Savoirs et situations dans les premiers apprentissages en programmation et en algorithmique

Jean-baptiste Lagrange et Janine Rogalski
LDAR, Université Paris-Diderot

Séminaire National de Didactique des Mathématiques - Paris, 6-7 novembre 2015

Nous sommes reconnaissants aux organisateurs d'avoir accepté notre proposition d'intervention au séminaire national.

Nous sommes particulièrement heureux d'intervenir au cours de cet après-midi entre une conférence et une table ronde portant sur les **interactions entre mathématiques et informatique dans l'enseignement.**

Notre intervention ne portera pas directement sur ce sujet mais plutôt sur des pré-conditions pour que les élèves puissent profiter de ces interactions, à savoir qu'ils réussissent leurs premiers apprentissages dans un domaine qu'on appelle dans certains contextes programmation et dans d'autres algoritmique.

Plan

1. Motivation et Hypothèses
2. Acquis des recherches en psychologie de la programmation
3. Approche didactique

Difficultés similaires

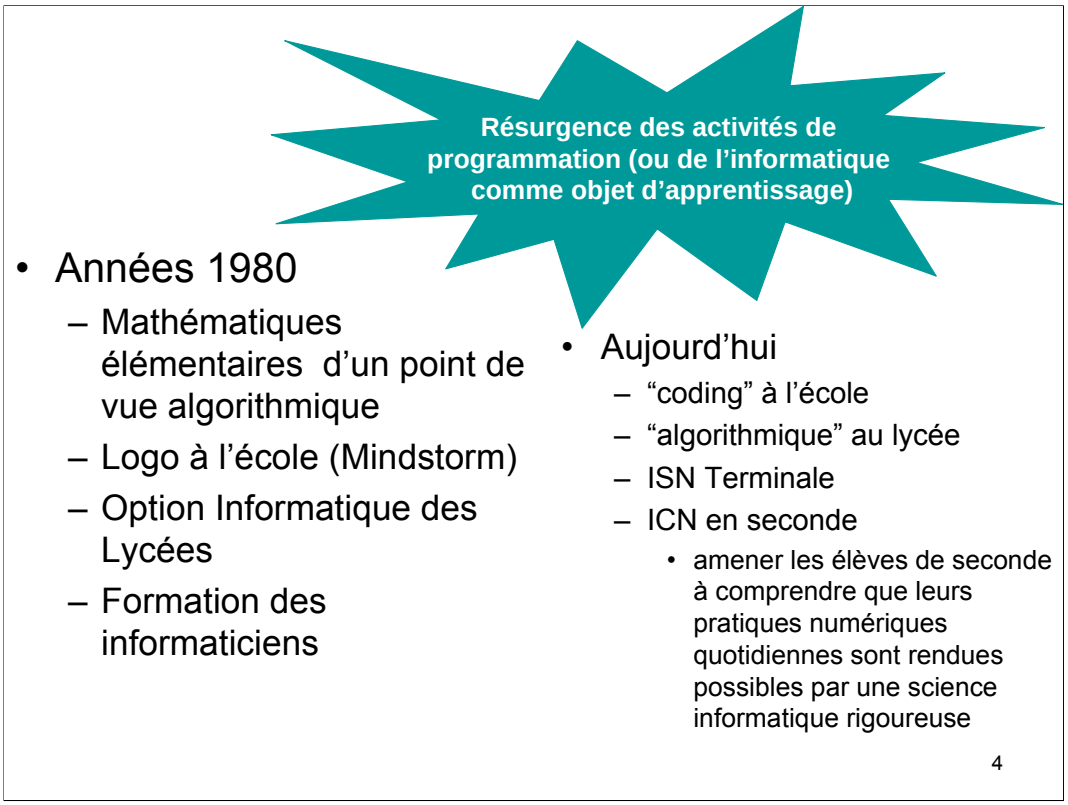
Algorithmique au lycée (2010)

↔

Option informatique (1985)

1. Représentations du traitement par la machine

2. Conditions et valeurs Booléennes



Pour préciser notre motivation il faut d'abord situer deux contextes qui présentent certaines similarités.

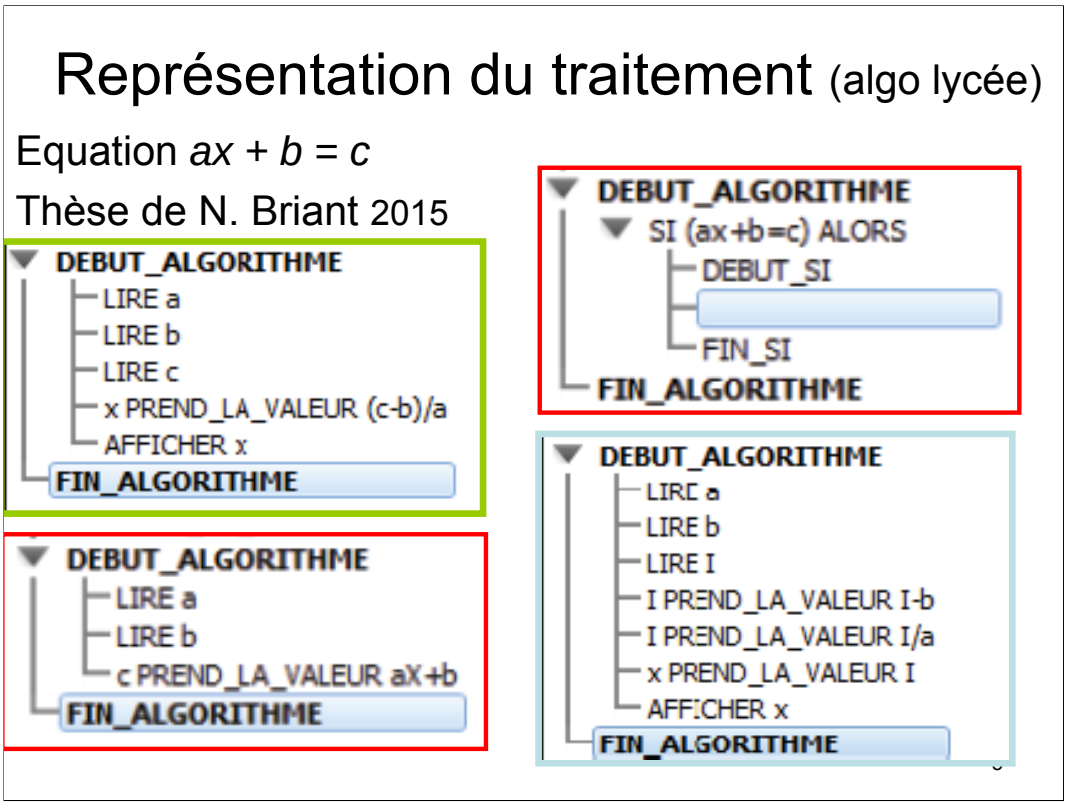
D'abord celui des années 1980 où l'ordinateur devient un objet assez largement accessible, mais où il ne permet pas encore de faire grand chose à part de la programmation.

