

# Otimizando o tempo de execução no processamento de imagens

Henrique Miyamoto e Thiago Benites

## I. CONTEXTUALIZAÇÃO

*Threads*, assim como processos, são mecanismos que permitem que um programa execute ações de forma aparentemente simultânea. A diferença é que *threads* possuem área de memória compartilhada, o que não ocorre com processos [1]. Apresentamos uma comparação de desempenho entre diferentes métodos para aplicação de brilho em uma imagem: usando múltiplas *threads*, usando multiprocessos, em uma única linha de execução, varrendo a matriz por linhas e por colunas. O objetivo é comparar e discutir os desempenhos de cada implementação a partir do tempo de execução real e de usuário de cada uma delas.

## II. DEMONSTRAÇÃO

Na comparação de desempenho, foram usadas imagens pequena (32x32 pixels), média (720x460) e grande (2592x1944) e foram medidos os tempos reais (tempo de relógio) e de usuário (tempo que a CPU gasta dentro dos processos). O processador usado para os testes tinha as seguintes especificações: *Intel(R) Core(TM) i5-7200U CPU @ 2.50GHz*.

Inicialmente, variou-se a quantidade de *threads* e processos em cada execução para otimizar esse número. Nesse procedimento, a imagem média foi usada como referência. A otimização do tempo de execução se dá para aproximadamente 800 *threads* e 1 processo, dentre os resultados avaliados.

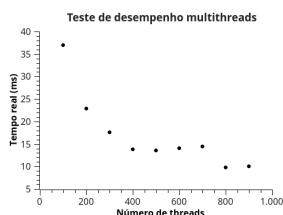


Fig. 1. Tempo real (ms) em função do número de *threads*.

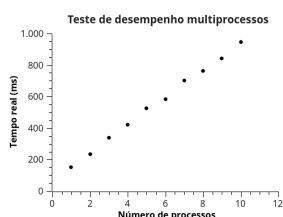


Fig. 2. Tempo real (ms) em função do número de multiprocessos.

\* Os arquivos do projeto estão disponíveis em <https://github.com/miyamotok/linguagem-processamento-imagem>.

A seguir, foram comparados os diferentes métodos de aplicação de brilho para cada imagem. Nesse teste, o número de *threads* e processos usados foi o número ótimo encontrado anteriormente.

TABLE I

TESTES DE DESEMPENHO PARA IMAGENS PEQUENA, MÉDIA E GRANDE

Método	Tamanho	Tempo real	Tempo de usuário
<i>Multithreads</i>	32x32	0,6650 ms	0,3650 ms
Multiprocessos	32x32	5,8270 ms	5,8280 ms
Varredura por colunas	32x32	0,0310 ms	0,0300 ms
Varredura por linhas	32x32	0,0210 ms	0,0200 ms
<i>Multithreads</i>	720x460	9,4800 ms	2,5760 ms
Multiprocessos	720x460	147,2950 ms	65,1070 ms
Varredura por colunas	720x460	9,2150 ms	9,2150 ms
Varredura por linhas	720x460	6,8940 ms	6,8930 ms
<i>Multithreads</i>	2592x1944	135,3250 ms	39,6660 ms
Multiprocessos	2592x1944	3819,1140 ms	637,9850 ms
Varredura por colunas	2592x1944	149,0920 ms	149,0950 ms
Varredura por linhas	2592x1944	142,3600 ms	142,3610 ms

## III. ANÁLISE

Observa-se que o método com menor tempo real de execução depende do tamanho da imagem utilizada. Para as imagens pequena e média, a varredura em uma única linha de execução por linhas apresenta melhor desempenho. Já para a imagem grande, este é alcançado pelo método *multithreads*. Uma explicação para essa diferença é a solução de compromisso inerente ao procedimento *multithreads*: existe um gasto computacional para a criação das *threads*, que só é compensado para imagens muito grandes, para as quais a execução paralela é vantajosa.

A implementação com múltiplos processos mostrou desempenho inferior às demais implementações. É possível que esse fato esteja relacionado ao gasto computacional necessário para criação da área de memória compartilhada e criação individual de cada processo. Em todos os casos, a varredura por linha apresentou tempo menor que por colunas, fato derivado de como a matriz foi definida. As linhas são posicionadas em sequência, logo, a análise se dá consecutivamente, pixel a pixel, enquanto que em colunas, não há essa sequência na posição de cada pixel.

Nos casos em que o tempo real é menor que o tempo de usuário, a aplicação do brilho é *CPU bounded* (i.e., o maior gargalo de execução está no processador). Por outro lado, no caso contrário, é *I/O bounded* (i.e., o gargalo está na entrada/saída de dados), pois o programa passa tempo esperando os dados.

## REFERÊNCIAS

- [1] MITCHELL, Mark; OLDHAM, Jeffrey e SAMUEL, Alex. *Advanced Linux Programming*. Indianapolis: New Riders, 2001.