COMBIEN, un EIAH pour les dénombrements : expérimentation et leçons pour l'ingénierie

Gérard Tisseau, Hélène Giroire, Françoise Le Calvez, Jacques Duma, Marie Urtasun

▶ **To cite this version:**

Gérard Tisseau, Hélène Giroire, Françoise Le Calvez, Jacques Duma, Marie Urtasun. COMBIEN, un EIAH pour les dénombrements : expérimentation et leçons pour l'ingénierie. Environnements Informatiques pour l'Apprentissage Humain 2003, Apr 2003, Strasbourg, France. pp.509-516. edutice-00000178

HAL Id: edutice-00000178

https://edutice.archives-ouvertes.fr/edutice-00000178

Submitted on 5 Nov 2003

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

COMBIEN, un EIAH pour les dénombrements : expérimentation et leçons pour l'ingénierie.

Gérard Tisseau*, Hélène Giroire*, Françoise Le Calvez, Jacques Duma***, Marie Urtasun****

**LIP6 Université Paris6, Boîte 169, Tour 46-0 2° étage,*
4 Place Jussieu, F-75252 Paris Cedex 05.
Gerard.Tisseau, HeleneGiroire@lip6.fr
***CRIP5, Université René Descartes, 45 rue des Saints Pères,*
F-75270 Paris Cedex 06.
Francoise.Le-calvez, Marie. Urtasun@math-info.univ-paris5.fr
****Lycée technique Jacquard, 2 rue Bouret, F-75019 Paris.*
dumajd@club-internet.fr

RÉSUMÉ. *Nous présentons le système* COMBIEN *dans le cadre d'une expérimentation. L'objectif pédagogique est de faire maîtriser les concepts, représentations et raisonnements des mathématiques discrètes élémentaires, et cela par le biais d'une activité de résolution de problèmes en dénombrement. Le cadre est celui d'une situation d'apprentissage présentant certains obstacles que l'apprenant ne peut pas contourner, car il est contraint de suivre une démarche structurée et explicite. Les exercices sont choisis pour favoriser des conflits cognitifs provoquant la réflexion à l'occasion des erreurs, signalées à chaque étape avec quelques explications, mais sans donner la solution. Nous présentons dans la partie démonstrative de EIAH03 une visualisation de l'utilisation de ces machines. Une expérimentation a montré que le logiciel était apprécié pour sa simplicité et sa démarche structurée et détaillée. Il présente cependant des manques, en particulier pour la consolidation des connaissances. Nous tirons de cette expérimentation quelques leçons pour l'ingénierie des EIAH.*

MOTS-CLÉS : *expérimentation, situation d'apprentissage, conflit cognitif, ingénierie des EIAH, dénombrement.*

1. Introduction

Le système COMBIEN est un EIAH dans le domaine des dénombrements. Il a déjà fait l'objet de plusieurs publications qu'on peut trouver sur notre site web [COMBIEN 02]. Nous présentons ici notre analyse du système à la lumière d'une expérimentation faite avec des étudiants (dont le compte rendu est disponible sur le même site), en nous rapportant aux grandes problématiques de conception des EIAH, et en en tirant quelques leçons pour l'ingénierie. Nous illustrons cet article par des opinions des étudiants sur le système, recueillies après l'expérimentation à l'aide d'un questionnaire ainsi que par des extraits d'enregistrements de certains groupes en train d'utiliser le système.

2. Objectifs pédagogiques

Définir un objectif pédagogique n'est pas une chose aisée, comme le montre par exemple l'évolution lente de la formulation des programmes d'enseignement des mathématiques en France. Il y a une trentaine d'années, ils étaient rédigés uniquement en termes de contenus, puis sont apparus progressivement des commentaires sur ce qui était exigible ou non, des thèmes de travaux pratiques. Cette évolution a abouti à trois aspects : les objectifs généraux (motivations, problématiques), les contenus, et les modalités de mise en œuvre (exemples de présentations, liens à faire). Depuis peu sont publiés des documents [CNDP 01] précisant encore plus les compétences à développer chez l'élève et par quel type de méthode (« veiller au caractère progressif et actif de l'apprentissage », « Questions et exemples seront systématiquement utilisés », « On veillera à la qualité des raisonnements », « le calcul est une activité qui est indispensable pour progresser »).