Les activités de programmation sont vues comme pouvant contribuer aux apprentissages dans les domaines scientifiques existants, les mathématiques en premier lieu. Le livre d' Arthur Engel math.. parait en allemand en 1977 et est traduit en français au début des années 1980. Le livre de Papert Mindstorm sur les apports de la programmation en Logo paraît aussi à la même époque. Cela se traduit par quelques pratiques et groupes de recherche dans les IREMs, sans toutefois qu'il y ait un impact sur le curriculum, au moins en France.

En revanche, l'institution met en place un enseignement centré sur l'informatique comme champ scientifique, l'option informatique des lycées. Elle se généralise très vite sous la pression des familles. Il s'agit d'offrir un nouveau champ scientifique, tout en soulignant les liens qu'il entretient avec les champs scolaires existants. Il ya une forte dimension programmation dans le descriptif de l'option. Le volume horaire est de 2,5 h par semaine, avec le thème central des **méthodes de construction de programmes** qui occupe la majorité du programme.

Tout cela s'arrête plutôt brusquement au tournant des années 1990. Les raisons sont complexes,. Une peut cependant être soulignée: c'est l'irruption des progiciels et aussi des applications dédiées comme le calcul formel et la géométrie dynamique qui donnent l'impression que l'enseignement tirera plus facilement profit de ces nouvelles applications que des activités de programmation.

Je ne m'étends pas sur la situation actuelle. On peut parler comme GL Baron d'un phénomène de résurgence, d'une prise de conscience de ce que les activités avec des progiciels ou des applications dédiées, d'une part ne sont pas si faciles à mettre en œuvre, et d'autre part, ne conduisent pas non plus facilement à la compréhension des principes et méthodes qui sous-tendent leur conception.



Comme vous le savez, les ingénieries mises en place par N. Briant visent à faire travailler les élèves sur les équations d'un point de vue objet via des activités de programmation.

Lors de son exposé, elle a signalé deux difficultés qui lui ont semblé mériter qu'on s'y attarde.

La première, concerne la programmation de la résolution d'une équation générique du premier degré.

On attend une solution comme encadrée en vert, où la solution générique est calculée par une formule, où toutefois le cas a=0 n'est pas traité, ce qui nécessiterait de construire une alternative.

C'est ce que produisent certains élèves,

Mais on rencontre aussi des productions comme les deux encadrées en rouge qui témoignent d'une incompréhension du langage, de la séquencialité, des variables, etc.. où les élèves essaient simplement de traduire l'équation avec la syntaxe du langage, soit par une affectation (en bas à gauche), soit par une condition (en haut à droite).

On rencontre aussi très souvent le type de résolution en bleu à droite, où la variable I prend les valeurs successives du second membre lors d'une résolution en papier/crayon. Comme le note N. Briant l'élève code les actions successives lors de cette résolution. Ce n'est certes pas gênant pour le fonctionnement: le programme est syntaxiquement correct et donne bien la solution, cependant ça reste distant de de la solution attendue.

Représentation du traitement (option info)

1. Lire X
   A←souschaine(X,1,1)
   L←longueur(X)
   B←souschaine(X,L,1)
   afficher A;B

Indique ce que l'ordinateur affiche pour l'entrée de « BONJOUR », de « B »

2. Ecris un programme pour que l'utilisateur ayant entré une chaîne, l'ordinateur affiche la chaine en passant la première lettre à la fin (BONJOUR → ONJOURB)

3. Ecris un programme pour que l'utilisateur ayant entré une chaîne, l'ordinateur intervertit la première lettre et la dernière (TALUS → SALUT)

2.
A←souschaine(X,2,L+souschaine(1,1))

3.
A←souschaine(X,2,L+souschaine(1,1))
A←souschaine(X,1,L-1+souschaine(5,1))

Tu avais mis un « + » ici, que signifie-t-il?

Ben, là… j'avais écrit BONJOUR donc X, à partir de la deuxième, on prend la longueur totale, plus la première lettre, on en prend une.

C'est quoi souschaine(1,1) …?

La longueur de X et 1 et 1 [6]

Dans ma thèse j'avais rencontré et commencé à traiter des difficultés analogues, mais comme c'était dans le cadre de l'option informatique, ça ne concernait pas directement l'algèbre, d'ailleurs je voulais éviter tout ce qui rappelait les mathématiques.

Donc en classe de Seconde, cela concernait les chaines de caractères. En février, les élèves avaient eu déjà une quarantaine d'heure en option et avaient été initiés au type chaîne de caractères. J'avais fait passer une petite épreuve pour repérer des élèves en difficulté et proposer des remédiation.

Dans le texte de l'épreuve, on rappelait d'abord les deux fonctions principales, longueur et souschaine.

Dans une première question, il s'agissait d'interpréter un petit programme. Cela donnait aussi une sorte de modèle pour la suite.

Dans les deux autres questions, il était demandé de créer un programme de même nature; la langage était volontairement proche d'une exécution « à la main » passer, intervertir, et pas comme un calcul à faire.

