Patrons d'exercices pour APLUSIX

Une étape du développement de l'EIAH occasion d'un travail entre didacticiens et informaticiens

Denis Bouhineau*, Alain Bronner, Hamid Chaachoua*** et Jean-François Nicaud***

** Université Joseph Fourier - LEIBNIZ - MeTAH*
46 Av F. Viallet
38031 Grenoble Cedex
Prenom.Nom@imag.fr
*** LIRDEF - IUFM de Montpellier*
2 Place Godechot
BP 4152, 34092 Montpellier Cedex
Prenom.Nom@montpellier.iufm.fr
**** IUFM – LEIBNIZ - MeTAH*
46 Av F. Viallet
38031 Grenoble Cedex
Prenom.Nom@imag.fr

RÉSUMÉ. *Au cours du développement de l'EIAH APLUSIX, il y a eu de nombreuses occasions d'un travail partagé entre informaticiens et didacticiens. Quelques-unes sont décrites succinctement. Lors de la mise en place de patrons d'exercices et d'une carte de tests, une coopération plus approfondie a eu lieu, en respectant les disciplines de chacun. Cette mise en place de patrons d'exercices est décrite plus longuement.*

MOTS-CLÉS : *algèbre, contrainte, domaine, hiérarchie, paramètre, pluridisciplinaire.*

1. Introduction

Au cours du développement de l'EIAH APLUSIX, il y a eu de nombreuses occasions d'un travail partagé entre informaticiens et didacticiens, par exemple la construction de conceptions d'élèves, ou la mise au point d'un diagnostic de la production d'élèves. Cependant, à chaque fois, informaticiens et didacticiens ont travaillé sur des objets différents. Au niveau plus technique de l'informaticien se trouvent des objets computationnels très formalisés ; les objets du didacticien, même s'ils parlent de la même chose, sont d'un autre ordre, plus littéraire et informel.

Une étape a été franchie lors de la construction de patrons d'exercices pour l'EIAH APLUSIX. Après une phase de travail selon un schéma producteur-consommateur hétérogène, le travail a pu s'effectuer sur des objets communs. L'article, après un bref exposé des premiers moments du travail commun entre informaticiens et didacticiens, décrit cette construction en montrant les évolutions qui se sont opérées de part et d'autre pour converger vers un objet commun de travail et les constructions qui ont été développées pour chaque discipline.

2. Premiers développement du projet APLUSIX

Le renouveau du projet APLUSIX a commencé en Juin 2000, à Nantes, dans un laboratoire d'informatique, au sein d'une équipe ne comportant pas de didacticien. Le cœur du système et les concepts qui régissent l'environnement ont donc été le fruit d'un travail en informatique seulement. L'objectif était de produire un micromonde d'apprentissage de l'algèbre avec une interface ergonomique fournissant des rétroactions épistémiques et pouvant aller en classe [NIC 2004].

Le premier travail commun informatique-didactique important a eu lieu en 2003 après le déplacement du projet APLUSIX à Grenoble : c'est la description de règles de calculs erronés utilisées par les élèves. Ce travail s'est effectué sur la base de protocoles récoltés par l'environnement APLUSIX lors d'utilisations en classe, les didacticiens faisant l'analyse et produisant les règles de transformations erronées. Plus de 600 heures d'utilisations d'APLUSIX ont été récupérées lors d'une expérimentation à l'automne 2002, menant à la production d'un ensemble de 157 règles de transformations correctes et 67 règles de transformations erronées. Un second travail a succédé concernant la mise au point d'un algorithme de diagnostic des productions des élèves : ce travail a été entrepris côté informatique, en utilisant l'ensemble de règles ainsi défini. D'autres exemples pourraient être donnés.

