Ph.D.)
en informatique

Mai 1996

Université de Montréal
Faculté des études supérieures

Cette thèse intitulée:

Génération intégrée de textes et de graphiques statistiques

présentée par:

Massimo Fasciano

a été évaluée par un jury composé des personnes suivantes:

Jean Vaucher	(président-rapporteur)
Guy Lapalme	(directeur de recherche)
Richard Kittredge	(membre du jury)
Laurence Danlos	(examinateur externe)

Thèse acceptée le 28 mai 1996

Sommaire

Cette thèse s'intéresse au problème de la génération intégrée de textes et de graphiques dans le cadre des rapports statistiques. Les graphiques et le texte sont deux médias très différents. Heureusement, lorsque leur intégration est bien effectuée, ils se complètent à merveille: une image permet de montrer alors qu'un texte permet de décrire. Cette complémentarité est très importante dans les rapports statistiques à cause de la nature de l'information à présenter. En effet, ces rapports cherchent à présenter une information très dense tout en s'assurant que le lecteur en retient les points essentiels. L'union des deux médias facilite énormément la tâche car les graphiques nous donnent une vue d'ensemble alors que le texte permet de cibler les détails intéressants.

Le modèle présenté dans cette thèse s'attaque au problème en établissant un ensemble de critères essentiels à la génération automatique de rapports statistiques. Les critères étudiés sont l'intention du rédacteur, les types des variables à présenter, les relations entre ces variables, ainsi que les valeurs des données. Quelques-uns de ces facteurs ont déjà été examinés par d'autres chercheurs, en général de façon superficielle, mais jamais dans un cadre unifié. Grâce au modèle unifié que nous présentons dans cette thèse, nous avons réalisé un prototype de générateur de rapports nommé PostGraphe. Ce système génère des rapports statistiques contenant du texte et des graphiques en se servant d'une description annotée des données à présenter. Les annotations utilisées correspondent aux critères établis dans le modèle théorique. Ainsi, l'utilisateur peut spécifier au système ses intentions (comparaison, évolution, répartition, corrélation, ...), les types des données à présenter (temporelles, numériques,

ordonnées, ...) et les relations entre les données (ex: les profits en fonction des années). Le système examine aussi les valeurs des données (écarts entre les extrèmes, nombre de valeurs à présenter) pour effectuer ses choix. PostGraphe est écrit en Prolog et produit des rapports en LaTeX contenant des figures PostScript.

Table des matières

Table des figures

Liste des tableaux

Remerciements

Je tiens d'abord à remercier mon directeur de recherche, Guy Lapalme, dont les conseils judicieux m'ont été indispensables. Je le remercie pour sa patience, ses encouragements et sa disponibilité tout au long de ma recherche.

Merci aux étudiants et aux professeurs du laboratoire Incognito pour m'avoir fourni un environnement de travail stimulant.

Merci à Micheline Bélanger, dont le projet de maîtrise a permis de lancer la recherche présentée dans cette thèse. Merci à Michel Gagnon qui a rendu disponible son générateur de texte PréTexte et qui m'a aidé à l'adapter à mon système.

Merci à Jean Vaucher, Richard Kittredge et Laurence Danlos pour leurs commentaires sur cette thèse.

Pour leur support financier, je remercie le CRSNG, le FCAR et l'Université de Montréal.

Un gros merci à mes parents qui m'ont toujours encouragé à continuer mes études.

Enfin, merci à Leila pour son amour et ses encouragements, sans lesquels je n'aurais jamais pu terminer cette thèse.

À Leila,

Chapitre 1

Introduction

1.1 Problématique: communication personne-machine

Depuis ses débuts, l'informatique souffre d'un problème fondamental qui retarde beaucoup son intégration dans notre vie quotidienne: la difficulté de la communication personne-machine. Ce problème est vaste et présente des obstacles tant au niveau technique que théorique. La communication avec une machine est très différente de la communication entre humains, mais pour que l'ordinateur soit facilement accepté par les non-informaticiens, il faut la rendre aussi naturelle.

Ceci est déjà une difficulté majeure, mais la réalité est encore moins encourageante. En effet, la communication avec une machine implique souvent un transfert de plus grandes quantités d'information qu'entre humains et les moyens à utiliser doivent donc être encore plus sophistiqués que ceux que l'on utiliserait pour reproduire un échange d'information entre deux personnes.

Le problème a donné naissance à deux sujets de recherche complémentaires: la communication personne vers machine et la communication machine vers personne. Alors que la première s'intéresse presqu'exclusivement à analyser les méthodes d'expression d'un humain, la seconde s'intéresse en plus à trouver et à raffiner de nouvelles

méthodes d'expression plus efficaces que celles utilisées couramment par les humains.

La communication de personne vers machine implique des problèmes complexes comme la *reconnaissance du langage naturel* qui travaille aux niveaux morphologique, grammatical et conceptuel, *la reconnaissance de la parole* qui ajoute le traitement de l'intonation et du débit ainsi que le problème physique de la reconnaissance des sons, et *la reconnaissance de la gestuelle* qui permet de façon générale la désambiguïsation de la communication (pointer pour identifier un référent) surtout dans le contexte bien particulier des petits ordinateurs portables qui ne disposent que d'un crayon électronique (pas de clavier) et qui devraient donc dépendre de la reconnaissance de la parole ou de l'écriture pour recevoir des commandes de l'usager. En combinant des modes de communication (pointer au crayon en parlant, cliquer à l'aide de la souris et écrire) on rend le message plus clair et plus concis et surtout plus proche de la communication naturelle entre humains.

Ces problèmes se présentent aussi de façon inverse dans la communication de machine vers personne. On peut ainsi produire automatiquement du texte pour l'affichage à l'écran (génération de texte) ou lecture par la machine (génération de parole); on peut aussi se servir d'un pointeur mobile pour préciser l'information dont on parle. Une différence avec les problèmes d'analyse de la communication humaine se situe au niveau de l'ambiguïté. En effet, en analyse, on doit traiter l'ambiguïté présente dans le discours de personne à machine alors qu'en génération on cherche plutôt à produire un discours sans ambiguïté plutôt que d'imiter l'humain de façon exacte.

Une autre grande différence est la quantité d'information à présenter. Les ordinateurs ont des capacités de traitement et de stockage impressionnantes mais malheureusement, les méthodes permettant de communiquer à l'humain ces données ne sont pas aussi développées et beaucoup d'information n'est donc pas exploitée à fond. On retrouve fréquemment ce genre de problème dans le domaine des statistiques où on doit souvent recueillir et traiter des quantités énormes de données. L'ordinateur est

l'outil idéal pour ces opérations mais il faut ensuite un effort considérable pour en extraire les résultats.

Une solution intéressante au problème de la transmission d'informations brutes réside dans l'utilisation de graphiques. On dit qu'une image vaut mille mots. Cela traduit le grand pouvoir d'expression des graphiques. Un graphique nous donne accès sous une forme très compacte à beaucoup d'information que notre système visuel peut décoder efficacement. C'est pour cela que les statisticiens les utilisent pour accompagner leurs analyses. Depuis que les ordinateurs ont atteint un niveau de performance permettant l'affichage rapide de graphiques, un grand nombre d'outils de conception ont fait leur entrée sur le marché. Ces outils vont du simple programme de dessin à deux dimensions qui se compare à une tablette de dessin assistée par ordinateur, au système d'animation tridimensionnel qui fournit une assistance très poussée au concepteur. À mi-chemin entre les deux, on retrouve les programmes de traitement de données tabulaires (tableur) qui permet d'afficher à l'aide de figures des données brutes ainsi que les résultats de traitements divers sur ces données.

Cependant, de façon générale les graphiques ne sont pas la solution parfaite aux problèmes de communication personne-machine. En effet, la langue naturelle permet d'exprimer clairement des détails qu'un graphique ne saurait capturer que de façon implicite. Un graphique contient une foule d'informations cachées qu'un texte peut concrétiser. Par exemple, un texte peut faire ressortir des tendances dans les données, ou attirer l'attention du lecteur sur une partie importante d'un graphique. L'usage de pointeurs et de gestuelle, lorsque possible, est aussi très utile dans ce contexte.

Depuis quelques années, un certain nombre de chercheurs qui travaillaient en génération de texte ont commencé à s'intéresser aux graphiques. Puisque tout système de génération de texte se décompose en une phase de planification du contenu (quoi dire) et une phase de génération finale du texte (comment le dire), ces chercheurs se sont dit que si le résultat de la planification pouvait être suffisamment général, on

pourrait éventuellement s'en servir comme point de départ pour un générateur de graphiques.

1.2 Notre approche à ce problème

Notre recherche applique l'intégration de textes et de graphiques à la génération automatique de **rapports statistiques**. En partant de données brutes et d'une représentation de l'intention du rédacteur, nous cherchons à produire des graphiques et un texte qui les accompagne.

Les aspects qui nous intéressent en particulier sont la planification des messages et la modélisation de l'intention de l'usager. La planification est intéressante car c'est elle qui détermine le contenu du rapport. C'est aussi dans cette phase que l'on décide quelles informations seront exprimées dans les graphiques et lesquelles dans le texte. Pour profiter des avantages respectifs des deux médias, on doit effectuer une planification conjointe, c'est-à-dire que les choix pour le texte doivent influencer les choix effectués pour le graphique, et vice-versa.

Le but d'un rapport est de produire une synthèse organisée à partir de données ou d'expériences concrètes. Bien qu'en principe un rapport doive être objectif, cette synthèse est quelquefois biaisée. Le rapport présente le point de vue de la personne ou de l'organisation qui le rédige. Pour y arriver, il faut souvent "tordre" un peu les faits, c'est-à-dire mettre en évidence ceux qui supportent nos conclusions et en éclipser d'autres. Cette caractéristique correspond parfaitement à un de nos intérêts de recherche, soit la modélisation de l'intention du rédacteur. Puisque la plupart des rapports présentent des informations subjectives, ils permettent de tester nos théories sur l'intention. Les rapports qui nous intéressent, les rapports statistiques, ont une particularité intéressante: les données sont nombreuses et peu évocatrices lorsque présentées de façon brute. En effet, il s'agit souvent de colonnes de nombres comme

on retrouve dans les chiffriers (ou tableurs) électroniques. Sans une analyse statistique préalable pour faire ressortir les points importants ainsi qu'une organisation et présentation très efficaces, l'interprétation de ces données est difficile, même pour un lecteur expert. Ainsi, dans un tel sous-domaine, l'analyse des données présentée par un générateur automatique aura une très forte influence sur l'interprétation du lecteur. Il est donc très important de modéliser les buts du rédacteur et leur influence sur le lecteur.

1.3 Intégration des graphiques et du texte

Il est rarement approprié de produire un rapport statistique contenant uniquement des graphiques, car ceux-ci sont habituellement trop "secs". Pour obtenir une analyse un peu plus détaillée et surtout plus subjective, il faut nuancer avec un commentaire en langue naturelle. Dans un texte, on insistera sur les aspects des données qui semblent les plus importants ou les plus pertinents pour notre application. Par exemple, on dirigera l'attention du lecteur sur une chute rapide des profits d'une compagnie durant une année et on lui conseillera d'en vérifier la cause.

L'aspect subjectif de l'analyse textuelle est aussi très important. En effet, à partir d'un même graphique, plusieurs interprétations des données sont possibles, mais lorsqu'on lui associe un texte, on détermine une interprétation.

En résumé, un graphique est idéal pour donner une vue d'ensemble sur les données et les présenter de façon objective, alors qu'un texte permet de fournir des analyses plus subjectives et de s'attarder sur des points précis des données.

Il semble donc naturel de combiner ces deux médias et ainsi profiter de leurs avantages respectifs. Cette combinaison peut se faire de deux façons: une juxtaposition simple ou une intégration complète. La juxtaposition d'un texte et d'un graphique indépendants donne déjà des résultats intéressants [Bélanger, 1990], mais pour vraiment profiter des deux médias, il faut faire une intégration beaucoup plus forte. Une intégration forte nous permet entre autres d'éviter des redondances entre le texte et

le graphique, de mettre en relief dans le graphique les éléments dont parle le texte, et elle permet aussi au texte de faire référence à une partie du graphique.

1.4 L'intention du rédacteur

Beaucoup de systèmes "intelligents" fonctionnent sans aucune intervention de l'usager et prennent les décisions nécessaires à l'aide d'heuristiques pour produire le résultat voulu.

Malheureusement, ce genre de situation n'est pas toujours possible car la qualité des résultats peut dépendre fortement de facteurs externes au système comme les goûts d'un utilisateur et, de façon générale, du contexte.

Dans notre projet, nous tenons compte de l'intention du rédacteur du rapport. Ainsi, le processus de génération du texte et du graphique est fortement influencé par le message que l'usager désire transmettre. Voici un exemple qui justifie l'importance du message: la situation représentée au tableau 1.1 peut être décrite par plusieurs textes assez différents qui laissent des impressions presque contradictoires chez le lecteur:

Année	Profits ($)
1971	10 000 000
1972	11 000 000
1973	11 000 000
1974	18 000 000
1975	16 000 000
1976	16 000 000
1977	15 000 000
1978	19 000 000

TAB. 1.1 - *Variation des profits d'une compagnie entre 1971 et 1978.*

1. Les profits de la compagnie ont stagné entre 1971 et 1973, puis entre 1975 et 1977 après une grosse augmentation en 1974.

2. Les profits de la compagnie ont augmenté de 8 millions entre 1971 et 1974, puis de 3 millions entre 1975 et 1978 après une chute de 2 millions.

3. Les profits de la compagnie ont diminué de 3 millions entre 1974 et 1977.

4. Les profits de la compagnie ont subi 3 ans d'augmentation, 2 ans de stagnation et 2 ans de diminution entre 1971 et 1978.

Les trois premières descriptions insistent respectivement sur la stagnation, l'augmentation et la diminution. La dernière description énumère le nombre d'années d'augmentation, de diminution et de stagnation. C'est une sorte de récapitulation. Elle est assez trompeuse car elle ne procède pas par ordre chronologique.

Le message général utilisé dans les 4 cas est l'évolution (description des changements dans le temps). On voit bien qu'à partir des mêmes données et du même type général de message, le résultat peut être totalement différent selon le point de vue considéré. Qu'il s'agisse de simple sélection ou encore de subjectivité de la part du rédacteur, on doit en tenir compte dans le processus de génération.

Par exemple, pour produire un rapport vantant les mérites d'une entreprise, on opterait pour une comparaison de type *évolution–augmentation* (la seconde description) qui donne priorité à la mise en relief des augmentations de profits.

Une partie importante de notre projet consiste à établir une classification hiérarchique des différents types de messages. L'aspect hiérarchique de la classification reflètera des situations comme celle de l'exemple précédent, où un type de message objectif très général (l'évolution) donne lieu à plusieurs messages subjectifs.

1.5 Présentation de la thèse

Dans cette thèse, nous présentons d'autres travaux dans notre domaine et dans des domaines connexes tels l'analyse statistique, la génération de textes, la génération

de graphiques et leur génération intégrée.

Nous présentons ensuite les différentes formes d'expression à notre disposition pour construire un rapport: les types de graphiques, les formes de texte, les formes intermédiaires.

Puis, nous montrons les éléments tels les intentions et le typage des données, nécessaires pour choisir les graphiques et les descriptions textuelles d'un rapport.

Nous montrons ensuite comment passer de la théorie à la pratique, en insistant sur les compromis que nous avons effectués dans le cadre de la réalisation de notre prototype PostGraphe.

Finalement, nous décrivons les détails d'implantation plus techniques de PostGraphe.

Chapitre 2

Travaux antérieurs

Parmi les travaux antérieurs pertinents à notre recherche, on retrouve des travaux théoriques sur la conception de présentations visuelles ainsi que des systèmes informatiques capables de générer automatiquement ces présentations. Parmi ces systèmes, certains se limitent à la génération de texte ou de graphiques, tandis que d'autres essaient d'intégrer ces deux médias.

Dans ce chapitre, nous passerons en revue les travaux les plus pertinents dans ces domaines et nous soulignerons les points importants de chacun ainsi que leur influence sur notre projet.

2.1 Conception de présentations

Les études sur la conception de présentations portent surtout sur l'utilisation efficace de graphiques. Ceci souligne l'importance du rôle des graphiques dans toute communication. Les travaux qui nous ont particulièrement intéressé portent sur les graphiques statistiques (tartes, barres, courbes, ...). De nombreux travaux ont aussi porté sur l'utilisation d'autres types de graphiques comme les photos, les organigrammes et les icônes; cependant, ces études étant peu pertinentes à notre travail, nous n'en ferons pas une description détaillée ici.

Les trois aspects importants de la conception de graphiques sont la compréhension de leur structure interne, de leur fonction dans une présentation ainsi que leur perception par le lecteur.

2.1.1 La structure des graphiques

L'étude de Jacques Bertin [Bertin, 1983] sur la structure des graphiques est le travail de référence dans ce domaine. Sa recherche définit de façon exhaustive les éléments de base, ainsi que les propriétés d'un graphique.

Bertin présente le graphique comme un ensemble de correspondances entre un nombre fini de *variables*, appelées composantes du graphique, et un *invariant* qui établit le lien entre ces variables. Par exemple, dans une relation entre le prix d'une voiture et son année de fabrication, les variables sont le prix et l'année alors que l'invariant est la voiture.

De plus, un graphique est caractérisé par le nombre de composantes, la *longueur* de chaque composante, et leur *niveau d'organisation*. Bertin définit la longueur d'une composante comme le nombre de valeurs distinctes qu'elle peut prendre. Par exemple, la longueur d'une variable booléenne est 2, alors que la longueur d'une variable prenant ses valeurs dans l'ensemble {rouge,vert,bleu} est 3. Le niveau d'organisation d'une composante dépend de deux propriétés: l'ordre et la continuité. Si la variable est continue (et donc ordonnée) on dit qu'elle est *quantitative*. Si elle est discrète (non-continue) et ordonnée, on dit qu'elle est *ordinale*. Si elle n'est ni continue ni ordonnée, alors on la qualifie de *nominale* (ou qualitative).

Le graphique est aussi caractérisé par la nature des *marques* visibles qu'il contient. Dans un espace à deux dimensions, 8 propriétés distinguent les marques les unes des autres. Les deux premières, qu'on qualifie de positionnelles ou spatiales, définissent

la position d'une marque dans le plan (l'abscisse et l'ordonnée). Les 6 autres (taille, saturation, texture, couleur, orientation et forme) sont appelées propriétés rétiniennes.

Bertin définit la notion d'efficacité d'un graphique en fonction du temps de perception nécessaire à en décoder toute l'information pertinente. Il fournit un ensemble de règles de construction qui permettent de choisir les propriétés (spatiales ou rétiniennes) qui vont produire la représentation la plus efficace.

Son ouvrage représente une source importante de techniques qui aident à construire des règles de décision pour la sélection d'un graphique. En particulier sa notion de niveau d'organisation a influencé, entre autres, les travaux de MacKinlay [Mackinlay, 1986a]. En effet, MacKinlay a étendu le modèle de Bertin et a implanté le système de présentation APT. Ce système ainsi que les extensions au modèle seront décrites à la section 2.2.1. Notre recherche étant fortement inspirée de celle de MacKinlay, les travaux de Bertin en forment donc indirectement la base.

2.1.2 La fonction des graphiques

Les études sur la fonction des graphiques visent à conseiller le rédacteur sur le choix du graphique le plus approprié pour transmettre un message particulier.

Zelazny [Zelazny, 1972; Zelazny, 1989] présente une méthodologie de sélection d'un graphique. Cette méthodologie ne s'adresse pas à un expert en composition graphique, mais plutôt à quelqu'un qui cherche à présenter des données à l'aide d'un graphique.

Zelazny souligne l'importance du message en précisant que le facteur le plus important dans la sélection d'un graphique est son but ou la signification visée. Par conséquent, la première phase de sa méthodologie consiste à déterminer exactement ce que l'on veut transmettre. L'auteur suppose que les messages correspondent à un des 5 types de comparaison élémentaires: la **décomposition** (représente des pour-

centages), la **position** (classement d'éléments entre eux), l'**évolution** (changements dans le temps), la **répartition** (séparation en catégories) et la **corrélation** (relation entre variables).

Il fournit ensuite une table de décision qui permet de passer du type de comparaison à un des 5 types de graphiques qu'il considère (tarte, barres, colonnes, points, courbe). Lorsqu'il y a ambiguïté (*Ex:* évolution → colonnes ou courbe), on utilise d'autres critères simples pour faire notre choix (*Ex:* colonnes si le nombre d'échantillons est petit, courbe sinon).

D'autres études sur le sujet ont abouti a des classifications et terminologies légèrement différentes mais, selon nous, moins utiles. C'est le cas de celle de J. Bojin et M. Dunand [Bojin et Dunand, 1982], qui utilisent un ensemble de graphiques de base légèrement différent. Ils font en plus la distinction entre *graphiques de structure* (tarte et histogramme) et *graphiques de performance* (barres verticales, barres horizontales, doubles barres). Cette classification est intéressante mais nous trouvons que leur modèle ne fait pas bien la distinction entre la structure et la fonction d'un graphique. Par exemple, dans leur modèle, l'histogramme est un graphique de base alors que selon nous, histogramme est la fonction, alors que colonnes est la forme ou structure du graphique.

Parmi les travaux dans le domaine, nous avons retenu celui de Zelazny à cause de sa simplicité et sa cohérence. Il nous a fourni une des bases les plus importantes pour notre recherche. En effet, nous avons étendu et réorganisé son modèle pour tenir compte de plus de figures et de messages. Nous ne nous attarderons pas ici sur ces extensions car une grande partie du chapitre 4 y est consacrée.

2.1.3 La perception des graphiques

Lorsqu'on fait ici référence à la perception des graphiques, on parle de perception visuelle et non d'interprétation, comme à la section précédente qui s'intéressait au sens des graphiques. Parmi les travaux qui traitent de ces problèmes de perception, celui de William S. Cleveland [Cleveland, 1980] reste la référence dans le domaine. En effet, celui-ci présente très clairement les problèmes ainsi que leurs solutions en s'appuyant sur des expériences psychologiques.

Cleveland identifie un ensemble de facteurs liés à la perception humaine. Il montre que certains graphiques trompent notre système visuel, transmettant ainsi un message erroné, et que d'autres le sur-estiment, et échouent ainsi dans leur tâche d'illustrer les données.

Parmi les problèmes présentés, Cleveland remarque que:

– Il est très difficile d'évaluer la différence entre deux fonctions sur un graphique. Pour ce faire, on devrait mesurer la distance verticale entre les deux courbes (la différence entre les ordonnées pour une abscisse donnée). Cependant, on a tendance à faire cette mesure orthogonalement aux courbes. Ceci est particulièrement flagrant dans le cas de deux courbes avec des pentes très abruptes (voir figure 2.1).

  Comme solution à ce problème, l'auteur suggère l'utilisation d'une courbe de la différence des deux fonctions. Le problème sera ainsi évité puisqu'on aura rendu explicite la différence qui était difficile à percevoir (voir figure 2.2).

  En comparant ces deux graphiques, on se rend compte que nos yeux nous jouent des tours, surtout de 2 à 5 et 6 à 9 sur l'axe des x. Entre 2 et 5, la différence entre les deux courbes est constante, mais sur le premier graphique (figure 2.1), on a

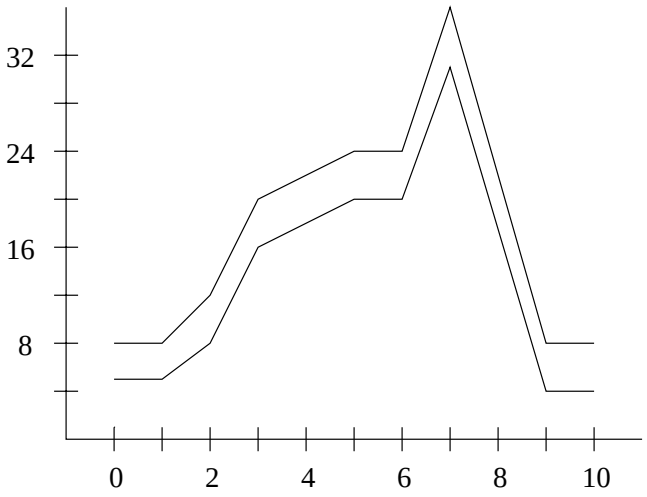


FIG. 2.1 - *Deux courbes superposées*

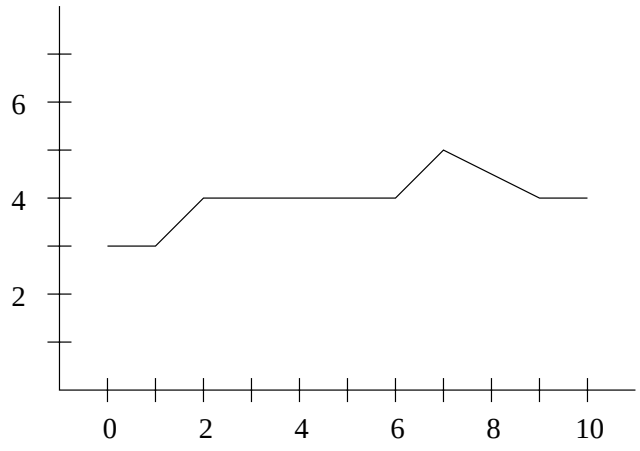


FIG. 2.2 - *Courbe de la différence entre les courbes de la figure 2.1*

l'impression qu'elle est petite entre 2 et 3 et qu'elle augmente entre 3 et 5. Par contre, entre 6 et 9, le premier graphique donne l'impression que la différence entre les deux courbes est constante alors qu'en fait elle augmente entre 6 et 7 et diminue entre 7 et 9.

– On a du mal à distinguer des angles ou des longueurs presqu'égaux à moins d'avoir un point de référence commun (voir figure 2.3).

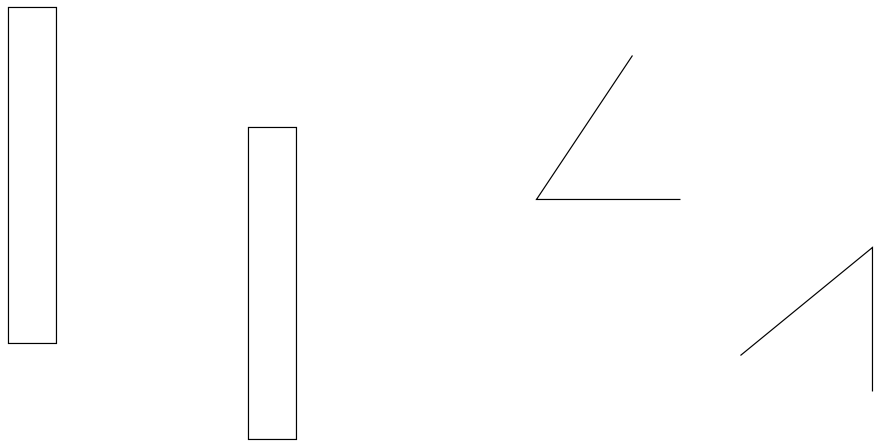


FIG. 2.3 - *Comparaison d'objets sans référence*

La solution triviale à ce problème est d'aligner les angles ou longueurs sur un point de référence commun (voir figure 2.4).

– L'œil évalue moins bien les variations de pente que les variations de distances ou longueurs. Les variations de pente sont importantes dans le cas de graphiques montrant une évolution. En effet, dans ce type de graphique, la pente représente le taux d'augmentation ou de diminution de la quantité étudiée. Dans des cas de pente très abrupte, comme à la figure 2.5, il est très difficile de percevoir les variations de la pente de la courbe.

FIG. 2.4 - *Comparaison des objets de la figure 2.3 avec référence*

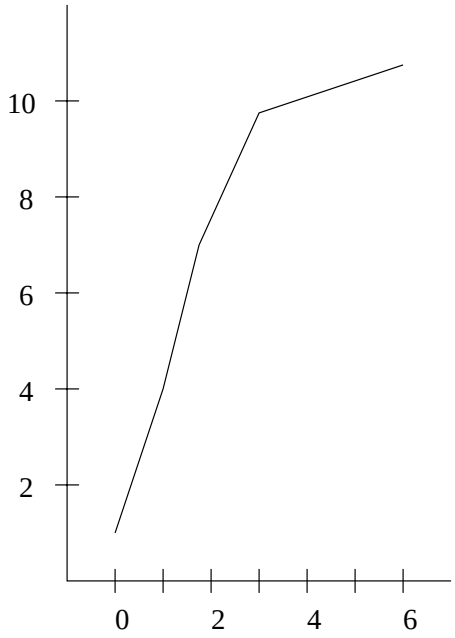


FIG. 2.5 - *Courbe à pente difficile à évaluer*

Dans cette situation, Cleveland propose d'utiliser un graphique de la pente: au lieu de faire une courbe montrant la variation d'une quantité, on fait une courbe qui montre la variation de sa pente. Ainsi, ce qui était une variation de pente devient une variation de distance/longueur, visuellement plus facile à percevoir. La figure 2.6 montre le résultat de cette méthode sur la courbe de la figure 2.5. On voit bien que ce type de graphique montre très clairement les différences de pente. En particulier, entre 0 et 3 sur l'axe des x, la pente de la courbe semble presque constante sur la figure 2.5 alors qu'on voit très bien ses variations à la figure 2.6.

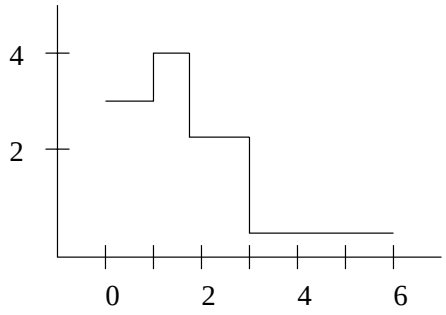


FIG. 2.6 - *Courbe de la variation de la pente de la courbe à la figure 2.5*

Ce qui rend l'ouvrage de Cleveland particulièrement intéressant, c'est qu'on y trouve des solutions à la plupart des problèmes présentés. En d'autres mots, il ne nous dit pas seulement quelles situations éviter mais aussi comment s'en sortir lorsqu'on se retrouve dans de telles situations.

Ses résultats nous ont été très utiles lors de la conception de notre système de réalisation de graphiques. En effet, nous évitons, autant que possible, de tomber dans les pièges qu'il a identifiés, en utilisant ses nombreuses suggestions.

2.2 Systèmes de génération automatique

On peut classifier les systèmes de génération automatique de présentations visuelles en trois catégories: ceux qui ne traitent que de la génération de graphiques, ceux qui ne traitent que de la génération de textes ainsi que ceux qui font l'intégration des deux médias.

2.2.1 Génération automatique de graphiques

Le système de génération de graphiques qui a eu le plus d'influence sur notre recherche est sans aucun doute APT[1]. Ce système, réalisé par MacKinlay dans le cadre de son Ph.D. [Mackinlay, 1986a], génère des graphiques statistiques de toute pièce plutôt que de choisir parmi une série de modèles pré-établis. Son approche permet de toujours trouver un graphique pour illustrer les données.

Il a identifié et classifié un ensemble de techniques d'*encodage graphique*. Une technique d'encodage graphique est une façon de représenter graphiquement la valeur d'une variable statistique. Par exemple, la valeur de la part de marché d'une compagnie peut être représentée graphiquement par la *position* d'un point le long d'un axe, par un *angle* dans un graphique en tarte, par la *superficie* d'un point (cercle) dans un graphique, etc... Ainsi, dans l'exemple précédent, *position*, *angle* et *superficie* sont des techniques d'encodage possibles pour la variable qui représente la part de marché.

Ces techniques sont un sur-ensemble des propriétés positionnelles et rétiniennes de Bertin [Bertin, 1983] décrites à la section 2.1.1. Notons que Mackinlay considère uniquement les variables visuelles de type nominal, ordinal ou quantitatif selon la nomenclature de Bertin.

Aux techniques d'encodage, il a associé des *langages graphiques primitifs*, qui sont des éléments de base permettant la construction de graphiques plus complexes. Par exemple, le langage graphique primitif associé à la technique d'encodage *position* est

1. A Presentation Tool

un *axe* (horizontal ou vertical).

Il a aussi déterminé un ensemble de *critères d'expressivité* pour les techniques d'encodage, c'est-à-dire quels types de relations et de variables pouvent être représentés par une technique.

Le tableau 2.1 illustre les critères d'expressivité pour les techniques d'encodage rétiniennes [Bertin, 1983].

	NOMINALE	ORDINALE	QUANTITATIVE
taille		√	√
saturation		√	√
texture	√	√	
couleur	√		
orientation	√		
forme	√		

TAB. 2.1 - *Critères d'expressivité pour les techniques d'encodage rétiniennes.*

Ainsi, une *forme* peut encoder une variable de type nominal mais pas une variable ordonnée (ordinale ou quantitative) car on ne perçoit pas les formes de façon ordonnée. Par contre, l'encodage par la *taille* du point a l'effet inverse. En effet, la taille est perçue de façon ordonnée (et continue) donc elle est adéquate pour les variables ordinales et quantitatives mais elle donne une fausse impression d'ordre si on l'utilise sur des variables nominales.

En s'inspirant de [Cleveland et McGill, 1984], Mackinlay définit ensuite une classification des techniques graphiques en fonction du type de variable utilisé. Le tableau 2.2 montre cette classification.

Lorsqu'il construit un graphique, APT essaie d'utiliser la technique d'encodage la

QUANTITATIVE	ORDINALE	NOMINALE
position	position	position
longueur	densité	couleur
angle	saturation	texture
pente	couleur	connexion
superficie	texture	inclusion
volume	connexion	densité
densité	inclusion	saturation
saturation	longueur	forme
couleur	angle	longueur
	pente	angle
	superficie	pente
	volume	superficie
		volume

TAB. 2.2 - *Classification par ordre décroissant d'efficacité des techniques d'encodage.*

plus efficace pour chaque variable à représenter. Ainsi, pour représenter une relation entre deux variables quantitatives, il va encoder chacune d'elles par sa position le long d'un axe, car la technique *position* est la plus efficace pour ce type de variable. Chaque couple de la relation sera donc représenté par un point dans le plan.

Malheureusement, certaines combinaisons de techniques d'encodage sont incompatibles. En effet, si on étend l'exemple précédent à 3 variables quantitatives, on se heurte à un problème: la troisième variable ne pourra pas être représentée par sa position le long d'un axe car les deux axes du plan ont déjà été attribués aux autres variables. Il faut donc représenter la troisième variable à l'aide d'une technique d'encodage moins efficace que la *position*.

Le système possède une base de connaissances lui permettant de déterminer quelles combinaisons de techniques d'encodage sont possibles. Ainsi, dans l'exemple précédent, la troisième variable serait encodée à l'aide de la *superficie* des points du graphique car les techniques d'encodage plus efficaces (position, longueur, angle et pente)

entrent en conflit avec le choix effectué pour les deux premières variables.

Notons que le système procède par *retour-arrière*, c'est-à-dire que lorsque le choix de la technique d'encodage d'une variable crée un conflit, il défait ce choix et en essaie un autre. Par conséquent, les variables qui sont traitées en premier bénéficient des techniques graphiques les plus efficaces et imposent des contraintes sur les variables suivantes. Pour une allocation plus efficace des techniques d'encodage, il faut donc ordonner nos variables par ordre décroissant d'importance.

La composition de graphiques permet d'étendre le nombre de méthodes applicables et ainsi augmenter l'efficacité du résultat. APT peut superposer deux graphiques si les axes horizontaux et verticaux sont compatibles. Par exemple, un graphique (Prix $\times$ Année) pour la voiture x peut être combiné à un autre graphique (Prix $\times$ Année) pour la voiture y et ainsi former un graphique (Prix $\times$ Année) pour les voitures x et y où les points des deux voitures seront différenciés par leur couleur ou autre technique rétinienne. APT peut aussi aligner deux graphiques ayant un axe compatible. Par exemple, un graphique (Prix $\times$ Consommation $\times$ Coût des réparations $\times$ Poids $\times$ Nom de la voiture) peut être décomposé en 4 graphiques à 2 variables alignés sur l'axe représentant les noms des voitures.

Comme le souligne Stephen M. Casner, auteur du système de réservations aériennes BOZ [Casner, 1991], un défaut majeur de APT est qu'il ignore totalement l'intention de l'usager, se basant uniquement sur des critères d'efficacité graphique. Comme nous verrons au chapitre 4, notre travail s'attarde à combler cette lacune de APT tout en essayant de garder ses points forts. Nous sommes donc capable de produire des présentations différentes à partir des mêmes données pour exprimer différentes intentions.

Une seconde lacune de APT, comblée aussi par notre recherche, est qu'il se limite aux variables nominales, ordinales et quantitatives, classification qui nous semble un

peu limitée, compte tenu de toute l'information qu'on peut tirer du type des variables. Notre modèle du typage des variables, décrit au chapitre 4, est beaucoup plus riche et permet donc des décisions plus précises quant au choix du graphique.

Notons que parmi les travaux antérieurs, on retrouve plusieurs autres types de générateurs de graphiques. Comme nous l'avons indiqué en début de chapitre, les systèmes qui nous intéressent directement sont liés d'une façon ou d'une autre au domaine des rapports statistiques. D'autres systèmes, comme par exemple APEX[2] [Feiner, 1987] de Steven Feiner, ont plutôt pour but l'explication de tâches et d'actions à l'aide de graphiques dans un univers à trois dimensions. Les gros systèmes multimédias actuels (COMET, WIP) (cf section 2.2.3) reposent sur cette même idée et y intègrent, entre autres, l'utilisation judicieuse du texte pour améliorer la qualité de la communication avec l'usager.

Après avoir présenté la génération automatique de graphiques, il nous semble important de mentionner un outil très utilisé pour la production semi-automatique de graphiques statistiques: les chiffriers électroniques (Lotus, Excel, ...). Ces outils permettent de conserver et de traiter des données tabulaires de façon automatique. Une des fonctions les plus utilisées de ces systèmes est la génération de graphiques à partir des données.

L'utilisation de ces programmes inclut généralement une interaction avec l'usager dans le but de connaître ses besoins. On pourrait donc être tenté d'affirmer qu'ils tiennent compte de l'intention de l'usager. Cependant, ces systèmes demandent souvent à l'usager de choisir un modèle de graphique dans une bibliothèque. Puisque l'usager doit fournir un modèle au système, celui-ci n'est guère plus qu'un bon module de réalisation interactif. Un bon système se doit de demander à l'usager ce qu'il compte faire avec les données et de déterminer quel modèle de graphique est le plus

2. Automated Pictorial EXplanations

approprié.

Le nombre de choix de graphiques est impressionnant et ce genre d'outil est très utile comme générateur de surface mais, pour quelqu'un qui ne sait pas comment produire un bon rapport, tous ces choix sont déroutants. Il serait donc préférable que ces systèmes réduisent le nombre de choix et posent des questions plus simples à l'usager en utilisant un bon modèle de ses intentions possibles.

2.2.2 Génération automatique de texte

Parmi les travaux en génération automatique de textes, nous allons présenter le système TEXT de McKeown qui forme la base de nombreux autres systèmes basés sur des schémas rhétoriques, entre autres le nôtre. Aussi, les systèmes ANA et LFS seront présentés car ils s'intéressent tous deux à la génération de rapports statistiques.

Le système TEXT

Le système TEXT [McKeown, 1985] de Kathleen McKeown est un système capable de répondre à des questions sur la structure d'une base de données. La génération du texte se fait en deux étapes. Tout d'abord, un module (la composante stratégique) s'occupe d'en déteminer le contenu ainsi que la structure. C'est ensuite le second module (la composante tactique) qui se charge de produire les phrases du texte en se servant bien-sûr des résultats du module précédent, ainsi que d'un lexique et d'une grammaire de la langue cible (anglais). La contribution scientifique de McKeown se situe cependant au niveau de la composante stratégique.

Des prédicats rhétoriques définissent le rôle d'une proposition dans le discours. *Équivalence*, *explication*, *analogie* et *alternatives* sont quelques exemples de prédicats rhétoriques. La proposition "Le bâtiment s'est écroulé *à cause de la violence du tremblement de terre*" est un exemple (ou instance) du prédicat *explication*. La

liste complète des prédicats rhétoriques du système est donnée dans [McKeown, 1985].

La structure du texte est déterminée par un schéma. Un schéma est un modèle qui détermine une combinaison des prédicats rhétoriques à utiliser pour obtenir l'effet désiré. Voici un exemple de schéma (*Identification*) utilisé par TEXT:

Identification
{Analogie/Constituants/Attributs/Renommage/Amplification}*
{Illustration particulière/Preuve}+
{Amplification/Analogie/Attributs}
{Illustration particulière/Preuve}

Notation: Le symbole * signifie 0 ou plus répétitions de l'entité qui le précède, le symbole + signifie 1 ou plus répétitions de l'entité qui le précède et le symbole / marque la disjonction (*ou*).

En appliquant ce schéma, le système pourrait arriver au texte suivant. Les prédicats rhétoriques utilisés sont indiqués entre crochets au début de la proposition concernée:

Saint-Sauveur (Québec) [Identification] Un important village de la région des Laurentides. [Attributs] Son économie est principalement axée sur les sports d'hiver, [Amplification] surtout le ski alpin. [Illustration particulière] Parmi ses centres de ski, on retrouve le Mont St-Sauveur, le Mont Habitant et le Mont Avila.

Dans [McKeown, 1985], McKeown définit de façon détaillée les 4 schémas utilisés par TEXT: identification, constituants (constituency), attributs (attributive) et contraste.

Pour régler les cas ambigus (plusieurs prédicats rhétoriques possibles), TEXT utilise

deux méthodes. D'abord, un sous-ensemble pertinent de la base de connaissances est déterminé à partir de la requête et est utilisé pour éliminer les propositions non-pertinentes.

Ensuite, TEXT regarde l'influence des prédicats possibles sur le "centre d'attention" (focus of attention) et tente de satisfaire les 4 critères suivants dans l'ordre indiqué:

1. Déplacement du centre d'attention sur un item introduit dans la proposition précédente.

2. Continuation sur le même sujet.

3. Retour à un sujet traité antérieurement.

4. Proposition avec le plus de liens implicites avec la proposition précédente.

Comme nous l'avons mentionné précédemment, la contribution majeure de TEXT à notre recherche est la notion de schéma rhétorique. Cette méthodologie est de nos jours considérée trop simple car les schémas sont statiques et doivent être développés pour chaque domaine discursif. D'autres méthodes plus élaborées ont été proposées: entre autres, on retrouve les relations de discours de la RST [Mann et Thompson, 1988] qui permettent de ne pas avoir un schéma figé mais plutôt de générer le texte dynamiquement. Nous avons cependant opté pour les schémas rhétoriques pour leur simplicité d'implantation de façon à concentrer nos efforts sur les graphiques et leur intégration au texte. De plus, dans un domaine aussi restreint que les rapports statistiques, les variations rhétoriques sont assez mineures et l'utilisation des schémas rhétoriques n'est pas un désavantage majeur.

Le système ANA

Le système ANA [Kukich, 1983] a été conçu pour générer des rapports boursiers de façon entièrement automatique à partir des cotes du Dow Jones. Le processus de génération se fait en 4 étapes:

Générateur de faits Ce module, écrit en *C*, s'occupe de calculer toutes les statistiques pertinentes à partir des données numériques brutes en provenance de New York. La sortie se fait sous forme de faits dans le langage *OPS-5*, utilisé par le reste du système.

Générateur de messages Ce module représente la partie sémantique du système. Il détermine le contenu du texte, c'est-à-dire les messages ou informations à transmettre. Les messages produits correspondent à une des dix classes considérées. Par exemple, on retrouve une classe pour les fluctuations intéressantes ainsi qu'une classe pour la situation du marché à la fermeture.

Organisateur du discours C'est à ce niveau qu'est déterminé l'ordre d'énonciation des différents messages du texte en provenance du générateur de messages. La méthode utilisée est figée car elle dépend uniquement du type de message à ordonner. Ce type de stratégie discursive ressemble beaucoup aux schémas rhétoriques de McKeown.

Module linguistique Ce dernier module est plus plus gros et le plus complexe du système. C'est lui qui détermine la forme exacte du texte (génération de surface). C'est ici que les choix grammaticaux et lexicaux sont effectués.

Quelques années plus tard, Chantal Contant a réalisé FRANA [Contant, 1985; Contant, 1986], une version française de ANA, en remplaçant le dernier module du système par un générateur francophone.

Le langage utilisé dans le domaine boursier est un sous-langage [Kittredge, 1982; Kittredge, 1983] du genre textuel des rapports statistiques. À ce titre, les systèmes ANA et FRANA nous ont permis d'avoir un bon point de départ et de référence pour le développement de nos schémas discursifs.

Le système LFS

LFS est un système de génération de rapports statistiques produisant des textes en français et en anglais. Son domaine d'application initial était les statistiques sur la *population active* au Canada [Iordanskaja et al., 1992]. Il a été ensuite adapté aux rapports sur la *vente au détail* ainsi que des rapports sur l'*indice des prix à la consommation*.

Le système supporte deux modes de fonctionnement:

1. Génération de rapports multi-paragraphes, sans graphiques.
2. Génération de courts résumés accompagnés d'illustrations[3]

Le module de réalisation de LFS est basé sur le modèle Sens-Texte [Mel'čuk et Pertsov, 1987]. Le modèle de planification, basé sur les graphes et/ou, a été étendu et implanté par B. Lavoie [Lavoie, 1994] dans le cadre de son projet de maîtrise.

LFS est capable de reproduire le style de rapports statistiques écrits par des humains dans des sous-domaines particuliers. Notre système s'intéresse plutôt à couvrir beaucoup de types de rapports statistiques, ce qui l'empêche de se concentrer sur le style. Les résultats obtenus sont donc plus génériques mais insatisfaisants pour des organisations particulières comme *Statistiques Canada* qui s'attendent à un format précis.

Selon [Lavoie, 1994], les lacunes de LFS sont:

1. Les plans textuels sont fortement reliés au domaine d'application. De plus, ils sont trop rigides, même à l'intérieur de ce domaine. Il suggère l'utilisation de la RST [Mann et Thompson, 1988].

3. L'emphase n'étant pas mise sur ce point dans LFS, nous le présentons dans la section sur les générateurs de texte.

2. Pas d'adaptation aux buts et style communicatifs du rédacteur. Il suggère une approche similaire à [Paris, 1991].

Notre approche souffre du même problème au niveau de la rigidité des plans mais elle s'adapte très bien à l'intention du rédacteur du rapport.

2.2.3 Intégration de graphiques et de texte

Les travaux en intégration de textes et de graphiques sont plus récents que ceux dans les domaines précédemment décrits. À l'Université de Montréal, les recherches dans le domaine ont commencé par le projet de maîtrise de Micheline Bélanger [Bélanger, 1990]. Notre travail est une suite logique de ce projet qui utilisait tel quel le modèle de MacKinlay pour la génération de graphiques et des modèles de phrases pré-établis pour le côté textuel. Profitant des expériences et de l'analyse du problème apportées par ce projet, nous avons continué la recherche surtout sur les fronts suivants:

- Extension du modèle de MacKinlay pour enrichir le système de types.
- Développement d'un modèle de l'intention du rédacteur permettant de produire des rapports correspondant à ses besoins.
- Intégration plus forte du texte et des graphiques en utilisant des coréférences. Dans le système de Bélanger, le contenu des graphiques et du texte générés sont pratiquement disjoints (le graphique présente les données tandis que le texte donne quelques statistiques importantes). Une intégration plus forte permet une meilleure cohésion entre les deux médias.

La majorité des autres systèmes dans le domaine s'inscrivent dans le cadre de projets multimédias à grande envergure, de plus en plus populaires. Les systèmes COMET, SAGE et WIP en sont trois bons exemples. Nous les présentons ici car il s'agit de projets fort intéressants qui nous ont inspiré dans nos recherches, même si leur apport a été moins direct que celui des travaux de MacKinlay ou Zelazny.

Le système COMET

Le système COMET[4] [Feiner et McKeown, 1991] a pour but la génération d'explications multimédias (graphiques 3D et textes) en réponse à des requêtes de l'usager. Le domaine de discours du système est l'entretien et la réparation d'un appareil radio militaire. En effet, COMET génère des séquences d'étapes expliquant comment corriger des problèmes de fonctionnement de cette radio.

Le contenu de l'explication est basé sur le type de requête, sur des informations contenues dans une série de bases de connaissances (sur les objets et actions impliqués, sur les diagnostiques et des connaissances géométriques pour la génération de graphiques 3D) ainsi que sur les connaissances (vocabulaire) et le but de l'utilisateur. La génération se fait en 6 étapes dont 2 en parallèle:

- Tout d'abord, un planificateur décide du contenu du message. Ceci est réalisé à l'aide de schémas rhétoriques de McKeown [McKeown et al., 1990] (voir la section 2.2.2).

- Ensuite, un module de coordination décide ce qui sera exprimé textuellement et ce qui sera exprimé graphiquement. Le graphique est utilisé pour mettre en relief des propriétés spatiales des objets (entourer la partie de la radio dont on va parler) alors que le texte sert à exprimer les actions abstraites et les relations. Par exemple, le texte pourrait exprimer la relation de causalité suivante: "appuyer sur le bouton rouge pour effacer l'affichage de la radio". Ce module sert aussi à s'assurer qu'il y a cohérence entre les deux médias. Ainsi, si on encadre une partie de la radio dans le dessin, on peut parler de la "partie encadrée" dans le texte.

- Les deux étapes suivantes se réalisent en parallèle: la génération du texte et du graphique.

    - Le générateur de texte fonctionne en deux phases: la sélection lexicale et la génération de phrases (d'abord, on identifie les mots, puis on les lie pour

4. COordinated Multimedia Explanation Testbed

former des phrases). Une caractéristique intéressante de ce générateur est qu'il tient compte des connaissances de l'utilisateur pour le choix de son vocabulaire (technique ou plus commun).

- Le générateur de graphiques [Seligmann et Feiner, 1991] nommé IBIS[5] permet de communiquer graphiquement les actions ou propriétés suivantes: la position, les propriétés physiques (taille, forme, couleur), l'état (pour un bouton), le changement d'état (appuyer sur un bouton) ainsi que des actions comme soulever, tirer, pousser et tourner.

- L'avant-dernière étape est l'organisation (ou mise en page) de l'information.

- Finalement l'impression (ou affichage) est effectuée.

Tout au long de ce processus, le système manipule et annote une hiérarchie d'entités que les auteurs appelent *logical forms* (formes logiques). Le tout est organisé en une grammaire fonctionnelle d'unification étendue que les auteurs appellent FUF[6] [Elhadad, 1991].

Le système SAGE

Le système SAGE [Roth et al., 1991] permet d'analyser des changements dans un système évolutif. Les modèles à dépendances hiérarchiques (en forme d'arbre) sont supportés ainsi que les modèles de style chiffrier électronique.

Par exemple, l'organisation des départements dans une entreprise entre dans la catégorie des dépendances hiérarchiques. Avec un tel modèle, le système pourrait retracer la cause de la diminution des profits de l'entreprise à un petit département au bas de la hiérarchie et illustrer le tout à l'aide d'un texte et d'un graphique (en forme d'arbre).

5. Intent-Based Illustration System

6. Functional Unification Formalism

Le premier module du système se charge de faire l'analyse des changements. Il détermine la portée de la requête et essaie de trouver les changements dans le système qui sont pertinents à celle-ci. C'est ce premier module qui détermine le contenu du message.

Ensuite un module graphique et un module textuel s'occupent de composer la présentation de façon intégrée:

Le module graphique Le traitement des graphiques de SAGE est inspiré du système APT de Mackinlay. Cependant, il lui apporte un certain nombre d'extensions:

- Notion de but. Le but est très facile à identifier, c'est la requête de l'usager.
- Types de données supplémentaires (*Ex:* types composés, données manquantes).
- Connaissances sur des procédés graphiques supplémentaires (*Ex:* le diagramme de Gantt qui décrit le début et la fin d'activités).
- Autres opérateurs de composition (combinaison) graphique (*Ex:* composer des arbres avec des graphiques statistiques).

