Proposta de uma técnica de Mineração em Grafos para identificação de gargalos em currículos de graduação

Jefferson de J. Costa[1], Flávia C. Bernardini[1,2], Danilo Artigas[2]

[1]Programa de Pós-Graduação em Engenharia de Produção e Sistemas Computacionais
Universidade Federal Fluminense – UFF
Campus de Rio das Ostras – Rua Recife, s/n, Jd. Bela Vista – CEP 28895-532
Rio das Ostras, RJ – Brasil

[2]Departamento de Computação – Instituto de Ciência e Tecnologia
Universidade Federal Fluminense – UFF
Campus de Rio das Ostras – Rua Recife, s/n, Jd. Bela Vista – CEP 28895-532
Rio das Ostras, RJ – Brasil

Abstract. *In the last years the increase of the number of entrants in higher education institutions is significantly. However, the number of graduates remains low. The main cause of this issue is the evasion and / or retention in higher education courses. Techniques and methods that assist the identification of causes and / or patterns are important in this area. The purpose of this paper is to propose a method of graph mining to identify bottlenecks retention in undergraduate curricula. To evaluate the proposed method, real data from a federal institution of higher learning were used.*

Resumo. *É significativo o aumento do número de ingressantes em instituições de ensino superior nos últimos anos, porém a quantidade de concluintes se mantém baixa. A principal causa dessa questão é a evasão e/ou retenção nos cursos de ensino superior. Técnicas e métodos que auxiliem na identificação de causas e/ou padrões para retenção são importantes nessa área. O objetivo deste trabalho é propor um método de mineração em grafos para identificação de gargalos de retenção em currículos de graduação. Para avaliar o método proposto, foram utilizados dados reais de uma instituição federal de ensino superior.*

1. Introdução

A quantidade de vagas oferecidas em instituições de ensino superior, em todo o Brasil, cresce a cada ano. De acordo com o Censo da Educação Superior do INEP, o total de alunos matriculados no ensino superior brasileiro, passou de sete milhões em 2012, o que representa um aumento de 4,4% no período de 2011-2012. O mesmo órgão aponta que 6.739.689 alunos se matricularam em algum curso superior no Brasil em 2011 e, em 2010, foram 6.379.299. Observa-se também, por outro lado, que o número de concluintes ainda é baixo. Em 2011, foram registrados 1.076.713 de alunos concluintes contra 873.839 em 2010. A evasão e/ou retenção dos alunos dos cursos de graduação no Brasil configura um dos grandes problemas do ensino superior (MANHÃES, 2011).

DOI: 10.5753/cbie.sbie.2015.1062

Para Silva Filho (2007), a evasão é um dos problemas que mais afligem as instituições de ensino superior em geral, não apenas no Brasil, mas em todos os países, e que afeta diretamente os resultados dos sistemas educacionais. Ainda, a perda/saída de universitários gera desperdícios sociais, acadêmicos e econômicos, tanto no setor público quanto no privado. A evasão implica uma ociosidade de professores, funcionários, equipamentos e espaços físicos.

Outro problema apontado por diversos autores é a retenção. Campello e Lins (2008) apontam, como principal impacto negativo da retenção, o fato dela não permitir que profissionais com nível superior ingressem no mercado de trabalho. Além disso, em algum momento, alunos retidos podem vir a evadir o curso. Como observado por esses autores, ainda existem poucas pesquisas no sentido de descobrir as razões que levam os universitários a abandonar o curso ou adiar sua conclusão. Parafraseando os autores, "faz-se, portanto, urgente que se estabeleça uma sistemática de avaliação que permita diagnosticar esta situação em diferentes regiões do país" (CAMPELO E LINS, 2008, p.2).

Podemos observar que diversos autores tentam explicar o que motiva os alunos a evadir um curso. Segundo Tinto (1987), as questões que podem justificar a evasão ou a permanência dos discentes em cursos superiores são muito pessoais e têm ligação direta com a interação dos alunos com o ambiente universitário. No entanto, ainda existem dúvidas quanto a validade do modelo de Tinto para os problemas brasileiros, pois o trabalho do autor foi desenvolvido com dados de universidades norte-americanas (ANDRIOLA ET AL., 2006). Isso mostra que ainda é necessário muito estudo sobre a realidade brasileira, para tentar explicar os problemas citados.

