https://eventos.utfpr.edu.br//sei/sei2018

Sistema de Correção Automatizada – Ambiente CD-MOJ

Automated Correction System – CD-MOJ Environment

Gustavo Martins de Souza
gustavooguto@gmail.com
Universidade Tecnológica Federal do Paraná – UTFPR – Campus Pato Branco, Pato Branco, PR, Brasil.

Bruno César Ribas
brunoribas@utfpr.edu.br
Universidade Tecnológica Federal do Paraná – UTFPR – Campus Pato Branco, Pato Branco, PR, Brasil.

RESUMO

O *Contest-Driven Meta Online Judge* (CD-MOJ) é uma plataforma de correção de algoritmos, cuja finalidade é corrigir de forma automatizada submissões de respostas a problemas propostos nas disciplinas de programação e algoritmos. O método de adição de *contests* no CD-MOJ depende de uma série de informações que devem ser seguidas rigorosamente, o que torna este processo complexo. Assim, esta pesquisa tem como objetivo atualizar e incrementar métodos de inserção de problemas e modalidades no Ambiente de Correção Automatizada. Com os novos *scripts elaborados,* o sistema apresentará um método automatizado de algumas funções que antes deveriam ser realizadas pelo professor, facilitando a forma como novos *contests* são disponibilizados aos alunos. O método aqui utilizado foi o levantamento de dados para identificar quais as modificações necessárias no sistema, a fim de modernizar a plataforma.

PALAVRAS-CHAVE: CD-MOJ. Sistema. Correção. Modernização.

ABSTRACT

The Contest-Driven Meta Online Judge (CD-MOJ) is an algorithm-correcting platform whose purpose is to automatically correct submissions of answers to proposed problems in programming disciplines and algorithms. The method of adding contests in the CD-MOJ depends on a series of information that must be followed strictly, which makes this process complex. Thus, this research aims to update and increase methods of insertion of problems and modalities in the Automated Correction Environment. With the new scripts, the system will present an automated method of some functions that should be carried out by the teacher, facilitating the way in which new contests are made available to the students. The method used here was to collect data to identify the necessary modifications to the system in order to modernize the platform.

KEYWORDS: CD-MOJ. System. Correction. Modernization.

Recebido: 31 ago. 2018.
Aprovado: 18 set. 2018.

INTRODUÇÃO

Nos últimos anos as competições acadêmicas na área da computação tem ganhado uma atenção maior entre muitas universidades. Isso se dá ao fato de algumas das grandes empresas como *Google*, *Facebook* contratarem ex-competidores. No Brasil essa competição é chamada de Maratona de Programação.

A Maratona de Programação nasceu de competições regionais classificatórias para finais mundiais de programação e é organizada pela Sociedade Brasileira de Computação desde 1996. Essa tem como público alvo estudantes de graduação e inicio de pós-graduação em áreas de Computação. A Maratona de Programação também promove aos competidores a capacidade de trabalho em equipe, com soluções otimizadas e criativas e a habilidade de conseguir trabalhar sob pressão (SOCIEDADE BRASILEIRA DE COMPUTAÇÃO, 2018).

Muitas universidades do Brasil desenvolvem competições locais a fim de selecionar equipes para participarem da Maratona (DE CAMPOS; FERREIRA, 2004). Além disso, vários professores utilizam sistemas de correção de algoritmos semelhantes aos utilizados na Maratona de Programação.

Atualmente, existem vários ambientes disponíveis para essa prática na sala de aula, porém poucos são disponibilizados como *Software* Livre, ou seja, muitos destes ambientes dependem de uma ferramenta imutável para que o professor possa criar ou selecionar problemas propostos aos alunos.

O *Contest-Driven Meta Online Judge* (CD-MOJ) é um ambiente desenvolvido por um professor da UTFPR – Campus Pato Branco e disponibilizado em *Software* Livre para auxiliar professores em disciplinas de programação e algoritmos a criar, adaptar ou reutilizar problemas de computação e disponibilizar aos alunos dessas disciplinas.

O CD-MOJ utiliza uma série de casos de testes que analisam o comportamento do algoritmo para uma determinada situação de entrada. Se o algoritmo a ser avaliado atende a todas as soluções requeridas pelo *online judge,* então o mesmo é considerado como uma submissão correta e otimizada.