Le module textuel Il s'inspire de TEXT [McKeown, 1985] pour l'organisation en schémas et de BLAH [Weiner, 1980] pour les explications. Une grande importance est donnée à la phase stratégique plutôt que tactique, c'est-à-dire à la structure et au contenu plutôt qu'à la génération de surface. Le module de génération de texte est donc très simplifié. Une dizaine de modèles rhétoriques sont présentés dont *changement*, *cause-effet*, *renforcement* et *dépendance*. Tout comme dans les explications de raisonnements pour systèmes experts [Weiner, 1980], SAGE doit faire de l'élaguage dans ses arbres de dépendances pour éviter de surcharger le texte.

Intégration texte/graphique Dans [Roth et al., 1991], les auteurs décrivent trois problèmes d'intégration qu'ils ont étudiés:

- Incompatibilité structurelle: Un texte et un graphique ne décomposent pas toujours l'information de la même façon. Ainsi, dans la représentation graphique d'un arbre, les fils sont près du père alors que dans une représentation textuelle, l'arbre peut être énuméré en pré-ordre, ce qui sépare le père de ses fils. Il est donc parfois difficile de faire le lien entre les deux.

- Cohésion: Lorsque le texte fait référence au graphique (*Ex:* ... situé dans le coin supérieur droit de la figure 3 ...), cela donne une impression de cohésion entre les deux. Pour que ce genre de référence soit possible, le générateur de texte doit avoir de l'information sur la structure du (ou des) graphiques produits (ou à produire).

- Redondance: Le texte énonce souvent des choses visibles dans le graphique (*Ex:* des valeurs numériques ou des dépendances du style père/fils pour un arbre). Un peu de redondance ne gène pas (bien au contraire), mais lorsqu'on en abuse, on surcharge inutilement la présentation.

Très récemment, Mittal et al. [Mittal et al., 1995] ont développé une extension à SAGE capable de générer de façon automatique des explications sur la structure d'un graphique. Par exemple, le texte peut servir à décrire la légende lorsque celle-ci est difficile à interpréter. Il faut remarquer que cette approche est très différente de l'utilisation conventionnelle du texte dans un système multimédias car elle décrit la forme plutôt que le contenu du graphique. Bien que très intéressante, l'utilité de cette application peut être mise en cause. En effet, si le générateur de graphiques avait choisi un graphique plus clair, il aurait été inutile d'expliquer comment l'interpréter. De façon générale, c'est le but visé par les systèmes de ce type. Cependant, il est important de noter que le système SAGE permet à l'utilisateur d'imposer ses préférences (bonnes ou mauvaises) sur le choix du graphique. Ainsi, la complexité du graphique n'est pas nécessairement attribuable à des mauvais choix au niveau du module de planification.

Le système WIP

Le système WIP a pour but la génération de présentations multimodales coordonnées à partir d'une représentation commune. Ces présentations sont adaptées à l'usager (ex: son expertise, sa langue de communication) ainsi qu'au contexte d'utilisation (ex: réparation rapide d'un appareil v.s. requête d'information générale sur ce même appareil). L'élaboration de la présentation est incrémentielle à tous les niveaux.

La détermination du contenu de la présentation est faite à l'aide des relations rhétoriques de la RST [Mann et Thompson, 1988]. Le choix du médium se fait dynamiquement lors de la planification du discours à l'aide de *stratégies de présentation*. Celles-ci indiquent la façon la plus efficace de présenter chaque type de message. Elles sont représentées par des plans indiquant, entre autres, son effet, ses conditions d'application, des sous-plans, ainsi que le médium de présentation.

La planification se fait en essayant de trouver des plans qui s'unifient avec un but global à réaliser. Les sous-plans sont ensuite traités récursivement jusqu'à l'obtention de stratégies de présentation primitives. La dernière étape consiste alors à générer le texte et le graphique à l'aide de modules de réalisation de bas niveau.

L'approche et les résultats de WIP, tout comme ceux de COMET, sont très intéressants et assez généraux dans le domaine des explications multimédias. Cependant, ce type de système n'a pas les mêmes buts qu'un système qui, comme le nôtre, produit des graphiques statistiques. Dans le cas des explications graphiques de procédures, le système dispose au préalable d'un croquis annoté de l'objet à décrire et il décide quelles composantes doivent être mises en valeur ou élidées et sous quel point de vue elles doivent être visualisées de façon à satisfaire le but de la présentation. Par contre, notre générateur de rapports part de données brutes et doit composer son graphique de toutes pièces. Non seulement, il faut décider quels éléments présenter (échelles, secteurs de tarte, légendes, ...) mais aussi la meilleure façon de dessiner ces éléments

et de les agencer de façon à rendre le résultat aussi clair que possible.

2.3 Résumé

Dans ce chapitre, nous avons présenté les travaux les plus influents sur notre recherche. Nous avons commencé par une description des travaux en conception de présentations visuelles. Ces travaux se divisent en 3 catégories: la structure des graphiques, la fonction des graphiques et la perception des graphiques.

Ensuite, nous avons passé en revue des travaux en génération automatique de présentations visuelles. Les 3 types de travaux présentés sont ceux qui ne traitent que de la génération de graphiques, ceux qui ne traitent que de la génération de textes ainsi que ceux qui font l'intégration des deux médias.

Chapitre 3

Formes d'expression

Pour présenter efficacement l'information dans un rapport, il faut être capable de se servir des formes d'expression appropriées et de bien les intégrer. La réalisation des formes d'expression est la dernière étape dans le processus de génération d'un rapport mais pour bien comprendre les étapes précédentes, surtout celle qui s'occupe du choix, il faut connaître les forces et faiblesses de chaque forme.

Parmi les formes d'expression pertinentes à un rapport statistique, on retrouve deux grandes catégories: les figures et le texte.

Les figures sont des éléments d'illustration plus ou moins flottants, c'est-à-dire que leur position dans le rapport n'est pas totalement fixe. Une figure doit se trouver près de l'information (textuelle ou autre) à laquelle elle est associée mais sa position exacte reste non-précisée: par exemple, elle peut se trouver en haut ou en bas d'une page de texte, ou encore seule sur une page avoisinante. Cette liberté de positionnement est très importante pour la mise en page d'un document. En effet, une figure est parfois trop grande pour être placée à un endroit particulier sans dépasser les limites de la page ou sans y laisser une grande zône vide. Le positionnement flexible des figures permet donc d'avoir un texte mieux distribué sur les pages du médium de présentation. Il faut noter en passant que si le médium de présentation n'est pas

paginé (ex: un rouleau de papier continu), il n'y a aucune raison de faire flotter les figures. Le flottement des figures, bien que très utile, est parfois dangereux car si la figure est trop éloignée de sa référence, elle perd son utilité car elle ne sera pas regardée au bon moment.

Dans cette grande catégorie des figures, on retrouve les tableaux, qui sont très importants même s'ils présentent l'information sous sa forme la plus brute, et aussi les graphiques statistiques (barres, colonnes, tartes, etc...) qui sont moins explicites que les tableaux mais très utiles pour présenter une vue d'ensemble.

Le texte d'un rapport, en plus de présenter des descriptions et analyses des données, représente la "colle" qui lie les figures (par des transitions, des références, etc...). Dans ce chapitre, nous verrons différentes façons d'utiliser le texte pour renforcer le message présenté dans les figures.

La distinction entre texte et graphique n'est pas toujours aussi évidente qu'on pourrait le croire. En effet, on utilise beaucoup de techniques graphiques pour améliorer l'apparence du texte. Ces techniques peuvent agir sur les caractères (gras, italique), sur les groupes de mots (guillemets, parenthèses), sur l'organisation de l'information en blocs (paragraphes, listes) ainsi que sur la mise en page globale (parges, espacement). Par exemple, une liste énumérée de phrases s'intègre de façon parfaite dans un texte mais elle présente la notion de position verticale qu'on retrouve dans beaucoup de graphiques statistiques. Nous présenterons ces techniques de mise en page à la fin de la section sur le texte.

3.1 Les figures

Parmi les figures qui nous intéressent, on retrouve les tableaux et les graphiques statistiques. Le point commun entre ces deux classes, c'est qu'elles n'imposent pas une continuité avec ce qui précède et ce qui suit. En général, elles sont ancrées au document à l'aide de références dans un texte. Les figures deviennent donc des objets

indépendants (flottants) qui peuvent être présentés séparément ou à la fin d'un document sans en changer le sens. Cependant, il faut noter que si une figure apparaît avant sa référence ou trop loin après, elle risque d'être regardée hors-contexte, ce qui va affaiblir son rôle dans la transmission du message.

3.1.1 Les tableaux

Étant donnée la simplicité apparente des tableaux et leur similitude avec les données brutes d'une base relationnelle, on pourrait croire que leur utilisation est simple et bien comprise. Ce n'est malheureusement pas le cas. Selon Ehrenberg [Ehrenberg, 1977], la plupart des tableaux sont mal conçus et mal présentés et nécessitent un effort de compréhension considérable même pour un expert. Cependant, en suivant quelques règles simples, on peut réussir à produire des tableaux clairs et expressifs. Ces règles sont d'autant plus intéressantes qu'une fois qu'on les a comprises dans le cas simple des tableaux, elles peuvent être appliquées à des graphiques statistiques plus complexes.

Tout d'abord, il faut se demander comment identifier un bon tableau. Selon P. Wright [Wright, 1980], 3 processus déterminent l'efficacité d'un tableau:

1. Compréhension. Le lecteur comprend-il l'organisation du tableau?

2. Recherche. Le lecteur sait-il où rechercher les réponses à ses questions? Cette recherche se fait-elle rapidement?

3. Interprétation. Le lecteur sait-il interpréter les réponses à ses questions? Le tableau contient-il tout ce qu'il faut pour cette interprétation ou faut-il consulter et comparer avec d'autres tableaux?

Le premier point (compréhension) est important, mais il devient primordial dans le cas des graphiques statistiques où la structure peut être très complexe. Le second (recherche) est certainement le moins respecté et c'est de là que viennent la plupart des problèmes de conception des tableaux. Le troisième (interprétation) est rarement

considéré mais dans le contexte des rapports multimédias, il est inévitable de se demander comment se servir du texte pour faciliter l'interprétation des figures.

On doit donc se demander comment on peut répondre à ces 3 critères, surtout le second. Selon Ehrenberg, un bon tableau doit mettre en évidence les tendances et exceptions pour quelqu'un qui sait les interpréter. Pour cela, il propose 4 règles de conception:

1. Arrondir fortement les nombres pour faciliter les comparaisons.

2. Inclure des moyennes: en plus de résumer, elles permettent de mieux visualiser les écarts.

3. Les comparaisons de nombres sont plus faciles en colonne qu'en ligne.

4. Si possible, ordonner l'information en mettant les plus grandes valeurs en haut. Le positionnement des grandes valeurs en haut facilite l'arithmétique mentale. L'ordre facilite les comparaisons.

Les tableaux 3.1 et 3.2 sont des exemples illustrant les 4 règles de Ehrenberg. Le premier tableau montre ce qu'on obtient lorsqu'on n'obéit pas aux règles. À part la lecture de valeurs, on ne peut pas vraiment en tirer grand chose. Le second essaie, autant que possible, de respecter les 4 règles.

Compagnie	A	B	C
Chiffre d'affaires (en M $)	20.053	100.1	80.2387
Employés	190.56	450	170.1

TAB. 3.1 - *Tableau ignorant les 4 règles de Ehrenberg*

On voit bien que l'arrondi des nombres en facilite la comparaison. La différence entre 100 et 20 est plus frappante qu'entre 100.1 et 20.053. Les chiffres superflus ont

Compagnie	Chiffre d'affaires (en Millions de $)	Employés
B	100	450
C	80	170
A	20	190
Moyenne	65	270

TAB. 3.2 - *Tableau respectant les 4 règles de Ehrenberg*

tendance à grossir le nombre. Ceci entre dans le point 2 de Wright.

À l'aide de la moyenne, on a un bon résumé du tableau pour comparer avec d'autres tableaux similaires (le point 3 de Wright). On a aussi une référence interne au tableau pour savoir quels éléments sont exceptionnels. Par exemple, on voit tout de suite que le chiffre d'affaires de A est nettement en dessous des autres. Dans un plus gros tableau, ce dernier point devient encore plus important et vient renforcer le point 2 de Wright.

Il suffit de regarder rapidement les deux tableaux pour se convaincre que la présentation des nombres en colonnes facilite leur comparaison. De plus, l'ordre des nombres de la première colonne (du plus grand au plus petit) simplifie encore plus la tâche. En effet, nous lisons de haut en bas et nous comparons en soustrayant donc il est très utile de voir la plus grande valeur avant la plus petite (pour comparer 20 et 100, on fait 100-20=80 ou encore 100/20=5). Il faut noter que nous n'avons pas pu respecter le règle 4 pour la seconde colonne. Il fallait choisir une des deux et celle de gauche est la plus importante (là encore, c'est celle qu'on lit en premier).

Par rapport aux graphiques statistiques présentés dans la prochaine section, les tableaux sont très utiles dans des situations où il est important de pouvoir lire avec précision toutes les valeurs.

Puisqu'on affiche presque toujours l'information en deux dimensions, on retrouve deux grandes catégories de tableaux: les tableaux linéaires (1 dimension) et les tableaux rectangulaires (2 dimensions). Cette terminologie est un peu trompeuse puisque tous les tableaux sont rectangulaires en apparence, mais lorsqu'on regarde l'organisation des données à l'intérieur du cadre du tableau, on se rend compte que parfois les données sont présentées en ligne et parfois en rectangle. Les tableaux 3.3 et 3.4 illustrent cette distinction:

Départ de	Temps vers destination					
	A	B	C	D	E	F
A	0	2	6.2	2.1	1	4
B	2	0	4	7	1.7	2
C	5.8	3	0	2	6	5.5
D	2	5	3	0	1	4
E	0.7	2	5	2	0	6
F	5	3	5	4	5	0

TAB. 3.3 - *Tableau à 2 dimensions*

Notez que lorsque le tableau rectangulaire utilise N lignes et M colonnes, le tableau linéaire peut occuper jusqu'à (N-1)*(M-1) lignes (ou colonnes s'il est présenté dans l'autre sens). Dans l'exemple ci-dessus, la version à deux dimensions occupe 7 lignes et 7 colonnes de données alors que la version à 1 dimension occupe 36 lignes et 3 colonnes de données.

Le tableau rectangulaire semble beaucoup plus adapté à une page de rapport puisqu'il la remplit de façon plus équilibrée. Cependant, dans certaines situations, le tableau linéaire est plus efficace. La première situation se présente lorsqu'on veut absolument économiser de l'espace dans une dimension. Par exemple, si on doit se

Départ	Destination	Temps
A	A	0
A	B	2
A	C	6.2
	⋮	
F	B	3
F	C	5
F	D	4
F	E	5
F	F	0

TAB. 3.4 - *Tableau à 1 dimension*

limiter à 5 lignes, on ne peut pas utiliser le tableau rectangulaire de l'exemple car celui-ci occupe 7 lignes. Par contre, on peut utiliser la transposée du tableau linéaire de l'exemple (3 lignes et 36 colonnes) à condition que ses 36 colonnes ne dépassent pas la largeur de la page. Cette situation est rare et l'utilisation du tableau linéaire dans ce contexte risque d'aller à l'encontre des 3 principes de Wright.

La situation la plus courante dans laquelle on utilise le tableau linéaire est celle du remplissage non-optimal. En effet, le tableau à deux dimensions devient très lourd lorsqu'il est presque vide. Un tableau est trop vide lorsque beaucoup de cases n'ont pas de valeur. Dans l'exemple précédent, on pourrait éliminer les zéros de la diagonale. Si la durée du voyage entre deux points était symétrique, on viderait aussi toute la partie sous cette diagonale. Si certains points n'étaient pas reliés directement (comme c'est le cas pour les vols aériens), on aurait aussi d'autres vides. Ainsi, on pourrait se retrouver avec un tableau qui contient 49 cases mais dont seulement une petite partie est remplie. Lorsque le ratio entre cases vides et cases remplies diminue beaucoup, il faut penser au tableau linéaire car celui-ci n'occupe que le nombre de cases utiles. À cause de sa géométrie, les cases vides n'ont pas besoin d'être affichées (voir tableaux

3.5 et 3.6). Le cas où on a plusieurs valeurs pour une case pose aussi des problèmes pour un tableau rectangulaire.

Départ de	Temps vers destination					
	A	B	C	D	E	F
A		2		2.1		
B			4			
C				2		
D						4
E						6
F						

TAB. 3.5 - *Tableau à 2 dimensions (relation à faible remplissage)*

Départ	A	A	B	C	D	E
Destination	B	D	C	D	F	F
Temps	2	2.1	4	2	4	6

TAB. 3.6 - *Tableau à 1 dimension (relation à faible remplissage)*

Lorsqu'on choisit le bon type de tableau, on minimise l'espace utilisé et on facilite la recherche dans le tableau (point 2 de Wright).

Au niveau des données, les dépendances fonctionnelles nous donnent des informations très importantes sur le type de tableau à choisir. Les dépendances fonctionnelles sont des relations entre variables indiquant que pour une valeur d'un sous-ensemble de variables (la *clé*), on aura une et une seule valeur pour le reste des variables. Par exemple, dans les tableaux ci-dessus, pour chaque paire de points *départ/destination*, on a une et une seule *durée*. On dit alors que *départ/destination* est la *clé* de la relation et qu'il y a dépendance fonctionnelle entre la durée et la clé. Dans un cas de

dépendance, on utilise un tableau rectangulaire.

Dans certains cas, il n'y a pas dépendance fonctionnelle. Cela implique que pour une valeur de la clé on peut avoir soit aucune soit plusieurs valeurs pour les autres variables. Dans ce cas, il faut regarder le ratio de remplissage pour déterminer quel tableau est plus approprié.

3.1.2 Les graphiques statistiques

Après avoir étudié la première grande catégorie de figures, les tableaux, nous allons maintenant examiner les propriétés et règles d'utilisation des graphiques statistiques. Il y a plusieurs sortes de graphiques statistiques mais ceux-ci partagent une propriété fondamentale: ils montrent une vue d'ensemble avant de montrer les détails.

Louis Timbal-Duclaux [Timbal-Duclaux, 1990] présente cette propriété pour montrer la différence entre le langage sonore et le langage visuel. Il souligne que le "langage sonore procède par addition et synthèse de sons élémentaires (des parties vers le tout)" alors que "le langage visuel fait l'inverse: c'est d'abord la tonalité qui est perçue, avant les détails qui la composent (c'est un fonctionnement *cerveau droit* et non *cerveau gauche*)".

La perception d'un texte se fait de façon presqu'identique à celle d'une conversation orale: on part des mots, on les assemble en phrases, en paragraphes, etc... Cependant, la structure globale d'un texte (paragraphes, sections, ...) est visible avant les mots et peut donner de l'information d'ensemble. Le poids de cette information, bien que non-négligeable, est cependant beaucoup plus faible que les détails apportés par les mots.

Dans le cas d'un tableau, on a une situation similaire au texte. En effet, le tableau a une structure d'ensemble, mais son rôle principal est de présenter les valeurs (nombres ou texte) se trouvant dans ses cases. Avant la lecture des valeurs, on n'a pas

vraiment d'idée sur son contenu. C'est pour cela que la règle 2 de Ehrenberg (inclure des moyennes) est très importante pour créer de façon artificielle cette vue d'ensemble.

Les graphiques statistiques ont pour rôle premier de permettre un survol rapide des données. Parfois, ils permettent ensuite une lecture précise des détails, mais contrairement au tableau, là n'est pas leur rôle principal. Cette grande force peut devenir une faiblesse lorsque le point 1 de Wright n'est pas respecté: en effet, si la structure globale d'un graphique est mal comprise, il perd son efficacité et peut même porter le lecteur à mal interpréter les tendances des données. L'étude de ce dernier point nous a aidé à modéliser la transmission de messages subjectifs (imposer l'interprétation du rédacteur au lecteur même si celle-ci ne découle pas naturellement des données).

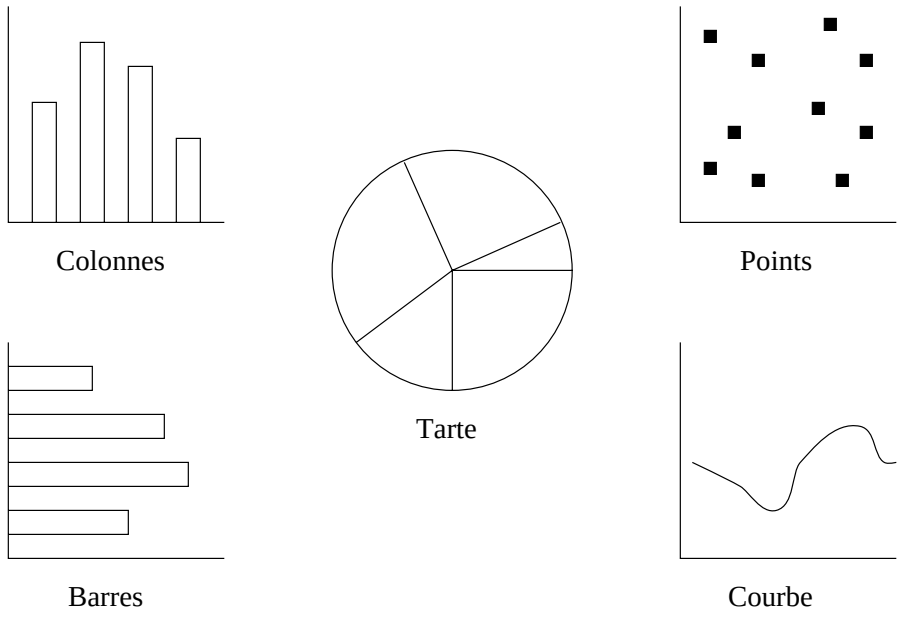


FIG. 3.1 - *Les 5 graphiques de base selon Zelazny*

Les graphiques statistiques de base de notre modèle sont ceux de Zelazny [Zelazny, 1989] (figure 3.1). Notre modèle de décision fait la distinction entre la forme (apparence) et la fonction d'un graphique. Nous avons choisi d'organiser cette section autour de la forme des graphiques et de présenter les différentes fonctions associées

à chaque forme. D'autres modèles, comme celui de J. Bojin et M. Dunand [Bojin et Dunand, 1982], confondent un peu forme et fonction. Leur modèle, en plus d'utiliser des graphiques de base légèrement différents, fait la distinction entre *graphiques de structure* (tarte et histogramme) et *graphiques de performance* (barres verticales, barres horizontales, doubles barres). Cette distinction est purement fonctionelle car les graphiques *histogramme* et *barres verticales* sont deux variations simples d'une même forme (*colonnes* dans notre modèle).

Nous allons commencer notre présentation par les graphiques les plus utilisés: les barres et les colonnes. Leur structure est similaire mais leur orientation leur donne des propriétés très différentes.

Les barres

Les graphiques en barres (figure 3.2) sont idéaux pour montrer des comparaisons de valeurs. L'orientation verticale de l'axe principal rend ce type de graphique inutilisable pour des données temporelles pour lesquelles il est recommandé d'utiliser un axe horizontal. Quelques contraintes d'utilisation: X doit être fonction de Y (il ne faut pas avoir 2 X pour un même Y). Ce type de graphique supporte 2 variables et il est difficile de l'étendre à 3 variables ou plus. Au lieu d'un graphique en barres à plus de 2 variables, il est préférable pour des raisons d'esthétique d'utiliser les colonnes.

Les barres présentent un avantage dû à leur orientation. En effet, selon Graham [Graham, 1937], le lecteur a tendance à surestimer la longueur d'une barre verticale (colonne). Puisque dans le graphique en barres, les rectangles sont orientés horizontalement, ce problème est évité.

Au niveau esthétique, de ce type de graphique est intéressant dans le contexte des comparaisons. En effet, les objets comparés ont souvent des noms assez longs (ex: des compagnies) et il est beaucoup plus facile de les écrire horizontalement sur ce type

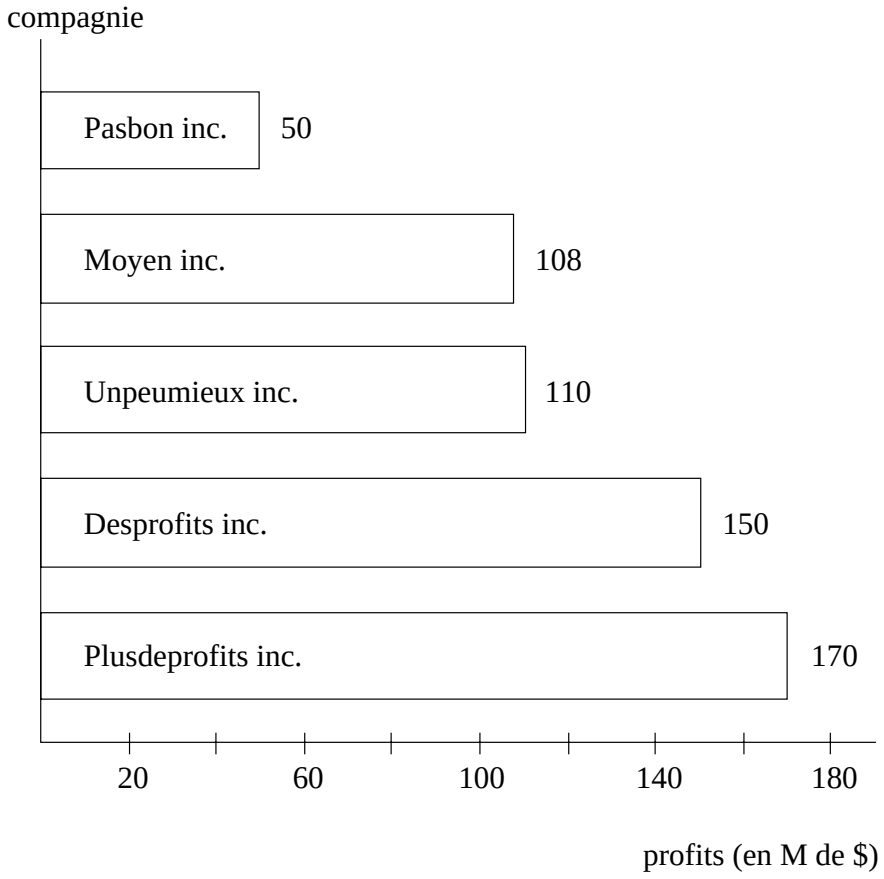


FIG. 3.2 - *Graphique en barres*

de graphique que verticalement sur un graphique en colonnes (figure 3.3). Les noms peuvent être placés à gauche de l'axe vertical ou à l'intérieur même des barres.

Pour insister sur le message de comparaison, on obéit à la règle 4 de Ehrenberg. Les entités à comparer sont donc triées non-pas par ordre alphabétique mais selon les valeurs à comparer. Ainsi, la plus "performante" [Bojin et Dunand, 1982], sera toujours plus près du point focal du graphique, c'est-à-dire l'origine de l'axe. Cela donne aussi une apparence plus équilibrée au graphique (si on plaçait la plus longue barre en haut, on aurait l'impression qu'il est sur le point de tomber). L'ajout de valeurs au bout des barres facilite l'interprétation des données [DeSanctis et Jarvenpaa, 1985]. Cela est dû en partie au fait que le lecteur ne lit pas les échelles [Vernon, 1946].

Les colonnes

Les graphiques en colonnes (figure 3.3) sont idéaux pour montrer des évolutions, des répartitions (histogrammes) et aussi pour des comparaisons à plus de 2 variables. L'orientation horizontale de l'axe principal rend ce type de graphique très efficace pour des données temporelles. Quelques contraintes d'utilisation: Y doit être fonction de X (il ne faut pas avoir 2 Y pour un même X) et le nombre de colonnes ne doit pas dépasser la douzaine. Si le nombre de colonnes est trop élevé, il est préférable de choisir une courbe. Comme pour les barres, l'ajout de valeurs au bout des colonnes facilite l'interprétation des données.

Dans le modèle de Bojin et Dunand, ce que nous appelons les colonnes se divise en deux types distincts de graphiques: l'histogramme et les barres verticales. La seule différence entre les deux est esthétique: dans l'histogramme, les colonnes sont toujours juxtaposées alors que pour les barres verticales, elles ne sont pas adjacentes. Cette différence de style est importante et doit être traitée comme telle dans le modèle, mais elle ne mérite pas une nouvelle catégorie de graphique. La juxtaposition des colonnes donne une impression de continuité à l'histogramme.

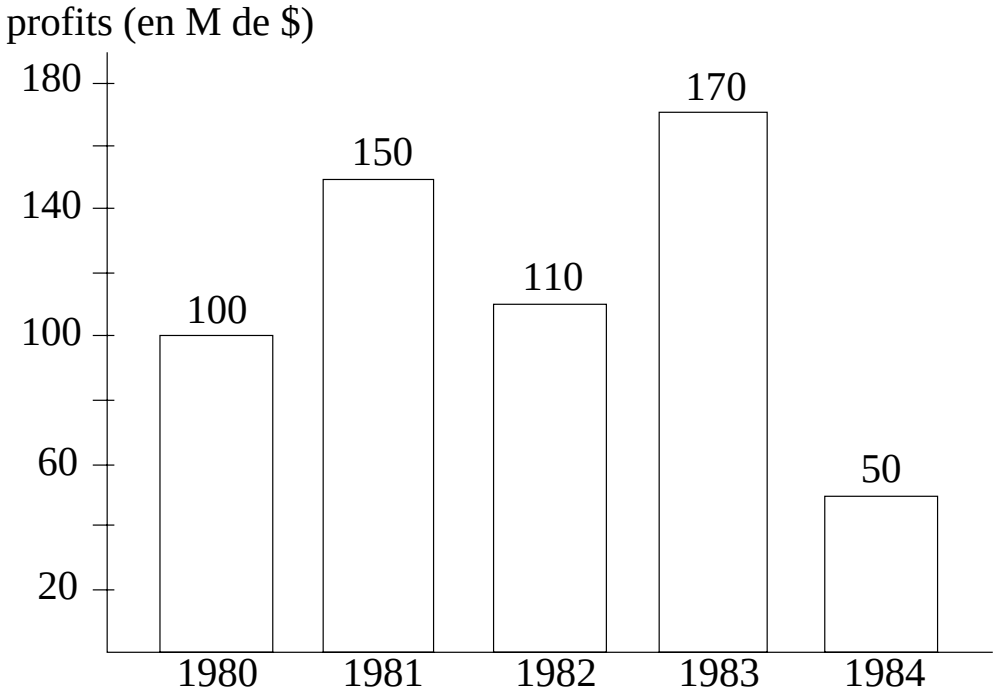


FIG. 3.3 - *Graphique en colonnes*

Les formes les plus utilisées de graphiques en colonnes sont présentées dans les figures suivantes: la figure 3.3 est un graphique en colonnes simples à 2 variables. Dans la figure 3.4, on utilise des colonnes multiples pour présenter des variables supplémentaires. La figure 3.5 est un histogramme (colonnes collées).

La figure 3.6 montre qu'il est possible d'ajouter des variables aux graphiques en colonnes sans avoir recours à des colonnes multiples. Dans cette figure, la largeur des colonnes ainsi que leur intensité (ton de gris) sont utilisées pour encoder une variable supplémentaire. En général, les colonnes multiples (figure 3.4) sont préférées par rapport à ce type d'encodage car leur lecture est beaucoup plus naturelle. Cependant, dans des situations exceptionnelles où les variables à encoder sont une largeur et une intensité ce type d'encodage peut être intéressant à cause du lien direct avec l'entité à encoder.

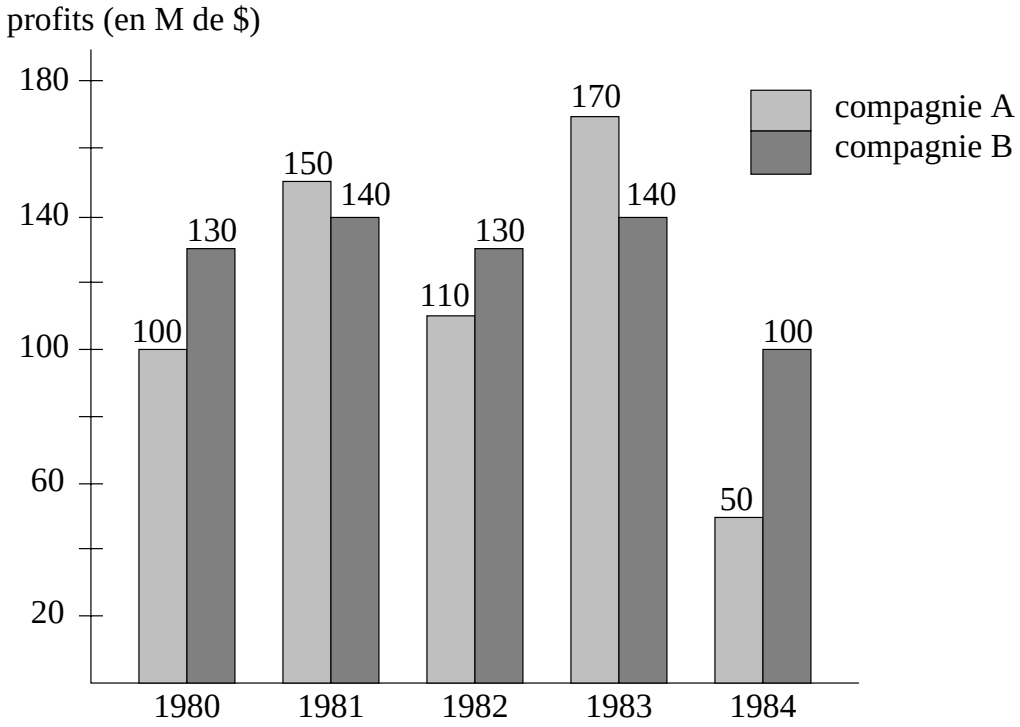


FIG. 3.4 - *Graphique en colonnes multiples*

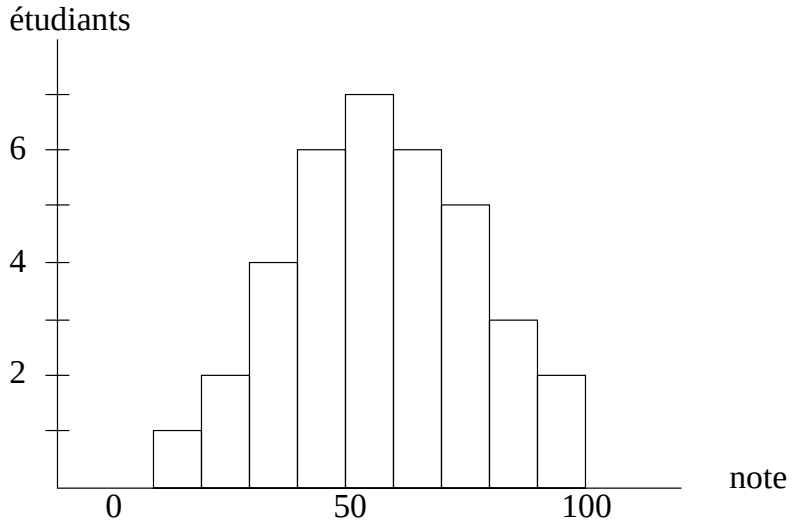


FIG. 3.5 - *Graphique en colonnes de type histogramme*

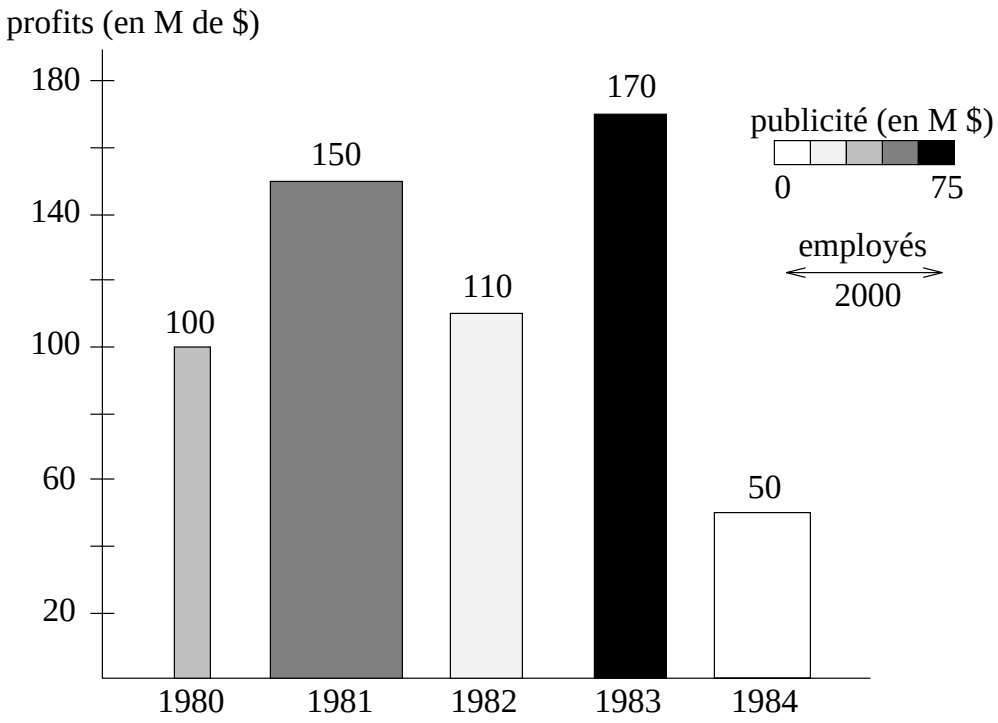


FIG. 3.6 - *Graphique en colonnes à intensité et largeur variable*

Le graphique présenté à la figure 3.7 est un cas spécial. En effet, ce graphique est difficile à classer car il partage des caractéristiques des colonnes et de la tarte. Il a la forme d'une colonne unique mais sa lecture ne se fait pas en mesurant la longueur de la colonne. La longueur de la colonne est fixe et il n'y a pas d'axe gradué comme pour le graphique en colonnes conventionnel. Tout comme la tarte, c'est un graphique partitionné. La colonne est divisée en sous-rectangles dont la longueur est proportionnelle à la part qui est représentée. Ce graphique, que nous appelons colonne partitionnée, est présenté ici car il est souvent utilisé en groupes de deux ou plus, et sous cette forme, il peut être facilement confondu avec le graphique en colonnes conventionnel. Bien que sa forme de surface soit celle d'une colonne, sa forme profonde (la façon dont il encode l'information) est celle d'une tarte et son utilisation sera présentée dans la section suivante.

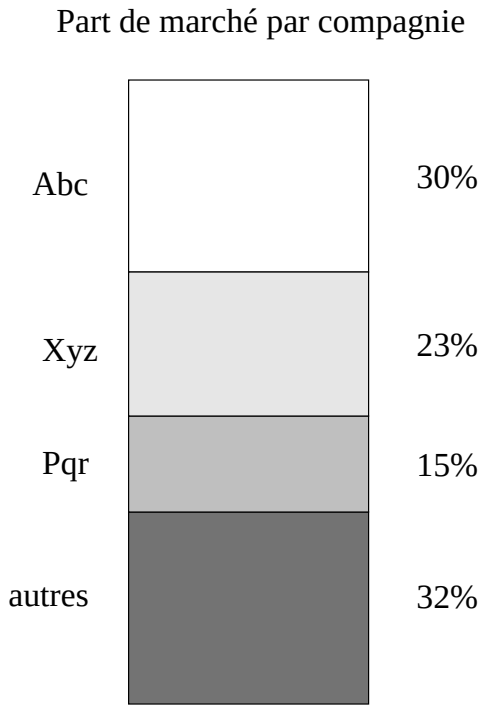


FIG. 3.7 - *Colonne partitionnée*

Les tartes

Les tartes (*pie chart* en anglais, *camembert* en France) (figure 3.8) sont très utiles pour exprimer des fractions d'un tout tant au niveau des comparaisons qu'au niveau des répartitions. Dans le premier cas, leur fonction est similaire à celle des barres, sauf qu'on compare des fractions (ex: comparer des parts de marché de compagnies). Dans le second, leur fonction est similaire à celle des colonnes utilisées dans le contexte d'un histogramme (ex: montrer le pourcentage d'auditeurs d'une émission dans chaque classe d'âge). Notons ici la distinction entre forme et fonction (histogramme est une fonction qui peut être remplie par les colonnes ou la tarte).

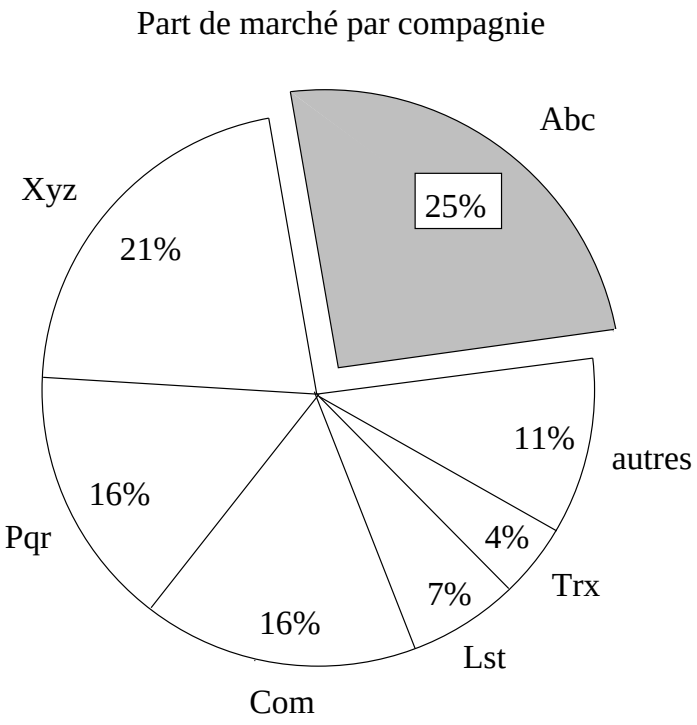


FIG. 3.8 - *Graphique en tarte*

L'éclatement ainsi que les couleurs permettent la mise en valeur de sections particulières. Pour ce qui est des contraintes d'utilisation: une des variables doit être ou jouer le rôle d'une étiquette et l'autre doit être fonction de celle-ci. De plus, si

le nombre de secteurs dépasse une valeur critique, il est fortement recommandé de grouper les plus petites dans un secteur “divers” ou “autres”. Cette valeur critique dépend des auteurs. Par exemple, selon Bertin [Bertin, 1983], la limite est 6. Selon Timbal-Duclaux [Timbal-Duclaux, 1990], elle est de 8. Selon notre expérience, la limite est floue et dépend de la différence de taille entre les secteurs, mais les valeurs de 6 et 8 peuvent servir de bornes sur le nombre idéal.

La tarte est un des graphiques les plus populaires et c’est sans aucun doute le plus utilisé pour montrer des décompositions. Pourtant, il s’agit d’un des graphiques les moins efficaces pour les deux raisons suivantes: premièrement, notre système visuel ne décode pas efficacement les angles que la tarte utilise pour représenter les valeurs. Deuxièmement, la tarte ne satisfait pas très bien le critère 3 de Wright. En effet, deux tartes sont presqu’impossibles à comparer même lorsque juxtaposées à cause de l’absence d’une référence horizontale ou verticale.

Notons qu’il existe deux variantes assez intéressantes de la tarte: la colonne partitionnée et le décagramme. La colonne partitionnée, présentée à la figure 3.7, corrige les lacunes de la tarte conventionnelle car elle n’utilise des longueurs à la place des angles et elle satisfait assez bien le critère 3 de Wright puisque plusieurs colonnes partionnées peuvent être alignées pour comparaison. Malheureusement, ce type de graphique est beaucoup moins utilisé que la tarte.

La seconde variante, le décagramme, est présenté à la figure 3.9. Il s’agit d’une tarte dans laquelle le cercle a été remplacé par un décagone (polygone régulier à 10 côtés). Le nombre 10 a été choisi car l’apparence générale du décagramme ressemble à celle du cercle, mais il y a assez de côtés pour introduire une référence intéressante: chaque côté représente une valeur de 10%. En comptant les côtés impliqués dans un angle, on peut mieux estimer sa valeur. Balogun [Balogun, 1978] a conclu, après une série de tests sur des usagers, que le décagramme était largement supérieur à la tarte (88.9% des usagers le préfèrent, 6.39% préfèrent la tarte et les autres sont indifférents).

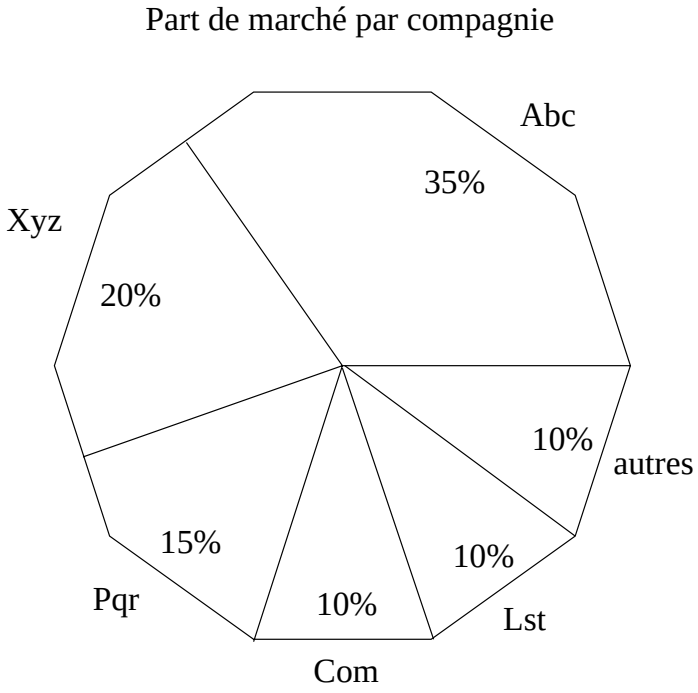


FIG. 3.9 - *Graphique en tarte de type décagramme*

Malgré ces résultats, ce graphique n'est presque jamais utilisé.

Les points

Les graphiques en points (figure 3.10) sont très appropriés pour exprimer des corrélations. Ils permettent de présenter un nuage de points (on ne sait pas d'avance s'il y a dépendance entre les variables) avec optionnellement une droite de régression pour indiquer la tendance.

Souvent, les points sont utilisés lorsque les contraintes des autres graphiques ne peuvent être satisfaites. En général, il n'y a pas de contrainte d'utilisation pour les points alors que les autres graphiques requièrent une dépendance fonctionnelle entre les variables.

La forme de base du graphique en points (figure 3.10) n'encode que 2 variables.

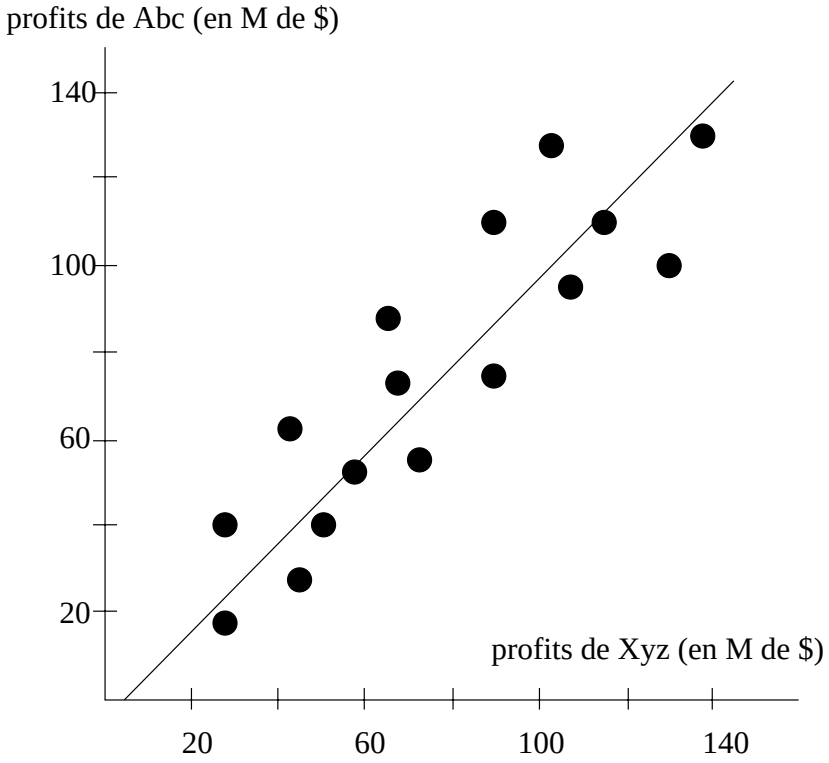


FIG. 3.10 - *Graphique en points*

En utilisant la surface des points ou l'intensité (figure 3.11), on peut encoder des variables supplémentaires. Ces deux techniques sont plus utilisées pour les points que pour les colonnes (section 3.1.2).

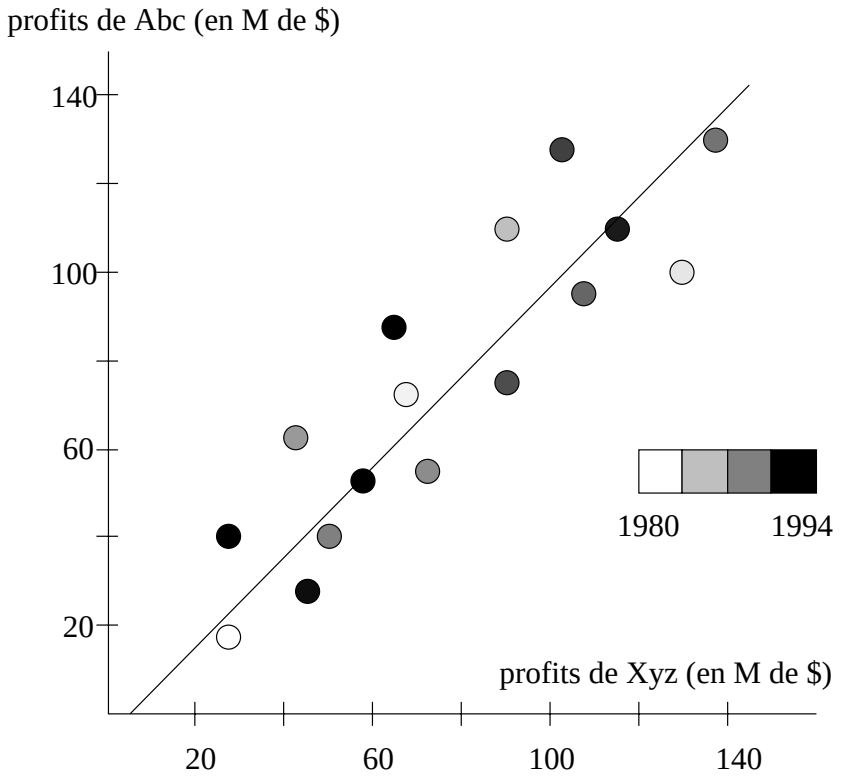


FIG. 3.11 - *Graphique en points à intensité variable*

Lorsque la 3^e variable d'un graphique en points est une étiquette, on peut l'encoder en utilisant des formes géométriques diverses (carré, triangle, ...) à la place des points conventionnels (figure 3.12). On peut aussi étiqueter directement les points lorsque l'espace le permet (figure 3.13). L'étiquetage direct est préférable à cause de sa meilleure lisibilité mais lorsque les étiquettes sont trop larges ou trop nombreuses, on se rabat sur les formes géométriques. Celles-ci nécessitent moins d'espace puisque la légende étiquetée se trouve hors de la zône principale du graphique.

