1M1-OS-02a-3

スマートオブジェクトの近接連携シナリオへの記号モデル検査の応用

Applying Symbolic Model Checking to Scenario Verification of Proximity-based Federations among Smart Objects

蓑田　玲緒奈　　湊　真一
Reona Minoda　Shin-ichi Minato

北海道大学　大学院情報科学研究科
Graduate School of Information Science and Technology, Hokkaido University

Verification of ubiquitous computing (UC) scenarios enables us to detect design-related faults of UC applications in advance before we actually implement them. In this paper, we propose a verification framework of CCRN, which is a description model of UC scenarios, as a new application field of symbolic model checking. To do so, we illustrate a method how to transform a scenario written in CCRN into a symbolic model checking problem. We also show experimentally that our framework make it possible to verify large scale UC scenarios which could not be verified in realistic time by our previous method.

1. はじめに

ユビキタス・コンピューティング (UC) では，様々なモノが連携し合う．これらのモノを本論文でスマートオブジェクト (Smart Object, SO) と呼ぶ．PC，携帯電話，組み込みコンピュータなどの計算や通信機能を有するデバイスはもちろん SO と見なすことができるが，食べ物や食器，コップなど一見すると計算や通信機能を有していないモノも，RFID タグなどを貼り付けてしまえば，SO と見なすことができる．本論文でフェデレーションとは複数の SO が近づいた時に起こる現象を指すことにする．フェデレーションの例としては「薬の瓶と飲み合わせの悪い食べ物が近づいたときに，ユーザーの持っているスマートフォンで注意を促す」といったことが考えられる．本論文で扱う UC シナリオとは複数の連携が介在する現象のことである．田中と Julia らによって UC シナリオを記述する触媒反応ネットワークモデル [Julia 16] が提案されたが，形式検証する手法が確立されておらず課題となっていた．これに対して筆者は，触媒反応ネットワークモデルに対してセグメントグラフと呼ばれる要素を導入した Context Catalytic Reaction Network (CCRN) を提案し，CCRN のモデル検査への変換手法を提案することによって，UC シナリオをモデル検査で検証することが可能になった [Minoda 16]．UC シナリオの検証は様々な UC アプリケーションの設計上の問題を，実装する前に予め見つけることを可能にした．しかしこの手法はスケーラビリティの向上が課題であった．本論文では，CCRN の検証を効率化するために記号モデル検査を用いた手法を提案する．

2. 準備

本節では，以後の説明に必要な定義，表記，要素について説明する．

2.1 基本的な定義と表記

X と Y を任意の集合とするとき，$X \cup Y$，$X \cap Y$，$X \setminus Y$ をそれぞれ X と Y の和集合，積集合，差集合とする．集合 X について，2^X を X の冪集合（すなわち，全ての部分集合）とし，$|X|$ を X の要素数とする．また，集合族（すなわち，集合の集合）M について，M に含まれる全ての集合の和集合と積集合をそれぞれ $\bigcup M$，$\bigcap M$ と表記する．

2.2 触媒反応ネットワーク

触媒反応ネットワークは元々生物の分野でタンパク質の代謝を分析する目的で Stuart Kauffman によって提案されたものである [Kauffman 02]．このモデルを基づいて，田中はこれを SO の複数のフェデレーションが相互に関わる UC シナリオを記述する方法として UC の分野に適用した [Tanaka 10]．本論文では，触媒反応ネットワークは後者を指すものとする．

触媒反応ネットワークは触媒反応の集合である．各々の触媒反応は複数の物質を入力としてそれを別の複数の物質に変換し，それを出力とする．また，各々の触媒反応は *context* と呼ばれる触媒を持つ．また，入力の物質に触媒を含ませることもでき，このような種類の触媒を *stimulus* と呼ぶ．触媒反応は全ての必要とされる SO が互いに近傍に位置するときに起こる．ここで SO の近傍領域の内側を *scope* という言葉で表現することにし，SO o の scope は SO o から通信できる SO の集合として表現する．文献 [Tanaka 10] では，全ての context と SO の scope が考慮されているが，本論文では context の scope のみを考慮するものとする．言い換えると，全ての必要な SO が context の scope に入った時に，その context に対応する触媒反応が起こるものと本論文では扱う．

