INSTITUTO FEDERAL DE MINAS GERAIS
CENTRO DE TECNOLOGIA
PROGRAMA DE BACHARELADO EM ENGENHARIA DE COMPUTAÇÃO

Guilherme Magno

VISUALIZAÇÃO E PREDIÇÃO DO COMPORTAMENTO DE PARTIDA DO JOGO LEAGUE OF LEGENDS UTILIZANDO GRAFOS.

Bambuí, MG
2018

Guilherme Magno

VISUALIZAÇÃO E PREDIÇÃO DO COMPORTAMENTO DE PARTIDA DO JOGO LEAGUE OF LEGENDS UTILIZANDO GRAFOS.

Trabalho de Conclusão de Curso apresentado ao Bacharelado em Engenharia de Computação do Instituto Federal de Minas Gerais (IFMG), como requisito parcial para a obtenção do grau de **Bacharel em Engenharia de Computação**

Orientador: Prof. Dr. Laerte Mateus

Bambuí, MG

2018

Magno, Guilherme

Visualização e predição do comportamento de partida do jogo League of Legends utilizando grafos. / por Guilherme Magno. – 2018.
38 f.: il.; 30 cm.

Orientador: Nome do orientador TEM Q CONFERIR ISSO
Trabalho de Conclusão de Curso - Instituto Federal de Minas Gerais, Centro de Tecnologia, Bacharelado em Engenharia de Computação, RS, 2018.

1. Modelo. 2. Latex. 3. Tcc. 4. Graduação. I. TEM Q CONFERIR ISSO, Nome do orientador. II.Visualização e predição do comportamento de partida do jogo League of Legends utilizando grafos. .

E-mail: Routmagno@gmail.com

Guilherme Magno

VISUALIZAÇÃO E PREDIÇÃO DO COMPORTAMENTO DE PARTIDA DO JOGO LEAGUE OF LEGENDS UTILIZANDO GRAFOS.

Trabalho de Conclusão de Curso apresentado ao Bacharelado em Engenharia de Computação do Instituto Federal de Minas Gerais (IFMG), como requisito parcial para a obtenção do grau de **Bacharel em Engenharia de Computação**

Aprovado em dia de mês de 2018:

Nome do orientador TEM Q CONFERIR ISSO, Dr. (UFSM)
(Presidente/Orientador)

Nome menbro banca Sobre nome, Me. (UFSM)

Nome menbro banca Sobre nome, Tecg. (UFSM)

Bambuí, MG

2018

DEDICATÓRIA

Dedico este trabalho a .....

AGRADECIMENTOS

Agradeço .....

"O valor de uma coisa depende da maneira como a abordamos mentalmente e não da coisa em si"

(JIGORO KANO)

RESUMO

VISUALIZAÇÃO E PREDIÇÃO DO COMPORTAMENTO DE PARTIDA DO JOGO LEAGUE OF LEGENDS UTILIZANDO GRAFOS.

AUTOR: GUILHERME MAGNO
ORIENTADOR: NOME DO ORIENTADOR TEM Q CONFERIR ISSO

Aqui você escreve o resumo. Lembrando no máximo 250 palavras para tcc e 500 palavras para tese ou dissertação.

Palavras-chave: Modelo. latex. tcc. graduação.

ABSTRACT

ABSTRACT TITLE

AUTHOR: GUILHERME MAGNO
ADVISOR: NOME DO ORIENTADOR TEM Q CONFERIR ISSO

Here you write the summary. Remembering a maximum of 250 words for tcc and 500 words for thesis or dissertation.

Keywords: Model. latex. tcc. graduation.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

MOBA	Arena de batalha online de multijogadores (do inglês *Multiplayer Online Battle Arena*)
API	Interface de Programação de Aplicativos (do inglês *Application Programming Interface*)
JSON	Notação de Objeto JavaScript (do inglês *JavaScript Object Notation*)
URL	Localizador Uniforme de Recursos (do inglês *Uniform Resource Locator*)
LOL	Liga das Lendas (do inglês *League of Legends*)

SUMÁRIO

1 INTRODUÇÃO

Atualmente muitos jogos tem conquistado a atenção e o tempo de muitas pessoas, com o surgimento da nova categoria, chamada *e-sports*, muitas pessoas têm dedicado o seu tempo aos jogos como forma de trabalho, já que a premiação só vem aumentando com o passar dos anos.

Premiação essa que, segundo uma pesquisa realizada pela ESPN (2017), teve um total no Mundial de *League of Legends* de 2017 ultrapassou 4 milhões de dólares. Mostrando que os jogos já estão tendo premiações consideráveis.

E como todos os jogos que existem equipes adversárias, no *League of Legends*, existem vencedores e perdedores. E este trabalho vem com a intenção de prever os vencedores de uma partida desse jogo utilizando grafos.

Quando descrevemos uma situação utilizando pontos e algumas ligações entre esses pontos, estamos modelando este cenário em forma de grafos. E quando algo está modelado em forma de grafos, podemos usar tecnologias que são aplicáveis em grafos.

1.1 OBJETIVOS

Os objetivos desse trabalho podem ser divididos em objetivos gerais e específicos, que podem ser encontrados nas próximas subseções, junto também com os resultados esperados.