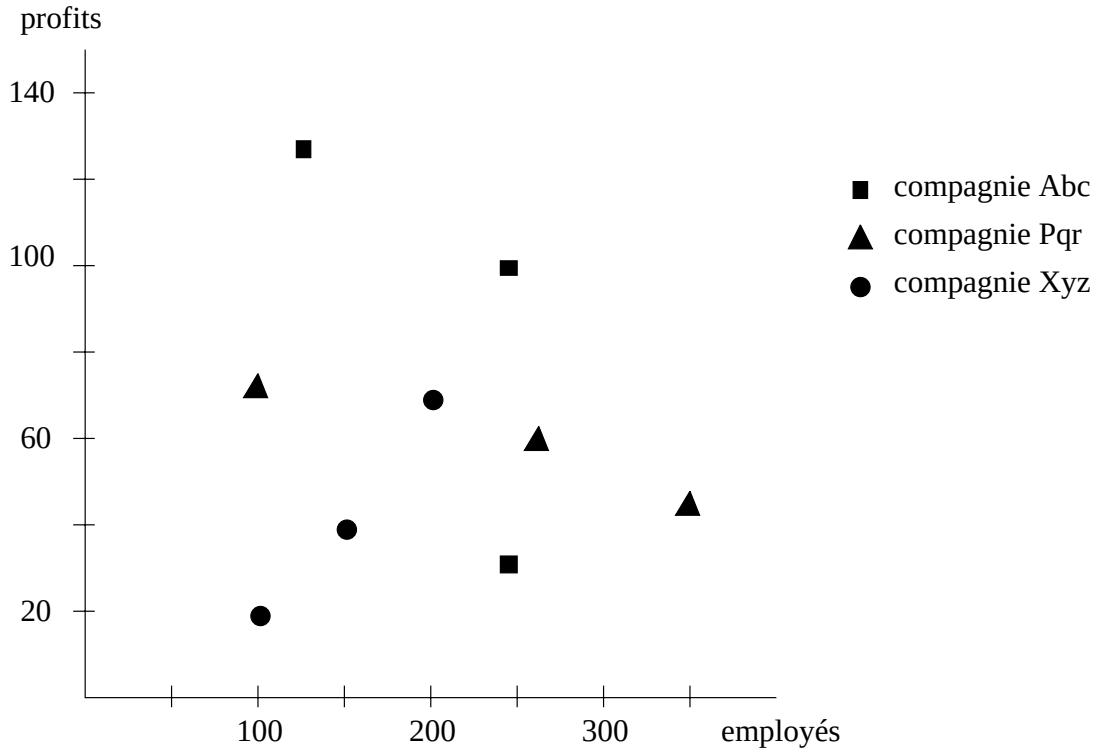


FIG. 3.12 - *Graphique en points à forme variable*

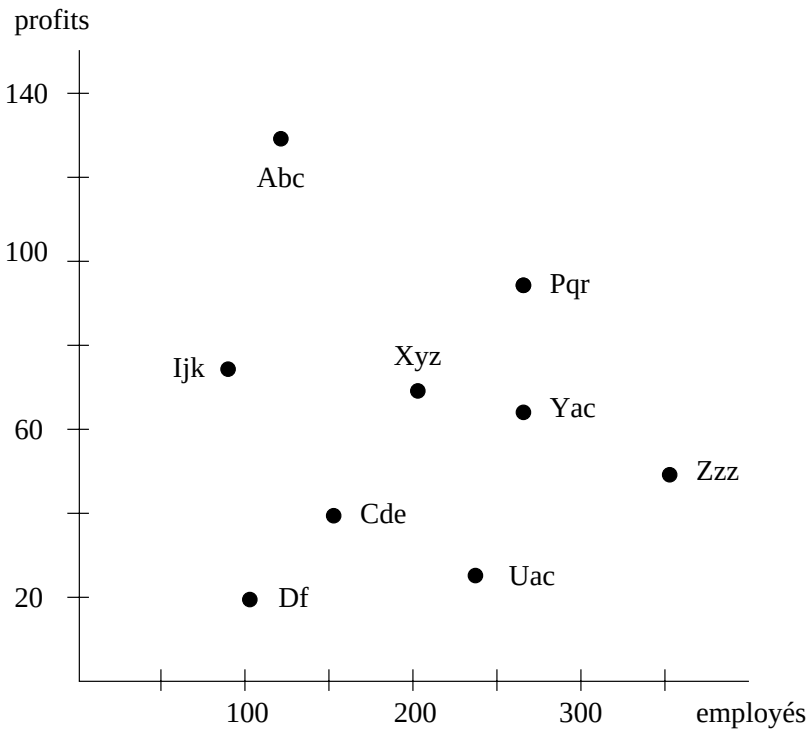


FIG. 3.13 - *Graphique en points étiquetés*

Les courbes

À première vue, on pourrait penser que les courbes (figure 3.14) sont des graphiques à points dans lesquels ceux-ci ont été reliés. Au niveau de l'apparence, c'est le cas, mais au niveau de la fonction, la ligne continue qui relie ces points souvent invisibles joue un rôle très important. En effet, elle impose une contrainte de dépendance fonctionnelle similaire à celle des colonnes et elle donne une impression naturelle d'évolution. Ceci rend la courbe fonctionnellement identique à des colonnes, même si sa forme est plus proche de celle des points.

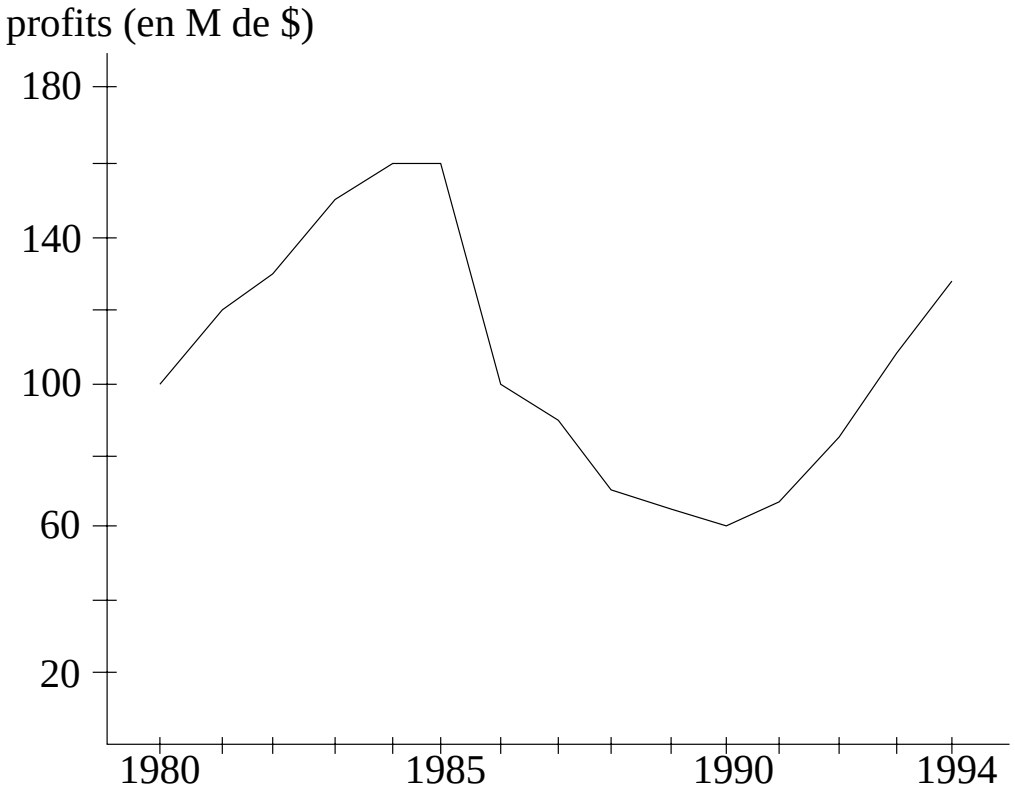


FIG. 3.14 - *Graphique en courbe*

La courbe est le graphique le plus utilisé pour présenter les données temporelles. Cependant, les études comparant courbes et colonnes montrent que les deux permettent une aussi bonne interprétation des tendances [Culbertson et Powers, 1959;

Jarvenpaa et Dickson, 1988].

Selon Culbertson et Powers [Culbertson et Powers, 1959], il est plus facile de déterminer les valeurs à partir d'un graphique en colonnes qu'à partir d'une courbe. En effet, les colonnes font mieux ressortir les valeurs individuelles.

Par contre, selon Jarvenpaa et Dickson [Jarvenpaa et Dickson, 1988], les barres et colonnes multiples (figure 3.3) sont beaucoup moins efficaces que les barres et colonnes simples (figure 3.4). L'alternance entre plusieurs séries de données brise la continuité et rend l'interprétation des tendances plus difficile.

Schutz [Schutz, 1961] conclut donc que les courbes multiples (figure 3.15) sont plus utiles qu'une courbe simple. Elles permettent des comparaisons faciles de plusieurs ensembles de données dans un graphique qui n'est pas surchargé, chose que les colonnes (ou barres) multiples ne réussissent pas à faire aussi bien.

3.2 Le texte

Dans un rapport illustré, le texte a deux rôles très importants: il décrit et il lie. La description textuelle permet de présenter certains éléments d'information difficiles à montrer graphiquement. L'aspect liaison est fondamental pour un rapport à cause de la nature flottante des figures. Le texte sert de point d'ancrage pour les références aux figures et permet de faire des transitions entre celles-ci.

Dans cette section, nous situons notre recherche en présentant une comparaison entre le texte des rapports statistiques et les types de texte utilisés dans d'autres disciplines. Ensuite nous montrons différentes méthodes permettant d'utiliser le texte dans un rapport statistique. Nous traitons le cas des légendes et celui des textes continus, en insistant dans les deux cas sur l'utilisation de références pour renforcer

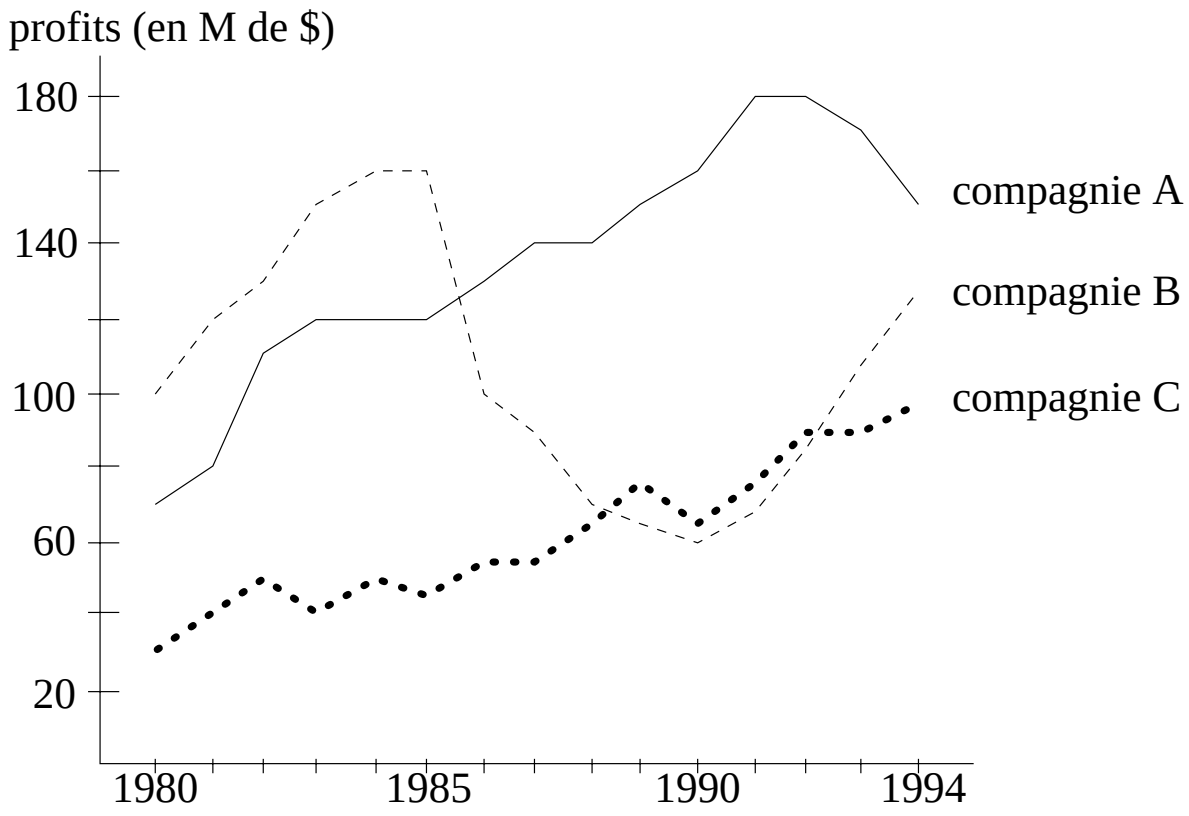


FIG. 3.15 - *Graphique à 3 courbes*

l'intégration texte-graphique. Finalement, nous présentons des techniques de mise en page du texte et leur rôle dans les rapports statistiques.

3.2.1 Comparaison des types de texte

Les textes utilisés dans les rapports statistiques présentent des différences importantes avec les textes procéduraux (recettes de cuisine, manuels d'instruction), les textes descriptifs et les textes narratifs. Ces différences se situent à deux niveaux: la structure globale et le niveau d'organisation de l'information.

Le niveau d'organisation d'un rapport est plus élevé que celui d'un texte narratif. En effet, le rapport doit être structuré de façon à ce que son analyse ait plus de poids. Pour un texte narratif, la structure est plus libre et suit l'événement qui est raconté. Les rapports n'offrent toutefois pas une structure aussi stricte que les textes procéduraux: tous les manuels d'instructions sont structurés de façon similaire, de même pour les recettes de cuisine. L'aspect structure générale n'est pas un de nos intérèts majeurs et à ce niveau, les rapports sont intéressants puisque leur structure est assez simple et stéréotypée et ne demande donc pas d'effort majeur de modélisation. Il est impossible d'étudier les textes procéduraux sans s'intéresser directement à leur structure. Le problème inverse se pose face aux textes narratifs à cause de la très grande liberté de structure: il faut se restreindre à un sous type très bien défini de façon à avoir une structure à suivre. On perd donc la généralité de l'application.

On peut aussi classifier les textes selon le niveau d'organisation de l'information qu'ils présentent. En effet, un rapport statistique est souvent construit à partir de données brutes comme des tableaux de nombres alors que d'autres textes partent d'information semi-structurée à plusieurs niveaux comme temporel (récit) ou spatial (description d'un objet). Parfois un texte peut même partir d'un autre texte, comme pour un résumé d'un texte plus long ou une critique de celui-ci.

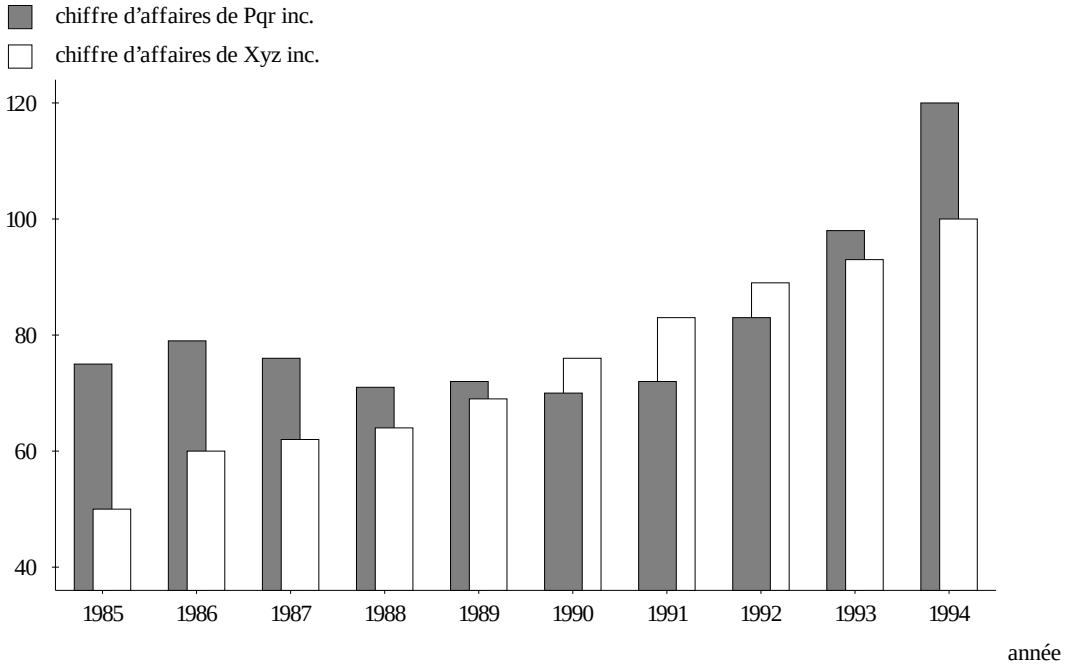
Cette classification est très importante car elle indique le degré de liberté dont on dispose pour produire un texte. Il est clair qu'il y a beaucoup plus de choix en partant de nombres que d'un texte organisé. Souvent, un résumé de texte narratif suivra l'ordre et la structure du texte original; on se limite alors au choix de l'information essentielle. La génération de texte pour les rapports statistiques peut profiter du degré de liberté pour mieux adapter le texte aux besoins du rédacteur mais elle doit aussi effectuer beaucoup de travail de structuration qui n'est pas nécessaire lorsque les données sont moins brutes.

3.2.2 Légendes textuelles dans les rapports statistiques

Dans une présentation multi-modale, on associe souvent un court texte à chaque figure pour en décrire brièvement le contenu. Ce type de texte est caractérisé par les deux éléments suivants: sa longueur et sa position. En effet, les légendes sont des textes très courts jumelés à la figure qu'ils décrivent.

Puisqu'une légende est limitée à une ou deux courtes phrases, son contenu doit être direct et précis. Elle doit se limiter à une description des points essentiels qui caractérisent sa figure et ne peut pas prendre le temps de la situer dans le reste du rapport, rôle qui est laissé à un texte continu. Parfois, la description reste d'ordre très général et on ne fait que présenter les données impliquées dans la figure alors que dans d'autres rapports, on place un court commentaire dans la légende. Les deux styles ont leur intérêt, et il est même possible de les combiner, comme on peut voir à la figure 3.16, de façon à profiter des avantages des deux.

Le jumelage de la légende et de la figure donnent un impact très important au texte. En effet, après avoir regardé rapidement la figure, le lecteur lit la légende. Son contenu est donc assimilé avant d'examiner la figure de façon plus détaillée. Ainsi, une légende objective bien formulée guidera le lecteur dans son inspection du graphique en dirigeant son attention vers les éléments et tendances les plus importants.



Xyz inc. et Pqr inc. ont vu leur chiffre d'affaires augmenter de 1985 à 1994. Pqr est restée en tête sauf entre 1990 et 1992.

FIG. 3.16 - *Évolution du chiffre d'affaires de Xyz inc. et Pqr inc.*

Une légende peut aussi orienter la perception du lecteur de façon plus subjective, et cette utilisation est beaucoup plus difficile à maitriser. Comme on va le voir à la section 4.2.2, le choix d'une figure ambiguë, mais pas trop, crée au niveau du lecteur un instant de confusion dont la légende peut profiter pour lui imposer une interprétation non-objective. Si la figure est trop ambiguë ou si le texte pousse l'interprétation trop loin, le lecteur perdra confiance et tirera ses propres conclusions d'une inspection visuelle plus détaillée. Il est donc important pour le rédacteur de bien doser la subjectivité et pour le lecteur d'être très méfiant des légendes. La figure 3.17 présente un exemple simple de ce type de légende. En effet, la partie visuelle montre deux rectangles non-alignés. La lecture naturelle de cette figure consiste à regarder rapidement les deux rectangles, lire la légende et puis possiblement examiner en détail les rectangles si on n'est pas saitsfait de l'interprétation. À cause de ce qu'on a vu à la section 2.1.3, l'ambigüité du graphique[1] est naturelle et peut passer inaperçue, ce qui permet à la légende d'imposer une interprétation inexacte de la réalité.

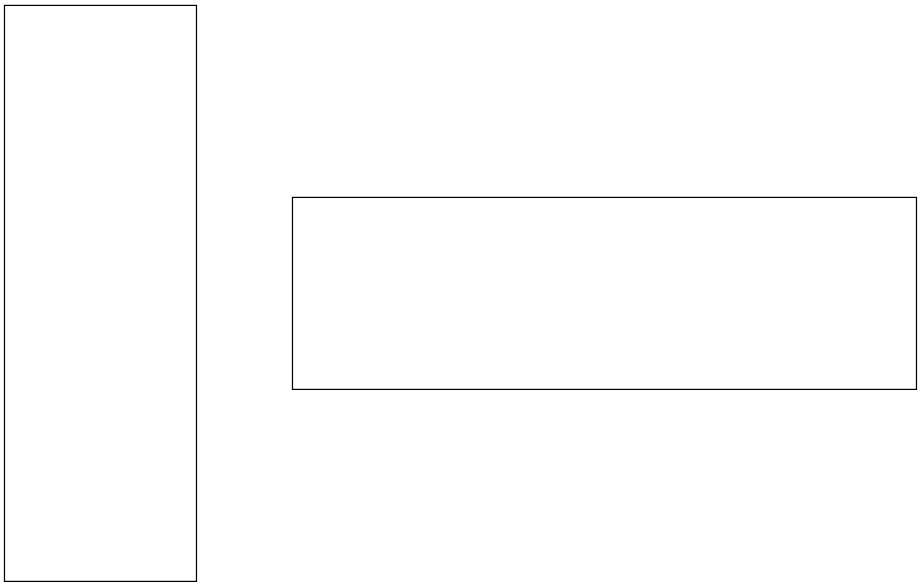


FIG. 3.17 - *Deux rectangles identiques orientés différemment*

1. Les deux rectangles ne sont pas de même longueur.

La proximité avec la figure rend les références inutiles. En effet, il est inutile de faire référence à un numéro de figure particulier dans la légende ou même de mentionner qu'on parle d'une figure. Puisque le texte et la figure sont liés physiquement, la portée de la légende est sous-entendue. Ainsi une légende comme "Les profits de la compagnie Incognito inc. ont diminué graduellement entre 1985 et 1990" est suffisante et préférable à "Comme on peut le voir à la figure ci-dessus, les profits de la compagnie Incognito inc. ont diminué graduellement entre 1985 et 1990".

Par contre, des références à des éléments internes à la figure sont utiles pour y diriger l'attention du lecteur plus rapidement. Par exemple, on parlera de la courbe rouge ou du secteur de tarte éclaté. Ce type de référence encourage l'utilisation de marques distinctives faciles à nommer dans les graphiques et les tableaux. Par exemple, l'utilisation de couleurs rend la référence plus facile que l'utilisation de pointillés différents (voir figure 3.18). En effet, un texte qui parle de la courbe en pointillés alternant long, long, court surprend un peu alors qu'une référence à la courbe verte est très naturelle.

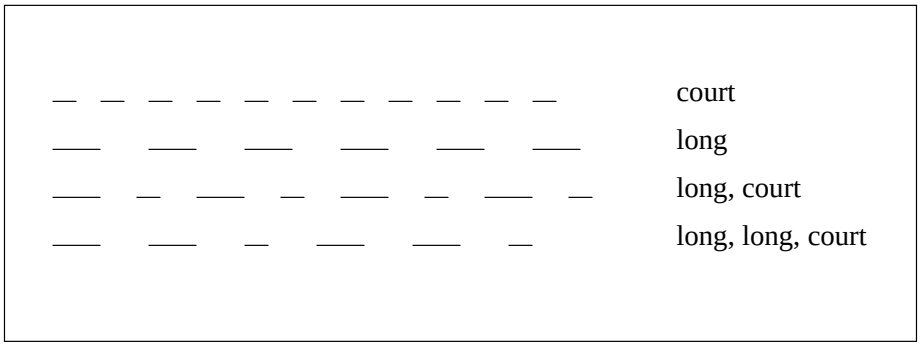


FIG. 3.18 - *Différents types de pointillés*

3.2.3 Texte continu dans les rapports statistiques

Le texte qui constitue le corps d'un rapport statistique a des propriétés très différentes de celui des légendes qui accompagnent les figures. En effet, ce texte étant long et continu, il peut se premettre de présenter plus d'information pout situer chaque

figure et de faire le lien entre celles-ci. De plus, le texte n'est plus juxtaposé aux figures, ce qui rend nécessaire l'utilisation de références à des figures parfois éloignées.

Dans le corps du rapport, il est possible de décrire une figure comme dans une légende, mais cette description aura un impact moins direct car elle ne sera pas lue en même temps que la figure. Souvent, pendant le temps perdu à chercher la figure, on a déjà oublié les détails de la description. Par conséquent, il est préférable de donner le plus possible d'information générale sur les données et les figures et de laisser les détails aux légendes. Ainsi, l'extrait suivant est efficace dans le corps du rapport:

La figure X.Y présente l'évolution des profits des compagnies A, B et C entre 1985 et 1990.

Par contre, le texte suivant devrait être réservé à une légende car il est plus précis:

Les profits des 3 compagnies ont diminué entre 1985 et 1990.

Ce format oblige cependant une lecture non-séquentielle de l'information puisqu'on doit absolument aller consulter la figure pour en avoir une description textuelle détaillée. Ainsi, on choisit donc d'inclure aussi dans le texte principal une description précise plus longue que celle de la légende, de façon à permettre au lecteur de parcourir le texte sans avoir à consulter les figures. Par exemple, on pourrait inclure l'extrait suivant dans le corps du rapport:

La figure X.Y présente l'évolution des profits des compagnies A, B et C entre 1985 et 1990. Les profits ont diminué graduellement pour les 3 compagnies (moins de 2 millions par an en moyenne) sauf en 1987 où elles ont chuté de 8 millions pour la compagnie C.

Le texte principal du rapport joue aussi le rôle de "colle" pour les figures. En effet, en plus de servir de point d'ancrage pour les figures flottantes à l'aide des références, il permet de lier celles-ci en faisant des transitions d'un thème à l'autre.

Ces transitions sont importantes lorsqu'une série de graphiques montrent plusieurs facettes d'un même sujet. Par exemple, l'extrait de texte suivant montre le passage d'une évolution à une corrélation.

La figure 3.19 montre l'évolution des profits de la compagnie Xyz inc. et de ses investissements en recherche et développement entre 1985 et 1994. Tous deux ont augmenté graduellement pendant la période en question (50 et 28 millions respectivement). Il semblerait que l'augmentation des profits soit liée à l'augmentation des investissements en r/d. Cette tendance est confirmée par la figure 3.20.

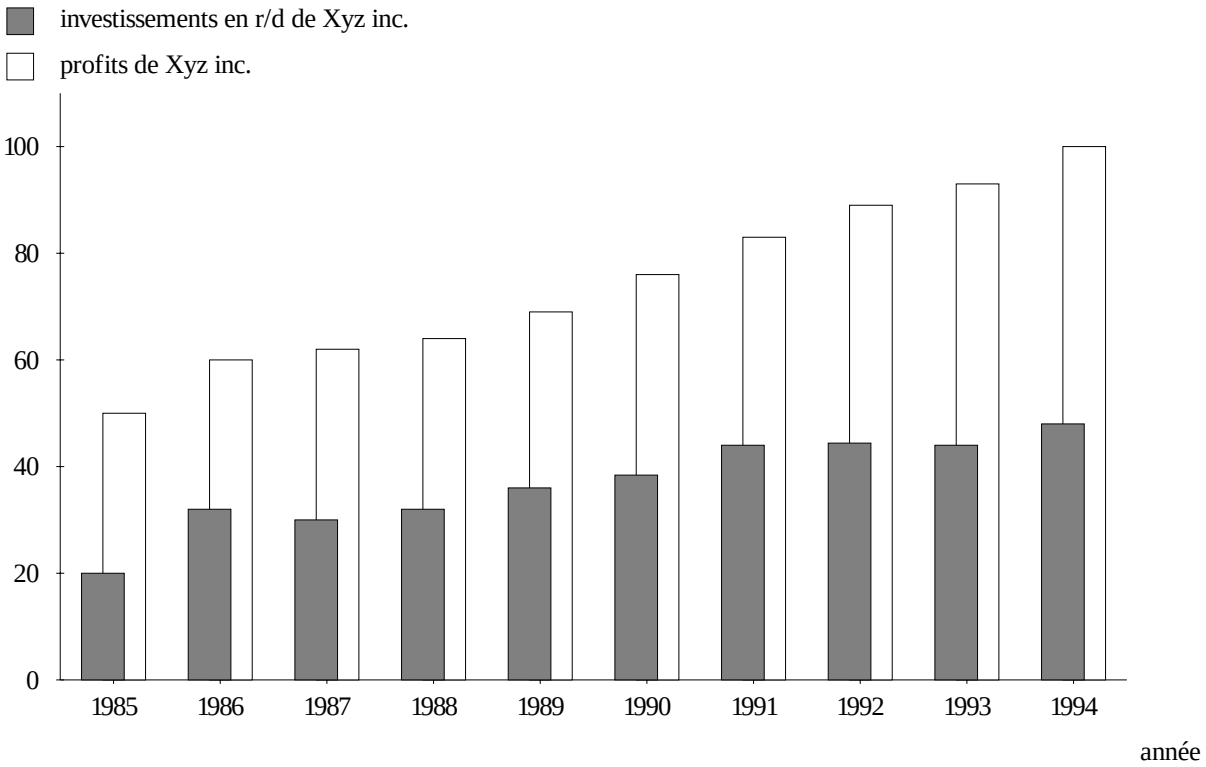


FIG. 3.19 - *Évolution des profits et investissements de Xyz*

Un autre aspect important à considérer dans le texte principal du rapport est celui des références trop précises. En effet, il est nécessaire d'inclure une référence

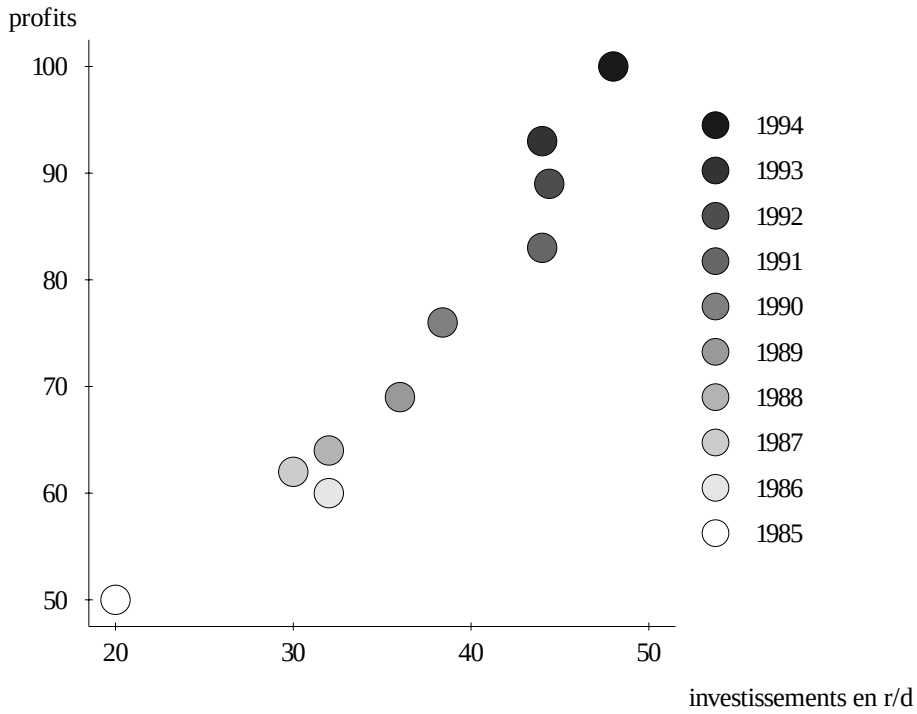


FIG. 3.20 - *Corrélation entre les profits et les investissements de Xyz*

à une figure avant d'en parler puisque celle-ci peut être sur une autre page que le texte. Cependant, il faut faire attention à ne pas rendre la référence trop précise. Les références à des parties d'une figure, bien que très utiles lorsqu'utilisées dans un texte adjacent à la figure, perdent beaucoup de leut efficacité lorsqu'il faut tourner 2 ou 3 pages pour les suivre.

3.2.4 Techniques de mise en page du texte

Dans tous les types de textes, on utilise un certain nombre de techniques de mise en page pour transmettre de l'information de façon implicite. Cette transmission d'information est possible à cause d'un ensemble de conventions qui régissent l'utilisation de ces techniques [Hovy, 1993; Pascual, 1993].

Ces techniques peuvent agir sur le style visuel du texte, sur l'organisation des blocs de texte ainsi que sur les marges et l'espacement.

Les techniques agissant sur le style visuel du texte fonctionnent soit au niveau des caractères (changement de police, majuscules) ou au niveau des groupes de mots (parenthèses, guillemets). Par exemple, les guillemets servent à marquer une citation.

Le paragraphe est la façon habituelle d'organiser l'information dans un texte. En général, un paragraphe consiste d'une suite de phrases reliées thématiquement et organisées séquentiellement. La fin du paragraphe est marquée d'un retour de chariot et son début est souvent marqué d'un alinéa. Cette organisation, bien que prédominante, n'est pas la seule façon d'agencer l'information. En effet, il est aussi possible d'agencer l'information sous forme de liste. Dans une liste, chaque élément, souvent une courte phrase, est séparé des autres par une marque visuelle. Les formes de liste les plus courantes sont les listes non-énumérées (figure 3.21), les listes énumérées (figure 3.22) et les listes de définitions (figure 3.23).

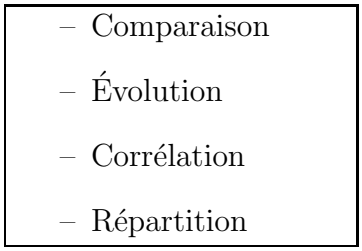


FIG. 3.21 - *Exemple de liste non-énumérée*

Dans la liste non-énumérée, le même marqueur graphique (souvent un cercle noir rempli) est utilisé au début de chaque élément. Dans la liste énumérée, les éléments sont ordonnés et numérotés. La numérotation sert donc de marqueur de début d'élément. Dans les listes de définitions, l'entité à définir est mise en relief par changement de son style visuel (ex: caractères gras) et débute chaque élément. Le reste de l'élément est la définition.

1. Allumer l'appareil
2. Appuyer sur la touche **MENU**
3. Appuyer 2 fois sur la touche **RUBRIQUE**
4. Appuyer sur la touche +
5. Appuyer sur la touche **MENU**
6. Appuyer simultanément sur les touches **OPTION** et **MÉMOIRE**

FIG. 3.22 - *Exemple de liste énumérée*

Camembert Fromage au lait de vache, à pâte molle et à croûte moisie.

Tarte Préparation faite d'une abaisse de pâte garnie d'un appareil salé ou sucré et cuite dans un moule.

FIG. 3.23 - *Format d'une liste de définitions*

Les listes sont utiles dans le cadre des rapports statistiques à cause de leur propriétés visuelles: elles s'apparentent à certains types de graphiques statistiques. À priori, toutes les listes ressemblent énormément aux graphiques en barres à cause de la succession verticale d'objets horizontaux. Cependant, à cause de la versatilité du texte, elles peuvent remplacer ou compléter d'autres graphiques comme les colonnes ou les courbes. Par exemple, la liste à la figure 3.24 joue le même rôle que le graphique à la figure 3.25 et conserve une partie des propriétés visuelles de ce graphique tout en utilisant des phrases complètes plutôt que des courbes. On garde la séparation horizontale visuelle entre les 3 compagnies, on perd la vue d'ensemble des courbes mais on gagne la puissance descriptive du texte.

Compagnie A Les profits ont été assez stables de 1987 à 1990.

Compagnie B Malgré quelques instabilités, les profits en 1990 sont à peine inférieurs à ceux de 1987.

Compagnie C Doublement des profits entre 1987 et 1990. L'augmentation a eu lieu surtout de 1989 à 1990.

FIG. 3.24 - *Comparaison de l'évolution des profits des compagnies A, B et C*

Les marges et l'espacement sont elles aussi utilisées pour transmettre de l'information. Par exemple, l'élargissement des marges pour un paragraphe marque en général une citation. En plus, elles servent à renforcer l'organisation du texte. Ainsi, on espace les paragraphes et on indente les listes pour mieux les mettre en valeur. Dans une série de listes imbriquées, le niveau d'indentation a une correspondance directe avec le niveau d'imbrication (figure 3.26).

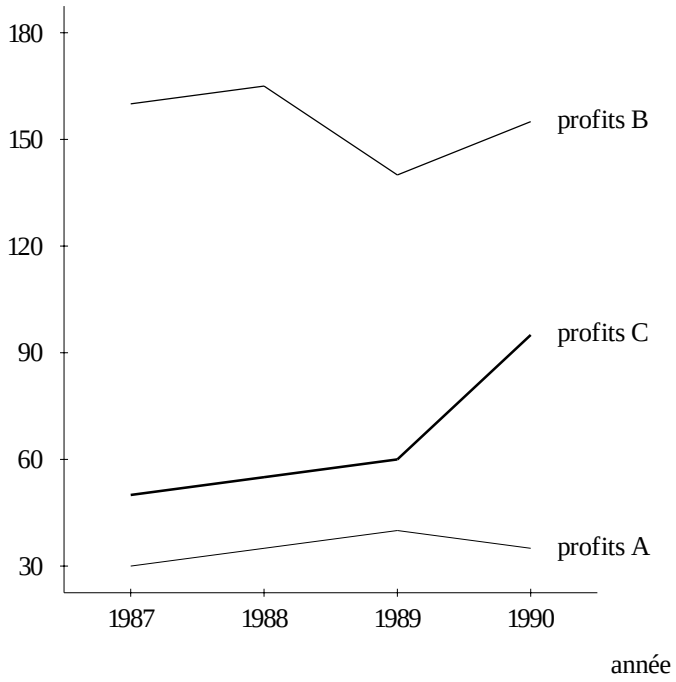


FIG. 3.25 - *Comparaison de l'évolution des profits des compagnies A, B et C*

- Section 1
  - Section 1.1
  - Section 1.2
    - Section 1.1.1
    - Section 1.1.2
  - Section 1.3
- Section 2
- Section 3
  - Section 3.1
  - Section 3.2

FIG. 3.26 - *Liste non-énumérée à plusieurs niveaux*

3.3 Résumé

Dans ce chapitre, nous avons présenté les formes d'expression utilisées dans les rapports statistiques. Ces formes d'expression se divisent en deux catégories: les figures et le texte. Le texte forme le corps d'un rapport alors que les figures sont souvent des objets flottants qui sont attachés au texte par des références.

Nous avons décrit deux types de figures: les tableaux et les graphiques statistiques. En général, les tableaux permettent une lecture exacte des données alors que les graphiques en montrent plutôt une vue d'ensemble.

Dans un rapport statistique, le texte sert à analyser les données et à lier les figures. L'utilisation du texte dans les rapports a été présentée en 3 parties: les légendes des figures, le corps du rapport et les techniques de mise en page.

Après cette présentation des formes d'expression et de leurs propriétés, nous présentons au chapitre suivant un modèle permettant de choisir la forme d'expression la plus appropriée pour exprimer chaque cas de données.

Chapitre 4

Choix d'une forme d'expression

Notre recherche a pour but de trouver un ensemble de critères permettant le choix de textes et de graphiques dans le contexte d'un rapport statistique.

Pour réaliser notre modèle, nous nous sommes surtout inspiré des travaux de Tufte [Tufte, 1983; Tufte, 1990] sur l'art de présenter les graphiques, de Bertin [Bertin, 1983] sur l'organisation et la structure des graphiques, ceux de Zelazny [Zelazny, 1989] sur la relation entre message et graphique, ainsi que les travaux de MacKinlay [Mackinlay, 1986a] qui ont abouti à un système informatique (APT) qui choisit et dessine des graphiques statistiques.

L'étude de MacKinlay fournit un algorithme basé sur une extension de la classification de Bertin (variables nominales, ordinales et quantitatives) et définit un ensemble de méthodes graphiques pour l'expression de chaque type de variable. Ces méthodes sont ordonnées et l'algorithme fait l'allocation des méthodes aux variables en évitant de générer des graphiques impossibles à réaliser (par exemple, 3 coordonnées spatiales dans un graphique à 2 dimensions).

Nous avons aussi intégré à notre modèle des résultats plus théoriques comme ceux de Tufte et de Zelazny. On retient de ces travaux une classification des propriétés d'un graphique tant au niveau de ses composantes élémentaires (Bertin, MacKinlay) qu'au

niveau global (Zelazny). Les propriétés des composantes élémentaires jouent surtout sur l'efficacité d'un graphique pour exprimer un type donné de variable. Par exemple, les positions spatiales sont plus utiles que les couleurs pour exprimer des variables continues. Les caractéristiques globales d'un graphique sont surtout utiles pour déterminer son rôle dans la transmission du message. Ainsi, des graphiques dont les éléments de base sont similaires peuvent avoir des rôles un peu différents (les barres sont adéquates pour comparer des valeurs, alors qu'on utilise toujours les colonnes ou les courbes pour des données temporelles). Notre classification des propriétés intègre ces résultats et les applique à un ensemble beaucoup plus général de graphiques.

Alors que plusieurs recherches se sont concentrées sur le modèle du lecteur [Paris, 1991], nous étudions plutôt celui du rédacteur du rapport. Cette étude nous a permis d'élaborer un ensemble de règles associant l'intention du rédacteur à des formes d'expression pertinentes.

Les rapports sont souvent biaisés par les buts du rédacteur. On doit donc tenir compte des rapports objectifs mais aussi des rapports subjectifs dans lesquels on essaie de faire "mentir" un peu les données. Cette subjectivité affecte également la façon dont les graphiques et le texte sont combinés. Cet aspect du problème a été très peu étudié dans le domaine. Nous avons survolé ce problème dans notre étude théorique, surtout au niveau de l'expression des évolutions. À cause de sa complexité et du manque de temps, nous n'avons pas poussé l'étude à tous les types d'intention.

4.1 Données et leurs propriétés

Les données brutes sont le facteur le plus déterminant d'un graphique puisqu'elles en définissent le contenu. Des propriétés plus générales ont aussi une influence très marquante: il s'agit d'informations sur le type des variables à traiter et sur l'existence de clés relationnelles parmi celles-ci.

4.1.1 Les données brutes

Les données brutes utilisées dans notre modèle sont organisées sous forme *tabulaire* ($n \times m$), tout comme dans le modèle relationnel. Elles se divisent en un ensemble de m *variables*, chacune représentée par une colonne du tableau. Dans le reste de cette thèse, nous utiliserons la notation suivante pour représenter les données:

- d_{ij} fait référence à l'élément du tableau de données se trouvant à la ligne i et à la colonne j.
- $d_{\bullet j}$ fait référence à la colonne j du tableau. Notons que chaque colonne représente une variable, elle-même nommée de façon unique pour faciliter les références.
- $d_{i\bullet}$ fait référence au tuple se trouvant à la ligne i.

Le tableau 4.1 permet de visualiser cette notation.

	$d_{\bullet 1}$	$d_{\bullet 2}$	$\ldots$	$d_{\bullet m}$
$d_{1\bullet}$	d_{11}	d_{12}		d_{1m}
$d_{2\bullet}$	d_{21}	d_{22}		d_{2m}
$\vdots$			$\ddots$	
$d_{n\bullet}$	d_{n1}	d_{n2}		d_{nm}

TAB. 4.1 - *Notation des données brutes*

Pour éviter les problèmes dans des cas limite, il faut examiner les données brutes. En effet, les deux situations suivantes sont les plus courantes:

- trop de données
- écarts trop grands

Le premier cas permet, entre autres, de choisir d'utiliser un graphique en courbe plutôt qu'un graphique en colonnes (lorsqu'on dépasse une douzaine de colonnes). Le

second pose des problèmes pour certains graphiques normalement très efficaces. Par exemple, à la figure 4.1, la lecture précise des petites valeurs est impossible à cause du choix d'échelle.

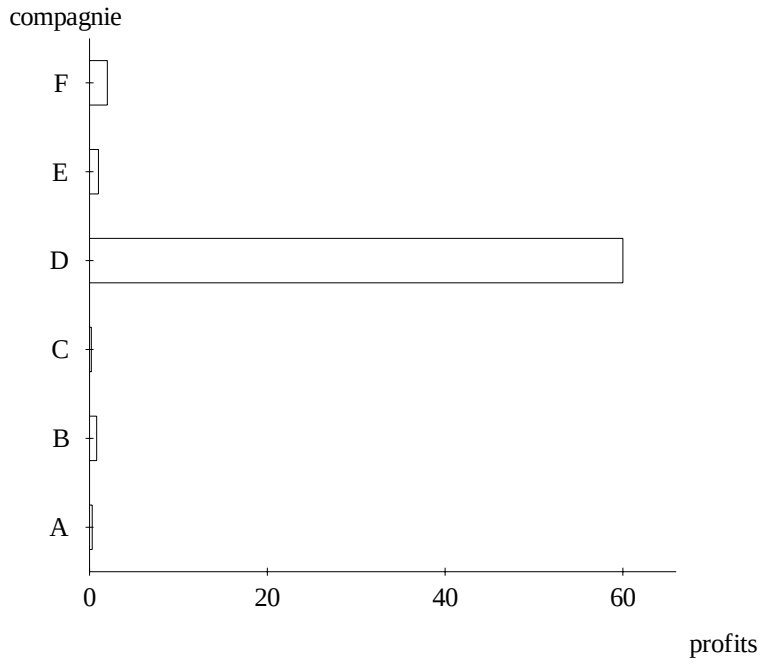


FIG. 4.1 - *Faible lisibilité à cause des écarts*

Pour corriger le problème, on peut utiliser un tableau dans lequel on n'a pas de problème d'échelle ou encore changer l'échelle de façon à ce que toutes les valeurs soient lisibles: échelle logarithmique ou échelle non-continue.

4.1.2 Les types

Dans notre modèle, nous nous servons d'informations de typage pour contrôler le choix et la réalisation du texte et des graphiques. Par exemple, nous limitons le choix de graphiques montrant une évolution au cas où la variable se trouvant sur l'axe horizontal est de type temporel. Cela évite de présenter des figures erronées comme par exemple une courbe montrant l'évolution des années par rapport aux profits d'une

compagnie (l'inverse est acceptable).

Notre système de types[1] est organisé en un graphe d'héritage multiple. Le graphe se décompose en un ensemble de facettes, chacune d'elles décrivant un aspect de la nature d'une variable. Le graphe est difficile à visualiser à cause des nombreux liens inter-facettes. Pour simplifier sa présentation, nous avons choisi une approche textuelle: la figure 4.2 montre les types classés par facette, indiquant pour chacun une liste de ses parents dans le graphe d'héritage. Notons que certains types sont paramétrés; par exemple, le type *intervalle* a comme paramètre les 2 bornes de l'intervalle en question. Pour simplifier la figure, les paramètres ont été omis mais seront discutés en détail dans le chapitre 6.

La notation suivante est utilisée pour tenir compte des types:

- $d_{\bullet j} :: t_j/T_j$ dénote la variable j de type principal t_j ayant comme types auxiliaires l'ensemble T_j.
- T_j est un ensemble de types $\{t_j^1, t_j^2, \ldots, t_j^q\}$.
- Lorsqu'on parle de la classe de la variable j, il s'agit de son type principal t_j. t_j est un type comme les autres placé dans un rôle dominant.
- $d_{\bullet j} :: t_j \equiv d_{\bullet j} :: t_j/\{\}$
- La notation T_j^n est utilisée pour tenir compte de l'héritage de types. T_j^n dénote l'ensemble des types de la variable j accessibles à travers n liens d'héritage. Ainsi $T_j^0 \equiv t_j/T_j$. Lorsque $n = *$, on fait référence à tous les types d'une variable accessibles par héritage.

Voici une description de chaque facette indiquant les choix permis par chacune d'elles:

1. Le mot *propriété* est aussi utilisé pour parler des types.

- Organisation
  - symbolique: énumération.
  - ordonnée.
  - nominale: symbolique.
  - ordinale: symbolique, ordonnée.
  - quantitative: ordonnée, nombre.
  - fractionnaire: quantitative.
- Format
  - étiquette: symbolique.
  - nombre: intervalle.
  - entier: nombre.
  - réel: nombre.
- Domaine
  - énumeration.
  - plus_grand.
  - plus_petit.
  - intervalle: plus_grand, plus_petit.
- Temps
  - temporelle: ordonnée.
  - mois: énumeration, temporelle, étiquette.
  - année: entier, symbolique, temporelle.
- Mesures
  - longueur.
  - position.
  - spatiale.
  - position_temporelle: temporelle, position.
  - date: position_temporelle.
  - durée: temporelle, longueur.
  - distance: spatiale, longueur.
  - position_spatiale: spatiale, position.
- Entités spécifiques
  - territoire: étiquette.
  - pays: territoire.
  - province: territoire.
  - pourcentage: réel, quantitative.
  - dollar: réel, quantitative.

FIG. 4.2 - *Classification des types*

Organisation

Cette facette reflète la classification de Bertin [Bertin, 1983] dans laquelle on retrouve 3 types de variables: nominale, ordinale et quantitative. Pour simplifier l'utilisation de cette classification et pour expliciter la sémantique des 3 types, nous avons ajouté un niveau dans la hiérarchie qui introduit 2 propriétés fondamentales à partir desquelles on peut déduire les 3 autres: variable symbolique (les valeurs de la variable jouent le rôle de symboles) et variable ordonnée (les valeurs de la variable ont un ordre pré-déterminé). Comme on peut le voir à la figure 4.2, les variables nominales et ordinales sont symboliques et les variables ordinales et quantitatives sont ordonnées. Une variable quantitative n'est pas symbolique car ses valeurs sont numériques et leurs propriétés numériques peuvent être utilisées. Une variable ordinale (ex: des notes sous forme de lettres et "+/-": A+, A, A-, B+, etc...) est symbolique et ordonnée.

À l'aide de ces propriétés, on peut faire des choix de graphiques beaucoup plus informés. Les propriétés *symbolique* et *ordonnée*, en particulier, sont beaucoup utilisées dans notre modèle. Par exemple, une tarte ne peut être choisie que si une de ses 2 variables est symbolique (les étiquettes des secteurs). Si une variable est ordonnée, ses valeurs ne pourront pas être réordonnées lors de la réalisation de certains graphiques comme les barres.

Format

Le format d'une variable (nombre, étiquette textuelle) est utile, entre autres, pour déterminer comment imprimer ses valeurs. C'est une propriété naturelle de la variable en question alors que l'organisation d'une variable est plus artificielle. En effet, une variable d'un format donné peut être organisée de plusieurs façons. Par exemple, des étiquettes peuvent être ordonnées ou pas. De plus, des nombres peuvent être traités comme des symboles dans certains cas: des modèles de voiture comme 626 ou 305.

Il est intéressant de noter que certains liens entre format et organisation sont

fixes: une variable quantitative est toujours un nombre et une étiquette est toujours symbolique.

Domaine

Les contraintes de domaine servent à indiquer que les valeurs d'une variable sont limitées à un ensemble donné. On traite les intervalles ouverts et fermés (l'intervalle fermé est une sous-classe des intervalles ouverts *plus_petit* et *plus_grand*) ainsi que les listes énumérées (ex: les mois de l'année).

Ce type de contrainte est utile, entre autres, pour préparer des échelles dans les graphiques.

Temps

Ce système est utile à deux niveaux: pour identifier quelles variables sont temporelles et aussi pour déterminer le type de donnée temporelle représentée.

Dans notre modèle, l'identification des variables temporelles est moins importante que dans un système comme celui de MacKinlay car l'utilisation d'intentions (évolution, en particulier) nous permet de les identifier implicitement par le contexte.

L'identification du type de donnée temporelle (les deux sous-types traités présentement sont année et mois) est utile pour savoir comment traiter et exprimer ces données, surtout dans un texte. Par exemple, lorsqu'on sait qu'une variable est de type mois, il est plus facile de calculer le temps écoulé entre les valeurs "Janvier" et "Mars", pour générer des locutions temporelles comme "Deux mois plus tard".

Mesures

Ce système est basé sur les notions de *position* et de *longueur*. Par héritage de ces propriétés et des propriétés *temporelle* et *spatiale* on peut obtenir plusieurs propriétés

intéressantes. Par exemple, une *durée* est une *longueur temporelle*. Une *distance* est une *longueur spatiale*. Une *date* est une *position temporelle*.

