

Bacharelado em Engenharia da Computação

DISCIPLINA: Arquitetura de Computadores II

PROFESSORA: Fischer Jônatas Ferreira

# Simulador de Memória Cache

# Mapeamento de Cache

Daniel Henrique  
Guilherme Gomes  
Matheus Nogueira

**Divinópolis – MG**

**Maio - 2018**

## 1. Introdução

### 1.1. Conceitos

O conceito de hierarquia de memória veio para suprir a necessidade de aumentar o desempenho dos computadores sem elevar seu preço consideravelmente, visto que uma memória muito rápida também se torna muito cara. Utilizando desse conceito memórias muito rápidas (cache) são colocadas em pequenas quantidades, porém mais próximas do processador e mapeiam as memórias mais lentas que por sua vez ficam mais distantes do processador, assim ao executar um processo, muitas informações estarão próximas ao processador e o ganho em performance é elevado. Os tipos de mapeamento de cache são:

**Mapeamento Direto:** Associa várias pequenas partes da memória principal a uma pequena parte da cache.

**Mapeamento Totalmente associativo:** Qualquer informação da memória principal pode ocupar qualquer parte da cache. Políticas de substituição são necessárias.

**Mapeamento Associativo por Conjunto:** Associa várias pequenas partes da memória principal a conjuntos de pedaços da cache. Políticas de substituição são necessárias.

Políticas de substituição são diretrizes do algoritmo responsáveis por escolher qual dado deve ser substituído na cache quando necessário. São elas:

- LRU: Menos recentemente usado;
- LFU: Menos frequentemente usado;
- FIFO: O primeiro que entra é o primeiro que sai;
- LIFO: O último que entra é o primeiro que sai.

Mais dois conceitos importantes para o trabalho:

- Hit: Quando o dado requisitado pelo processador já se encontra na cache, não é necessário procurar na memória principal;
- Miss: Quando o dado requisitado pelo processador não se encontra na cache, é necessário procurar na memória principal e carregar o dado para a cache, utilizando das políticas de substituição se necessário.

### 1.2. Objetivo

O objetivo deste trabalho prático é familiarizar com os principais conceitos estudados sobre Hierarquia de Memória. Foram estudados três tipos de projeto de organização da cache (Mapeamento direto, associativo por conjunto e totalmente associativo). Este trabalho destina-se a utilizar de forma prática esses projetos de cache. Dessa forma, será necessário a implementação desses tipos de cache e a análise da execução para grupos de entradas previamente fornecidos.