Dans le système COMBIEN, l'objectif perceptible par l'apprenant est d'acquérir de bonnes performances dans un certain type d'exercices de dénombrement. Mais ce n'est pas l'objectif essentiel : il s'agit surtout de maîtriser les concepts associés, de se former des représentations mentales efficaces, de mettre en forme des raisonnements utilisant les notions de base de la logique et des ensembles, de se familiariser avec les activités de modélisation conceptuelle et de programmation. Il s'agit d'un objectif *large* et à long terme plus difficile à évaluer qu'un objectif *focalisé* et à court terme.

3. Le domaine et sa modélisation

Un EIAH a pour but d'aider un apprenant à acquérir des connaissances et des compétences dans un certain *domaine*. Souvent, et en particulier en mathématiques, il s'agit d'un domaine qui a déjà été formalisé et qui fait l'objet d'un enseignement institutionnel. Cependant, la formalisation institutionnelle est souvent abstraite et générale, et bien qu'elle soit suffisante en théorie pour résoudre les problèmes du domaine, elle ne répond pas entièrement aux besoins d'un EIAH. Dans le domaine

des dénombrements par exemple, nous avons dû introduire des concepts pour représenter les problèmes et leurs solutions (construction, étape, filtre, propriété, multiplicité), et des connaissances relatives à ces concepts (contraintes relatives à une construction correcte, comparaison de deux constructions, méthode constructive).

Ce besoin d'augmenter la théorie institutionnelle est manifeste dans d'autres EIAH, par exemple dans APLUSIX [NICAUD 89], ce qui a amené les concepteurs à formuler une « théorie cognitive et computationnelle de l'algèbre » [GELIS 94]. Le but n'est pas d'abolir la théorie institutionnelle, mais de fournir des moyens d'expression et de représentation adaptés à un système interactif à visée pédagogique. Les augmentations se placent principalement au niveau méta : on représente explicitement ce qu'est un problème, une solution, une erreur, une étape de raisonnement, etc. On spécialise aussi la théorie générale en introduisant des concepts auxiliaires correspondants aux problèmes particuliers traités, ce qui facilite l'expression des connaissances et des explications. Ces concepts manquent dans la théorie institutionnelle parce que celle-ci se veut minimale et générale, et ne présente que les bases qui suffisent logiquement. D'autre part, elle ne s'intéresse qu'aux raisonnements *justes* mais elle ne cherche pas de critères pour analyser des raisonnements *faux*, ce qui nécessite une connaissance plus profonde.

L'expérimentation semble montrer que l'introduction de concepts explicites permet de structurer la démarche et de mieux y réfléchir. Utiliser explicitement un vocabulaire (« construction », « étape », « propriété ») permet aux apprenants de prendre conscience d'un cadre dans lequel ils peuvent exploiter les connaissances théoriques générales, et de l'intérêt d'un tel cadre : un étudiant parle de « sentiment d'être aidé par la démarche imposée… (donne un cadre clair) ». Ils ont apprécié le fait que la démarche était structurée et les obligeait à détailler la solution « on apprend à bien décomposer la résolution de l'exercice en étapes », « oblige à expliciter clairement son raisonnement et à décomposer en plusieurs étapes simples ».

