Elaboração de horários de disciplinas de cursos de graduação usando Algoritmos Genéticos

Abias de Santana, Marcus Carvalho

[1]Departamento Ciências Exatas (DCX) - Universidade Federal da Paraíba (UFPB)
Rua da Mangueira, s/n, Companhia de Tecidos Rio Tinto
CEP 58297-000 - Rio Tinto - PB - Brasil

{abias.santana,marcuswac}@dcx.ufpb.br

1. Resumo

Atualmente a elaboração de quadros de horários na UFPB é um processo manual e custoso, que exige a participação tanto dos professores quanto das chefias de departamento com diversas reuniões para chegarem a uma solução razoavelmente satisfatória, o que acaba por exigir muito tempo e recursos humanos.

O trabalho tem como objetivo auxiliar na elaboração e análise de quadros de horários de instituições de ensino em geral, para a elaboração e testes da ferramenta usamos como base os cursos de Sistemas de informação e de Licenciatura em ciência da computação da UFPB campus IV, no entanto, a solução foi pensada para ser aplicada em qualquer instituição de ensino.

Este trabalho apresenta a implementação de uma ferramenta de análise de quadro de horários e de uma ferramenta de geração de quadros de horários otimizados com uso de algoritmos genéticos. As duas soluções tem como base um esquema de acoplamento de funções que avaliam diferentes características de um quadro de horários, a ferramenta de análise gera um relatório do individuo com que tipo de problemas o quadro de horários apresenta e onde, com base nas funções pré-estabelecida, o gerador de quadro de horários usa estas funções para gerar uma solução com uso de algoritmos genéticos.

1

2. Introdução

Elaborar os horários de disciplinas em cursos de graduação é uma tarefa complexa e de grande impacto tanto no processo de aprendizagem dos discentes quanto na eficiência das atividades exercidas pelos docentes, visto que um horário mal elaborado pode acarretar em um mau uso do tempo durante o período de aulas. Alguns problemas causados por horários mal elaborados são: (1) aulas muito fragmentadas em um mesmo dia, com buracos entre as aulas, pode gerar tempo ocioso dos alunos e dos professores entre as aulas; (2) uma sequência grande de aulas consecutivas de um mesmo tema ou do mesmo professor em uma turma seria cansativo para alunos e professores, afetando o processo de aprendizagem; (3) dias com apenas 1 disciplina ou turma pode trazer um aumento nos custos para alunos e professores, que às vezes moram longe do campus e teriam que se deslocar mais

[1]Trabalho de conclusão de curso, sob orientação do professor Marcus Carvalho submetido ao Curso de Bacharelado em Sistemas de Informação do Centro de Ciências Aplicadas e Educação (CCAE) da Universidade Federal da Paraíba, como parte dos requisitos necessários para obtenção do grau de BACHAREL EM SISTEMAS DE INFORMAÇÃO.

vezes ao campus para atividades isoladas em alguns dias. Outros de importância crítica para o funcionamento do quadro de horários, que podem tornar um quadro de horários inviável logo em um primeiro momento, como professores alocados para duas aulas no mesmo horário ou colisão de horários de dois professores para mesma turma.

Atualmente nos cursos de Bacharelado em Sistemas de Informação (BSI) e Licenciatura em Ciência da Computação (LCC) da Universidade Federal da Paraíba (UFPB), campus IV, os horários são elaborados de maneira manual em um trabalho conjunto das coordenações dos dois cursos e da chefia de departamento, o que gera uma quantidade considerável de carga de trabalho a estes professores e, mesmo com esta mobilização em sua confecção, por muitas vezes acaba resultando em um horário pouco satisfatório para algumas das partes: professores e alunos.

Considerando a complexidade e importância desta tarefa, este trabalho se propõe a auxiliar o processo de elaboração de horários de disciplinas de cursos de graduação. O objetivo geral do trabalho é criar uma ferramenta que avalia a viabilidade e eficiência de soluções de horários e automatiza a busca pelo horário mais adequado de acordo com critérios levantados por professores e alunos, com o uso de algoritmos genéticos. Os objetivos específicos deste trabalho são:

- Estabelecer critérios para seleção de melhor horário, com uma função de custo e pesos para cada critério;
- Criar uma ferramenta que avalie uma solução, indicando o custo de uma solução com base nos critérios definidos e permitindo a comparação de soluções. A ferramenta deve ser flexível para permitir a fácil adição, remoção e configuração de critérios.
- Criar um método para buscar a melhor solução de forma automática, visando minimizar a função de custo considerando os critérios estabelecidos e usando uma abordagem de algoritmos genéticos.

O restante do artigo está estruturado da seguinte forma: a seção 3 apresenta a fundamentação teórica; a seção 4 discute os trabalhos relacionados; a seção 5 descreve a ferramenta proposta neste trabalho; a seção 6 mostra como a avaliação da ferramenta foi conduzida em um cenário real; e, finalmente, a seção 7 apresenta as conclusões do trabalho.