図 1 は単一の触媒反応の例である．この触媒反応では，context としてゲート c_1 があり，ユーザーは電話 a，ヘッドセット b，IC カード c の 3 つの SO を持っている．ユーザーが c_1 の scope に入ると，c_1 は a と b に連携するように働きかける．この動作は s がきっかけとして起こる．この動作の後，電話 a とヘッドセット b はフェデレーションを形成する．本論文では，a と b によるフェデレーションを a と b の連結（すなわち，ab）で表現する．この動作では c_1 と s は触媒として働いている．特に s はこの触媒反応の stimulus である．この反応を図 1 の右側のように表記する．

触媒反応ネットワークには，図 2 に示すような 4 種類の触媒反応がある．4 種類の触媒反応は大きく 2 つのグループに分けられ，1 つは合成反応のグループ（図 2 (i) と (ii)），もう 1 つは分解反応のグループ（図 2 (iii) と (iv)）である．図 1 の触媒反応は図 2(i) の種類に属している．また，図 2 (ii) のような stimulus がない触媒反応も考えることができる．図 2 (ii) では，SO a と SO b を持ったユーザーが stimulus なしで context c_2 の scope に入ることだけをきっかけに，c_2 は a と b に連携するように働きかける．同じようにして図 2 (iii) や (iv) のような分解反応も考えることができる．図 2 (iii) のような種類の反応では，ユーザーが ab というフェデレーションを形成している SO a と b とさらに SO s を持って context c_3 の scope に入ると，c_3 は stimulus s をきっかけとしてフェデレーション ab を分解する．図 2 (iv) の反応は図 2 (iii) の stimulus がない反応である．

触媒反応の出力となる SO は，stimulus として他の触媒反応を促進したり，他の触媒反応の入力の SO となったりする．このようにして，複数の触媒反応により触媒反応ネットワーク

連絡先: 北海道大学大学院情報科学研究科
〒 060-0814 札幌市北区北 14 条西 9 丁目
E-mail: minoda@meme.hokudai.ac.jp

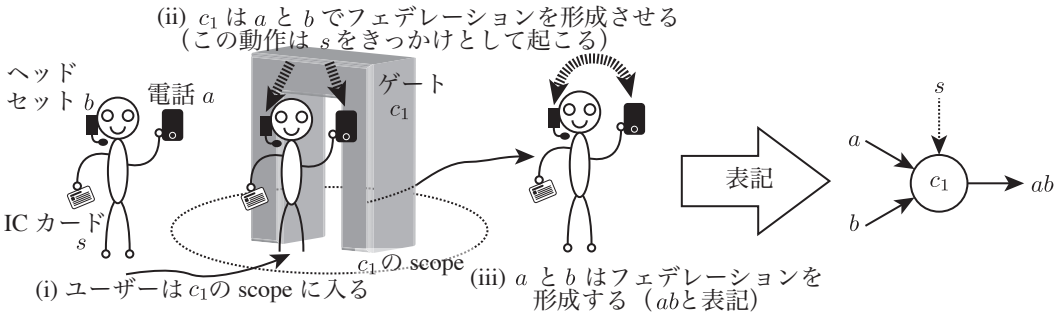

図 1: 触媒反応の例

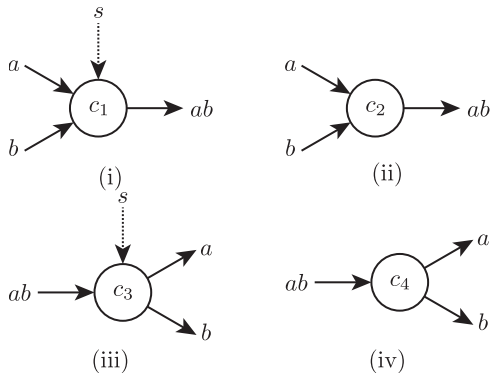

図 2: 4 種類の触媒反応

を形成する．

ここで，触媒反応ネットワークを形式的に定義する．まず O を SO の集合とする．フェデレーションを形成している SO o_f を $o_f \in 2^O \setminus \emptyset$ where $|o_f| > 1$ と定義する．なお，$|o_f| = 1$ の時は o_f はフェデレートしていない単一の SO とみなせる．次に触媒反応を以下のように定義する．

