APPENDIX A

A.1 Pseudocode of OTAB algorithm

Algorithm 1 OTAB Algorithm

Input: \( G = (V, E), s, A, T. \)
Output: Broadcast Scheduling \( TTS : V \rightarrow 2^N \).
1: Apply Dijkstra’s algorithm on the graph \( G \) to construct the shortest path tree \( T_{SP} \) rooted at the source node \( s \).
2: Assign \( \text{MaxLatency}(T_{SP}) \) to \( D \), and divide \( V \) into different layers \( L_0, L_1, \ldots, L_D \) based on \( T_{SP} \).
3: Invoke Algorithm 2 to construct the MIS’es and the broadcast tree, and color the parent nodes with two coloring methods \( f_1 \) and \( f_2 \). Use \( m_1 \) and \( m_2 \) to denote the numbers of colors required by \( f_1 \) and \( f_2 \) respectively.
4: \( t \leftarrow 0 \)
5: for \( i \leftarrow 1 \) to \( D \) do
6: if \( L_i \neq \emptyset \) then
7: for each node \( u \in M_i \) do
8: Schedule node \( u \)'s parent \( P(u) \) to transmit at time slot \( t + f_1(P(u))[T] + j \), where \( j \) equals to \((i - 1) \mod |T|\), and add this time slot into \( TTS(P(u)) \).
9: \( t \leftarrow t + m_1[T] \)
10: for each node \( w \in L_i \setminus M_i \) do
11: Schedule node \( w \)'s parent \( P(w) \) to transmit at time slot \( t + f_2(P(w))[T] + j \), and add this time slot into \( TTS(P(w)) \).
12: \( t \leftarrow t + m_2[T] \)

Algorithm 2 Construct the MIS’es and the broadcast tree, and color the parent nodes

**Step I:** Construct the MIS’es
1: Partition \( V \setminus \{s\} \) into different subsets \( U_0, U_1, \ldots, U_{|T|-1} \) according to the nodes’ active time slots.
2: for \( j \leftarrow 0 \) to \( |T| - 1 \) do
3: Construct an MIS \( Q_j \) of \( G[U_j] \) layer by layer
4: for \( i \leftarrow 1 \) to \( D \) do
5: Divide all the nodes at layer \( L_i \) into two subsets: an IS \( M_i \) and a subset \( L_i \setminus M_i \), where \( M_i \) belongs to \( Q_j \) and \( j \) equals to \((i - 1) \mod |T|\).

**Step II:** Construct the broadcast tree
1: \( T_B \leftarrow (V_B, E_B) \), \( V_B \leftarrow V, E_B \leftarrow \emptyset \)
2: for \( i \leftarrow 1 \) to \( D \) do
3: Use the rule (if a node covers most nodes that do not have a parent yet, this node is first chosen as the parent node) to do:
   1: Choose some nodes at higher layers as the parent nodes of nodes in \( M_i \)
   2: Choose some nodes in \( M_i \) as the parent nodes of nodes in \( L_i \setminus M_i \) and nodes at lower layers with the same active time slot
4: Collect the parent nodes of nodes in \( M_i \) and \( L_i \setminus M_i \) into two sets \( W_1 \) and \( W_2 \) respectively based on the chosen order of these parent nodes, where \( j \) equals to \((i - 1) \mod |T|\).
5: Collect each pair \( (P(u), u) \) into \( E_B \) for each node \( u \in L_i \).

**Step III:** Color the parent nodes
1: for \( j \leftarrow 0 \) to \( |T| - 1 \) do
2: Apply a proper D2-coloring method to color the parent nodes in \( W_j \) by the front-to-back ordering. Use \( f_{1,j} \) to denote this coloring method.
3: Apply a proper D2-coloring method to color the parent nodes in \( W_j \) by the smallest-degree-last ordering. Use \( f_{2,j} \) to denote this coloring method.

A.2 Details and Pseudocode of UTB Algorithm

UTB algorithm is proposed for the all-to-all MLBSDC problem under the unit-size message model. It contains two processes. During the first process, all the messages are gathered to one special node. During the second process, these messages are broadcast one by one from this special node to all the other nodes. The pseudocode of UTB algorithm is given in Algorithm 3.

