



Arhitectura Sistemelor de Calcul



Computer Science
& Engineering
Department

Universitatea Politehnica Bucuresti
Facultatea de Automatica si Calculatoare

cs.ncit.pub.ro

curs.cs.pub.ro



Permutari Elementare

2

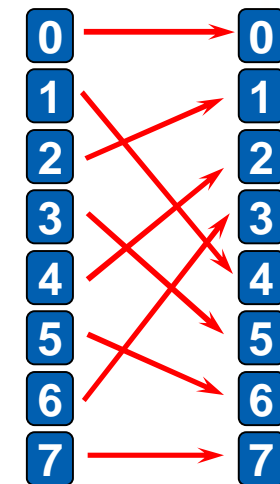
- Specifica conexiunea între n resurse ca o funcție bijectivă pe o mulțime ordonată
- Se pot realiza în mai multe feluri:
 1. Permutarea de bază – Base Line Permutation
 2. Permutarea cu intercalare perfectă – Shuffle Permutation
 3. Permutarea cu negare de bit – Negate Bit Permutation
 4. Permutarea fluture – Butterfly Permutation
 5. Permutarea cu ordine inversă – Bit Reversal Permutation
 6. Permutarea cu incrementare modulo – Increment Permutation
- Definirea acestor permutări se poate face pe baza reprezentării binare a adreselor resurselor din mulțime (Flanders): $ADR = (a_n a_{n-1} \dots a_1 a_0)$



1 – Base Line Permutation

3

- Bitul cel mai nesemnificativ devine cel mai semnificativ:
 $P_base(a_n a_{n-1} \dots a_1 \underline{a_0}) = (\underline{a_0} a_n a_{n-1} \dots a_1)$
- Permutarea de baza este intercalarea perfecta inversa!
- Pentru 8 resurse avem $P_base(a_2 a_1 \underline{a_0}) = (\underline{a_0} a_2 a_1)$
- Variante posibile sunt
 - Permutarea de baza k inferioara:
 - Cei mai nesemnificativi k biti din adresa
 - $P_b_kinf(a_n \dots a_k \underline{a_{k-1} \dots a_1 a_0}) = (a_n \dots a_k \underline{a_0} a_{k-1} \dots a_1)$
 - Asigura conexiunea **in** submultimi de resurse
 - Permutarea de baza k superioara:
 - Cei mai semnificativi k biti din adresa
 - $P_b_ksup(a_n \dots \underline{a_{n-k+1} a_{n-k}} \dots a_0) = (\underline{a_{n-k+1}} a_n \dots a_{n-k+2} a_{n-k} \dots a_0)$
 - Asigura conexiunea **intre** submultimi de resurse
- TA: Variantele p_base pentru 8 resurse





2 – Shuffle Permutation

4

- Bitul cel mai semnificativ devine cel mai nesemnificativ:

$$P_shuffle(\underline{a_n a_{n-1} \dots a_1 a_0}) = (a_{n-1} \dots a_1 a_0 \underline{a_n})$$

- Pentru 8 resurse avem $P_shuffle(\underline{a_2 a_1 a_0}) = (a_1 a_0 \underline{a_2})$

- Variante posibile sunt

- Permutarea shuffle k inferioara:

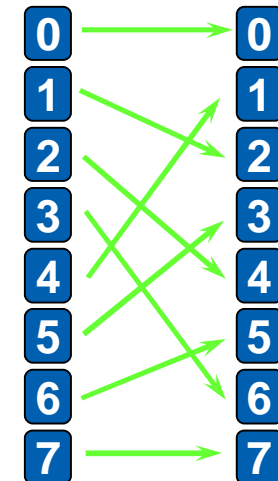
- Cei mai nesemnificativi k biti din adresa
- $P_shuffle_kinf(a_n \dots a_k \underline{a_{k-1} \dots a_1 a_0}) = (a_n \dots a_k \dots a_1 a_0 \underline{a_{k-1}})$
- Utilizare posibila: conexiunea memoriilor – adresa de baza a unui modul + adresa in cadrul acelui modul

- Permutarea shuffle k superioara:

- Cei mai semnificativi k biti din adresa
- $P_shuffle_ksup(\underline{a_n \dots a_{n-k+1}} a_{n-k} \dots a_0) = (a_{n-1} \dots a_{n-k+1} \underline{a_n} a_{n-k} \dots a_0)$

- Permutarea shuffle q # c, intre q·c resurse (nu putere a lui 2):

- $P_shuffle_q\#c(\underline{ADR}) = q \cdot V_{ADR} \bmod (q \cdot c - 1)$ pentru $0 \leq V_{ADR} < q \cdot c - 1$
- $P_shuffle_q\#c(\underline{ADR}) = V_{ADR}$ pentru $V_{ADR} = q \cdot c - 1$
- $V_{ADR} = 0, 1, 2, \text{ etc, } n-1$ succesiv

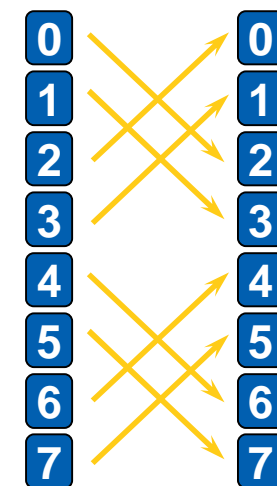
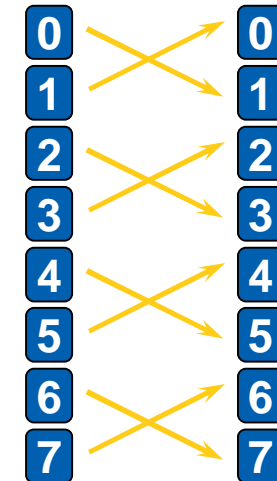




