情報処理学会研究報告
IPSJ SIG Technical Report

Vol.2010-SLDM-144 No.27
Vol.2010-EMB-16 No.27
Vol.2010-MBL-53 No.27
Vol.2010-UBI-25 No.27
2010/3/26

ユビキタスコンピューティング環境におけるルール処理機構の安全性について

佐野渉二[†1]　寺田努[†2]　塚本昌彦[†2]

ユビキタスコンピューティング環境では，周囲の状況を把握するセンサや，LED，ブザー，アクチュエータなどの入出力機器を用いて我々の行動が支援される．入出力機器を制御する方法の 1 つとして，筆者らの研究グループでは，ECA ルールというイベント駆動型ルールを用いて入出力機器を制御するデバイスを提案している．このデバイスでは，複数のルールを連鎖的に実行させることで複雑な処理を行える反面，意図しないルールの連鎖によってルールの無限連鎖が形成され，システムに不具合が生じることがある．本研究では，ルール実行の無限連鎖を検出する手法について検討する．

On Safety in Rule Processing Mechanisms in Ubiquitous Computing Environments

Shoji Sano ,[†1] Tsutomu Terada [†2]
and Masahiko Tsukamoto[†2]

In ubiquitous computing environments, various services are provided utilizing input/output control devices such as sensors, LED, buzzers, and actuators. As one of such devices, we have proposed a device for controlling input/output with event-driven rules called ECA rules. Although this device can execute complicated procedures by using several linked rules, a system may have problems because unexpected chains of rules make infinite loop of rules. In this paper, we propose a method that detects infinite loop of rules dynamically.

†1 神戸大学大学院自然科学研究科
Graduate School of Science and Technology, Kobe University
†2 神戸大学大学院工学研究科
Graduate School of Engineering, Kobe University

1. はじめに

環境に埋め込まれた多くのコンピュータを用いて，我々の行動を支援するユビキタスコンピューティング環境[1] に対する注目が集まっている．これを実現するため，センサ，LED，ブザーやアクチュエータなどの入出力機器を制御する方法の 1 つとして，ECA ルール[2] というイベント駆動型ルールを用いたデバイス[3),4)] がある．このデバイスでは，ルールの追加，削除などのコマンドにより動的にルールを交換できる．1 つ 1 つのルールは単純なものであるが，複数のルール実行を連鎖させることで複雑な処理も行える一方で，意図しないルールが連鎖することがある．この意図しないルール実行の連鎖のために，ルール実行の無限連鎖が形成され，システムに不具合が生じる．例えば，室温を調節するシステムで，部屋の温度が熱くなりすぎたり，明るさを調節するシステムで，部屋の明かりが点いたり消えたりを繰り返すなど不利益を被ることがあり，このような場合，システムを停止させるなどの処置を行う必要がある．

そこで本研究では，ルールの無限連鎖を検出する手法について考える．ルール実行の異常動作を検出する手法としては，これまでにアクティブデータベースとよぶ ECA ルールで動作するデータベースシステムにおいて，事前検出手法[5]，直前検出手法[6]，実行時検出手法[7] が提案されている．事前検出手法では，システムが稼働していない状態でルール実行の連鎖関係を表すトリガグラフと呼ばれる有効グラフを用いて異常動作を検出する．直前検出手法では，ネットワーク構成が変化する際に，動的に必要な情報だけをやり取りしてトリガグラフを再構築することにより異常動作を検出する．実行時検出手法では，システム稼働しながら，ルール実行の連鎖が起こるごとに連鎖数をカウントするルールカウンタや連鎖が起きている時間を計測するためのタイムスタンプを用いることで，リアルタイムに異常動作を検出する．しかし，事前検出手法や直前検出手法では，トリガグラフを構築する際にあるルールから次に連鎖するルールに対して有効パスをはることをすべてのルールに対して行う必要があり，処理能力が非力なデバイスを用いるユビキタスコンピューティング環境で用いるのは現実的でない，ルール実行時の検出手法については，本研究でも検討するが，アクティブデータシステムにおける研究では，異常動作はメッセージ伝播のみに起因するのに対し，ユビキタスコンピューティング環境では，出力機器の実行とそれに反応する入力機器の連鎖関係も考慮する点で環境が異なる．

