社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE.

LoRa ネットワークにおける位相調整制御を用いた 動的送信スケジューリング手法

坂東　力†　　中尾　彰宏†

† 東京大学, 東京都文京区本郷 7-3-1,113-0033
E-mail: †ricky1231@g.ecc.u-tokyo.ac.jp, ††nakao@nakao-lab.org

あらまし　低消費電力かつ広域な IoT ネットワークを実現する無線通信技術として、Low Power Wide Area (LPWA) に注目が集まっている。無線センサーでは周期的に環境センシングを行うことが多いが、LPWA の一種である LoRa では MAC 層プロトコルに ALOHA を用いており、センサー数の増加によって間欠通信のタイミングが重なり、パケット衝突が増加するという課題がある。そこで本稿では、位相調整制御により各センサーの送信タイミングをずらし、動的にトラフィックを平滑化するスケジューリング手法を提案する。さらにネットワークシミュレーター NS-3 を用いて、パケット到達率の改善について提案手法の有効性を評価する。
キーワード　IoT, LPWA, LoRa, 周期トラフィック, ALOHA, 位相調整制御

Dynamic Scheduling Method Using Phase-Shift Manipulation for LoRa Wireless Network

Riki BANDO† and Akihiro NAKAO†

† The University of Tokyo, 7-3-1, Hongo，Bunkyo-Tokyo, 113-0033, Japan
E-mail: †ricky1231@g.ecc.u-tokyo.ac.jp, ††nakao@nakao-lab.org

Abstract　Low Power Wide Area (LPWA) is attracting attention as a wireless communication technology to realize low power consumption and wide-area IoT networks. Wireless sensors often perform environmental sensing periodically, but LoRa, one type of LPWA, uses ALOHA for the MAC layer protocol, which causes an increase in packet collisions due to overlapping of intermittent communication as the number of sensors increases. In this study, we propose a scheduling method that dynamically smoothes the traffic by shifting transmission timing of each sensor with phase-shift manipulation. Furthermore, we use a network simulator NS-3 to evaluate the effectiveness of the proposed method in terms of improving packet delivery ratio.
Key words　IoT, LPWA, LoRa, Periodic traffic, ALOHA, Phase-shift scheduling

1. はじめに

IoT システムにおける通信では、多数の端末がネットワークに接続され、一回に送信されるデータサイズが比較的小さいという特徴がある。加えて、センサーは屋外に設置される場合もあり電力供給が見込めるとは限らないので、バッテリーを搭載して電力消費を抑える必要がある。省電力かつ安価に広域な無線センサーネットワークを実現するために注目されているのが、 LPWA と呼ばれる無線技術である。LPWA は Low Power Wide Area の略で、省電力かつ長距離の通信が可能な一方で、通信速度が数 kbps から数百 kbps 程度と低速である。主な LPWA 方式としては、免許不要で利用できる SIGFOX・LoRa 、免許が必要な周波数帯を用いる eMTC・NB-IoT などがある。

LoRa は Long Range の略で、現在最も普及している LPWA 方式の一つである。チャープ拡散というスペクトル拡散方式の拡張により、ノイズに強い長距離の通信が可能である [1] [2]。LoRa では ISM バンドと呼ばれる免許不要の低周波数帯が用いられ、最大伝送速度は 250 kbps 程度、最大伝送距離は 10 km 程度である。LoRa では MAC 層プロトコルとして ALOHA を採用しており、各端末が自律的にランダムアクセスを行う。IoT システムでは端末が周期的な間欠通信を行うことが多く、ALOHA によるアクセス制御では端末の増加に伴い通信タイミングが重なり、パケット衝突が増加する懸念がある。そこで本研究では、サーバー側から各端末に遅延付加の命令を送信し位相調整を行うことで、動的にパケット衝突を解消し帯域の利用

効率を向上させる手法を提案する。

本稿の構成は次のとおりである。まず 2 章では LoRa 通信の概要を説明する。3 章では LoRa 通信における衝突回避の関連研究を示す。4 章では提案手法を述べ、5 章ではシミュレーション結果を示す。6 章では今後の課題を述べ、7 章で本稿を総括する。