定義 1 (触媒反応) O と C をそれぞれ *SO* と *context* の集合とする．触媒反応はタプル (c, M, N) と定義される．ただし，

- $c \in C$, $M \subseteq 2^O \setminus \emptyset$, $N \subseteq 2^O \setminus \emptyset$
- $\forall o_f \forall o'_f \in M.(o_f \neq o'_f \rightarrow o_f \cap o'_f = \emptyset)$
- $\forall o_f \forall o'_f \in N.(o_f \neq o'_f \rightarrow o_f \cap o'_f = \emptyset)$
- $\bigcup M = \bigcup N$
- $(|M \cap N| + 1 = |N|, |M| > |N|) \vee$
  $(|M \cap N| + 1 = |M|, |M| < |N|)$ $(*)$

である．

なお，(*) で示した最後の条件の前半と後半部分はそれぞれ合成反応と分解反応で満たすべき条件を表している．

ここで，この定義を用いた触媒反応の例を与える．$C = \{c_1, c_3\}, O = \{a, b, s\}$ とすると，図 2 (i) と (iii) の触媒反応はそれぞれ $(c_1, \{\{a\}, \{b\}, \{s\}\}, \{\{a, b\}, \{s\}\})$, $(c_3, \{\{a, b\}, \{s\}\}, \{\{a\}, \{b\}, \{s\}\})$ と定義することができる．

最後に触媒反応ネットワークを以下のように定義する．

定義 2 (触媒反応ネットワーク) 触媒反応ネットワークは触媒反応の集合である．

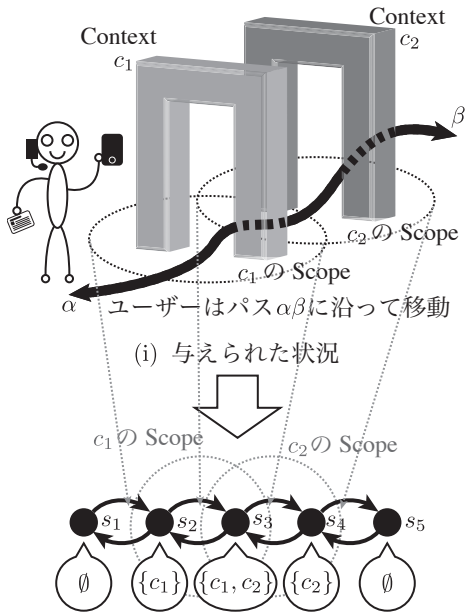

図 3: セグメントグラフの例

2.3 Context Catalytic Reaction Network

前節で述べたように，触媒反応は必要とされる SO が全て対応する context の scope に入ると起こる．触媒反応ネットワークの性質を状態遷移系として分析するためには SO の移動という現象を形式化する必要がある．例として，図 3 (i) では，context c_1 と c_2 があり，それらの scope は重なっている部分がある．ユーザーは図 3 (i) に示したようなパス $\alpha\beta$ に沿って移動できる．このような状況をセグメントグラフを用いて図 3 (ii) のように表現できる．ユーザーはセグメントグラフ上を移動するものとして，ユーザーは常にセグメントグラフのいずれかの頂点に位置するものとする．セグメントグラフの各頂点にはその場所に存在する context の scope の集合が対応づけされている．このようにして，図 3 (i) のように context の scope に重なりがあったとしても離散構造として表現できる．セグメントグラフは以下のように定義される．

定義 3 (セグメントグラフ) C を *context* の集合とする．セグメントグラフ G はタプル (S, E, F) である．ただし，

- S はセグメントの有限集合
- $E \subseteq S \times S$ は *2* つのセグメント間の有効辺
- $F : S \rightarrow 2^C$ はセグメントに対応する *context* の *scope* の集合が返る関数

である．

Context Catalytic Reaction Network (CCRN) [Minoda 16] は，触媒反応ネットワークに関係する SO の状況を表現する離散構造である．CCRN はセグメントグラフと触媒反応ネットワークの組み合わせで定義される．

定義 4 (Context Catalytic Reaction Network) *Context Catalytic Reaction Network (CCRN)* はタプル $(O, C, R, G, L_{FIX}, l_0)$ である．ただし，

- O は SO の集合
- C は *context* の集合
- R は触媒反応の集合（すなわち，触媒反応ネットワーク）
- G はセグメントグラフ (S, E, F)
- $L_{FIX} \subseteq O \times S$ は固定された SO の位置
- $l_0 \in S$ は移動する SO の初期に位置するセグメント（移動する SO は $O \setminus \{o \in O \mid \exists s \in S.((o, s) \in L_{FIX})\}$ と表せる）