Entités spécifiques

Cette catégorie regroupe des classes plus concrètes qui servent dans des problèmes particuliers. Elles ne font pas partie du modèle de base et pourraient être décrites dans les données. Elles sont définies directement dans le système car elles sont souvent utilisées. Dans cette catégorie, on retrouve entre autres *dollar*, *pourcentage* et *pays*.

4.1.3 Les clés relationnelles

La classification de types présentée précédemment nous permet de caractériser chaque variable mais elle ne nous donne pas d'outils pour représenter les relations entre variables: quelles variables dépendent de quelles autres. Pour représenter ces relations, nous allons utiliser la notion de clé que l'on retrouve dans le modèle des bases de données relationnelles [Date, 1988].

Pour définir la notion de *clé*, il faut préciser les termes suivants: *relation*, *tuple* et *attribut*. Une *relation* est un tableau de données dont les lignes sont appelées *tuples* et les colonnes *attributs*. Les attributs sont donc les variables de la relation.

La définition de *clé*, telle que donnée par C. J. Date [Date, 1988], est la suivante: une *clé candidate* est un identificateur unique pour une relation. Par définition, une relation a au moins une clé candidate: l'ensemble de ses attributs. Lorsqu'une relation a plusieurs clés candidates, on choisit l'une d'entre elles comme clé primaire et les autres sont alors appelées clés auxiliaires.

Plus formellement, soit R une relation avec les attributs $d_{\bullet 1}, d_{\bullet 2}, \ldots, d_{\bullet n}$. L'ensemble d'attributs K de R tel que $K = (d_{\bullet k_1}, d_{\bullet k_2}, \ldots, d_{\bullet k_p}), k_i \neq k_j, 1 < k_i < k_j < n$ est appelé *clé candidate* de R si et seulement si il satisfait les deux conditions sui-

vantes:

1. **Unicité:** 2 tuples distincts de R ne peuvent pas avoir la même valeur pour K.
2. **Minimalité:** Aucun des $d_{\bullet k_i}$ ne peut être éliminé de K sans détruire la propriété précédente.

Par exemple, dans les données du tableau 4.2, (B) est une clé (unique et minimale) pour la relation (A, B, C, D) car pour une valeur de (B), on n'a qu'un seul tuple de valeurs de (A, B, C, D).

Selon la définition stricte, les couples (A, B), (B, C) et (B, D) ne sont pas des clés car ils ne respectent pas la propriété 2 (minimalité). Ils respectent cependant la propriété 1 et ceci les rend très utiles pour exprimer des dépendances entre variables. Dans notre modèle, nous généralisons le concept de *clé* à ces *clés non-minimales* et nous utilisons le terme *clé minimale* pour référer aux clés qui respectent les deux propriétés de la définition formelle.

Il faut noter que si (B) est une clé pour (A, B, C, D) alors (A, B, C) est une clé pour (A, B, C, D) et (B) est une clé pour (A, B, C) car on peut toujours ajouter des éléments à la clé et en enlever à la relation sans perturber la propriété 1 de la définition. Toutes ces variations peuvent être déduites de la forme canonique qui est la clé *minimale* pour la relation *maximale*.

Par contre, (A, C) n'est pas une clé pour (A, B, C, D) car au couple $(A = 1, C = 3)$ correspondent deux tuples distincts $(1, 2, 3, 4)$ et $(1, 4, 3, 4)$.

Les clés sont très utiles car elles nous indiquent quelles variables sont fonction de quelles autres (si (A) est une clé pour (A, B, C) alors on sait que B et C sont fonction de A). En particulier elles permettent d'écarter très rapidement des graphiques pour lesquels il est nécessaire d'avoir au plus une image par point de l'axe (horizontal ou vertical). Par exemple, une courbe conventionnelle nécessite une relation fonctionelle du type $y = f(x)$ ((x) doit être clé de (x, y)), où x et y correspondent aux valeurs

A	B	C	D
1	2	3	4
1	3	2	4
1	4	3	4

TAB. 4.2 - *Exemple de relation simple*

sur les axes du graphique. Par contre, les graphiques de type "points" n'imposent pas cette contrainte et il est tout-à-fait possible d'obtenir un "nuage" de points répartis dans les 2 dimensions.

De plus, dans les cas où la relation fonctionnelle n'est pas nécessaire, on peut quand même utiliser cette information pour ordonner les variables dans un graphique. Ainsi, on donnera les positions les plus visibles aux variables clés dans un graphique même si celui-ci n'impose pas de contraintes de ce type. Par exemple, si on choisit un tableau à trois colonnes pour montrer les profits de deux compagnies en fonction de l'année, on placera les années dans la première colonne, puisque c'est celle qu'on a tendance à voir en premier (lecture de gauche à droite).

4.2 Les intentions

4.2.1 Classification des intentions

L'intention du rédacteur affecte directement le message à présenter dans un rapport. Une très bonne étude sur la correspondance entre le message et le graphique a été faite par Gene Zelazny dans [Zelazny, 1989]. Dans cet ouvrage, il identifie 5 messages[2] de base et 5 graphiques très utilisés dans les rapports et il explique le lien entre ces deux ensembles (figure 4.3).

2. Zelazny les appelle *types de comparaison*, mais il affirme qu'ils correspondent directement aux messages que le rédacteur peut transmettre.

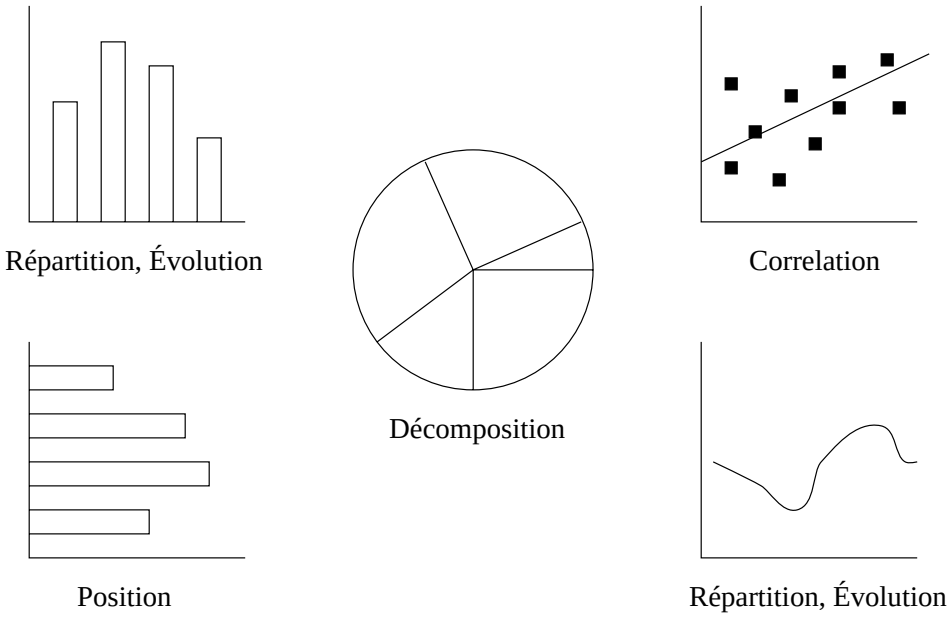


FIG. 4.3 - *Les associations message-graphique de Zelazny*

La décomposition Dans une décomposition, on cherche à montrer des valeurs en tant que fraction d'un tout. La tarte, de par sa nature, est presque toujours le meilleur graphique pour montrer une décomposition.

La position Ce type de message s'applique au classement d'éléments entre eux. Par exemple, comparer les profits de plusieurs entreprises. Le graphique à barres est le plus approprié pour ce type de message. Il rend la comparaison des valeurs facile (surtout si elles sont triées). Le graphique en colonnes est à éviter car il donne une impression d'évolution (on associe le temps à l'axe horizontal) et aussi parce que selon Graham, le lecteur a tendance à surestimer la longueur d'une colonne, rendant ainsi le graphique moins précis.

L'évolution Ce type de message vise à montrer des changements dans le temps, comme par exemple, la variation des profits d'une entreprise au fil des années. Le graphique en colonnes (données non-continues) et la courbe (données continues) sont appropriés pour ce type de message. Dans les deux graphiques, le temps est toujours

représenté horizontalement.

La répartition C'est une répartition de fréquence dans laquelle on cherche à montrer combien d'échantillons se trouvent dans chacun des intervalles considérés. Les graphiques qui correspondent à ce type de message sont les colonnes et la courbe.

La corrélation Une comparaison de corrélation montre la relation entre deux variables. Elle permet de tester si le comportement d'une variable est lié à celui d'une autre. On utilise souvent le graphique en points (avec une droite de régression) pour montrer une corrélation.

Cette étude est un très bon point de départ mais elle est incomplète à deux points de vue: tout d'abord les 5 graphiques traités par l'auteur ne sont pas suffisants pour tous les rapports; les messages utilisés, quoique très utiles, semblent manquer d'organisation: décomposition et position ont des points communs qui ne sont pas traités (les deux comparent), évolution est trop générale, etc... Nous avons donc étendu son étude à des graphiques plus complexes tout en réorganisant les messages qu'il a définis. Notons que dans notre modèle, le terme *intention* est utilisé pour décrire le but communicatif du rédacteur alors que le terme *message* est réservé au contenu de la transmission. Zelazny ne fait pas de distinction entre ces deux notions.

Pour effectuer cette réorganisation, nous introduisons la notion d'intentions modificatrices. Dans notre modèle, nous traitons 5 intentions de base: lecture, évolution, comparaison, corrélation et répartition. Il est possible d'appliquer des modificateurs à ces 5 intentions pour en obtenir des variations intéressantes. Le tableau 4.3 résume notre classification et la compare à celle de Zelazny.

La notation suivante est utilisée pour formaliser une intention simple. Une notation

Notre classification		**Classification de Zelazny**
Message de base	Modificateur	
lecture		
évolution		évolution
évolution	augmentation	
évolution	diminution	
évolution	stagnation	
évolution	récapitulation	
corrélation		corrélation
comparaison		position
comparaison	décomposition	décomposition
répartition		répartition
répartition	proportion	

TAB. 4.3 - *Classification des intentions*

plus générale sera présentée à la section 4.2.3.

- $i(v_1)$ représente une intention unaire.
- $i(v_1, v_2)$ représente une intention binaire.
- v_n est un $d_{\bullet j}$ ou une liste de $d_{\bullet j}$.

Une intention unaire agit sur une variable ou un ensemble de variables. Une intention binaire agit sur une variable ou un ensemble par rapport à une autre variable ou un autre ensemble. Par exemple, on parle de l'évolution d'une variable par rapport à une autre donc *évolution* est binaire. Le tableau 4.4 indique l'arité de chaque intention et spécifie aussi le type de chaque paramètre (variable simple ou ensemble). *Comparaison* a deux formes binaires: la forme utilisant comme premier paramètre une variable simple compare les valeurs de cette variable entre elles; la forme utilisant un ensemble compare en parallèle les valeurs de cet ensemble de variables. Notez que

l'arité et le nombre de variables impliquées sont deux concepts distincts. Par exemple, *comparaison* est d'arité 2 mais on peut comparer un nombre arbitraire de variables puisque le premier paramètre peut être un ensemble. Nous n'avons pas trouvé d'intentions d'arité supérieure à 2 mais notre modèle peut les traiter sans aucun problème puisqu'il ne fait aucune supposition sur l'arité.

UNAIRES
lecture(variable)
décomposition(variable)
proportion(ensemble)
BINAIRES
comparaison(variable,ensemble)
comparaison(ensemble,ensemble)
évolution(variable,variable)
correlation(variable,variable)
répartition(variable,ensemble)

TAB. 4.4 - *Arité des intentions*

Voici maintenant une description détaillée de chaque type d'intention avec les modificateurs qui s'y appliquent:

L'intention *lecture* exprime le besoin de lire des valeurs individuelles plutôt que de regarder uniquement les tendances globales. C'est le facteur majeur qui permet la discrimination entre les tableaux et les graphiques. Il est aussi très important dans le choix et la formulation du texte qui peut accompagner les figures. En effet, une information simple comme la moyenne d'une série de données peut être indiquée par une marque visuelle dans le graphique lorsque sa valeur précise n'est pas importante, mais elle peut être exprimée dans le texte lorsque celle-ci est importante (Ex: La moyenne des profits des 3 compagnies est de 45 millions de $). Dans notre modèle, la

lecture est une intention sans modificateur; elle n'existe pas dans le modèle de Zelazny.

Nous traitons 4 modificateurs pour l'évolution (augmentation, diminution, stagnation et récapitulation). Ces modificateurs permettent d'introduire des éléments subjectifs au message qui sont très apparents à la figure 4.4.

Corrélation n'a pas de modificateur et correspond au message du même nom dans le modèle de Zelazny.

Comparaison sous sa forme de base correspond au message *position* de Zelazny. Lorsqu'on lui applique le modificateur *décomposition*, on obtient le message *décomposition* de Zelazny. Ainsi, dans notre modèle, ce que Zelazny appelle la décomposition n'est qu'une forme spéciale de comparaison où on utilise des fractions au lieu des valeurs.

On applique le même raisonnement à la répartition. Sa forme de base est une répartition par valeurs (ex: 4 compagnies ont un chiffre d'affaires entre 30 et 50 millions de $) et elle a un modificateur *proportion* qui permet d'avoir une version fractionnaire du message (ex: 34% des auditeurs ont entre 13 et 19 ans).

4.2.2 Effet des intentions sur le texte et les graphiques

L'intention du rédacteur a un effet très important sur l'utilisation du texte et des graphiques. Cette influence n'est pas tout à fait la même dans un rapport subjectif que dans un rapport objectif.

Au niveau des rapports objectifs, l'effet de l'intention du rédacteur est assez claire. Certains graphiques sont mieux adaptés à certains messages (Zelazny):

- décomposition: tarte
- évolution: colonnes ou courbe

Année	Profits ($)
1971	10 000 000
1972	11 000 000
1973	11 000 000
1974	18 000 000
1975	16 000 000
1976	16 000 000
1977	15 000 000
1978	19 000 000

AUGMENTATION: Les profits de la compagnie ont augmenté de 8 millions entre 1971 et 1974, puis de 3 millions entre 1975 et 1978 après une chute de 2 millions.

DIMINUTION: Les profits de la compagnie ont diminué de 3 millions entre 1974 et 1977.

STAGNATION: Les profits de la compagnie ont stagné entre 1971 et 1973, puis entre 1975 et 1977 après une grosse augmentation en 1974.

RÉCAPITULATION: Les profits de la compagnie ont subi 3 ans d'augmentation, 2 ans de stagnation et 2 ans de diminution entre 1971 et 1978.

FIG. 4.4 - *Adaptation du texte à l'intention du rédacteur*

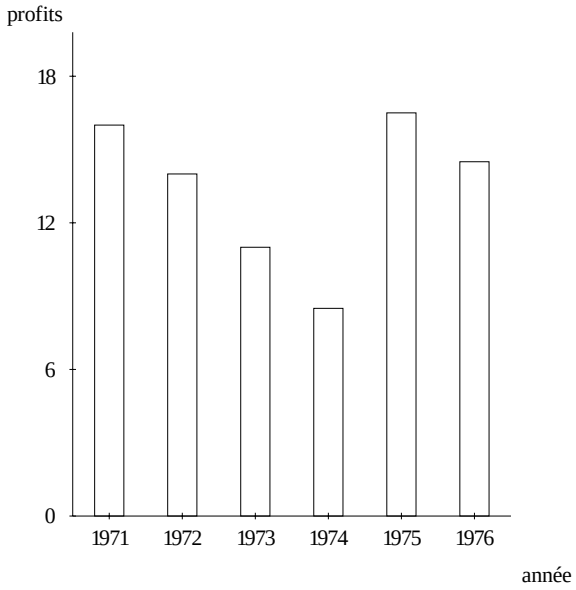
– corrélation: points

Au niveau du texte, le contenu change complètement, comme on peut le voir à la figure 4.5. Cet exemple présente les mêmes données (profits d'une compagnie entre 1971 et 1976) à travers deux messages différents. La partie supérieure montre les données à travers un message d'évolution alors que celle du bas insiste plutôt sur la comparaison des années.

Pour ce qui est des rapports plus subjectifs, les effets sont beaucoup plus subtils et dépendent aussi du lecteur. Un graphique subjectif est plus difficile à réaliser qu'un texte subjectif car toute omission dans un graphique est suspecte. Dans un texte, l'omission est pratique courante pour le rendre plus compact. Toutefois, comme on peut le voir à la figure 4.6, deux graphiques similaires peuvent tout de même transmettre des messages très différents. La partie du haut insiste beaucoup sur les fluctuations et combine texte et graphique de façon conventionnelle. Le message qu'on y présente est honnête tant au niveau du texte que du graphique. Par contre, le mariage du texte et du graphique dans la partie du bas est assez spécial. Le message présenté dans le texte est extrèmement biaisé, à un tel point qu'il ne correspond plus beaucoup aux données. En effet, il insiste sur la grosse augmentation et parle d'une série de petites diminutions, oubliant cependant de mentionner que la somme de ces diminutions dépasse l'augmentation. Le choix d'un graphique pour accompagner ce texte est assez délicat: si on omet le graphique, le lecteur se doute qu'on lui cache des choses et si on en montre trop dans le graphique, le lecteur voit bien que notre texte n'est pas honnête. On a choisi de reprendre le graphique du haut en alignant les variations à 0, ce qui rend beaucoup plus difficile la sommation de toutes les petites diminutions, tout en présentant assez d'information pour ne pas soulever de doutes.

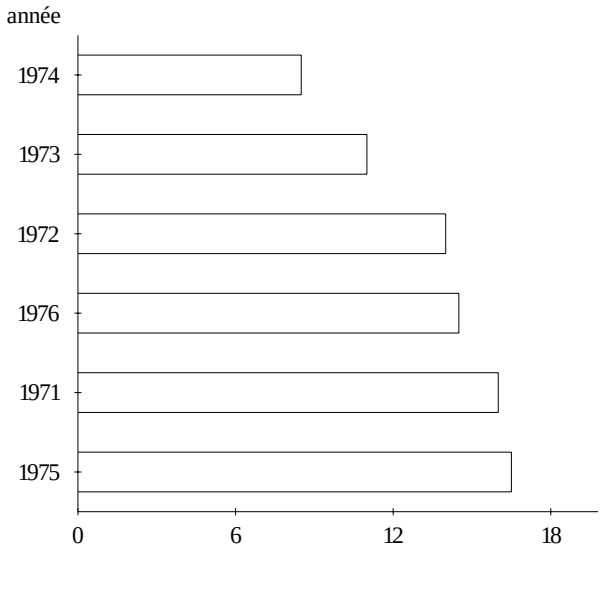
La figure 4.6 montre donc une intégration texte/graphique hors de l'ordinaire: utiliser le graphique pour rassurer un peu le lecteur tout en essayant de le déjouer en présentant les données de façon à appuyer le mieux possible le texte. On voit donc

ÉVOLUTION



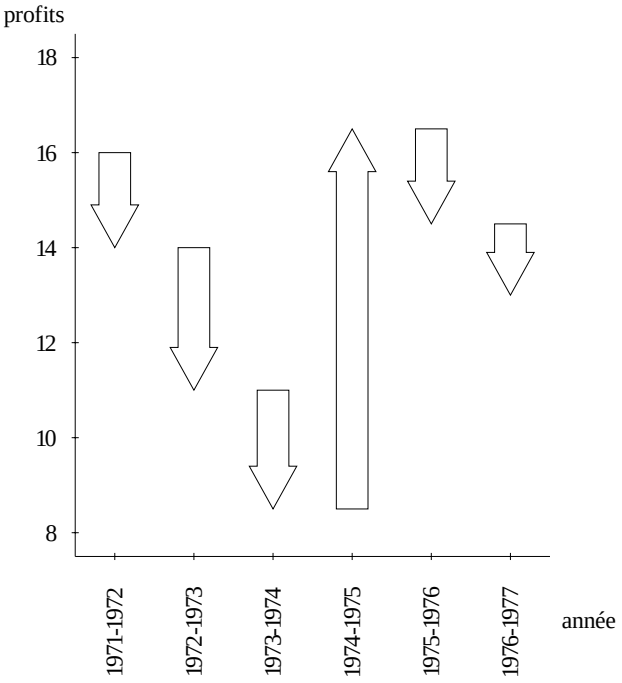
Globalement, les profits ont diminué malgré une forte reprise de 1974 à 1975.

COMPARAISON

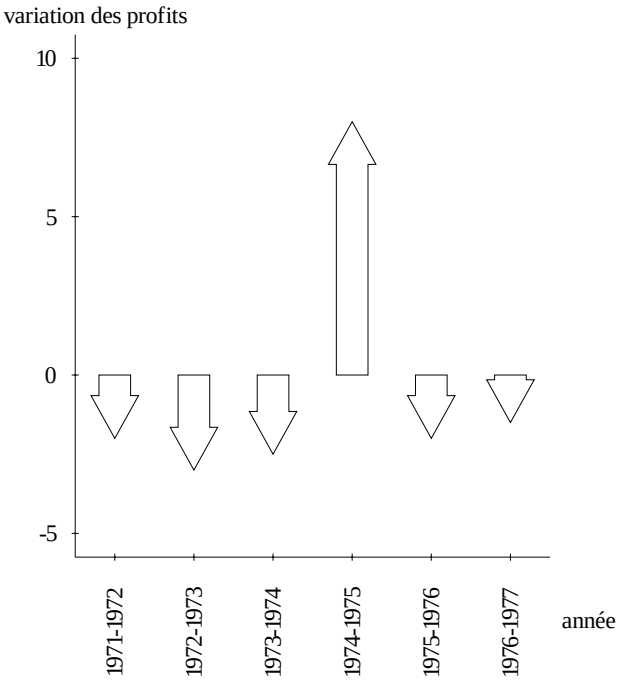


Les profits ont été à leur plus fort en 1975 et en 1971. Ils ont été à leur plus bas en 1974, avec environ la moitié de leur valeur de 1975.

FIG. 4.5 - *Effet des messages évolution et comparaison*



Globalement, les profits ont diminué malgré une forte reprise de 1974 à 1975.



Les profits ont subi une grosse augmentation de 1974 à 1975 et des petites diminutions les autres années.

FIG. 4.6 - *Intégration texte/graphique pour un message subjectif*

que l'intégration plus conventionnelle qui vise en général à combiner pour mettre en valeur les points saillants ne s'applique pas toujours aux cas subjectifs.

4.2.3 Groupement d'intentions

On retrouve souvent plusieurs intentions à l'intérieur d'un rapport statistique. Par exemple, à la figure 4.5, on distingue deux intentions exprimées à partir des mêmes données. Dans cette figure, les deux intentions sont distinctes, mais ce n'est pas toujours le cas. Par exemple, la figure 4.7 montre un graphique dans lequel deux intentions (évolution et comparaison) sont groupées pour produire un message hybride de comparaison-évolution.

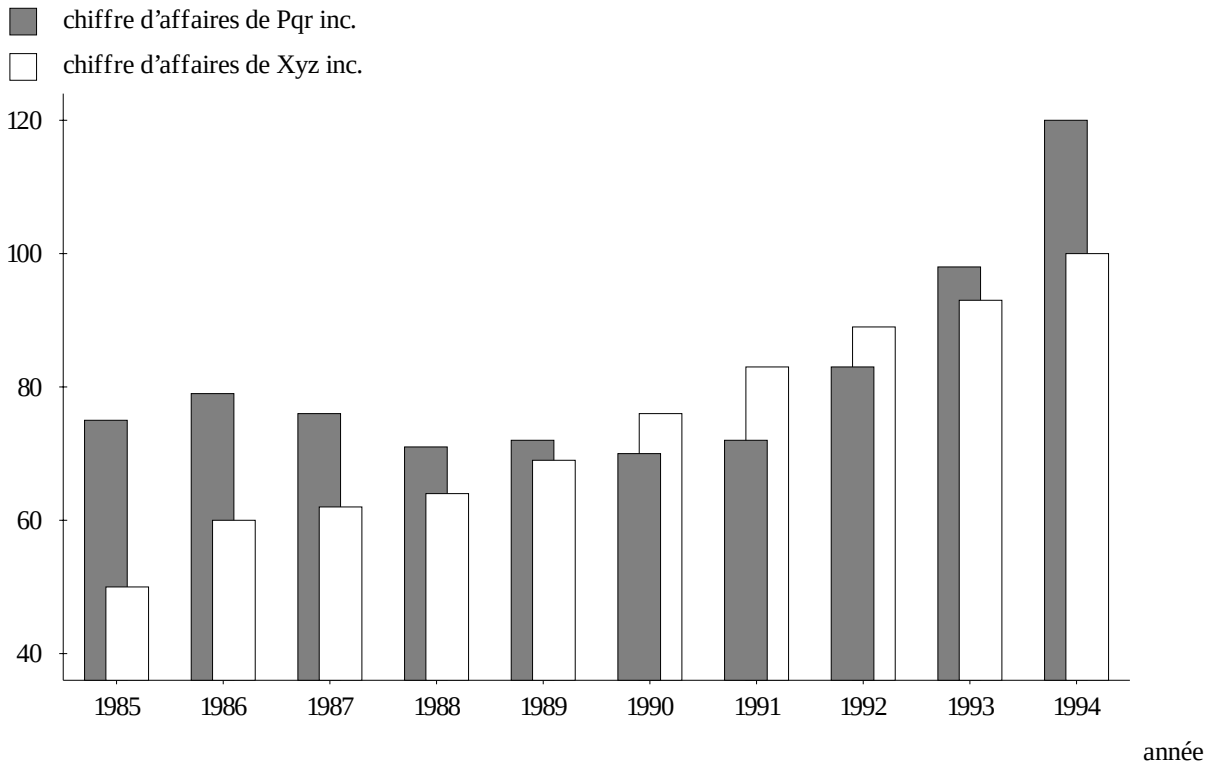


FIG. 4.7 - *Groupement d'évolution et de comparaison*

Le groupement d'intentions est un phénomène complexe. Dans l'exemple précé-

dent, on parle d'un message hybride de comparaison-évolution, mais est-ce la même chose qu'un message d'évolution-comparaison? En fait, dans cet exemple, la différence est assez subtile car on n'utilise qu'un graphique, mais avec un texte, on voit tout de suite la différence entre comparer des évolutions et montrer l'évolution de la comparaison: La phrase *Le chiffre d'affaires de Xyz a augmenté moins rapidement que celui de Pqr entre 1992 et 1994* compare des évolutions alors que la phrase *Pqr est restée en tête sauf entre 1990 et 1992* montre l'évolution de la comparaison entre les deux compagnies.

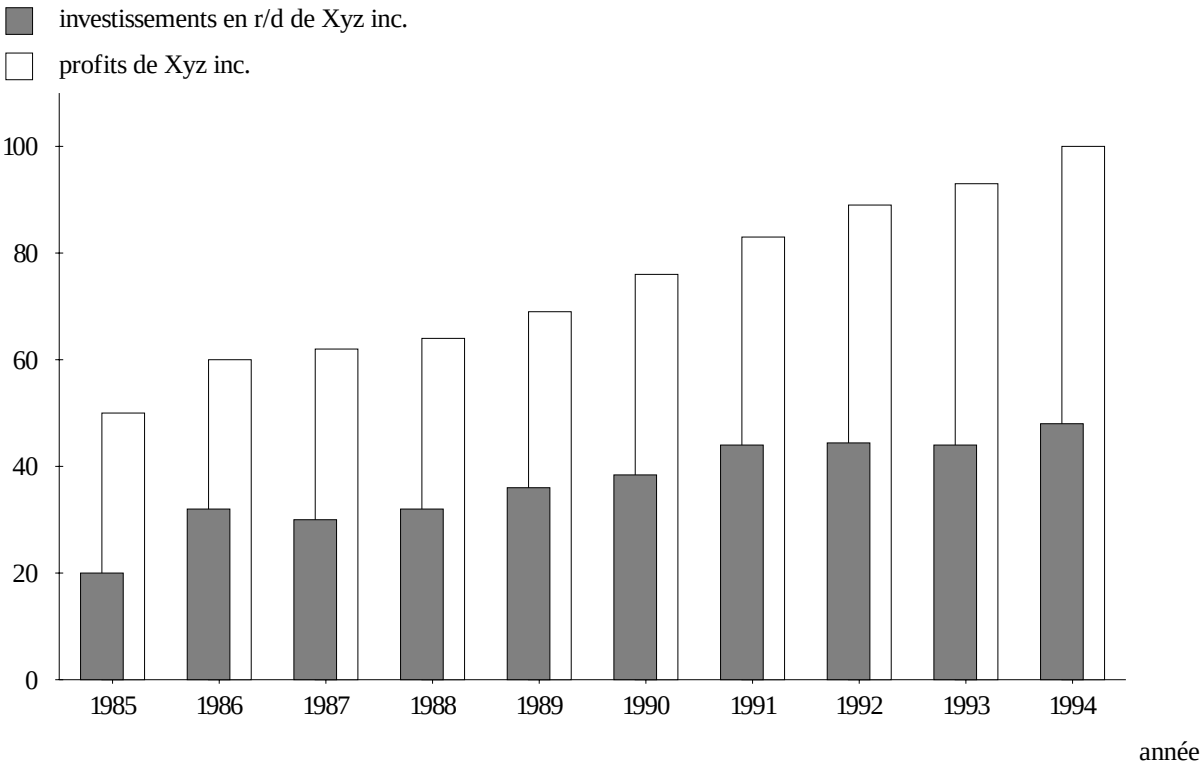


FIG. 4.8 - *Superposition de corrélation et d'une composition de comparaison et évolution*

Dans les deux exemples ci-dessus, on a composé deux intentions pour obtenir une intention hybride, mais il est possible de grouper deux intentions sans que celles-ci ne

forment un nouveau message. Elle seront exprimées ensemble (par exemple, dans la même figure) mais resteront distinctes. Ainsi, à la figure 4.8, on peut distinguer une certaine corrélation entre les profits et les investissements mais cette corrélation est indépendante du message hybride de comparaison et évolution. On parlera alors de superposition d'intentions.

Nous utilisons la notation suivante pour représenter une intention sous sa forme générale:

$$IG == IS \mid [IG_1, IG_2, \ldots, IG_n] \mid \{IG_1, IG_2, \ldots, IG_n\}$$

Une intention générale IG est une intention simple IS (section 4.2.1), une liste (ordonnée) d'intentions générales (composition) ou un ensemble (non-ordonné) d'intentions générales (superposition).

En résumé, on peut faire la composition ou la superposition des intentions et dans le cas de la composition, l'ordre a une importance; on peut donc parler d'intention dominante. Tous les groupements sont possibles, mais la comparaison intervient plus souvent que les autres intentions, parfois même comme "parasite" car le lecteur a tendance à comparer les objets ou quantités visibles même si là n'est pas le but direct du message.

4.3 Résumé

Le modèle présenté dans cette section utilise plusieurs sources d'information complémentaires pour faire des choix de formes d'expression dans le cadre de rapports statistiques. Les sources d'information étudiées sont:

- Les données brutes
- Les types des variables
- Les relations entre variables

– Les intentions du rédacteur

Chacune de ces sources apporte un élément utile au processus de décision. Un modèle ne peut donc pas être complet sans considérer toutes les 4. Cependant, il est impossible d'affirmer que notre modèle est complet parce qu'il tient compte de ces informations. En effet, beaucoup d'information contextuelle est nécessaire pour produire un excellent rapport statistique. Ce type d'information est très difficile à formaliser car il traite souvent de causalité (ex: la chute des profits de 1983 est due à une grève). Malgré tout, en puisant dans ces 4 sources en même temps, notre modèle est capable de prendre des décisions beaucoup plus informées.

Les intentions du rédacteur permettent d'établir le but du rapport. Elles sont donc l'élément directeur du processus de génération. Les types des variables permettent de produire du texte et des légendes beaucoup plus lisibles en donnant au générateur de rapports une meilleure connaissance des unités. Les types des variables et les relations entre variables sont très importants pour choisir entre des graphiques ayant la même fonction mais des structures très différentes. Finalement, les données brutes doivent être considérées pour éviter des cas où une forme d'expression est incapable de montrer le nombre de valeurs ou les écarts de valeurs présents dans les données.

Notons en terminant que ces 4 sources apportent parfois des informations contradictoires qu'il faut pouvoir pondérer. Par exemple, si on revient à la figure 4.1 (p. 78), on se rend compte que le graphique est peu lisible à cause des écarts mais que si l'intention du rédacteur est de montrer que D est loin en tête, alors le graphique est excellent.

Chapitre 5

De la théorie à la pratique

Pour vérifier la validité des résultats théoriques présentés dans cette thèse, nous avons implanté PostGraphe, un prototype de générateur de rapports statistiques multimédias. Dans les deux chapitres suivants, nous montrerons comment, en se servant de notre modèle théorique, le système est en mesure de produire des graphiques et du texte pertinents et de les combiner. Nous rappellerons tout d'abord dans ce chapitre les points d'influence majeurs sur la planification d'un rapport statistique, puis nous verrons les compromis effectués pour arriver à une implantation opérationnelle. Le prochain chapitre portera sur les détails d'implantation plus techniques.

5.1 Points d'influence majeurs sur la planification

Comme nous l'avons vu au chapitre 4, il est important de tenir compte d'un ensemble de facteurs pour choisir les textes et graphiques à inclure dans un rapport statistique. Parmi ces facteurs, on retrouve l'intention du rédacteur, le type des variables impliquées, leurs valeurs, ainsi que les relations entre ces variables.

Il est important de rappeler que l'intention du rédacteur joue un rôle prédominant dans la rédaction d'un rapport. En effet, comme nous avons vu, les mêmes données peuvent être exprimées de plusieurs façons très différentes en fonction du message à

transmettre (par exemple, voir figures 4.4 (p.91) et 4.5 (p.93)). Si on n'identifie pas le but communicatif des différents éléments du rapport, on risque de transmettre un message erroné.

L'étude des variables joue aussi un rôle très important dans la production d'un rapport. Les types des variables nous donnent des indications sur la structure des éléments du rapport. Par exemple, alors qu'une variable de type continu sera mieux représentée par une courbe, la nature d'une variable discrète sera mise en valeur par des colonnes. Pour raffiner le processus de sélection, il faut aussi tenir compte des valeurs des variables. Parfois, le nombre de valeurs peut avoir une forte influence sur les choix effectués. Ainsi, une variable de type discret avec 200 valeurs sera plutôt traitée comme continue. Dans d'autre situations, l'écart entre les valeurs a une influence. En effet, comme le montre la figure 4.1 à la page 78, un bon choix peut être remis en question lorsque les écarts entre les valeurs sont extrêmes.

5.2 Architecture d'un planificateur de rapports

L'architecture idéale d'un planificateur de rapports décompose le processus en une série de tâches consécutives et indépendantes. Une telle architecture permet la mise à jour facile d'un module ou sa substitution par un autre module équivalent. Un bon exemple de ceci est le système FRANA [Contant, 1985], dans lequel le module de réalisation textuelle de ANA [Kukich, 1983] a été remplacé par une version française équivalente.

Parmi les étapes importantes d'un générateur de rapports statistiques multimédias, on retrouve la sélection du contenu (quoi dire?), la sélection du médium (par quels outils l'exprimer?), l'organisation du discours (comment ordonner et agencer l'information?) ainsi que la réalisation finale des éléments textuels et graphiques du rapport (comment le dire?).

Idéalement, la sélection du contenu doit se faire en connaissant et en utilisant l'intention du rédacteur pour mieux cibler la nature du message à transmettre. Lorsque son intention n'est pas connue ou n'est pas assez précise, on devrait disposer de méthodes statistiques pour extraire des données les points saillants qui seront alors à la base du contenu du rapport.

L'étape de sélection du médium permet de déterminer la meilleure façon d'exprimer chaque élément d'information du rapport. Certains messages passent mieux de façon graphique, certains s'expriment mieux de façon purement textuelle, alors que d'autres profitent des deux médias. Par exemple, une évolution globale se présente mieux graphiquement alors qu'un message insistant sur une augmentation très localisée passe mieux dans un texte. Un message indiquant la moyenne des valeurs d'une variable peut utiliser les deux médias: une phrase dans le texte donnant la valeur précise, associée à une ligne pointillée dans le graphique situant les autres valeurs par rapport à cette moyenne.

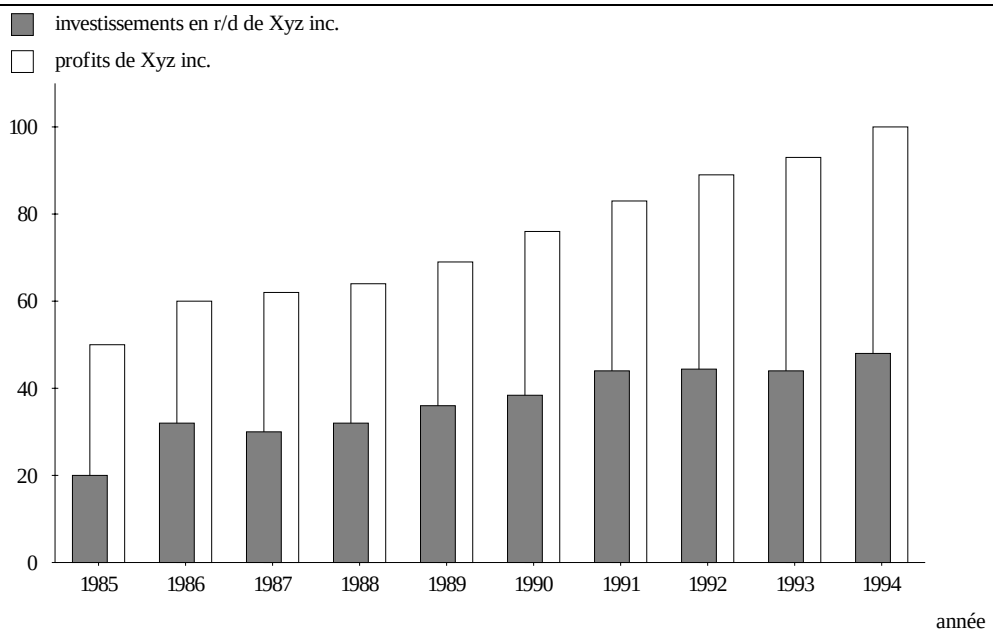
L'étape d'organisation du discours permet non seulement de déterminer l'ordre des informations à présenter mais aussi les liens entre celles-ci. Les liens entre les sources d'informations (textes et graphiques) sont toujours importantes mais dans un long rapport, l'ordre et la structure sont primordiales. Le séquencement des informations peut être d'ordre temporel (par saison dans un rapport de commerce), d'ordre géographique (par région dans un rapport de chômage) ou encore par contenu (par sujet). Lorsque la quantité d'information à présenter est très grande, il est très important de tenir compte de l'organisation par sections et de faire des divisions comme indiqué ci-dessus. Lorsqu'on présente moins d'information, les liens entre les informations deviennent beaucoup plus importants car ils déterminent de façon complète l'organisation du rapport. Ainsi, dans un court rapport de deux pages, on parlera d'un seul sujet, mais la disposition des données sur ces deux pages sera encore plus importante que si le rapport était plus long car c'est la seule structure visible.

Il existe plusieurs types de liens entre les unités d'information d'un rapport. La plus évidente est celle qui lie un texte explicatif à un graphique. En effet, on associe souvent un court texte explicatif à un graphique, utilisant ainsi le graphique pour montrer une vue d'ensemble et le texte pour insister sur les détails importants. On retrouve aussi, entre autres, des liens implicites entre deux graphiques complémentaires (par exemple, deux graphiques alignés sur une même variable) et des liens de transition explicites entre deux messages différents. La figure 5.1 montre un exemple d'une telle transition entre un message d'évolution et un message de corrélation.

Dans un modèle idéal, la dernière phase, la réalisation, n'a plus qu'à se servir d'un générateur de surface textuel et d'un logiciel de graphiques statistiques pour terminer le processus de génération. Toute l'information importante a été déterminée dans les phases précédentes et il ne reste qu'à appliquer tous ces choix pour visualiser le résultat.

5.3 Un modèle plus réaliste

Dans notre description de l'architecture idéale d'un générateur de rapports, quelques points sont restés flous. En effet, bien que la modularité d'un tel modèle soit très intéressante, on voit difficilement où certains processus importants entrent en jeu. Par exemple, on peut se demander à quel moment on fait le choix du type de graphique. Logiquement, ce choix devrait se faire quelque part entre la sélection de médium et la réalisation finale. À ce sujet, on remarque un point très important qui nous fait réfléchir sur la validité du modèle "idéal": le choix du graphique n'est pas ponctuel. En effet, pour choisir le meilleur graphique, on a besoin de connaître au moins le choix de médium, la structure du rapport, ainsi que les contraintes de réalisation. Ainsi, on doit savoir si on a un graphique seul ou un jumelage texte/graphique, on doit considérer que certaines combinaisons de graphiques fonctionnent mieux ensemble, contribuant ainsi à une meilleure structure du rapport et on doit aussi tenir compte des limites physiques du médium (à partir de combien de colonnes doit-on passer à



La figure ci-dessus montre l'évolution des profits de la compagnie Xyz inc. et de ses investissements en recherche et développement entre 1985 et 1994. Tous deux ont augmenté graduellement pendant la période en question (50 et 28 millions respectivement). Il semblerait que l'augmentation des profits soit liée à l'augmentation des investissements en r/d. Cette tendance est confirmée par la figure ci-dessous.

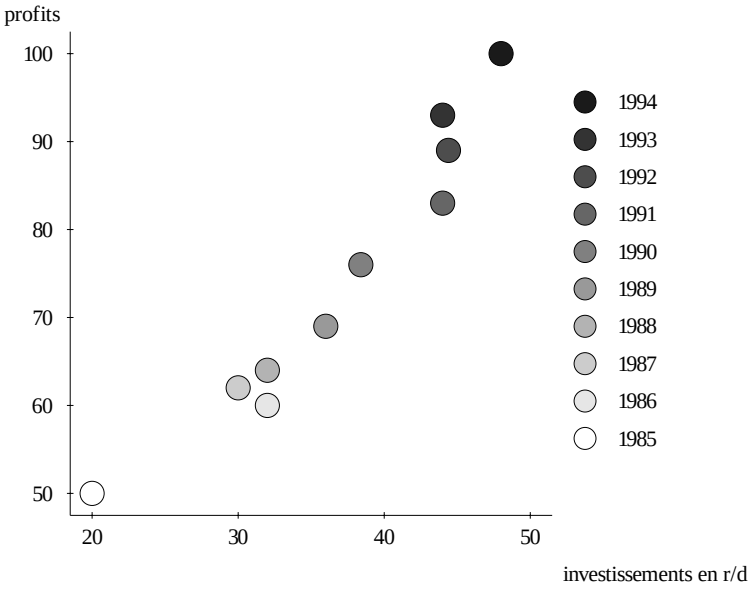


FIG. 5.1 - *Transition entre évolution et corrélation*

la courbe pour des raisons de lisibilité?). Ce point remet donc en question l'architecture modulaire présentée auparavant car elle montre qu'un aspect fondamental du processus est distribué dans beaucoup de modules. Les modules perdent donc leur indépendance et deviennent difficiles à remplacer.

Un autre problème du modèle linéraire (tâches consécutives indépendantes) vient du fait qu'il est trop déterministe. En effet, pour qu'un tel modèle puisse bien fonctionner, chaque module doit agir comme un oracle. Si un module prend une mauvaise décision, on ne peut plus revenir en arrière et il faut essayer de compenser dans un des modules suivants. Ce problème se présente aussi dans les systèmes de génération de texte mais le niveau de rétroaction est plus faible dans ces systèmes, puisqu'ils n'ont pas de contraintes de réalisation graphiques.

Dans les rapports statistiques, le choix des graphiques vient compliquer les choses. En effet, comme nous l'avons énoncé précédemment, ce choix se fait à plusieurs niveaux. Ainsi, un choix peu raffiné est possible après la sélection du médium et ce choix devient de plus en plus raffiné jusqu'à la réalisation finale où on peut tenir compte de contraintes pratiques comme la taille de la page ou le nombre de couleurs de l'imprimante. Pour minimiser les effets de ce problème, deux approches sont possibles: procéder par raffinements successifs ou essayer de faire le bon choix dès le début. Les raffinements sucessifs peuvent se faire soit en gardant une liste de candidats à chaque niveau ou en utilisant un mécanisme de retour-arrière. Ces deux options sont équivalentes: elles ne sacrifient pas tellement la modularité du système mais elles sont peu efficaces. Par contre, on peut essayer de faire le bon choix le plus tôt possible incluant des heuristiques à tous les niveaux. Ces heuristiques brisent l'indépendance des modules en tentant de prédire les choix faits par les modules suivants mais ce désavantage est largement compensé par le gain d'efficacité qu'elles apportent au processus de génération.

Une heuristique, tout en restant simple, peut aider beaucoup à prendre les bonnes

décisions. Par exemple, on peut associer une taille approximative à chaque type de graphique (en général, les courbes sont très compactes mais les tableaux prennent beaucoup de place) de façon à éviter de choisir un graphique qui risque de dépasser les contraintes d'espace. La vraie taille d'un graphique n'est connue que lorsque le module de dessin le réalise mais il est utile d'en avoir une approximation dans les phases de planification.

Malheureusement, l'utilisation d'heuristiques décentralise les connaissances d'un module et rend la mise à jour ou le remplacement d'un module difficile. Par exemple, lorsqu'on modifie la réalisation d'un type de graphique du système, on doit tout d'abord modifier le module de réalisation mais on doit aussi en vérifier les conséquences sur les heuristiques des autres modules. De telles corrections pourraient être nécessaires si on a ajouté de la couleur à un graphique ou si on a déplacé la position de sa légende, changeant peut-être ainsi sa géométrie.

5.4 Choix et compromis dans le système PostGraphe

Dans notre prototype, le système PostGraphe, nous avons essayé de trouver un compromis permettant de garder l'implantation simple, efficace et de produire des résultats valables. L'aspect modulaire du modèle de génération idéal est celui qui a le plus souffert. En effet, pour obtenir un niveau d'efficacité raisonnable, nous avons introduit beaucoup de codépendances entre les modules sous forme d'heuristiques. Afin de ne pas trop compliquer l'implantation, nous avons simplifié certains processus, en particulier le choix du médium. Après avoir examiné de nombreux rapports, nous avons constaté que le texte et les graphiques étaient souvent utilisés ensemble pour transmettre un même message. Puisque le but de notre recherche était surtout l'étude de l'intégration du texte et des graphiques, nous avons décidé d'éliminer le choix de médium et de toujours générer une paire texte/graphique pour chaque message. Par contre, le contenu de chacun varie en fonction du contenu de l'autre et du message à présenter. Au niveau du réalisateur, nous n'avons pas pu nous contenter

d'une solution simple. Nous aurions préféré sacrifier un élément d'aussi bas niveau plutôt que de simplifier le choix de médium. L'utilisation d'un logiciel de dessin indépendant aurait beaucoup simplifié notre implantation. Au cours de notre recherche, nous avons essayé d'utiliser le système X-Lisp-Stat [Tierney, 1990] ainsi que les primitives graphiques de LaTeX [Lamport, 1985]. Malheureusement, en pratique, trop de choix de haut niveau dépendent de détails aussi simples que le nombre de couleurs ou le positionnement d'étiquettes textuelles dans un graphique. En implantant le réalisateur de toutes pièces, nous avons pu l'intégrer dans la chaîne de décisions du système, permettant ainsi des oracles plus précis ou au moins une approche retour-arrière en cas de difficulté majeure.

Nous décrirons maintenant les étapes suivies par le système pour produire un rapport, en insistant sur l'évolution des données de leur entrée dans le système jusqu'au résultat final.

Dans l'entrée du système PostGraphe, on retrouve 3 annotations qui viennent complémenter les données brutes. Il s'agit d'informations sur le type des variables à traiter, sur l'existence de clés relationnelles parmi celles-ci, ainsi qu'une série de directives exprimant l'intention du rédacteur. La valeur théorique de ces informations a été présentée au chapitre 4 et leur syntaxe précise en Prolog sera présentée au chapitre 6. Voici maintenant une description de leur utilisation dans PostGraphe:

5.4.1 Les types

Le système de types permet d'associer à chaque variable du problême un ensemble de propriétés ainsi qu'une unité. Comme on l'a vu au chapitre 4, les propriétés sont organisées en un graphe d'héritage multiple. Parmi les groupes de propriétés traitées, on retrouve l'organisation, le domaine, les propriétés temporelles, les propriétés de format (ou nature) des variables, les propriétés de mesure, ainsi qu'un ensemble de propriétés servant à décrire des entités spécifiques. Les propriétés sont paramétrées, ce qui permet de préciser leur fonction. Par exemple, la propriété énumération du

groupe des contraintes sur le domaine accepte un paramètre qui spécifie les valeurs possibles de la variable énumérée qu'on décrit.

Dans l'entrée, on spécifie la propriété (classe) "mère" de chaque variable et une liste de propriétés auxiliaires. Les propriétés auxiliaires ont priorité sur celles qui sont héritées de la classe mère. Ainsi, on peut redéfinir une partie des caractéristiques d'une variable, ou encore définir des types temporaires en combinant des propriétés d'autres types. L'algorithme de recherche effectue le parcours suivant du réseau de propriétés: d'abord la classe mère, puis chacune des propriétés auxiliaires, ensuite un parcours en profondeur des réseaux de chaque propriété auxiliaire, et finalement le parcours en profondeur du réseau de la classe mère.

En plus des mécanismes de base décrits plus haut, le système de type effectue des opérations d'inférence et des affectations automatiques. Les opérations d'inférence permettent de traverser les liens d'héritage dans le sens inverse dans certains cas bien précis. Par exemple, une variable ordinale est ordonnée et symbolique (par héritage), mais une règle d'inférence indique que la condition nécessaire et suffisante pour être ordinale est d'être ordonnée et symbolique. Ainsi, on peut faire une inférence dans les deux sens dans ce cas bien précis (ces règles sont peu nombreuses mais très utiles).

D'autre part, les affectations automatiques permettent d'adapter dynamiquement le système de types aux données. En effet, le module de lecture des données peut ajouter automatiquement des propriétés auxiliaires qui auront priorité sur celles de la classe mère. Cette caractéristique est présentement utilisée au niveau des contraintes sur le domaine pour ajuster les intervalles et énumérations aux valeurs présentes dans les données. Par exemple, si les données contiennent des références à 3 mois de l'année, alors la propriété énumération sera ajoutée aux propriétés auxiliaires avec la liste de ces 3 mois comme paramètre. Ainsi, sans perdre l'information sur le nom des 12 mois de l'année (dans la classe mois), on précise cette information pour la variable de l'exemple.

Le traitement des unités est assez simple. En effet, on dispose d'un mécanisme qui permet d'associer une unité à une propriété (ex: pourcentage $\mapsto$ %). Cette association est optionnelle et un algorithme se charge de trouver l'unité lorsque le lien n'existe pas. L'algorithme qui trouve l'unité d'une variable se sert du graphe d'héritage des propriétés. Cependant, le graphe est navigué en mode simple (on ne regarde que la première superclasse qui est supposée la plus proche). Si on ne trouve pas d'unité, alors on prend le nom de la propriété comme unité.

5.4.2 Les clés relationnelles

Le chapitre 4 présente les clés relationnelles et leur utilité au niveau du choix et de l'organisation de graphiques. PostGraphe est capable de calculer automatiquement les clés comme dans l'exemple du tableau 4.2. Cependant, les résultats obtenus ne sont pas toujours conformes à la sémantique des variables impliquées. En effet, dans des exemples réels, les variables s'appellent rarement A,B,C,D. Elles représentent plutôt des unités et objets de la vie courante (des années, des profits, ...) auxquels on associe un certain sens. Le tableau 5.1 montre un exemple ou la façon habituelle de calculer les clés donne des résultats un peu étranges. Si on ne tient pas compte de la nature des variables et qu'on suit la définition de clé, on trouve que {profits} est une clé de la relation complète {année,compagnie,profits} (car toutes les valeurs de profits sont différentes). C'est correct selon la définition mais pas de façon générale lorsqu'on regarde la nature des variables en question. En effet, on s'intéresse à voir les profits exprimés en fonction d'autre que chose, plutôt que l'inverse. Dans ce cas-ci la clé intéressante pour la relation est {année,compagnie}. On voudrait donc pouvoir exclure *profits* de la clé tout en laissant le système faire le calcul automatique.

