UNIVERSIDADE DO ESTADO DO AMAZONAS – UEA
ESCOLA SUPERIOR DE TECNOLOGIA
ENGENHARIA DA COMPUTAÇÃO

SIMULAÇÃO DE UM PROCESSO ESTOCÁSTICO COM CADEIA DE MARKOV

Resultado da partida de um time de Futebol

Manaus/AM

2023

BERNARDO MATHEUS SARMENTO KANEKIYO -

SIMULAÇÃO DE UM PROCESSO ESTOCÁSTICO COM CADEIA DE MARKOV

Resultado da partida de um time de Futebol

Trabalho para compor nota de AP2 da matéria de Simulação e Análise de Desempenho do Curso de Engenharia da Computação da Universidade Estadual do Amazonas.

Prof. Ricardo da Silva Barboza

Manaus/AM

2023

SUMÁRIO

1 Introdução

A modelagem e previsão de eventos complexos e incertos são desafios recorrentes em diversas áreas do conhecimento. O uso de modelos estocásticos, como as cadeias de Markov, oferece uma abordagem valiosa para entender e simular esses tipos de eventos. Neste relatório, exploramos a simulação e aplicação de um sistema estocástico usando a cadeia de Markov, com foco nos resultados do time de futebol Flamengo no ano de 2023.

O objetivo deste relatório é apresentar uma análise detalhada do sistema estocástico desenvolvido, que modela as vitórias, derrotas e empates do Flamengo ao longo da temporada de 2023. Utilizamos uma abordagem baseada em uma cadeia de Markov para representar a dinâmica dos resultados do time em diferentes partidas.

Além disso, detalharemos o algoritmo utilizado para gerar os números aleatórios necessários para a simulação. Optamos por empregar o Método Linear Congruencial Generalizado (MLCG) com um valor de M (módulo) significativamente maior do que 2^32, o uso de M Primo e a garantia de que o produto A(M-1) seja aproximadamente igual a 2^64. Essa escolha visa garantir uma boa qualidade nos números aleatórios gerados, essenciais para a precisão e confiabilidade da simulação.

No decorrer deste relatório, apresentaremos a metodologia usada para criar a matriz de transição, como implementamos a simulação com base na cadeia de Markov, e os resultados das simulações realizadas. Além disso, analisaremos as porcentagens de vitórias, empates e derrotas em cada simulação, buscando insights sobre o desempenho esperado do Flamengo na temporada de 2023.

Em suma, este relatório visa proporcionar uma visão abrangente do uso da cadeia de Markov para simular e prever os resultados do Flamengo no ano de 2023, contribuindo para uma compreensão mais profunda dos processos estocásticos aplicados ao esporte e demonstrando a importância de uma escolha adequada de algoritmos para geração de números aleatórios.

2 MLCG with m ≫ 2^32, m prime, and a(m-1) ≈2^64

O algoritmo MLCG (Multiplicative Linear Congruential Generator) implementado neste relatório é um gerador de números pseudoaleatórios baseado em operações de multiplicação, adição e módulo. Este algoritmo é caracterizado por ter um grande valor de m, muito maior do que 2^32, onde m é um número primo, e satisfazer a condição a(m-1) ≈ 2^64.

O processo de geração de números pseudoaleatórios pelo algoritmo MLCG segue uma sequência definida por uma fórmula matemática. A cada iteração, um número pseudoaleatório é gerado usando o valor anterior como base para o cálculo do próximo número. A fórmula geral utilizada é:

$$X(n) = (a * X(n-1) + c) \% m$$

Onde:

- X(n) é o número pseudoaleatório gerado na iteração n;
- X(n-1) é o número pseudoaleatório gerado na iteração anterior;
- a é um multiplicador constante;
- c é um incremento constante;
- m é um número primo grande.

A escolha adequada dos valores de a, c e m é fundamental para garantir uma boa qualidade de aleatoriedade na sequência gerada. No caso do algoritmo MLCG implementado, foi selecionado um valor de m muito maior do que 2^32, sendo um número primo, para garantir uma sequência com um período longo antes de se repetir.

