7V-06

形式手法を用いた通信エラーを考慮した無線センサネットワークの検証

池田愛大† 宮崎敏明†

†会津大学大学院コンピュータ理工学研究科

1. はじめに

無線センサネットワーク(WSN)は環境モニタリングに有用である。また、各センサノード(以下、単にノード)内で実行されるプログラムを変更することで、当該ノードや WSN 全体の振る舞いを変更できる。一方、全てのノードが同時に動作し、相互に影響を及ぼし合うため、WSN 全体の振る舞いは複雑である。そのため、WSN の動作検証には、従来からシミュレーションを用いるのが一般であるが、カバレッジが十分なテストケースを用意することが困難である。

本問題に対する1つの解決法として、形式検証を適用することが試みられてきた[1,3,4,6]。後藤ら[1]は、ノードの動作を UML を用いて記述し、それをモデル検査ツール SPIN[2]の入力言語である Promela に変換して検証を試みた。Oleshchuk[3]は、ノードの動作仕様を直接 Promela を用いてモデリングし、SPIN によって検証した。筆者らは、ノードの実装を意識して、C 言語のサブセットを用いてノード動作を記述し、そこから Promela 記述を生成して SPIN で検証するシステムを提案した[4]。さらに、CSP[5]に基づく Funclet+という独自言語で WSN の動作仕様を記述し、検証コードと実機ノードへの C コードを得る手法を提案した[6]。上記の研究は、形式検証手法を導入し、WSN の動作仕様の無矛盾性を網羅的に検証することに成功しているが、時間概念が扱えず遅延や時間制約に関する仕様を陽に記述できない。本稿では時間概念を拡張した CSP である Timed CSP[7]を導入し、従来の形式手法では、時間概念の欠如により扱えなかった通信エラー処理も検証可能とする。加えて、各ノードへの実行コードを入力仕様から直接得る環境の提案も行う。

2. 提案システム

図 1 に、提案システムの概要を示す。ユーザは、ノードの動作を記述した動作定義ファイルと環境設定ファイルを準備する。動作定義ファイルは、C 言語に似た独自言語で記述する。環境設定ファイルには、WSN 内のノード数、パケットロスの発生頻度、パケットロスに対するエラー処理などの起動タイミングを規定するタイマの設定などの環境オプションを記述できる。トランスレータは上記 2 つのファイルを入力とし、RTS 形式の検証用コード、実機ノードで動作させる C コードを生成する。RTS は Timed CSP に基づく形式検証ツール PAT (Process Analysis Toolkit)[8]の入力形式である。生成された RTS コードは入力動作仕様と動作環境を反映しており、それが PAT によって検証できれば、バグのない動作定義であることが保証できる。また、トランスレータにより生成された C コードは、その入力動作仕様を的確に表現しているため、コンパイル後に実機ノードへ安全に実装できる。

3. 通信モデルと通信エラーを考慮した通信ライブラリ

本稿で取り扱う通信モデルを図 2 に示す。データの送受信は、関数 get()/send()のみで行う。関数 get()は、送信元ノードが宛先ノードにデータを要求するときに用いる。関数 send()は、送信元ノードのデータを宛先ノードに送るときに用いる。get 通信では宛先ノードからの返答データが要求応答(ACK)そのものを兼ねる。図 2 において、通信エラー処理のタイムアウト処理は、パケットロスなどの通信エラーにより、予め定めた時間内に「(3)ACK 承認」処理が完了しなければ「(1)データ要求/データ送信」処理を繰り返す動作と定義する。上記動作を定義するには、時間概念が必要である。ここでは、各種条件に応じた get()/send()関数群をライブラリとして用意し、トランスレータが、ユーザの与えた環境設定ファイル内の記述に従って、適切な関数 get()/send()を選択し、生成する RTS コードに反映するようにした。図 3 に関数 send()の例を示す。1～2 行目は送信元ノードの振る舞いである。まず、1 行目で自身が保有するデータの送信と ACK 待ち状態への移行を行っている。2 行目は ACK 承認の処理と、承認が指定時間以内に完了しなかった場合の通信エラー処理であるタイムアウト及び再送を行っている。3 行目は宛先ノード側の処理で、受け取ったデータを内部に格納した後、ACK を送信元ノードに返信する一連の動作を記述している。

