Prenons la main de la mémoire. Ma découverte de l'informatique (1964-1973) - Témoignage

Pierre Lescanne

Prenons la main de la mémoire. Ma découverte de l'informatique (1964–1973)

Pierre Lescanne
Professeur émérite à l'École normale supérieure de Lyon, LIP (UMR 5668 CNRS ENS Lyon UCBL)

« Les mathématiciens méprisent, avec beaucoup d'aristocratisme, les calculateurs. »

François Le Lionnais, « Un Certain disparate »[1].

Cet article présente mon initiation à l'informatique durant la période 1964-1973 et je vais tenter de retracer en quoi consistait cette initiation, sachant que comme tous les gens de ma génération, je suis essentiellement autodidacte ; en effet, l'enseignement de l'informatique que nous recevons avant et après 1970 est minimal. Nous prenons quelques cours certes, mais l'essentiel de notre apprentissage se fait par la lecture de livres ou d'articles, peu nombreux, parfois par la redécouverte de concepts, par quelques rares conférences et par les écoles de printemps et d'été qui se mettent en place. Notre méconnaissance de l'anglais est un énorme handicap. En effet, dans ma formation du collège et du lycée, je n'ai appris que l'allemand, le latin et le grec, ce qui n'est d'aucune aide, compte tenu du fait que, dès cette époque, même les Allemands et les Suisses publient en anglais. De plus, immédiatement acquis, les concepts et les contenus scientifiques doivent être enseignés aux autres dans cette construction ininterrompue d'une science nouvelle à laquelle nous participons.

1 Entretien avec Jean-Marc Lévy-Leblond et Jean-Baptiste Grasset (chap. 37, 1976), publié sur *oulipo.net* [URL : https://oulipo.net/fr/un-certain-disparate].

1957	Installation d'une IBM 604, machine à programmes câblés
1958	Installation d'un IBM 650, premier ordinateur
1958	Création du cours d'Analyse et calcul numérique
1959	Création du centre de calcul
1964	Mon entrée à la Faculté des Sciences de Nancy
1965	Installation d'une CAE 510
1967	Création du département informatique de l'IUT
1968	Ma nomination comme assistant délégué (non titulaire) – mon premier cours de programmation
1970	Installation du CII-10070
1971	Ma thèse de 3e cycle
1971-72	Mon service militaire
1972	Ma nomination comme maître assistant à l'Université de Nancy II
1973	Création de l'équipe de recherche associée au CNRS
1974	Mon détachement au CNRS
1976	Création du CRIN, laboratoire associé au CNRS
1979	Ma thèse d'État
1980	Mon départ au MIT

Table 1 – Les étapes de l'informatique universitaire à Nancy et de ma carrière

La table 1 retrace les principales étapes de l'émergence de l'informatique à Nancy, ainsi que ma propre carrière, qui s'y insère. Cependant, j'adopterai ici dans ma démarche une approche parfois un peu technique et globalement subjective. En effet, je n'analyserai pas l'apparition de l'informatique dans le milieu universitaire nancéien, mais je dirai comment j'ai vécu ces années d'apprentissage et de construction de la science qu'est maintenant l'informatique. Je présenterai comment cette activité s'est, chez moi, articulée avec un enseignement de mathématiques, compte tenu de ma formation initiale et de la rareté des postes en informatique à cette époque et compte tenu aussi de mon détachement au CNRS.

Mon but est de faire connaître aux générations plus jeunes, qui ont toujours baigné dans l'informatique, ce que nous, les pionniers, avons vécu. Mais comme je ne considère que le petit bout de la lorgnette, j'invite donc le lecteur qui souhaite un peu plus d'objectivité sur les débuts de l'informatique à Nancy à écouter la conférence enregistrée de Marion Créhange[2] ou à lire, les écrits de Claude Pair[3], de Marion Créhange et Marie-Christine Haton (2014), de Pierre-Éric Mounier-Kuhn (2010) ainsi que ceux dont je suis l'auteur ou le co-auteur (2021 ; 1996). De plus une table donne en annexe C, une description succincte des langages de programmation que j'ai appris à utiliser, tandis que l'annexe A est un glossaire des termes techniques que j'utilise.

Mes études supérieures

Je suis né à Dakar le 22 mars 1947 et j'ai fait mes études secondaires à Colmar. Au moment des études supérieures, à l'automne 1964, je choisis Nancy, parce que ma grand-mère y habite et que je peux loger chez elle, mais aussi parce que je sais qu'en mathématiques c'est un endroit où s'y font de bonnes choses. Ayant lu en terminal, le livre édité par François Le Lionnais, *Les grands courants de la pensée mathématique* (1948), je sais que Nancy jouit d'une réputation d'excellence héritée du temps où elle hébergeait les bourbakistes, et je connais un peu les enjeux de la recherche en ce domaine, sans toutefois, bien sûr, les maîtriser. Quand les cours de Claude Georges, notre professeur en première année, nous conduisent à aborder les « familles sommables » (les « séries[4] » étant considérées par lui comme un peu trop contraintes) dès le mois de décembre et un peu plus tard au mois de janvier « l'intégrale de Stieltjes » sur les « fonctions à variation bornée » (là encore parce que « l'intégrale de Riemann » sur les « fonctions continues » n'est pas assez générale), je suis moins submergé que mes condisciples, pour lesquels cette « pédagogie » de choc fait l'effet d'une douche glacée[5].

2 Marion Créhange, « Le compilateur Algol 60 sur IBM 1620 », Colloque Claude Pair, 14 juin 2019 [URL : https://videos.univ-lorraine.fr/index.php?act=view&id=7762].

3 Claude Pair, « Notions sur la théorie des langages » (cours rédigé par Alain Quéré), Université de Nancy, Faculté des Sciences, Institut universitaire de Calcul automatique. Archives personnelles de l'auteur.

4 En mathématiques, les « séries » sont les sommes infinies qui peuvent diverger ou converger. Dans les « familles sommables », il n'y a pas d'ordre sur les termes. Si l'on compare l'enseignement de Mathématique générale et physique, ou MGP, que l'on appelle propédeutique et qui est la première année de faculté des sciences, avec celui des classes préparatoires aux grandes écoles, on note que les séries n'étaient pas au programme de la première année de math-sup, mais seulement à celui de la deuxième année de math-spé, tandis que les familles sommables n'étaient pas évoquées du tout.

5 Je découvre plus tard que Claude George fonde son cours d'analyse de MGP, sur le livre de Walter Rudin *Principles of Mathematical Analysis* ; ce livre est destiné aux étudiants de quatrième année de bachelor ou de première de master des universités américaines. Le cours de Math 1 de Pierre Eymard (deuxième année de fac pour nous) est fondé sur le livre de Walter Rudin *Real and Complex Analysis* que nous nous procurons

Nous étions dans l'enseignement supérieur depuis un peu plus de deux mois et n'avions jamais entendu parler au lycée de séries et d'intégrales. En 1967, je me suis retrouvé « licencié », la licence étant le diplôme qui permettait de se présenter aux concours de l'enseignement et au DEA[6]. Mais je sais que je veux faire de la recherche et un DEA de math me paraît la chose la plus normale. Je m'y inscris par conséquent. Mais j'ai plusieurs activités annexes : je fais de la voile, je joue au rugby (avec mes 78 kg, comme je suis grand, je suis deuxième ligne !), je suis investi à l'UNEF (Union Nationale des Étudiants de France). Quand Mai 68 éclate, je suis au front à la Faculté des sciences de Nancy, où rétrospectivement on peut dire que ça a été relativement calme, quoique j'apprendrai ultérieurement que deux éminents professeurs en ont été très affectés. Nous, les étudiants syndicalistes, « reconstruisons » l'université et rencontrons les professeurs syndiqués, dont l'un d'eux est au SGEN[7] et avec lequel, je sympathise ; c'est Jean-Louis Ovaert[8].

Rebelle[9], je suis dans mon engagement au syndicat étudiant, et rebelle je suis en mathématiques. De jeunes assistants montent, dès 1967, un séminaire de recherche alternatif, dans une annexe du département de Mathématiques, rue Sellier à Nancy[10] et nous y sommes conviés ; nous nous y rendons, Jean-Luc Rémy[11] et moi, comme des conspirateurs. Quand il s'agit de choisir mon sujet de mémoire de DEA, Pierre Eymard, qui le dirige, nous propose de lire un article de mathématiques, parmi une liste qu'il nous soumet, pour en faire ensuite un compte rendu écrit. Comme au séminaire alternatif, j'ai entendu parler des « mesures idempotentes » je choisis l'article de P. J. Cohen[12] de 1960, « On a conjecture of Littlewood and idempotent measures ». Mes collègues du séminaire alternatif me conseillent en fait de lire et de rapporter sur un autre article intitulé

d'ailleurs dans un achat groupé sur la recommandation de notre professeur. Ce livre est très clairement, dans l'esprit de son auteur, un cours de master (Bac+5). C'est d'ailleurs, dans ce livre que je me suis initié à l'anglais.

6 Le DEA était le Diplôme d'Études Approfondies. Il correspondait *grosso modo* au master d'aujourd'hui après quatre ans d'études.

