CSC346/646 Parallel Computation Spring 2015

Analysis of Parallel Floyd-Warshall's Algorithm

Pseudocode for parallel Floyd-Warshall's algorithm is given below. Figure 1 illustrates the data communication pattern.

Input: A weighted graph $G = (V, E)$ with weight function $f : V \times V \to \mathcal{R}^+ \cup \{+\infty\}$.
Let n denote the number of vertices in V. Number the vertices $V = \{v_1, v_2, v_3, ..., v_n\}$.

Output: An $n \times n$ matrix C such that $C_{i,j}$ is the cost of the shortest path from v_i to v_j.

Key Idea: Define a matrix $C_{i,j}^{(k)}$ as the cost of the shortest (restricted) path from v_i to v_j that goes through intermediate vertices numbered no higher than k.

Method:

Check that the number of processes p is a perfect square.
Process rank 0 reads the $n \times n$ input matrix ; check that n is divisible by $\sqrt{p}$.
Block-distribute each $\frac{n}{\sqrt{p}} \times \frac{n}{\sqrt{p}}$ part of the input matrix

0. Initialize the $\frac{n}{\sqrt{p}} \times \frac{n}{\sqrt{p}}$ block portion of the cost matrix $C^{(0)}$

```
// Let P_{i,j} denote the processor in block-row i and block column j

for k = 1 to n do
{
1.      each P_{i,j} that has a segment of the k-th row of C^(k-1) broadcasts it to P_{*,j}
2.      each P_{i,j} that has a segment of the k-th column of C^(k-1) broadcasts it to P_{i,*}
3.      each process P_{i,j} computes its n/sqrt(p) x n/sqrt(p) part of C^(k)
}
```

4. Process rank 0 collects the $\frac{n}{\sqrt{p}} \times \frac{n}{\sqrt{p}}$ parts of $C^{(n)}$ and writes $C^{(n)}$ to a file.

Parallel Time:

We begin our analysis assuming the cost matrix $C^{(0)}$ is block-distributed and properly initialized using the values of the weight function f, i.e., we omit the time taken by step 0. Also, we omit the time taken by step 4 in our analysis here. Intuitively, we can think of the parallel time as the sum of the computation time and the communication time. Let n denote the size of the problem (i.e. $n = |V|$) and let p denote the number of processors. We have:

$$T_p(n,p) \;=\; \text{parallel computation time} \;+\; \text{communication time}$$

We derive the parallel computation time by counting the number of update operations. Observe that the updates are done concurrently for each block size $\frac{n}{\sqrt{p}} \times \frac{n}{\sqrt{p}}$. The factors of our expression for parallel computation time are illustrated in the following diagram:

$$n \left(\frac{n}{\sqrt{p}}\right)^2 \;=\; \frac{n^3}{p}$$

n iterations

There are $\frac{n}{\sqrt{p}} \times \frac{n}{\sqrt{p}}$ entries in each block

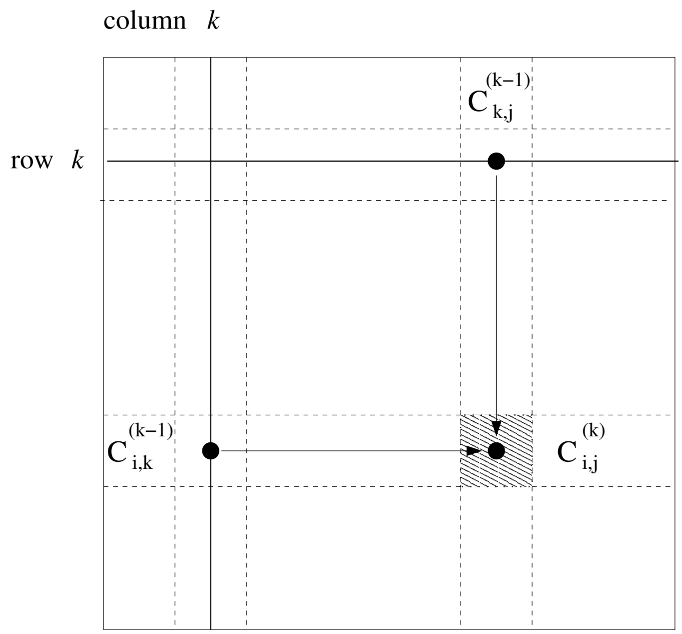

Figure 1: Data dependency for update step: $C^{(k)}_{i,j} \;=\; \min\left(C^{(k-1)}_{i,j}\;,\; C^{(k-1)}_{i,k} + C^{(k-1)}_{k,j} \right)$

