7S-02

無線センサネットワークにおける 柔軟な協調動作実現のためのルール拡張

久場有紗† 横田裕介†

日本女子大学†

1.はじめに

一般的な無線センサネットワークシステムでは，ホスト PC が全ノードからのデータを集約し，また各ノードの制御を行う．しかし，この手法だとホスト PC やネットワークにかかる負担が大きい．そこで本研究では，各ノードが能動的に自身の動作を決定可能なシステムを提案する．ノード同士の協調動作によってこれを実現するためには動作ルールを記述する必要があるが，本稿ではこのルールに変数機能を付与し，ルール記述を拡張する．また，変数機能を用いて実際のアプリケーションのルールを記述し，有用性を確認した．

2.ノード間での協調動作によるネットワーク

一般的な無線センサネットワークでは，ホスト PC やネットワークに負担がかかるという問題点がある．その解決策の一つとして，ノード同士の協調動作を利用し，ノード自身がデータ送信のタイミングやセンシング間隔などの次に行うべき動作を判断し，切り替える方法が挙げられる [1].

2.1 一般的な無線センサネットワークの課題

一般的なシステムの動作を図 1 に示す．まず全ノードがホスト PC へ向けてデータを送信する．ホスト PC からの距離が離れていて直接通信できない場合は，送信先を隣接ノードとし，バケツリレーのように運んでいく．データを受け取ったホスト PC は，それらを処理判断した後，各ノードへと制御命令を出す．命令に従い，ノードはそれぞれ動作を変更したり，維持したりする．この手法はホスト PC に大きな負担がかかるという問題がある．また，ノード数が増加した場合，ネットワークへの負荷も比例して増大する．

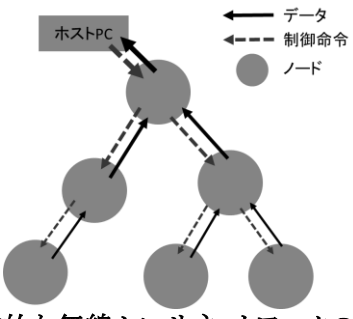

図 1. 一般的な無線センサネットワークの動作

Rule Extension for Flexible Cooperative Operation of Nodes in Wireless Sensor Networks

† Arisa KUBA, Yusuke YOKOTA, Japan Women's University

2.2 状態遷移による動作変更

2.1 で示した課題を解決するための提案手法について述べる．ノードには 2 種類以上の状態を用意しておき，センシング結果によって，必ずいずれかの状態を取ることにする．加えて，各状態ごとに，センシング間隔や観測データをホスト PC へと送信するタイミングなど，ノードが行う動作を予め定めておく．これにより，図 2 に示すように，ノードが状態遷移を行えば，それに伴ってノードの動作も変更される．

2.3 イベント通信に基づく状態遷移

ノードが取るべき状態はセンシング結果によって決定付けられるが，状態遷移を行うタイミングはイベント通信に基づく．イベントは合図のようなもので，これを受信したノードが状態遷移を行う．イベントの送信条件も予め各ノードに設定しておく．例を図 3 に示す，あるノード A のセンシング結果から，ノードの動作を変更する必要が生じたとき，ノード A はネットワーク内の隣接する他ノードにイベントを送信する．イベントを受信したノード B は状態遷移を行う．つまり，ノード B の動作は変更されたことになる．このように，ノード同士が協調動作することによって，自身の次の行動を自己決定することが可能となる．

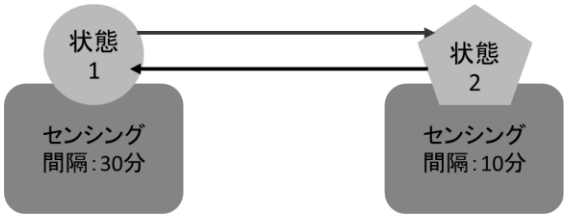

図 2. 状態遷移による動作変更の一例

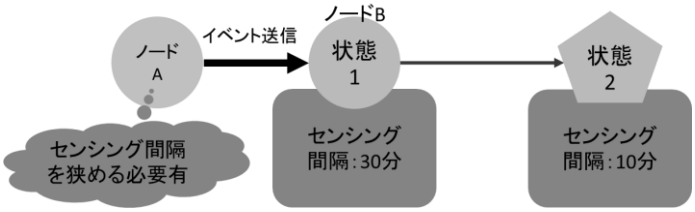

図 3. イベント送信に基づく状態遷移の一例

3.ルール処理部の拡張

2.2 および 2.3 で述べた動作は各ノードが持つルールとして記述される．このルール記述には，JSON (JavaScript Object Notation) というテキストベースのデータフォーマットを用いる．

