Comparação de métodos de cálculo da raiz quadrada inversa

Bruno Fusieger

Universidade Estadual de Maringá (UEM)
Maringá – PR – Brasil

ra112646@uem.br

Resumo. *Este artigo apresenta uma comparação entre os métodos de cálculo da raiz quadrada inversa por dois métodos de Newton-Rapson e Gary Tarolli. Também foi aplicada um método de aproximação por shifts de bits. Foram analisadas a precisão e o número de multiplicações de cada método.*

1. Introdução

O cálculo da raiz quadrada inversa é uma área de interesse de áreas da computação como a computação gráfica, o processamento digital de sinais, entre outros. Para este artigo, foi feita uma comparação entre métodos que aproximam o valor da raiz quadrada inversa no formato de precisão simples da IEEE 754.

Os métodos explorados foram, Newton-Raphson da raiz quadrada, onde usa-se o cálculo da raiz quadrada e depois inverte-se o número, Newton-Raphson da raiz quadrada inversa onde o cálculo da raiz quadrada é direto, e o método de Gary Tarolli.

Para melhorar as estimativas inicias dos métodos baseados em Newton-Raphson também foi usado uma aproximação inicial baseada em shifts de bits.

2. Aproximação por shift de bits

A aproximação da raiz quadrada tira vantagem do padrão IEEE 754 de 32 bits, assim como o método desenvolvido por Gary Tarolli que será descrito a frente.

Primeiramente, o formato pode ser descrito pela equação:

$$x = (-1)^s(1+f)2^{e+b}$$

Onde s é o bit de sinal, f a fração da mantissa normalizada, e o expoente e b o polarizador. No padrão de 32 bits temos $b = 127$ e como a raiz quadrada admite apenas valores positivos, teremos sempre $s = 0$, portanto:

$$x = (1+f)2^{e+b}$$

Definiremos que um valor representado no formato IEEE 754 pode ser interpretado também com um inteiro de 32 bits, denotado daqui em diante por x_{int}. Temos a seguinte aproximação usando a interpretação por inteiro de x, onde B é o número de bits da mantissa, e b o polarizador.

$$log_2(x) \approx x_{int} \cdot 2^{-B} - b$$

Considere agora que desejamos calcular a raiz quadrada de x, ou seja:

$$\sqrt{x} = y \implies x^{\frac{1}{2}} = y \implies log_2(y) = \frac{1}{2} log_2(x)$$

Podemos então aplicar a aproximação do logaritmo descrita anteriormente e substituir $log_2 x$, obtendo assim:

$$log_2(y) \approx \frac{1}{2}(x_{int} \cdot 2^{-B} - b) \approx log_2(\sqrt{x})$$

Por fim, ao manipular essa fórmula conseguimos extrair a seguinte expressão, que representa uma estimativa inicial para a raiz quadrada de x:

$$\sqrt{x} \approx \frac{x_{int} - 2^B}{2} + \frac{b+1}{2}2^B$$

Para o padrão IEEE de 32 bits de precisão, que tem 23 bits de mantissa e 7 bits de polarizador podemos simplificar a expressão para a seguinte:

$$\sqrt{x} \approx \frac{x_{int} - 2^{23}}{2} + 2^{29}$$

Com essa fórmula podemos criar o seguinte código para a a aproximação da raiz quadrada, observe que a grande vantagem dessa aproximação é que ela não envolve multiplicações.

```
float approxSqrt(float x) {
     union {
          float f;
          u_int32_t k;
     } val = { .f = x };

     val.k -= 1 << 23;
     val.k >>= 1;
     val.k += 1 << 29;

     return val.f;
}
```

2. Aproximação da raiz quadrada por Newton-Raphson

Neste caso vamos usar o método de newton-raphson para calcular a raiz quadrada de A e ao final calcular a reciproca da raiz quadrada de A obtida, esse método portanto sempre contará com uma divisão ao final.

Usando a equação de recorrência de Newton-Raphson, definida genericamente por:

$$x_{k+1} = x_k + \frac{f(x_k)}{f'(x_k)}$$

Para calcular a raiz quadrada, primeiro vamos definir a função *f*:

$$f(x) = x^2 - A = 0$$

Onde A é o valor de entrada da função, representado no padrão de precisão simples da IEEE. Com isso temos a seguinte derivada:

$$f'(x) = 2x$$

A partir disso tem-se a seguinte expressão de recorrência:

$$\sqrt{x} = x_{k+1} = \frac{1}{2}(x_k + \frac{A}{x_k})$$

Tomaremos x_0 como o valor pela função *approxSqrt* definida anteriormente, com isso teremos como resultado a seguinte função.

```
float newtonRaphson(float a) {
    union {
        float f;
        u_int32_t k;
    } val = { .f = approxSqrt(a) };

    val.f = (val.f + a / val.f);

    val.f = val.f / 2.0f;

    return 1.0f / val.f;
}
```