以下，2 章ではルール処理機構について説明し，ルール処理機構における安全性の定義を行う．3 章ではルール実行の無限連鎖検出手法について提案し，4 章で提案手法に対してト

Vol.2010-SLDM-144 No.27
Vol.2010-EMB-16 No.27
Vol.2010-MBL-53 No.27
Vol.2010-UBI-25 No.27
2010/3/26

ラフィック量やストレージ量の影響について評価を行う．最後に 5 章でまとめる．

2. ルール処理機構の安全性について

2.1 ユビキタス環境におけるルール処理機構

ユビキタスコンピューティング環境において，センサやアクチュエータなどを制御するデバイスを用いたシステムを考える．本稿では，センサなどを入力機器，アクチュエータなどを出力機器と記し，単にデバイスと記す場合，それら入出力機器を制御するデバイスを指す．

本研究では，入出力機器を制御するために ECA ルールを用いたデバイスについて考える．ECA ルールは，イベント，コンディション，アクションの 3 つの構成により動作の記述を行い，記述の簡易さから，動作を動的に切り替えるシステムでよく用いられる．イベントとしては，メッセージの受信，入力機器によるデータの取得，コンディションはそのデータが満たす条件，アクションとしてはメッセージの送信，出力機器の動作設定などが挙げられる．

あるルールのアクションを起こすことにより，一定時間内に任意のルールのイベントが発火することを，ルール実行の連鎖という．ルール実行の連鎖には，メッセージの送信アクションによって他のデバイスの受信イベントが発火したり，電気をつけるアクションによって照度センサが反応するなど，アクションとイベントの関係を考慮に入れてルールを設定することでルール実行の連鎖が起こる意図した連鎖と，システム設計の際に考慮に入れていないルール実行の連鎖がシステム運用時に起こる場合の意図しない連鎖がある．意図しない連鎖は，ある出力機器を制御したときに予期しなかった入力機器が反する場合や複数のアプリケーションを 1 つのシステムで実現する際に他のアプリケーションで使用したルールを考慮せずにルールを設定する場合，または複数人で同じシステムを使用するとき，他の人が組み込んだルールを考慮せずにルールを追加する場合などに生じる可能性がある．

2.2 ルール実行の無限連鎖の例

照度センサによる照明の明るさ制御システムにおいて，図 1 のように，照明の明るさを制御するデバイス 1，照度センサデータを取得するデバイス 2 に対し，デバイス 1 にはルール 1，2，デバイス 2 にはルール 3 がそれぞれ格納されているとする．

- ルール 1：メッセージ ‘A’ を受信したら (イベント)，100[lux] 暗くする (アクション)．
- ルール 2：メッセージ ‘B’ を受信したら (イベント)，100[lux] 明るくする (アクション)．
- ルール 3：照度センサのメッセージを取得したときに (イベント)，500[lux] 以下であるとき (コンディション)，デバイス 1 にメッセージ ‘B’ を送信する (アクション)．

このとき，仮にルール 3 のイベントが発火する場合，ルール 3 のアクションにより，ルール 2 のイベントが発火する．これは，上述の意図したルール連鎖の例である．ここで，省エネを考え，照度センサのデータを取得するデバイス 3 を用いて，ルール 4 を追加する．

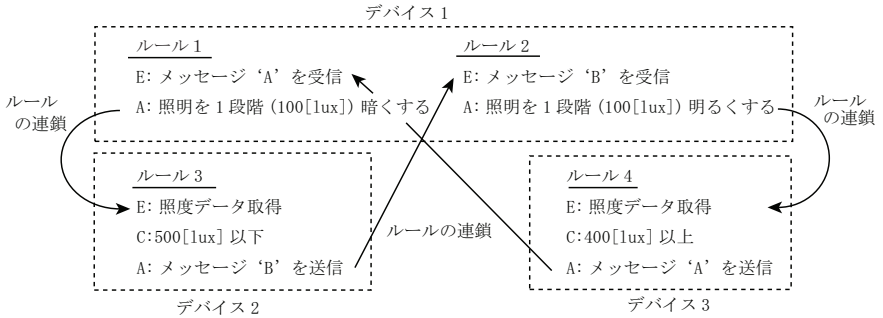

図 1 ルールの無限ループの例

- ルール 4：照度センサのデータが 400[lux] 以上になったとき (イベント)，デバイス 1 にメッセージ ‘A’ を送信する (アクション)．

例えば明るさが 550[lux] のとき，ルール 4 → ルール 1 → ルール 3 → ルール 2 → ルール 4 のようなルール実行の連鎖が無限に続いてしまう．このように意図しないルールの連鎖が行われることにより，システムが正常に動作しなくなる可能性がある．