2. LoRa 通信の概要

LoRa の変調方式では、時間軸方向にサイクリックシフトしたチャープ信号を用いて通信を行い、通信情報の最小単位はシンボルで定義される。信号の拡散率 (Spreading Factor: SF) は 7 から 12 までの整数によって設定でき、一つのシンボルに SF ビットの通信データが含まれる [3]。シンボルレート (RS) ・帯域幅 (BW) ・ SF 値には以下の関係が成り立つ [4]。

$$RS = \frac{BW}{2^{SF}}$$

SF 値を上げることでシンボルレートが小さくなり、すなわち信号が拡散されノイズ耐性が高くなる。よって受信感度が向上し伝送距離が伸びるが、伝送速度が低下しフレームあたりの帯域占有時間が増加する。

LoRaWAN とは、物理層に LoRa を用いた無線ネットワークを指す。帯域幅は 62.5 kHz・125 kHz・250 kHz・500 kHz などの設定が可能である。LoRaWAN のトポロジーはネットワークサーバー・ゲートウェイ・エンドデバイスから構成され、スター型の構成を取る。LoRa による無線通信はエンドデバイスとゲートウェイ間で行われ、ゲートウェイとネットワークサーバー間では IP 通信が行われる。各デバイスは 32 bit のデバイスアドレス (DevAddr) で識別される。

LoRaWAN はクラス A ・クラス B ・クラス C の三つの通信クラスに分かれる。クラス A は主にエンドデバイスからサーバーへの通信と ACK 応答、クラス B はビーコンによるサーバーからエンドデバイスへの通信もサポートし、クラス C は常時受信スロットを開きダウンリンクの通信が出来るようにクラス A を拡張したものである [5] 。本稿では消費電力が最も少ないクラス A を用いる。クラス A では、受信スロットはフレーム送信の一定時間後に開かれ、送信と送信の間はスリープすることで電力の消費を抑える。スリープ中はフレームを受信することは出来ない。

デューティサイクルとは、チャネルごとにエンドデバイスに許容される送信周期のことで、チャネルによって 1 % か 0.1 % が設定される [1] 。デューティサイクルを d 、チャネル数を n 、 SF として i を用いたときの転送時間 (Time on Air: ToA) を T_{a_i} とする。 LoRaWAN の MAC プロトコルでは、純粋な ALOHA に加えて、

$$\frac{nd}{T_{a_i}}$$

のパケット送信率を超えられないという制約がある [6]。すなわち、逆数の

$$\frac{T_{a_i}}{nd}$$

のパケット送信間隔を開ける必要がある。パケット送信要求があっても、この周期に合うまでパケットの送信は延期される。この制約はゲートウェイのダウンリンク通信にも適用される。さらに LoRa ゲートウェイは半二重通信となっており、フレームの受信と送信を同時に行うことは出来ない。

LoRa では、非確認 (unconfirmed) メッセージと確認 (confirmed) メッセージの二種類を利用できる [3]。非確認メッセージでは送信の到達は確認せず、受信側は ACK 応答を発行しない。確認メッセージでは、受信側は ACK 応答を発行するため送信側がメッセージの到達を確認できるが、ACK 応答の受信のため消費電力は大きくなる。さらに、クラス A では端末が受信ウィンドウを開くタイミングはメッセージ送信の一定時間後である。この様子を図 1 に示す。デバイスはメッセージを送信してから Rx Delay1 経過後に受信ウィンドウ Rx1 を開く。この間に ACK が届かなければ、Rx1 を閉じてから Rx Delay2 経過後に下りチャネルで受信ウィンドウ Rx2 を開く。もし Rx2 でも ACK 応答を受け取れなければ、デバイスは前回とは異なるチャネルを用いてフレームの再送を行う。