1.1.1 Objetivos Gerais

Criar um modelo computacional capaz de exibir resumos dos dados armazenados e predizer um vencedor no jogo *League of Legends* baseando-se num histórico de partidas usando redes complexas.

1.1.2 Objetivos Específicos e Resultados Esperados

1. Construir um *webservice* para visualização;
2. Calcular deterministicamente quantas vezes cada herói ganhou contra/com outro herói;
3. Testar o resultado obtido em outra amostra de dados;
4. Documentar os resultados.

1.2 JUSTIFICATIVA

Devido a crescente popularidade dos videogames no mundo e seus campeonatos mundiais valendo milhões de dólares, a criação deste modelo computacional capaz de visualizar e predizer a possibilidade de vitórias e derrotas, [continua...]

2 FUNDAMENTAÇÃO TEÓRICA

Nessa parte será apresentados conceitos para melhor entendimento deste trabalho e da modelação utilizada. A seção 2.1 elucida o que são grafos e o capítulo 2.2 explica a variação de grafo utilizado para modelar o problema proposto. Depois na seção 2.3 é esclarecido o ambiente que será abstraído, explicando algumas regras básicas do jogo.

2.1 GRAFOS

Quando descrevemos uma situação utilizando pontos e ligações entre algum desses pontos, como o exemplo os pontos sendo pessoas e as ligações entre as pessoas sendo as amizades feitas, a abstração matemática desse tipo dá lugar ao conceito de grafo (LUCCHESI, 1979).

Segundo VIANA (2007) "Um grafo $G(V, E)$ pode ser definido como um conjunto de vértices V, e um conjunto de conexões E "e ele continua: "Cada elemento do conjunto E, associa dois elementos do conjunto V, assim, se $(u, v) \in E$, então existe uma conexão entre os vértices u e o vértice v .". Sendo eles classificados como vizinhos ou adjacente (FEOFILOFF; KOHAYAKAWA; WAKABAYASHI, 2011).

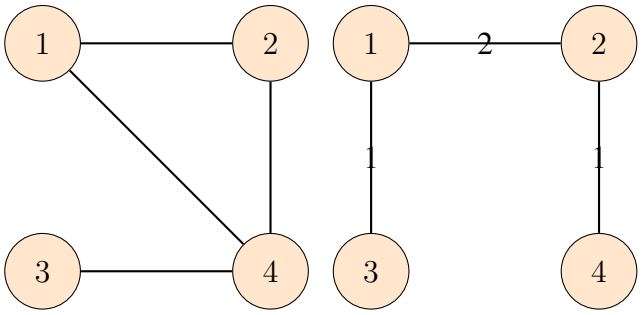
2.1.1 Binário e Não-binário

Os grafos também podem ser classificados como binários e não-binários, sendo os binários suas arestas representadas por 1 (se existem) ou 0 (se não existem) e os não-binários quando existe um conjunto W onde informa a intensidade da interação de uma aresta entre dois vértices (VIANA, 2007). Na Figura 1a temos um grafo binário onde se $(u, v) \in E$ então ela está representada no grafo, já na Figura 1b além da existência da conexão entre os vértices, está representado o conjunto W informando o peso ou aresta daquela aresta.

2.1.2 Vizinhança

FEOFILOFF; KOHAYAKAWA; WAKABAYASHI (2011) diz sobre a vizinhança de um grafo que "O conjunto de vértices X de um grafo G é o conjunto de todos os vértices que tem algum vizinho em X", e diz que esse conjunto de vértices pode ser chamado de $\Gamma_G(X)$. Exemplificando, o conjunto $\Gamma_G(1)$ da Figura 1a são os vértices 2 e 4 e a aresta $(2, 4)$ sendo representados na Figura 2.

Figura 1 – Exemplo de grafos.

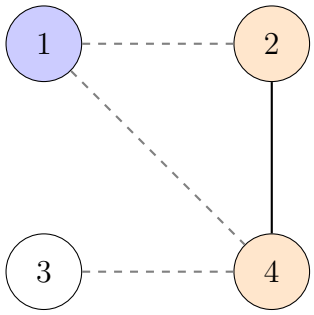


(a) Grafo binário com quatro vértices e quatro arestas.

(b) Grafo não-binário com quatro vértices e três arestas.

Fonte: Autor.

Figura 2 – Exemplo de vizinhança de um vértice de um grafo.



Fonte: Autor.

2.1.3 Grau

O grau de um vértice v é definido por FEOFILOFF; KOHAYAKAWA; WAKABAYASHI (2011) sendo a quantidade de arestas que chegam no vértice v, sendo denotado por $g(v)$. Continuando ainda no exemplo da Figura 1a o $g(1)$ é 2 pois apenas as arestas $(1, 2)$ e $(1, 4)$ incidem no vértice 1.

O grau máximo de um grafo G é o número do vértice que tem o maior grau presente nesse grafo, ou seja $\Delta(G) = max\{g(v) : v \in V(G)\}$. E o grau mínimo de um grafo G é o grau do vértice com menor grau, $\delta(G) = min\{g(v) : v \in V(G)\}$. E ainda, um grafo é regular se $\delta(G) = \Delta(G)$ e é k-regular se $\delta(G) = \Delta(G) = k$ (FEOFILOFF; KOHAYAKAWA; WAKABAYASHI, 2011).