2.3 ルール処理機構の安全性の定義

ルール実行の無限連鎖が起こる条件として以下の 2 つを考える．

条件 1. 一定回数以上ルール実行が連鎖する

条件 2. ルールの連鎖経路が 1 つ以上のループを形成する

この 2 つの条件を同時に満たさないとき，ルール実行の無限連鎖が起きていないと判定し，ルール処理機構が安全であるものとする．

3. ルールの無限連鎖検出手法

3.1 想 定 環 境

ユビキタス環境において，ルール実行の無限連鎖およびその検出手法を考える上で関係する項目を挙げる．

- 入出力機器の出力と入力の関係
  出力機器の制御に伴い，反応する入力機器との関係を考慮する必要がある．例えば，入

Vol.2010-SLDM-144 No.27
Vol.2010-EMB-16 No.27
Vol.2010-MBL-53 No.27
Vol.2010-UBI-25 No.27
2010/3/26

力機器として照度センサを考える場合，照明器具を制御することで照度センサが反応する．これ以外にも，昼間，アクチュエータを制御しカーテンを開けることで光が差し込み，照度センサが反応する場合，アクチュエータの制御と照度センサ間でルール実行の連鎖が起こるなど，出力機器と入力機器に関連がなさそうな場合にも関連する場合がある．

- 出力機器の影響範囲
  シーリングライトなど全体照明の照明器具では部屋全体に光照射の影響を及ぼすが，スポットライトなど部分照明の照明器具では一定の範囲にしか影響を及ぼさない．このように出力機器に応じて影響を及ぼす範囲が異なる．
- デバイスの位置関係
  前項の出力機器により影響を及ぼす範囲が異なるため，入力機器，出力機器の位置関係によりルール実行の連鎖関係が変わる場合がある．例えば，部分照明の照明器具を用いる場合，フォトトランジスタの位置により反応するかしないかが決定する．特に入力機器，または出力機器が可動な場合は，常にその位置関係が変わる．
- 通信ネットワーク
  デバイス間のメッセージ通信により，ルール実行の連鎖が起きる場合に影響する．また，ルール実行の連鎖が生じたときにその様子を把握するには，デバイス間のメッセージとしてその情報を送る必要がある．そのため，すべてのデバイス間で通信が可能かどうかにより検出手法にも影響すると考えられる．

これらに対して本稿では，

- すべてのルール実行の連鎖関係があらかじめ分かっている
- すべてのデバイスが同じ通信ネットワークに接続されている

環境を想定する．つまり，入出力機器の出力と入力の関係，出力機器の影響をおよぼす範囲，デバイスの位置関係は既知であり，システム稼働中にデバイスの位置関係は変化しない．通信ネットワークについては，任意の 2 つのデバイスにおいて常にメッセージの送受信を行えるシステムとする．

3.2 提案手法

ルール実行の無限連鎖を検出手法は，ルール実行のサブシステムとして稼働するため，ルール実行に影響がないように動作すべきである．条件 1. については，ルールの連鎖が起きるごとにカウントアップするルールカウンタを伝播させることで判定する．条件 2. を検出するための手法として，ルール実行の無限ループを検出するために，オブジェクト伝播法，メモリ格納法を提案する．両手法ともイベントが発火し，コンディションが成立後，アクションを実行する前に行う．

各デバイスに付与する ID を n，デバイス n に格納されている m 番目のルールをを R_m^n とし，ルール実行の連鎖経路をルール ID の集合で表す．例えば，ルール R_1^1, R_1^2, R_1^3 の順でルール実行の連鎖が起きた場合，このルール実行の連鎖経路を $C_k = \{R_1^1, R_1^2, R_1^3\}$ と表す．ここで，k はルール実行の連鎖を区別するための ID である．以下では，ルール R_1^1 のイベントが発火して始まったルール実行の連鎖が，ルール $R_0^1, R_0^2, \cdots, R_0^{n-1}$ の順に伝播し，ルール R_0^n のイベント，コンディションが成立した際のルール実行の無限連鎖検出手法について説明する．

