smvp.pas

February 21, 2008

Contents

1 smvp **1**

2 mul4 **2**

3 init **2**

1 smvp

```
$r-
program smvp ;
{Usage of the program
smvp digit
where digit is 1, 2 , 3 or 4
it will run the matrix vector product using one of 4 algorithms
1 is the built in dot product operator
}
const
    cols = 4096;
    rows = 8192;
    alpha =1;
    beta =2;
    runs =30;
type
    t = real ;
    tv = array [1..cols ] of t ;
    tv2 = array [1..rows ] of t ;
    tm = array [1..rows ,1..cols ] of t ;

var
    Let A ∈ tm;
    Let B ∈ tv;
    Let C ∈ tv2;
    Let start, finish, delay ∈ double;
    Let i ∈ integer;

    Let s ∈ string;
```

procedure *mul4* (**var** A :*tm* ; **var** X :*tv* ;**var** Y :*tv2*); (see Section 2)

procedure *init* ; (see Section 3)
begin
init;

else $s \leftarrow$ *paramstr* (1);
writeln('method ' , s);
start$\leftarrow$ *secs*;
for $i \leftarrow 1$ **to** *runs* **do**
case **ord**(s_1) - **ord**('0') **of**
1:$C \leftarrow (A.B) \times \alpha + \beta \times C$;
2:$C \leftarrow (\sum A_{\iota_0} \times B) \times \alpha + \beta \times C$;
3:$C \leftarrow A_{\iota_0}.B \times \alpha + \beta \times C$;
4:*mul4* (A, B, c);
end ;
finish$\leftarrow$ *secs*;
delay$\leftarrow$ *finish* - *start*;
writeln('matrix size' , *rows* $\times$ *cols* : 8, ' average of ' , *runs* : 4, ' runs ' , $\frac{1000 \times (delay)}{runs}$, 'ms ' , $\frac{(2 \times rows \times cols) \times (runs/delay)}{1e6}$, ' mflops');
end .

2 mul4

procedure *mul4* (**var** A :*tm* ; **var** X :*tv* ;**var** Y :*tv2*);
var
Let *tmp* $\in$ t;
Let i, $j \in$ integer;
begin
for $i \leftarrow 1$ **to** *rows* **do**
begin
tmp$\leftarrow$ 0;
for $j \leftarrow 1$ **to** *cols* **do**
tmp$\leftarrow$ *tmp* + $A_{i,j} \times X_j$;
$Y_i \leftarrow$ *tmp* $\times (\alpha) + (Y_i \times \beta)$;
end ;
end ;

3 init

procedure *init* ;
begin

$A \leftarrow (\iota_0 \times \iota_1) \wedge 13$;

$B \leftarrow 1$;
end ;