Cahiers GUTenberg

HTML → LaTeX → PDF, OU L'ENTRÉE DE TeX DANS L'ÈRE DE L'HYPERTEXTE

Yannis Haralambous

Cahiers GUTenberg, nº 19 (1995), p. 127-147.

<http://cahiers.gutenberg.eu.org/fitem?id=CG_1995___19_127_0>

HTML → LaTeX → PDF, ou l'entrée de TeX dans l'ère de l'hypertexte*

Yannis HARALAMBOUS

187, rue Nationale, 59800 Lille, France,
Internet : Yannis.Haralambous@univ-lille1.fr

Résumé. Nous décrivons le processus de production d'hyper-documents électroniques en passant par LaTeX et *Adobe Acrobat*. Après une discussion générale sur les avantages et inconvénients de LaTeX dans ce domaine, nous donnons une description de chaque étape ainsi que certaines précautions à prendre dans le but d'obtenir des documents Acrobat efficaces.

Le lecteur trouvera dans cet article la description des utilitaires DVIHPS (développé par Tom Rokicki et Sebastian Rahtz), `repere` et `recticrt` (développé par l'auteur), ainsi que les principes de base du format PDF.

Abstract. *In this paper we describe the process of creation of electronic hyperdocuments via LaTeX and Adobe Acrobat. After a general discussion on the advantages and disadvantages of LaTeX in this field, we give a detailed description of each step and a lot of caveats for the user willing to obtain efficient Acrobat documents.*

The reader will find in this paper a description of the software tools DVIHPS (developped by Tom Rokicki and Sebastian Rahtz), `repere` *and* `recticrt` *(developped by the author) as well as the basic principles of the PDF format.*

Mots clef : Acrobat DVIHPS PDF `repere` `recticrt`

1. Introduction

1.1. Hypertexte et LaTeX, quel rapport ?

Contrairement à l'*hyper*marché, l'*hyper*tension, l'*hyper*activité, l'*hyper*trophie et les vitesses *hyper*soniques, où le préfixe « *hyper* » exprime la grande quantité, voire l'excès, l'*hypertexte* n'est pas un texte très grand, mais un texte muni d'une

*. Cet article correspond à la présentation faite à Nanterre, le 19 janvier 1995 lors de la journée *Diffusion des documents électroniques.*

structure interne que certains logiciels d'affichage écran peuvent exploiter pour permettre la navigation à travers les pages du document.

Il y a donc un rapport trivial entre la notion d'hypertexte et le système de balisage de document LaTeX : tous les deux ajoutent de la structure au document. Par exemple, la notion de référence croisée de LaTeX correspond tout à fait à la notion de lien hypertexte.

La différence essentielle entre les deux concepts est le désintéressement de TeX vis-à-vis de la sortie écran. En effet, TeX manipule des boîtes destinées à contenir des caractères ou des images. La tâche de remplacement de ces boîtes par les caractères/images eux-mêmes succombe aux divers *pilotes* écran, télécopieur ou imprimante. TeX étant surtout un outil de composition typographique, l'écran n'est utilisé que pour l'épreuvage, et un affichage écran n'est jamais considéré comme le but final d'une compilation TeX.

Ce désintéressement de TeX vis-à-vis de l'écran est encore plus important parce que les constructions PostScript, introduites dans un fichier DVI par le biais de commandes `\special` ne sont généralement pas visibles à l'écran[1]. L'affichage écran est donc d'autant plus décevant dans ces cas-là, comparé au résultat imprimé.

Il s'avère que, pour la première fois dans son existence, TeX devient extrêmement utile pour créer des documents dont le but est d'être lus à l'écran. En effet, LaTeX est tout à fait propice à la production automatique de liens hypertexte, et les méthodes dont nous allons parler dans cet article permettent une conversion automatique vers un document hypertexte de tout document LaTeX déjà existant ! Il convient d'insister sur le fait qu'un tel document garde toute la qualité typographique de LaTeX, et peut être imprimé exactement de la même manière qu'avant.

1.2. Le schéma général

TeX et LaTeX lisent un fichier contenant le texte d'un document muni de balises structurelles et visuelles, et créent un deuxième fichier qui décrit la page imprimée avec une très grande précision. Ce fichier-sortie de TeX est appelé DVI (*DeVice Independent* = indépendant de plate-forme) parce qu'il ne contient que des données abstraites : l'emplacement de chaque caractère sur la page, le nom de la police dans laquelle le pilote trouvera les pixels de ce caractère, son code dans cette police, etc.

La visualisation/impression d'un fichier DVI suppose donc l'accesibilité d'un certain nombre de polices de caractères. Cela est toujours possible dans le cadre de systèmes informatiques reliés par réseau (stations de travail, gros systèmes), mais devient problématique dans le cadre de l'informatique personnelle. La situation devient encore plus grave lorsqu'on souhaite distribuer des documents

