

Evaluating the Overhead of Setting up TDMA Communication in Wireless Sensor Networks

AS 2009

Yu Li

Advisor: Federico Ferrari
Professor: Prof. Dr. Lothar Thiele

Zurich
19th February 2010

Acknowledgements

I would like to thank my supervisor Federico Ferrari for his constant support during this semester thesis. He proposed three ideas about scheduling propagation algorithms and assisted me revising the report and presentation material. He also spent much time discussing with me about the problems during project.

1. Introduction.....	1
2. Implementation of Algorithms.....	3
2.1. Implementation of Basic Algorithm.....	3
2.1.1. Slot specification.....	3
2.1.2. Sending and Receiving.....	6
2.1.3. Transmission Range	7
2.1.4. Radio on, radio off and switching	8
2.1.5. Radio-on time calculation	9
2.1.6. Time sequence diagram.....	10
2.2. Implementation of Extended Algorithm.....	12
2.2.1. Basic Idea.....	12
2.2.2. Adjustment of Slot Specification.....	13
2.2.3. Radio-on time calculation.	14
2.2.4. Time sequence diagram.....	14
2.3. Implementation of Odd-even Slot Algorithm.....	15
2.3.1. Basic Idea.....	15
2.3.2. Adjustment of slot specification.....	16
2.3.3. Radio-on time calculation.	17
2.3.4. Time sequence diagram.....	18
3. Link Failure	20
3.1. Time-out.....	20
3.2. Revised slot specification and state machine chart	21
3.2.1. Basic Algorithm	21
3.2.2. Extended Algorithm	23
3.2.3. Odd-even Slot Algorithm	24
4. Results Analysis.....	26
4.1. Evaluation Metrics	26
4.1.1. Parent-children Ratio	26
4.1.2. Total Radio-on Time and Total Phase Length	26
4.2. Evaluation of Algorithms w/o Link Failure	27
4.2.1. Evaluation Results.....	27
4.2.2. Formal Analysis of the Results.....	31
4.2.3. Evaluation Results Analysis.....	36
4.3. Evaluation of Algorithms with Link Failure	37
Conclusions.....	42
Bibliography	43

1. Introduction

Because of the limited available energy supply to a Wireless Sensor Network (WSN) node, an energy efficient MAC protocol is necessary to guarantee the long-term operation of the WSN. Up till now, a wide variety of protocols, such as combinations of contention based and TDMA schemes, distributed TDMA schemes, have been proposed. However a pure TDMA, collision-free scheme is still not a fully understood area.

This thesis attempts to make a contribution to the design of the pure-TDMA, collision-free scheme protocol. The protocol has three parts which are synchronization phase, scheduling propagation phase and the slotted TDMA communication phase. See figure 1.1.

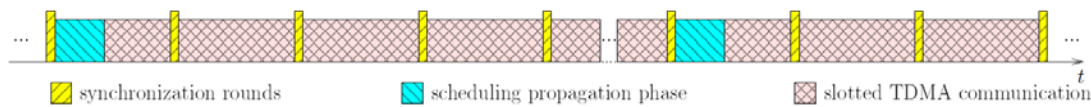


Figure 1.1: Targeted TDMA organization of a sensor network

Synchronization phase needs to be executed frequently in order to keep the WSN in a globally synchronized state, because the communication phase is strictly slotted and therefore needs clocks to be regularly synchronized. A large quantity of synchronization protocols for WSN are feasible and aim at global synchronization, such as (Time-sync Protocol for Sensor Networks) TPSN, (Flooding Time Synchronization Protocol) FTSP.

Communication phase is designed to operate periodically for regular data collection. The communication phase is split into many equal-length slots. Slots are assigned exclusively to a node for transmission, which means that during a certain slot only one transmission is allowed to occur, i.e. only one node can transmit while only one receiver can receive. During each round of data collection, information originates from the leaf node which acquires environmental information by employing the sensor. Then it arrives at the sink node or base station after being forwarded by several intermediate nodes. The information is transported along a tree structure. The tree construction is not discussed in this thesis. We assume that the tree already exists

Scheduling propagation phase is regularly called when the transmission tree needs reconstruction or some nodes leave or join the tree. This phase precedes the communication phase and plays a role of organizing the communication phase. As we just said, communication phase consists of many equal-length slots, which need to be exclusively assigned to a certain node. The slot assignment is performed in this phase. By taking advantage of the tree topology, every node is traversed according to a certain tree traversal order, starting from the root. At each node, certain communication slots are taken, which intrinsically guarantees the exclusive distribution of communication slots.

In the slots assignment, a slot budget mechanism is used. A slot budget corresponds to slots that

are still available for the communication phase. At the beginning of the scheduling phase, slot budget corresponds to entire range of all the communication slots, while the size of slot budget shrinks during assignment, after each node makes its reservation. Obeying the traversal order, each node is provided with a slot budget by its parent. It then reserves a fraction of slot budget for communication. Nodes must be taken from the tail to the head of the given slot budget, so that children always transmit before the parent during the communication phase. Afterwards remaining slots will be distributed to children.

In this thesis, we focus on the scheduling propagation phase, and propose three algorithms for communication slots assignment in this phase. All of them are analytically evaluated by studying the relationship between a wide topology variation and some performance metrics, such as duty cycle, radio-on time.

The remaining parts of the thesis are organized as follows. Chapter 2 gives specific introduction on three scheduling propagation algorithms. Chapter 3 explains the adjustment on three algorithms brought by link failure. Chapter 4 first introduces the evaluation metrics such as parent-children ratio, total phase length and total radio-on time and then analyzes the results both without link failure and with link failure. Chapter 2 and 3 are algorithm part, while chapter 4 is evaluation part

2. Implementation of Algorithms

In this section, three algorithms will be introduced to finish scheduling propagation. Even though three algorithms differ themselves in details, they possess the same basic procedure of scheduling propagation.

The basic procedure works as follows. Focusing on one node, it will, in the beginning of each slot, sense the carrier. If the node successfully senses the carrier, it will receive the slot budget. After a few internal operations, it will send the budget to its children. When all its children are traversed and all of them have reserved the slot in the budget, this node will then send the remaining budget back to its parent.

2.1. Implementation of Basic Algorithm

2.1.1. Slot specification

We will firstly show the entire final slot specification as follows, and then explain the details and design thinking. We also attach a table 2.2 which explains the details about every fragment in slots. The explanation and timing calculation are based on the results of [2]. Data about slot specification is also acquired from [2], but some changes are made.

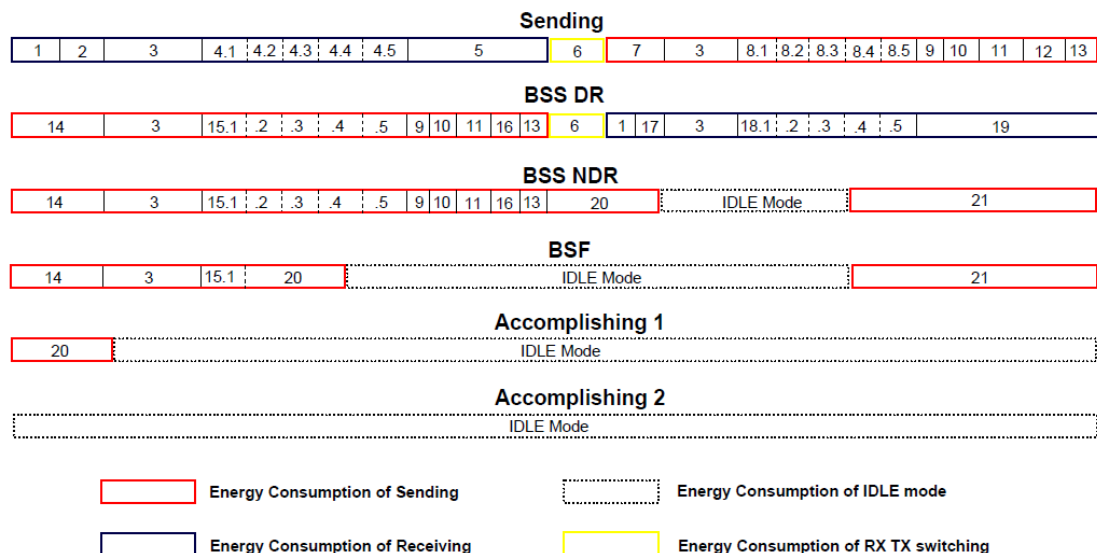


Figure 2.1: Slot specification of basic algorithm. In the figure, BSS DR stands for Beacon Sense Success & Data Receiver. BSS NDR stands for Beacon Sense Success & Not Data Receiver. BSF stands for Beacon Sense Failure.

In order to understand table 2.2, we have to firstly specify different types of payload. There are three types of payload, which are showed in table 2.1.

Payload types	Explanation	Number of Bytes
SBI	Slot Budget Information.	8
SRI	Slot Remains Information.	7
TSA	TX Success Acknowledgement	3

Table 2.1: Explanation of three payload types

Information which SBI payload carries is used for informing the children nodes of the available slots. 8-byte room is required to take some important parameters, such as Payload type (1 byte), myID (1 byte), myHop (1 byte), recID (1 byte), slotBegin (2 bytes), slotEnd (2 bytes) [3].

Information which SRI payload carries is employed for reporting the taken slots and the remaining slots to the parent node. 7-byte room is required to take parameters, such as payload type (1 byte), myID (1 byte), recID (1 byte), myBegin (2 bytes), childBegin (2 bytes) [3].

Different from SBI payload and SRI payload, TSA has a function of successful receiving of transmission. Sending nodes must wait for this small confirmation packet whenever it sends a SBI or SRI packet. Because it only plays a role of guarantee of receiving, only 3-byte room is need, where parameters, such as payload type (1 byte), myID (1 byte), recID (1 byte), are included.

At last, in order to understand the explanation in table 2.2, some definitions are need.

d_{SBI} Number of bytes in the SBI payload

d_{SRI} Number of bytes in the SRI payload

d_{TSA} Number of bytes in the TSA payload

No.	Explanation for each fragments in one slot	Timing Calculation (μs)
1	Packet generation and software updating	23
2	SBI payload and length information sent from MSP430 to CC2420 radio transmit buffer via SPI	$17 + 3 \times (d_{SBI} + 1)$
3	Radio calibration	192
4	Transmission of SBI (Slot Budget Information)	
4.1	Preamble	10×32
4.2	SFD (Start of Frame Delimiter), used to guarantee receiver's successful reception of the packet	2×32
4.3	Length	32

4.4	Payload	$d_{SBI} \times 32$
4.5	CRC	2×32
5	Waiting for receiver reading SBI information from CC2420 RTB (radio transmit buffer) to MSP430 via SPI	$= t_9 + t_{10} + t_{11} + t_{16} + t_{13}$ $= 8 + 54 + (17 + 3) +$ $(17 + 3 \times (d_{SBI} + 2)) + 8$
6	Mode switching between TX and RX	8
7	Waiting for TSA (TX Success Acknowledgment) sender finishing tasks	$= t_1 + t_{17} = 23 +$ $(17 + 3 \times (d_{TSA} + 1))$
8	Reception of TSA	
8.1	Preamble	4×32
8.2	SFD	2×32
8.3	Length	32
8.4	Payload	$d_{TSA} \times 32$
8.5	CRC	2×32
9	Notification of packet arrival	8
10	Communication stack overhead	54
11	Reading length information from CC2420 to MSP430 via SPI	$17 + 3$
12	Reading payload and CRC information from CC2420 to MSP430 via SPI	$17 + 3 \times (d_{TSA} + 2)$
13	Software updating	8
14	Waiting for SBI sender finishing task 1,2	$= t_1 + t_2 = 23 + 17 + 3 \times (d_{SBI} + 1)$
15	Reception of SBI	
15.1	Preamble (Carrier Sense)	10×32
15.2	SFD	2×32
15.3	Length	32
15.4	Payload	$d_{SBI} \times 32$
15.5	CRC	2×32
16	Receiving SBI payload and CRC information from RTB to MSP430 via SPI	$17 + 3 \times (d_{SBI} + 2)$
17	TSA payload and length information sent from MSP430 to CC2420 RTB via SPI	$17 + 3 \times (d_{TSA} + 1)$
18	Transmission of TSA	
18.1	Preamble	4×32
18.2	SFD	2×32
18.3	Length	32
18.4	Payload	$d_{TSA} \times 32$
18.5	CRC	2×32

19	Waiting for receiver reading TSA information from CC2420 RTB to MSP430 via SPI	$= t_9 + t_{10} + t_{11} + t_{12} + t_{13}$ $= 8 + 54 + (17 + 3) +$ $(17 + 3 \times (d_{TSA} + 2)) + 8$
20	Mode switching from TX,RX to IDLE mode	50
21	Mode switching from IDLE to TX, RX	192

Table 2.2: Radio-on time of fragments in different slots types. The above values are acquired from the specifications of TMOTE SKY, which contains radio CC2420 and processor MSP430.

2.1.2. Sending and Receiving

Either sending or receiving slot consists of three basic blocks. They are basic sending block, basic receiving block and switching (from RX to TX, or vice versa) block.

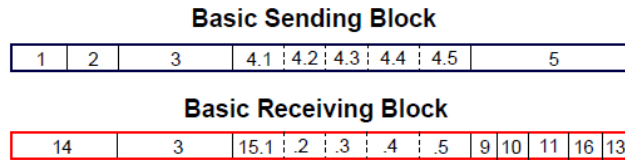


Figure 2.2: Basic Sending Block and Basic Receiving Block

As we can see from figure 2.2, the basic sending block in the sending slot includes fragments 1,2,3,4, and 5. From [2] we know that every piece of information to be sent has to be firstly sent from MSP430 to radio software stack. This information exchange is performed through SPI interface. The transmission via SPI will bring some delay. No.2 in table 2.2 shows the duration of this procedure, which depends on the amount of data d_{SBI} .