3 – Negate Bit Permutation

5

- Bitul cel mai nesemnificativ este negat
 $P_neg_bit(a_n a_{n-1} \dots a_1 a_0) = (a_n a_{n-1} \dots a_1 \bar{a}_0)$:
- $P_neg_bit(a_2 a_1 a_0) = (a_2 a_1 \bar{a}_0)$
- Varianta posibila este negarea bitului k:
 $P_neg_bit_k(a_n a_{n-1} \dots a_{k-1} \dots a_1 a_0) = (a_n a_{n-1} \dots \bar{a}_{k-1} \dots a_1 a_0)$
- $P_neg_bit(a_2 a_1 a_0) = (a_2 \bar{a}_1 a_0)$





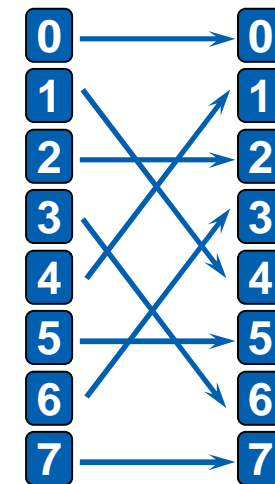
4 – Butterfly Permutation

6

- Se interschimba bitul cel mai semnificativ cu cel mai nesemnificativ:

$$P_butterfly(\underline{a_n \dots a_1 a_0}) = (a_0 a_{n-1} \dots a_1 a_n)$$

- $P_butterfly(a_2 a_1 a_0) = (a_0 a_1 a_2)$
- Butterfly este o permutare simetrica
- Variante posibile sunt



- Permutarea butterfly k inferioara:

- $P_butterfly_kinf(a_n \dots a_k \underline{a_{k-1} \dots a_1 a_0}) = (a_n \dots a_k a_0 \dots a_1 a_{k-1})$
- Oferă posibilitatea de conectare între resursele clasei

- Permutarea butterfly k superioara:

- $P_butterfly_ksup(\underline{a_n \dots a_{n-k+1}} a_{n-k} \dots a_1 a_0) = (a_{n-k+1} a_{n-1} \dots a_n a_{n-k} \dots a_0)$
- Oferă posibilitatea de conectare între clase



5 – Bit Reversal Permutation

7

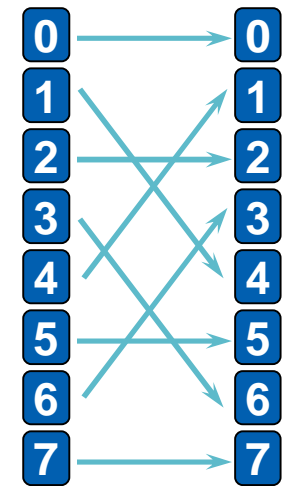
- Se interschimba complet ordinea bitilor:
 $P_{\text{reversal}}(\underline{a_n \dots a_1 a_0}) = (a_0 a_1 \dots a_n)$
- $P_{\text{reversal}}(a_2 a_1 a_0) = (a_0 a_1 a_2)$
- Cu aceasta permutare se pot obtine echivalente
- Variante posibile sunt

- Permutarea reversal k inferioara:

- $P_{\text{reversal_kinf}}(a_n \dots a_k \underline{a_{k-1} \dots a_1 a_0}) = (a_n \dots a_k a_0 a_1 \dots a_{k-1})$

- Permutarea reversal k superioara:

- $P_{\text{reversal_ksup}}(\underline{a_n \dots a_{n-k+1}} a_{n-k} \dots a_1 a_0) = (a_{n-k+1} \dots a_n a_{n-k} \dots a_0)$

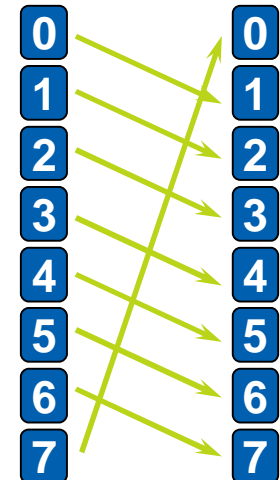




6 – Increment Permutation

8

- Se incrementeaza valorile modulo 2^n :
$$P_increment(ADR) = (V_{ADR} + 1) \bmod (2^{n+1})$$
- Aceasta este o conexiune inelara intre module alaturate
- 1 din $(V_{ADR} + 1)$, poate fi orice constanta
- Variante posibile sunt
 - Permutarea increment k inferioara:
 - Realizeaza conexiuni in cadrul submultimii de resurse
 - Permutarea increment k superioara:
 - Realizeaza conexiuni intre submultimile de resurse





What Next?

9

- Q & A?
- Next time:
 - Arhitecturi MIMD – Caracteristici
 - Sisteme multiprocesor/multicalculator
 - Sisteme cu memorie partajata:
 - UMA
 - NUMA/CC-NUMA
 - COMA
 - Sisteme cu memorie distribuita
 - Sisteme strans/slab cuplate