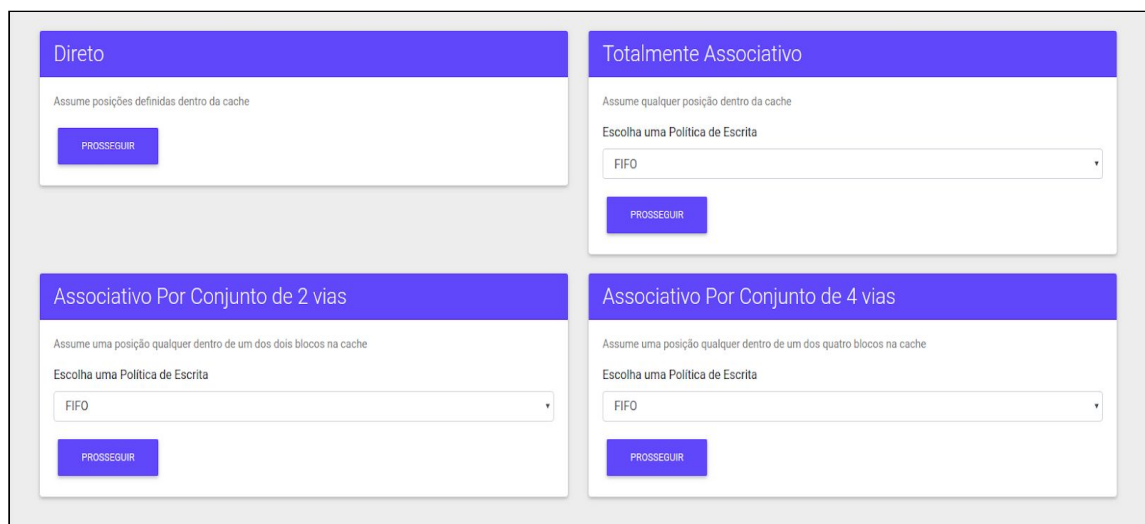
### 1.3. Funcionamento do Simulador

O simulador deverá ler dois arquivos, um deles será a memória principal e o outro será o programa a ser executado. Ele deve processar os dados de acordo com os tipos de mapeamento e computar as taxas de hit e miss, para assim facilitar a análise de cada “programa” em cada tipo de mapeamento.

## 2. Implementação

Para a implementação do trabalho, foi desenvolvido um sistema web, utilizando-se de linguagem principal baseada em JavaScript, através do framework Angular 5, que utiliza TypeScript (um *port* do JavaScript ECMA 16 mais moderno e com conceitos de programação ligadas a orientação de objetos).

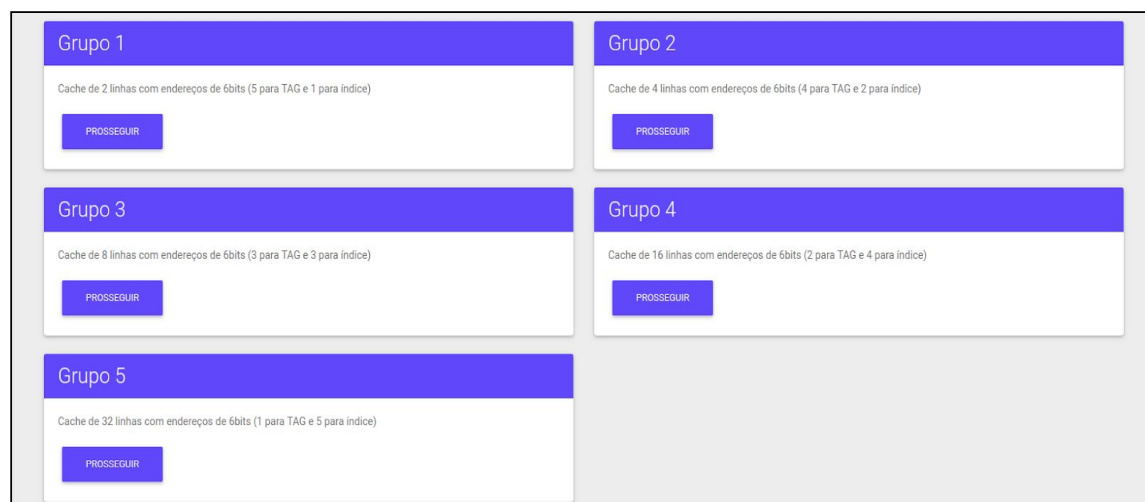
Foram escolhidas duas políticas a serem usadas nas caches Totalmente Associativa e Associativa por Conjunto, sendo elas a FIFO e a LIFO. Após escolher o método, é possível escolher o grupo e gerar os resultados dos programas referentes de cada, com gráficos e tabela dos hits e miss.



A interface apresenta quatro painéis de configuração para diferentes tipos de cache:

- Direto:** Assume posições definidas dentro da cache. Botão: PROSSEGUIR.
- Totalmente Associativo:** Assume qualquer posição dentro da cache. Escolha uma Política de Escrita: FIFO (dropdown). Botão: PROSSEGUIR.
- Associativo Por Conjunto de 2 vias:** Assume uma posição qualquer dentro de um dos dois blocos na cache. Escolha uma Política de Escrita: FIFO (dropdown). Botão: PROSSEGUIR.
- Associativo Por Conjunto de 4 vias:** Assume uma posição qualquer dentro de um dos quatro blocos na cache. Escolha uma Política de Escrita: FIFO (dropdown). Botão: PROSSEGUIR.