Na Universidade Federal Fluminense (UFF), a realidade não é diferente. Ainda, por se tratar de uma instituição pública federal, os impactos negativos causados pelo abandono de um aluno refletem em toda a sociedade. Um aluno trancado (ocioso) ou aquele aluno que decidiu abandonar o curso representa um investimento perdido, ou seja, dinheiro público desperdiçado. Segundo dados da própria instituição, em 2013, 69.107 alunos se matricularam na Universidade. Neste mesmo ano, 15.181 alunos trancaram suas matrículas e apenas 2.085 estavam com o *status* de formado. Esses dados mostram que são necessárias ações para tentar diminuir a quantidade de trancamentos e aumentar a quantidade de concluintes.

Em (PECLY E MELLO, 2013), os autores propõem o uso da teoria dos grafos para realizar análises de cunho didático em relação ao fluxograma de um curso de graduação e identificar qual o tempo mínimo para conclusão do curso, a disciplina central do curso, dentre outras questões. Nesse trabalho, somente o fluxograma (ou currículo) do curso foi utilizado para análise. No entanto, os dados de histórico dos alunos também poderiam ser utilizados para extração de padrões para responder questões quanto ao percurso do fluxograma do curso. Assim, utilizar mineração de padrões frequentes em grafos poderia oferecer um ferramental para o problema em questão. No entanto, como detalhado adiante neste trabalho, não identificamos nenhum algoritmo aplicável ao nosso problema.

Baseado na realidade de retenção dos cursos de graduação e por não termos encontrado nenhum método aplicável ao nosso problema, o objetivo deste trabalho é propor um método para identificação de disciplinas que os alunos têm grande dificuldade para

cursar em currículos de graduação. Para atingir esse objetivo, representamos as grades curriculares e os históricos dos alunos como grafos. Para identificação das disciplinas em caminhos de maior custo, propomos um método de mineração em grafos. Para avaliar o método proposto, utilizamos um currículo de um curso de Ciência da Computação da UFF.

Este trabalho está dividido como segue: na Seção 2, são apresentados conceitos de grafos, importantes para a compreensão deste trabalho. Na Seção 3, são apresentadas algumas técnicas de mineração em grafos. Na Seção 4, é apresentada a técnica proposta. Na Seção 5, é apresentado o estudo de caso realizado. Na Seção 6, são apresentados os resultados obtidos.

2. Conceitos de Grafos

As definições de grafos descritas a seguir são baseadas em (SZWARCFITER, 1988; THULASIRAMAN E SWAMY; 2011). Um grafo direcionado G = (V, E) consiste de um conjunto finito não-vazio V e um conjunto finito E de pares ordenados de elementos distintos de V. Os elementos em V são denominados vértices do grafo G, e os pares $e = (u,v) \in E$ são as arestas de G, sendo $\{u,v\} \in V$. Cada aresta $e \in G$ possui um peso $c(e)$ associado. Um subgrafo de um grafo G = (V, E) é definido como qualquer grafo H = (V', E') tal que $V' \subseteq V$ e $E' \subseteq E$.

Um caminho de comprimento k é uma sequência de vértices $(v_0, ..., v_k)$ tal que $v_{i-1},v_i \in E$ para $1 \leq i \leq k$, e todos os vértices da sequência são distintos. Um ciclo de comprimento k é uma sequência de vértices $(v_0, ..., v_k)$ tal que $v_{i-1},v_i \in E$ para $1 \leq i \leq k$, $v_0 = v_k$ e $(v_0, ..., v_{k-1})$ é um caminho. Um grafo que não possui ciclos é denominado acíclico. Um DAG (do inglês *Directed Acyclic Graph*) é um grafo direcionado e acíclico. Um grafo G = (V, E) é denominado conexo quando existe um caminho entre cada um dos pares de vértices de G, e é desconexo caso contrário.

No problema de identificação de retenção em currículos de graduação, representamos os currículos dos cursos (as grades curriculares) como grafos. Nesse grafo G, cada disciplina será representada por um vértice e a relação de pré-requisito entre duas disciplinas será representada por arestas. Assim, se a disciplina v é pré-requisito de w então a aresta (v, w) pertencerá ao grafo G. Também serão incluídos os vértices "Ingresso" e "Conclusão", que representam, respectivamente, o ingresso e a formatura do aluno no curso. Criaremos arestas ("Ingresso", v) para todo vértice v que represente disciplinas sem pré-requisitos no curso, e arestas (v, "Conclusão") para todo vértice v que represente disciplinas que não são pré-requisitos de nenhuma outra no curso. Dessa forma o grafo G representa o fluxograma do curso e possui uma origem e um destino bem definidos.

