Table of Contents

[Khảo sát sự sai số 8](#_Toc65283143)

**Sơ lược về số thực.**

* Số thực là một tập hợp các số vô tỉ và hữu tỉ, bao gồm cả số dương, số 0 và số âm.
* Tính liên tục và trù mật của số thực.
* Xét tập hợp số hữu tỉ .

**Sự chuẩn hoá**

Với là cơ số, p là số lượng các chữ số có thể biểu diễn, sự chuẩn hoá được định nghĩa như sau:

Biểu diễn còn có thể được viết lại dưới dạng:

**Ví dụ:**

|  |  |  |
| --- | --- | --- |
| Số chưa chuẩn hoá: | Số đã chuẩn hoá | Cách diễn đạt khác |
|  |  |  |
|  |  |  |

Kiến trúc dấu chấm động theo chuẩn IEEE-754

|  |  |  |
| --- | --- | --- |
| Sign (S) | Exponent (E) | Fraction (F) |

**S:** bit dùng xác định dấu (0: dương, 1: âm).

**E:** phần số mũ Exponent.

**F:** Phần định trị

Công thức chung biểu diễn giá trị số dấu chấm động với M = :

|  |  |  |
| --- | --- | --- |
|  | Chi tiết | Ý nghĩa |
| **bias** | được tính bởi vớik là số lượng bit trong vùng E | Độ lệch |
| **p** | số lượng bit có trong vùng Fraction | độ chính xác |
|  | bit tại vị trí {\displaystyle {\frac {s}{b^{\,p-1}}}\times b^{e},} | Giá trị bit |

Ta cũng có thể tính M theo cách sau:

**Lưu ý:** Do phần fraction ở hệ nhị phân ở dạng normalized luôn là 1.

Do đó:

1. Giá trị 1 trước **binary point** được ngầm hiểu là 1 mà không cần phải lưu trữ.
2. Điều này có nghĩa là với p = 24(Single) ta có số bit trong vùng Fraction sẽ là p – 1 = 23 (Tương tự với p = 53(Double) sẽ là 52 bit).

|  |  |
| --- | --- |
| **Single Precesion** | **Double Precesion** |
| 1bit Sign + 8bit Exponent + 23bit Fraction(p = 24) | 1bit Sign + 11bit Exponent + 52bit Fraction(p = 53) |
| |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | 1 | 8 | 7 | … | 1 | 24 | 22 | … | 2 | 1 | |  |  | | | |  | | | | | | |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | 1 | 11 | 10 | … | 1 | 53 | 50 | … | 2 | 1 | |  |  | | | |  | | | | | |

BIỂU DIỄN DẤU CHẤM ĐỘNG

Để đơn giản ta sử dụng định dạng Single Precession (cơ chế của định dạng Double Precession cũng hoạt động tương tự).

**Cách chuyển từ hệ 2 sang hệ 10 (số 32bit).**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | … | 0 |

|  |  |
| --- | --- |
| ) | |
|  |  |
|  |  |

Vậy kết quả khi chuyển từ hệ nhị phân sang hệ 10

Cách chuyển từ hệ 10 sang hệ 2

|  |  |  |
| --- | --- | --- |
| Phần Dấu | Phần Nguyên | Phần Thập Phân |
| 1(S = 0) |  |  |

* Tìm dãy số nhị phân cho phần nguyên và phần thập phân
  + Phần Nguyên:
  + Phần Thập Phân:

|  |  |  |
| --- | --- | --- |
|  |  | 1 |
|  |  | 1 |
|  |  | 0 |
|  | (Dừng) | 1 |

