34 Lezione

giovedì 23 maggio 2019 16:17

Esempio di utilizzo di Hadoop

Sequenceiq / hadoop docker

docker pull ↗

VECTORIZATION

In general 4/5 principles to respect: 16:53

→ Countable

→ the block of code that we use for vectorization should have a unique entry and exit point, (less than do not diverge)

→ Not diverging code

→ Innermost loop is the one that is vectorized

→ You don't have function calls.

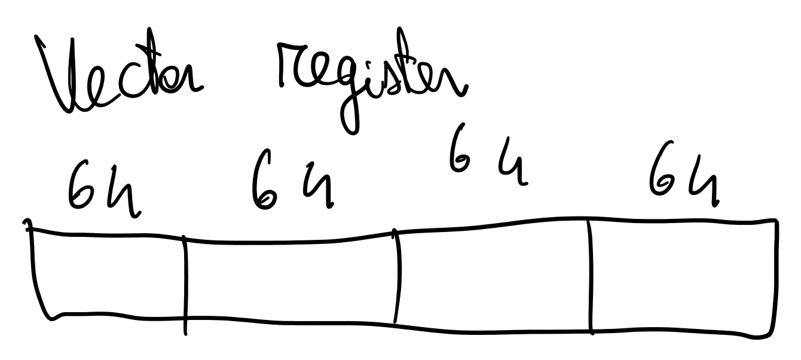
→ No dependencies in the different parallel iteration.

What we should look at?

→ alignment: memory accesses and in particular the shape of memory access is more than redundant.

for $(\forall i)$

$$X[i] = a[i] + b[i]$$



What I try to do is to compile the code in a kind of load vector.

we can have a load vector in Rx_1
another load vector in Rx_2
and an add vector
$Rx_1 + Rx_2 = Rx_3$
then the store vect
$x[i] = Rx_3$

Another big problem with data accesses

for $(\forall i)$

$$a[i] = b[c[i]]$$

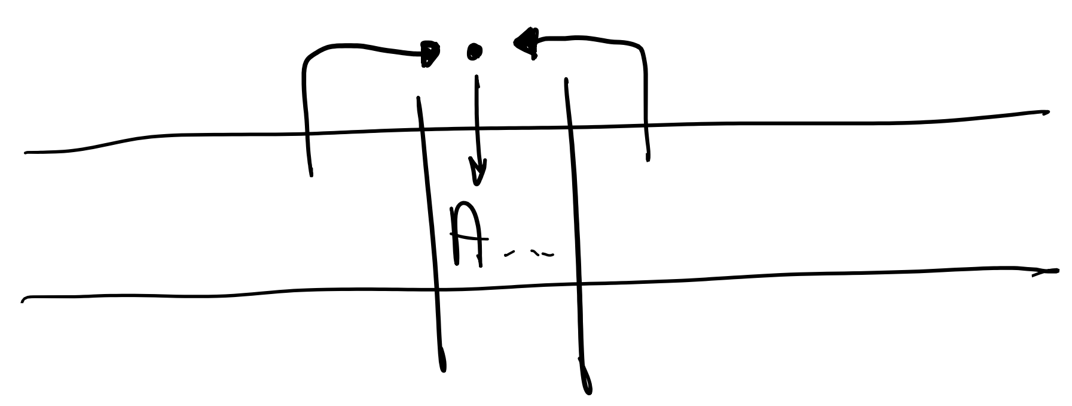
No continuous memory access means be careful of str... and ... 17:18

Data dependencies:

RAW Read after write

WAR write after read

WAW



17:20

If I write the different vector this doesn't .. any dependency

for (int $i=1$; $i<n-1$; i++)

$$\cancel{X}[i] = \frac{(x[i] + x[i-1] + x[i+1])}{3}$$

y

Problem of Alignment/aliasing

Suppose that we want to write the procedure: mcpy (T*, T*, Size)

for ($i=0$; $i<$ size; i++)

$*(p1)++ = *(p2)++$

We can use the __restrict__ keyword to say that this pointer is different from the others. I do:

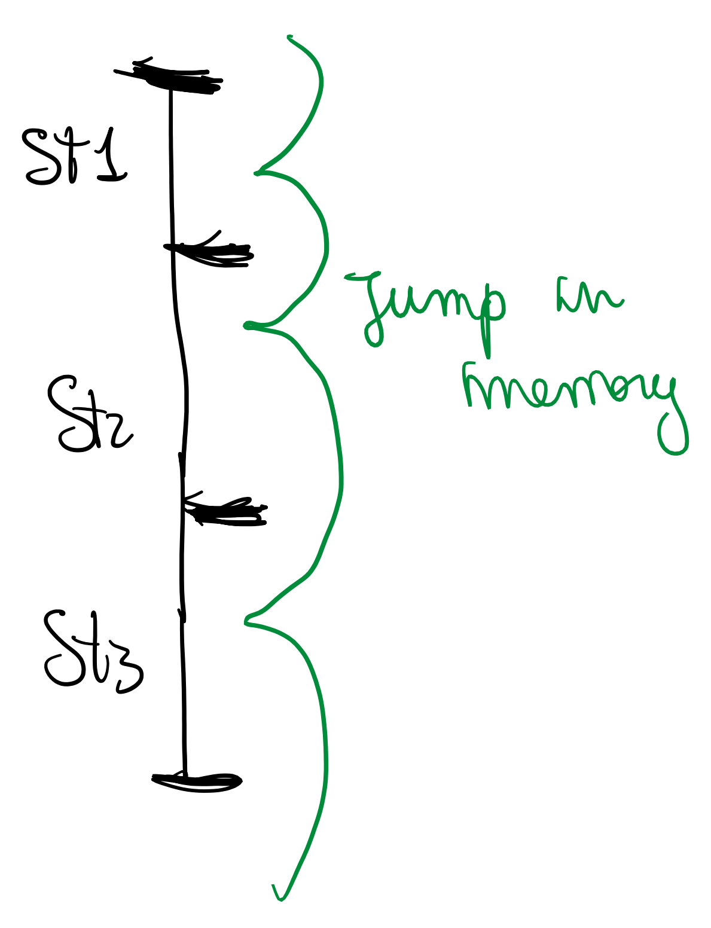
mcpy (T * p1 __restrict__, T* p2 __restrict__ ...)

```
#pragma GCC ivdep
    for ( ... )
```

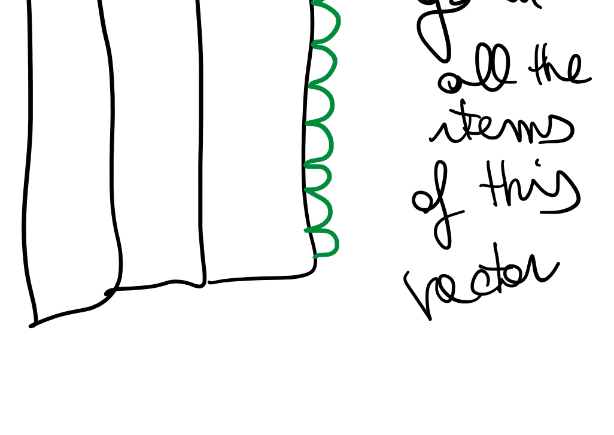
Another thing common to be discussed:

vect of struct vs struct of vect

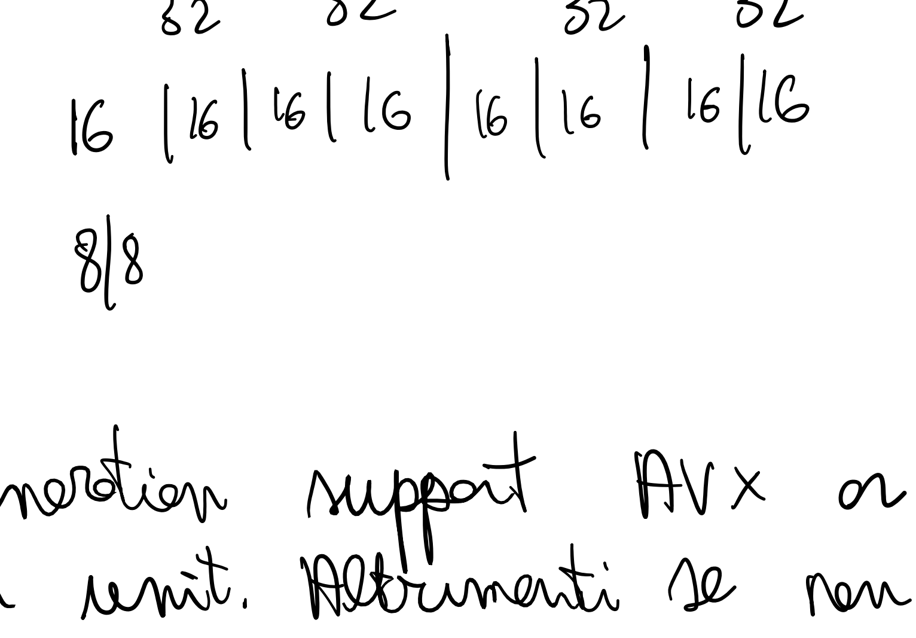
If you have to do some data parallel computation: you have in memory:



If I have a vector for the ID and a vector for name



Data sizes are important: 17:32



Latest generation support AVX or AVX2 version unit. Altrimenti se non metto il flag -mavx2 va ... 17:36

-O3 in gcc includes the vectorization

-ftree-vectorized

you can use the -fopt-info-vec-all to have during the compilation phase the information about who successfully vectorized and who received an error.