3. Génération automatique de patrons d'exercices

3.1. Principes de la génération des exercices : notions de patron

Le but est de construire automatiquement des listes d'exercices pour l'entraînement ou l'apprentissage relativement aux différents types de problèmes algébriques (calculer, développer, factoriser, résoudre) en prenant en compte, dans

une certaine mesure, le curriculum français, sans y être attaché strictement. La mise en œuvre s'est effectuée en articulant une dimension didactique pour la détermination d'une hiérarchie de familles d'exercices, de leurs patrons, des variables et des domaines intervenant dans ces patrons, et une dimension informatique pour la mise en œuvre effective, l'implémentation de la hiérarchie avec héritage, l'instanciation des patrons, etc.

La notion centrale est celle de patron. Les patrons sont organisés en une hiérarchie dont les feuilles sont appelées des patrons effectifs et les nœuds des patrons abstraits. Les patrons effectifs comportent une expression algébrique, ex : « a(bx + c) + d(- ex + f) = gx + h », contenant des variables instanciées par un tirage aléatoire prenant en compte le domaine de définition des variables et des contraintes à respecter pour l'exercice obtenu.

3.2. La construction des patrons : la dimension didactique

Les patrons sont des formes algébriques et sémantiques dans la mesure où ils vont être construits relativement à des problèmes algébriques typés : calculer, développer, factoriser un polynôme, résoudre une équation, une inéquation ou un système d'équations. La structure hiérarchique commence avec 4 grands types de problèmes.

Développons l'exemple des patrons de type « Factoriser ». Ce type contient les patrons fabriqués en vue du type de problèmes : « Factoriser un polynôme P(x) d'une variable ». Le but étant d'engendrer des exercices effectifs, nous avons restreint les formes en fonction de certaines sous-variables. L'idée est d'engendrer différentes complexités pour les expressions proposées. Il ne s'agit pas seulement d'une complexité intrinsèque liée à la forme, mais aussi liée à la difficulté de la factorisation. Par exemple $(2x + 9)(5x + 1) + (3x - 4)(5x + 1) + (7x + 5)(5x + 1)$ est moins complexe à factoriser que $\frac{2}{9}x^2 - 3$ EMBED alors que la forme de la dernière ne contient que deux termes. Les résultats de recherche en didactique de l'algèbre, l'étude des programmes de l'enseignement secondaire et les difficultés repérées par les enseignants conduisent à prendre en compte comme premières variables didactiques potentielles [BROUSSEAU 1997] : la diversité des règles à utiliser, les formes associées et le nombre de transformations à mettre en œuvre ; la présence de transformations intermédiaires pour certains termes ou facteurs ; la profondeur de la factorisation ; le degré, la forme des polynômes et du facteur commun ; la présence ou non d'un facteur – 1 ; la présence ou non d'un facteur 1 implicite.

La plupart de ces variables peuvent être reprises pour les autres grands types de problèmes algébriques, mais certaines sont spécifiques du type « factoriser », comme celles de la dernière de la liste. La hiérarchie et la combinaison de ces variables conduisent à une première typologie. En complexifiant les formes par un jeu raisonné des variables précédentes, nous avons obtenu 5 grands types abstraits dans le type Factoriser, se répartissant en 14 sous-catégories (non décrites ici) :

- Type Factoriser 1 : Polynôme somme de termes constants et de monômes avec un facteur commun, autrement dit polynômes de forme générale AB + AC avec A, B et C constants ou monômes de degré 1 ou plus.

- Type Factoriser 2 : Polynôme somme de produits formés d'un facteur commun de degré 1 et de facteurs constants ou monômes, autrement dit polynômes de forme générale AB + AC avec A du premier degré, B et C de degré 1 au maximum

- Type Factoriser 3 : Polynôme somme de produits du premier degré, autrement dit polynômes de forme générale AB + AC avec A du premier degré, B et C constant ou de degré 1.