である．

2.4 モデル検査

モデル検査は状態遷移系の性質を検証する手法である．モデル検査では状態遷移系としてクリプキ構造 [Kripke 63] が用いられる．クリプキ構造の性質は様相論理と呼ばれる論理によって記述される．様相論理は主に計算木論理 (computational tree logic, CTL) と線形時相論理 (linear temporal logic, LTL) の 2 つが用いられることが多いが，本論文では LTL をクリプキ構造の性質を記述する論理として用いる．

モデル検査問題は，直感的には，あるクリプキ構造 M が与えられた様相論理式 ϕ を満たすかどうかを判定する判定問題である．モデル検査問題はクリプキ構造 M の状態数が有限の場合はグラフ探索問題に帰着できることが知られているが，大規模なクリプキ構造に対してモデル検査すなわちグラフ探索を行うのはコストがかかる．そこで，クリプキ構造に含まれる全状態や遷移関係を明示的に保持せず，これらをブール関数で表現し，これを二分決定グラフ (BDD) でコンパクトに格納したものを用いて大規模な状態遷移系を効率的にモデル検査を行う**記号モデル検査**が McMillan らによって提案された [Burch 92]．記号モデル検査では，任意の状態集合 S を変数ベクトル $\boldsymbol{s}$ を用いて以下のブール関数 $S(s)$ で表現する．

$$S(\boldsymbol{s}) = \begin{cases} \text{True} & \boldsymbol{s} \in S \\ \text{False} & \text{otherwise} \end{cases}$$

これを用いて初期状態 $I \subseteq S$ を表すブール関数も同様に $I(s)$ として表現できる．さらに，状態 $\boldsymbol{s} \in S$ から $\boldsymbol{s}' \in S$ への遷移関係 $R \subseteq S \times S$ を以下のブール関数 $T(\boldsymbol{s}, \boldsymbol{s}')$ として表現する．

$$T(\boldsymbol{s}, \boldsymbol{s}') = \begin{cases} \text{True} & (\boldsymbol{s}, \boldsymbol{s}') \in R \\ \text{False} & \text{otherwise} \end{cases}$$

これらは実際に記号モデル検査を行う検査器の内部では BDD として保持されているが，記号モデル検査問題に帰着させるには BDD を直接検査器に与える必要はなく，状態を表す変数ベクトル $\boldsymbol{s}$，$S(\boldsymbol{s})$，$I(\boldsymbol{s})$，$T(\boldsymbol{s}, \boldsymbol{s}')$ をブール関数として与えればよい．記号モデル検査を行う実装として，NuSMV2 (**New** **S**ymbolic **M**odel **V**erifier version **2**)[Cimatti 02] が有名である．

3. CCRN の記号モデル検査による検証

本節では，CCRN を記号モデル検査の問題に帰着する手法を提案する．CCRN $(O, C, R, (S, E, F), L_{\text{FIX}}, l_0)$ で保持される状態には，L_{FIX} で特定のセグメントに固定されておらず，ユーザーが持ち運ぶ SO $O_{\text{MOB}} = O \setminus \{o \in O \mid \exists s \in S.((o, s) \in L_{\text{FIX}}\}$ がどのセグメント $s \in S$ に位置するかという状態と，$2^{|O|} - 1$ 通りのフェデレーション o_f が存在するか否かという状態の 2 種類がある．前者は $|S|$ の状態数があり，後者は $2^{2^{|O|}-1}$ の状態数があり，仮に明示的に全状態数を数え上げると $|S| \times 2^{2^{|O|}-1}$ という膨大な状態数になる．まず記号モデル検査の問題に帰着させるための最初のステップとして状態を表す変数ベクトル $\boldsymbol{s}$ を定義する．先の議論から，O_{MOB} がセグメント $s \in S$ に位置する状態を $segment = s$ と表現し，フェデレーション o_f が存在する状態を $fed(o_f) = \text{True}$ と表現する．これらを用いて状態ベクトル $\boldsymbol{s}$ を

$$\boldsymbol{s} = (segment, \underbrace{fed(o_f), fed(o_f'), \cdots}_{2^{|O|}-1})$$

