MODELAGEM DE ESTUDANTES PARA UM SISTEMA TUTOR INTELIGENTE BASEADO EM RESTRIÇÕES

Students modeling for an intelligent tutor system based on contraints

Carlos Alberto Botelho

Unisantos - Universidade Católica de Santos–Santos-SP
Universidade IMES – São Caetano do Sul-SP

RESUMO

Este artigo aborda a técnica de modelagem baseada em restrições (Constraint Based Modelling - CBM) para a modelagem do estudante e do domínio do conhecimento em sistemas tutores inteligentes. Na primeira parte, é feita uma revisão histórica do desenvolvimento de sistemas tutores inteligentes e uma descrição da arquitetura genérica de sistemas tutores desse tipo. Na segunda parte, é feita uma descrição das características básicas da técnica de modelagem baseada em restrições. Na terceira parte, são descritos alguns sistemas que utilizam essa técnica e, finalmente, são relatadas algumas conclusões a respeito das vantagens do uso dessa técnica.

Palavras-chave: CBM, sistemas tutores inteligentes, técnicas de modelagem.

ABSTRACT

This paper describes the modeling technique based on constraints (Constraint Based Modeling - CBM) for the knowledge domain in intelligent tutor systems and the student modeling. In the first part, a historic revision of intelligent tutoring systems development and a description of generic architecture of tutoring this kind of systems are done. In the second part, the basic characteristics of constraint based modeling techniques are described. In the third part, some systems which use this technique are described, and, finally, conclusions regarding the advantages of using this technique are reported.

Keywords: CBM, intelligent tutoring systems, modeling techniques.

1. INTRODUÇÃO

O processo de ensino utilizando os computadores como ferramenta, teve inicio no final da década de 1950, quando o psicólogo americano B. F. Skinner, baseado na teoria comportamentalista, propôs um método de ensino, que chamou de máquina de ensinar. Nesse método, o material a ser ensinado se divide em módulos seqüenciais, cada módulo terminando com questões a serem respondidas pelo estudante. Acertando as respostas, o estudante é autorizado a passar para o módulo seguinte.

Com base nesse método, foram criados os primeiros programas de computador voltados para o ensino: o Computer Aided Instruction, conhecidos por CAI.

Nos sistemas de ensino suportados pelo CAI, o aluno segue uma seqüência finita e predeterminada de passos. A cada passo, o conhecimento adquirido pelo estudante é testado e, se corresponde ao correto, passa para o próximo passo; caso contrário, o conhecimento ainda não assimilado é novamente exposto ao estudante e o teste é refeito. Essa operação se repete até que o estudante responda corretamente ao teste.

Esses sistemas têm como característica, a transmissão do conhecimento feita sempre da mesma forma, independente do perfil do estudante. São hoje conhecidos como "viradores de páginas".

Durante a década de 1960, foram feitos muitos investimentos na produção de sistemas CAI, pelo governo norte-americano, por empresas da área de Tecnologia da Informação, tais como IBM, Digital, RCA e por universidades. O sistema CAI mais bem-sucedido até hoje é o PLATO (Programmed Logic for Automatic Teaching Operations), desenvolvido em parceria entre a Computer Education Research Laboratory, da Universidade de Illionois, a Control Data Corporation e a National Science Foundation, para computadores de grande porte.

Em 1982, Sleeman e Brown (SLEEMAN e BROWN, 1982 *apud* FONSECA JR., 2004) revisaram o estado da arte nos sistemas CAI e, com base na psicologia cognitivista, na abordagem construtivista do ensino e nas técnicas de Inteligência Artificial, criaram uma nova maneira de usar o computador no ensino, que denominaram Sistemas Tutores Inteligentes (STI).

Ao contrário dos sistemas CAI tradicionais, que atuam sempre da mesma maneira para todo estudante, os STI atuam conforme as características de cada estudante.

O quadro abaixo, mostra com clareza as principais diferenças entre os sistemas CAI e os sistemas STI.

Aspecto	CAI	STI
Origem	Educação	Ciência da Computação
Bases Teóricas	Skinner (behaviorista)	Psicologia Cognitivista
Estruturação e Funções	Uma única estrutura algoritmicamente predefinida, onde o aluno não influi no seqüenciamento	Estrutura subdividida em módulos, cujo seqüenciamento se dá em função das respostas do aluno
Estruturação do Conhecimento	Algorítmica	Heurística
Modelagem do Aluno	Avaliam a última resposta	Tentam avaliar todas as respostas do aluno durante a interação
Modalidades	Tutorial, exercício e prática, simulação e jogos educativos	Socrático, ambiente interativo, diálogo bidirecional e guia

Fonte: Giraffa (1997, p. 19).

Segundo Jonassen (1993 *apud* FONSECA JR., 2004), um STI deve atender a três requisitos:

1. O conteúdo deve ser codificado de modo que o sistema possa rapidamente acessar as informações, fazer inferências ou resolver problemas.

2. O sistema deve ser capaz de avaliar a aquisição deste conhecimento pelo aluno.

3. As estratégias tutoriais devem ser projetadas para reduzir a discrepância entre o conhecimento do especialista e o conhecimento do aluno.

Para Urretavizcaya (2001), as características mais importantes de um STI são:

1. Conhecimento do domínio restrito e claramente articulado.

2. Conhecimento sobre o aluno que permita dirigir e adaptar o ensino.

3. Seqüência de ensino não-predeterminada.

4. Produção de diagnósticos mais detalhados e adaptados ao aluno.

5. Melhora na interação entre tutor e aluno, permitindo que o aluno faça perguntas ao tutor.

A arquitetura tradicional de um STI, demonstrada na **Figura 1**, contém quatro componentes básicos (WENGER, 1987 *apud* URRETAVIZCAVA, 2001).

