Assistance à la conception de techniques de diagnostic des connaissances

Sébastien Lallé*,, Vanda Luengo*, Nathalie Guin****

** LIG METAH,Université Joseph Fourier*
110 av. de la Chimie, BP 53, 38041 Grenoble cedex 9
{sebastien.lalle, vanda.luengo}@imag.fr
*** Université de Lyon, CNRS*
Université Lyon 1, LIRIS, UMR5205, F-69622, France
nathalie.guin@ liris.univ-lyon1.fr

RÉSUMÉ. *Cet article propose une méthodologie d'assistance pour la conception de techniques de diagnostic des connaissances. La principale contribution est une méthodologie indépendante tant du domaine que du type de technique de diagnostic. Elle permet ainsi la conception de plusieurs techniques de diagnostic pour un domaine sans nécessiter la modélisation de chaque outil indépendamment. La méthode est semi-automatique : elle repose sur une ontologie des connaissances utilisée pour guider un algorithme d'apprentissage automatique. Une expérimentation évalue cette méthodologie sur deux domaines différents, la chirurgie orthopédique et la lecture de l'anglais. Les résultats montrent que les techniques de diagnostic conçues présentent une précision de la prédiction supérieure à la classe majoritaire, en validation croisée. Les tests statistiques RMSE et AIC permettent également de comparer les performances des techniques conçues.*

MOTS-CLÉS : *diagnostic des connaissances, outil auteur, apprentissage automatique*

1. Introduction

Le diagnostic des connaissances est le processus qui, prenant en entrée des traces d'apprenants, infère ou met à jour un modèle des connaissances, sues ou non, de l'apprenant. Les traces sont la collection temporellement située des interactions des apprenants avec l'EIAH. Le diagnostic des connaissances, en estimant l'évolution de l'état des connaissances d'un apprenant, peut être utilisé pour fournir des rétroactions, personnaliser l'EIAH, choisir les prochains exercices à pratiquer, etc. Il existe dans la littérature des techniques génériques de diagnostic des connaissances, réutilisables dans plusieurs domaines. Dans un précédent article [LALLE et al. 12], nous avons défini la notion de technique de diagnostic comme l'association entre un modèle générique de diagnostic (par exemple le Knowledge Tracing [CORBETT & ANDERSON 95]) et une implémentation (réseaux bayésiens, règles de production, modèles linéaires…). Un modèle de diagnostic est épistémique (inclut un modèle des connaissances), global (raisonne sur un ensemble de traces pour déterminer pour chaque apprenant les connaissances sues et non sues du modèle des connaissances), et générique (le modèle ne dépend pas du domaine). L'objet de cet article étant l'assistance à la conception de techniques de diagnostic, telles que définies ci-dessus, nous ne détaillons pas la formalisation des modèles de diagnostic ici.

Le développement d'une technique de diagnostic des connaissances est une tâche complexe, dépendant du domaine, de l'utilisation visée dans l'EIAH, et des paradigmes de modélisation et d'implémentation existants. Actuellement, certains outils existants pour la construction d'une technique de diagnostic sont indépendants du domaine, i.e. permettent de construire une même technique de diagnostic dans plusieurs domaines (par exemple, le Bayesian Knowledge Tracing a été appliqué en algèbre, chimie, chinois…), mais dépendent d'une technique de diagnostic. Cela signifie que ces outils de construction ne proposent qu'une seule technique, sans possibilité d'en considérer d'autres. Le verrou que cet article aborde est donc l'assistance à la conception de techniques de diagnostic des connaissances, de façon indépendante tant du domaine que des techniques de diagnostic – i.e. construire plusieurs techniques de diagnostic. Nous proposons pour ce problème une méthodologie d'assistance, réifiée dans une première plateforme pour son évaluation. La principale question que pose ce verrou est la possibilité d'assister la conception sans duplication du travail pour chaque technique de diagnostic.

L'utilisateur cible de notre plateforme est un concepteur de techniques de diagnostic. Pour un expert en diagnostic des connaissances, la plateforme doit permettre de construire, évaluer et comparer diverses techniques de diagnostic sur plusieurs domaines et plusieurs jeux de traces. Pour un expert d'autres domaines des EIAH mais souhaitant réutiliser une technique de diagnostic pour un EIAH donné, la plateforme doit permettre d'assister le développement et de réduire le coût de programmation, mais aussi fournir des résultats permettant au concepteur de choisir la technique de diagnostic la plus adaptée au domaine et aux traces considérées.

L'article présente en premier lieu un état de l'art sur le sujet, puis la méthodologie de conception de techniques de diagnostic fondée sur un algorithme d'apprentissage semi-automatique. Nous montrons ensuite les résultats obtenus sur deux domaines différents, avant de conclure et de dresser quelques perspectives de recherche.

2. Travaux sur la conception de techniques de diagnostic

Il existe deux principales approches pour la conception assistée de techniques de diagnostic : les outils auteurs et l'apprentissage automatique.

