Initiale Permutation der DES-Chiffre

Yusuf Enes Tamer

Juni 2025

1 Permutation auf Basis einer alternierenden Funktion

Es geht um den Permutation-Algorithmus des Data Encryption Standards. Mit der ist es möglich die Anordnung der vorgegeben 64 Bit-Werte zu vertauschen. Diese 64-Bit-Werte können in einer Matrix befinden, weshalb die Vertauschungen in einem 2D-Feld stattfinden. Ein Element im 2D-Feld wird durch die Zeilennnummer sowie auch gleichzeitig von der Spaltennummer addressiert. Bei einer höheren Speicherkomplexität kann das 2D-Feld dupliziert werden, sodass die Bit-Werte in einer unterschiedlichen Anordnung in ein weiteres Feld geladen werden. Die Vielfachheit eines Feldes kann sich positiv auf die Laufzeit wirken, sodass die Permutation in einer schneller Zeitkomplexität stattfindet.

Das 2D-Feld wird folgendermaßen nummeriert:

```
    5.  1.  6.  2.  7.  3.  8.  4.
1.
2.
3.
4.
5.
6.
7.
8.
```

Die Spaltennummer legt die Byte-Nummer fest, dagegen die Zeilennummer die Bit-Reihenfolge. Die unterschiedliche Anordnung der Byte-Nummern ist folgendermaßen festgelegt worden:

```
(8-3)  -> 5  <-----+
                   |  -4
(8-7)  -> 1  <---- +
                   |  +5
(8-2)  -> 6  <-----+
                   |  -4
(8-6)  -> 2  <-----+
                   |  +5
(8-1)  -> 7  <-----+
                   |  -4
(8-5)  -> 3  <-----+
                   |  +5
(8-0)  -> 8  <-----+
                   |  -4
(8-4)  -> 4  <-----+
```

Die Permutation wird damit auf Basis einer alternierenden Addition/ Subtraktion Operation durchgeführt.Dieser Algorithmus kann in der Programmier-Sprache (C++) geschrieben werden:

```
for ( int j=0; j<8; j++){
BitArrayV2[j][1 - 1]=BitArray[5 - 1][j];
BitArrayV2[j][2 - 1]=BitArray[1 - 1][j];
BitArrayV2[j][3 - 1]=BitArray[6 - 1][j];
BitArrayV2[j][4 - 1]=BitArray[2 - 1][j];
BitArrayV2[j][5 - 1]=BitArray[7 - 1][j];
BitArrayV2[j][6 - 1]=BitArray[3 - 1][j];
BitArrayV2[j][7 - 1]=BitArray[8 - 1][j];
BitArrayV2[j][8 - 1]=BitArray[4 - 1][j];}
```

$$\sum_{n=1}^{4} 5n - \sum_{j=n-1}^{n} 4j$$

Somit ist das Ziel die Permutation mindestens auf 2 Feldern durchzuführen, wo dabei die Zeilen und Spalten alterniert werden (Mirror). Diese Alternierung kann mit der vorliegenden Summen-Formel berechnet werden.

```
        for (int i = j; i<j+8; i++)
        { if (i <= 3)
        {
BitArrayV3[0][i] = BitArrayV2[8 - 2][7 - 1 - 2 * i];
BitArrayV3[1][i] = BitArrayV2[8 - 4][7 - 1 - 2 * i];
BitArrayV3[2][i] = BitArrayV2[8 - 6][7 - 1 - 2 * i];
BitArrayV3[3][i] = BitArrayV2[8 - 8][7 - 1 - 2 * i];
BitArrayV3[4][i] = BitArrayV2[8 - 1][7 - 1 - 2 * i];
BitArrayV3[5][i] = BitArrayV2[8 - 3][7 - 1 - 2 * i];
BitArrayV3[6][i] = BitArrayV2[8 - 5][7 - 1 - 2 * i];
BitArrayV3[7][i] = BitArrayV2[8 - 7][7 - 1 - 2 * i];}
        else
                {switch (i)
                        {
                        case 4:
                                BitArrayV3[0][i] = BitArrayV2[8 - 2][7];
```

Nachdem das Mirror-Feld erstellt wurde kann die einzelne Selektion von den jeweiligen Bits beginnen. Dafür wird eine weitere Formel verwendet, die für die Adressierung dazu da ist:

$$\sum_{x=0}^{1}\sum_{i=1}^{4}(8-2i-x)$$

Der zweite Index kann folgendermaßen berechnet werden:

$$\sum_{i=0}^{3}(7-2*i-1)+(7-2*i)$$