Pour la première question, pas de souci, les résultats sont donnés correctement sauf un élève qui donne B au lieu de BB pour l'entrée de la lettre B.

Pour la seconde question, ça se gâte nettement, environ la moitié des élèves ne donnent pas un réponse qui représente le calcul d'une chaîne.

Pour la troisième question, on a le même type de réponse ou pas de réponse.

Voici une réponse typique (encadré rouge). On voit qu'il y a bizarrement l'addition d'une variable représentant probablement la longueur (mais non affectée) et d'un appel à souschaine incorrect puisqu'avec deux arguments seulement.

Si l'on regarde la réponse à la question 3, elle est cohérente au moins au début. L'élève a d'abord passé le premier caractère en dernier, puis il s'occupe du dernier caractère. On remarque qu'il affecte le résultat à la même variable dans les deux lignes.

Concernant le profil de ces élèves en difficulté: ils avaient choisi l'option parce qu'ils ou elles voulaient faire des sciences, mais ils ou elles étaient rebutés par les maths, en particulier l'algèbre. Ils étaient très passifs et recopiaient les programmes après que le prof. avait donné

Conditions et Booléens (algo lycée)

Tant que (A<>495) ou (A<>0)

P : ..vous les filles devant, vous m'avez dit que le test qu'on fait là c'est A différent de 495 **ou** A différent de 0, au fond, tu m'as dit. . .

« *les filles devant » sont passées d'une condition d'arrêt* (A=495) ou (A=0) *à une condition de continuation* (A≠495) ou (A ≠ 0)

E3 : A différent de 0 **et** A différent de 495.

« au fond » a nié correctement la condition de continuation

P : Alors, qu'est-ce qui est correct, qu'est-ce qui ne l'est pas ? … Oui ?

la prof lance le débat

E7 : Ben moi ça me parait bizarre qu'un nombre soit égal à deux nombres différents.

« une fille devant » réagit à E3: pour sortir de la boucle, A devrait être égal à 0 et à 495

7

Toujours dans le contexte actuel de l'algorithmique au lycée, voici un épisode à propos des conditions, ici dans une itération avec sortie en début de boucle, donc de forme tant que. L'épisode est observé dans le cadre d'un mémoire de master mettant en place une ingénierie sur laquelle je reviendrai en seconde partie de l'exposé.

On souhaite que l'itération se termine quand une variable prend une des deux valeurs 495 ou zéro.

On est dans une bonne classe de terminale qui fait de l'algorithmique depuis la seconde.

L'auteure du mémoire qui a préparé l'ingénierie la met en place en prenant la place de l'enseignant (P dans le dialogue) Le professeur de la classe est là en tant qu'observateur (O).

Il s'agit d'une discussion collective qui fait suite à ce qu'un groupe (les filles devant) a écrit la condition sous la forme Tant que (A<>495) ou (A<>0) et ne comprennent pas pourquoi l'exécution part en boucle.

Donc l'enseignante essaie de provoquer un débat. Un autre groupe E3 corrige sans plus d'explication. Une élève fait une intervention peu explicite, mais on imagine qu'elle pense à ce qui ferait sortir de la boucle, donc en fait la négation de la condition (A<>495) et (A<>0), qu'elle conçoit comme (A=495) et (A=0),

Conditions et Booléens (algo lycée)

Tant que (A<>495) ou (A<>0)

P : Voilà, donc, quand tu dis le et, tu vois, un nombre qui est différent de deux nombres, c'est vrai, tu vas trouver des nombres qui sont différents de 495 **et** différents de 0. .. Mais tu vas jamais satisfaire la condition être égal à 495 **et** à 0. … Finalement, ton algorithme, il s'arrêtera quand tu obtiendras un nombre égal à 495 **et** à 0, d'accord ? Et ça tu vas pas pouvoir le trouver.

Explication embrouillée du prof

E3 : On l'a fait **fonctionner** et ça a **marché**.

P : Tu l'as fait **tourner** et ça **marche**. . .

Validation empirique

O : Tu l'as fait. . . **Ça devrait pas marcher.**

P : Oui, mais, **AlgoBox** il a des conditions d'évaluation des. . .

…

P : Bon, on va y réfléchir, c'est une question à laquelle il faut réfléchir …

8

L'enseignante reprend cette idée, mais de façon assez confuse, puisqu'elle considère en même temps une condition de continuation correcte « des nombres qui sont différents de 495 et différents de 0 » et une condition d'arrêt fausse « être égal à 495 et à 0 »

De toute façon la discussion tourne court, quand l'élève qui a corrigé propose une validation pragmatique « On l'a fait fonctionner et ça a marché », reprise par le professeur. Le débat est relancé par l'observateur qui a du être embrouillé par l'explication car il semble très étonné que la condition avec le et fonctionne. Mais la discussion reste sur le plan pragmatique, l'enseignante faisant référence au logiciel et non plus à la logique. Après une phase assez confuse, l'enseignante clôt le débat.

Dans son mémoire, l'enseignante dit qu'elle ne s'était pas préparée à ce problème et a donc été prise au dépourvu. On peut le comprendre. La question est pourquoi une enseignante a priori bien formée, intéressée par l'algorithmique puisqu'elle fait son mémoire sur ce sujet est prise au dépourvu. Pourquoi ne reprend-elle pas la remarque de E7, en l'exprimant en termes de condition d'arrêt, et en soulignant que la condition dans le tant que est une condition de continuation, négation de la condition d'arrêt ? Ceci suggère que les difficultés liées aux conditions (valeurs logiques de type booléen) sont sous-estimées. La ressemblance entre ces conditions et celles que l'on exprime dans les conditions « ordinaires » est trompeuse dès que les conditions deviennent un peu complexes, notamment lorsqu'elles imposent l'emploi de connecteurs

Conditions et Booléens (option info)

J'ai 5 amis : Marie, Marc, Jean, Janine et Jean.

