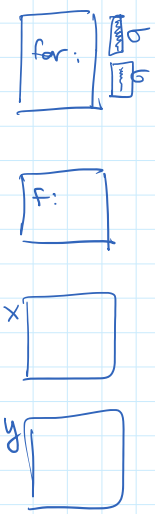
martedì 21 novembre 2017 16:21

POLITICA ON-DEMAND

```
for( ) {
        x[i] = f(y[i]);
}
```



event_process IC

↑

@ for

Cerchi la 1ª (prima) linea del codice della C1

LOAD $R_{base\,y}$, R_i, R_-

caricamento di y[0] ---- y[σ-1]

CALL R_f, R_{ret}

Caricamento del codice f

STORE $R_{base\,x}$, R_i, R_-

Caricamento di x[0] ---- x[σ-1]

(non avrà un fault vero e proprio)

martedì 21 novembre 2017 16:33

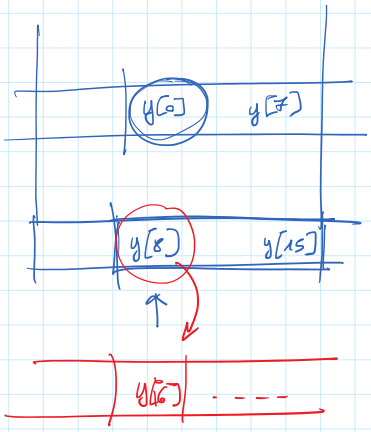
POLITICA PREFETCH

Se accedo ad un indirizzo X ⟶ linea di cache

⇒ faccio in modo asincrono di caricare in cache anche la riga successiva

prefetch(x)
(y)

LOAD




```
for( <N )
    x[i] = f(y[i])
```

$$\#fault = fault(codice) + 2\frac{N}{\sigma}$$

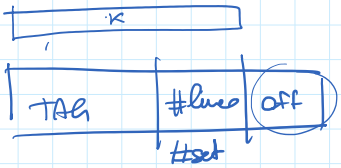
Con prefetch

y[0] - y[7]
x[0] - x[7]

$$= fault(codice) + 2$$

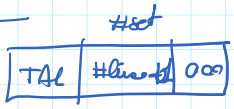
(1)

LOAD
STORE
CLO



k .. ∞∞
k .. (σ-1)

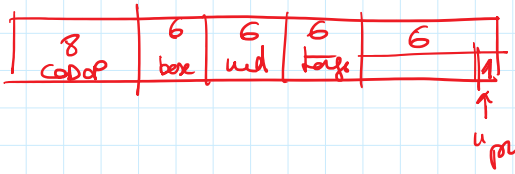
prefetch

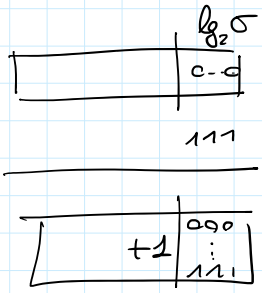


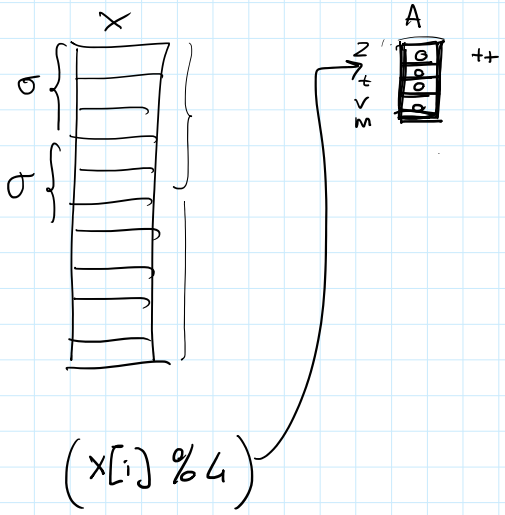

```
-- prefetch(z[17]);
```

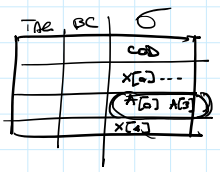
LOAD R_{base}, Rindice, Rtarget, "prefetch"

flag





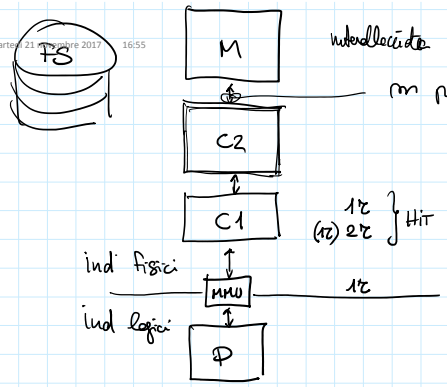




LOAD Rhash A, Rind, Rtag, "non deallocare"

STORE

martedì 21 novembre 2017 16:55



interallacciata $m < \sigma$ $km = \sigma$

m parola alla volta

prefetch

on demand

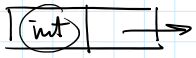
COME si comportano M – Ci nei confronti dei processi.