と定義することができる．全状態集合を表す $S(\boldsymbol{s})$ は本論文では任意の $\boldsymbol{s}$ について True であるようなブール関数とし，$T(\boldsymbol{s}, \boldsymbol{s}')$ で制約を設けるアプローチをとる．初期状態 $I(\boldsymbol{s})$ は以下のように表現する．

$$I(\boldsymbol{s}) = \quad (segment = l_0) \wedge \left(\bigwedge_{o_f \in 2^O \setminus \{\}, |o_f|=1} fed(o_f) = \text{True} \right) \wedge \left(\bigwedge_{o_f \in 2^O \setminus \{\}, |o_f|>1} fed(o_f) = \text{False} \right)$$

つぎに，$T(\boldsymbol{s}, \boldsymbol{s}')$ を定義することを考える．定義を簡便な表現にするため，いくつかの補助変数を定義する．まず，セグメントグラフ上のあるセグメント s からエッジ e を辿り別のセグメント s' へ SO O_{MOB} が移動する遷移を

$$S_e \triangleq (segment = s) \wedge (segment' = s')$$

と定義する．触媒反応による SO のフェデレーションの存在状態の変化に関する遷移も考える．まず，触媒反応が一切起こらなかった場合の遷移を

$$R_\emptyset \triangleq \bigwedge_{o_f \in 2^O \setminus \{\}} \left(fed(o_f) = fed'(o_f) \right)$$

さらに，触媒反応 $r = (c, M, N)$ が起こった際に SO のフェデレーションの存在状態が変化する遷移を

$$R_r \triangleq \bigwedge_{o_f \in N} \left(fed'(o_f) = \text{True} \right) \wedge \bigwedge_{o_f \in M} \left(fed'(o_f) = \text{False} \right) \wedge \bigwedge_{o_f \in (2^O \setminus \{\}) \setminus (M \cup N)} \left(fed(o_f) = fed'(o_f) \right)$$

と定義し，触媒反応 r が起こるフェデレーションの状態の条件を

$$RC_r \triangleq \bigwedge_{o_f \in M} (fed(o_f) = \text{True})$$

と定義する．つぎに，セグメントグラフ上のセグメント $s \in S$ から $s' \in S$ へエッジ e を辿って移動した際に反応する触媒反応の集合 R_{app} は以下のように与えられる．

$$R_{\text{app}}(e) \triangleq \{(c, M, N) \in R \mid c \in F(s'), O(c) \supseteq \bigcup M\} \text{ where}$$

$$O(c \in C) = O_{\text{MOB}} \cup \{o \in O \mid \exists s'' \in S.(c \in F(s''), (o, s'') \in L_{\text{FIX}}\}$$

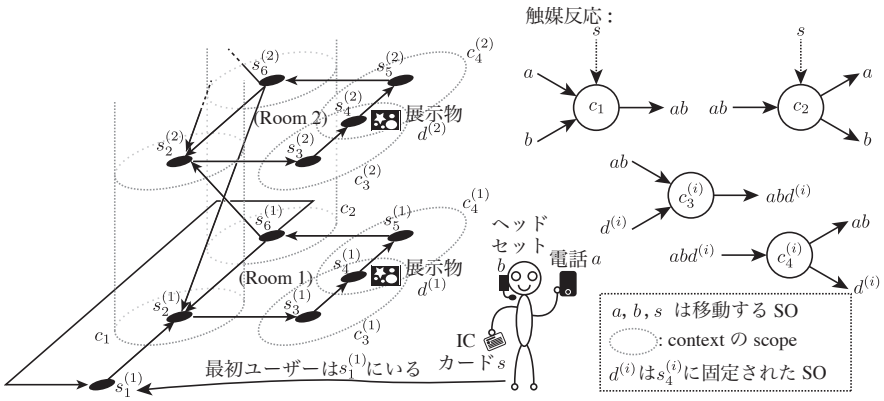

図 4: 実験で用いた CCRN で記述された UC シナリオ

表 1: 実験結果

実験インスタンス				素朴な手法 [Minoda 16]	
n	$\|O\|$	$\|C\|$	$\|S\|$	CPU (s)	MEM (MB)
1	4	4	6	0.01	13.81
2	5	6	11	0.04	16.50
3	6	8	16	0.41	48.46
4	7	10	21	8.69	656.75
5	8	12	26	273.56	13,088.76
6	9	14	31	N/A	MEM. Out