1. Sauf pour les fortunés d'entre nous qui utilisent des systèmes opérationnels munis de *Display PostScript*.

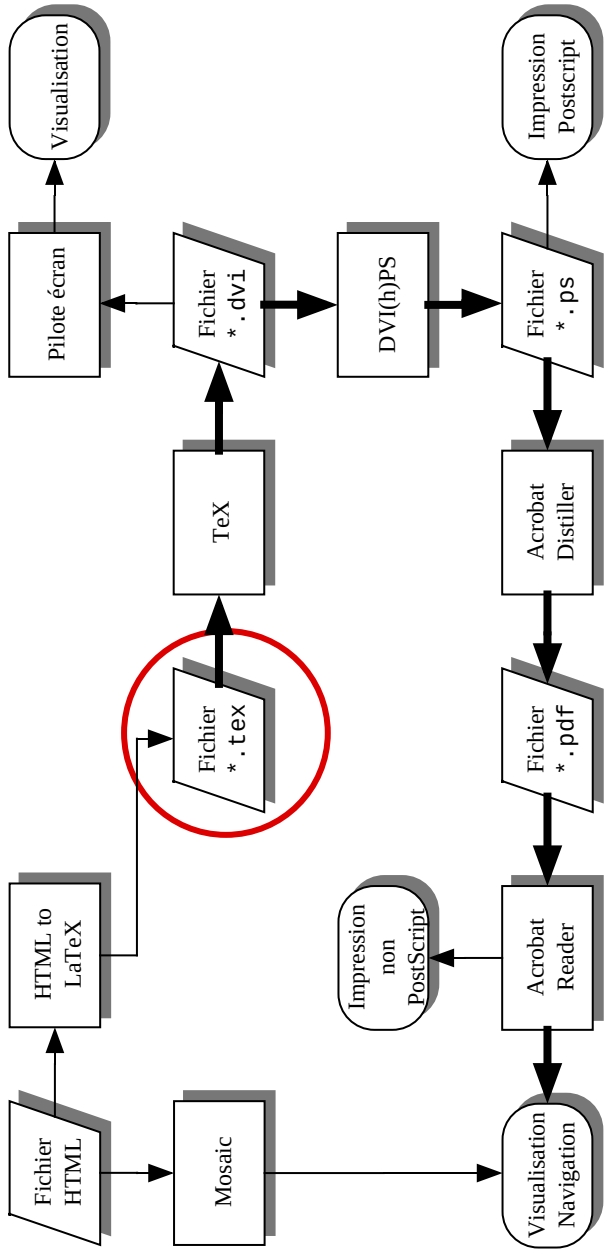

FIGURE 1 - *Schéma général*

électroniques : un document susceptible d'être visualisé et imprimé par le plus grand nombre de personnes peut difficilement être distribué en format DVI (cela ne concernerait alors que la communauté TEX, et encore il faudrait se limiter aux polices obsolètes CM, toujours très répandues aux États-Unis). En particulier, il serait impossible de mettre sur une disquette le document et suffisament d'utilitaires pour qu'il puisse être visualisé et imprimé *sur le champ*, sans qu'un système TEX soit installé.

Finalement, les liens hypertexte ne sont pas prévus dans la syntaxe du format DVI ; toute tentative de développement de pilote hypertexte utilisant le format DVI nous conduirait donc à la définition d'un nouveau format « Hyper-DVI », avec tous les problèmes de compatibilité et de réticence de la part de la communauté TEX qui est fière de la stabilité de ces outils. Il semble donc que le format DVI n'est pas le candidat idéal pour un format de fichier rapidement exploitable et suffisament interactif pour permettre l'intégration de liens hypertexte.

Que faire alors ? le choix évident pour un tel candidat — du moins, aujourd'hui — est le format PDF (*Portable Document Format*) d'Adobe Systems. Il s'agit d'une extension du langage PostScript, très proche de la syntaxe de fichiers produits par le logiciel Illustrator, avec deux grandes innovations vis-à-vis du langage PostScript pur : la visualisation et impression sur toute plate-forme, indépendamment des polices utilisées dans le document, et l'intégration de fonctionnalités hypertexte.

Nous reparlerons du format PDF (cf. 6.1). Voyons comment le passage par ce format s'intègre dans le processus de production de document (électronique ou imprimé). Dans le diagramme de la figure 1, les fichiers entrée-sortie sont indiqués par des symboles obliques, les logiciels par des rectangles, et les opérations (visualisation, impression etc.) par des symboles aux coins arrondis. Les flèches grasses indiquent le cheminement de base, auquel cet article est dédié.

Le fichier `*.tex` est notre point de départ (il est inscrit dans un cercle rouge). À partir de ce fichier (ainsi que de fichiers de style, et de métriques de polices), TEX produira un fichier `*.dvi`. Si on utilise le format LATEX, et si on a pris soin d'inclure l'extension `hyperref`, les commandes de références croisées, citations bibliographiques, sommaire et indexation produiront des liens hypertexte, inclus dans le fichier `*.dvi` à l'aide d'encapsulations `\special`.

Observons qu'un autre point de départ possible est un fichier HTML (en haut à gauche dans notre diagramme). En effet, un fichier HTML peut très facilement être converti en LATEX, et les liens hypertexte contenus dans le premier peuvent être conservés dans le deuxième.

Mais revenons à notre fichier `*.dvi`. On peut l'exploiter directement, en passant par un pilote écran, ou une imprimante non-PostScript. Mais on peut aussi le convertir en PostScript à l'aide du logiciel DVIPS. Une version étendue de DVIPS, du nom de DVIHPS (« DVI vers *Hyper*PostScript ») permet la conservation des liens hypertexte éventuels, contenus dans le fichier `*.dvi`. Une fois le fichier PostScript obtenu, on peut l'imprimer sur imprimante PostScript (ou sur

imprimante non-PostScript en passant par GhostScript), ou le convertir au format PDF, à l'aide du logiciel Adobe Acrobat Distiller. Ce dernier interprétera les liens hypertexte et les inclura dans le document PDF.

Finalement, pour visualiser le document PDF on se sert de Adobe Acrobat Reader, logiciel distribué gratuitement, tournant sur Macintosh, Windows, DOS, Solaris. Ce logiciel nous permet aussi de naviguer dans le(s) document(s), et de l'imprimer sur imprimante non-PostScript.

On s'aperçoit que si le point de départ était un document HTML, toutes les fonctionnalités hypertexte ont été conservées, mais qu'on a gagné en sus la présentation typographique de LaTeX. Un document PDF est une copie fidèle du document imprimé (il peut être photocomposé et produire ainsi un imprimé professionnel avec images couleur, graphiques etc.), offrant en plus la navigation hypertexte, dans le document ou à travers le réseau (la navigation à travers le réseau n'est possible qu'avec la version 2 des logiciels Adobe Acrobat).

1.3. Conclusions

La structuration du document exigée par LaTeX est appliquée à un nombre de plus en plus grand de domaines: l'exemple le plus frappant est sans doute celui du synthétiseur vocal (cf. [5]), à l'usage des non-voyants, capable de prononcer une formule mathématique à la manière d'un mathématicien, et de permettre la navigation dans le document sonore à l'aide de touches spéciales du magnétophone.