Algorithm 3 UTB Algorithm

Input: \( G = (V, E), A, T. \)
Output: Broadcast Scheduling \( TTS : V \rightarrow 2^N \).
1: Find a special node \( s \) such that the maximum latency of the shortest path tree \( T_{SP} \) rooted at this node is the minimum.
2: Assign \( \text{MaxLatency}(T_{SP}) \) to \( D \), and divide \( V \) into different layers \( L_0, L_1, \ldots, L_D \) based on \( T_{SP} \).
3: Invoke Algorithm 2 to construct the MIS’es and the broadcast tree, and color the parent nodes with two coloring methods \( f_1 \) and \( f_2 \). Use \( m_1 \) and \( m_2 \) to denote the numbers of colors required by \( f_1 \) and \( f_2 \) respectively.
4: Set \( m' \) as \( \text{max}(m_1, 0 \leq j \leq |T| - 1) \) and set \( m'' \) as \( \text{max}(m_2, 0 \leq j \leq |T| - 1) \).
5: Invoke Algorithm 4 to gather the messages of all the nodes to node \( s \).
6: Node \( s \) sends the message \( l \) \((1 \leq l \leq n) \) at time slot \( t_l \), where \( t_l = t + (l - 1)(m_1 + m'' - 1) \), and \( t \) is the time when the data gathering completes.
7: for \( i \leftarrow 1 \) to \( n \) do
8: \( t' \leftarrow t_l \)
9: for \( i \leftarrow 1 \) to \( D \) do
10: if \( L_i \neq \emptyset \) then
11: for each node \( u \in M_i \) do
12: Schedule node \( u \)'s parent \( P(u) \) to transmit at time slot \( t' + f_1(P(u))[T] + j \), where \( j \) equals to \((i - 1) \mod |T|\), and add this time slot into \( TTS(P(u)) \).
13: for each node \( w \in L_i \setminus M_i \) do
14: Schedule node \( w \)'s parent \( P(w) \) to transmit at time slot \( t' + f_2(P(w))[T] + j \), and add this time slot into \( TTS(P(w)) \).
15: \( t' \leftarrow t' + m''[T] \)

Algorithm 4 Data Gathering

1: Construct the BFS tree \( T_{BFS} \) rooted at node \( s \).
2: Assign \( \text{MaxDepth}(T_{BFS}) \) to \( \beta \), and divide \( V \) into \( S_0, S_1, \ldots, S_\beta \).
3: \( t \leftarrow 0 \)
4: while node \( s \) has not received \( n - 1 \) messages from other nodes do
5: for \( i \leftarrow 0 \) to \( 2 \) do
6: for all \( k \equiv l \mod 3 \) and \( 0 \leq k \leq \beta \) do
7: Find a node \( u \in S_k \), such that one of its neighboring nodes \( v \in S_{k'}(k' > k) \) has a message to transmit or forward.
8: Schedule node \( v \) to transmit at time slot \( t+A(u) \), and add this time slot into \( TTS(v) \).
9: \( t \leftarrow t + |T| \)
10: return \( TTS, t \)

First, UTB algorithm finds a special node \( s \) from \( V \) such that the maximum latency of the shortest path tree \( T_{SP} \) rooted at this node is the minimum. We compute the maximum latency of the shortest path tree rooted at every node, and find the node \( s \). Then we construct the maximal independent sets and
that used in the OTAB algorithm. The difference is that, we use $l_1$ schedule the broadcast of these messages. We assume that node $n_1(a)$. We assume that node

We take the same network as that in Example 1

in the first group of subsets $S_0$, $S_1$, $S_2$, ..., $S_B$ according to their depths in $T_{BFS}$, where $B$ is the maximum depth in $T_{BFS}$.

Then the transmissions are scheduled in an interleaving manner in the subsets with an interval depth of three. That is, in the first group of subsets $S_0$, $S_3$, $S_6$, ..., we find a node $u$ from each subset such that its neighboring node $v$ in the subset with a greater depth has a message to transmit or forward. Then the transmission from node $v$ to node $u$ is scheduled at time slot $t + A(u)$, where $t$ is the current time slot. These transmissions can finish in $|T|$ time slots, so the current time slot $t$ increases by $|T|$ time slots to avoid the collision. The next group of subsets with an interval depth of exactly three (i.e., $S_1$, $S_4$, $S_7$, ...) are handled in a similar way, so do the final group of subsets $S_2$, $S_5$, $S_8$, ... . This process repeats until node $s$ receives $n - 1$ messages from other nodes.

Once the messages are gathered to node $s$, we start to schedule the broadcast of these messages. We assume that the messages are numbered by $1, 2, ..., n$, which is the order to be sent by node $s$. Node $s$ starts to send one message every $(m_1 + m_2)|T|$ time slots. For example, node $s$ will start to send the message $l (1 \leq l \leq n)$ at time slot $t + (l - 1)(m_1 + m_2)|T|$. The scheduling of the transmissions of each message is similar to that used in the OTAB algorithm. The difference is that, we use $m_1$ and $m_2$ to replace $m_1j$ and $m_2j$ respectively. This process is repeated $n$ times for $n$ messages.