MÉTODOS

Existem plataformas que podem ser utilizadas como *online judges* tanto em competições quanto para resolução de problemas em particular, como por exemplo, o BOCA Online Administrator, SPhere Online Judge (SPOJ) e o KATTIS (DE CAMPOS; FERREIRA, 2004 p.3; ESTIMA 2013 p.29).

Com o grande crescimento no âmbito da programação, a Sociedade Brasileira de Computação usa, na Maratona de Programação, o BOCA. Ele é uma plataforma *online d*esenvolvida inicialmente para gerenciar competições de programação. Existe uma distribuição baseada no Ubuntu, denominada Maratona Linux que inclui o BOCA. (DE CAMPOS; FERREIRA, 2004 p.3).

O SPOJ é uma plataforma de submissão de algoritmos para problemas computacionais, porém surge como um ambiente avulso, que não somente é destinada a competidores. Atualmente o sistema tem suporte para mais de 40

linguagens de programação e milhares de exercícios pra se resolver, com suporte inclusive em português (ESTIMA 2013 p.29).

O KATTIS, utilizado pela *International Collegiate Programming Contest* (ICPC), considerada como a competição mais importante no mundo da programação, é um sistema *online*, ou seja, essa plataforma não precisa ser instalada na máquina do usuário, basta apenas que o mesmo acesse esse ambiente de programação através do navegador. Outras competições também podem ser gerenciadas por ele (ESTIMA 2013 p.29).

REFERENCIAL TEÓRICO

Quando um usuário está utilizando o seu computador ele está, indiretamente, interagindo com o *shell* do sistema, seja ele GNU/Linux, iOS, Microsoft Windows ou qualquer outro sistema operacional. Um tipo de *shell* é o textual, fornecendo uma linguagem de comunicação, como por exemplo, o Bash.

O Bash lê comandos, traduz para uma forma que o núcleo do computador entenda e por fim, executa as tarefas repassadas a ele. O Bash foi desenvolvido e disponibilizado em 1989 pela *Free Software Foundation* e é o interpretador padrão utilizado nos sitemas GNU/Linux (STALLMAN, 2000, p.7). O CD-MOJ é desenvolvido na linguagem de programação interpretada Bash.

O CD-MOJ está hospedado em uma plataforma de código-fonte com controle de versão, denominada (GIT). Plataformas com controle de versão permitem ao dono de um projeto desfazer alterações com facilidade fazendo com que erros sejam facilmente revertidos, também permitem que vários usuários tenham acesso aos códigos-fonte e façam as suas alterações sem que afetem o projeto principal, podendo estes enviar as suas modificações ou correções de *bugs* ao proprietário do projeto que poderá escolher se deve ou não acatar a estas modificações (MASON, 2006, p.11-12).

Por se tratar de um sistema *web* o CD-MOJ precisa de um servidor *web*. O servidor utilizado na presente pesquisa é o Apache. Este é um servidor *open-source*, ou seja, é um sistema de código aberto. O Apache é o responsável por receber e aceitar as solicitações dos usuários e então retornar as informações em forma de arquivos e paginas *Web*.

O CD-MOJ

O CD-MOJ é um ambiente de correção automatizada, ou seja, para cada problema proposto pelo professor o aluno pode submeter uma solução e então, depois de uma rigorosa correção, o sistema retorna uma resposta a esse aluno dizendo se a sua submissão foi ou não aceita e, se for o caso, demonstrar ao mesmo que tipo de erro foi encontrados em seu código. O CD-MOJ é um sistema *stateless*, isto é, cada resposta submetida por cada aluno é tratada de forma independente.

Para que o CD-MOJ funcione corretamente, o servidor web precisa estar corretamente configurado para suportar os *scripts* desenvolvidos no Bash. Somente após a conclusão desta, o sistema poderá ser instalado.

Depois de instalado, dois *daemons* devem ser executados na máquina pra que consigam manter o contato com o servidor *web*. São estes dois *daemons* (executar-julgador. sh e executar-corretor. sh) que farão a maior parte das comunicações com o servidor. Eles têm a função de verificar as requisições de login, publicar novos *contests*, enviar as submissões feitas pelos alunos e informar ao aluno se a sua solução para o problema proposto está correta ou se alterações devem ser feitas no código.

