ALGORITHMIQUE ET APPRENTISSAGE DE LA PREUVE

Simon MODESTE, Sylvain GRAVIER
ERTé Maths à Modeler,
Irem de Grenoble

Cécile OUVRIER-BUFFET
ERTé Maths à Modeler
Irem de Paris 7

Résumé : *Dans cet article, nous présentons tout d'abord une étude épistémologique sur la place et le rôle de l'algorithme dans la science mathématique. Nous étudions les différents aspects de l'algorithme suivant une dichotomie outil-objet, puis nous développons le lien privilégié qu'il entretient avec la preuve. En s'appuyant sur cette étude, nous proposons une analyse des programmes du lycée ainsi que des manuels. Nous proposons, dans un troisième temps, une situation de recherche en classe mettant en jeu l'algorithme. Les résultats d'expérimentations de cette situation montrent comment la construction d'algorithmes, leur preuve et l'analyse de leur complexité peuvent être questionnées en classe.*

Introduction

L'algorithme est un objet méconnu. Souvent vu comme un objet de l'informatique, on l'associe à la programmation. Cependant, l'algorithme est avant tout un objet des mathématiques. « Avant tout », car la notion d'algorithme précède de beaucoup l'informatique. Le terme trouve d'ailleurs son origine dans le nom *al-Khwarizmi*, nom de l'auteur du plus ancien traité d'algèbre connu, datant du IXème siècle. Bien sûr, la formalisation de ce concept doit beaucoup aux avancées de l'informatique. Cette dernière a aussi enrichi les mathématiques de nouvelles questions. Si bien que la recherche de procédures effectives est, aujourd'hui encore plus, une problématique centrale des mathématiques. Le rôle de l'algorithme est si important qu'il est au centre d'une discipline propre, l'algorithmique, à l'intersection entre mathématique et informatique et dont l'essor actuel est considérable.

Face à cette évolution de la science mathématique et à la présence grandissante de l'informatique et de ses applications autour de nous, l'enseignement des mathématiques doit être requestionné. En 2000, la commission Kahane proposait l'introduction d'une « part d'informatique dans l'enseignement des sciences mathématiques et dans la formation des maîtres » (Kahane, 2000, p. 1). Aujourd'hui, c'est dans les programmes de seconde que l'algorithmique trouve une place. Cette présence dans l'enseignement mathématique pourrait entraîner des malentendus entre les communautés mathématique et informatique, notamment concernant la légitimité des enseignants de mathématiques à enseigner l'algorithmique.

Et bien que ces derniers puissent penser que l'informatique est loin de leurs préoccupations, l'algorithmique relève bien de problématiques mathématiques. Se pose aussi une question de formation. A priori, peu d'enseignants ont étudié l'algorithmique en tant que telle dans leur cursus et l'algorithme risque de devenir un simple outil informatique pour les mathématiques. On peut alors s'interroger : quel accueil les enseignants de mathématiques vont-ils faire à l'algorithmique ?

Il semble donc essentiel d'aborder la question de l'introduction de l'algorithme dans l'enseignement des mathématiques d'un point de vue didactique. Ces questions ont été peu abordées et la didactique s'est surtout intéressée à la programmation et à l'outil algorithme plus qu'à l'objet mathématique. On peut citer des travaux récents tels que Nguyen (2005) qui s'intéresse à l'introduction d'éléments de programmation à l'aide de la calculatrice ou Schuster (2004) qui étudie l'introduction de problèmes d'optimisation combinatoire au secondaire. On peut aussi citer les thèses de Cartier (2008) et Ravel (2003), où l'algorithme est étudié dans des domaines mathématiques particuliers : la théorie des graphes en spécialité mathématique de terminale ES pour Cartier et l'arithmétique de la spécialité mathématique en terminale S pour Ravel. Dans cet article, c'est l'algorithme en tant qu'**objet** mathématique que nous allons étudier, sans nous placer dans un champ mathématique en particulier.

Pour mieux appréhender les questions que soulève l'algorithme, il faut se pencher sur son rôle dans les mathématiques et sur les différents aspects qu'il revêt. En s'appuyant sur la dialectique outil-objet développée par Régine Douady (1986), on peut les classer suivant qu'ils relèvent de l'aspect « outil » du concept ou de l'aspect « objet » : l'algorithme n'est pas uniquement un outil pour la résolution de problèmes mais c'est aussi un objet mathématique à part entière, pour lequel il existe un cadre d'étude précis. Précisons :

> *Ainsi, nous disons qu'un concept est outil lorsque nous focalisons notre intérêt sur l'usage qui en est fait pour résoudre un problème. Un même outil peut être adapté à plusieurs problème, plusieurs outil peuvent être adaptés à un même problème. Par objet, nous entendons l'objet culturel ayant sa place dans un édifice plus large qui est le savoir savant à un moment à un moment donné, reconnu socialement.*
>
> (Douady, 1986, p. 9)

Dans une première partie, nous allons étudier plus en profondeur le concept d'algorithme afin de mieux comprendre en quoi il peut être objet d'enseignement en mathématiques. Nous nous intéresserons ensuite à la place du concept d'algorithme dans les programmes et manuels. Ce sera l'objet d'une deuxième partie. La troisième partie se centrera sur l'étude d'une situation pour manipuler l'objet algorithme.

1. — L'algorithme, un objet qui questionne l'activité mathématique et l'enseignement

1.1 *Définition de l'algorithme, différents aspects pour un même concept*

Pour présenter ces différents aspects, nous nous appuyons sur des extraits de la définition du terme « algorithme » issue du dictionnaire des mathématiques (Bouvier et *al.*, 2005). Cet ouvrage présente l'algorithme d'un point de vue généraliste et évoque les approches mathématique et informatique du concept.

> *Un algorithme est une suite finie de règles à appliquer dans un ordre déterminé à un nombre fini de données pour arriver avec certitude (c'est-à-dire sans indétermination ou ambiguïté), en un nombre fini d'étapes, à un certain résultat et cela indépendamment des données. Un algorithme ne résout donc pas un problème unique mais toute une classe de problèmes ne différant que par les données mais gouvernés par les mêmes prescriptions.*
>
> (Bouvier et *al.*, 2005, p. 27)

Cette définition, qui est assez courante dans les ouvrages généralistes, met en avant trois aspects de l'algorithme. Le premier aspect est d'être **soumis à un enjeu de vérité, de preuve** : un algorithme doit fonctionner avec *certitude* quel que soit le problème donné. Un autre aspect est que l'algorithme est un **outil de résolution de problèmes**. Plus précisément, un algorithme permet de résoudre une classe de problèmes. Le troisième aspect qui apparaît ici est qu'un algorithme est **effectif**, il s'applique à des données *finies* et résout un problème en un nombre *fini* d'étapes : il peut être mis en œuvre par un opérateur. La notion d'algorithme est indissociable de cet aspect, comme le souligne Chabert :

> *Aujourd'hui sous l'influence de l'informatique, la finitude devient une notion essentielle contenue dans le terme algorithme, le distinguant de mots plus vagues comme procédé, méthode ou technique.* […] *Finitude du nombre des opérations et du nombre des données, mais finitude aussi de la résolution, c'est-à-dire que chaque étape doit pouvoir être réalisée selon un processus fini -ce qui n'est pas le cas, par exemple, du quotient de deux nombres réels incommensurables. On parle aussi de procédé effectif, c'est-à-dire permettant d'obtenir effectivement le résultat (en un temps fini).*
>
> (Chabert, 1994, p. 6)

Une autre question se pose alors : Tout ce qui est effectif, une formule par exemple, est-il un algorithme ?

Pour Bouvier et *al.* (2005) la réponse est oui. Plus précisément, une formule peut être vue comme un algorithme dont les prescriptions ne varient pas en fonction des données. Il semble donc que la notion d'algorithme englobe celle de procédure effective, mais nous verrons plus loin que ces « algorithmes-formules » ont un intérêt limité du point de vue de l'algorithmique (paragraphe 2.2).

Le développement de l'informatique a bien sûr une influence importante sur l'algorithmique et soulève des questions d'implémentation de l'algorithme. Ces questions de programmation et de langages informatiques ne seront pas étudiées ici. Comme le souligne Bouvier et *al.* (2005), des questions se posent aussi quant à l'efficacité de la mise en œuvre de ces algorithmes : la question des temps de calcul, c'est-à-dire le nombre d'étapes élémentaires qu'effectue un algorithme en fonction de la taille des données, mais aussi la question de la taille mémoire, c'est-à-dire la place nécessaire pour stocker les différentes structures de données, correspondant à la complexité en temps et en espace de l'algorithme.

