Théorie des nombres et ordinateurs

par Henri Cohen

Le texte qui suit ~~reproduit~~ est la transcription d'une conférence faite en octobre 1983 aux élèves de première année de l'E.N.S. Ulm. Son but était de situer et d'expliquer un projet informatique avant la réalisation de ce projet. ~~Un~~ résultat (un peu inattendu pour l'auteur) a été la proposition de quatre élèves de collaborer à ce projet.

Je suis un mathématicien particulièrement intéressé par la théorie des nombres (ou arithmétique), l'une des deux plus vieilles branches des mathématiques avec la géométrie. Je voudrais tout d'abord très subjectivement et de façon très chauvine expliquer pourquoi ce sujet m'attire. Il y a bien entendu l'aspect très simple des problèmes que l'on y rencontre, explicables pour la plupart à des élèves du secondaire, alors que les moyens mis en œuvre pour les résoudre (dans les rares cas où on y arrive !) sont sophistiqués et puisent dans presque toutes les branches des mathématiques.

Mais ~~plus~~ mon sentiment profond est que la théorie des nombres est une véritable science de la nature au même titre, disons, que la physique : les beautés et les structures que l'on y découvre préexistent. Cette analogie avec la physique peut être d'ailleurs poussée très loin. Il y a d'abord une

partie expérimentale, permettant de faire des conjectures plausibles. Il y a ensuite la partie théorique consistant à démontrer lesdites conjectures. Pour cela on doit souvent utiliser, sinon créer, des branches des mathématiques assez éloignées. De même en physique, certains problèmes on nécessité la création de nouveaux outils mathématiques.

Avec l'avènement de l'ère informatique, la théorie des nombres a reçu un outil qui permet à la dialectique entre théorie et expérimentation d'être beaucoup plus efficace. Une utilisation typique peut être la suivante : 1) Obtenir suffisamment de données expérimentales sur lesquelles on puisse faire des conjectures 2) Vérification de ces conjectures par des passages ~~en machine~~ extensifs en machine (parfois plusieurs semaines de temps machine !)
3) Parfois aussi aide aux démonstrations elles mêmes, soit en complétant une démonstration (si on a démontré un énoncé pour $n \geq 10^6$ il "suffit" ensuite de le passer en machine pour $n < 10^6$), soit en guidant la démonstration.

On voit que de plus en plus l'ordinateur joue en théorie des nombres un rôle analogue à celui des petites ou grosses machines utilisées par les physiciens expérimentaux. Toutefois il faut faire un constat de pauvreté : les physiciens peuvent se payer des appareils coûtant

trois cent milliards de centimes (le tunnel sous le Jura du CERN) alors que les arithméticiens (et les mathématiciens purs en général) n'ont que de faibles moyens: typiquement quelques microordinateurs de valeur inférieure à 70 000 francs et des accès sur des gros centres où, bien que le coût de l'heure de calcul soit modéré, il est impossible de faire les passages extensifs mentionnés ci-dessus.

Il n'est difficile d'expliquer cette extravagance de crédits accordés aux physiciens (j'en suis personnellement fort content pour eux, d'autant plus que la physique théorique me passionne) mais je voudrais souligner qu'il y a exactement autant d'intérêt pour la Science avec un grand S (c'est à dire beaucoup de detecter des particules W que de découvrir (grâce à de nombreux essais expérimentaux) puis à démontrer un lien entre la dérivée au point 1 de fonctions L de courbes elliptiques et la hauteur de certains points de Heegner [ ]. Je veux dire par là que ces résultats (W ou L) n'ont aucun intérêt pratique ni même envisageable, mais que par contre ils dévoilent des structures naturelles très belles qui enrichissent notre compréhension (avec un grand C) du monde

Je voudrais revenir maintenant sur le matériel informatique disponible par les arithméticiens.

- D'une part les gros centres ont l'avantages: ils ont énormément de logiciels, les machines sont en général rapides avec une grosse place mémoire.

Défauts : prix élevé d'utilisation, accès encombré.

- D'autre part les microordinateurs. Avantages : prix négligeable, accès immédiat, possibilité de "bricoler".

Défauts : souvent peu de logiciels (scientifiques en particulier), machines lentes et peu de place mémoire.

On voit donc que les deux types sont en gros complémentaires.

Evolution actuelle et prévisible.

