Project Discussions

Dijkstra's Algorithm: Parallel Formulation

- Two parallelization strategies
  - execute each of the *n* shortest path problems on a different processor (source partitioned),
  - use a parallel formulation of the shortest path problem to increase parallelism (source parallel).

Dijkstra's Algorithm: Source Partitioned Formulation

- Use n processors, each processor P_i finds the shortest paths from vertex v_i to all other vertices by executing Dijkstra's sequential single-source shortest paths algorithm.
- It requires no interprocess communication (provided that the adjacency matrix is replicated at all processes).
- The parallel run time of this formulation is: $\Theta(n^2)$.
- It can only use n processors

Recall: Floyd Warshall Algorithm

- All Pair Shortest Path algorithm
  - Defines the shortest distance d in terms of “smaller” problems?
- Note: a path exists between two vertices v_i, v_j , if
  - there is an edge from v_i to v_j; or
  - there is a path from v_i to v_j going through intermediate vertices from set $\{v_1\}$; or
  - there is a path from v_i to v_j going through intermediate vertices drawn from set $\{v_1, v_2\}$; or
  - …
  - there is a path from v_i to v_j going through intermediate vertices drawn from set $\{v_1, v_2, \dots v_k\}$;

Floyd's Algorithm

From our observations, the following recurrence relation follows:

$$d_{i,j}^{(k)} = \begin{cases} w(v_i, v_j) & \text{if } k = 0 \\ \min\left\{d_{i,j}^{(k-1)}, d_{i,k}^{(k-1)} + d_{k,j}^{(k-1)}\right\} & \text{if } k \geq 1 \end{cases}$$

This equation must be computed for each pair of nodes and for *k = 1, n*. The serial complexity is $O(n^3)$.

