MPI* Info

Représentation des Nombres

TD

Hasley William

1 Exercice 1

Question 1. Avec 4 bits, nous pouvons représenter $2^4 = 16$ valeurs distinctes. Le caractère signé de ces valeurs nous indique que nous pouvons représenter l'intervalle d'entiers $[\![-8, 7]\!]$ avec ces quatre bits.

Question 2. Nous obtenons ainsi pour représentation :

- $0 \quad = \overline{0000}^2$
- $1 \quad = \overline{0001}^2$
- $-1 \quad = \overline{1111}^2$
- $2 \quad = \overline{0010}^2$
- $-5 \quad = \overline{1011}^2$
- $7 \quad = \overline{0111}^2$

L'entier représenté par $\overline{1000}^2$ est -8, car le complément à deux redonne la même écriture binaire, qui correspond à l'écriture binaire non signée de 8. (autrement, $x + 2^4 = x + 16 = 8 \implies x = -8$). Alors que $\overline{0101}^2 = 5$.

2 Exercice 2

Question 1. Le décalage vaut $e = 2^{3-1} - 1 = 3$. Les exposants possibles sont donc $\{-3, -2, -1, 0, 1, 2, 3, 4\}$. On rappelle que le décalage e vaut $2^{(n-1)} - 1$ pour n bits d'exposant, et que la puissance calculée est $p - e$. Nous retirerons néanmoins les puissances représentées par les écritures $\overline{000}^2$ et $\overline{111}^2$ pour des raisons qui apparaîtrons plus tard. La plage effective des puissances disponible est alors $[\![-2, 3]\!]$

Question 2. Nous obtenons ainsi pour représentation :

- $1 \quad = \overline{00110000}^2 = (-1)^0 \times 2^{(3-3)} \times 1.000$
- $-1 \quad = \overline{10110000}^2 = (-1)^1 \times 2^{(3-3)} \times 1.000$
- $\frac{1}{2} \quad = \overline{00100000}^2 = (-1)^0 \times 2^{(2-3)} \times 1.000$
- $-5 \quad = \overline{11010100}^2 = (-1)^0 \times 2^{(5-3)} \times 1.25$

Question 3. Par définition de la mantisse (comprise dans $[1, 2[$), pour trouver $x = 1 + \epsilon$, le plus petit nombre strictement supérieur à 1, nous devons avoir un exposant donnant 1.

Ainsi, $x = \overline{00110001}^2 = (-1)^0 \times 2^0 \times \left(1 + \frac{1}{16}\right) = \frac{17}{16} = 1.0625$

Question 4. Un calcul direct donne $\overline{11100101}^2 = (-1) \times 2^{(6-3)} \times \left(1 + \frac{1}{4} + \frac{1}{16}\right) = -8 - 2 - \frac{1}{2} = -10.5$

Question 5. Rappelons que les nombres représentés sont dits normalisés lorsque le bit implicite de la mantisse vaut 1 (en notation scientifique, il est attendu d'avoir un chiffre des unités non-nul, or en binaire, seul 1 convient). Lorsque l'écriture binaire de l'exposant est différente de $\overline{000}^2$ et $\overline{111}^2$, le nombre est normalisé. Sinon, ce nombre est dit "dénormalisé", et le bit implicite de la mantisse devient zéro. Ce format sert à gérer les zéros, $+\infty$ et NaN.

Ainsi, le plus petit nombre strictement positif normalisé provient de l'écriture $\overline{00010000}^2 = 2^{(1-3)} = 0.25$, alors que le plus grand strictement positif normalisé s'écrit $\overline{01101111}^2 = 2^{(6-3)} \times \left(1 + \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \frac{1}{16}\right) = 8 + 4 + 2 + 1 + 0.5 = 15.5$

Question 6. Nous remarquons que la meilleure approximation inférieure de π se fait avec un exposant biaisé de 1, via l'écriture binaire $\overline{01001001}^2 = 3.125$. De même, la meilleure approximation supérieure donne $\overline{01001010}^2 = 3.25$.

Ainsi, la meilleure approximation de π est l'approximation inférieure : $\overline{01001001}^2 = 3.125$

3 Exercice 3

La fonction suivante convient :

```c
int pow_of_ten(void){
  int cur_pow = 0;
  double cur_ten = 1;
  while(cur_ten / 10.0 > 0){
    cur_pow--;
    cur_ten /= 10.0;
  }

  return cur_pow;
}
```

Nous obtenons via cette fonction : $k = -323$.

4 Exercice 4

Question 1. Le décalage vaut $2^{(5-1)} - 1 = 15$.

Question 2. Les nombres donnés ont pour représentation :

- $0 = \overline{0000000000000000}^2$
- $1 = \overline{0011110000000000}^2 = (-1)^0 \times 2^{(15-15)} \times 1.000$
- $-1 = \overline{1011110000000000}^2 = (-1)^1 \times 2^{(15-15)} \times 1.000$
- $2 = \overline{0100000000000000}^2 = (-1)^0 \times 2^{(16-15)} \times 1.000$
- $7 = \overline{0100011100000000}^2 = (-1)^0 \times 2^{(17-15)} \times 1.75$

- $-259 \quad = \overline{1101110000001100}^2 = (-1)^0 \times 2^{(23-15)} \times 1.01171875$

Question 3. De même que précédemment, le plus petit nombre strictement supérieur à 1 est représenté par $\overline{0011110000000001}^2$, et vaut $1 + \frac{1}{2^{10}} = 1.0009765625$

Question 4. Le plus petit nombre normalisé est représenté par $\overline{0000010000000000}^2$ et vaut $2^{-14} \simeq 6.1 \times 10^{-5}$. Alors que le plus grand nombre normalisé est représenté par $\overline{0111101111111111}^2$, valant 65504.

Question 5. L'écriture $\overline{0100001001001000}^2$ représente le nombre $2^{(16-15)} \times \left(1 + \frac{1}{2} + \frac{1}{16} + \frac{1}{128}\right) = 3.140625$

5 Exercice 5

Question 1. La fonction suivante convient :

```
bool test_associativite(double a, double b, double c){
  return (((a + b) + c) == (a + (b + c)));
}
```

Question 2. Un exemple classique est le triplet 0.1, 0.2 et 0.3. Nous pouvons vérifier que le test d'associativité échoue sur cette entrée.

Question 3. Afin de répondre à cette question, nous proposons l'algorithme suivant :

```
double probabilite_assoc(void){
  long success = 0;
  for(int a = 0; a < 100; a++){
    for(int b = 0; b < 100; b++){
      for(int c = 0; c < 100; c++){
      if(test_associativite(a / 10.0, b / 10.0, c / 10.0)) success++;
      }
    }
  }

  return ((double)success) / (101.0 * 101.0 * 101.0);
}
```

Nous obtenons une probabilité de ne pas être associatif valant 24.6%

Question 4. Les cas limites NaN montrent que NaN $\neq$ NaN (en effet, NaN est réalisé par un exposant rempli de 1, et une mantisse non-nulle. Ceci mène à pouvoir représenter NaN de deux manières différentes, et à affirmer leur différence).

Question 5. La multiplication n'est pas associative, et le même triplet $0.1, 0.2$ et 0.3 le montre.

Question 6. Le triplet $0.1, 0.2, 0.3$ montre que la multiplication flottante n'est pas distributive : $0.2 \times (0.1 + 0.3) \neq 0.2 \times 0.1 + 0.2 \times 0.3$