Exploatarea ierarhiei de memorie

Un exemplu intuitiv

Un student aflat în biblioteca facultăţii are pe masă o serie de cărţi

Subiectul căutat nu se regăseşte în cărţile aflate pe masă

Studentul se reîntoarce la rafturi şi extrage o nouă carte

Existența unui număr mare de cărţi pe masă reduce timpul de căutare

Probabilitatea de căutare nu este aceeaşi pentru toate cărţile din bibliotecă

Un program nu accesează toată secțiunea sa de cod sau de date cu aceeaşi probabilitate.

Principiul localizării - stă la baza modului de operare a programelor

stabilește faptul că programele accesează o porțiune relativ redusă a spațiului lor de adrese la orice moment de timp

Localizarea temporală - localizare în timp - dacă se face referire la un anumit obiect, este posibil ca acesta să fie referit din nou cât de curând

Localizarea spațială - localizare în spațiu - dacă se face referire la un anumit obiect din memorie, obiectele ale căror adrese sunt învecinate cu acesta, tind să fie adresate cât de curând.

Principiul localizării temporare este folosit la implementarea memoriei unui calculator sub forma unei ierarhii de memorie.

O ierarhie de memorii poate fi alcătuită din mai multe niveluri, însă datele la un moment dat sunt copiate doar între două niveluri adiacente.

Unitatea minimă de informație care poate fi prezentă sau absentă într-o ierarhie de memorie se numește bloc.

Unitatea minimă de informație care poate fi prezentă sau absentă într-o ierarhie de memorie se numește bloc.

Regăsirea informației necesare procesorului într-un bloc de memorie superior se numește **HIT**

Neregăsirea datelor pe nivelul superior se numește **MISS**

Rata de succes - fracțiunea din accesele la memorie ce au găsit datele în nivelul superior de memorie.

Rata de eşec = 1 - rata de succes fracţiunea din accesele la memorie care nu au găsit datele în nivelul superior de memorie.

Timpul de succes - reprezintă timpul necesar accesului la un nivel superior al ierarhiei de memorie, ce include și timpul necesar determinării tipului de acces.

Penalizarea de eșec - reprezintă timpul necesar înlocuirii blocului din nivelul superior cu blocul corespunzător din nivelul inferior, incluzând și timpul trimiterii acestui bloc către procesor.

Construirea sistemelor de memorie afectează:

1. modul în care SO-ul administrează memoria și perifericele
2. modul în care compilatoarele generează codul
3. modul în care aplicațiile folosesc mașina de calcul

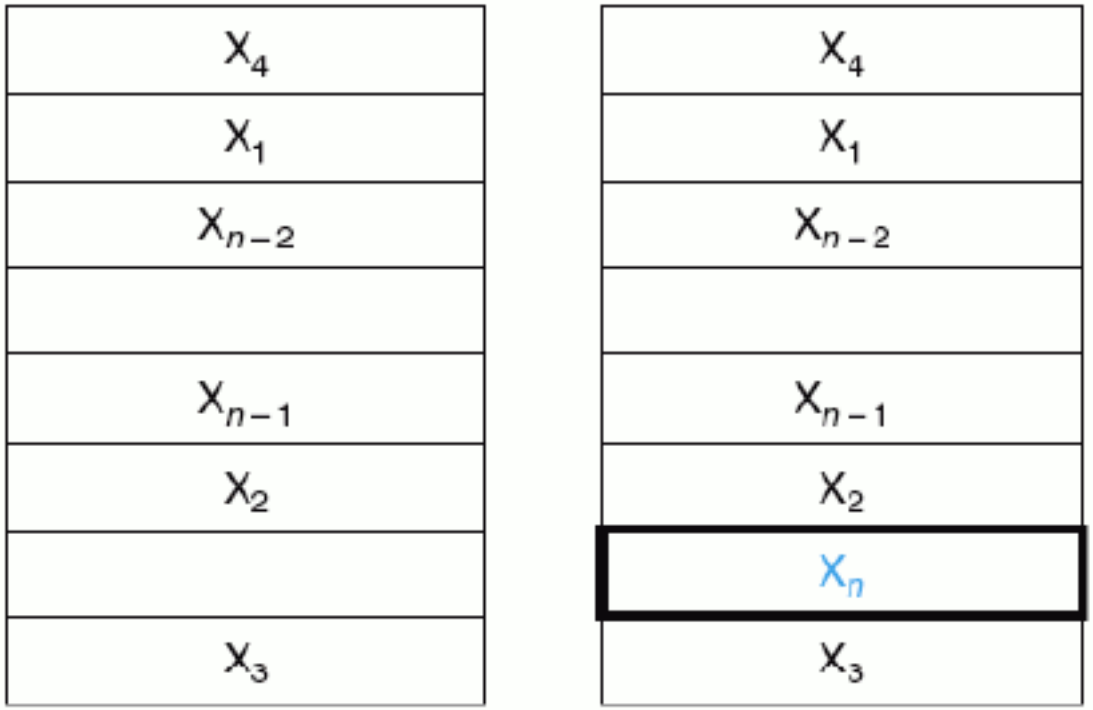
Principiu de bază

Programele prezintă localizare temporală cât și localizare spaţială.