Na Figura 1 é ilustrado o grafo de um currículo do curso de Bacharelado em Ciência da Computação, da UFF, que também é utilizado em nosso estudo de caso. Para gerar o grafo, utilizamos a ferramenta Gephi (GEPHI, 2015). A lista dos códigos de todas as disciplinas e seus respectivos nomes pode ser visualizada na Tabela 1. A partir desse grafo, os históricos dos alunos são mapeados de maneira a identificar o peso de cada aresta (u,v) da grade curricular, indicado pela quantidade de vezes que um aluno teve que cursar a disciplina u antes de poder cursar a disciplina v.

É interessante em nosso trabalho identificar os caminhos mais custosos nos currículos dos cursos e nos históricos dos alunos que concluíram algum curso de graduação. O caminho mais custoso de um grafo é o caminho que possui o maior custo, ao somar as arestas do caminho, dentre todos os possíveis caminhos no grafo. No nosso problema, estamos interessados em identificar o caminho mais longo do vértice "Ingresso" ao vértice "Conclusão". Neste trabalho, chamamos o caminho mais longo de caminho mais custoso. Como as arestas de G representam pré-requisitos de disciplinas de um curso de graduação, então G é um DAG. O problema de determinar um caminho mais custoso em um DAG pode ser resolvido utilizando um algoritmo de tempo linear. Tal algoritmo é uma adaptação do algoritmo de Bellman-Ford, e encontra-se descrito em detalhes em (DASGUPTA, PAPADIMITRIOU E VAZIRANI, 2010). Neste trabalho, implementamos tal algoritmo para obter os dados de retenção dos alunos em caminhos mais longos, ou mais custosos.

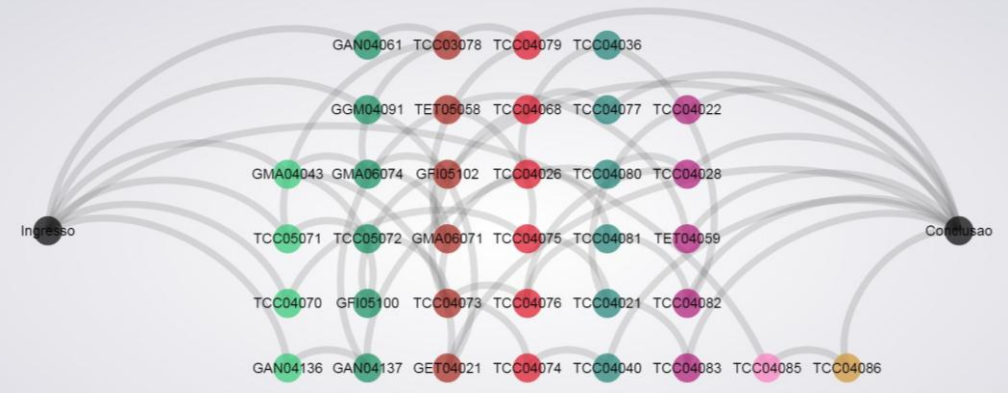

Figura 1 - Currículo do curso de Bacharelado em Ciência da Computação da UFF, utilizado no estudo de caso

Tabela 1 – Códigos e Nomes das Disciplinas do Currículo exibido na Figura 1

Código	Nome	Código	Nome
GAN04061	Álgebra Linear	TCC04070	Organização de Computadores I
GAN04136	Introdução a Álgebra	TCC04073	Estruturas de Dados I
GAN04137	Lógica para Ciência da Computação	TCC04074	Estruturas de Dados II
GET04021	Estatística XI	TCC04075	Organização de Computadores II
GFI05100	Física Geral e Experimental XVIII	TCC04076	Linguagens de Programação
GFI05102	Física Geral e Experimental XX	TCC04077	Análise e Projeto de Algoritmos
GGM04091	Análise Combinatória	TCC04079	Métodos Numéricos II
GMA04043	Cálculo Diferencial e Integral Aplicado I	TCC04080	Linguagens Formais e Teoria da Comp.
GMA06071	Equações Diferenciais Aplicadas	TCC04081	Engenharia de Software I
GMA06074	Cálculo Diferencial e Integral Aplicado II	TCC04082	Engenharia de Software II
TCC03078	Métodos Numéricos I	TCC04083	Redes de Computadores I
TCC04021	Sistemas Operacionais I	TCC04085	Projeto de Aplicação I
TCC04022	Sistemas Operacionais II	TCC04086	Projeto de Aplicação II
TCC04026	Banco de Dados	TCC05071	Programação de Computadores I
TCC04028	Compiladores	TCC05072	Programação de Computadores II
TCC04036	Projeto de Banco de Dados	TET04059	Microprocessadores
TCC04040	Inteligência Artificial	TET05058	Circuitos Digitais
TCC04068	Algoritmos em Grafos		