4. Situation d'apprentissage

La situation d'apprentissage proposée dans le système COMBIEN se présente comme une activité de résolution de problèmes : l'apprenant doit résoudre une série d'exercices, dans un ordre prévu pour être progressif. Le domaine des mathématiques discrètes est spécifique en ce qui concerne les démarches de preuve et de modélisation [GRENIER 98]. Contrairement à de nombreux types de problèmes mathématiques, la résolution ici ne consiste pas à enchaîner des inférences pour déduire de nouveaux faits ou à enchaîner des règles de réécriture pour transformer une expression. Ici, on part d'un système de contraintes et on s'intéresse à l'ensemble des objets (appelés *configurations*) vérifiant ces contraintes, le but étant de déterminer leur nombre. La difficulté est d'abord de l'ordre de la représentation et de la modélisation.

La démarche proposée ici consiste à définir une *construction* qui permettrait d'engendrer une énumération exhaustive des configurations satisfaisant les contraintes. En ce sens, l'activité proposée se rapproche d'une activité de *programmation*. D'ailleurs un des étudiants l'a bien vu : il a parlé de « l'écriture du *programme* », alors que ce mot ne figurait nulle part dans les instructions ou l'interface. Cela fait intervenir en fait différentes activités : modélisation, spécification, utilisation rigoureuse d'un langage, tests, preuve de programme. C'est la variété de ces activités, leur intégration dans un même thème et leur proximité avec l'outil informatique qui nous ont fait choisir ce domaine et ce type de démarche.

5. Approche pédagogique : situation-problème, conflit cognitif

La stratégie que nous utilisons pour favoriser l'apprentissage est celle des « situations-problèmes » [MEIRIEU 93]. Il s'agit de proposer une tâche présentant certains obstacles, dans un dispositif qui introduit d'une part des *contraintes* empêchant de contourner les obstacles mais qui fournit d'autre part des *ressources* permettant d'acquérir les connaissances pour surmonter les obstacles. Cette stratégie tend à provoquer des conflits cognitifs que l'apprenant doit essayer de résoudre.

Le système COMBIEN semble réaliser une situation-problème opérationnelle, d'après nos observations et les appréciations des étudiants. Ce que les étudiants ont apprécié correspond à ce nous attendions d'une telle situation : le système oblige l'apprenant à expliciter son raisonnement en suivant une méthode structurée, ce qui entraîne une réflexion permettant de construire une solution claire et correcte. D'autre part, cette méthode n'est pas radicalement nouvelle pour les apprenants, elle ne fait que formaliser rigoureusement une approche intuitive, comme le montrent les solutions qu'ils proposent (parfois fausses) avant l'utilisation du système. Vous trouverez dans la rubrique communication démonstrative sur le Web une démonstration détaillée de l'utilisation du système.

L'*obligation* est ressentie au début comme une gêne, un inconfort, d'abord parce qu'elle impose à l'apprenant d'affronter certains obstacles embarrassants sans pouvoir les contourner, ensuite parce qu'elle limite parfois ses possibilités d'expression, et enfin parce qu'elle semble détourner de ce qui est, pour la plupart des étudiants, l'objectif principal : obtenir la formule finale de dénombrement. Effectivement, nous avons constaté que la résolution du premier exercice demandait plus de temps que les autres (mais peut-être est-ce aussi une question d'ergonomie). Mais assez vite, après la résolution de quelques exercices, l'obligation est ressentie comme une *aide*, un garde-fou, un guide. L'*explicitation* permet de rendre visibles les concepts intéressants et ainsi de se rendre compte de leur existence et de leur importance. Et enfin la *structuration* permet d'organiser ces concepts en une méthode, ce qui permet de former des intentions, des prévisions, des plans.