1. Módulo do domínio, que contém a matéria a ser ensinada.

2. Módulo do aluno, que contém as características do aluno.

3. Módulo do tutor, que contém as estratégias e táticas para a aplicação no processo de ensino.

4. Módulo da interface, que contém a forma de comunicação entre o tutor e o aluno.

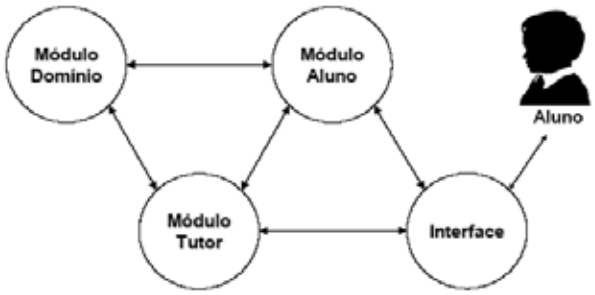

Figura 1: Os componentes de um Sistema Tutor Inteligente [Goulart e Giraffa, 2001].

O módulo do domínio do conhecimento deve conter todo o conhecimento a ser ensinado, mas a sua representação completa pode ser muito dispendiosa e o acesso aos dados muito lento.

O módulo do estudante deve conter informações sobre o nível de conhecimento, objetivos de aprendizagem, motivação, crenças e características pessoais do estudante, mas a sua representação completa é um problema classificado como intratável por Self (1990) devido à quantidade de informações necessárias.

Self demonstra que, dado um problema, se quizermos mapear todas as soluções corretas e erradas desse problema para um estudante, a quantidade de alternativas erradas pode chegar a um valor exponencial, o que caracteriza um problema intratável.

Segundo Mitrovic e Ohlsson (1999), embora seja impossível a modelagem completa do estudante, existem algumas técnicas que permitem algum tipo satisfatório de modelagem, embora incompleto. Entre essas estão "model tracing", na qual é feito um diagnóstico a cada passo, na construção da solução do problema; redes Bayesianas, na qual é estimada a probabilidade para um conjunto predefinido de elementos do conhecimento; modelagem através

de um conjunto "fuzzy" e "modelagem baseada em restrições."

2. MODELAGEM BASEADA E RESTRIÇÕES

Segundo Ohlsson (1996), mesmo conhecendo a maneira correta de realizar uma tarefa, cometemos erros na sua execução. No entanto, a repetição sucessiva da mesma tarefa corrigindo os erros cometidos, leva à eliminação progressiva dos erros. Essa técnica é denominada "aprendizagem através dos erros".

Ohlsson (1994 *apud* MITROVIC *et al*., 2003), baseado na teoria da aprendizagem através dos erros, e em que uma solução só pode estar correta se não violar qualquer um dos princípios do domínio do conhecimento, propôs a modelagem do domínio do conhecimento e do domínio do estudante, baseada em restrições (Constraint-Based Modelling).

Segundo Mitrovic (2003), a técnica CBM procura ter a representação do estado do processo do conhecimento em que se encontra o estudante num dado momento, não se importando com o que foi feito anteriormente. Enquanto não violar qualquer regra contida no domínio do conhecimento, o estudante pode prosseguir no desenvolvimento da aprendizagem.

A representação do conhecimento é feita de maneira abstrata, através de um par ordenado chamado "restrição de estado". O par é constituído de uma cláusula de relevância e uma cláusula de satisfação e tem o formato de uma regra na forma: se a condição de relevância for encontrada, então a condição de satisfação deve ser atendida.

Por ser feita de maneira abstrata, a técnica de modelagem baseada em restrições é independente do domínio do conhecimento e da técnica pedagógica, uma vez que o domínio do conhecimento é constituído de um conjunto de pares, sendo cada par associado com uma ação instrucional a ser aplicada, qualquer que seja o domínio a ser ensinado e a técnica pedagógica a ser aplicada.

O domínio do conhecimento é constituído de um conjunto de restrições que representam os princípios básicos sobre os quais este se sustenta **(Figura 2)**.

C_{r1}: a abertura de um dos ângulos da base de um triângulo isósceles é conhecida (θ_1) e o estudante calculou a abertura do outro ângulo da base (θ_2).

C_{s1}: *a abertura do ângulo* θ_2 *deve ser igual à abertura do ângulo* θ_1.

C_{r2}: a abertura de um dos ângulos da base de um triângulo isósceles é conhecida (θ_1) e o estudante calculou a abertura do outro ângulo da base (θ_2), igual à de θ_1.

C_{s2}: θ_2 *é um ângulo da base.*

C_{r3}: as aberturas de dois ângulos de um triângulo (θ_1 e θ_2) são conhecidas e o estudante calculou a abertura do terceiro ângulo θ_3.

C_{s3}: $\theta_3 = 180 - \theta_1 - \theta_2$.

Figura 2: Três restrições que verificam se a abertura de um ângulo está correta (Mitrovic *et al*., 2003).

O processo de verificação de uma restrição é extremamente simples: basta, num determinado estado da solução de um problema, verificar quais restrições são relevantes nesse estado e, em seguida, verificar se as condições de satisfação para as restrições relevantes estão presentes no estado.

O modelo do estudante é o conjunto de restrições violadas, ou seja, aquelas que não estão satisfeitas num determinado estado da solução do problema. Essas restrições representam os conceitos do domínio que o estudante ainda não conhece perfeitamente. Esse tipo de modelo do estudante é conhecido como "overlay" pois contém apenas parte do modelo do domínio. A presença de uma restrição indica, portanto, que o estudante ainda não entendeu o item do domínio do conhecimento representado por ela.

Ohlsson (1994) afirma que não se pode de-

duzir que o fato do estudante não ter violado uma restrição indica que ele entendeu o conhecimento representado por ela, uma vez que o sucesso pode ser acidental. Por outro lado, a violação de uma restrição indica com segurança que o estudante não entendeu o conhecimento representado por ela.