Example 2. We take the same network as that in Example 1 to explain the UTB algorithm. The network is shown in Fig. 1(a). We assume that node $s$ is the special node. The shortest path tree and the broadcast tree are shown in Fig. 1(b) and Fig. 1(c) respectively. During the data gathering process, we first construct the BFS tree $T_{BFS}$ rooted at node $s$ as shown in Fig. 1(d). All the nodes are divided into five subsets, i.e., $S_0 = \{s\}$, $S_1 = \{v_1, v_2, v_3\}$, $S_2 = \{v_4, v_5, v_6\}$, $S_3 = \{v_7, v_8, v_9\}$ and $S_4 = \{v_{10}, v_{11}\}$. The group of subsets $S_0$ and $S_3$ is first handled, e.g., node $s$ in $S_0$ receives the message of node $v_1$ at time slot $t + A(s) = 2$, and node $v_0$ receives the message of node $v_{10}$ at time slot $t + A(v_0) = 0$. The other groups of subsets are handled in a similar manner until node $s$ receives 11 messages from all the other nodes. During the broadcast process, node $s$ broadcasts all the messages one by one to other nodes every $(m_1 + m_2)|T| = (2 + 1) \times 4 = 12$ time slots.

A.3 Details and Pseudocode of UNB Algorithm

UNB algorithm is proposed for the all-to-all MLBSDC problem under the unbounded-size message model. The pseudocode of UNB algorithm is given in Algorithm 5.

Algorithm 5 UNB Algorithm

Input: $G = (V, E)$, $A, T$.
Output: Broadcast Scheduling $TTS : V \rightarrow 2^N$.
1: Find a special node $s$ such that the maximum latency of the shortest path tree $T_{SPT}$ rooted at this node is the minimum.
2: Assign MaxLatency($T_{SPT}$) to $D$, and divide $V$ into different layers $L_0, L_1, ..., L_d$ based on $T_{SPT}$.
3: Invoke Algorithm 2 to construct the MIS’es and the broadcast tree, and color the parent nodes with two coloring methods $f_1j$ and $f_2j$. Use $m_1j$ and $m_2j$ to denote the numbers of colors required by $f_1j$ and $f_2j$, respectively.
4: Invoke Algorithm 6 to aggregate the messages to node $s$.
5: Node $s$ sends the combined message at time slot $t$ when the data aggregation completes.
6: for $i \leftarrow 1$ to $D$ do
7: \hspace{1em} if $L_i \neq \emptyset$ then
8: \hspace{2em} for each node $u \in M_i$ do
9: \hspace{3em} Schedule node $u$’s parent $P(u)$ to transmit at time slot $t + f_1j(P(u))|T| + j$, where $j$ equals to $(i - 1) \mod |T|$, and add this time slot into $TTS(P(u))$
10: \hspace{2em} $t \leftarrow t + m_1j|T|
11: \hspace{1em} for each node $w \in L_i \setminus M_i$ do
12: \hspace{2em} Schedule node $w$’s parent $P(w)$ to transmit at time slot $t + f_2j(P(w))|T| + j$, and add this time slot into $TTS(P(w))$
13: \hspace{2em} $t \leftarrow t + m_2j|T|
Recall that in this case a node can combine the messages as one message, and can send the combined message in one time slot. The UNB algorithm also contains two processes: data gathering and broadcasting. Note that the node responsible for data gathering only needs to send one combined message after receiving all the other messages. The data gathering process is different from that in the UTB algorithm. It can be thought as a data aggregation process, which has been extensively studied in [28], [29]. However, none of these algorithms considers the duty-cycled scenarios.

Similar to UTB, UNB finds the special node $s$, divides all the nodes into different layers, constructs the MIS’es and the broadcast tree, and colors all the parent nodes with two coloring methods $f_1j$ and $f_2j$. The data aggregation process works from the bottom layer to the top layer. At each layer $L_i$, nodes in $L_i \setminus M_i$ first transmit the messages to their parent nodes
Algorithm 6 Data Aggregation

1: \(t \leftarrow 0\)
2: Every node combines its received messages before transmitting the messages.
3: for \(i \leftarrow D\) down to 1 do
4: \(X_i \leftarrow L_i \setminus M_i, Y_i \leftarrow \{\text{Parent nodes of nodes in } L_i \setminus M_i\}\)
5: \(X_2 \leftarrow M_2, Y_2 \leftarrow \{\text{Parent nodes of nodes in } M_i\}\)
6: for \(k \leftarrow 1\) to 2 do
7: \(c \leftarrow 0, X \leftarrow X_i, Y \leftarrow Y_i, t' \leftarrow t\)
8: while \(X \neq \emptyset\) do
9: \(\text{Find a minimal cover } Z \subseteq Y \text{ of } X\).
10: for each node \(z \in Z\) do
11: \(\text{Find a neighbor } x \in X \text{ of } z\).
12: \(\text{Schedule node } x \text{ to transmit at time slot } t + c|T| + A(z), \text{ and add this time slot into } TTS(x)\).
13: if \(t' < t + c|T| + A(z) + 1\) then
14: \(t' \leftarrow t + c|T| + A(z) + 1\)
15: \(X \leftarrow X \setminus \{x\}\)
16: \(Y \leftarrow Z\)
17: \(c \leftarrow c + 1\)
18: \(t \leftarrow \lceil t' / |T| \rceil / |T|\)
19: return \(TTS, t\).