7 Syndicat général de l'Éducation nationale, CFDT (Confédération française démocratique du travail).

8 René Cori, Anne Michel-Pajus et Robert Rolland, *Jean-Louis Ovaert – Un homme d'action et de convictions*. Brochure IREM-APMEP, juillet 2015 [URL : http://numerisation.irem.univ-mrs.fr/PS/IPS15006/IPS15006.pdf].

9 Ce qualificatif m'a été donné, des années après, par Françoise Bellegarde, pour qui, je suis scientifiquement et universitairement rebelle. Françoise Bellegarde (1942-2016), qui a d'abord été professeure de mathématique en lycée, fera, avec Claude Pair, une thèse de 3e cycle sur la compilation, puis, avec moi, une thèse d'État sur la programmation fonctionnelle. Elle séjournera plusieurs années aux États-Unis, puis sera professeure à Besançon, jusqu'à sa retraite.

10 Le département de mathématiques est situé rue de la Craffe, au cœur de la faculté des sciences où nous avons nos cours. Le bâtiment est aujourd'hui occupé par un collège.

11 Jean-Luc Rémy est le plus brillant étudiant en mathématique de ma génération. Il fera une carrière de chercheur en informatique au CNRS et sera connu pour son algorithme de génération aléatoire d'arbres binaires. *Cf.* Wikipédia francophone, « Algorithme de Rémy » [URL : https://fr.wikipedia.org/w/index.php?title=Algorithme_de_R%C3%A9my&oldid=182172482].

12 Médaille Fields, en 1966, mais je ne fais pas le lien.

« A simple proof of the theorem of P. J. Cohen[13] » (Amemiya & Ito, 1964). Cela plaît à Pierre Eymard, spécialiste d'analyse harmonique, thème de l'article et donc au jury, puisque j'obtiens mon DEA sans problème en septembre 1968.

Suite au décès d'une enseignante chercheuse, dans un accident de voiture[14], un poste d'assistant se libère. Claude Pair, que je ne connais pas encore à l'époque, en parle à son grand ami Jean-Louis Ovaert et je me retrouve embauché, pour enseigner les maths. Ce sera en PC1 (première année du cycle Physique-Chimie). À l'époque, les recrutements se font à la bonne franquette. Je fais un choix, pour l'année 1968-1969 : je commence à enseigner tout en préparant l'agrégation de maths, si ça passe, tant mieux, sinon, je commencerai une thèse de troisième cycle dans une discipline à définir entre maths et la toute nouvelle informatique. Ovaert m'a dit du bien de son ami Claude Pair, et les mathématiques pures, qui m'avaient beaucoup plu en tant que matière d'enseignement, me tentent moins comme domaine de recherche, car elles me paraissent ne traiter que de sujets avec un moindre intérêt social, comme c'est le cas des « mesures idempotentes ». On ne peut pas raconter à son frère ou à sa sœur ce qu'on fait et parfois même pas à un autre mathématicien. D'autre part, la recherche en mathématiques pures, malgré la dynamique du séminaire alternatif ne semble pas être faite en équipe et, si équipes il y a, elles sont à Paris. La présence d'un vrai leader qui s'implique, d'une ambiance, d'un domaine plein de promesses me titille. Quand l'étape de l'agrégation est franchie avec succès, mon choix est fait, je ferai de l'informatique[15], mais je garderai toujours de bons contacts avec les mathématiciens et avec les mathématiques.

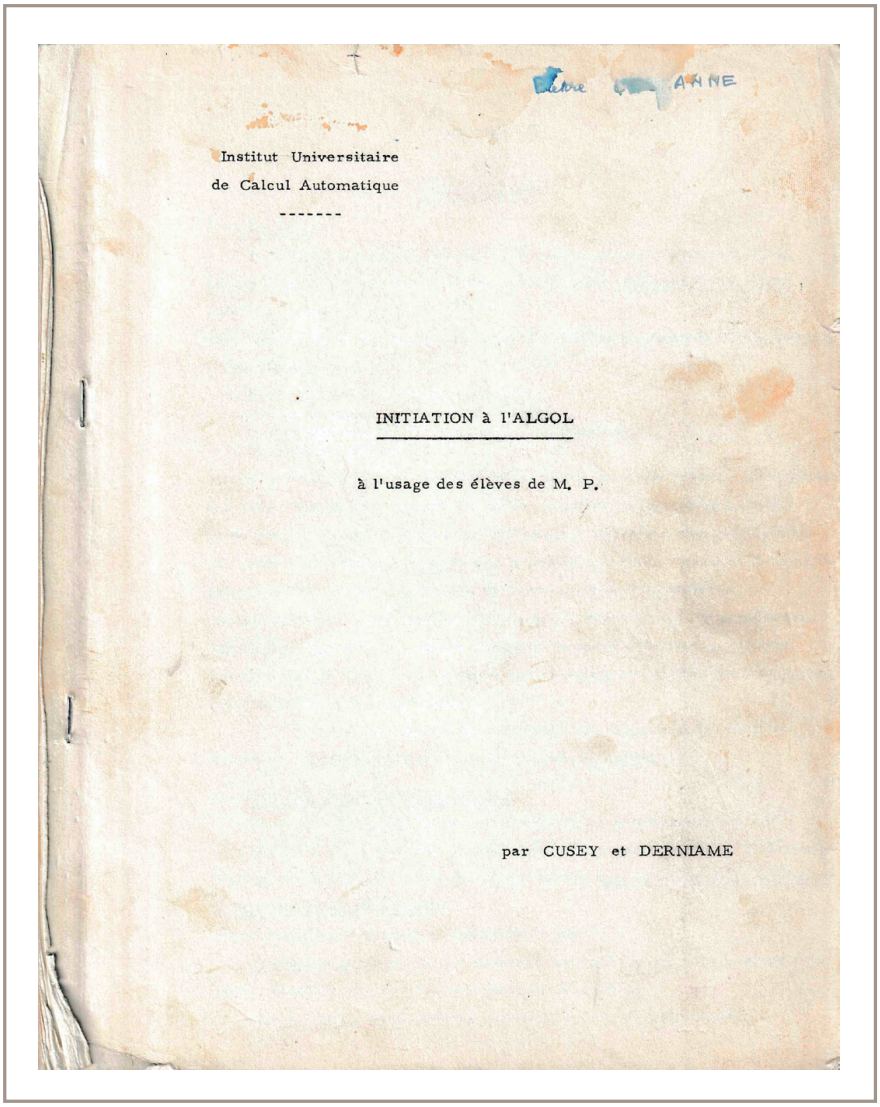
Je suis assistant à la Faculté des sciences jusqu'en 1971, année de mon service militaire. Claude Pair enseigne le cours d'algèbre et je suis une année son assistant pour les travaux dirigés. En 1970, avec Robert Mainard[16], mon professeur de physique de MGP, qui en est à l'origine, je participe à la mise sur pied d'une filière de premier cycle Mathématiques, Physique et Technologies, qui se veut être une alternative aux autres filières scientifiques, à savoir MP (Mathématiques et Physique) et PC (Physique et Chimie) par son contenu et son orientation résolument technologique. Je suis chargé d'un enseignement intégré (cours et TD) de mathématiques devant une promotion d'une vingtaine d'étudiants. J'ai 23 ans, je ne suis guère plus âgé qu'eux.

13 On notera le terme « *the theorem* », comme s'il n'y a aucune ambiguïté sur le théorème de P. J. Cohen dont il s'agit.

14 Joëlle Rousseau.

15 Nous employions le terme depuis au moins 1967, date de l'inauguration du département Informatique de l'IUT qui doit sa création au Ministre Christian Fouché, qui avait fait l'annonce d'un département de « carrière de l'informati*que* », au lieu de « carrière de l'informati*on* ». Les deux départements seront alors créés.

