Escalonador de Atividades usando Algoritmo de Subida na Montanha

Carla Nobrega, Clarissa Lippi, Eduardo Machado, Hans Hudson, Robson Luiz, Enzo Tiezzi, Igor Moreira, Newton Delbuque, Marco Aurélio, Flávio Augusto, Thiago Souza, Caio Zanelato

Orientador: Fabricio Barth

Faculdade de Tecnologia Bandeirantes – BandTec
Rua Estela, 268 – 04011-001 – São Paulo – SP – Brasil

1. Introdução

A faculdade BandTec oferece em sua grade curricular um apoio psicopedagógico com o objetivo de suprir as necessidades de seus alunos além do âmbito técnico. O Programa H, como é chamado, auxilia os alunos a desenvolver qualidades humanas como liderança, trabalho em grupo, autoconhecimento, entre outros quesitos que, assim como o conhecimento técnico, são indispensáveis a um profissional de destaque.

Uma das formas de orientar e guiar os alunos em seu desenvolvimento é através de alguns treinamentos organizados pela equipe do programa H, estes treinamentos tem o objetivo de auxiliar os alunos a melhorar alguma competência não tão desenvolvida.

Dado o grande número de alunos e a quantidade de treinamento que o programa H pode executar por semana, a organização da agenda de treinamento dispende muito tempo para ser elaborada e sua adaptação a mudanças repentinas de disponibilidade acaba tornando-se lenta e trabalhosa.

Sendo assim, o objetivo deste trabalho é desenvolver um escalonador que retorna uma agenda de treinamento maximizando a quantidade de alunos atendidos, considerando a quantidade de treinamentos que o programa H precisa executar em uma semana e a disponibilidade de todos os alunos que precisam receber este treinamento.

Uma premissa adotada para o desenvolvimento deste trabalho é a de que apenas um tipo de treinamento é realizado por semana, mas que ele pode ser executado de 1 até 10 vezes por semana.

São Paulo, 06 de outubro de 2015.

2. Solução Proposta

Durante o desenvolvimento deste trabalho foi implementado uma função de escalonamento que recebe como parâmetros:

- a quantidade de treinamentos que devem ser ministrados naquela semana. Este valor pode variar de 1 até 10, e;
- a disponibilidade na agenda de cada aluno que precisa receber o treinamento. A disponibilidade de um aluno é representada por um vetor de tamanho 10 com valores lógicos. A posição 1 do vetor significa se o aluno está disponível nas duas primeiras aulas da segunda-feira, a posição 2 do vetor representa se o aluno está disponível nas duas últimas aulas da segunda-feira, a posição 3 representa a situação das duas primeiras aulas da terça-feira e assim sucessivamente até às duas últimas aulas da sexta-feira. Sempre que o valor for TRUE neste vetor significa que o aluno tem disponibilidade para o treinamento naquele horário. O segundo parâmetro da função de escalonamento é uma lista de vetores que significa a disponibilidade de todos os alunos que precisam receber o treinamento.

Algumas premissas foram adotadas para o desenvolvimento deste trabalho:

- apenas um tipo de treinamento será ministrado por semana. Poderão ser vários treinamentos, entre 1 e 10, mas sempre serão sobre o mesmo assunto. Isto simplifica a escolha da lista de alunos que será utilizada como segundo parâmetro da função, e;
- todos os treinamentos tem duração de duas horas-aula. Isto também simplifica a representação da disponibilidade dos alunos. É por este motivo que a disponibilidade de cada aluno pode ser representada por um vetor de valores lógicos de tamanho dez, 2 valores para cada dia.

Para o desenvolvimento deste trabalho foi escolhido um algoritmo do tipo subida na montanha. Algoritmos do tipo subida na montanha fazem uso de uma função de otimização para escolher sempre o estado que otimiza o objetivo da implementação. É uma técnica que usa um laço repetitivo que se move de forma contínua no sentido do valor crescente da função de otimização, ou seja, encosta acima, até alcançar um pico. Não mantém uma árvore de busca, e assim a estrutura de dados do nó atual só precisa registrar o estado e o valor de sua função de otimização. A técnica não examina antecipadamente valores de estados além dos vizinhos imediatos do estado corrente [1].

Para evitar problemas de máximos locais, o algoritmo implementado neste trabalho faz uso de um percurso aleatório [1] quando encontra máximos locais. A função de otimização implementada neste trabalho conta o número de alunos atendidos em cada estado. A geração de sucessores acontece de forma aleatória. Para garantir que a solução encontra a solução ótima, todos os estados considerados como máximos locais são armazenados e comparados entre si.

3. Validação e Resultados

Para verificar se a aplicação está realmente atendendo a demanda dos objetivos, foram realizados testes no sistema. Na tabela 1 são apresentados alguns dos cenários de casos de testes utilizados para validação do projeto. Além disso, foram realizados testes funcionais que atenderam os seguintes requisitos:

- Verificar se o sistema respeita aos critérios de apenas 4 treinamentos por semana;
- Verificar se treinamento é agendado apenas quando a quantidade máxima de alunos é escalonada, ou seja, quando o resultado é ótimo;
- Verificar se a quantidade mínima de alunos para por treinamento é respeitada;
- Verificar se o sistema respeita os treinamentos disponíveis versus horários livres de alunos para treinamento, juntamente com a quantidade mínima e máxima de alunos por treinamento;

Para garantir que a solução não tenha nenhum problema referente a máximos locais, todos os casos de teste foram executados 20 vezes e sempre retornaram o resultado ótimo.