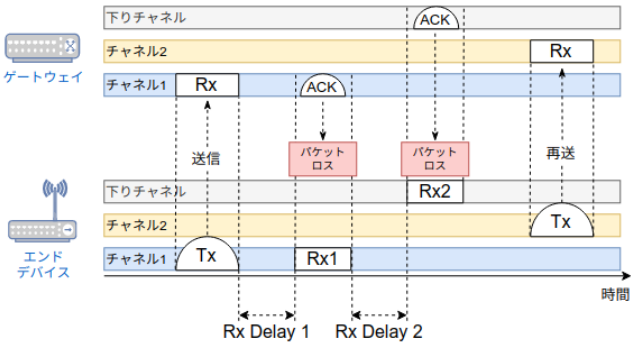

図 1　メッセージ確認の様子

3. 関 連 研 究

LoRa 通信におけるパケット衝突の回避についての既存研究としては、 MAC プロトコルとしてキャリアセンス多元接続 (Carrier Sense Multiple Access: CSMA) を用いる手法 [7] や、複数のゲートウェイで信号を受信する手法 [8] がある。文献 [7] では、CSMA によるわずかな電力消費の増加と引き換えに、パケット衝突率を大幅に減少できることが示されている。しかし CSMA を用いると、受信感度の違いからデバイスで認識できないフレームがゲートウェイに届き、結果的に ALOHA と同じ状況になる場合がある、という点が指摘されている [9]。また文献 [8] ではゲートウェイを複数設置することによりカバレッジが広がり通信の安定性が向上することが報告されているものの、ゲートウェイあたりの接続デバイス数が向上するわけではないので、根本的な解決とはなっていない。

LoRa 通信で信頼性のある送受信を行うには、確認メッセージを利用する方法がある。しかし LoRaWAN では、確認メッセージの ACK 応答やメッセージの再送により帯域の利用効率が著しく低下するという問題がある。文献 [10] のシミュレーションでは、全てのデバイスが確認メッセージを送信した場合、非確認メッセージを送信する場合に比べてパケット衝突が増加

し、ネットワークのグッドプットは最大 85 % 程度減少することが報告されている。主な原因としては、ゲートウェイの送信フレームが増加する結果、一度に一つのフレームしか送信できないゲートウェイはデューティーサイクルによって送信が制限される点や、ゲートウェイが半二重通信で ACK 応答を送信中に受信したフレームはロスしてしまう点が挙げられる [11]。

この問題を回避するため、ゲートウェイとデバイスで完全な時刻同期を行い、各デバイスが送信するタイムスロットをずらして ACK 応答を集約する方式が提案されている ([12] [13] [14])。この方式では、ACK 応答を一つのパケットにまとめて送信することで帯域の圧迫を解消し、ゲートウェイからの送信のデューティーサイクルの制約も満たせるため、スループットを大幅に改善することができる。しかしこの方式では、集約した ACK 応答が失われた場合に全てのデバイスが再送を行う必要があり、スループットの低下が大きい。また、時刻同期によって各デバイスが送信できるタイムスロットは固定されているので、再送が行われるほどサーバーに到達するデータとデバイス側で取得したデータにギャップが生じる。加えて、ネットワーク参加要求に対する ACK 応答が返ってこなければデバイスは送信スケジュールに参加できず、ネットワーク加入時には送信タイミングの同期のためにデバイス側での計算が必要である。帯域の状況は刻々と変化するため、偶発的なネットワークの遅延やデバイスの参加に対応するためにはサーバーとの同期を定期的に行う必要があり、その度に再計算の必要が生じデバイスとしての消費電力は増加するという問題がある。

本稿に近いアプローチとして、文献 [9] がある。これは、ゲートウェイと各デバイスに通信タイミングを管理する位相振動子モデルを導入し、空きスロットを把握してゲートウェイからの ACK 応答に位相更新の命令を載せることで、全デバイスの位相が等間隔にずれた逆相同期状態を自律分散的に実現する手法を提案している。しかし文献 [9] でのシミュレーションでは、30 分のデータ送信周期に対してデバイス数が 50 台と少なく、ACK 応答の増加に伴う帯域利用効率の低下に関しての提案手法の有効性は議論されていない。本稿では、より帯域が輻輳し ACK 応答のロスが生じるような状況を想定し、サーバー側からの中央制御の位相調整による提案手法の評価を行う。

以上の通り LoRa 通信における輻輳回避の研究は多く存在するが、多数のデバイスがサーバーに周期的なトラフィックを送る場合に動的なパケットロスの解消を行う研究はほとんど成されていない。そこで本稿では、確認メッセージの ACK 応答を利用してネットワークサーバーからデバイスごとに遅延付加要求を通知し、位相調整を行うことで、動的にパケットロスを解消し帯域の利用効率を高める手法を提案する。