Nous décrirons dans cet article une autre application: la création de livres électroniques, dont la présentation n'a rien à envier aux livres traditionnels (puisqu'ils peuvent être imprimés sans perte de qualité) et qui offrent un minimum d'interactivité: navigation hypertexte entre sommaire/index/liste de références bibliographiques et texte, dans le texte même à l'aide des références croisées LaTeX, et finalement navigation entre documents, situés sur la même machine ou sur le réseau.

Dans le reste de cet article nous allons étudier chacune des étapes du processus, indiqué par des flèches grasses sur le diagramme de la fig. 1 (et sur celui de la fig. 2 qui en est une extension).

2. HTML → LaTeX

Le système de balisage HTML est défini suivant la norme SGML (le lecteur trouvera la DTD complète dans [3]). Il s'agit d'un petit nombre de balises, destinées principalement à l'affichage écran. Ainsi on trouve plusieurs styles de texte, aussi bien logiques (emphatique, très emphatique, citation, adresse etc.) que visuels (italiques, gras, souligné etc.), mais aucune balise pour des structures aussi fondamentales que les notes de bas de page, les tableaux etc.

Il va de soi que LaTeX est un système de balisage infiniment plus riche que HTML, et donc la conversion entre les deux formats, dans le sens « HTML → LaTeX » est triviale[2]. En effet, il s'agit de convertir simplement

1. certaines balises en environnement LaTeX, par exemple `<CITE>` en `\begin{quotation}` et `</CITE>` en `\end{quotation}` ;
2. d'autres balises en commandes LaTeX avec argument : `<EM>un mot</EM>` en `\emph{un mot}`, etc. ;
3. quelques rares balises en commandes LaTeX sans argument, comme `<P>` qui peut être converti en une ligne blanche ou en `\par` ;
4. les lettres accentuées en syntaxe LaTeX : `&eacute;` donnera `\'{e}`, `&Ccedilla;` donnera `\c{C}` et ainsi de suite.

Deux cas particuliers se présentent :

1. Les balises HTML qui n'ont pas d'équivalent immédiat LaTeX, par exemple `<STRONG>` qui indique un style fortement emphatique. Dans ce cas, la démarche appropriée serait de définir des nouveaux environnements LaTeX, avec éventuellement le même nom, et de laisser à l'utilisateur le choix de leur représentation visuelle, tout en donnant une représentation par défaut. Ainsi par exemple, `<STRONG>fortement emphatique</STRONG>` pourrait être converti en

   ```
   \begin{STRONG}fortement emphatique\end{STRONG}
   ```

   lequel environnement pourrait être défini comme

   ```
   \newenvironment{STRONG}{\bfseries\itshape}{\/}
   ```

   dans un fichier `html2ltx.sty` et qui produira donc « ***fortement emphatique*** » dans le document LaTeX.

2. Les balises de fonctions hypertexte. Dans ce cas, il convient d'utiliser la syntaxe utilisée par l'extension `hyperref`, que nous utiliserons ensuite (cf. 3) pour inclure dans le fichier DVI précisement ces informations. Voici les commandes explicites :

   (a) pour définir une *cible* (une « ancre » dans le jargon HTML), on délimitera sous HTML le texte choisi par les balises `<A`

2. La conversion dans l'autre sens est décrite dans [2] dans ce même volume. Pour palier l'absence de certaines structures le système LaTeX → HTML de Nikos Drakos convertit en image les tableaux et les formules mathématiques LaTeX. Une méthode peu othodoxe (le fichier HTML obtenu est un mélange de texte et d'images représentant du texte) mais drôlement efficace : tout document LaTeX peut ainsi être converti au format HTML, quelque soit sa complexité... dans le pire des cas le document consistera en une page de titre et une série d'images.

`NAME="mot-cle">` (où `mot-cle` est le mot-clé donné à la cible en question), et `</A>`. La commande LaTeX équivalente sera `\hyperdef{}{mot-cle}{}{...}` où `...` est le texte choisi ;

(b) pour définir un *lien* vers une cible dans le même document, on délimitera sous HTML le texte choisi par les balises `<A HREF="#mot-cle">` (où `mot-cle` est le mot-clé donné à la cible en question), et `</A>`. La commande LaTeX équivalente sera `\hyperref{}{mot-cle}{}{...}` où `...` est le texte choisi ;

(c) pour définir un *lien* vers un autre document, on délimitera sous HTML le texte choisi par les balises `<A HREF="adresse">` (où `adresse` est l'adresse `http` du document en question), et `</A>`. La commande LaTeX équivalente sera `\hyperref{adresse}{}{}{...}` où `...` est le texte choisi.

3. LaTeX → DVI

Disons-le tout de suite : tout document LaTeX 2_ε, sans exception, peut produire un document électronique, par le seul ajout de la ligne

```
\usepackage{hyperref}
```

dans le préambule. Les commandes LaTeX qui vont servir à créer des liens hypertexte sont :

- `\label`, `\ref` et `\pageref` (références croisées) ;
- `\chapter`, `\section`, `\subsection`, etc. (subdivision du document) ;
- `\index` (création d'index) ;
- `\cite` (création de liste de références bibliographiques).

Il n'y a donc rien à changer dans le source du document LaTeX, à moins de vouloir ajouter des nouvelles fonctionalités hypertexte, dans lequel cas on pourra utiliser les commandes internes `\hypertarget` et `\hyperlink`, ou définir des nouvelles commandes, en se servant des commandes « privées » `\hyper@anchor` et `\hyper@link` (cf. [4] pour plus de détails).