3. Fundamentação teórica

Algoritmos genéticos são um tipo de algoritmo evolutivo que tem como base a lógica da seleção natural das espécies proposta por Charles Darwin (Lacerda and Carvalho 1999). Este tipo de algoritmo faz uso dos critérios como hereditariedade, mutação, seleção natural e recombinação da seleção natural.

A Figura 1 mostra um processo típico de algoritmo genético. O primeiro passo consiste em definir uma população inicial, composta por um grupo de indivíduos que carregam uma série de genes , também conhecido como genoma. Um indivíduo nada mais é do que uma possível solução; uma população seria o conjunto destes indivíduos; um gene seria uma das características que este indivíduo carrega e o genoma seria o conjunto destes genes. A população inicial geralmente é formada por uma série de indivíduos com características geradas aleatoriamente.

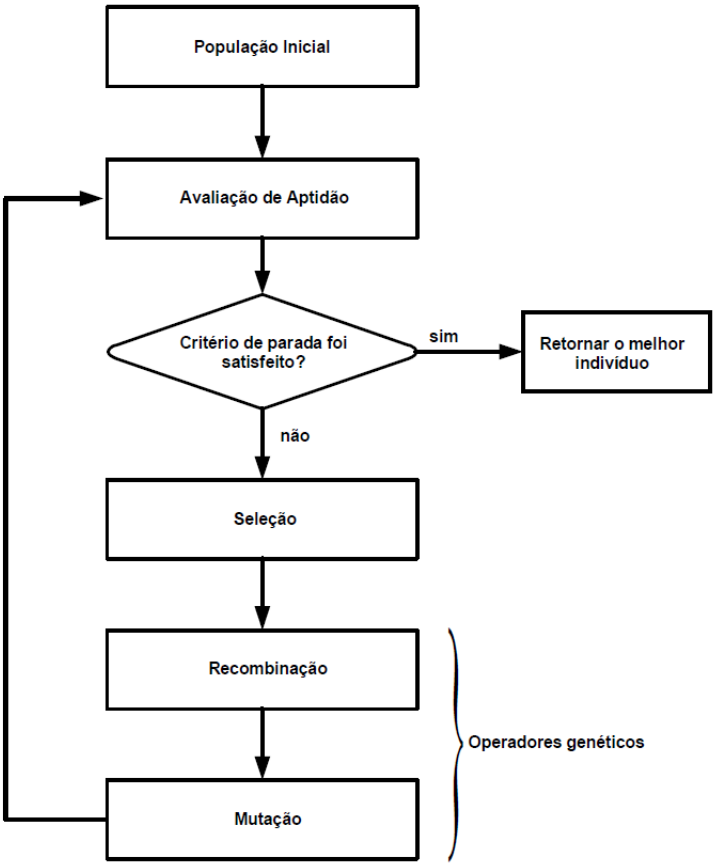


Figura 1. Exemplo gráfico do fluxo do algoritmo genético. Fonte: (Lacerda and Carvalho 1999)

Após a criação da população inicial e após a criação de cada nova população, os indivíduos destas populações são submetidos a uma função de aptidão, que avalia o seu desempenho de acordo com as características desejadas, conhecida como função de *fitness*. A função de *fitness* usa para a avaliação uma lógica de bonificação para os que se adequam melhor aos requisitos estabelecidos na função ou punição aos que não possuam características desejáveis. Este sistema de penalização e bonificação pode utilizar um sistema de pesos, que possibilita à função saber quais os critérios prioritários e quais os de menor prioridade. Este sistema de critérios pode ser divididos em dois tipos: *hard constraints*, que são restrições de extrema importância que tornam uma solução inviável caso não sejam cumpridas por um indivíduo; e *soft constraints*, que são restrições de menor impacto na qualidade da solução, mas que também devem ser contabilizados durante a avaliação e seleção do melhor indivíduo.

Após a aplicação da função de *fitness*, os indivíduos mais bem avaliados serão elegidos para a criação dos novos indivíduos, onde este novo indivíduo possui uma mistura das características de seus pais; este processo é chamado de recombinação ou cruzamento (*crossover*). Feito o processo de cruzamento, o novo indivíduo recebe uma mudança em um ou mais dos seus genes a fim de prover uma maior variabilidade genética na população; este processo de alteração dos genes é conhecido como mutação (*mutation*).

O processo de aplicação da função de *fitness*, *crossover* e *mutation* irá se repetir

até que esta série de novos indivíduos se acumule em uma quantidade suficiente para a formação de uma nova população. A cada vez que a função fitness é aplicada é verificado se as condições de parada foram atendidas, caso não, os processos de *crossover*, *mutation* e novamente de *fitness* se repetirão e após a aplicação do fitness a nova população a condição de parada será verificada novamente, este processo se repete até que a condição de parada seja atendida, caso esta condição seja atendida, o melhor ou os N melhores indivíduos daquela população seriam retornados como as melhores soluções.

4. Trabalhos relacionados

Esta seção discute trabalhos que são relacionados com o problema proposto no presente artigo, dos quais foram avaliadas as ferramentas usadas, métodos, objetivos, contextos e resultados obtidos.

