Ayudantía
SCT - Semana - 02

Nota: las respuestas y otras notas útiles aparecerán en cajas de color verde.

1. En la próxima pelicula de Marvel^{TM}, *Captain America: Civil War*, Tony Stark debe trabajar en condiciones extremas y su armadura sufrirá grandes daños. En la configuración original de la armadura se utiliza doble precisión, sin embargo luego de utilizar la armadura en batallas de grueso calibre ha perdido 5 bits.

Considere que los 5 bits mencionados se pueden perder de los bits utilizados en la mantisa o en el exponente. Si los bits se pierden en el exponente, se debe ajustar el shift y los casos especiales según corresponda.

Recuerda que en doble precisión hay:

- 1 bit de signo
- 11 bits de exponente (para representar $2^{11} = 2048$ diferentes valores en máquina)
- 52 bits de mantisa (para representar 2^{52} diferentes valores en máquina)

a) Considerando que Tony Stark hace cálculos utilizando números en el rango $[2^{-30},\ 2^{30}]$ y requiere tener la mayor precisión disponible, ¿es más conveniente perder los bits de la mantisa o del exponente de la representación de punto flotante?

Entre 2^{-30} y 2^{30}, existen 61 valores posibles que puede tomar el exponente:

- 30 valores entre 1 y 30
- 30 valores entre -1 y -30
- el valor 0

A Tony Stark le bastan solo 6 bits para representar estos 61 valores, pues 6 bits pueden representar $2^6 = 64$ valores diferentes.

Más específicamente, en una máquina que use 6 bits de exponente, de los 64 valores posibles, se usarían: los 62 valores del medio para representar directamente los exponentes entre -30 y 31 aplicando un bias de $2^5 - 1 = 31$; el valor más bajo (000000) para representar números subnormales y el 0; y el valor más alto (111111) para representar el infinito y el NaN.

En doble precisión, hay 11 bits para el exponente. Si Tony Stark perdiera 5 bits en el exponente, le quedarían 6. Esto no es un problema, porque 6 bits son suficientes.

Por otro lado, si perdiera 5 bits en la mantisa que normalmente tiene 52, pasaría de poder representar valores con 52 dígitos de precisión a valores con hasta solo 47 dígitos de precisión, lo cual sí le afecta porque requiere la mayor precisión posible.

Por lo tanto, le conviene más perder los bits del exponente.

b) Si se eliminan los 5 bits de la mantisa, ¿se modifica el valor de ϵ_{mach}? Si su respuesta es positiva, obténgalo.

ϵ_{mach} es la diferencia entre 1 y el número inmediatamente mayor a 1 que se pueda representar.

El valor de ϵ_{mach} depende de cuántos bits se usan para la mantisa: si se pierden bits, su valor sí cambia. Si, normalmente, hay 52 bits de mantisa, indicando que se puede representar valores con 52 dígitos de precisión, entonces el número binario que le sigue a 1 es

$$1.\underbrace{0000000...0}_{51\text{ ceros}}1$$

donde los primeros 51 de los 52 dígitos después de la coma son 0, y el último es 1. Entonces, ϵ_{mach} es normalmente:

$$\epsilon_{\text{mach}} = 1.\underbrace{0000000...0}_{51\text{ ceros}}1 - 1 = 0.\underbrace{0000000...0}_{51\text{ ceros}}1 = 2^{-52}$$

Sin embargo, si Tony Stark pierde 5 bits de la mantisa, pasa de tener 52 dígitos a solo 47, lo que hace que el número siguiente a 1 sea:

$$1.\underbrace{00...0}_{46}1$$

y esto provoca que ϵ_{mach} aumente su valor a:

$$\tilde{\epsilon}_{\text{mach}} = 1.\underbrace{00...0}_{46}1 - 1 = 0.\underbrace{00...0}_{46}1 = 2^{-47}$$

En general, en una máquina con m bits de mantisa, $\epsilon_{\text{mach}} = 2^{-m}$.

c) Si se eliminan los 5 bits de la mantisa, ¿cuál es el menor número (mayor que 0) representable?

Como todavía hay 11 bits en el exponente, estos pueden representar $2^{11} = 2048$ valores distintos en máquina, de los cuales: los 2046 del medio se usan para representar los exponentes entre -1022 y 1023; el más alto (11111111111) se usa para el infinito y el NaN; y el más bajo (00000000000) se usa para los "números subnormales" y el 0.

Los números subnormales permiten acceder a números aún más pequeños de lo normal, más pequeños que 2^{-1022}, a cambio de perder precisión. Sigue habiendo un producto por 2^{-1022} (donde -1022 es el exponente normal más bajo que se puede obtener), pero la parte entera del factor izquierdo ya no es 1, sino 0.