4. 提 案 手 法

本稿では単一チャネルでの通信を想定し、複数のデバイスが一つのゲートウェイを介して、一つのネットワークサーバーに周期的にメッセージを送信する状況を仮定する。送信周期は一定で、各デバイスが送るフレームの拡散率・パケットサイズは全て等しいものとする。ACK 応答はメッセージと同一のチャネルで送信し、ACK の受信には Rx1 ウィンドウのみ用いる。また、ACK 受信失敗時にフレームの再送は行わない。

提案手法では帯域をスロットで管理し、ランダムにネットワークに加入するデバイスの送信タイミングを、ACK 応答を選択的に返すことで適切なスロット位置にずらす。スロット長 (Slot Length) に関しては ACK 応答は考慮せず、フレームの ToA よりやや大きく設定する。スロット長に余裕をもたせることで送信時のゆらぎによるフレームの衝突を防ぐことができるが、スロット長が大きすぎると帯域の利用効率が低下するため、通信の安定性を考慮して適切に設定する必要がある。ここでは、スロット長の単位量 (Slot Unit Size) を定め、フレームの ToA と天井関数を用いて以下のように計算する。

$$SlotLength = ceil(ToA/SlotUnitSize) * SlotUnitSize$$

提案手法は、

1. ランダムバックオフによる帯域資源の探索
2. フレーム受信情報による帯域資源のコンパクション

の大きく 2 つに分かれる。

4.1 ランダムバックオフによる帯域資源の探索

デバイスはネットワーク加入時、ネットワークサーバーに対してパケットを送信し通信を試みるが、一定回数 ACK 受信に失敗し続けたら、送信周期にランダムなバックオフ時間を付加し送信タイミングをずらす。

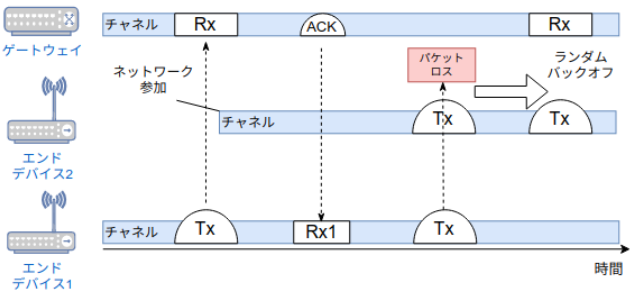

図 2 ランダムなバックオフ時間の付加

各デバイスはこの操作を ACK 受信が初めて成功するまで繰り返し、帯域資源の探索を行う。デバイスは ACK 受信に成功したら、ネットワークへの加入に成功したものとして送信タイミングを固定し、以後 ACK 受信の成否によらずバックオフ時間の付加は行わない。バックオフ時間は予め最大値を固定値で指定し、付加する際にランダムに生成する。

4.2 フレーム受信情報による帯域資源のコンパクション

ネットワークサーバーでは、届いたフレームのフレームヘッダからデバイスアドレスを抽出して送信元デバイスを識別し、デバイスアドレスをキーとして以下の情報を管理する。

timestamp: フレームの到達時刻
nextDevAddr: 次のフレームのデバイスアドレス
nextDevDuration: 次のフレーム到達までの時間差

これらの情報はフレームが到着する度に更新し、また生存時間 (Time to Live: TTL) を設定して、フレーム到達時刻から

の経過時間が TTL よりも長ければ該当デバイスは離脱したものとして扱うことで、偶発的なデバイスの離脱にも対応する。加えて、ネットワークサーバーでは到達したフレームのデバイスアドレスを両端キューで管理し、新しいデバイスアドレスのフレームが到着する度に末尾にアドレスを挿入して順番を保持する。

次に、各デバイスに通知する遅延量の計算について述べる。両端キューの末尾からデバイスアドレスを取り出し、対応するフレームに対して ACK が送信可能かを確認する。具体的には、一周期前の情報から次のフレームまでの時間差分を連鎖的に遡ることで、対象フレームを受信後 Rx Delay 経過後に ACK を送信する間に到達するフレームが無いことを確認する。この様子を図 3 に示す。図 3 では、エンドデバイス 1 に対して ACK 応答を送信する場合、ACK 応答が送信完了する前に別のフレームの受信が始まるため、ACK 応答は送信できないと判断できる。ここで、フレームの衝突によりサーバー側では情報が存在せず、帯域が利用可能だと誤認識し ACK 応答のロスを招く場合があるが、これは前述のランダムバックオフ機構により周期を重ねるにつれて解消され、帯域が占有されて ACK 送信が不可能だと正しく認識される。