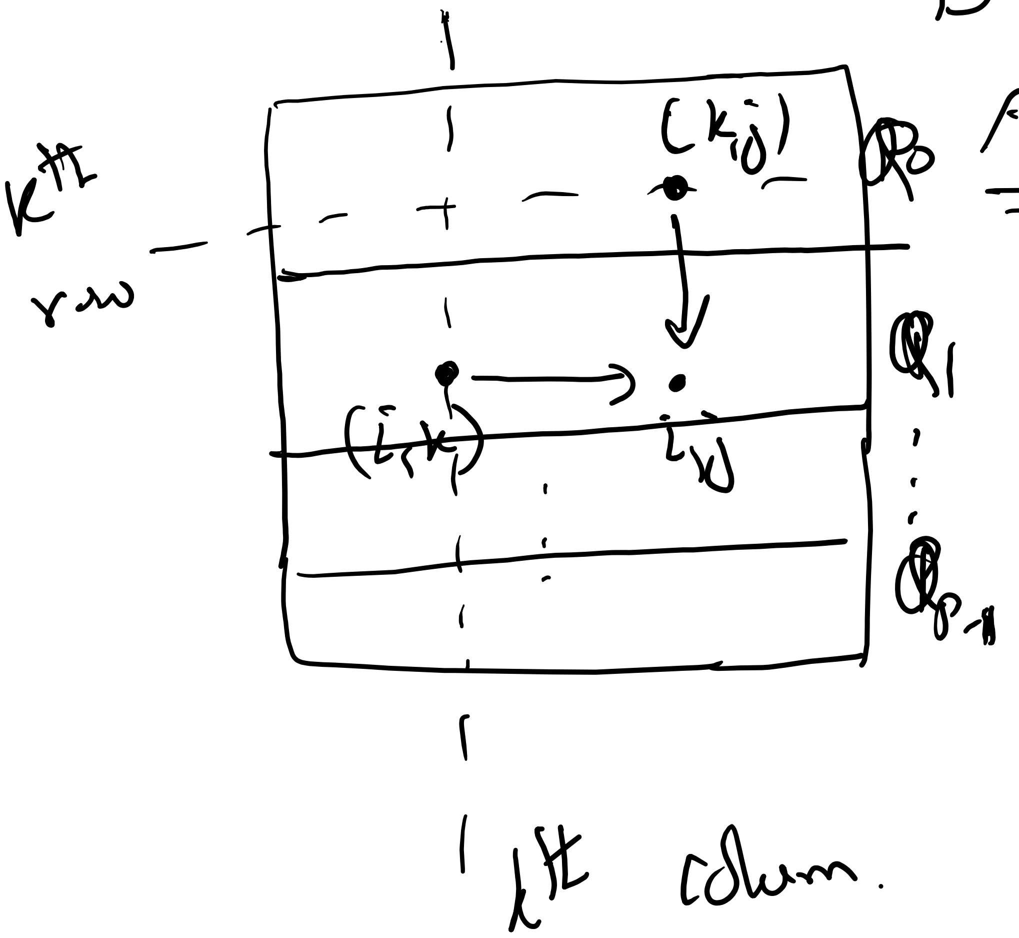
Floyd's Algorithm

procedure FLOYD_ALL_PAIRS_SP(A)
begin
    $D^{(0)} = A$;
    for $k := 1$ **to** n **do**
        for $i := 1$ **to** n **do**
            for $j := 1$ **to** n **do**
                $d_{i,j}^{(k)} := \min\left(d_{i,j}^{(k-1)}, d_{i,k}^{(k-1)} + d_{k,j}^{(k-1)}\right);$
end FLOYD_ALL_PAIRS_SP

Broadcast the k^{th} row to all processors.

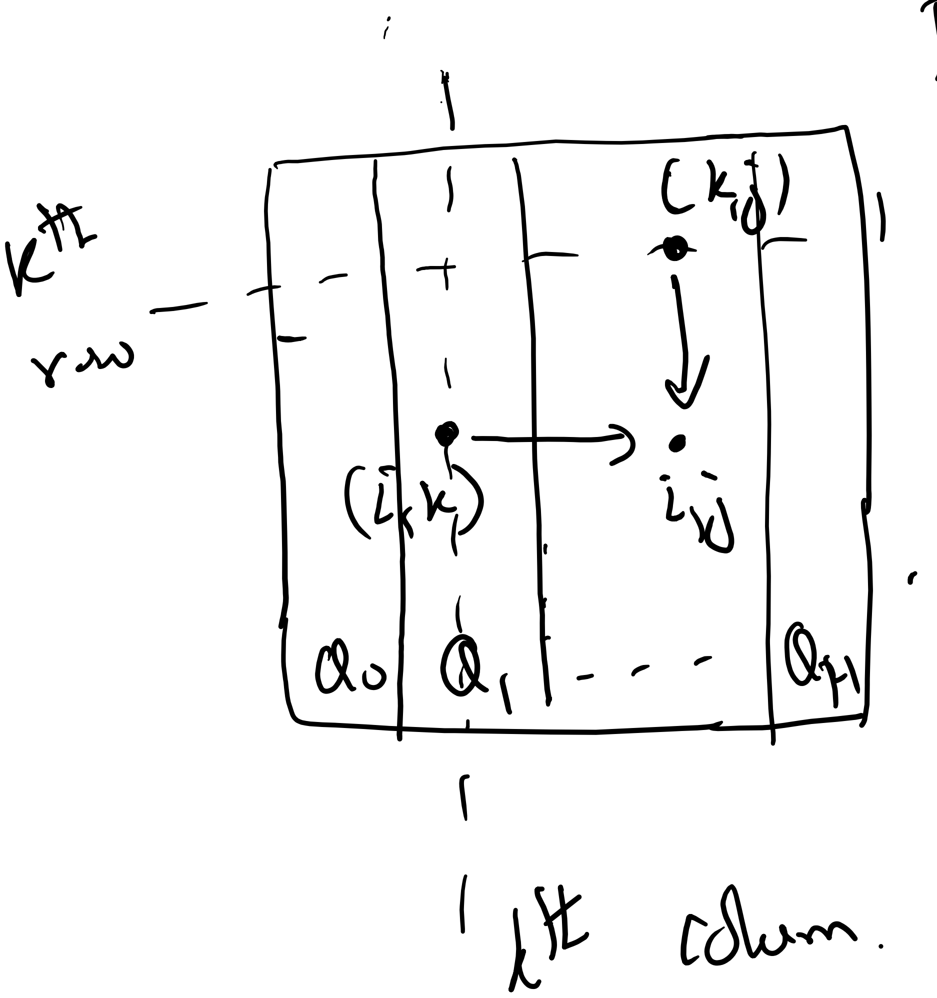
Floyd's all-pairs shortest paths algorithm. This program computes the all-pairs shortest paths of the graph *G = (V,E)* with adjacency matrix *A*.

