SGSO – Um Sistema Baseado em Lógica *Fuzzy* para Geração e Correção de Simulados e Provas via *Web*

Paulo Afonso Parreira Junior[1], Heitor Augustus Xavier Costa[2]

[1] GDMS – Grupo de Desenvolvimento e Manutenção de Software
Departamento de Computação – Universidade Federal de São Carlos
Caixa Postal 676 – CEP 13.565-905 – São Carlos – SP - Brasil

[2] PqES – Grupo de Pesquisa em Engenharia de Software
Departamento de Ciência da Computação – Universidade Federal de Lavras
Caixa Postal 3037 – CEP 37.200-000 – Lavras – MG – Brasil

paulo_junior@dc.ufscar.br, heitor@ufla.br

Abstract. *In this paper, a tool (SGSO) to build and to correct tests was developed using fuzzy logic to choose the questions. These tests can be accomplished via Web. The questions are chosen automatically in accord to difficulty level of the evaluation defined by teacher. So, it is possible to select some questions, based in a knowledge base maintained by SGSO, which represents the difficulty level selected.*

Resumo. *Neste trabalho, foi desenvolvida uma ferramenta (SGSO) para gerar e corrigir simulados e provas via Web utilizando lógica fuzzy para escolher as questões. Esta escolha é feita automaticamente a partir do nível de dificuldade da avaliação definido a priori pelo professor. Baseado nesta informação e em uma base de conhecimentos mantida pelo sistema, o SGSO é capaz de selecionar dentre inúmeras questões, aquelas que melhor representam o nível de dificuldade selecionado.*

1. Introdução

Com a revolução da disseminação e da transferência de conhecimentos promovida pela Internet, o mercado global tem se tornado cada vez mais exigente, requerendo diferenciais nos candidatos a empregos [Nozawa; Oliveira, 2006]. Nesse contexto, muitas pessoas buscam participar de projetos de Inclusão Digital visando melhor qualificação profissional.

A motivação deste trabalho foi baseada em alguns problemas observados no decorrer do curso de Introdução à Microinformática do Projeto de Qualificação Profissional da UFLA [UFLA, 2009], a saber: i) resultados de provas teóricas e práticas eram insatisfatórios; ii) resolução de exercícios em sala de aula atrasava o cronograma do curso; e iii) não havia *feedback* contínuo para o professor sobre o desempenho da turma. Podem ser identificadas algumas causas para estes problemas: i) nível de dificuldade das provas estava além do necessário para avaliar o conhecimento dos alunos; ii) tarefa de elaborar exercícios e transferi-los para o quadro-negro demandava tempo; e iii) professor aplicava uma única avaliação ao final do curso.

Segundo [Nozawa; Oliveira 2006], no processo de ensino e aprendizagem, tão importante quanto a escolha dos conteúdos e das técnicas a serem utilizadas é conhecer

o estado cognitivo do grupo ou, se possível, de cada aprendiz. A avaliação é um instrumento que visa verificar o que foi apreendido pelos alunos e a qualidade do ensino aplicado pelo professor. As avaliações contínuas durante o processo de ensino e aprendizagem são essenciais para o mapeamento do conhecimento do estudante, de modo que o seu modelo de aprendizagem possa ser estabelecido.

Contudo, uma avaliação fácil não examina eficientemente o aluno, pois ele poderia estudar o mínimo necessário. Por outro lado, preparar uma prova difícil, buscando descobrir o que o aluno não assimilou também não é o melhor, pois poderia desanimar/desestimular o aluno. Por isso, o ideal é o equilíbrio, ou seja, uma avaliação nem fácil nem difícil; o professor deve cobrar os conhecimentos fundamentais da matéria e inserir uma ou duas questões de um grau de dificuldade maior. Desta forma, ele tem como mensurar o conhecimento do aluno.

Neste contexto, o objetivo deste trabalho é apresentar um sistema baseado em lógica *fuzzy* para gerar e corrigir simulados e provas via *Web*, SGSO – Sistema Gerador de Simulados *Online*. O SGSO auxilia professores e alunos nos seguintes aspectos:

- **Auxílio ao professor**: i) armazena históricos de provas e de simulados resolvidos, mantendo *feedback* contínuo do desempenho da turma; ii) contém um módulo que permite gerar provas e simulados automaticamente com opção de seleção do nível de dificuldade (somente para geração de provas), adequando o nível da avaliação ao desempenho da turma; e iii) permite elaboração de provas por tópicos do conteúdo;

- **Auxílio ao aluno**: i) permite auto-avaliação com a realização de simulados; ii) permite estudar de qualquer lugar onde exista acesso a Internet; e iii) armazena o histórico de provas e simulados realizados para acompanhamento.