オブジェクト伝播法

オブジェクト伝播法では，ルールカウンタ，連鎖 ID，ルールの連鎖経路の組をオブジェクトとして，ルールの連鎖が生じるごとに伝播させる．ルールカウンタが設定値以上になったとき，このデータを参照することで連鎖経路にループが生じているかを判定する．つまり，オブジェクト伝播法では，まず，デバイス $n-1$ に問い合わせて，ルールカウンタ，ルール実行の連鎖経路 $\{R_0^1, R_0^2, \cdots, R_0^{n-1}\}$ を取得する．次に，ルールカウンタが設定値を越えていなければ，アクションを実行し，ルールカウンタが設定値を越えていれば，連鎖経路からループが形成されているか判定し，ループが検出されていなければ，ルール R_0^n のアクションを実行する．

メモリ格納法

メモリ格納法では，連鎖 ID，ルールの連鎖経路の組を各デバイスのメモリに格納する．ルールカウンタが設定値以上になったとき，ルール実行連鎖のルールを逆順にたどり，連鎖経路にループが生じているかを判定する．つまり，メモリ格納法では，まず，デバイス $n-1$ に問い合わせて，ルールカウンタ $n-1$，ルール実行の連鎖経路 $\{R_{n-1}^{n-1}\}$ を取得する．次に，ルールカウンタが設定値を越えていなければ，アクションを実行し，ルールカウンタが設定値を越えていれば，デバイス $n-1$ に連鎖 ID，ルール実行の連鎖経路の組 $\langle C_k, R_0^{n-1}\rangle$ を送信する．さらにデバイス $n-1$ では，自身の連鎖 ID，ルール ID を加えたルール実行の連鎖経路の組 $\langle C_k, \langle R_0^{n-1}, R_0^{n-2}\rangle\rangle$ を 1 つ前に連鎖経路であるデバイス $n-2$ に送信する．これをはじめにイベントが起こったルールを持つデバイスまで送信することを繰り返し，得られたルール実行の連鎖経路の組 $\langle C_k, \langle R_0^{n-1}, R_0^{n-2}, \cdots R_0^0\rangle\rangle$ からループが形成されているか判定し，ループが検出されていなければ，ルール R_0^n のアクションを実行する．

Vol.2010-SLDM-144 No.27
Vol.2010-EMB-16 No.27
Vol.2010-MBL-53 No.27
Vol.2010-UBI-25 No.27
2010/3/26

表 1 ルール実行の無限連鎖検出手法

	無限ループ検出手法	デバイスカウンタ
手法 1	オブジェクト伝播法	未使用
手法 2	メモリ格納法	未使用
手法 3	オブジェクト伝播法	使用
手法 4	メモリ格納法	使用

表 2 評価パラメータ

パラメータ	基本値
デバイス数	30
1 デバイスがもつ平均ルール数	5
コンディションの最小確率	80
イベントの自然発火数	3
ルールカウンタ	15

デバイスカウンタ

デバイスに連鎖 ID，ルール ID，ルールカウンタの組を保持するデバイスカウンタを設ける．デバイスカウンタを設けることで，ルールの連鎖経路を求める際に，同じデバイスが 2 回以上通る場合はすぐに検出できる．

4. 評　　価

表 1 に示すルール実行の無限連鎖を検出する 4 つの手法に対して，シミュレーション評価を行った．シミュレータは Windows XP 上で Visual Studio C#.NET 2005 を用いて作成した．シミュレータでは，デバイスやルールおよびそれを構成するイベント，コンディション，アクションは，すべて ID で区別する．ルール実行の連鎖関係は，各アクション ID において，反応するイベント ID を対応させることにより設定する．また，1 つのデバイスがもつ平均ルール数，ルール実行の最大連鎖数，平均連鎖数を設定することで，それを満たすルール群を各デバイスに分布できる．本評価では，ルール実行の無限連鎖を検証する過程におけるトラフィック量とストレージ量について評価を行うため，無限ループが起こらないようにルールを設定した．