Quando uma nova submissão acontece, o *script* julgador.sh é invocado pelo seu *daemon* (executa-julgador.sh) para que ele possa então redirecionar a resposta enviada a um diretório específico das novas submissões. Outro *daemon* chama o *script* corrige.sh, que tem a função de pegar a resposta presente no diretório citado anteriormente e enviar pra o *online judge* escolhido pelo professor e aguardar o resultado da submissão. Depois de receber a resposta do servidor do site que avaliou a solução do aluno, o *script* cria um arquivo na pasta de submissões para que o julgador.sh possa ler e retornar ao aluno se sua submissão é válida ou não.

As modificações realizadas no CD-MOJ têm o intuito de ampliar as funcionalidades do sistema e facilitar a forma com que problemas são sugeridos pelos professores. No entanto, essas alterações devem ser feitas sem que o sistema sofra mudanças significativas, ou seja, serão pequenas mudanças que farão de fato muita diferença para o usuário.

AS MODIFICAÇÕES

O projeto está dividido em 2 etapas, na primeira, foram realizados todos os levantamentos necessários com os professores que utilizam a plataforma para que as primeiras modificações necessárias sobre os *scripts* originais fossem executadas. Em seguida, as novas funcionalidades foram implementadas.

A primeira versão do sistema necessitava que o professor criasse um arquivo de texto chamado *contest-description.txt* seguindo rigorosamente a instruções descritas no *site* para que o arquivo fosse criado corretamente e pudesse então ser enviado para o sistema poder exibir o novo contest na página inicial do ambiente de programação. Na figura 1 observam-se as exigências que devem ser seguidas pelo usuário para que a elaboração desse arquivo fosse feita de forma correta.

Figura 1 – Modo antigo de adicionar novos *contests*.

CONTEST_ID	ID do contest em um nome padrao UNIX, sem espaco
"Nome Completo do Contest"	Nome completo do contest para aparecer na tela inicial, deve ser escrito com ASPAS
INICIO-da-prova	Inicio da prova em segundos, no padrão UNIX gere com o comando date, por exemplo, se a prova deve começar às 15:00 de hoje: `date --date="15:00:00 today" +%s`
TERMINO-da-prova	Termino da prova em segundos, gere com o comando date, por exemplo, se a prova deve terminar hoje às 17:00: `date --date="17:00:00 today" +%s`
N	Numero inteiro com a quantidade de problemas da prova, depois serão N linhas com as descrições dos problemas
SITE ID "Nome Completo" Nome_Pequeno link-enunciado	N linhas com esse formato, onde cada elemento representa: SITE: pode ser spoj-br spoj-www ID: é o ID do problema no SITE "Nome completo": nome full do problema, entre ASPAS Nome_pequeno: pode ser Letra ou numero mas coloque em ordem nesse arquivo link-enunciado: pode ser: site , redireciona pro SITE um link inicando por http:// none para nao ter um enunciado o nome de um arquivo dentro do diretorio enunciados/
M	Número Inteiro representando a quantidade de usuários cadastrados
login:senha:Nome Completo	M linhas com os usuarios que tem permissão de logar nesse contest. Todo login que terminar com .admin será considerado um usuário do tipo Administrador e terá acesso a várias funcionalidades administrativas, como acesso contínuo às Estatísticas mesmo sem ter PARTIALSTATISTIC=1.

Fonte: Contest-Driven Meta Online Judge (2018).

Verifica-se na Figura 1 que nas cinco primeiras linhas épossível identificar que o CD-MOJ utiliza para distinguir os *contests,* o nome do *contest* que deverá aparecer na tela inicial, o tempo de início e de fim em segundos e o número total de problemas.

Nas linhas seguintes vem o conjunto de informações que cada problema possui. Primeiro, o *site* que julgará o problema, a identificação deste, o título, uma letra de A Z para que o CD-MOJ possa organizá-los e por fim, o enunciado. Existem duas diferentes formas de anexar o enunciado do problema, uma é colocando-o em um diretório com o nome de enunciados e o outro é adicionando um *link* pra que o CD-MOJ consiga realizar o *download* do enunciado.

Após adicionar cada um dos problemas em suas respectivas linhas o arquivo segue contendo na linha seguinte o número total de usuários e por fim, uma lista de todos os alunos que podem acessar o contest com seu login, senha e nome completo.