La construction de techniques de diagnostic est intégrée dans plusieurs outils auteurs, i.e. des environnements permettant de construire tout ou partie d'un EIAH en limitant le coût de programmation. Eon [MURRAY 03], CTAT [ALEVEN et al. 06], SimStudent [MATSUDA et al. 05], Cognitive Model SDK [BLESSING & GILBERT 08] et ASPIRE [MITROVIC et al. 06] permettent de construire un modèle du domaine et des connaissances, puis une technique de diagnostic de type : système expert basé sur des règles dans Eon, Knowledge Tracing [CORBETT & ANDERSON 95] pour CTAT, Simstudent et Cognitive Model SDK, et Constraint-based [OHLSSON 94] pour ASPIRE. Dans Eon, CTAT, Cognitive Model SDK et ASPIRE, le modèle des connaissances est créé via l'interface graphique. La technique de diagnostic est développé à la main dans Eon, CTAT et Cognitive Model SDK (écriture de règles ou de contraintes) ; ASPIRE peut lui générer des contraintes à partir du modèle du domaine (exprimé par une ontologie), mais les contraintes plus complexes doivent être écrites manuellement. SimStudent permet de construire aussi bien le modèle de connaissances que la technique de diagnostic à partir d'exemples.

L'inconvénient des outils auteur est la nécessité de concevoir plusieurs composants de l'EIAH, notamment l'interface, avant le développement de la technique de diagnostic des connaissances. Cela limite la possibilité de changer d'outil auteur en cours de développement, ainsi que de réutiliser un EIAH ou des composants d'EIAH existants, dont des interfaces complexes (comme dans le cas d'un simulateur ou d'un micromonde). Seul le Cognitive Model SDK adopte une approche modulaire pour le développement d'une technique de diagnostic. La seconde limite est la restriction à une seule technique de diagnostic, sans possibilité de changer. Créer plusieurs techniques pour les comparer requiert d'utiliser plusieurs outils auteur, et donc de construire plusieurs fois les mêmes composants d'un EIAH. Enfin, Eon, CTAT et en partie ASPIRE requièrent d'écrire des règles de production ou des contraintes manuellement, ce qui représente un coût de programmation. SimStudent propose une solution à ce problème en apprenant à partir d'un ensemble d'exemples de résolution d'un problème les règles de production associées à chaque étape du problème.

Une seconde approche pour la création de techniques de diagnostic est l'apprentissage automatique, c'est-à-dire l'extraction d'une technique de diagnostic à partir de traces d'apprenants collectées dans un domaine. Il en résulte une technique de diagnostic dite apprise ou instanciée. Dans le cas de techniques génériques de diagnostic, nous parlons d'instanciation automatique de la technique au domaine. Plusieurs auteurs ont discuté l'apprentissage automatique de modèles d'apprenants : [SISON & SHIMURA 98] pour des librairies d'erreurs, [DESMARAIS et al. 06] pour l'Item-Response Theory, ou [GONZALES-BRENES & MOSTOW 12] pour un réseau bayésien dérivé du Knowledge Tracing. ADVISOR [BECK et al. 00] permet également l'apprentissage d'un modèle d'apprenant ainsi que d'un outil de feedback au moyen d'un modèle linéaire. Les modèles d'apprenant appris, évalués soit par des experts, soit par des méthodes statistiques (taux de bonne prédiction, vraisemblance), ont montré de bons résultats.

Le problème de l'apprentissage automatique est l'interprétabilité des techniques de diagnostic apprises, qui peuvent être opaques pour un humain. C'est particulièrement le cas pour les variables cachées (non observées dans les traces) comme les connaissances. Beck [BECK 07] a également montré que les algorithmes d'apprentissage peuvent extraire des informations contre-intuitives, par exemple une connaissance qui serait toujours diagnostiquée comme maîtrisée dès le premier exercice la mettant en jeu ; il s'agit d'un

problème de plausibilité. Il est dans ce cas difficile pour un expert d'utiliser la technique de diagnostic pour des tâches comme le feedback ou la personnalisation de l'EIAH, posant la question de l'utilité du diagnostic. En second lieu, tout comme les outils auteur, ces algorithmes ne permettent d'apprendre qu'une seule technique de diagnostic.

En conclusion, il existe un verrou pour proposer des outils permettant de construire plusieurs techniques de diagnostic. C'est ce problème que nous étudions dans cet article. Notre approche et la plateforme résultante permettent la création de techniques de diagnostic des connaissances pour un EIAH existant, sans nécessiter de concevoir un EIAH complet, à la différence de nombreux outils auteurs. Pour éviter au concepteur de modéliser chaque technique indépendamment, nous utilisons également un algorithme d'apprentissage, conçu pour instancier à partir de données des techniques génériques de diagnostic. Cet algorithme est semi-automatique (c'est-à-dire qu'il nécessite un certain degré d'intervention humaine) pour garantir l'interprétabilité, la plausibilité et l'utilité du diagnostic.

3. Méthode semi-automatique d'instanciation de techniques génériques de diagnostic

Le problème est d'instancier un ensemble de techniques génériques de diagnostic, i.e. comme défini en introduction l'association d'un modèle générique de diagnostic et d'une implémentation. Pour cet article, nous formulons le postulat suivant : un modèle générique de diagnostic des connaissances peut être défini comme un graphe orienté G et un ensemble de contraintes sur les cardinalités du graphe. Par exemple, le Knowledge Tracing est un graphe à trois types de nœuds (problèmes, étapes du problème, connaissances cognitives), avec les deux contraintes suivantes : une étape a pour parent un et un seul problème, et une étape a pour fils une et une seule connaissance.

Pour éviter le travail fastidieux de modéliser chaque technique séparément, notre méthode utilise un algorithme d'apprentissage permettant d'instancier chaque modèle de diagnostic. Toutefois, extraire un modèle sémantiquement riche à partir de traces quelconques n'est pas un problème informatique solvable dans un cas général. Un apport de sémantique est nécessaire. De plus, nous faisons l'hypothèse qu'il est possible de conserver l'interprétabilité et la plausibilité de la technique. Enfin, des connaissances peuvent ne pas être observées dans les traces. Pour pallier à ces problèmes, l'algorithme d'apprentissage est guidé par une ontologie des connaissances, dont les éléments sont liés aux traces pour apporter la sémantique. Une seule ontologie est réalisée pour l'ensemble des techniques, car elle est liée aux traces et non à chaque technique de diagnostic.