シミュレーションで考慮するパラメータを表 2 に示す．コンディションについては，実際にはセンサ値などで成立するか判断するが，シミューレーションにおいては確率的に成立するものとし，コンディションの最小確率をパラメータとした．例えば，コンディションの最小確率が 80% のときは，ルール生成時にコンディションが成立する確率を 80〜100% のうちでランダムに決定する．自然発火イベントは，カーテンやドアを開けるなど，人の動作により発火するイベントや誤作動により発火するイベントを考慮したものであり，一定時間にイベントが一定数発火するものとする．評価において変化させるパラメータ以外は表 2 の値を使用する．連鎖 ID，ルール ID など一つのデータを送る，または格納するデータ量を 1 とし，一定時間行ったものを 100 セット行い，その平均値を計算した．

4.1 評 価 結 果

ルール実行の無限連鎖検出手法とトラフィック量と必要ストレージ量の関係を調べるために，表 2 のパラメータをそれぞれ変化させた場合の評価を行った．デバイスカウンタは，無限ループが起こっている場合に素早く検出するためのものであり，無限ループが起こらないようにした本評価では，デバイスカウンタの使用の有無によりトラフィック量は変化しない．このため，トラフィック量に関しては，オブジェクト伝播法，メモリ格納法についてのものを示す．

デバイス数を変化させた場合の結果を図 2 に示す．横軸はデバイス数，縦軸は各デバイスにおける平均トラフィック量または最大ストレージ量である．トラフィック量，ストレージ量については，デバイス数の変化によらずほぼ一定の値を示した．また，トラフィック量については，ルール実行の連鎖が比較的少ない環境では，メモリ格納法の方が少ないが，ルール実行の連鎖が比較的多い環境では，オブジェクト伝播法の方が少ない．ストレージ量については，手法 1，手法 2，手法 3，手法 4 の順で多くなる．また，デバイスカウンタを設けることで，ストレージが一定量増えることが確認できる．なお，1 つのデバイスがもつ平均ルール数を変化させた場合も同様の結果を示したため，記載を省略する．

コンディションの最小確率，自然発火イベント数の変化による結果をそれぞれ図 3，図 4 に示す．横軸はそれぞれコンディションの最小確率，自然発火イベント数であり，縦軸は各デバイスにおける平均トラフィック量または最大ストレージ量である．コンディションの最小確率，または自然発火イベント数が大きくなるに従い，トラフィック量，ストレージ量は単調増加することが分かる．各手法とトラフィック量，ストレージ量の大小関係については，デバイス数を変化させた場合と同じ順序となった．

ルールカウンタの設定値に対する結果を図 5 に記す．横軸はルールカウンタの設定値，縦軸は各デバイスにおける平均トラフィック量または最大ストレージ量である．トラフィック量については，ルールカウンタの設定値を小さく設定する場合，ルールの連鎖経路を調べる回数が多くなり，オブジェクト伝播法の方がよいが，ルールカウンタの設定値を大きく設定

Vol.2010-SLDM-144 No.27
Vol.2010-EMB-16 No.27
Vol.2010-MBL-53 No.27
Vol.2010-UBI-25 No.27
2010/3/26