- Type Factoriser 4 : Polynômes développés liés aux identités remarquables

- Type Factoriser 5 : Polynôme complexe, somme de polynômes de types Factoriser 3 ou Factoriser 4. Le facteur commun n'est pas toujours apparent, une transformation intermédiaire et/ou une factorisation du type correspondant aux formes Factoriser 4 (liées aux identités remarquables) sont nécessaires. Le facteur commun est de la forme ax + b.

Les patrons se présentent ainsi comme des formes qui ouvrent encore des possibilités trop importantes ou pas assez réalistes dans certains contextes scolaires. Plus généralement les expressions doivent respecter la coutume didactique de certains niveaux de classe : les nombres doivent être simples et les calculs simplifiables en général. Pour obtenir des patrons effectifs, il s'avère nécessaire d'injecter d'autres variables pour définir le domaine D des coefficients, comme : le nombre de termes ; la nature des coefficients et leurs types d'écriture (entière, décimale, fractionnaire, etc.) ; les propriétés arithmétiques intrinsèques ou réciproques de certains coefficients (diviseurs, multiples, facteurs communs, etc.).

Des questions se sont alors posées dans l'équipe. S'il était clair pour chacun d'entre-nous qu'il fallait prendre en compte la nature des coefficients, les choix des écritures mathématiques et les instructions du langage informatique ont fait l'objet de débats dont nous restituons quelques questions : Est-il préférable de faire deux patrons ou de programmer directement un opérateur [+ ou –] pour avoir des termes a + b ou a – b ? Suffit-il de dire que b est un nombre relatif ? Engendre-t-on directement des rationnels en les déclarant comme tels, ou bien les engendre-t-on comme fractions à partir des entiers ? Des problèmes mathématiques et informatiques s'entrecroisent ici.

La génération d'exercices effectifs de ces patrons demande encore une restriction du domaine des coefficients prenant en compte la taille des nombres. Il est difficile de donner des critères de taille dans l'absolu. Nous nous sommes référés aux usages, notamment ceux issus des manuels. Ces questions ont fait l'objet d'échanges dans le groupe pour arriver à un compromis entre informaticiens et didacticiens.

Au final, une structure hiérarchique de profondeur 5 avec 98 patrons abstraits et 436 patrons effectifs a été obtenue.

3.3. La construction des patrons : la dimension informatique

La figure 1 donne un exemple de patron. La structure utilisée comporte une dizaine de champs. Les deux premiers (Nom, SorteDe) permettent de définir une hiérarchie de patrons et d'hériter d'attributs. Parmi les autres champs visibles, le lecteur reconnaîtra le patron lui-même, la définition du domaine pour chaque

variable et les contraintes à respecter. Parmi les champs non-visibles (obtenus par héritage), notons le type de problème associé au patron, le temps moyen accordé pour résoudre le problème, une pseudo fréquence d'apparition du patron.

```
{[nom FactorDistSTD23Z]
 [sorteDe FactorDistSTD]
 [patron <<ax^2+bx>>]
 [domaine ((c entier+ petit)(d entier+ petit)(e entier*
petit))]
 [avec ((<> c 1) (:= a (* d c)) (:= b (* e c)))]}

{[nom PatronEqDg1Groupement17]
 [sorteDe PatronEqDg1Groupement1]
 [patron <<[a/b]x+[c/e]=d/e>>]
 [domaine ((a parmi (-4 -3 -2 -1 1 2 3 4 5))(b parmi (1 2 3 4
5))
             (c parmi (-4 -3 -2 -1 1 2 3 4 5 6 7 8 9))
             (d parmi (-4 -3 -2 -1 1 2 3 4 5 6 7 8 9))
             (f parmi (2 3)))]
  [avec ((NEstPasDivisiblePar a b)(NEstPasDivisiblePar c b)

(NEstPasDivisiblePar d b) (<> d c)(:= e (* f b)) (<> d e))]}
```

Figure 1. *Deux patrons dont le patron FactorDistSTD23Z utilisé figure 2*

Pour instancier un patron, nous utilisons un mécanisme de tirage aléatoire non trivial qui assure le respect des contraintes, en invalidant certains patrons Par rapport au projet Wims ou à la spécification d'IMS-QTI [IMS-QTI 2004], ces contraintes constituent une originalité. Certaines de ces contraintes sont en fait des affectations, d'autres sont de vraies contraintes, elles permettent des vérifications locales ou globales et facilitent l'expression de certaines propriétés (divisibilité, degré d'une expression, existence de carré). Après instanciation des variables, des calculs supplémentaires sont effectués : suppressions des éléments neutres, utilisation des éléments absorbants, mise en forme.