3. Mineração em Grafos

Aggarwal e Wang (2010) afirmam que a teoria de grafos pode ser aplicada em diversos domínios: análise de dados químicos, bioinformática, redes sociais, análises de *links* na *web* e redes de computadores. Ainda assim, os grafos geralmente são utilizados apenas quando as relações entre os vértices são muito importantes para a aplicação. O padrão extraído dos dados normalmente é representado por grafos, subgrafos ou expressões mais abstratas dos padrões implícitos nos dados (COOK E HOLDER, 2007).

Existem três abordagens principais na mineração de grafos: classificação, agrupamento e descoberta de padrões (ou subestruturas) frequentes (AGGARWAL E WANG, 2010). A classificação de grafos abrange duas tarefas diferentes. A primeira constrói um modelo para predizer o rótulo de classe do grafo todo. Já a segunda tarefa, também denominada propagação de rótulo, tenta predizer os rótulos de classe dos vértices em um grande grafo. A construção de agrupamentos é indicada quando desejamos dividir um conjunto de dados em grupos que possuem objetos similares. Os algoritmos de construção de agrupamentos em grafos geralmente são divididos em duas categorias: algoritmos de construção de agrupamentos de nós e algoritmos de construção de agrupamentos de grafos.

A descoberta de subgrafos frequentes consiste em duas etapas. Na primeira fase, os candidatos à subestrutura frequente são gerados, enquanto a frequência de cada candidato é verificada na segunda etapa. A maioria dos estudos relacionados à descoberta de subgrafos frequentes foca na primeira etapa, já que o segundo passo envolve o problema de isomorfismo de subgrafos, que é NP-completo (COOK E HOLDER, 2007). Existem diversos algoritmos de descoberta de subgrafos frequentes baseados no algoritmo *Apriori*. Um dos problemas desses algoritmos é que o espaço de busca de padrões em grafos cresce exponencialmente com o tamanho do padrão, o que traz um gargalo computacional. Outros algoritmos foram propostos para tentar resolver o referido problema, baseados na técnica *Pattern Growth*, que estende padrões a partir de um único padrão.

O gSpan, baseado em *Pattern Growth*, foi projetado para reduzir a geração de grafos duplicados. Além disso, garante a descoberta do conjunto completo de grafos frequentes (COOK E HOLDER, 2007). O gSpan adota o método heurístico de busca em profundidade para percorrer os grafos. Inicialmente, um vértice é escolhido aleatoriamente, e os outros são marcados conforme são visitados. O conjunto de vértices visitados é expandido repetidamente até que uma árvore de busca em profundidade completa seja construída. A diferença do gSpan para a busca em profundidade está na maneira como as arestas podem ser adicionadas à árvore de busca em profundidade. O algoritmo gSpan é utilizado para grafos não-direcionados. Uma desvantagem do gSpan é que funciona bem apenas com conjuntos de dados de tamanho moderado.

O mSpan é um algoritmo criado por Li et al. (2009) com o objetivo de minerar grafos direcionados e rotulados, baseado no gSpan. O mSpan gera um código mínimo dos pesos das arestas e um código de sequência de nós abstrato para identificar um grafo direcionado, com o objetivo de resolver o problema de isomorfismo de subgrafos e da redundância na expansão do grafo. O algoritmo também cria um código de direção de uma aresta, baseado na direção da sua expansão, e registra isso dentro do rótulo da

aresta para identificar um padrão de grafo direcionado. O algoritmo foi proposto com o objetivo de analisar uma base de dados de uma instituição financeira chinesa e tentar descobrir caminhos de lavagem de dinheiro. Os grafos foram gerados considerando as contas bancárias como vértices, e as informações das transações entre as contas como arestas. A direção das arestas era gerada de acordo com o fluxo de dinheiro. Como o objetivo era encontrar características comuns de lavagem de dinheiro, os vértices não foram considerados durante o processo de mineração. Portanto, o mSpan é um algoritmo que não considera os vértices de um grafo no processo de mineração, considerando somente as arestas do grafo.