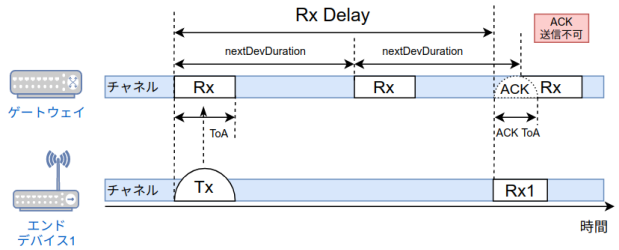

図 3　ACK 応答送信可否の確認

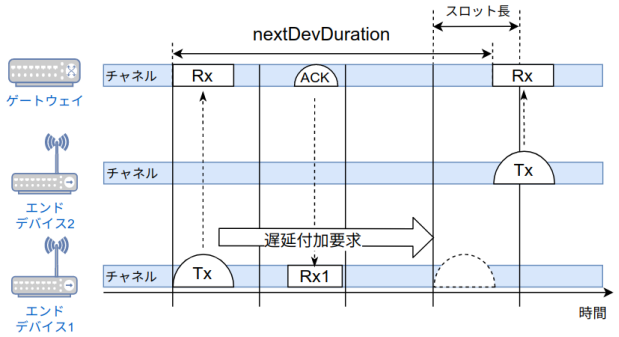

図 4　遅延付加要求の送信

ACK が送信可能だと判定できれば、ACK 応答のペイロードに遅延量の情報を載せることで、各デバイスに遅延付加要求を通知する。遅延量の計算方法について述べる。まずネットワークサーバーは、一番最初に到着したフレームのタイムスタンプを基準に、帯域におけるスロット位置を算出する。そして対象フレームの次に到着するフレームについて、フレームの ToA を考慮した直前の利用可能なスロット開始位置までずらすこととし、現在時刻との差分を遅延量とする。デバイスは受け取った遅延量だけ動作を遅延させ、次回のフレーム送信のタイミングをずらす。この操作によって、対象フレームの直前のフレームは nextDevDuration の値が増加するため、遅延付加の際により多く遅延量を指定することが出来る。もし ACK が送信不可能だと判定したら、デバイスアドレスを両端キューの先頭に挿入し、操作が一巡した後に再び判定を行う。

このように、新しいアドレスから順に遅延付加を行うのは、遅延付加によって生まれた余剰資源を次のフレームの計算に活かすためである。もし古いアドレスから順に遅延付加を行うのであれば、遅延量は直後のフレーム位置を上限とするため、場所が入れ替わるだけで根本的に各デバイスの送信タイミングを詰めることはできない。

以上の操作を繰り返して両端キューが空となったら、フレーム受信情報のリセットを行い各デバイスに対して遅延量を計算し直す。これによって、デバイスの離脱や新規デバイスの参加により古くなった送信スケジュールを更新し、定期的にフレームの衝突が少ない状態を保つことが出来る。提案手法においてデバイス側で求められるのは、バックオフ時間の生成・付加、そして届いた ACK 応答のペイロードを読み取り送信周期を変更することのみなので、動作として余分な電力消費を必要としないという特徴がある。

表 1　シミュレーションパラメータ

Parameter	Value
周波数	868.1 MHz (EU リージョン)
帯域幅 (BW)	125 kHz
デューティーサイクル	1 %
Rx Delay	1 s
ペイロードサイズ	25 bytes
ACK ペイロードサイズ	4 bytes
シミュレーション時間 (ST)	50000 s, 100000 s
送信周期	300 s
デバイス数 (N)	20, 40, 60, 80, 100, 120, 150, 180
拡散率 (SF)	12
最大バックオフ時間	10 s
スロット長の単位量	150 ms

5.　性 能 評 価

離散事象シミュレーターである NS-3 [15] の lorawan モジュール [16] を用いてシミュレーションを行い、提案手法の性能を評価する。主なパラメーターを表 1 にまとめる。