A construção de um conjunto completo de restrições é uma tarefa que demanda tempo e é difícil de ser realizada. Pode-se, no entanto, construir o conjunto de restrições gradativamente, com a adição de uma restrição quando esta for necessária.

Segundo Martin (1999), a construção de um conjunto de restrições para modelar um domínio do conhecimento esbarra em dois problemas: a ausência ou duplicação de restrições e a especificidade de uma restrição.

No primeiro caso, na inclusão de novos tipos de problemas, é muito difícil avaliar quais restrições já existem e quais necessitam ser incluídas, o que pode levar à construção de um conjunto incompleto de restrições.

No segundo caso, a especificidade de uma restrição pode levar à avaliação como incorreta a resposta a um novo problema, mesmo que esta resposta esteja correta segundo conceitos ainda não representados no conjunto de restrições.

A seguir, são descritas experiências de tutores implementados com base em CBM.

3. ALGUNS SISTEMAS QUE USAM CBM

3.1. SQL-Tutor

O SQL-Tutor foi desenvolvido por Mitrovic (1997) para ensinar a linguagem SQL de acesso a bases de dados, para estudantes universitários, utilizando a técnica da modelagem baseada em restrições, aplicada na representação do domínio do conhecimento e no modelo do estudante.

O sistema é composto de uma interface, uma base de conhecimento, um modelador do estudante e um módulo pedagógico que determina o "timing" e o conteúdo das ações pedagógicas (**Figura 3**).

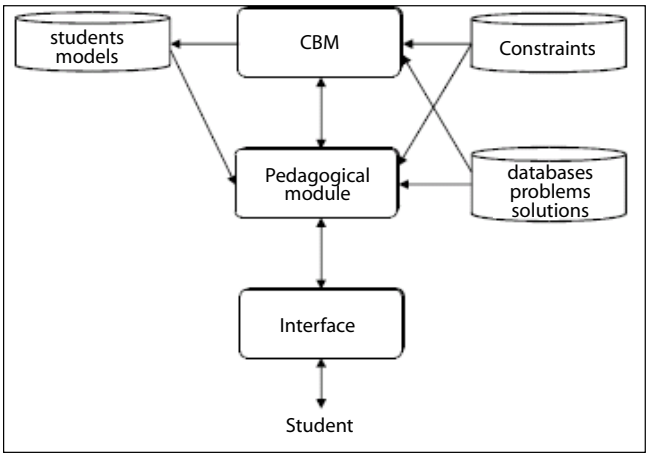

Figura 3: Arquitetura de um SQL-Tutor (Mitrovic; Ohlsson, 1999).

3.1.1. Interface

O módulo da interface solicita a identificação do estudante, empregada para localizar o modelo desse estudante. Caso seja a primeira entrada do estudante no sistema, é criado um modelo para esse estudante.

A tela principal do SQL-Tutor (**Figura 4**) é composta de três partes que são visíveis durante todo o tempo que durar a interação do estudante com o sistema. Na parte superior, é apresentado o texto do problema corrente; na parte do meio, são apresentadas as cláusulas do comando SELECT e, na parte inferior, o esquema do banco de dados escolhido.

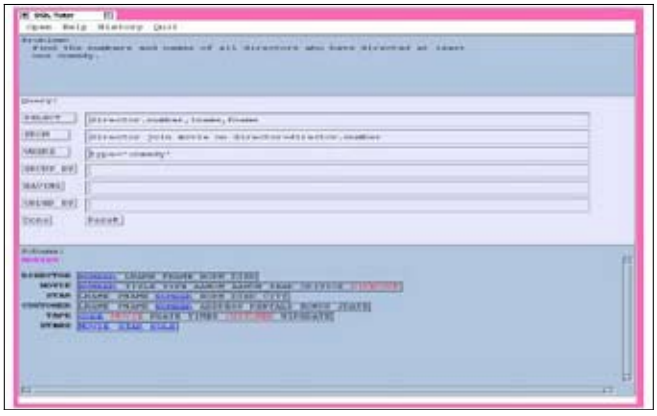

Figura 4: Interface do SQL-Tutor (Mitrovic, 1997).

A análise da solução do problema é feita somente quando for acionada pelo estudante. Nesse momento, o módulo pedagógico envia a solução proposta pelo estudante para o modelador do estudante que analisa a solução, identifica erros e atualiza o modelo do estudante. Caso haja erros, o módulo pedagógico apresenta a mensagem de "feedback" apropriada.

3.1.2. Base do conhecimento

A base do conhecimento é um conjunto desordenado de restrições. Essas restrições podem ser do tipo sintático, que se referem ao texto da solução do estudante ou do tipo semântico, que se referem à relação entre a solução do estudante e a solução ideal.

As restrições são declaradas como classes (**Figura 5**) na linguagem LISP. Cada restrição é identificada por um número (number), que é único, o padrão de relevância (rel-pattern), o padrão de satisfação (sat-pattern), uma lista de problemas (problems), para os quais a restrição é relevante, uma descrição da restrição (descr) e uma mensagem de "feedback", para o caso de erro e, por fim, a cláusula (clause) do comando SELECT, para o qual a restrição se aplica e é usada para gerar o feedback de tipo de erro.

```
(defclass     constraint  ()
((number      :type       integer  :accessor  number       :initarg  number)
(rel-pattern  :type       cons     :accessor  rel-pattern  :initarg  rel-pattern)
(sat-pattern  :type       cons     :accessor  sat-pattern  :initarg  sat-pattern)
(problems     :type       cons     :accessor  problems     :initarg  problems)
(descr        :type       string   :accessor  descr        :initarg  descr)
(clause       :type       string   :accessor  clause       :initarg  clause)))
```

Figura 5: Declaração da classe restrição (Mitrovic, 1997).

```
(p 186
"Check the numerical constant you are using in the WHERE clause!"
(and (not (null (where ss)))
     (not (null (where is)))
     (bind-all '?n (find-names ss 'where) bindings)
     (attribute-in-db (find-schema (current-database *radni*)) '?n)
     (equalp (find-type '?n) 'numeric)
     (match '(?*d1 ?n ">" (?is ?c1 numericp) ?*d2) (where ss) bindings)
     (match '(?*d3 ?n ">=" (?is ?c2 numericp) ?*d4) (where is) bindings))
(equalp (string-to-number '?c2) (1+ (string-to-number '?c1)))
"WHERE")
```

Figura 6: Exemplo de uma restrição (Mitrovic, 1997).