Este artigo está organizado como segue: a seção 2 reúne trabalhos encontrados na literatura que utilizam recursos computacionais e técnicos similares ao SGSO para a geração e para a correção automática de provas; a seção 3 aborda principais conceitos de lógica *fuzzy* usados para implementar o processo de escolha das questões das provas; a seção 4 apresenta a modelagem *fuzzy* do SGSO e suas principais funções; a seção 5 apresenta resultados obtidos a partir do uso do SGSO em uma turma do curso de Introdução à Microinformática comparando-os com resultados de outras turmas que não o utilizaram; e a seção 6 apresenta algumas considerações finais e propostas para trabalhos futuros.

2. Trabalhos Relacionados

Aplicações de lógica *fuzzy* podem ser encontradas em várias áreas do conhecimento. Na educação, pode-se destacar o trabalho de Ribeiro (2007), que desenvolveu um software para apoiar professoras do Ensino Fundamental durante a tarefa de avaliar seus alunos, sobretudo, no que diz respeito à fase de notação numérica (atribuição de notas). A lógica *fuzzy* foi aplicada como ferramenta para manipulação e conversão de dados qualitativos em quantitativos. Esta abordagem leva em consideração um processo de avaliação de cunho predominantemente qualitativo.

Rieder; Brancher (2004) apresentaram um jogo interativo de matemática que usa recursos computacionais inteligentes, oferecendo informações personalizadas ao professor do desempenho do aluno dentro de um ambiente tridimensional de ensino.

Para tal, cada ação do usuário é monitorada usando regras de inferência *fuzzy* em um sistema de tutoria inteligente. Em Fabri (2002), a lógica *fuzzy* é usada no acompanhamento do desempenho de alunos em cursos de Ensino a Distância (EAD), determinando se o aluno está ou não apto a avançar um módulo. Queiroz (2002) apresenta um algoritmo para geração automática de currículo para ambientes de EAD, utilizando um agente pedagógico que usa camadas cognitivas, por intermédio da lógica *fuzzy* e redes de Petri.

Além destes trabalhos, pode-se encontrar na literatura algumas propostas de ferramentas que auxiliam professores no processo de avaliação, permitindo a geração de provas e de simulados via *Web* [Moodle, 2009; Silva *et al*., 2007; Nozawa; Oliveira, 2006; Silva, 2005; Abrão *et al*., 2004; Martins *et al*., 2001; Cunha *et al*., 2001; Costa; Xexéo, 1996; Pimentel; Hagui, 1996; Gordin *et al*., 1996].

Contudo, a proposta deste trabalho apresenta inovações em relação aos demais trabalhos apresentados anteriormente, a saber: i) o SGSO automatiza a tarefa de classificação das questões cadastradas, sendo capaz de gerar provas automaticamente, baseando-se em dados históricos de provas resolvidas e em uma base de conhecimento fornecida pelo próprio professor; ii) oferece a possibilidade do professor selecionar o nível de dificuldade da prova que o SGSO deve gerar; e iii) utiliza conceitos da lógica *fuzzy* no processo de seleção das questões das provas.

3. Lógica *Fuzzy*

A lógica *fuzzy* foi criada em 1965 por Lotfi A. Zadeh [Zadeh, 1965], professor do Departamento de Engenharia Elétrica da Universidade da Califórnia em Berkeley e pode ser entendida como uma generalização da lógica clássica que admite infinitos valores lógicos intermediários entre a falsidade e a verdade.

De acordo com Souza (2004), a diferença entre a lógica *fuzzy* e a lógica clássica é que a primeira possuir mais de dois resultados distintos, o que não ocorre com a segunda. Esses resultados não são expressos de forma bem definida, mas lingüisticamente como: “difícil”, “muito difícil”, “fácil” e “muito fácil”. Tais valores estão contidos em um conjunto *fuzzy*. Cada conjunto *fuzzy*, A, é definido em termos de relevância a um conjunto universal, *U*, por uma função denominada de função de pertinência, associando a cada elemento *x* um número, $\mu A(x)$, no intervalo fechado [0,1] que caracteriza o grau de pertinência de *x* em A. O fator de pertinência pode assumir qualquer valor entre 0 e 1, representando completa exclusão e completa pertinência, respectivamente. A função de pertinência tem a forma $\mu A: X \rightarrow [0, 1]$.