E quando modelamos um sistema real em forma de grafos vamos ver na seção seguinte que esse grafo recebe um nome especial, sendo chamado de rede complexa.

2.2 REDES COMPLEXAS

Sobre redes complexas VIANA (2007) diz que é utilizado o termo redes complexas quando um grafo representa um sistema físico real, então levando em conta isso, um grafo do jogo LOL pode ser considerado como uma rede complexa.

Para que seja possível classificar os resultados adequadamente, serão apresentados três modelos que se destacam no estado da arte segundo ALBERT; BARABáSI (2002): As redes *small worlds*, as redes livres de escala e as redes aleatórias. E também será explicado sobre o coeficiente de aglomeração .

2.2.1 *Redes Small Worlds*

As redes *small worlds* são redes que o caminho entre dois nós são relativamente pequenos. (continua...).

2.2.2 Redes Livres de Escala

Nas redes livres de escala um nó tem a probabilidade $P(k)$ de possuir k arestas obedecendo a lei da potência $P(k) \sim k^{-y}$ (ALBERT; BARABáSI, 2002; ANTIQUEIRA et al., 2005). Segundo VIANA (2007) nas redes livres de escala muitos nós tem poucas arestas e poucos nós se ligam a muitos.

2.2.3 Redes Aleatórias

Segundo VIANA (2007) as redes aleatórias são um sistema formado por E arestas e N vértices, onde as arestas são distribuídas aleatoriamente. CUNHA RECUERO (2004) apud (tenho que ver certinho) exemplifica esse tipo de rede como uma festa, onde “bastava uma conexão entre cada um dos convidados de uma festa, para que todos estivessem conectados ao final dela” e que quanto mais conexões forem criadas, maior a chance de serem criados grupos de pessoas que de tempos em tempos se relacionavam com outros grupos e que poderiam concluir que esses nós se relacionavam de forma randômica.

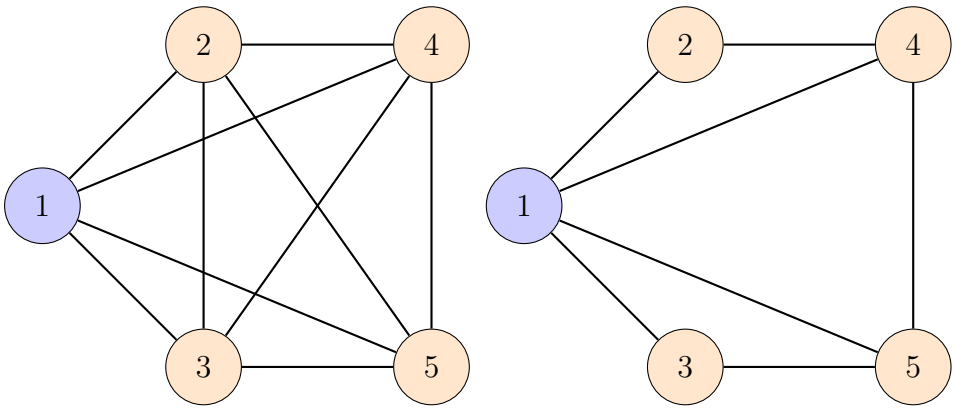
2.2.4 Coeficiente de Aglomeração

Segundo VIANA (2007) o coeficiente de aglomeração mede o quão conectado estão os nós da rede ou do grafo. ANTIQUEIRA et al. (2005) define o coeficiente de aglomeração sendo:

$$CA_i = \frac{E_i}{k_i(k_i - 1)}$$

ANTIQUEIRA et al. (2005) continua: "Sendo para cada vértice i existe k_i arestas, que os ligam a outros k_i vértices. Se esses k_i vértices estivessem ligados diretamente à todos os outros vértices do conjunto, haveriam $k_i(k_i - 1)$ arestas entre eles. E assumindo E_i o número de arestas que existentes entre os k_i vértices. O coeficiente da rede inteira é a média de todos CA_i".

Figura 3 – Coeficiente de aglomeração.



Fonte: Autor.

Na Figura 3a, assumindo o peso de todas as arestas como 1, o coeficiente de aglomeração do vértice em azul é 1, já na Figura 3b o coeficiente do vértice em azul segundo a fórmula é $\frac{1}{2}$.

2.3 *LEAGUE OF LEGENDS*

O jogo *League of Legends* é um jogo classificado como arena de batalha online de multi jogadores (do inglês *Multiplayer Online Battle Arena*) ou conhecido também como MOBA, que é um estilo de jogo onde duas equipes se enfrentam em um campo de batalha e cada jogador controla o seu personagem, mais chamado de herói ou campeão. O objetivo do MOBA é derrotar a equipe adversária destruindo a construção principal da equipe inimiga.

A arena onde acontece o jogo é uma arena onde normalmente o mapa inicialmente é espelhado, ou seja, o lado que cada time está não oferece vantagens exclusivas. O mapa é composto de três caminhos até a base inimiga.

Figura 4 – Mapa do jogo de *League of Legends*

Fonte: Autor (2018).

A Figura 4 mostra como é realmente o mapa do jogo, sendo que os círculos mostram a localização da construção principal, e os triângulos as construções de suporte de cada equipe, sendo azul uma equipe e vermelho a outra.