O trabalho de (Lobo 2005) propôs uma solução do problema de horário escolar via algoritmo genético paralelo. Foi aplicado na Universidade Presidente Antônio Carlos – UNIPAC – em Conselheiro Lafaiete-MG. Fez uso de Java e Corba para implementar a heurística de otimização na construção de quadro de horários. Utilizou um sistema de "ilhas" a fim de aplicar computação paralela ao algoritmo. Conseguiu atingir 92,3% da solução ótima. Também fez uma comparação com e sem o uso de paralelismo, onde conseguiu atingir o resultado em 78 segundos com o uso do paralelismo, e em 140 segundos sem o paralelismo, um ganho de 62 segundos.

O trabalho de (Ciscon et al. 2005) teve como objetivo a eliminação de janelas e aulas isoladas no horário dos professores. Foi elaborado um algoritmo utilizando um sistema de penalização, construído utilizando a linguagem Java. Não fez diretamente nenhuma diferenciação entre hard requisites e soft requisites, e não utilizou uma solução ótima para avaliação do desempenho do algoritmo.

(Vieira and Macedo 2011) propuseram um sistema de alocação de horários de cursos universitários aplicado no Departamento de Computação da Universidade Federal de Sergipe. Além de utilizar requisitos normalmente utilizados neste tipo de solução (ex: disciplina atende horário do curso, professor dá aula em dois lugares ao mesmo tempo, etc), também levou em conta o interesse dos professores em cada disciplina. No entanto, a ferramenta foi idealizada unicamente para o Departamento de Computação da Universidade Federal de Sergipe, sem considerar uma flexibilização da ferramenta para outros departamentos e para outros critérios.

O trabalho de (Freitas et al. 2007) apresentou a ferramenta Kairós, baseada em algoritmos genéticos para a geração de tabela de horário escolar, que foi criada para satisfazer os interesses da Faculdade Ruy Barbosa (FRB). As grades de horários obtida a partir desta ferramenta atingiu uma média entre 1 a 3 conflitos de horários e uma média de 2 a 5 horários em branco em cada bateria de testes, ao todo foram 5 bateria de testes.

5. Uma ferramenta extensível para a definição de horários baseada em algoritmos genéticos

A solução proposta neste trabalho foi feita com a modelagem e implementação de um algoritmo genético. O diferencial desta ferramenta em relação aos trabalhos relacionados é: ter seu código aberto, implementar restrições ainda não vistas em outras ferramentas

e ser extensível para criar diferentes restrições e funções de aptidão para se adequar à realidade de outros cursos.

A linguagem Python foi escolhida para a construção da ferramenta, devido à facilidade em gerar estruturas de dados capazes de representar as entidades necessárias, além da existência de bibliotecas que ajudam na manipulação dos dados e implementação da solução. As diferentes etapas do algoritmo genético foram elaboradas conforme descritas nas subseções a seguir, seguindo o processo apresentado anteriormente na Figura 1.

5.1. Representação da solução

Para modelar as entidades do sistema, optou-se pelo uso de um dicionário para representação do indivíduo (solução), do cromossomo (conjunto de horários definidos para cada aula) e da população (conjunto de indivíduos em uma geração). As gerações de indivíduos foram mapeadas em um dicionário, onde a chave corresponde ao número sequencial da geração e o valor corresponde a um outro dicionário que representa a população naquela geração. Para a população de cada geração, existe um dicionário que representa o conjunto de indivíduos daquela população. Cada indivíduo é representado por um cromossomo, implementado também com dicionários. No dicionário de mais alto nível, cada item é uma disciplina, onde a chave é o código da disciplina. Este dicionário de disciplinas é mapeado para a lista de aulas daquela disciplina. Cada aula tem a representação do gene do cromossomo (o horário definido para aquela aula, naquele indivíduo), além de informações adicionais da disciplina, curso, período e professor. Além do valor do gene, que corresponde ao horário alocado para cada aula, cada aula também traz informações adicionais como o nome da disciplina, código, curso e período, para ajudar no cálculo da função *fitness*.

A Figura 2 apresenta a estrutura das entidades modeladas, com vários níveis de dicionários e as relações entre eles. A Figura 3 mostra um exemplo de indivíduos em uma população, onde cada indivíduo apresenta informações sobre: sala, período, professor e horário alocado (gene).

5.1.1. Representação do cromossomo

O cromossomo é composto por um dicionário que possui como chave o código da disciplina e o valor é a lista de aulas daquela disciplina. Cada aula representa um gene do cromossomo, que armazena cinco características referentes a aula. Exemplo de cromossomo e genes: 42: [[’A1’, ’p3’, ’João’, ’h3’, ’segunda’, ’Matemática’, ’si’],[’A2’, ’p5’, ’José’, ’h3’, ’segunda’, ’Inglês’, ’si’]], temos entre cada colchete a representação de um gene contendo sete informações referentes a aula. Tomemos como exemplo o primeiro gene, [’A1’, ’p3’, ’João’, ’h3’, ’segunda’, ’Matemática’, ’si’]: o primeiro valor separado por vírgula representa sala em que a aula se dará, o segundo valor representa o período em que esta disciplina está disponível, terceiro valor representa o professor responsável pela disciplina, o quarto valor representa o horário em que acontecerá a aula, o quinto valor representa o dia da aula, o sexto valor representa a disciplina que será lecionada na aula e, por fim, o sétimo valor que representa o curso para o qual a aula está disponível. Assim tomando como exemplo o primeiro gene [’A1’, ’p3’, ’João’, ’h3’, ’segunda’, Matemática’, ’si’] temos:

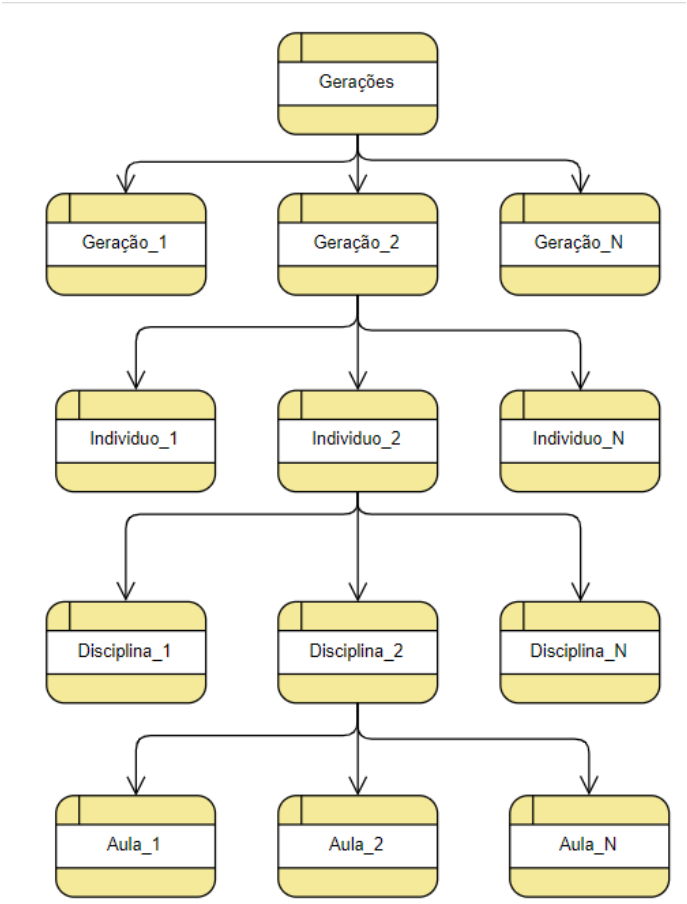


Figura 2. Estrutura das entidades representadas pelo algoritmo genético.

- *Sala*: A1
- *Período*: p3
- *Professor*: João
- *Horário*: h3
- *Dia*: segunda
- *Disciplina*: Matemática
- *Curso*: SI

O horário no qual cada aula será alocada é o atributo que o algoritmo genético vai modificar para buscar a melhor solução. Os outros atributos são fixos, passados como entrada, e são usados para computar restrições. Então, de forma simplista, um cromossomo é representado aulas, onde cada aula está em um índice (gene) e o seu valor representa o horário no qual a aula estará alocada. Os horários podem se repetir (mais de uma aula pode ter o mesmo horário), mas cada aula só terá um horário alocado em uma solução.

5.2. Entrada do algoritmo e população inicial

Para a criação da população inicial, a ferramenta recebe como entrada 3 arquivos no formato CSV:

- *disciplinas.csv*: cada linha representa uma disciplina ofertada. As colunas são: nome da disciplina, código, curso, período no curso, quantidade de aulas e professor que irá ministrá-la.

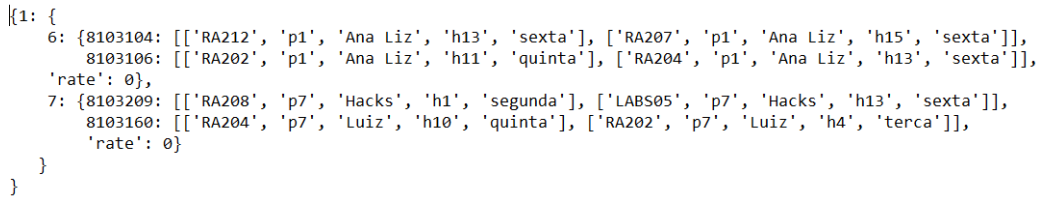


Figura 3. Exemplo da representação indivíduos em uma população.