3.1. Créer un fichier PDF avec répère

Comme on le voit sur la fig. 3, le logiciel Acrobar Reader nous donne la possibilité d'avoir une table des matières hierarchique et interactive sur la partie gauche de la fenêtre Acrobat. Le logiciel DVIHPS — du moins dans sa version actuelle

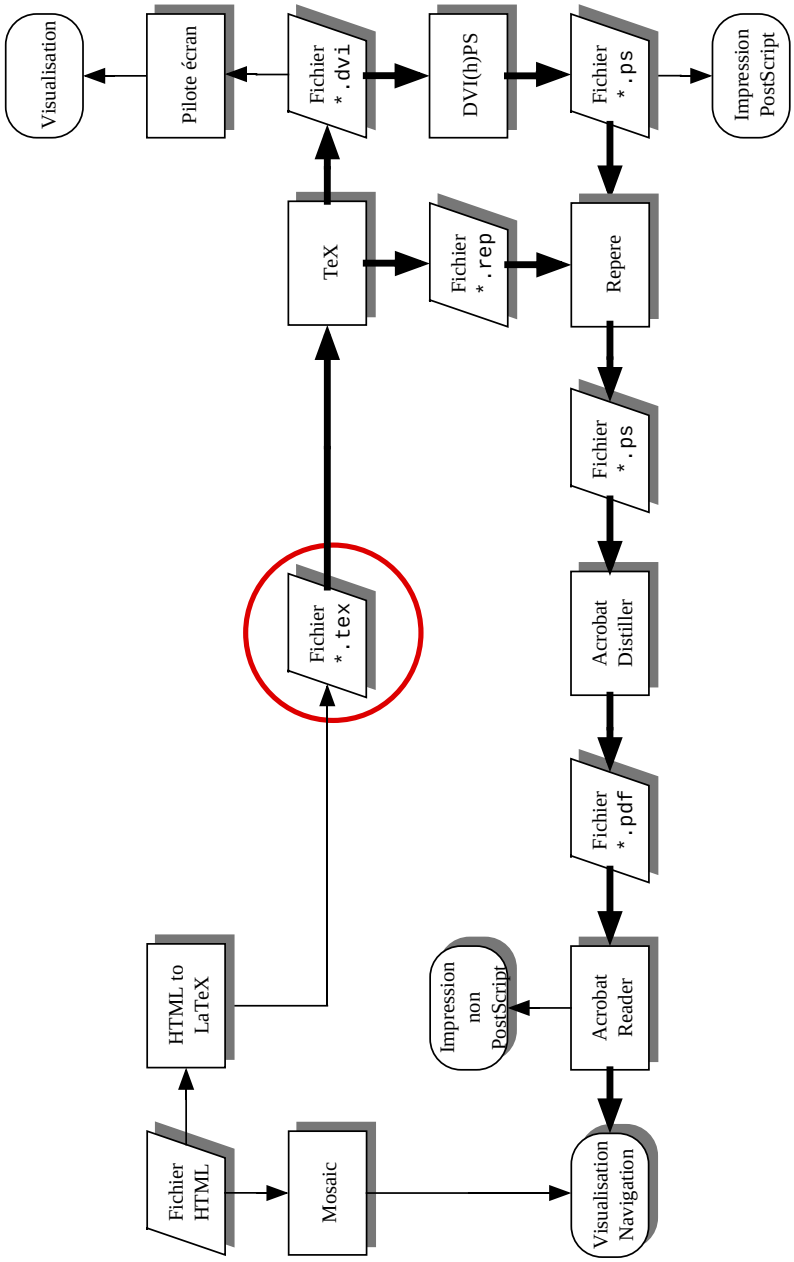

FIGURE 2 - *Schéma général pour obtenir un fichier PDF avec répère*

— ne produit pas cette table de matières automatiquement. Pour pallier cette lacune, l'auteur a développé un utilitaire, du nom de `repere`, et un style LaTeX (`repere.sty`). Pour obtenir un exécutable de `repere` sur sa plate-forme, l'utilisateur lancera d'abord l'utilitaire Flex (version $\geq$ 2.4.6) sur `repere.lex`, avec l'option de ligne de commande `-8` :

```
flex -8 repere.lex
```

et ensuite son compilateur C préféré sur le fichier de sortie de Flex (`lex.yy.c` sous UNIX ou Macintosh, `lexyy.c` sous DOS etc.).

En fig. 2, le lecteur trouvera le schéma général de la fig. 1, auquel on a ajouté le passage par `repere`.

Voici la procédure exacte : pour produire une table de matières PDF,

- on ajoutera la commande `\usepackage{repere}` dans le préambule du document, *après* la commande `\usepackage{hyperref}` ;
- on remplacera les commandes `\part`, `\chapter`, `\section`, `\subsection` et `\subsubsection` qu'on souhaite faire entrer dans la table des matières, par `\Part`, `\Chapter`, `\Section`, `\Subsection` et `\Subsubsection` resp. ;
- LaTeX produira alors un nouveau fichier auxiliaire, d'extension `.rep` ;
- après le passage par DVIHPS, on lancera `repere` avec comme fichier d'entrée, la concaténation

  1. du fichier `foo.rep` créé par LaTeX ;
  2. du fichier `foo.ps` produit par DVIHPS ;
  3. et de nouveau du fichier `foo.rep`.

  Sous UNIX (et tout autre système opérationnel disposant de « tubes ») cela peut se faire très facilement à l'aide d'une ligne de commande du type

  ```
  cat foo.rep foo.ps foo.rep | repere > nouveau-foo.ps
  ```

- le fichier `nouveau-foo.ps` produit par `repere` est une copie exacte du fichier `foo.ps` avec quelques lignes de plus, qui permettront à Acrobat Distiller de créer une table de matières PDF.

La partie la plus délicate de cette opération est la conversion des titres des différentes subdivisions du texte dans le codage PDF. En effet, ce codage est un amalgame des codages Macintosh Standard, Adobe Standard et Windows ANSI. La difficulté provient du fait que LaTeX fait une expansion de commandes lorsqu'il écrit ces chaînes de caractères dans le fichier `*.rep`. Un caractère accentué peut donc être décrit de différentes manières, suivant le codage sortie de LaTeX, les commandes d'accentuation etc. L'utilitaire `repere` pourra difficilement reconnaître

chaque écriture et la remplacer par le caractère 8-bit PDF correspondant. Il se peut donc que l'utilisateur ait à faire quelques corrections dans les titres de subdivisions de la table de matières du fichier PDF. Heureusement, cela n'affecte en rien l'interactivité : les liens hypertexte entre cette table de matières et le texte du document PDF seront conservés.