Os sistemas *fuzzy* utilizam um conjunto de regras do tipo “*Se-Então*” baseadas em variáveis lingüísticas [Zadeh, 1978; Lee, 1990]. Inicialmente, as variáveis de entrada sofrem um processo de “fuzzificação”, ou seja, é feito um mapeamento do domínio de números reais para o domínio *fuzzy*. Em seguida, efetua-se a inferência sobre o conjunto de regras *fuzzy* obtendo os valores dos termos das variáveis de saída. O mecanismo de inferência define a maneira de como as regras são combinadas, provendo base para tomada de decisões [Mendel, 1995]. Há muitos procedimentos inferenciais na lógica *fuzzy*, porém os mais usados são Mamdani [Fuzzy, 1996] e Takagi-Sugeno-Kang [Mendel, 2001]. Por fim, as variáveis de saída sofrem um processo de “defuzzificação” que consiste em converter dados *fuzzy* para valores numéricos precisos. Para isto, na

inferência de Mamdani, são utilizadas várias técnicas, tais como valor máximo, média dos máximos, média local dos máximos, centro de gravidade, ponto central da área e o centro da média [Driankov *et al*., 1993; Dutta 1993; Fuzzy, 1996].

Uma regra *Se* (antecedente) *Então* (conseqüente) é definida pelo produto cartesiano *fuzzy* dos conjuntos *fuzzy* que a compõem. A inferência de Mamdani agrega as regras usando o operador lógico *OU* modelado pelo operador máximo e, em cada regra, o operador lógico *E* é modelado pelo operador mínimo.

A lógica *fuzzy* vem tornando possível aproximar a máquina do raciocínio humano, propondo soluções mais realistas a problemas que apenas o cérebro humano era capaz de interpretar e resolver [Santos, 2003].

4. SGSO – Sistema Gerador de Simulados *Online*

4.1. Modelagem *Fuzzy*

Neste trabalho, foram definidas uma variável de entrada (“Nível de Dificuldade”) e três variáveis de saída (“Porcentagem de Questões Fáceis”, “Porcentagem de Questões Intermediárias” e “Porcentagem de Questões Difíceis”). A Figura 1 mostra a representação gráfica das variáveis e de seus termos lingüísticos. Os gráficos referentes às outras duas variáveis de saída foram omitidos, pois se assemelham ao gráfico da “Porcentagem de Questões Fáceis”.

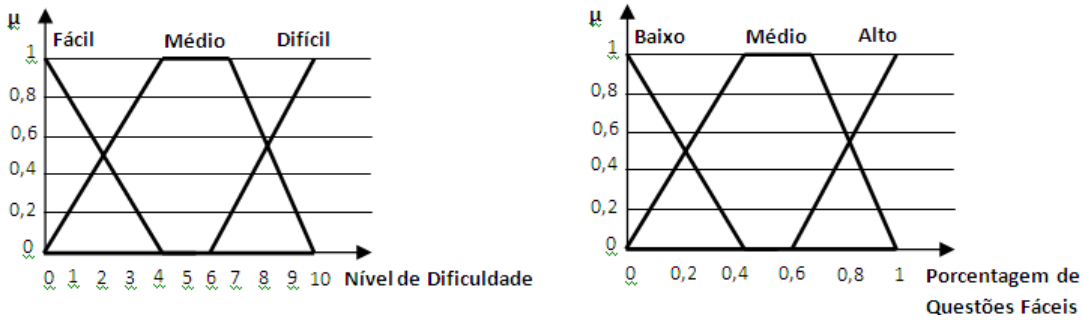

Figura 1 – Representação Gráfica das Variáveis Lingüísticas “Nível de Dificuldade” e “Porcentagem de Questões Fáceis”

A variável “Nível de Dificuldade” corresponde ao grau de dificuldade de uma prova e pode assumir valores reais entre 0 e 10. Esta variável é composta dos termos: “Fácil”, “Médio” e “Difícil”. De modo análogo, as variáveis lingüísticas de saída representam a quantidade de questões fáceis, médias e difíceis que estarão contidas em uma prova. Estas variáveis são compostas dos termos: “Baixo”, “Médio” e “Alto”.

Em seguida, foram feitas algumas escolhas: i) o método de inferência de Mamdani foi escolhido, pois as funções de pertinência de saída do modelo proposto são conjuntos *fuzzy*, o que não é permitido no método Takagi-Sugeno-Kang; ii) operador de intersecção escolhido foi o mínimo e o de união foi o máximo, pois são os mais utilizados [Mendel, 2001]; e iii) método de “defuzzificação” escolhido foi Centro de Gravidade. A Figura 2 apresenta a base de regras *fuzzy* do SGSO que é composta de 9 (nove) regras “*Se-Então*”.