Depuis deux ans sont apparus sur le marché des machines à base de microprocesseurs très performants, les trois principaux étant le MC 68000, NS 16032, Z8000, et depuis peu le couple 8086-8087, où le 8087 sert de processeur arithmétique rapide.

En gros ils travaillent à 10^6 instructions par seconde, ont des registres de 32 bits mais manipulent le plus souvent des données de 16 bits (c'est pourquoi on les appelle des 16/32 bits); leurs instructions internes sont en petit nombre et très puissantes, et en particulier sont bien adaptés aux langages structurés de type PASCAL.

La génération suivante commence déjà à apparaître sur le marché et sera en pleine expansion en 1985. Ils ont pour noms MC68020, NS32032, Z80000, sont 4 à 8 fois plus rapides, travaillent entièrement sur 32 bits et ont des attributs qui n'existaient auparavant que sur les grosses machines, telles que mémoire cache, mémoire virtuelle, coprocesseurs arithmétiques. La taille de la mémoire devient sans

limite réelle, et son prix est en chute libre. Tout ceci fait que les gros ordinateurs sont rapidement entrain de perdre une grande partie de leur intérêt. De plus avec des architectures dites parallèles, on peut atteindre des vitesses beaucoup plus grandes pour un prix beaucoup plus bas ~~faible~~ non exhorbitant ?

La conclusion que je tire de ceci est que l'avenir appartient aux microordinateurs, sauf pour quelques applications précises où une grosse centralisation est nécessaire, telle que par exemple la gestion d'une base de données. Toutefois le problème qui se pose avec la plus grande acuité est celui du logiciel : la trop grande variété de matériel existant est un obstacle à son développement, et favorise des ordinateurs très répandus tels que l'Apple II et l'IBM PC.

Enfin le marché des 16/32 bits pose un problème particulier : les constructeurs étant persuadés (à mon avis à tort) qu'il ne se développera que lentement, on ne trouve pratiquement que du très haut de gamme, avec des logiciels sophistiqués (par exemple Lisa d'Apple, basé sur 68000) mais destinés à la gestion donc à 90% inutile pour un scientifique, et beaucoup trop chers. Il commence à y en avoir à prix presque raisonnable (40000F) comme le M68 de SORD. Une autre solution est d'acheter un ordinateur monocarte (7000F) mais qui ne dispose que d'un logiciel réduit (éditeur écran, assembleur, basic). La situation est en pleine évolution et à suivre.

et alors ?

Problèmes spécifiques à la théorie des nombres -

Tout d'abord il est nécessaire de pouvoir manipuler des entiers ou des réels avec beaucoup de chiffres (typiquement 20 ou 30), non pas pour le plaisir, mais à cause du fait que la plupart des expériences de théorie des nombres nécessitent d'aller jusque là pour avoir assez de données, et également parceque certains algorithmes utilisent des grands nombres dans des calculs intermédiaires -

D'autre part on a en permanence besoin de tables de nombres premiers ou de tests de primalité, de méthodes de factorisation, de fonctions arithmétiques particulières. Dans certaines applications on doit également manipuler des polynomes ou des séries entières -

Surement quelque chose de ce genre doit exister ? La réponse est oui, MAIS

1°) Uniquement sur de grosses machines (ce n'est pas fondamental, voir ci dessus)

2°) La plupart du temps ~~des~~ tels ~~langages~~ sont très lents, car interprétés, ce qui est catastrophique en théorie des nombres

3°) Souvent malcommodes d'emploi, et ne contiennent pas les besoin spécifiques des arithméticiens -

Exemples typiques : MACSYMA, REDUCE (interprétés donc lents, mais commodes), SAC2 (compilé donc nettement plus rapide, mais très malcommode).

C'est ce qu'on appelle des systèmes de calcul formel, et ils sont fort justement très appréciés des

physiciens.

Notre projet.

Il consiste à essayer de réaliser l'ambitieux program ci-dessus sur un micro ordinateur en diminuant au maximum les inconvénients énoncés.

