Column 1: Profilers

Estratégias de Medição e Otimização de Código

Sérgio Alexandre Gobbi Rebello
Gabriel Henrique Garces de Deus
Miguel Büge Paganini

2026

1 Introdução e o Conceito de Profiling

Jon Bentley define o *profiler* como o "estetoscópio"do programador. Em vez de basear a otimização em intuição, a ferramenta fornece dados reais sobre a execução:

- **Contagem de Linhas:** Identifica quantas vezes cada instrução foi executada.
- **Perfil de Tempo:** Mostra quais funções consomem a maior fatia da CPU.

2 Programa P1: A Abordagem de Força Bruta

```
int prime(int n) {
    int i;
    for (i = 2; i < n; i++)
        if (n % i == 0) return 0;
    return 1;
}

main() {
    int i, n = 1000;
    for (i = 2; i <= n; i++)
        if (prime(i)) printf("%d\n", i);
}
```

3 Programa P2: O Custo da Função sqrt()

```
int prime(int n) {
    int i;
    for (i = 2; i <= (int)sqrt((float)n); i++)
        if (n % i == 0) return 0;
    return 1;
}
```

4 Programa P3: Otimização via Variável Auxiliar

```
int prime(int n) {
    int i, bound;
    bound = (int)sqrt((float)n);
    for (i = 2; i <= bound; i++)
        if (n % i == 0) return 0;
    return 1;
}
```

5 Programa P4: Identificando Erros Lógicos

```
int prime(int n) {
    if (n % 2 == 0) return 0;
    if (n % 3 == 0) return 0;
    if (n % 5 == 0) return 0;
    bound = (int)sqrt((float)n);
    for (i = 7; i <= bound; i = i + 2)
        if (n % i == 0) return 0;
    return 1;
}
```

6 Programa P5: Eficiência Máxima

```
int prime(int n) {
    if (n % 2 == 0) return (n == 2);
    if (n % 3 == 0) return (n == 3);
    if (n % 5 == 0) return (n == 5);
    for (i = 7; i * i <= n; i = i + 2)
        if (n % i == 0) return 0;
    return 1;
}
```

7 Análise Comparativa de Desempenho

Algoritmo	**N = 1.000**	**N = 10.000**	**N = 100.000**
P1 (Simples)	2.4s	169s	—
P3 (Raiz fixa)	1.4s	15s	192s
P5 (Sem sqrt)	0.3s	3.5s	64s
Q1 (Crivo)	0.2s	1.2s	10.4s

8 Conclusões

- **Dados vs. Intuição:** O caso do P2 prova que o que parece mais rápido pode ser mais lento sem medição.
- **Cobertura de Testes:** O *profiler* serve como ferramenta de qualidade ao expor partes do código que não se comportam como esperado.
- **Escalabilidade:** Em sistemas de larga escala, o *profiling* é o único meio seguro de aplicar o esforço de engenharia.