Un nouvel aspect est ici évoqué : **la complexité**.

Les questions d'existence d'algorithme pour certains problèmes et de leur complexité ont conduit à un cadre théorique d'étude de l'algorithme en tant qu'objet : l'hypothèse fondamentale de la théorie des algorithmes est que tout algorithme peut être réalisé par une machine de Turing particulière et aussi, par conséquent par une machine de Turing universelle [1]. Cela constitue un autre aspect de l'algorithme.

Un dernier aspect de l'algorithme peut être soulevé, l'aspect **constructif** de l'algorithme : En ce qui concerne les problèmes d'existence, qui divisent les mathématiciens sur le plan philosophique, les algorithmes permettent d'apporter une méthode de construction de l'objet reconnue par l'ensemble des mathématiciens (Bouvier et *al.*, 2005).

Parmi les différents aspects relevés certains font référence à l'outil, d'autres à l'objet. Regarder l'algorithme en tant qu'**objet**, c'est s'intéresser aux questions de bon fonctionnement, de domaine de validité, de complexité et de description des algorithmes. Ce sont des problématiques de l'**algorithmique**. Les aspects preuve, complexité, machine de Turing et constructif réfèrent à l'objet algorithme.

1 Une machine de Turing est un modèle abstrait du fonctionnement des appareils mécaniques de calcul, créé par Alan Turing afin de donner une définition précise du concept d'algorithme. Pour plus de détails on pourra consulter : Pierre Wolper. Introduction à la calculabilité, cours et exercices corrigés, 3ème éd. Dunod (2006).

Regarder l'algorithme en tant qu'**outil**, c'est s'intéresser à l'utilisation que l'on en fait pour résoudre des problèmes. Les aspects résolution de problème, effectivité et formule réfèrent à l'outil algorithme.

Ce sont donc les questions de construction, de preuve et de complexité des algorithmes qui font vivre l'algorithme en tant qu'objet des mathématiques et c'est autour d'elles que notre étude s'articule.

1.2 *L'algorithmique fait-elle appel à un mode de pensée spécifique ?*

Knuth (1985), qui exerce indépendamment des activités mathématiques et informatiques, s'interroge sur les différences qui existent entre la *pensée mathématique* et la *pensée algorithmique* (l'algorithmique étant, pour lui, entendue au sens large, comme la science informatique). Il repère dans l'activité du mathématicien neuf grands *modes de pensée* et remarque que six d'entre eux sont communs à la pensée algorithmique :

— la manipulation de formules
— la représentation d'une réalité
— la réduction à des problèmes plus simples
— le raisonnement abstrait
— les structures d'informations
— les algorithmes.

Il ajoute deux catégories à la pensée algorithmique qui lui semblent ne pas être présentes dans la pensée mathématique [2] :

— la notion de complexité
— la notion d'affectation symbolisée par := ou $\leftarrow$. Précisons ici que Knuth fait référence

2 Knuth souligne tout de même que certains textes mathématiques sont extrêmement proches de ce qu'il appelle la pensée algorithmique, en particulier un livre d'analyse constructive : Bishop E. Foundations of constructive Analysis. McGraw Hill (1967).

à une notion d'affectation dynamique : l'affectation successive de différentes valeurs à une même variable au cours des différentes étapes d'un processus.

La notion de complexité semble donc propre à l'algorithmique. Knuth fait remarquer que, dans les résultats constructifs rencontrés en mathématiques, le plus souvent, on ne tient pas compte du « coût » de la construction, pour lui tous les ingrédients d'un algorithme ne sont pas rassemblés.

La notion d'affectation n'est pas sans rappeler la distinction entre *variable mathématique* et *variable informatique*. Une variable mathématique est un symbole représentant un élément non spécifié ou inconnu d'un ensemble ou jouant un rôle de marque place (en logique par exemple). Une variable informatique désigne un emplacement dans la mémoire, son contenu peut changer. L'opération d'affectation (relation antisymétrique) diffère de la relation d'égalité (évidemment symétrique).

L'algorithmique fait appel à des raisonnements communs aux mathématiques, mais aussi à un mode de pensée spécifique. Il semble essentiel que l'étude de l'algorithme en tant qu'objet permette la confrontation à ce mode de pensée. Une préoccupation centrale des mathématiques est de fournir des démonstrations de ses résultats, il est donc inévitable, si l'on regarde l'algorithme comme un objet des mathématiques, d'interroger le lien algorithme-preuve.

1.3 *Algorithme et preuve*

L'algorithme entretient un lien étroit avec la preuve, de diverses manières. Pour mieux les comprendre commençons par regarder un exemple :

Algorithme de recherche du pgcd

L'existence du *pgcd* de deux entiers peut être prouvée comme suit :

Preuve : Soient a et b deux entiers avec $a \geq b$. Notons D l'ensemble des diviseurs communs à a et b. Comme $1 \mid a$ et $1 \mid b$, on a $D \neq \emptyset$, et D est inclus dans $\{1, \ldots, a\}$. D est borné non vide, donc possède un plus grand élément d noté $pgcd(a,b)$.

Le *pgcd* peut être obtenu par l'algorithme d'Euclide[3] :

Euclide(a,b) :
  $r_1 \leftarrow a$
  $r_2 \leftarrow b$
  tant que $r_2 \neq 0$
    $r_1 = q.r_2 + r$ *(division euclidienne)*
    $r_1 \leftarrow r_2$
    $r_2 \leftarrow r$
  retourner r_1

Il n'est pas évident que cet algorithme fonctionne et construise le *pgcd* de a et b. Il est nécessaire de le prouver :

Preuve :
— Preuve de la correction[4]. Notons a_k, b_k et q_k les valeurs successives prises par r_1, r_2 et q après k pas de la boucle *tant que*. On a pour tout k : $a_k = q_k.b_k + b_{k+1}$.

3 On utilise ici quelque notations propores à l'informatique et à la logique pour simplifier la présentation.
4 c'est-à-dire : preuve que l'algorithme donne bien le *pgcd*.

Donc pour tout entier d, ($d \mid a_k$ et $d \mid b_k$) ssi ($d \mid a_{k+1}$ et $d \mid b_{k+1}$). D'où $pgcd(a_k, b_k) = pgcd(a_{k+1}, b_{k+1})$.
Ainsi lorsque l'algorithme s'arrête après n pas : $pgcd(a,b) = pgcd(a_n, 0) = a_n$.
- Preuve de terminaison [5]. On remarque que la suite (b_k) est strictement décroissante. Donc la condition d'arrêt $b_k = 0$ sera vérifiée après un nombre fini d'itérations et l'algorithme s'arrêtera.

Revenons maintenant sur les liens qu'entretiennent algorithme et preuve.

Preuve d'algorithme

Tout d'abord, pour présenter un algorithme et affirmer qu'il résout une famille de problèmes, il convient d'en donner une démonstration. Il faut être certain qu'il aboutit au bon résultat quelque soit le problème en instance, c'est ce que l'on appelle la **correction**, et pouvoir garantir que ce résultat sera atteint en un nombre fini d'étapes, c'est la **terminaison**. C'est ce que nous avons illustré dans l'exemple précédent.

De plus un algorithme peut être utilisé au cours d'une preuve en tant qu'inférence (par exemple lorsque l'on fait appel à l'algorithme du *pgcd* pour prouver l'existence des coefficients de Bezout). La preuve de l'algorithme donné est alors un point indispensable à la validation de la démonstration concernée.

La preuve d'algorithmes revêt aussi une importance particulière depuis l'utilisation d'algorithme dans des preuves récentes, en tant que substitut au mathématicien pour des tâches non accessibles à l'homme (théorème des quatre couleurs par exemple).

L'algorithme dans l'activité mathématique

Les preuves dites constructives (la preuve par induction en est un exemple) peuvent en général donner lieu à un algorithme. Il peut être nécessaire d'expliciter les algorithmes sous-jacents. Ce type de preuves constructivistes concerne le plus souvent des problèmes d'existence d'objets ou de reconnaissance d'une classe d'objets.

Afin de démontrer l'existence d'un objet ayant une certaine propriété *(P)* les méthodes constructives exhibent de tels objets. En mathématiques, la question de la caractérisation de tous les objets vérifiant *(P)* s'avère une question de recherche souvent difficile. Un moyen de répondre à cette question est l'énumération de tous ces objets : cette technique peut aussi faire appel à un algorithme. Cependant le problème d'énumération est parfois encore trop difficile. De plus, l'algorithmique permet d'aborder des questions de reconnaissance des objets vérifiant *(P)* : étant donné un objet *O,* un algorithme de reconnaissance permet de dire si *O* vérifie *(P)*. Ces questions sont clairement de nature mathématique et ne peuvent être abordées sans le concept d'algorithme.