No.4 in table 2.2 shows the transmission duration of different information. “Preamble” is indispensable, because by transmitting a long signal, the sending node can tell nodes in the certain range that some information will be sent after the preamble section. The receiving nodes can decide whether it will keep radio working by performing Carrier Sense. Apart from “Payload”, some overhead, such as “SFD”, “Length”, “CRC”, have also to be transmitted in order to guarantee the correctness of transmission.

No.5 is necessary because it takes some time for the receiver to transfer the data from radio software stack to MSP430, which is the same procedure but with reversed direction as the No.2, therefore the sender has to wait until receiver has properly executed all the information.

Similarly, basic receiving block includes fragments No.14, 3, 15, 9, 10, 11, 16, 13. The receiver has to wait for the sender for its information transfer from MSP430 to radio software stack. Then

it receives the actual signals, and finally sends the signal back to the MSP430.

We notice that in figure 2.1, every sending and BSS DR slot pair has two basic sending-receiving blocks. Nearly every fragment of two basic sending-receiving blocks is the same but the ones related to payload.

2.1.3. Transmission Range

In the actual scenario, the power of signal decreases when the transmission distance increases. Thus, out of certain range, the transmitted signal can not be properly received. In order to simulate this scenario, we assume here that only the nodes one-hop away from the sender can hear the signal. As described in figure 2.3, when node 1 is sending, only nodes 0, 2, 3, 4 can hear the signals.

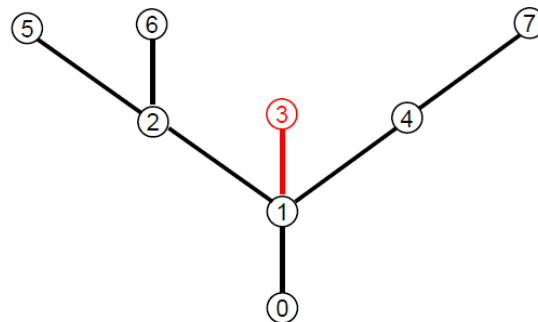


Figure 2.3: Transmission range explanation

When a node is sending, say node 1, to the receiver, say node 3, all nodes apart from node 1 can be divided into 3 groups according to different receiving effect.

The first group, only node 3, is the receiver, who executes the slot type 2. It does, at the beginning, fragment 15.1, the carrier sense. Because node 3 is in the transmission range of node 1, the signal arriving at the node 3 is strong enough for a successful carrier sense. Then node 3 follows the rest fragments to receive the information aimed at it. As it knows that it is the actual recipient of the packet, it continues doing TSA. The slot type executed by the first group is called BSS DR (Beacon Sense Success & Data Receiver).

The second group includes node 0, 2, and 4, who are also in the transmission range of node 1, but they are not the recipient of the packet. Similar to nodes in group 1, these nodes have to execute a basic receiving block as well, because all the nodes in the transmission range can sense the beacon and afterwards have to receive the information, though information is possibly not for it. However, after finishing the basic receiving block, group 2 nodes will find out that the information is actually not for them. Then they can sleep in the rest of slot. What has to be mentioned here is that both the radio turning on and turning off need some time. That's why fragment 20 and 21 is added in the slot. The slot type used by the second type is defined as BSS NDR (Beacon Sense Success

& Not Data Receiver).

The third group includes the remaining nodes, i.e. are nodes 5, 6, and 7. They are out of the transmission range. So after the failure of carrier sensing, they will get knowledge that the incoming packet is still far from them and can sleep in the rest of the slot. This slot type is called BSF (Beacon Sense Failure).

2.1.4. Radio on, radio off and switching

We notice that in the slot specification there are four different modes with different energy consumption. These modes are Sending, Receiving, RX/TX switching and IDLE mode. As we have assumed before, the energy consumption of first three modes are equal, while the fourth one is negligible.

On account of existence of different modes, the transitional fragment should be added between the different modes. For example, between Sending and IDLE a radio turn off fragment is needed; between IDLE and Sending a radio turn on fragment is needed; Between Sending and Receiving a switching fragment is needed as well.

In order to determine whether we need a transitional fragment at the beginning or at the end of the slot, we have to collect knowledge about the previous slots and the coming slots. The state machine chart for the basic algorithm, which clearly describes the sequence of slots, is shown in figure 2.4.

Fragment no.6 is required between basic sending block and basic receiving block. Fragment no.6 plays a role as mode switching. But we can see from figure 2.4:

- After slot Sending, only BSS NDR, BSF, or Accomp. 1 follow. And all of them begin with receiving mode. So there is no need to add an additional switching fragment at the end of the Sending slot.
- After slot BSS DR, only Sending follows. BSS DR ends with sending mode, but Sending begins with sending mode as well. There is thus no need to add transitional fragment.

For radio on and radio off fragments, both in BSF and in BSS NDR, we have to turn the radio on at the end of the slot and turn the radio off after finishing tasks in the slot.

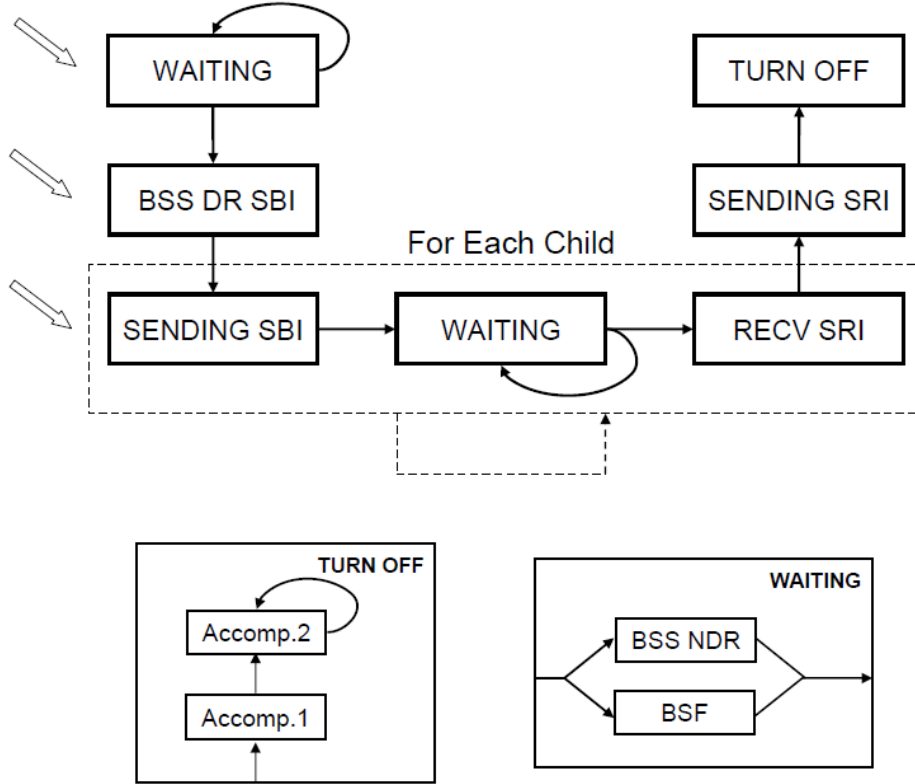


Figure 2.4: State machine chart of basic algorithm. The three hollow arrows in the left par of the figure indicate the three possible starting states. The states “WAITING” and “TURN OFF”, they each include two substates, which are showed in the two blocks in the lower part of the figure.

2.1.5. Radio-on time calculation

Now, we can calculate the radio-on time for each slot type.

The radio-on time of slot Sending is

$$t_{Sending} = \sum_{k=1}^{13} t_k + t_3$$

The value of t_k can be found in table 2.2.

The radio-on time of other slots are as following:

$$t_{BSSDR} = \sum_{k=9}^{11} t_k + \sum_{k=13}^{19} t_k + 2 \times t_3 + t_6 + t_1$$

$$t_{BSSNDR} = \sum_{k=9}^{11} t_k + \sum_{k=13}^{15} t_k + t_3 + t_{20} + t_{21}$$

$$t_{BSF} = t_3 + t_{14} + t_{15.1} + t_{20} + t_{21}$$

$$t_{A1} = t_{20}$$

$$t_{A2} = 0$$

2.1.6. Time sequence diagram

Time sequence diagram provides information about how many slots of every type one node goes through, and then the duty cycle of a node can be calculated.

Time sequence diagram is constructed by using three rounds, i.e. slot types are divided into three groups. We firstly arrange the Sending slots and BSS DR slots for each node. In the second round we will distribute Accomplishing 1 and Accomplishing 2 slots. Finally BSS NDR and BSF slots are allocated.

In the first round, recursion is used. What we describe below is a recursive function, which means it will call itself. Details about first round are showed below.

- Step 1 The current node is allocated. At the beginning, the current node is the sink node.
- Step 2 Send SBI packet to the kth child. The ongoing slot of sender is allocated as Sending, while the one of receiver is BSS DR.
- Step 3 The parameter describing the current state is increased by one.
- Step 4 Call the recursion function (Call itself, i.e. go to step 1)
- Step 5 Send SRI packet back from the kth child to the current node. The ongoing slot of sender as Sending, while receiver BSS DR.
(Do the steps 2 to 5 until all children have been traversed)
- Step 6 Check the recursion ending condition

In the second round, we check, for the time sequence of each node, where the Sending slot of SRI is. Since SRI Sending signals the end of all the tasks of a node, the node will turn into sleeping mode after sending the SRI. So we can check from the end to the beginning of a node phase, where the first sending is, which will be the SRI sending. Then the first slot after Sending should be Accomplishing 1, and the others should be Accomplishing 2.

In the third round, BSF and BSS NDR slots are allocated. Before this round, only some of slots are allocated with specific slot types. So for a certain node, it has some slots specified "Sending" or "BSS DR". It has some empty slots as well. In this round, we would fill the rest slots with "BSF" and "BSS NDR". We finish this by checking every slot of each node and then allocating this slot with either "BSS NDR" or "BSF".

At this round, the slots which are taken into consideration are the ones which are still not allocated anything yet. We will check every slot from the beginning to the end. In each iteration, we consider nodes whose slots are still unoccupied. If one is in the transmission range, then BSS NDR is allocated. Otherwise, BSF allocated.

Given the time sequence diagram, duty cycle of each node and the average duty cycle can be calculated with the following expressions.

$$DC_k = \frac{n_{Sending} t_{Sending} + n_{BSSDR} t_{BSSDR} + n_{BSSNDR} t_{BSSNDR} + n_{BSF} t_{BSF} + t_{A1}}{T_p}$$

$$\overline{DC} = \frac{1}{N} \sum_{k=0}^N DC_k$$

Where $n_{Sending}$ is the number of “Sending” slots one certain node has. $n_{BSSDR}, n_{BSSNDR}, n_{BSF}$ are the number of “BSS DR”, “BSS NDR”, “BSF” slots, respectively. $t_{Sending}$ is the radio-on time of “Sending” slot, $t_{BSSDR}, t_{BSSNDR}, t_{A1}$ are the radio-on time of “BSS DR”, “BSS NDR”, “A1” respectively. N is the total number of nodes.

The next value we require is the phase length, which is the entire time needed by the scheduling propagation phase. It can be calculated by using the following expression:

$$T_p = t_{slot} \cdot N_{slotnum} = t_{Sending} \cdot N_{slotnum}$$

In the expression, T_p is the time of the scheduling propagation phase; t_{slot} is the time of a slot;

$N_{slotnum}$ is the number of slots in the whole phase. Because every slot in the phase has equal length, the whole phase length can be expressed by multiplying of slot length and slot number. What's more, the every slot type has the same length as well, so the length radio-on time of the slot “Sending” can be used to substitute the slot length. Because it is one of slot types that need more time (together with BSS DR)

At last, the average radio-on time, can be acquired:

$$\overline{T_{radio-on}} = \overline{DC} \cdot T_p$$

A sample time sequence diagram looks as the following.

SLOT	1	2	3	4	5	6	7	8	9	10	11	12
Node6	4	4	4	4	4	4	4	4	4	2	1	*
Node5	4	3	4	2	1	*	-	-	-	-	-	-
Node4	4	2	1	*	-	-	-	-	-	-	-	-
Node3	3	4	4	4	4	4	3	4	2	1	2	1
Node2	3	4	4	4	4	4	2	1	*	-	-	-
Node1	2	1	2	1	2	1	*	-	-	-	-	-
Node0	1	3	4	3	4	2	1	2	1	3	4	2

- 1 denotes “Sending”
- 2 denotes “BSS DR”
- 3 denotes “BSS NDR”
- 4 denotes “BSF”
- * denotes “Accomplishing 1”
- denotes “Accomplishing 2”

The result is based on the following topology.

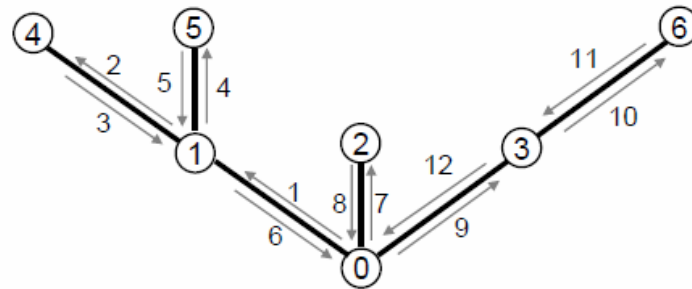


Figure 2.5: The topology used to calculate the sample result. In the figure, numbers along each line represent the order in which links are traversed. i.e. the link with 1 is traversed first, while the link with 12 is traversed at last.