シミュレーションでは一つのゲートウェイの半径 1 km 以内にデバイスがランダムに配置され、ネットワークサーバとの通信を任意のタイミングで開始する。ペイロードサイズは 25 bytes とし、ヘッダーが付いてフレームサイズは 34 bytes となる。フレームサイズによる ToA は文献 [17] から計算でき、SF 値・フレームと ACK 応答の転送時間・デューティサイクルに伴う最小送信間隔をまとめると表 2 の通り。表 2 より SF12 かつ送信周期 300 s でメッセージを送信する場合、フレームの転送には約 1.65 s、ACK 応答の転送には約 1.15 s を要するため、

一周期に送信できるフレーム数は非確認メッセージの場合で最大 180 個程度、確認メッセージで全フレームに ACK を返す場合は最大 100 個程度となる。またスロット長の単位量は 150 ms なので、この場合のスロット長は 1650 ms と計算できる。

表 2　SF 値ごとのフレーム・ACK 応答の転送時間

SF 値	フレーム転送時間	ACK 応答転送時間
SF7	0.077056s	0.0046336
SF8	0.133632s	0.082432
SF9	0.246784s	0.164864
SF10	0.452608s	0.288768
SF11	0.905216s	0.577536
SF12	1.64659s	1.15507 s

実験では、提案手法を用いた場合 (Proposed) のデバイス数 (N)・シミュレーション時間 (ST) の違いによるパケット到達率 (Packet Delivery Ratio : PDR) を、同じ初期状態から確認メッセージで全てのフレームに ACK 応答を返す場合 (Confirmed) と非確認メッセージで通信を行う場合 (Unconfirmed) とで比較を行う。実験結果における提案手法の数値は、10 回の試行の平均値とする。

5.1　シミュレーション時間による変化

まず $(N, SF, ST) = (80, 12, 100000)$ の時のシミュレーション経過時間と PDR の関係を示したのが図 5 である。この図より、Proposed は Confirmed の場合より全体を通して PDR は高く、その改善率は時間の経過に伴って上昇することが分かる。またシミュレーション開始から 3000 s 手前までは、提案手法は Unconfirmed の場合よりも PDR が低い。これは、初めてネットワークに参加するデバイスへの ACK 応答によってフレームの衝突が発生することが原因である。一方、Proposed では ACK は選択的に送信されるので、3000 s 以降は提案手法の PDR は Unconfirmed を上回り、100000 s 経過時点で PDR は Unconfirmed に比べて 20% 程度高い値となっている。

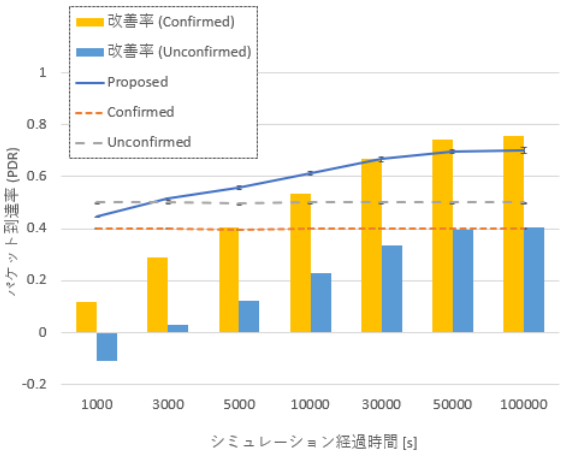

図 5　シミュレーションの経過時間による PDR の変化

デバイスが重複なく連続して到達する区間をイテレーションとして区切り、提案手法適用前後でのイテレーション内のデバイス数の変化を表したグラフが図 6 である。この図より、シミュレーション開始直後はパケット衝突によりデバイスは 33 台程度しかサーバーに到達していないが、提案手法実施後は 60 台程度に増加していることが分かる。100 回目のイテレーション以降デバイス数が振動するのは、帯域の輻輳によりネットワークに接続できていないデバイスは ACK 応答が届きにくくなるため、サーバーに到達しているフレームがランダムバックオフにより帯域を探索しているフレームと衝突し、次のイテレーションで再び到達することを繰り返すのが原因である。

図 7 は提案手法実施前後での nextDevDuration の分布を示したものである。図 7 の提案手法実施前後の nextDevDuration の平均・最大値・最小値に着目すると、実施前より実施後の方が値が小さくなり、フレームがサーバーに到達する間隔が狭まっていることが分かる。