3.2. Précautions à prendre au niveau TEX

Le fait qu'un fichier DVI servait jusqu'à ajourd'hui presque exclusivement à l'impression nous oblige à prendre certaines précautions lors de la préparation de ce fichier, quand notre intention sera de le convertir ultérieurement au format PDF.

Ces précautions concernent principalement le choix des polices utilisées dans le document. En effet le plus grand problème pour un logiciel comme Acrobat qui prétend pouvoir afficher et imprimer n'importe quel fichier PostScript, est la disponibilité des polices PostScript utilisées dans le document. 99% des polices PostScript existantes (et il y en a des milliers...) sont commerciales, et leur utilisation implique donc une transaction préalable entre l'utilisateur et la société qui en détient les droits. Comment faire alors pour distribuer un document ? comment être sûr que chaque lecteur du document aura à sa disposition les mêmes polices PostScript? que faire dans le cas contraire?

Adobe a résolu ce problème en développant une nouvelle technologie de polices : les polices *Multiple Master*. Le principe est assez proche de METAFONT : il s'agit de meta-polices[3], et en modifiant la chasse de leurs caractères, le pilote écran et impression (Super-ATM) peut simuler des polices PostScript non disponibles. Deux polices Multiple Master accompagnent Acrobat Reader : une police romaine et une sans empattements. Toute police non disponible sera remplacée par une de ces deux polices « génériques ».

Toute police non disponible? oui, toute police non disponible, y compris toute police de symboles mathématiques, ou de caractères spéciaux (phonétiques, non-latins etc.). En utilisant des polices non-standard on peut donc rapidement avoir des résultats catastrophiques.

Pour y remédier, Acrobat Distiller propose deux solutions :

1. l'inclusion de la police PostScript « exotique » dans le fichier PDF. Cela empêche la substitution Multiple Master, mais pose éventuellement des problèmes de copyright ;
2. dans le cas de TEX, l'utilisation de polices PK (non-PostScript). Le problème de copyright ne se pose pas, puisque seuls quelques bitmaps seront inclus dans le fichier PDF, mais la visualisation sous Acrobat Reader en souffre. En effet, Acrobat Reader n'est pas optimisé pour afficher des caractères

3. En fait, leurs « meta-propriétés » sont ridicules, comparées à celles des polices METAFONT.

bitmap, et il faut souvent passer à un agrandissement élévé pour arriver à les déchiffrer. Par contre, l'impression se fait sans problème (puisqu'en fin de compte la résolution des caractères bitmap inclus dans le fichier correspond à la résolution de l'imprimante, plutôt qu'à celle de l'écran).

Une troisième solution consiste à éviter le problème, en n'utilisant que des polices « universelles », c'est-à-dire invariablement présentes sous tout système opérationnel : ainsi on n'est pas amené à inclure des polices dans le document (et donc les problèmes de copyright ne se posent pas) mais on garde pour autant la visualisation PostScript, qui est nettement meilleure que la visualisation de caractères bitmap. Ces polices sont Times, Helvetica, Courier, Symbol, Palatino, New Century Schoolbook et Bookman.

Enfin, une ultime précaution : il faut éviter d'utiliser des polices *virtuelles utilisant des assemblages de signes*. Autrement dit, il faut passer par des re-encodages de polices « à la DVIPS », plutôt que par des constructions de caractères accentués par la combinaison de la lettre et de l'accent, comme le fait `fontinst.sty`. La raison en est qu'Acrobat permet la recherche de chaînes de caractères dans un document : si les caractères accentués sont en réalité des assemblages de lettres non-accentuées et d'accents, on ne pourra pas faire une recherche utilisant des caractères acentués 8-bit. Le mot « dégénéré », par exemple, sera codé comme

```
de<accent aigu>ge<accent aigu>ne<accent aigu>re<accent aigu>
```

dans le document PDF, et une recherche de la chaîne `dégénéré` (où « `é` » est un caractère 8-bit Macintosh ou Windows ou ISO Latin-1), ne pourra qu'échouer.

Pour obtenir des polices virtuelles *qui se limitent à re-encoder les caractères*, on pourra utiliser l'utilitaire `afm2tfm` de la distribution DVIPS, avec les options de ligne de commande `-T extex.enc` (codage de Cork) et `-v` suivi du nom de fichier VPL souhaité. Dans le fichier de configuration `psfonts.map` de DVIPS, il faut alors ajouter les commandes

```
" ExtendedTeXEncoding ReEncodeFont " < extex.enc
```

aux définitions des polices en question (en supposant que le fichier `extex.enc` se trouve dans le même repértoire que `psfonts.map`, sinon il faut remplacer `extex.enc` par son chemin d'accès complet).

Voici un exemple : l'auteur a obtenu la police réelle `rtimes.tfm` et la police virtuelle `vtimes.vf` (avec fichier métrique `vtimes.tfm`) à partir du fichier AFM `Times-Roman.AFM`, à l'aide des commandes suivantes :

```
afm2tfm Times-Roman.afm -T extex.enc -v vtimes.vpl rtimes.tfm
vftovp vtimes.vpl -o vtimes.vf
```

Pour l'utiliser il a ajouté la ligne

```
rtimes Times-Roman " ExtendedTeXEncoding ReEncodeFont " <extex.enc
```

au fichier `psfonts.map`, et il a créé un fichier `T1vtime.fd` contenant, entre autres, les lignes

```
\DeclareFontFamily{T1}{vtime}{}
\DeclareFontShape{T1}{vtime}{m}{n}{
   <->vtimes
}{}
\endinput
```

Pour utiliser cette nouvelle famille de polices il a suffi d'inclure les lignes

```
\usepackage{t1enc}
\def\rmdefault{vtime}
```

dans le préambule du fichier LaTeX.

Pour les formules mathématiques on peut utiliser les polices Times et Symbol: plusieurs paquets de macros/polices virtuelles permettent de le faire, entre autres aussi *MathTimes* de Michael Spivak.