No problema por nós representado, cada grafo de grade curricular é um grafo dirigido, acíclico e conexo, e é importante ter conhecimento dos vértices que fazem parte dos subgrafos frequentes. Os dois algoritmos que melhor se adequam ao nosso problema são o mSpan e o gSpan. No entanto, o gSpan foi construído para minerar grafos não-dirigidos, e o mSpan não considera os vértices no processo de mineração. Ainda, é importante observar que a saída de todos os algoritmos de mineração em grafos para descoberta de padrões frequentes pode ser qualquer elemento de um grafo, como um único nó muito frequente ou uma simples aresta mais frequente. No nosso caso, o foco é descobrir caminhos mais longos e mais frequentes. A ordenação dos caminhos e as arestas são importantes também. Sendo assim, propomos um método para encontrar arestas frequentes nos currículos que melhor se adequasse ao nosso problema, descrito na próxima seção.

4. HeavierEdge: Identificação das arestas mais pesadas em um currículo de graduação

O objetivo do método HeavierEdge é identificar disciplinas que causam retenção em um curso de graduação. Esse processo pode ser visto como identificar quais pré-requisitos possuem alto índice de reprovação, levando o aluno a percorrer caminhos mais custosos na grade curricular.

Na Seção 2, foi descrita a representação de um currículo de graduação, onde cada vértice representa uma disciplina em um currículo, e cada aresta (u,v) é a ligação entre duas disciplinas, sendo u pré-requisito da disciplina v. Como o interesse está em analisar gargalos de currículos, o histórico de um aluno que já foi graduado no currículo em análise deve ser representado levando em consideração a estrutura curricular do curso.

O método HeavierEdge considera como entrada:

- O grafo da grade curricular G = (V, E);
- O conjunto de grafos $\mathbf{H} = \{G'_1, ..., G'_N\}$ que representam os históricos de todos os N alunos que concluíram a grade curricular G;

A saída do método são todas as arestas de G consideradas mais custosas dos caminhos mais custosos de cada histórico. No Algoritmo 1 são apresentados os passos do algoritmo para obter as arestas ordenadas em ordem decrescente. No passo 1, os vértices e as arestas do grafo G, do currículo fornecido como entrada ao algoritmo, são copiados para um grafo G*. No passo 2, os custos das arestas do grafo G* são inicializados com o valor zero. No passo 3.1, para cada grafo de histórico de aluno, encontre o caminho mais custoso utilizando o algoritmo de Bellman-Ford. Para cada aresta equivalente do

caminho encontrado em 3.1, some o custo de cada aresta em G* (passos 3.2 e 3.3). Normalize os custos de cada aresta em G* pelo número de alunos (passo 4). Ordene, em ordem decrescente, as arestas do grafo pelo custo das arestas.

5. Um estudo de caso

Para realizar o nosso estudo de caso, utilizamos o currículo apresentado na Seção 2 e ilustrado na Figura 1. Em relação aos dados dos históricos dos alunos, foram utilizados 303 históricos de alunos que concluíram o curso nos anos de 2004 a 2014. Inicialmente, calculamos as arestas mais custosas considerando todos os históricos, sem considerar o caminho mais longo.

Algoritmo 1 – Algoritmo para encontrar as arestas ordenadas

1. Faça uma cópia dos vértices e das arestas do grafo G em G*;
2. Para cada $e_i^* \in$ G* faça:
   1. $c(e_i^*) = 0$;
3. Para cada grafo $G'_i \in \mathbf{H}$ faça:
   1. Encontre o caminho C'_i mais longo, ou mais custoso, de G'_i utilizando o algoritmo de Bellman-Ford;
   2. Para cada aresta $e_i = (u,v) \in C'_i$, encontre a aresta equivalente $e_i^* = (u,v) \in$ G*;
   3. Faça $c(e_i^*) = c(e_i^*) + c(e_i)$;
4. Para cada $e_i^* \in$ G* faça:
   1. $c(e_i^*) = c(e_i^*) / N$;
5. Ordenar, em ordem decrescente pelo custo de cada aresta $e_i^* \in$ G*, considerando somente as arestas cujo custo $c(e_i^*) > 0$.