O processamento de uma solução apresentada pelo estudante é feito em duas etapas (**Figura 7**). Na primeira etapa, a solução é verificada contra uma rede que contém as condições de relevância para o problema. Serão selecionadas apenas as condições que se apresentam como relevantes para a solução. Na segunda etapa, são selecionadas as condições de satisfação que correspondem às soluções selecionadas como relevantes.

Figura 7: Fluxograma do processamento de uma solução do estudante (Mitrovic, 1997).

3.1.3. Modelo do estudante

O modelo do estudante é representado segundo a estrutura de classe demonstrada na **figura 8**.

```
(defclass         student  ()
((name            :type    string  :accessor  student-name     :initarg   :name)
(solved-problems  :type    cons    :accessor  solved-problems  :initform  nil)
(knows            :type    cons    :accessor  knows            :initform  nil)
(stype            :type    atom    :accessor  stype            :initarg   stype)))
```

Figura 8: Declaração da classe estudante (Mitrovic , 1997).

Segundo descrição apresentada por Mitrovic (1997), a entrada knows é uma lista de estruturas que representam a informação histori-

ca sobre o uso das várias restrições. Para cada restrição usada pelo estudante, é feita uma lista na forma:

(constr-no relevant used correct)

Onde constr-no é o número da restrição (único), relevant é o indicador de quantas vezes essa restrição foi relevante para a solução ideal e used é o número de vezes que a restrição foi relevante para a solução do estudante e correct é o número de vezes que a restrição foi relevante e foi satisfeita pela solução do estudante.

Ainda segundo Mitrovic (1997), a informação contida na entrada knows é atualizada da seguinte maneira: o processamento da solução do estudante produz duas listas de restrições. A primeira contém as restrições satisfeitas e a segunda contém as restrições violadas. Se uma restrição é satisfeita, os indicadores used e correct são incrementados e se a mesma restrição é relevante para a solução ideal, o indicador relevant também é incrementado. No caso das restrições violadas, apenas o indicador used é incrementado.

3.1.4. Módulo pedagógico

O módulo pedagógico gera mensagens de "feedback" baseado no modelo do estudante e seleciona exercícios, baseado nas restrições violadas e nas restrições cuja condição de relevância ainda não foram utilizadas pelo estudante.

Ao contrário de outros ITS, que usam a técnica do "model tracing", que atuam a cada passo do estudante, o SQL-Tutor analisa a solução proposta pelo estudante e apresenta os correspondentes "feedbacks" apenas quando comandado pelo estudante.

3.2. CAPIT: um sistema inteligente para "capitalização" e pontuação.

O CAPIT (Mayo *et al.*, 2000) é um sistema para o ensino de regras de "capitalização" (uso de letras maiúsculas) e pontuação na língua inglesa.

O tutor CAPIT foi desenvolvido para aplicação no ensino de crianças na faixa de idade entre 10 e 11 anos, o que torna importante a facilidade e a motivação para o uso do sistema. Essa característica foi determinante para o desenho da interface com o usuário (**Figura 10**).

A arquitetura do CAPIT (**Figura 9**) é constituída das bases de dados de restrições, problemas e modelos dos estudantes, da interface do usuário, do modelador do estudante e do módulo pedagógico.

O processo de tutoria inicia com o comando do estudante para a seleção de um novo problema. Quando o estudante submete a solução de um problema ao tutor, esta é passada para o modelador do estudante, que verifica quais restrições são relevantes para essa solução e quais restrições são satisfeitas. As restrições violadas são passadas para o módulo pedagógico, onde é selecionada e apresentada uma mensagem de "feedback".

A base do conhecimento do CAPIT cobre o seguinte domínio, na língua inglesa:

- Utilização de letras maiúsculas em sentenças.
- Utilização de letras maiúsculas em nomes de pessoas e lugares.
- Uso do ponto final em sentenças.
- Contração do is e do not com apóstrofos.
- Anotação de posse com apóstrofos.

- Separação de sentenças com o uso de vírgulas.
- Separação de itens numa lista, com o uso de vírgulas.

- Uso de aspas na apresentação de fala.

- Pontuação correta do pronome possessivo its.

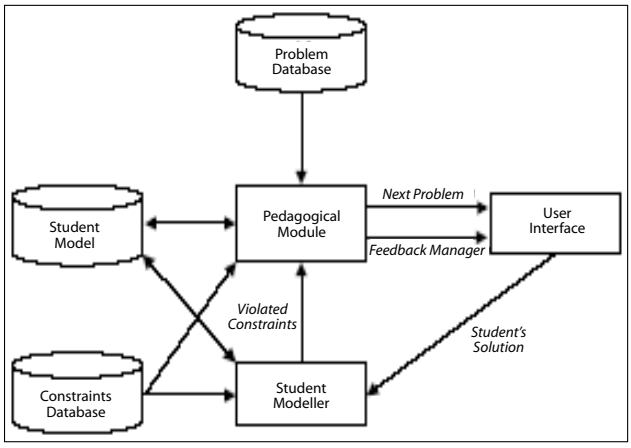

Figura 9: A estrutura do CAPIT (Mayo *et al.*, 2000).