16 Robert Mainard fut ensuite le troisième président de l'Université scientifique de Nancy.


Institut Universitaire
de Calcul Automatique

INITIATION à l'ALGOL

à l'usage des élèves de M. P.

par CUSEY et DERNIAME

Figure 1 – Mon polycopié d'Algol 60

Mes premiers cours d'informatique

Si je veux un jour faire de l'informatique, il faut que j'apprenne la programmation ! Or, à l'automne 1968, un cours d'Algol 60 est proposé et est enseigné par Christiane Legras[17]. Je me souviens de ce cours au tableau noir dans une salle du bâtiment du Centre de calcul, en face du Trésor de la langue française, avenue de la Libération, près de la place Godefroy de Bouillon. Le bâtiment est

17 Fille de Jean Legras (Legras, 2008), fondateur du centre de calcul de Nancy et initiateur du calcul numérique à Nancy. Jean Legras était à l'origine mécanicien avec une thèse sur l'aile portante, mais il deviendra un spécialiste d'analyse numérique, qu'il enseigne à la faculté des sciences et dans les écoles d'ingénieur. Il est l'auteur de plusieurs livres d'enseignement de l'analyse numérique (Legras, 1963, 1968).

moderne, différent de ceux de l'Institut de mathématiques et de physique de la Porte de la Craffe. Nous avons eu comme support de cours le polycopié « Initiation à l'Algol » à l'usage des élèves de Maths-Physique par Cusey et Derniame[18] (*cf.* figure 1 et plan du cours en Annexe B). Au milieu des instructions[19] *début*, *fin*, *si, alors*, *sinon*, et des *;* de cette version francisée d'Algol 60, Christiane nous propose un exemple que je ne comprends pas ; en effet, je n'ai pas fait d'analyse numérique, ni *a fortiori* de programmation, ni encore moins d'algorithmique et je ne reconnais pas, dans l'instruction *x : = (x / 2) + (1 / x);*, la suite… qui calcule √,2. Pour quelqu'un qui, comme moi, a appris une certaine dose de maths, c'est vexant ! Mais ça me fait comprendre sur quoi, en pédagogie, peut reposer un blocage. Ça me fait aussi comprendre que l'enseignement de mathématiques pures que j'ai suivi est particulièrement abstrait. Mais je me souviendrai de cette suite, car je la prendrai, comme exemple, neuf mois plus tard, pour introduire ma leçon d'agrégation intitulé suite de Cauchy ; en effet, cette suite est effectivement un bel exemple de suite de rationnels qui ne converge pas vers un rationnel. Pour calculer cette suite il faut utiliser une boucle *POUR* dont la sémantique (ce qu'elle signifie) nous est expliquée dans l'organigramme de la figure 2. Étant donné qu'aujourd'hui, j'éprouve des difficultés à comprendre cet organigramme, je me demande ce qu'il en était en 1969. En fait, dans ce cours d'Algol 60, nous apprenons plus la syntaxe que la sémantique. Mais nous apprenons surtout comment écrire un programme qui sera accepté par le compilateur. Nous ne savons pas encore qu'affirmer qu'un programme est correct, ce n'est pas cela. Nous commencerons à le comprendre quand nous élaborerons la « Théorie des programmes ». De plus, maîtriser Algol 60, c'est maîtriser les entrées-sorties, comme cela est décrit dans le chapitre IV du polycopié (Annexe B). C'est indispensable, mais un peu rébarbatif et cela n'a pas l'élégance des autres constructions. Leur nécessité s'impose si l'on veut écrire des programmes qui produisent des résultats qui sortiront sur une imprimante, car c'est la seule sortie possible, lisible par le commun des mortels. Les autres médias, à savoir les rubans perforés et les cartes perforées ne sont pas d'une grande utilité pour tester si notre exercice a été concluant. Comme le dit le polycopié, ces instructions sont spécifiques à la CAE 510[20]. À part cela, Algol 60 me plaît beaucoup et l'algorithmique qu'il sous-tend aussi. Avec ce cours, il n'y a pas de passage

18 Michel Cusey et Jean-Claude Derniame. « Initiation à l'Algol à l'usage des élèves de M. P. », Institut Universitaire de Calcul Automatique, 1968. Polycopié.

19 Voir le glossaire en fin d'article.

20 La CAE 510 est une machine qui n'a pas de système d'exploitation. Elle est donc mono-utilisateur et nécessite pas mal de manipulations pour la faire fonctionner comme l'on veut. Les concepteurs du compilateur ALGOL 60 de cette machine n'ont pas pu s'appuyer sur un système d'exploitation pour leurs entrées et sorties et donc ils ont dû tout concevoir à partir de zéro. De plus, le langage machine est spécifique. Ils n'ont pas pu envisager une réutilisation de leur travail.

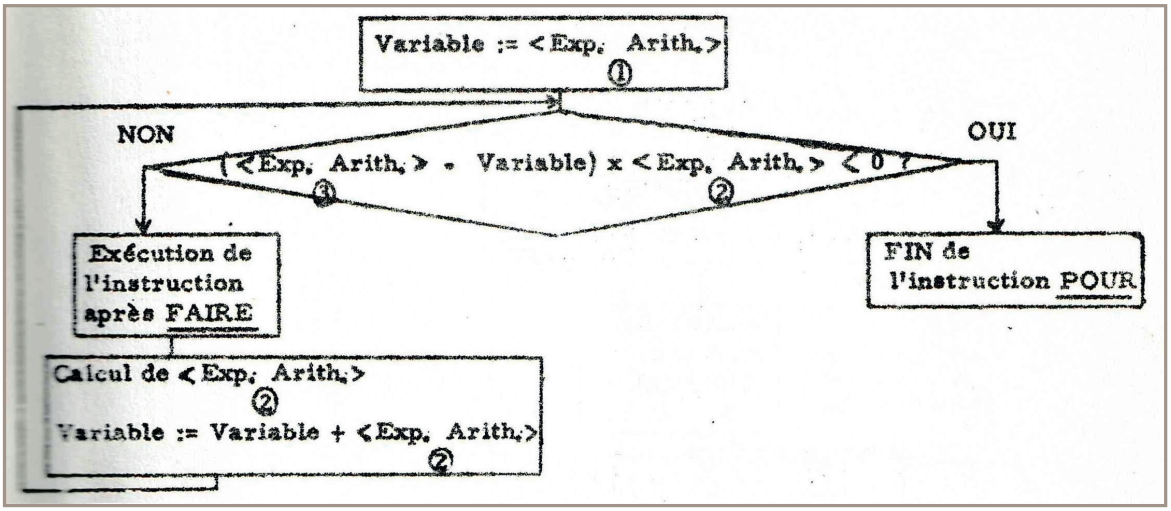


Figure 2 – L'explication de la boucle POUR par un organigramme, dans le polycopié

sur machine. Plus tard, j'essaie mon premier programme Algol 60 (très probablement un calcul de factoriel) sur la CAE 510 du centre de calcul avec sortie sur son imprimante, qui est une machine à écrire IBM à boule (figure 7). Auparavant, il aura fallu charger le compilateur qui est sur un ruban de papier perforé. Tout un art ! Il ne faut surtout pas le déchirer ! Quelque temps après, je suis un cours de COBOL, mais je comprends rapidement que ce langage n'est pas pour moi. Je n'en retiens que ses quatre divisions et ses PIC 999, car, si j'ai bien compris, l'important dans ce langage, ce sont les formats.

Mon ami Jacques Guyard[21] m'a raconté une anecdote qui illustre bien le monde informatique d'alors. Fasciné par un cours d'Algol 60 (peut-être le même que le mien) où on avait dû lui exposer un autre grand classique du genre, à savoir le triangle de Pascal, il veut passer à la pratique. Or il vit dans un monde de physiciens où le langage de référence est FORTRAN. Son laboratoire possède des programmes qui calculent des nombres similaires utilisés par les physiciens. Ces programmes et leurs sous-programmes ne sont pas récursifs et Jacques Guyard s'est dit qu'en les rendant récursifs il en simplifierait le code. C'est ce qu'il fait. Le compilateur ne réagit pas à ces appels imbriqués de sous-programme et produit un code exécutable. Mais bien sûr, ce code fait n'importe quoi. En se renseignant, il comprend que FORTRAN à la différence d'Algol 60 n'autorise pas les appels imbriqués de sous-programme, ce qui est le principe de la récursivité et que le compilateur FORTRAN ne prend pas la peine d'en avertir l'utilisateur, tant la récursivité paraît incongrue, mais il « compile » néanmoins le pro-

21 Par la suite, Jacques Guyard est devenu enseignant-chercheur en informatique, professeur et directeur de l'École d'ingénieur ESIAL, qui deviendra Télécom Nancy.

gramme. En FORTRAN, un appel à un sous-programme n'est rien de plus qu'un branchement[22].

Pour compléter ma formation, je prends un « vrai » cours de programmation : le certificat C1 de la maîtrise d'Informatique qui vient d'être créée. J'y apprends les fondements de la programmation de l'époque à savoir les registres, l'adressage indirect, le calcul d'instruction, les instructions de base comme LOAD ou les branchements, bref le langage machine ainsi que l'assembleur et son langage. J'y apprends aussi ce qu'est un système d'exploitation ou comment une machine « boote ». Le professeur est Jean-Claude Derniame, maître-assistant[23] à l'époque et son assistant pour les travaux dirigés est Jacques Ducloy[24], qui a étudié à l'ENSEM quand j'étais à la faculté. Il a suivi un cours similaire, l'année précédente, ce qui l'autorise à être notre enseignant. D'une année sur l'autre le contenu du cours évolue au gré de l'acquisition des connaissances et des progrès de la technologie. Donc on ne peut même pas dire que Jacques ait suivi le même cours l'année précédente, mais il prend sa fonction très à cœur. Ainsi comme le centre de calcul a acquis un nouvel ordinateur : un CII 10070, nous apprendrons son langage d'assembleur, le SYMBOL, et son métalangage d'assembleur, le METASYMBOL. La documentation de ces langages, surtout du METASYMBOL est insuffisante et probablement incohérente, si bien que nous devrons nous habituer à voir la doxa changer d'une semaine sur l'autre. Nous coopérons avec notre enseignant et ami pour essayer de dégager les concepts de ce langage qui mélange, dans la même syntaxe approximative, le langage d'assembleur et ce que l'on peut dire sur lui. En effet, puisqu'en assembleur, on répète plusieurs fois les mêmes instructions à des nuances près, une suite d'instructions en METASYMBOL, n'est pas un programme, à proprement parler, mais une description visant à produire, puis à faire interpréter un programme d'assembleur en SYMBOL. On s'y perd un peu, entre tel mot clé qui appartient au langage d'assembleur et tel autre qui appartient à son métalangage. Qu'est-ce que dit le manuel ? Le dit-il vraiment ? D'ailleurs nous n'avons que des photocopies d'extraits parmi ceux qui sont pertinents. Et puis, ce cours n'est que théorique, puisque nous n'avons pas encore la machine.