Reprenons l'exemple des preuves d'existence. Pour un même résultat, on peut distinguer deux types de preuves :

— Les preuves d'existence « théoriques », sans construction explicite de l'objet : c'est souvent le cas des preuves par l'absurde ou des preuves probabilistes. La preuve d'existence du *pgcd* donnée plus haut en est un exemple.

5 c'est-à-dire : preuve donne le résultat après un nombre fini d'étapes.

— Les preuves constructives parmi lesquels il faut distinguer les preuves algorithmiques qui donnent une méthode de construction des objets et la construction explicite des objets considérés.

Le plus satisfaisant est sans doute de bénéficier d'une construction explicite et d'un algorithme de reconnaissance de la classe d'objets. On dira alors que le problème de reconnaissance relatif au théorème d'existence est entièrement résolu.

Complexité, optimalité et preuve

L'étude de la complexité d'algorithmes est une autre problématique en lien avec l'activité de preuve. La complexité peut être abordée sous divers angles dont deux sont classiques dans l'étude d'algorithmes : la *complexité au pire* et la *complexité en moyenne* [6]. Se pose alors la question de recherche d'algorithmes optimaux pour un problème donné. Là encore, la preuve joue un rôle fondamental. L'apprentissage de la preuve est une problématique centrale de l'enseignement des mathématiques. Au vu des liens que l'algorithme entretient avec la preuve, il est naturel de se poser la question suivante : *L'algorithme peut-il être un levier efficace pour l'apprentissage de la preuve ?*

2. — L'algorithme dans l'enseignement : outil ou objet ?

2.1 *Les intérêts d'un enseignement d'algorithmique*

La commission Kahane (2000) souligne les intérêts d' « introduire une part d'informatique dans l'enseignement des sciences mathématiques et dans la formation des maîtres ». La commission avance pour cela des arguments que nous résumons ici :

- l'esprit algorithmique, présent implicitement dans l'enseignement, pourrait être travaillé et mis en lumière grâce aux instruments de l'algorithmique ;
- la programmation permet la formalisation du raisonnement ;
- les questions d'effectivité des algorithmes mettent en jeu les mathématiques ;
- traitement des données et calculs par informatique sont courants dans les autres disciplines scientifiques ;
- l'informatique a changé les mathématiques :
  - — en permettant d'aborder les objets sous un autre angle,
  - — en apportant de nouvelles questions,
  - — en créant de nouveaux domaines des mathématiques, aujourd'hui en plein essor,
  - — en transformant l'activité du mathématicien grâce à de nouveaux outils.

La commission propose des contenus pour cet enseignement d'informatique : les questions de représentation et d'approximation des nombres en informatique, les concepts de base de la programmation et de l'algorithmique.

Elle donne trois exemples : l'étude des graphes, la recherche d'enveloppes convexes dans le plan et l'étude des fractions continues. Il semble que cette proposition ait été entendue au vu des programmes de seconde pour la rentrée 2009.

6 Ces deux notions de complexité seront définies précisément à la fin du paragraphe 3.1.

2.2 *Place actuelle et rôle de l'algorithme dans les programmes et les manuels* [7]

Nous avons souligné plus haut que considérer l'algorithme sous son aspect objet conduit à se poser les questions de la construction, la preuve, la complexité et l'optimalité des algorithmes considérés. Nous ne nous intéresserons donc ici qu'aux algorithmes permettant un questionnement sur au moins un de ces points (au regard de l'algorithmique comme savoir savant et non du point de vue des programmes ou des manuels). Pour préciser notre démarche, regardons sur des exemples d'algorithmes issus des programmes si ces questions sont en jeu.

L'algorithme d'Euclide soulève clairement toutes ces questions, nous nous intéresserons donc à lui dans les programmes et manuels. C'est aussi le cas des algorithmes de recherche par dichotomie ou du pivot de Gauss. Les questions de tri de données permettent aussi de tels questionnements.

Par contre, bien que les résolutions d'équations du premier degré ou du second degré avec le discriminant puissent être vue comme des algorithmes, elles ne soulèvent pas ces questions de construction, de preuve et de complexité d'algorithmes. Ce sont en quelque sorte des formules, que l'on peut appliquer de manière systématique et implémenter informatiquement mais qui n'interrogent pas l'objet mathématique algorithme. Nous laisserons ces « algorithmes » de côté dans nos analyses des programmes et des manuels.

Dorénavant, si cela n'est pas précisé, lorsque nous emploierons le terme algorithme, nous exclurons les formules. Nous entendrons par algorithmique la discipline du savoir savant qui étudie l'algorithme en tant qu'objet mathématique. Lorsque nous parlerons par exemple, d'introduire des notions d'algorithmique, nous entendons par là un questionnement sur la construction d'algorithmes, leur preuve, l'étude de leur complexité etc.

Ces précisions faites, nous pouvons étudier la place et le rôle de l'algorithme (formules exclues, donc) dans les programmes et les manuels. Cette étude s'est appuyée sur la dialectique outil-objet (Douady, 1986), sur une approche praxéologique des contenus et sur une analyse écologique des savoirs (Chevallard, 1998). Nous relatons ici les principaux résultats de cette étude (pour plus de détails, voir Modeste (2009)).

Programmes de lycée

La place de l'algorithme n'est spécifiée que dans les programmes de première et terminale L dans le cadre de l'option de mathématiques, et en terminale ES dans le cadre de la spécialité mathématiques. Dans le cas de la série littéraire, l'algorithmique est un domaine transversal qui doit permettre d'aborder les mathématiques de manière pratique et concrète et de distinguer résolutions théorique et effective de problèmes. Les élèves doivent être entraînés à des techniques relatives à l'algorithmique : décrire, interpréter et mettre en œuvre des algorithmes simples. Cela semble s'articuler avec l'enseignement obligatoire de mathématiques-informatique (notamment concernant l'utilisation des outils informatiques). On est proche d'un enseignement de mathématiques appliquées.

Dans la spécialité mathématiques de terminale ES, l'algorithmique est amenée par la théorie des graphes. L'algorithme est pré-

7 Avant application des programmes de seconde à la rentrée 2009.

senté comme un outil indispensable pour traiter des graphes complexes, certains algorithmes doivent être présentés mais la majorité des exercices doit se résoudre sans recours à l'algorithme. Ici aussi, l'accent est mis sur l'aspect appliqué des mathématiques, mais cela passe plutôt par la modélisation.

Dans les deux cas, aucun exposé théorique d'algorithmique ne doit être fait.

Dans la série scientifique, l'algorithme est très peu présent et ne fait l'objet d'aucun travail particulier. Au vu de ce qui est dit de l'algorithmique et son intérêt, dans les programmes de première et terminale L, on peut s'interroger sur son absence dans la série S. On peut faire l'hypothèse que le constat serait le même concernant la place des mathématiques appliquées.

Dans les trois filières, seul l'aspect outil de l'algorithme est mis en valeur par les programmes (avec une légère différence pour la série L, où la question de la complexité est évoquée à propos de l'algorithmique mais ne réapparaît pas dans les contenus).

D'autre part, on constate que trois domaines sont privilégiés par les programmes pour l'introduction d'algorithmes :

— L'arithmétique (collège, seconde, première et terminale L (option), terminale ES (spécialité)).

— L'analyse, dans le cas d'approximations (première L (option) et terminale S).

— La théorie des graphes (Terminale ES (spécialité)).

Malgré cela, dans ces domaines, beaucoup d'algorithmes sont évités alors que tout est favorable à leur introduction. C'est notamment le cas en arithmétique à différents niveaux ou en spécialité de terminale ES, dans le cadre de la théorie des graphes, où l'on peut s'interroger sur ce qui a motivé le choix des algorithmes au programmes [8]. On note aussi que d'autres domaines pourraient être des champs d'introduction d'algorithmes : simulation et modélisation, géométrie, résolution de systèmes d'équations, calcul matriciel.

Manuels du lycée

Nous avons choisi d'analyser un manuel pour chaque classe des filières générales du lycée. Pour la seconde et la filière S, nous avons choisi la collection Terracher (Hachette). Mais cette collection n'existe pas pour les autres filières. Pour la filière ES, nous avons choisi les manuels de la collection Transmath (Nathan). Pour la spécialité de la filière L, il semble qu'il n'y ait pas de manuels. Pour répondre à nos questions, concernant cette filière, une étude des pratiques enseignantes en classe aurait été nécessaire. Nous n'avons pas pu la mettre en œuvre ici.