4. DVI → (Hyper)PostScript

Comme TeX, DVIPS est aussi un exemple typique de logiciel domaine public de très haute qualité. Étant unanimement reconnu comme le meilleur traducteur DVI vers PostScript, DVIPS va très probablement devenir la norme dans ce domaine. D'ailleurs, de plus en plus de systèmes TeX proposent au même temps un pilote écran (ou imprimante non-PostScript) spécifique au système opérationnel sous-jacent, et une implémentation de DVIPS pour l'impression PostScript.

Sebastian Rahtz a modifié le logiciel DVIPS (de Tom Rokicki) pour lui permettre de gérer des liens hypertexte. Une telle intervention « chirurgicale » était-elle nécessaire? Il s'avère que oui, et nous allons voir pourquoi.

Pour définir un lien hypertexte, Acrobat Distiller nécessite (au moins) deux informations: la zone active de départ, et l'écran d'arrivée du lien. Ces informations doivent lui être commuiquées sous forme de coordonnées PostScript (72èmes du pouce à partir de l'origine des coordonnées PostScript, en bas à gauche de la feuille).

Pour déterminer les coordonnées exactes de la cible d'un lien hypertexte, qui peut se situer plusieurs pages *après* la zone active de départ, il *faut* post-processer le fichier DVI: il serait, sinon impossible, du moins horriblement compliqué, d'obtenir ces coordonnées au niveau LaTeX. Et puisqu'on nécessite un post-processeur DVI, autant prendre directement DVIPS, qui travaille en coordonnées PostScript, et tient compte de tous les changements d'échelle au de translation de l'origine des coordonnées qui peuvent intervenir.

Ainsi, Sebastian Rahtz a obtenu une version « ravalée » de DVIPS, qu'il appelle DVIHPS (« DVI vers *Hyper*PostScript »). L'auteur espère que ces modifications feront bientôt partie du DVIPS standard.

Au plan pratique, DVIHPS est utilisé de la même manière que DVIPS, à une exception près : si l'utilisateur désire produire un fichier PostScript avec les informations hypertexte destinées à Acrobat Distiller, il devra utiliser l'option de ligne de commande supplémentaire `-z` (z *minuscule*).

À noter que le préambule du fichier PostScript produit contiendra du code PostScript qui désactivera toutes les commandes hypertexte, si jamais le fichier est interprêté par un autre interprêteur qu'Acrobat Distiller. Ce même code vérifiera également la version d'Acrobat Distiller (par exemple, la version 1 de ce logiciel ne permet pas de liens entre documents distincts : seuls ces liens seront désactivés lors de l'interprétation).

5. PostScript → PDF

Cette étape, qui est certainement la plus longue du point de vue temps d'exécution, est entièrement prise en charge par le logiciel Adobe Acrobat Distiller (un logiciel assez onéreux, mais malheureusement indispensable pour produire des document PDF solides). Acrobat Distiller est au même temps un très bon débogueur de code PostScript, et un bon interpréteur : n'ayant ni imprimante PostScript couleur à sa disposition, ni le système Display PostSript d'épreuvage PostScript à l'écran, l'auteur utilise Acrobat Distiller pour visualiser des documents TeX coloriés.

6. Prévisualisation, navigation et impression du fichier PDF

Ces opérations se font à l'aide du logiciel Adobe Acrobat Reader, distribué gratuitement. Plusieurs fonctions de recherche, navigation, copie de texte[4], etc. sont proposées à l'utilisateur, à l'aide de « boutons » et de menus. Notons que ce logiciel tourne sur Macintosh ainsi que sous DOS, Windows et UNIX (Solaris et bientôt d'autres systèmes), avec exactement la même interface utilisateur.

Pour comparer l'affichage écran d'un fichier TeX, l'utilisateur trouvera dans la fig. 3 une copie d'écran Acrobat Reader sur Macintosh, et dans la fig. 4 la version imprimée du même document. L'extension LaTeX `hyperref` permet à l'utilisateur de choisir la présentation des zones actives de lien hypertexte (par défaut en rouge) ainsi que des zones-cible (par défaut en vert). Le format PDF permet en outre d'encadrer les zones actives.