Figura 1: Tela para escolha do Método de cache



A interface apresenta cinco painéis de configuração para diferentes grupos de cache:

- Grupo 1:** Cache de 2 linhas com endereços de 6bits (5 para TAG e 1 para índice). Botão: PROSSEGUIR.
- Grupo 2:** Cache de 4 linhas com endereços de 6bits (4 para TAG e 2 para índice). Botão: PROSSEGUIR.
- Grupo 3:** Cache de 8 linhas com endereços de 6bits (3 para TAG e 3 para índice). Botão: PROSSEGUIR.
- Grupo 4:** Cache de 16 linhas com endereços de 6bits (2 para TAG e 4 para índice). Botão: PROSSEGUIR.
- Grupo 5:** Cache de 32 linhas com endereços de 6bits (1 para TAG e 5 para índice). Botão: PROSSEGUIR.

Figura 2: Tela para escolha do Grupo

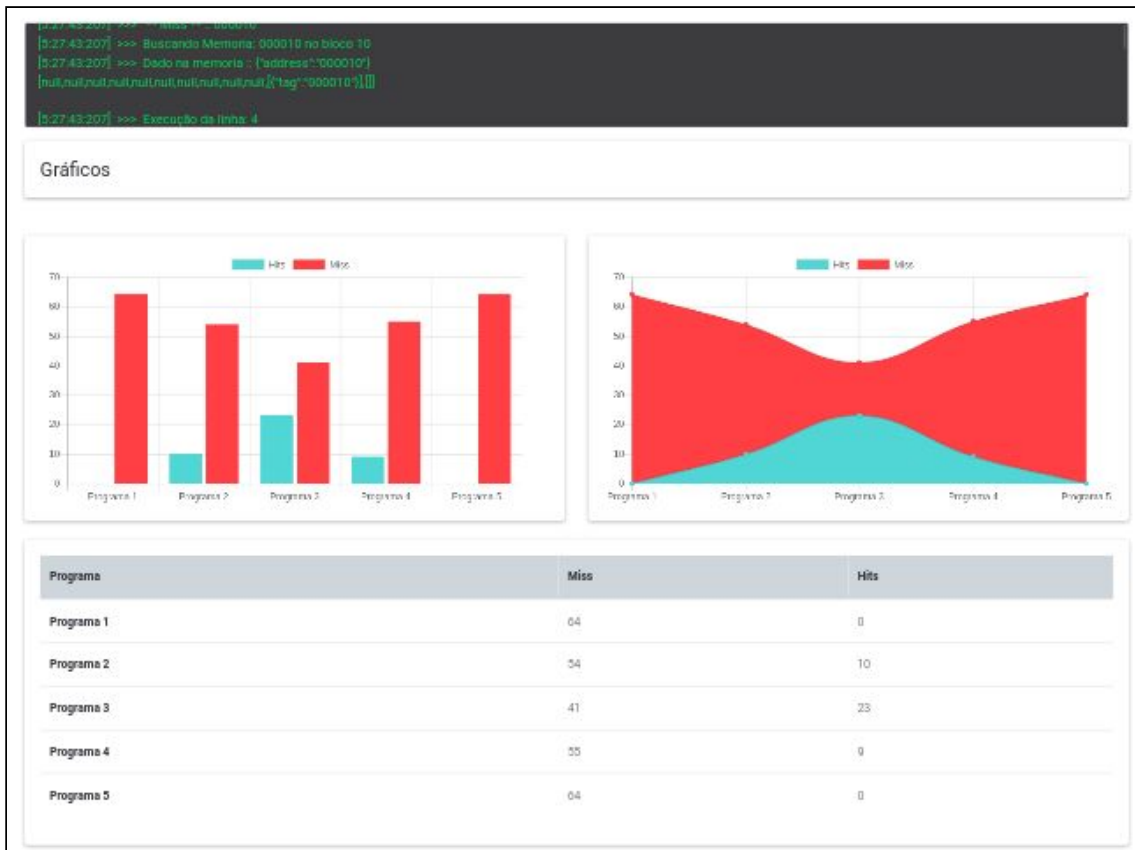


Figura 3: Tela com a apresentação dos resultados

O simulador completo está acessível no repositório “guilnorth/arquitetura-cache” em <https://github.com/guilnorth/arquitetura-cache>