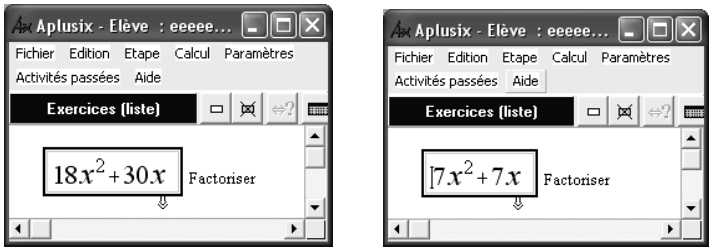

Figure 2. *Deux tirages aléatoires pour le patron FactorDistSTD23Z défini figure1*

Au fur et à mesure de l'avancement du travail, nous avons été obligés d'améliorer la robustesse de l'analyseur syntaxique et lexical que nous utilisions, ses reprises sur erreurs et les messages d'erreur qu'il fournissait, de telle sorte que les utilisateurs (didacticiens) gagnent de l'autonomie dans l'écriture des patrons.

4. Conclusion

Notre projet est clairement pluridisciplinaire, mais on peut se demander quel fonctionnement réel à cette pluridisciplinarité. Ce n'est pas une simple addition de points de vue disciplinaires. Le cadre de travail et la méthodologie ont permis d'établir une véritable coopération entre deux disciplines et des paradigmes autonomes pour atteindre un objectif commun. Bien sûr, les informaticiens auraient pu travailler de manière isolée, notamment en raison de leur expérience en mathématique, et produire des patrons. Un produit aurait pu être fourni « clé en main » aux consommateurs didacticiens et enseignants. On peut penser qu'alors, d'une part, le langage aurait été premier et que, d'autre part, les patrons n'auraient pas eu cette richesse qualitative et quantitative vis-à-vis de la structure et de la hiérarchie. Le cadre a en fait permis des boucles d'interaction entre les membres de notre communauté plurielle à partir des avancées de chacun. Lors de l'écriture formalisée des patrons, les problèmes rencontrés ont déplacé diverses dimensions, d'une part épistémique, avec la modification des patrons, voire de la structure, et d'autre part, informatique, en adaptant le langage de description lui-même.

Ce type de travail pluridisciplinaire ouvre la voie à des compléments. En perspectives, côté informatique, la réalisation d'une interface conviviale de description de patrons pour l'enseignant, permettant de tester les patrons imaginés ; côté didactique, la construction de patrons ayant un lien avec le curriculum de chaque pays et la mise au point d'un tableau qui indique pour chaque niveau scolaire les familles d'exercices du curriculum.

5. Bibliographie

[Brousseau 1997] Brousseau G., "Theory of didactical situations in mathematics", *Kluwer Academic Publishers*, 5.Dordrecht, 1997

[IMS-QTI 2004] IMS Question and Test Interoperability: "Item Implementation Guide" & "Item Information Model", V2.0 Editor Steve Lay, University of Cambridge, http://www.imsglobal.org/question/qti_item_v2p0pd page rédigée en Juin 2004.

[Nicaud 2004] Nicaud, J.F., Bouhineau, D. and Chaachoua H. "Mixing microworld and CAS features in building computer systems that help students learn algebra", in *Int^{al} Journal of Computers for Mathematical Learning*, Vol. 9.2, Springer-Verlag, 2004.