Com o início do jogo cada jogador escolhe um herói diferente, onde cada herói tem um conjunto de características únicas, como habilidades especiais, seu impacto no jogo, na equipe adversária e na equipe aliada.

Depois de escolher os heróis de cada equipe, cada jogador deve procurar adquirir recursos no jogo e objetivos para conseguir vantagens. Os recursos são limitados por equipe e por

tempo, ou seja, deve ser bem escolhido quem ficará com a maior parte dos recursos da equipe.

2.4 ESTADO DA ARTE

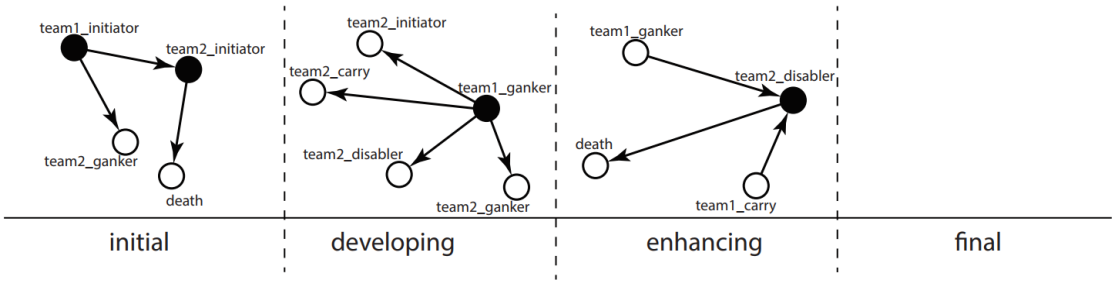
VOU REFAZER

No estado da arte, estudos com jogos de gênero MOBA são poucos, mas relacionados a redes complexas e predição existem diversos.

2.4.1 Identifying Patterns in Combat that are Predictive of success

O trabalho de Yang *et al*, relacionou a posição dos jogadores em uma batalha do MOBA com redes complexas, ao predizer o time vencedor daquela batalha.

Figura 5 – As estruturas gráficas de uma batalha de MOBA



Fonte: (YANG; HARRISON; ROBERTS, 2014).

Yang *et al* modela os dados de um combate, em redes complexas, e treina uma arvore de decisão usando as melhores características tiradas dessa rede complexa. Depois de treinado a árvore, é identificado regras de combate que são preditivas de sucesso, e depois essas regras são traduzidas de volta em um padrão de combate específico usando uma técnica chamada mineração de subgrafo frequente.

2.4.2 Análise de uma Métrica Alternativa para Predição de Laços Sociais em Grafos Lei de Potência

Feito por DANIELEWICZ, ela mostra uma visão geral do problema de predição de laços sociais, analisa os modelos de geração de grafos principalmente que seguem a lei da potência, no âmbito de formação de laços sociais e depois consegue uma métrica para predição de laços. Ela faz testes usando técnicas diferentes, mostrando seus resultados.

3 METODOLOGIA

Este capítulo apresenta as ferramentas e algoritmos utilizados neste projeto tanto no processo de extração do conhecimento, quanto na visualização dos dados. A primeira seção irá explicar como os dados estão disponíveis e de que informações eles carregam. A segunda clarifica o que é um banco de dados, e porque foi utilizado. A terceira fala de que maneira se consegue os dados, verifica se os dados estão completos e armazena-os em um banco de dados.

3.1 API DO *LEAGUE OF LEGENDS*

A Riot Games, desenvolvedora e dono do jogo *League of Legends*, fornece uma Interface de Programação de Aplicativos (do inglês *Application Programming Interface*) ou chamado de API, para que terceiros consigam acessar dados sobre os jogos.

A Riot games disponibiliza o acesso as informações à patir da URL[1] de modo que é gerado uma chave válida por 1 ano para projetos cadastrados. O acesso à essa API é por URL utilizando uma função disponível pela Riot Games. Neste trabalho, usaremos apenas a função *MATCH-V3* que é uma função que retorna os dados de uma partida já terminada.

A função *MATCH-V3* é disponibilizada publicamente pela Riot Games, retornando um conjunto de informações no formato JSON sobre a partida passada por parâmetro, quando essa partida existe. A Figura 6 resume um exemplo de uso da função acessando a URL[2], acessada em 21 de março de 2018 pelo autor, sendo que $minhachave$ tem que ser substituída por uma chave privada válida.

[1] http://developer.riotgames.com
[2] https://br1.api.riotgames.com/lol/match/v3/matches/1381102031?api_key=minhachave

Figura 6 – Exemplo de retorno do uso da função *MATCH-V3*. Sendo as informações divididas em informações da partida, informações dos times e informações dos participantes.


```
"gameId":1381102031,
"platformId":"BR1",
"gameCreation":1526653171409,
"gameDuration":1533,
"queueId":420,
"mapId":11,
"seasonId":11,
"gameVersion":"8.10.229.7328",
"gameMode":"CLASSIC",
"gameType":"MATCHED_GAME",
"teams":[ ⊟
   { ⊟
      "teamId":100,
      "win":"Win",
      "firstBlood":true,
      "firstTower":true,
      "firstInhibitor":true,
      "firstBaron":true,
      "firstDragon":true,
      "firstRiftHerald":true,
      "towerKills":10,
      "inhibitorKills":2,
      "baronKills":1,
      "dragonKills":2,
      "vilemawKills":0,
      "riftHeraldKills":1,
      "dominionVictoryScore":0,
      "bans":[ ⊞ ]
   },
   { ⊞ }
],
"participants":[ ⊟
   { ⊟
      "participantId":1,
      "teamId":100,
      "championId":420,
      "spell1Id":12,
      "spell2Id":4,
      "highestAchievedSeasonTier":"PLATINUM",
      "stats":{ ⊞ },
      "timeline":{ ⊞ }
   },
   { ⊞ },
```


Fonte: Autor.