Algorithm 6 iteratively, and then do the nodes in \(M_i\). Since the scheduling method proposed in [29] is so far the best, we modify this method to schedule the transmissions. The difference is that a transmitter node transmits the message at the active time slot of the receiver node in one scheduling period. We take the following example to illustrate the data aggregation process.

Fig. 2. An example to illustrate the data aggregation process in UNB algorithm.

Example 3. As shown in Fig. 2(a), nodes \(v_4, v_5, v_6, v_7\) and \(v_8\) are all in \(L_2 \setminus M_i\) (or \(M_i\)), and their parent nodes are \(v_1, v_2\) and \(v_3\). The solid lines denote the parent-child relationship, and the dashed lines denote the neighboring relationship. The children nodes send the messages to their parent nodes iteratively, and the transmitting time increases by \(|T|\) after each iteration to avoid the collision. In the first iteration, every parent’s proprietary child (not adjacent to other parents) transmits the message. As shown in Fig. 2(b), nodes \(v_4, v_6\) and \(v_8\) transmit the messages at time slots \(t + A(v_1), t + A(v_2)\), and \(t + A(v_1)\) respectively. In the second and third iterations, nodes \(v_3\) and \(v_5\) transmit the messages to node \(v_2\) at time slots \(t + |T| + A(v_2)\) and \(t + 2|T| + A(v_2)\) respectively as shown in Fig. 2(c) and Fig. 2(d).

Fig. 2. An example to illustrate the data aggregation process.

Fig. 3. An example to illustrate the Prune method.

A.4 Details of Prune Method

Before detailing the Prune method, we first give an example to illustrate the idea of this method. As shown in Fig. 3, five nodes \(v_1, v_2, v_3, v_4\) and \(v_5\) share the same active time slot. Nodes \(v_1, v_2\) and \(v_4\) are at layer \(L_i\), and nodes \(v_3\) and \(v_5\) are at layer \(L_2\), where \(i_1 < i_2\), and \(i_2 \mod |T| = 1\) and \(i_2 \mod |T| = 2\) respectively. The Prune method modifies the second phase as follows. In this phase, each node \(v \in M_i\) with children nodes of the same active time slot, we schedule this node to transmit at time slot \(t + f_2(v)|T| + j\), where \(j\) equals to \((i - 1) \mod |T|\). We call our three algorithms together with the Prune method as OTAB-P, UTB-P and UNB-P respectively.

APPENDIX B

B.1 NP-hardness Analysis

Lemma 1. Both the one-to-all and the all-to-all MLBSDC problems are NP-hard.

Proof: The NP-hardness of the one-to-all MLBSDC problem has been proved in [13]. We only need to prove that the all-to-all MLBSDC problem is NP-hard. If we set \(T\) as \(0\), then all the nodes are always active, and the all-to-all MLBSDC problem reduces to the conventional all-to-all MLBS problem. Since the conventional all-to-all MLBS problem has been proved to be NP-hard in [2], this lemma holds. \(\square\)
B.2 Correctness Analysis

Theorem 1. Both the OTAB and OTAB-P algorithms provide correct and collision-free broadcast schedulings.

Proof: This proof is given in Section 5.1 of the paper. □

Theorem 2. Both the UTR and UTB-P algorithms provide correct and collision-free broadcast schedulings.

Proof: After finding the special node $s$, the messages will be gathered to this node. For each node $u$ in the subset $S_k$, only if its neighboring node $v$ in the subset with a greater depth has a message to transmit or forward, the transmission from node $v$ to node $u$ is scheduled. In addition, the transmissions always occur from the nodes far away from node $s$ to the nodes close to node $s$ or node $s$, so ultimately node $s$ will receive all the messages. Afterward, node $s$ broadcasts the messages to all the other nodes by applying the similar method to OTAB or OTAB-P. The correctness of OTAB and OTAB-P have been proved, so UTB and UTB-P are correct.

The data gathering process works in an interleaving manner. Only the transmissions to nodes in the subsets whose depths have an exact interval of three are simultaneous. Moreover, only one transmission is scheduled for each subset. Clearly, these transmissions are collision-free.