6. Systèmes apparentés

Les systèmes apparentés ne sont pas à chercher en mathématiques classiques (algèbre, analyse, géométrie), à cause des spécificités mentionnées plus haut. Le système le plus proche dont nous ayons connaissance est SQL-Tutor [MITROVIC 98]. SQL-Tutor est un Tuteur Intelligent où la tâche de l'apprenant est d'écrire une requête en SQL (langage de bases de données) pour traduire une question posée informellement. Cette tâche ressemble par bien des aspects à celle que nous proposons à l'élève dans COMBIEN. Là où SQL-Tutor demande : « écrire une requête SQL qui, si elle était exécutée, fournirait toutes les données vérifiant telles propriétés », COMBIEN demande : « écrire une construction qui, si elle était exécutée, fournirait toutes les configurations vérifiant telles propriétés ». Mais nous ajoutons : « puis, en analysant cette construction (sans l'exécuter), en déduire le nombre de configurations solutions ». Les étapes d'une construction ressemblent effectivement à des requêtes SQL. Par exemple, dans l'étape : « choisir 1 élément dont la couleur n'est pas Cœur et la couleur n'est pas Pique », la propriété pourrait s'écrire en SQL : WHERE COULEUR <> 'Cœur' AND COULEUR <> 'Pique', COULEUR étant un attribut et 'Cœur' et 'Pique' étant des valeurs. Dans COMBIEN, cette construction est exprimée par l'élève interactivement et de manière guidée, alors qu'elle s'écrit dans un langage formel dans SQL-Tutor, où l'objet de l'apprentissage est le langage SQL lui-même.

7. Choix des exercices

Après une étude exhaustive des erreurs possibles dans le cadre du système, nous avons choisi des exercices donnant l'occasion de provoquer des conflits cognitifs. Cela a permis d'obtenir des exercices de difficultés variées, et considérés comme « non classiques » par certains enseignants qui ont testé le système. Les opinions des étudiants sont diverses : « ils sont intéressants et pas trop difficiles », « ils rassemblent toutes les questions susceptibles d'être posées », « variés et de difficulté croissante », mais aussi « ils sont tous plus ou moins du même type ». En effet, tous les exercices parlaient de jeux de cartes. Il y a d'autres exercices, mais les 9 proposés demandaient déjà une séance de presque deux heures. De même, il y a d'autres interfaces pour d'autres types de problèmes, mais elles n'ont pas été testées lors de cette expérimentation.

La plupart des « pièges » ont effectivement fonctionné. Par exemple, l'exercice « avec un jeu de 32 cartes, combien peut-on former de mains de 5 cartes avec exactement 2 piques et exactement 2 cœurs ? » provoque fréquemment la construction suivante : « je choisis 2 piques, puis je choisis 2 cœurs, et j'ai terminé », ce qui est faux puisqu'alors la main n'a que 4 cartes, pas 5. La machine indique : « vous n'avez que 4 cartes, alors que vous en voulez 5 d'après votre propre définition de l'univers ». Cette contradiction crée un conflit cognitif qui, nous l'espérons, permet d'assimiler une règle : dans une construction, il faut parfois introduire des contraintes qui ne figuraient pas explicitement dans l'énoncé (on veut 1 carte ni cœur ni pique). L'observation d'un groupe ayant fait cette erreur au début a permis de voir

qu'effectivement ce principe avait été compris : 5 exercices plus tard, un des élèves propose de terminer trop tôt, un autre s'en aperçoit et dit qu'il faut continuer : « Mais on est obligé, sinon on n'aurait pas 18 cartes ».

8. Les erreurs

Les interventions tutorielles du système se produisent lorsque l'apprenant commet des erreurs, détectées lors de certains *moments de validation*. Des choix importants interviennent donc dans la conception d'un EIAH : décider quels sont les moments de validation, quelles erreurs l'interface empêche de se produire (en ne fournissant que des options correctes par exemple) et pour celles qui restent possibles quelle doit être la réaction du système et quand elle doit avoir lieu (immédiatement ou de manière différée). Cependant ces choix sont difficiles à faire a priori et il est inévitable d'avoir à tester différentes versions du système avec des choix différents.