La plateforme possède une base de techniques de diagnostic, i.e. la formalisation des modèles génériques et des méthodes d'inférences pour les implémenter. L'objectif est d'instancier (d'appliquer) ces techniques au domaine à partir de traces, de sorte que les traces du concepteur et les connaissances de son domaine puissent être prises en compte. Il en résulte des techniques de diagnostic applicables aux traces ou à l'EIAH du concepteur. La plateforme assiste chaque concepteur dans cette tâche, via le processus suivant : le concepteur crée une ontologie des connaissances du domaine et la met en correspondance avec ses traces. Puis l'algorithme instancie au domaine chaque technique générique de sa base à partir des traces, de l'ontologie et de la mise en correspondance, et apprend les paramètres.

Quatre techniques de diagnostic sont actuellement proposées dans la plateforme, dont les détails des implantations peuvent être trouvés dans [LALLE et al. 12] :

– Knowledge Tracing [CORBETT & ANDERSON 95] et Additive Factor Model [CEN et al. 08] (deux modèles cognitivistes), deux implémentations (respectivement avec un modèle de Markov caché et un modèle linéaire) pour les diagnostics cognitivistes ;

– Constraint-based [OHLSSON 94], implanté avec des contraintes (règle SI/SINON) auxquelles sont associées une régression ;

– Control-based [MINH CHIEU et al. 10], implanté avec un réseau bayésien dynamique.

3.1. Modélisation des traces et des connaissances

L'ontologie centrée sur les connaissances d'un domaine donné est fondée sur une partie d'une ontologie existante : CREAM-C [NKAMBOU et al. 97], qui est compatible avec toutes les techniques de diagnostic considérées. L'ontologie se compose de deux parties : les capacités (ou connaissances) et les transitions exprimant les informations observables dans les traces. Le terme transition est utilisé pour suggérer la transition de l'état observable d'un problème à un autre. Deux niveaux d'ontologies peuvent être construits :

– ontologie de haut niveau, où les classes de l'ontologie représentent les variables présentes dans les traces ;

– ontologie détaillée, où les classes peuvent représenter les variables des traces ainsi que des exemples de valeurs de ces variables dans les traces.

La figure 2 montre deux exemples d'ontologies, une de haut niveau (gauche) et une détaillée (droite), dans le domaine de la chirurgie orthopédique. Les classes héritent des deux classes Capacity et Transition, exprimant le lien entre éléments observables et connaissances. L'ontologie détaillée montre trois exemples d'individus, un pour la classe Operator (action) et deux pour la classe Skill (connaissance). Les flèches pleines sont les relations d'héritage et les fines flèches grises les relations d'appartenance d'un individu à une classe. Les flèches pointillées expriment les relations objet entre classes ; des relations objet sont proposées par défaut, comme « a une capacité » de Transition à Capacité, et la relation inverse « a une transition », ou la relation « a pour préalable » de Capacité vers Capacité. Le concepteur peut définir de nouvelles relations pour enrichir la sémantique.

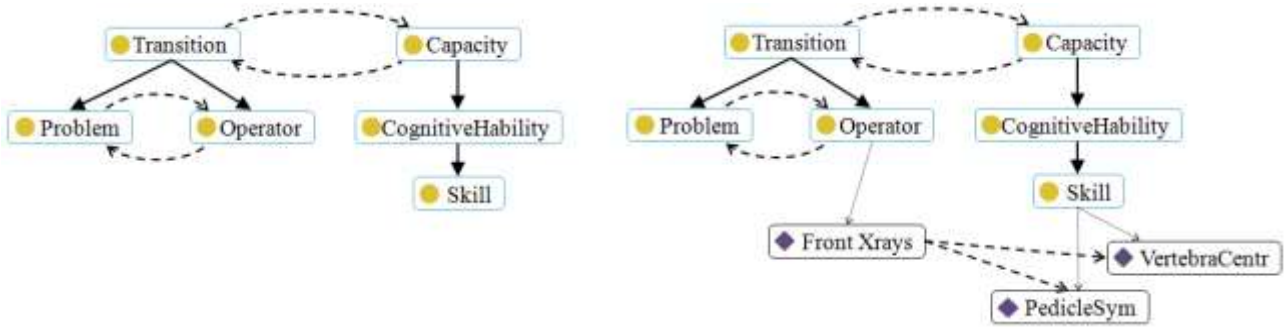


Figure 1. *Deux exemples d'ontologie, de haut niveau à gauche, détaillée à droite. Les ronds désignent les classes de haut niveau et les losanges les classes du niveau détaillé. Les flèches pleines épaisses sont les relations d'héritage, les flèches pointillées les relations objets et les flèches grises fines les relations d'appartenance à une classe.*

Le concepteur crée ensuite une mise en correspondance entre l'ontologie et les traces : une ou plusieurs classes ou individus de l'ontologie sont associés à un ou plusieurs éléments observés dans les traces. Cette mise en correspondance permet de lier aux traces la sémantique apportée par l'ontologie. Plusieurs degrés d'informations expertes sont possibles en fonction : du niveau de l'ontologie (haut niveau ou niveau détaillé) ; de la complétude de l'ontologie ; de la complétude de la mise en correspondance. Notre plateforme ne permet pas d'estimer la quantité d'information experte requise pour instancier les techniques de diagnostic, mais l'approche automatique permet de tester et évaluer

plusieurs degrés de complétude de l'ontologie, d'abord de haut niveau puis au niveau détaillé.