22 Le grand spécialiste de la récursivité, Henry Gordon Rice (le célèbre théorème de Rice est une de ses contributions majeures), écrit en 1965 un article (Rice, 1965) où il montre, ce dont on n'est pas certain à l'époque, à savoir qu'en FORTRAN on peut coder n'importe quelle fonction récursive au sens mathématique du terme, au prix du codage de la pile de récursion par des tableaux. Il implante pour cela la célèbre fonction d'Ackermann, dont il donne un programme FORTRAN d'une trentaine de lignes et un organigramme pour en décrire la structure, alors que le code Algol 60 de la même fonction tient en quelques lignes et n'a pas besoin d'explications supplémentaires.

23 Il devenu professeur à l'Université de Lorraine.

24 Jacques Ducloy a fait l'*École nationale supérieure d'électricité et de mécanique* (ENSEM). Il deviendra ingénieur de recherche au CNRS, d'abord à l'IUCA (Institut universitaire de calcul automatique), puis à l'INIST (Institut de l'information scientifique et technique), puis au LORIA, qui est le laboratoire de recherche en informatique de l'Université de Nancy.

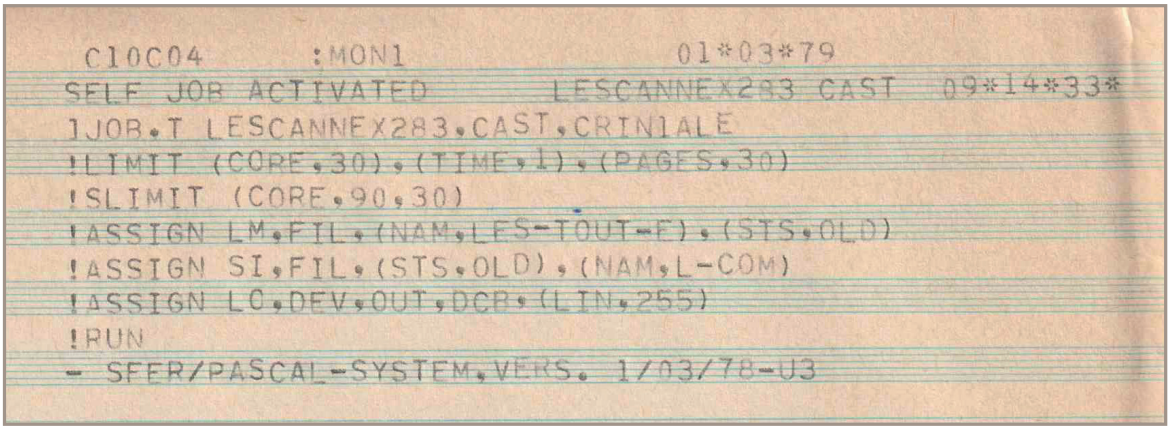


Figure 3 – Listing avec ses « cartes de commandes »

Nous apprenons aussi les cartes de commande du système d'exploitation, qui sont des consignes à l'ordinateur. On parle de « cartes » puisque *a priori* nous entrons nos programmes par cartes perforées. Même quand nous rentrerons les programmes à la console, la syntaxe bizarre par carte (ou ligne) de commande sera conservée. La figure 3 est issue d'un listing relatant, plusieurs années plus tard, l'exécution du programme Pascal de calcul symbolique dont je parle plus loin. Puisque le système d'exploitation fonctionne en traitement par lot[25], il faut lui dire comment prendre en compte mon travail (mon *job*) quand ce sera le moment, pour lui, de le faire. La syntaxe des « cartes » est incohérente et incompréhensible et n'a absolument aucune logique. La seule cohérence entre toutes les cartes est le fait qu'elles commencent par un point d'exclamation « ! », suivi du nom de la commande. Il y a les cartes LIMIT et SLIMIT qui énoncent les limites en temps, en mémoire interne et en mémoire externe dans lequel mon programme a le droit de s'exécuter et les tailles des sorties qu'il a le droit d'imprimer[26]. Si je choisis des limites trop grandes, mon programme sera rejeté du lot. Il y a la carte RUN qui demande l'exécution d'un programme donné, ici le sous-système SFER/PASCAL-SYSTEM qui sert d'environnement à mon programme. La pire des cartes de commande est la carte ASSIGN qui indique où lire les données, soit sur un disque fixe, soit sur un disque amovible, soit sur un lecteur de bande magnétique, soit sur un lecteur de cartes perforées et où produire les sorties sur les mêmes, auxquels il faut ajouter une imprimante à tambour. Chaque support physique (*device*) a sa propre syntaxe, puisque, avec le recul, j'ai l'impression qu'elle a été conçue sur un canevas commun par des pro-

25 Je découvrirai les systèmes en temps partagé au Massachusetts Institute of Technology (MIT) en 1980.

26 Il n'est pas rare malgré ces contraintes de voir l'imprimante échapper à son dresseur et produire en musique des pages de 0 et de 1. Cette musique anticipe celle des *Tambours du Bronx*. Ça fera du papier pour les classes de maternelles !

grammeurs différents qui l'ont définie au moment d'écrire cette partie de logiciel. Mon *listing* de la figure 3 comporte trois cartes ASSIGN. Si ma mémoire est bonne, l'une dit où le système doit trouver mon programme à exécuter, c'est-à-dire dans le fichier LES-TOUT-E, où il doit trouver les données de mon programme, où il doit écrire les résultats de mon programme, en l'occurrence sur l'imprimante à tambour et que ladite imprimante utilise du DCB, c'est-à-dire du « décimal-codé-binaire[27] ». La figure 3 illustre bien le type de sortie produit par une telle imprimante. Quand le programme est rejeté du lot à cause d'une carte de commande erronée, c'est l'humiliation, car il faudra attendre le prochain tour (le prochain lot), qui peut être dans un quart d'heure ou une demi-heure ou le lendemain. Nous sommes très conservateurs et nous nous contentons de recopier les exemples qu'on nous a donnés sur des pages manuscrites photocopiées, en n'essayant de ne commettre aucune erreur dans les parenthèses, les virgules et les mots clés et en essayant d'identifier les parties qui changent pour notre programme et celles qui ne sont pas susceptibles de changement. Il est clair que cette pédagogie n'est pas basée sur des principes et des concepts, mais sur des exemples à imiter, un peu comme les scribes babyloniens apprenaient l'algorithmique (Knuth, 1972).

Pour l'initiation pratique, nous avons, comme pour toute formation technologique, ce que nous appellerions aujourd'hui un projet semestriel, en l'occurrence un projet logiciel[28] puisqu'il s'agit d'une formation à l'informatique. Nous devons écrire un assembleur (un programme de traduction) d'un langage d'assembleur *ad hoc* vers une machine virtuelle[29] et donc simulée, dite machine C. Le tout est écrit dans un autre langage d'assembleur. C'est un processus un peu fastidieux, car le langage n'est pas convivial, mais réaliste, quoique, dans un but didactique, il soit nettement simplifié par ses concepteurs, par rapport à un langage d'assembleur réel de l'époque. Il n'en demeure pas moins que cette activité m'a beaucoup appris. Avant l'arrivée du CII 10070, nous avons la possibilité de passages sur la CAE 510, la seule machine qui nous est accessible. Pour cela, il faut écrire le programme sur des bordereaux *ad hoc*, puis transmettre ces bordereaux à une perforeuse[30] qui nous rend dans un casier un paquet de cartes ; puis munis de notre paquet de cartes, nous pouvons accéder à la CAE 510, dans une salle spéciale qui lui

27 On notera que les restes de la carte utilisent l'anglais, mais que l'abréviation DCB est du français, puisque « décimal codé binaire » se dit en anglais *binary coded decimal* et s'abrège en BCD.

28 Le mot « *logiciel* » venait d'être créé, nous ne l'utilisions pas encore. Il aura plus de succès que les noms « binon » (*bit*) et « bogue » (*bug*) créés en même temps.

29 « Virtuelle » signifie qu'il ne s'agit pas d'un vrai ordinateur mais d'un ordinateur fictif émulé par un logiciel.

