# PAA – Relatório do Trabalho Prático 3


**Danilo Ferreira e Silva**[1]

[1]Departamento de Ciência da Computação – UFMG

danilofs@dcc.ufmg.br


## 1. Introdução

Neste trabalho foi projetado um algoritmo para resolver o problema de encontrar times matematicamente eliminados de um campeonato, ou seja, que não possuem mais chances de ficar em primeiro lugar (ou empatado em pontos). Para tal, o algoritmo analisa o máximo de pontos que um time pode alcançar comparado aos seus adversários, considerando todos os jogos que ainda serão disputados. Um exemplo de entrada é dado na tabela 1.

**Tabela 1. Exemplo de entrada do problema**

| Time | Vitórias | Derrotas | A jogar | vs. A | vs. B | vs. C | vs. D |
|---|---|---|---|---|---|---|---|
| A | 83 | 71 | 8 | 0 | 1 | 6 | 1 |
| B | 80 | 79 | 3 | 1 | 0 | 0 | 2 |
| C | 78 | 78 | 6 | 6 | 0 | 0 | 0 |
| D | 77 | 82 | 3 | 1 | 2 | 0 | 0 |

## 2. Modelagem da solução

Seja $T = \{t_1, t_2, ..., t_n\}$ o conjunto de $n$ times da divisão. Seja $t_x$ o time de interesse que deseja-se verificar a possibilidade matemática. O conjunto dos concorrentes a $t_x$, ou seja, os times de $T$ com exceção de $t_x$, será chamado de $T^x$.

Como estamos interessados na existência de alguma chance de $t_x$ não ser superado por outro time, podemos considerar as seguintes condições como verdadeiras, pois elas sempre levam ao melhor cenário para $t_x$:

- $t_x$ ganhará todas as partidas que restam a ele;
- os demais times perderão todas as partidas que não sejam contra times da mesma divisão.

Com isso limitamos nossa análise apenas para o conjunto de jogos $J$ que sejam entre times $i, j \in R$. Todos estes jogos terão algum vencedor, e estes pontos serão distribuídos a algum time dentro do próprio conjunto $T^x$.

O problema pode ser modelado como uma rede de fluxo onde desejamos distribuir $|J|$ vitórias para os times $T^x$. No entanto, desejamos restringir essa distribuição de pontos de tal forma que nenhum time supere $t_x$.

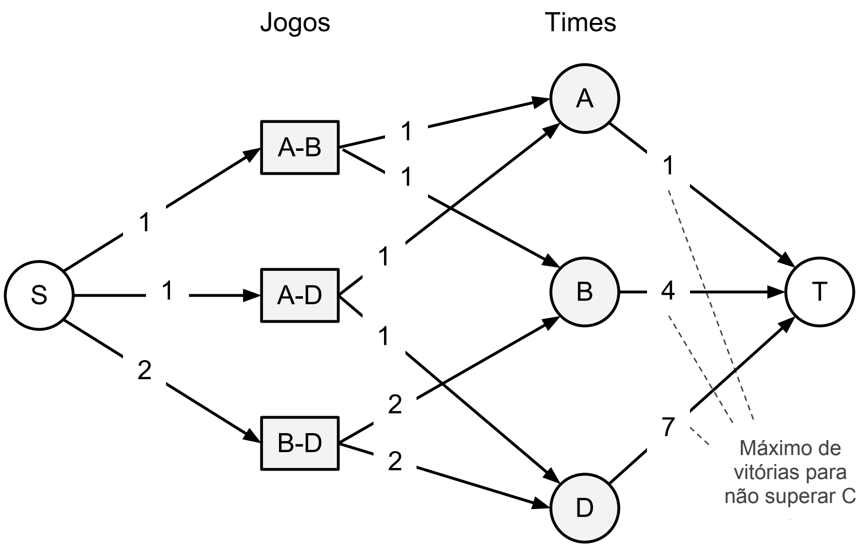


**Figura 1. Problema da tabela 1 modelado como uma rede de fluxos**

No figura 1 temos o grafo da rede de fluxos associada ao problema dado na tabela 1, onde analisamos as chances do time C. Partindo do vértice $s$ temos arestas que ligam a cada partida que deverá ser disputada, onde a capacidade da aresta é o número de partidas. Saindo de cada partida temos duas arestas que levam aos times que a disputam. A princípio qualquer um dois dois times pode ganhar todas as partidas, portanto a capacidade de ambas as arestas é a quantidade de jogos disputados.

Finalmente, saindo de cada time temos uma única aresta ligada ao destino $t$. A capacidade destas arestas deve ser atribuída de forma que o time não possa ganhar mais partidas que $t_x$. No exemplo dado, C atinge no máximo 84 vitórias. Portanto B, que já possui 80, não pode ganhar mais do que 4 partidas.