$D(n \times n)$



At the k^{th} step:

$$\min(d(i,j),\ d(i,k)+d(k,j))$$

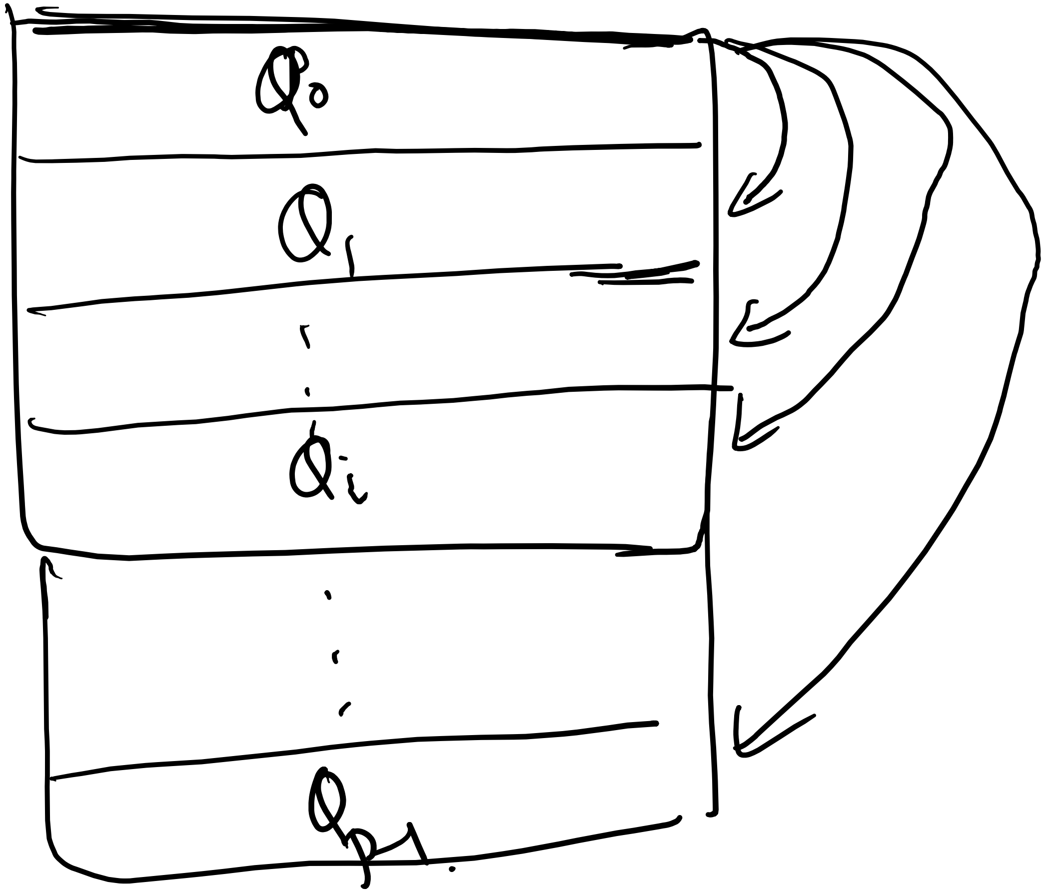