- *horarios.csv*: cada linha representa um slot de horário disponível para a alocação das disciplinas. As colunas são: identificador do horário, dia da semana e hora de início

Com base nestes arquivos de entrada, o algoritmo monta a população inicial com um indivíduo da seguinte forma. Para cada uma das D disciplinas passadas no arquivo de entrada, verifica-se a sua quantidade x de aulas e então x genes do cromossomo (solução) são usados para indicar os horários nos quais as aulas desta disciplina ocorrerão. Portanto, o cromossomo será composto por $N = D * \bar{x}$ genes, onde cada gene representa o horário de uma aula de uma disciplina, sendo $\bar{x}$ a média da quantidade de aulas por disciplina. Os indivíduos da população inicial terão horários aleatórios alocados para cada aula, gerados através de uma distribuição uniforme com valores $h \in [1, H]$, onde H é a quantidade de horários disponíveis e h representa o índice do horário escolhido, dentre os horários disponíveis passados como entrada no arquivo de horários. Os valores gerados para h serão sorteados permitindo repetição. Teremos, então, indivíduos da população inicial compostos por N genes, sendo N a quantidade de aulas semanais das disciplinas definido durante a inicialização do algoritmo vezes a quantidade disciplinas, os quais possuem valores aleatórios de horários entre 1 e H.

5.3. Restrições e função de aptidão

Inicialmente foram discutidos alguns critérios que seriam relevantes tanto para um quadro de horários geral para cursos de graduação, quanto no contexto dos cursos relacionados a este trabalho. Algumas ideias de restrições foram obtidas a partir de uma reunião de professores do Departamento de Ciências Exatas (DCX) da UFPB para elaboração dos horários dos cursos de Bacharelado em Sistemas de Informação (BSI) e Licenciatura em Ciência da Computação (LCC).

Dentre opções de critérios levantados, foi implementado um subconjunto de restrições na ferramenta para permitir a sua aplicação e avaliação inicial. No entanto, a ferramenta foi desenvolvida de tal forma que as restrições podem ser facilmente estendidas, sendo facilmente plugadas novas funções que representam novos requisitos adequados a outros contextos. Cada função de restrição indica a quantidade de violações daquela restrição encontrada em uma solução (indivíduo).

O conjunto das R restrições consideradas e a quantidade de violações são utilizados para compor a função de *fitness*. Mais especificamente, o usuário que deseja executar o algoritmo passa uma lista de funções de restrição com seus respectivos pesos. A função de *fitness* calcula o somatório da quantidade de violações v_r de cada restrição r multiplicada pelo peso w_r associada ao custo (penalidade) daquela restrição. Quanto menor o valor da função de *fitness*, melhor será a solução. Portanto, busca-se escolher o indivíduo

que minimize o valor do *fitness*, indicando um menor custo ou penalidade daquela solução de horário. O *fitness* é calculado conforme equação a seguir:

$$F = \sum_{r=1}^{R} v_r * w_r$$

As restrições podem ser classificadas em rígidas (*hard*) e suaves (*soft*), de tal forma que se uma restrição rígida for violada a solução é considerada inviável. Na prática, restrições rígidas terão pesos muito maiores do que as suaves. Por padrão, será definido peso 1000 para restrições rígidas e peso 10 para restrições suaves. Apenas soluções com *fitness* menores que 1000 serão consideradas viáveis. Apesar dos valores padrão, diferentes valores de pesos para cada restrição e de limiares de solução viável podem ser definidos.

Nesta primeira versão, a ferramenta tem implementadas as seguintes funções de restrição:

- **R1:** disciplinas do mesmo curso e mesmo período não podem ter aulas no mesmo horário.
- **R2:** disciplinas ministradas pelo mesmo professor não podem ter aulas no mesmo horário.
- **R3:** aulas da mesma disciplina não podem ser no mesmo dia.

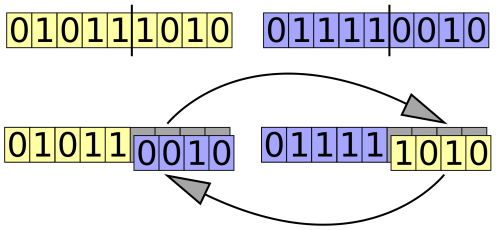
As restrições $R1$ e $R2$ são rígidas (peso 1000 para cada violação), enquanto a restrição $R3$ é suave (peso 10 para cada violação).

5.4. Seleção

Para a seleção de indivíduos, foi escolhido o sistema de *torneio*, onde é escolhido uma amostra de tamanho N de uma população a qual a função *fitness* já foi aplicada e selecionamos o indivíduo mais bem avaliado desta amostra, esta operação se repete até que seja atingido a quantidade de indivíduos igual ao tamanho da população, que é a quantidade necessária para a aplicação do *crossover*.

5.5. Crossover

Para o *crossover* foi selecionado o sistema de *recombinação em um ponto* onde é escolhido um ponto aleatório da sequência dos genes do cromossomo de dois indivíduos e uma das metades desse cromossomos trocam de posição entre os indivíduos, gerando dois novos indivíduos.



5.6. Mutação

Durante a mutação é escolhido um gene aleatório de cada indivíduo é gerado um novo gene com uma nova combinação aleatória de características. Após submetida a mutação a população resultante se torna a população atual.

6. Metodologia

Para os testes durante a implementação do algoritmo, foram utilizados os dados referentes às disciplinas e horários dos cursos graduação em Sistemas de Informação e de Licenciatura em Ciência da Computação da UFPB, Campus IV.