* + Gộp Phần Nguyên và Phần Thập Phân lại:
* Normalize kết quả:
* Tính toán số mũ
  + Ta có phần mũ sau khi chuẩn hoá là

|  |  |
| --- | --- |
|  |  |

Vậy kết quả biểu diễn nhị phân của

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | … | 0 |

MIỀN GIÁ TRỊ SỐ THỰC

Diagram

Description automatically generated

Hình ảnh được lấy từ: <https://slideplayer.com/slide/14875529/>

**DẠNG CHUẨN**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 1 | Tất cả các bit đều là 0 |
| CỰC TIỂU | | | | | **Single precesion** | | | | | | **Double precesion** |
| S | | | | |  | | | | | |  |
|  | | | | |  | | | | | |  |
| Significand | | | | |  | | | | | |  |
| Value | | | | |  | | | | | |  |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 1 | 1 | 1 | | 1 | 1 | 1 | 0 | Tất cả các bit đều là 1 |
| CỰC ĐẠI | | | | | **Single precesion** | | | | | | **Double precesion** |
| S | | | | |  | | | | | |  |
|  | | | | |  | | | | | |  |
| Significand | | | | |  | | | | | |  |
| Value | | | | |  | | | | | | ( |

**DẠNG KHÔNG CHUẨN**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | Tất cả bằng 0 | | 0 | 0 | … | 0 | 0 | 1 |
| CỰC TIỂU | | **Single precesion** | | | | | | | | **Double precesion** |
| S | |  | | | | | | | |  |
|  | |  | | | | | | | |  |
| Significand | |  | | | | | | | |  |
| Value | |  | | | | | | | |  |

|  |  |  |  |
| --- | --- | --- | --- |
| 0 | Tất cả bằng 0 | | Tất cả các bit đều là 1 |
| CỰC ĐẠI | | **Single precesion** | | **Double precesion** |
| S | |  | |  |
|  | |  | |  |
| Significand | |  | |  |
| Value | |  | |  |

MẤT MÁT THÔNG TIN

Mất mát thông tin phát sinh trong quá trình lưu trữ (Rounding and truncate error).

* Biểu diễn số thực trên máy tính.
* Ép kiểu dữ liệu.
* Làm tròn số.

Do không phải giá trị bất kì nào cũng có thể biểu diễn được trên máy tính. Cho nên với một giá trị x bất kì cần biểu diễn, ta sẽ tìm được hai giá trị a và b trên máy gần x nhất thoả điều kiện và sẽ được biểu diễn bằng giá trị này, lúc này mất mát thông tin sẽ xảy ra.

Sự mất mát thông tin được biểu diễn qua 2 cách đánh giá:

Rounding Error =

Relative Error =

Tuỳ trường hợp mà ta nên sử dụng cách đánh giá sao cho phù hợp, xét 2 ví dụ bên dưới.

***Ví dụ 1***

*Biểu diễn số , với độ chính xác là 3.*

**Đáp:** Với độ chính xác này có thể được biểu diễn thành 3.14 hoặc 3.15

* Lỗi 3.14 so với 3.14159 là:
* Lỗi 3.15 so với 3.14159 là:

Ta nhận thấy rằng độ lỗi của 3.15 là lớn hơn vậy x sẽ được biểu diễn thành số 3.14.

***Ví dụ 2***

*Đánh giá độ lỗi của 3.14159 được làm tròn về 3.14 so với độ lỗi của một phép tính có giá trị chính xác là 11.1 nhưng bị lỗi trong quá trình làm tròn nên giá trị tính toán là 11.2.*

**Đáp:** Giá trị lỗi được tính toán:

* Lỗi 3.14 so với 3.14159 là: (\*)
* Lỗi 11.1 so với 11.2 là: (\*\*)

Do 2 độ lỗi được tính toán trên các đơn vị khác nhau nên Relative Error một độ lỗi tương đối được áp dụng là hợp lý nhất:

* Lỗi 3.14 so với 3.14159 là:
* Lỗi 11.1 so với 11.2 là:

Từ đây ta mới có thể kết luận sự mất mát của (\*\*) lớn gấp 18 lần (\*).

**Nhận xét:**

Với cách đánh giá ở ***ví dụ 1*** chỉ thích hợp cho đánh giá các giá trị cùng 1 tiêu chuẩn. Nhưng đối với việc đánh giá giữa các tập dữ liệu dựa trên các tiêu chuẩn khác nhau cách tính % như ***ví dụ 2*** phù hợp và đúng thực trạng hơn.