Regra 1: Se (nd é Fácil) **então** (pqf é Alto);
Regra 2: Se (nd é Médio) **então** (pqf é Médio);
Regra 3: Se (nd é Difícil) **então** (pqf é Baixo) ;
Regra 4: Se (nd é Fácil) **então** (pqi é Médio);
Regra 5: Se (nd é Médio) **então** (pqi é Médio);
Regra 6: Se (nd é Difícil) **então** (pqi é Médio);
Regra 7: Se (nd é Fácil) **então** (pqd é Baixo);
Regra 8: Se (nd é Médio) **então** (pqd é Médio);
Regra 9: Se (nd é Difícil) **então** (pqd é Alto);

Legenda: nd = Nível de Dificuldade;
pqf = Porcentagem de Questões Fáceis;
pqi = Porcentagem de Questões Intermediárias;
pqd = Porcentagem de Questões Difíceis;

Figura 2 – Conjunto de Regras *Fuzzy* do SGSO

A Figura 3 apresenta um cenário de execução das três primeiras regras apresentadas anteriormente. Além disso, esta figura apresenta a saída real *Z* de um sistema de inferência do tipo Mamdani gerada a partir do valor de entrada 2,5. Este valor de entrada corresponde ao grau de dificuldade desejado pelo usuário.