3.2. Algorithme d'instanciation des techniques

Comme introduit dans l'état de l'art, les méthodes d'apprentissage automatique dépendent du modèle de diagnostic, et de l'implémentation utilisée (réseau bayésien, logique…). Notre approche consiste à utiliser des algorithmes différents pour chaque implémentation possible dans la plateforme, mais relevant tous d'une méta-stratégie, afin de proposer un algorithme unique et équitable. De ce point de vue, l'approche par score est particulièrement adaptée : elle consiste à coupler un algorithme d'exploration des solutions possibles et un score permettant d'évaluer ces solutions. Nous utilisons une seule méthode d'exploration, et proposons un score applicable à toutes les techniques de diagnostic. L'algorithme met donc en jeu un espace des solutions possibles (issues des traces *T*), un algorithme d'exploration des solutions *search()*, c'est-à-dire de sélection de solutions candidates à évaluer, et un score *S* mesurant la qualité de chaque solution candidate. *T* et *S* sont des paramètres de l'algorithme *search()*.

Dans notre plateforme, nous réutilisons un algorithme d'exploration et des scores existant pour chaque implémentation, et proposons un second score unique *S'* pondérant *S*. *S'* prend en compte le modèle de diagnostic *MD* en cours d'apprentissage, ainsi que l'ontologie O et la mise en correspondance de l'ontologie vers les traces Map. *Search()* et S traitent le problème de l'apprentissage du meilleur modèle (ayant le meilleur score) dans les traces, c'est donc un problème d'apprentissage automatique. Le score *S'* apporte une contrainte sur le score *S* afin que le modèle statistique appris satisfasse le modèle de diagnostic *MD*, donc soit une instance de la technique générique de diagnostic considérée. *MD* étant défini comme un métamodèle formulé comme un graphe et un ensemble de contraintes sur le graphe (cf. plus haut), un modèle satisfaisant *MD* se définit comme un graphe, instance du métamodèle, n'enfreignant aucune des contraintes. L'algorithme résout le problème de l'instanciation de *MD* étant donné *T*, *O* et *Map*. Par la suite, nous considérons T comme une table (une collection de variables), sans contrainte sur son format.

L'algorithme fonctionne en trois étapes :

– recherche du sous-ensemble de variables (dans les traces ou classes de l'ontologie) maximisant la probabilité de satisfaire *MD*. Par exemple, déterminer quelles variables dans les traces représentent les Knowledge Component pour le Knowledge Tracing. Cette étape est réalisé par l'algorithme *search()* et le score. Une solution candidate générée par *search()* est donc une association possible entre *MD* et les variables des traces ;

– extraction du domaine (des valeurs possibles) de ces variables, par exemple l'ensemble des Knowledge Components. Dans l'ontologie, ce sont les classes du niveau détaillé ;

– apprentissage automatique des paramètres si requis, par exemple les distributions de probabilité d'un réseau bayésien ou les coefficients d'un modèle linéaire.

On définit une matrice *R* dont les lignes sont les variables de *MD* et les colonnes les variables des traces. *R(md,t)* est la probabilité que la variable *t* de *T* représente la variable *md* de *MD*. La matrice est initialisée à 1 si *t* a été liée par le concepteur à une classe de l'ontologie *O* (via la mise en correspondance), 0,5 sinon. Ces probabilités servent à calculer le score *a*S*S'* (a et S' *biaisant* le score S), puis sont mises à jour à chaque itération de l'algorithme d'exploration. Le calcul de *a* et *S'* est défini ci-dessous.

Soit $N=(\mu, \sigma 2)$ une gaussienne centrée en 1 ($\mu = 1$), $\sigma 2$ étant fixé de sorte que $N(0.5) = max(S)/2$, et $N(R(t, md))$ le score correspondant à la probabilité R(t, md), alors le score S' est le produit des $N(R(t, md))$ pour tout *t* de *T* et *md* de *MD* de la solution candidate.

Le coefficient *a* est un paramètre calculé de sorte que *min(S)*a*S'(0.8)=max(S)*S'(0.5)*. Cette contrainte permet de vérifier la propriété suivante : une probabilité supérieure ou égale à 0,8 dans la matrice donnera toujours un meilleur score qu'une probabilité aléatoire (0.5), quel que soit *S*. Le meilleur score est celui maximisant *a*S*S'* et tel que la solution candidate satisfasse *MD*.

La matrice *R* est finalement mise à jour par inférence, en considérant : le score, la satisfaction du modèle de diagnostic *MD*, et la compatibilité avec l'ontologie *O*. La satisfaction de *MD* et *O* est réalisée par une fonction de comparaison de graphe.

L'algorithme d'exploration locale est relancé jusqu'à convergence de *R*. La technique de diagnostic instanciée retournée par l'algorithme est la solution candidate satisfaisant *MD* avec le plus haut score.