Dans les manuels étudiés, nous avons fait les constats suivants :

— Le concept d'algorithme est présent mais très peu mis en valeur institutionnellement. D'une part, l'algorithme n'est pas complètement défini, voire souvent pas du tout défini [9]. D'autre part, il ne fait pas l'objet d'une présentation spécifique et les algorithmes sont souvent présents de manière implicite sans qu'un travail spécifique ne soit réalisé (travail de description, de preuve ou d'analyse).

8 Voir (Cartier, 2008).

9 La seule définition de l'algorithme que nous avons rencontré dans ces manuels le présente comme une « méthode permettant de résoudre un problème et qui se présente sous la forme d'une suite d'opérations élémentaires obéissant à un enchaînement déterminé ».

— Les algorithmes proposés dans la partie « cours » sont rarement prouvés rigoureusement ; souvent, on n'a pas même l'ébauche d'une preuve. Cependant, deux domaines mathématiques laissent une place à la preuve : l'arithmétique en spécialité de terminale S et la théorie des graphes en spécialité de terminale ES où quelques algorithmes sont prouvés ou justifiés (notamment l'algorithme d'Euclide pour le *pgcd* et le théorème d'existence d'un cycle eulerien dans un graphe), mais où d'autres preuves pourraient être traitées.

— La complexité des algorithmes de la partie « cours » n'est questionnée dans aucun manuel et de manière extrêmement rare dans les exercices (il s'agit alors de compter des multiplications ou des additions).

— Des algorithmes sont très rarement mobilisés dans des preuves. Cela est tout de même présent de manière sporadique dans les spécialités de terminales S et ES [10] (preuve de l'existence des coefficients de Bezout ou preuve du théorème d'existence de chemins eulériens).

— Un unique type de tâche est réellement développé relativement à l'algorithme : la mise en œuvre sur des données, c'est-à-dire un type de tâche relatif à l'*aspect outil*.

Nous pouvons dire que l'algorithme occupe strictement la même place que celle prévue dans les programmes. Seul son aspect outil est développé, comme si l'algorithme n'avait qu'une existence en tant que technique de résolution. Quant au lien de l'algorithme à la preuve, il est très peu exploité par les manuels.

10 Concernant la réintroduction de l'arithmétique en terminale S (spécialité) on pourra consulter la thèse de Ravel (2003) où la preuve et l'algorithme sont abordés. Pour les questions concernant la théorie des graphes en terminale ES (spécialité) on pourra consulter la thèse de Cartier (2008).

2.3 *En conclusion*

Nous avons vu que l'algorithme est présent dans le savoir à enseigner, mais sa place et son rôle sont très restreints et son lien à la preuve n'est pas exploité : sa place est limitée à certains domaines mathématiques (arithmétique, approximation en analyse et théorie des graphes) où il n'apparaît essentiellement que comme outil. Nous avons montré dans la première partie, que l'algorithme en tant qu'objet aurait toute sa place dans l'enseignement, en particulier pour son lien à la preuve. Actuellement, son rôle est limité à celui d'outil, comme nous l'avons montré dans cet état des lieux alors qu'il existe d'autres aspects de l'algorithme qui pourraient être étudiés (la complexité par exemple) et qui ne sont pas du tout traités dans les programmes et manuels.

Le lien de l'algorithme à la preuve n'est ni évoqué dans les programmes ni exploité par les manuels : on ne rencontre pas beaucoup de preuves d'algorithme et les algorithmes sont rarement mis en lien avec les preuves. En résumé, dans les programmes et les manuels de l'enseignement secondaire français, il semble que l'algorithme ne vit que dans la niche technique de résolution. Mais d'autres niches existent pour lui dans le savoir savant : la niche preuve et bien sûr la niche algorithmique elle-même.

3. — Une situation pour l'algorithme

Nous souhaitons montrer ici que l'on peut proposer des situations permettant l'accès aux différents aspects de l'algorithme et notamment à l'aspect objet. Nous allons, pour cela, nous intéresser à un problème de recherche de fausses pièces.

3.1 *Présentation du problème*

Un problème de fausses pièces

Dans un ensemble de pièces, indiscernables à la vue ou au toucher, se trouvent des fausses pièces. Les vraies pièces pèsent toutes le même poids, les fausses aussi mais leur poids est différent de celui des vraies. A l'aide d'une balance Roberval à deux plateaux et sans poids, comment peut-on retrouver les fausses pièces ? Quelle est la méthode qui permet de les retrouver en effectuant le moins possible de pesées ?

On peut définir plusieurs variantes de ce problème en jouant sur les variables de la situation suivantes :

- Poids des fausses pièces :
  — On sait au départ que les fausses pièces sont plus lourdes que les vraies (ou plus légères).
  — On ne sait rien.
- Le nombre de fausses pièces :
  — Fixé à l'avance.
  — Compris dans un intervalle donné.
  — Non connu à l'avance.

Ce problème ainsi que la plupart de ses variantes, ne sont pas encore complètement résolus pour la recherche en mathématiques [11]. Nous limiterons notre étude à quelques variantes, en jouant sur les variables didactiques, que nous choisirons pour permettre une mise en œuvre d'algorithmes et leur étude par des élèves.

L'algorithmique dans ce problème

Ce problème relève bien d'une problématique d'algorithmique : il s'agit de construire un algorithme de recherche des fausses pièces, de le prouver et d'étudier son optimalité. L'énoncé du problème ne fait appel à aucune notion mathématique particulière mise à part la notion d'optimalité, mais qui peut-être comprise au sens intuitif. L'appropriation de ce problème nous paraît donc relativement facile.

De plus, la construction d'une méthode, même grossière du point de vue de la complexité, peut être mise en place. Pour prouver qu'une méthode de recherche donnée trouve toujours les fausses pièces, un raisonnement s'appuyant sur une logique intuitive suffit la plupart du temps (disjonction de cas par exemple).

La question la plus délicate semble donc être celle de l'optimalité. En effet, elle soulève le problème du choix d'un critère d'optimalité. Deux critères peuvent être utilisés : la complexité au pire et la complexité en moyenne.

La complexité au pire est le nombre de pesées qu'effectue un algorithme dans le pire des cas. Autrement dit, la complexité au pire pour un algorithme donné est le nombre maximum d'étapes (ici les pesées) qu'il effectue, maximum calculé sur l'ensemble des instances d'une taille donnée (ici pour un nombre fixé de pièces). La complexité en moyenne est la moyenne des nombres d'étapes effectuées par l'algorithme, moyenne calculée sur l'ensemble des instances d'une taille donnée (généralement on suppose les instances équiprobables). Elles s'expriment toutes les deux en fonction de la taille de l'instance, c'est-à-dire ici, en fonction du nombre de pièces. Ici, nous ne nous intéresserons pas au problème de l'optimalité pour la complexité en moyenne. Cela ne veut pas dire que cette question n'est pas intéressante du point de vue mathématique ou didactique.

11 On pourra consulter, entre autres, (Tosic, 1983), (Pyber, 1986) ou (Aigner, Li, 1997).

Malgré ce choix, nous le verrons dans l'analyse mathématique du problème, ce sont les questions de complexité et d'optimalité qui peuvent demander le plus d'outils mathématiques (récurrence, puissances, logarithmes,...).

3.2 *Etude mathématique du problème*

Le problème des pesées est un problème d'optimisation. La résolution de tels problèmes repose sur l'étude de deux sous-problèmes :

- P_M : Construire une méthode de recherche de la fausse pièce pour un problème donné
- P_O : Prouver l'optimalité de la méthode.

On distinguera parmi les stratégies de résolution de P_O celles qui relèvent d'un argument d'optimalité locale et celles qui relèvent d'un argument d'optimalité globale.

P_M permet de travailler sur la condition suffisante et P_O sur la condition nécessaire. Parfois les deux sous-problèmes sont traités simultanément : on construit une méthode et sa construction prouve son optimalité. Nécessairement cela s'appuiera sur un argument d'optimalité locale. Pour le problème d'optimalité P_O on peut définir le problème dual P_O* : Si l'on fixe un nombre de pesées *p*, combien de pièces peut-on traiter avec *p* pesées ? Parfois le passage à P_O* pourra s'avérer fructueux pour répondre à P_O.

On s'intéresse maintenant aux différentes stratégies envisageables en fonction des variables suivantes : le nombre de fausses pièces et les informations sur leur poids :
— Le nombre de fausses pièces peut être connu (*1*, *3*, compris entre *0* et *2*...) ou non.
— Le poids de la fausse pièce peut être connu (plus lourd par exemple) ou non.