On peut en tirer une leçon pour l'ingénierie [TCHOUNIKINE 02] : l'architecture d'un EIAH doit permettre de changer facilement ces choix pour essayer différentes possibilités. Dans COMBIEN, cette flexibilité est obtenue par la *réification* (chaque concept important, comme celui de schéma d'erreur ou de moment de validation, est représenté par un objet informatique structuré, paramétrable, possédant des connaissances procédurales et déclaratives), la *structuration* (les objets font partie de structures et leur position dans ces structures est utilisée par les mécanismes, par exemple pour savoir quelles erreurs tester et quand), et la *déclarativité* (les connaissances, les paramétrages et les structures sont exprimés le plus possible sous forme déclarative, dans un langage conçu pour l'occasion : Descript). Le système possède une base de schémas d'erreurs implémentée suivant ces principes. Cette représentation rejoint par certains aspects celle qui est utilisée dans SQL-Tutor, fondée sur des contraintes (Constraint-Based Modeling [OHLSSON 92]).

Les opinions des étudiants sont parfois contradictoires : « Les erreurs sont signalées immédiatement » (cité dans les points forts), « Il est dommage que les erreurs soient données en cours d'exercices et non à la fin » (cité dans les points faibles). Cela montre bien, comme le souligne [MATHAN 02], que le débat sur le « bon » délai de réaction aux erreurs (immédiat ou différé) est faussé car la réponse dépend du modèle de performance visé et des styles individuels d'apprentissage. Nous prévoyons d'introduire un paramètre permettant de choisir l'instant du signalement des erreurs (immédiatement ou à la fin).

Il y a deux façons de montrer des erreurs : soit en disant explicitement qu'il y a une erreur et en expliquant quel principe est mis en défaut, soit en donnant un contre-exemple, c'est-à-dire une configuration (engendrée par la construction de l'étudiant) qui ne vérifie pas les contraintes de l'énoncé. Cette deuxième technique est plutôt une *visualisation* des erreurs qu'une *explication*. Cette visualisation peut être faite actuellement dans un dispositif qui montre (à la demande de l'étudiant) un exemple aléatoire de configuration engendrée. Mais comme la génération est

aléatoire, la configuration ne présente pas forcément d'erreur. Nous prévoyons donc de modifier le dispositif pour qu'il engendre systématiquement un contre-exemple s'il y en a un. Ce problème se retrouve dans les environnements de modélisation/simulation, dans lesquels la simulation peut montrer à l'apprenant les conséquences de sa modélisation erronée [**HIRASHIMA 98].**

9. Conclusion

Le système COMBIEN réalise une situation-problème : il donne à l'apprenant un but immédiatement compréhensible (répondre à la question « combien ? ») ; ce but n'est pas atteignable directement (il nécessite plusieurs étapes d'action, de représentation et de raisonnement) et il n'y a pas de procédure algorithmique systématique pour y arriver ; l'activité peut produire des conflits cognitifs (parfois l'énoncé contient la contrainte « il faut 3 cœurs » et pourtant une étape disant « je tire 3 cœurs » est fausse) ; la démarche et la solution ne sont pas évidentes, mais elles sont ressenties comme accessibles et le sont effectivement (les 9 exercices ont été réussis par des débutants lycéens), ce qui fait que les apprenants se prennent au jeu et s'engagent activement dans la recherche ; il y a plusieurs façons d'aboutir à la solution (un étudiant dit : « les différents raisonnements valables pour arriver à la solution peuvent être explicités : pas un seul chemin possible ») ; pour aboutir à la solution, il faut mettre en œuvre (et parfois découvrir) certaines démarches et connaissances qui sont l'objectif de l'apprentissage, et cela est rendu incontournable grâce aux contraintes imposées.