$D(n \times n)$

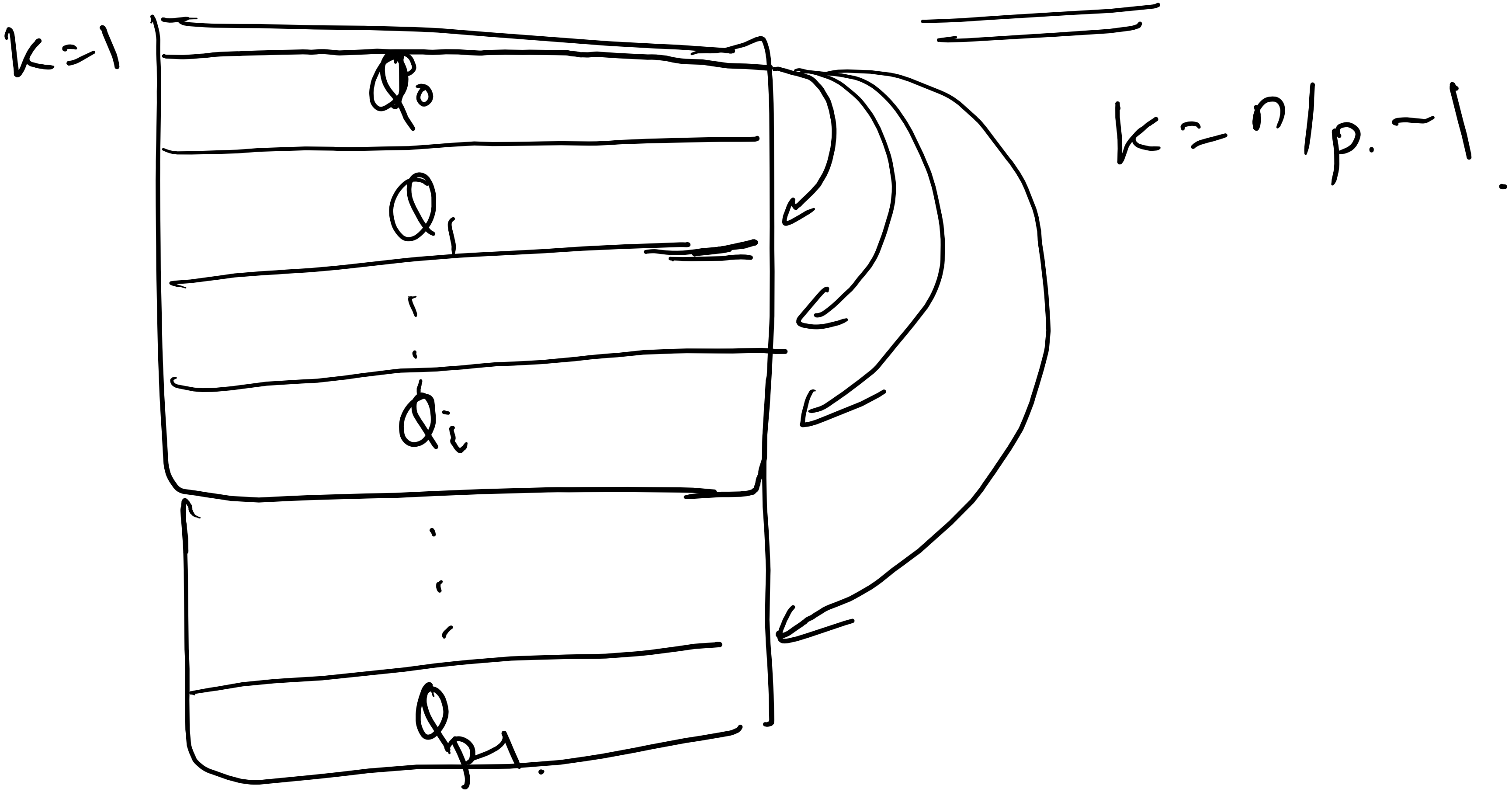
At the k^{th} step:

$$\min\big(d(i,j),\ d(i,k)+d(k,j)\big)$$

$k=0.$



$k = 1.$

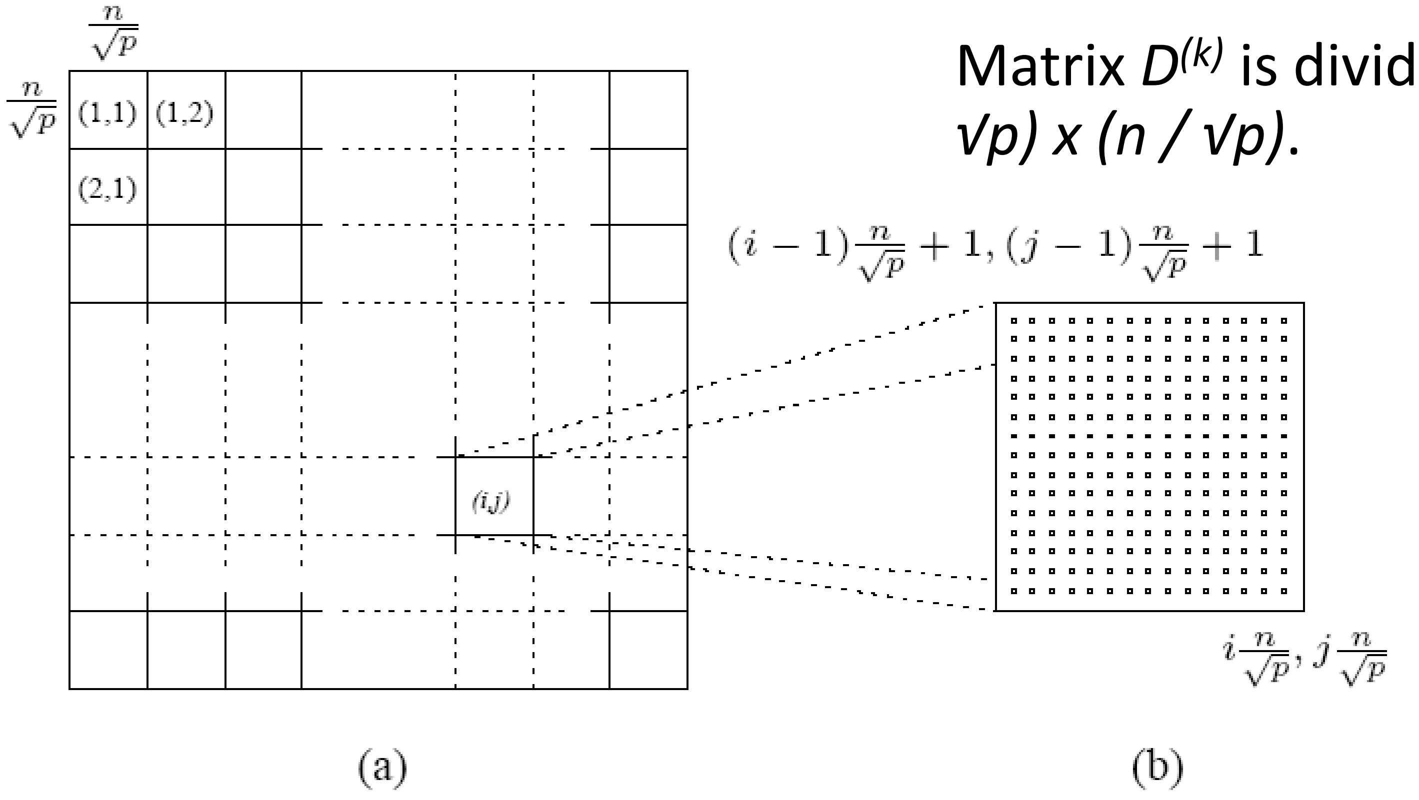


$D(n \times n).$

Floyd's Algorithm: Parallel Formulation Using 2-D Block Mapping

- Each processor updates its part of the matrix during each iteration.
- To compute $d_{l,k}^{(k-1)}$ processor $P_{i,j}$ must get $d_{l,k}^{(k-1)}$ and $d_{k,r}^{(k-1)}$.
- In general, during the k^{th} iteration, each of the $\sqrt{p}$ processes containing part of the k^{th} row send it to the $\sqrt{p}$ - 1 processes in the same column.
- Similarly, each of the $\sqrt{p}$ processes containing part of the k^{th} column sends it to the $\sqrt{p} - 1$ processes in the same row.