Principes : — Tout d'abord, il ne s'agit pas d'écrire un système de calcul formel (ce qui est d'ailleurs un travail énorme) mais seulement la partie dont on a besoin en théorie des nombres (et qui inclut des choses que les systèmes de calcul formel classiques ne possèdent pas)

— Il y aura un grosse bibliothèque scientifique écrite principalement en C (le langage des systèmes UNIX et CP/M 68K) ou/et en langage assembleur 68000. Ceci aura pour conséquence : 1°) Une très grande rapidité ; 2°) (Transportabilité ?, ca le langage C (ou dans une moindre mesure le 6800 est très répandu 3°) Possibilité d'améliorations avec peu de travail supplémentaire quand la nouvelle génération des vrais 32 bits sera à un prix accessib (exemple le 68020 comprend tout le code 68000 mais est beaucoup plus rapide).

— Enfin et surtout il y aura un langage spécifique pour utiliser la puissance de la bibliothèque, et devant satisfaire à deux critères 1°) Il doit être compilé (bien qu'une version interprétée soit également désirable, la communicat

homme-machine étant importante dans la phase expérimentale d'une recherche en théorie des nombres)

2°) Il doit être très proche, et même si possible contenir, un langage déjà existant pour que son apprentissage soit facile. Comme candidats je vois essentiellement Pascal, Ada, C et Modula 2. J'ai écarté C car ce n'est pas un langage vraiment adaptable à l'utilisation de types scientifiques très différents, et Modula 2 car bien que très intéressant, il est pour l'instant trop peu répandu. Enfin Ada est un monstre également fascinant et bien adapté, mais j'attendrai encore quelques années avant de le voir sur un microordinateur... Donc malgré tous ses défauts bien connus, j'ai choisi Pascal.

Par contre le compilateur du langage sera écrit en C, qui est parfaitement adapté à cette tâche. Plus précisément dans la version actuelle du projet nous envisageons d'écrire un préprocesseur en C qui transforme notre langage en du vrai Pascal, puis bien sûr une compilation Pascal ordinaire terminera le travail.

Détail de la bibliothèque et du langage

-- Tout d'abord il y aura un grand nombre de types de base incluant des entiers ou réels de précision arbitraire (mais déclarée), des rationnels, des nombres complexes, des vecteurs, matrices, polynomes et développements limités.

- Sur ces types un grand nombre d'opérations et de fonctions seront prédéfinies avec bien sûr mélange le plus large possible des types.
- On pourra (comme en Pascal) créer aisément de nouveaux types, mais aussi si possible créer des opérations nouvelles entre types.
- Les fonctions arithmétiques usuelles (y compris la factorisation) seront disponibles immédiatement.

Conclusion et remarques générales

Je ne sais pas si j'arriverai à mener ce projet à son terme. Bien que longue, l'écriture de la bibliothèque scientifique ne me pose pas de problèmes. Par contre, n'étant pas informaticien l'écriture du compilateur (ou plutôt du préprocesseur) m'effraye passablement. Des collègues m'affirment que les utilitaires Unix Lex et Yacc feront presque tout le travail pour moi; ~~enfin~~ verra bien... Il est bien évident que toute personne qui veut m'aider soit par ses conseils soit en participant effectivement au projet, est la bienvenue. Il faut noter que les industriels ne s'intéressent absolument pas à ce genre de choses (et d'ailleurs s'intéressent très peu à ce qui n'est pas application de type gestion).
D'autre part les informaticiens ont leur propre problèmes et bien que prêts à m'aider ponctuellement ne désirent pas participer au projet. Resteraient enfin des étudiants de 3e cycle : mais vu le salaire

rémunérant qu'ils trouveront après dans le privé, ils préfèrent presque tous ~~s'occuper de problèmes qui les intéressent~~ se former à des techniques qui leur serviront dans leur carrière.

En résumé, pour l'instant on est obligé de faire tout soi-même (toutefois voir entête). Lié à ce constat est le fait regrettable à mon avis que beaucoup d'informaticiens français (à l'exception notable des théoriciens et en particulier de l'École de Schützenberger) ont une formation mathématique plus qu'insuffisante. Je pense qu'il est nécessaire que soient formés des gens ayant une haute qualification à la fois en mathématiques et en informatique, et les élèves des écoles normales supérieures sont typiquement parmi ceux qui pourraient aisément atteindre cette qualification.

?

Lié à ce constat est le fait regrettable que beaucoup d'informaticiens français (à l'exception notable des théoriciens et en particulier de l'École de Schützenberger) ont une formation mathématique plus qu'insuffisante.