2.2. Implementation of Extended Algorithm

2.2.1. Basic Idea

The extended algorithm is based on the idea that if a node fails to sense the carrier, it will not receive a slot budget (SBI), nor a remaining budget (SRI) in the next slot. Therefore it can safely go to sleep for one slot.

Proof. We prove the idea in two situations. First, a certain node receives BSS NDR from its parent. Second, one certain node receives BSS NDR from its child.

As an example for the first case, in figure 2.5 supposing node 0 is sending packet to node 1. We focus on the node 2. It will sense the carrier and get a BSS NDR from its parent node 0. In this case, node 1 sends the packet further to its children in the next slot. Even if node 1 hadn't any child, one more slot will be taken for it to send back to node. Thus node 2 can safely sleep in the next slot.

As an example for the second case, in figure 2.5 supposing that node 1 is sending to node 4. And node 0 can get BSS NDR from its child, node 1. Because in the next slot, node 1 has to receive a packet from node 4, node 0 can safely sleep in the coming slot #

2.2.2. Adjustment of Slot Specification

As mentioned in the idea, a new slot type, sleeping slot, is needed. The adjusted slot specification is showed below[2].

In figure 2.6, we can see that a single fragment 21 is included in the newly added Sleeping slot. What's more, changes are made to BSS NDR, the fragment 21 is removed. In order to understand the changes, we have to refer to the state machine chart of extended algorithm. The state machine chart indicates that after BSS NDR, Sleeping comes. So the fragment 21 which is originally allocated to the BSS NDR can be moved to the end of Sleeping slot. In addition, after Sleeping slot, BSS DR or BSF or BSS NDR come. The beginning parts of all three slot type are all receiving mode, so a transitional fragment is not needed.

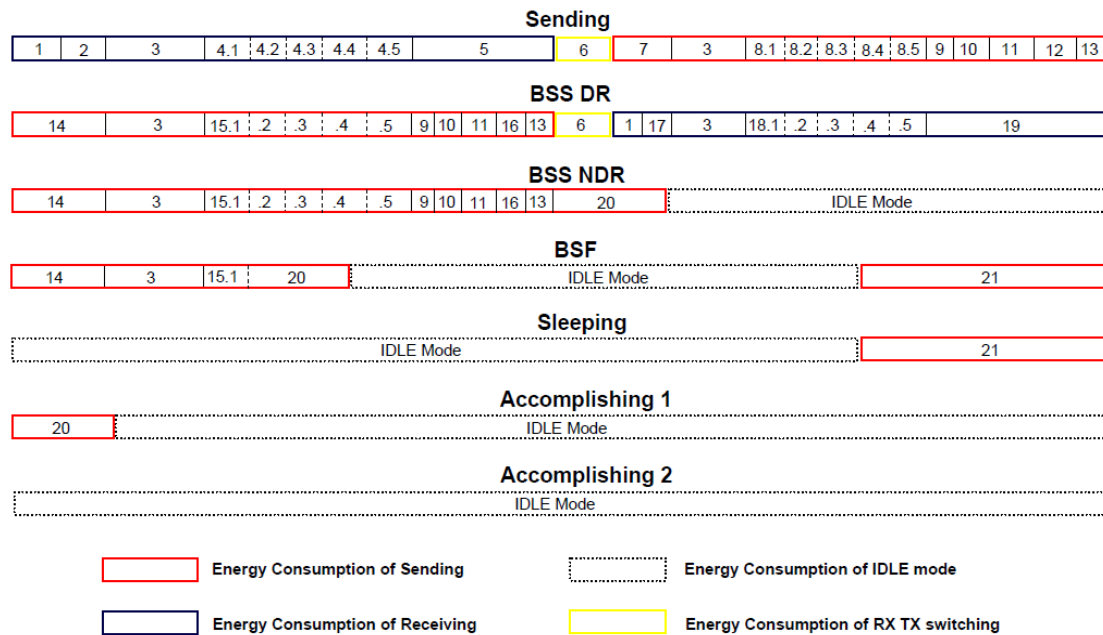


Figure 2.6: Slot specification of extended algorithm.

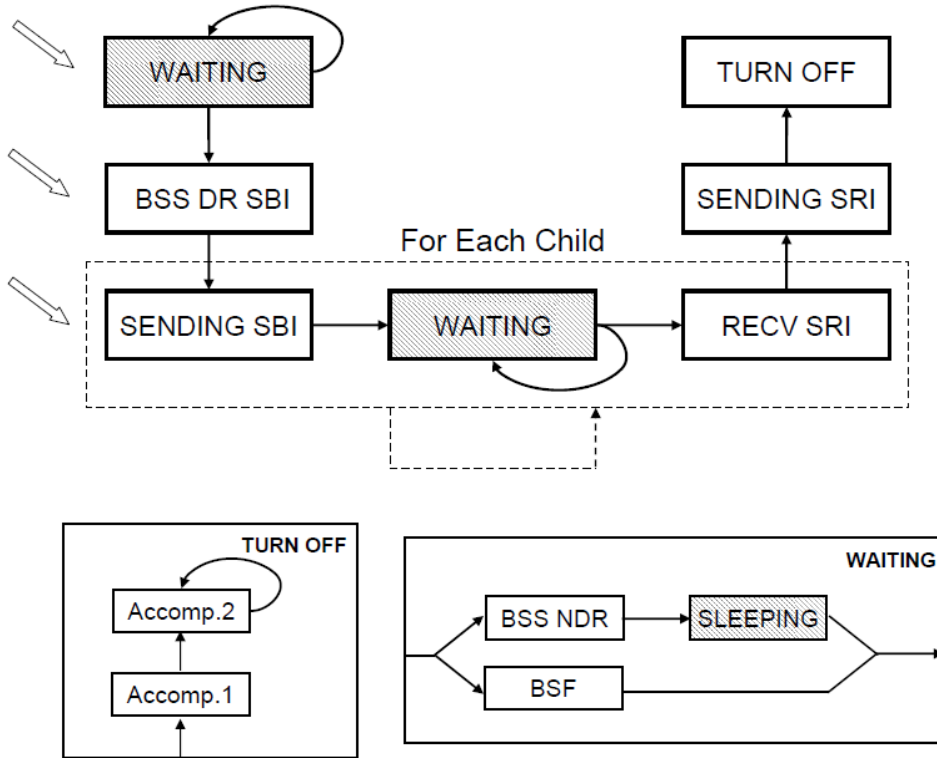


Figure 2.7: State machine chart of extended algorithm. The three hollow arrows in the left part of the figure indicate the three possible starting states. The states “WAITING” and “TURN OFF”, they each include several substates, which are shown in the two blocks in the lower part of the figure. The shadowed blocks, such as “WAITING”, “SLEEPING”, stress that some blocks (“WAITING”) are changed, or some new blocks (like “SLEEPING”) are added

2.2.3. Radio-on time calculation.

Because of adjustment of BSS NDR slot, the expression of it should be changed accordingly.

$$t_{BSSNDR} = \sum_{k=9}^{11} t_k + \sum_{k=13}^{15} t_k + t_3 + t_{20}$$

And the radio-on time of Sleeping slot is

$$t_{Sleeping} = t_{21}$$

2.2.4. Time sequence diagram

The only change to the programming is in the third round, i.e. when BSF and BSS NDR are allocated. The new sleeping slot is also allocated during the third round. If a certain slot of one

node is allocated BSS NDR, then the next slot of this slot should be allocated as Sleeping slot.

According to the following expressions, we can calculate the duty cycle.

$$DC_k = \frac{1}{T_p} (n_{Sending} t_{Sending} + n_{BSSDR} t_{BSSDR} + n_{BSSNDR} t_{BSSNDR} + n_{BSF} t_{BSF} + n_{Sleeping} t_{Sleeping} + t_{A1})$$

$$\overline{DC} = \frac{1}{N} \sum_{k=0}^N DC_k$$

$$T_p = t_{Sending} \cdot N_{slotnum}$$

$$\overline{T_{radio-on}} = \overline{DC} \cdot T_p$$

Here, a sample output based on the same input as algorithm 1 is showed below.

SLOT	1	2	3	4	5	6	7	8	9	10	11	12
Node6	4	4	4	4	4	4	4	4	4	2	1	*
Node5	4	3	0	2	1	*	-	-	-	-	-	-
Node4	4	2	1	*	-	-	-	-	-	-	-	-
Node3	3	0	4	4	4	4	3	0	2	1	2	1
Node2	3	0	4	4	4	4	2	1	*	-	-	-
Node1	2	1	2	1	2	1	*	-	-	-	-	-
Node0	1	3	0	3	0	2	1	2	1	3	0	2

0 denotes "Sleeping"

1 denotes "Sending"

2 denotes "BSS DR"

3 denotes "BSS NDR"

4 denotes "BSF"

* denotes "Accomplishing 1"

- denotes "Accomplishing 2"

2.3. Implementation of Odd-even Slot Algorithm

2.3.1. Basic Idea

The odd-even slot algorithm is based on the rule that SBI can be only transmitted in odd slots, and SRI only in even slots.

If this restriction upon the scheduling propagation is fixed, then following modifications apply.

- Nodes who have not received any SBI can sleep in all even slots.

According to the rule, SBI can be only received in odd slots, thus nodes waiting for their SBI do not have to check even slots.

- Nodes who have sent out SBI and are still waiting for the feedback, SRI, have no need to check the odd slots.

According to the rule, SRI can only arrive in even slots, therefore nodes waiting for SRI do not have to check odd slots.

2.3.2. Adjustment of slot specification

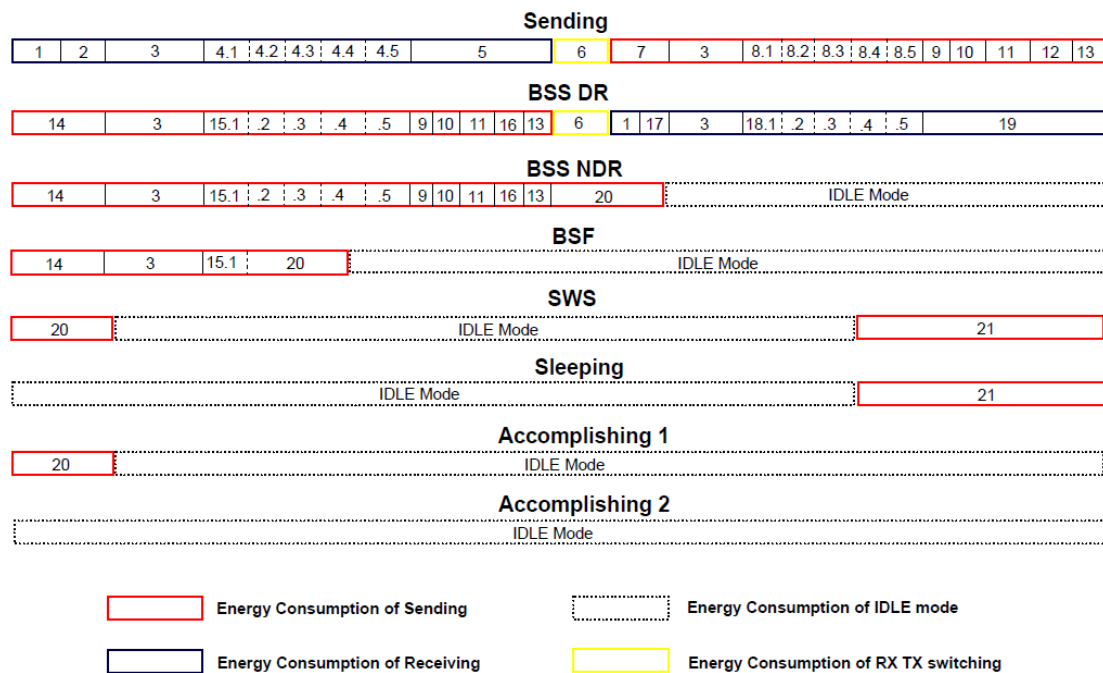


Figure 2.8: Slot specification of odd-even slot algorithm [2].

As we notice from the above slot specification,

- A new slot SWS(Sending Waiting Slot) is added.

According to the basic idea, SBI only in odd slots, and SRI only in even slots. There could possibly be a case when a node wants to send SBI to a child, immediately after it has received a SBI packet from the parent. In this case, since two consequent packets are both SBI, the node has to wait for one more slot before sending its SBI. Then this transitional SWS slot must be added.

However, the node knows it will send SBI in next odd slot. So in order to save energy, it can turn off the radio after receiving the first SBI and turn it on before sensing the second SBI.

- The fragment 21 in both BSS NDR and BSF is removed: They are always followed by a “Sleeping” slot.

As usual, we first draw the state machine chart of the odd-even slot algorithm, and then explain the adjustments.