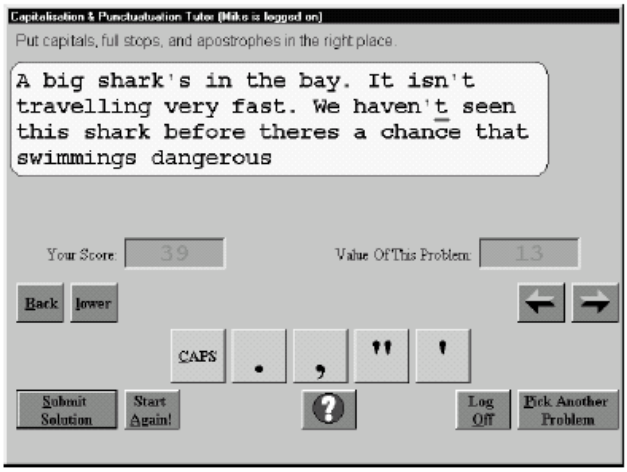

Figura 10: A tela de inteface do CAPIT (Mayo *et al.*, 2000).

As restrições são constituidas de três partes (**Figura 11**): a condição de relevância (Cr), a condição de satisfação (Cs) e uma mensagem que deverá ser mostrada, no caso da restrição ser violada. Além disso, a maioria das restrições é associada a uma página com uma explicação textual que é mostrada quando o estudante aciona o botão "Why?".

Os problemas são representados pela ligação entre cada palavra do texto com um conjunto de "tags", que se referem às condições de relevância (veja exemplo na **Figura 12**).

Cada condição de relevância é uma "tag" que deverá ser associada a cada palavra no texto a ser analisado. Uma palavra poderá ser associada a várias "tags". A condição de satisfação é uma expressão regular contra a qual a palavra relacionada com a "tag" que corresponde à condição de relevância será analisada.

Comparando os exemplos das **Figuras 11 e 12**, a "tag" DEFAULT, que consta na condição de relevância (a) da **Figura 11**, é relevante para o problema apresentado na **Figura 12**, pois está relacionada com as palavras teacher e your. Por outro lado, a "tag" NAME-OF-PERSON não é relevante para o problema, pois não está associada a qualquer palavra do texto apresentado.

	Cr	*Cs*
(a)	{DEFAULT, L-CASE}	^[a-z0-9%SYMBOLSET%]*$
	This word doesn't need any capital letters!	
(b)	{NAME-OF-PERSON}	^[%SYMBOLSET%]*[A-Z0-9]
	Each word in a person's name should start with a capital!	
(c)	{DIRECT-QUOTE-ENDING-SENTENCE}	[^%SYMBOLSET%]+((\.+"+)|"+|\.+)?$
	The full stop should be within the quotation marks!	
(d)	{ITS-POSSESSIVE-PRONOUN}	[^']s$
	No apostrophe is required in *its*!	

Figura 11: exemplos de restrições (Mayo *et al.*, 2000).

The	SENTENCE-START,NO-PUNC
teacher	DEFAULT
said,	WORD-PRECEDING-DIRECT-QUOTE, L-CASE,ONE-PUNC
"Open	DIRECT-QUOTE-START,ONE-PUNC
your	DEFAULT
books."	DIRECT-QUOTE-ENDING-SENTENCE, L-CASE,TWO-PUNC

Figura 12: Representação da frase: The teacher said, "Open your books." (Mayo *et al.*, 2000).

3.3. Kermit

Kermit (SURAWEERA; MITROVIC, 2002) é um sistema tutor inteligente para a modelagem de dados baseada no conceito entidade-relacionamentos. A sua estrutura é constituída de uma interface com o estudante, um modelador do estudante e um módulo pedagógico.

O sistema KERMIT não contém um solucionador de problemas, mas contém uma base de dados de problemas, que é um conjunto de problemas com as suas soluções ideais, especificadas por um especialista humano. Em cada problema são descritas as especificações de uma base de dados a ser projetada pelo estudante. O texto de cada problema é representado internamente por "tags" que fazem a ligação com a solução ideal. Para verificar uma solução apresentada pelo estudante, o sistema compara essa solução com a solução correta, usando as restrições existentes no domínio do conhecimento.

Da mesma forma que no SQL-Tutor, as restrições são de dois tipos: sintáticas, que se referem á apresentação do texto e semânticas, que se referem à comparação entre a solução apresentada e a solução correta.

A interface do estudante (**Figura 13**), tem três janelas. A superior, contém o texto que especifica o problema e uma janela de lista de níveis de ajuda. A intermediária, é o espaço onde o estudante desenvolve a solução do problema. Na janela inferior, é apresentada a mensagem de "feedback".

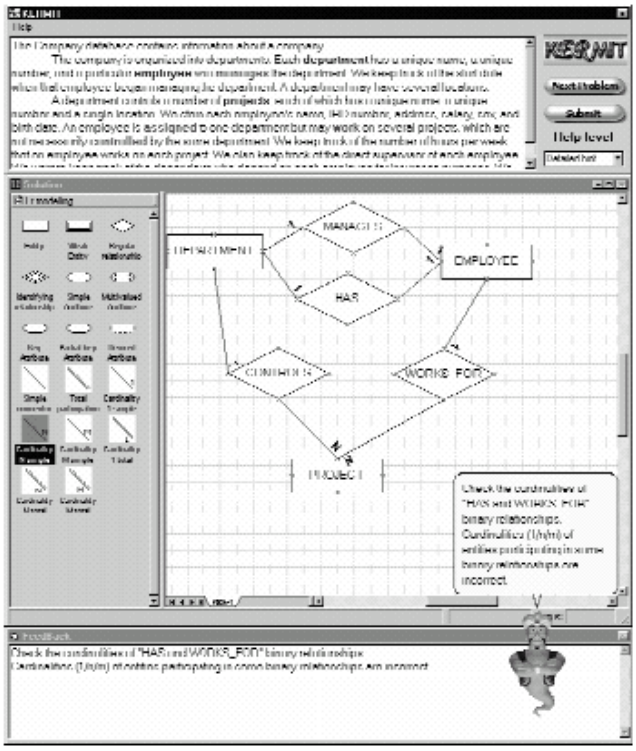

Figura 13: A interface com o estudante do sistema KERMIT.

