1. Node and Edge Eliminations

We define node and edge eliminations in Algorithm 1.

**Algorithm 1** Node and edge eliminations.

```plaintext
1: function NODEELIMINATION(G)
2:     if exist a node \( l_j \) with a single in-edge \( e_1 = (l_i, l_j) \) and a single out-edge \( e_2 = (l_j, l_k) \) then
3:         \( e' = (l_i, l_k) \)
4:         \( G' = G - l_j - e_1 - e_2 + e' \)
5:         return \( G' \)
6:     else
7:         return \( G \)
8: end function

10: function EDGEELIMINATION(G)
11:     if exist two edges \( e_1 = (l_i, l_j) \) and \( e_2 = (l_i, l_j) \) then
12:         \( e' = (l_i, l_j) \)
13:         \( G' = G - e_1 - e_2 + e' \)
14:         return \( G' \)
15:     else
16:         return \( G \)
17: end function
```

**Theorem 1.** Assume \( G' = \text{NodeElimination}(G) \) and \( l_j \) is the eliminated layer. If \( S_o' \) is an optimal strategy for \( G' \), then \( S_o = S_o' + \hat{c}_j \) is an optimal strategy for \( G \), where

\[
\hat{c}_j = \arg\min_{c_j} \{t_c(n_j, c_j) + t_x(n_j, c_j) + t_x(e_1, c_j) + t_x(e_2, c_j, c_k)\}
\]

**Proof.** It is equivalent to prove that \( t_o(G, S_1) \geq t_o(G, S_o) \) for any other strategy \( S_1 \) we assume layer \( l_i \) has parallelization configuration \( c_{i1} \in S_1 \). We prove this inequality by using the following path.

\[
t_o(G, S_1) \geq t_o(G', S_1) \geq t_o(G', S_o') = t_o(G, S_o)
\]

**Proof of Equation 2.** The difference between \( t_o(G, S_1) \) and \( t_o(G', S_1) \) is

\[
t_o(G, S_1) - t_o(G', S_1) = t_x(e_1, c_{j1}) + t_x(e_2, c_{j1}) - t_x(e', c_{j1})
\]

This is because all other layers except \( l_j \) use the same configurations in \( t_o(G, S_1) \) and \( t_o(G', S_1) \), and therefore all cost functions non-related to \( l_j \) are eliminated in the subtraction. The remaining parts are \( l_j, e_1, \) and \( e_2 \), which no longer exist in \( G' \) after node elimination, and \( e' \) that is added to \( G' \). Recall that \( t_x(e', \cdot, \cdot) \) is defined as follows.

\[
t_x(e', c_i, c_k) = \min_{c_j} \{t_c(l_j, c_j) + t_x(l_j, c_j) + t_x(e_1, c_i, c_j) + t_x(e_2, c_i, c_k)\}
\]

Combining Equation 5 and 6, we have \( t_o(G, S_1) \geq t_o(G', S_1) \).

**Proof of Equation 3.** Since \( S_o' \) is an optimal strategy for \( G' \), the inequality holds by definition.

**Proof of Equation 4.** Similarly, the difference between \( t_o(G', S_o') \) and \( t_o(G, S_o) \) is

\[
t_o(G, S_o) - t_o(G', S_o') = t_x(l_j, \hat{c}_j) + t_x(e_1, \hat{c}_j) + t_x(e_2, \hat{c}_j, c_k)
\]

This is because \( S_o = S_o' + \hat{c}_j \), and therefore all cost functions non-related to \( l_j \) are eliminated. We can prove Equation 4 by bringing Equation 1 into Equation 7.

**Theorem 2.** Assume \( G' = \text{EdgeElimination}(G) \), and \( S_o' \) is an optimal strategy for \( G' \), then \( S_o = S_o' + \hat{c}_j \) is an optimal strategy for \( G \).

**Proof.** We can use the same path to prove this theorem.

**Proof of Equation 2.** The difference between \( t_o(G, S_1) \) and \( t_o(G', S_1) \) is

\[
t_o(G, S_1) - t_o(G', S_1) = t_x(e_1, c_{j1}) + t_x(e_2, c_{j1}) - t_x(e', c_{j1})
\]
Recall that \( t_X(e', \cdot, \cdot) \) is defined as follows.

\[
t_X(e', c_i, c_j) = t_X(e_1, c_i, c_j) + t_X(e_2, c_i, c_j)
\]  
(9)

Combining Equation 8 and 9, we have \( t_o(G, S_1) = t_o(G', S_1) \).

**Proof of Equation 3.** The inequality holds since \( S_o' \) is an optimal strategy for \( G' \).

**Proof of Equation 4.** The difference between \( t_o(G', S_o') \) and \( t_o(G, S_o) \) is

\[
t_o(G, S_o) - t_o(G', S_o')
= t_X(e_1, c_i, c_j) + t_X(e_2, c_i, c_j) - t_X(e', c_i, c_j)
= 0
\]  
(10)