Ierarhiile de memorie folosesc avantajul localizării temporale păstrând datele accesate recent cât mai aproape de procesor.
Ierarhiile de memorie folosesc avantajul localizării spațiale prin mutarea blocurilor conținând cuvinte învecinate din memorie în nivelurile superioare ale ierarhiei.

În anii ’60 s-a folosit cuvântul ***cache*** pentru a desemna nivelul ierarhiei de memorie aflat între UCP și memoria principală.

PRINCIPIILE DE BAZĂ ALE MEMORIILOR CACHE



Iniţial cuvântul de date X_n nu se găsește în cache

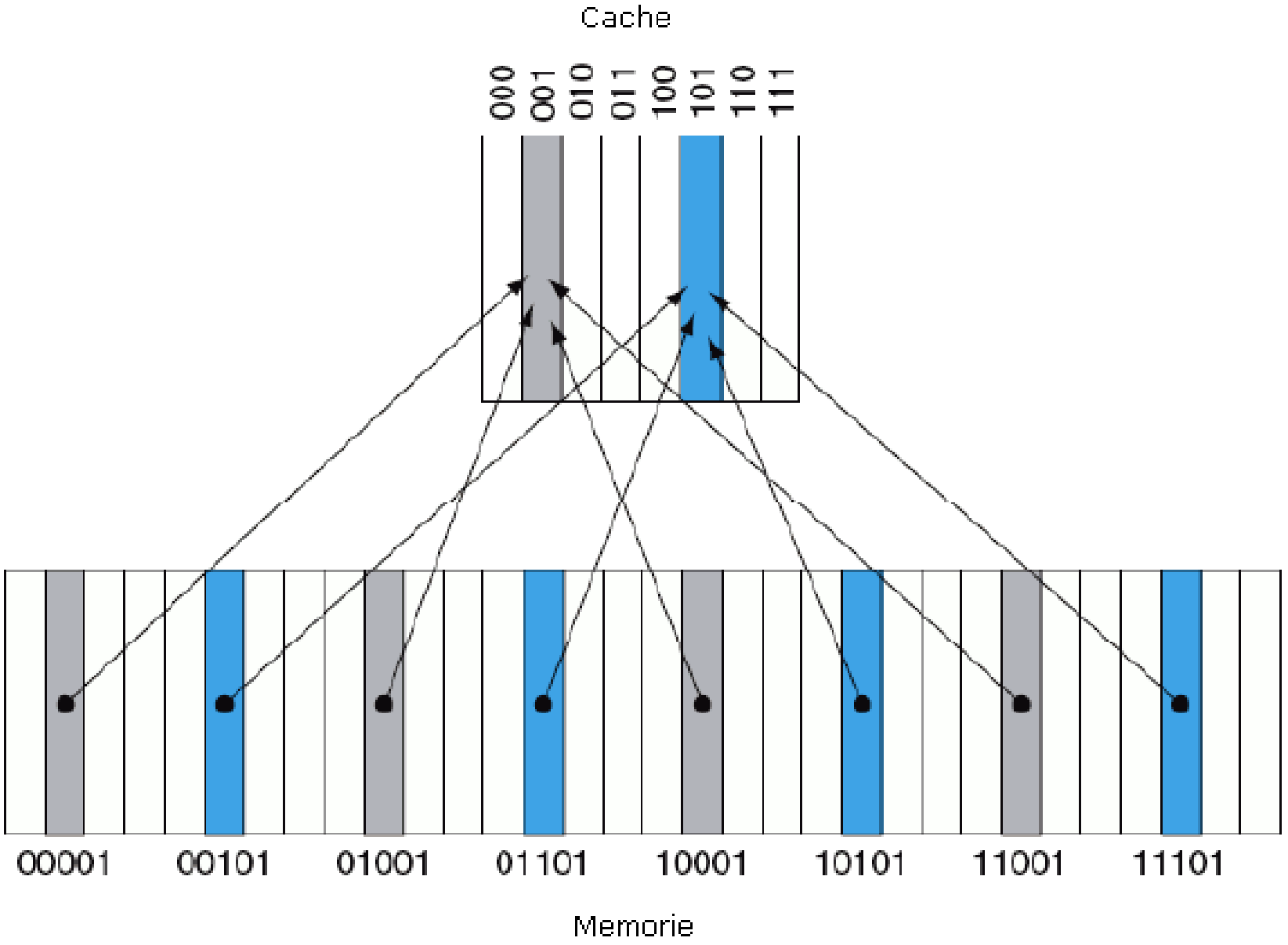
Cum se poate determina dacă o dată este în memoria cache ?