1. "Marie et Jean ne s'entendent pas"
2. "Marc et Marie ne viendront pas l'un sans l'autre
3. "Si j'invite Janine, il faut que j'invite Luc ou Jean, mais on ne peut pas les inviter tous les trois ensemble"

Ecrire un programme qui permet de savoir si une invitation est possible.

Solution attendue

```
display 'Marie invité(e)  (O/N)'
read Rep
Marie :=(Rep='O' )
….
C1:=Not (Marie and Jean)
C2:=(Marc=Marie)
C3:=...
If C1 and C2 and C3 then
     display 'Possible'
 else
     display 'Impossible'
```

Une solution « élève » typique

```
display 'Marie invité(e)  (O/N)'
read Rep
If Rep='O' then
   Marie :=true
else
    Marie :=false
If (Marie and Jean) then
        display 'Impossible'
else
        if  (Marc<>Marie) then
                display
                        'Impossible'
        else            ……
```

9

J'avais identifié déjà ce phénomène dans ma thèse lors de l'option informatique

Un exemple. J'avais entrepris une ingénierie sur les booléens avec des élèves plutôt matures puisqu'ils étaient dans leur deuxième année d'option et dans une classe scientifique.

J'avais imaginé le petit problème suivant. Il peut se traiter avec des booléens, type auquel le élèves avaient été initiés, ce qui donne une résolution assez simple si l'on déclare 5 variables booléennes correspondant aux cinq amis.

Pour chaque condition on forme une petite expression booléenne. C'est proche de l'intuition pour les deux premières, beaucoup moins pour la troisième. Ce que j'ai constaté, c'est que ces élèves n'avaient pas trop de mal à utiliser les variables booléennes pour exprimer des conditions tant qu'il est possible de le faire de façon congruente avec l'énoncé, mais avaient par contre beaucoup de difficultés avec d'autres conditions.

Dans une solution typique, l'élève utilise une alternative pour affecter une condition à une variable booléenne. Ensuite il imbrique des alternatives, structure qui lui permet de transposer les deux premières conditions directement de l'énoncé dans le langage, mais est bloqué pour la prise en compte de la troisième. Mon interprétation était que les élèves concevaient des variables booléennes à deux états, mais ne les concevaient pas comme calculables. Mettre en évidence ces difficultés et leur persistance, ne veut pas dire qu'elles en soient pas traitables, c'était l'objets des ingénieries proposées dans ma thèse

Difficultés similaires

Algorithmique au lycée (2010)

↔

Option informatique (1985)

1. Représentation du traitement par la machine … influencée par les traitements « ordinaires »
2. Conditions et valeurs Booléennes … comprises comme non calculables et non affectables

Hypothèse

- *Des savoirs sont en jeu qui ne relèvent ni des mathématiques enseignées ni de la « science informatique » au sens strict.*
- *Ils supposent des situations d'enseignement/apprentissages spécifiques.*

Directions

- Acquis des recherches en psychologie de la programmation
- Approche didactique

11

Ce qui nous conduit à notre hypothèse

Dans les activités de programmation s'adressant à des débutants, quels que soient les objectifs recherchés (info pour l'option informatique, math pour l'algorithmique, voire math et info) *Des savoirs sont en jeu qui ne relèvent ni des mathématiques enseignées ni de la « science informatique » au sens strict. Ils supposent des situations d'enseignement/apprentissages spécifiques.* …

Si on n'identifie pas les savoirs et qu'on ne recherche pas des situations adéquates, on s'expose à ce que les objectifs ne soient pas atteints pour bon nombre d'élèves. Nous allons nous intéresser à cette hypothèse dans deux directions

Tout d'abord ce qu'on appelle la psycho de la programmation, ce pour quoi je passerai la parole à Janine,

Ensuite je reprendrai la parole pour voir ce que pourrait être une approche didactique à partir des rares travaux de ce domaine.

- Ici s’insère l’exposé de Janine Rogalski
- (diapos Rogalski_SemNat_Nov2015.pdf)

Du « constructivisme naïf » à la TSD

```
AV 10 TD 90
AV 10 TD 90
AV 10 TD 90
AV 10 TD 90
```

```
Pour Carre :A
 repete 4
   [AV :A TD 90]
Fin
```

```
Pour Dessin :N
 Carre :N
 si :N>0 [Dessin :N-1]
Fin
```

CRAHAY, M. (1987). Logo, un environnement propice à la pensée procédurale ? *Revue française de pédagogie*

Résultats décevants

Tendance des enseignants

- Laisser agir les enfants
- Se concentrer sur une intervention individuelle
- Faire à leur place

- Outil de progrès cognitif
- Conditions
  - Une théorie de l'expertise qui décrit la structure de connaissance
  - Une théorie de l'acquisition qui décrit le processus de construction de la connaissance
  - Une théorie de l'intervention qui suggère les conduites d'enseignement

13

Donc, je reprends la parole, ce que je souhaiterais, c'est que des recherches démarrent et souligner qu'avec la TSD nous avons un outil pour cela.

Pourquoi la TSD ? Parce que dans les enseignements à des débutants on observe souvent un point de vue constructiviste naïf. Le cas de LOGO dans les années 1980 est représentatif et il a été bien analysé par Crahay. Logo est un langage où l'on écrit des procédures qui peuvent s'appeler les unes les autres.

J'ai mis un exemple en haut, on peut faire un carré de taille donnée avec 8 instructions en direct, on peut aussi créer une procédure qui permet de dessiner un carré générique. Ensuite on peut appeler cette procédure dans une autre procédure et une procédure peut aussi s'appeler elle-même. Il a été avancé à l'époque que les activités de programmation en Logo conduiraient à développer chez les enfants la pensée procédurale, c'est-à-dire la capacité à penser une série d'actions comme une procédure paramétrable et réutilisable dans un traitement plus complexe.

Il ya eu beaucoup d'expérimentations sur Logo de par le monde, ce qui a permis des évaluations. Crahay fait une revue des évaluations, et conclut qu'elles ont toutes montré des résultats décevants.

Il met en cause particulièrement les pratiques des enseignants qui au départ laissent une grande liberté aux apprenants dans la construction de programme, mais ensuite se limitent à de interventions individuelles auprès des élèves et finissent par écrire le programme à leur place.