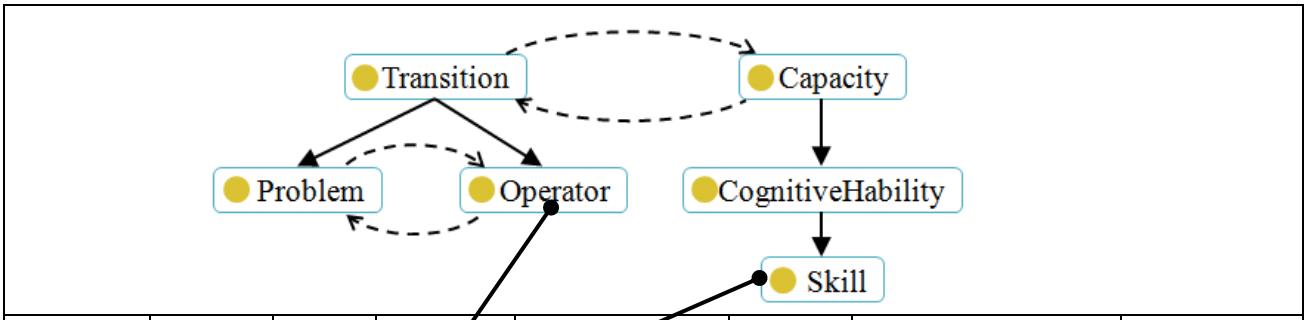
La seconde étape extrait des traces et de l'ontologie toutes les valeurs possibles pour des variables de la technique de diagnostic. La troisième étape utilise des algorithmes de la littérature pour apprendre les paramètres des techniques si requis.

Concernant l'implantation dans notre plateforme, l'algorithme de recherche locale utilisé est fondé sur Hill-Climbing [LANGLEY et al. 87], et le score *S* est DAG-Search [NEAPOLITAN 04] pour les modèles probabilistes graphiques, un algorithme de couverture glouton pour les règles SI/SINON [CORMEN et al. 01]. Pour les paramètres, l'algorithme EM [DEMPSTER et al. 77] est utilisé pour toutes les techniques actuellement.

3.3. Exemple d'application de l'algorithme

Nous montrons un exemple d'exécution de l'algorithme sur le domaine de la chirurgie orthopédique via les traces de TELEOS [MINH CHIEU et al. 10]. Cet exemple est donné pour l'instanciation de la technique Knowledge Tracing, et l'ontologie de haut niveau de la Figure 1 est utilisée. Un exemple des traces est donné Figure 2, où Vertèbre représente l'exercice, Action une interaction avec l'EIAH, Représentation désigne les types d'expression des problèmes et des actions, et Variables de situation est le label du résultat du diagnostic comportemental. La mise en correspondance est réalisée entre ces traces et l'ontologie (Figure 2).

La matrice *R* est initialisée comme montré Tableau 1 (gauche), où les lignes représentent *MD* et les colonnes les variables des traces *T*. *R(KC, Connaissance)* est initialisé à 1 car héritant de la classe Capacity dans l'ontologie. Or, le modèle de diagnostic du Knowledge Tracing ne comporte qu'une seule variable de type connaissance, *KC*. En revanche pour les transitions (observées), il n'est pas possible à ce stade d'identifier dans les traces les variables *Problem* et *Step*. L'algorithme génère les premiers candidats en partant d'un modèle à une variable ; la première étape est montrée Figure 3, avec la solution candidate à gauche et le score à droite. La solution est composée d'une seule variable *Vertèbre* assimilée à la première variable du modèle de diagnostic, Problème. Le modèle *MD* exprimant que le problème est composé d'une ou plusieurs étapes, les candidats explorés par l'algorithme de recherche locale sont {Fils(Vertèbre) = Action, Fils(Vertèbre) = Représentation, Fils(Vertèbre) = Connaissance, Fils(Vertèbre) = Variable de situation} (illustré Figure 3 partie gauche). Les scores *S* et *S'* sont ensuite calculés. La Figure 3 partie droite montre la gaussienne utilisée pour le calcul du score : l'axe des abscisses représente les probabilités issues de la matrice *R* et les ordonnées la valeur du score *S'*. Le but étant d'identifier les étapes du problème dans les traces, la ligne Step de la matrice *R* est prise en compte pour le score : toutes les valeurs valent 0,5 sauf *R(Step, Connaissance)* qui vaut 0. La matrice est ensuite mise à jour (Tableau 1, droite). La probabilité *R(Step, Action)* augmente le plus car le score est le plus élevé pour cette solution.



Apprenant	Vertèbre	Date	Action	Connaissance	Représentation	Variable de situation	Diagnostic comportement
Baptiste	T12	12/05	Radio face	Vertèbre centrée	Radio	Dist_RF_centrageVert	Correct
Baptiste	T12	12/05	Radio face	Symétrie pédicules	Radio	Dist_RF_symetriePedic	Correct

Figure 2. *Exemple de traces de TELEOS et la mise en correspondance entre les variables des traces et l'ontologie.*

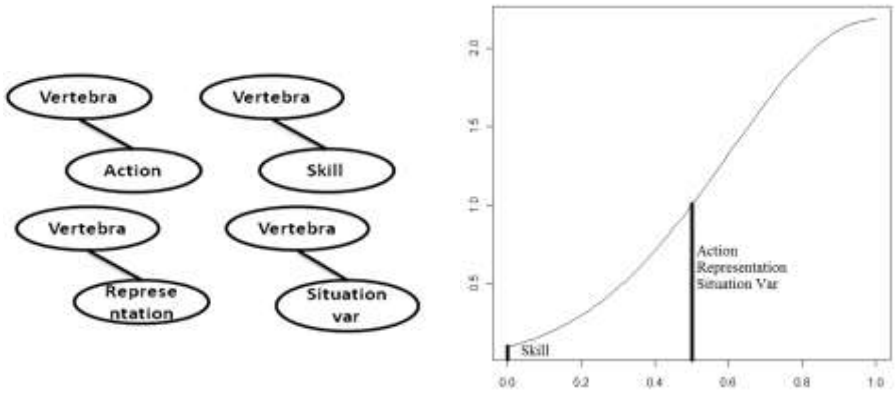


Figure 3. *Illustration d'une itération de l'algorithme, avec les solutions candidates à gauche et le calcul du score S' à droite, au moyen de la matrice R donnée Tableau 3.*