Na Tabela 2 são exibidos, para cada aresta do currículo exibido na Figura 1, as arestas do grafo, o custo absoluto de cada aresta e o custo relativo. Deve ser observado que as arestas cujo custo relativo foi igual a um indica que todos os alunos cursaram uma única vez, e portanto foram retiradas da tabela. Nessa tabela, pode ser observado que as três primeiras disciplinas que levam às sete arestas mais custosas são: GMA06074 (Cálculo Diferencial e Integral Aplicado II); GFI05100 (Física Geral e Experimental XVIII); e GMA04043 (Cálculo Diferencial e Integral Aplicado I). Tais resultados são esperados em um curso de Ciência da Computação.

Na Tabela 3 são exibidos os dados obtidos utilizando o método HeavierEdge. Pode ser observado nessa tabela que todas as disciplinas do currículo aparecem em alguma aresta custosa, pertencente a algum caminho mais longo de ao menos um aluno. Em um primeiro momento, o leitor pode estranhar o fato de aparecer arestas na qual a disciplina de pré-requisito é “Ingresso”, ou seja, quando o aluno iniciou o curso. No entanto, para tornar o grafo conexo, a entrada do aluno no curso foi representada com o vértice “Ingresso”, e o custo de “Ingresso” para qualquer disciplina sempre possui o valor um. Assim sendo, pode-se verificar nos resultados que a disciplina TCC05071 (Programação de Computadores I) apareceu em 166 caminhos mais longos dos 303 alunos do curso; a disciplina GMA04043 (Cálculo Diferencial e Integral I) apareceu em 67 caminhos mais longos; GAN04136 (Introdução à Álgebra), em 66 caminhos mais longos; TCC04085 (Projeto de Aplicação I), em 3 caminhos mais longos; e TCC04070 (Organização de

Computadores I), em 1 caminho mais longo. Nesse caso, observamos que não somente as disciplinas das áreas de matemática e física apareceram nos caminhos mais longos.

Tabela 2 – Arestas mais custosas

Aresta e_i	Custo absoluto $c(e_i)$	Custo relativo $c(e_i) / N$	Aresta e_i	Custo absoluto $c(e_i)$	Custo relativo $c(e_i) / N$
GMA06074 - GMA06071	486	1.60	TCC04021 - TCC04022	373	1.23
GMA06074 - GFI05102	486	1.60	TCC04021 - TCC04083	373	1.23
GMA06074 - TCC03078	486	1.60	TCC04080 - TCC04028	371	1.22
GFI05100 - GFI05102	479	1.58	GAN04136 - GAN04137	362	1.19
GMA04043 - GFI05100	479	1.58	GAN04137 - TCC04080	354	1.17
GMA04043 - GET04021	479	1.58	GAN04137 - TET05058	354	1.17
GMA04043 - GMA06074	479	1.58	TET04059 - Conclusão	352	1.16
GFI05102 - Conclusão	462	1.52	TCC04070 - TCC04075	346	1.14
TCC04086 - Conclusão	461	1.52	TCC04083 - Conclusão	346	1.14
TCC04075 - TCC04021	434	1.43	TCC04082 - Conclusão	342	1.13
TCC04077 - Conclusão	428	1.41	TCC04073 - TCC04068	335	1.11
GMA06071 - TCC04079	420	1.39	TCC04073 - TCC04026	335	1.11
GMA06071 - GFI05102	420	1.39	TCC04073 - TCC04074	335	1.11
TCC05072 - TCC04073	417	1.38	TCC04073 - TCC04076	335	1.11
TCC05071 - TCC03078	409	1.35	TCC04081 - TCC04082	333	1.10
TCC05071 - TCC05072	409	1.35	TCC03078 - TCC04079	332	1.10
GAN04061 - TCC03078	404	1.33	TCC04028 - Conclusão	331	1.09
GET04021 - Conclusão	397	1.31	TCC04036 - TCC04082	329	1.09
TCC04022 - Conclusão	396	1.31	TCC04085 - TCC04086	328	1.08
TCC04026 - TCC04036	391	1.29	TCC04040 - Conclusão	326	1.08
TCC04026 - TCC04081	391	1.29	TCC04074 - TCC04040	324	1.07
TCC04068 - TCC04077	379	1.25	TET05058 - TCC04075	316	1.04
TCC04079 - Conclusão	376	1.24	TET05058 - TET04059	316	1.04
TCC04076 - TCC04081	374	1.23			