Dans notre étude on se limitera aux cas où le nombre de fausses pièces n'excède pas un : c'est-à-dire lorsqu'il y a exactement une fausse pièce ou au plus une fausse pièce. On s'intéresse donc aux quatre problèmes suivants :

- $P_{1,+}$: 1 fausse pièce, plus lourde.
- $P_{1,+/\_}$: 1 fausse pièce, plus lourde ou plus légère.
- $P_{0/1,+}$: 0 ou 1 fausse pièce, plus lourde.
- $P_{0/1,+/\_}$: 0 ou 1 fausse pièce, plus lourde ou plus légère.

On présentera ici les stratégies et les résultats mathématiques pour $P_{1,+}$. Pour les autres cas on pourra consulter (Modeste, 2009).

Stratégies pour $P_{1,+}$:

Dans une démarche de recherche on est souvent amené à étudier des petits cas afin d'essayer de mieux comprendre l'enjeu du problème et de pouvoir, par exemple, émettre des conjectures. Cela donne lieu à la stratégie suivante :

• Stratégie expérimentale S_{EXP} :

On traite les petits nombres de pièces (par exemple par énumération de cas). Cela peut aboutir à des « généralisations » formulées comme conjectures. D'autre part, pour un nombre de pièces fixé, si l'on a traité tous les cas plus petits, on peut énumérer toutes les pesées possibles (de manière plus ou moins organisée) et déduire la méthode optimale pour ce cas. Cette méthode s'appuie sur un argument d'optimalité locale. Elle devient très vite fastidieuse mais permet d'avoir un grand nombre d'exemples sur lesquels s'appuyer pour faire des conjectures ou les invalider.

Pour répondre à ces conjectures, il convient de formaliser ce qu'est une pesée. Un premier point de vue est qu'effectuer une pesée, c'est prendre deux ensembles de pièces de même taille et de les comparer pour savoir lequel des deux est le plus lourd. Cette vision peut donner lieu à la stratégie S_{DICHO}.

• Stratégie de dichotomie S_{DICHO} :

Pour traiter n pièces, on compare « la moitié » des pièces sur chaque plateau. Le déséquilibre permet de garder $[n/2]$ pièces à tester[12], on réitère ensuite le processus. Si n est impair, un équilibre de la balance est possible et indique que la fausse pièce est celle laissée de côté. Cela donne un algorithme en $[log_2(n)]$ pesées au pire.

Cette stratégie ne conduit pas à un algorithme optimal, il suffit pour s'en convaincre de regarder le problème pour *9* pièces. Avec S_{DICHO} il faudra *3* pesées alors que S_{EXP} donne une méthode en *2* pesées. L'optimalité de cet algorithme peut reposer sur la conception erronée : plus on pèse de pièces à la fois, plus on est efficace (argument d'optimalité locale). En fait, cette méthode ne tient pas assez compte du fait qu'une pesée donne aussi de l'information sur l'ensemble des pièces non pesées.

En effet, effectuer une pesée, c'est faire trois tas : deux de même taille A et B que l'on compare, et un tas C que l'on laisse de côté. Trois résultats sont alors possibles :
— A est plus léger que B et alors A contient la fausse pièce.
— A est plus lourd que B et alors B contient la fausse pièce.
— A et B sont de même poids et c'est le tas C qui contient la fausse pièce.

En s'appuyant sur cette observation, on peut mettre en œuvre la stratégie suivante.

• Stratégie de trichotomie S_{TRICHO} :

On divise le tas de pièces restantes en trois paquets de « même taille » i.e. tels que

$$|A| = |B| \text{ et } \big|\,|A|-|C|\,\big| \leq 1.$$

Pour n pièces, avec $3^{k-1} < n \leq 3^k$, cet algorithme demande k pesées, autrement dit $[log_3(n)]$ pesées ou encore cette méthode permet de traiter en k pesées jusqu'à 3^k pièces.

Proposition. Cet algorithme est optimal pour la complexité au pire[13].

Une autre façon de concevoir une pesée, est de n'interpréter le résultat que comme un équilibre ou un déséquilibre, sans tenir compte de quelles pièces sont sur le plateau le plus lourd. Cela peut conduire à la stratégie :

• Stratégie par étalon S_{ETAL} :

On utilise un ensemble E de pièces identifiées comme vraies pour tester un ensemble A de pièces inconnues par comparaison. Soit les pièces de A sont vraies et elles s'ajoutent à l'ensemble E des pièces talons, soit on sait que A contient la fausse pièce et il ne reste plus qu'à la retrouver. Pour ce faire on peut, par exemple, identifier des bonnes pièces dans A grâce à E et les enlever jusqu'à n'en avoir plus qu'une.

• Stratégie de passage par un autre problème S_{PROB} :

12 On notera entre crochets *[ ]* la partie entière inférieure.

13 Pour une preuve, voir Modeste (2009).

On se ramène à des problèmes déjà traités ou qui paraissent plus simples (voire on déduit de leur optimalité celle de la méthode construite).

3.3 *Analyse de la situation*

Une Situation de Recherche en Classe

Pour observer la construction d'algorithmes, leur preuve ainsi que l'étude de leur complexité et de leur optimalité, il est intéressant de mettre les apprenants en situation de résolution de problème en groupe, face à un problème pour lequel ils peuvent adopter une démarche proche de celle du chercheur (essais-erreurs, conjectures, preuve, simplification du problème,...). Les Situations de Recherche en Classe, dites SiRC, développées et étudiées par l'équipe Maths à Modeler semblent se prêter particulièrement bien à ce type d'études et le problème des pesées nous paraît complètement adapté à ce cadre. D'après les travaux de (Grenier, Payan, 2003), (Godot, Grenier, 2004) et (Cartier et al., 2006), les SiRC se caractérisent par les aspects suivants :

a) Le problème abordé doit être proche de la recherche actuelle.

b) La question initiale doit être facile d'accès et ne doit pas être formalisée en termes mathématiques pour faciliter l'entrée dans le problème.

c) Des stratégies d'approche simples doivent exister pour permettre d'entrer dans une démarche de recherche.

d) De nombreuses stratégies de recherche peuvent être mises en place, les méthodes de résolutions ne sont pas désignées.

e) Une question résolue peut amener à se poser d'autres questions : il n'y a que des critères de fin locaux.

Variables de la situation

On s'intéresse ici aux différentes variables de la situation, et notamment à celles des deux types suivants :

- Les variables de recherche
- Les variables didactiques

Les *variables de recherche* sont les variables de la situation qui sont à disposition de l'élève pour organiser son travail de recherche.

> *L'élève se retrouve en effet en position de « chercheur » mais aussi en situation de « gestionnaire » de sa propre recherche : c'est lui qui choisit et modifie les valeurs des variables de recherche.*
>
> (Cartier et *al.*, 2006)

> *Elles déterminent la compréhension et l'intérêt de la question, son ouverture à de nouvelles questions, l'élargissement des stratégies de recherche, les possibilités de transformation du problème (modélisation).*
>
> (Grenier & Payan, 2003)

Les *variables didactiques* sont celles parmi les variables de la situation, dont les modifications peuvent provoquer un changement de stratégie chez l'apprenant (Brousseau, 1982). Dans la situation étudiée, nous avons repéré les variables didactiques suivantes :

Parmi les variables du problème :

- Le nombre de fausses pièces :

Le problème peut devenir relativement complexe, si l'on s'autorise plus d'une fausse pièce : par exemple le problème à deux fausses pièces n'est à ce jour pas complètement résolu pour la recherche en mathématiques. Pour le problème où l'on sait qu'il y a au maximum une fausse pièce, pour n'importe quelles valeurs des autres variables, on peut savoir si le nombre de fausses pièces est 0 ou 1 en une ou deux pesées ce qui favorisera une stratégie du type S_{PROB}.

Au contraire, si l'on sait qu'il y a exactement une fausse pièce, la question de rechercher une fausse pièce qui n'existe peut-être pas n'interfère plus et des stratégies comme S_{DICHO} ou S_{TRICHO} pourront être mises en place directement.

• Le poids des fausses pièces ou plutôt les informations sur leur poids :

Si l'on sait qu'elles sont plus lourdes (ou plus légères, c'est le même problème), ou si l'on ne le sait pas, les stratégies envisageables vont sensiblement différer. Dans le problème $P_{1,+/\_}$ par exemple : Ne pas connaître le poids de la fausse pièce pousse à rechercher cette information avant de rechercher la fausse pièce, c'est-à-dire renforce la stratégie S_{PROB}.