De todas essas informações retornadas pela função, as que foram armazenadas para o projeto por participante são:

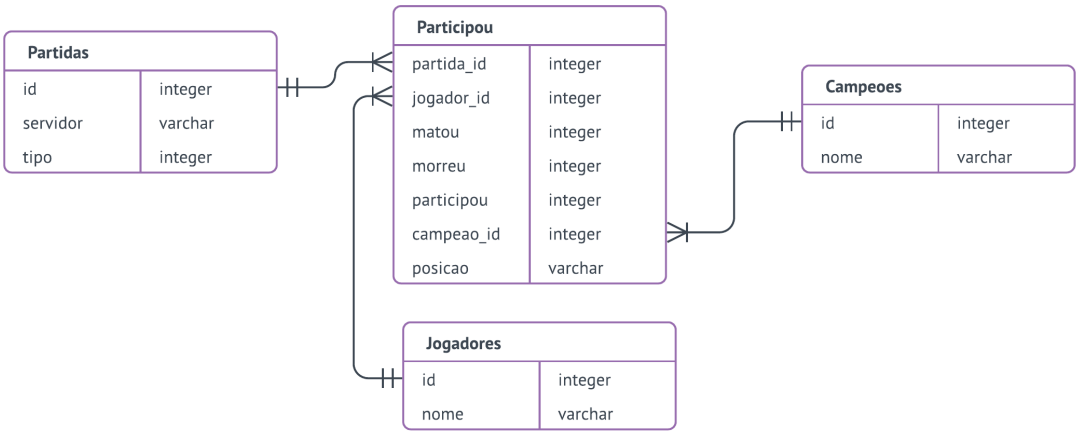
1. *gameId*. Identificador único da partida;
2. *kills*, *deaths* e *assists*. Informações que falam, respectivamente, quantas vezes esse jogador matou, morreu e participou na morte de outrem;
3. *win*. Se ele ganhou;
4. *championId*. Qual campeão ele escolheu;
5. *lane*. Em qual posição ele tava jogando;
6. *platformId*. Em qual servidor ele jogava;
7. *queueId*. E qual o tipo de partida.

Sabendo qual informação vamos armazenar, ficou escolhido a linguagem *Python* para ser desenvolvido o algoritmo no qual obterá as informações de forma não manual e o sistema de gerenciamento de banco de dados MySQL. Na próxima seção será apresentado sobre banco de dados e na seção 3.3 será esclarecido como foram obtidos o *dataset* das partidas.

3.2 BANCO DE DADOS

(continua...)

Figura 7 – Esquema usado do Banco de Dados.



Fonte: Autor.

3.3 CONSEGUINDO OS DADOS

Python é uma linguagem de programação dentre muitas, assim como existem várias linguagens faladas e escritas por nós humanos, existem várias linguagens de programação onde geralmente cada uma se destaca em pelo menos um quesito.

Com isso, foi criado um algoritmo para a aquisição dos dados de partidas pela API do *League of Legends* em *Python*, que usa a biblioteca *requests* para conseguir acessar a API por URL's, e a biblioteca json para transformar o JSON retornado pela requisição da URL em um objeto em *Python*.

O aplicativo verifica se a partida requisitada existe e se os dados necessários estão completos, salvando no banco de dados apenas as partidas válidas. O pseudo-código se encontra no Algoritmo 5.

Algoritmo 1: AQUISIÇÃO DOS DADOS DAS PARTIDAS

```
início
    i = 1000000000; // Esta é a id de uma partida aleatória na
     ultima versão do jogo
    chave = minha chave privada de acesso;
    se Existe arquivo com ultimo partida lido então
        i = ultima partida lida;
    repita
        pedido = requisita URL Da API(i, chave);
        se pedido.status == 200 então (          // Se a partida existe
            JSON = carrega JSON Do Pedido(pedido);
            se JSON é válido então
                Armazena os dados em um banco de dados;
            i = i + 1;
        fim
    até Até ser pausado;
    Salva i no arquivo;
fim
```

Onde com esse algoritmo foram armazenadas 28597 jogos válidos diferentes, e foi decidido usar apenas as partidas ranqueadas 5 contra 5, já que estas são as partidas responsáveis pela classificação dos jogadores dentro dos jogo, e as partidas não ranqueadas são consideradas como amistosas. Com esse escopo, o número de partidas usados foram ao todo 11717 partidas ranqueadas diferentes. Na tabela 1 é possível ver uma pequena fatia dos dados salvos.