Pour traiter ce problème, on spécifie deux informations supplémentaires (optionnelles) dans l'entrée du système: la liste de variables qui peuvent servir de clés et la liste de variables qui peuvent ne pas servir de clés. À l'aide de cette information, le système calcule tous les ensembles {clé, relation} pour des relations maximales. Cela

Année	Compagnie	Profits
1970	Xyz inc.	3 000 000
1975	Xyz inc.	4 000 000
1979	Xyz inc.	5 000 000
1970	Pqr inc.	20 000 000
1975	Pqr inc.	24 000 000
1979	Pqr inc.	23 000 000
1970	Abc inc.	7 000 000
1975	Abc inc.	6 000 000
1979	Abc inc.	3 000 000

TAB. 5.1 - *Exemple de relation pour le calcul automatique de clés*

nous donne un contrôle assez fin sur quelles relations sont liées à quelles clés sans nous obliger à les écrire explicitement. Ainsi, on peut forcer l'exclusion de certaines variables des clés ou encore limiter ces variables à être dans les clés.

5.4.3 Les intentions du rédacteur

L'intention du rédacteur est l'information qui permet à PostGraphe de choisir quoi dire, et jusqu'à un certain point, comment le dire. Cette information est organisée en sections correspondant aux sections dans le rapport final. Dans l'entrée du système, on l'exprime sous forme d'une liste de listes de prédicats où chaque sous-liste correspond à une section du rapport. Notre algorithme de planification, décrit plus en detail à la section 5.4.4, réorganise l'information à l'intérieur des sections mais n'agit pas du tout entre les sections. Il est donc possible d'isoler les intentions par thèmes en étant certain que le planificateur n'essaiera pas de les mélanger.

Chaque intention peut être vue comme une contrainte sur l'expressivité des figures choisies. On essaie de trouver le plus petit ensemble de figures qui couvre efficacement

les intentions de l'usager. Les intentions utilisées dans notre implantation sont celles que nous présentons à la section 4.2.1 sauf que les modificateurs de *évolution* ne sont pas utilisés, afin de simplifier le système.

5.4.4 Planification

Dans PostGraphe, la planification par schémas est utilisée autant pour les graphiques que pour le texte. Puisqu'on a décidé de transmettre chaque message à l'aide des deux médias, on planifie toujours en deux phases: les graphiques, puis le texte dont le contenu s'adapte au graphique. L'algorithme utilisé s'inspire de l'algorithme utilisé par MacKinlay dans APT [Mackinlay, 1986a].

Tout d'abord, voici un petit rappel sur l'algorithme de MacKinlay: on part d'un ensemble de variables typées (quantitative, ordinale, nominale) et on dispose d'une table indiquant les méthodes graphiques les plus appropriées pour exprimer chaque type de variable. Il existe plusieurs façons d'exprimer chacune d'elles et il faut trouver une figure qui les traite toutes ou séparer les variables en sous-groupes plus faciles à traiter. On vérifie ensuite si le résultat est réalisable et si c'est le cas on le retourne immédiatement. L'algorithme n'essaie pas directement de maximiser l'expressivité finale mais suppose que les variables sont listées par ordre d'importance (il alloue les meilleures méthodes d'expression aux premières et les autres prennent ce qui reste).

Cette méthode présente plusieurs défauts assez importants. En effet, elle ne tient compte que d'un ensemble très limité de types (3), elle travaille sur des variables individuelles plutôt que sur des relations globales et elle ne traite que des concepts graphiques. Le traitement des variables individuelles est très intéressant car il permet non seulement d'étudier la composition élémentaire des graphiques mais aussi de simplifier et d'accélérer l'algorithme. Cependant, il introduit deux problèmes importants: la phase de réalisation qui suit est ambiguë et les phénomènes inter-variables sont ignorés. L'ambiguïté de la réalisation vient du fait que plusieurs graphiques qui

expriment les variables selon les mêmes méthodes peuvent avoir des formats très différents. Par exemple, une courbe, des points, des barres ainsi que des colonnes expriment deux variables à l'aide de leur position le long d'un axe. Cependant, il existe des différences importantes entre ces 4 graphiques (les lignes dans une courbe, les rectangles et leur orientation dans les barres et colonnes). Ces différences permettent entre autres d'exprimer des phénomènes inter-variables comme la comparaison (plus facile sur la courbe que les points), la corrélation (plus facile sur les points que la courbe).

Notre système présente les différences suivantes: il ne part pas d'une liste de variables à exprimer mais plutôt d'un ensemble d'intentions (inter- ou intra-variables) à satisfaire. Le résultat de notre algorithme est un schéma pour chaque groupe d'intentions. Les schémas en question servent aussi bien pour générer du texte que des graphiques. Par exemple, pour comparer les variables A et B et voir l'évolution de A par rapport C, utiliser le schéma X. De plus, il n'impose pas d'ordre particulier car tous les choix qu'il fait sont pondérés, ce qui lui permet d'avoir une fonction globale de "qualité" maximisable. Cette maximisation complique l'exploration des solutions car il n'est plus possible de retourner la première solution réalisable. À priori, il faudrait essayer tous les groupes d'intentions pour trouver ceux qui peuvent être le mieux exprimés tout en essayant de minimiser le nombre de ces sous-groupes. En utilisant une série d'heuristiques, on peut réduire considérablement le nombre de groupes candidats, évitant ainsi une explosion combinatoire.

Pour des raisons d'efficacité, nous avons donc découpé le processus en 4 étapes:

1. **Phase de groupement.** À l'aide d'une heuristique, on identifie les intentions les plus "compatibles" et on les groupe. La compatibilité entre intentions a été définie de façon empirique en essayant de minimiser les groupements inutiles tout en maximisant les groupes utiles. La construction des groupes commence avec des groupes de 1 intention et essaie à chaque itération de construire des

groupes ayant une intention de plus qu'à l'itération précédente. Cette phase est très rapide et identifie la plupart des groupes intéressants. Cependant, elle laisse passer certaines combinaisons moins fréquentes, et cela a parfois comme conséquence de trouver plusieurs fois le même schéma dans le rapport pour des intentions différentes. La phase 4 a été introduite pour éliminer ce problème. Il est très difficile de considérer toutes les possibilités intéressantes sans causer une explosion combinatoire.

2. **Phase de composition.** Cette phase utilise une table, tout comme l'algorithme de MacKinlay, pour trouver comment exprimer graphiquement chaque intention. Le but est de trouver si chaque groupe d'intentions est réalisable et à l'aide de quels types de schémas. Les entrées de la table sont pondérées, ce qui fait que le résultat de cette phase est une liste de figures candidates triées de la plus à la moins efficace. Voici, en guise d'exemple, 3 entrées de la table. La première entrée indique que l'intention `comparaison([A,B],[X])` est exprimée avec une pondération de 80 par le schéma de réalisation `colonnes3` et fait intervernir la variable X suivie de la liste de variables Ys (à condition que A et B appartiennent à Ys).

comparaison([A,B],[X])	colonnes3	80	`[X|Ys]`	A ∈ Ys, B ∈ Ys
comparaison([Y],[X])	barres1	100	`[X,Y]`	
evolution(A, X)	courbe2	100	`[X|Ys]`	A ∈ Ys

3. **Phase de vérification et réalisation.** Lorsqu'on arrive à cette phase, il ne reste qu'à vérifier si les schémas d'expression sont réalisables et à appeler les routines de génération de bas niveau. Un schéma d'expression peut ne pas être réalisable pour des raisons purement physiques (la figure associée est trop grosse, on n'a pas assez de tons de gris distinguables, ...), à cause de contraintes sur les types ou les clés relationnelles, ou parce que certaines contraintes de la table ci-dessus n'ont pas pu être verifiées auparavant et échouent ici. Les informations sur les types et sur les clés sont aussi très utiles pour l'organisation des données à passer au module de réalisation.

4. **Phase de post-optimisation.** Cette phase contrôle l'insertion des schémas dans le rapport final après leur réalisation. Elle est inévitable vu les approximations apportées par les heuristiques. Lorsque la phase de groupement ne met pas ensemble deux intentions qui peuvent être exprimées par le même schéma, on retrouve deux fois le même schéma dans le résultat. Il faut donc éliminer cette redondance en éliminant un des deux et en fusionnant les ensembles d'intentions traités par chacun.

5.5 Un rapport généré automatiquement par PostGraphe

Nous présentons ici un exemple d'entrée (figure 5.2) et de sortie de PostGraphe. L'entrée est formulée sous forme d'un terme Prolog dont la syntaxe sera décrite en détail à la section 6.1.1. L'exemple présente les profits des compagnies A, B et C de 1987 à 1990. Les intentions du rédacteur se divisent en deux sections: la première doit présenter les 3 variables impliquées, c'est-à-dire *année*, *compagnie* et *profits*. La seconde doit présenter la comparaison des profits entre les compagnies et l'évolution des profits par rapport aux années.

PostGraphe génère ses rapports sous forme d'un document LaTeX [Lamport, 1985] avec figures PostScript [Adobe Systems Incorporated, 1990b; Adobe Systems Incorporated, 1985] incorporées. La sortie du système contient des annotations spéciales qui ne seraient pas affichées dans un produit fini. Ces informations permettent de retracer facilement les choix effectués par PostGraphe et donc de détecter les erreurs plus facilement. Il s'agit d'informations sur les intentions utilisées dans chaque section et dans chaque figure et aussi sur le type de schéma utilisé pour chaque figure ou texte. En plus, pour chaque intention utilisée dans une figure ou un texte, un entier entre parenthèses indique la qualité du choix effectué par rapport à cette intention. La valeur est comprise entre 0 et 100. Plus elle est élevée, plus l'intention a été réalisée

```
data(% noms des variables
     [année,compagnie,profits],
     % types des variables
     [annee/[symbolique],
      etiquette,
      dollar/[pluriel(profit)]],
     % candidats pour les clés
     [année,compagnie],
     % non-candidats pour les clés
     [profits],
     % intentions du rédacteur
     [% section 1
      [presentation(année),
       presentation(compagnie),
       presentation(profits)],
      % section 2
      [comparaison([profits],[compagnie]),
       evolution(profits,année)]],
     % les données brutes
     [[1987,'A',30],
      [1988,'A',35],
      [1989,'A',40],
      [1990,'A',35],
      [1987,'B',160],
      [1988,'B',165],
      [1989,'B',140],
      [1990,'B',155],
      [1987,'C',50],
      [1988,'C',55],
      [1989,'C',60],
      [1990,'C',95]]).
```

FIG. 5.2 - *Exemple d'entrée de* PostGraphe

correctement.

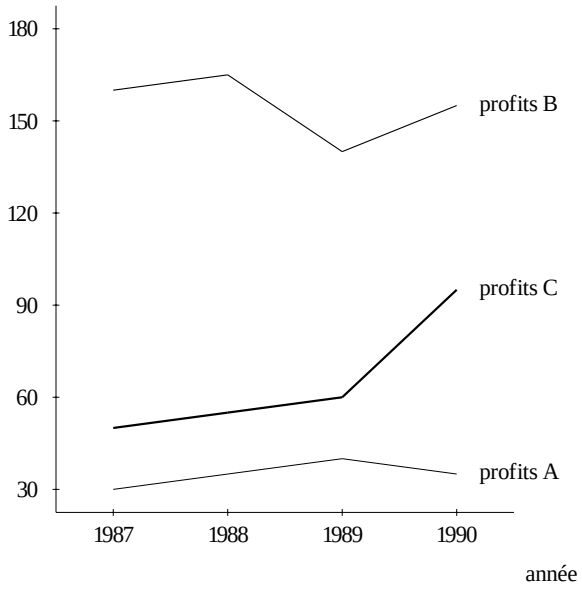
La sortie complète est présentée à l'annexe A. Nous ne présentons à la figure 5.3 qu'une version simplifiée sans les annotations. Dans la première section du rapport, PostGraphe a choisi d'utiliser un tableau, ce qui lui a permis de présenter les données associées à toutes les variables sans transmettre de message particulier. Par contre, dans la seconde section, le système a décidé de superposer 3 courbes dans la même figure, de façon à transmettre le mieux possible le groupe de deux intentions demandées par l'utilisateur, c'est-à-dire comparaison et évolution des profits. Parmi les graphiques candidats, les courbes superposées ont été choisies car elles représentent le meilleur compromis entre la comparaison et l'évolution. Ces deux intentions ont été exprimées avec des efficacités respectives de 69 et 95. Le résultat pour la comparaison n'est pas excellent, mais celui pour l'évolution compense largement, ce qui ramène la moyenne d'efficacité à 72 pour ce graphique.

Grâce au système de pondération associé au choix des schémas, il est possible d'altérer le comportement de PostGraphe en utilisant des annotations spéciales dans l'entrée. Ces annotations et leur effet sur l'algorithme de planification seront expliquées en détail au chapitre 6. En voici une explication rapide: les deux annotations utilisées, `>` et `*`, sont associées à des intentions pour indiquer respectivement un seuil minimal de satisfaction ou une pondération différente pour l'intention annotée. Les annotations `>50` et `*1` sont utilisées par le système lorsque l'usager ne spécifie pas de préférence. Ainsi, la seconde série d'intentions de l'exemple de la figure 5.2 correspond à:

```
[comparaison([profits],[compagnie])>50*1,
 evolution(profits,année)>50*1]
```

Si on décide de changer la pondération de la première intention, on change son importance dans le calcul du meilleur groupement d'intentions. Ceci a pour effet de changer le choix du schéma le plus approprié. Par exemple, si on multiplie cette pon-

année	1987	1988	1989	1990
compagnie	profits	profits	profits	profits
A	30	35	40	35
B	160	165	140	155
C	50	55	60	95



De 1987 à 1989 les profits de la compagnie A ont augmenté de 30 $ à 40 $. Jusqu'en 1990 ils ont diminué de 40 $ à 35 $.

De 1987 à 1988 les profits de B ont augmenté de 160 $ à 165 $. Pendant 1 année ils ont diminué de 25 $. Jusqu'en 1990 ils ont augmenté de 140 $ à 155 $.

De 1987 à 1990 les profits de C ont augmenté de 50 $ à 95 $.

FIG. 5.3 - *Exemple de sortie de* PostGraphe

dération par 10, on va obtenir la figure 5.4 à la place des courbes de la figure 5.3. L'agencement des colonnes dans cette nouvelle figure favorise la comparaison, tout en montrant aussi l'évolution. Les résultats de 80 et 60 pour cette figure lui donnent une valeur globale de 78.2 $((80 * 10 + 60)/11)$, alors que celle des courbes de la figure 5.3 serait ici de 71.4 $((69 * 10 + 95)/11)$. Sans la pondération de 10 pour la comparaison, les résultats respectifs auraient été de 70 et 72, favorisant ainsi les courbes.

Tout en gardant une pondération de 10 pour la comparaison, il est possible d'imposer un seuil pour l'évolution. Ainsi, en spécifiant un résultat minimal de 65 pour l'expression de l'évolution, on écarte la figure 5.4 car son résultat est de 60. La grande pondération pour la comparaison écarte aussi les courbes de la figure 5.3. C'est donc la figure 5.5 qui est générée.

Si on remplace la pondération de 10 par un seuil de 80 pour la comparaison, le système est incapable de produire une seule figure satisfaisant toutes les contraintes. Il doit donc générer deux figures indépendantes. Ces deux figures utilisent les mêmes graphiques que les figures 5.4 et 5.3.

Il est important de noter que le haut degré de liberté présent dans cet exemple aurait été éliminé si on s'était plutôt intéressé aux moyennes des profits. Ainsi, comme on peut le voir aux figures 5.6 et 5.7, le groupement des deux intentions devient impossible avec les moyennes, ce qui élimine les variations précédentes.

5.6 Résumé

Dans ce chapitre, nous avons présenté l'architecture idéale d'un générateur de rapports statistiques. Cette architecture est composée d'un ensemble de modules indépendants qui fonctionnent de façon déterministe. Nous avons ensuite expliqué que certains facteurs présents dans la génération multimédias rendent cette architecture inefficace. En particulier, le choix de type de figure est très difficile à isoler et nécessite

```
[comparaison([profits],[compagnie])*10,
 evolution(profits,année)]
```

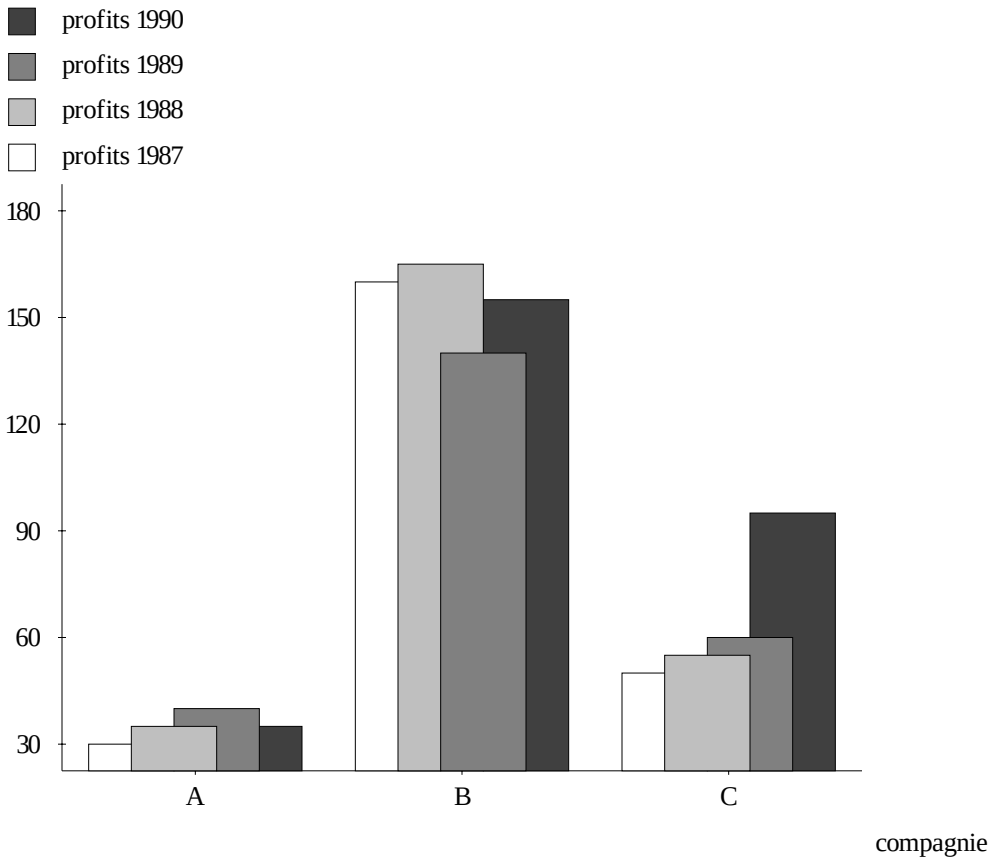


FIG. 5.4 - *[Schéma:* colonnes2*]. comparaison de profits entre compagnie (80). évolution de profits par rapport à année (60).*

```
[comparaison([profits],[compagnie])*10,
 evolution(profits,année)>65]
```

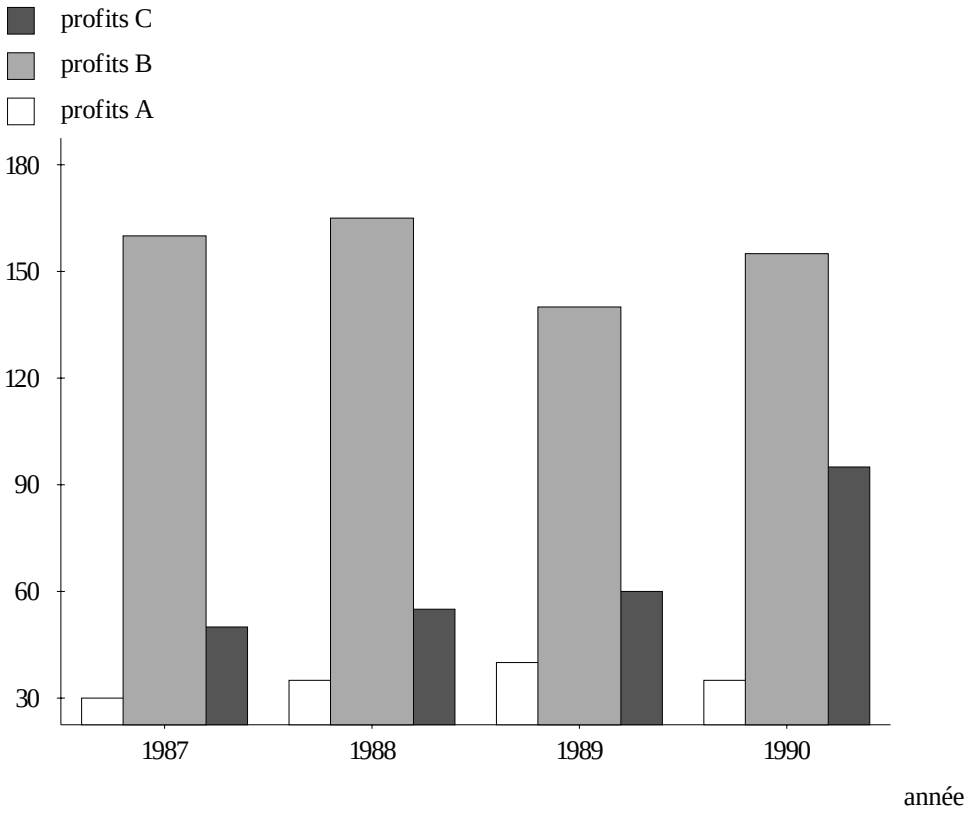


FIG. 5.5 - *[Schéma:* colonnes2*]. comparaison de profits entre compagnie (70). évolution de profits par rapport à année (85).*

```
[comparaison([reduce(moyenne,profits)],[compagnie]),
 evolution(reduce(moyenne,profits),année)]
```

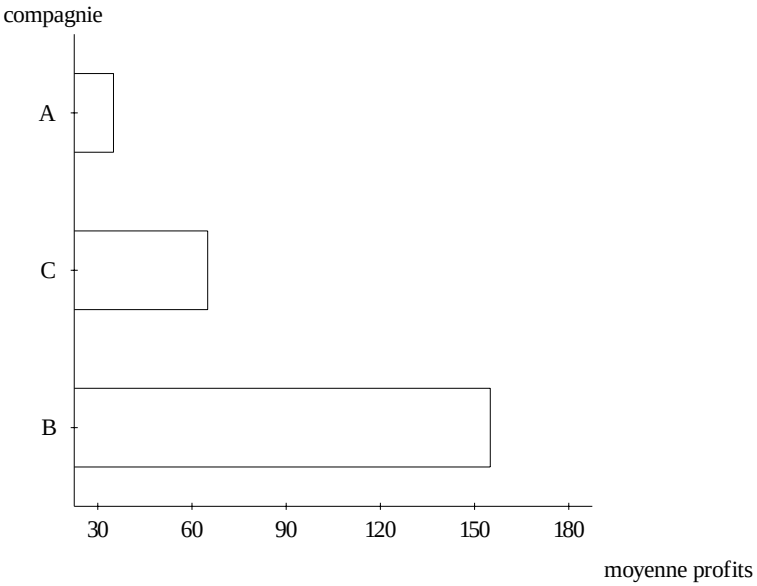


FIG. 5.6 - *[Schéma:* barres1*]. comparaison de la moyenne de profits entre compagnie (100).*

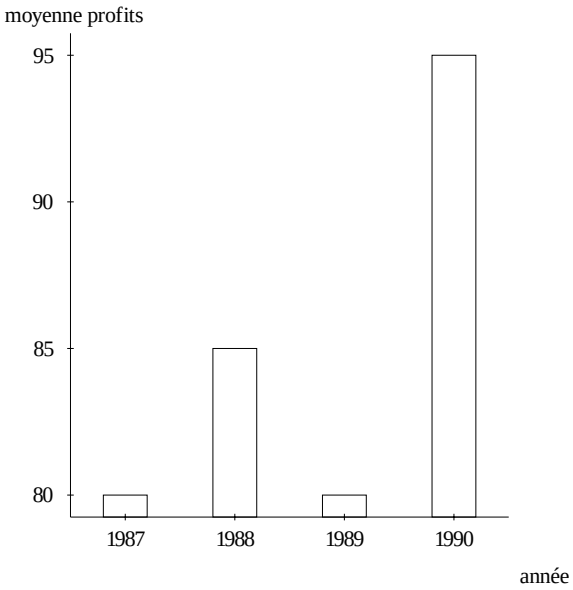


FIG. 5.7 - *[Schéma:* colonnes1*]. évolution de la moyenne de profits par rapport à année (94).*

la coopération de plusieurs modules ou beaucoup de non-déterminisme.

Nous avons ensuite montré les compromis qui étaient nécessaires pour réaliser un générateur de rapports statistiques multimédias. Nous avons présenté les aspects importants de notre système PostGraphe, tout en restant à un niveau très algorithmique. Dans le chapitre suivant, nous présentons des détails beaucoup plus techniques sur notre prototype, en utilisant plutôt un découpage par module Prolog.

Chapitre 6

L'implantation de PostGraphe

Comme nous l'avons vu au chapitre précédent, nous avons réalisé PostGraphe surtout pour vérifier notre modèle théorique sur la génération de rapports statistiques multimédias. Dans ce même chapitre, nous avons montré qu'il avait été nécessaire d'ignorer certains aspects du modèle théorique afin de simplifier le prototype. Le modèle a été simplifié surtout en profondeur plutôt qu'en largeur. Ainsi, les différents aspects de notre théorie se reflètent dans les modules de PostGraphe, sans toutefois être développés dans tous leurs détails.

Le système PostGraphe est implanté en Prolog et se divise en 5 modules principaux (≈3700 lignes) ainsi qu'un ensemble de fonctions utilitaires (≈1000 lignes). De plus, le système utilise le générateur de texte PréTexte [Gagnon, 1993] (≈10000 lignes), lui aussi écrit entièrement en Prolog. Les modules principaux de PostGraphe sont:

1. La saisie de données (≈250 lignes)
2. Le système graphique (≈2000 lignes)
3. Le traitement des types (≈150 lignes)
4. La planification (≈300 lignes)
5. Les schémas (figures et texte) (≈900 lignes)

Dans ce chapitre, nous passerons en revue les modules du prototype en présentant leur interface extérieure ainsi que les détails de leur implantation en Prolog.

6.1 La saisie de données

Dans PostGraphe, la saisie des données est un processus actif, c'est-à-dire que le système lit un fichier d'entrée et applique une série de transformations aux données pour les rendre plus faciles à traiter. Les transformations sont très simples et visent soit à déduire certaines annotations manquantes, soit à précalculer certains résultats de façon à accélérer le processus de génération.

6.1.1 L'entrée du système

L'entrée du prototype PostGraphe doit respecter la syntaxe générale suivante:

data(noms,types,clés,non-clés,intentions,données).

Voici maintenant une description de la notation des paramètres d'entrée. Cette notation est une adaptation en Prolog de la notation présentée au chapitre 4. La syntaxe de Prolog est utilisée pour les listes et les termes mais la notation reste équivalente.

Notez que n représente le nombre de variables, a,b,c,d,e et f sont des entiers compris entre 1 et n et i, j, k, l, m et p des entiers quelconques. Les parties entre accolades sont optionnelles.

- *noms:* $[v_1,v_2,\ldots,v_n]$
- *types:* $[t_1,t_2,\ldots,t_n]$
  - t_i: t_i^0 { / [$t_i^1,\ldots,t_i^k$] }
  - t_i^j: $nom\_type_i^0(param_{i\,1}^j,\ldots,param_{i\,p}^j)$

- *clés:* [v_a,v_b,...,v_c]
- *non-clés:* [v_d,v_e,...,v_f]
- *intentions:* [$section_1$,$section_2$,...,$section_l$]
  - $section_i$: [$intention\_ponderee_1$,...,$intention\_ponderee_m$]
  - $intention\_ponderee_i$: $intention_i$ { > seuil_i } { * poids_i }
  - $intention_i$: notation présentée au chapitre 4, exemples au tableau 6.1.
- *données:* [$d_{1\bullet}$,$d_{2\bullet}$,...,$v_{n\bullet}$]
  - $d_{i\bullet}$: [d_{i1},d_{i2},...,v_{in}]

Un exemple d'entrée est présenté à la figure 6.1. Comme on peut le voir dans la notation générale et dans la figure 6.1, l'entrée du système contient les données brutes sous forme de tuples précédées de 4 annotations supplémentaires: le nom de chaque variable (colonne) de données, le type de chaque variable, l'information permettant de calculer les clés relationnelles ainsi que les intentions du rédacteur.

Les noms et les types des variables doivent être listés dans le même ordre que les colonnes de la liste de tuples. Comme on peut le voir dans la notation générale, la syntaxe des types est un peu plus complexe que celle qui est utilisée à la figure 6.1. En effet, la forme générale (`TypePrincipal/[TypeAux1,...,TypeAuxN]`) permet d'exprimer les types auxiliaires d'une variable en plus de son type principal. On utilise le terme *propriété* pour faire référence au type principal et aux types auxiliaires. Comme on l'a vu au chapitre 4, les propriétés peuvent être paramétrées. Dans l'entrée, chacune d'elles est représentée par un terme Prolog dont l'arité dépend du nombre de paramètres de la propriété. Par exemple, le terme `intervalle(De,A)` est utilisé pour représenter la propriété *intervalle* et ses 2 paramètres (borne inférieure et supérieure). Une variable qui possède cette propriété est donc bornée par `De` et `A`.

```
data(% noms des variables
     [mois,province,chômage,activité,emploi],
     % types des variables
     [mois,province,pourcentage,pourcentage,pourcentage],
     % variables pouvant faire partie des clés relationnelles
     [mois,province],
     % variables pouvant ne pas faire partie des clés relationnelles
     [chômage,activité,emploi],
     % intentions du rédacteur (par section)
     [% section 1
      [presentation(mois),
       presentation(province),
       presentation(chômage),
       presentation(activité),
       presentation(emploi)],
      % section 2
      [evolution(reduce(moyenne,chômage),mois),
       evolution(emploi,mois),
       comparaison([emploi],[province]),
       evolution(reduce(moyenne,activité),mois),
       repartition(chômage,[mois,province]),
       proportion([province]),
       repartition(reduce(moyenne,chômage),[mois]),
       repartition(reduce(moyenne,chômage),[province]),
       correlation(activité,emploi),
       comparaison([reduce(moyenne,emploi)],[province]),
       comparaison([reduce(moyenne,activité)],[province]),
       decomposition(reduce(moyenne,emploi)),
       comparaison([reduce(moyenne,chômage),reduce(moyenne,activité)],
                   [mois])]],
     % les données brutes sous forme de tuples
     [[mai,'Terre-Neuve',18.600000,54.000000,43.900000],
      [mai,'Île-du-Prince-Édouard',16.100000,65.600000,55.000000],
      ...
      [mai,'Alberta',8.400000,72.500000,66.400000],
      [mai,'Colombie-Britannique',9.800000,66.200000,59.700000],
      [juin,'Terre-Neuve',20.300000,53.200000,42.400000],
      [juin,'Île-du-Prince-Édouard',17.300000,65.500000,54.200000],
      ...
      [juillet,'Nouvelle-Écosse',12.100000,71.600000,54.100000],
      [juillet,'Nouveau-Brunswick',12.800000,78.600000,51.100000],
      [juillet,'Québec',11.500000,63.600000,56.300000],
      ...
      [juillet,'Alberta',8.500000,63.000000,66.800000],
      [juillet,'Colombie-Britannique',9.900000,66.600000,60.100000]]).
```

FIG. 6.1 - *Exemple d'entrée du système*

Les intentions sont spécifiées par section. Une section de l'entrée reflète directement une section du rapport final. La syntaxe formelle des intentions a été présentée au chapitre 4. Le tableau 6.1 présente quelques exemples illustrant la syntaxe des prédicats Prolog permettant d'exprimer les intentions et indique les quelques différences par rapport à la théorie.

INTENTION DE BASE	SENS
presentation(mois)	La présentation de la variable **mois**. Cette intention s'appelle *lecture* dans la théorie.
comparaison([profits],[région])	La comparaison des **profits** selon les **régions**. Dans la théorie, le premier paramètre est une variable simple.
comparaison([dépenses,profits],[année,mois])	La comparaison en parallèle des **dépenses** et des **profits** pour chaque paire **année/mois**.
evolution(profits,année)	L'évolution des **profits** par rapport aux **années**.
correlation(dépenses,profits)	La corrélation entre **dépenses** et **profits**.
repartition(profits,[province])	La répartition des **profits** par rapport aux **provinces**.
repartition(profits,[province,mois])	La répartition des **profits** par rapport aux paires **province/mois**.
MODIFICATEUR	SENS
decomposition(profits)	Expression des **profits** sous forme fractionnaire dans une comparaison.
proportion([mois])	Expression des **mois** sous forme fractionnaire dans une répartition.
proportion([province,mois])	Expression des paires **province/mois** sous forme fractionnaire dans une répartition.

TAB. 6.1 - *Exemples d'intentions en Prolog*

À cette syntaxe de base s'ajoutent deux annotations optionnelles: le seuil et le poids. En Prolog, une intention `I` peut s'écrire `I>Seuil*Poids`. Ces informations permettent respectivement de rejeter des plans si une intention n'y est pas assez bien exprimée et de donner plus de poids à une intention dans l'évaluation de la qualité d'un plan. Ces détails seront explicités dans la section sur le planificateur.

Mentionnons en terminant qu'au niveau de la saisie des données, PostGraphe ajoute une série de faits à sa base de connaissances pour permettre un accès plus rapide aux différentes composantes de la structure d'entrée. Par exemple, une série de faits Prolog `d(I,J,Valeur)` permet un accès beaucoup plus rapide à la valeur d_{ij} des données, surtout lorsque $i * j$ est élevé. Sans cette optimisation, on devrait parcourir 2 listes de longueurs i et j de façon linéaire pour trouver une valeur.

6.1.2 Les types

Le module de saisie de données interagit avec le module de types à deux niveaux: l'instanciation de types partiels et la précompilation des liens d'héritage.

L'instanciation de types partiels est un mécanisme qui permet de s'assurer que les paramètres de certains types reflètent les valeurs présentes dans l'entrée. Ce mécanisme nécessite une collaboration entre le module de saisie et le module de types. Comme on va le voir à la section 6.2, il est possible de laisser des variables libres dans la description d'un type. Ces variables libres devront être instanciées par le module de saisie en se servant des données. Ce mécanisme est utilisé pour les deux propriétés suivantes:

```
enumeration(Liste)
intervalle(Min,Max)
```

Lors de la lecture des données, le module de saisie interroge le système de types pour savoir si la variable courante hérite d'une des deux propriétés ci-dessus. Si c'est le cas, le module de saisie vérifie si les paramètres des propriétés sont instanciées. Si les paramètres ne sont pas totalement instanciés, il calcule la valeur correspondante et instancie le paramètre.

Pour instancier le paramètre `Liste` ci-dessus, il détermine la liste des valeurs $d_{i\bullet}$ pour la variable courante. Pour `Min` et `Max`, il calcule le minimum et maximum

des $d_{i\bullet}$. Par exemple, dans les données de la figure 6.1, la variable *mois* est de type *mois*. Dans le module de types, on peut voir que le type *mois* hérite, entre autres, de `enumeration(_)`. Puisque ce type est partiellement instancié, le module de saisie calcule la liste de valeurs pour la variable *mois* et instancie la propriété à la valeur `enumeration([mai,juin,juillet])`.

L'instanciation de propriétés se fait à l'aide du mécanisme des propriétés auxiliaires. En effet, comme on va voir à la section 6.2, une propriété auxiliaire a priorité sur les parents du type principal. Il suffit donc d'ajouter à la fin de la liste des types auxiliaires de la variable la propriété à instancier. Dans l'exemple ci-dessus, le type de la variable *mois* deviendra *mois/[enumeration([mai,juin,juillet])]*. Puisque les types auxiliaires sont examinées dans l'ordre, l'ajout se fait à la fin de la liste car l'entrée peut déjà contenir une instanciation manuelle et toute annotation manuelle doit avoir priorité sur les ajouts automatiques.

La seconde interaction entre le module de saisie et le module de types est beaucoup plus simple. En effet, pour chaque variable de l'entrée, le module de saisie demande au module de types de précalculer tous ses liens d'héritage. Cette opération accélère le processus de génération qui interroge souvent le module de types avec les mêmes requêtes de base. Puisque les mêmes requêtes reviennent souvent, il est intéressant de les précalculer. De plus, il est possible de précompiler les types et de modifier d'autres parties indépendantes de l'entrée comme les intentions sans avoir à recompiler à chaque fois.

6.1.3 Les relations

À l'aide de la méthode présentée au chapitre 5, le système calcule un ensemble de clés minimales pour des relations maximales. Il considère aussi dans ses calculs les pseudo-variables statistiques obtenues par réduction (moyenne, minimum, maximum, ...). Il est important de considérer ces variables statistiques même si elles ne sont pas

présentes dans les données car la réduction change les relations. En précalculant ces pseudo-relations, on évite de refaire les calculs à chaque utilisation d'une statistique.

Pour chaque relation, le système calcule aussi un facteur de remplissage. Ce facteur est calculé en déterminant $\mid d_{\bullet j} \mid$, le nombre de valeurs uniques pour chaque variable $d_{\bullet j}$. En faisant le produit de tous les $\mid d_{\bullet j} \mid$, on obtient le nombre maximal de tuples pouvant être construits à l'aide des variables. En divisant le nombre de tuples réels par ce maximum, on obtient le facteur de remplissage. Ce facteur permet, entre autres, de déterminer la proportion de cases vides dans un tableau (voir le chapitre 3).

6.1.4 Les intentions

La seule transformation des intentions effectuée au niveau de la saisie des données est liée au seuil et au poids. En effet, ces informations sont optionnelles et le système doit leur attribuer des valeurs par défaut lorsque celles-ci sont omises. Les valeurs choisies sont 50 pour le seuil et 1 pour le poids. Ces valeurs ont été choisies car elles sont très neutres. En effet, un seuil de 50 n'écarte que les intentions très mal transmises et un poids de 1 ne donne pas d'importance spéciale à une intention. L'utilisation de ces deux annotations sera présentée plus en détail à la section 6.3.2.

6.2 Le traitement des types

Comme on l'a vu au chapitre 4, l'utilisation d'un système de types bien structuré fournit de l'information importante permettant de choisir comment exprimer les intentions de l'usager. Dans ce même chapitre, nous avons présenté le système de types dans notre modèle théorique. Notre système de types est organisé en une série de graphes d'héritage.

Dans notre module de traitement des types, on retrouve 7 prédicats importants (voir tableau 6.2): 3 servent à définir les types et 4 à interroger le système sur les

types. Les 3 prédicats de définition couvrent l'héritage de propriétés, l'inférence de propriétés et les unités des propriétés. Le terme propriété est utilisé pour parler du type principal (classe) ainsi que des types auxiliaires d'une variable. Les règles de notre système d'unités sont présentées en annexe à la section B.1. Les 4 prédicats d'interrogation permettent de déterminer le type principal d'une variable, ses types auxiliaires, ainsi que son unité.

Définition
`heritage_propriete(P,[P1,P2,...,Ph])`
`inference_propriete([P1,...,Pi],P)`
`unite_propriete(P,U)`
Interrogation
`instance_de(Variable,Classe)`
`types_aux_de(Variable,TypesAux)`
`propriete_de(Variable,Propriété)`
`unite_de(Var,Unite)`

TAB. 6.2 - *Prédicats du module de types*

Le prédicat `heritage_propriete/2` est utilisé pour définir une propriété `P` en spécifiant ses parents `[P1,P2,...,Ph]` dans le graphe d'héritage.

Le prédicat d'inférence de propriété `inference_propriete/2` est un outil qui étend les possibilités du graphe d'héritage en permettant d'inférer automatiquement certaines propriétés à partir d'autres. Il indique qu'une variable ayant `[P1,...,Pi]` comme propriétés acquiert automatiquement la propriété P. Ceci simplifie la composition de types dans l'entrée du système et évite des oublis dans cette même situation. Par exemple, soit une variable $d_{\bullet j}$ telle que *plus_grand(0)* $\in T_j^*$ et[1] *plus_petit(10)* $\in T_j^*$. L'existence de la règle

```
inference_propriete([plus_grand(X),plus_petit(Y)],intervalle(X,Y))
```

1. Rappel: T_j^* est l'ensemble des propriétés (directes + héritées) de $d_{\bullet j}$

nous permet d'inférer que *intervalle(0,10)* $\in T_j^*$. Ce type d'inférence se comporte comme de l'héritage inverse dans le cas où une série de propriétés est une condition nécessaire et suffisante à l'existence d'une autre. Ainsi, on retrouve dans la définition du graphe d'héritage le fait

```
heritage_propriete(intervalle(X,Y),[plus_grand(X),plus_petit(Y)])
```

qui représente l'inverse de l'inférence de propriétés correpondante. Avec les deux règles ci-dessus, on peut donc affirmer que *intervalle(X,Y)* $\in T_j^* \equiv$ *plus_grand(Y)* $\in T_j^* \wedge$ *plus_petit(X)* $\in T_j^*$.

Le prédicat `unite_propriete/2` associe l'unité `U` à la propriété `P`. Par exemple, le fait

```
unite_propriete(pourcentage,'%')
```

indique que la propriété *pourcentage* est associée à l'unité *%*.

Pour récupérer de la base de données les propriétés d'une variable, on dispose des deux prédicats suivants:

```
instance_de(Variable,Classe)
types_aux_de(Variable,TypesAux)
```

Le premier associe à une variable son type principal, c'est-à-dire sa classe dans le graphe d'héritage. Ce type est unique. Le second associe une variable à la liste de ses types auxiliaires. Notons que le type principal a une place spéciale mais c'est une propriété tout comme les types auxiliaires. Ces deux prédicats interrogent la base de connaissances de l'entrée et ne font aucune inférence pour déterminer les propriétés héritées (ils retournent donc T_j et non T_j^*). Ils sont utilisés de façon interne par le prédicat `propriete_de/2` mais peuvent être utiles indépendamment de celui-ci.

Le prédicat `propriete_de/2` permet de savoir si une variable possède une propriété (*Propriété* $\in T_j^*$) Ce prédicat parcourt le graphe d'héritage par retour-arrière, permettant ainsi d'énumérer aussi toutes les propriétés dont une variable hérite. De

plus, le prédicat se sert de l'inférence de propriétés pour compléter ses resultats. La figure 6.2 montre le code Prolog commenté de ce prédicat.

```
% si Var est une instance directe de Prop (type principal) alors Prop
% est une propriété de Var.
propriete_de(Var,Prop) :-
    instance_de(Var,Prop).
% si Prop fait partie de la liste des types auxiliaires de Var, alors
% Prop est une propriété de Var.
propriete_de(Var,Prop) :-
    proprietes_de(Var,ListeProp),
    member(Prop,ListeProp).
% s'il existe un chemin dans le graphe entre un des types auxiliaires de
% Var et Prop, alors Prop est une propriété de Var.
propriete_de(Var,Prop) :-
    proprietes_de(Var,ListeProp),
    member(Var2,ListeProp),
    chemin_proprietes(Var2,Prop).
% s'il existe un chemin dans le graphe entre le type principal de
% Var et Prop, alors Prop est une propriété de Var.
propriete_de(Var,Prop) :-
    instance_de(Var,Var2),
    chemin_proprietes(Var2,Prop).
```

FIG. 6.2 - *Code Prolog commenté de* `propriété_de`

Notons ici qu'on a examiné le type principal avant les types auxiliaires, puis récursivement les parents des types auxiliaires avant ceux du type principal. Cela reflète le fait qu'on ne peut pas changer la nature même (classe) d'une variable mais que l'on peut altérer ses propriétés en lui ajoutant des propriétés auxiliaires dans l'entrée du système. Ce mécanisme est très utile au niveau de l'entrée des données car il permet la redéfinition très précise de certaines propriétés, même si celles-ci sont héritées de façon très éloignée. Par exemple, on peut indiquer au système de traiter des nombres comme des étiquettes en ajoutant la propriété *étiquette* dans la liste des types auxiliaires d'une variable dont le type principal hérite de *nombre*.

Le prédicat `unite_de/2` permet d'interroger le système sur l'unité d'une variable.

L'unité est utilisée lorsqu'on imprime une valeur de la variable (ex: 40 cm). L'héritage des unités est beaucoup plus direct que celui des propriétés. En effet, le graphe d'héritage est parcouru en mode simple en considérant comme seul point de départ la classe (type principal) de la variable. Pour effectuer le parcours en mode simple, on ne considère que le premier parent dans les liens d'héritage.

De plus, une règle par défaut permet d'utiliser le nom du type de base comme nom de l'unité lorsqu'aucune unité n'a été spécifiée. Le parcours très simple du graphe est dû en partie à cette règle par défaut. En effet, on évite l'héritage multiple pour que l'unité reste très proche de la variable en question. S'il n'est pas possible de trouver une unité en mode simple, on préfère retomber sur le cas par défaut qui utilise le nom de la variable. Cela évite de toujours hériter les unités de propriétés génériques comme *nombre* ou *étiquette* dont presque toutes les variables héritent par héritage multiple. Les unités des propriétés génériques sont définies comme des chaînes vides car, par exemple, lorsqu'on ne connait pas la nature exacte d'une variable de valeur 30 héritant de *nombre* il est préférable d'écrire *30* plutôt que la forme bizarre *30 nombres*. Par contre, on doit utiliser la règle par défaut et écrire *30 voitures* plutôt que *30* lorsqu'on sait que le type de l'objet est *voiture*.

6.3 La planification

L'algorithme général de planification de PostGraphe, tel que présenté au chapitre 5, se divise en 4 étapes:

1. Phase de groupement d'intentions

2. Phase d'évaluation de la composition

3. Phase de vérification et réalisation

4. Phase de post-optimisation

Nous allons maintenant présenter les détails d'implantation de chacune de ces étapes ainsi que les structures de données impliquées.

6.3.1 Groupements d'intentions

La phase de groupement d'intentions s'occupe de déterminer les intentions qui peuvent être combinées de façon à produire des intentions composées. Le type de groupement effectué ici est de type *superposition* (voir chapitre 4).

Le groupement automatique d'intentions est un compromis qui permet de réduire le nombre de messages à produire tout en essayant de minimiser la perte d'expressivité. Puisque chaque message produit en général une figure et une description textuelle, en minimisant leur nombre, on réduit la taille du rapport. À cette étape, on essaie de prévoir quels groupes seront intéressants, alors que les étapes suivantes du processus déterminent avec précision si le compromis est valable. Puisqu'on utilise une heuristique, les choix que l'on fait ne sont pas toujours exacts. Il est possible de choisir un groupement candidat qui sera rejeté plus loin tout comme il est possible de laisser tomber un groupement qui aurait été acceptable. L'étape de groupement, bien que peu précise, est cependant extrèmement utile car elle évite une explosion combinatoire du nombre de groupes. Le code Prolog de notre heuristique est présenté à la figure 6.3. Le prédicat `groupable/2` vérifie la compatibilité des deux intentions qui lui sont passées comme paramètres. Les règles de notre heuristique ont été déterminées de façon empirique en mesurant quels groupements contribuaient aux étapes suivantes de l'algorithme.

6.3.2 Évaluation du groupement d'intentions

Le but de cette étape est de trouver le meilleur schéma d'expression capable d'exprimer chaque ensemble d'intentions. Pour réaliser efficacement cette tâche, nous avons bâti une table d'inférence permettant de trouver rapidement tous les schémas exprimant une intention. La table nous donne en plus une mesure de la qualité du

```
% 2 évolutions sur la même variable
groupable(evolution(C,B),evolution(A,B)) :-
    % attention aux réductions! ex: on ne veut pas grouper l'évolution
    % de A et l'évolution de la moyenne de A (ça n'aurait aucun sens)
    not(reduce_clash(C,A)),!.
% une comparaison d'une liste de variables (incluant A) par
% rapport à B et une évolution de A par rapport à B.
groupable(comparaison(L,[B]),evolution(A,B)) :-
    member(A,L),!.
% comparaison et évolution d'une même variable par rapport à 2
% variables différentes.
groupable(comparaison([A],[C]),evolution(A,B)).
% comparaison et correlation par rapport à la même variable.
groupable(comparaison([A],[B]),correlation(A,B)) :- !.
% comparaison et décomposition d'une variable.
groupable(comparaison([A],[B]),decomposition(A)) :- !.
% comparaison de 2 ensembles de variables par rapport à B.
groupable(comparaison(L1,[B]),comparaison(L2,[B])) :-
    % on vérifie s'il y a problème avec les réductions
    L1=[A|_],L2=[C|_],
    not(reduce_clash(C,A)),
    !.
% répartition et proportion par rapport à A.
groupable(proportion(A),repartition(_,A)) :- !.
% présentation de 2 variables.
groupable(presentation(A),presentation(B)) :-
    % attention aux réductions!
    not(reduce_clash(A,B)),!.
```

FIG. 6.3 - *Heuristique de superposition d'intentions*

résultat ainsi que les contraintes à vérifier avant de l'appliquer.

La table d'inférence est représenté par les prédicats Prolog `c_fig/6` pour les graphiques et `c_txt/6` pour le texte. La syntaxe du prédicat est la suivante:

```
c_fig(Intention,Schéma_instance,Schéma_classe,Variables,Qualité,Contraintes)
c_txt(Intention,Schéma_instance,Schéma_classe,Variables,Qualité,Contraintes)
```

L'intention utilise la syntaxe habituelle. La classe de schéma représente le type d'objet utilise pour exprimer le message (ex: courbe, tableau). L'instance de schéma est le schéma particulier de cette classe qui s'applique à l'intention considérée. Par convention, nous notons les instances avec le nom de la classe suivie d'un entier (ex: courbe7). La liste de variables est celle qui sera passée en paramètre au module de schémas 6.4 avec le nom de l'instance. C'est la liste des variables impliquées dans le message. La qualité est un entier de 0 à 100 indiquant si l'expression de l'intention par ce schéma est bonne. Plus la valeur est élevée, plus la qualité est bonne. Comme nous l'avons mentionné à la section 6.1.4, une valeur de 50 représente une qualité à peine acceptable. Les contraintes sont une liste d'éléments de la forme `[V1 / V2]` signifiant que lorsque `V1` et `V2` sont instanciées, il faut vérifier que `V1` appartient à la liste `V2`. Si les deux variables ne sont pas instanciées, il y a gel de la contrainte en attendant une phase de l'algorithme qui pourra les instancier.

Voici un exemple commenté d'une ligne de la table `c_fig`:

```
c_fig(comparaison([Y], [T]), points6, points, [X,Y|Z], 55, [T / Z]).
```

Cette entrée signifie que le schéma `points6` de la classe `points` permet d'exprimer l'intention `comparaison([Y], [T])` dans une figure utilisant une liste de variables `[X,Y|Z]` à condition que `T` appartienne à la liste `Z`. La qualité de l'expression est de 55 (passable). La table complète est présentée en annexe à la section B.2. Le choix des facteurs de qualité a été fait de façon empirique en s'inspirant toutefois beaucoup des résultats théoriques présentés aux chapitres 3 et 4.

La phase d'évaluation du groupement des intentions procède donc de la façon suivante pour chaque groupe d'intentions candidat:

1. Trouver dans la table un schéma qui exprime la première intention. Noter la qualité et les contraintes.

2. Vérifier si les autres intentions peuvent être représentées avec le même schéma. Noter la qualité et les contraintes.

3. Si la qualité tombe en bas du seuil (voir section 6.1), échouer.

4. Si toutes les intentions peuvent être exprimées avec le même schéma, celui-ci reçoit un facteur de qualité obtenu en faisant la moyenne pondérée des facteurs de qualité pour chaque intention. La pondération vient de la saisie des données (section 6.1).