Il conclut que les activités de programmation en Logo peuvent être l'occasion de progrès cognitif incomparable, mais que il faudrait pour cela

1) une théorie de l'expertise qui décrit la performance terminale ou la structure de connaissance achevée que

nous espérons faire acquérir par l'apprenant;

2) une théorie de l'acquisition qui décrit le processus de construction de la connaissance ou de la performance

vises ou autrement dit les étapes par lesquelles les sujets transitent lorsqu'ils acquièrent cette compétence;

3) une théorie de l'intervention qui suggère quelles conduites d'enseignement sont susceptibles de stimuler

le processus d'acquisition et qui fournit. par le fait même une information utile pour savoir comment s'y prendre avec l'apprenant»

Je pense que l'on peut étendre la réflexion à l'ensemble des activités de programmation et proposer la théorie des situations didactiques comme cadre général correspondant aux différentes conditions.

Une approche « théorie des situations » pour les premiers apprentissages

- 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.

- LAGRANGE J.B., GUY, M.N. (2015) Planification et connaissances mathématiques dans une situation d'apprentissage au lycée : l'algorithme de Kaprekar. Petit X n°97

14

En fait il n'y a pas eu beaucoup de recherches adoptant ce cadre de la TSD. La seule que je connaisse est la thèse de Nguyen Chi Thanh menée dans le contexte des programmes de math du lycée des années 2000.

Et plus récemment dans le contexte de l'algorithmique au lycée, un travail modeste mené dans le cadre de l'encadrement d'un mémoire de Master, et paru dans la revue PetitX. Je remercie à ce propos les relecteurs et éditrices qui m'ont apporté une aide considérable, justement à propos de la TSD.

Nguyen C.T. (2005)

- Solidarité fondamentale entre les mathématiques et l'informatique
  - écologie pas évidente de l'algorithmique et de programmation dans l'enseignement secondaire.
- La structure itérative (évolution des machines)
  - Mémoire effaçable
    (emplacement physique correspond à une donnée évoluant au cours du traitement)
  - Branchement contrôlé par la mémoire
    (le traitement « boucle » jusqu'à une valeur spécifiée)
  - Programme comme donnée en mémoire
- Un milieu et une situation fondamentale

15

A l'époque Nguyen Chi Thanh a présenté son travail et publié des articles, mais comme ça ne semble pas être beaucoup repris, je vais en dire quelques mots.

Il étudie lui aussi les rapports entre mathématiques et informatique et conclut à une solidarité fondamentale.

Il s'intéresse à l'itération et montre que c'est une structure qui émerge dans l'évolution des calculatrices ou machines mathématiques de la calculatrice simple à la machine de babbage qui était capable de tabuler des fonctions de façon automatique, grâce au fait qu'un qu'un emplacement mémoire correspondant à une donnée invariante au cours d'un calcul répétitif peut prendre successivement des valeurs différentes, et le contrôle de l'exécution non plus par l'opérateur, mais par la machine en fonction de l'état de la mémoire, qui permet notamment l'itération.

Et finalement l'ordinateur moderne où le processeur exécute un programme qui est lui-même considéré comme une donnée, ce qui permet le branchement contrôlé de façon beaucoup plus souple

Et donc NCT construit un milieu et une situation fondamentale

- « Soit f une fonction. Calculer les images par cette fonction de m nombres espacés d'un pas p dans un intervalle donné»

  f(x)=x²+1, intervalle [-3,2], pas 0,2 puis 0,03

  Calculatrice avec mémoires STO

1. Ecrire la suite des touches nécessaire pour la faire exécuter par un robot

   -3 Sto A A x A+1=
   -3.2 Sto A A x A+1=
   -3.4 Sto A A x A+1=
   -3.6 Sto A A x A+1=
   ……..

2. Ecrire un groupe de touches à faire répéter par le robot

Groupe de touches à répéter :

. A + 0,03 → STO B
. B x B + 1 =
. B STO A

3. Ecrire un message le plus court possible pour que le robot fasse tout le calcul de lui-même

-3 STO A
répéter 166 fois A x A + 1 =
A + 0,03 STO A

16

La situation fondamentale est basée sur la tabulation de fonctions.

Les situations sont prévues pour une classe de Seconde. En voici une résumée.

Les élèves ont des calculatrices simples, sans parenthèses, mais disposant de mémoires A, B, C pouvant stocker le résultat d'un calcul. La situation met aussi en scène un robot capable d'appuyer sur les touches de la calculatrice à partir d'une liste de touches fournie sur papier.

La première tâche, surtout dans la seconde modalité implique l'écriture fastidieuse d'une série de touches. Assez vite les élèves voit l'intérêt d'une suite invariante qui pourrait être répétée.

Mais cela implique une rupture, puisqu'on ne peut plus entrer individuellement les valeurs de la variable.

Comme on voit ici, ça peut passer par l'utilisation d'une seconde mémoire, l'élève ne voulant sans doute pas que la valeur dans la mémoire A soit perdue.La troisième tâche est ensuite sans difficulté, sauf que les élèves se trompent généralement dans le nombre d'itérations, la valeur 1,98 est calculée, mais son image n'est pas affichée.

- Limites
  - Contexte tabulation
  - Répétition
- Points forts
  - Variable itérative
  - Construction plutôt que monstration
- Influence sur la compréhension
  - Initialisation de la variable itérative

17

L'ingéniérie a certes ses limites puisque l'itération se limite à la répétition et reste liée au contexte de la tabulation de fonctions. Néanmoins, elle met bien en valeur l'iée de variable itérative.

Généralement quand on introduit l'itération à des élèves, c'est par monstration. On montre un programme et on explique comment il marche. Il n'est pas étonnant que, sans construction, de nombreux élèves soient en difficulté.

Un bénéfice, semble-t-il, est que l'on ne rencontre pas l'erreur consistant à placer l'initialisation de la variable itérative dans le corps de boucle, alors que cette erreur a été repérée comme courante et persistante chez les débutants.