**O time analisado está eliminado se, e somente se, o fluxo máximo da rede for menor que** $|J|$. Se o fluxo for igual a $|J|$ existe uma atribuição de resultados a todas as partidas de $J$ em que $t_x$ não é eliminado.

### 2.1. Justificativa para a eliminação

Outra informação que podemos obter da rede de fluxos no caso de eliminação é o conjunto de times $R$ que terão um número de vitórias agregadas tal que algum deles garantidamente superará $t_x$.

Seja $C = (S, T)$ uma partição de $V$, tal que $C$ é um corte s–t mínimo da rede de fluxos. Sabemos que a capacidade do corte mínimo corresponde ao fluxo máximo da rede. Vamos definir $E_c$ como o conjunto de arestas $\{e(u, v) \in E \mid u \in S, v \in T\}$, ou seja as arestas cuja origem está em $S$ e destino está em $T$.

Podemos dizer que qualquer aresta que não esteja no conjunto $E_c$ pode ter sua capacidade aumentada e, ainda assim, o fluxo máximo se manteria o mesmo, visto que a capacidade do corte mínimo se manteve.

Explorando este fato, suponha que exista uma aresta $e(t_i, t) \notin E_c$ , onde $t_i$ é um time qualquer e $t$ é o vértice destino. Mesmo se aumentássemos indefinidamente a capacidade da mesma, ou seja, o número de partidas que $t_i$ pudesse ganhar, $t_x$ ainda seria eliminado. Dessa forma, podemos afirmar que $t_i$ não é necessário para explicar a eliminação de $t_x$.

Começando com todos os times em $R$ e removendo todo time $t_i$ tal que $e(t_i, t) \notin E_c$, ficamos com o conjunto de times que explicam a eliminação de $t_x$.

## 3. Análise de complexidade

Como discutido acima, verificar a eliminação de um time $t_x$ envolve a construção de uma rede de fluxos e encontrar o fluxo máximo da mesma. Isso foi feito utilizando o algoritmo Edmonds-Karp, uma variante do Ford-Fulkerson que usa a busca em largura para encontrar caminhos de aumento. A complexidade do mesmo é $O(VE^2)$.

Dada uma entrada com $n$ times, o quantidade máxima de confrontos distintos entre eles é $(n-1)^2/2$. Portanto, cada em cada grafo construído temos:

$$\begin{aligned} V &= 1 + (n-2)^2/2 + (n-1) + 1 = (1/2)n^2 + n - 1 = O(n^2) \\ E &= 3(n-2)^2/2 + (n-1) = (3/2)n^2 + n - 7 = O(n^2) \end{aligned}$$

Portanto, o custo total no pior caso é:

$$\begin{aligned} C &= nO(VE^2) \\ &= nO(n^2(n^2)^2) \\ &= O(n^7) \end{aligned}$$

Na figura 2 temos o gráfico do tempo de execução pelo tamanho da entrada, expresso pelo número de times $n$. Nele podemos observar o crescimento rápido da curva, que é condizente com a complexidade $O(n^7)$.

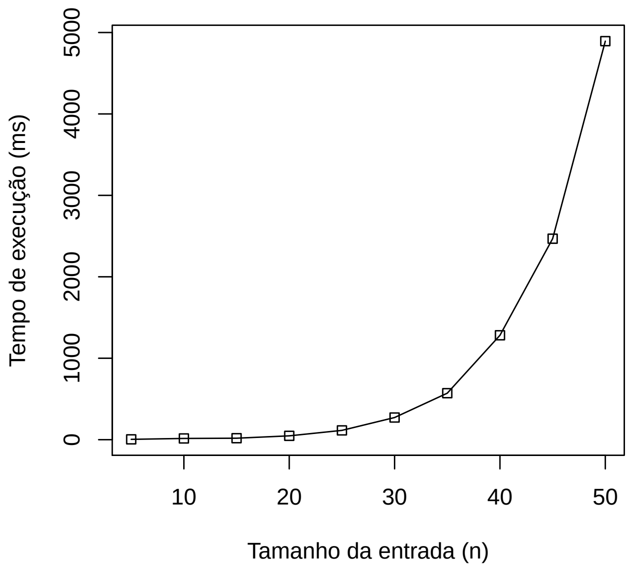


**Figura 2. Tempo de execução por número de times** $n$

Em termos de espaço, o custo é apenas $O(n^2)$, tanto para armazenar a entrada quando o grafo necessário.

## 4. Conclusão

Neste trabalho foi desenvolvido um algoritmo para detectar times eliminados em um campeonato. Para tal o problema foi modelado em um grafo que expressa uma rede de fluxos. O problema de verificar se um time estava eliminado foi então reduzido em encontrar o fluxo máximo deste grafo.

A complexidade final do algoritmo foi $O(n^7)$. Em experimentos foi constatado que o crescimento do tempo de execução de fato é muito elevado. Embora o algoritmo seja polinomial, ele começa a se tornar inviável para entradas cujo o número de times seja mais do que algumas dezenas.