30 Une « perforeuse » est une employée du centre de calcul. Sans paraître sexiste, il faut avouer que cette fonction est toujours occupée par une femme à l'époque. Elle perfore (fait des trous dans) les cartes en se servant d'une énorme machine bruyante, qui s'appelle une « perforatrice ».

est dédiée. Auparavant, il faut réserver la salle. Mon groupe de projet est composé de Roger Mohr[31] et de Marcel Navet[32]. Le seul créneau disponible, que nous voulons suffisamment long pour pouvoir faire plusieurs passages en cas d'erreurs, est la nuit du samedi au dimanche. L'un d'entre nous a retiré, auprès de la secrétaire, la clé du centre de calcul qui nous permettra de rentrer et sortir nuitamment. Nous commençons le long processus : d'abord en démarrant la CAE 510 en positionnant les *switchs* du mot sur lequel l'ordinateur *boote*. Le programme qui démarre l'ordinateur est sur une bande magnétique, donc nous n'avons rien à faire qu'attendre que la phase de démarrage soit complète. D'ailleurs « attendre » est l'une de nos principales activités. Pour le compilateur du langage dans lequel la machine C est simulée, il nous faut le charger, car il est sur un ruban perforé. Le lancement de la lecture se fait sur la machine à écrire IBM à boule qui est la console de contrôle opérateur. Les différentes phases du processus sont décrites dans un classeur qui doit rester toujours près de l'ordinateur, sur une table. La plupart du texte est dactylographié, mais des modifications ont été ajoutées au stylo à bille après qu'ont été raturées les consignes obsolètes. Nouvelle attente ! Puis sur une nouvelle instruction à la console, la CAE engloutit notre paquet de cartes perforées. Nouvelle attente ! La machine simulée commence à lire notre programme (nos données), puis se met à calculer. Chaque phase de calcul a son propre bruit. Avant la lecture de la bande magnétique c'est relativement silencieux, seul le clignotement des diodes rouges montre que l'ordinateur calcule. Quand vient son tour, on entend surtout le léger son du dispositif qui la maintient tendue, en gros un petit *tchak* toutes les demi-secondes. La lecture du ruban perforé s'apparente à un sifflement. Puis vient le *tacatac* régulier de la lecture des cartes perforées[33]. Zut, il y a une erreur ! Nous discutons pour en trouver l'origine et proposer une suggestion de correction. Quand nous la localisons, nous utilisons la perforatrice mise à la disposition du public pour perforer une ou plusieurs nouvelles cartes. C'est reparti jusqu'à la prochaine erreur ! Vers onze heures du soir, par inadvertance, je tape sur le clavier de la console (l'IBM à boule). L'ordinateur se plante et le processus que nous avions entamé vers 19h, doit être recommencé (*boot*, chargement du compilateur, lecture du programme de simulation de la machine virtuelle, lecture de nos cartes, premiers calculs). Mes condisciples, après avoir râlé, préfèrent en sourire. Cela aurait pu leur arriver, à ce qu'ils prétendent. Au lieu de quitter le centre de calcul vers minuit, dans une

31 Roger Mohr a été professeur à l'École des Mines de Nancy, puis à l'Ensimag de Grenoble dont il deviendra le directeur (Tombre *& al.*, 2017).

32 Il deviendra chargé de recherche au CNRS en physique à Orléans et son fils Nicolas Navet est professeur d'informatique à l'Université du Luxembourg.

33 Les imprimantes à tambour viendront plus tard et, trop bruyantes, seront installées dans des pièces séparées. Elles ont le *po-pom po-pom* qui les caractérise, qui devient très rythmé quand elles impriment un *dump* mémoire pour un chercheur qui n'a pas pu identifier la source de son erreur et utilise l'ultime recours pour traquer ce que nous n'appelons pas encore un *bug*.

Figure 4 – Une imprimante à tambour, capot ouvert
Source : Wikimedia Commons.

prévision optimiste, nous abandonnons les lieux vers 2h du matin. Notre assembleur prototype traduit presque toutes les instructions. Nous sommes contents. Quelques petites mises au point, avec un meilleur créneau horaire, permettront de rendre un rapport et un projet acceptables. En ce qui nous concerne, comme nous ne sommes pas de véritables étudiants, c'est plutôt pour la gloire !

C'est aussi à cette époque que je suis des cours de Claude Pair sur les automates finis, les langages à contexte libre, les grammaires et l'analyse syntaxique. À ce cours est associé un polycopié qu'a rédigé Alain Quéré[34] et qui correspond à des notes qu'il a prises lors d'un cours précédent[35]. C'est à cette occasion que j'apprends l'importance du point-fixe en informatique[36]. Je suis aussi des cours sur les graphes et les algorithmes de cheminement, théorie dans laquelle Claude Pair et Jean-Claude Derniame ont joué un rôle de pionniers et où aussi le point-fixe est fondamental (Derniane & Pair, 1971 ; Derniane, 2020[37]).

Ma thèse de troisième cycle

Pour Claude Pair, les problèmes centraux de l'informatique naissante sont l'analyse syntaxique et les automates. Il s'intéresse aussi aux arbres (syntaxiques) dont la structure ressemble par certains aspects aux chaînes de caractères, en les généralisant, et qui jouent un rôle fondamental en analyse syntaxique. On doit pouvoir donc décrire des automates d'arbres et définir des langages d'arbres reconnus par des automates d'arbres, que l'on appelle langages reconnaissables. D'autre part, un théorème dit que, pour les chaînes de caractères, les langages reconnaissables par automates sont aussi ceux qui sont engendrés par les opérations algébriques de bases sur les langages dont la fameuse « opération étoile » ; pour cette raison on les appelle « langages réguliers ». Ils sont aussi ceux qui sont solutions d'équations d'un certain type et pour cela ils sont algébriques. Claude[38] veut donc que je généralise ce résultat aux arbres. Mais il se trouve que je suis tombé sur des articles qui proposent une généralisation encore plus grande, en permettant d'énoncer ce théorème sur des structures plus générales, à savoir des algèbres abstraites, dont les chaînes de caractères et les arbres ne sont que des cas particuliers

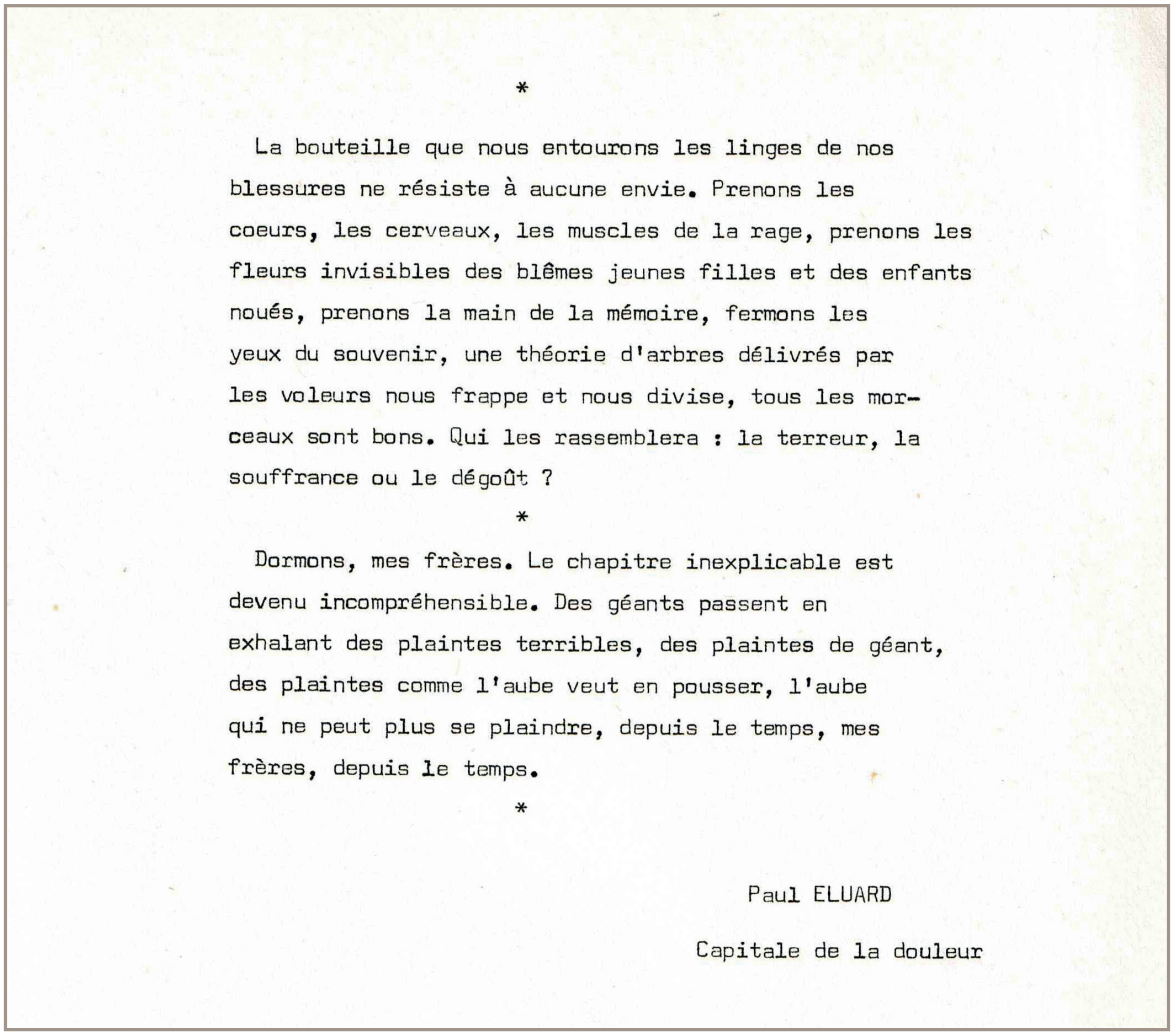
34 Alain Quéré a été maître de conférences à Nancy. Il deviendra directeur de l'Unité Inria-Lorraine, puis directeur du Centre informatique national de l'enseignement supérieur (CINES) à Montpellier.