Os dados referentes às disciplinas foram carregados de um arquivo CSV que continham os dados das disciplinas como, código, nome, créditos, curso e professor. Após o carregamento é gerada a população inicial com o tamanho de 50 indivíduos com cromossomos com genes gerados aleatoriamente. Criada a população inicial é aplicada a função *fitness* à população, que observa os seguintes requisitos: (1) verifica se existem aulas de disciplinas do mesmo período no mesmo dia e horário caso encontrado aplica um peso de 1000 para cada quebra deste requisito no indivíduo, (2) verifica se existem aulas de disciplinas do mesmo professor no mesmo dia e horário, caso encontrado aplica um peso de 1000 para cada quebra deste requisito no indivíduo, (3) verifica se existem aulas da mesma disciplina no mesmo dia, caso encontre aplica um peso de 10 ao para cada quebra deste requisito no indivíduo, (4) verifica a existência de aulas consecutivas, caso encontrada aplica um peso de 10 para cada quebra deste requisito ao indivíduo.

Aplicada a função *fitness* a população são verificadas duas condições de parada, (1) se chegou a milésima geração, (2) encontrou um indivíduo com *fitness* igual a zero. Caso uma dessas condições sejam satisfeitas é exibido o relatório com qual o melhor indivíduo e caso tenha quebrado algum requisito quais e como, e gerados dois arquivos CSV um com o quadro de horário obtido para o curso de LCC e outro para o curso de SI.

Aplicada função *fitness* a população e caso as condições de parada não tenham sido satisfeitas, vem a fase de seleção por torneio, o torneio seleciona 10 indivíduos aleatórios da população e seleciona o indivíduo com o melhor *fitness* dentre os selecionados no torneio o vencedor do torneio irá para a fase de cruzamento, o processo de torneio se repete até que sejam selecionados 50 indivíduos.

Terminado o torneio os 50 indivíduos vão para o processo de *crossover* onde dois indivíduos têm seus cromossomos divididos em um ponto aleatório e um dos lados é selecionado aleatoriamente e trocado com o outro indivíduo, gerando um novo indivíduo, o processo se repete até que sejam gerados 50 novos indivíduos.

Feito o *crossover* essa nova população de indivíduos é submetida a um processo de mutação onde um gene aleatório do cromossomo é marcado e substituído por um novo gene aleatório.

Feito isso a nova população gerada após a mutação é submetida a função *fitness* e o ciclo se repete até que uma das condições de parada sejam satisfeitas.

6.1. Métricas de avaliação

Para avaliar a qualidade do algoritmo e as soluções geradas, usamos o *fitness* do melhor indivíduo de cada geração como métrica.

7. Resultados

Até o momento, foi implementado um protótipo da ferramenta de elaboração de quadros de horários de cursos, com base em algoritmos genéticos, que é de código aberto e flexível para customização de restrições, pesos, funções de parada, etc.

o algoritmo desenvolvido foi capaz de encontrar soluções ótimas com os dados dos cursos de Bacharelado em Sistemas de Informação e de Licenciatura em Ciência da Computação, para o período 2020.1 e para as restrições definidas até o momento. No entanto, é possível que a medida que o número de restrições aumente, o tempo necessário para encontrar a solução ótima aumente ou até mesmo se torne inviável. Apesar disso, a solução desenvolvida é capaz de se adaptar a diferentes requisitos e condições de parada, desde que estes obedeçam o padrão estabelecido na ferramenta.

A Figura 4 apresenta os resultados da execução do algoritmo usando uma população de 50 indivíduos durante 1000 gerações. O melhor valor de *fitness* encontrado dentro de toda a execução foi de 10, ou seja, apenas uma *soft constraint* de peso 10 foi violada.