Based on the proof above, the transmissions during the first phase of the broadcast process in UTB-P are collision-free. During the second phase of the broadcast process in UTB-P, the transmissions from nodes in $M_i$ are based on the colors of these nodes. We consider two kinds of transmissions from nodes in $M_i$ and from nodes in $M_j$. Similarly, if the nodes in $M_i$ do not share the same active time slot with the nodes in $M_j$, these two kinds of transmissions are collision-free. If they share the same active time slot, we can use the similar proof to that of Theorem 1 to prove that these two kinds of transmissions are collision-free. □

Theorem 3. Both the UNB and UNB-P algorithms provide correct and collision-free broadcast schedulings.

Proof: First, we consider the data aggregation process. The messages are aggregated from the bottom layer to the top layer. At each layer $L_n$, nodes in $L_{n+1}$ first deliver their messages to their parent nodes at higher layers or in $M_i$. Then the messages of nodes in $M_i$ are delivered to their parent nodes at higher layers. Therefore, node $s$ can ultimately receive all the $n$ messages. The broadcast process is similar to that in OTAB or OTAB-P, so UNB and UNB-P are correct.

The transmissions during the broadcast processes of UNB and UNB-P are collision-free according to Theorem 1. We only need to prove that the transmissions during the data aggregation process are collision-free. The data aggregation process works layer by layer. In each iteration at each layer, the proprietary neighbors are scheduled to deliver messages to the nodes in the minimal cover. These transmissions are apparently collision-free, and hence this theorem holds. □

B.3 Approximation Ratio Analysis

Lemma 2. To color the parent nodes, the required numbers of colors $m_{1j} \leq 5$ and $m_{2j} \leq 12$, where $0 \leq j \leq |T| - 1$.

Proof: First, we consider the set $W_{2j}$, which is an IS of $G$. According to [3], twelve colors are sufficient to color all the nodes in $W_{2j}$ by the smallest-degree-last D2-coloring method, so it follows that $m_{2j} \leq 12$.

We prove $m_{1j} \leq 5$ by contradiction. If $m_{1j} > 5$, we assume that node $v$ in $W_{1j}$ is colored with 5. This node must have one child node in $Q_j$. Based on the front-to-back coloring method, there must be five children $v_1, v_2, v_3, v_4$ and $v_5$ ordered before node $v$ in $W_{1j}$. These nodes are colored with the numbers from 0 to 4, and each of them has at least one child node in $Q_j$ which is adjacent to node v, because otherwise node $v$ can be colored with one number in $\{0, 1, 2, 3, 4\}$. Since these children nodes are different, node $v$ has more than five neighboring nodes in the MIS $Q_j$ which is contradictory [25]. □

Theorem 4. The approximation ratios of both the OTAB and OTAB-P algorithms are at most $17|T|$.

Proof: This proof is given in Section 5.2 of the paper. □

Lemma 3. $R + (n - 2)|T| + 1$ is the lower bound for the all-to-all MLBSDC problem under the unit-size message model, where $R$ is the graph-theoretic radius of the network.
Proof: We first consider the graph center \( s_c \) of the network. The maximum depth of the BFS tree \( T_{BFS} \) rooted at node \( s_c \) is the minimum among the BFS trees rooted at all the nodes in the network. This maximum depth is the graph-theoretic radius \( R \). It is easy to find that each packet should be transmitted at least \( R \) times to reach all the nodes. To complete the all-to-all broadcast task, the total number of transmissions is at least \( nR \). Therefore, we can find a node which transmits at least \( R \) times. Furthermore, this node requires to receive \( n - 1 \) messages from other nodes. Since a node cannot transmit or receive messages at the same time, the minimum latency for this node to complete these operations should be \( R + (n - 2)|T| + 1 \) time slots, which provides a lower bound for the all-to-all MLBSDC problem under the unit-size message model.

Lemma 4. For the all-to-all MLBSDC problem under the unit-size message model, the maximum latency \( D \) of the \( T_{SPT} \) rooted at the special node \( s \) is no larger than \((n - 2)|T|\).

Proof: Before we prove this lemma, we first prove that \( D \leq R|T| \). We consider the graph center \( s_c \). We construct the shortest path tree rooted at node \( s_c \) based on the latency defined in Eq. (1). We denote by \( D_{s_c} \) the maximum latency of this shortest path tree, and consider the \( T_{BFS} \) rooted at \( s_c \). The maximum latency of the \( T_{BFS} \) is bounded by \( R|T| \) if we do not take the collision into account. Since the shortest path tree provides the minimum latency between node \( s_c \) and any other nodes, we have that \( D_{s_c} \leq R|T| \). Furthermore, according to the definition of \( D \), it follows that \( D \leq D_{s_c} \leq R|T| \).