35 Pair, « Notions sur la théorie des langages », *op. cit.*, 1968.

36 Le point-fixe est au cœur du livre de Livercy (1968), car il est le pivot de la récursivité. Livercy est le nom d'auteur du collectif composé de Jean-Pierre Finance, Monique Grandbastien, Pierre Lescanne, Pierre Marchand, Roger Mohr, Alain Quéré et Jean-Luc Rémy [URL : http://denif.ens-lyon.fr/data/programmation_enslyon/2007_sem2/biblio/Livercy.pdf].

37 Voir aussi Pauline Bolignano et Thierry Viéville, « La découverte scientifique… qui mérite le mérite ? », *Binaire* – Blog sur *lemonde.fr,* juillet 2020 [URL : https://www.lemonde.fr/blog/binaire/2020/07].

38 À l'époque, je ne me serais pas permis de l'appeler par son prénom.



*

La bouteille que nous entourons les linges de nos blessures ne résiste à aucune envie. Prenons les coeurs, les cerveaux, les muscles de la rage, prenons les fleurs invisibles des blêmes jeunes filles et des enfants noués, prenons la main de la mémoire, fermons les yeux du souvenir, une théorie d'arbres délivrés par les voleurs nous frappe et nous divise, tous les morceaux sont bons. Qui les rassemblera : la terreur, la souffrance ou le dégoût ?

*

Dormons, mes frères. Le chapitre inexplicable est devenu incompréhensible. Des géants passent en exhalant des plaintes terribles, des plaintes de géant, des plaintes comme l'aube veut en pousser, l'aube qui ne peut plus se plaindre, depuis le temps, mes frères, depuis le temps.

*

Paul ELUARD

Capitale de la douleur

Figure 5 – La couverture de ma thèse de troisième cycle et la citation d'Éluard

et plus généralement encore en prenant comme cadre la théorie mathématique des catégories, si bien que, par ironie, je mets en exergue de ma thèse (1971) un poème d'Éluard[39], avec en particulier les trois extraits « *une théorie d'arbres* [...] *nous frappe et nous divise* », « *Le chapitre inexplicable est devenu incompréhensible* » et « *Prenons la main de la mémoire*[40] » qui s'appliquent bien, je pense, à mon travail. De fait, je peux généraliser l'étoile et démontrer un théorème d'équivalence des reconnaissables, des réguliers et des algébriques[41]. Ce résultat n'a pas d'écho. Plus tard, en 1976, je publie (en français) dans la revue d'informatique française *RAIRO Infor-*

39 Paul Éluard, « Silence de l'Évangile » dans le recueil, *Mourir de ne pas mourir*. Le titre de cet article, « Prenons la main de la mémoire », a été extrait de ce poème.

40 Ici au sens informatique du terme.

41 Claude Pair, « Colloque en l'honneur de Pierre Lescanne », Archives de l'auteur, 29 mai 2006, LORIA, Nancy.

matique théorique et applications, un article qui n'est cité, d'après le moteur de recherche d'articles académiques Google Scholar[42], que trois fois. Ainsi va la recherche en informatique française dans les années 1970 ! En arrivant au MIT en 1980, je constate que la recherche française en informatique est insignifiante pour nos collègues américains, tandis qu'en France, en informatique théorique, il existe une rivalité Paris-Province.

En 1971, c'est le moment pour moi de faire mon service militaire. Étant agrégé, je peux bénéficier de mon statut pour l'effectuer comme professeur de mathématiques à l'École de Maistrance de l'Aéronautique navale à la base de Fréjus-Saint-Raphaël. Je suis logé sur la base et pour aller de ma chambre à la plage, il me faut contourner le tennis, ça aurait pu être pire ! Dans cette ambiance fréjussienne qui mélange le loisir et la Marine, ma famille me manque beaucoup, mais je souffre aussi de l'isolement scientifique. Je comble cela en continuant une recherche à la suite de ma thèse, mais celle-ci s'élève dans l'abstraction. Claude Pair en particulier m'envoie des documents à lire, en l'occurrence une thèse à évaluer (figure 6).

Heureusement, il est prévu une école d'été d'informatique (la deuxième du genre) à Neuchâtel en Suisse. J'ai raté la première édition, qui a eu lieu à Alès et je le regrette[43]. Au cours de cette école, au demeurant très sympathique et familiale, les jeunes chercheurs sont invités à présenter leurs travaux récents. J'expose les miens, mais je ne sais pas bien m'expliquer, je ne donne aucune intuition, je m'envole dans l'abstraction. Éluard a raison et, au premier rang de l'assistance, l'un de mes meilleurs amis pleure (littéralement) de rire, tant ce que je dis est abscons. Il me faut atterrir.

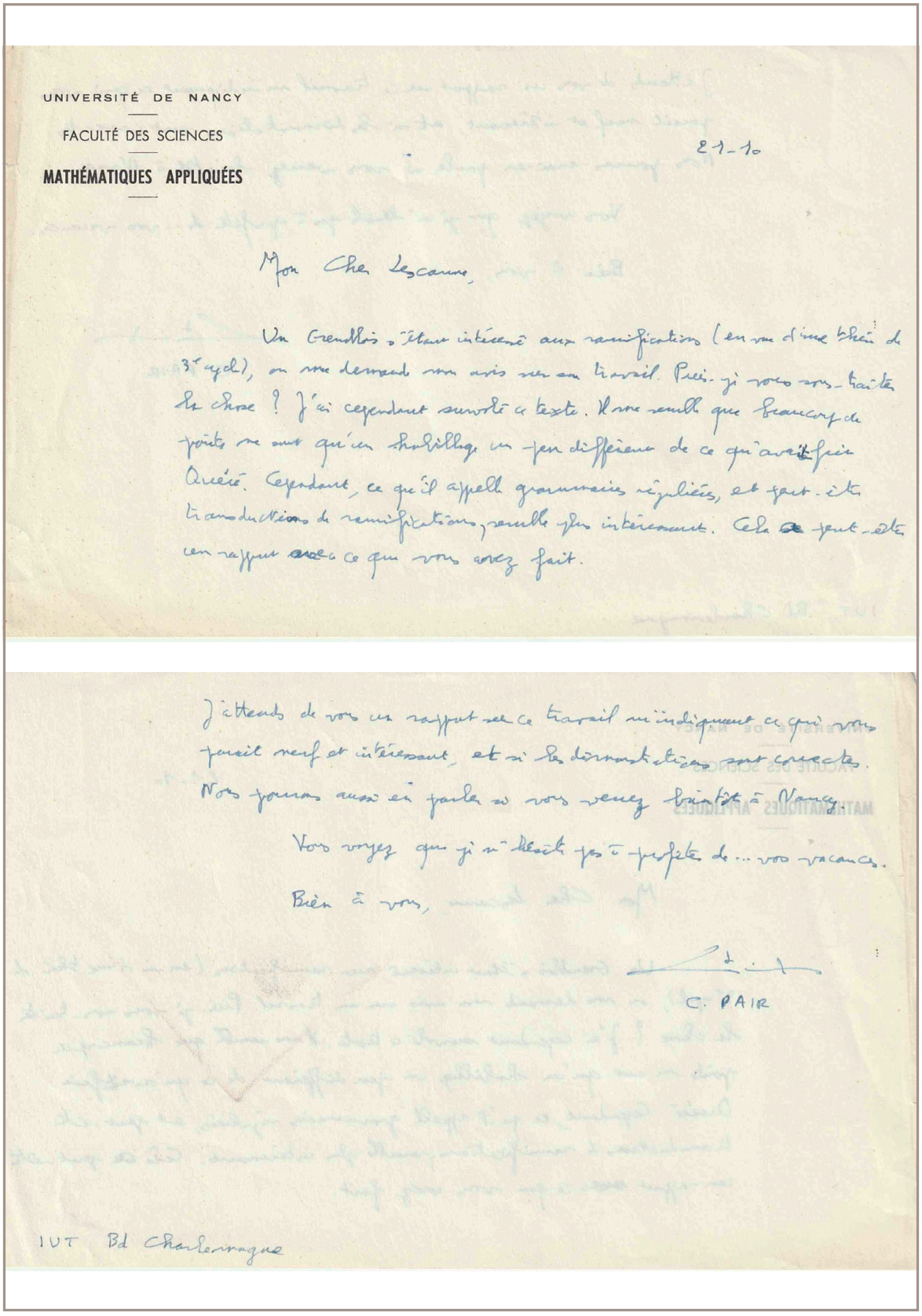
Au même moment, c'est-à-dire en 1972, je rencontre une théorie qui va jouer un rôle important dans ma recherche ultérieure, ainsi que dans mon enseignement[44]. En effet, je découvre le livre de Peter Wegner intitulé *Programming languages, information structures, and machine organization* (1968) qui, entre autres choses, me fait connaître le lambda-calcul. Pour compléter mon initiation, je fais acheter par la bibliothèque de mathématiques le petit fascicule dactylographié de Hindley, Lercher & Seldin (1972), qui, bien que ce ne soit pas dans son titre, présente plus complètement le lambda-calcul. C'est aussi à ce moment que Claude Pair rapporte d'une réunion du groupe de travail de l'IFIP[45], les photocopies des transparents de Dana

42 N.D.E. : Moteur de recherche spécialisé en citations académiques. Ses résultats sont limités par l'ensemble d'articles de revues indexés dans sa base de données.