Ou au contraire, on peut être amené à penser que le poids de la fausse pièce ne peut être déterminé et se tourner vers une stratégie de type S_{ETAL} , qui n'utiliserait par exemple que l'information que le poids de la fausse pièce est différent de celui d'une vraie.

Si l'on connaît à l'avance le poids de la fausse pièce ($P_{1,+}$), la première de ces stratégies n'a pas de sens. La deuxième, bien que pouvant aboutir à une méthode de recherche acceptable, sera effacée par des stratégies moins complexes et utilisant toute l'information sur le poids de la fausse pièce (S_{DICHO}, S_{TRICHO} …).

• Le nombre de pièces :

Lorsque l'on veut résoudre un cas particulier du problème avec un nombre de pièces fixé, ce nombre peut influencer les stratégies. Par exemple, un petit nombre de pièces permet la stratégie S_{EXP} mais dès que l'on s'interroge sur un grand nombre de pièces, ce type de stratégie devient impossible à mettre en œuvre (notamment une énumération de tous les cas devient très vite fastidieuse). Ou encore pour les problèmes $P_{1,+}$ ou $P_{0/1,+}$, un nombre pair de pièces poussera à partager les pièces en deux tas et à toutes les peser ensemble et favorisera S_{DICHO} . De même si l'on cherche sur un cas à *3k+1* ou *3k+2* pièces, S_{TRICHO} peinera à apparaître face à d'autres stratégies.

Ces trois variables, nombre de fausses pièces, poids des fausses pièces et nombre de pièces sont des variables de recherche de la situation.

Parmi les autres variables de la situation nous avons distingué deux variables didactiques :

• L'énoncé du problème : La présentation du problème et l'ordre dans lequel sont présentés les différentes variantes peut avoir une influence sur les méthodes de résolutions choisies pour chacun d'eux, de même que le choix des exemples préliminaires qui peuvent être proposés.

• Le matériel disponible : La présence d'une balance Roberval et de « pièces » peut avoir une influence sur la dévolution de l'énoncé du problème et sur la méthodologie de recherche. Cela peut favoriser les stratégies de type S_{EXP}, mais cela peut aussi entraver le passage à des grands nombres de pièces et créer des difficultés à énumérer tous les cas possibles pour un algorithme donné.

Hypothèses sur les stratégies

Le calcul de la complexité des algorithmes proposés sera, d'après nous, souvent interrogé. Mais il nous semble que cela aboutira plus rarement car cela met en jeu plus d'outils mathématiques. En résumé nous retenons les hypothèses suivantes :

H_1 : Pour $P_{1,+}$, la stratégie naturelle est S_{DICHO} . Les stratégies S_{EXP} et S_{TRICHO} viennent plus tard.

H_2 : L'optimalité de S_{TRICHO} sera peu abordée et lorsque ce sera le cas ce sera très majoritairement avec un argument d'optimalité locale.

H_3 : Pour les autres variantes, la stratégie naturelle est S_{PROB} pour revenir à $P_{1,+}$.

H_4 : L'optimalité sera peu abordée et lorsque ce sera le cas elle sera déduite de celle de $P_{1,+}$.

Cadre d'expérimentation

Nous avons expérimenté cette situation dans deux classes (chacune d'une vingtaine d'étudiants répartis en cinq groupes) :

— Un cours de mathématiques de la formation des PE_2, à l'Iufm de Créteil.

— Option « Jeux combinatoires et raisonnements mathématiques » proposé à l'université Joseph Fourier et ouvert aux étudiants de première et deuxième années des licences scientifiques.

Les séances se sont déroulées suivant un même plan :

1) Présentation du problème et constitution des groupes.

2) Recherche en groupe sur le problème donné. Ce moment a duré entre 1 heure et 1 heure et demi. Pendant tout ce temps, les gestionnaires de situation ont circulé dans les groupes pour suivre leurs avances et répondre aux questions.

3) Préparation de la présentation orale. Chaque groupe a présenté ensuite son travail à la classe (résultats, conjectures, questions en suspend...).

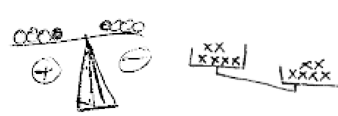
4) Institutionnalisation.

L'étude a porté sur les brouillons des étudiants, leurs présentations orales et des enregistrements vidéos de certains groupes en travail de recherche.

3.4 *Résultats de l'expérimentation*

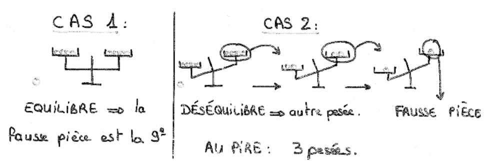
Stratégies utilisées

- Nous avons retrouvé la plupart des stratégies présentes dans l'analyse mathématique. Tous les groupes ont utilisé une représentation de la balance (plus ou moins schématique) pour symboliser une pesée.



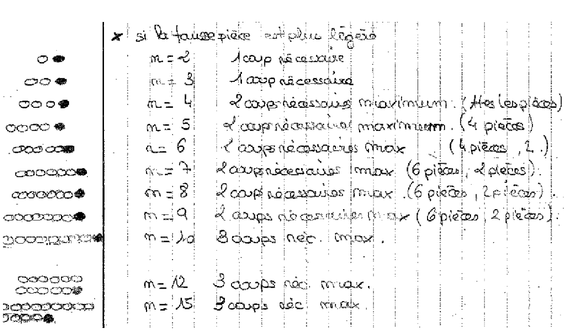
Représentations d'une pesée

- Que ce soit pour $P_{1,+}$, $P_{1,+/\_}$ ou $P_{0/1,+}$, la stratégie la plus présente est S_{DICHO} ou ses variantes. Elle est développée dans plus de la moitié des productions et évoquée dans toutes. Les groupes filmés évoquent clairement cette stratégie dès le début. Cela semble confirmer les hypothèses sur les stratégies. Cette stratégie donne souvent lieu à un algorithme qui est parfois énoncé en langage courant, d'autres fois représenté par un schéma (dans ce cas il est généralement présenté sur un exemple).



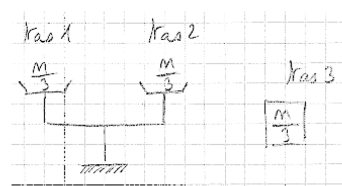
La stratégie de dichotomie.

• On retrouve aussi beaucoup la stratégie S_{EXP}, elle peut être utilisée dans deux buts : chercher la méthode optimale ou évaluer la complexité d'une méthode. On l'a clairement identifiée dans la moitié des groupes observés. Elle permet à ces groupes de valider ou d'invalider des méthodes, d'en conjecturer d'autres et de faire des hypothèses sur le nombre de pesées. On retrouve très clairement cette stratégie dans les échanges oraux, où elle est utilisée pour étudier S_{TRICHO} et conjecturer la complexité de l'algorithme construit. Les nombres de pièces plus petits sont, par exemple, utilisés pour traiter les cas suivants. Cela semble aider à repérer le lien de récurrence existant.



La stratégie expérimentale.

• Une autre stratégie que l'on retrouve souvent est la stratégie est S_{TRICHO}. On la retrouve clairement chez quatre groupes et elle semble être évoquée chez deux autres. Elle donne lieu à un algorithme, parfois difficile à préciser selon le nombre de pièces. Comme pour S_{DICHO} l'algorithme est présenté en langage courant ou par un schéma. L'optimalité de cet algorithme est souvent conjecturée mais rarement prouvée. Elle est clairement évoquée dans les discussions des groupes et apparaît dans les deux cas (parfois assez tôt).



La stratégie de trichotomie.

• S_{PROB} est aussi beaucoup utilisée chez les groupes qui ont traité plusieurs problèmes : pour résoudre $P_{1,+/\_}$ ou $P_{0/1,+}$, ils cherchent à se ramener au problème $P_{1,+}$ qui leur semble peut-être plus simple. Sur les 7 ou 8 groupes qui ont traité plusieurs problèmes, quatre ont eu clairement recours à cette stratégie.