Next, we prove that \( R \leq n - 2 \) by contradiction. Without loss of generality, we consider \( n \geq 3 \). Obviously \( R \) is smaller than \( n \). If \( R \) is greater than \( n - 2 \), \( R \) must be \( n - 1 \). In this case, all the nodes must be connected one by one. If we choose an intermediate node, and construct the BFS tree rooted at this node, we can find that the newly constructed BFS tree has a maximum depth smaller than \( R \), which contradicts the assumption.

Theorem 5. The approximation ratios of both the UTB and UNB-P algorithms are at most \( 17|T| + 20 \).

Proof: Since the worst-case broadcast latency of UTB and UTB-P is the same, we only need to prove the approximation ratio of UTB algorithm. We first analyze the latency of the data gathering process shown in Algorithm 4. This algorithm gathers the messages to node \( s \) in an interleaving manner, and node \( s \) receives one message every \( 3|T| \) time slots. So it will take \( 3|T|(n - 1) \) time slots for node \( s \) to receive all the \( n - 1 \) messages.

Next, we analyze the latency of the broadcast process of \( n \) messages from node \( s \). Since node \( s \) sends one message every \((m_1^n + m_2^n)|T| \) time slots, the last message will be transmitted at time slot \( 3|T|(n - 1) + (m_1^n + m_2^n)|T|(n - 1) \). According to Theorem 4, the last message requires at most \( 17|T|D \) time slots to reach all the other nodes. In addition, we consider the case \( n \geq 3 \), and clearly the graph-theoretic radius \( R \geq 1 \). We denote by \( D_{UTB} \) the latency of UTB. According to Lemma 2, Lemma 3, and Lemma 4, it follows that

\[
D_{UTB} \leq 3|T|(n - 1) + (m_1^n + m_2^n)|T|(n - 1) + 17|T|D \\
\leq 3|T|(n - 1) + 17|T|(n - 1) + 17|T|D \\
\leq 20|T|(n - 2) + 20|T| + 17|T|(n - 2)|T| \\
= (17|T| + 20)(n - 2)|T| + 20|T| \\
< (17|T| + 20)(n - 2)|T| + (17|T| + 20)(R + 1) \\
= (17|T| + 20)(R + (n - 2)|T| + 1)
\]

Lemma 5. The latency of the data aggregation process shown in Algorithm 6 is at most \((\Delta + 5)|T|D \) time slots.

Proof: The messages are gathered from the bottom layer to the top layer in the \( T_{SPT} \). We analyze the latency of two phases during the data aggregation process at each layer. During the first phase, the nodes in \( L_1 \backslash M_i \) are first scheduled to transmit messages to their parent nodes. According to Algorithm 6, one node will always exist in the minimal cover during the whole iterations, and its degree decreases by at least 1 after each iteration. Therefore, the number of iterations is bounded by the maximum node degree \( \Delta \). In addition, the active time slot of every node is bounded by \(|T| - 1\). So the latency of the first phase is at most \((\Delta - 1)|T| + (|T| - 1) + 1 = \Delta|T| \) time slots. Since one node has at most five neighboring nodes in an IS, the latency of the second phase is at most 5|T| time slots. In the worst case, there are \( D + 1 \) layers, so the latency of the data aggregation process is bounded by \((\Delta + 5)|T|D \).

Theorem 6. The approximation ratios of both the UNB and UNB-P algorithms are at most \((\Delta + 22)|T|\).

Proof: Since UNB and UNB-P have the same worst-case broadcast latency, we only need to prove the approximation ratio of UNB algorithm. First, we claim that \( D \) is a trivial lower bound for the all-to-all MLBSDC problem under the unbounded-size message model. We analyze the latency of two processes in UNB. During the first process, the messages are gathered from the bottom layer to the top layer in the shortest path tree. According to Lemma 5, this process will take at most \((\Delta + 5)|T|D \) time slots to finish. During the second process, node \( s \) transmits the combined message to all the other nodes. The latency of this process is at most \( 17|T|D \) according to Theorem 4. We combine the latency of two processes as \((\Delta + 5)|T|D + 17|T|D = (\Delta + 22)|T|D \).

B.4 Overhead Analysis

Lemma 6. To solve the one-to-all MLBSDC problem, the minimum total number of transmissions \( M_{opt} \) is at least \( |Q|/5 \), where \( Q = \bigcup_{0 \leq j \leq |T| - 1} Q_j \).