Dacă o dată există în memoria cache cum poate fi ea gasită ?

!!!!! Fiecare cuvânt poate fi memorat doar într-o anumită locație a memoriei cache => corespondenţă directă.

Corespondenţa dintre adrese și locaţiile memoriei cache se determină astfel:

(Adresa blocului) modulo (numărul de blocuri din memoria cache)



Cum determinăm dacă o dată este în memoria cache ?

Cum determinăm dacă o dată este validă sau nu ?

Metoda cea mai des folosită este cea de a aduna un ***bit de validare*** pentru a indica dacă o locație conţine o adresă validă.

Accesarea unei memorii cache cu corespondeță directă

Cerere	Adresa	HIT / MISS	Blocul din cache
22	10110		
26	11010		
22	10110		
26	11010		
16	10000		
3	00011		
16	10000		
18	10010		

Index	V	Marcaj	Date
0	N		
1	N		
10	N		
11	N		
100	N		
101	N		
110	N		
111	N		

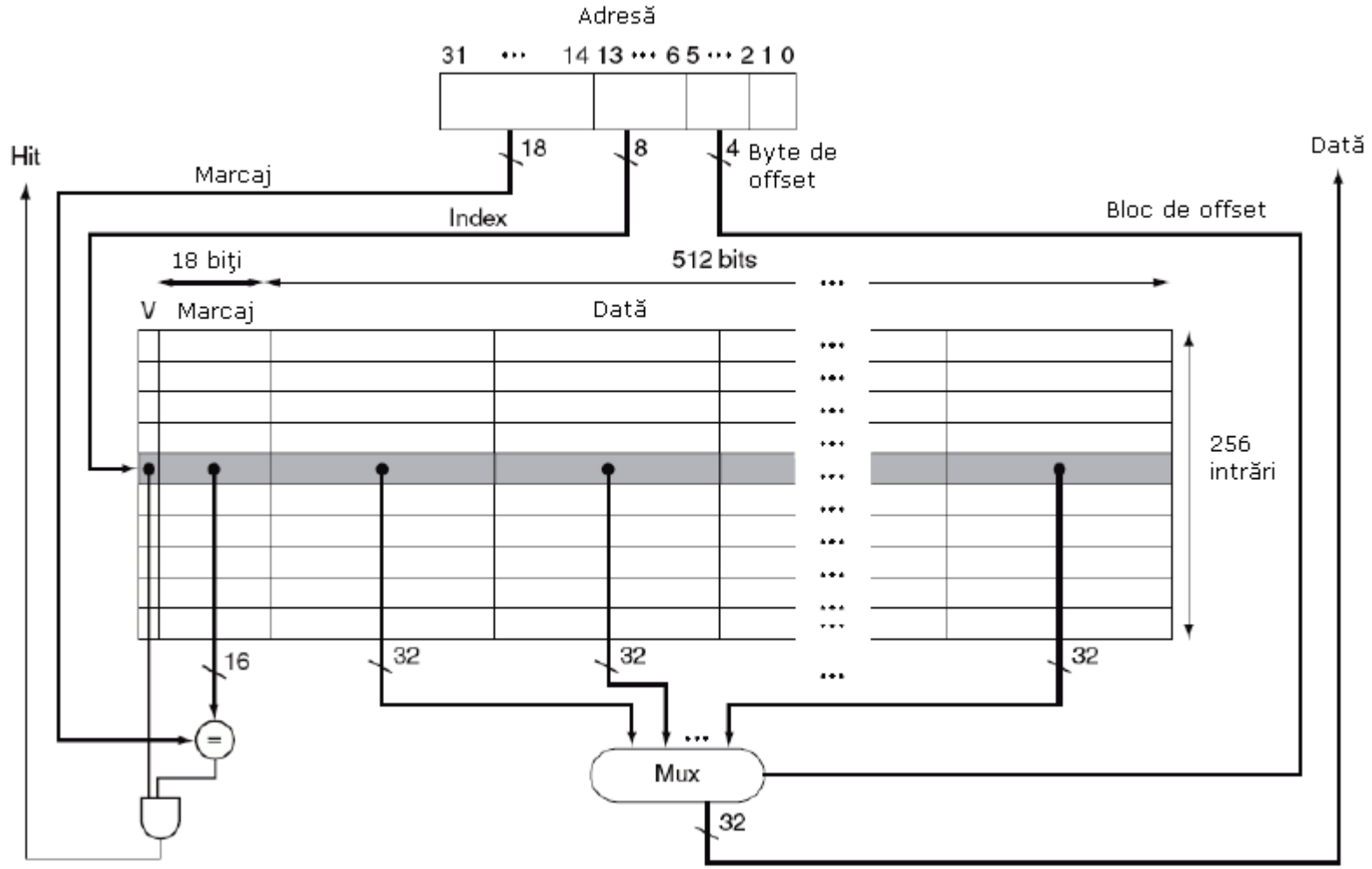
Index	V	Marcaj	Date
0	N		
1	N		
10	N		
11	N		
100	N		
101	N		
110	Y	10	mem (10110)
111	N		