43 J'assisterai à celle de Tarbes en 1974 et à celle de Rabat en 1975.

44 À l'École normale supérieure de Lyon, à partir de 1997, j'enseignerai le lambda-calcul à des générations d'élèves.

45 L'IFIP (International Federation for Information Processing) regroupe au niveau international les sociétés savantes en informatique.



UNIVERSITÉ DE NANCY

FACULTÉ DES SCIENCES

MATHÉMATIQUES APPLIQUÉES

21-10

Mon Cher Lescaux,

Un Grenoblois s'étant intéressé aux ramifications (en vue d'une thèse de 3e cycle), on me demande mon avis sur ce travail. Puis-je vous soumettre la chose ? J'ai cependant survolé ce texte. Il me semble que beaucoup de points ne sont qu'un habillage un peu différent de ce qu'avait fait Quéré. Cependant, ce qu'il appelle grammaires régulières, et peut-être transductions de ramifications, semble plus intéressant. Cela se peut-être en rapport avec ce que vous avez fait.

J'attends de vos nouvelles sur ce travail me indiquant ce qui vous paraît neuf et intéressant, et si les démonstrations sont correctes. Nous pourrons aussi en parler si vous venez bientôt à Nancy.

Vous voyez que je n'désire pas t'perfide de ... vos vacances.

Bien à vous,

C. PAIR

IUT Bd Charlemagne

Figure 6 – Une lettre de Claude Pair, reçue pendant mon service militaire

Scott[46] sur le modèle du lambda-calcul que ce dernier vient d'inventer (Scott, 1970). Je suis fasciné et pour faire partager mon enthousiasme, j'écris, en français, pour mes collègues, une petite note sur le sujet (Lescanne, 1973). Je ne renouerai avec le lambda-calcul qu'en 1993 (Lescanne, 1994).

Pendant mon service militaire, j'ai été promu maître assistant, puis nommé à l'Université Nancy 2, l'université de lettres et droit. Je continue à enseigner les mathématiques, cette fois-ci aux psychologues. Quant au contenu, c'est assez cool et avec mon collègue Jean-Pierre Deschaseaux, enseignant en mathématiques, nous essayons d'intéresser les étudiants, qui sont réfractaires aux mathématiques. Nous nous investissons donc beaucoup dans la pédagogie, mais lorsqu'un collègue enseignant de psychologie se lamente du grand nombre d'étudiants eu égard aux perspectives professionnelles et nous invite à être en pointe pour la sélection, car les maths sont un bon outil pour cela, alors que la psychologie en tant que discipline d'enseignement ne peut pas jouer ce rôle, nous opposons un refus catégorique. Sans le lui dire, mais en le faisant savoir aux étudiants, nous mettons au point un calcul savant de coefficients, pour qu'un étudiant qui a assisté à tous ses partiels, sans cependant remettre jamais une copie complètement blanche, soit assuré d'avoir la moyenne. Un jour que les étudiant(e)s sont en grève, je vais prendre un café avec le piquet de grève et nous discutons de tout et de rien, notamment de l'intérêt des mathématiques dans le cursus de psychologie. Une étudiante qui, je le comprends bien, n'est pas enthousiasmée par mes cours, me le dit de façon cachée : « *Oh Monsieur vous devriez assister au cours d'histoire du cinéma, c'est formidable*[47] *!* ». Je n'enseigne toujours pas d'informatique, ni quelque chose qui y ressemble, mais un dossier pour la création d'une MIAGE[48] à Nancy est déposé. On me demande d'écrire le programme de mathématiques et je m'y colle, puis je me retrouve à l'enseigner devant la première promotion : graphes et langages entre autres, mon enseignement se rapproche de l'informatique, puisque j'aborde enfin les structures mathématiques de l'informatique. Dans la plaquette de présentation j'essaie de trouver les mots qui puissent motiver des étudiants de sciences économiques à rejoindre notre nouvelle formation[49].

L'enseignement des mathématiques doit se fixer trois objectifs :

- Donner des outils qui serviront soit à d'autres enseignements, soit dans la

46 Dana Scott est grand logicien américain qui s'est intéressé aux fondements de l'informatique.

47 Le cours est enseigné par Roger Viry-Babel, un an plus âgé que moi, avec qui je ne prétends pas rivaliser sur la forme comme sur le fond. Grand cinéphile, Roger Viry-Babel (1946-2006) fut aussi cinéaste, journaliste et pionnier de l'enseignement de l'audiovisuel à l'université.

48 Maîtrise d'informatique appliquée à la gestion des entreprises.

49 Les enseignants de la Miage. MIAGE, « Maîtrise de méthodes informatiques appliquées à la gestion », Plaquette de présentation, 1973-1974, 1973. Archives de l'auteur.

Figure 7 – Une machine à écrire IBM à boule
Source : Etan J. Tal via Wikimédia Commons.

formation permanente, soit dans la vie professionnelle.
- Introduire la démarche algorithmique et la modélisation de situations concrètes.
- Développer l'esprit de rigueur dans la formulation et la résolution des problèmes.

On voit que plus que le contenu, c'est la démarche qui compte. C'est aussi une initiation à l'algorithmique.

En parallèle, je donne des cours au département d'informatique de l'IUT de Nancy, toujours des cours de mathématiques, toujours dans le même esprit, mais qui incluent l'algèbre de Boole et les fameux diagrammes de Karnaugh, que j'apprends pour l'occasion.

Un premier constat

À l'issue de cette période, j'ai conscience de ne pas savoir assez d'informatique. Certes, le sujet me plaît, mais j'ai surtout enseigné des mathématiques et n'ai fait que des recherches

assez théoriques et formelles. D'ailleurs cette conscience de notre inculture est un constat commun parmi les collègues et a conduit à créer l'école d'été de l'« Association française pour la cybernétique économique et technique » (AFCET). La décennie qui vient me fera aborder une certaine forme d'informatique, où je mettrai les mains dans le cambouis ou plus exactement sur le clavier. Mais c'est une « informatique à la française », qui, avec ses pauvres moyens, est décalée et isolée par le plan Calcul, mais qui est pleine de promesses et qui révélera que le bon niveau mathématique des chercheurs français est un atout.

Bibliographie (citée) de l'auteur

Lescanne P. (1971). « Étude de quelques théories des langages et généralisation du théorème de Kleene ». Thèse de Spécialité, Université Henri Poincaré – Nancy 1, Juin 1971.

Lescanne P. (1973). « Introduction au lambda-calcul » [en ligne]. Site personnel de Pierre Lescanne [URL : http://perso.ens-lyon.fr/pierre.lescanne/PUBLICATIONS/lambda.html].

Lescanne P. (1976). « Équivalence entre la famille des ensembles réguliers et la famille des ensembles algébriques ». *RAIRO Informatique théorique et applications*, 10(8), pp. 57-81.

Lescanne P. (1994). « From lambda-sigma to lambda-upsilon a journey through calculi of explicit substitutions ». *In* H.-J. Boehm, B. Lang & D. M. Yellin (eds.). *Conference Record of POPL'94 : 21st ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages* (Portland, Oregon, USA, January 17-21). ACM Press, pp. 60-69.

Lescanne P. (1996). « La science informatique ». *In* G. Grignon (dir.). *Encyclopédie Illustrée de la Lorraine. Histoire des Sciences et des Techniques.* Metz : Éditions Serpenoises, pp. 105-116.

Créhange M., Lescanne P. & Quéré A. (2021). « L'informatique de Claude Pair ». *Bulletin de la société informatique de France*, 1024 (à paraître).

Bibliographie générale

Cohen P. J (1960). « On a conjecture of Littlewood and idempotent measures ». *American Journal of Mathematics*, 82(2), Apr., pp. 191-212.

Créhange M. & Haton M.-C. (2014).

« L'informatique universitaire à Nancy : un demi-siècle de développement ». *Technique et Science Informatiques*, 33(1-2), pp. 127-141.

Derniame J.-C. (2020). « À propos du cheminement dans les graphes » [en ligne]. *Bulletin de la société informatique de France*, 1024/16, novembre [URL : https://www.societe-informatique-de-france.fr/wp-content/uploads/2020/11/1024-numero-16_Article19.pdf].

Derniame J.-C. & Pair C. (1971). *Problèmes de cheminement dans les graphes* (Monographies d'informatique/AFCET). Paris : Dunod.