Après convergence, le résultat de la première étape est donné Figure 4 partie gauche, qui fournit une mise en correspondance entre le modèle de la technique générique de diagnostic (Knowledge Tracing ici) et les traces. L'algorithme s'exécute de la même manière pour toutes les techniques incluses dans la plateforme, grâce aux modèles de diagnostic. La deuxième étape de l'algorithme parcourt les traces pour extraire les valeurs possibles de chaque variable du modèle de diagnostic. L'exemple est fondé sur les traces de la Figure 2.

	Vertèbre	Action	Représentation	Connaissance	Variable de situation
Problem	0.5	0.5	0.5	0	0.5
Step	0.5	0.5	0.5	0	0.5
KC	0	0	0	1	0

	Vertèbre	Action	Représentation	Connaissance	Variable de situation
Problem	0.59	0.54	0.5	0	0.5
Step	0.54	0.62	0.5	0	0.5
KC	0	0	0	1	0

Tableau 1. *Gauche : initialisation de la matrice R. Droite : matrice R mise à jour par l'algorithme après une itération.*

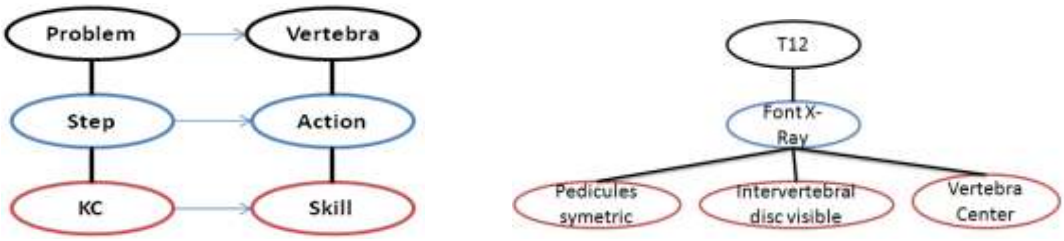


Figure 4. *Gauche : structure de la technique de diagnostic apprise, associant les variables des traces aux variables correspondantes du modèle de diagnostic MD. Droite : détail de la technique de diagnostic instanciée au domaine après extraction des valeurs possibles de chaque variable (les couleurs correspondent avec la partie gauche).*

4. Expérimentations

4.1. Présentation des domaines et des traces

Nous avons évalué notre proposition dans deux domaines au moyen de traces d'apprenants précédemment collectées lors d'expérimentations. Le premier domaine est la chirurgie orthopédique avec des traces collectées par TELEOS [MINH CHIEU et al. 10]. Les expérimentations comptaient six apprenants et un chirurgien expert, soit 2695 traces obtenues pour 37 connaissances distinctes, où une trace est une interaction avec l'EIAH. Ces traces sont coûteuses à obtenir car les expérimentations ont eu lieu à l'hôpital et le domaine est mal défini. Le second domaine est la lecture de l'anglais maternel pour les enfants, avec le Reading Tutor [MOSTOW & AIST 01]. 50 enfants ont utilisé l'EIAH durant une année scolaire, soit 240 204 traces obtenues et 58 connaissances distinctes, où une trace est un mot lu par un enfant.

Nous avons testé deux ontologies par domaine, une de haut niveau et une détaillée. L'ontologie détaillée comprend les mêmes classes que l'ontologie de haut niveau, mais la moitié des classes détaillées seulement a été modélisée.

4.2. Résultats

Nous avons évalué les techniques de diagnostic instanciées par des mesures statistiques calculées en validation croisée, 75 % des traces ayant été utilisées pour l'apprentissage des techniques et 25 % pour leur évaluation (par répartition aléatoire des apprenants). Les mesures utilisées sont la précision de la prédiction, i.e. le taux de bonne prédiction au temps t de la réponse de l'apprenant, correcte ou incorrecte, au temps t+1, la racine de l'erreur moyenne au carré (RMSE), qui mesure une approximation du taux d'erreur, et le critère AIC, qui mesure la vraisemblance (likelihood) du modèle pénalisée par sa complexité, définie ici comme le produit du nombre d'apprenants par le nombre de connaissances par le nombre de paramètres par connaissance requis par la technique. Le Tableau 5 montre les résultats pour TELEOS et le Tableau 6 pour le Reading Tutor. Pour la précision de la prédiction, l'intervalle de confiance à 95 % est précisé entre parenthèses.

La classe majoritaire (la valeur la plus fréquente d'une variable) est de 54 % pour TELEOS et 74 % pour le Reading Tutor. Toutes les précisions de la prédiction sont donc supérieures ou égales à la classe majoritaire, hormis le Constraint-Based pour le Reading Tutor. Les intervalles de confiance montrent que les différences entre les précisions des techniques ne sont pas significatives sur ces deux jeux de traces ; cela suggère également que l'algorithme d'apprentissage donne des résultats proches pour toutes les techniques. Les RMSE sont toutes cohérentes avec la précision de la prédiction. Ces résultats sur la prédiction montrent donc que la meilleure technique pour TELEOS est le Control-Based en

utilisant l'ontologie détaillée, et les deux modèles cognitivistes (Knoweldge Tracing et Additive Factor Model) sont les meilleurs pour le Reading Tutor, avec des différences statistiquement non significatives. Le test AIC pénalise le Control-based, en raison du grand nombre de paramètres modélisés dans le réseau bayésien dynamique.