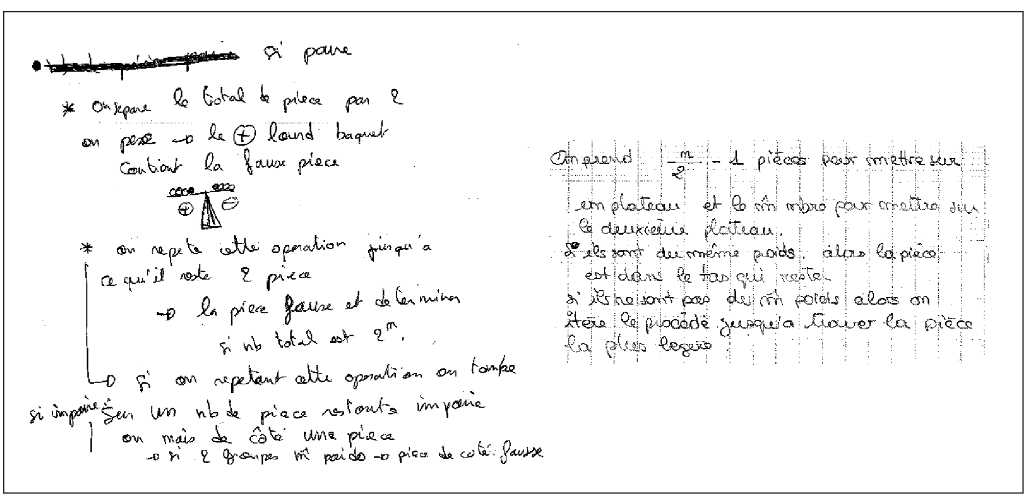
• On retrouve aussi des variantes de ces stratégies, par exemple mélangeant S_{TRICHO} et S_{DICHO}, ou d'autres encore, proposant des partages des pièces dans d'autres proportions. On retrouve aussi dans une production la stratégie S_{ETAL}.

Il semble donc que la stratégie la plus naturelle soit S_{DICHO} et que S_{EXP} puisse être une bonne stratégie pour rejeter la dichotomie. Viennent alors les autres stratégies et notamment S_{TRICHO} . Il apparaît aussi que le passage par $P_{1,+}$ soit un recours naturel dans la résolution des autres variantes proposées. La majorité des stratégies évoquées dans l'analyse mathématique sont présentes et leur ordre prévu dans l'analyse a priori semble se confirmer.

Algorithmes

• Les étudiants produisent des algorithmes sous deux formes principalement : en langage courant et sous forme de schémas (arbres ou succession de pesées). Dans le deuxième cas l'algorithme est souvent présenté sur un exemple. Il est alors difficile de savoir si l'algorithme a été généralisé à un nombre quelconque de pièces ou non. La construction d'algorithmes apparaît très nettement dans les échanges oraux à divers moments. Les descriptions sont très claires et l'on retrouve des termes tels que « la même procédure sera appliquée » ou « jusqu'à ce que tu tombes sur plus qu'une pièce » qui mettent en évidence une itération. Les algorithmes construits sont tous des algorithmes valides de recherche de la fausse pièce, il semble que la question de produire un ou plusieurs algorithmes pour le problème ne pose pas de grosses difficultés.

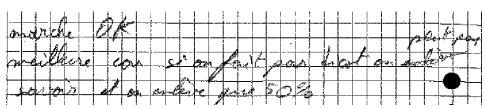
• Se pose la question de la preuve de ces algorithmes. Elle transparaît très peu dans les documents écrits des étudiants mais on peut constater que dans la majorité des productions, tous les cas possibles pour la fausse pièce sont pris en compte. On constate aussi l'utilisation (en apparence) d'invariants pour traiter moins de cas (symétrie, cas similaires, retour à un nombre de pièces déjà traité...). Concernant la preuve des algorithmes, l'étude des échanges oraux nous en dit un peu plus. En effet, la preuve des algorithmes transparaît à de nombreux endroits et de manière très nette. Elle a lieu en même temps que la construction des algorithmes, par disjonction des différents cas : « on a soit *2k* pièces soit *2k+1* pièces » ou « Si c'est ni dans l'un ni dans l'autre... » par exemple.


si paire
* On separe le total de piece par 2
on pese → le ⊕ lourd baquet contient la fausse piece
* on repete cette operation jusqu'a ce qu'il reste 2 piece
→ la piece fausse et determiner si nb total est 2^n.
→ si on repetant cette operation on tombe
si impaire sur un nb de piece restants impaire
on mais de côté une piece
→ si 2 groupes m̂ poids → piece de côté fausse

On prend $\frac{n}{2} - 1$ pièces pour mettre sur un plateau et le m̂ nbre pour mettre sur le deuxième plateau.
Si ils sont du même poids, alors la pièce est dans le tas qui reste.
si ils ne sont pas du m̂ poids alors on itère le procédé jusqu'à trouver la pièce la plus legère.

Deux algorithmes en langage courant.

• Les algorithmes sont parfois comparés entre eux, sur des exemples ou sur un nombre quelconque de pièces, ce qui permet de rejeter, pour un algorithme trouvé, la conjecture selon laquelle il est optimal. La stratégie S_{EXP} développée par plusieurs groupes peut aussi relever de la comparaison d'algorithmes. Les comparaisons d'algorithmes ne sont faites que sur des exemples ou par des arguments d'efficacité locale (sur une pesée). On voit clairement ces comparaisons dans les échanges des étudiants.



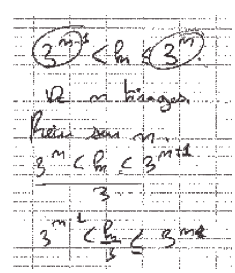
Comparaison des stratégies de trichotomie et dichotomie.

• La complexité, ou plutôt le nombre de pesées, est questionnée par plusieurs groupes, souvent sans succès. Cependant quelques groupes émettent des hypothèses sur ce nombre de pesées, souvent à l'aide de S_{EXP}. Certains vont jusqu'à produire une preuve de la complexité de leur méthode que nous préciserons dans la partie concernant la preuve. Dans les échanges, la complexité est questionnée et l'on retrouve des calculs (ou des tentatives de calcul) de cette complexité. Par exemple, « On peut calculer le nombre de pesées mais ça dépend de *n* », évoque clairement la complexité. L'optimalité est aussi questionnée et ce sont plutôt les échangent oraux qui nous en informent. De manière générale, il semble que l'enjeu d'optimisation ait vécu dans l'activité des groupes. Le critère de complexité au pire a été sujet à discussion dans les groupes. Les questions d'optimalité ont aussi donné lieu, par exemple, à des discussions sur le passage d'un problème à un autre et la conservation de l'optimalité, qui sont des questions importantes de l'algorithmique.

Démarche de preuve

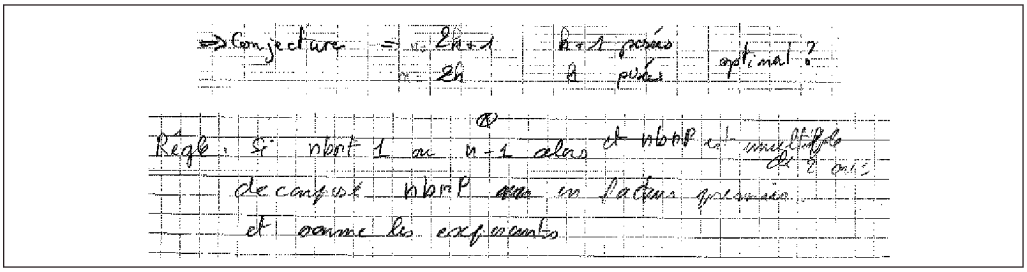
D'après les résultats présentés jusqu'ici, un enjeu de preuve semble être présent dans l'activité des étudiants. Elles peuvent être des preuves d'algorithmes, des preuves de complexité ou des preuves d'optimalité. Voyons plus précisément quels sont les éléments constitutifs de cette activité.

• D'une part, quelques preuves sont présentes dans les productions écrites, la difficulté est qu'elles ne sont pas toujours très formalisées et il est difficile de les identifier clairement. L'étude des échanges oraux nous a permis de mieux repérer ces preuves, nous les avons déjà évoquées dans les paragraphes précédents.



Une preuve par récurrence.

• D'autre part, des hypothèses, des conjectures et des questions sont présentes sur les productions. Elles sont significatives d'une démarche de preuve, mais sont, pour beaucoup, formulées oralement. Citons comme exemples : « Ah ouais, c'est les plus équilibrés en fait, je pense... », « Mais 5 tu l'auras pas avant 90. » ou « Mais pourtant c'est la meilleure méthode. Enfin on suppose que. »...


⇒ Conjecture ⇒ n = 2k+1   k+1 pesées   optimal ?
n = 2k   k pesées

Règle : Si nbre 1 ou n-1 alors ~~et nbre est multiple de 2 ou~~
décompose nbre ~~en~~ en facteurs premiers
et somme les exposants

Exemples de conjectures.