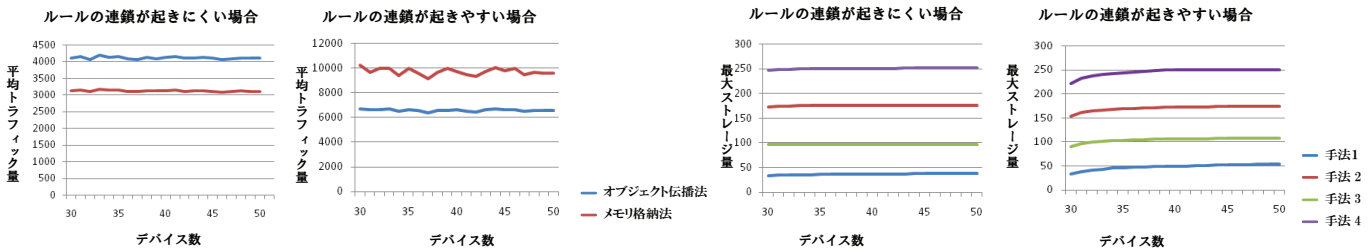

図 2 デバイス数に関する評価結果

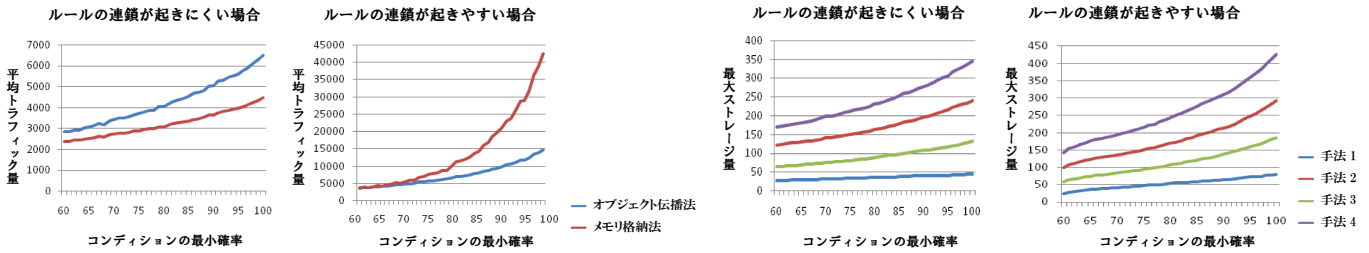

図 3 コンディションの最小確率に関する評価結果

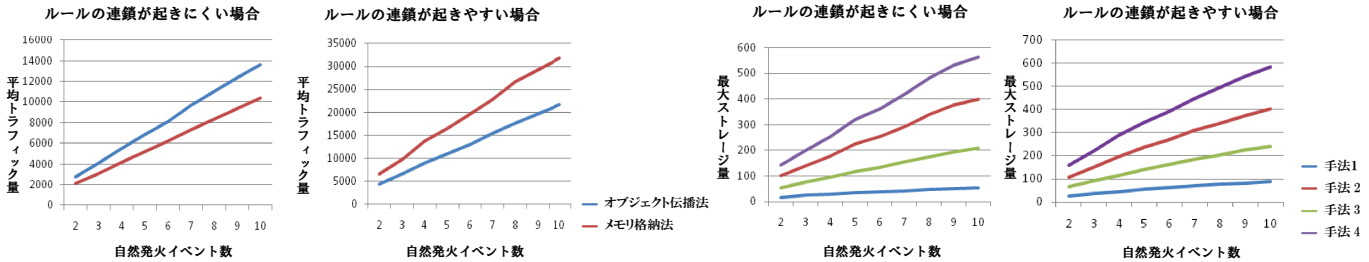

図 4 自然発火イベントに関する評価結果

する場合は，メモリ格納法がよいことが分かる．

5. 考　　察

5.1 ルール実行の無限連鎖の検出能力

本研究におけるルール処理実行の安全性の定義では，すべてのルール実行の無限ループを検出することを目指したものであるため，ユーザが意図して起こしたルールの無限ループも検出する．しかし，ユーザが設定するルールに人為的ミスが含まれている場合を考慮すると，設定されたルールから，どのルールの連鎖を意図しているかどうかは判別できない．このため，ルール実行の意図したループを関連する各デバイスにあらかじめ登録しておき，無限ループが検出された場合にそれを構成するルールの連鎖がすべて意図するものかどうかを判定する機構を組み込むことにより，ユーザが意図しないルールの無限ループのみを検出することができると考えられる．

Vol.2010-SLDM-144 No.27
Vol.2010-EMB-16 No.27
Vol.2010-MBL-53 No.27
Vol.2010-UBI-25 No.27
2010/3/26