**Unit in the last place**

Ở ví dụ 2 ta nhận thấy còn một điểm chưa được thống nhất là cần phải chuẩn hoá các độ lỗi về đơn vị như thế nào để tiện cho việc so sánh. Ví dụ như thì ta nên chọn số nào. Từ một bài báo khoa học[8] đã chỉ ra cách mà ta có thể thống nhất được đơn vị unit này. Ta gọi đó là một “unit in the last place” hay “ulp”.

ULP được tính bằng khoảng cách nhỏ nhất giữa 0 và một số bất kì nhỏ nhất mà máy tính có thể biểu diễn được(dựa hệ số và chuẩn hệ thống hiện tại). Các giá trị x được biểu diễn dựa trên ulp được kí hiêu là Ulp(x), được biểu diễn như sau:

Chứng minh[8] cũng đã chỉ ra rằng:

Giá trị chặn trên của biểu thức được giới hạn bởi một lượng , được gọi là hệ số epsilon machine.

Với những số biểu diễn quá nhỏ như Relative Error ta có thể biểu diễn lại chúng bằng số để thuận lợi cho việc đánh giá:

***Ví dụ:***

**Đề bài:** Với cách biểu diễn số lỗi 3.14 so với 3.14159 là rất nhỏ như ở ví dụ trên ta có thể biểu diễn lại dựa theo hệ số là:

**Biểu diễn Rounding Error theo ULPs**

1. Tính Rounding error: .
2. Chuẩn hoá giá trị expected: .
3. Tính ulps: .
4. Chuẩn hoá RE theo ULP.

|  |  |
| --- | --- |
| **0.02 và 0.017(p=3)** | **3.14 và 3.14159(p=3)** |
| **99.2 và 98.8** | **12.35 và 12.4** |
| **0.1 và 0.0292** | **0.03 và 0.03480 (p =4)** |

Mất mát thông tin trong quá trình tính toán.

1. Cộng 2 số có chênh lệch quá lớn (Phần mũ quá khác biệt)

(Mất mát trung bình) (Loss of significance).

1. Trừ 2 số gần bằng nhau

(Mất mát lớn) (Loss of significance).

1. Nhân 2 số

Underflow hoặc Overflow nếu phần mũ của 2 số quá lớn hoặc quá nhỏ.

1. Chia 2 số

Có thể gây ra underflow.

**Đối với phép cộng**

Cộng 2 số có chênh lệch quá lớn (Phần mũ quá khác biệt) vấn đề này được gọi là “Loss of significance”, khi máy thực hiện phép cộng ta cần chuẩn hoá 2 số về cùng phần mũ (exponent) để thực hiện phép cộng, việc dịch bit về bên phải sẽ làm cho dãy bit tràn ra ngoài và rò rĩ lượng bit này.

**Đối với phép trừ**

Xảy ra khi trừ hai số thực mà hai số này gần nhau. Kết quả sẽ có thể mang lại một giá trị gần đúng rất xấu so với số chính xác, vấn đề này được gọi là “Catastrophic cancellation” **[7]** được hiểu như một sự mất mát nghiêm trọng.

Chứng minh**[7]**:

Gọi và với Ta có:

Do đó, Related error của phép trừ hai số sấp sỉ ta có thể thấy được là: . Sai số này sẽ rất lớn và tệ hại nếu và là những số gần nhau (Có nghĩa là với ta có ).

**ĐẢM BẢO ĐỘ CHÍNH XÁC TÍNH TOÁN BẰNG GRS**

**(Được sử dụng trong IEEE-754)**

Trong quá trình tính toán 3 bit GRS sẽ được thêm vào như bên dưới như những nhân tố trung gian để đảm bảo độ chính xác của phép tính. Sau khi quá trình tính toán kết thúc, các bit này sẽ được huỷ đi nhằm giảm thiểu chi phí phần cứng.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sign | Exponent | Fraction | Guard | Round | Sticky |

**G (Guard bit):** Chỉ cần 1 bit để tránh đi sự mất mát đáng kể của việc dịch bit sang phải thoát khoải vùng lưu trữ khi thực hiện việc chuẩn hoá, (để giải thích rõ hơn[8]).