## 2.1 Dificuldades e Observações

Alguns pontos importantes a serem citados que tiveram impacto durante o desenvolvimento do simulador foram:

- Nem todos endereços requisitados pelos programas encontram-se no arquivo de memória correspondente, fazendo necessário validar esse caso antes de aceitar o dado vindo da cache.
- Para o mapeamento associativo por conjunto, o tamanho do índice é definido pelo logaritmo do número de vias na base dois. Como foi fixado em duas e quatro vias, o tamanho do índice sempre vai ser um ou dois.
- A cache totalmente associativa do grupo um de duas vias, torna-se uma cache de mapeamento direto, por seu número de vias ser igual ao número de linhas e o mesmo ocorre com a cache totalmente associativa do grupo quatro, quando em quatro vias.
- A cache associativa por conjunto do grupo dois, quando em quatro vias, não é possível, visto que o número de linhas é inferior ao número de vias.
- O arquivo de memória “Grupo2/m2.txt” possui, nas linhas 62 a 64, números não binários, o que não afeta o processo em geral.

### 3. Experimento

#### 3.1. Grupo 01

Os dados obtidos no simulador, são os seguintes:

Grupo 1	Programa 1	Programa 2	Programa 3	Programa 4	Programa 5
Nº hits	0	3	17	5	0
Nº miss	64	61	47	59	64

Tabela 1: Mapeamento direto, cache com 2 linhas, memória 1

Grupo 1	Programa 1	Programa 2	Programa 3	Programa 4	Programa 5
Nº hits	0	3	17	4	0
Nº miss	64	61	47	60	64

Tabela 2: Mapeamento totalmente associativo, cache com 2 linhas, memória 1

Grupo 1	Programa 1	Programa 2	Programa 3	Programa 4	Programa 5
Nº hits	0	3	17	5	0
Nº miss	64	61	47	59	64

Tabela 3: Mapeamento associativo por conjunto de 2 vias, cache com 2 linhas, memória 1

Mapeamento associativo por conjunto de 4 vias não é possível nesse caso pois há apenas 2 linhas.

#### 3.2. Grupo 02

Grupo 2	Programa 1	Programa 2	Programa 3	Programa 4	Programa 5
Nº hits	0	1	13	6	0
Nº miss	64	63	51	58	64

Tabela 1: Mapeamento direto, cache com 4 linhas, memória 2

Grupo 2	Programa 1	Programa 2	Programa 3	Programa 4	Programa 5
Nº hits	0	1	13	6	0
Nº miss	64	63	51	58	64

Tabela 2: Mapeamento totalmente associativo, cache com 4 linhas, memória 2

Grupo 2	Programa 1	Programa 2	Programa 3	Programa 4	Programa 5
Nº hits	0	1	13	6	0
Nº miss	64	63	51	58	64

Tabela 3: Mapeamento associativo por conjunto de 2 vias, cache com 4 linhas, memória 2

Grupo 2	Programa 1	Programa 2	Programa 3	Programa 4	Programa 5
Nº hits	0	1	13	6	0
Nº miss	64	63	51	58	64

Tabela 4: Mapeamento associativo por conjunto de 4 vias, cache com 4 linhas, memória 2

### 3.3. Grupo 03

Grupo 3	Programa 1	Programa 2	Programa 3	Programa 4	Programa 5
Nº hits	0	11	23	7	0
Nº miss	64	53	41	57	64

Tabela 1: Mapeamento direto, cache com 8 linhas, memória 3

Grupo 3	Programa 1	Programa 2	Programa 3	Programa 4	Programa 5
Nº hits	0	10	23	9	0
Nº miss	64	54	41	55	64