Tabela 1 – Exemplo de *subset* salvo no banco de dados

match_id	kills	deaths	assists	win	champ_id	lane	player_id	platform	type
1282000002	11	10	7	0	67	BOTTOM	16112724	BR1	420
1282000002	2	8	15	0	412	BOTTOM	18604874	BR1	420
1282000002	7	4	7	0	34	MIDDLE	19281809	BR1	420
1282000002	7	6	4	0	5	JUNGLE	21304223	BR1	420
1282000002	5	7	12	0	98	TOP	651014	BR1	420
1282000002	0	6	23	1	44	BOTTOM	18592234	BR1	420
1282000002	19	5	5	1	222	BOTTOM	7170345	BR1	420

Fonte: Autor.

Na etapa seguinte são contabilizados quantas vezes cada campeão jogou com outro campeão, seja no mesmo time ou no time adversário, calculando quantas vitórias em oposição e quantas vitórias em junção. E com as informações já processadas foi montado um *web service*, que será explicado na seção 3.5, utilizando uma biblioteca chamada D3.js para uma melhor visualização dos dados obtidos , biblioteca essa que será explanado na próxima seção.

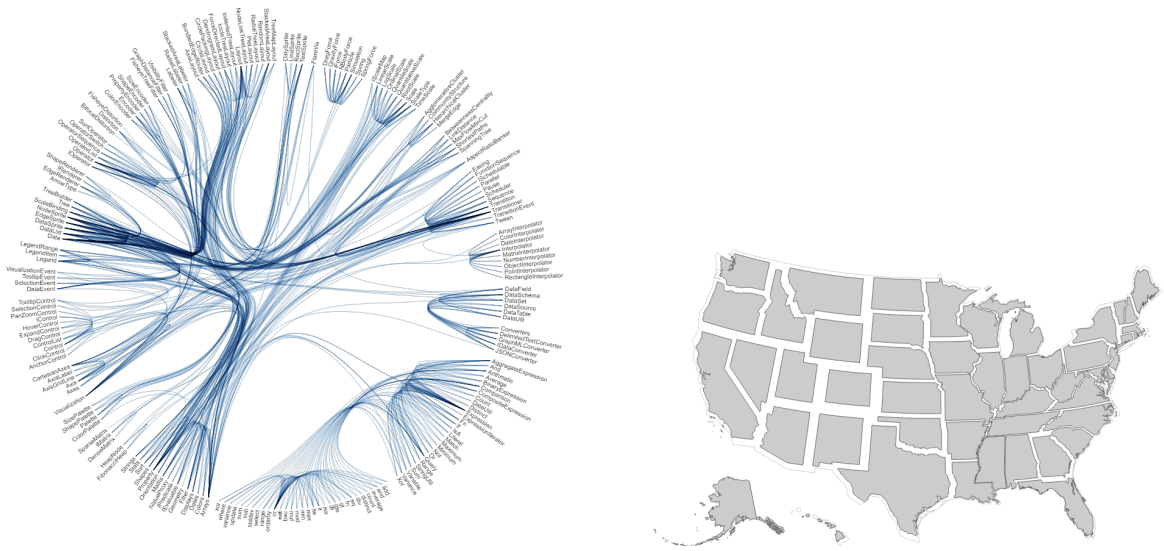
3.4 D3.JS

A biblioteca D3.js é uma biblioteca em JavaScript especializada em dar vida aos dados. Segundo ZHU (2013, tradução do autor) "Em certo sentido, o D3 é uma biblioteca JavaScript especializada que permite criar incríveis visualizações de dados usando uma abordagem mais simples (baseada em dados), aproveitando os padrões da *Web* existentes", e a organização oficial continua :

> O D3.js é uma biblioteca em JavaScript para manipulação de documentos por dados. D3 te ajuda a trazer vida para os dados utilizando HTML, SVG e CSS. A ênfase da biblioteca D3 nos padrões da *web* dá-lhe todos os recursos dos navegadores modernos sem te prender a um *framework* proprietário, combinando poderosos componentes de visualização e uma aproximação orientada por dados da manipulação do DOM. (BOSTOCK, 2016, tradução do autor)

Com essa biblioteca é possível dar vida à informações e dados, como por exemplo a Figura 8 que possui duas das diversas maneiras de se exibir informações com o D3.js . E com o D3 foi gerado o grafo do *web service*, programa este que será clarificado na seção seguinte.

Figura 8 – Exemplo de visualização de dados utilizando o D3.js.



Fonte: (BOSTOCK, 2016).

