
```
1. procedure PRIM_MST( $V, E, w, r$ )
2. begin
3.    $V_T := \{r\}$ ;
4.    $d[r] := 0$ ;
5.   for all  $v \in (V - V_T)$  do
6.     if edge  $(r, v)$  exists set  $d[v] := w(r, v)$ ;
7.     else set  $d[v] := \infty$ ;
8.   while  $V_T \neq V$  do
9.     begin
10.      find a vertex  $u$  such that  $d[u] := \min\{d[v] | v \in (V - V_T)\}$ ;
11.       $V_T := V_T \cup \{u\}$ ;
12.      for all  $v \in (V - V_T)$  do
13.         $d[v] := \min\{d[v], w(u, v)\}$ ;
14.      endwhile
15.   end PRIM_MST
```

Algorithm 10.1 Prim's sequential minimum spanning tree algorithm.

```
1. procedure DIJKSTRA_SINGLE_SOURCE_SP( $V, E, w, s$ )
2. begin
3.    $V_T := \{s\}$ ;
4.   for all  $v \in (V - V_T)$  do
5.     if  $(s, v)$  exists set  $l[v] := w(s, v)$ ;
6.     else set  $l[v] := \infty$ ;
7.   while  $V_T \neq V$  do
8.     begin
9.       find a vertex  $u$  such that  $l[u] := \min\{l[v] | v \in (V - V_T)\}$ ;
10.       $V_T := V_T \cup \{u\}$ ;
11.      for all  $v \in (V - V_T)$  do
12.         $l[v] := \min\{l[v], l[u] + w(u, v)\}$ ;
13.      endwhile
14. end DIJKSTRA_SINGLE_SOURCE_SP
```

Algorithm 10.2 Dijkstra's sequential single-source shortest paths algorithm.

```
1. procedure FLOYD_ALL_PAIRS_SP(A)
2. begin
3.    $D^{(0)} = A$ ;
4.   for  $k := 1$  to  $n$  do
5.     for  $i := 1$  to  $n$  do
6.       for  $j := 1$  to  $n$  do
7.          $d_{i,j}^{(k)} := \min(d_{i,j}^{(k-1)}, d_{i,k}^{(k-1)} + d_{k,j}^{(k-1)})$ ;
8.   end FLOYD_ALL_PAIRS_SP
```

Algorithm 10.3 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 .

1. **procedure** FLOYD_2DBLOCK($D^{(0)}$)
2. **begin**
3. **for** $k := 1$ **to** n **do**
4. **begin**
5. 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;
6. 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;
7. each process waits to receive the needed segments;
8. each process $P_{i,j}$ computes its part of the $D^{(k)}$ matrix;
9. **end**
10. **end** FLOYD_2DBLOCK

Algorithm 10.4 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.

```
1. procedure JOHNSON_SINGLE_SOURCE_SP( $V, E, s$ )
2. begin
3.    $Q := V$ ;
4.   for all  $v \in Q$  do
5.      $l[v] := \infty$ ;
6.    $l[s] := 0$ ;
7.   while  $Q \neq \emptyset$  do
8.     begin
9.        $u := \text{extract\_min}(Q)$ ;
10.      for each  $v \in \text{Adj}[u]$  do
11.        if  $v \in Q$  and  $l[u] + w(u, v) < l[v]$  then
12.           $l[v] := l[u] + w(u, v)$ ;
13.        endwhile
14.   end JOHNSON_SINGLE_SOURCE_SP
```

Algorithm 10.5 Johnson's sequential single-source shortest paths algorithm.