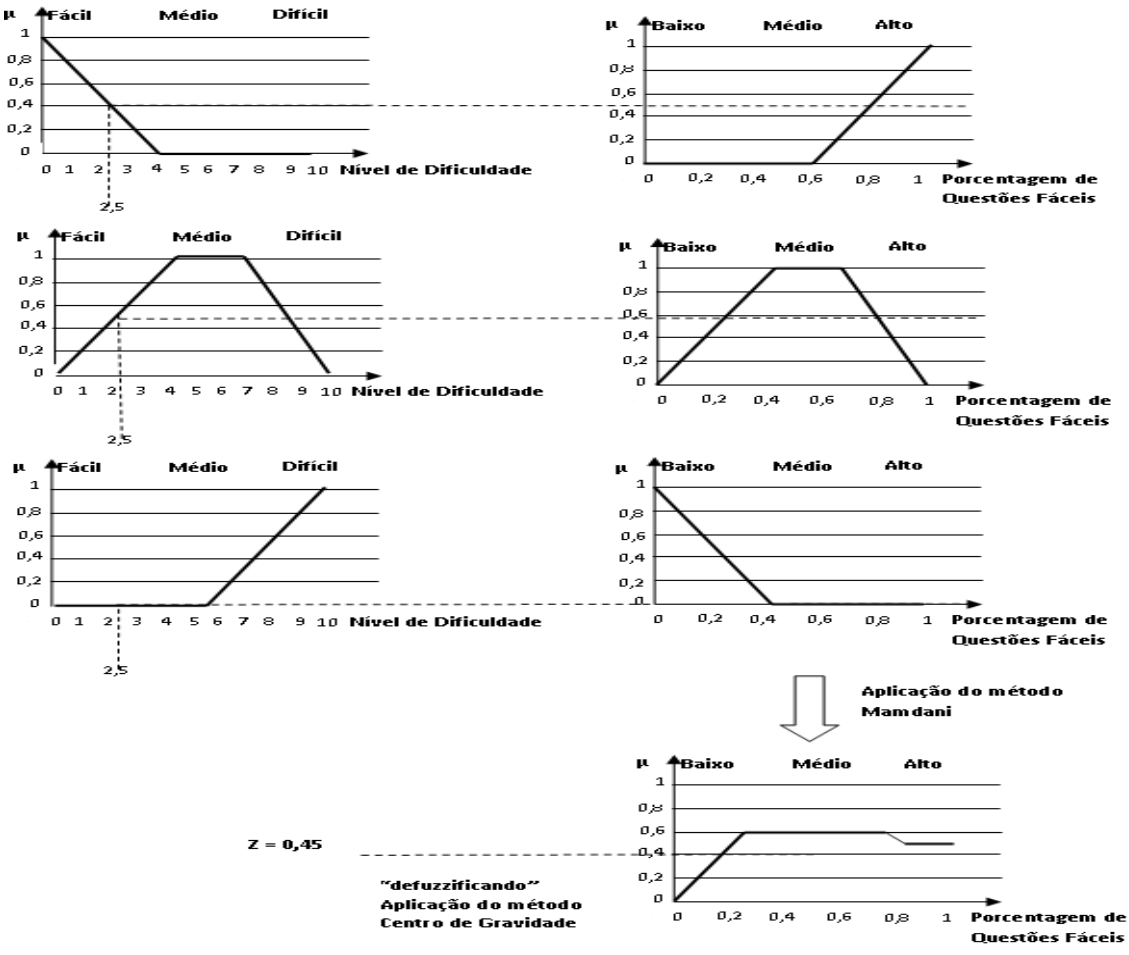

Figura 3 – Cenário de Execução de Três Regras do Sistema *Fuzzy*

A saída *Z* representa a porcentagem de questões fáceis que deve ser adicionada à prova para obter o nível de dificuldade adequado. Como explicado, o modelo definido neste trabalho possui três variáveis de saída, assim, dado um grau de dificuldade *x*, o SGSO retorna como resultado uma tripla ordenada *(y, w, z)* que representa,

respectivamente, porcentagem de questões fáceis, intermediárias e difíceis incorporadas à prova. Para classificar uma questão quanto ao seu nível de dificuldade (Fácil, Médio ou Difícil), é utilizado o fator índice de erros da questão, calculado a partir de:

$$IEQ = (NE / (NA + NE)) * 10,$$

sendo *IEQ* o índice de erros de uma determinada questão, *NA* o número de acertos e *NE* o número de erros da questão. Desta forma, pode-se observar que, quanto mais próximo de 10 estiver o valor de *IEQ*, mais difícil é a questão. Toda questão recebe um nível de dificuldade inicial definido pelo usuário. Este procedimento é necessário para garantir que o nível de dificuldade inicial da questão não seja incoerente. À medida que simulados e provas são realizados, os valores *NE* e *NA* são atualizados.

4.2. Funcionalidade do SGSO

4.2.1 Interface

Segundo Nozawa; Oliveira (2006), se a interface do sistema não possuir apresentação adequada, o usuário, no caso o aluno, pode sentir-se desmotivado, confuso, perplexo ou sobrecarregado, dificultando a aprendizagem. Assim sendo, optou-se por estruturar a interface do SGSO com poucos recursos visuais para que eles não tirem a concentração durante a navegação. A página inicial do sistema possui um *menu* de acesso às suas funções, uma seção para apresentar o conteúdo solicitado e uma seção para apresentar informações sobre o projeto ao qual a ferramenta é aplicada.

4.2.2 Atores do SGSO

Há cinco atores[1], relacionados por herança, que interagem com o SGSO: “Visitante”, “Aluno”, “Instrutor”, “Tutor” e “Administrador”. O ator “Visitante” pode “Gerar Simulados”. O ator “Aluno” herda a ação do “Visitante” e pode “Efetuar *Login/Logout*”, “Visualizar Histórico”, “Alterar Senha”, “Editar Perfil” e “Visualizar Tarefas”. O ator “Instrutor” herda as ações de “Aluno” e pode “Editar Configurações”, “Gerar Prova”, “Manter Tópicos”, “Manter Questões” e “Manter Alunos”. O ator “Tutor” herda ações de “Instrutor” e pode “Manter Instrutores”. O ator “Administrador” herda ações de “Tutor” e pode “Manter Cursos” e “Manter Tutores”.

4.2.3 Manutenção de Questões

Os atores “Instrutor”, “Tutor” e “Administrador” podem manter o cadastro de questões. É importante salientar que o ator deve informar um nível de dificuldade inicial para a questão (número real entre 0 e 10), sendo que, quanto mais próximo de 10, mais difícil é a questão. Este valor será alterado pelo SGSO à medida que esta questão seja resolvida nos simulados e nas provas, segundo a fórmula apresentada na seção 4.1. Quanto menos acertos esta questão tiver, maior será seu grau de dificuldade. Em seguida, o ator deve cadastrar as alternativas para a questão. É possível cadastrar várias alternativas para uma questão, mas uma proporção entre alternativas certas e erradas deve ser respeitada. Esta proporção é definida no módulo de configuração do curso.

[1] Um ator representa um papel que um ser humano, um dispositivo de hardware ou outro software desempenha, interagindo com o software [Booch *et al*. 2004].

4.2.4 Gerar Simulados e Provas

Para gerar um simulado, o ator (todos têm permissão) deve selecionar a quantidade de questões e a lista de tópicos do conteúdo a serem abordados. A Figura 4 ilustra a página de geração de simulados e a Figura 5 apresenta um simulado gerado.