5. Répéter le tout avec un autre schéma de départ.

6. Lorsque tous les groupes candidats supérieurs au seuil ont été calculés, ils sont triés et passés à la phase suivante pour vérification.

6.3.3 Vérification et réalisation

Lorsqu'on arrive à cette étape, on dispose d'une liste de candidats très prometteurs ordonnés par facteur de qualité. Le travail est presque terminé. Tout ce qu'il reste à faire, c'est de vérifier de façon encore plus stricte les groupements. Pour ce faire, on instancie directement les schémas de réalisation dont les préconditions procédurales se chargent de tester l'applicabilité du schéma en question. Idéalement, on ne devrait pas avoir beaucoup d'échecs. En pratique, il y en a beaucoup plus que prévu. La raison est simple: la table d'inférence ne vérifie pas les valeurs des variables car elle travaille au niveau des intentions. Il est donc possible d'avoir des échecs imprévus. Par exemple, un schéma de colonnes peut échouer parce qu'on doit y présenter 50 valeurs. Lors d'un échec, on passe au candidat suivant dans la liste. Dans l'exemple

précédent, on passerait à la courbe après les colonnes.

Les schémas seront décrits à la section 6.4.

6.3.4 Post-Optimisation

La dernière phase de l'algorithme de planification est une phase de post-optimisation. Cette phase est nécessaire à cause des approximations faites en début de processus (les groupements initiaux). En effet, l'heuristique de groupement ne considère pas tous les cas; il est donc possible de traiter séparément des intentions qui pourraient s'exprimer par le même schéma. Si l'algorithme les exprime par des schémas différents, on accepte ce choix. Si, par contre, ils sont exprimés par le même schéma, on détecte la redondance et on fait un groupement pour éviter d'avoir plusieurs fois la même figure ou le même texte dans le rapport.

En phase de post-optimisation, un groupement se fait en éliminant un des schémas redondants et en ajoutant ses intentions à l'autre.

6.4 Les schémas (figures et texte)

Dans notre système, les schémas sont représentés par les prédicats Prolog `figure/4` pour les figures et `texte/4` pour le texte. La syntaxe est la suivante:

```
figure(Schéma_instance,Schéma_classe,Variables,Procédure)
texte(Schéma_instance,Schéma_classe,Variables,Procédure)
```

Les 3 premiers paramètres ont été décrits à la section sur la table d'inférence. Le dernier paramètre sert à retourner une procédure qui pourra être évaluée au moment opportun pour réaliser le texte ou la figure associé. Le code Prolog des schémas du système est présenté en annexe C.

Le corps d'un schéma se divise en 3 sections informelles:

– Préconditions

– Transformation

– Exécution

La partie *préconditions*, sert à effectuer des tests divers sur l'applicabilité du schéma. Tout prédicat Prolog est permis. Par exemple, dans le schéma *colonnes2*, on retrouve les tests suivants:

```
type(X,symbolique),
type(Y,symbolique),
relcle(_,[X,Y],L),member(Z,L),
```

À l'aide de ces tests, on vérifie d'abord que les variables X et Y sont symboliques. Puis on vérifie que X,Y est une clé d'une relation incluant Z. En d'autres mots, on vérifie que X,Y est une clé de X,Y,Z.

La partie transformation est parfois utilisée pour passer d'une configuration à une autre de façon à utiliser une même procédure graphique pour deux schémas différents. Voir *colonnes5* dans l'annexe pour un exemple de transformation.

Finalement, la partie exécution prépare les paramètres pour les routines graphiques de PostGraphe ou pour les routines textuelles de PréTexte. La procédure obtenue n'est pas exécutée tout de suite mais retournée sous forme encapsulée dans le 4^e paramètre du prédicat du schéma.

Dans, les schémas textuels, l'interface avec le réalisateur est plus complexe qu'au niveau des schémas graphiques. En effet, puisque le réalisateur de graphiques est incorporé dans PostGraphe, ses primitives ont été conçues en fonction des besoins du système et donc nécessitent une interface minimale. Ce n'est pas le cas pour le texte, puisque nous utilisons un générateur indépendant qui utilise une syntaxe et

des structures très différentes. Pour plus de détails sur l'interface entre les schémas textuels et le générateur de texte, voir la section 6.6.

6.5 Le système graphique

Notre système graphique comporte 3 modules. Le module principal se charge des traitements de plus haut niveau (graphiques statistiques et tableaux) alors que les 2 autres s'occupent de détails de plus bas niveau (graphiques simples et chaînes de caractères).

6.5.1 Traitement des graphiques statistiques et tableaux

Au coeur de ce module, on retrouve le prédicat `pic/1`. Celui-ci a comme unique argument une structure décrivant le type de graphique à réaliser. Chaque structure correspond à une forme particulière de tableau ou de graphique; un exemple d'utilisation ainsi qu'un résultat graphique est présenté pour chaque figure en annexe C. Les structures possibles sont présentées au tableau 6.3. La forme générale d'une de ces structures est `G(V1,...,VN,L1,...,LM,R,Opt)` où les paramètres représentent:

- `G`: le type de graphique.
- `V1,...,VN`: les noms des variables impliquées. Pour les graphiques acceptant un nombre non-fixe de variables, on remplace un `Vi` par une liste de noms. Dans ce cas particulier, nous plaçons un "s" à la fin du nom de la variable Prolog pour rappeler qu'elle représente une liste.
- `L1,...,LM` ($M \leq N$): les valeurs des `Vi` correspondants. Ces valeurs servent à produire les échelles, ce qui explique leur absence partielle dans le cas des tableaux. Selon la nature de la variable, deux formes sont possibles:
  - `enumeration([X1,X2,...,XK])` pour une liste de valeurs discrètes.
  - `intervalle(XMin,XMax)` pour un intervalle continu de XMin à XMax.

Tout comme pour les `Vi`, les `Li` peuvent être elles aussi sous forme de liste (la même notation est utilisée).

- `R`: la relation complète (liste de tuples).
- `Opt`, s'il existe peut être
  - un nom de style ou de variante de figure (ex: flèches pour les colonnes).
  - une liste d'entiers `i` indiquant les tuples `Ti` de la relation à mettre en relief (ex: pour la tarte).

On retrouve plusieurs variantes dans chaque catégorie. Les modules en amont dans le processus de génération s'occupent de choisir la forme optimale. Notons que ce module peut être utilisé de façon indépendante à partir d'un autre système Prolog. Ce prédicat imprime le code PostScript [Adobe Systems Incorporated, 1990b; Adobe Systems Incorporated, 1985] correspondant à la figure.

La figure 6.4 montre une des règles Prolog du prédicat `pic` pour le graphique en barres. Dans cet exemple, on distingue deux catégories de prédicats: les prédicats de type *ps* et ceux de type *gr*. Les prédicats de type *ps* seront décrits en détail à la section 6.5.2. Ceux de type *gr* représentent les outils élémentaires du module de traitement des figures. Ils permettent de réaliser les composants d'une figure (les axes, les légendes, ...) ou bien de calculer la position et les dimensions de ceux-ci (position des axes, distance entre colonnes, ...). Le prédicat `map` utilisé régulièrement dans le système est une adaptation logique de son équivalent dans les langages fonctionnels. `map(+Pred,+XIn,+XOut,+LIn,-LOut)` instancie successivement `XIn` avec chaque valeur de la liste `LIn`. Après chaque instanciation, `Pred` est évalué et `XOut` est ajouté à une liste temporaire. À la fin du processus, cette liste est retournée dans `LOut`. Habituellement, `Pred` utilise `XIn` comme paramètre d'entrée et `XOut` comme paramètre de sortie. Par exemple, `map((Y is X+1),X,Y,[1,2,3],L)` va instancier `L` à `[2,3,4]`. Dans l'exemple de la figure 6.4, l'appel à `map` n'utilise pas les paramètres `XOut` et

TAB. 6.3 - *Types de graphique/tableau*

`LOut` car on ne veut que produire un effet de bord, dans ce cas une impression, pour chaque élément de la liste `V`.

```
pic(barres(X,Y,LX,LY,R)) :-
    % nouvelle figure
    ps_debut,
    % dessiner les axes
    gr_axe(horizontal,LX,PX,LAX),
    gr_axe(vertical,LY,PY,LAY),
    % dessiner les légendes des axes
    gr_legende(horizontal,X,LAX),
    gr_legende(vertical,Y,LAY),
    % dessiner une barre pour chaque tuple de données
    map((% trouver la position de la donnée courante sur les 2 axes
         gr_axe_pos(PX,A,PA),
         gr_axe_pos(PY,B,PB),
         % déterminer la largeur de la barre (même code que pour colonne)
         gr_colonne_delta(PY,Delta),
         D2 is Delta/4,
         PB1 is PB-D2,
         PB2 is PB+D2,
         % dessiner la barre
         ps_line(0,PB1,PA,PB1),
         ps_line(PA,PB1,PA,PB2),
         ps_line(PA,PB2,0,PB2)),
        [A,B],_,R,_),
    % fin de la figure
    ps_fin.
```

FIG. 6.4 - *Code pour le graphique en barres*

6.5.2 Traitement des graphiques simples en PostScript

Les prédicats du système graphique dont le nom commence par `ps_` représentent la couche de plus bas niveau du système graphique. Ils permettent de générer le code PostScript [Adobe Systems Incorporated, 1990b; Adobe Systems Incorporated, 1985] correspondant à des éléments graphiques très simples (lignes, rectangles, cercles, ...). On peut les remplacer par des prédicats équivalents qui utilisent une autre interface

que PostScript, comme par exemple les primitives graphiques de LaTeX [Lamport, 1985]), sans modifier les fonctions de type *gr*. Nous avons opté pour PostScript à cause de sa très grande puissance et flexibilité tant pour l'impression sur papier que pour l'affichage sur écran. Le tableau 6.4 présente les prédicats disponibles dans ce module. Un exemple d'utilisation est présenté aux figures 6.5 et 6.6.

`ps_dot(X,Y)`	Un point
`ps_dot(X,Y,Forme)`	Un point à forme spéciale (6 formes)
`ps_circle(X,Y,Rayon)`	Un cercle
`ps_circle(X,Y,Rayon,Gris)`	Un cercle rempli d'un ton de gris
`ps_rectangle(XMin,YMin,XMax,YMax)`	Un rectangle
`ps_rectangle(Gris,XMin,YMin,XMax,YMax)`	Un rectangle rempli d'un ton de gris
`ps_box(RectEnglob)`	Un rectangle englobant (2 ou 4 points)
`ps_poly(Points,Gris)`	Un polygone rempli d'un ton de gris
`ps_fleche(X1,Y1,X2,Y2)`	Une flèche verticale
`ps_fleche(Gris,X1,Y1,X2,Y2)`	Une flèche verticale remplie d'un ton de gris
`ps_secteur(X,Y,Rayon,A1,A2,Style)`	Un secteur de l'angle A1 à A2 à style variable
`ps_line(X1,Y1,X2,Y2)`	Une ligne
`ps_line(Epaisseur,X1,Y1,X2,Y2)`	Une ligne à épaisseur variable

TAB. 6.4 - *Prédicats PostScript*

6.5.3 Traitement des chaînes de caractères en PostScript

À cause de sa complexité, le traitement des chaînes de caractères n'a pas été inclus dans le module précédent. La complexité est due à la gestion du positionnement de chaque caractère. Ainsi, le module doit se servir d'une base de faits Prolog indiquant les dimensions et propriétés exactes de tous les caractères des polices utilisées.

Les 2 prédicats fournis par ce module sont:

```
ps_show(+Font,+Size,+X,+Y,+PX,+PY,+Ss,-[X1,Y1,X2,Y2])
ps_rshow(+Angle,+Font,+Size,+X,+Y,+PX,+PY,+Ss,-[X1,Y1,X2,Y2,X3,Y3,X4,Y4])
```

Ils permettent respectivement d'afficher une chaîne de caractères (représentée par `Ss`, une liste d'atomes Prolog) horizontalement ou à un angle arbitraire. Ces fonctions

```
% debut de la figure
ps_debut,
% le repere
ps_line(0,-200,0,200), ps_line(-200,0,200,0),
% 7 points (quartier Bas-Gauche du dessin)
ps_dot(-60,-150),
map((Y is -120+Style*20,ps_dot(-60,Y,Style)),
     Style,_,[0,1,2,3,4,5],_),
% 4 cercles (Haut-Gauche)
ps_circle(-50,160,20),
map((Y is 50+Gris*100,ps_circle(-150,Y,10,Gris)),
    Gris,_,[0,0.3333,0.6666,1],_),
% 2 rectangles (Bas-Droite)
ps_rectangle(60,-60,100,-40),
ps_rectangle(0.5,60,-100,100,-140),
% un polygone (Haut-Droite)
ps_poly([[100,100],[120,100],[130,140],[120,155],[90,160]],0.2),
% 2 fleches (Bas-Droite)
ps_fleche(160,-75,180,-10),
ps_fleche(0.5,160,-100,170,-160),
% 3 secteurs (Haut-Droite)
ps_secteur(-60,60,40,30,75,0),
ps_secteur(-60,60,40,75,175,1),
ps_secteur(-60,60,40,175,240,2),
% 3 lignes (Haut-Droite)
ps_line(90,20,130,20),
ps_line(4,90,40,130,40),
ps_line(8,90,60,130,60),
% 4 chaines de caracteres (Bas-Gauche)
ps_rshow(58,-150,-50,right,top,['Ceci est ','un test'],Box),
ps_box(Box),
ps_show(-150,-150,left,top,['Ceci est ','un test'],_),
ps_show(-150,-150,center,bottom,['Ceci est ','un test'],_),
% fin de la figure
ps_fin
```

FIG. 6.5 - *Exemple d'utilisation des prédicats PostScript*

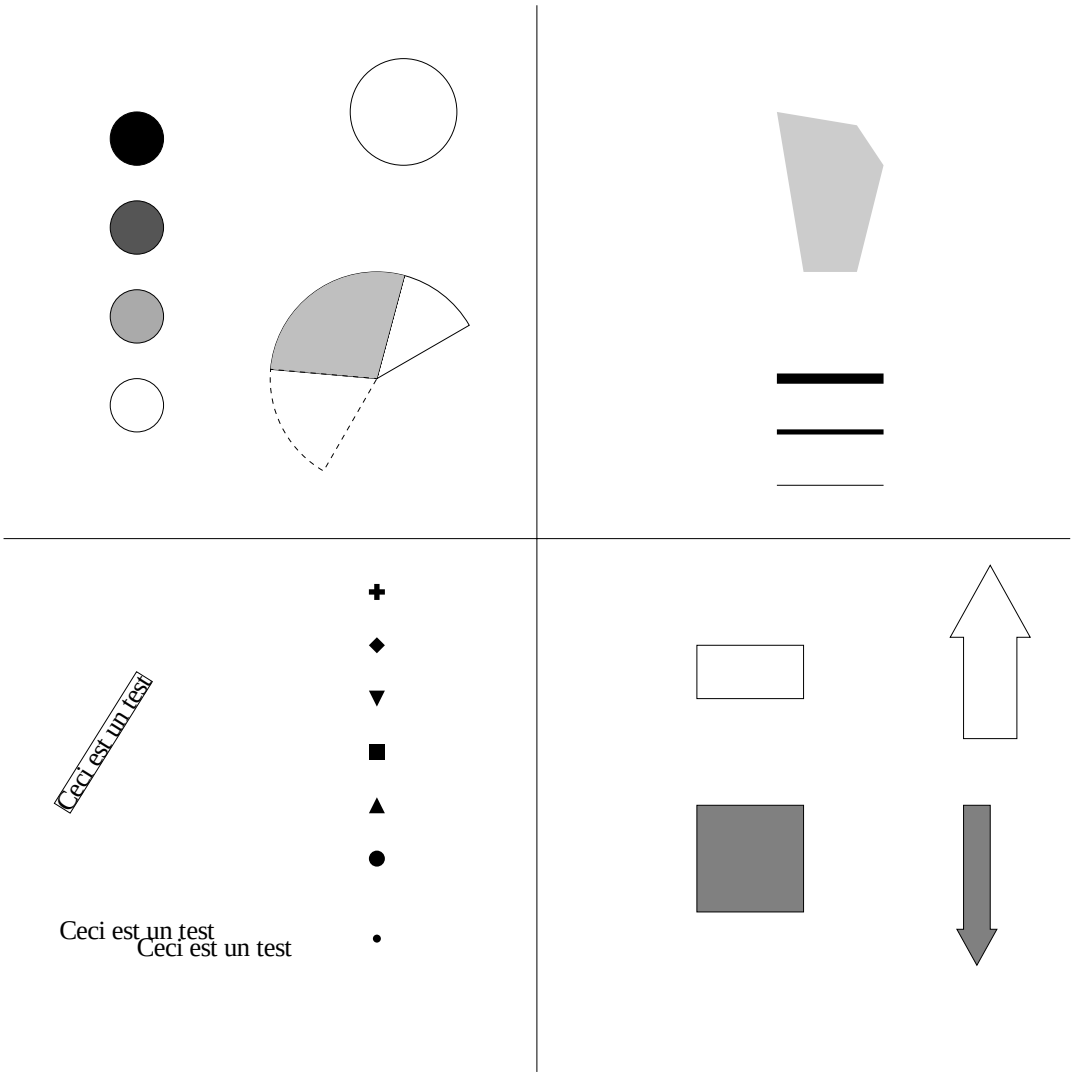


FIG. 6.6 - *Résultat de l'exemple de la figure 6.5*

sont disponibles sous plusieurs formes écourtées, mais les formes présentées ci-dessus permettent le maximum de contrôle. Il est possible de spécifier la police de caractères (`Font`), sa taille en points (`Size`), la position de la chaîne (`X` et `Y`), son alignement (centre, haut, bas, gauche ou droite) par rapport à `X`,`Y` (`PX` et `PY`). Le prédicat `r_show` reçoit en plus l'angle de rotation (`Angle`). Les 2 prédicats retournent le rectangle englobant (*bounding box*) de la chaîne (voir figure 6.7). La forme simple de rectangle englobant est définie par seulement 2 points (les abscisses et ordonnées minimales et maximales de l'objet à englober). C'est la plus utilisée car elle définit un rectangle avec un angle nul par rapport au repère. Ainsi, elle permet de mesurer l'espace occupé par une figure quelconque sur une feuille de papier ou un écran. La forme plus générale définit un rectangle dont l'orientation correspond à celle de l'objet à englober (par exemple, une courbe ou un mot). On doit alors spécifier les 4 points du rectangle. Pour obtenir la forme simple à partir de la forme à 4 points, il suffit de prendre le minimum et le maximum des abscisses et ordonnées. La forme à 4 points est intéressante lorsqu'on veut encadrer ou souligner un objet dessiné à un angle quelconque. Un exemple d'utilisation de ces deux prédicats est présenté aux figures 6.5 et 6.6.

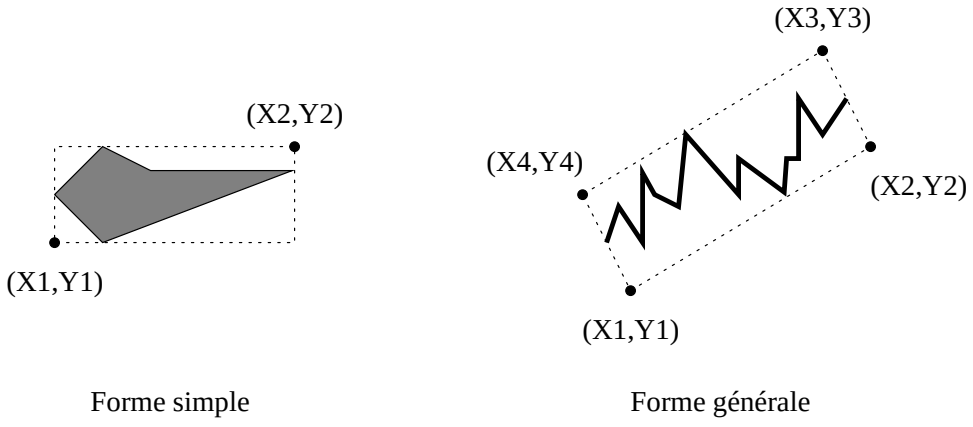


FIG. 6.7 - *Le rectangle englobant*

La base de connaissances nécessaire au positionnement des caractères est très complexe mais inévitable. Nous avons donc mis au point un ensemble d'outils à l'aide du langage Perl [Wall, 1993] pour la générer automatiquement à partir des spécifica-

tions contenues dans une police PostScript [Adobe Systems Incorporated, 1990a]. La figure 6.8 montre un extrait de notre base de données. Les 4 faits Prolog utilisés sont `charmap/2`, `charbox/4`, `kernpair/4` et `fontbox/2`. Voici maintenant un aperçu de leur syntaxe ainsi qu'une description de leur fonction et paramètres:

```
charmap(Code_Dans_L_Alphabet,Nom_Caractere_PostScript).
```

Le fait `charmap` associe un code dans l'alphabet utilisé (ISO-Latin-1 dans notre cas) à un nom de caractère PostScript. En remplaçant les faits `charmap`, il est possible de fonctionner avec d'autres alphabets comme celui du MacIntosh.

```
charbox(Nom_Police_PostScript,Nom_Caractere_PostScript,
        Espacement_Inter_Caracteres,
        [Rectangle_Englobant_X_Min,
         Rectangle_Englobant_Y_Min,
         Rectangle_Englobant_X_Max,
         Rectangle_Englobant_Y_Max]).
```

De son côté, `charbox` donne l'espacement inter-caractères ainsi que le rectangle englobant (voir la figure 6.7) pour un caractère d'une police particulière. L'espacement inter-caractères est défini comme étant le déplacement horizontal du pointeur courant après l'affichage du caractère. En pratique, cela revient souvent à mesurer la distance entre la partie gauche du caractère courant et celle du suivant. Notons que l'espacement inter-caractères est souvent un peu plus grand que la largeur du rectangle englobant pour éviter que les caractères ne se touchent.

```
kernpair(Nom_Police_PostScript,
         Nom_Caractere_PostScript_Gauche,
         Nom_Caractere_PostScript_Droit,
         Ajustement_Horizontal).
```

À son tour, `kernpair` indique l'ajustement entre des paires de caractères d'une police; par exemple, on colle beaucoup le A et le V à cause de leurs formes inclinées, tandis que A et p ne le sont pas du tout.

```
fontbox(Nom_Police_PostScript,
        [Rectangle_Englobant_X_Min,
         Rectangle_Englobant_Y_Min,
         Rectangle_Englobant_X_Max,
         Rectangle_Englobant_Y_Max]).
```

Finalement, le fait `fontbox` indique le rectangle englobant maximal pour les caractères d'une police.

Les unités utilisées dans ces faits correspondent à celles d'une police PostScript standard. Ainsi, pour une police de taille X points, X équivaut à 1000 unités. Par exemple, pour une police de 12 points, un espacement inter-caractères de 500 équivaut à 6 points.

```
charmap(62,'greater').
charmap(63,'question').
charmap(64,'at').
charmap(65,'A').
charmap(66,'B').
charmap(67,'C').
...
charbox('Times-Roman','x',500,[17,0,479,450]).
charbox('Times-Roman','y',500,[14,-218,475,450]).
charbox('Times-Roman','yacute',500,[14,-218,475,678]).
charbox('Times-Roman','ydieresis',500,[14,-218,475,623]).
charbox('Times-Roman','yen',500,[-53,0,512,662]).
charbox('Times-Roman','z',444,[27,0,418,450]).
charbox('Times-Roman','zcaron',444,[27,0,418,674]).
charbox('Times-Roman','zero',500,[24,-14,476,676]).
...
kernpair('Times-Roman','A','U',-55).
kernpair('Times-Roman','A','V',-135).
kernpair('Times-Roman','A','W',-90).
kernpair('Times-Roman','A','Y',-105).
kernpair('Times-Roman','A','p',0).
...
fontbox('Times-Roman',[-168,-218,1000,898]).
```

FIG. 6.8 - *Spécifications de polices PostScript*

6.6 Le générateur de texte

6.6.1 Généralités sur PréTexte

La composante de réalisation textuelle utilisée par PostGraphe est une version modifiée du générateur PréTexte [Gagnon, 1993]. PréTexte est basé sur la théorie de la grammaire systémique [Berry, 1975; Berry, 1976; Matthiessen et Bateman, 1991] et se spécialise dans le traitement de l'information temporelle, tant au niveau des locutions adverbiales que des verbes. PréTexte a été choisi pour 2 raisons: premièrement, il est entièrement écrit en Prolog, ce qui a facilité son intégration avec PostGraphe; deuxièmement, il est spécialisé en traitement de l'information temporelle et ce genre d'information est très utile dans les rapports statistiques, puisqu'on y exprime souvent des évolutions de quantités dans le temps.

Trois sources d'information sont utilisées par PréTexte au cours de son processus de génération: l'*environnement*, la *grammaire* et le *tableau noir*. L'environnement contient des informations sur le message à transmettre ainsi qu'une base de données sémantiques et lexicales. La grammaire contient le réseau systémique et le tableau noir contient de l'information dynamique sur la structure syntaxique à produire.

Pour décrire le message à transmettre, on doit spécifier sa représentation conceptuelle ainsi que sa représentation sémantique. La représentation conceptuelle comprend une description des objets de l'univers et leurs propriétés non-temporelles, une description des constantes et occurrences temporelles, ainsi que les relations entre ces constantes et occurrences. La représentation sémantique contient une description des messages à exprimer, et fixe pour chacun le moment d'énonciation, la référence temporelle, la perspective temporelle, la localisation temporelle ainsi que le centre d'attention. Des exemples d'entrée pour la version non-modifiée de PréTexte sont disponibles dans [Gagnon, 1993]. Un exemple d'entrée pour notre version modifiée sera présenté plus loin (figures 6.9 et 6.10).

Dans la base de connaissances, on retrouve le dictionnaire, qui décrit les concepts à exprimer, ainsi que le lexique, qui décrit comment les exprimer.

La grammaire contient le réseau systémique de PréTexte et se divise en 4 catégories d'information: les descripteurs de systèmes, les connexions entre systèmes, les règles de sélection et les règles de réalisation. Les descripteurs de systèmes indiquent les traits de chaque système et un trait par défaut. Les connexions indiquent quels autres systèmes doivent être explorés lorsqu'un trait est choisi dans un système. Les règles de sélection sont des tests permettant de déterminer quel trait doit être choisi dans un système. Les règles de réalisation sont des procédures à réaliser lorsqu'un trait est choisi. Elles permettent la mise à jour d'informations globales, incluant des présélections de traits pour d'autres systèmes.

Le *moteur* de PréTexte contrôle le processus de génération et obtient toute l'information dont il a besoin à l'aide d'un *module d'interface*. Le module d'interface permet d'isoler le moteur du format des données, permettant ainsi des modifications faciles à ce niveau sans répercutions majeures sur le reste du système.

Le moteur ajoute de l'information au tableau noir lors de la production des constituants et le *solveur* se sert de cette information pour produire leur structure finale.

6.6.2 Modifications à PréTexte

Les modifications effectuées lors de l'adaptation de PréTexte pour PostGraphe sont décrites ici. Pour plus de détails sur les autres aspects de PréTexte, en particulier le moteur du générateur, voir [Gagnon, 1993].

La structure du message donnée en entrée à PréTexte a dû être modifiée pour tenir

compte des formes syntaxiques utiles à PostGraphe. Les figures 6.9 et 6.10 montrent un exemple d'entrée dans lequel apparaissent ces modifications.

Tout d'abord, au niveau de la description des propriétés non-temporelles des objets, les traits suivants ont été ajoutés:

id(I) Permet d'identifier un objet.

connu Indique que l'objet est connu (déterminé).

de(D) Indique un lien d'appartenance.

quantite(Q) Indique une quantité précise ou floue (pluriel, singulier).

Au niveau de la description des occurrences, les traits suivants ont été ajoutés:

variation(V,U) Indique une variation de valeur V et d'unité U.

qualificateur(Q) Permet de qualifier le processus (ex: grosse augmentation).

Un trait `contexte` a été ajouté au message pour indiquer qu'un élément de la description d'un objet a déjà été mentionné. Par exemple, le trait

```
contexte === [onto===[c1],
              id===[]],
```

indique que la nature de `c1` est connue (une compagnie, aux figures 6.9 et 6.10) mais que son indentificateur n'a pas encore été mentionné. La gestion du contexte est importante dans un rapport pour éviter de répéter trop souvent la même information.

La grammaire de PréTexte contient maintenant 95 systèmes, parmi lesquels un quart ont été ajoutés ou modifiés pour tenir compte des modifications à la structure du message. Une liste complète des descripteurs de systèmes est présentée en annexe à la section B.4 selon la syntaxe décrite dans [Gagnon, 1993].

```
% Les objets de l'univers
univers([m1,c1,p1]).
% Constantes temporelles
constante(n,incldans(constante(ct0))).
% Les proprietes non temporelles des objets
desc(m1,[onto(moyenne),connu,de(p1)]).                  % m1 = la moyenne de p1
desc(c1,[onto(compagnie),id('Toto inc.'),connu]).       % c1 = la
                                                        % compagnie Toto inc.
desc(p1,[onto(profit),connu,quantite(pluriel),de(c1)]). % p1 = les profits
                                                        % de c1.
% Description des occurrences
occurrence(o1,[processus(augmenter),acteur(m1),   % o1 = m1 augmente
               variation(1000,dollar)]).          % de 1000 dollars.
occurrence(o2,[processus(augmenter),acteur(m1),   % o2 = m1 augmente
               variation(8000-9000,dollar)]).     % de 8000 à 9000 dollars.
occurrence(o3,[processus(augmenter),acteur(m1),   % o3 = m1 augmente beaucoup
               variation(1000,dollar),            % (1000 dollars).
               qualificateur(beaucoup)]).
% Relations entre occurrences
les_relations([avant(n,o1),            % n est avant o1
               avant(o2,n),            % o2 est avant n
               avant(o3,n)]).          % o3 est avant n
```

FIG. 6.9 - *Exemple de représentation conceptuelle pour PréTexte*

```
% Les messages à exprimer
input([
      % La moyenne des profits de la compagnie Toto inc.
      % augmentera de 1000 dollars.
      [message === evenement(o1),
       contexte === [onto===[],
                     id===[]],
        enon===constante(n),
        ref===nil,
        persp===nil,
        loc===nil,
        focus===m1],
      % 2 jours plus tard la moyenne des profits de Toto inc. a
      % augmenté de 8000 dollars à 9000 dollars.
      [message === evenement(o2),
       contexte === [onto===[c1],    % on sait qu'on parle d'une compagnie
                     id===[]],
        enon===constante(n),
        ref===constante(ct1),
        persp===nil,
        loc===apres([ct2,_,_],[ct1,_,_],duree(2,jour)),
        focus===m1],
      % De 1995 à 1996 la moyenne des profits de la compagnie a
      % augmenté beaucoup ( 1000 dollars ).
      [message === evenement(o3),
       contexte === [onto===[],
                     id===[c1]],     % on connait le nom de la compagnie
       enon===constante(n),
       ref===nil,
       persp===nil,
       loc===etendue([ct4,_,_],[ct3,annee,1995],[ct5,annee,1996]),
       focus===m1]
]).
```

FIG. 6.10 - *Exemple de représentation sémantique pour PréTexte*

6.6.3 Interface entre PréTexte et PostGraphe

L'interface entre le planificateur de PostGraphe et le générateur PréTexte se fait à l'intérieur des schémas textuels de PostGraphe. L'utilisation de ces schémas a été décrite à la section 6.4 et leur code apparait en annexe à la section B.3.2. L'interface contenue dans ces schémas a pour but de passer de la structure interne utilisée par PostGraphe au format présenté aux figures 6.9 et 6.10.

Comme nous l'avons vu à la section 6.4, un schéma a 4 paramètres:

Schéma_instance Le type particulier de schéma (ex: evolution2).

Schéma_classe La catégorie de schéma (ex: evolution).

Variables La liste des variables à traiter.

Procédure Une procédure encapsulée retournée par le schéma. Elle sera exécutée plus tard lorsqu'il faudra produire la forme finale du rapport.

À partir de la liste de variables, l'interface génère l'ensemble des faits `desc` de PréTexte. Les 4 transformations suivantes sont appliquées (`N` et `M` sont des entiers quelconques différents obtenus par `gensym/2`):

1. **V** ⇒
   `desc(vN,[onto(V),connu]).`
   (`V` est au singulier)

2. **V** ⇒
   `desc(vN,[onto(VS),connu,quantite(pluriel)]).`
   (`V` est au pluriel,
   `VS` est le singulier de `V`)

3. **reduce(Stat,V)** ⇒
   `desc(vN,[onto(Stat),connu,de(vM)]),`

`desc(vM,[onto(V),connu]).`

(`V` est au singulier)

4. **reduce(Stat,V)** $\Rightarrow$

   `desc(vN,[onto(Stat),connu,de(vM)]),`

   `desc(vM,[onto(VS),connu,quantite(pluriel)]).`

   (`V` est au pluriel,

   `VS` est le singulier de `V`)

Par exemple, à partir de la variable `reduce(moyenne,profits)` (la moyenne des profits), elle va produire les faits

```
desc(v0,[onto(moyenne),connu,de(v1)]).
desc(v1,[onto(profit),connu,quantite(pluriel)]).
```

Le schéma obtient la relation complète à partir des variables à l'aide du prédicat `gentuples/2`. En se servant des données de la relation, il peut construire les occurrences et le message. Ces constructions varient de schéma en schéma. Pour les schémas de type `evolution`, les données sont parcourues afin d'identifier des périodes d'augmentation ou de diminution. Une fois les zones déterminées, les occurrences correspondantes sont générées. Par exemple, à partir de la séquence de profits `[10,20,30,40,20]` en dollars, l'interface va générer les occurrences

```
occurrence(oP,[processus(augmenter),acteur(vN),variation(10-40,dollar)]).
occurrence(oQ,[processus(diminuer),acteur(vN),variation(40-20,dollar)]).
```

où `P` et `Q` sont des entiers différents obtenus à l'aide de `gensym/2`. `vN` est l'identificateur d'un objet *profit* défini préalablement dans un fait `desc`.

4 techniques supplémentaires sont utilisées pour profiter de mécanismes de Pré-Texte. Nous les présentons à travers un exemple simple, de façon à les rendre plus claires. L'exemple décrit l'évolution des profits de 3 compagnies entre 1987 et 1990. Les données brutes de l'exemple sont présentées au tableau 6.5.

	A	B	C
1987	30	160	50
1988	35	165	55
1989	40	140	60
1990	35	155	9

TAB. 6.5 - *Profits (en $) des compagnies A, B et C de 1987 à 1990*

Le traitement de cet exemple utilise le schéma `evolution2` de PostGraphe. La première phase de réalisation de ce schéma consiste à organiser les données pour chaque compagnie sous forme de tranches dans lesquelles le signe de la variation des profits ne change pas. Ensuite, chacune de ces tranches est exprimée par une phrase en évitant de ne pas répéter d'information redondante.

Pour éviter la redondance, nous choisissons 4 procédés linguistiques que PréTexte sait exprimer:

– Variété au niveau de l'adverbial temporel:

  Lors de la première mention d'une compagnie, on utilise l'adverbial temporel d'étendue (De 1987 à 1988) pour bien situer la tranche d'évolution. Ensuite, on utilise des durées (Pendant 1 année). Lors de la dernière mention, on utilise un intervalle fermé à droite (Jusqu'en 1990) pour situer de façon claire la fin des données pour une compagnie.

– Pronominalisation:

  La pronominalisation est utilisée lors de toute mention du groupe nominal sujet sauf la première de chaque compagnie. Lors de la première mention, il faut exprimer le groupe nominal pour préciser de quelle compagnie on parle.

– Ellipse de la catégorie ontologique:

Le mot compagnie n'est exprimé que lors de la première mention du sujet dans le schéma (ellipse de la catégorie ontologique). Lors des mentions subséquentes de celui-ci, on suppose que le lecteur sait qu'il s'agit de compagnies (les profits de B).

– Remplacement d'un intervalle par une différence:

Lors de la première et dernière mention de la variation des profits pour chaque compagnie, on indique les deux bornes de l'intervalle (avant et après). Toutes les autres références utilisent une différence (ont diminué de 25 dollars).

La figure 6.11 montre le résultat obtenu en appliquant ces techniques aux données du tableau 6.5.

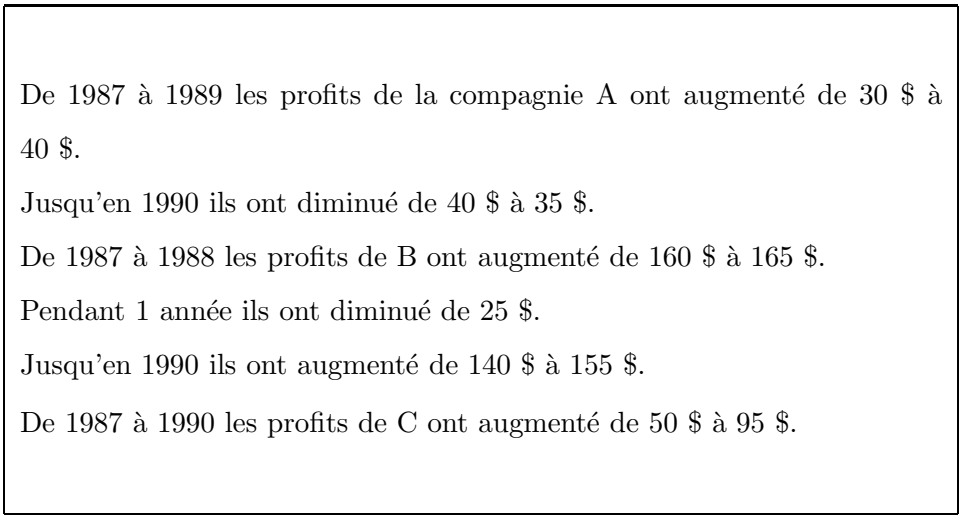


FIG. 6.11 - *Exemple de sortie du schéma textuel* `evolution2`

6.7 Résumé

Dans ce chapitre, nous avons présenté les différents modules Prolog du système PostGraphe: la saisie des données, le traitement des types, la planification, les schémas, le système graphique et l'interface avec le système PréTexte.

La section sur la saisie des données explique le format des données d'entrée du système et les traitements effectués par PostGraphe au moment de la lecture de ces données.

La section sur le traitement des types explique l'implantation en Prolog de notre graphe d'héritage des types et des mécanismes associés.

La section sur la planification décrit les différentes étapes du processus de génération, en insistant sur le planificateur et son contrôle sur les autres modules.

Une section est consacrée à la description du format de nos schémas graphiques et textuels.

Les 3 parties du système graphique de PostGraphe sont décrites en détail: la réalisation de tableaux et graphiques statistiques, la réalisation de primitives graphiques en PostScript et la réalisation de chaînes de caractères en PostScript.

En plus de la description de tous les modules de PostGraphe, nous avons aussi détaillé les modifications apportées au système PréTexte pour l'adapter à PostGraphe.

Chapitre 7

Conclusion

7.1 Résumé de la thèse

Notre thèse s'attaque au problème de la communication personne-machine en étudiant le sous-domaine des rapports statistiques. Nous avons travaillé sur un modèle théorique permettant la génération automatique de ce type de rapports à partir de données brutes annotées. Nous avons ensuite réalisé un prototype nommé PostGraphe pour vérifier le modèle.

Cette thèse a décrit notre modèle théorique en deux phases. Tout d'abord, nous avons décrit les formes d'expression pouvant intervenir dans un rapport statistique (chapitre 3). Parmi ces formes d'expression, on retrouve les figures (tableaux, graphiques statistiques) et le texte (corps du rapport et légendes). Nous présentons les propriétés de chaque type de forme d'expression en insistant sur les règles qui régissent leur utilisation dans un rapport.

Une seconde phase (chapitre 4) s'intéresse aux facteurs qui influencent les décisions prises au cours de la génération d'un rapport statistique. Ces facteurs déterminent quelles annotations devront accompagner les données brutes dans l'entrée d'un système de génération automatique. Notre recherche a montré qu'il était important de

considérer les facteurs suivants afin de pouvoir effectuer des choix informés: l'intention du rédacteur, les types des variables, les relations entre les variables et les valeurs des variables. Notre approche intègre tous ces facteurs dans un même modèle permettant ainsi à un générateur automatique de profiter de toute cette information en même temps.

Notre implantation est elle aussi décrite en deux phases: tout d'abord (chapitre 5), nous présentons une transition entre notre modèle théorique et notre prototype PostGraphe en expliquant les choix et compromis qui ont été nécessaires pour arriver à un système simple et efficace. Cette description est une bonne introduction au fonctionnement algorithmique de PostGraphe. La phase suivante (chapitre 6) décrit plus en détail les "entrailles" du système. Nous découpons cette description par module Prolog plutôt qu'en fonction des algorithmes utilisés.

7.2 Travaux futurs

Le problème présenté dans cette thèse est très complexe. Pour le résoudre, il faut tenir compte d'un grand nombre de facteurs. Ceci explique notre choix d'une approche en largeur au lieu d'une approche en profondeur.

À cause de l'approche en largeur que nous avons choisie, nous avons traîté plusieurs aspects du problème en parallèle, mais nous n'avons pas pu tous les approfondir. La lacune majeure de notre implantation est son traitement très superficiel du texte, dû surtout à un manque de temps de notre part. De plus, d'autres éléments beaucoup mieux traîtés dans PostGraphe n'ont pas reçu la même importance que dans notre modèle théorique. Nous décrivons ici ces faiblesses ainsi que des solutions proposées amenant à des travaux futurs.

7.2.1 Le planificateur

Comme nous l'avons vu aux chapitres 5 et 6, le planificateur de PostGraphe utilise un algorithme de groupement pondéré d'intentions pour choisir ses schémas. Cet algorithme utilise une table, présentée en annexe à la section B.2, pour faire le lien entre les intentions du rédacteur et les schémas. La table indique, entre autres, la valeur du choix de chaque schéma potentiel pour une intention donnée. À l'aide de cette information, le planificateur essaie de construire des groupements d'intentions pouvant être exprimés par un même schéma. Grâce au système de pondération, les résultats peuvent être comparés et les meilleurs groupements sont retenus pour la génération du rapport.

Cette approche présente deux problèmes majeurs: les groupements ne peuvent être que des superpositions d'intentions et la table est difficile à mettre à jour. Comme nous l'avons vu à la section 4.2.3, il existe deux types de groupements d'intentions: la superposition et la composition. La superposition est un groupement additif dans lequel les intentions sont totalement indépendantes. La composition est un groupement multiplicatif dans lequel chaque intention modifie le sens des autres et dans lequel les intentions n'ont pas la même importance relative. Ainsi, par composition de comparaison et évolution, on pourra créer des nouvelles intentions comme "montrer l'évolution de la comparaison entre n variables" ou encore "comparer les évolutions de n variables". Par superposition de ces deux intentions, on ne fait qu'exprimer dans un même texte ou graphique la comparaison des valeurs de n variables et l'évolution des valeurs des mêmes n variables.

La différence entre ces deux types de groupements a une grande influence sur le mécanisme de planification. En effet, l'indépendance des intentions dans une superposition permet l'utilisation d'une table qui associe des intentions individuelles à des schémas individuels. Dans le cas de la composition, il faudrait tenir compte dans la table de tous les groupes possibles ainsi que l'ordre des intentions dans ces groupes.

Ceci rendrait le modèle par table totalement inutilisable. Pourtant, la composition est un mécanisme très naturel qui ne peut pas être écarté dans un système complet. Ceci ouvre donc une avenue de recherche au niveau de la création d'un algorithme permettant de calculer, si possible, la valeur d'un groupement composé à partir d'une table linéaire telle qu'utilisée dans PostGraphe. Un tel algorithme nécessiterait certainement un retour au modèle théorique pour déterminer l'influence exacte des intentions dominantes sur le message global (voir section 4.2.3).

Le second problème de notre approche est la mise à jour de la table de poids. Au départ, les poids initiaux de la table avaient été déterminés à partir d'une version simplifiée du modèle théorique présenté aux chapitres 3 et 4. Après des tests informels auprès d'utilisateurs du système, certains poids ont dû être ajustés pour obtenir des résultats conformes aux attentes de ces utilisateurs. Nous avons constaté que ces modifications ne convergeaient pas toujours. En effet, le système de poids est beaucoup plus flexible que des règles booléennes mais les répercussions d'un changement sont aussi plus difficiles à détecter. Parfois, un changement dans un poids ayant pour but de faire choisir un schéma dans un contexte donné n'affecte pas tout de suite les autres choix. La variation du poids est parfois absorbée dans le calcul pondéré. Après plusieurs modifications, on s'aperçoit qu'un choix devient tout d'un coup erroné mais la cause est difficile à retracer puisqu'elle correspond rarement à un changement qui vient d'être effectué. Ce problème est difficile à résoudre et nécessiterait peut-être un système d'auto-apprentissage capable d'ajuster les poids du système en fonction des réactions de l'utilisateur. Un tel système permettrait aussi au système de s'ajuster aux préférences d'une organisation ou d'un individu dans l'utilisation de certains schémas.

Comme nous l'avons vu au chapitre 5, à cause du manque de temps et de ressources, nous avons choisi de simplifier certains aspects du processus de planification. Un de ces aspects est le choix du médium. En effet, PostGraphe essaie toujours de produire un couple texte/graphique pour chaque groupement d'intentions. Cette simplification est valable, puisqu'il est très rare d'utiliser du texte ou des graphiques

seuls dans les rapports statistiques. Cependant, il arrive parfois qu'un utilisateur ait besoin d'un rapport purement textuel ou d'une série de graphiques isolés. Dans cette situation, PostGraphe ne produit pas de très bons résultats car il n'a pas été doté d'un mécanisme lui permettant d'exprimer chaque idée à l'aide de chaque médium. Puisqu'il suppose la disponibilité du texte et des graphiques, notre système n'essaie pas d'exprimer dans un médium ce que l'autre peut mieux transmettre.

7.2.2 Les schémas

Au niveau de l'implantation des schémas de PostGraphe, on retrouve quelques lacunes moins importantes que celles du planificateur. Elles sont dues, pour la plupart, à un manque de temps qui nous a empéché de compléter certaines bases de connaissances dejà en place. Un seul problème, lié aux préconditions des schémas, est un problème de conception qui pourrait remettre en jeu une partie des mécanismes de génération.

Comme nous l'avons mentionné plus tôt, la génération de texte de PostGraphe est très faible. Ceci est d'autant plus malheureux que beaucoup de travail a été effectué pour modifier le générateur PréTexte (voir section 6.6) et pour réaliser une interface avec notre système. Après ce travail de préparation, seulement deux schémas textuels d'évolution ont été écrits, alors que la partie graphique du système en compte 21. De plus, les deux schémas textuels existants s'intègrent assez mal aux graphiques (pas de références, répétitions par rapport au contenu du graphique). Il serait donc important de travailler sur la mise au point d'autres schémas textuels pour PostGraphe et sur le raffinement éventuel des deux qui existent, de façon à mieux les intégrer aux graphiques.

Malgré ses 21 schémas graphiques, PostGraphe n'offre pas la même diversité que beaucoup de logiciels capables de produire des graphiques statistiques. Cependant,

les choix fournis couvrent les graphiques les plus utilisés et transmettent les intentions du rédacteur de façon claire et précise. Puisque chaque ajout de schéma complique le processus de décision, en particulier la gestion de la pondération des intentions, nous avons décidé de nous retreindre au sous-ensemble de graphiques qui nous semblaient les plus efficaces. De plus, nos graphiques se limitent toujours à 2 dimensions. Là encore, il s'agit d'un choix visant à simplifier énormément le générateur. Parmi les graphiques dits à 3 dimensions, on retrouve deux grandes catégories: les graphiques utilisant les 3 dimensions pour présenter de l'information et les graphiques à 2 dimensions embellis par l'ajout de profondeur. La première catégorie pose un problème majeur dans le processus de décision. En effet, pour présenter des données tridimensionnelles, il faut nécessairement les projeter sur les 2 dimensions du médium de présentation (écran ou papier) et l'efficacité du graphique dépend beaucoup du choix de la perspective pour cette projection. Le choix de la perspective est très difficile à automatiser et n'aboutit pas toujours à une solution. Ainsi, il est souvent préférable d'utiliser un système interactif qui laisse le choix de la projection au lecteur. L'intégration d'un tel système exploratoire avec notre approche plus automatique dépassait le cadre de notre recherche mais pourrait être abordé dans le cadre de recherches complémentaires. La seconde catégorie de graphiques à 3 dimensions est très utilisée dans les chiffiers électroniques et vise à embellir les graphiques en leur ajoutant une composante de profondeur artificielle. Cette approche n'exploite pas le potentiel de la 3^e dimension mais en hérite une partie des problèmes. En effet, l'affichage de ces graphiques nécessite une projection en 2 dimensions et cette opération peut introduire des distortions des longueurs et distances, rendant le graphique moins lisible. C'est pour cette raison que nous avons choisi d'ignorer cette option, optant plutôt pour des graphiques moins impressionnants mais plus lisibles. Par contre, notre intérêt pour les graphiques subjectifs nous empêche de les écarter complètement, puisqu'en combinant le choix d'une perspective à l'organisation interne d'un graphique, on peut produire des graphiques qui font facilement mentir les données. Toute recherche s'intéressant à approfondir l'étude de la subjectivité dans les graphiques statistiques se doit d'examiner l'influence de ces "faux" graphiques à 3 dimensions.

Il faut mentionner que lors du remplissage de la table des poids du planificateur, nous avons choisi de laisser de côté les intentions plus subjectives qui ont été présentées aux sections 3.2.2 et 4.2.2. Ceci est surtout dû au fait que ce sont des intentions qui s'expriment beaucoup plus efficacement avec du texte. Puisque les schémas de texte sont très peu développés, il aurait été difficile de remplir la table. Cependant, tous les mécanismes sont déjà en place et il suffirait de développer des schémas traitant ces intentions pour les inscrire à la table de poids.

Le dernier problème associé aux schémas du système est plus délicat, puisqu'il remet en question un choix d'implantation. En effet, les deux mécanismes de sélection qui agissent au niveau des choix du planificateur sont les entrées de la table de poids, dont l'influence est directe et aussi les préconditions des schémas, qui peuvent faire échouer un choix effectué par le planificateur. Chaque schéma commence par une série de conditions à vérifier avant de pouvoir l'appliquer. Ces conditions sont dures (ou booléennes), par opposition au système de poids qui est plus flexible. Les préconditions peuvent soit échouer, soit réussir; il n'y a pas d'échec ou de réussite partiels. Pour la plupart de ces conditions, c'est exactement ce qu'il faut, puique celles-ci testent des contraintes dures comme le type d'une variable.

Cependant, les conditions qui traitent des valeurs des données ne s'intègrent pas très bien à ce modèle. En effet, la contrainte qui pose le plus de problèmes dans PostGraphe est celle qui détermine le nombre de valeurs d'une variable pour décider qu'un schéma graphique utilisant des colonnes n'est pas approprié. Ainsi, si ce nombre dépasse 10, tout graphique en colonnes est directement rejeté au profit des schémas utilisant des courbes. Cette situation est problématique dans le cas où le nombre est très proche de 10, puisque 11 colonnes devraient être un peu moins valables que 10 au lieu d'être simplement rejetées. Pour résoudre ce problème, il faudrait ajouter une phase au processus de planification dans laquelle des contraintes pondérées sont évaluées et modifient les résultats de la table de poids actuelle. Les contraintes dures

resteraient au niveau des préconditions des schémas. Cette idée semble assez simple mais elle demande une étude sur l'importance relative de ces contraintes pondérées par rapport aux poids des intentions. Si cette importance est surestimée, on retombe dans la situation actuelle où l'influence des contraintes est trop forte. Si elle est sous-estimée, la situation empire, puisqu'elles n'auront pas assez d'influence sur le processus de génération.