3.4. AlgoLC – um sistema para o ensino e aprendizagem de algoritmos utilizando um companheiro de aprendizagem colaborativo

Petry (2005) propõe um sistema para o ensino de algoritmos, que denominou AlgoLC, utilizando um companheiro de aprendizagem, com a aplicação da técnica de Modelagem Baseada em Restrições.

3.4.1. Arquitetura

A arquitetura do AlgoLC (**Figura 14**) mostra um sistema com os mesmos componentes já vistos nos anteriores, acrescido do módulo do companheiro de aprendizagem.

O modelo do domínio contém as informações corretas na forma de restrições que não podem ser violadas pelo estudante.

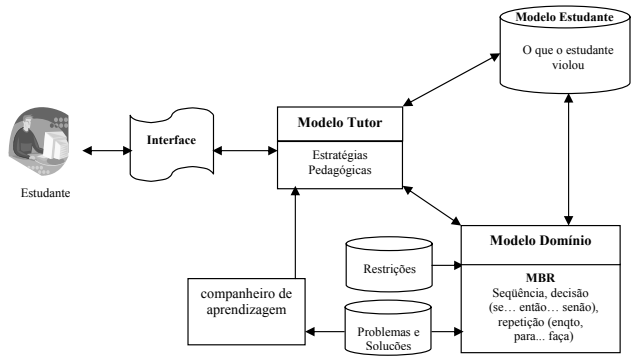

Figura 14: A estrutura do AlgoLC (Petry, 2005).

O modelo do estudante contém, além das suas características pessoais, as restrições violadas e não violadas. Esses dados são utilizados pelo tutor para selecionar a seqüência de abordagem da matéria.

O modelo do tutor contém as estratégias pedagógicas que tratam do conhecimento sobre o ensino do domínio.

O companheiro de aprendizagem interage com o modelo do tutor e com o estudante, para orientar o processo de aprendizagem através das informações recebidas do tutor. No caso do protótipo descrito por Petry (2005), o companheiro de aprendizagem reage a cada linha do algoritmo digitada pelo estudante, apresentando a mensagem de "feedback" apropriada no caso de violação de alguma restrição. Se nenhuma restrição for violada, o companheiro de aprendizagem não apresenta qualquer reação.

As restrições estão organizadas em quatro níveis, correspondendo ao nível de dificuldade desejado pelo estudante: declaração de variáveis, estruturas seqüenciais, estruturas de decisão e estruturas de repetição. São 33 (trinta e três) restrições para a abordagem da solução de 6 (seis) problemas. A título de exemplo, a restrição número 10:

10. Cr: Se (enquanto variável_1 <= variável_2) então:

Cs: O valor da variável_1 deve ser incrementado até ser maior que a variável_2 (senão acionar companheiro de aprendizagem).

3.4.2. Funcionamento

Na **Figura 15**, é demonstrado o funcionamento do sistema.

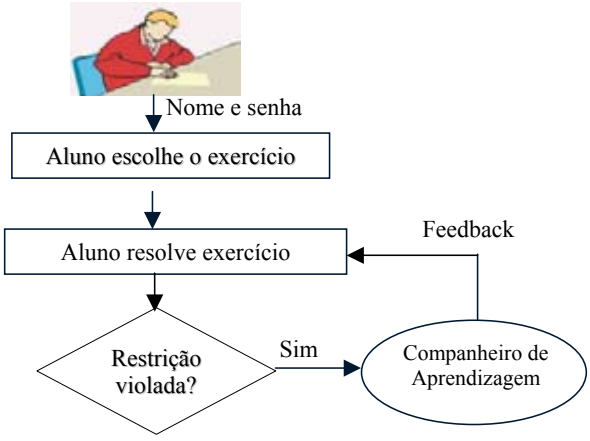

Figura 15: Fluxograma de funcionamento do AlgoLC (Petry, 2005).

Após se identificar e escolher o processo a seguir na tela representada na **Figura 16**, o estudante recebe a tela de trabalho representada na **Figura 17**, onde escolhe o grau de dificuldade do exercício que deseja realizar. Na área "Algoritmo", o estudante desenvolve o algoritmo que resolve o exercício proposto.

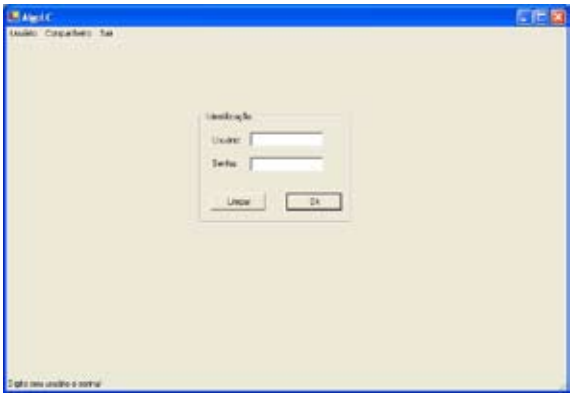

Figura 16: Tela inicial do AlgoLC (Petry, 2005).

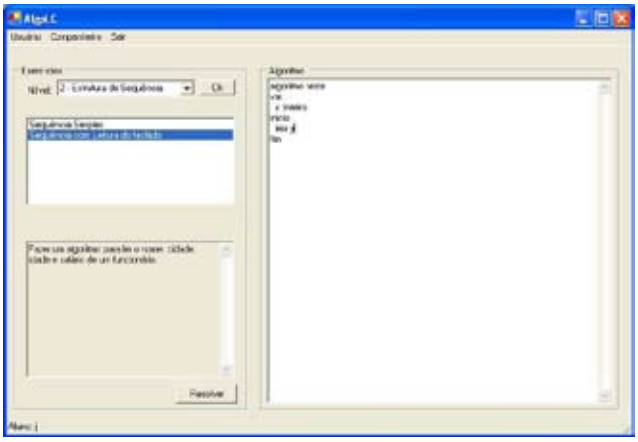

Figura 17: Tela de trabalho do AlgoLC (Petry, 2005).

