CS-406 HW-3
Page-1

Görkem YAR
27970

Q1:

Graph (a)

1) Maximum degree of concurrency : 8
2) Critical path length : 4 nodes 3 length
3) Maximum Speedup: $\frac{15}{4} = 3.75$
4) Minimum number of processors for max speed up = 8
5) Maximum Speed up with 4 threads = $\frac{15}{5} = 3$

Graph (b)

1) Maximum degree of concurrency = 8
2) Critical Path length = 4 nodes 3 length
3) Maximum speed up = $\frac{15}{4}$ = 3.75
4) Minimum number of processor for max speed up = 8.
5) Maximum speed up with 4-threads = $\frac{15}{5} = 3$

Graph (c)

1) Maximum degree of concurrency: 8
2) Critical Path Length = 7 nodes 6 length
3) Maximum Speed-up = $\frac{14}{7} = 2$
4) Minimum number of processors for max speed up : 3
5) Maximum speed up with 4 threads = $\frac{14}{7} = 2$

Graph (d)

1) Maximum degree of concurrency: 8
2) Critical path length = 8 nodes 7 length
3) Maximum Speed up = $\frac{15}{8} = 1.75$
4) Miniman Number of processor for max speed up = 2
5) Maximum Speed up with 4 threads = $\frac{15}{8} = 1.75$

Q2:

revised code

```
int temp;
for (j=1; j<n; j++){
    temp= c[j]
    for(i=1; i<n; i++){
        c[j][i] = c[j-1][i-1] + temp
```

* improvement-1: change loop order to improve spatial locality. In the previous version the code access the matrix column by column. Now the access is done by row by row. As a result, the cache miss rate will reduce. (Cache Locality)

* improvement-2: Using a temp variable to store c[j]. Instead of accessing it from memory multiple times, we put it in a register. This reduced memory accesses. (Locality in registers) (Tempo ral Locality)

Gürkem UAR
27970

Q3: Using Amdahl's law:

Serial fraction $1-r$; paralel fraction r; T_s represents serial time
$T_P(P) = (1-r) \cdot T_s + \frac{r \cdot T_s}{P}$; T_P represents parallel time

Amdahl's law for speedup:

$$\leq T_s / T_P(P)$$

$$= \frac{T_s}{(1-r) \cdot T_s + \frac{r \cdot T_s}{P}} = \frac{1}{(1-r) + \frac{r}{P}}$$

if we make $\frac{r}{P}$ to small compared to $1-r$ then it can be negligible.

$$\lim_{P \to \infty} \frac{1}{(1-r) + \frac{r}{P}} = \frac{1}{1-r} \;//$$

speedup maximum

Q4:

a) Latency: The time it takes for the data to travel from memory to it's destination. It is the measurement of delay and unit of latency is seconds.

Bandwidth: The maximum rate (capacity) for the data to transfer from memory. It can be measured as bit/seconds or byt/seconds.

* Low latency means faster response, high bandwidth means more data can be transmitted at once.

b) Spatial Locality: It means use of data elements within relatively close in memory (Location). If a particular memory accessed nearby locations will be accessed as well.

Temporal Locality: Refers to the reuse of specific data within a small time interval.

Görkem YAR
27970 ~~Gorkem~~

Q5:

10,6 Tera flops = $10{,}6 \cdot 10^{12}$ floating point operations per second

720 GBytes = $\frac{720}{4} \cdot 10^9 = 180 \cdot 10^9$ floats per second from global memory

$\to$ sizeof(float)

$$\frac{10{,}6 \cdot 10^{12}}{180 \cdot 10^9} = 58{,}89$$ floating point operations per float.

on average 59 operations.

Q6:

a) 50.000 / 2048 = 24,414 $\to \boxed{25}$ blocks each block has 1024 threads

b) Each warp in CUDA is 32 threads each block has 1024 threads

so each block has 1024/32 = 32 warps

total warps = 25 . 32 = 800 WARPS

c) 1024 * 25 = 25600 threads in total.

d) Let's consider the following threads with block numbers:

blockIdx.x = 24

blockDim.x = 1024

Id of the last blocks first thread $\to$ 49.152

50000 - 49152 = 848 is the threadIdx of the first thread that does not do any job.

block = 25 (last block) warp = 848/32 = 26 is the warp number

e) The control diverges for the $\underbrace{\text{threads}}_{\text{warps that are 26 or higher}}$ that higher idx or warp number in line 5 and 7.

Also, the first $\underbrace{\text{848 threads}}_{\text{first 26 warps less than 26}}$ of the last block diverge in line 7 as well.

Görkem YAR
27170 ~~[illegible]~~

Q7:

a) Using Amdahl's law: The unparalelizeable part of a program determines the maximum speed up.

Serial	Fraction	40%
parallel	fraction	60%

$$T_P(P) = \%40 \cdot T_S + \frac{\%60 \cdot T_S}{P}$$

$$\text{Speed up} = \frac{T_S}{\frac{T_S \cdot 2}{5} + \frac{3\, T_S}{5 \cdot P}} = \frac{1}{\frac{2}{5} + \frac{3}{5 \cdot P}}$$

if $\lim_{P \to \infty} = \frac{5}{2} = 2{,}5$ times speed up

to much cores, parallel work

b) theoretically maximum speed up with p processor is $\underline{p}$. However, in practice the maximum speed up can exceed p (we call this superlinearity).
examples
- N queens problem: parallel version can have less work since the solution might appear in one parallel block much quicker than doing all serial work.
- Also, the parallel executions can improve cache-hit ratios this is called resource-based superlinearity.

c)
$$E = \frac{S}{P} = \frac{T_S}{P \cdot T_P} = \frac{n}{P \cdot (n/p + \log p)} = \frac{n}{n + p\log p}$$

d) $E = \Theta(1)$ to be cost optimal. The cost optimality can also be written in the form $P \cdot T_P = \Theta(T_S)$.

$$P(n/p + \log p) = \Theta(n)$$
$$= n + p\log p = \Theta(n)$$
$$\Rightarrow p\log p = \Theta(n)$$
$$\Rightarrow p\log p \lesssim kn$$

$\frac{n}{\log n}(\log n - \log\log n) \rightarrow$ $\boxed{P = \frac{n}{\log n}}$ by substitution

$n - \frac{\log\log n}{\log n} \le k.n$ holds

to small