3. Aproximação da raiz quadrada inversa por Newton-Raphson

Para este caso vamos calcular a equação de recorrência de Newton-Raphson para obter diretamente a raiz quadrada inversa de A, neste caso temos f e f' como:

$$f(x) = \frac{1}{x^2} - A = 0$$

$$f'(x) = -\frac{2}{x^3}$$

A partir disso tem-se a seguinte expressão de recorrência:

$$\frac{1}{\sqrt{x}} = x_{k+1} = x_k(\frac{2}{3} - \frac{A}{2}x_k^2)$$

Aqui também tomaremos x_0 como o resultado de *approxSqrt*, no entanto, é preciso inverter a entrada antes de passa-la para a função de aproximação, logo teremos:

$$x_0 = \text{approxSqrt}(1/A)$$

Por fim temos o seguinte código em C:

```
float invNewtonRaphson(float a) {
     float a2 = a / 2.0f;

     float x = approxSqrt(1 / a);

     x = x * (1.5f - (a2 * x * x));

    return x;
}
```

4. Aproximação por Gary Tarolli

O método de Gary Tarolli consiste em duas coisas, primeiro usar uma aproximação para a raiz quadrada de 2^{127} e usa-lá, interpretada como um inteiro, ou seja 0x5f375a86. Depois interpretando o valor de entrada A como um inteiro, realizar um shift de 1 bit para direita e diminuí-lo da aproximação definida anteriormente.

Ao final do algoritmo ainda é realizada uma iteração do método de Newton-Raphson para melhorar a aproximação. Esse método resulta no seguinte código:

```
float garyTarolli(float a) {
    float a2 = a / 2.0f;

    union {
        unsigned int k;
        float x;
    } u = {.x = a};

    u.k = 0x5f375a86 - (u.k >> 1);

    u.x = u.x * (1.5f - a2 * u.x * u.x);

    return u.x;
}
```

5. Comparação

Para a comparação foi calculado o erro absoluto entre os três métodos quando comparados com a reciproca da função da biblioteca padrão. O espaço de análise foi um espaço linear de 1 até 200 com 50 amostras.

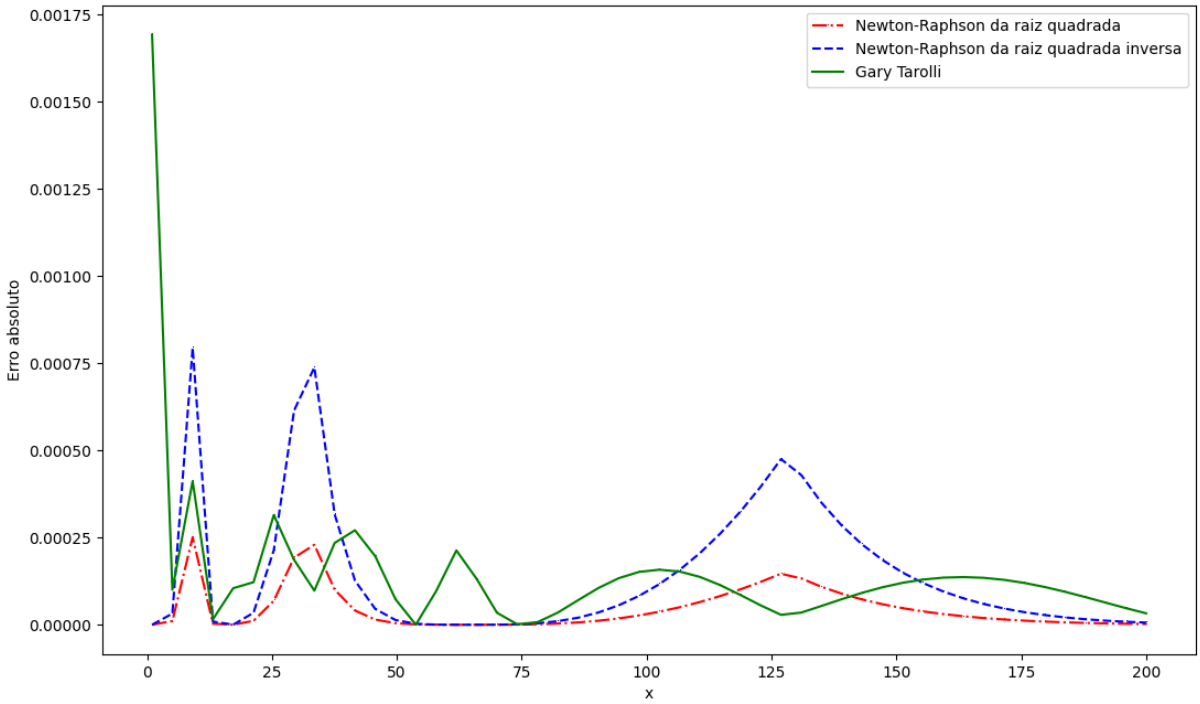

Figura 1. Comparação do erro absoluto

Como pode ser observado todos os métodos se mantém com pelo menos duas casas decimais de precisão, no entanto, quando comparamos o número de multiplicações temos: Newton-Raphson da raiz quadrada com 3 multiplicações, Newton-Raphson da raiz quadrada inversa com 4 multiplicações e o método de Gary Tarolli com também quatro multiplicações. Isso demonstra que o método usando Newton-Raphson da raiz quadrada tem desempenho melhor enquanto mantém a mesma precisão dois demais.