3 [1] Write a parallel program using parbegin/pend for the precedence graph given below.

```
parbegin
begin
    S2
    S5 end
    S3 begin
    S4 parbegin
    S6
    S7 parpend
    S8 end
parend
S9
```
3. Suppose the time-complexity of a parallel algorithm on a weighted graph is \(2n^2 n/p + 3pn\), where \(n\) is the number of nodes of the graph and \(p\) is the number of processors. What would be the optimum number of processors to use when \(n\) is 600?

\[
det \ y = \frac{2n^2}{p} + 3pn \quad -(i)
\]

Plugging in the values of \(n = 600\),

\[
y = \frac{2 \times (600)^2}{p} + 3 \times (600) \times p
\]

Differentiating both sides,

\[
\frac{dy}{dp} = \frac{-2 \times (600)^2}{p^2} + (3 \times 600) \quad -(ii)
\]

For the eqn \((i)\) to be either maxima or minima,

\[
\frac{dy}{dp} = 0
\]

\[
- \frac{720000}{p^2} + 1800 = 0
\]

Or

\[
p^2 = \frac{720000}{1800} = 400
\]

Or \(p = \pm 20\). Note: \(p > 0\).

Differentiating both sides of \((ii)\),

\[
\frac{d^2y}{dp^2} = \frac{2 \times 2 \times (600)^2}{p^3} \quad -(iii)
\]

Plugging both \(p = 20\) & \(p = -20\) into \((iii)\),

When \(p = 20\)

\[
\frac{1440000}{8000} = +180
\]

When \(p = -20\)

\[
\frac{-1440000}{-8000} = -180
\]

\[
\therefore eqn(i) \text{ reaches minima when } p = +20
\]

\[
\therefore \text{No of processors to be used } = +20.
\]
2. [3]  
(1) Show the forest after the first stage of the Sollin’s MST algorithm on the following weighted graph.

(2) How many iterations are needed in the worst case when Sollin’s algorithm is applied to a weighted graph of 200 nodes?

The no. of iterations for the worst case of Sollin’s is \( \log_2 \).

\[ \text{no. of iterations required for 200 nodes} = \log_2 200 = 7.64 \approx 8 \]

\[ \therefore \text{no. of iterations required} = 8. \]
[4] Solve the following recurrence relation. What is the time-complexity in big-O notation?

\[ T(n) = \begin{cases} 
1, & n = 2 \\
2T(n/2) + (n-1), & n > 2 
\end{cases} \]

de\[ n = 2^k \] (i)

\[ T(n) = 2T\left(\frac{n}{2}\right) + (n-1) \]
\[ = 2 \left[ 2T\left(\frac{n}{2^2}\right) + \left(\frac{n}{2^2}\right) - 1 \right] + (n-1) \]
\[ = 2^2 T\left(\frac{n}{2^2}\right) + (n-2) + (n-1) \]
\[ = 2^2 \left[ 2T\left(\frac{n}{2^3}\right) + \left(\frac{n}{2^3}\right) - 1 \right] + (n-2) + (n-1) \]
\[ = 2^3 T\left(\frac{n}{2^3}\right) + (n-2^2) + (n-2) + (n-1) \]
\[ = 2^3 T\left(\frac{n}{2^3}\right) + 3n - \left(2^2 + 2 + 1\right) \]

Iterating the loop \( k \) times

\[ = 2^k T\left(\frac{n}{2^k}\right) + kn - \left(\sum_{i=1}^{k} 2^i\right) \]
\[ = 2^k T\left(\frac{n}{2^k}\right) + kn - \left(\frac{2^k - 1}{2 - 1}\right) \]

Since \( 2^k + 2^{k-2} + \ldots + 2^1 + 2^0 = 2^k \)
from eqn (i)

\[ = nT\left(\frac{n}{n}\right) + n \log_2 n - (n-1) \]
Since \( T(1) = 1 \)
No!

\[ = n \log_2 n + 1 \]

\[ : O(n) = (n \log_2 n) \]
Show how the following 16 values would be sorted by the following algorithms/architectures. (Choose 3)

7, 9, 10, 2, 3, 6, 16, 1, 14, 5, 15, 8, 4, 11, 13, 12

(1) Batcher's bitonic merge
(2) Stone's perfect shuffle
(3) 2-D mesh with snake order
(4) Odd-even sorting network
(5) Merge-split sort with p = 4

(3) 2-D Mesh with snake order.

Step 1: Row Sort

Step 2: Column Sort

Step 4: Column Sort

Step 5: Row Sort

Final Sorted Order.
(1) Batchelor's Bitonic Sort

Step 1: Converting the input to a bitonic sequence

Step 2 will actually

do the sort on the bitonic

sequence which we

created in Step 1

1, 2, 3, 6, 7, 9, 10, 16

15, 14, 13, 12, 11, 8, 5, 4

Ascending ← Descending

Step 2 is continued in next page
The bitonic sequence, which is the input for the sort, is taken from Step 1 (previous page).
(4) Odd - Even Sorting Network