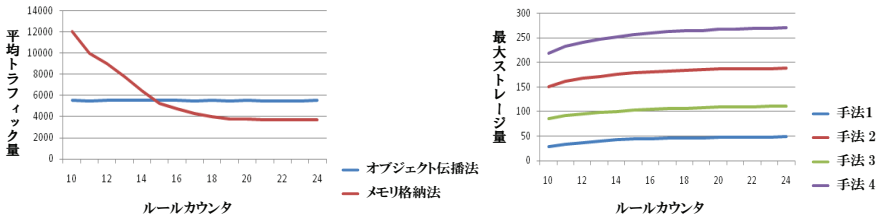

図 5 ルールカウンタに関する評価結果

5.2 ルールカウンタの決定指針

無限連鎖の検出手法を用いる際にはルールカウンタを設定する必要がある．ルールカウンタを小さくする場合には，ルールの連鎖が少ないときから無限ループが生じているかを検証するため，ルール実行のループが生じたときに素早く検出できる．しかし，無限ループの検証回数が多くなるため，トラフィック量は増加する．このため，使用する出力機器に無限ループが起こると危険を及ぼすものが含まれている場合には，ルールカウンタを小さく設定したり，メッセージを送信する際に，中継デバイスが多いマルチホップ通信を使用する場合はルールカウンタを少し大きく設定したりするなど，システムの安全性とトラフィック量を考慮した値を設定することが望ましい．

5.3 ルール実行無限連鎖検出手法の決定指針

ユビキタス環境で入力機器を制御するデバイスは設置場所を限定しないために，メモリ量などの制約がある．そのため，ルール実行無限連鎖検出手法において，デバイスのストレージ量を少なくしたい場合はオブジェクト伝播法の使用が有用である．一方，デバイスのメモリ量にそれほど制約がない場合は，トラフィック量が少なくなるように手法を使い分ける．つまり，トラフィック量については，ルール実行の連鎖数とルールカウンタ値に関係するため，ルール実行の連鎖数に対して，ルールカウンタを大きく設定する場合はメモリ格納法，小さく設定する場合はオブジェクト伝播法が適すると考えられる．

6. まとめと今後の課題

本稿では，ルール処理機構の安全性を確保するためにルールの無限連鎖を検出する手法について述べた．本手法を用いることで，ユビキタスコンピューティング環境においてルールの異常動作を検出できる．今後の課題としては，ネットワークが一部で分断されていたり，ルールの連鎖関係があらかじめ分からない状況などでのルールの無限連鎖検出手法について考える予定である．また，実機で使用することにより，提案手法がルール実行に影響を与えないことを示す．

謝辞 本研究の一部は，日本学術振興会科学研究費補助金基盤研究 (A) “ユビキタス環境のための全体プログラミング方式”(20240007) の研究助成によるものである．ここに記して謝意を表す．

参 考 文 献

1) M. Weiser：“The Computer for the Twenty-first Century,” *Scientific American*, Vol. 265, No. 3, pp. 99–104, 1991.
2) J. Widom, S. Ceri：“Active Database Systems,” *Triggers and Rules for Advanced Database Processing*, Morgan Kaufmann Publishers, 1996.
3) T. Terada, M. Tsukamoto, K. Hayakawa, T. Yoshihisa, Y. Kishino, A. Kashitani, and S. Nishio：“Ubiquitous Chip: a Rule-based I/O Control Device for Ubiquitous Computing,” In *Proceedings of 2nd International Conference on Pervasive Computing (Pervasive 2004)*, pp. 238–253, 2004.
4) S. Sano, T. Yoshihisa, and M. Tsukamoto: “Design and Implementation of Device with Alterable Functions for Ubiquitous Computing,” In *Proceedings of International Symposium on Ubiquitous Multimedia Computing (UMC2008)*, pp. 226–231 (2008).
5) 村瀬 亨, 塚本 昌彦, 西尾 章治郎: “移動体計算環境におけるアクティブデータベースの安全性解析手法,” 情報処理学会論文誌, Vol. 43, No. 12, pp. 3785–3793 (2002).
6) 寺田 努, 塚本昌彦, 西尾章治郎: “移動体計算環境におけるアクティブデータベースの動的トリガグラフ構築機構の設計と実装,” 情報処理学会論文誌（データベース）, Vol. 43, No. SIG12(TOD16), pp. 52–63 (2002).
7) 寺田 努, 莫 君, 村瀬 亨, 塚本 昌彦, 西尾 章治郎: “移動体計算環境におけるアクティブデータベースの ECA ルール実行監視機構の設計と実装,” 情報処理学会研究報告 (データベースシステム研究会 99-DBS-119), Vol. 99, No. 7, pp. 369–374 (1999).