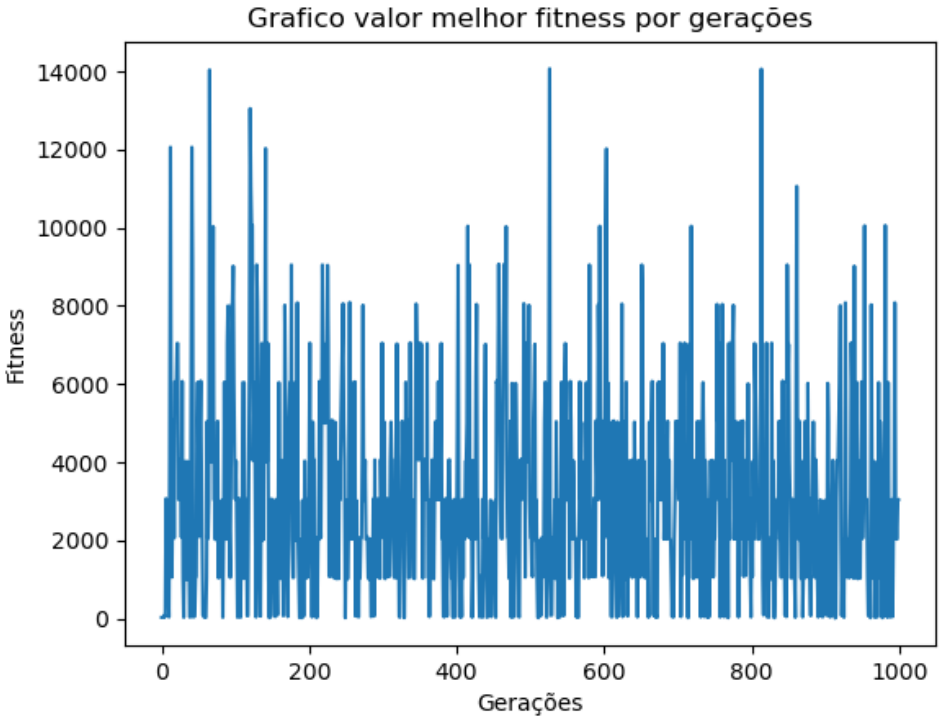


Figura 4. Valor do fitness do melhor indivíduo por geração

Também foi desenvolvida uma funcionalidade de avaliação de quadros de horários já prontos com base nas restrições pré-definidas, indicando alertas sobre restrições sendo violadas. Isto é útil quando, por exemplo, a coordenação elabora um horário manualmente e que verificar inconsistências. Esta funcionalidade recebe um CSV com o quadro de horários e, caso queira além do *fitness* do quadro, também é possível receber um relatório de quais restrições não estão sendo respeitadas por quais partes do cromossomo do indivíduo.

8. Conclusões

A ferramenta desenvolvida é capaz de lidar com a elaboração de quadros de horários, baseada em algoritmos genéticos, de maneira flexível do ponto de vista das restrições e das condições às quais é desejado atingir na elaboração do quadro. Ela também é capaz

de avaliar soluções geradas previamente com base nas restrições aplicadas ao algoritmo, o que por se só trás um grande ganho de tempo na avaliação dos quadros de horários elaborados pelos professores, além de tornar os quadros submetidos a avaliação mais confiáveis quanto ao respeito às diretrizes estabelecidas pelos professores durante a sua elaboração.

Referências

[Ciscon et al. 2005] Ciscon, L. A., Oliveira, A. C., Hipolito, T. R., Alvarenga, G. B., and Roullier, A. C. (2005). O problema de geracao de horarios: um foco na eliminacao de janelas e aulas isoladas. In *XXXVII Brazilian Symposium of Operational Research*.

[Freitas et al. 2007] Freitas, C. C., Guimaraes, P. R., Neto, M. C., and BARBOZA, F. J. (2007). Uma ferramenta baseada em algoritmos geneticos para a geracao de tabela de horario escolar. *SETIMA ESCOLA REGIONAL DE COMPUTACAO Bahia-Sergipe. Vitria da Conquista:[sn]*.

[Lacerda and Carvalho 1999] Lacerda, E. G. and Carvalho, A. (1999). Introdução aos algoritmos genéticos. *Sistemas inteligentes: aplicações a recursos hídricos e ciencias ambientais*, 1:99–148.

[Lobo 2005] Lobo, E. L. M. (2005). Uma solucao do problema de horario escolar via algoritmo genetico paralelo. *Centro Federal de Educaao Tecnologica de Minas Gerais*.

[Vieira and Macedo 2011] Vieira, F. and Macedo, H. (2011). Sistema de alocacao de horarios de cursos universitarios: um estudo de caso no departamento de computacao da universidade federal de sergipe. *Scientia Plena*, 7(3).

Apêndice I - Melhor horário para os cursos de Sistemas de Informação e Ciência da Computação

Tabela 1. Quadro de horários do curso de Sistemas de informação obtido da melhor solução encontrada

Disciplina	Período	Professor	Hora	Dia
Introdução à Programação TUR 2	p1	Ayla	8	sexta
Introdução à Programação TUR 2	p1	Ayla	13	terca
Introdução ao Computador	p1	Ana Liz	8	quarta
Introdução ao Computador	p1	Ana Liz	13	quarta
Metodologia Científica	p1	Substituto	13	terca
Metodologia Científica	p1	Substituto	8	sexta
Administração I	p1	Eliane	13	segunda
Administração I	p1	Eliane	13	segunda
Matemática Elementar	p1	Laudelino	8	quinta
Matemática Elementar	p1	Laudelino	10	quinta
Alg. e Estrutura de Dados 1	p3	Wagner	8	quinta
Alg. e Estrutura de Dados 1	p3	Wagner	13	sexta
Prog. Orientada a Objetos - POO	p3	Raquel	8	terca
Prog. Orientada a Objetos - POO	p3	Raquel	10	quarta
Álgebra Linear e Geom. Analitica	p3	Wendell	8	segunda
Álgebra Linear e Geom. Analitica	p3	Wendell	8	quarta
Banco de Dados I	p5	Leandro	13	terca
Banco de Dados I	p5	Leandro	8	terca
Eng. de Software Aplicada	p5	Juliana/Yuska	8	terca
Eng. de Software Aplicada	p5	Juliana/Yuska	13	sexta
Sistemas Operacionais	p5	Raquel	10	quinta
Sistemas Operacionais	p5	Raquel	13	sexta
Redes de Computadores	p5	Hacks	13	terca
Redes de Computadores	p5	Hacks	13	segunda
Pesquisa Aplicada SI	p5	Adson	13	segunda
Pesquisa Aplicada SI	p5	Adson	13	quarta
Desenv. de Sist. Corporativos	p7	Leandro	8	quarta
Desenv. de Sist. Corporativos	p7	Leandro	13	quarta
Gestão da Qualidade SW	p7	Yuska	8	quarta
Gestão da Qualidade SW	p7	Yuska	8	terca
Sistemas Distribuídos	p7	Marcus	10	quarta
Sistemas Distribuídos	p7	Marcus	8	terca
Auditoria e Seg. de Sistemas	p7	Hacks	8	terca
Auditoria e Seg. de Sistemas	p7	Hacks	10	quinta
Sistemas de Apoio a Gestão	p7	Luiz	10	segunda
Sistemas de Apoio a Gestão	p7	Luiz	8	terca
Intelig. Artificial e Sistemas Espec.	p7	Marcus	8	quinta
Intelig. Artificial e Sistemas Espec.	p7	Marcus	8	sexta