Tabela 3 – Resultados obtidos com o algoritmo HeavierEdge

Aresta e_i	Custo absoluto $c(e_i)$	Custo relativo $c(e_i) / N$	Aresta e_i	Custo absoluto $c(e_i)$	Custo relativo $c(e_i) / N$
TCC05071 - TCC05072	245	0.81	TCC04068 - TCC04077	45	0.15
TCC05072 - TCC04073	240	0.79	TCC04073 - TCC04068	33	0.11
Ingresso - TCC05071	166	0.55	TCC04021 - TCC04083	31	0.10
TCC04026 - TCC04036	152	0.50	TCC04083 - Conclusão	30	0.10
TCC04082 - Conclusão	152	0.50	TCC04081 - TCC04082	29	0.10
GMA06074 - GMA06071	151	0.50	TCC04076 - TCC04081	24	0.08
TCC04073 - TCC04026	141	0.47	TCC04079 - Conclusão	24	0.08
GMA04043 - GMA06074	140	0.46	TCC04086 - Conclusão	19	0.06
TCC04036 - TCC04082	127	0.42	TCC04026 - TCC04081	16	0.05
GFI05102 - Conclusão	123	0.41	GMA06071 - TCC04079	15	0.05
TCC04075 - TCC04021	114	0.38	TCC04073 - TCC04076	11	0.04
GMA06071 - GFI05102	106	0.35	GFI05100 - GFI05102	7	0.02
TCC04022 - Conclusão	98	0.32	TCC04028 - Conclusão	6	0.02
GAN04136 - GAN04137	88	0.29	GMA06074 - TCC03078	5	0.02
GAN04137 - TET05058	82	0.27	GMA04043 - GFI05100	4	0.01
TCC04021 - TCC04022	73	0.24	TCC03078 - TCC04079	4	0.01
TET05058 - TCC04075	71	0.23	TCC04085 - TCC04086	4	0.01
Ingresso - GMA04043	67	0.22	Ingresso - TCC04085	3	0.01
TCC04077 - Conclusão	67	0.22	TCC04076 - TCC04028	3	0.01
Ingresso - GAN04136	66	0.22	TCC04070 - TCC04075	3	0.01

É interessante observar a comparação dos números obtidos nas duas tabelas. Como foi previamente descrito, os números apresentados na Tabela 2 são esperados. Por esse motivo, esperávamos que a maioria dos alunos possuíssem, nos caminhos mais longos de seus históricos, as disciplinas que apareceram como arestas mais custosas. No entanto, as três arestas mais custosas dos caminhos mais longos dos alunos no curso são as que envolvem disciplinas de Programação, como pode ser observado nas cinco primeiras arestas (ou linhas) na Tabela 3: Programação de Computadores I (TCC05071), Programação de Computadores II (TCC05072), Banco de Dados (TCC4026) e TCC4082 (Engenharia de Software II). Sendo assim, o algoritmo HeavierEdge mostra resultados interessantes para serem analisados por coordenadores de curso e gestores de universidades pois, ao analisar em número absoluto somente a quantidade de reprovações por disciplina, que indiretamente são os números exibidos na Tabela 2, os resultados já são conhecidos e esperados em cursos de Ciência da Computação e outros cursos de graduação da área de exatas. No entanto, utilizando o método HeavierEdge e analisando seus resultados, exibidos na Tabela 3, descobrimos dados interessantes, que mostram que as arestas mais custosas nos caminhos mais longos de um curso de computação são justamente as disciplinas principais ligadas a esse curso.

6. Conclusões e Trabalhos Futuros

Neste trabalho, é proposto o algoritmo HeavierEdge para identificação de arestas mais custosas em caminhos mais longos executados por alunos em currículos de graduação. Tal algoritmo foi por nós proposto pois não encontramos na literatura algoritmos de mineração em grafos que pudessem atender a nossa necessidade de mineração. Realizamos um estudo de caso utilizando o método proposto e dados de um currículo de um curso de Bacharelado em Ciência da Computação da Universidade Federal Fluminense (UFF). O esperado é que em cursos de computação as disciplinas que representam maior gargalo no curso são as disciplinas de matemática e física. Esse resultado foi obtido quando analisamos somente as arestas mais custosas, ou seja, as que possuem maiores índices de reprovação. No entanto, ao utilizarmos o método HeavierEdge, descobrimos que diversas disciplinas de Computação aparecem muitas vezes nos caminhos mais longos dos históricos. Tal resultado não era esperado, trazendo uma possibilidade de novos conhecimentos para o coordenador desse curso em particular. Sendo assim, esse estudo de caso nos leva a acreditar que novos conhecimentos podem ser descobertos em relação à realidade dos cursos de graduação em geral quanto ao percurso do currículo por parte dos alunos.