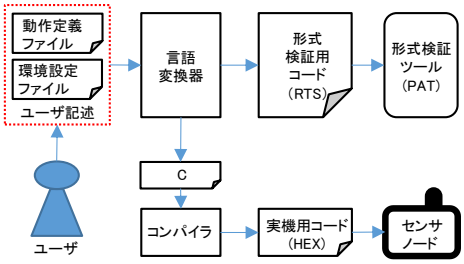

図 1 提案システムの概要

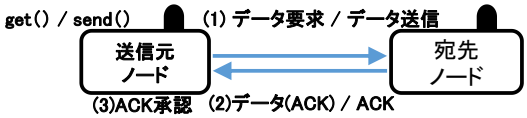

図 2 通信モデル

4. 形式検証及び実機ノードによる確認

具体的な例題を用いて提案システムの動作確認を行う。

```
//送信元ノードの振る舞い
1:   SEND(self,dest,func_id,param,time) =
         send!dest.self.func_id.param  ->>
         SEND_checkAck(self,dest,func_id,param,time);
2:   SEND_checkAck(self,dest,func_id,param) =
         (ack?self.dest.func_id ->> Skip;) timetout[time]
         SEND(self,dest,func_id,param,time);

//宛先ノードの振る舞い
3:   SEND_recieve(self) = send?self.from.func_id.param ->>
     dataput(func_id, param)  ->> ack!from.self.func_id ->> Skip;
```

図 3 通信エラーを考慮した通信ライブラリの一例

Verification of Wireless Sensor Network Considering Communication Errors Using Formal Method
†Akihiro Ikeda, †Toshiaki Miyazaki
†Graduate School of Computer Science and Engineering, The University of Aizu

4.1. 例題概要

例題の動作概要を図 4 に示す。防犯システムを想定し、起点ノードが人を検知した際には、検知状態に移行し、警戒状態を中継ノード、終点ノードへとマルチホップするものである。環境設定ファイルの中で WSN 内のノード数を 3 としている。1～8 行目の関数_MOTION()は人感センサに関する処理を規定している。関数_MOTION()は、環境設定ファイル内で、5 秒周期で起動するように設定している。2～7 行目は、この関数を起動した際、人感センサが反応していた場合に実行される処理である。2 行目で引数 value の値が 1(人感センサの反応あり)なら、3 行目で自身が起点ノードか確認する。起点ノードであれば、赤色 LED を点灯(4 行目)した後、5 行目で他のノードと通信する。9～14 行目は、関数 hopnext()の定義である。メッセージを受信した際、自身の黄色 LED を点灯し警戒状態に移行することと、自身が終点ノードでなければ次ノードの当該関数 hopnext()へアクセスを行う。この関数がアクセスを受け起動されると、まず、10 行目で黄色 LED を点灯して自身を警戒状態にする。次に、11～13 行目で、自身が終点ノードでなければ次ノードの当該関数 hopnext()にアクセスする。

4.2. 検証結果と実機ノードによる動作確認

4.1 で説明した動作に対し、3 番目の終点ノードは 0 から始まる配列の添字 2 で表され、LED の状態を持つ配列 leds[]において OFF、YELLOW が定義されているとき、絶対時間が 10 秒かつマルチホップの終点である 3 番目のノードが黄色 LED を点灯していない(警戒状態になっていない)場合が存在し得るかを問うアサーション式は以下のようになる。

current_time == 10 && leds[2] != YELLOW

もし、パケットロスのある環境でこの式が Valid と評価されれば、マルチホップ通信が失敗した際、ユーザの意図と異なる振る舞い(時間経過しても終点ノードが警戒状態ではない)が存在することを示す。NOT Valid と評価されればマルチホップ通信が失敗してもエラー処理により、最終的にユーザの意図した振る舞いを達成することを確認できる。PAT による検証結果を表1に示す。表1の結果より、パケットロスの発生する環境では終点のノードまで警戒状態が行き渡らない可能性があり、データを再送する通信エラー処理によってそれを回避できることがわかる。