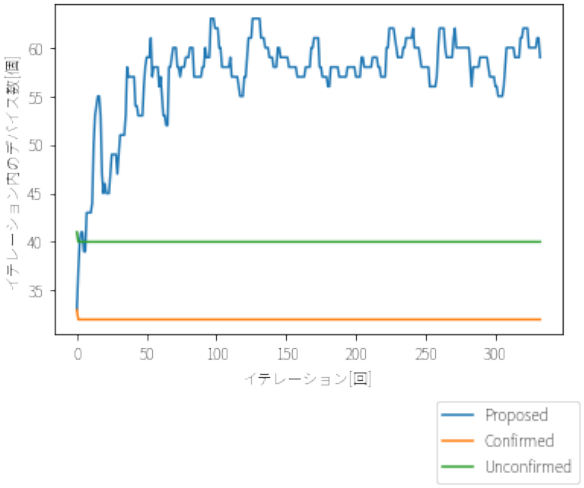

図 6　イテレーション内のデバイス数の変化

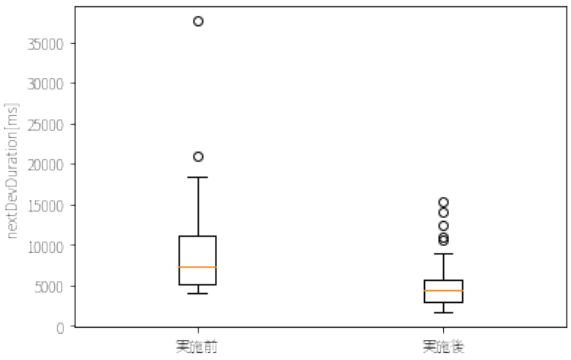

図 7　提案手法実施前後の nextDevDuration の分布の変化

5.2　デバイス数による比較

次に $(ST, SF) = (100000, 12)$ とした時のデバイス数 (N) と PDR の関係を図 8 に示す。

図 8 より、デバイス数の増加に従って提案手法実施後の PDR は線形に減少することが分かる。PDR の改善率に関しては初期状態の違いによる揺らぎはあるものの、N=60,80 の場合が高くなっており、これはデバイス数が少ない場合は帯域に余裕があって初期状態でのフレームが衝突が少なく、またデバイス数が多い場合については帯域の輻輳により位相調整の余裕がないため、提案手法による PDR 改善の効果が限定的になるためだと考えられる。

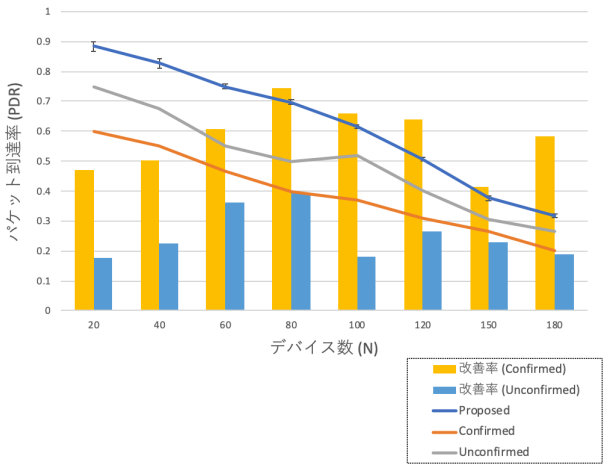

図 8　ネットワーク参加デバイス数による PDR の変化

6. 今後の課題

本稿で述べた提案手法では ACK 応答がデバイスに到達することを仮定して制御を行うが、輻輳の度合いによる ACK 応答の到達確率やデバイス数との関係性については明らかにできていない。また現状では、遅延付加の情報を載せた ACK 応答が別のフレームと衝突しデバイスを制御できないという状況が生じるが、制御信号としての ACK 応答に優先度をつけるなど、フレーム数が増加し ACK 応答の送受信が困難となるような状況でも、効果的に位相調整制御を行うことができる仕組みを検討したい。さらに、異なる送信周期のデバイスや複数のトラフィックパターンの機器が混在する場合などについて、提案手法の改善を行う予定である。また、具体的な電力消費量の評価や提案手法の実機を用いた場合の有効性の検証も今後の課題とする。