4. Et ce sera la première fois qu'on pourra faire des copies de texte à partir d'un dérivé de fichier DVI.

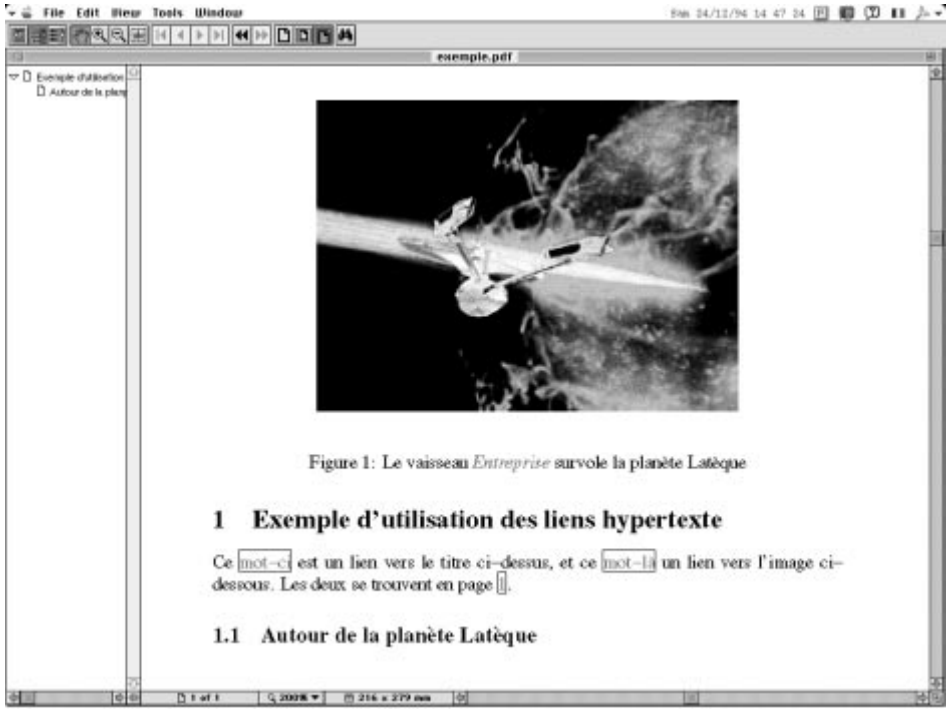

FIGURE 3 - *Copie d'écran Macintosh : un exemple de fichier PDF créé sous LaTeX*

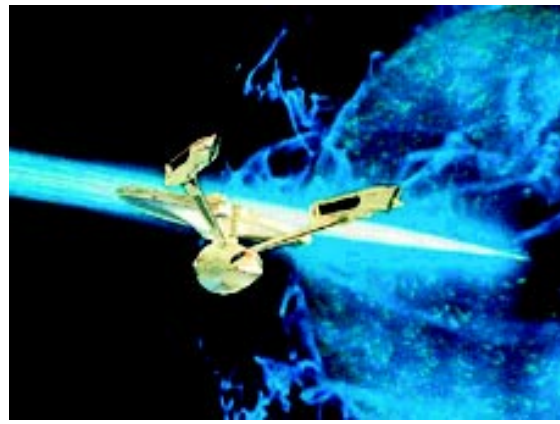

Figure 1: Le vaisseau *Entreprise* survole la planète Latèque

1 Exemple d'utilisation des liens hypertexte

Ce mot-ci est un lien vers le titre ci-dessus, et ce mot-là un lien vers l'image ci-dessous. Les deux se trouvent en page 1.

1.1 Autour de la planète Latèque

FIGURE 4 - *Résultat de l'impression du fichier PDF*

6.1. Quelques informations sur le format PDF

Le format PDF est encore plus hermétique et indéchiffrable pour le commun des mortels que le langage PostScript lui-même. Néanmoins il est intéressant de connaître un peu sa structure, pour effectuer, le cas échéant, quelques infimes modifications à la présentation du fichier (le format PDF est encore très récent et on manque cruellement d'utilitaires pour modifier des documents PDF).

Un fichier PDF est un fichier texte, entièrement écrit en 7-bit. Il consiste en quatre parties : le préambule (*header*), le corps (*body*), la table de références croiées (*cross-reference table*) et l'épilogue (*trailer*). Le préambule, pour la version 1, ne comporte qu'une seule ligne : `%PDF-1.0`. Le corps, est composé d'*objets* : chaque page est un objet, les liens, les notes, les entrées de repère, les codages de police, les descripteurs de polices, les systèmes de description de couleur sont également des objets. L'intérêt d'utiliser des objets vient du fait qu'on peut ainsi changer l'ordre, insérer ou retirer des pages, sans briser les liens hypertexte existants : l'ordre des pages est gardé dans la table des références croisées, les pages « retirées » restent en fait dans le document, et ne sont donc que virtuellement retirées. Chaque modification va entraîner la création d'une nouvelle table de références croisées dans l'épilogue du document. Les applications de visualisation PDF commencent donc la lecture du document par la fin, et récupèrent dans la table de références croisées des pointeurs vers les différents objets du document.

La plupart des objets sont compressés et puis codés en 7-bit : quatre méthodes de compression peuvent être utilisées : Lempel-Ziv, *run length*, CCITT Fax groupe 3 ou 4, JPEG ; ensuite deux méthodes peuvent être utilisées pour le passage à 7-bit : la notation héxadecimale ou la notation « modulo 85 ».

Adobe Acrobat Distiller permet de desactiver la compression, mais cela ne présente que peu d'intérêt parce qu'aucun utilitaire n'est prévu pour permettre la compression *a posteriori* des objets PDF modifiés.

Nous décrirons alors uniquement quelques objets non compressés, qui peuvent par conséquence être librement modifiés par l'utilisateur. Néanmoins il faut noter que toute modification du fichier PDF (à part une, que nous verrons plus bas) nécessite impérativement une mise à jour de la table de références croisées : en effet, cette table contient pour chaque objet PDF, son décalage en octets par rapport au début du fichier. Chaque objet a un numéro, c'est d'ailleurs la première donnée d'un objet. Les objets ne sont pas forcement disposés par ordre croissant de numéro dans le fichier PDF. La table de références croisées contient une ligne pour chaque objet ; cette ligne consiste en le décalage en octets de l'objet par rapport au début du fichier (un nombre à 10 chiffres), suivi d'un blanc, d'un nombre à 5 chiffres qui dénote le nombre de fois ce numéro d'objet a été modifié, encore un blanc, et la lettre « n ». Si l'objet est effacé, le numéro d'objet est donc disponible et cette ligne change de syntaxe : le nombre à dix chiffres indique le numéro du prochain objet libre dans la table (il est nul s'il s'agit du dernier objet libre) et la lettre en fin de ligne est « f ».

Pour chaque modification effectuée à un objet, il faut reporter le décalage résultant à tous les objets qui suivent physiquement dans la disposition du fichier. Également il faut changer un nombre qui se trouve à la fin du fichier et qui indique le décalage de la table de références croisées à partir du début du fichier.

Un exemple : le fichier PDF de la fig. 3 comporte 20 objets ; en voici les deux derniers, la table de référence croisées et l'épilogue :

```
1 0 obj
<<
/CreationDate (Dim 25 Déc 1994)
/Producer (Acrobat Distiller 1.0 pour Macintosh)
>>
endobj
3 0 obj
<<
/Pages 5 0 R
/Outlines 16 0 R
/Type /Catalog
>>
endobj
xref
0 21
0000000000 65535 f
0000043515 00000 n
0000042654 00000 n
0000043617 00000 n
0000040662 00000 n
0000042693 00000 n
0000000010 00000 n
0000039522 00000 n
0000040847 00000 n
0000042235 00000 n
0000042360 00000 n
0000042487 00000 n
0000042777 00000 n
0000042897 00000 n
0000043017 00000 n
0000043211 00000 n
0000043137 00000 n
0000043373 00000 n
0000039543 00000 n
0000040641 00000 n
0000042612 00000 n
trailer
<<
/Size 21
/Root 3 0 R
/Info 1 0 R
>>
```

```
startxref
43683
%%EOF
```

Le nombre 43683 indique le décalage de la table de réf. croisées à partir du début du fichier.