Após concluir esta etapa de construção do arquivo o mesmo deve ser compactado com o diretório dos enunciados para que então possa ser enviado ao sistema para que o mesmo consiga disponibilizar o *contest* criado na página inicial do CD-MOJ. Na Figura 2 podemos ver qual deve ser o resultado final do arquivo *contest-description.txt*.

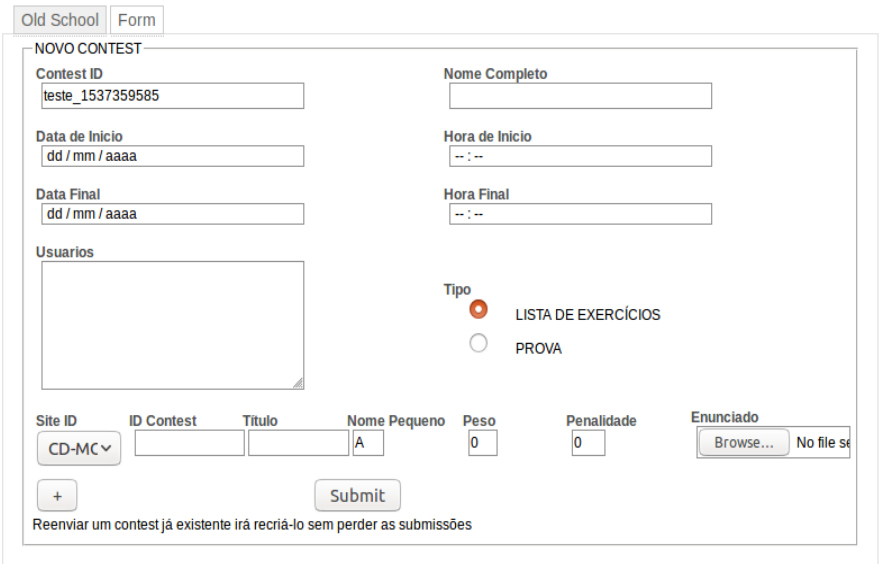
Figura 2 – Exemplo de formato arquivo *contest-description.txt*.

```
teste
"Prova de Teste do Sistema"
1372801500
1373492700
3
spoj-br PLACAR "Quem vai ser reprovado?" A reprovado.txt
cdmoj cartasfora "Jogando Cartas Fora" B jogando-cartas-fora.pdf
spoj-www TEST "Life, the Universe, and Everything" C site
5
ribas.admin:senhasecretaADMIN:Bruno Ribas Aministrador
ribas:fiesta:Buno Ribas
ricardo:busao:Ricardo Oliveira
vinicius:hyundai:Vinicius Ruoso
willian:gol:Willian Zalewski
```

Fonte: Contest-Driven Meta Online Judge (2018).

Na implementação proposta foi inserido ao sistema um formulário (Figura 3) onde cabe ao professor apenas preencher os campos exibidos. Algumas variáveis foram adicionas em relação ao estilo anterior, como o modo prova, o peso sobre cada problema e a penalidade sobre cada submissão errada. O peso sobre cada problema mostra ao aluno qual é aquele que exigirá mais das suas capacidades de programar. A penalidade faz com que o aluno analise cuidadosamente o código que deseja enviar e utilize seus próprios casos de teste.

Figura 3 – Formulário para inclusão de novos *contests*.



Fonte: Contest-Driven Meta Online Judge (2018).

RESULTADOS E DISCUSSÕES

Para que a atualização no padrão de adição dos novos *contests* pudesse acontecer, alguns *scripts* novos tiveram que ser adicionados ao CD-MOJ, como o *new-contest-form.html, formulario.sh, new-contest.sh.* Estes 3 arquivos são os

responsáveis por criar o formulário, enviar os dados obtidos nas caixas de texto, processar os dados recebidos, criar o arquivo *contest-description.txt* e compactá-lo junto com os enunciados em um único arquivo.

O primeiro *script* a ser elaborado foi o *new-contest-form.html*. É este o responsável por exibir o formulário e coletar o conteúdo das caixas de texto. Quando um usuário encaminha os dados para o servidor ele gera um arquivo que contém este conteúdo e envia para o *script* que de fato gera o *contest-description.txt* e compacta junto com os enunciados.