7.3 Contribution à la science

Malgré les avenues de recherche non explorées, notre approche contribue à la science en apportant un certain nombre de nouveautés à la génération automatique de rapports statistiques.

Tout d'abord, le modèle proposé, contrairement à ses prédécesseurs, utilise les intentions ou buts du rédacteur pour planifier le contenu du rapport. Cela permet à PostGraphe de générer des rapports qui sont mieux adaptés aux attentes de l'utilisateur.

De plus, nous avons développé un système de types avec héritage multiple et traitement des unités beaucoup plus complet que le typage des variables à 3 catégories (nominale, ordinale, quantitative) tel que proposé par MacKinlay [Mackinlay, 1986a; Mackinlay, 1986b]. Ceci permet des choix plus raffinés tant au niveau du texte que des graphiques en donnant au générateur des connaissances précises et bien organisées sur les données à présenter.

Pour raffiner encore plus les choix de notre générateur, nous avons tenu compte des relations fonctionnelles entre les variables. Les relations permettent une meilleure structuration du texte et des graphiques puisqu'elles indiquent quelles variables sont fonction de quelles autres. Au niveau des graphiques, elles permettent de décider où placer les variables. Au niveau du texte, elles permettent de différencier entre des

expressions comme *les profits de la compagnie* et *la compagnie des profits*.

Nous tenons aussi compte du nombre et des valeurs des données à présenter pour choisir des formes d'expression plus appropriées. En effet, nos règles de planification se servent de contraintes sur les données pour rejeter certains choix dans des conditions limite. Par exemple, lorsque le nombre de valeurs dépasse la dizaine dans un graphique en colonnes, il est préférable d'utiliser une courbe.

Aucun de ces 4 facteurs n'a été exploré en détail par nos prédécesseurs. De plus, bien que certains éléments aient été utilisés sous une forme simplifiée dans des systèmes de génération, aucun modèle intégrant les 4 n'a été proposé jusqu'ici. Nous proposons un modèle qui exploite ces 4 sources d'information et qui s'applique aussi bien au texte qu'aux graphiques tout en utilisant une entrée semblable à celle que l'on retrouve dans les tableurs.

Annexe A

Exemple de sortie du système

Cette annexe contient la sortie complète de PostGraphe correspondant à l'entrée présentée à la page 114. La section 5.5 présente cette sortie sans les annotations.

DÉBUT DU RAPPORT GÉNÉRÉ

Nouvelle section (3 intentions à traiter).

nouvelles intentions: présentation de année. présentation de compagnie. présentation de profits.

année / compagnie	1987 profits	1988 profits	1989 profits	1990 profits
A	30	35	40	35
B	160	165	140	155
C	50	55	60	95

FIG. A.1 - *[Schéma:* tableau1*]. présentation de année (100). présentation de compagnie (100). présentation de profits (100).*

Nouvelle section (2 intentions à traiter).

nouvelles intentions: comparaison de profits entre compagnie. évolution de profits par rapport à année.

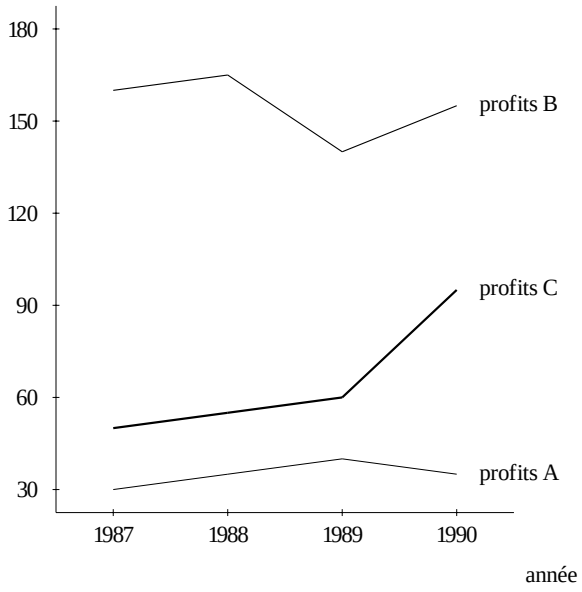


FIG. A.2 - *[Schéma:* courbe3*]. comparaison de profits entre compagnie (69). évolution de profits par rapport à année (95).*

[Schéma: *evolution2*]**. évolution de profits par rapport à année (100).**

De 1987 à 1989 les profits de la compagnie A ont augmenté de 30 $ à 40 $. Jusqu'en 1990 ils ont diminué de 40 $ à 35 $.

De 1987 à 1988 les profits de B ont augmenté de 160 $ à 165 $. Pendant 1 année ils ont diminué de 25 $. Jusqu'en 1990 ils ont augmenté de 140 $ à 155 $.

De 1987 à 1990 les profits de C ont augmenté de 50 $ à 95 $.

FIN DU RAPPORT GÉNÉRÉ

Annexe B

Extraits de code

B.1 Règles du système d'unités

Cette section contient le code Prolog correspondant aux règles du système d'unités de PostGraphe. L'implantation du module associé à ces règles est décrite à la section 6.2. La section 4.1.2 présente les aspects théoriques du système de types.

```
% Propriétés d'organisation
heritage_propriete(symbolique,[enumeration(_)]).
heritage_propriete(ordonnee,[]).
heritage_propriete(nominale,[symbolique]).
heritage_propriete(ordinale,[symbolique,ordonnee]).
heritage_propriete(quantitative,[ordonnee,nombre]).
heritage_propriete(fractionnaire,[quantitative]).

% Propriétés sur le domaine
heritage_propriete(enumeration(_),[]).
heritage_propriete(plus_grand(_),[]).
heritage_propriete(plus_petit(_),[]).
heritage_propriete(intervalle(X,Y),[plus_grand(X),plus_petit(Y)]).

% Propriétés temporelles
heritage_propriete(temporelle,[ordonnee]).
heritage_propriete(mois(annee),
                   [etiquette,
                    enumeration(['janvier','février','mars','avril',
                                 'mai','juin',
                                 'juillet','août','septembre','octobre',
```

```
                                'novembre','décembre']),
                    temporelle]).
heritage_propriete(mois,[mois(annee),enumeration(_)]).
heritage_propriete(annee,[entier,symbolique,temporelle]).

% Propriétés de format
heritage_propriete(etiquette,[nominale]).
heritage_propriete(nombre,[intervalle(_,_)]).
heritage_propriete(reel,[nombre]).
heritage_propriete(entier,[reel]).

% Propriétés sur les mesures
heritage_propriete(longueur,[]).
heritage_propriete(position,[]).
heritage_propriete(spatiale,[]).
heritage_propriete(position_temporelle,[temporelle,position]).
heritage_propriete(date,[position_temporelle]).
heritage_propriete(duree,[temporelle,longueur]).
heritage_propriete(distance,[spatiale,longueur]).
heritage_propriete(position_spatiale,[spatiale,position]).

% Propriétés diverses
heritage_propriete(pluriel(_),[]).

% Propriétés dérivées (objets réels)
heritage_propriete(territoire,[etiquette]).
heritage_propriete(pays,[territoire]).
heritage_propriete(province,[territoire]).
heritage_propriete(pourcentage,[reel,quantitative]).
heritage_propriete(dollar,[reel,quantitative]).

% Inférence de propriétés
inference_propriete([plus_grand(X),plus_petit(Y)],intervalle(X,Y)).
inference_propriete([symbolique,ordonnee],ordinale).

% Unités
unite_propriete(dollar,'$').
unite_propriete(pourcentage,'%').
unite_propriete(etiquette,'').
```

B.2 La table d'inférence du planificateur

Le code Prolog présenté dans cette section représente la table d'inférence utilisée par PostGraphe pour déterminer quels schémas correspondent à quelles intentions du rédacteur. Le processus de planification, pendant lequel intervient cette table, est présenté aux sections 5.4.4 et 6.3.

```
c_fig(comparaison([X], [Y]), tableau2, tableau, Z, 30, [X / Z,Y / Z]).
c_fig(comparaison([Y], [X]), courbe2, courbe, [X|Y], 77, [Y / Y]).
c_fig(comparaison([T], [X]), tableau1, tableau, [X,Y|Z], 30, [T / Z]).
c_fig(comparaison([T], [Y]), tableau1, tableau, [X,Y|Z], 30, [T / Z]).
c_fig(comparaison([Z,T], [X]), courbe2, courbe, [X|Y], 75, [Z / Y,T / Y]).
c_fig(comparaison([Z], [X]), colonnes3, colonnes, [X|Y], 80, [Z / Y]).
c_fig(comparaison([Z,T], [X]), colonnes3, colonnes, [X|Y], 80, [Z / Y,T / Y]).
c_fig(comparaison([X], [T]), points6, points, [X,Y|Z], 55, [T / Z]).
c_fig(comparaison([X], [T]), points7, points, [X,Y,Z,T], 55, []).
c_fig(comparaison([X], [Y]), points1, points, [X,Y], 55, []).
c_fig(comparaison([X], [Z]), points2, points, [X,Y,Z], 55, []).
c_fig(comparaison([X], [Z]), points3, points, [X,Y,Z], 55, []).
c_fig(comparaison([X], [Z]), points4, points, [X,Y,Z], 55, []).
c_fig(comparaison([X], [Z]), points5, points, [X,Y,Z], 55, []).
c_fig(comparaison([X], [Z]), points7, points, [X,Y,Z,T], 55, []).
c_fig(comparaison([Y], [T]), points6, points, [X,Y|Z], 55, [T / Z]).
c_fig(comparaison([Y], [T]), points7, points, [X,Y,Z,T], 55, []).
c_fig(comparaison([Y], [X]), barres1, barres, [X,Y], 100, []).
c_fig(comparaison([Y], [X]), colonnes1, colonnes, [X,Y], 70, []).
c_fig(comparaison([Y], [X]), courbe1, courbe, [X,Y], 69, []).
c_fig(comparaison([Y], [X]), points1, points, [X,Y], 55, []).
c_fig(comparaison([Y], [X]), tarte1, tarte, [X,Y], 75, []).
c_fig(comparaison([Y], [X]), tarte2, tarte, [X,Y], 50, []).
c_fig(comparaison([Y], [Z]), points2, points, [X,Y,Z], 55, []).
c_fig(comparaison([Y], [Z]), points3, points, [X,Y,Z], 55, []).
c_fig(comparaison([Y], [Z]), points4, points, [X,Y,Z], 55, []).
c_fig(comparaison([Y], [Z]), points5, points, [X,Y,Z], 55, []).
c_fig(comparaison([Y], [Z]), points7, points, [X,Y,Z,T], 55, []).
c_fig(comparaison([Z], [X]), colonnes2, colonnes, [X,Y,Z], 80, []).
c_fig(comparaison([Z], [X]), courbe3, courbe, [X,Y,Z], 79, []).
c_fig(comparaison([Z], [Y]), colonnes2, colonnes, [X,Y,Z], 70, []).
c_fig(comparaison([Z], [Y]), courbe3, courbe, [X,Y,Z], 69, []).
c_fig(correlation(X, Y), points1, points, [X,Y], 100, []).
c_fig(correlation(X, Y), points2, points, [X,Y,Z], 100, []).
c_fig(correlation(X, Y), points3, points, [X,Y,Z], 100, []).
c_fig(correlation(X, Y), points4, points, [X,Y,Z], 100, []).
c_fig(correlation(X, Y), points5, points, [X,Y,Z], 100, []).
```

```
c_fig(correlation(X, Y), points6, points, [X,Y|Z], 100, []).
c_fig(correlation(X, Y), points7, points, [X,Y,Z,T], 100, []).
c_fig(proportion(Y), tarte3, tarte, [X|Y], 100, []).
c_fig(decomposition(Y), tarte1, tarte, [X,Y], 100, []).
c_fig(decomposition(Y), tarte2, tarte, [X,Y], 100, []).
c_fig(evolution(Z, X), courbe2, courbe, [X|Y], 100, [Z / Y]).
c_fig(evolution(Z, X), colonnes3, colonnes, [X|Y], 80, [Z / Y]).
c_fig(evolution(Y, X), colonnes1, colonnes, [X,Y], 94, []).
c_fig(evolution(Y, X), colonnes5, colonnes, [X,_,Y], 90, []).
c_fig(evolution(Y, X), courbe1, courbe, [X,Y], 100, []).
c_fig(evolution(Z, X), colonnes2, colonnes, [X,Y,Z], 85, []).
c_fig(evolution(Z, Y), colonnes2, colonnes, [X,Y,Z], 60, []).
c_fig(evolution(Z, X), courbe3, courbe, [X,_,Z], 95, []).
c_fig(presentation(X), tableau2, tableau, L, 95, [X / L]).
c_fig(presentation(Z), courbe2, courbe, [X|Y], 90, [Z / Y]).
c_fig(presentation(T), points6, points, [X,Y|Z], 50, [T / Z]).
c_fig(presentation(T), tableau1, tableau, Z, 100, [T / Z]).
c_fig(presentation(T), points7, points, [X,Y,Z,T], 50, []).
c_fig(presentation(X), barres1, barres, [X,Y], 100, []).
c_fig(presentation(X), colonnes1, colonnes, [X,Y], 100, []).
c_fig(presentation(X), colonnes2, colonnes, [X,Y,Z], 100, []).
c_fig(presentation(X), colonnes3, colonnes, [X|Y], 100, []).
c_fig(presentation(X), courbe1, courbe, [X,Y], 100, []).
c_fig(presentation(X), courbe2, courbe, [X|Y], 100, []).
c_fig(presentation(X), courbe3, courbe, [X,Y,Z], 100, []).
c_fig(presentation(X), points1, points, [X,Y], 80, []).
c_fig(presentation(X), points2, points, [X,Y,Z], 80, []).
c_fig(presentation(X), points3, points, [X,Y,Z], 80, []).
c_fig(presentation(X), points4, points, [X,Y,Z], 80, []).
c_fig(presentation(X), points5, points, [X,Y,Z], 80, []).
c_fig(presentation(X), points6, points, [X,Y|Z], 80, []).
c_fig(presentation(X), points7, points, [X,Y,Z,T], 80, []).
c_fig(presentation(X), tarte1, tarte, [X,Y], 100, []).
c_fig(presentation(X), tarte2, tarte, [X,Y], 100, []).
c_fig(presentation(Y), barres1, barres, [X,Y], 80, []).
c_fig(presentation(Y), colonnes1, colonnes, [X,Y], 80, []).
c_fig(presentation(Y), colonnes2, colonnes, [X,Y,Z], 90, []).
c_fig(presentation(Y), colonnes3, colonnes, [X|Y], 90, []).
c_fig(presentation(Y), courbe1, courbe, [X,Y], 90, []).
c_fig(presentation(Y), courbe3, courbe, [X,Y,Z], 75, []).
c_fig(presentation(Y), points1, points, [X,Y], 80, []).
c_fig(presentation(Y), points2, points, [X,Y,Z], 80, []).
c_fig(presentation(Y), points3, points, [X,Y,Z], 80, []).
c_fig(presentation(Y), points4, points, [X,Y,Z], 80, []).
c_fig(presentation(Y), points5, points, [X,Y,Z], 80, []).
c_fig(presentation(Y), points6, points, [X,Y|Z], 80, []).
c_fig(presentation(Y), points7, points, [X,Y,Z,T], 80, []).
```

```
c_fig(presentation(Y), tarte1, tarte, [X,Y], 50, []).
c_fig(presentation(Y), tarte2, tarte, [X,Y], 50, []).
c_fig(presentation(Z), colonnes2, colonnes, [X,Y,Z], 80, []).
c_fig(presentation(Z), courbe3, courbe, [X,Y,Z], 85, []).
c_fig(presentation(Z), points2, points, [X,Y,Z], 100, []).
c_fig(presentation(Z), points3, points, [X,Y,Z], 60, []).
c_fig(presentation(Z), points4, points, [X,Y,Z], 50, []).
c_fig(presentation(Z), points5, points, [X,Y,Z], 70, []).
c_fig(presentation(Z), points7, points, [X,Y,Z,T], 70, []).
c_fig(repartition(X, Y), colonnes4, colonnes, [X|Y], 100, []).
c_fig(repartition(X, Y), tarte3, tarte, [X|Y], 90, []).

c_txt(evolution(V,T),evolution1,evolution,[T,V],99,[]).
c_txt(evolution(V,T),evolution2,evolution,[T,V,_],100,[]).
```

B.3 Les schémas

Les règles Prolog listées ici correspondent aux schémas de PostGraphe. Elles sont décrites aux sections 6.4 (généralités et graphiques) et 6.6.3 (texte). PostGraphe les utilise pour vérifier les conditions d'utilisation des schémas et aussi pour réaliser ceux-ci.

B.3.1 Graphiques

```
%%
%  TARTES
%%

figure(tarte1,tarte,[X,Y],P) :-
    % si X est nominale, on trie ses valeurs...
    type(X,nominale), type(Y,quantitative),
    cle([X],L), member(Y,L),
    gentuples([X,Y],T), sort((Y1 < Y2),[_,Y1],[_,Y2],T,T2),
    etiquette(X,XE), etiquette(Y,YE),
    pict(tarte(XE,YE,T2,[]),P).

figure(tarte2,tarte,[X,Y],Procedure) :-
    % si X est ordonnee, on ne trie pas ses valeurs,
    % mais on met en relief le max...
    type(X,ordonnee), type(Y,quantitative),
    cle([X],L),member(Y,L),
    gentuples([X,Y],T), positions(T,PO),
    map2(true,[TuX,TuY],Po,[TuY,Po],T,PO,TPO), TPO=[H|TT],
    foldl((X1 > X2 -> R=[X1,P1] ; R=[X2,P2]),[X1,P1],[X2,P2],R,H,TT,[_,PMAX]),
    etiquette(X,XE), etiquette(Y,YE),
    pict(tarte(XE,YE,T,[PMAX]),Procedure).

figure(tarte3,tarte,[Z|ZR],Procedure) :-
    % repartition...
    type(Z,quantitative),
    cle(ZR,L),member(Z,L),
    gentuples([Z|ZR],TU),
    typechelle(Z,TU,0,intervalle(ZMin,ZMax)),
    mapnth(0,TU,ZL), mk_repart(ZMin,ZMax,ZL,EX,EY,LL0),
    mapnth(1,LL0,LL1), somme(LL1,S1),
    map((PCT is X1 * 100 / S1),[Label,X1],[Label,PCT],LL0,LL2),
    sort((M1 > M2),[_,M1],[_,M2],LL2,LL),
    etiquette(Z,ZE),
```

```
    pict(tarte(ZE,'% par classe',LL,[]),Procedure).

%%
%  BARRES
%%

figure(barres1,barres,[X,Y],Procedure) :-
    % barres standard (inversion de X et Y)...
    type(X,symbolique),
    cle([X],L),member(Y,L),
    gentuples([Y,X],T),
    typechelle(Y,T,0,EY), typechelle(X,T,1,enumeration(EX0)),
    (type(X,nominale) ->
     % si X est nominale, alors on trie...
     sort((urank(X1,Y,R1), urank(X2,Y,R2), R1 @< R2),[X1,_],[X2,_],T,T2),
     sort((pmember([_,X1],T2,P1), pmember([_,X2],T2,P2), P1 > P2),X1,X2,EX0,EX);
     EX=EX0),
    etiquette(X,XE), etiquette(Y,YE),
    pict(barres(YE,XE,EY,enumeration(EX),T),Procedure).

%%
%  COLONNES
%%


figure(colonnes1,colonnes,[X,Y],Procedure) :-
    % colonnes standard...
    type(X,symbolique),
    cle([X],L),member(Y,L),
    gentuples([X],TX), length(TX,LTX), LTX =< 10, gentuples([X,Y],T),
    typechelle(X,T,0,EX), typechelle(Y,T,1,EY),
    etiquette(X,XE), etiquette(Y,YE),
    pict(colonnes(XE,YE,EX,EY,T),Procedure).

figure(colonnes2,colonnes,[X,Y,Z],Procedure) :-
    % multi-colonnes avec dependance 2D...
    type(X,symbolique), type(Y,symbolique),
    relcle(_,[X,Y],L),member(Z,L),
    gentuples([X],TX), length(TX,LTX), gentuples([Y],TY), length(TY,LTY),
    LTX =< 10, LTY =< 6, LTX*LTY =< 30,
    gentuples([X,Y,Z],T), gentuples([reduce(liste,Y),X,reduce(liste,Z)],T1),
    map(true,[_,X1,Z1s],[X1|Z1s],T1,T2),
    typechelle(X,T,0,EX), typechelle(Y,T,1,enumeration(EYs)),
    typechelle(Z,T,2,EZ),
    etiquette(X,XE), etiquette(Z,ZE),
    map(concatom([ZE,' ',EY],EY2),EY,EY2,EYs,EY2s),
    pict(colonnes(XE,EY2s,EX,EZ,T2,juxtaposition),Procedure).
```

```
figure(colonnes3,colonnes,[X|Ys],Procedure) :-
    % multi-colonnes (plusieurs Y)...
    type(X,symbolique),
    cle([X],L),map(member(Y,L),Y,_,Ys,_),
    gentuples([X],TX), length(TX,LTX), length(Ys,LTY),
    LTX =< 10, LTY =< 6, LTX*LTY =< 25, length(Ys,LYs),LYs>1,
    gentuples([X|Ys],T),
    typechelle(X,T,0,EX), typechelle(Ys,T,1,EY),
    etiquette(X,XE), map(etiquette(Y,YE),Y,YE,Ys,YEs),
    pict(colonnes(XE,YEs,EX,EY,T,juxtaposition),Procedure).

figure(colonnes4,colonnes,[Z|ZR],Procedure) :-
    % repartition...
    type(Z,quantitative),
    cle(ZR,L),member(Z,L),
    gentuples([Z|ZR],TU), typechelle(Z,TU,0,intervalle(ZMin,ZMax)),
    mapnth(0,TU,ZL), mk_repart(ZMin,ZMax,ZL,EX,EY,LL),
    etiquette(Z,ZE),
    pict(colonnes(ZE,'# par classe',EX,EY,LL),Procedure).

figure(colonnes5,colonnes,[X,Y,Z],Procedure) :-
    % colonnes séparées avec dependance 2D...
    type(X,symbolique), type(Y,symbolique),
    relcle(_,[X,Y],L),member(Z,L),
    gentuples([X],TX), length(TX,LTX),
    gentuples([Y],TY), length(TY,LTY), conclist(TY,YL),
    LTX =< 10, LTY =< 6, LTX*LTY =< 30,
    map((
        gentuples([X,select(Y,(_Y=YEl),_Y),Z],T1),
        map(true,[X1,_,Z1],[X1,Z1],T1,T),
        typechelle(X,T,0,EX), typechelle(Z,T,1,EZ),
        etiquette(X,XE), etiquette(Z,ZE),
        concatom([ZE,' ',YEl],ZE2)
        ),YEl,colonnes(XE,ZE2,EX,EZ,T),YL,Procedures),
    pict(Procedures,Procedure).

%%
%  POINTS
%%

figure(points1,points,[X,Y],Procedure) :-
    % points standard...
    gentuples([X,Y],T),
    typechelle(X,T,0,EX), typechelle(Y,T,1,EY),
    etiquette(X,XE), etiquette(Y,YE),
    pict(points(XE,YE,EX,EY,T),Procedure).
```

```
figure(points2,points,[X,Y,Z],Procedure) :-
    % points etiquetes...
    type(Z,etiquette),
    gentuples([X,Y,Z],T),
    typechelle(X,T,0,EX), typechelle(Y,T,1,EY), typechelle(Z,T,2,EZ),
    etiquette(X,XE),
    etiquette(Y,YE), etiquette(Z,ZE),
    pict(points(XE,YE,ZE,EX,EY,EZ,T,etiquette),Procedure).

figure(points3,points,[X,Y,Z],Procedure) :-
    % points a forme variable...
    type(Z,nominale),
    gentuples([X,Y,Z],T),
    typechelle(X,T,0,EX), typechelle(Y,T,1,EY), typechelle(Z,T,2,EZ),
    etiquette(X,XE), etiquette(Y,YE), etiquette(Z,ZE),
    pict(points(XE,YE,ZE,EX,EY,EZ,T,forme),Procedure).

figure(points4,points,[X,Y,Z],Procedure) :-
    % points a intensite variable...
    type(Z,symbolique),
    gentuples([X,Y,Z],T),
    typechelle(X,T,0,EX), typechelle(Y,T,1,EY), typechelle(Z,T,2,EZ),
    etiquette(X,XE), etiquette(Y,YE), etiquette(Z,ZE),
    pict(points(XE,YE,ZE,EX,EY,EZ,T,gris),Procedure).

figure(points5,points,[X,Y,Z],Procedure) :-
    % points a surface variable...
    type(Z,quantitative),
    gentuples([X,Y,Z],T),
    typechelle(X,T,0,EX), typechelle(Y,T,1,EY), typechelle(Z,T,2,EZ),
    etiquette(X,XE), etiquette(Y,YE), etiquette(Z,ZE),
    pict(points(XE,YE,ZE,EX,EY,EZ,T,surface),Procedure).

figure(points6,points,[X,Y|Zs],Procedure) :-
    % multi-points :-)
    map(type(Z,quantitative),Z,_,Zs,_),
    length(Zs,LZs), LZs > 1, gentuples([X,Y|Zs],T),
    typechelle(X,T,0,EX), typechelle(Y,T,1,EY), typechelle(Zs,T,2,EZ),
    etiquette(X,XE), etiquette(Y,YE),
    map(etiquette(Z,ZE),Z,ZE,Zs,ZEs),
    pict(points(XE,YE,ZEs,EX,EY,EZ,T,surface),Procedure).

figure(points7,points,[X,Y,Z1,Z2],Procedure) :-
    % points a surface et intensite variable...
    type(Z1,quantitative), type(Z2,symbolique),
    gentuples([X,Y,Z1,Z2],T),
```

```
    typechelle(X,T,0,EX), typechelle(Y,T,1,EY),
    typechelle(Z1,T,2,EZ1), typechelle(Z2,T,3,EZ2),
    etiquette(X,XE), etiquette(Y,YE), etiquette(Z1,ZE1), etiquette(Z2,ZE2),
    pict(points(XE,YE,ZE1,ZE2,EX,EY,EZ1,EZ2,T),Procedure).

%%
%  COURBES
%%

figure(courbe1,courbe,[X,Y],Procedure) :-
    % courbe standard...
    type(X,ordonnee),
    (type(X,symbolique) ->
     gentuples([X],TX), length(TX,LTX), LTX > 10 ; true),
    type(Y,quantitative),
    cle([X],L),member(Y,L),
    gentuples([X,Y],T),
    typechelle(X,T,0,EX), typechelle(Y,T,1,EY),
    etiquette(X,XE), etiquette(Y,YE),
    pict(courbe(XE,YE,EX,EY,T),Procedure).

figure(courbe2,courbe,[X|Ys],Procedure) :-
    % courbes superposees...
    type(X,ordonnee),
    (type(X,symbolique) ->
     gentuples([X],TX), length(TX,LTX), LTX > 10 ; true),
    map(type(Y,quantitative),Y,_,Ys,_),
    cle([X],L),map(member(Y,L),Y,_,Ys,_),
    gentuples([X|Ys],T),
    typechelle(X,T,0,EX), typechelle(Ys,T,1,EY),
    etiquette(X,XE), map(etiquette(Y,YE),Y,YE,Ys,YEs),
    pict(courbe(XE,YEs,EX,EY,T),Procedure).

figure(courbe3,courbe,[X,Y,Z],Procedure) :-
    % courbes superposees avec dependance 2D...
    type(X,ordonnee),
    type(X,symbolique),
    type(Y,symbolique),
    relcle(_,[X,Y],L),member(Z,L),
    gentuples([X],TX), length(TX,LTX),
    gentuples([Y],TY), length(TY,LTY), LTY < 4,
    gentuples([X,Y,Z],T), gentuples([reduce(liste,Y),X,reduce(liste,Z)],T1),
    map(true,[_,X1,Z1s],[X1|Z1s],T1,T2),
    typechelle(X,T,0,EX), typechelle(Y,T,1,enumeration(EYs)),
    typechelle(Z,T,2,EZ),
    etiquette(X,XE), etiquette(Z,ZE),
    map(concatom([ZE,' ',EY],EY2),EY,EY2,EYs,EY2s),
```

```
    pict(courbe(XE,EY2s,EX,EZ,T2),Procedure).

%%
%  TABLEAUX
%%

figure(tableau1,tableau,[X,Y|Zs],Procedure) :-
    % tableau 2D...
    type(X,symbolique), type(Y,symbolique),
    (cle([X,Y],L);cle([Y,X],L)),!,
    map(member(Z,L),Z,_,Zs,_),
    remplissage([X,Y],Zs,_,_,R), R > 0.25,
    gentuples([X,Y|Zs],T),
    typechelle(X,T,0,enumeration(EX)), typechelle(Y,T,1,enumeration(EY)),
    etiquette(X,XE), etiquette(Y,YE),
    map(etiquette(Z,ZE),Z,ZE,Zs,ZEs),
    pict(table(XE,YE,ZEs,EX,EY,T),Procedure).

figure(tableau2,tableau,Zs,Procedure) :-
    % tableau de tuples (1D)...
    gentuples(Zs,T),
    length(T,LT), LT =< 10,
    map(etiquette(Z,ZE),Z,ZE,Zs,ZEs),
    pict(table(ZEs,T),Procedure).
```

B.3.2 Texte

```
%%%
% Textes d'évolution
%%%

% Schémas (Préconditions, Transformation, Réalisation)

texte(evolution1,evolution,[T,V],Proc) :-
        % évolution simple: 1 variable (V) + le temps (T)
        type(V,quantitative), type(T,temporelle),
        gentuples([T,V],D),
        prepare_texte(evolution(T,V,D),Res),
        texte_procedure(evolution1(debut,T,V,Res),Proc).

texte(evolution2,evolution,[T,V,V2],Proc) :-
        % évolutions multiples: 1 évolution simple (V,T)
        % pour chaque valeur de V2
        type(V,quantitative), type(T,temporelle),
        cle([T,V2],L),member(V,L),
        gentuples([T,V,V2],D),
        prepare_texte(evolution(T,V,V2,D),Res),
        texte_procedure(evolution2([],[],T,V,V2,Res),Proc).

% Implantation de la partie réalisation des schémas
% (interface avec PréTexte)

texte_realise(evolution1(_,_,_,[])).
texte_realise(evolution1(Debut,T,V,[[T1,T2,Val1,Val2]|Reste])) :-
        instance_de(T,TT),
        unite_de(V,VV),
        u_diff(T2,T1,T,DeltaT), u_diff(Val2,Val1,V,DeltaV),
        gensym(v,G0), gensym(v,G1), gensym(v,G2),
        gensym(o,O1), gensym(o,I1),
        gensym(ct,CT0), gensym(ct,CT1), gensym(ct,CT2), gensym(ct,CT3),
        %
        % les objets
        %
        (V = reduce(Stat,Vari) ->
            pretexte(desc(G0,[onto(Stat),connu,de(G1)])),
            Acteur=G0;
            Vari=V,
            Acteur=G1),
        (unite_pluriel(Vari,VSing) ->
            pretexte(desc(G1,[onto(VSing),connu,
                              quantite(pluriel)]));
            pretexte(desc(G1,[onto(Vari),connu]))),
```

```
        %
        % les actions
        %
        (DeltaV > 0 -> Pro = augmenter ; Pro = diminuer),
        (Debut = debut ->
           Debut2 = milieu,
           Loc=etendue([CT1,_,_],[CT2,TT,T1],[CT3,TT,T2]),
           Variation = Val1-Val2;
           (Reste = [] ->
              Loc=termpar([CT1,_,_],[CT3,TT,T2]),
              Variation = Val1-Val2;
              Debut2 = Debut,
              Loc=duree(DeltaT,TT),
              Variation is abs(DeltaV))),
        pretexte(occurrence(O1,[processus(Pro),acteur(Acteur),
                                variation(Variation,VV)])),
        %
        % les relations
        %
        pretexte_add(les_relations,avant(O1,n)),
        (Debut = debut ->
            Cont = [];
            Cont = [onto===[Acteur]]),
        pretexte(constante(n,incldans(constante(CT0)))),
        %
        % le message
        %
        pretexte_add(input,[message===evenement(O1),
                            contexte===Cont,
                            enon===constante(n),
                            ref===nil,
                            persp===nil,
                            loc===Loc,
                            focus===Acteur]),
        texte_realise(evolution1(Debut2,T,V,Reste)).

texte_realise(evolution2(_,_,_,_,_,[])).
texte_realise(evolution2(Contexte,IdLast,T,V,V2,
              [[T1,T2,Val1,Val2,Id]|Reste])) :-
        instance_de(T,TT),
        unite_de(V,VV),
        u_diff(T2,T1,T,DeltaT), u_diff(Val2,Val1,V,DeltaV),
        gensym(v,G0), gensym(v,G1), gensym(v,G2),
        gensym(o,O1), gensym(o,I1),
        gensym(ct,CT0), gensym(ct,CT1), gensym(ct,CT2), gensym(ct,CT3),
        %
        % les objets
```

```
%
(V = reduce(Stat,Vari) ->
    pretexte(desc(G0,[onto(Stat),connu,de(G1)])),
    Acteur=G0;
    Vari=V,
    Acteur=G1),
(unite_pluriel(Vari,VSing) ->
    pretexte(desc(G1,[onto(VSing),connu,
                      quantite(pluriel),de(G2)]));
    pretexte(desc(G1,[onto(Vari),connu,de(G2)]))),
pretexte(desc(G2,[onto(V2),connu,id(Id)])),
%
% les actions
%
(DeltaV > 0 -> Pro = augmenter ; Pro = diminuer),
(not(IdLast = Id) ->
   Loc=etendue([CT1,_,_],[CT2,TT,T1],[CT3,TT,T2]),
   IDCONT=[],
   Variation = Val1-Val2;
   IDCONT=[id===[G2]],
   ((Reste = [];Reste=[[_,_,_,_,ID]|_],not(ID=Id)) ->
      Loc=termpar([CT1,_,_],[CT3,TT,T2]),
      Variation = Val1-Val2;
      Loc=duree(DeltaT,TT),
      Variation is abs(DeltaV))),
(member(onto,Contexte) -> Cont0=[onto===[G2]|IDCONT];Cont0=IDCONT),
((member(onto===[G2],Cont0),member(id===[G2],Cont0)) ->
    Cont=[onto===[Acteur]];Cont=Cont0),
(Contexte = [] -> Contexte2=[onto];Contexte2=Contexte),
pretexte(occurrence(O1,[processus(Pro),acteur(Acteur),
                        variation(Variation,VV)])),
%
% les relations
%
pretexte_add(les_relations,avant(O1,n)),
pretexte(constante(n,incldans(constante(CT0)))),
%
% le message
%
pretexte_add(input,[message===evenement(O1),
                    contexte===Cont,
                    enon===constante(n),
                    ref===nil,
                    persp===nil,
                    loc===Loc,
                    focus===Acteur]),
texte_realise(evolution2(Contexte2,Id,T,V,V2,Reste)).
```

```
% Prétraitement (regroupement de variations de même signe)

prepare_texte(evolution(T,V,D),D2) :-
    regroupe_evolutions(D,[],D2).
prepare_texte(evolution(T,V,V2,D),D2) :-
    sort((VA = VB -> urank(TA,T,R1),urank(TB,T,R2),R1@>R2 ; VA @> VB),
         [TA,_,VA],[TB,_,VB],D,D3),
    regroupe_evolutions2(D3,[],D2).

regroupe_evolutions2([],OutR,Out) :-
    rev(OutR,Out).
regroupe_evolutions2([[T,V,E]|R],[],Out) :-
    regroupe_evolutions2(R,[[T,T,V,V,E]],Out).
regroupe_evolutions2([[T,V,E]|R],[[T1,T2,V1,V2,E]|R2],Out) :-
    (V2 >= V1, V > V2 ; V2 =< V1, V < V2 ; V2 = V1, V = V2),!,
    regroupe_evolutions2(R,[[T1,T,V1,V,E]|R2],Out).
regroupe_evolutions2([[T,V,E]|R],[[T1,T2,V1,V2,E]|R2],Out) :-
    !,
    regroupe_evolutions2(R,[[T2,T,V2,V,E],[T1,T2,V1,V2,E]|R2],Out).
regroupe_evolutions2([[T,V,E1]|R],[[T1,T2,V1,V2,E2]|R2],Out) :-
    regroupe_evolutions2(R,[[T,T,V,V,E1],[T1,T2,V1,V2,E2]|R2],Out).

regroupe_evolutions([],OutR,Out) :-
    rev(OutR,Out).
regroupe_evolutions([[T,V]|R],[],Out) :-
    regroupe_evolutions(R,[[T,T,V,V]],Out).
regroupe_evolutions([[T,V]|R],[[T1,T2,V1,V2]|R2],Out) :-
    (V2 >= V1, V > V2 ; V2 =< V1, V < V2 ; V2 = V1, V = V2),!,
    regroupe_evolutions(R,[[T1,T,V1,V]|R2],Out).
regroupe_evolutions([[T,V]|R],[[T1,T2,V1,V2]|R2],Out) :-
    regroupe_evolutions(R,[[T2,T,V2,V],[T1,T2,V1,V2]|R2],Out).
```

B.4 Systèmes de PréTexte

Cette section liste les descripteurs de systèmes de la version de PréTexte qui a été adaptée à PostGraphe. La section 6.6 décrit PréTexte et son adaptation à PostGraphe.

```
systeme(constituant,[proposition,nominal,nom,
                     pronom,article,verbal,verbe,
                     adverbial,preposition,adverbe,adjectif],
        proposition).
systeme(type_prop,[complete,partielle_nominale,
                   partielle_adverbiale],complete).
systeme(niveau_prop,[primaire,secondaire],primaire).
systeme(mode_prop,[commande,enonce,intensionnel],enonce).
systeme(processus,[materiel,mental,relationnel,evolutif],erreur).
systeme(voix,[active,passive],erreur).
systeme(adjoint_secondaire,[oui,non],non).
systeme(extension_active,[explicite,implicite],erreur).
systeme(reflexivite,[reflexif,non_reflexif],non_reflexif).
systeme(extension_passive,[explicite,implicite],erreur).
systeme(aspect,[situation,evenement],erreur).
systeme(type_sit,[resultante,ouverte],erreur).
systeme(localisation,[passe,present,futur,atemporelle],erreur).
systeme(circ_temporelle,[oui,non],non).
systeme(circ_temp_thematise,[oui,non],oui).
systeme(reference_acteur,[anaphorique,non_anaphorique],
        non_anaphorique).
systeme(reference_but,[anaphorique,non_anaphorique],
        non_anaphorique).
systeme(genre_gn,[masculin,feminin],erreur).
systeme(nombre_gn,[singulier,pluriel],singulier).
systeme(reference,[textuelle,exterieure],exterieure).
systeme(anaphore,[designation,pronominalisation,
                  repetition],pronominalisation).
systeme(emphase_designation,[oui,non],non).
systeme(design_gn,[definie,indefinie],erreur).
systeme(quant_gn,[quantifie,non_quantifie],erreur).
systeme(ident_gn,[directe,indirecte],erreur).
systeme(referent_dir,[pays,autre],erreur).
systeme(id_ind,[identifie,non_identifie],erreur).
systeme(complexite,[compose,simple],simple).
systeme(complementation,[attributif,autre],autre).
systeme(specificateur,[oui,non],non).
systeme(type_specificateur,[position,autre],non).
systeme(type_nom,[propre,commun],commun).
systeme(genre_nom,[masculin,feminin],erreur).
```

```
systeme(nombre_nom,[singulier,pluriel],erreur).
systeme(type_pro,[interrogatif,personnel,relatif,
systeme(position_pro,[conjoint,disjoint],erreur).
systeme(fonction_pro,[subjectif,objectif,reflexif],objectif).
systeme(extension_pro,[direct,indirect],erreur).
systeme(pro_anim,[anime,inanime],erreur).
systeme(personne_pro,[1,2,3],erreur).
systeme(genre_pro,[masculin,feminin,nil],nil).
systeme(nombre_pro,[singulier,pluriel],erreur).
systeme(def_art,[defini,indefini,demonstratif],erreur).
systeme(genre_art,[masculin,feminin],erreur).
systeme(nombre_art,[singulier,pluriel],erreur).
systeme(loc_temps,[passe,present,futur],present).
systeme(tv_passe,[imp,pqp,pc_ps],erreur).
systeme(tv_present,[pres,pc_acc],erreur).
systeme(tv_futur,[fs,f_ant],erreur).
systeme(auxiliaire,[avoir,etre],avoir).
systeme(prolong_verbe,[requis,non_requis],non_requis).
systeme(prep_verbe,[destination,origine],erreur).
systeme(modificateur_verbal,[present,absent],absent).
systeme(verbe_mode,[participe,imperatif,indicatif,infinitif],
        infinitif).
systeme(temps_part,[passe,present],passe).
systeme(accord_part,[variable,invariable],invariable).
systeme(genre_verbe,[masculin,feminin,nil],nil).
systeme(nombre_verbe,[singulier,pluriel,nil],nil).
systeme(personne_verbe,[1,2,3,nil],nil).
systeme(temps_verbe,[passe,imparfait,present,futur],present).
systeme(adverbial_parenthese,[oui,non],non).
systeme(type_adverbial,[temporel,variation,autre],erreur).
systeme(type_variation,[bornee,delta],erreur).
systeme(forme_variation,[complete,concise],complete).
systeme(qte_ancrage,[unique,double],erreur).
systeme(ancrage,[anaphorique,deictique,autonome],erreur).
systeme(ancrage1,[anaphorique,deictique,autonome],erreur).
systeme(ancrage2,[anaphorique,deictique,autonome],erreur).
systeme(niveau_loc,[primaire,secondaire],primaire).
systeme(aspect_loc,[duratif,ponctuel],erreur).
systeme(des_duree_fermee,[directe,relative],erreur).
systeme(type_duree,[bornee,quantifiee],erreur).
systeme(ancrage_duree,[ante,post,double],erreur).
systeme(persp_duree,[interne,externe],erreur).
systeme(des_zone_temp,[directe,relationnelle],erreur).
systeme(ref_int,[occurrentielle,chronologique],erreur).
systeme(nomin_occ,[nominalisee,non_nominalisee],non_nominalisee).
systeme(zone_ref,[ellipse,present],erreur).
systeme(type_relation,[anteriorite,posteriorite,entre,deb,term],
```

```
        erreur).
systeme(repere_loc,[enonciation,reference,nil],erreur).
systeme(ref_cadre,[implicite,explicite],erreur).
systeme(denom_zone,[directe,pos_dans_ordre],erreur).
systeme(pos_repere_loc,[positionne,non_positionne],erreur).
systeme(persp_enon,[ponctuel,duratif],erreur).
systeme(unite_cadre,[minute,heure,jour,semaine,mois,
                     saison,annee,indefini],indefini).
systeme(exp_decalage,[implicite,explicite],explicite).
systeme(ref_agent_loc_occ,[idem,differents],erreur).
systeme(id_decalage,[defini,indefini],erreur).
systeme(val_dec,[1,2,autre],erreur).
systeme(cat_prep,[temporelle,autre],erreur).
systeme(genre_adj,[masculin,feminin],erreur).
systeme(nombre_adj,[singulier,pluriel],erreur).
systeme(cat_adj,[numeral,autre],erreur).
systeme(cat_adj_num,[cardinal,ordinal],erreur).
systeme(cat_adv,[temporel,autre],erreur).
```

Annexe C

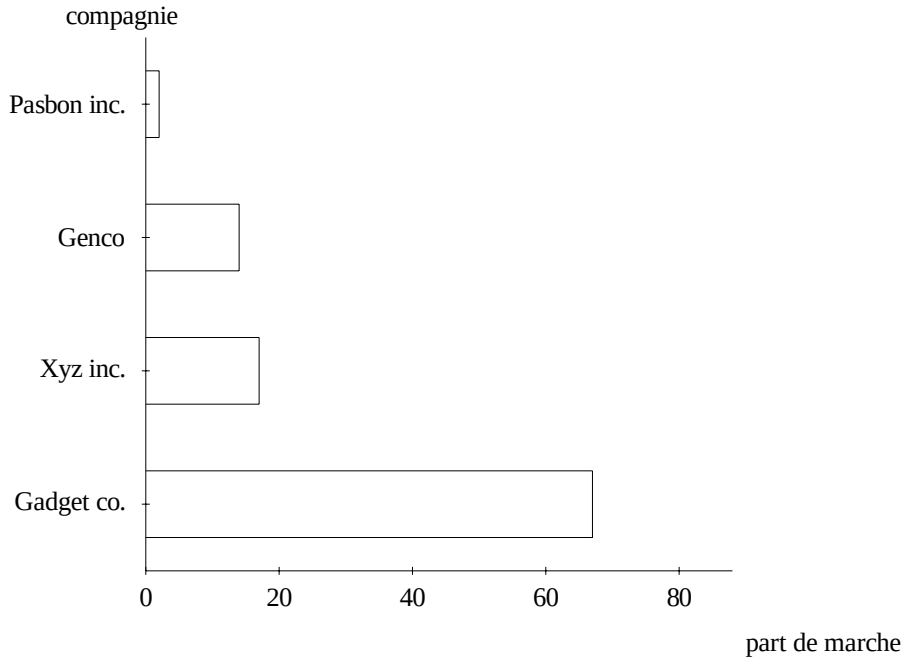
Exemples d'utilisation du module graphique

Cette annexe présente une série d'exemples d'utilisation du module graphique de PostGraphe. Ce module est utilisé de façon interne par le système pour réaliser les graphiques statistiques et les tableaux mais il a été conçu pour être utilisable de façon directe en Prolog sans avoir à passer par le reste de PostGraphe. Pour les détails d'implantation du module et pour une description de la syntaxe de ses prédicats, voir la section 6.5.1.

Les données utilisées dans ces exemples sont fictives, sauf pour les exemples C.4.7, C.5.1, C.5.2 et C.6, qui sont inspirés de données de *Statistiques Canada* sur le chômage et l'emploi, et les exemples C.2.6 et C.2.7 qui ont été produits pour la thèse de doctorat de Leila Kosseim [Kosseim, 1995].