ルールでは，ノードの状態の種類，各状態におい

ての動作，イベントの送信条件など，ノード同士での協調動作に必要な項目を定めている[2]．これを適宜，ルール処理部で参照してノードの動作を制御する．しかし，これまでのルール設計では，変数を用いるようなアプリケーションを考慮していない．そこで，変数も扱うことができるようにルール記述を拡張する．

3.1 変数機能の必要性

アプリケーションにおいて，過去のデータ保存が可能になれば，現在のデータとの比較や過去のデータの傾向から今後の予測などを行うことも可能になる．また，複数の状態間でデータを共有するためには，一時的な格納場所を確保し，どの状態からでも参照可能にすることが考えられる．よって，より幅広いアプリケーションを実現するためにも，変数機能は重要といえる．将来的には，過去のデータ保管はローカル変数，複数の状態間でのデータ共有ならグローバル変数と，目的に応じて変数の種類の使い分けも考えられると良い．

3.2 ルール内での記述

ルールは INITIAL，STATES，TRANSITION の三要素からなり，INITIAL はノードの初期状態を，TRANSITION はノードの状態遷移に関して定めている．今回はノードの各状態における動作を定める STATES の部分を拡張，改変した．図 4 はノードが状態 1 の場合の具体例である．STATES は更に，ノードの現在の状態を表す SID，センシングに関する Sensing，イベント送信に関する IF で構成される．変数とプログラム内で用意した配列とを結びつけるのはルール処理部で，Variable から変数 cur，max を参照し，配列の要素にそれぞれ割り当てる．以降，プログラム内で変数名を参照すれば，対応する配列の要素が自動的に呼び出されることになる．Condition では変数同士の比較，Substitution では代入のためにこの機能が用いられる．

```
{
  "INITIAL": "S1",
  "STATES": [
    {
      "SID": "S1",
      "Sensing": {
        "Interval": 1000,
        "Variable": {
          "cur": 0,
          "max": 1
        }
      },
      "IF": {
        "Condition": [
          "cur",
          "max",
          ">"
        ],
        "Action": {
          "SendEvent": {
            "Event": "E1",
            "Node": [
              "N1",
              "N2"
            ]
          },
          "Substitution": {
            "left": "max",
            "right": "max"
          }
        }
      },
    },
```

図 4. ルール拡張部分の一例

4.アプリケーションへの応用例

拡張したルール処理機構の有用性を示すために，1 日の中で 1 時間ごとに気温を計測し，その時点での最高値を通知するアプリケーションを作成する．

4.1 設計

このアプリケーションでは，観測値を格納するための変数と，その時点での最高値を格納するための変数の 2 種類を扱っている．

ノードとして Arduino M0 Pro に XBee を搭載したものを使用した．ネットワーク内には温度センサ搭載のセンシング用ノードと、LED 搭載の対ユーザ通知用ノードが存在する．センシング用ノードは午前 5 時から 1 時間に 1 回，気温を計測する．計測時に，これまで観測した気温の最高値と比較してそれを上回るようならば，最高値を更新する．最高値が更新された際に，センシング用ノードはユーザ通知用ノードへとイベントを送信する．ユーザ通知用ノードは，イベントを受信したら LED を点滅させるよう動作変更する． 24 時間が経過したら，最高値はリセットされる．

4.2 実装および動作確認

図 4 はこのアプリケーションのルール記述の一部となっている．今回はテストのため，センシング間隔を大幅に短縮して実装し，正常な動作を確認することができた．今後は，更に幅広いアプリケーションに対応できるよう，より普遍的なルール記述を目標とする．

5.おわりに

本研究では無線センサネットワークを用いるアプリケーションの幅を広げるために，ルール処理部を拡張し，変数機能を付与した．また，実際のアプリケーション上での実装を行った．

今後は，グローバル変数とローカル変数の使い分けや，更に複雑な状態遷移の条件でも対応できるよう機能拡張を検討したい．

参考文献

[1]冨森英生，横田裕介，大久保英嗣：ルールベースの問合せ処理機構による協調型センサネットワークの実現，情報処理学会研究報告，2009-MBL-48，pp.57-64(2009)

[2]吉田麻衣子，横田裕介：イベント通信に基づく協調型センサネットワークシステムの実装，第 78 回全国大会講演論文集，第 1 分冊，pp.223-224(2016)