LAGRANGE J.B., GUY, M.N. Petit X n°97

- « l'algorithmique a une place naturelle dans tous les champs des mathématiques et les problèmes posés doivent être en relation avec les autres parties du curriculum » Programme du lycée
- L' « algorithme » de Kaprekar
  - Planification (savoir « algorithmique »)
  - Numération (savoir mathématique)

18

J'en vient donc à la petite étude menée avec Marie Noelle Guy dans le cadre de son mémoire de master, qui va me permettre de préciser d'une part l'interaction math-algorithmique et une mise ne œuvre modeste de la théorie des situations.

Dans cette étude on s'est placé dans le contexte du programme du lycée quand il précise que l'algorithmique a une place naturelle dans tous les champs des mathématiques et les problèmes posés doivent être en relation avec les autres parties du curriculum, en faisant l'hypothèse que ça pouvait être une base pour concevoir une situation d'apprentissage.

On a retenu un travail autour de l' algorithme de Kaprekar que je vais présenter ensuite, parce qu'il nous a semblé qu'il mettait en jeu, à côté des savoirs mathématiques, des savoirs en programmation ou algorithmique, relatifs à la planification. La planification je vais expliquer de quoi il s'agit, ça a un peu à voir avec la modularité dont j'ai parlé tout à l'heure. Côté savoir mathématique, ça concerne la représentation des nombres, qu'on peut associer à la numération ou plus généralement à l'arithmétique. Notons que cette ingénierie s'insère dans une ingénierie plus vaste se poursuivant par la preuve de la terminaison de l'algorithme, partie traitée par Dominique Laval dans son travail de thèse.

L'« algorithme » de Kaprekar

1. Choisir un nombre entier de trois chiffres.
2. Former le nombre obtenu en arrangeant les chiffres du nombre choisi en 1. dans l'ordre croissant.
3. Former le nombre obtenu en arrangeant les chiffres du nombre choisi en 1. dans l'ordre décroissant.
4. Calculer la différence des nombres obtenus en 2. et 3.
5. Recommencer (à partir de l'instruction 2.) avec le résultat obtenu en 4, jusqu'à obtenir un nombre déjà obtenu.

Traitement par un dispositif « arithmétique »

Deux représentations (nombre, triplet de chiffres)

Etapes à construire

a. Séparer les trois chiffres pour les trier
b. Recomposer des nombres après le tri pour soustraction

19

Donc, si vous ne connaissez pas, voici le traitement connu sous le nom de Kaprekar. Pour simplifier je le présente pour les nombres à trois chiffres, mais monsieur Kaprekar l'avait étudié pour des nombres à 4 chiffres et sans machine, ce qui était une performance.

Avec trois chiffres, vous avez le cas trivial des multiples de 111 qui donne zéro lui-même invariant.

A partir d'autres nombres, on atteint 495 qui est un autre invariant en un maximum de 5 itérations.

Notre situation repose sur le fait que dans ce traitement, le nombre courant est considéré sous deux représentations. Pour le tri (arrangement dans l'ordre croissant ou décroissant) aux étapes 2 et 3, c'est un triplet de chiffres et pour la soustraction c'est un nombre au sens habituel du terme. A la main, la différence de représentation n'apparait pas. Vous n'avez pas conscience d'isoler des chiffres et de les ordonner quand vous voyez 495, vous passez tout de suite à 954 et 459.

En revanche, si vous vous voulez faire exécuter le traitement à une machine « arithmétique », c'est-à-dire disposant des 4 opérations dans les entiers et d'instructions de comparaison, il va falloir considérer le nombre courant sous ses deux représentations et donc construire des étapes de conversion du nombre ordinaire en triplet et inversement.

La situation donc met en jeu d'une part la planification puisqu'il faut penser des étapes, et d'autre part l'arithmétique pour construire les conversions.

ROGALSKI, J., SAMURÇAY, R. (1990) Acquisition of programming knowledge and skills. *Psychology of programming*..

- **Schéma** : structure utilisée dans le traitement des informations pour atteindre des objectifs à petite échelle,
- **Plan :** ensemble organisé de schémas

Top-Down
on pense le plan et on spécifie les schémas élémentaires.

Bottom-up
on fait l'inventaire des schémas disponibles et on les organise en plan.

Débutants
Pas de répertoire de schémas.
Plans transposés d'une exécution manuelle. 20

Un cadre théorique pour la planification. On le trouve dans un des chapitres d'un ouvrage de référence au tournant des années 90 Psychology of programming.. qui fait le point sur l'activité de programmation d'un point de vue expert et du point de vue des débutants.

Les auteures distinguent les schémas : structure utilisée dans le traitement des informations pour atteindre des objectifs à petite échelle, et les **Plans qu'on peut voir comme des** ensembles organisés de schémas.

Les experts combinent généralement une approche descendent (top-down) et ascendante bottom-up. En gros si on voit bien où on va, on construit un plan et ensuite on s'intéresse à des sous-objectifs et aux structures correspondantes. Quand on voit moins bien où on va, on peut partir des schémas disponibles.

Le problème des débutants est qu'ils ne disposent pas d'un répertoire de schémas et que les plans auxquels ils pensent sont directement transposés d'une exécution manuelle qui rend difficile la spécification des sous-objectifs pour un dispositif informatique. Donc ils ne peuvent faire ni du top-down ni du bottom-up

Vous voyez que pour Kaprekar si on calque le plan sur l'exécution manuelle, on est coincé au moment de produire les nombres dans l'ordre croissant et décroissant.

Hypothèse

- Schémas de décomposition et de recomposition d'un nombre non disponibles initialement chez l'élève.
- Prise de conscience
  - de la nécessité d'aménager le plan du traitement (découpage en étapes).
  - par confrontation au dispositif « arithmétique ».

21

Donc l'hypothèse à la base de la situation est que

Les schémas de décomposition et de recomposition d'un nombre ne sont pas disponibles chez l'élève

Et que par confrontation au dispositif « arithmétique » il peut se produire une prise de conscience

> de la nécessité d'aménager le plan du traitement, et plus généralement de ce qu'un plan convenant à un traitement manuel doit être adapté en fonction des capacités du dispositif destiné à l'exécuter.