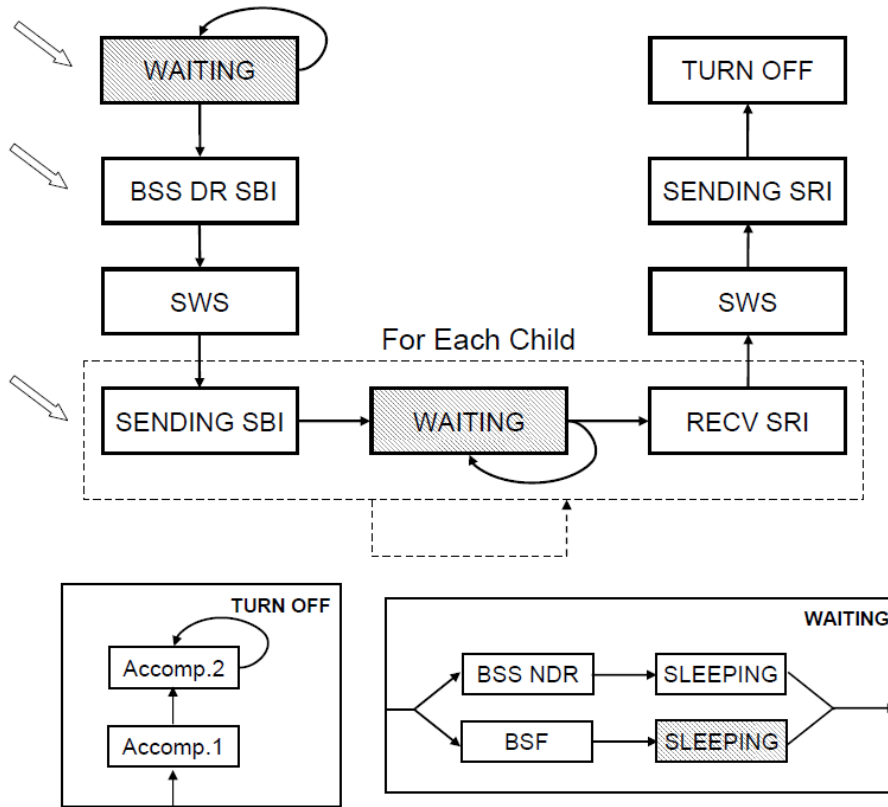


Figure 2.9: State Machine Chart of odd-even slot algorithm. The three hollow arrows in the left part of the figure indicate the three possible starting states. The states “WAITING” and “TURN OFF”, they each include several substates, which are shown in the two blocks in the lower part of the figure. The shadowed blocks, such as “WAITING”, “SLEEPING”, stress that some blocks (“WAITING”) are changed, or some new blocks (like “SLEEPING”) are added.

We can see from figure 2.9 that after BSS NDR or BSF, it is Sleeping that always follows. We don’t need to turn on the radio at the end of BSS NDR or BSF. We can turn the radio of a certain node off until the end phase of Sleeping and turn on the radio at the end.

From the extended algorithm, we know that the slot after BSS NDR is a “Sleeping” slot. What’s more, whatever a node is waiting for, either SBI or SRI, it only has to check the odd or even slot, so after BSS NDR or BSF, a Sleeping always follows.

2.3.3. Radio-on time calculation.

Because of adjustment of slot specification, the expression of radio-on time of BSF slot has to be

changed accordingly.

$$t_{BSF} = t_3 + t_{14} + t_{15.1} + t_{20}$$

And the radio-on time of SWS slot is

$$t_{SWS} = t_{20} + t_{21}$$

2.3.4. Time sequence diagram

In the first round, recursion is used. What we describe below is a recursive function, which means it will call itself. Details about first round are showed below.

- Step 1 The current node is allocated. At the beginning, the current node is the sink node.
- Step 2 If the current slot is not odd slot, then wait for one more slot, the current slot assigned SWS
- Step 2' Send SBI packet to the kth child. The ongoing slot of sender is allocated as Sending, while the one of receiver is BSS DR.
- Step 3 The Parameter describing the current state is increased by one.
- Step 4 Call the recursion function (Call itself, i.e. go to step 1)
- Step 5 If the current slot is not even slot, then wait for one more slot, the present slot assigned SWS
- Step 5' Send SRI packet back from the kth child to the current node. The ongoing slot of sender as Sending, while receiver BSS DR.
(Do the step 2 to step 5 until all N children have been traversed)
- Step 6 Check the recursion ending condition

More changes to the programming should be made in the third round.

- If a certain slot of one node is allocated BSS NDR, then the next slot of this slot should be allocated as Sleeping slot.
- If a certain slot of one node is allocated BSF, then the next slot of this slot should be allocated as Sleeping slot.

$$DC_k = \frac{1}{T_p} (n_{Sending} t_{Sending} + n_{BSSDR} t_{BSSDR} + n_{SWS} t_{SWS} + n_{BSSNDR} t_{BSSNDR} + n_{BSF} t_{BSF} + n_{Sleeping} t_{Sleeping} + t_{A1})$$

Sample output on the same topology as for basic algorithm and extended algorithm.

SLOT	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Node6	4	0	4	0	4	0	4	0	4	0	4	0	2	1	*	-
Node5	4	0	3	0	2	1	*	-	-	-	-	-	-	-	-	-
Node4	4	0	2	1	*	-	-	-	-	-	-	-	-	-	-	-
Node3	3	0	4	0	4	0	4	0	3	0	2	5	1	2	5	1

Node2	3	0	4	0	4	0	4	0	2	1	*	-	-	-	-	-
Node1	2	5	1	2	1	2	5	1	*	-	-	-	-	-	-	-
Node0	1	4	0	4	0	4	0	2	1	2	1	4	0	4	0	2

0 denotes "Sleeping"

1 denotes "Sending"

2 denotes "BSS DR"

3 denotes "BSS NDR"

4 denotes "BSF"

5 denotes "SWS"

* denotes "Accomplishing 1"

- denotes "Accomplishing 2"

3. Link Failure

In the previous chapter, most important settings for three algorithms, such as slot specification, state machine chart, and time sequence diagram, are all based on an assumption that the possibility of success is always 100%, in other words, retransmission is redundant. However, in order to improve the preciseness of algorithm evaluation, retransmission caused by link failure is supposed to be considered.

In the following parts of this chapter, we will first clarify some definitions, such as time-out, retransmission number, and consequently explain the revised slot specifications and state machine charts.

3.1. Time-out

Thanks to the existence of CRC, used to check whether failure occurs during transmission, link failure can be found after each interaction between sender and receiver. What the sender needs to do after a detected failing transmission is to retransmission. There might a case that the receiver gets the message eventually after several times of retransmission, however an event can possibly happen that even after a huge quantity of retrying, receiver still can not receive it. So, as a part of the algorithm settings, we should set a time-out in order to limit the retransmission times, otherwise, the algorithm will cause the traversal to a deadlock because of infinite retransmission. Even if for the case of limited retransmission, a time-out is needed to lower the waste of power.

Time-out is defined as the maximum transmission number which means that Reattempt is not made after this number of transmission. For example, if the time-out is set as 4, after 4 times of failing sending attempts from the sender, the sender will choose another link and discard the present link.

Additionally, there is a constraint about the selection of time-out. The time-out can not be odd numbers. Because in algorithm 2, we set a rule that if a node is in BSS NDR slot, in the next slot it can sleep. Supposing that we set the MSN as 5, in this way, node 0 will transmit to node 1, as we can see from figure 3.1, after five times, the transmission fails at last, and then node 0 will turn to node 2 and send to it. When the node sends at first, node 2 is in BSS NDR, so it can sleep during second transmission of node 0. And therefore the node 2 is also in BSS NDR in the last transmission attempt of node 0. So the node 2 will wrongly sleep when node 0 sends to itself. According to the analysis, time-out has to be a even number, so we randomly choose a time-out as 4. Realistic time-out selection should be based on the physical layer specification of signal communication. Here the selection of 4 is for simplicity.

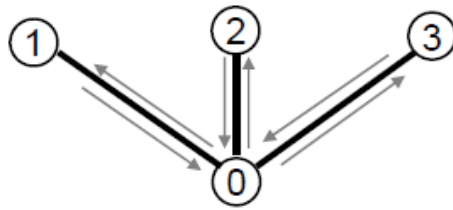


Figure 3.1: Explanation of constraints on the selection of time-out.

In the implementation of algorithms with link failure, we firstly randomly generate a link failure array constrained by an average transmission number, which is also the user input. Each value in this array represents the sending number in corresponding link during the tree traversal. Specifically speaking, if, for the link node 0 to node 1, 4 is assigned, then transmission is tried for 4 times. If -1 is assigned, then transmission is tried for time-out times (4 times), and this link is ignored.

In figure 3.2, two topologies are provided as examples for transmission number and time-out distribution. In the left case, as the traversal is going, transmission must be retried at each link according to the number given beside the up and down link line in figure 2.3. In the right case, because a time-out is assigned to the link from node 0 to node 1, then the whole sub-tree having node 1 as the top parent node is disposed of. The subsequent traversed node will be node 2.

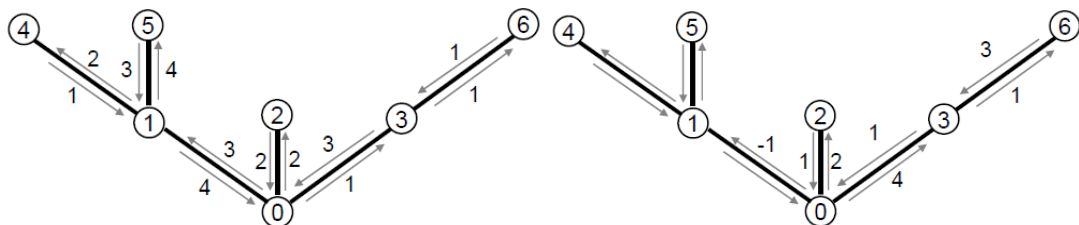


Figure 3.2: A case of topology with randomly distributed transmission number w/o time-out (left); a case of topology with time-out(right).

3.2. Revised slot specification and state machine chart

3.2.1. Basic Algorithm

Some minor variations can be noticed in both figure 3.3 and figure 3.4 compared to the original ones. In the state machine chart, owe to consideration of unstable links, the slot “Sending” will go back to itself, which brings about the corresponding changes to the slot specification, where the fragment 6 (a fragment for mode switching) is appended as the tail of both “Sending” slot and “BSS DR” slot, because a “Sending” or “BSS DR” precedes another “Sending” or “BSS DR”. If “Sending”, for example, repeats itself, then because of the mode difference of header and tail of

the “Sending” slot, a fragment 6 has to be added.

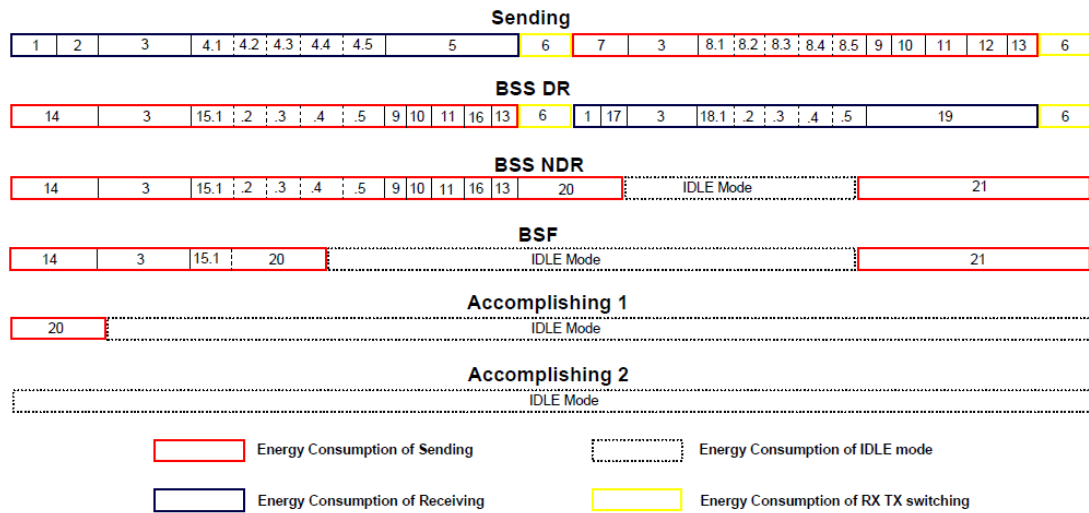


Figure 3.3: Slot specification for basic algorithm with link failure [2]

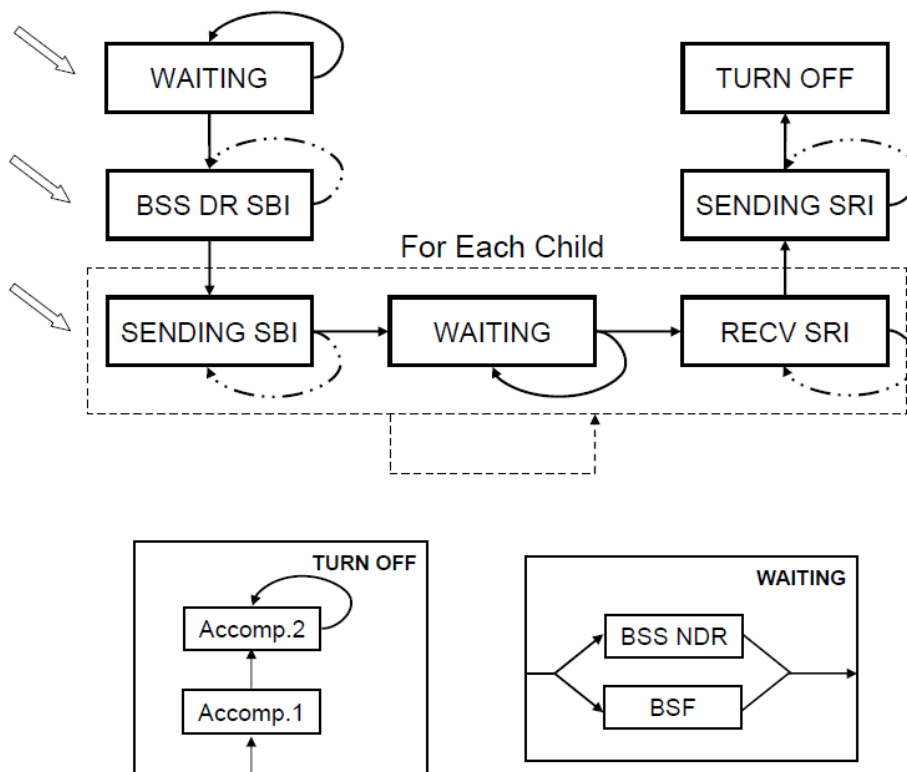


Figure 3.4: State machine chart for basic algorithm with link failure. The three hollow arrows in the left part of the figure indicate the three possible starting states. The states “WAITING” and “TURN OFF”, they each include several substates, which are showed in the two blocks in the lower part of the figure. The dashed curves, which are around “BSS DR SBI”, “SENDING SBI”, “SENDING SRI”, “RECV SRI”, in the figure show that these curves are new compared to the situation without link failure.