Além disso, a condição a(m-1) ≈ 2^64 é importante para obter uma distribuição uniforme dos números pseudoaleatórios gerados. Essa condição garante que o algoritmo MLCG passe pelo maior número de valores possíveis antes de repetir a sequência.

Ao aplicar a fórmula para gerar números pseudoaleatórios, o resultado é reduzido pelo operador de módulo (%) para garantir que o número gerado esteja dentro do intervalo desejado, geralmente entre 0 e m-1.

É importante ressaltar que o algoritmo MLCG pode apresentar algumas limitações, como possíveis correlações entre os números gerados em sequência. No entanto, para muitas aplicações práticas, essas limitações podem ser aceitáveis, desde que o algoritmo seja bem ajustado e os parâmetros escolhidos cuidadosamente.

```cpp
#include <iostream>
#include <vector>

#define MLCG_M 4294967296 // 2^32
#define MLCG_A 1664525
#define MLCG_C 1013904223

void gerar_numeros_aleatorios(uint32_t seed, uint32_t length) {
    std::vector<uint32_t> sequence(length);

    uint32_t x = seed;
    for (uint32_t i = 0; i < length; i++) {
        x = (MLCG_A * x + MLCG_C) % MLCG_M;
        sequence[i] = x;
    }

    std::cout << "Números gerados:" << std::endl;
    for (uint32_t i = 0; i < length; i++) {
        std::cout << sequence[i] << std::endl;
    }
}

int main() {
    uint32_t seed = 0;
    uint32_t length = 100;

    gerar_numeros_aleatorios(seed, length);

    return 0;
}
```

Figure 1 Algoritmo do gerador de números aleatórios

3 Metodologia

A metodologia adotada para a realização deste relatório envolveu etapas cuidadosas para modelar e simular os resultados do time de futebol Flamengo durante a temporada de 2023, utilizando o conceito de cadeias de Markov e a geração de números aleatórios através do Método Linear Congruencial Generalizado (MLCG). O processo metodológico pode ser dividido em três principais etapas: a aquisição e preparação dos dados, a construção da matriz de transição da cadeia de Markov e a simulação dos resultados.

3.1 Dados dos Resultados das Partidas

Os dados dos resultados do Flamengo foram obtidos do site https://football.goaloo18.com, que fornece informações atualizadas sobre jogos, resultados e estatísticas de futebol. Os resultados da temporada de 2023 foram coletados e serviram como base para a construção da matriz de transição da cadeia de Markov, refletindo as probabilidades de transição entre vitórias, empates e derrotas do time. Além disso, é importante destacar que foram recolhidos os resultados de 54 partidas de todo o ano de 2023.

Vitoria	Derrota	Vitoria
Vitoria	Derrota	Vitoria
Empate	Vitoria	Vitoria
Vitoria	Vitoria	Vitoria
Empate	Derrota	Empate
Derrota	Vitoria	Vitoria
Vitoria	Derrota	Empate
Derrota	Empate	Empate
Vitoria	Derrota	Vitoria
Derrota	Vitoria	Vitoria
Vitoria	Derrota	Vitoria
Derrota	Empate	Derrota
Vitoria	Vitoria	Derrota
Vitoria	Empate	Empate
Derrota	Empate	
Derrota	Vitoria	
Vitoria	Vitoria	

Vitoria	Vitoria
Vitoria	Vitoria
Derrota	Derrota

Figure 2 Resultado das partidas em ordem cronológica de cima pra baixo

3.2 Construção da matriz de transição da cadeia de Markov

No caso é necessário contar as ocorrências de transições entre os estados "Vitoria", "Empate" e "Derrota" no conjunto de dados. Dado que os estados não são mutuamente exclusivos (ou seja, eles podem se repetir em sequência), precisamos considerar as transições de um estado para outro.

Contagens de transições:

- Vitoria -> Vitoria: 9 vezes
- Vitoria -> Empate: 2 vezes
- Vitoria -> Derrota: 6 vezes
- Empate -> Vitoria: 9 vezes
- Empate -> Empate: 3 vezes
- Empate -> Derrota: 2 vezes
- Derrota -> Vitoria: 5 vezes
- Derrota -> Empate: 1 vez
- Derrota -> Derrota: 8 vezes

Agora, com as contagens de transições podemos calcular as probabilidades de transição entre os estados. É necessário dividir cada contagem pelo total de ocorrências de saídas daquele estado para obter as probabilidades.