Le Milieu

- Données à traiter
  - Objets mathématiques
    rétroactions renvoient à leurs propriétés mathématiques,
  - Objets « codifiés » (choix de représentations)
- Le langage : rétroactions résultant de
  - contraintes d'expression,
  - contraintes de représentations des données

  perçues de façon formelle,

  et/ou en référence au dispositif

22

Précisons le milieu.

On a d'abord des données à traiter. Ici il s'agit d'objets mathématiques, d'une part ils ont des propriétés mathématiques, par exemple la soustraction du nombre avec les chiffres dans l'ordre décroissant et du nombre dans l'ordre croissant doit donner un nombre entier positif sinon on s'est trompé quelque part.

Mais ce sont aussi des objets à codifier pour un traitement exprimé dans un langage symbolique..

On a donc aussi des rétroactions du langage au sens de code symbolique destiné à être exécuté par un dispositif. Les rétroactions résultent de contraintes d'expression et de représentation

perçues de façon formelle, (cette écriture n'est pas « conforme »)

Ou en référence à un dispositif (l'ordinateur ne va pas comprendre) et souvent les deux à la fois.

Choix

- Classe de Terminale
- Algobox
  - Langage exécutable
  - Connu des élèves
  - Pas de procédures
  - Déclaration des variables
  - Un seul type numérique
  - Itération Tant que

- Fonctions arithmétiques
- Listes
- Nombres à trois chiffres
  - continuation sur ≠ 495 et ≠0
  - condition à la charge des élèves

Mettre l'accent sur la représentation des données (choix des variables) pour engager une dynamique de planification productive

23

Les choix pour la mise en œuvre.Tout d'abord le niveau de classe.
La terminale pour 2 raisons: c'est le niveau où l'on fait de l'arithmétique et aussi, la situation suppose chez les élèves des connaissances concernant le langage.

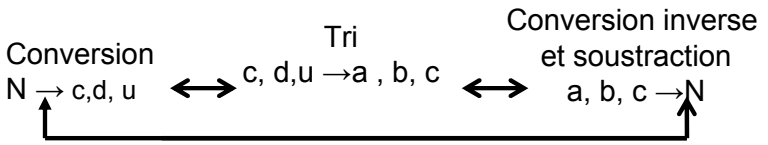
Le choix d'un langage symbolique. Dans l'étude, c'est algobox, je suppose que l'outil est connu ici. Les principales caractéristiques ; il est exécutable en ce sens qu'un environnement sur ordinateur permet à la fois l'écriture d'un texte avec des contraintes syntaxiques et son exécution. Nous l'avons choisi principalement parce que c'est l'outil habituel des élèves. Les élèves connaissent aussi le tableur, nous exposons dans l'article les raisons pour lesquelles Algobox est plus adapté. Il faut noter que Algobox ne permet pas l'écriture de procédure et on pourrait penser que c'est une limitation pour une démarche de planification. Il offre un seul type numérique (pas d'entiers). Algobox privilégie l'itération tant que, et donc impose, comme on l'a vu dans la première partie de considérer une condition de continuation, plutôt qu'une condition d'arrêt. Nous faisons aussi le choix de limiter aux fonctions arithmétiques et au traitement de listes.

D'autres choix: considérer des nombres à trois chiffres ce qui permet un algorithme de tri sans itération, et de considérer une condition de continuation sur deux valeurs, ce qui est plus simple que le repérage d'un invariant ou d'un cycle et permet de laisser la conception de cette condition à la charge des élèves (voir la diapo 8)..

Un choix essentiel est de mettre l'accent sur la représentation des données (choix des variables) pour engager une dynamique de planification productive

Phases

1. Variables à déclarer (Identification des chiffres comme données non explicites dans un traitement « manuel »)
2. Discussion collective sur
   - ébauche de plan (« étapes »),
   - Spécification des schémas en « sous-algorithmes »
     - conversion du nombre en la suite de ses chiffres,
     - tri des trois chiffres,
     - conversion inverse et soustraction
3. Ecriture des « sous-algorithmes » en a-didacticité
4. Organisation des « sous-algorithmes » en a-didacticité



24

Les 3 phases en résultent.

On va demander aux élèves quelles variables sont à déclarer, et on s'attend à ce que certains prennent conscience de la nécessité d'introduire une ou des variables pour les chiffres du nombre courant, soit une liste à trois éléments numériques, soit plus simplement trois variables numériques.Ensuite on peut penser que cette prise de conscience va être partagée dans la classe et qu'un plan incluant des étapes ou sous-algorithmes de conversion vont être identifiées.

Dans une seconde phase, les élèves conçoivent les sous-algorithmes. Dans un esprit de modularité, les sous-algorithmes sont confiés à des groupes différents.

La troisième phase consiste à rédiger l'algorithme complet en organisant les sous-algorithmes dans une itération. Il ne s'agit d'une simple concaténation. En effet les sous-algorithmes ont des déclarations de variables spécifiques qui doivent être harmonisées. De même pour le nombre entré dans le premier sous-algorithme qui doit être harmonisé avec le nombre rendu par le 3ème. Ils ont aussi à construire l'itération avec une condition d'arrêt adéquate.

Scénario

- **Travail à la maison**
  - exécuter à la main le traitement sur quelques valeurs,
  - suggérer un test d'arrêt pour l'itération,
  - répertorier les variables nécessaires
- **Première phase** (20 mn)
  - Synthèse du travail à la maison. l'enseignante se contente de mettre en commun ces réponses et de les soumettre à la validation de la classe L'enseignante suggère de segmenter l'algorithme en traitements élémentaires, mais laisse aux élèves l'identification de ces traitements.
- **Seconde phase**
  - Sur ordinateur, 3 groupes traitent la conversion du nombre en chiffres, 3 groupes traitent le tri des chiffres et 2 groupes traitent la conversion inverse (45 mn)
  - Mise en commun (20 mn)
- **Travail à la maison**
  - assembler trois sous-algorithmes afin de proposer une première version
- **Troisième phase**
  - Discussion sur la première version de l'algorithme
  - Par groupes sur ordinateurs
    - Exécution
    - Instructions de sorties de façon à obtenir des observables sur le comportement de la suite des résultats.