7. 結　　論

本稿では、複数のエンドデバイスが周期的にネットワークサーバーにメッセージを送信する状況において、サーバー側から各エンドデバイスに対して送付する ACK 応答で遅延付加の要求を通知し位相調整を行い、またデバイス側ではランダムバックオフにより帯域の探索を行うことで、フレームの衝突を動的に解消し帯域の利用効率を高める手法を提案する。さらに本稿では NS-3 を用いてシミュレーションを行い、提案手法の有効性を評価する。得られた結論としては、提案手法を用いることで確認メッセージを用いる場合と比べて最大 30% 程度、また非確認メッセージを用いる場合と比べて最大 20% 程度、ネットワークサーバーへのパケット到達率を改善できる。

文　　献

[1] L. Vangelista, A. Zanella, and M. Zorzi, “Long-range iot technologies: The dawn of lora,” Future access enablers of ubiquitous and intelligent infrastructures, pp.51–58, Springer, 2015.
[2] F. Sforza, “Communications system,” March 26 2013. US Patent 8,406,275.
[3] 鄭立, IoT ネットワーク LPWA の基礎: SIGFOX, LoRa, NB-IoT, RIC Telecom, 2017.
[4] M.A. Ertürk, M.A. Aydın, M.T. Büyükakkaşlar, and H. Evirgen, “A survey on lorawan architecture, protocol and technologies,” Future Internet, vol.11, no.10, p.216, 2019.
[5] L. Alliance, “Lorawan v1.1 specifiication,” 2017. https://lora-alliance.org/sites/default/files/2018-04/lorawantm_specification_-v1.1.pdf.
[6] F. Adelantado, X. Vilajosana, P. Tuset-Peiro, B. Martinez, J. Melia-Segui, and T. Watteyne, “Understanding the limits of lorawan,” IEEE Communications Magazine, vol.55, no.9, pp.34–40, Sep. 2017.
[7] T.H. To and A. Duda, “Simulation of lora in ns-3: Improving lora performance with csma,” 2018 IEEE International Conference on Communications (ICC), pp.1–7, IEEE, 2018.
[8] A.J. Wixted, P. Kinnaird, H. Larijani, A. Tait, A. Ahmadinia, and N. Strachan, “Evaluation of lora and lorawan for wireless sensor networks,” 2016 IEEE SENSORS, pp.1–3, IEEE, 2016.
[9] 小南大智, 合原一究, and 村田正幸, “Lpwa ネットワークにおける基地局負荷の分散を考慮した自律分散型送信スケジュール手法,” 電子情報通信学会技術研究報告= IEICE technical report: 信学技報, vol.117, no.353, pp.127–132, 2017.
[10] A.l. Pop, U. Raza, P. Kulkarni, and M. Sooriyabandara, “Does bidirectional traffic do more harm than good in lorawan based lpwa networks?,” GLOBECOM 2017-2017 IEEE Global Communications Conference, pp.1–6, IEEE, 2017.
[11] V. Di Vincenzo, M. Heusse, and B. Tourancheau, “Improving downlink scalability in lorawan,” ICC 2019-2019 IEEE International Conference on Communications (ICC), pp.1–7, IEEE, 2019.
[12] D. Zorbas, K. Abdelfadeel, P. Kotzanikolaou, and D. Pesch, “Ts-lora: Time-slotted lorawan for the industrial internet of things,” Computer Communications, vol.153, pp.1–10, 2020.
[13] 長谷川洋平, 鈴木一哉, and 登内敏夫, “Lorawan 通信のためのマルチユーザ確認応答フレーム集約方式,” 電子情報通信学会技術研究報告 = IEICE technical report : 信学技報, vol.118, no.124, pp.185–190, URL=, July 2018.
[14] G. Yapar, T. Tugcu, and O. Ermis, “Time-slotted aloha-based lorawan scheduling with aggregated acknowledgement approach,” 2019 25th Conference of Open Innovations Association (FRUCT), pp.383–390, IEEE, 2019.
[15] “The ns-3 network simulator.” https://www.nsnam.org/.
[16] D. Magrin, M. Capuzzo, S. Romagnolo, and M. Luvisotto, “Lorawan ns-3 module.” https://github.com/signetlabdei/lorawan, 2019.
[17] Semtech Corporation, “Sx1272/3/6/7/8 lora modem designer’s guide,” 2013.