Tabela 2: Mapeamento totalmente associativo, cache com 8 linhas, memória 3

Grupo 3	Programa 1	Programa 2	Programa 3	Programa 4	Programa 5
Nº hits	0	10	23	9	0
Nº miss	64	54	41	55	64

Tabela 3: Mapeamento associativo por conjunto de 2 vias, cache com 8 linhas, memória 3

Grupo 3	Programa 1	Programa 2	Programa 3	Programa 4	Programa 5
Nº hits	0	10	23	9	0
Nº miss	64	54	41	55	64

Tabela 4: Mapeamento associativo por conjunto de 4 vias, cache com 8 linhas,

memória 3

### 3.4. Grupo 04

Grupo 4	Programa 1	Programa 2	Programa 3	Programa 4	Programa 5
Nº hits	0	16	21	14	0
Nº miss	64	48	43	50	64

Tabela 1: Mapeamento direto, cache com 16 linhas, memória 4

Grupo 4	Programa 1	Programa 2	Programa 3	Programa 4	Programa 5
Nº hits	0	14	21	14	0
Nº miss	64	50	43	50	64

Tabela 2: Mapeamento totalmente associativo, cache com 16 linhas, memória 4

Grupo 4	Programa 1	Programa 2	Programa 3	Programa 4	Programa 5
Nº hits	0	14	21	15	0
Nº miss	64	50	43	49	64

Tabela 3: Mapeamento associativo por conjunto de 2 vias, cache com 16 linhas, memória 4

Grupo 4	Programa 1	Programa 2	Programa 3	Programa 4	Programa 5
Nº hits	0	14	21	17	0
Nº miss	64	55	43	47	64

Tabela 4: Mapeamento associativo por conjunto de 4 vias, cache com 16 linhas, memória 4

### 3.5. Grupo 05

Grupo 5	Programa 1	Programa 2	Programa 3	Programa 4	Programa 5
Nº hits	0	22	22	18	0
Nº miss	64	42	42	46	64

Tabela 1: Mapeamento direto, cache com 32 linhas, memória 5

Grupo 5	Programa 1	Programa 2	Programa 3	Programa 4	Programa 5
Nº hits	0	22	22	21	0
Nº miss	64	42	42	43	64

Tabela 2: Mapeamento totalmente associativo, cache com 32 linhas, memória 5

Grupo 5	Programa 1	Programa 2	Programa 3	Programa 4	Programa 5
Nº hits	0	22	22	21	0
Nº miss	64	42	42	43	64

Tabela 3: Mapeamento associativo por conjunto de 2 vias, cache com 32 linhas, memória 5

Grupo 5	Programa 1	Programa 2	Programa 3	Programa 4	Programa 5
Nº hits	0	22	22	21	0
Nº miss	64	42	42	43	64

Tabela 4: Mapeamento associativo por conjunto de 4 vias, cache com 32 linhas, memória 5

### 3.6. Organizando os Dados

Mapeamento Direto					
Hits	Grupo 01	Grupo 02	Grupo 03	Grupo 04	Grupo 05
Programa 01	0	0	0	0	0
Programa 02	3	1	11	16	22
Programa 03	17	13	23	21	22
Programa 04	5	6	7	14	18
Programa 05	0	0	0	0	0

Tabela 1: Mapeamento direto.

Totalmente Associativo					
Hits	Grupo 01	Grupo 02	Grupo 03	Grupo 04	Grupo 05
Programa 01	0	0	0	0	0
Programa 02	3	1	10	14	22
Programa 03	17	13	23	21	22
Programa 04	4	6	9	14	21
Programa 05	0	0	0	0	0



Tabela 2: Mapeamento totalmente associativo.

Associativo por Conjunto de 2 vias					
Hits	Grupo 01	Grupo 02	Grupo 03	Grupo 04	Grupo 05
Programa 01	0	0	0	0	0
Programa 02	3	1	10	14	22
Programa 03	17	13	23	21	22
Programa 04	5	6	9	15	21
Programa 05	0	0	0	0	0

Tabela 3: Mapeamento associativo por conjunto de 2 vias.