Quando o usuário envia todas as informações coletadas das caixas de texto, o navegador gera um arquivo temporário com tudo o que foi digitado no formulário. Porém esse arquivo contém informações adicionais que precisam ser filtradas e, em alguns casos, devem ser editadas. Então, a tarefa do *script* formulario.sh é muito simples, é dele o trabalho de filtrar e editar essas informações.

Depois de coletar todas as informações necessárias o formulario.sh cria o arquivo *contest-description.txt,* a pasta de enunciados, compacta os dois e finalmente envia para o diretório *submissions* para que o *contest* seja publicado na página inicial.

O CD-MOJ ainda aceita que novos *contests* sejam adicionados a partir do modo antigo. O *new-contest.sh*, é quem identifica qual é a forma que o usuário deseja inserir o novo *contest* e então redirecionar o mesmo para a seção do modo antigo ou formulário.

CONSIDERAÇÕES FINAIS

O novo método de inserir *contests* foi à elaboração de um formulário que contém todas as exigências do modo antigo. Algumas variáveis foram adicionadas ao CD-MOJ e agora o sistema conta com um método de peso e penalidade por problema adicionado.

Alguns arquivos tiveram que ser criados para que as funcionalidades do formulário fossem possíveis. O sistema possui um novo arquivo que cria o formulário, coleta o conteúdo das caixas de texto e envia para o *script* que filtra as informações necessárias, faz os ajustes nos dados coletados e grava as informações no arquivo *contest-description.txt* e o envia juntamente com os enunciados para o diretório responsável por receber os novos *contests*.

Apesar de receber as atualizações, o CD-MOJ ainda possui o formato inicial de submissão de problemas. Dando ao usuário a oportunidade de escolher qual é a forma que mais lhe agrada quando desejar criar novos *contests*.

O trabalho conta com alguns projetos futuros para que a plataforma possa auxiliar a universidade em eventos que visam selecionar alunos para participar da Maratona de Programação. A inclusão de uma nova funcionalidade, o Modo Progressivo, onde seguirá alguns padrões do novo Modo Prova, como o peso e a penalidade por problema. Porém, com uma nova forma de aplicar as penalidades. Para que os alunos sejam incentivados a encontrar soluções de forma rápida, será implementada, uma forma adicional de penalidade que levará em conta o tempo de prova. Um limite de submissões por problema também

será aplicado, aumentando a dificuldade aos competidores e obrigando-os a aumentarem seus conhecimentos sobre a linguagem de programação utilizada.

AGRADECIMENTOS

O presente trabalho foi realizado com o auxilio da Universidade Tecnológica Federal do Paraná - UTFPR, pois sem esse apoio não seria possível à realização de tal pesquisa.

REFERÊNCIAS

CD-MOJ. Disponível em:<https://moj.naquadah.com.br>. Acesso em: 20 ago.2018.

DE CAMPOS, C. P.; FERREIRA, C. E. . BOCA: um sistema de apoio a competições de programação (BOCA: A Support System for Programming Contests). Disponível em: <https://www.ime.usp.br/~cassio/boca/campos-ferreira-wei2004.pdf>. Acesso em: 24 ago.2018.

ESTIMA, Pedro Miguel Oliveira. **Plataforma de apoio à aprendizagem de linguagens de programação**. 2013. Dissertação de Mestrado. Universidade de Aveiro. Disponível em: <https://ria.ua.pt/bitstream/10773/11778/1/tese.pdf>. Acesso em: 29 ago.2018.

MASON, Mike. **Pragmatic Version Control Using Subversion**. The Pragmatic Programmers LLC, 2006. p.11-12. Disponível em: <http://117.3.71.125:8080/dspace/viewer/web/viewer.html?file=http://117.3.71.125:8080/dspace/bitstream/DHKTDN/6736/1/6141.Pragmatic%20version%20control%20using%20subversion%20%282nd%20ed%29.pdf>. Acesso em 30 ago.2018.

SOCIEDADE BRASILEIRA DE COMPUTAÇÃO. **XXIII Maratona de Programação**. 2018. Disponível em: <http://maratona.ime.usp.br>. Acesso em: 21 set.2018.

STALLMAN, Richard. O projeto GNU. **Data GramaZeroZero—Revista de Ciência da Informação**, n. 1, 2000. Disponível em: <http://www.brapci.inf.br/_repositorio/2010/01/pdf_5ceeeb7f7e_0007418.pdf>. Acesso em: 28 ago.2018.