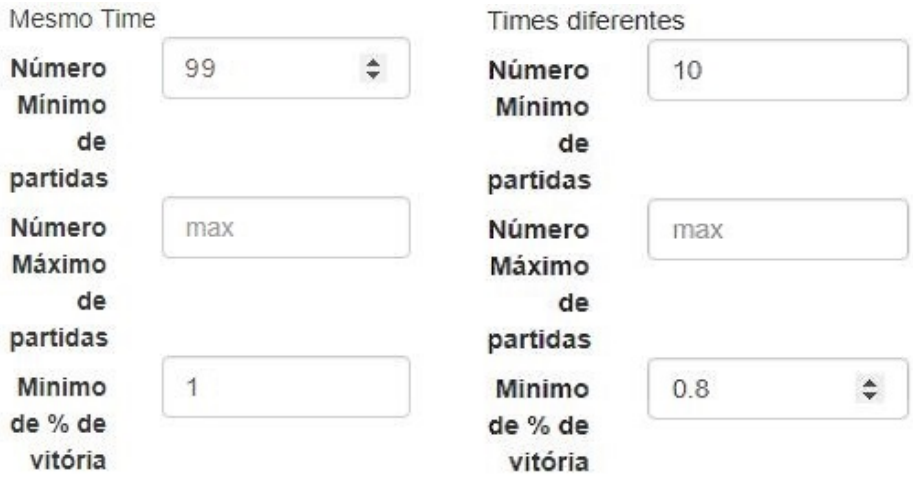
3.5 *WEB SERVICE*

O *web service* foi criado utilizando o *framework* Flask, que como RONACHER (2010, tradução do autor) diz "Flask é um micro *framework* para Python baseado em Werkzeug, Jinja 2 e em boas intenções.".

Com esse serviço será possível um relatório geral onde o usuário será capaz de ver quais são os campeões melhores contra os outros e quais são as duplas mais favoráveis, podendo filtrar quais campeões podem participar da pesquisa e quais não podem. Também será exequível a predição de uma partida, onde se seleciona os campeões de cada equipe e o sistema tenta antever quem vencerá.

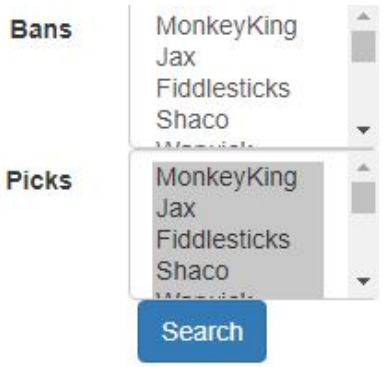
Na figura 9 é possível ver um exemplo de uso do serviço para uma visão geral das informações, já na figura 10 é possível ver o uso da ferramenta para predição de uma partida. Na seção 3.6 será esclarecido a maneira que essa predição é feita.

Figura 9 – Exemplo de uso do *web service* para visão geral dos dados.

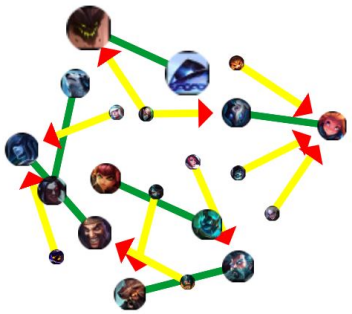


(a) Filtro dos dados para campeões da mesma equipe.

(b) Filtro dos dados para adversários.



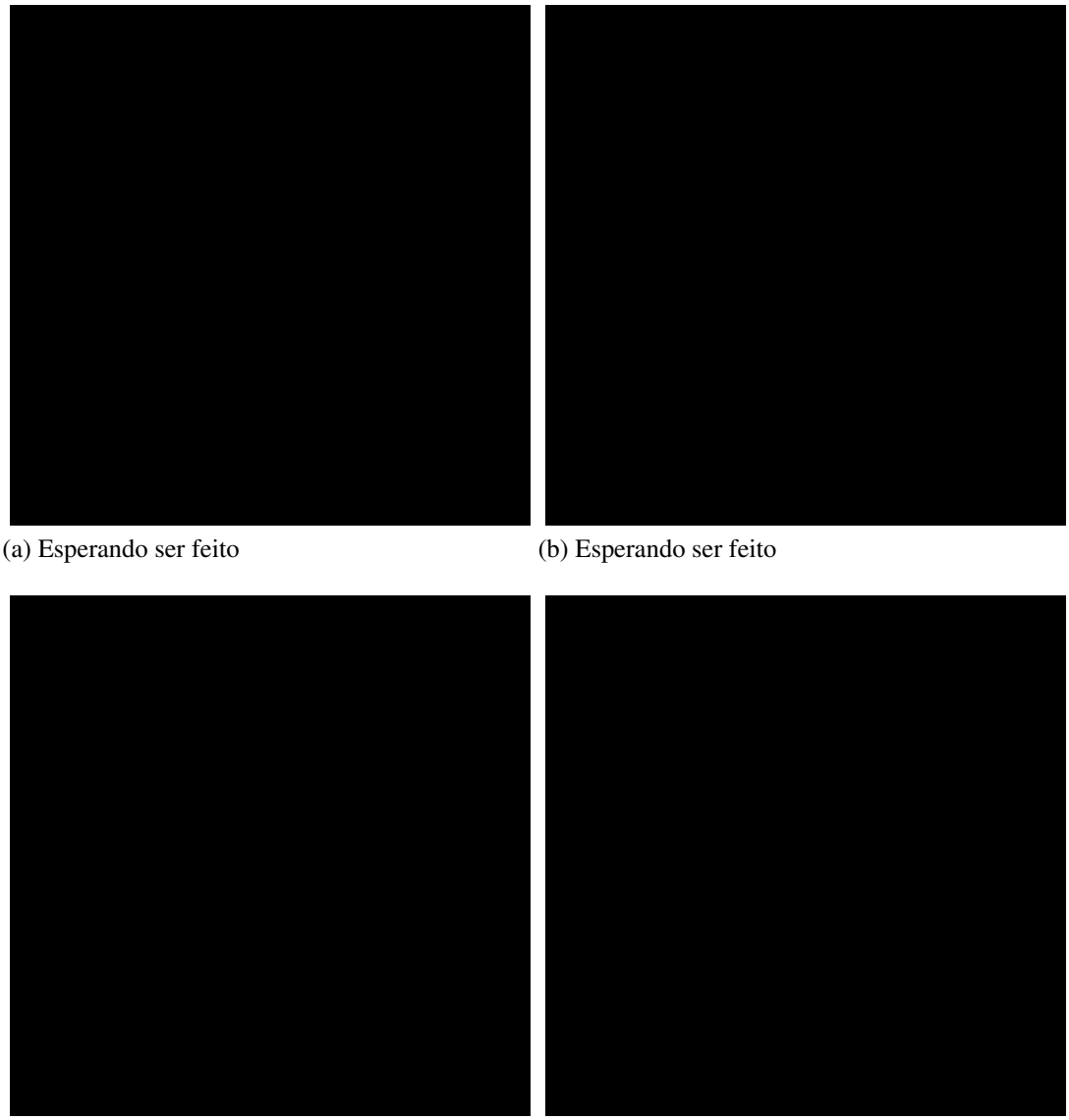
(c) Filtro para escolher quais campeões podem participar (*picks*) e quais nao podem (*bans*).