Atualmente, estamos trabalhando em outros algoritmos para detecção dos caminhos completos mais longos dos currículos, dentre outros padrões que possam ser considerados interessantes para serem exibidos. Ainda, pretendemos fazer mais estudos com outros cursos de graduação para avaliar a qualidade dos novos conhecimentos a serem descobertos.

Agradecimentos

Agradecemos à Superintendência de Tecnologia da Informação (STI) da UFF pelo apoio e por ter nos fornecido os dados para o desenvolvimento deste trabalho. Agradecemos também aos revisores pelos importantes comentários para a melhoria deste trabalho.

Referências Bibliográficas

AGGARWAL, C. C.; WANG, H. (2010) *Managing and Mining Graph Data*. Springer.

ANDRIOLA, W. B.; ANDRIOLA, C. G.; MOURA, C. P. (2006) Opiniões de docentes e de coordenadores acerca do fenômeno da evasão discente dos cursos de graduação da Universidade Federal do Ceará (UFC). Ensaio: Aval. Pol. Públ. Educ., Rio de Janeiro, v.14, n.52, p.365-382.

CAMPELLO, A. V. C.; LINS, L. N. (2008) Metodologia de análise e tratamento da evasão e retenção em cursos de graduação de instituições federais de ensino superior. XXVIII Encontro Nacional de Engenharia De Produção, RJ, 13p.

CHERKASSKY, B. V., GOLDBERG, A. V., RADZIK, T. (1994) Shortest paths algorithms: Theory and experimental evaluation. In: ACM-SIAM Symposium on Discrete Algorithms, pages 516–525.

COOK, D. J.; HOLDER, L. B. (2007) *Mining Graph Data*. Nova Jersey: John Wiley & Sons.

DASGUPTA, S.; PAPADIMITRIOU, C.; VAZIRANI, U. (2010) *Algoritmos*. McGraw Hill Brasil.

GEPHI (2015) Gephi: The Open Graph Viz Platform. Disponível em http://gephi.github.io/. Acessado em 26 de maio de 2015.

LI, Y. et al. (2009) A directed labeled graph frequent pattern mining algorithm based on minimum code. In: *Third International Conference on Multimedia and Ubiquitous Engineering (MUE'09)*, Qingdao, China, p. 353-359.

MANHÃES, L. M. B.; CRUZ, S. M. S.; COSTA, R. J. M.; ZAVALETTA, J.; ZIMBRÃO, G. (2011) Previsão de estudantes com risco de evasão utilizando técnicas de mineração de dados. In: Anais do XXII Simpósio Brasileiro de Informática na Educação, Aracaju-SE, p. 150 - 159.

PECLY, P.H.D.; MELLO, J.C.C.B.S. (2013) A Teoria dos Grafos na Análise do Fluxograma do Curso de Engenharia de Produção da UFF. *Relatórios de Pesquisa em Engenharia de Produção*, v.13, Série B. n.3, p. 20-33.

SILVA FILHO, R.L.L. et al. (2007) A evasão no ensino superior brasileiro. Cadernos de Pesquisa, v. 37, n. 132, p. 641-659.

SZWARCFITER, J.L. (1988) *Grafos e algoritmos computacionais*. Rio de Janeiro: Campus.

THULASIRAMAN, K.; SWAMY, M. N. S. (2011) *Graphs: theory and algorithms*. Canada: John Wiley & Sons.

TINTO, V. (1987) *Leaving college: Rethinking the causes and cures of student attrition*. University of Chicago Press.

WASHIO, T.; MOTODA, H. (2003) State of the art of graph-based data mining. In: International Conference on Knowledge Discovery and Data Mining. *ACM SIGKDD Explorations Newsletter*, New York, USA, v. 5, n. 1, p. 59 – 68, 2003.