図 5 に、トランスレータによって生成された C コードをコンパイルし、実機ノードで実行したときのスナップショットを示す。(A)は「通信エラー処理なし」、(B)は「通信エラー処理あり」の場合である。どちらの場合も、人感センサが反応し、左端の起点ノードが警戒状態をマルチホップした際、パケットロスによる通信エラーが発生した後の状態である。通信エラー処理を行わない(A)では、起点ノードは赤色 LED を点灯し検知状態だが、起点ノードから中継ノード(中央)への通信が失敗しているため、中継ノードと終点ノード(右端)はどちらも黄色 LED が点灯していない。一方、図 5(B)では、通信エラー処理、すなわち、環境設定ファイルで指定した秒数で、データの再送処理を行うので中継ノード、終点ノードともに黄色 LED が点灯し、情報が正しく伝搬していることがわかる。

5. おわりに

本稿では、センサノードの動作仕様を C 言語に類似した入力言語で入力すると、Timed CSP に基づいて、動作仕様を検証するとともに、実機センサノードで動作する検証済みの C コードも自動生成するシステムを提案した。本提案システムでは、形式検証で時間概念を扱えるようにしたことにより、従来法では困難であったパケットロス時の再送要求などのエラー処理の動作仕様の記述およびその形式検証も可能となった。

謝辞 本研究の一部は、総務省戦略的情報通信研究開発推進制度(SCOPE No.121802001)の支援を受けて実施したものである。

```
// SELFID       :自身のノード番号を示す特殊変数
// MAX_NODE 3   :ネットワーク内にノードがいくつあるかを表す定数
 1:   void _MOTION(int value){   // 人感センサを用いたイベント
 2:      if(value == 1){          // 人感センサに反応があれば
 3:         if(SELFID == 0){      // 自身が起点ノードなら
 4:         __LedRed__(1);        // 赤色 LED を点灯し人検知状態へ
 5:         sendList(hopnext, SELFID+1); // 自身の次のアドレスの
 6:         }                             // ホップネクスト関数に
                                          // 対して通信を行う
 7:      }
 8:   }
 9:   void hopnext(int from){       // 自身の次のノードに対し
                                    // 通信を行う
10:      __LedYellow__(1);         // 黄色 LED を点灯し警戒状態へ
11:      if(SELFID < MAX_NODE-1){  // 自身が終点ノードでなければ
12:         sendList(hopnext, SELFID+1); // 次ノードへアクセス
13:      }
14:   }
```

図 4 入力言語による動作定義

表 1 PAT による例題の検証結果

通信エラー処理	検証結果	検証結果の意味
なし	Valid	10 秒経過時点で終点ノードが警戒状態に移行しない場合あり
あり	NOT Valid	10 秒経過時点で終点ノードが警戒状態に移行しない場合なし

図 5 実機ノードによる動作確認

参考文献

[1] 後藤亮馬,和崎克,“UML-PROMELA 変換器を用いた ZigBeeIP/RPL プロトコルにおけるノード探索仕様の検証,” FIT2014, vol. 13, no. 1, pp. 159-162. 2014.

[2] G. Holzmann, “The SPIN Model Checker: Primer and Reference Manual,” Addison-Wesley, 2003.

[3] V. Oleshchuk, “Ad-hoc sensor networks: modeling, specification and verification. ,” 2nd IEEE Intertional Work on Intelligent Data Acquisition and Advanced Computing Systems Technology and Applications, pp. 76-79, Sept. 2003.

[4] 秋山直輝, 池田愛大, 宮崎敏明, ”C 言語で記述した無線センサネットワーク動作の形式検証法の提案 (An Approach to Formal Verification for Wireless Sensor Network Behavior Specified Using C Language),” 情報処理学会第 80 回全国大会(早稲田大学), 7S-01, March 2018.

[5] C.A.R. Hoare, “Communicating Sequential Processes,” Communications of the ACM, vol. 21, no.8, pp. 666-677, August 1978, DOI:10.1145/359576.359585.

[6] T. Miyazaki, N. Akiyama, “Formal approach to produce verified programs for wireless sensor nodes,” IEEE 10th International Symposium on Communication Systems, Networks and Digital Signal Processing (CSNDSP) ,. pp. 1-6, 2016.

[7] G. M. Reed and A. W. Roscoe, “A Timed Model for Communicating Sequential Processes,” Theoretical Computer Science, pp. 314–323, 1988.

[8] “PAT: Process Analysis Toolkit,” http://sav.sutd.edu.sg/PAT/ (Available 2018/01/08).