Pour les deux niveaux d'ontologie, des résultats quasiment similaires sont obtenus avec le Reading Tutor, car la même technique a été apprise, modulo seulement l'approximation due à l'apprentissage des paramètres. Pour TELEOS, la précision de la prédiction et le test RMSE sont meilleurs pour les techniques instanciées à partir de l'ontologie détaillée, mais de façon statistiquement significative seulement pour le Control-based d'après les intervalles de confiance de la précision. Les intervalles de confiance de la précision du Contraint-based et du Knowledge Tracing sont également proches de la signifiance (significatifs mais avec un intervalle de confiance à 80 %). Sur les deux domaines, l'ontologie détaillée n'a donc eu un impact significatif que dans un cas. Sans utiliser aucune ontologie, nous avons mesuré à titre de test une baisse de la précision de la prédiction de 16 % en moyenne et un RMSE plus élevé de 0,14 en moyenne sur toutes les techniques.

Pour le Reading Tutor, des techniques de diagnostic (de type Knowledge Tracing) développées par des experts en conception de diagnostic ont été évaluées dans de précédents travaux et donnaient des précisions de 72 % à 87 % [XU & MOSTOW 12]. Les résultats obtenus par notre plateforme sont donc compris dans cet intervalle. Toutefois, une telle comparaison est informative et non scientifique (les données traitées n'étant pas les mêmes et la technique de diagnostic obtenant 87 % de précision non testée dans notre expérimentation).

Technique de diagnostic	**Ontologie de haut niveau**			**Ontologie détaillée**		
	Précision	**RMSE**	**AIC**	**Précision**	**RMSE**	**AIC**
Knowledge Tracing	66% (+/- 2,8 %)	0.32	7 914	71% (+/- 2,7 %)	0.32	7 829
Additive Factor Model	69% (+/- 2,9 %)	0.34	7 899	70% (+/- 2,8 %)	0.32	7 897
Constraint-based	68% (+/- 3,7 %)	0.33	7 472	73% (+/- 3,6 %)	0.31	7 305
Control-based	67% (+/- 3,5 %)	0.36	10 109	75% (+/- 3,3 %)	0.30	10 194

Tableau 5. *Résultat pour le domaine de la chirurgie orthopédique.*

Technique de diagnostic	**Ontologie de haut niveau**			**Ontologie détaillée**		
	Précision	**RMSE**	**AIC**	**Précision**	**RMSE**	**AIC**
Knowledge Tracing	78% (+/- 3,2 %)	0.577	226 723	78% (+/- 3,2 %)	0.572	226 126
Additive Factor Model	78% (+/- 3,9 %)	0.586	215 894	78% (+/- 3,9 %)	0.584	214 447
Constraint-based	72% (+/- 4,1 %)	0.62	214 351	72% (+/- 4,1 %)	0.61	215 003
Control-based	74% (+/- 3,5 %)	0.594	244 879	74% (+/- 3,5 %)	0.59	244 655

Tableau 6. *Résultat pour le domaine de la lecture.*

5. Conclusion

Nous proposons dans cet article une méthodologie générique d'assistance pour le développement d'un ensemble de techniques de diagnostic, reposant sur un algorithme d'apprentissage semi-automatique, guidé par une ontologie. Les résultats montrent des

précisions de prédiction supérieures à la classe majoritaire dans deux domaines différents, la chirurgie orthopédique et la lecture.

Par rapport aux outils auteur existants, notre plateforme adopte une stratégie modulaire (centrée sur un composant des EIAH, le diagnostic des connaissances). Il n'existe également pas à notre connaissance de méthode permettant le développement de plusieurs techniques de diagnostic. En contrepartie, l'approche est semi-automatique pour garantir à la fois l'interprétabilité des techniques de diagnostic et la prise en compte de techniques de diagnostic variées. Le concepteur doit donc fournir un travail de modélisation que n'exigent pas les méthodes d'apprentissage automatique. Il se pose également la question du biais apporté par le concepteur lors du design de l'ontologie. Notre plateforme s'inscrit plutôt dans l'assistance à la conception, permettant aux concepteurs de tester et évaluer statistiquement et par comparaison diverses approches de modélisation de l'ontologie et divers jeux de traces. Les performances d'une technique de diagnostic (en termes de précision) étant variables selon les domaines [GONG et al. 11], une approche permettant la comparaison entre plusieurs techniques est d'autant plus pertinente. Enfin, notre plateforme requiert la collecte de traces au préalable, via des expérimentations, un micromonde, un simulateur, un EIAH existant, une base de partage de traces… Pour le moment, nous considérons des traces enrichies, où il est possible d'évaluer les productions ou le comportement des apprenants (par exemple, une réponse est-elle correcte ou incorrecte), sans raisonnement sur les connaissances mobilisées ou non mobilisées par l'apprenant.

Les perspectives de nos travaux sont d'améliorer l'interface pour l'heure rudimentaire de notre plateforme, afin d'expérimenter son utilisabilité, de tester la méthodologie sur d'autres domaines, notamment avec des jeux de traces incomplets ou sur des domaines où les connaissances sont mal formalisées, et d'assister le concepteur dans le choix du niveau de l'ontologie (haut niveau ou détaillé) le plus pertinent pour ses traces.

Remerciements

Nous remercions la région Rhônes-Alpes pour la bourse de thèse qui soutient ce travail, ainsi que les projets LISTEN et TELEOS pour les traces mises à notre disposition.