C.1 Barres

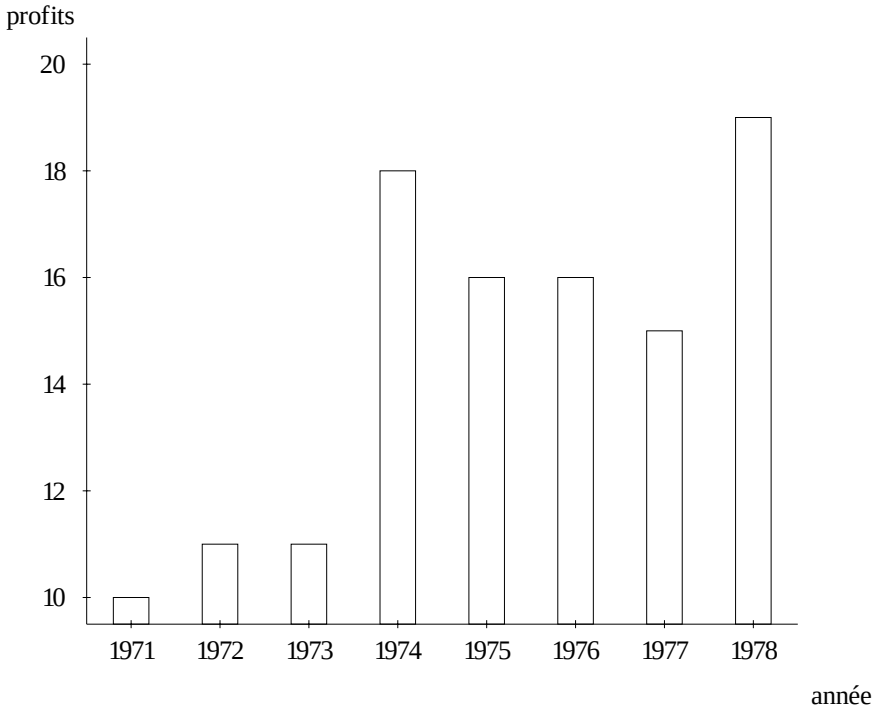
```
barres('part de marche',compagnie,
       intervalle(2,67),
       enumeration(['Gadget co.','Xyz inc.','Genco','Pasbon inc.']),
       [[67,'Gadget co.'],
        [17,'Xyz inc.'],
        [14,'Genco'],
        [2,'Pasbon inc.']])
```



C.2 Colonnes

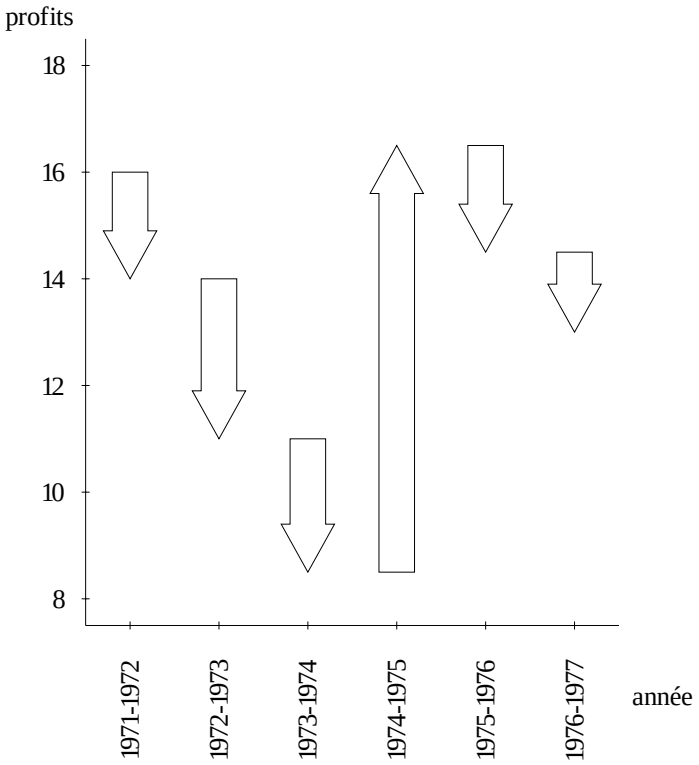
C.2.1 Simples

```
colonnes('année',profits,
         enumeration([1971,1972,1973,1974,1975,1976,1977,1978]),
         intervalle(10,19),
         [[1971,10],
          [1972,11],
          [1973,11],
          [1974,18],
          [1975,16],
          [1976,16],
          [1977,15],
          [1978,19]])
```



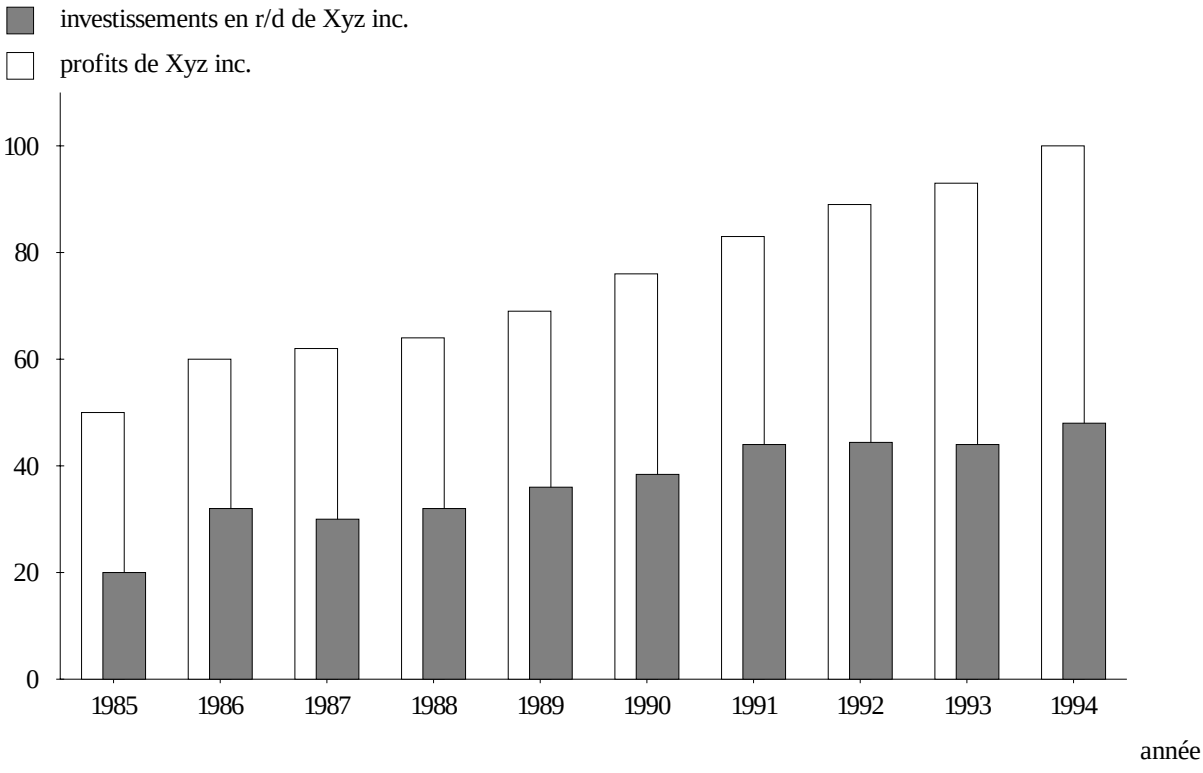
C.2.2 Flèches

```
colonnes('année',profits,
         enumeration(['1971-1972','1972-1973','1973-1974',
                      '1974-1975','1975-1976','1976-1977']),
         intervalle(8.500000,16.500000),
         [['1971-1972',16,14],
          ['1972-1973',14,11],
          ['1973-1974',11,8.5],
          ['1974-1975',8.5,16.5],
          ['1975-1976',16.5,14.5],
          ['1976-1977',14.5,13]],
         fleches)
```



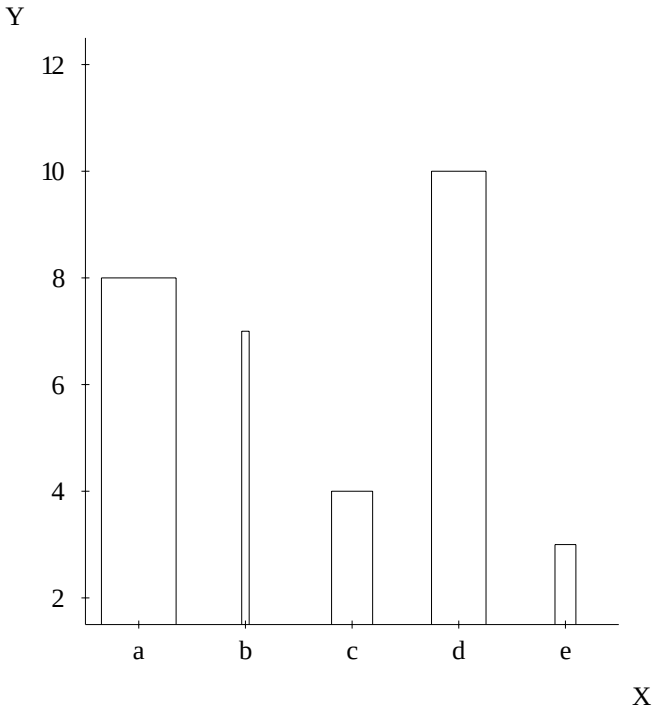
C.2.3 Juxtaposition

```
colonnes('année',
         ['profits de Xyz inc.',
          'investissements en r/d de Xyz inc.'],
         enumeration([1985,1986,1987,1988,1989,1990,
                      1991,1992,1993,1994]),
         intervalle(0,100),
        [[1985,50,20],
         [1986,60,32],
         [1987,62,30],
         [1988,64,32],
         [1989,69,36],
         [1990,76,38.4],
         [1991,83,44],
         [1992,89,44.4],
         [1993,93,44],
         [1994,100,48]],
        juxtaposition)
```



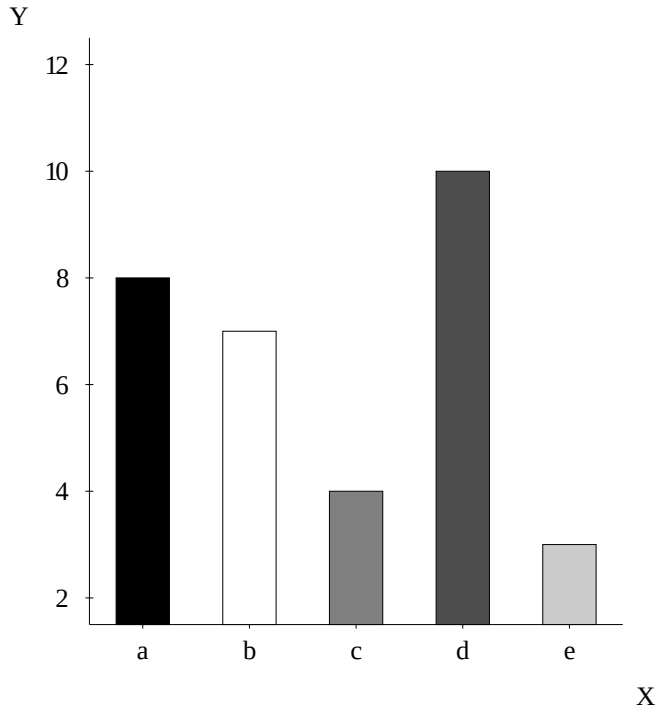
C.2.4 Largeur

```
colonnes('X','Y','Z',
         enumeration([a,b,c,d,e]),
         intervalle(3,11),
         intervalle(10,20),
        [[b,7,10],
         [c,4,15],
         [a,8,20],
         [e,3,12],
         [d,10,17]],
        largeur)
```



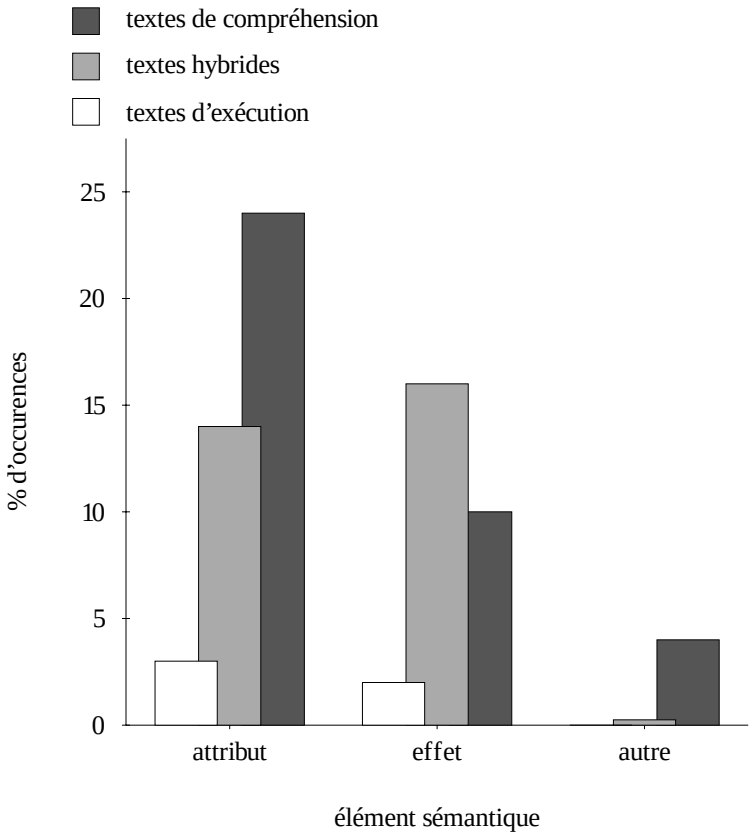
C.2.5 Gris

```
colonnes('X','Y','Z',
         enumeration([a,b,c,d,e]),
         intervalle(3,11),
         intervalle(10,20),
        [[b,7,10],
         [c,4,15],
         [a,8,20],
         [e,3,12],
         [d,10,17]],
        gris)
```



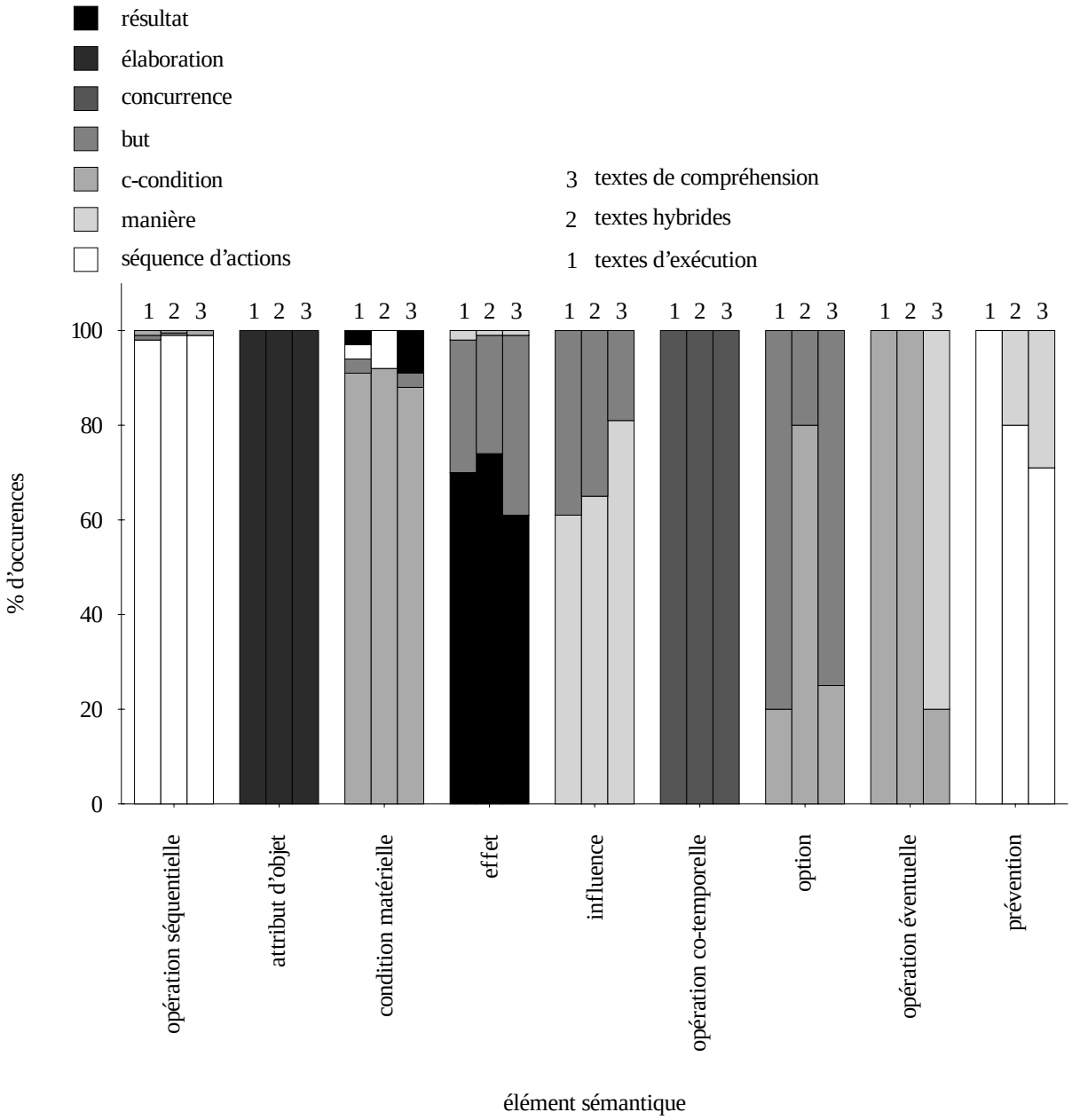
C.2.6 Juxtaposition 2

```
colonnes('élément sémantique','% d'occurences',
         ['textes d'exécution',
          'textes hybrides',
          'textes de compréhension'],
         enumeration([attribut,effet,autre]),
         intervalle(0,24),
        [[attribut,3,14,24],
         [effet,2,16,10],
         [autre,0,0.250000,4]],
        'juxtaposition2')
```



C.2.7 Super Juxtaposition

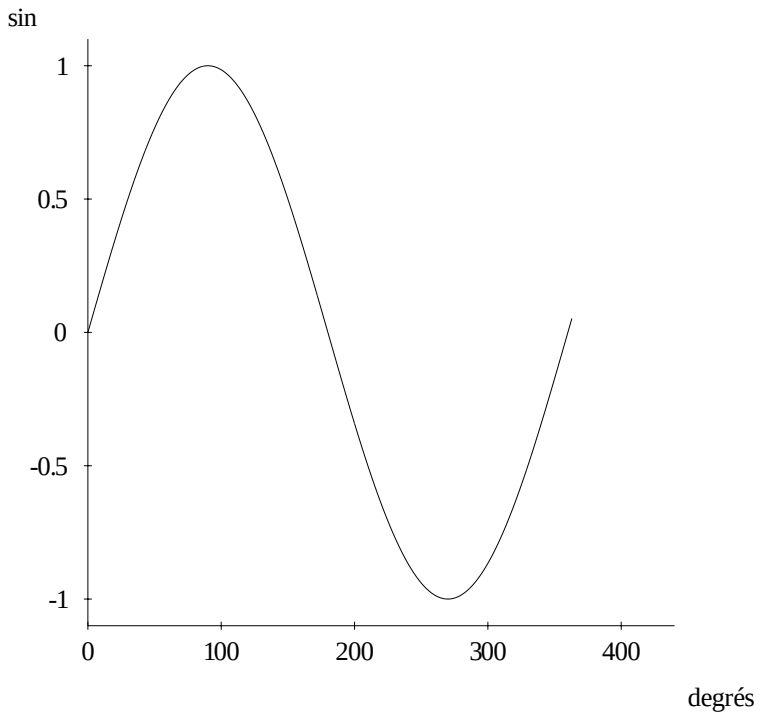
```
colonnes('élément sémantique','% d'occurences',
         ['textes d'exécution','textes hybrides',
          'textes de compréhension'],
         ['séquence d'actions','manière','c-condition',
          but,concurrence,'élaboration','résultat'],
         enumeration(['opération séquentielle','attribut d'objet',
                      'condition matérielle',effet,influence,
                      'opération co-temporelle',option,
                      'opération éventuelle','prévention']),
         enumeration([but,'séquence d'actions','c-condition',
                      concurrence,'manière','élaboration',
                      'résultat']),
         intervalle(0,100),
         [['opération séquentielle','séquence d'actions',98,99,99],
          ['opération séquentielle',but,1,0.500000,0],
          ['opération séquentielle','c-condition',1,0.500000,1],
          ['condition matérielle','c-condition',91,92,88],
          ['condition matérielle',but,3,0,3],
          ['condition matérielle','séquence d'actions',3,8,0],
          ['condition matérielle','résultat',3,0,9],
          ['attribut d'objet','élaboration',100,100,100],
          [effet,'résultat',70,74,61],
          [effet,but,28,25,38],
          [effet,'manière',2,1,1],
          [influence,'manière',61,65,81],
          [influence,but,39,35,19],
          ['opération co-temporelle',concurrence,100,100,100],
          [option,'c-condition',20,80,25],
          [option,but,80,20,75],
          ['prévention','séquence d'actions',100,80,71],
          ['prévention','manière',0,20,29],
          ['opération éventuelle','c-condition',100,100,20],
          ['opération éventuelle','manière',0,0,80]],
         superjuxta)
```


résultat
élaboration
concurrence
but
c-condition
manière
séquence d'actions
3 textes de compréhension
2 textes hybrides
1 textes d'exécution
% d'occurences
0
20
40
60
80
100
1 2 3
opération séquentielle
attribut d'objet
condition matérielle
effet
influence
opération co-temporelle
option
opération éventuelle
prévention
élément sémantique

C.3 Courbes

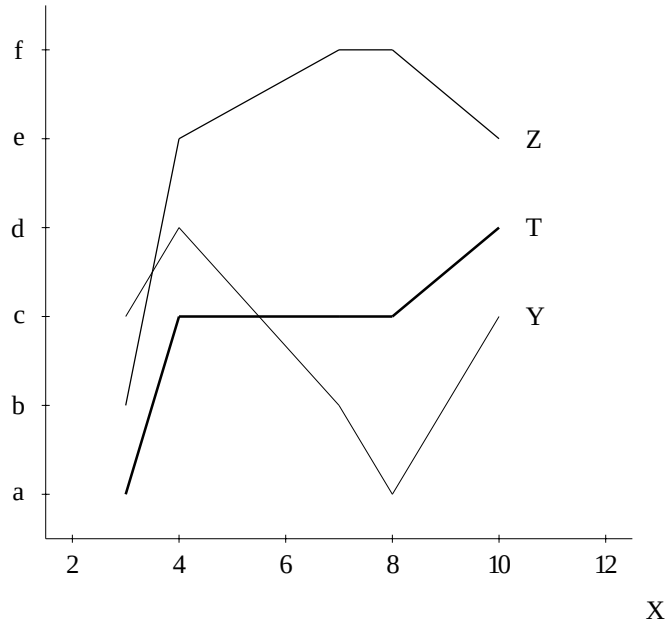
C.3.1 Simples

```
courbe('degrés',sin,
       intervalle(0,360),
       intervalle(-1,1),
      [[0,0],
       [1,0.017452],
       [2,0.034899],
       ...,
       [359,-0.017452],
       [360,-0.000000]])
```



C.3.2 Superposées

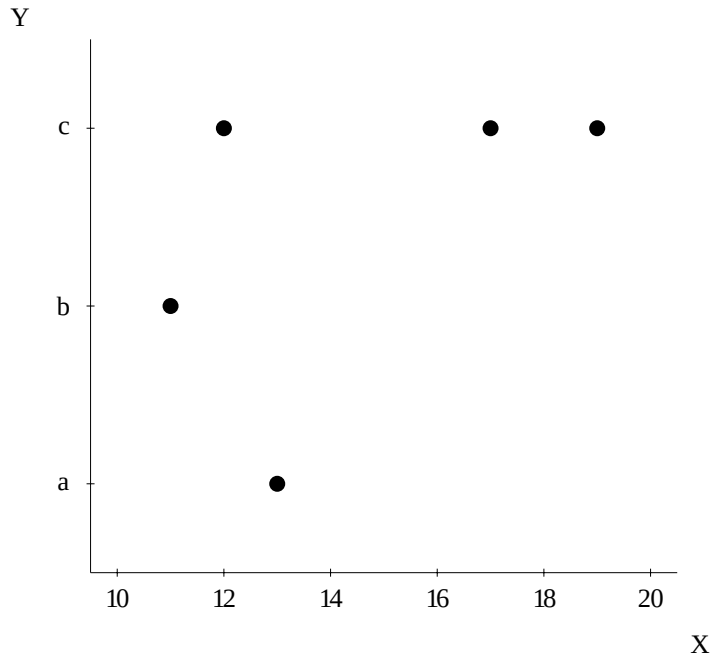
```
courbe('X',['Y','Z','T'],
       intervalle(3,11),
       enumeration([a,b,c,d,e,f]),
       [[7,b,f,c],
        [4,d,e,c],
        [8,a,f,c],
        [3,c,b,a],
        [10,c,e,d]])
```



C.4 Points

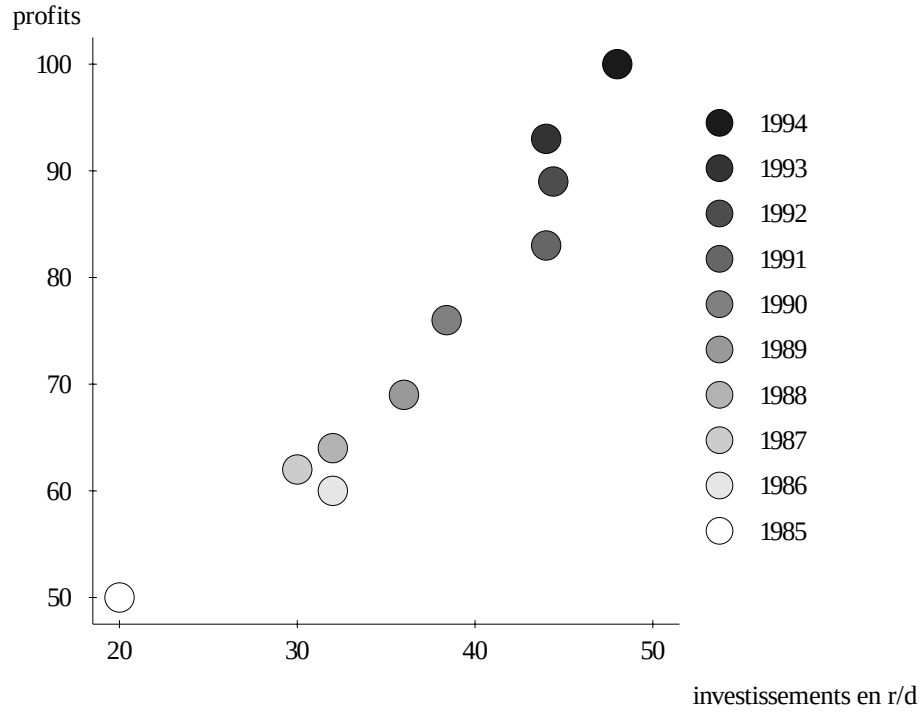
C.4.1 Simples

```
points('X','Y',
       intervalle(10,20),
       enumeration([a,b,c]),
       [[11,b],
        [17,c],
        [13,a],
        [12,c],
        [19,c]])
```



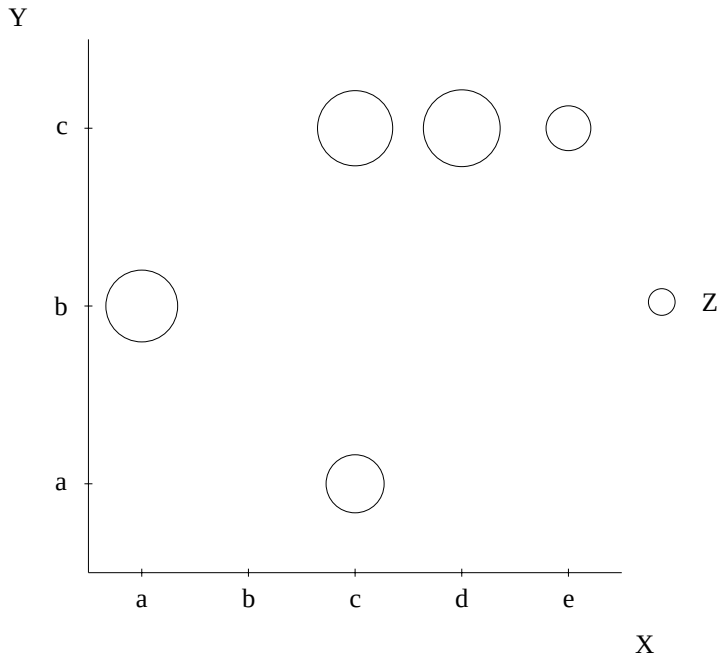
C.4.2 Gris

```
points('investissements en r/d',profits,'année',
       intervalle(20,48),
       intervalle(50,100),
       enumeration([1985,1986,1987,1988,1989,1990,
                    1991,1992,1993,1994]),
       [[20,50,1985],
        [32,60,1986],
        [30,62,1987],
        [32,64,1988],
        [36,69,1989],
        [38.4,76,1990],
        [44,83,1991],
        [44.4,89,1992],
        [44,93,1993],
        [48,100,1994]],
       gris)
```



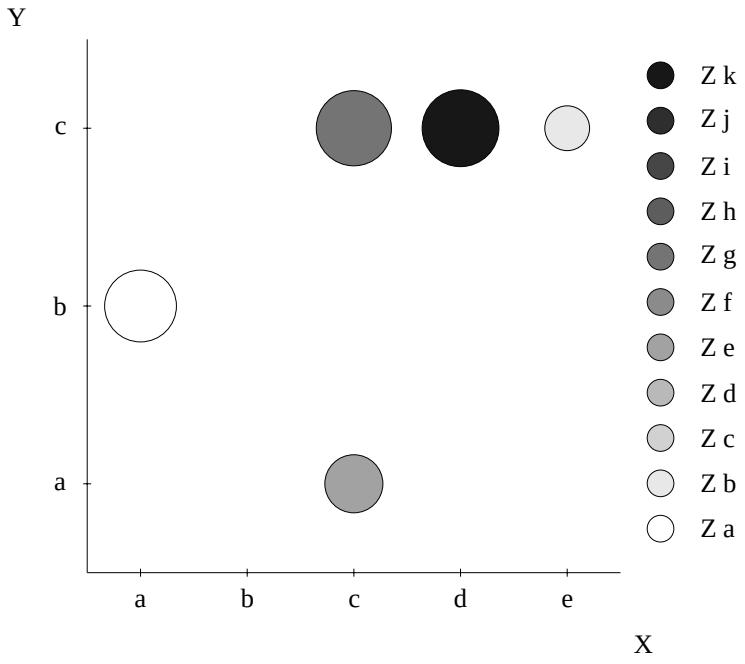
C.4.3 Surface

```
points('X','Y','Z',
       enumeration([a,b,c,d,e]),
       enumeration([a,b,c]),
       intervalle(0,20),
       [[a,b,17],
        [e,c,5],
        [c,a,10],
        [c,c,19],
        [d,c,20]],
       surface)
```



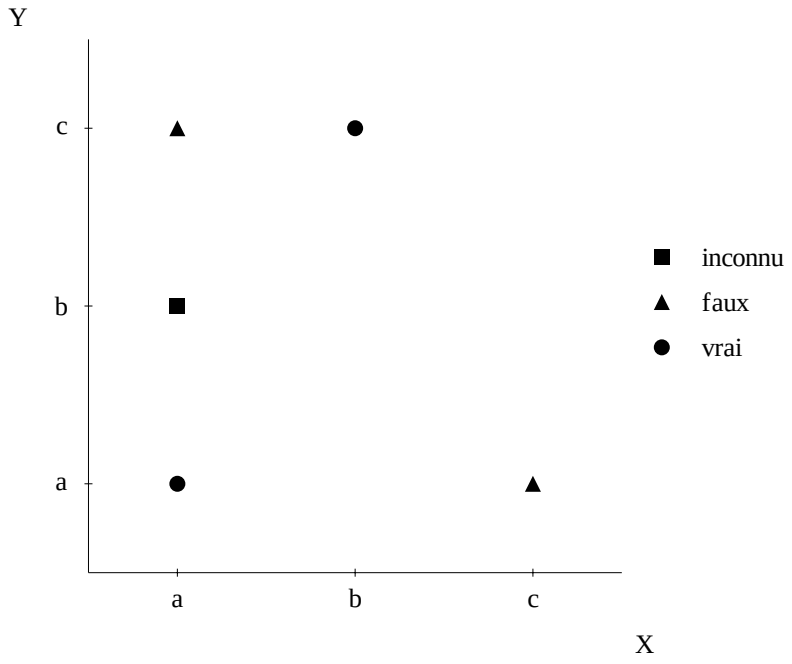
C.4.4 Surface et gris

```
points('X','Y','Z','T',
       enumeration([a,b,c,d,e]),
       enumeration([a,b,c]),
       intervalle(0,20),
       enumeration([a,b,c,d,e,f,g,h,i,j,k]),
      [[a,b,17,a],
       [e,c,5,b],
       [c,a,10,e],
       [c,c,19,g],
       [d,c,20,k]])
```



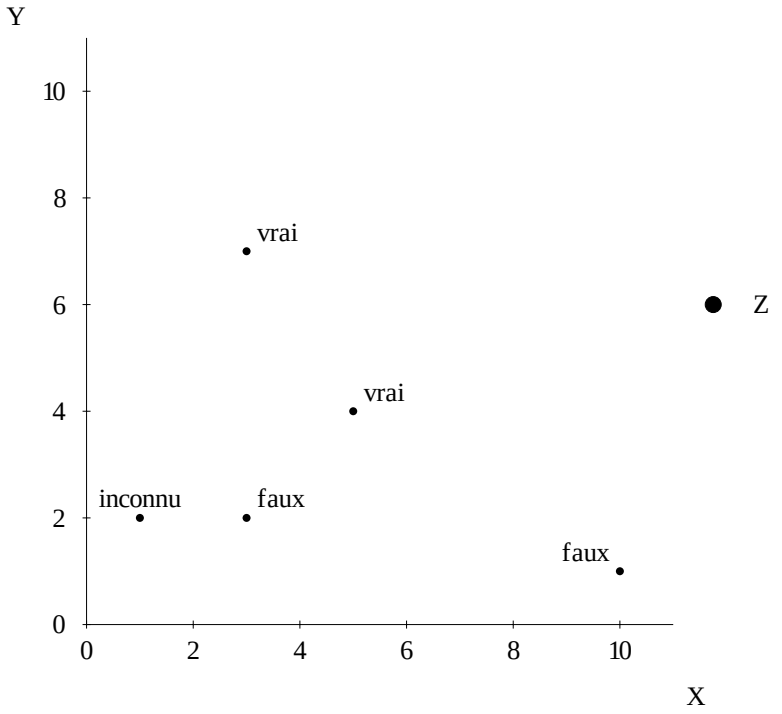
C.4.5 Forme

```
points('X','Y','Z',
       enumeration([a,b,c]),
       enumeration([a,b,c]),
       enumeration([vrai,faux,inconnu]),
       [[a,b,inconnu],
        [a,a,vrai],
        [c,a,faux],
        [b,c,vrai],
        [a,c,faux]],
       forme))
```



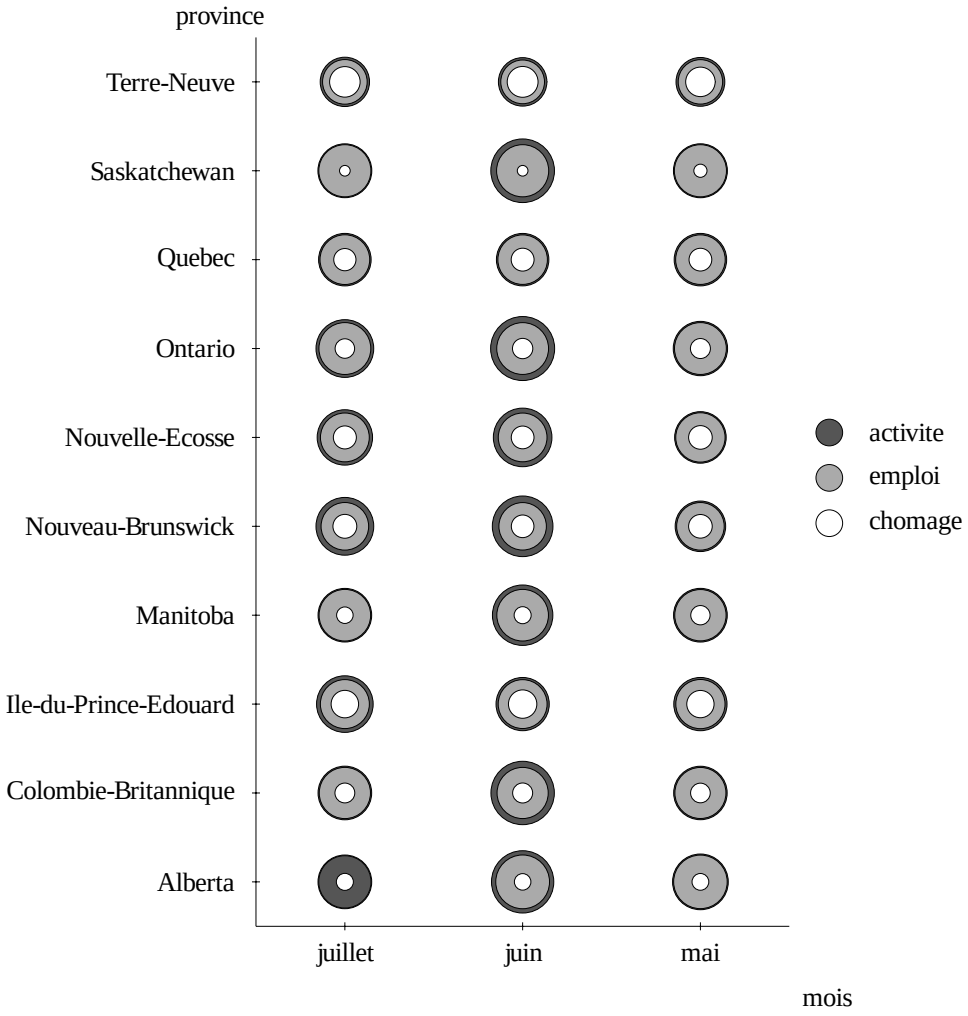
C.4.6 Étiquette

```
points('X','Y','Z',
       intervalle(0,10),
       intervalle(0,10),
       enumeration([vrai,faux,inconnu]),
       [[1,2,inconnu],
        [3,2,faux],
        [3,7,vrai],
        [10,1,faux],
        [5,4,vrai]],
       etiquette)
```



C.4.7 Surface et superposition

```
points(mois,province,[chomage,emploi,activite],
       enumeration([juillet,juin,mai]),
       enumeration(['Alberta','Colombie-Britannique',
                    'Ile-du-Prince-Edouard','Manitoba',
                    'Nouveau-Brunswick','Nouvelle-Ecosse',
                    'Ontario','Quebec','Saskatchewan',
                    'Terre-Neuve']),
       intervalle(7.2,98.8),
       [[juillet,'Alberta',8.5,66.8,63],
        ...,
        [mai,'Terre-Neuve',18.6,43.9,54]],
       surface)
```



C.5 Tableaux

C.5.1 1 dimension

```
table([x,y,z],
      [[juillet,'Colombie-Britannique',9.9],
       [juillet,'Ile-du-Prince-Edouard',16.3],
       [juillet,'Manitoba',8.3],
       ...,
       [mai,'Saskatchewan',7.4],
       [mai,'Terre-Neuve',18.6]])
```

x	y	z
juillet	Colombie-Britannique	9.9
juillet	Ile-du-Prince-Edouard	16.3
juillet	Manitoba	8.3
juillet	Nouveau-Brunswick	12.8
juillet	Nouvelle-Ecosse	12.1
juillet	Ontario	9.7
juillet	Quebec	11.5
juillet	Saskatchewan	7.2
juillet	Terre-Neuve	19.9
juin	Alberta	8.3
juin	Colombie-Britannique	10
juin	Ile-du-Prince-Edouard	17.3
juin	Nouveau-Brunswick	11.8
juin	Nouvelle-Ecosse	11.9
juin	Ontario	10.2
juin	Quebec	12
juin	Saskatchewan	7.2
juin	Terre-Neuve	20.3
mai	Alberta	8.4
mai	Colombie-Britannique	9.8
mai	Ile-du-Prince-Edouard	16.1
mai	Manitoba	9.6
mai	Nouveau-Brunswick	12.6
mai	Nouvelle-Ecosse	12.6
mai	Ontario	10
mai	Quebec	11.9
mai	Saskatchewan	7.4
mai	Terre-Neuve	18.6

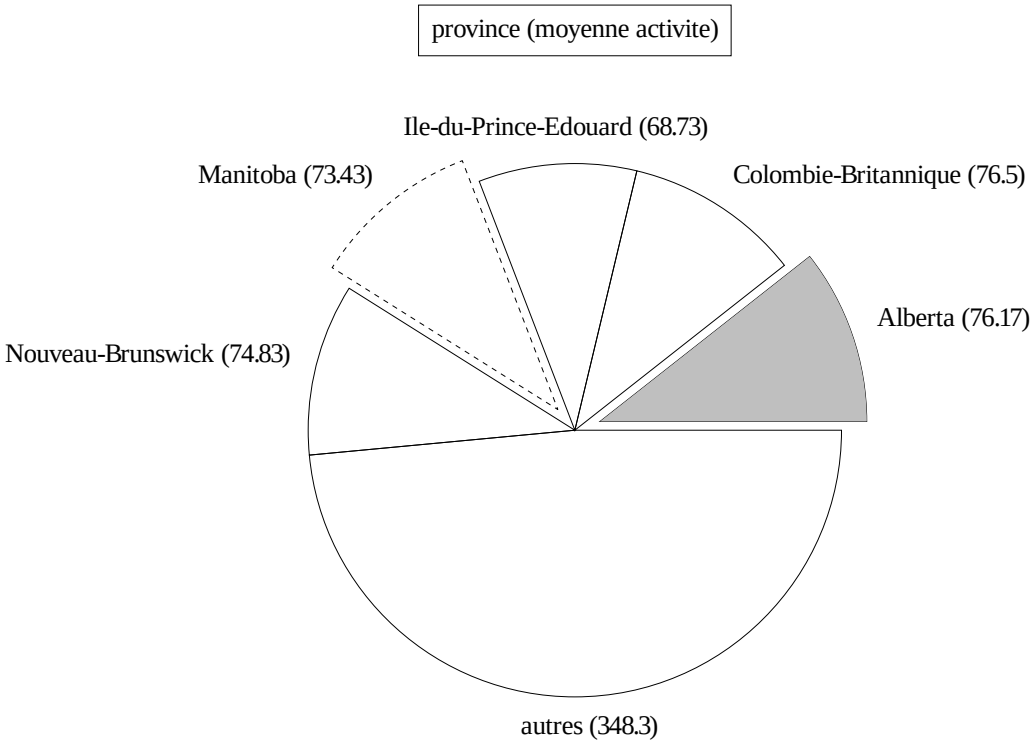
C.5.2 2 dimensions

```
table(x,y,['z1','z2','z3'],
      [mai,juin,juillet],
      ['Alberta',
       'Colombie-Britannique',
       'Ile-du-Prince-Edouard',
       'Manitoba',
       'Nouveau-Brunswick',
       'Nouvelle-Ecosse',
       'Ontario',
       'Quebec',
       'Saskatchewan',
       'Terre-Neuve'],
      [[mai,'Terre-Neuve',18.6,54,43.9],
       [mai,'Ile-du-Prince-Edouard',16.1,65.6,55],
       [mai,'Nouvelle-Ecosse',12.6,61,55.3],
       [mai,'Nouveau-Brunswick',12.6,58.3,51],
       ...,
       [juillet,'Colombie-Britannique',9.9,66.6,60.1]])
```

x	mai	juin	juillet	mai	juin	juillet	mai	juin	juillet
y	z1	z1	z1	z2	z2	z2	z3	z3	z3
Alberta	8.4	8.3	8.5	72.5	93	63	66.4	67	66.8
Colombie-Britannique	9.8	10	9.9	66.2	96.7	66.6	59.7	60	60.1
Ile-du-Prince-Edouard	16.1	17.3	16.3	65.6	65.5	75.1	55	54.2	54.5
Manitoba	9.6	8.5	8.3	66.8	86.9	66.6	60.4	61.2	61.1
Nouveau-Brunswick	12.6	11.8	12.8	58.3	87.6	78.6	51	50.8	51.1
Nouvelle-Ecosse	12.6	11.9	12.1	61	81.1	71.6	55.3	53.9	54.1
Ontario		10.2	9.7		98.8	78.5		61.7	61.9
Quebec	11.9		11.5	63.5		63.6	55.9		56.3
Saskatchewan	7.4	7.2	7.2	67.5	97.5	67.3	62.5	62.6	62.5
Terre-Neuve	18.6	20.3	19.9	54	53.2	55.3	43.9	42.4	44.3

C.6 Tartes

```
tarte(province,'moyenne activite',
      [['Alberta',76.166667],
       ['Colombie-Britannique',76.5],
       ['Ile-du-Prince-Edouard',68.733333],
       ['Manitoba',73.433333],
       ['Nouveau-Brunswick',74.833333],
       [autres,348.3]],
      [0,3])
```



Bibliographie

Adobe Systems Incorporated (1985). *PostScript Language Tutorial and Cookbook*. Addison-Wesley, Reading, MA, USA.

Adobe Systems Incorporated (1990a). *Adobe Type 1 Font Format—Version 1.1*. Addison-Wesley, Reading, MA, USA.

Adobe Systems Incorporated (1990b). *PostScript Language Reference Manual*. Addison-Wesley, Reading, MA, USA, édition 2.

André, E., Finkler, W., Graf, W., Rist, T., Schauder, A. et Wahlster, W. (1993). WIP: the automatic synthesis of multimodal presentations. Dans [Maybury, 1993].

André, E., Herzog, G. et Rist, T. (1994). Multimedia Presentation of Interpreted Visual Data. *Proceedings of* 12^{th} *National Conference on Artificial Intelligence (AAAI-94), Workshop on the Integration of Natural Language and Vision Processing*, pages 74–82.

André, E. et Rist, T. (1994). Referring to World Objects with Text and Pictures. *Proceedings of the* 15^{th} *International Conference on Computational Linguistics (COLING-94)*, pages 530–534.

André, E. et Rist, T. (1995). *Generating Coherent Presentations Employing Textual and Visual Material*, chapitre à paraitre dans AI Review.

Balogun, O. Y. (1978). The decagraph: A substitute for the pie graph? *The Cartographic Journal*, 15:78–85.

Bélanger, M. (1990). Génération intégrée de textes et de graphiques. Mémoire de maîtrise, Département d'informatique et de recherche opérationnelle, Université de Montréal.

Berry, M. (1975). *An Introduction to Systemic Linguistics, Volume 1 – Structures and Systems*. St. Martin Press, New York.

Berry, M. (1976). *An Introduction to Systemic Linguistics, Volume 2 – Levels and Links*. St. Martin Press, New York.

Bertin, J. (1983). *Semiology of Graphics*. The University of Wisconsin Press. Translated by William J. Berg.

Bojin, J. et Dunand, M. (1982). *Documents et exposés efficaces: messages, structure du raisonnement, illustrations graphiques*. Éditions d'Organisation, Paris.

Burnhill, P., Hartley, J. et Young, M. (1976). Tables in text. *Applied Ergonomics*, 7(1):13–18.

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

Chambers, J. M., Cleveland, W. S., Kleiner, B. et Tukey, P. A. (1983). *Graphical Methods for Data Analysis*. Statistics/Probability. Wadsworth.

Cleveland, W. S. (1980). *The Elements of Graphing Data*. Wadsworth Advanced Books and Software.

Cleveland, W. S. et McGill, M. E. (1988). *Dynamic Graphics for Statistics*. Statistics/Probability. Wadsworth.

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

Contant, C. (1985). Génération automatique de texte: application au sous-langage boursier français. Mémoire de maîtrise, Département de linguistique et de philologie, Université de Montréal.

Contant, C. (1986). Génération automatique de rapports boursiers français et anglais. *Revue québécoise de linguistique*, 17(1):197–222.

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

Date, C. J. (1988). *An Introduction to Database Systems*, volume I. Addison-Wesley, édition 4.

DeSanctis, G. et Jarvenpaa, S. L. (1985). An investigation of the "tables versus graphs" controversy in a learning environment. Dans Gallegos, L., Welke, R. et Wetherbe, J., éditeurs, *Proceedings of the 6^{th} international conference on information systems*, pages 134–144.

Eells, W. C. (1926). The Relative Merits of Circles and Bars For Presenting Component Parts. *Journal of the American Statistical Association*, XXI(154):119–132.

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

Elhadad, M. (1991). FUF: the Universal Unifier. User Manual Version 5.0. Rapport technique, Columbia University.

Fasciano, M. et Lapalme, G. (1994). Génération intégrée de textes et de graphiques statistiques. 62^{e} congrès de l'ACFAS, colloque sur le traitement automatique du français écrit.

Fasciano, M. et Lapalme, G. (1996). Postgraphe: a system for the generation of statistical graphics and text. Dans *to appear in "Proceedings of the 8th International*

Workshop on Natural Language Generation (INLG-96)". Herstmonceux, Sussex, UK.

Feiner, S. et McKeown, K. (1990). Coordinating Text and Graphics in Explanation Generation. Dans *Proceedings of the* 8^{th} *National Conference on Artificial Intelligence (AAAI-90)*, volume 1, pages 442–449.

Feiner, S. et McKeown, K. (1991). Automating the generation of coordinated multimedia explanations. *Multimedia Information Systems*, 24(10):33–41.

Feiner, S. K. (1987). Computer generation of pictorial explanations. Rapport technique CS-87-30, Brown University.

Gagnon, M. (1993). *Expression de la localisation temporelle dans un générateur de texte*. Thèse de doctorat, Département d'informatique et de recherche opérationnelle, Université de Montréal. Publication 888.

Gagnon, M. et Lapalme, G. (1996). From conceptual time to linguistic time. *Computational Linguistics*, 22(1):91–127.

Graham, J. L. (1937). Illusory trends in the observation of bar graphs. *Journal of Experimental Psychology*, 6(20):597–608.

Hartley, J. (1981). Eighty Ways of Improving Instructional Texts. *IEEE Transactions on Professional Communication*, PC-24(1):17–27.

Hartley, J., Young, M. et Burnhill, P. (1975). On the typing of tables. *Applied Ergonomics*, 6(1):39–42.

Holzgang, D. A. (1987). *Understanding PostScript Programming*. Sybex, 2021 Challenger Driver, Suite 100, Alameda, CA 94501, USA.

Holzgang, D. A. (1989). *PostScript Programmer's Reference Guide*. Scott, Foresman and Company, Glenview, IL, USA.

Hovy, E. H. (1993). Automated discourse generation using discourse structure relations. *Artificial Intelligence*, 63:341–385.

Iordanskaja, L., Kim, M., Kittredge, R., Lavoie, B. et Polguère, A. (1992). Generation of extended bilingual statistical reports. *Proceedings of the* 15^{th} *International Conference on Computational Linguistics (COLING-92)*, pages 1019–1023.

Jarvenpaa, S. L. et Dickson, G. W. (1988). Graphics and Managerial Decision Making: Research Based Guidelines. *Communication of the ACM*, 31(6):764–774.

Kittredge, R. (1982). Variation and homogeneity of sublanguages. Dans Kittredge, R. et Lehrberger, J., éditeurs, *Sublanguage: Studies of Language Restricted Semantic Domains*, chapitre 4, pages 107–137.

Kittredge, R. (1983). Semantic processing of texts in restricted sublanguages. *Computers and mathematics with applications*, 9(1):45–58.

Kosseim, L. (1995). *Planification de textes d'instructions: Sélection du contenu et de la structure rhétorique*. Thèse de doctorat, Département d'informatique et de recherche opérationnelle, Université de Montréal. Publication 994.

Kosslyn, S. M. (1985). Graphics and Human Information Processing. *Journal of The American Statistical Association*, 80(391):499–512.

Kruskal, W. (1982). Criteria for Judging Statistical Graphics. *Utilitas Mathematicas*, 21B:283–311.

Kukich, K. (1983). Knowledge-based report generation: A technique for automatically generating natural language reports from databases. Dans *Proceedings of the ACM SIGIR Meeting*, pages 246–250. ACM.

Lamport, L. (1985). *LaTeX, A Document Preparation System: User's Guide and Reference Manual*. Addison-Wesley, Reading, MA, USA.

Lavoie, B. (1994). Étude sur la planification de texte: Application au sous-domaine de la population active. Mémoire de maîtrise, Université de Montréal.

Leung, Y. et Apperley, M. (1993). E^3: Towards the Metrication of Graphical Presentation Techniques for Large Data Sets. Dans Bass, L. J., Gornostaev, J. et C., U., éditeurs, *Lecture Notes in Computer Science: Human-Computer Interaction*, pages 125–140. Springer-Verlag.

Macdonald-Ross, M. (1977). Graphics in Texts. *Review of Research in Education*, 5:49–85.

Mackinlay, J. D. (1986a). *Automatic Design of Graphical Presentations*. Thèse de doctorat, Computer Science Department, Stanford University.

Mackinlay, J. D. (1986b). Automating the Design of Graphical Presentations of Relational Information. *ACM Transactions on Graphics*, 5(2):110–141.

Mahon, B. (1977). Statistics and decisions: The Importance of Communication and the Power of Graphical Presentation. *Journal of the Royal Statistical Society*, 140(Part 3):298–323.

Mann, W. et Thompson, S. (1988). Rhetorical Structure Theory: towards a functional theory of text organization. *TEXT*, 8(3):243–281.

Matthiessen, C. et Bateman, J. (1991). *Text Generation and Systemic-Functional Linguistics*. Pinter, London.

Maybury, M. (1991). Planning Multimedia Explanations Using Communicative Acts. *Proceedings of* 10^{th} *National Conference on Artificial Intelligence (AAAI-91)*, 1:61–66.

Maybury, M., éditeur (1993). *Intelligent Multimedia Interfaces*. AAAI Press, Cambridge, MA.

McKeown, K., Feiner, S., Robin, J., Seligmann, D. et Tanenblatt, M. (1992). Generating Cross-References for Multimedia Explanation. Dans *Proceedings of the* 10^{th} *National Conference on Artificial Intelligence (AAAI-92)*, pages 9–16, San Jose, Californie.

McKeown, K. R. (1985). Discourse strategies for generating natural language text. *Artificial Intelligence*, 27(1).

McKeown, K. R. et al. (1990). Language generation in COMET. Dans [Mellish et al., 1990].

Mellish, C., Dale, R. et Zock, M., éditeurs (1990). *Current Research in Language Generation*. Academic Press.

Mel'čuk, I. et Pertsov, N. (1987). A brief outline on the meaning-text theory and the corresponding linguistic model. Dans *Surface Syntax of English: A Formal Model within the Meaning-Text Framework*, chapitre 1, pages 12–45. John Benjamins Publishing Company, Amsterdam/Philadelphia.

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

Paris, C. L. (1991). The role of the user's domain knowledge in generation. *Computational Intelligence*, 7:71–93.

Pascual, E. (1991). *Représentation de l'architecture textuelle et génération de texte*. Thèse de doctorat, Université Paul Sabatier, Toulouse.

Pascual, E. (1993). The Modeling of Text Formatting for Text Generation. Rapport technique, IRIT, Toulouse.

Roth, S. E., éditeur (1988). *Real World PostScript*. Addison-Wesley, Reading, MA, USA.

Roth, S. F. et Mattis, J. (1990). Data Characterization for Intelligent Graphics Presentation. *Empowering People: CHI'90 Conference Proceedings*, pages 193–200.

Roth, S. F., Mattis, J. et Mesnard, X. (1991). Graphics and natural language as components of automatic explanation. Dans [Sullivan et Tyler, 1991], chapitre 10.

Schutz, H. (1961). An Evaluation of Formats for Graphic Trend Displays — Experiment II. *Human Factors*, 3:99–107.

Seligmann, D. et Feiner, S. (1991). Automated Generation of Intent-Based 3D Illustrations. *Computer Graphics*, 25(4):123–132.

Sullivan, J. W. et Tyler, S. W., éditeurs (1991). *Intelligent User Interfaces*. Frontier Series. ACM Press.

Tierney, L. (1990). *Lisp-Stat: An Object-Oriented Environment for Statistical Computing and Dynamic Graphics*. Wiley.

Timbal-Duclaux, L. (1990). *La Communication écrite scientifique et technique*. ESF Entreprise moderne d'édition, Paris.

Tufte, E. R. (1983). *The Visual Display of Quantitative Information*. Graphics Press.

Tufte, E. R. (1990). *Envisioning Information*. Graphics Press.

Tukey, J. W. (1977). *Exploratory Data Analysis*. Addison-Wesley.

Vernon, M. D. (1946). Learning from graphical material. *British Journal of Psychology*, 3(36):145–158.

Wall, L. (1993). *PERL: Practical Extraction and Report Language, Release 4.0 Patchlevel 36*. UNIX Manual Page.

Weiner, J. (1980). BLAH, a system which explains its reasoning. *Artificial Intelligence*, 15:19–48.

Wright, P. (1977). Presenting technical information: A survey of research findings. *Instructional Science*, 6:93–134.

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

Zelazny, G. (1972). *Choosing & Using Charts*. McKinsey & Company Inc.

Zelazny, G. (1989). *Dites-le avec des graphiques*. InterÉditions.