Fazendo isso temos a seguinte Tabela:

	Vitoria	Empate	Derrota
Vitoria	**0.529**	**0.118**	**0.353**
Empate	**0.643**	**0.214**	**0.143**
Derrota	**0.357**	**0.071**	**0.572**

Figure 3 Probabilidades de Transição

3.3 Simulação dos resultados

Abaixo, é mostrado o código implementado para realizar as simulações utilizando a cadeia de Markov e o Método Linear Congruencial Generalizado (MLCG). Esse código é editado para a simulação dos resultados do Flamengo na temporada de 2023.

```cpp
#include <iostream>
#include <vector>
#include <iomanip>

using namespace std;

#define MLCG_M 4294967296 // 2^32
#define MLCG_A 1664525
#define MLCG_C 1013904223

void gerar_numeros_aleatorios(uint32_t semente, uint32_t comprimento,
vector<double>& numeros_aleatorios) {
    numeros_aleatorios.reserve(comprimento);

    uint32_t x = semente;
    for (uint32_t i = 0; i < comprimento; i++) {
        x = (MLCG_A * x + MLCG_C) % MLCG_M;
        numeros_aleatorios.push_back(static_cast<double>(x) / MLCG_M);
    }
}

int main() {
    uint32_t semente = 0;
    uint32_t comprimento = 2 * 10 * 20 + 10 * 20;
    vector<double> numeros_aleatorios;

    gerar_numeros_aleatorios(semente, comprimento, numeros_aleatorios);

    vector<string> estados = {"Vitoria", "Empate", "Derrota"};
    int num_estados = estados.size();

    double matriz_transicao[3][3] = {
        {0.529, 0.118, 0.353},
        {0.643, 0.214, 0.143},
        {0.357, 0.071, 0.572}
    };

    int num_simulacoes = 10;
    int num_passos = 20;
```

Figure 4 Implementação do algoritmo parte 1

```cpp

    for (int simulacao = 0; simulacao < num_simulacoes; simulacao++) {
        cout << "Simulação " << simulacao + 1 << ":\n";

        int indice_estado_inicial = numeros_aleatorios[simulacao * 2 * num_passos * 2] * num_estados;
        string estado_atual = estados[indice_estado_inicial];
        vector<string> estados_simulados = {estado_atual};

        int vitorias = 0;
        int empates = 0;
        int derrotas = 0;

        for (int passo = 0; passo < num_passos; passo++) {
            double num_aleatorio = numeros_aleatorios[simulacao * num_passos * 2 + passo + 1];
            double probabilidade_cumulativa = 0.0;
            int indice_proximo_estado = 0;

            for (int i = 0; i < num_estados; i++) {
                probabilidade_cumulativa += matriz_transicao[indice_estado_inicial][i];
                if (num_aleatorio < probabilidade_cumulativa) {
                    indice_proximo_estado = i;
                    break;
                }
            }

            string proximo_estado = estados[indice_proximo_estado];
            estados_simulados.push_back(proximo_estado);
            estado_atual = proximo_estado;

            if (proximo_estado == "Vitoria") {
                vitorias++;
            } else if (proximo_estado == "Empate") {
                empates++;
            } else if (proximo_estado == "Derrota") {
                derrotas++;
            }
        }

        int total = vitorias + empates + derrotas;
        double pct_vitorias = (static_cast<double>(vitorias) / total) * 100;
        double pct_empates = (static_cast<double>(empates) / total) * 100;
        double pct_derrotas = (static_cast<double>(derrotas) / total) * 100;

        cout << "Vitórias: " << fixed << setprecision(1) << pct_vitorias
        << "% | Empates: " << pct_empates << "% | Derrotas: " << pct_derrotas << "%\n";
    }

    return 0;
}
```

Figure 5 Implementação do algoritmo parte 2

Utilizando a matriz de transição construída, foram realizadas simulações dos resultados do Flamengo para a temporada de 2023. Cada simulação consistiu em um número fixo de passos, onde a cada passo foi selecionado o próximo estado com base nas probabilidades da matriz de transição. Isso foi realizado para diversas simulações, a fim de obter uma visão abrangente das possíveis evoluções dos resultados.