Bibliographie

[ALEVEN et al. 06] Aleven, V., McLaren, B., Sewall, J., and Koedinger, K., « The cognitive tutor authoring tools (ctat): Preliminary evaluation of efficiency gains », *International Conference Intelligent Tutoring Systems*, 2006, p. 61–70.

[BECK 07] Beck, J. E., « Difficulties in inferring student knowledge from observations (and why you should care) », *Educational Data Mining: Supplementary Proceedings of the 13th International Conference of Artificial Intelligence in Education*, 2007, p. 21–30.

[BECK et al. 00] Beck, J. E., Woolf, B. P., and Beal, C. R., « ADVISOR: a machine-learning architecture for intelligent tutor construction », *AAAI Conference on Artificial Intelligence*, 2000, p. 552–557.

[BLESSING & GILBERT 08] Blessing, S., and Gilbert, S., « Evaluating an authoring tool for model-tracing intelligent tutoring systems », *International Conference on Intelligent Tutoring Systems*, 2008, p. 204–215.

[CEN et al. 08] Cen, H., Koedinger, K., and Junker, B., « Comparing two IRT models for conjunctive skills », *International Conference on Intelligent Tutoring Systems*, 2008, p. 796–798.

[CORBETT & ANDERSON 95] Corbett, A. T., and Anderson, J. R., « Knowledge tracing: Modeling the acquisition of procedural knowledge », *User Modelling and User-Adapted Interaction*, vol. 4, 1995, p. 253–278.

[CORMEN et al. 01] Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C., *Introduction To Algorithms*, MIT PRess, 2001.

[DEMPSTER et al. 77] Dempster, A. P., Laird, N. M., and Rubin, D. B., « Maximum Likelihood from Incomplete Data via the EM Algorithm », *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 39, n° 1, 1977, p. 1–38.

[DESMARAIS et al. 06] Desmarais, M. C., Meshkinfam, P., and Gagnon, M., « Learned student models with item to item knowledge structures », *User Modeling and User-Adapted Interaction*, vol. 16, n° 5, 2006, p. 403–434.

[GONG et al. 11] Gong, Y., Beck, J. E., and Heffernan, N. T., « How to Construct More Accurate Student Models: Comparing and Optimizing Knowledge Tracing and Performance Factor Analysis », *International Journal of Artificial Intelligence in Education*, vol. 21, n° 1, 2011, p. 27–46.

[GONZALES-BRENES & MOSTOW 12] Gonzáles-Brenes, J., and Mostow, J., « Dynamic Cognitive Tracing: Towards Unified Discovery of Student and Cognitive Models », *International Conference on Educational Data Mining*, 2012, p. 49-56.

[LALLE et al. 12] Lallé. S., Luengo, V., and Guin, N., « Méthodologie d'assistance pour la comparaison de techniques de diagnostic des connaissances », *Conférence Technologies de l'Information et de la Communication pour l'Enseignement*, 2012, p. 11-19.

[LANGLEY et al. 87] Langley, P., Gennari, J. H., and Iba, W., *Hill-Climbing Theories of Learning*, Defense Technical Information Center, 1987.

[MATSUDA et al. 05] Matsuda, N., Cohen, W. W., and Koedinger, K. R., « Building cognitive tutors with programming by demonstration », *International Conference on Inductive Logic Programming*, 2005, p. 41-46.

[MINH CHIEU et al. 10] Minh Chieu, V., Luengo, V., Vadcard, L., and Tonetti, J., « Student modeling in complex domains: Exploiting symbiosis between temporal Bayesian networks and fine-grained didactical analysis », *International Journal of Artificial Intelligence in Education*, vol. 20, n° 3, 2010, p. 269–301.

[MITROVIC et al. 06] Mitrovic, A., Suraweera, P., Martin, B., Zakharov, K., Milik, N., and Holland, J., « Authoring constraint-based tutors in ASPIRE », *International Conference on Intelligent Tutoring Systems*, 2006, p. 41–50.

[MOSTOW & AIST 01] Mostow, J., and Aist, G., « Evaluating tutors that listen: An overview of Project LISTEN », *Smart Machines in Education: The coming revolution in educational technology*, AAAI Press, 2001, p. 169–234.

[MURRAY 03] Murray, T., « Eon: Authoring tools for content, instructional strategy, student model, and interface design », *Authoring Tools for Advanced Technology Learning Environments Toward cost-effective adaptive, interactive, and intelligent educational software*, Springer, 2003, p. 309–340.

[NEAPOLITAN 04] Neapolitan, R. E., *Learning Bayesian networks*, Pearson Prentice Hall, 2004.

[NKAMBOU et al. 97] Nkambou, R., Gauthier, G., and Frasson, C., « Un modèle de représentation des connaissances relatives au contenu dans un système tutoriel intelligent », *Sciences et techniques éducatives*, vol. 4, n° 3, 1997, p. 299–330.

[OHLSSON 94] Ohlsson, S., « Constraint-based student modeling », *NATO ASI Series F Computer and Systems Sciences*, vol. 125, 1994, p. 167–189.

[SISON & SHIMURA 98] Sison, R., and Shimura, M., « Student modeling and machine learning », *International Journal of Artificial Intelligence in Education*, vol. 9, n° 1-2, 1998, p. 128–158.

[XU & MOSTOW 12] Xu, Y., and Mostow, J., « Comparison of methods to trace multiple subskills: Is LR-DBN best? », *International Conference on Educational Data Mining*, 2012 p. 41-48.