Recall our cost model for broadcasting a message size m among q processors (in a hypercube) is given by:

$$T_{\text{Bcast}}(q) \;=\; (t_c \,+\, t_w m)\log_2(q)$$

For the communication time in parallel Floyd-Warshall, we consider the two broadcast operations, Refer to Figure 1 and recall that each block- row and each block-column contains $\sqrt{p}$ blocks, therefore the number of processors participating in each row-broadcast and each column broadcast is $\sqrt{p}$. In each broadcast, a row-segment (or column-segment) is sent. There are $\frac{n}{\sqrt{p}}$ data items in each segment. Distributing $\frac{n}{\sqrt{p}}$ data items in one broadcast is more efficient than $\frac{n}{\sqrt{p}}$ broadcast operations, each sending one data item. Using these observations, we have the following expression for communication time:

$$n\;\left[\;2\left(t_c \,+\, t_w \tfrac{n}{\sqrt{p}}\right)\log_2(\sqrt{p})\;\right]$$

n iterations

Two broadcasts

There are $\frac{n}{\sqrt{p}}$ data items broadcast among $\sqrt{p}$ processors

Adding the computation and communication time, we have:

$$T_{\text{par}}(n,p) \;=\; \frac{n^3}{p} \;\;+\;\; n\;\left[\;2\;\left(t_c \,+\, t_w\frac{n}{\sqrt{p}}\right)\log_2(\sqrt{p})\;\right] \tag{1}$$

Recall for logarithms of any base b we know:,

$$\log_b(\sqrt{x}) = \frac{1}{2}\log_b(p)$$

This property allows us to simplify equation (1) and we have:

$$T_{\text{par}}(n,p) \;=\; \frac{n^3}{p} \;+\; n\left(t_c + t_w\frac{n}{\sqrt{p}}\right)\log_2(p) \tag{2}$$

Parallel Cost:

$$\begin{aligned} pT_{\text{par}}(n,p) &= n^3 \;+\; np\left(t_c + t_w\tfrac{n}{\sqrt{p}}\right)\log_2(p) \\ &= n^3 \;+\; t_c np\log_2(p) \;+\; t_w n^2\sqrt{p}\log_2(p) \end{aligned} \tag{3}$$

Sequential Time:

Counting update operations, we see there are n^3 such operations. Therefore, we can write our sequential time as:

$$T_s \;=\; n^3$$

Parallel Overhead:

$$\begin{aligned} T_o(n,p) &= pT_{\text{par}}(n,p) - T_s(n) \\ &= n^3 \;+\; t_c np\log_2(p) \;+\; t_w n^2\sqrt{p}\log_2(p) \;-\; n^3 \\ &= t_c np\log_2(p) \;+\; t_w n^2\sqrt{p}\log_2(p) \end{aligned} \tag{4}$$

Iso-efficiency Equation: $T_s(n) \;=\; K\,T_o(n,p)$

$$n^3 \;=\; K\left(\, t_c np\log_2(p) \;+\; t_w n^2\sqrt{p}\log_2(p) \,\right) \tag{5}$$

When there is more than one term in our expression for the overhead, T_o, we can consider these terms one at a time. To understand the reasoning behind treating the terms of T_o individually, we recall our simplified expression for efficiency:

$$E \;=\; \frac{1}{1+\frac{T_o(n,p)}{T_s(n)}} \tag{6}$$

Referring to equation (6), we observe that if the overhead strictly dominates[1] the sequential time, then the ratio $\frac{T_o(n,p)}{T_s(n)}$ diverges to infinity, and the efficiency E converges to zero.

If the overhead $T_o(n,p)$ consists of a sum of several terms, the ratio $\frac{T_o(n,p)}{T_s(n)}$ will diverge to infinity if any one of those terms strictly dominates $T_s(n)$. Therefore, it suffices to consider the asymptotically dominant term of $T_o(n,p)$. We re-consider equation (5). Under the very reasonable assumption that $n >> \sqrt{p}$, the dominant term of $T_o(n,p)$ is:

$$t_w n^2\sqrt{p}\log_2(p)$$

We can then consider a modified iso-efficiency equation using only the dominant term. We have:

$$n^3 \;=\; K\left(t_w n^2\sqrt{p}\log_2(p)\,\right) \tag{7}$$

Cancelling a factor of n^2 from both sides of equation (7) and combining constants we find:

$$n \;=\; \hat{K}\,\sqrt{p}\log_2(p) \tag{8}$$

While equation (8) can not be simply re-written to express p as a function of problem size n, it does provide an implicit relationship between n and p. Equation (8) implicitly describes how much p should increase when the problem size n is increased, and still maintain a level of efficiency which does not diminish to zero.

[1] In the sense of little "o".