Associativo por Conjunto de 4 vias					
Hits	Grupo 01	Grupo 02	Grupo 03	Grupo 04	Grupo 05
Programa 01	0	0	0	0	0
Programa 02	0	1	10	14	22
Programa 03	0	13	23	21	22
Programa 04	0	6	9	17	21
Programa 05	0	0	0	0	0

Tabela 4: Mapeamento associativo por conjunto de 4 vias.

3.7. Gráficos

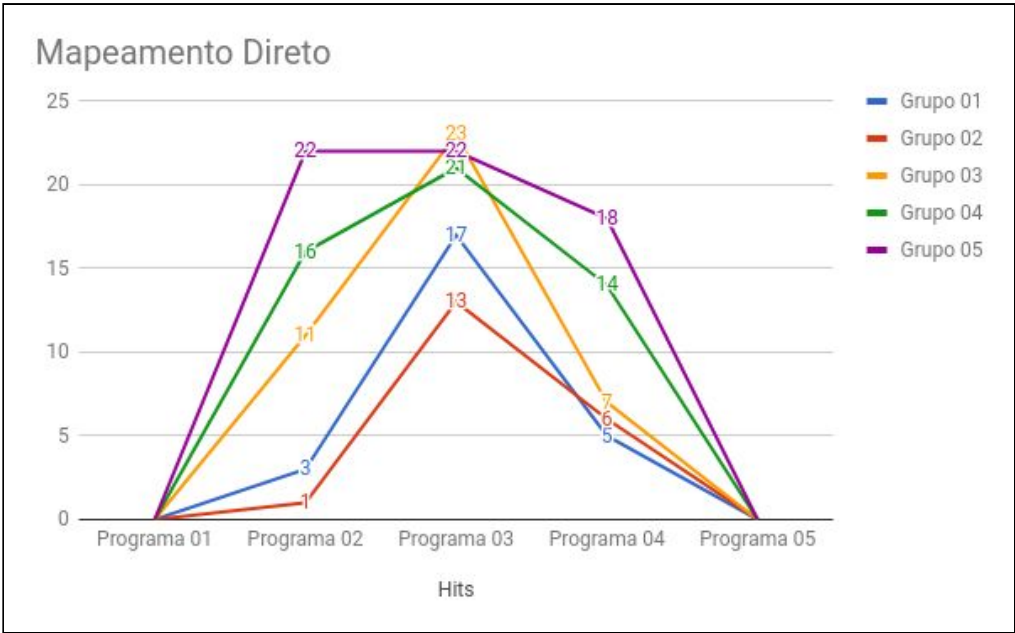


Gráfico 1: Direto, número de Hits.