Por lo tanto, el número positivo más pequeño se obtiene con el menor exponente normal posible ($-\text{bias}+1 = -(2^{p-1}-1)+1 = -1023+1 = -1022$) y la menor mantisa posible que no sea 0. En este caso, como perdimos 5 bits de mantisa y nos quedan 47, este es un número cuyos dígitos después de la coma son 46 ceros y un 1 al final. **El número positivo más pequeño es, entonces:**

$$0.\underbrace{00...0}_{46}1 \times 2^{-1022} = 2^{-47} \times 2^{-1022} = 2^{-1069}$$

En general, el número más pequeño en una máquina de m bits de mantisa y p bits de exponente está dado por $\epsilon_{\text{mach}} \times 2^{-\text{bias}+1}$, es decir, $2^{-m} \times 2^{-2^{p-1}+2}$.

d) Si se eliminan los 5 bits del exponente, ¿cuál es el primer entero que no se puede representar?

Esta es una pregunta trampa, porque, **normalmente, perder bits en el exponente no influye** en cuál es el primer entero no representable, que depende principalmente de los bits en la mantisa. **La excepción es cuando la pérdida es demasiado grande, y en esta pregunta sí es el caso.**

Hay que entender que, por ejemplo, para valores de exponente 0, como por ejemplo el 1, que se representa así:

$$1.\underbrace{0000000...00}_{52\text{ ceros}} \times 2^0$$

un incremento en el último bit de la mantisa incrementa el valor en

$$0.\underbrace{0000000...0}_{51\text{ ceros}}1 \times 2^0 = 2^{-52} = \epsilon_{\text{mach}}$$

pero, para valores de exponente más grande, como el valor 4 que tiene un exponente 2:

$$1.\underbrace{0000000...00}_{52\text{ ceros}} \times 2^2$$

incrementar el último bit resulta en un cambio o gap más grande:

$$0.\underbrace{0000000...0}_{51\text{ ceros}}1 \times 2^2 = 2^{-52} \times 2^2 = 2^{-50} = 4\epsilon_{\text{mach}}$$

En general, si el exponente es x, el gap está dado por $2^{-52} \times 2^x = 2^{x-52}$, o $2^x\epsilon_{\text{mach}}$.

Hasta cierto punto, todos los enteros son representables. Sin embargo, cuando el exponente x se vuelve muy grande, el incremento o gap $2^x\epsilon_{\text{mach}}$ puede terminar siendo mayor a 1, lo que significa que nos iremos saltando algunos enteros.

Para encontrar el valor x para el cual esto ocurre:

$$\text{gap} = 2^x \times \epsilon_{\text{mach}} > 1 \quad \Longrightarrow \quad 2^x \times 2^{-52} > 1 \quad \Longrightarrow \quad 2^{x-52} > 2^0 \quad \Longrightarrow \quad x - 52 > 0 \quad \Longrightarrow \quad x > 52$$

Es decir, en $x = 52$, el incremento al aumentar el último bit es de exactamente 1, y todos los enteros con ese exponente se pueden representar. Sin embargo, al llegar a $x = 53$, llegamos al número entero $1.000... \times 2^{53}$, a partir del cual el incremento o gap pasa a ser 2. Es decir, solo podemos pasar del 2^{53} al $2^{53} + 2$, saltándonos el valor del medio, $2^{53} + 1$. Este es el primer entero que no se puede representar en precisión doble.

Normalmente, los bits del exponente no influyen en este valor que está principalmente determinado por la mantisa (m bits de mantisa normalmente implican que el primer entero no representable es $2^{m+1} + 1$).

El único caso en que los bits del exponente sí cambiarían este valor es cuando, de partida, no se puede llegar al exponente 53 debido a que hay muy pocos bits.

En este caso, si Tony Stark pierde 5 bits de exponente, los 6 restantes solo le permiten representar exponentes entre -30 y 31. No se puede representar el exponente 53.

Por lo tanto, en realidad, el primer entero no representable bajo estas condiciones es 2^{32}, no $2^{53}+1$. Todos los enteros con exponente 31 sí se pueden representar, así que es 2^{32} el primero no representable.

En general, este valor en una máquina de m bits de mantisa y p de exponente está dado por el mínimo entre estos dos valores:

- $2^{m+1} + 1$, que ocurre cuando el exponente no interfiere
- $2^{\text{bias}+1} = 2^{2^{p-1}}$, que ocurre cuando sí interfiere

e) Si se eliminan los 5 bits del exponente, ¿cuál es el menor número (mayor que 0) representable?

Similar a la pregunta c), hay que usar el menor exponente y mantisa posibles.

En este caso, como solo quedan 6 bits de exponente, se puede representar $2^6 = 64$ valores de los cuales 62 se usan para representar los exponentes entre -30 y 31. Por lo tanto, la potencia con el menor exponente posible es 2^{-30}.

Con los 52 bits de mantisa intactos, la menor mantisa posible es aquella que consiste de 51 ceros y 1 uno. En números subnormales, corresponde a

$$0.\underbrace{0000000...0}_{51 \text{ ceros}}1 = 2^{-52}$$

Por lo tanto, el menor número positivo representable es:

$$2^{-52} \times 2^{-30} = 2^{-82}$$