3.2.2. Extended Algorithm

Similar changes are made for the extended algorithm.

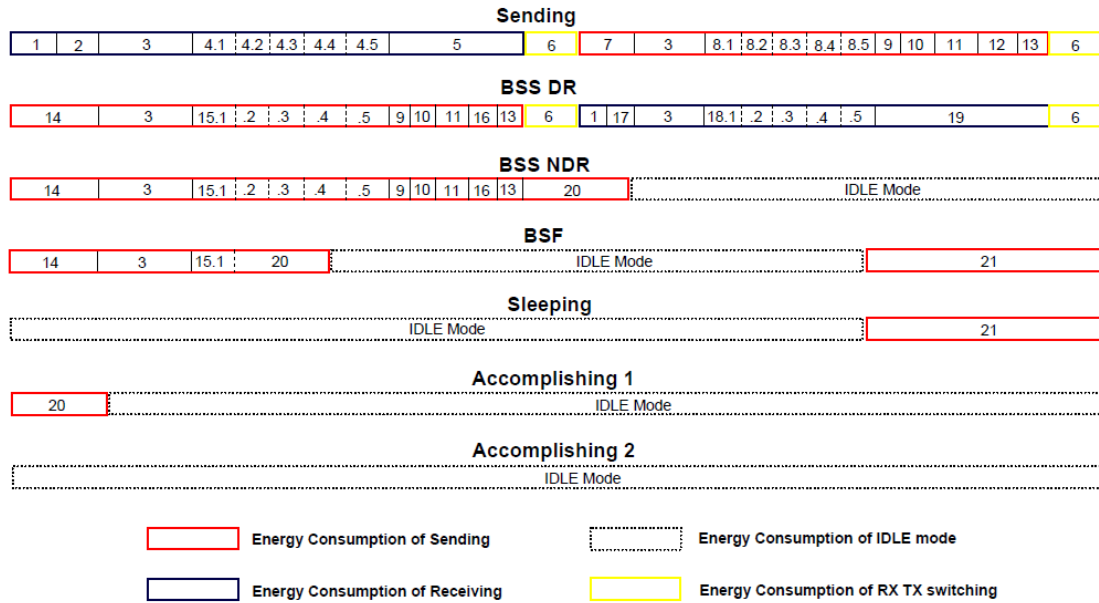


Figure 3.5: Slot specification for extended algorithm with link failure [2].

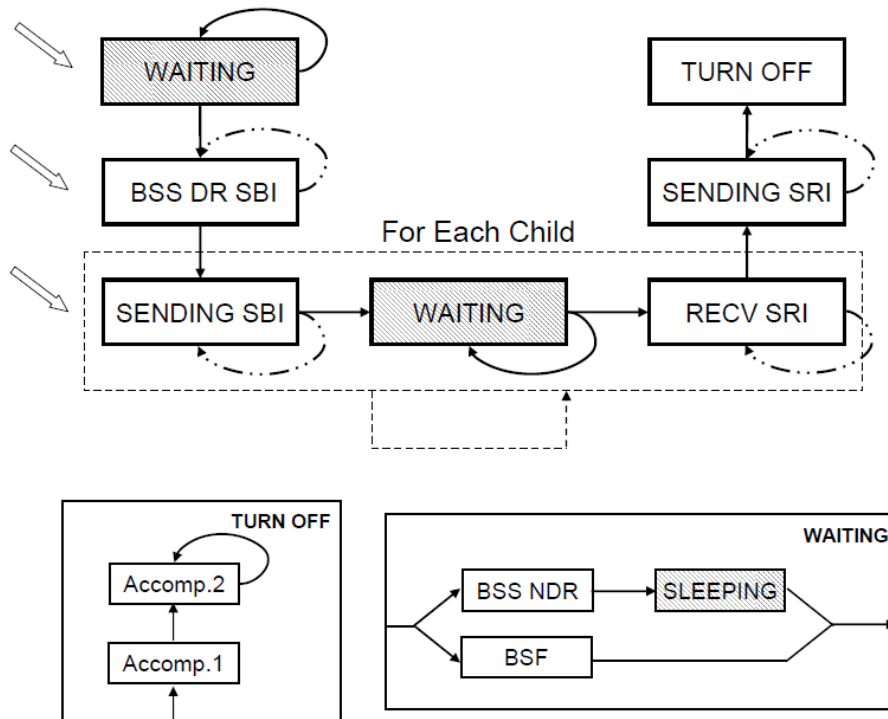


Figure 3.6: State machine chart for extended algorithm with link failure. The three hollow arrows in the left part of the figure indicate the three possible starting states. The states “WAITING” and “TURN OFF”, they each include several substates, which are showed in the two blocks in the lower part of the figure. The shadowed blocks, such as “WAITING”, “SLEEPING”, stress that some blocks (“WAITING”) are changed, or some new blocks (like “SLEEPING”) are added. The

dashed curves, which are around “BSS DR SBI”, “SENDING SBI”, “SENDING SRI”, “RECV SRI”, in the figure show that these curves are new compared to the situation without link failure.

3.2.3. Odd-even Slot Algorithm

Similar to the previous two algorithms, odd-even slot algorithm will also replicate the slot “Sending” and “BSS DR”. But different to the previous ones, in order to react the two slots, sender has to skip the subsequent slot and resend in the third slot. Because the type of next slot is different to the first one, the sender has to wait for the right slot type, which will occur in the third slot.

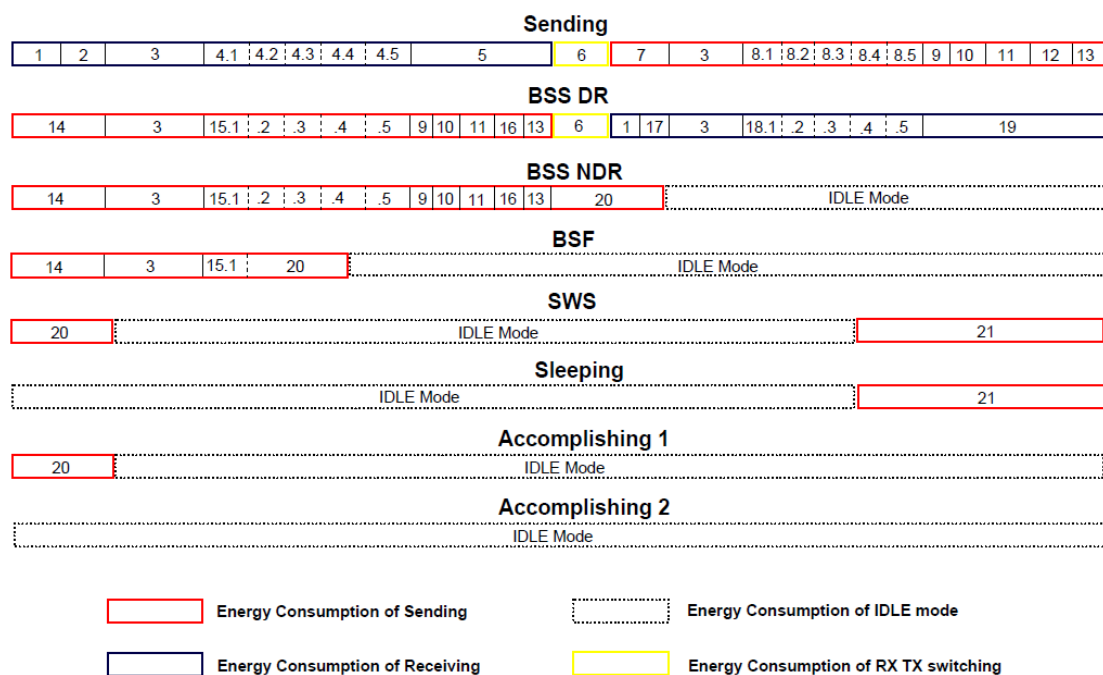


Figure 3.7: Slot specification for odd-even slot algorithm with link failure [2]

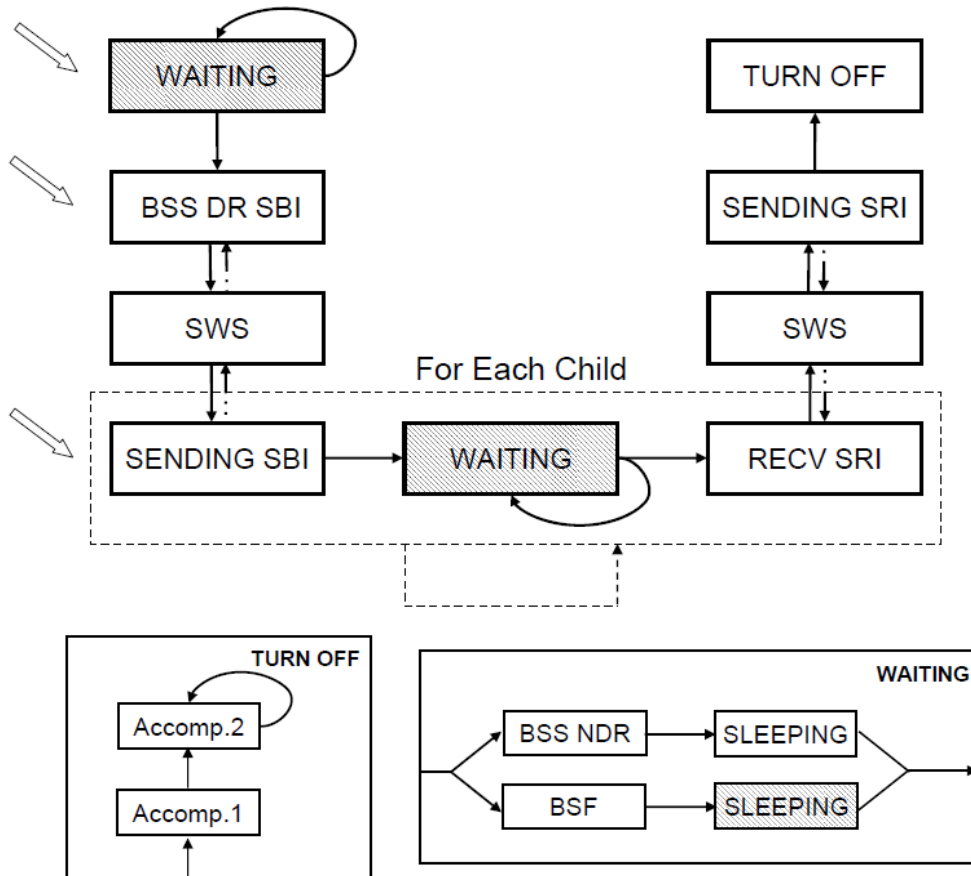


Figure 3.8: State machine chart for odd-even slot algorithm with link failure. The three hollow arrows in the left part of the figure indicate the three possible starting states. The states “WAITING” and “TURN OFF”, they each include several substates, which are shown in the two blocks in the lower part of the figure. The shadowed blocks, such as “WAITING”, “SLEEPING”, stress that some blocks (“WAITING”) are changed, or some new blocks (like “SLEEPING”) are added. The dashed lines, which are around “SWS”, in the figure show that these lines are new compared to the situation without link failure.

4. Results Analysis

4.1. Evaluation Metrics

4.1.1. Parent-children Ratio

In order to reflect how the input topology is constructed, we define the parent-children ratio as following:

$$\sigma = \frac{N_{parent}}{N_{child}}$$

which is the ratio of the parent number to children number. As in a tree the nodes in a certain topology must be child except the sink node, total children number should be $N-1$, where N is the total nodes number. Then the above expression can be changed

$$\sigma = \frac{N_{parent}}{N-1}$$

σ has a range of $[\frac{1}{N-1}, 1]$. The upper bound is hit when the parent number equals to $N-1$, which actually represents a pure sequence topology as showed in left part of figure 5.1. Except the leaf nodes, every node in the topology is a parent. The lower bound occurs when the parent number equals to 1, which is what the right part of the same figure shows. One parent node coordinates $N-1$ child nodes.

If the parent-children ratio is near the lower bound, then the parent number is low and the tree depth is small. The topology inherently contains larger parallel structures. If the parent-children ratio grows, the depth goes up, then more deep sequential structures can be constructed. Therefore the parent-children ratio reflects the amount of parallel and sequential structures in a certain topology.

4.1.2. Total Radio-on Time and Total Phase Length

In order to evaluate the performance of a certain algorithm, two metrics are used:

- Total radio-on time
- Total phase length

These two metrics evaluate an algorithm from different perspectives. Total phase length evaluates an algorithm from latency perspective. It can be computed by using the following expression,

$$T_p = t_{slot} \cdot N_{slotnum} = t_{Sending} \cdot N_{slotnum}$$

Where T_p is the time of the scheduling propagation phase; t_{slot} is the time of a slot; N_{slotum} is the number of slots in the whole phase; $t_{Sending}$ is the time of the sending slot. Because the sending slot is always the longest slot, the slot length corresponds to $t_{Sending}$.

Total radio-on time evaluates an algorithm from energy perspective. The total phase length can be calculated as

$$\overline{T_{radio-on}} = \overline{DC} \cdot T_p$$

Where, $\overline{T_{radio-on}}$ is the total radio-on time, \overline{DC} is average duty cycle, and T_p is total phase length which we can calculate from the above equation.