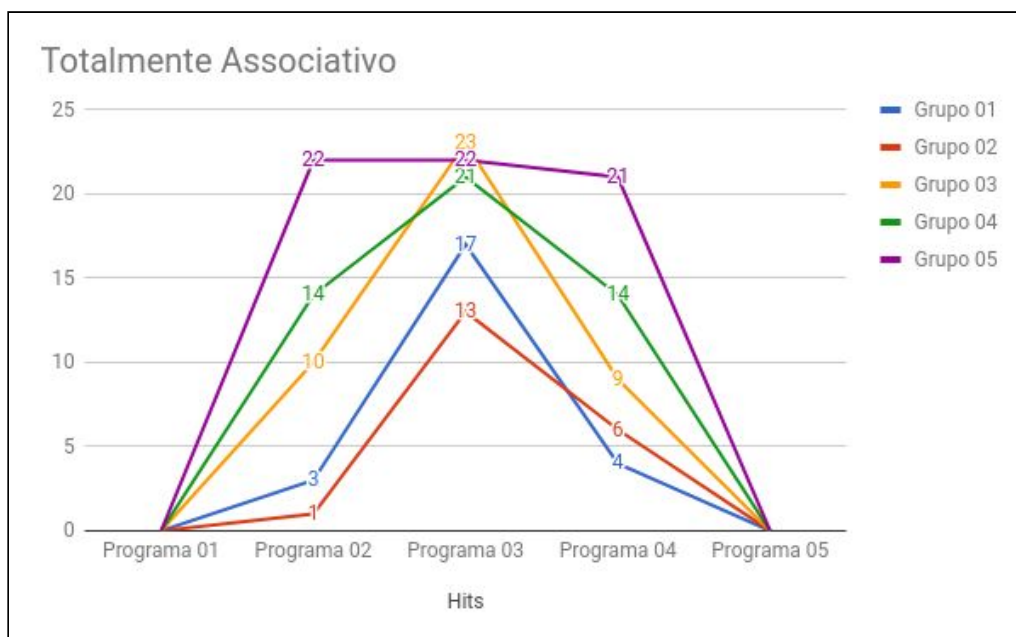


Gráfico 2: Totalmente Asociativo, número de Hits.

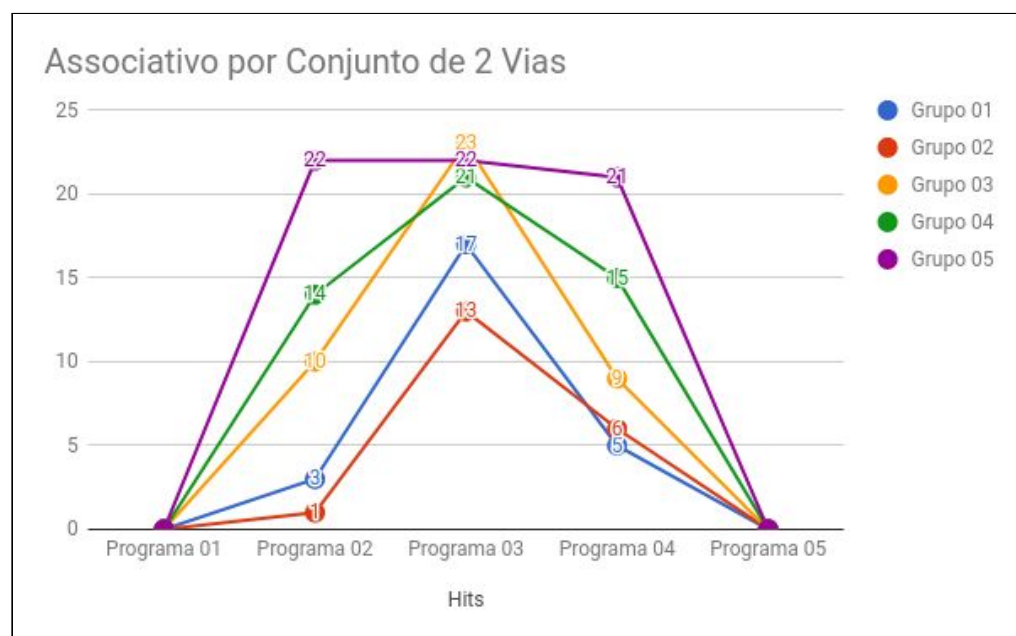


Gráfico 3: Asociativo por conjunto de 2 vias, número de Hits.