Hindley J. R., Lercher B. & Seldin J. P. (1972). *Introduction to Combinatory Logic* (vol. 7 of London mathematical Society, Lecture note series). Cambridge (UK) : Cambridge University Press.

Amemiya I. & Ito T. (1964). « A simple proof of the theorem of P.J. Cohen ». *Bull. Amer. Math. Soc.,* 70(06), pp. 774-776.

Knuth D. E. (1972). « Ancient Babylonian Algorithms ». *Commun. ACM,* 15(7), pp. 671-677.

Le Lionnais F. (dir.) (1948). *Les Grands Courants de la pensée mathématique.* Marseille : Cahiers du Sud.

Legras B. (2008). *Jean Legras – Mathématicien lorrain, précurseur de l'informatique à Nancy, fondateur de l'Institut universitaire de calcul automatique*. Laxou : Groupe Dialog'Guyo (auto-édition).

Legras J. (1963). *Précis d'analyse numérique*. Paris : Dunod.

Legras J. (1963). *Initiation à l'analyse numérique*. Paris : Dunod.

Livercy C. (1978). *Théorie des programmes*. Paris : Dunod.

Mounier-Kuhn P.-É. (2010). *L'Informatique en France, de la seconde guerre mondiale au plan Calcul. L'émergence d'une science.* Paris : PUPS (2021).

Pair C. (1990). « CRIN. The history of a laboratory ». *Annals of the History of Computing,* 12(3), July / September, pp. 159-166.

Rice H. G. (1965). « Recursion and iteration ». *Commun. ACM,* 8(2), pp. 114-115.

Scott D. (1970). « Lattice-theoretic models for the lambda calculus ». *IFIP WG 2.2*, Bulletin n° 5.

Tombre K., Quan L., Horaud R., Gros P., Schmid C. & Sturm P. (2017). « In Memoriam Roger Mohr ». *Bulletin de la société informatique de France*, 1024/11, September, pp. 91-98 [URL : https://hal.inria.fr/hal-01598085].

Wegner P. (1968). *Programming languages, information structures, and machine organization*. New York : McGraw-Hill.

Annexe A – Glossaire

Adresse : La mémoire d'un ordinateur est divisée en emplacements qui ont chacun une adresse.

Algèbre de Boole : Une structure algébrique qui sert de fondement mathématique à l'architecture des ordinateurs.

Analyse syntaxique : Opération qui consiste à examiner une chaîne de caractères pour en extraire la structure en composants, en vue d'une traduction.

Algorithme : Ordonnancement des calculs en vue de résoudre un problème. Un algorithme est mis en œuvre par un programme.

Algorithmique : Science qui étudie les algorithmes.

Arbre syntaxique (ou arbre) : Structure vers laquelle l'analyse syntaxique traduit une chaîne de caractères.

Assembleur : Programme, relativement simple, qui traduit un programme en langage d'assembleur en un programme en langage machine.

Automate : Dispositif (programme) qui traite de l'information.

Booter : Démarrer un ordinateur en lui indiquant la première instruction qu'il doit effectuer. Sur un ordinateur d'aujourd'hui, cela est fait par une instruction à laquelle l'utilisateur lambda n'a pas accès.

Calcul d'instruction : Un programme qui s'exécute est amené à répéter plusieurs fois la même instruction ou presque. Le calcul d'instruction est une technique absconse et obsolète qui consiste à mettre en œuvre ce « ou presque » et à modifier les instructions en programme au vol. Cela se fait en considérant les instructions du langage machine comme des valeurs numériques sur lesquelles on peut calculer.

Compilation : Action de traduire un programme dans un langage évolué (*cf.* tableau de l'annexe C) vers un programme du langage d'assembleur d'une machine.

Compilateur : Programme qui effectue la compilation.

Diagramme ou table de Karnaugh : Méthode graphique pour trouver ou simplifier une fonction logique à partir de sa table de vérité.

Éditeur : Logiciel permettant de créer et modifier un fichier. Dans un éditeur ligne à ligne, les commandes s'appliquent à chaque ligne et le résultat n'est pas immédiatement visible. Dans un éditeur plein écran, un curseur permet de voir où se fait la modification

qui est immédiatement visible.

Étoile : En théorie des langages, opération sur un langage qui produit un autre langage. Par exemple l'étoile du langage {ab}, à un seul mot, s'écrit {ab}* et est {ε,ab,abab,ababab,...}.

Grammaire : Ensemble de règles décrivant mathématiquement la structure d'un langage formel. Les grammaires les plus fréquemment étudiées par les informaticiens, sont les grammaires qui définissent les langages à contexte libre.

Graphe : Structure faite de points reliés par des traits sur laquelle on peut exécuter de jolis algorithmes.

Instruction : La composante de base d'un programme.

***Job* :** Tâche qui doit être exécutée par un ordinateur. En français, tâche.

Lambda calcul : Formalisme inventé par le logicien Alonzo Church pour abstraire la notion de calcul. Le lambda calcul a des liens avec la machine de Turing, mais il lui est légèrement antérieur et est plus mathématique.

Langage : En théorie des langages, un langage est un ensemble de mots ou un ensemble d'arbres. Parmi les langages, on distingue des langages reconnaissables, des langages réguliers, des langages algébriques.

Langage d'assembleur : Langage de très bas niveau qui représente le langage machine sous une forme lisible par un humain.

Langage machine : Langage dont les instructions sont des mots machines (donc une suite de bits) qui est fait pour être interprété directement par la machine ou le processeur. Il est presque totalement incompréhensible (directement) pour un humain.

LOAD : Instruction de chargement d'une valeur dans un registre depuis un emplacement de la mémoire.

Mot : En théorie des langages, un mot est un ensemble de lettres.

Mot clé : Dans un langage de programmation, certains mots ont un usage réservé et ne peuvent pas être utilisés comme variable, par exemple *if, then, else, while,* etc.

Mot vide : En théorie des langages le mot vide est le mot qui n'a aucune lettre. Il s'écrit aujourd'hui ε. Claude Pair écrivait le mode vide "Λ" (ou lambda majuscule), probablement parce que c'était plus facile à écrire avec une machine à écrire.

Point-fixe : Solution d'une équation de la forme x=f(x).

Programmation fonctionnelle : Façon d'aborder les programmes comme des fonctions

à évaluer. Le programmeur ne donne pas l'organisation des calculs mais laisse le soin au compilateur de le faire.

Récursivité : Propriété d'un programme d'être récursif.

Récursif : Un sous-programme (une procédure) est récursif (récursive), s'il (elle) contient dans son code un appel à lui-même (elle-même).

Réécriture : (Pour faire simple) la réécriture est la théorie de la simplification des formules.

Registre : Pour accélérer le calcul, un ordinateur possède des emplacements de mémoire à accès plus rapide que le reste de la mémoire. Ces emplacements spécifiques sont appelés des registres.

Régulier : En théorie des langages, un langage est régulier s'il peut être engendré par des opérations simples, à savoir l'union, la concaténation et l'étoile.

Système d'exploitation : Logiciel de base d'un ordinateur, qui donne accès à la machine, gère ses entrées et sorties, permet à plusieurs personnes de l'utiliser et fait fonctionner plusieurs processus en même temps.

Temps partagé : Quand plusieurs utilisateurs ou plusieurs processus peuvent s'exécuter en même temps on parle de temps partagé, en anglais *time sharing*.

Traitement par lot : Quand les programmes exécutés par un ordinateur sont exécutés les uns après les autres, on parle de traitement par lots, en anglais *batch*. Le contraire est le temps partagé.

Théorie des catégories : Théorie mathématique ayant un haut niveau d'abstraction.

Triangle de Pascal : Méthode inventée par Blaise Pascal pour calculer des nombres, à savoir le nombre de combinaisons de n éléments p à p.

Annexe B - Plan du polycopié « Initiation à l'Algol »

- Chapitre I : Généralités
- Chapitre II : Notions permettant d'écrire des programmes simples
- Chapitre III : Nombres, Variables, Expressions arithmétiques simples, Fonctions, Affectations
- Chapitre IV : Les instructions d'entrées et de sorties
- Chapitre V : ALLERA ; Instructions conditionnelles et composées
- Chapitre VI : Les ordres de bouclage : Instruction POUR
- Chapitre VII : Expressions booléennes
- Chapitre VIII : Blocs et déclarations
- Chapitre IX : Les procédures
- Chapitre X : Compléments : Expression conditionnelle, Aiguillage, Expression de désignation, commentaire, La partie valeur, Instruction vide
- Annexe
- Exercices d'Algol
  Parmi les exercices, j'ai noté que ce que nous appelons aujourd'hui un tri est appelé un classement des éléments d'un tableau par valeurs croissantes (Exercice 5.4).

Annexe C – Quelques langages de programmation

FORTRAN	1957	Premier langage de programmation
ALGOL 60	1960	Premier langage avec récursivité et structure de blocs
COBOL	1959	Langage pour la gestion
SYMBOL	1968	Langage d'assembleur du CII 10070
PASCAL	1970	Langage issu d'ALGOL 60, avec des structures de données et une recherche de la simplicité