4.2. Evaluation of Algorithms w/o Link Failure

In chapter 2, given a certain topology, the total radio-on time of traversing the same topology is outputted. The results are given by using three algorithms. The goal of this chapter is to evaluate the performance of the three algorithms based on a particular topology.

As we can imagine, three algorithms perform with difference when executed in different topologies. Thus a comprehensive evaluation of the three algorithms must take the thorough variations of topologies into consideration. By evaluating topologies with different structure, we find out that the hierarchical structure might become a factor that influences the performance of algorithms. Hierarchical structure, specifically speaking, is related to the combination of parallel and sequence structure. The two structures are showed in figure 4.6. After trying some cases, we find out a rule that the odd-even slot algorithm does much better work than others in a topology containing more parallel structure, which is not yet verified and might be wrong.

Next, we will first propose a definition which describes whether a topology resembles more to sequence structure or more to parallel structure. Then we enumerate the all existent topologies and calculate this “parallel-sequence” factor and total radio-on time. And then study the relationship between the input and output.

4.2.1. Evaluation Results

Total Radio-on Time vs Parent-children Ratio

Figure 5.1 shows the relationship between total radio-on time and parent-children ratio, when total nodes number is 15. Three curves can be seen in the figure 5.1, the brightest curve represents the results of basic algorithm. The darkest one shows how the odd-even slot algorithm performs; and

the grey one is about the extended algorithm.

Y-axis is total radio-on time, which describes the latency of three algorithms. X-axis is parent-children ratio, which is a factor indicating topology variation. As discussed above, when the total nodes number is fixed, in this case 15, the parent-children ratio only has 14 possibilities, where parent number grows from 1 to 14. Thus 14 samples are plotted for each algorithm in figure 4.1.

An errorbar at each sample value shows the lower and upper bound and the sampled value is thus always located between the lower and upper bound. Even when the total nodes number and parent-children ratio are fixed, there are still a large amount of topology possibilities pointing to the same parent-children ratio. So for a certain parent-children ratio, we can get many total radio-on times. So what we do is to calculate the worse case, best case and the average case. The marker (including star, circle, square) in the figure is the average value, while the top and bottom of errorbar represents the worst case and best case respectively.

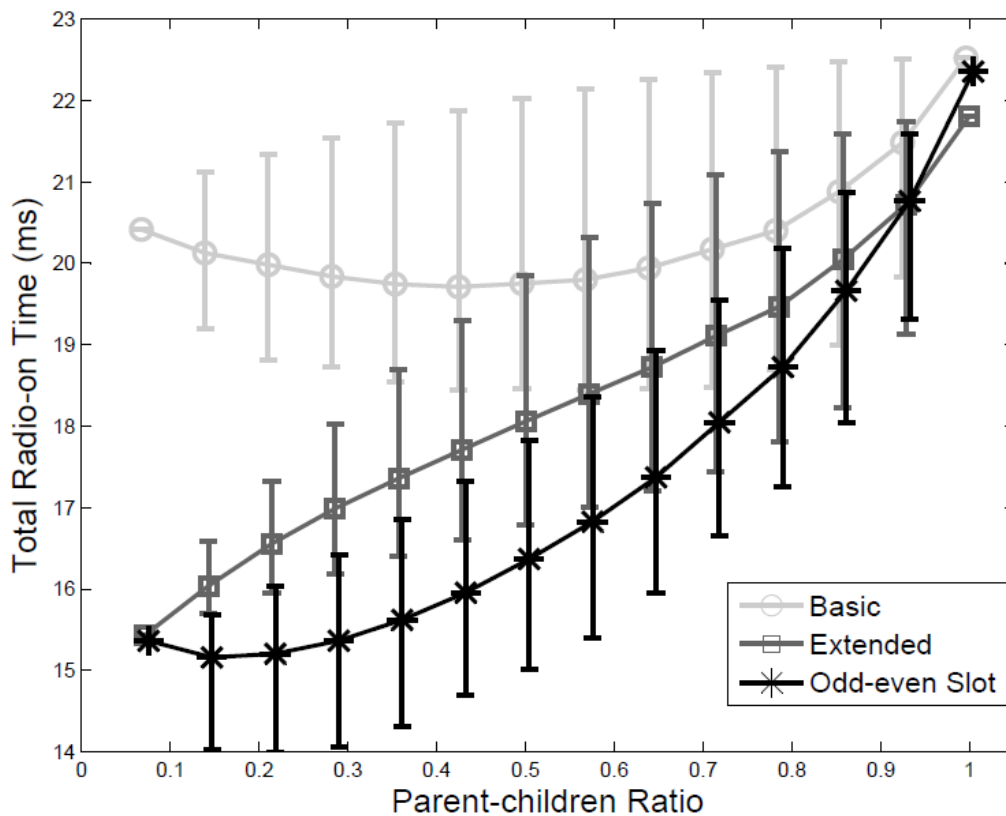


Figure 4.1: Relationship of total radio-on time against parent-children ratio (Total nodes number is 15)

Changing the total nodes number, we can get the similar results to the previous one, as we can see in the figure 4.2 and figure 4.3.

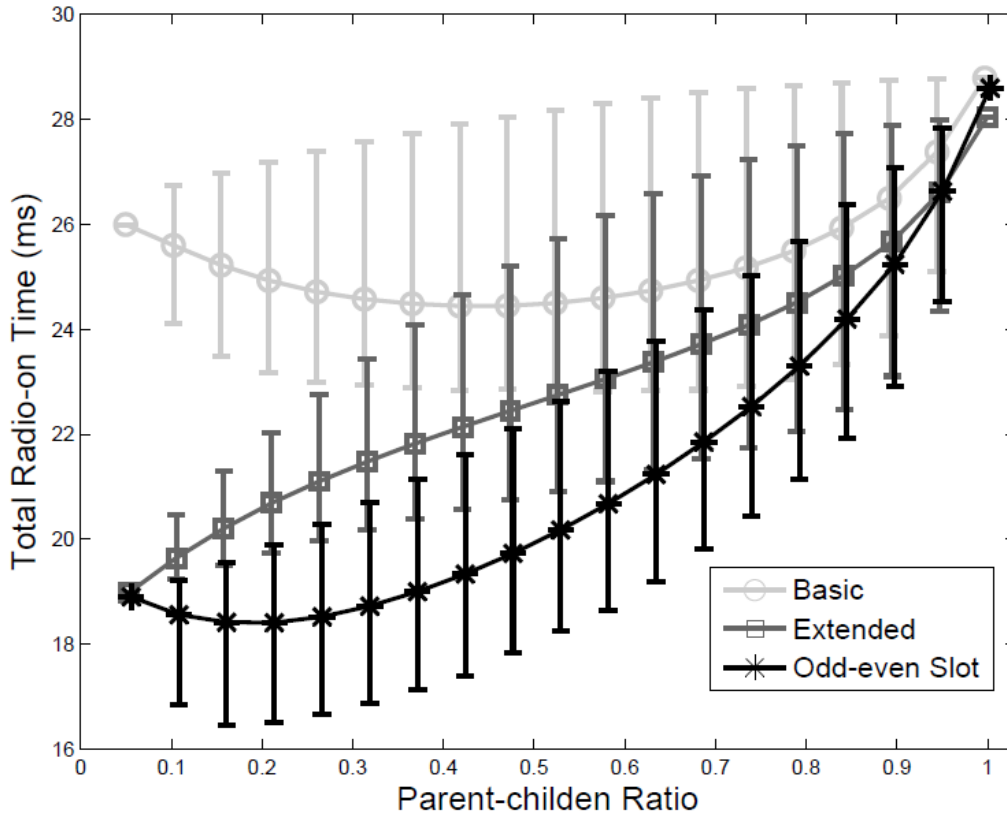


Figure 4.2: Relationship of total radio-on time against parent-children ratio (Total nodes number is 20)

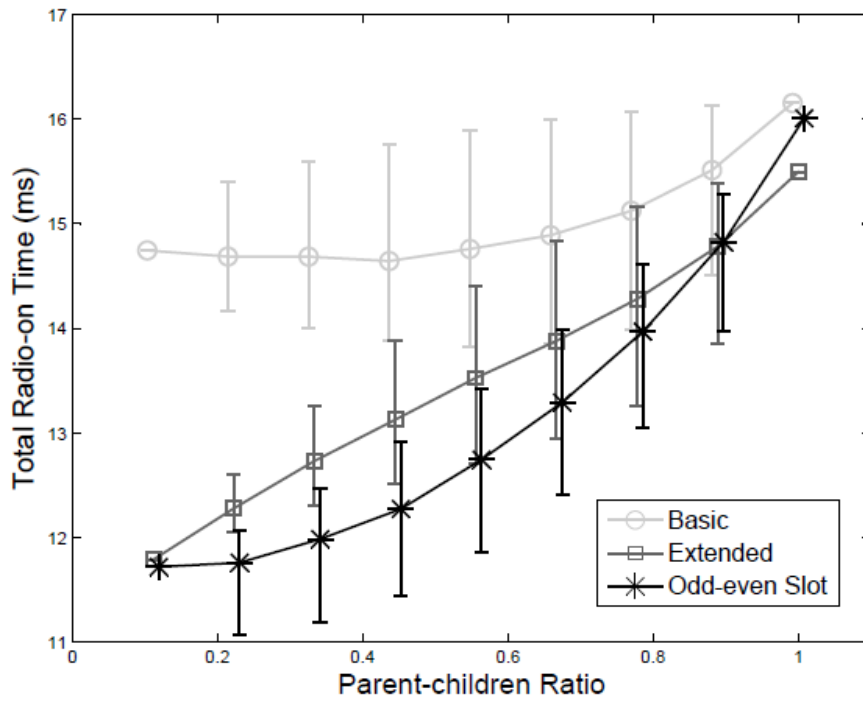


Figure 4.3: Relationship of total radio-on time against parent-children ratio (Total nodes number is 10)

Total Phase Length vs Parent-children Ratio

Figure 5.4 shows the relationship between total phase length and parent-children ratio. 15 nodes are deployed. Compared to the previous graph, there is no errorbar in this graph, which shows that traversal of topologies corresponding to the same parent-children ratio costs the same phase length. What's more, the figure below shows two facts: when using basic algorithm and extended algorithm, total phase length is independent on parent-children ratio, whereas when using odd-even slot algorithm, the total phase length increases with an increased parent-children ratio.

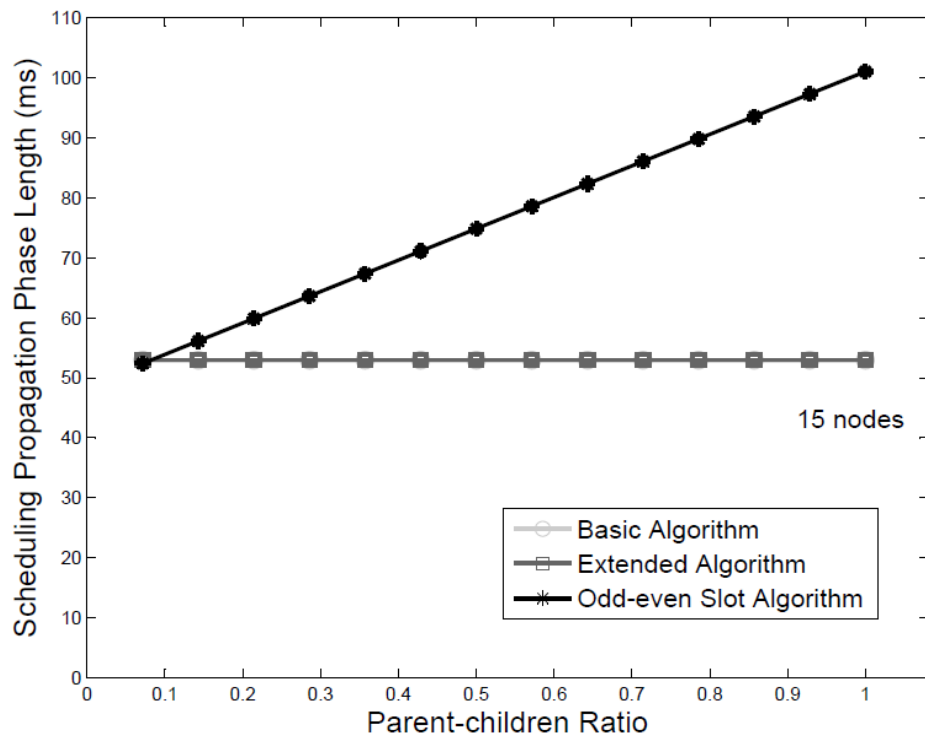


Figure 4.4: Relationship of total phase length against parent-children ratio (Total nodes number is 15)

4.2.2. Formal Analysis of the Results

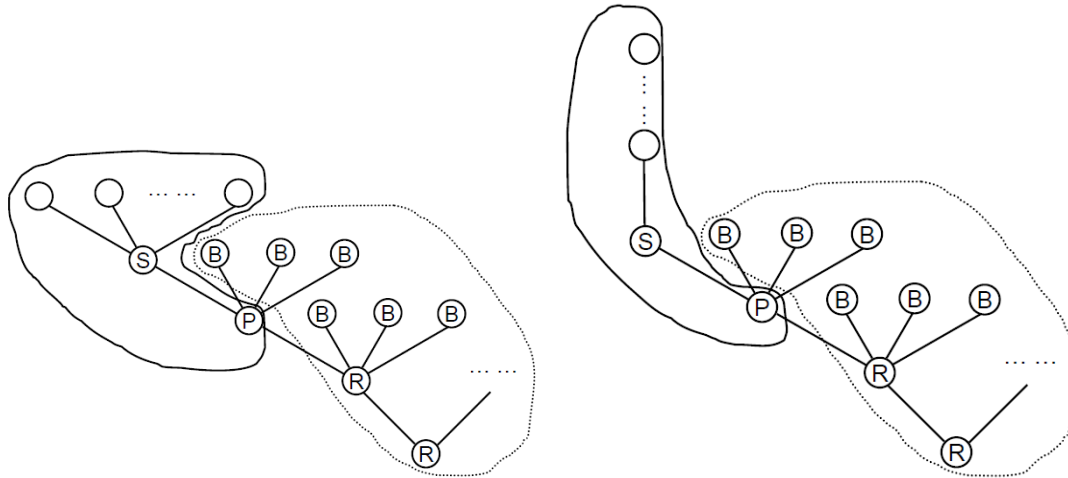


Figure 4.5: Situation of formal analysis

In order to analyze the influence the sequential and parallel structures impose on the two algorithms (we only compare extended algorithm and odd-even slot algorithm, because basic algorithm performs always worse than the extended algorithm does), we suppose the above situation. The traversal begins from the node S, through all of its children, back to S. We want to evaluate the radio-on time of all the nodes during this time. The traversed part can be either a parallel structure or a sequential structure, as showed surrounded with solid line in figure 4.5. The parent of S is also included, because it is also in the transmission range of S and therefore is influenced.