Figura 4 – Geração de Simulados

Figura 5 – Realização de Simulado

As questões são exibidas ao lado direito da página. O ator deve marcar as alternativas que deseja e clicar em “Enviar”. Logo após, é exibida uma página com o resumo do simulado (Figura 6) apresentando na parte superior a quantidade de questões que o ator acertou e errou, o seu desempenho e o nível do simulado (Fácil, Médio, Difícil). Este resumo é armazenado no seu histórico, a menos que ele seja um “Visitante”.

Analogamente, para gerar uma prova, os atores “Instrutor”, “Tutor” e “Administrador” devem selecionar a quantidade de questões, a lista tópicos do conteúdo, o nível de dificuldade e a opção se desejam ou não disponibilizar a prova aos alunos naquele momento. Uma prova disponível aparece na lista de tarefas dos alunos.

4.2.5 Configurações

A plataforma Java [Java, 2009] e a API *FuzzyJ Toolkit* [FuzzyJ, 2009]foram escolhidas para implementar o modelo *fuzzy* do SGSO, pois são gratuitas e apresentam características de portabilidade. A API *FuzzyJ Toolkit* fornece métodos e classes que auxiliam no processo de modelagem *fuzzy*, permitindo fácil definição de variáveis lingüísticas, de conjuntos de regras *fuzzy* e de termos lingüísticos e a escolha do método de inferência e do método de “defuzzificação”.

No SGSO, existe uma configuração específica para cada curso cadastrado. Quando um novo curso é inserido, uma configuração padrão é criada, podendo ser alterada usando o *menu* “Minhas Configurações” (Figura 6). Para isto, o ator deve ser “Instrutor”, “Tutor” ou “Administrador”. Estes atores podem alterar alguns parâmetros, como: i) quantidade máxima de questões; ii) quantidade de alternativas por questões; e iii) quantidade de questões certas para cada questão errada.

5. Resultados

O SGSO foi usado nas aulas do curso de Introdução à Microinformática do Projeto de Qualificação Profissional da UFLA [UFLA, 2009]. Ele foi apresentado para a turma do 1º semestre de 2008 que compreendeu os meses de Maio a Julho com aulas duas vezes por semana em um laboratório de computadores. O SGSO foi utilizado continuamente, gerando simulados no início das aulas e abordando tópicos apresentados na aula anterior. Inicialmente, o SGSO foi apresentado aos alunos por meio de uma demonstração realizada em horário de aula, visando explicar o seu funcionamento e permitir que os alunos se familiarizassem com sua interface.

A Figura 7 apresenta algumas observações coletadas relativas à melhoria no desempenho da turma em relação às notas obtidas nos simulados, após a apresentação e o uso do SGSO. Notou-se que, com o SGSO, houve aumento da média das notas dos alunos em relação aos simulados realizados em sala de aula. Observa-se que a média, que antes estava abaixo de 6,0, chegou a variar entre 8,0 e 9,0 (4º ao 9º simulado) e atingiu seu valor máximo no 10º simulado.