Proof: First, we assume that there is an optimal algorithm for the one-to-all MLBSDC problem, which provides a broadcast scheduling with the minimum total number of transmissions. We denote by \( F_{opt} \) the set of the forwarding nodes chosen by this algorithm, and by \( M_{opt} \) the minimum total number of transmissions. The forwarding nodes may
transmit many times to inform its neighboring nodes with different active time slots, and we denote by \(N_v(u)\) the total number of transmissions required by node \(u\) in \(F_{opt}\). We create \(N_v(u)\) virtual nodes \(u_1, u_2, \ldots, u_{N_v(u)}\) for each node \(u\), and denote by \(V_v\) the set of all the virtual nodes created for nodes in \(F_{opt}\). Clearly, \(M_{opt}\) equals to \(|V_v|\).

Recall that one node has at most five neighboring nodes in an IS. When one virtual node \(v\) transmits the message at time slot \(t\), it informs at most five nodes in \(Q_j\), where \(j = t \mod |T|\). In order to inform all the nodes in \(Q_j(0 \leq j \leq |T| - 1)\), the total number of virtual nodes should be at least \(|Q_j|/5\). If two nodes belong to \(Q_j\), and \(Q_{j'} (j \neq j')\) respectively, they should have different active time slots, and the virtual nodes responsible for informing these two nodes should be different. Therefore, to inform all the nodes in \(Q = \bigcup_{0 \leq j \leq |T| - 1} Q_j\), the total number of virtual nodes which equals to \(M_{opt}\) should be at least \(|Q|/5\), i.e., \(|V_v| = M_{opt} \geq |Q|/5\).

**Lemma 7.** To solve the all-to-all MLBSDC problem under the unit-size message model, the minimum total number of transmissions \(M'_{opt}\) is at least max\(\{nR, n|Q|/5\}\), where \(Q = \bigcup_{0 \leq j \leq |T| - 1} Q_j\).

**Proof:** According to the proof of Lemma 3, we can achieve that \(M'_{opt} \geq nR\). Since one message requires at least \(|Q|/5\) transmissions to reach all the nodes according to Lemma 6, and the messages have to be sent by one without combination, so we can achieve that \(M'_{opt} \geq n|Q|/5\). Hence, this lemma holds.

**Lemma 8.** To solve the all-to-all MLBSDC problem under the unbounded-size message model, the minimum total number of transmissions \(M''_{opt}\) is at least \(n\).

**Proof:** Since every node has to transmit its message to the other nodes, the minimum total number of transmissions \(M''_{opt}\) is at least \(n\). Hence, this lemma holds.

**Theorem 7.** The total numbers of transmissions scheduled by the OTAB, OTAB-P, UTB, UTB-P, UNB and UNB-P algorithms are at most 15, 10, 17, 12, 4 and 3 times as large as the minimum total numbers of transmissions respectively.

**Proof:** First, we analyze the overhead of the OTAB and OTAB-P algorithms. The scheduling in the OTAB algorithm proceeds layer by layer. At each layer \(L_i\), the message is first broadcasted to each node in \(L_i\). Only one transmission is required to inform node \(u\), so the number of transmissions during the first phase is bounded by \(|M_i|\). The message is then broadcasted to nodes in \(L_i \setminus M_i\). Recall that the parent nodes of these nodes are in \(M_i\) or \(M_j\), where \(i < j\), and \((i - 1)\mod |T|\) and \((i - 1)\mod |T|\) are both equal to \(j\). Specifically, if \(1 \leq i \leq |T|\), these parent nodes are in \(M_i\). Otherwise, these parent nodes are in \(M_{i-|T|}\) or \(M_{j}\). These parent nodes only require to transmit once to inform the nodes in \(L_i \setminus M_i\). Hence the number of transmissions during the second phase is bounded by \(|M_{i-|T|}| + |M_i|\).

Therefore, the total number of transmissions scheduled by OTAB is bounded by \(\sum_{1 \leq i \leq D} |M_i| + |M_{i-|T|}| + |M_i| \leq 2\sum_{1 \leq i \leq D} |M_i| + \sum_{1 \leq i \leq D} |M_{i-|T|}| < 3\sum_{1 \leq i \leq D} |M_i|\). Note that, \(\sum_{1 \leq i \leq D} |M_i| = \sum_{0 \leq j \leq |T| - 1} Q_j = |Q|\). According to Lemma 6, \(M_{opt} \geq |Q|/5\). So the total number of transmissions scheduled by OTAB is at most \(3|Q|/\left(\frac{|Q|}{5}\right) = 15\) times as large as \(M_{opt}\).

We then consider the OTAB-P algorithm. Based on the proof above, the number of transmissions during the first phase at each layer \(L_i\) is bounded by \(|M_i|\). During the second phase, each node \(v\) in \(M_i\) with children nodes in \(L_i \setminus M_i\) or at lower layers with the same active time slot transmits only once to inform its children nodes. Therefore the number of transmissions during this phase is bounded by \(|M_i|\). We combine these two numbers of transmissions as \(|M_i| + |M_i| = 2|M_i|\). The total number of transmissions scheduled by OTAB-P is bounded by \(\sum_{1 \leq i \leq D} 2|M_i| = 2\sum_{0 \leq j \leq |T| - 1} |Q_j| = 2|Q|\), which is at most \(2|Q|/\left(\frac{|Q|}{5}\right) = 10\) times as large as \(M_{opt}\).