For sake of clear explanation, we must first specify some symbols:

- t_1, t_1' Radio-on time of "Sending" in extended algr. and odd-even slot algr., respectively.
- t_2, t_2' Radio-on time of "BSS DR" in extended algr. and odd-even slot algr., respectively.
- t_3, t_3' Radio-on time of "BSS NDR" in extended algr. and odd-even slot algr., respectively.
- t_4, t_4' Radio-on time of "BSF" in extended algr. and odd-even slot algr., respectively.
- t_0, t_0' Radio-on time of "Sleeping" in extended algr. and odd-even slot algr., respectively.
- TRT Total Radio-on Time (TRT) of all the nodes in particular period for extended slot algorithm.
- TRT' Total Radio-on Time (TRT) of all the nodes in particular period for odd-even slot algorithm.

The radio-on time of selected odd slot types and even slot types in odd-even slot algorithm and the time of selected types of slots in extended algorithm are showed in table 4.1. Even though there are two types of slots in odd-even slot algorithm, we only use the values of odd slots because of

the subtle difference of odd slots and even slots.

No.	Slot Type	Extended (us)	Odd slot (us)	Even slot (us)
1	Sending	1898	1898	1852
2	BSS DR	1898	1898	1852
3	BSS NDR	1182	1182	1144
4	BSF	821	629	626
6	SWS 2	X	250	250
0	Sleeping	192	192	192

Table 4.1: Radio-on time of selected slot types of extended and odd-even slot algorithms.

We first analyze the parallel structure. Three different cases will be considered. The first case is nodes in the solid-lined area, the second case is nodes in the dashed-lined area marked with B, the third case is nodes in the dashed-lined area marked with R. The radio-on time of nodes in solid line area and dashed line area will be calculated separately, because all the nodes in dashed line area are outside of the transmission range of S. They can be called waiting nodes. In the area bounded by dashed line, some nodes, marked by R, in the odd-even slot algorithm are waiting for SRI nodes, while others in the same area is waiting for SBI. So we have to classify the nodes into three types. we will analyze according to the following order.

- Case 1: Total radio-on time of nodes in solid-lined area (parallel structure traversal).
- Case 2: Total radio-on time of nodes waiting for SBI in dashed-lined area (parallel structure traversal).
- Case 3: Total radio-on time of nodes waiting for SRI in dashed-lined area (parallel structure traversal).

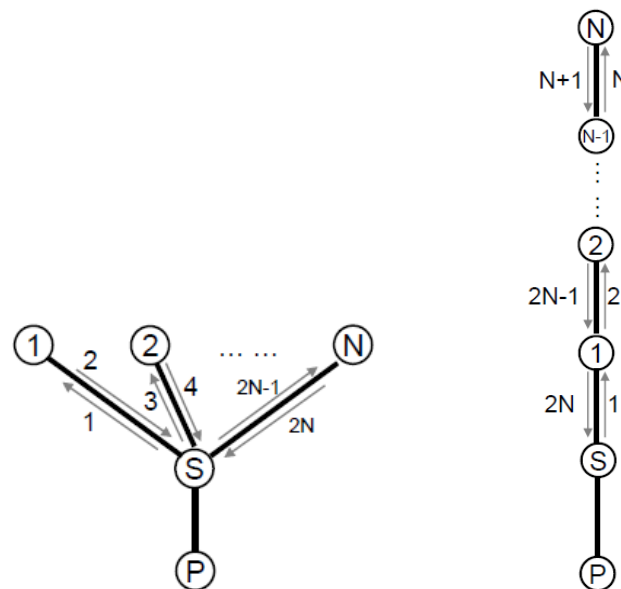


Figure 4.6: Enlargement of solid-lined area. Traversal is evaluated in a parallel structure (left) and in a sequential structure (right)

CASE 1

For extended algorithm, the total radio-on time of all the nodes is

$$\begin{aligned}
TRT &= \sum_{i=1}^N T_i + T_S + T_P \\
T_i &= (t_2 + t_1) + (t_3 + t_0)(i-1) + t_{20} \\
T_S &= (t_1 + t_2)N \\
T_P &= (t_3 + t_0)N
\end{aligned}$$

Where T_i is the radio-on time of node 1...N during the traversal period. The traversal order is from 1 to N. T_S is the radio on time of node S while T_P is the radio-on time of node P. N is the children number of S in the solid-lined area.

For odd-even slot algorithm, the total radio-on time of all the nodes is

$$\begin{aligned}
TRT' &= \sum_{i=1}^N T_i' + T_S' + T_P' \\
T_i' &= (t_2' + t_1') + (t_3' + t_0')(i-1) + t_{20}' \\
T_S' &= (t_1' + t_2')N \\
T_P' &= (t_3' + t_0')N
\end{aligned}$$

Because $t_1 = t_1'$, $t_2 = t_2'$, $t_3 = t_3'$, $t_0 = t_0'$, $t_{20} = t_{20}'$, $TRT = TRT'$. For case 1, the total radio-on time of all the nodes in the solid-lined area calculated by using two algorithms are the same.

CASE 2

By using table 4.2, we can acquire the total radio-on time of extended algorithm and odd-even slot algorithm.

The total radio-on time for extended algorithm is

$$TRT = t_4 \cdot 2N \cdot N_{dashSBI}$$

The total radio-on time for odd-even slot algorithm is

$$\begin{aligned}
TRT' &= (t_4' \cdot N + t_0' \cdot N) \cdot N_{dashSBI} \\
&= (t_4 \cdot N) \cdot N_{dashSBI}
\end{aligned}$$

An equation is used above $t_4 = t_4' + t_0'$. In addition, $N_{dashSBI}$ is the number of nodes which are in the dashed-lined area and wait for SBI. As we can see, the consumed time in algr.3 is only half of that in algr.2.

	1	2	3	4	...	2N
Algr. 2	4	4	4	4	...	4
Algr. 3	B	R	B	R		R

	4	0	4	0		0
--	---	---	---	---	--	---

Table 4.2: Partial time sequence diagram for nodes of case 2.

CASE 3

Similarly, by using table 4.3 below, we can get the total radio-on time of both algorithms.

The total radio-on time for extended algorithm is

$$TRT = t_4 \cdot 2N \cdot N_{dashSRI}$$

The total radio-on time for odd-even slot algorithm is

$$\begin{aligned} TRT' &= (t_4' \cdot N + t_0' \cdot N) \cdot N_{dashSRI} \\ &= (t_4 \cdot N) \cdot N_{dashSRI} \end{aligned}$$

Where $N_{dashSRI}$ is the number of nodes which are in the dashed-lined area and wait for the SRI.

In analogy with case 2, the radio-on time of related nodes for odd-even slot algorithm is half the time of the extended algorithm.

	1	2	3	4	...	2N
Algr. 2	4	4	4	4	...	4
Algr. 3	B	R	B	R		R
	0	4	0	4		4

Table 4.3: Partial time sequence diagram for nodes of case 3.

We then analyze the situation that a sequential structure is in the solid-lined area. Again, three cases are considered in this structure.

- Case 4: Total radio-on time of nodes in solid-lined area (sequential structure traversal).
- Case 5: Total radio-on time of nodes waiting for SBI in dashed-lined area (sequential structure traversal).
- Case 6: Total radio-on time of nodes waiting for SRI in dashed-lined area (sequential structure traversal).

CASE 4

In table 4.4 below, we derive, after sequential structure traversal, the number of slots executed for each slot type.

Slot type	Sending	BSS DR	Sleeping	BSF	BSS NDR
Extended	2N	2N	N-1	$\frac{3}{2}N^2 - \frac{3}{2}N$	N-1
Slot Type	Sending	BSS DR	Sleeping	BSF	

Odd-even	2N	2N	$\frac{3}{2}N^2 + \frac{5}{2}N - 2$	$\frac{3}{2}N^2 + \frac{1}{2}N$	
----------	----	----	-------------------------------------	---------------------------------	--

Table 4.4: Radio-on time comparison between extended algorithm and odd-even slot algorithm.

Each column in table 4.4 shows the total number of a certain type of slots which all nodes in this case have used in the whole scheduling propagation phase. Both the values acquired by using algorithm 2 and by using algorithm 3 are provided. For example, if we use algorithm no.2, then all together 2N “Sending” slots are employed in the whole phase. This value is calculated by considering all the nodes.

As we can notice that some elements have first order of N, while some have the second order of N, because the total radio-on time should be decided mainly by the elements with second order. There are two slot types for algorithm 3 which have the second order of N. But

$$t_4 = t_4' + t_0'$$

which means the radio-on time of slot “BSF” in the extended algorithm equals to that of slot “BSF” and “Sleeping” in odd-even slot algorithm. So if we just consider the second order of N, extended algorithm and odd-even slot algorithm should have the same radio-on time. And this part is main part of radio-on time. So in case 4, radio-on time of both algorithms are nearly the same.

CASE 5

The total radio-on time for extended algorithm is

$$TRT = t_4 \cdot 2N \cdot N_{dashSBI}$$

The total radio-on time for odd-even slot algorithm is

$$\begin{aligned} TRT' &= (t_4' + t_0') \cdot (2N - 1) \cdot N_{dashSBI} \\ &= t_4 \cdot (2N - 1) \cdot N_{dashSBI} \end{aligned}$$

An equation is used above $t_4 = t_4' + t_0'$. In addition, $N_{dashSBI}$ is the number of nodes which are in the dashed-lined area and wait for SBI.

As we can see, the consumed time in algr.3 is approximately the same as in algr.2.

CASE 6

The total radio-on time for extended algorithm is

$$TRT = t_4 \cdot 2N \cdot N_{dashSRI}$$

The total radio-on time for odd-even slot algorithm is

$$\begin{aligned} TRT' &= (t_4' + t_0') \cdot (2N - 1) \cdot N_{dashSRI} \\ &= t_4 \cdot (2N - 1) \cdot N_{dashSRI} \end{aligned}$$

Where $N_{dashSRI}$ is the number of nodes which are in the dashed-lined area and wait for the SRI.

The radio-on time got by using extended algorithm is nearly the same as the time by using odd-even slot algorithm.

To summarize the above seemingly complex formal analysis, we make table 4.5 which is intended to make a rough comparison of two algorithms in two situations, which are the parallel structure being traversed and the sequential structure being traversed. So the conclusion is that the odd-even slot algorithm performs better than extended only under the situation of parallel structure.

Structure	Parallel			Sequential		
Case	1	2	3	4	5	6
Comparison of radio-on time	Nearly same	Half*	Half*	Nearly same	Nearly same	Nearly same

Table 4.5: Summary of radio-on time difference of two algorithms for 6 cases.

* half means that the radio-on time by using odd-even slot algorithm is half of the time by using extended algorithm.

* nearly same means that by omitting some negligible elements, TRT and TRT' will have the same value.

4.2.3. Evaluation Results Analysis

Total Radio-on Time vs Parent-children Ratio

From figure 4.1, 4.2, and 4.3, we can state the following:

- 1) Basic algorithm has always a longer radio-on time than the other two algorithms.
- 2) Odd-even is always better than extended with medium parent-children ratios(i.e., less than 0.8).
- 3) With low parent-children ratios, extended algr. and odd-even slot algr. show great benefits
- 4) With high parent-children ratios, the difference between extended algr./odd-even slot algr. And the basic algorithm is much lower.

First Item

We first explain the first item. We know from the slot specification of the basic algorithm and extended (see figure 2.1 and 2.6) that the only difference between two algorithms is the movement of fragment 21 from the original place slot "BSS NDR" to the slot "Sleeping". In the time sequence diagram, every sequential "BSS NDR" and "BSF" in basic algorithm is replaced by "BSS NDR" and "Sleeping" in extended algorithm. Basic algorithm is thus worse than extended algorithm in all the cases. Thus in the following analysis, only the extended algorithm and odd-even slot algorithm are considered for the comparison.

Second Item

At medium ratios, when using odd-even slot algorithm, waiting nodes don't have to check every slot, which saves more energy.

Third Item

At low ratios, the topology looks like the left part of figure 4.6, which is a parallel structure. Whenever node S sends to any of its children, the other children can sense the transmission, and can save energy by sleeping in the next slot. This mechanism is used in extended algorithm and odd-even slot algorithm. In basic algorithm, there is no sleeping slot, which fails to save energy. So high benefits can be noticed at low parent-children ratio.

Fourth Item

At high ratios, the topology becomes more like the right part of figure 4.6, which is actually a sequential structure. In this structure, the mentioned mechanism has no effect, because all the nodes stand in a line. Except the recipient, nobody can sense the transmission. So little energy can be saved by using this mechanism.