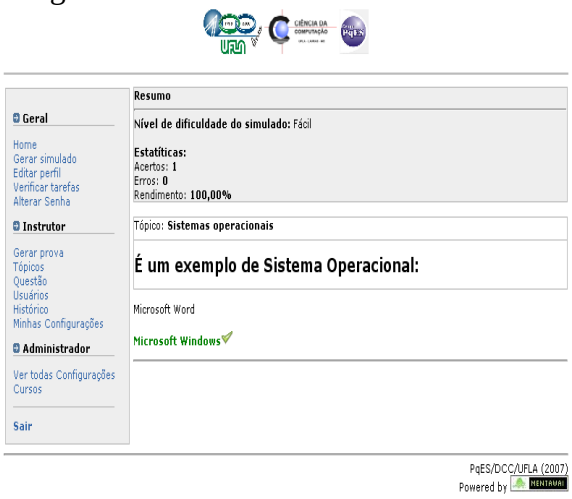

Figura 6 – Resultado do Simulado

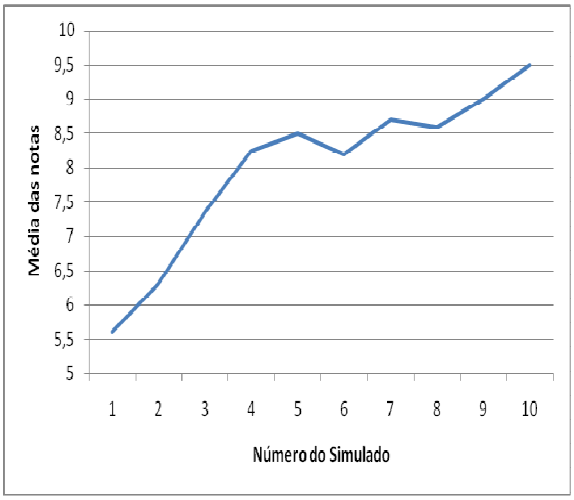

Figura 7 – Desempenho dos Alunos do Curso de Introdução à Microinformática

A Figura 8 mostra o perfil de duas turmas analisadas, turma do 1º período de 2008 que usou o SGSO e turma do 2º período de 2007 que não usou o SGSO, sendo que ambas possuíam aproximadamente 20 alunos com nível de conhecimento de informática básica aproximado. A Figura 9 mostra a comparação entre as médias destas turmas. Baseado nestas figuras, pode-se perceber que o uso do SGSO como auxílio à aprendizagem de informática básica alterou o ânimo e a motivação dos alunos. Mesmo sem seu uso mediado pelos instrutores, os alunos mencionavam a ferramenta durante as aulas, relatando seus estudos extra-classe (em casa, no trabalho e em *lanhouses*).

6. Conclusões

Embora existam diversos geradores automáticos de provas, o diferencial do SGSO é o uso da lógica *fuzzy* no suporte à escolha das questões. Às vezes, o professor de uma disciplina ao elaborar uma prova pode equivocar-se e fazê-la muito difícil ou muito fácil. Esta característica não é determinada pela qualidade do professor, mas pelos alunos que cursam a disciplina. Desta forma, o SGSO fornece subsídio importante e

relevante para a elaboração de uma prova com nível de dificuldade adequado aos alunos da disciplina.

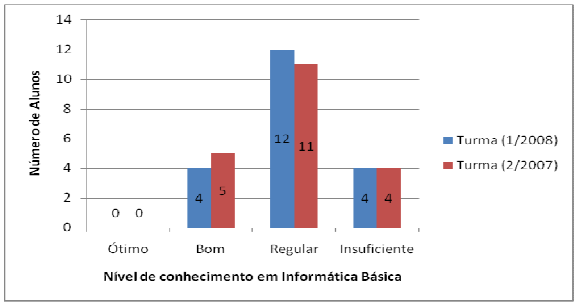

Figura 8 – Conhecimento em Informática Básica

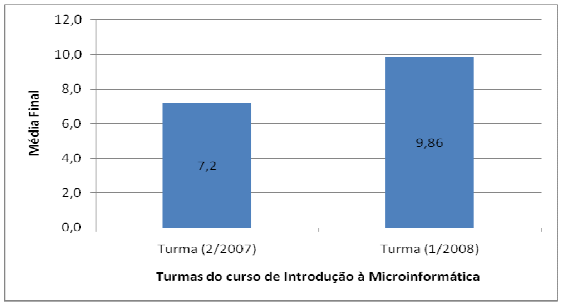

Figura 9 – Médias das Turmas

Há a intenção de implantar o SGSO nos cursos de *lato-sensu* modalidade de tutoria à distância ofertados pelo Departamento de Ciência da Computação da Universidade Federal de Lavras. Além disso, está prevista a implantação do SGSO em um cursinho pré-vestibular mantido em parceria pela Universidade Federal de Lavras e a Prefeitura Municipal de Lavras. Desta forma, espera-se pela abertura de novas turmas para que mais resultados possam ser medidos e comparados com os resultados anteriores, tornando os resultados mais significativos.

Referências Bibliográficas

Abrão, I. C., Rayel, F., Abrão, M. A. V. L. “QUESTCOMP: Ferramenta para Avaliação de Aprendizado à Distância”. World Conference on Engineering and Technology Education, Guaruja. Engineering Education in the Changing Society. 2004

Booch, G.; Rumbaugh, J.; Jacobson, I. “Unified Modeling Language Reference Manual. Addison Wesley. 2ª Edição – 2004.

Costa, R. M. E. M.; Xexéo, G. P. “A Internet nas Escolas: Uma Proposta de Ação”. VII Simpósio Brasileiro de Informática na Educação. Belo Horizonte. Novembro 1996.

Cunha, L. M.; Gerosa, M. A.; Fuks, H.; Lucena, C. J. P. “Desenvolvimento e Aplicação de Cursos Totalmente a Distância na Internet”. XIX Workshop de Educação em Informática. 2001.

Driankov, D.; Hellendoorn, H.; Reinfrank, M. “An Introduction to Fuzzy Control”. Springer-Verlag. 1993.