A cada linha digitada, o AlgoLC verifica a violação de alguma restrição e, se houve violação, é apresentada uma mensagem de erro correspondente à restrição violada (**Figura 18**).

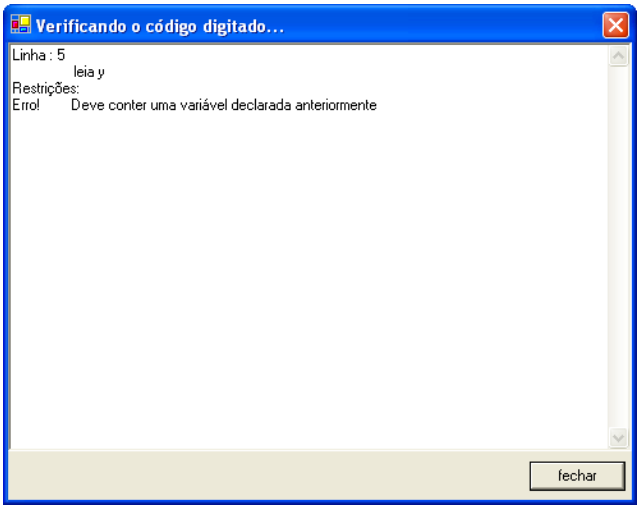

Figura 18: Tela de demonstração de violação de restrição (Petry, 2005).

Quando o exercício é completado, é apresentada uma tela (Figura 19), indicando que não há mais erros a corrigir.

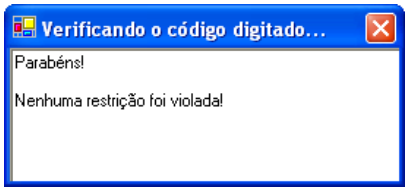

Figura 19: Tela de conclusão (Petry, 2005).

O sistema apresenta também relatórios demonstrando o desempenho do estudante (Figuras 20 e 21).

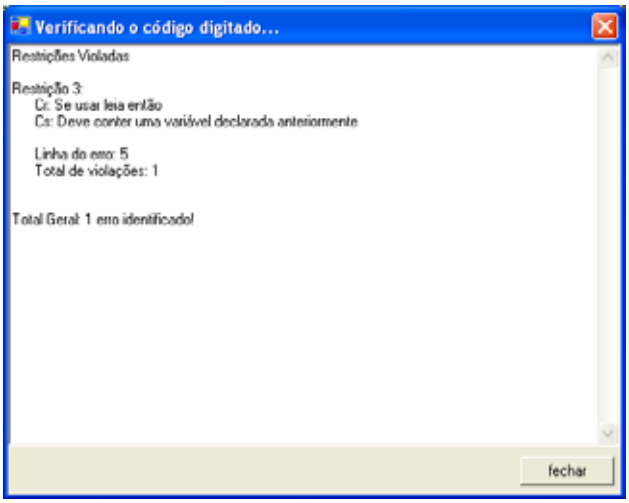

Figura 20: Relatório por estudante (Petry, 2005).

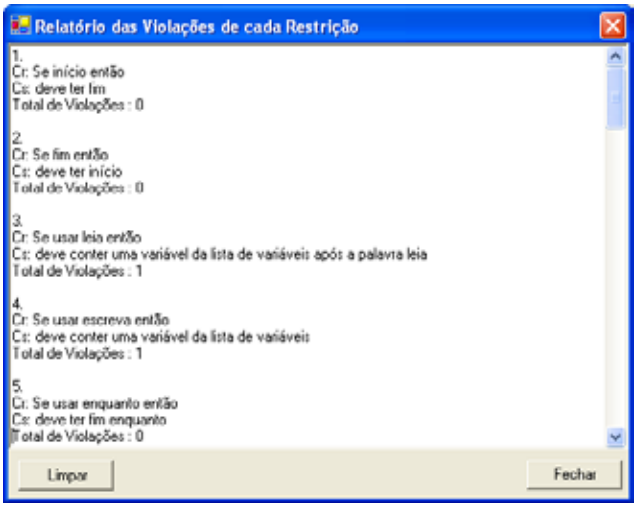

Figura 21: Relatório por restrição (Petry, 2005).

4. CONCLUSÃO

Examinando os sistemas acima descritos pode-se ver que apenas o objeto do conhecimento é diferente mas, em todos eles, a estrutura é a mesma, contendo cláusula de relevância e cláusula de satisfação para cada cláusula de relevância. A essas duas informações são acrescentadas mensagens de "feedback", contadores de ocorrências de violações e não violações e

outras informações, dependendo do projeto do sistema tutor.

A grande vantagem que se observa na modelagem baseada em restrições, é a possibilidade da estruturação da base do conhecimento de forma independente do domínio objeto da tutoria que é representado pelos conteúdos das cláusulas. Assim, é possível a construção de um sistema tutor genérico, no qual as restrições são alimentadas por um especialista no domínio, sem que seja necessariamente conhecedor de técnicas de desenvolvimento de programas de computador.

Além disso, é possível melhorar as restrições e, conseqüentemente, o modelo do estudante, com o sistema tutor já disponível para uso, bastando acrescentar novas restrições à base de conhecimento do sistema.

5. UMA PROPOSTA PARA A MODELAGEM DO ESTUDANTE

A modelagem do estudante consiste na aquisição dos dados pessoais do estudante, dos dados sobre a sua escolaridade e dos dados sobre o seu conhecimento do domínio.

