METAPOST et le problème de Truchet

Denis Roegel

5 juillet 1995

Le problème de Truchet devait être résolu soit par l'emploi de macros `PSTricks`, soit par une solution *PostScript* directe. Je n'ai suivi aucune de ces voies, non pas parce qu'elles m'apparaissaient difficiles, mais parce que je ne pouvais pas m'investir dans ces langages. J'ai préféré profiter de cet exercice pour perfectionner ma connaissance de METAPOST. Je crois qu'il vaut mieux maîtriser un langage unique, plutôt que d'en connaître beaucoup superficiellement.

METAPOST[1] est un langage graphique dérivé de METAFONT et partage donc un certain nombre de caractéristiques avec `PSTricks`/TEX. C'est un *macrolangage*—tout comme TEX—et un programme METAPOST génère des fichiers *PostScript* pouvant être réutilisés par TEX ou éventuellement un autre logiciel. `PSTricks` génère aussi du *PostScript* mais les commandes y sont directement écrites dans le fichier `.dvi`. Parmi les acquis de METAFONT, on trouve en particulier la structure déclarative : il est possible de spécifier les coordonnées des points clés par des ensembles d'équations linéaires. Avec un peu d'habitude, cette structure déclarative devient très naturelle et accroît notablement la souplesse des figures produites.

Ma solution au problème de Truchet n'est pas un exercice METAPOST typique, en ce sens qu'elle n'utilise pas ces potentialités déclaratives. J'ai plutôt tiré profit des caractéristiques suivantes :

- la possibilité de déclarer des tableaux de dimension quelconque, dont le nombre d'éléments n'est pas connu à l'avance ; par exemple `numeric tilecolor[][][]` déclare un tableau de nombres (entiers ou réels) qui pourront être accédés par trois index ; ceci n'est qu'un exemple, car en général, le nom d'une variable de tableau est composé de *tags* et de *suffixes*, un peu comme les paramètres d'une macro TEX peuvent être séparés par des délimiteurs. . . ;
- la présence des structures de contrôle usuelles : `for`, `forever`, `exitif`, `endfor` ;
- la génération de nombres aléatoires (distributions uniforme et normale) ;
- le tracé de courbes de Bézier passant par certains points et soumis à des contraintes ; les courbes sont des objets de type `path`, elles peuvent subir diverses transformations (translations, rotations, déformations) et être coloriées, par défaut dans le système RGB ;
- le remplissage de zones délimitées par une courbe de type `path` ;
- le découpage en *macros* pour l'organisation des fonctionnalités et en particulier la possibilité d'évaluer ou non les paramètres d'une commande (type `expr` ou `text`).

Dans la solution que je propose, un pavage consiste en un tableau de pavés et pour chacun d'eux, la donnée de son orientation et la coloration de ses lignes et parties. Les algorithmes employés sont les suivants :

lignes leur coloriage repose sur une opération plus simple, le *suivi* d'une ligne : une ligne encore non coloriée est choisie et parcourue d'un bout à l'autre ; on change de couleur à chaque nouvelle ligne.

surfaces leur coloriage repose sur l'observation que deux couleurs suffisent à colorier tout pavage du type considéré ; l'algorithme consiste à colorier un premier pavé et à diffuser ses couleurs sur ses voisins, jusqu'à avoir rempli tout le pavage ; le remplissage est donc réalisé pavé par pavé au sein même de METAPOST ; il eut été possible de définir une macro METAPOST prenant en argument un point d'un pavage, construisant la courbe bordant la région sélectionnée et remplissant alors cette région avec les commandes de METAPOST ; une telle macro peut s'avérer intéressante dans un cas plus général, mais pas pour un problème aussi simple et abordé avec une structure de tableau comme j'ai choisi de le faire.

---

1. METAPOST a été développé par John D. Hobby, ATT Bell Laboratories, USA. Le manuel et d'autres documents sont disponibles sur CTAN ainsi que sur `http://www.loria.fr/tex`.

METAPOST offre de très nombreuses autres possibilités, en particulier l'intégration de portions de texte dans les figures grâce à l'exécution de sous-processus TEX. Les sorties générées sont des fichiers *PostScript*, pouvant être inclus directement par `\includegraphics` dans un fichier LATEX, sans que des calculs complexes doivent encore être faits par TEX.