Index	V	Marcaj	Date
0	N		
1	N		
10	Y	11	mem (11010)
11	N		
100	N		
101	N		
110	Y	10	mem (10110)
111	N		

Index	V	Marcaj	Date
0	Y	10	mem (10110)
1	N		
10	Y	11	mem (11010)
11	N		
100	N		
101	N		
110	Y	10	mem (10110)
111	N		

Acest tip de accesare permite folosirea principiului localizării temporale - cuvintele accesate recent le înlocuiesc pe cele accesate mai puţin recent.

FOLOSIREA LOCALIZĂRII SPAŢIALE



Se doreşte ca blocul memoriei cache să fie mai mare decât lungimea unui cuvânt

Determinarea blocului din memoria cache pentru o anumită adresă

(Adresa blocului) modulo (Numărul de blocuri din memoria cache)

unde

Adresa blocului = adresa cuvântului / numărul de cuvinte din bloc

EXEMPLU

Se consideră o memorie cache cu 64 de blocuri de date, fiecare cu dimensiunea de 16 octeți. Care este numărul blocului corespunzător adresei de octet 1200 ?

Eşecurile şi succesele de scriere

Un bloc de date conține mai mult de un cuvânt => nu se poate să scriem doar marcajele și datele.

Considerații:
1. două adrese de memorie X și Y au același bloc corespondent C în memloria cache
2. Blocul are 4 cuvinte și conține adresa Y
3. Scriem la adresa X prin simpla suprapunere a datelor și a marcajului din blocul C

Conform considerațiilor de mai sus, ce se întamplă după operația de scriere ?

Spre deosebire de cazul blocurilor de 1 cuvânt, în cazul blocurilor de date cu mai multe cuvinte, eșecurile la scriere necesită o citire din memorie.

Îmbunătățirea performanței în cazul folosirii principiului de localizare temporară

Presupunem că următorii octeți de adrese sunt ceruți de către un program

16, 24, 20

și nici una din aceste adrese nu se găsește în memoria cache.

Dacă folosim o memorie cache cu blocuri de date de 4 cuvinte, atunci un eșec

la adresa 16 va determina încărcarea în memoria cache a blocului care conține

Adresele 16, 20, 24 și 28.

Din exemplul prezentat se observă că vom avea un singur eșec. Dacă blocul de date ar fi de 1 cuvânt, câte eșecuri am fi avut ?

Măsurarea și îmbunătățirea performanțelor

Timpul de execuție pentru UCP este format din ciclurile de ceas în care UCP execută programul cerut și cele în care UCP-ul așteaptă executarea transferurilor în și din memorie

Timpul UCP = (ciclurile de ceas UCP pt execuție + ciclurile de ceas staționare a UCP

datorate memoriei) x durata unui ciclu de ceas

Ciclurile de ceas staționare datorate memoriei = cicluri staționare pentru citire + cicluri staționare pentru scriere

Ciclurile de ceas de staționare la citire = citiri/program x rata de eșec la citire x penalizarea de eșec la citire

Ciclurile de staționare datorate memoriei = nr de accese la memorie/program x rata de eșec x penalizarea de eșec = nr de accese la memorie/program x nr de eșecuri/instrucțiune x penalizarea de eșec

Exemplul 1 – se va efectua în clasă

Se consideră programul gcc având o rată de eșec de instrucțiuni de 2% și una de date de 4%. Dacă mașina de calcul are un CPI de 2 fără staționări din cauza memoriei și o penalizare de eșec de 40 cicluri de ceas pentru toate eșecurile, să se determine cât de rapid ar funcționa aceeași mașină dacă memoria cache ar fi Considerată ca fiind perfectă., fără eșecuri.