Tabela 2. Quadro de horários do curso de Ciência da Computação obtido da melhor solução encontrada

Disciplina	Período	Professor	Hora	Dia
Introdução à Programação TUR 2	p1	Ayla	8	sexta
Introdução à Programação TUR 2	p1	Ayla	13	terca
Álgebra Linear e Geom. Analitica	p3	Wendell	8	segunda
Álgebra Linear e Geom. Analitica	p3	Wendell	8	quarta
Intelig. Artificial e Sistemas Espec.	p7	Marcus	8	quinta
Intelig. Artificial e Sistemas Espec.	p7	Marcus	8	sexta
IP	p1	Vanessa/Rafael	8	segunda
IP	p1	Vanessa/Rafael	13	sexta
Metodologia	p1	Thaise	13	quinta
Metodologia	p1	Thaise	8	terca
Mat. Elementar	p1	Laudelino	8	terca
Mat. Elementar	p1	Laudelino	10	segunda
Calculo 1	p1	Josevandro	13	sexta
Calculo 1	p1	Josevandro	10	terca
Scaico	p1	Int. Comp	8	quinta
Scaico	p1	Int. Comp	13	terca
Ling. Programação	p2	Ayla	8	quinta
Ling. Programação	p2	Ayla	8	segunda
Arquitetura I	p2	Augusto	13	quinta
Arquitetura I	p2	Augusto	8	quarta
Vetorial	p2	Wendhel	13	terca
Vetorial	p2	Wendhel	13	quarta
Cálculo II	p2	Josevandro	13	segunda
Cálculo II	p2	Josevandro	10	quinta
Estr. de Dados	p3	Wagner	13	terca
Estr. de Dados	p3	Wagner	13	segunda
Arquitetura II	p3	Rafael	13	quinta
Arquitetura II	p3	Rafael	10	quarta
Lógica	p3	Substituto	10	segunda
Lógica	p3	Substituto	13	terca
Álgebra Linear	p3	Wendhel	13	terca
Álgebra Linear	p3	Wendhel	8	terca
POO	p3	Rodrigo Vilar	13	quarta
POO	p3	Rodrigo Vilar	10	terca
Mod. de dados	p4	Pasqueline	8	quarta
Mod. de dados	p4	Pasqueline	13	segunda
SO	p4	Raquel	8	quinta
SO	p4	Raquel	10	segunda
Mod. de dados	p4	Pasqueline	13	sexta
Mod. de dados	p4	Pasqueline	10	quarta
Pesquisa Aplicada	p4	Pasqueline	10	quinta
Pesquisa Aplicada	p4	Pasqueline	8	segunda
Ling. Declarativas	p4	Joelson	13	sexta
Ling. Declarativas	p4	Joelson	8	segunda
Empreend.	p5	DSCA	8	sexta
Empreend.	p5	DSCA	10	terca
Teoria da Comp.	p5	Joelson	10	quinta
Teoria da Comp.	p5	Joelson	10	quarta
Banco de Dados	p5	Vanessa	8	terca
Banco de Dados	p5	Vanessa	8	sexta
Redes	p5	Augusto	13	terca
Redes	p5	Augusto	8	quarta
Sistemas Multimídia	p6	Thaise	10	quarta
Sistemas Multimídia	p6	Thaise	13	quinta
Análise e Projeto	p6	Rodrigo Vilar	13	quinta
Análise e Projeto	p6	Rodrigo Vilar	10	sexta
Estatística	p6	Joseilme	8	quarta
Estatística	p6	Joseilme	13	quarta
EAD	p7	Flávia	8	quarta
EAD	p7	Flávia	10	quinta
Eng. de Software	p7	Leandro	8	segunda
Eng. de Software	p7	Leandro	10	quinta
PMI	p7	Flávia	13	segunda
PMI	p7	Flávia	8	terca
ESII	p7	Thaise	13	terca
ESII	p7	Thaise	10	sexta