Total Phase Length vs Parent-children Ratio

Some points from figure 4.4 are explained.

- 1) Basic algorithm and extended algorithm have a fixed total phase length, i.e. it is independent from the parent/children ratio.
- 2) Odd-even slot algorithm has a total phase length that increases with the parent/children ratio.

First Item

In figure 4.4, the line representing basic algorithm and the one for extended algorithm are overlapping, because the only change in extended algorithm is extension on sleeping mechanism, which can save energy but can not make the phase length shorter.

The overlapping line also shows that at different ratios the total phase length has the same value: When the total nodes number is fixed, then the link number is fixed. And two transmissions are done at each link, so the total phase length is fixed.

Second Item

When the ratio rises, the topology becomes more like the right part of figure 4.6. When the topology becomes more and more sequential, more transitional slots need to be added between receiving and sending.

4.3. Evaluation of Algorithms with Link Failure

Total Radio-on Time vs Parent-children Ratio

In this section, we will use the same method as the one employed in last section to explain the results of the figure 4.7, when link failures are taken into account.

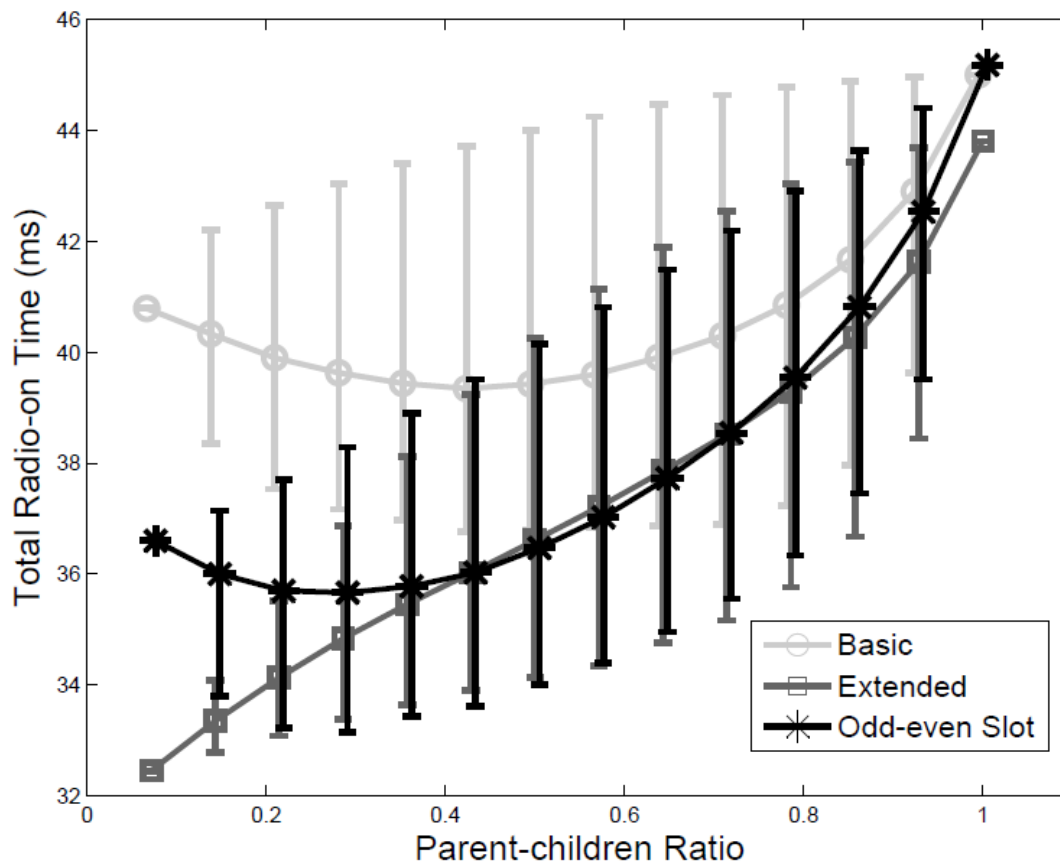


Figure 4.7: Relationship between total radio-on time and parent-children ratio, when two transmissions at each link are considered (Total nodes number is 15)

As we have done in the last section, we make a table 4.6 to show the radio-on time difference between two algorithms.

	Parallel Structure
Case 1	Odd-even wastes $2Nt_6$ ' time units more than Extended
Case 2	Odd-even uses Nt_4 time units less than extended
Case 3	Odd-even uses Nt_4 time units less than extended
	Sequential Structure
Case 4	Nearly the same
Case 5	Nearly the same
Case 6	Nearly the same (odd-even slightly worse)

Table 4.6: Summary of radio-on time difference for 6 cases under two-time sending scenario.

We can see from table 4.6 that A sequential structure, two algorithms have nearly the same performance, while the difference comes up when the parallel structure is evaluated. For case 1, odd-even slot algorithm has a worse performance, however in the other two cases (case 2 and case

3), the algorithm shows some advantages. By using the above data, we can explain the figure 4.7.

- Low parent-children ratio

At the lowest parent-children ratio, which is actually case 1, the results comply with the formal analysis. A relatively great difference can be noticed in figure 4.7 at this point.

- medium parent-children ratio

When the parent-children ratio increases, more nodes operating in case 2 and 3 will occur, which makes the odd-even slot algorithm slightly better than the extended algorithm.

- high parent-children ratio

Because of the slightly worse performance of odd-even slot algorithm compared to extended one in case 6, the odd-even algorithm performs worse than the extended algorithm.

In the scenario of figure 4.7, we have taken the link failure into consideration, but the two-time transmission at each link is not so realistic. So in the next scenario, a more reasonable assumption is given. Compared to the last scenario, we assign random transmission numbers to each link. But these transmission numbers in each link are carefully selected, in order to obey to a constraint on the average number and maximum transmission number. Thus, as the inputs, the average and maximum transmission numbers should be given.

Figure 4.8 shows how three algorithms perform in this scenario. Some obvious changes can be noticed.

- Odd-even slot algorithm performs always worse than extended algorithm.

Because at each link random times of transmission will be made, a node in odd-even slot algorithm has to wait for another slot to before every retransmission, which makes the performance of this algorithm worse and worse, as the transmission number increases. But for a node in extended algorithm, it is not required to wait for the available slot before retransmission, which shows better scalability.

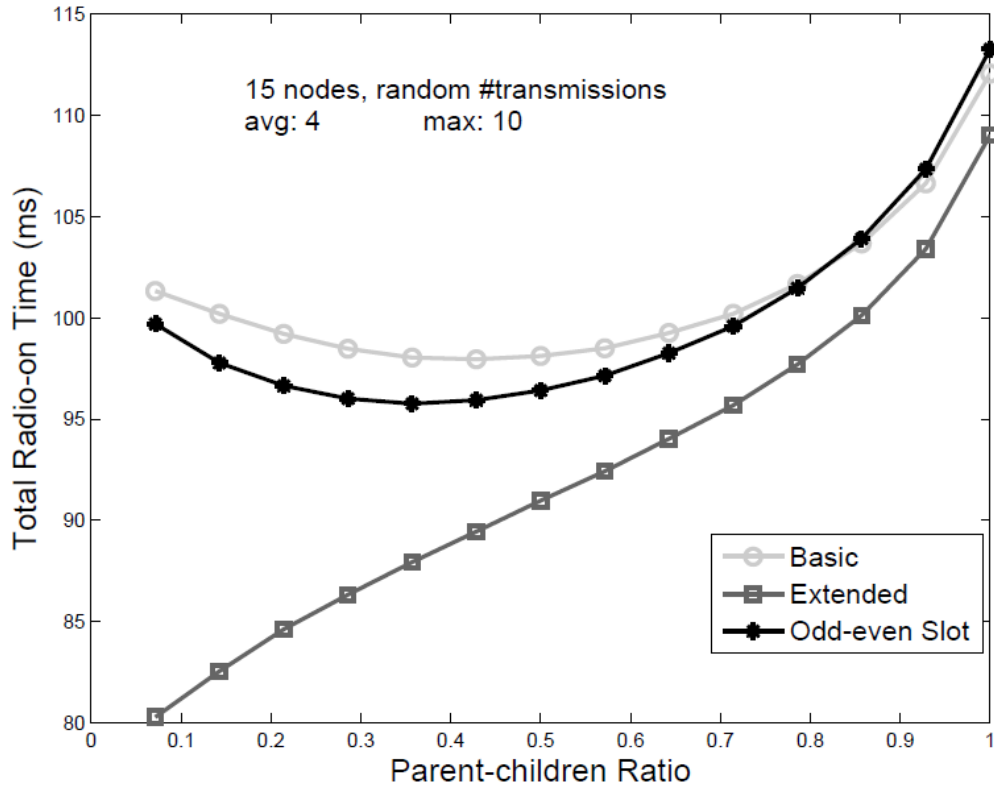


Figure 4.8: Relationship of total radio-on time against parent-children ratio, with an average of four transmissions at each link, 10 maximum transmissions (Total nodes number is 15)

Total Phase Length vs Parent-children Ratio

Figure 4.9 studies the relationship between total phase length against parent-children ratio. Similar to the previous figures, basic algorithm and extended algorithm are overlapping, because these two algorithms have the same latency. The odd-even slot algorithm instead has an increasing phase length and the phase length of odd-even slot algorithm is nearly twice as long as the other two. That is because we impose a random transmission number obeying average of 4 and maximum of 10, in most time, we have to do retransmissions. Whenever we have to retransmit, we have to waste one more slot, in order to wait for the next legitimate slot. So in most time, twice as many slots as the other two algorithms are used to transmit.

What's more, we notice a slight increasing tendency for odd-even slot algorithm. This slight change is caused by variation of topology, which transforms from parallel to sequential, where the same type of information is continuously transmitted.

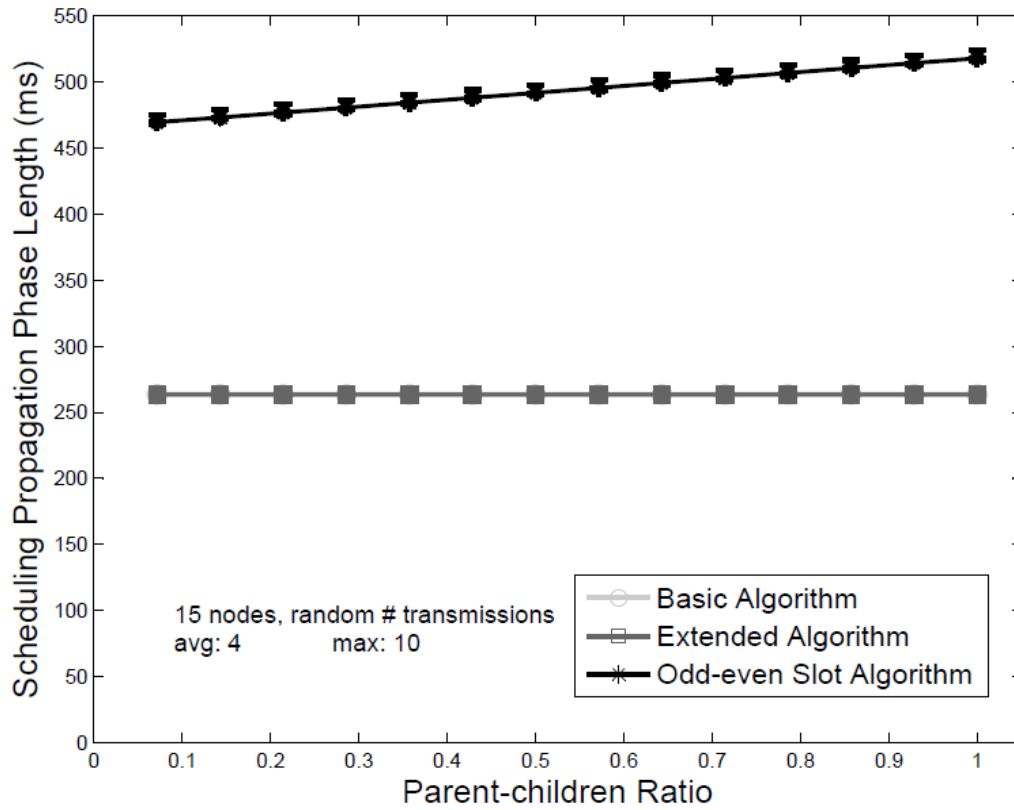


Figure 4.9: Relationship of total phase length against parent-children ratio, on average four times transmission at each link, maximum transmission will be 10, which means transmission number will not larger than 10 (Total nodes number is 15)

Conclusions

In this paper, we propose three algorithms for scheduling propagation. The first one, basic algorithm, finishes the scheduling by traversing every node in the tree. The second one, extended algorithm, introduces “Sleeping” state. The third one is based on an idea of odd and even slots.

Then we finish a JAVA framework to evaluate phase length and energy consumption. Given a certain topology, the JAVA program outputs the latency (phase length) and average radio-on time (energy consumption) in the scheduling propagation phase.

At last, link failure is considered. We change slot specifications and state machine charts of three algorithms accordingly. Results without link failure and ones with link failure are analyzed.

Bibliography

- [1] 2.4 GHz IEEE 802.15.4/ZigBee-ready RF Transceiver. CC2420. Chipcon Products from Texas Instruments.
- [2] K.K. Chintalapudi, L.Venkatraman. On the design of MAC protocols for Low-Latency Hard Real-Time Discrete Control Applications Over 802.15.4 Hardware. 2008 International Conf. on Info. Processing in Sensor Networks.
- [3] F.Ferrari. Proposal for Semester Thesis at the Department of Information Technology and Electrical Engineering.