Le fait d'utiliser un environnement informatique pour réaliser cette situation présente des avantages : il contrôle systématiquement le respect des *contraintes* empêchant de contourner l'apprentissage (ici l'obligation d'utiliser une méthode constructive et d'expliciter les étapes) ; il fournit des *ressources* aussi bien explicites (les aides, les messages d'erreur) qu'implicites (le langage utilisé, l'organisation et le contrôle de l'interface) ; il offre des moyens de *visualisation* (exemples et contre-exemples engendrés dynamiquement et en contexte) ; il libère de certaines *tâches annexes* non pertinentes pour l'objectif visé (calculs numériques, mise en forme soignée) ; il empêche les *erreurs non instructives* (inhibition automatique de certaines possibilités, listes contrôlées de valeurs acceptables).

La mise au point d'un EIAH réalisant ces fonctions avec une bonne efficacité pédagogique n'est pas possible uniquement à partir de conceptions a priori. Il est nécessaire d'essayer et de tester de nombreuses variantes. Pour cela, il faut que l'architecture permette de configurer ces variantes aussi simplement que possible. Nous avons trouvé que des outils efficaces dans ce but étaient la réification, la structuration et la déclarativité.

Toute situation-problème nécessite une *consolidation* ultérieure pour que les acquis soient explicités et réutilisés. Actuellement, cette consolidation n'est pas faite dans le système. Nous prévoyons d'ajouter des dispositifs qui permettraient de l'amorcer (par exemple demander une explicitation de la preuve d'équivalence entre énoncé et construction).

L'évaluation de l'efficacité du système pour l'apprentissage est difficile à cause du type d'objectif, large et à long terme. Elle se fera nécessairement sur une période plus longue qu'une simple expérimentation ponctuelle, et elle devra faire intervenir les interfaces correspondant aux différents types de problèmes, suivant la progression prévue.

9.1. Références bibliographiques

[CNDP 01] Ministère de l'Éducation nationale, Direction de l'Enseignement scolaire, « Mathématiques classe de première des séries générales », *série Accompagnement des programmes*, CNDP, 2001.

[GELIS 94] GELIS J.M., Eléments d'une théorie cognitive et computationnelle de l'algèbre. Application au cas de la factorisation d'expressions polynomiales. Thèse de l'Université de Paris XI, 1994.

[GRENIER 98] GRENIER D., PAYAN Ch., « Spécifités de la preuve et de la modélisation en mathématiques discrètes ». *Recherches en Didactique des Mathématiques*, Vol. 18, n°1, 1998, p. 59-100.

[HIRASHIMA 98] HIRASHIMA T., HORIGUCHI T., KASHIHARA A., TOYODA J., : « Error-Visualization by Error-Based Simulation », *International Journal of AIED*, Vol.9, 1998, pp.17-31

[MATHAN 02] MATHAN S.A., KOEDINGER K.R., « An Empirical Assessment of Comprehension Fostering Features », *ITS 2002*, Biarritz, LNCS 2363, Springer-Verlag, 2002, pp. 330-343,

[MITROVIC 98] Mitrovic A., « Experiences in Implementing Constraint-Based Modeling in SQL-Tutor ». *Proceedings of ITS'98*, pp. 414-423.

[MEIRIEU 93] MEIRIEU P., « Objectif obstacle et situations d'apprentissage ». *La pédagogie : une encyclopédie pour aujourd'hui*, ESF, Paris, 1993, p. 289-300.

[NICAUD 89] NICAUD J.F., « APLUSIX : Un système expert pédagogique et un environnement d'apprentissage dans le domaine du raisonnement algébrique ». *Technique et Science Informatiques*, Vol. 8, n°2, Hermès, 1989.

[OHLSSON 92] Ohlsson S., « Constraint-based student modeling ». *Journal of Artificial Intelligence in Education*, 3(4), p. 429-447.

[TCHOUNIKINE 02] TCHOUNIKINE P., « Pour une ingénierie des Environnements Informatiques pour l'Apprentissage Humain », *Information-Interaction-Intelligence*, Volume 2, n°1, Cépaduès-Edition 2002.

9.2. Références sur le web

[COMBIEN 02] http://www.math-info.univ-paris5.fr/combien/