Second, we analyze the overhead of the UTB and UTB-P algorithms. During the data gathering process, the message of every node in \(S_k\) \((1 \leq k \leq \beta)\) requires being transmitted \(k\) times to reach the special node in \(S_0\). So the number of transmissions during this process is \(\sum_{1 \leq k \leq \beta} \sum_{1 \leq j \leq |S_k|} \beta \sum_{1 \leq i \leq j} |S_i| = \beta(n - 1)\). It follows that \(\beta \leq d \leq 2R\), where \(d\) is the graph-theoretic diameter of the network. According to Lemma 7, \(M'_{opt} \geq nR\), and \(\beta(n - 1)/M'_{opt} \leq 2R(n - 1)/nR < 2\), so the number of transmissions during the data gathering process is at most 2 times as large as \(M_{opt}\).

Using the similar proof to that for OTAB and OTAB-P, we can prove that the broadcast processes of UTB and UTB-P require at most \(3n|Q|\) and \(2n|Q|\) transmissions respectively. According to Lemma 7, \(M'_{opt} \geq n|Q|/5\), so the numbers of transmissions scheduled by UTB and UTB-P during the broadcast process are at most 15 and 10 times as large as \(M_{opt}\) respectively. We combine two numbers of transmissions during two processes, and thus achieve that the total numbers of transmissions of UTB and UTB-P are at most 17 and 12 times as large as \(M_{opt}\) respectively.

Finally, we analyze the overhead of UNB and UNB-P algorithms. During the data aggregation process, every node except the special node can combine its received message as one message and only transmits the combined message once to its parent node, so the number of transmissions during this process is \(n-1\). According to the proof for OTAB and OTAB-P, the numbers of transmissions during the broadcast process in UNB and UNB-P are at most \(3|Q|\) and \(2|Q|\) respectively. Since \(|Q| \leq n - 1\), the total numbers of transmissions scheduled by UNB and UNB-P are at most \(4(n-1)\) and \(3(n-1)\) respectively. According to Lemma 8, \(M''_{opt} \geq n\), and thus the total numbers of transmissions of UNB and UNB-P are at most 4 and 3 times as large as \(M''_{opt}\) respectively. Hence this theorem holds.

**B.5 Time Complexity Analysis**

**Theorem 8.** The time complexity of all the OTAB, OTAB-P, UTB, UTB-P, UNB and UNB-P algorithms is at most \(O(|T|^2 n^2 + n^3)\).

**Proof:** First, we analyze the time complexity of OTAB and OTAB-P algorithms. The running time of constructing the shortest path tree by applying Dijkstra’s algorithm is \(O(n^2)\) [27]. It takes \(O(n)\) time to divide all the nodes into different layers. It takes at most \(O(|T|^2 n^2)\) time to construct the MIS’es and takes at most \(O(|T|^2 n^2 + n^3)\) time to construct
the broadcast tree. The D2-coloring process can be completed in $O(n^2)$ time. The running time of scheduling the broadcast is at most $O(n)$. We combine all the running time and achieve that the time complexity of the OTAB algorithm is at most $O(|T|^2 n^2 + n^3)$. The major difference between OTAB and OTAB-P is the second phase of the broadcast process. It takes at most $O(n)$ time to schedule the broadcast in OTAB-P, so the time complexity of OTAB-P is also at most $O(|T|^2 n^2 + n^3)$.

Second, we analyze the time complexity of UTB and UTB-P algorithms. It takes at most $O(n^3)$ time to find the special node and at most $O(n)$ time to divide all the nodes. Based on the proof for OTAB and OTAB-P, the running time of Algorithm 2 is at most $O(|T|^2 n^2 + n^3)$. It takes at most $O(n^2)$ time to gather the messages and to schedule the broadcast in both the UTB and UTB-P algorithms. We combine all the running time and achieve that the time complexity of both the UTB and UTB-P algorithms is at most $O(|T|^2 n^2 + n^3)$.

Finally, we analyze the time complexity of UNB and UNB-P algorithms. The major difference between UNB and UTB is the data gathering process and the broadcast process. In both UNB and UNB-P, it takes at most $O(n^3)$ time to complete the data gathering process, and takes at most $O(n)$ time to schedule the broadcast. Therefore, based on the proof for UTB and UTB-P, we can achieve that the time complexity of both the UNB and UNB-P algorithms is at most $O(|T|^2 n^2 + n^3)$. \(\square\)