Nous allons modifier l'objet 1 : tout d'abord nous allons corriger une erreur d'Acrobat Distiller sous système opérationnel Macintosh français : la date comporte des caractères 8-bit (le « é » de « Déc ») que Distiller a omis de convertir au codage PDF. La boîte de dialogue destinée à afficher certaines informations, parmi lesquelles la date de création du document, affichera un caractère erroné à la place du « é ». Il faut remplacer ce caractère 8-bit par le code du « é » dans le codage PDF : en notation PDF octale, `\351`. On va donc remplacer un octet (é) par 4 octets (`\351`) : le décalage sera de +3.

Il s'agit de l'objet 1, et le seul objet qui le suit dans le fichier est l'objet 3. Il faut donc changer la troisième ligne de la table de références croisées (on commence à compter par 0 : la ligne

```
0000000000 65535 f
```

est en fait la zéro-ième. La première ligne de la table est

```
0000043515 00000 n
```

elle correspond à l'objet contenant les informations. La troisième ligne est

```
0000043617 00000 n
```

et elle correspond au seul objet se trouvant après l'objet que nous allons modifier (effectivement le nombre 43617 est le plus grand dans la table, il s'agit donc bien du dernier objet dans le fichier). On ajoutera 3 au nombre 43617, ainsi qu'au nombre 43683 à la fin du fichier (la table de réf. croisées s'est aussi trouvée décalée de trois octets). Le fichier devient donc :

```
1 0 obj
<<
/CreationDate (Dim 25 D\351c 1994)
...
xref
0 21
0000000000 65535 f
0000043515 00000 n
0000042654 00000 n
0000043620 00000 n
...
startxref
43686
%%EOF
```

De même, pour ajouter quelques informations supplémentaires à l'objet 1, on comptera le décalage final et corrigera la table de réf. croisées et son pointeur en conséquence :

```
1 0 obj
<<
/CreationDate (Dim 25 D\351c 1994)
/Producer (Acrobat Distiller 1.0 pour Macintosh)
/Author (Yannis Haralambous)
/Title (Exemple de fichier PDF)
/Subject (Conversion HyperLaTeX vers PDF)
/Creator (LaTeX, DVIHPS, et autres...)
>>
endobj
...
xref
0 21
0000000000 65535 f
0000043515 00000 n
0000042654 00000 n
0000043762 00000 n
...
startxref
43828
%%EOF
```

Pour automatiser l'opération de réctification de la table de références croisées, l'auteur a développé un utilitaire flex, du nom de `recticrt`. Pour obtenir un exécutable de `recticrt` sur sa plate-forme, l'utilisateur lancera d'abord l'utilitaire Flex (version $\geq$ 2.4.6) sur `recticrt.lex`, avec l'option de ligne de commande `-8` :

```
flex -8 recticrt.lex
```

et ensuite son compilateur C préféré sur le fichier de sortie de Flex (`lex.yy.c` sous UNIX ou Macintosh, `lexyy.c` sous DOS etc.). L'utilisation de `recticrt` est très simple :

`resticrt <` *ancien fichier* `>` *nouveau fichier*

Si la table de références croisées du fichier est correcte, `recticrt` la laissera intacte.

Une autre modification très courante consiste à rendre invisible le filet encadrant les zones actives de liens hypertexte. Chaque lien hypertexte est défini par un objet PDF du type

```
12 0 obj
<<
```

```
/Type /Annot
/Subtype /Link
/Rect [ 148 399 177 411 ]
/Border [ 1 1 1 ]
/Dest [ 4 0 R /FitH 607 ]
>>
endobj
```

Les commandes `/Rect` et `/Dest` déterminent les coordonnées des zones actives de départ et d'arrivée. Les paramètres de `/Border` sont

1. le rayon horizontal du coin du filet ;
2. le rayon vertical du coin du filet ;
3. l'épaisseur du trait.

Les valeurs par défaut sont `1␣1␣1`. En remplaçant le troisième paramètre par `0`, on rend le filet invisible. Si on fait attention à ne pas inclure ou effacer des blancs, le décalage d'octets n'est pas modifié. Il s'agit donc d'une modification qui n'entraîne pas de mise à jour de la table de références croisées et de son pointeur.

Le lecteur trouvera la description complète du format PDF dans [1].

7. Disponibilités

Adobe Acrobat Reader est disponible sur plusieurs serveurs publics. Adobe Acrobat Distiller peut être procuré par Adobe Systems Inc.

DVIHPS (la version modifiée de DVIPS) est — à la date de redaction de cet article — encore sous β-test. Sa disponibilité (ou l'intégration de ses fonctionalités à DVIPS, qui contrairement à TEX et METAFONT n'est absolument pas figé) seront annoncés par les voies habituelles.

L'utilitaire `repere`, le style LATEX correspondant et l'utilitaire `recticrt` peuvent être récupérés par le serveur public `ftp.ens.fr`, ils sont dans le répertoire `/pub/tex/yannis/acrobat`. Le lecteur y trouvera également les exemples décrits dans cet article, ainsi que l'article lui-même (en Times, pour les raisons expliquées en 3.2) sous forme `.tex`, `.aux`, `.bbl`, `.rep`, `.dvi`, `.ps` et `.pdf`.

Références bibliographiques

[1] Adobe Systems Inc, *Portable Document Format Reference Manual*. Addison Wesley, 1 edition, 1993.

[2] M. Goossens, « LaTeX – HTML aller et retour », *Cahiers GUTenberg* n° 19, janvier 1995 (ce cahier), 98–120.

[3] M. Goossens, « Introduction pratique à SGML », *Cahiers GUTenberg* n° 19, janvier 1995 (ce cahier), 27–58.

[4] S. Rahtz, Hypertext marks in LaTeX. 1994.

[5] T.V. Raman, An audio view of TeX documents. In *13th Annual TeX Users Group Meeting, Portland, 1992*, 1992.