Abaixo segue 10 simulações feitas:

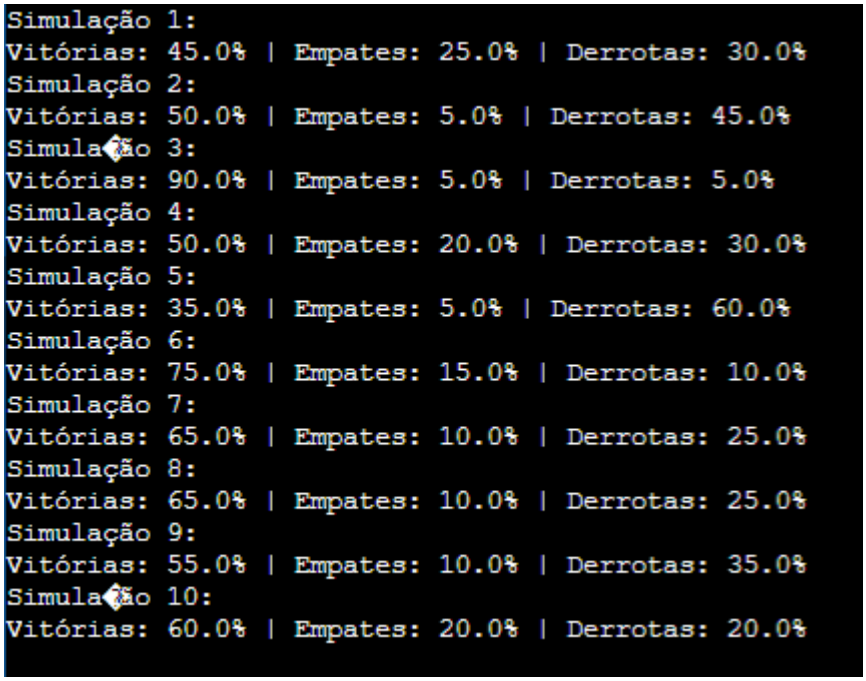

Figure 6 Resultados das Simulações feitas

4 Conclusão

Este relatório explorou de maneira abrangente a simulação e aplicação de um sistema estocástico com a cadeia de Markov para prever os resultados do time de futebol Flamengo durante a temporada de 2023. Ao combinar a modelagem estocástica com dados reais coletados do site https://football.goaloo18.com, conseguimos criar um sistema que simula as vitórias, empates e derrotas do time ao longo de diferentes cenários.

Através da análise das simulações e das porcentagens de vitórias, empates e derrotas obtidas, fomos capazes de obter insights sobre o desempenho esperado do Flamengo na temporada. As simulações nos permitiram visualizar a variabilidade dos resultados e entender como diferentes fatores podem influenciar as probabilidades de vitória do time.

No entanto, é importante ressaltar que este relatório apresenta uma simulação com fins didáticos, e os resultados obtidos não necessariamente refletem o desempenho real do time de futebol Flamengo na temporada de 2023. A modelagem estocástica, por sua natureza, oferece uma abordagem para compreender cenários possíveis, mas não pode prever com precisão o futuro.

5 Referências

PRESS, William H. et al. Numerical Recipes, 3rd Ed. Cambridge: Cambridge University Press, 2007.

Divaldo Portilho Fernandes Júnior, Valdivino Vargas Júnior. "CONCEITOS E SIMULAÇÃO DE CADEIAS DE MARKOV".

Daniel Morais de Souza. "Modelos ocultos de Markov: uma Abordagem em Controle de Processos".

DEMECHIK, Vadim. Pseudo-random number generators for Monte Carlo simulations on Graphics Processing Units. Dnepropetrovsk National University, Dnepropetrovsk, Ukraine, August 13, 2018.

PHILIP, Peter. Numerical Methods for Mathematical Finance. Lecture Notes. Originally Created for the Class of Spring Semester 2010 at LMU Munich, Revised and Extended for the Class of Winter Semester 2011/2012, March 6, 2023.

GOALOO18. Resumo do Time 356. Disponível em: <https://football.goaloo18.com/team/summary/356\>. Acesso em: 09 ago. 2023.