C1 (piccola!) ⇓ commutazione di contesto ⇒ FLUSH ⇐ ∀ rigo della cache è libero

(in caso di WRITE BACK ⇒ dopo aver scritto nei livelli superiori le linee modificate)

C2 ↑ sono più grandi ($> (\#HS)$) alla commutazione di contesto non si fa nulla ▷ 0

Esercizio
martedì 21 novembre 2017 17:21

programma ①: ocarre una lista N 

e calcola la somma di tutti gli elementi

vs.

programma ②: scorre un vettore N
e calcola la somma di tutti gli elementi

} calcolare T_c

Cache 1° livello set assoc 4 vie $\sigma = 8$ 1k linee
M interallacciata 4 moduli $\tau_M = 40\tau$ $T_{tr} = 4\tau$

①
```
S = 0;
p = testa;
while( p != null ) {
    S = S + p.info;
    p = p.next;
}
```

②
```
S = 0;
for(int i=0; i<N; i++)
    S = S + x[i];
```

WORKING SET
⇓
#fault

WS = { codice: "while"
dati: S, p (reg) }

WS = { codice: "for"
dati: S, i, N
linea con x[i] (PREFETCH) }

NON È WS!

	località	riuso
codice	✓	✓
x	✓	—
i, s	?	✓

↳ riuso x scalari ⇒ Reg!

```
S = ∅;
p = testa;
while ( p != null ) {
    S = S + p.info;
    p = p.next;
}
```

Hp: null ≡ ∅

```
MOV  R0, R3        ; ADD R0, R0, R3
MOV  Rtesta, Rp    ; ADD Rtesta, R0, Rp
while: · IF= Rp, R0, fine
       · LOAD Rp, R0, Rinfo
       · ADD Rinfo, R3, R3   ; S = S + p.info
       · LOAD Rp, #1, Rp     ; p = p.next;
       · GOTO while
fine . END
```

N volte

$$T_{cid} = 2\underbrace{(2\tau + t_a}_{\text{dato+ds}} + \underbrace{\tau)}_{\text{exec ALU}} + N\big(\underbrace{5}_{\text{manca nei conti un ×5 (5 tempi di chiamata sdf ho 5 istr nel corpo del while!)}}(2\tau+t_a) + \underbrace{2\tau}_{IF=} + \underbrace{2(2\tau+t_a)}_{LOAD} + \underbrace{(1\tau)}_{ADD} + \underbrace{(1\tau)}_{GOTO}\big) + \underbrace{(2\tau}_{\substack{IF= \\ \text{delta pass}}} \overbrace{2\tau+t_a}$$

$$= \underbrace{6\tau + 2t_a}_{init} + N\underbrace{(10\tau + 3t_a)}_{\uparrow} + \underbrace{4\tau + t_a}_{finale}$$

t_a: Hp: cache c1 set assoc "Hennessy Patterson" $\Rightarrow 1\tau$

$+1\tau$ rimo

$$t_a = 2\tau$$

$$T_{cid} = (6\tau + 2t_a) + N(10\tau + 3t_a) + (4\tau + t_a)$$
$$(10\tau) + N(16\tau) + (6\tau) = 16\tau + N(16\tau)$$
$$= (N+1)(16\tau)$$

#fault

$\sigma = 8$

1 fault × il codice

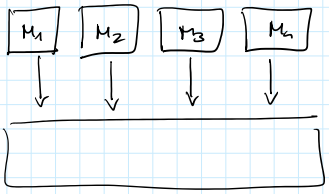
fault × dati ?

N fault × dati

$\Big\}$ #fault = $(N+1)$

Cache 1° livello set assoc 4 vie $\sigma = 8$ 1k word

M interallacciata 4 moduli $\tau_M = 4\tau$ $T_{tr} = 4\tau$



$$2(\underbrace{2\tau}_{4\tau} + T_{tr}) + \frac{\sigma}{m}\tau_M + \tau$$

$$10\tau + \underbrace{8\tau}_{\text{e } 4\tau} + \tau$$

$T_{trasf} = 91\tau$

$$T_c = T_{cid} + \#fault \times T_{trasf} = (N+1)\,16\tau + (N+1)\,91\tau$$
$$(N+1)16\tau + (N+1) * 91\tau$$

martedì 21 novembre 2017 18:01

for( ) { s = s + x[i]; }

WS { Codice: codice for
dati: linea convertita di x (x[i])

#fault ?

codice → 1 (2)

dati → N / σ

LOAD $R_{base\,x}$, R_i, R_{temp}, "PREFETCH"
ADD R_s, R_{temp}, R_s

#Fault = 1 (dati)?

~~dati~~ #fault = 2