• On retrouve aussi l'expression d'une nécessité de recourir à la preuve dans les discussions, par exemple dans un groupe : « Je pense quand même qu'il y a une raison un peu plus... logique », « Ben allez, faut le prouver. », « Une récurrence pour prouver quoi ? » ou « Si il te la prouve pas comment tu vas le croire ? »... Ou dans un autre groupe : « Je pense plutôt qu'on est pas assez rigoureux. » ou « On va faire un raisonnement en probabilités... ».

• L'utilisation de contre-exemples permet aussi de déceler une activité de preuve. Cependant, il est difficile de voir sur les productions écrites si des contre-exemples sont utilisés. On peut en repérer mais ils sont sûrement plus souvent évoqués à l'oral. Les comparaisons de méthodes entre elles constituent aussi des contre-exemples à l'optimalité de l'une des deux. S_{EXP} permet de construire de nombreux exemples qui peuvent servir de contre-exemples à l'optimalité de certaines méthodes. C'est ce que semble faire l'un des groupes : « Si tu fais *4*, *4* et *2*. Si t'enlèves les *2*, en gros t'auras, t'auras perdu une pesée quoi. » pour rejeter une stratégie de découpage en *2*, *2*, *1*.

Comparaison des deux publics

Même si nous avons présenté ensembles les résultats des deux expérimentations, il est nécessaire de discuter des différences et des points communs entre les deux publics observés.

• Concernant les stratégies mises en œuvre, on rencontre chez les deux publics les stratégies S_{DICHO} et S_{EXP} de façon courante. La stratégie S_{PROB} est aussi commune aux deux expérimentations. Cependant, les PE_2 n'ont pas soulevé S_{TRICHO} que l'on ne retrouve que chez les étudiants de licences scientifiques.

• Concernant les algorithmes, leur construction est mise en œuvre par les deux publics mais le passage au cas général semble plus difficile pour les PE_2 et tous ne l'ont pas traité. Comme la preuve de ces algorithmes se fait généralement en parallèle de leur construction, on la retrouve aussi chez les deux publics. L'enjeu d'optimisation semble bien présent dans les deux expérimentations et la notion de complexité au pire paraît assez naturelle pour tous les étudiants. Des tentatives de calcul de cette complexité existent aussi bien dans les deux expérimentations, mais il semble qu'elles n'aboutissent pas chez les étudiants de PE_2. Concernant l'enjeu de preuve, on le retrouve à des degrés différents dans les deux cas. Cependant des preuves d'optimalité ne sont présentes que chez les étudiants de L_1 et L_2.

En conclusion

On constate, sans surprise, que le rapport à la preuve des deux publics auprès desquels nous avons expérimenté est différent. Ces résultats, concernant la mise en place de preuves semblent aller dans le sens de ceux de Schuster qui a étudié l'introduction de problèmes d'optimisation combinatoire dans l'enseignement secondaire (Schuster, 2004). Il constate que la construction d'algorithmes peut vivre en classe dès le grade 9 et que la preuve n'est réellement mise en œuvre qu'aux grades 11 et 12 de manière autonome [14]. Il serait pertinent d'étudier le rapport à la preuve des différents publics concernés par cette situation.

Cette situation a permis à certains groupes de mettre en œuvre la construction d'algorithmes et de questionner certains aspects de l'algorithme (preuve, complexité, optimalité) qui sont absent des programmes et manuels. Ces aspects sont du côté objet de l'algorithme. Concernant les hypothèses sur les stratégies, elles se sont vérifiées ici. La construction et la preuve d'algorithmes ont été effectuées par une majorité d'étudiants. La question de l'optimalité a été rarement résolue et les groupes qui l'ont traité ont privilégié l'optimalité locale. Pour beaucoup d'étudiants, traiter l'optimalité demanderait sûrement un temps plus long.

Cette situation semble donc porteuse d'un potentiel pour la manipulation de l'algorithme en tant qu'objet. Il semble donc important de poursuivre son étude sur une période plus longue (5 à 6 séances de 2 heures par exemple pour une SiRC classique). Cette étude pourrait être mise en œuvre avec différents publics (du primaire à l'université et en formation d'enseignants) afin d'étudier le rapport de chacun à l'algorithme et de comprendre l'évolution de ce rapport au fil de la situation.

14 Le grade 9 correspond à la tranche d'âge 15-16 ans et les grades 11 et 12 à la tranche 17-19 ans.

Conclusion

L'algorithme peut être vu sous différents aspects, dont certains ont un potentiel réel pour l'enseignement des mathématiques et en particulier pour l'apprentissage de la preuve. Malheureusement, ces aspects semblent absents des programmes et manuels de l'enseignement secondaire en France et sont donc difficiles à exploiter par les enseignants. En effet, l'algorithme ne joue dans les programmes et les manuels que le rôle d'un outil. Pourtant, il semble que l'on peut questionner les différents aspects de ce concept dans certaines situations, et tenter de faire apparaître l'algorithme comme un véritable objet mathématique.

En proposant le « problème des pesées » à des étudiants, sous la forme d'une situation de recherche, nous avons pu constater que la construction d'algorithmes, leur preuve et leur étude pouvaient vivre en classe. Cependant, l'étude de leur complexité reste une question difficile. La complexité est une notion spécifique à l'algorithmique (Knuth, 1985) et son étude passe nécessairement par l'objet algorithme. Cette notion pourrait trouver sa place au sein d'un enseignement de mathématiques et même redonner du sens à certains des objets enseignés.

L'algorithme est sur le point de prendre une place plus grande dans l'enseignement français et il est nécessaire de suivre cette évolution et de l'accompagner.

Références

Aigner M. , Li A. (1997), Searching for counterfeit coins, *Graphs and Combinatorics*, 13(1), 9-20.

Bouvier A. , George M. , Le Lionnais F. (2005), *Dictionnaire des mathématiques*, Puf.

Brousseau G. (1982), Les objets de la didactique des mathématiques, *Contributions à la seconde école d'été de didactique des mathématiques*, pp 10-33.

Cartier L. (2008), *Le graphe comme outil pour enseigner la preuve et la modélisation*, Thèse de l'Université Joseph Fourier. http://tel.archives-ouvertes.fr/tel-00416598/fr/.

Cartier L., Godot K. , Knoll E. , Ouvrier-Buffet C. (2006), Les situations-recherche : Apprendre à chercher en mathématiques, *Annales du colloque Espace Mathématique Francophone*, Sherbrooke, Canada.

Chabert J.-L. (1994), *Histoire d'algorithmes - du caillou à la puce*, Belin.

Chevallard Y. (1998), Analyse des pratiques enseignantes et didactique des mathématiques : L'approche anthropologique, *Actes de l'U.E. de la Rochelle*.

Douady R. (1986), Jeux de cadres et dialectique outil-objet, *Recherches en didactique des mathématiques*, 7(2), 5-31.

Godot K., Grenier D. (2004), Research situations for teaching : a modelization proposal and examples, *Proceedings of the 10th International Congress for Mathematics Education, ICME 10*, Copenhague.

Grenier D., Payan C. (2003), Situations de recherche en « classe » essai de caractérisation et proposition de modélisation, *Les cahiers de laboratoire Leibniz*, 92, http://www-leibniz.imag.fr/Les-Cahiers/Cahiers2003.html.

Kahane J.-P. (2000), *Commission de réflexion sur l'enseignement des mathématiques, informatique et enseignement des mathématiques*, http://smf.emath.fr/Enseignement/CommissionKahane/.

Knuth D. E. (1985), Algorithmic thinking and mathematical thinking, *The American Mathematical Monthly*, 92(1), 170-181.

Modeste S. (2009), *La place et le rôle de l'algorithme dans l'enseignement : vers un apprentissage de la preuve*, Mémoire de Master 2, Université Joseph Fourier, Grenoble, www.mathsamodeler.net/~smodeste.

Nguyen C. T. (2005), Étude didactique de l'introduction d'éléments d'algorithmique et de programmation dans l'enseignement mathématique secondaire à l'aide de la calculatrice, Thèse de l'université Joseph Fourier, Grenoble.

Pyber L. (1986), How to find many counterfeit coins? *Graphs and Combinatorics*, 2(1), 173-177.

Ravel L. (2003), *Des programmes à la classe : étude de la transposition didactique interne. Exemple de l'arithmétique en terminale S spécialité mathématique*, Thèse de l'université Joseph Fourier, Grenoble.

Schuster A. (2004), About traveling salesmen and telephone networks - combinatorial optimization problems at high school, *ZDM,* 36(2), 77-81.

Tosic R. (1983), Two counterfeit coins, *Discrete Mathematics*, 46(3), 295-298.