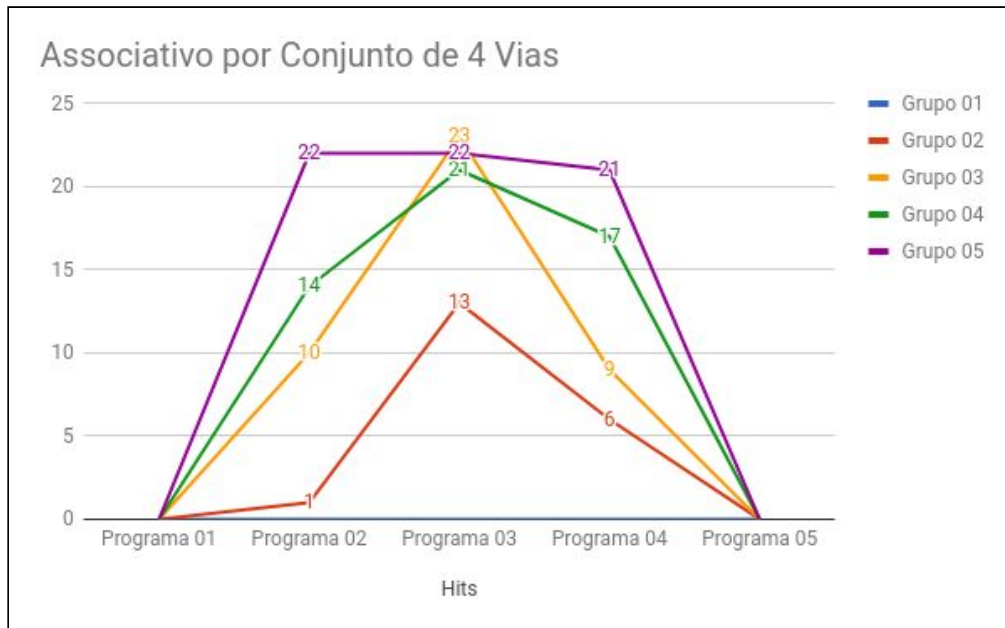


Gráfico 4: Associativo por conjunto de 4 vias, número de Hits.

### 3.8. Conclusões e observações do experimento

Como pode-se observar, a medida em que vai aumentando o número de linhas da cache, a taxa de hits também vai aumentando na maioria dos programas, exceto no Programa 3, onde uma cache com mais de 8 linhas já é o suficiente e até melhor do que caches muito grandes. Assim, número de linhas é importante, mas um programa bem construído e otimizado é de igual importância para o ganho de performance.

Outros pontos a se levar em conta são:

- O Programa 1 não permite um hit na cache, independente do tipo de mapeamento, isso pelo fato de não fazer requisições duplicadas;
- O Programa 2 permanece com a taxa de hits próxima independente do tipo de mapeamento, isso devido ao tipo de política de substituição escolhida (FIFO), já que ela prioriza a localidade temporal e o Programa 2 possui repetições de tempos em tempos;
- O Programa 3 é o que possui maior número de hits, por fazer requisições seguidas de um mesmo termo inúmeras vezes;
- O Programa 4 se comporta de maneira parecida com o Programa 2 e suas taxas permanecem próximas em caches com número de linhas maiores que 4. Em número de linhas de 4 ou menores a diferença é maior e o Programa 2 sai na frente;
- O programa 5 não permite um hit na cache pelos mesmos motivos do programa 1.

#### **4. Conclusão**

De acordo com o experimento, pode-se dizer que vários fatores influenciam na taxa de hits na cache de um programa, o número de linhas da cache é de fundamental importância, assim como a otimização do programa em execução e o tipo de mapeamento, sendo o último o que se verifica menor diferença de resultado nas circunstâncias propostas. Porém em meios mais complexos essa diferença pequena pode pesar. No caso estudado, essa diferença é irrelevante na maioria dos casos, exceto no Programa 4 no Grupo 5, onde o mapeamento direto tem uma taxa abaixo dos demais casos, e o Programa 1 no Grupo 1 que não aceita o Mapeamento associativo por conjunto de 2 vias, por ser uma cache muito pequena.

Um ponto a se ressaltar é a dificuldade na análise mais profunda dos programas 2, 3 e 4, por serem diferentes nos 5 grupos, apesar de seguir um padrão similar seria uma análise mais confiável se todos fossem totalmente iguais.

#### **5. Referências**

ROSE, César A. F. De; MORAES, Fernando Gehm. ARQUITETURA DE COMPUTADORES II: UNIDADE 2: GERÊNCIA DE MEMÓRIA. Disponível em: <<https://www.inf.pucrs.br/~flash/orgarq/aulas/memoria>>. Acesso em: 10 maio 2018.