Relatório da atividade 04 - Diretivas do compilador

Grupo:
Artur Kenzo - 15652663
Daniel Jorge - - 15446861
Gabriel Phelippe - 15453730
Jhonatan Barboza - 15645049

---

1. Introdução

Este relatório apresenta uma análise comparativa do impacto das diferentes flags de otimização do GCC no desempenho, tempo de compilação e tamanho dos executáveis. O estudo foi realizado utilizando dois programas de benchmark da Computer Language Benchmarks Game (CLBG).

2. Metodologia

2.1 Arquitetura de Teste

- **Processador:** 13th Gen Intel® Core™ i5-1335U × 12
- **Sistema Operacional:** Linux Ubuntu
- **Compilador:** GCC versão 13.x

2.2 Programas Selecionados

Dois programas foram escolhidos do CLBG para representar diferentes características computacionais:

1. **fasta.c** - Programa de geração de sequências FASTA
    - Fonte: https://benchmarksgame-team.pages.debian.net/benchmarksgame/program/fasta-gcc-3.html
    - Características: Operações de I/O intensivas, manipulação de strings
2. **n-body.c** - Simulação de sistema n-corpos
    - Fonte: https://benchmarksgame-team.pages.debian.net/benchmarksgame/program/nbody-gcc-9.html
    - Características: Cálculos matemáticos intensivos, uso de instruções AVX

2.3 Configurações de Compilação Testadas

- **Sem otimização:** `gcc -o executavel_base codigo_fonte.c`
- **-O1:** `gcc -O1 -o executavel_o1 codigo_fonte.c`
- **-O2:** `gcc -O2 -o executavel_o2 codigo_fonte.c`
- **-O3:** `gcc -O3 -o executavel_o3 codigo_fonte.c`
- **-Os:** `gcc -Os -o executavel_os codigo_fonte.c`

Nota: Para n-body.c foram adicionadas as flags `-mavx -lm` devido ao uso de instruções AVX e funções matemáticas.

2.4 Métricas Avaliadas

1. **Tempo de Compilação:** Média de 10 compilações usando o comando `time`
2. **Tamanho do Executável:** Medido em KB usando `stat`
3. **Tempo de Execução:** Média de 10 execuções com parâmetro 50000000

3. Resultados

3.1 Programa fasta.c

Configuração	Tempo Comp.(s)	Tamanho(KB)	Tempo Exec.(s)	Speedup
Sem otimização	0.0720	16.29	3.5530	1.0x
-O1	0.0960	16.06	2.3560	**1.51x**
-O2	0.1130	16.01	2.3570	**1.51x**
-O3	0.1180	16.01	2.3380	**1.52x**
-Os	0.1050	16.01	3.5470	1.00x

3.2 Programa n-body.c

Configuração	Tempo Comp.(s)	Tamanho(KB)	Tempo Exec.(s)	Speedup
Sem otimização	0.3690	19.81	19.5510	1.0x
-O1	0.4580	15.71	3.3410	**5.85x**
-O2	0.4860	15.72	3.3740	**5.80x**
-O3	0.4930	15.72	2.6540	**7.37x**
-Os	0.4810	15.75	3.3790	5.79x

4. Análise dos Resultados

4.1 Tempo de Execução

fasta.c

- **Melhor performance:** -O3 (52% mais rápido)
- **Comportamento consistente:** -O1, -O2 e -O3 apresentam performance similar (~51-52% melhoria)
- **-Os neutro:** Performance equivalente ao código não otimizado
- **Ganho moderado:** Máximo de 1.52x de speedup

n-body.c

- **Melhor performance:** -O3 (7.37x mais rápido)
- **Ganho significativo:** Todas as otimizações oferecem speedup > 5x
- **Progressão clara:** O1 ≈ O2 ≈ Os < O3
- **Alto potencial de otimização:** Devido à natureza matemática intensiva

4.2 Tempo de Compilação

- **fasta.c:** Aumento modesto (0.07s → 0.14s máximo)
- **n-body.c:** Aumento mais significativo (0.37s → 0.50s)
- **Padrão geral:** Tempo de compilação cresce com nível de otimização
- **Trade-off:** Tempo adicional de compilação vs. ganho de performance

4.3 Tamanho do Executável

fasta.c

- **Redução mínima:** 16.29KB → 16.01KB (1.7% de redução)
- **Impacto baixo:** Todas as otimizações resultam em tamanho similar

n-body.c

- **Redução significativa:** 19.81KB → 15.72KB (20.6% de redução)
- **Benefício consistente:** Todas as otimizações reduzem o tamanho

5. Discussão

5.1 Características dos Programas

fasta.c demonstra um perfil típico de programa I/O-bound, onde as otimizações têm impacto limitado devido aos gargalos de entrada/saída. O comportamento anômalo do -O3 pode ser explicado por over-optimization que interfere com o cache ou pipeline do processador.

n-body.c representa um programa CPU-bound ideal para otimizações, utilizando cálculos matemáticos intensivos e instruções vetoriais (AVX). O ganho substancial com -O3 reflete a eficácia das otimizações avançadas em código computacionalmente intensivo.

5.2 Implicações Práticas

Os resultados demonstram que as flags -O1, -O2 e -O3 oferecem performance equivalente no programa fasta.c, com speedups entre 1.51x e 1.52x, tornando -O1/-O2 opções particularmente atrativas devido ao menor tempo de compilação. Esta consistência entre os níveis de otimização sugere que o programa atinge rapidamente um teto de performance, onde otimizações mais agressivas não produzem benefícios proporcionais ao esforço computacional adicional.

A flag -Os apresenta comportamento neutro no fasta.c, mantendo performance equivalente ao código não otimizado, o que a qualifica como opção viável quando a prioridade é exclusivamente a redução do tamanho do executável. Em contraste, o programa n-body.c responde dramaticamente às otimizações, evidenciando que diferentes algoritmos e padrões computacionais requerem análise empírica específica para determinar a estratégia de otimização mais adequada.

6. Conclusões

A análise com execuções múltiplas demonstra que otimizações são altamente específicas às características de cada programa. Enquanto n-body.c alcançou speedups ainda mais dramáticos (7.37x) devido à sua natureza computacionalmente intensiva, fasta.c apresentou comportamento mais consistente e previsível,

com todas as flags de otimização (-O1, -O2, -O3) oferecendo ganhos similares de aproximadamente 50%. As flags -O1/-O2 mantêm-se como escolhas seguras, oferecendo excelente custo-benefício.

A flag -O3 confirmou-se como a melhor opção para programas matemáticamente intensivos, enquanto para programas I/O-bound oferece apenas ganhos marginais sobre -O1/-O2. A execução de múltiplas iterações revelou-se fundamental para obter resultados confiáveis, evidenciando a importância de benchmarks robustos na tomada de decisões sobre otimizações de compilação.