Os dados pessoais são necessariamente, o nome e a data do nascimento. Sobre a escolaridade se deve saber o curso e a turma onde o estudante está inserido. Sobre o conhecimento do domínio, deve ser utilizada a modelagem baseada em restrições.

5.1. Como modelar o estudante

Como já visto em tópicos anteriores, a modelagem baseada em restrições consiste no uso de duas cláusulas: a de relevância e a de satisfação, para a representação do conhecimento. No modelo do estudante, serão registradas apenas as restrições violadas, ou seja, aquelas em que ocorreu a relevância, mas o estudante não conseguiu encontrar qualquer caminho para o atendimento da cláusula de satisfação. As restrições em que a cláusula de satisfação foi encontrada pelo estudante, não serão registradas no modelo do estudante.

5.2. Forma de registro das restrições

Para adequação a tecnologia disponível para o registro de informações, a estrutura de dados do modelo do estudante devera ser na forma de "registro", contendo, em cada registro, a identificação da restrição violada e uma indicação de que essa restrição foi posteriormente satisfeita ou substituída por uma outra restrição e, neste caso, qual a identificação da restrição substituta.

5.3 A identificação de uma restrição

Usando o modelo de hierarquia, uma restrição devera ser identificada por um código indicando o tópico ao qual pertence essa restrição seguido de um código indicando a restrição.

No modelo do domínio, esse código será relacionado com a definição da restrição, ou seja, com a cláusula de relevância e com a cláusula de sa-tisfação.

No modelo pedagógico, esse código será empregado para a localização da seqüência de abordagem da matéria.

No modelo do estudante, como já visto, será usado para indicar as restrições violadas.

REFERÊNCIAS BIBLIOGRÁFICAS

FONSECA JR., R. D. **Um modelo para sistemas inteligentes adaptativos**. Dissertação apresentada ao Departamento de Ciência de Computação da Universidade de Brasília, 2004.

GIRAFFA, L.M. M. **Seleção e adoção de estratégias de ensino em sistemas tutores inteligentes**. Exame de qualificação de doutorado em Ciência da Computação pela Universidade Federal do Rio Grande do Sul, Porto Alegre, 1997.

JONASSEN, D. H. The physics tutor: integrating hypertext and expert systems. **Journal of Educational Technology Systems**, v. 22(1), pp.19-28, 1993.

MARTIN, B. Constraint-based modeling: representing student knowledge. **New Zealand Journal of Computing**, 1999.

MAYO M.; MITROVIC A. ; McKENZIE J. CAPIT: an intelligent tutoring system for capitalisation and punctuation. In Kinshuk, Jesshope C. and Okamoto T. (Eds.) **Advanced learning technology**: design and development issues, Los Alamitos, CA: IEEE Computer Society (ISBN 0-7695-0653-4), pp. 151-154, 2000.

MIRTOVIC, A. SQL-Tutor: a preliminary report. **Technical report** n. TR-COSC 08.97, Christchurch, New Zealand: Computer Science Department, University of Canterbury, 1997.

MITROVIC, A.; KOEDINGER K. R. ; MARTIN, B. A Comparative analysis of cognitive tutoring and constraint-based modeling. In: Brusilovsky, P., *et al*. (eds.) **Proc. 9th International Conference on User Modelling UM2003**, PittsFONSECA JR., R. D. Um modelo para sistemas inteligentes adaptativos, Dissertação apresentada ao Departamento de Ciência de Computação da Universidade de Brasília, 2004.

MITROVIC, A.; OHLSSON, S. Evaluation of a constraint-based tutor for a database language. **International Journal of Artificial Intelligence in Education**, n.10, p. 238-256, 1999.

MITROVIC, A.; *et al*. Constraint-based tutors: a success story. **Proceedings of the 6th International Conference on Intelligent Tutoring Systems**, p. 377 – 387, 2002.

MITROVIC, A.; MARTIN, B.; MAYO, M. **Using evaluation to shape ITS design**: results and experiences with SQL-Tutor _user modeling and user - adapted interaction,12, 2-3; ABI/INFORM Global, p.243-279, 2002.

PETRY, P. G. **Um sistema para o ensino e aprendizagem de algoritmos utilizando um companheiro de aprendizagem colaborativo**. Dissertação submetida à Universidade Federal de Santa Catarina como parte dos requisitos para a obtenção do grau de Mestre em Ciência da Computação, 2005.

OHLSSON, S. Constraint-based student modelling. IN: **Student modelling**: the key to individualized knowledge-based instruction, Springer-Verlag. p.168-189, 1994.

REFERÊNCIAS BIBLIOGRÁFICAS

OHLSSON, S. Learning from performance errors. **Psychological Review**, n.103, p. 241-262, 1996.

SELF, J. A. Bypassing the intractable problem of student modeling. IN: C. Frasson and G. Gauthier (eds.), **Intelligent tutoring systems**: at the crossroads of artificial intelligence and education. Norwood: Ablex, p.107-123, 1990.

SLEEMAN, D. ; BROWN, J. S. **Introduction:** intelligent tutoring systems. Academic Press, New York, 1982.

SURAWEERA, P.; MITROVIC, A.; MARTIN, B. **A knowledge acquisition system for constraint-based intelligent tutoring systems**, 2001.

SURAWEERA, P.; MITROVIC, A. Kermit: a constraint-based tutor for database modeling. **Proceedings of the 6th International Conference on Intelligent Tutoring Systems**, p.377 – 387, 2002.

URRETAVIZCAVA, L. M. Sistemas Inteligentes en el ámbito de la educación. **Revista Iberoamericana de Inteligencia Artificial**, n.12, p.5-12, 2001.

WENGER, E. **Artificial intelligence and tutoring systems:** computational and cognitive approaches to the communication of knowledge. Morgan Kaufmann Publishers Inc., 1987.