**R(Round bit) và S(Sticky bit):** Đây là 2 bit cần thiết cho việc việc làm tròn (rounding).

|  |  |
| --- | --- |
| Guard bit | Round bit và Sticky bit |
| テキスト  低い精度で自動的に生成された説明 | テキスト が含まれている画像  自動的に生成された説明 |

Nguồn: <https://slideplayer.com/slide/14875529/>

Quy tắc làm tròn dựa trên GRS:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| g | r | s | Bit cuối Fraction | Action |
| 0 | --- | --- | --- | Truncate |
| 1 | 0 | 1 | --- | Increament |
| 1 | 1 | 0 | --- |
| 1 | 1 | 1 | --- |
| 1 | 0 | 0 | 0 | Truncate |
| 1 | 0 | 0 | 1 | Increament |

**Note:** Truncate đơn giản là xoá đi phần bit cuối sau dãy Fraction và Increament là việc cộng bit 1 vào phần Fraction.

**BẢNG TÓM TẮT VỀ CHUẨN IEEE-754**

|  |  |  |  |
| --- | --- | --- | --- |
| Single-Precision | Exponent = 8 | Fraction = 23 | Value |
| Normalized | 1…254 | Anything |  |
| Denormalized | 0 |  |  |
| Zero | 0 | 0 |  |
| Infinity | 255 | 0 |  |
| NaN | 255 |  | NaN |

|  |  |  |  |
| --- | --- | --- | --- |
| Double-Precision | Exponent = 11 | Fraction = 52 | Value |
| Normalized | 1…2046 | Anything |  |
| Denormalized | 0 |  |  |
| Zero | 0 | 0 |  |
| Infinity | 2047 | 0 |  |
| NaN | 2047 |  | NaN |

**Các quy tắc tính toán đặc biệt**

|  |  |
| --- | --- |
| **Phép tính** | **Kết quả** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

Nguồn: <http://indico.ictp.it/event/7659/session/10/contribution/44/material/1/0.pdf>

References:

1. <https://sites.ualberta.ca/~kbeach/comp_phys/fp_err.html>

2. <https://slideplayer.com/slide/14875529/>

3. <https://slideplayer.com/slide/14875529/>

4. <https://expressmagazine.net/posts/view/1252/cac-phep-toan-can-ban-tren-so-nhi-phan>

5. Bài giảng kiến trúc máy tính và hợp ngữ (Thầy Phạm Tuấn Sơn – ĐH Khoa Học Tự Nhiên).

6. <https://www.geeksforgeeks.org/ieee-standard-754-floating-point-numbers/>

7. <https://en.wikipedia.org/wiki/Catastrophic_cancellation>

8. <https://docs.oracle.com/cd/E19957-01/806-3568/ncg_goldberg.html>

9. <http://pages.cs.wisc.edu/~david/courses/cs552/S12/handouts/guardbits.pdf>

Tool convert online:

<https://www.h-schmidt.net/FloatConverter/IEEE754.html>

**Phần viết chương trình**

Viết chương trình chuyển đổi số float.

1. Chuyển đổi từ 10 sang 2.
2. Chuyển đổi từ 2 sang 10.
3. Cho nhập vào chuẩn đang sử dụng.
4. Tính toán sai số khi chuyển đổi.
   1. Absolute error
   2. Relative error
   3. Error tính theo epsilon.
5. Hàm các phép tính trên số thực.
   1. Phép cộng
   2. Phép trừ
   3. Phép nhân
6. Hàm mô phỏng các giá trị và xuất ra bản báo cáo.

Phần sử dụng báo cáo đã duyệt để kiểm chứng các kết quả thực nghiệm.

Reference: <https://www.geeksforgeeks.org/program-for-conversion-of-32-bits-single-precision-ieee-754-floating-point-representation/>