Analyse a posteriori

1. Prise de conscience des deux représentations

   On va prendre *A en nombre initial,* ***on va prendre*** *B arrangé en ordre décroissant, C en ordre croissant,* ***on va faire*** *la différence de, euh, C moins B. O****n va le remettre*** *dans A, et* ***on va refaire*** *avec B et C.*

   *Ben si A c'est une variable où c'est un nombre, pour qu****'il*** *puisse le ranger dans l'ordre croissant et décroissant, faut d'abord séparer les chiffres des dizaines, des unités et des centaines. Donc il faut trois variables….*

2. Identification des étapes effectuée par les élèves et écriture des sous-algorithmes à la charge des élèves

   Trois écritures différentes pour la décomposition et le tri, une pour la recomposition

   « ***L'ordinateur*** *pour arranger les chiffres en ordre décroissant, ça va être important* ».

   « faut voir ce que ***AlgoBox*** *est capable de faire. »*

3. L'assemblage des sous-algorithmes à la charge des élèves

   Les groupes qui n'avaient pas traité toutes les étapes ont rencontré plus de difficultés que les autres.

3. Une difficulté dans la compréhension de l'itération

   nécessité d'avoir une donnée de même nature en entrée et en sortie de la boucle

26

Quelques éléments d'analyse à posteriori dans chacune des phases

Dans la première phase, on assiste bien à une prise de conscience de la nécessité de deux représentations sous l'influence du dispositif.

Pour illustrer, deux prises de parole successives par deux élèves, l'un fait référence au traitement manuel « on va prendre » et présent un choix naïf des variables, dans un schéma itératif, qui lui est correct.

Le second élève fait lui référence au dispositif (« il ») et à ses capacités et identifie la nécessité de variables pour les chiffres.

Comme prévu aussi, les élèves, dans une phase collective identifient les étapes et par groupe sur ordinateur conçoivent les sous-algorithmes. Les solutions sont variées, et là aussi on constate à nouveau que le dispositif joue un grand rôle dans la réflexion des élèves

Quelques difficultés intéressantes sont signalées dans la dernière phase.

Etude de Ressources

	Identification Etapes	Décomposition du nombre	Tri des chiffres	Reconstruction du nombre	Construction algorithme
Math'x	Donnée par le manuel	Partiellement à la charge des élèves.	Guidé par l'énoncé	A la charge des élèves	Pas à la charge des élèves
2nde 1	À la charge de l'enseignant	Très guidée par les énoncés	Guidé par les énoncés	À la charge des élèves	Très guidée par l'énoncé
2nde 2	Suggérée par l'enseignante	Suggérée par l'enseignante	Suggéré par l'enseignante :	À la charge des élèves nombreuses difficultés	À la charge des élèves
1ère S	À la charge de l'enseignant	L'enseignant donne l'algorithme final	L'enseignant donne l'algorithme final	À la charge des élèves	À la charge des élèves

27

Pour évaluer l'intérêt de la situation nous avons comparé à des mises en œuvre de situations autour de l'algorithme de Kaprekar trouvés dans un manuel et dans une ressource de l'IREM de Besançon qui présente des mises en œuvre dans 3 classes. Les ressources sont présentées dans l'article de PetitX. Je ne détaille pas l'analyse que nous en faisons. Disons seulement que la mise en œuvre est différente quand à la part de responsabilité laissée aux élèves dans la construction de l'algorithme. L'identification des étapes pour un traitement algorithmique est donnée soit par l'énoncé, soit par l'enseignant. Certains sous-algorithmes sont à la charge des élèves, particulièrement la conversion inverse du triplet au nombre qui est jugée facile, mais en fait ne l'est pas tant que cela pour les élèves qui n'ont pas opéré la décomposition.

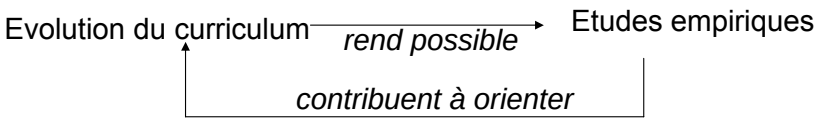
On peut donc considérer qu'il y a un gain certain dans la situation mise en place dans le mémoire en termes d'implication des élèves dans la construction de l'algorithme..

- *Des savoirs sont en jeu qui ne relèvent ni des mathématiques enseignées ni de la « science informatique » au sens strict.*
- *Ils supposent des dispositifs d'enseignement/apprentissages spécifiques.*

Quatre dimensions indissociables dans les savoirs en jeu	•Traitement •Dispositif (exécution en différé)	•Langage •Représentation des données

Savoirs: compréhension du langage, itération, planification
Situations donnant aux élèves toute leur part dans l'élaboration
Identifier le milieu et la possibilité de phases a-didactiques
Tirer parti pour cela de recherches en psychologie de la programmation.

.



28

Pour conclure, je reviens sur notre hypothèse et le repérage de 4 dimensions indissociables

Dans l'exposé, on a commencé à explorer certains savoirs, la compréhension du langage et des ses contraintes, l'itération, la planification.

On a essayé de montrer qu'il faut concevoir des situations qui donnent aux élèves la responsabilité de l'élaboration du programme ou de l'algorithme.

Et donc la nécessité de concevoir un milieu adéquat et des phases a-didactiques et qu'il est intéressant de tirer parti pour cela de recherches en psychologie de la programmation

En ce qui concerne les évolutions curriculaires, on peut les voir comme on veut. Pour moi, je les vois comme une opportunité pour un terrain expérimental et j'espère que de nombreuses recherches empiriques vont se développer. On peut espérer qu'en retour ces recherches vont participer à orienter l'évolution en délimitant le champ des possibles.