(d) Exemplo de grafo exibido pelo *web service*, sendo as setas vermelhas mostrando quem é bom contra quem, e as linhas continuas quem é bom com quem. O tamanho do circulo é a frequência daquela ligação e a cor é a força da ligação.

Fonte: Autor.

Figura 10 – Exemplo de uso do *web service* predição da partida.



(a) Esperando ser feito (b) Esperando ser feito

(c) Esperando ser feito (d) Esperando ser feito

Fonte: Autor.

3.6 PREDIÇÃO DOS DADOS

Como ainda não conseguimos dados suficientes, não chegamos até aqui.

Algoritmo 2: ALGORITMO PRINCIPAL

início
Entrada: *Dataset*, Nó raiz, Número de grupos
Saída: Salva os grupos
Raiz = vértice raiz;
N = numero de clusters;
G = Carrega vértices e arestas do arquivo de entrada;
MatrizAdjacencia = distancia_euclidiana(G);
AGM = prim(MatrizAdjacencia, Raiz)
Cluster = Cria_Clusters(AGM, N)
Salva os grupos no arquivo de saida;
fim

Algoritmo 3: Distância Euclidiana

início
Entrada: Arquivo .CSV
Saída: Matriz de Adjacência
fim

Algoritmo 4: Prim

início
Entrada: Matriz de Adjacência, Nó raiz
Saída: Arvore Geradora Mínima (AGM)
fim

Algoritmo 5: Clusterização

início
Entrada: AGM, Número de Grupos
Saída: Grupos separados
fim

4 CONCLUSÃO

Está é a conclusão do trabalho ....

REFERÊNCIAS

ALBERT, R.; BARABáSI, A.-L. Statistical mechanics of complex networks. **Reviews of modern physics**, [S.l.], v.74, n.1, p.47, 2002.

ANTIQUEIRA, L. et al. Modelando textos como redes complexas. In: III WORKSHOP EM TECNOLOGIA DA INFORMAçãO E DA LINGUAGEM HUMANA. **Anais...** [S.l.: s.n.], 2005. p.22–26.

BOSTOCK, M. D3. js-data-driven documents (2016). **URL: https://d3js. org**, [S.l.], 2016.

CUNHA RECUERO, R. da. Teoria das redes e redes sociais na internet: considerações sobre o orkut, os weblogs e os fotologs. In: XXVII CONGRESSO BRASILEIRO DE CIêNCIAS DA COMUNICAç AO. XXVII INTERCOM. **Anais...** [S.l.: s.n.], 2004.

DANIELEWICZ, G. Análise de uma métrica alternativa para predição de laços sociais em grafos lei de potência. **?**, [S.l.], 2016.

ESPN. **Premiação do Mundial de League of Legends ultrapassa US 4 milhões**. 2017.

FEOFILOFF, P.; KOHAYAKAWA, Y.; WAKABAYASHI, Y. Uma introdução sucinta à teoria dos grafos. , [S.l.], 2011.

LUCCHESI, C. L. **Introdução à teoria dos grafos**. [S.l.]: IMPA, 1979.

RONACHER, A. Welcome—flask (a python microframework). **URL: http://flask.pocoo.org/(visited on 02/04/2018)**, [S.l.], p.38, 2010.

VIANA, M. P. **Metodologia das redes complexas para caracterização do sistema de Havers**. 2007. Tese (Doutorado em Ciência da Computação) — Universidade de São Paulo.

YANG, P.; HARRISON, B. E.; ROBERTS, D. L. Identifying patterns in combat that are predictive of success in MOBA games. In: FDG. **Anais...** [S.l.: s.n.], 2014.

ZHU, N. Q. **Data visualization with D3. js cookbook**. [S.l.]: Packt Publishing Ltd, 2013.

APÊNDICES

.1 TESTE

ANEXOS

ANEXO A – Título do Anexo

Este é o anexo A