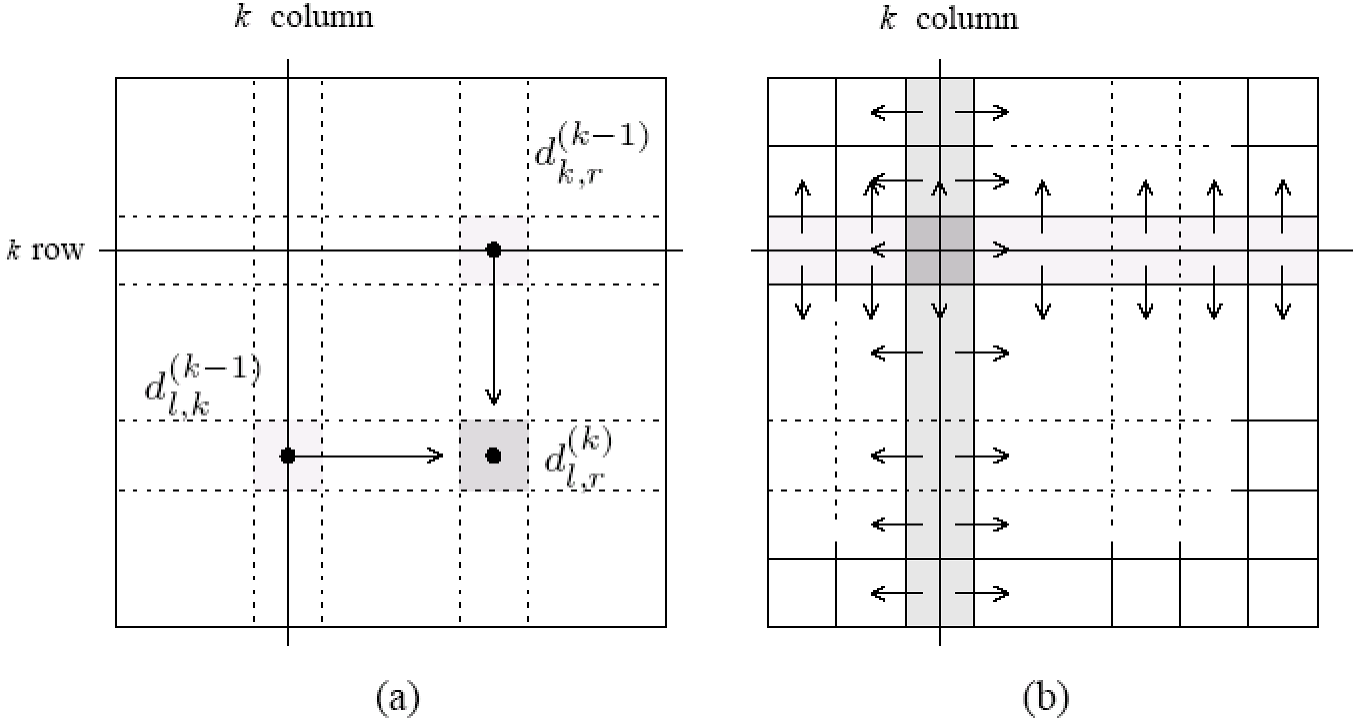
Floyd's Algorithm: Parallel Formulation Using 2-D Block Mapping



Matrix $D^{(k)}$ is divided into p blocks of size $(n / \sqrt{p}) \times (n / \sqrt{p})$.

(a) Matrix $D^{(k)}$ distributed by 2-D block mapping into $\sqrt{p} \times \sqrt{p}$ subblocks, and (b) the subblock of $D^{(k)}$ assigned to process $P_{i,j}$.

Floyd's Algorithm: Parallel Formulation Using 2-D Block Mapping



(a) Communication patterns used in the 2-D block mapping. When computing $d_{i,j}^{(k)}$, information must be sent to the highlighted process from two other processes along the same row and column. (b) The row and column of $\sqrt{p}$ processes that contain the k^{th} row and column send them along process columns and rows.

Floyd's Algorithm: Parallel Formulation Using 2-D Block Mapping

procedure FLOYD_2DBLOCK($D^{(0)}$)
begin
    for $k := 1$ **to** n **do**
    begin
        each process $P_{i,j}$ that has a segment of the k^{th} row of $D^{(k-1)}$;
            broadcasts it to the $P_{*,j}$ processes;
        each process $P_{i,j}$ that has a segment of the k^{th} column of $D^{(k-1)}$;
            broadcasts it to the $P_{i,*}$ processes;
        each process waits to receive the needed segments;
        each process $P_{i,j}$ computes its part of the $D^{(k)}$ matrix;
    end
end FLOYD_2DBLOCK

Floyd's parallel formulation using the 2-D block mapping. $P_{*,j}$ denotes all the processes in the j^{th} column, and $P_{i,*}$ denotes all the processes in the i^{th} row. The matrix $D^{(0)}$ is the adjacency matrix.

Floyd's Algorithm: Parallel Formulation Using 2-D Block Mapping

- During each iteration of the algorithm, the k^{th} row and k^{th} column of processors perform a one-to-all broadcast along their rows/columns.
- The size of this broadcast is $n/\sqrt{p}$ elements, taking $\Theta((n \log p)/\sqrt{p})$ steps
- The synchronization step takes $\Theta(\log p)$.
- The computation time is $\Theta(n^2/p)$.

Dijkstra's Algorithm: Source Parallel Formulation

- In this case, each of the shortest path problems is further executed in parallel. We can therefore use up to n^2 processors.
- Given p processors ($p > n$), each single source shortest path problem is executed by p/n processors.
- Using previous results, this takes time:

$$T_P = \overbrace{\Theta\left(\frac{n^3}{p}\right)}^{\text{computation}} + \overbrace{\Theta(n \log p)}^{\text{communication}}.$$

- For cost optimality, we have $p = O(n^2/\log n)$ and the isoefficiency is $\Theta((p \log p)^{1.5})$.