Foi realizada uma sequência de 20 cenários de testes onde cada teste foi executado 20 vezes, totalizando 400 execuções. Onde o tempo médio para resposta foi de 1,44 milissegundos e o desvio padrão foi de 1,33 milissegundos.

Alinhado com os objetivos deste projeto, os resultados esperados para o sucesso do mesmo é que o sistema combine (realize um match) entre os horários e datas disponíveis dos alunos que precisarão realizar um curso de capacitação, e seu instrutor. O treinamento é agendado apenas se houver um dia na semana na qual a maior parte dos alunos cadastrados tenham disponibilidade. No entanto, a faculdade pode realizar quatro treinamentos por semana. A definição dos treinamentos é feita nos quatro dias que mais possui alunos disponíveis.

4. Considerações Finais

O objetivo deste trabalho foi propor um sistema de escalonamento levando-se em consideração restrições de agenda e da equipe de treinamento. Para alcançar este objetivo foi utilizado um algoritmo do tipo subida na montanha. Os resultados encontrados comprovam que esta solução é factível e pode ser implantada em ambiente de produção. Todos os cenários de teste validados retornaram configurações ótimas e o tempo de processamento é em media 1,5 milissegundos, o que comprova que o algoritmo pode ser usado para a solução proposta.

O programa foi testado para um total de 20 alunos com necessidade de realizar um treinamento de reforço. Levando – se em consideração uma universidade inteira, sendo que não serão todos com necessidade de procura e disponibilidade de horário, este programa ainda será útil e funcional para que seus objetivos sejam atingidos.

São Paulo, 06 de outubro de 2015.

Quanto às pessoas que utilizaram o programa, a principal restrição apresentada é em relação a alocação do mesmo, pois, é necessário que a máquina possua o programa instalado. Não é um sistema desenvolvido para ser acessado de várias plataformas, via web.

Além disso, sua interface não é amigável, dificultando o uso do mesmo. Deve haver treinamentos e pessoas capacitadas a utilizá-lo.

Para esse modelo não foram testados todos os estados possíveis, pois deixaria a busca lenta. Por esse motivo, ao encontrar o resultado 80% ótimo, ele será considerado como máximo global, ou seja, será considerado o resultado ótimo.

5. Referências bibliográficas

[1] NORVIG, PETER,RUSSELL, STUART. **Inteligência Artificial**. 3° ed. Brasil: Elsevier Brasil, 2014.

São Paulo, 06 de outubro de 2015.

Tabela 1: Resultados de testes

Entrada	Instruções de teste	Resultado esperado	Resultado obtido	Ponto de controle	Ponto de observação
Todos os alunos disponíveis em todas as aulas	Selecionar o teste 1, com todos os valores "true"	Definir um treinamento invalido ou não definir treinamento.	sem treinamento definido.	Não Há	No mundo real não havera cadastro de aluno sem aulas
Todos os alunos com aula apenas segunda, no segundo horario	Selecionar o teste 2, com essa caracteristica	Apresentar datas de treinamento na segunda feira do segundo horario	sexta primerio horario, 20 alunos atingidos, execução em 2ms	não há	Gera resultado como qualquer um dos dias possiveis
10 alunos com apenas terça primeiro horario em aula e 10 com apenas sexta ultimo horario em aula.	Selecionar o teste 3, com essa caracteristica	Apresentar datas de treinamento em qualquer dia, exceto terça primeiro horario ou sexta segundo horario	quarta segundo horario, 20 alunos afetados, 2ms	Não há	Realizar alguns testes e ver se não apresenta dia invalido
Todos os alunos em aula, exceto segunda nos dois horarios	Selecionar o teste 4, com essa caracteristica	Qualquer dia, exceto segunda nos dois horarios, com treinamento para todos os alunos	quinta primeiro horario, 20 alunos afetados, 2 ms	Não há	Validar se difere de 20 alunos ou apresenta treinamento na segunda.
Todos alunos livres apenas segunda e quarta no primeiro horario	Selecionar o teste 5, com essa caracteristica	Qualquer dia, exceto segunda e quarta no primeiro horario, para todos os 20 alunos.	quarta segundo horario, 20 alunos afetados, 6 ms	Não há	Definição de dias para aplicar o treinamento é randomico, pode apresentar qualquer um dos dias
Alunos 1 a 7 em aula apenas terça no primeiro horario, 8 a 13 em aula apenas segunda no primeiro horario, 9 a 17 em aula apenas quinta no primeiro horario e 18 a 20 apenas sexta primeiro horario	Selecionar o teste 6, com essa caracteristica	Recomendar o dia para treinamento como quarta primeiro horario ou quanquer dia no segundo horario	terça segundo horario, 20 alunos afetados, 1ms	Não há	Validar se não apresenta resultado diferente do proposto

São Paulo, 06 de outubro de 2015.