Dutta, S. “Fuzzy Logic Applications: Technological a Strategic Issues”. IEEE Transactions on Engineering Management. 40(3):237-254. 1993.

Fabri, J. A.; Fabri, M. G. S. “Ferramenta Fuzzy para Acompanhamento do Desempenho dos alunos nos Cursos a Distância”. XXII Congresso da Sociedade Brasileira de Computação. Workshop de Informática na Escola. Campinas. 2002.

Fuzzy. “Fuzzy Open-Loop Attitude Control for the FAST Spacecraft”, San Diego - CA. Proc. of the NASA AIAA, Guidance, Navigation and Control Conference, Julho 1996

FuzzyJ Toolkit (2009). “The NRC FuzzyJ Toolkit”. Disponível em: http://www.iit.nrc.ca/IR_public/fuzzy/fuzzyJToolkit2.html. Acessado em: Março de 2009.

Gordin, D. N.; Gómez, L. M.; Peã, R. D.; Fishman, B. J. “Using the World Wide Web to Build Lerning Communities in K-12”. The Journal of Computer-Mediated Communication. v 2, n 3, December, 1996.

Java (2009). Disponível em: http://www.sun.com/java/. Acessado em: Março de 2009.

Lee, C. C. “Fuzzy Logic in Control Systems: Fuzzy Logic Controller (Part I)”. IEEE Transactions on Systems, Man an Cybernetics, 20(2):404 – 418. 1990.

Martins, A. R.; Carneiro, M. L. F.; Fabre, M. C. J. M.; Keller, R. S. “O Suporte em Educação a Distância”. XIX Workshop de Educação em Informática. 2001

Mendel, J. M. “Fuzzy Logic Systems for Engineering: A Tutorial”. Proceedings of the IEEE. v. 83. Issue 3. p. 345-377. 1995.

Mendel, J. M. “Uncertain Rule-Based Fuzzy Logic Systems: Introduction and New Directions”. Londres: Prentice Hall P. T. R. 2001.

Moodle (2009). Disponível em: http://moodle.org/. Acessado em: Março de 2009.

Nozawa, E. H.; Oliveira, E. H. T. “Simulador e-JLPT: Um Software de Apoio Educacional com Enfoque em Hipermídia Adaptativa”. XVII SBIE. DF. 2006.

Pimentel, M. G. C; Hagui, S. H. “Usando a WWW como Ferramenta de Apoio ao Ensino”. VII Simpósio Brasileiro de Informática na Educação. Novembro 1996.

Queiroz, B.; Lopes, C.; Fernandes, M. “Geração Automática de Currículo para um Sistema Educacional Baseado na Web”. XIII SBIE. 515-517. 2002

Ribeiro, A. P. M. “A Avaliação da Aprendizagem: Aplicação de um Modelo Fuzzy para se Obter Notas Mais Justas na Disciplina de Língua Portuguesa. Dissertação de Mestrado”. UFC. 2007.

Rieder, R.; Brancher, J. D. “Aplicação da Lógica Fuzzy a Jogos Didáticos de Computador – A Experiência do Mercadão GL”. In: VII Congresso Ibero Americano de Informática Educativa, 2004, Monterrey.

Santos, G. J. C. “Lógica Fuzzy”. Monografia de conclusão do curso de Matemática da Universidade Estadual de Santa Cruz. UESC. 2003.

Silva, T. V. A. da. “Uso do Modelo PMBoK® Adaptado ao Desenvolvimento de Uma Ferramenta de Aplicação de Simulados Via Web”. Monografia de Final de Curso. Universidade Federal de Lavras. 75p. 2005.

Silva, T. V. A. da; Costa, H. A. X.; Resende, A. M. P. “Adaptação do Modelo PMBoK ao Desenvolvimento de um Projeto de Pequeno Porte – Projeto SIMIUS”. Revista do CCEI, v. 11, p. 102-113. 2007.

Souza, O. T. L. “Desenvolvimento de um Modelo Fuzzy para Determinação do Latente com Aplicação em Sistemas de Irrigação”. Dissertação de Mestrado, UNESP. 2004

UFLA. “Projeto de Qualificação Profissional”. Universidade Federal de Lavras. UFLA. Disponível em: http://www.proex.ufla.br/semear.htm. Acessado em: Março de 2009.

Zadeh, L. A. “Fuzzy Sets – Information and Control”. 8 : 338-53. 1965.

Zadeh, L. A. “Fuzzy Sets as a Basis for a Theory of Possibility. Fuzzy Sets and Systems”. 1978.