実験インスタンス				提案手法	
n	$\|O\|$	$\|C\|$	$\|S\|$	CPU (s)	MEM (MB)
1	4	4	6	0.01	13.41
2	5	6	11	0.02	15.24
3	6	8	16	0.04	19.61
4	7	10	21	0.09	31.64
5	8	12	26	0.24	67.85
6	9	14	31	0.78	188.54
7	10	16	36	2.85	636.56
8	11	18	41	10.93	2349.27
9	12	20	46	78.12	9368.13
10	13	22	51	455.73	13075.79
11	14	24	56	N/A	MEM. Out

備考: 「MEM. Out」は計算機の主記憶不足で実験を中断したことを示す

さらに，セグメントグラフの各エッジ $e \in E$ について以下のような遷移関係を定義する．

$$T_e \triangleq \begin{cases} S_e \wedge R_\emptyset & R_{\mathrm{app}}(e) = \emptyset \\ \bigvee_{r \in R_{\mathrm{app}(e)}} \left(S_e \wedge RC_r \wedge R_r \right) \vee & \\ \left(S_e \wedge \neg \left(\bigvee_{r \in R_{\mathrm{app}(e)}} RC_r \right) \wedge R_\emptyset \right) & \text{otherwise} \end{cases}$$

これらを用いて $T(\boldsymbol{s}, \boldsymbol{s}')$ は以下のように定義する．

$$T(\boldsymbol{s}, \boldsymbol{s}') = \bigvee_{e \in E} T_e$$

4. 実験と評価

スケーラビリティの評価を行うため実験を行った．本実験では，図 4 で表される CCRN で記述された UC シナリオを検証対象とした．このシナリオは n 階建ての博物館の UC シナリオであり，各階に廊下と展示室が 1 部屋ある．博物館の入り口と出口にはユーザーの電話とヘッドセットが音声ガイドとしての役割を果たすための触媒反応に対応する context c_1 と c_2 がある．各展示室 i には展示物 $d^{(i)}$ があり，$d^{(i)}$ に対応する説明をユーザーが受けるための触媒反応に対応する context $c_3^{(i)}$ と $c_4^{(i)}$ がある．セグメントグラフ中の有向辺 $(s_6^{(i)}, s_2^{(i+1)})$ と $(s_6^{(i+1)}, s_2^{(i)})$ は各階の間の階段に対応する．このような UC シナリオで n の大きさを変えながら，以下の LTL を満たすかどうかの検証を行う．

$$\mathbf{G}(segment = s_1^{(1)} \rightarrow \mathbf{F}(segment = s_4^{(n)} \rightarrow fed(\{a, b, d^{(n)}\}) = \mathrm{True}))$$

この LTL 式の直感的な意味は，ユーザーが博物館の入り口に入ったならば，最上階の展示室の展示物の説明を受けることが常にできるという意味である．実験には CPU が Core i7 3820QM，主記憶 16GB の計算機を用い，モデル検査器は NuSMV Version 2.6.0 を用い，素朴な手法 [Minoda 16] と本論文の提案手法を比較した．素朴な手法では，CCRN をモデル検査の問題に帰着させる際に全ての状態を明示的に記述するエンコーディングを行っているため，例え NuSMV2 を用いていたとしても記号モデル検査の利点は殆ど生かせていない手法である．実験結果は表 1 の通りである．表の左側が実験インスタンス，表の右側はそれを検証するのに要した CPU 時間と使用主記憶である．この実験環境では，素朴な手法を用いると 5 階の博物館のインスタンスで主記憶の制約から限界に達しているが，提案手法では 10 階の博物館のインスタンスまで同様の環境で検証することが可能になった．この結果から，記号モデル検査を用いることで，より多様な UC シナリオの検証が現実的な計算コストで行えるようになったと言える．

5. むすび

本論文では，CCRN で記述された UC シナリオを記号モデル検査を用いて効率的に検証するための帰着手法を提案した．これにより，素朴な手法 [Minoda 16] よりも大規模な UC シナリオの検証が現実的な計算コストで行えるようになった．しかしながら，変数を指数オーダーの個数用いているなど本手法にも改善の余地が残っており，今後の課題である．

謝辞

本研究の一部は JSPS 科研費基盤 (S) 15H05711 の助成による．

参考文献

[Burch 92] Burch, J., Clarke, E., McMillan, K., Dill, D., and Hwang, L.: Symbolic model checking: 10^{20} States and beyond, *Information and Computation*, Vol. 98, No. 2, pp. 142–170 (1992)

[Cimatti 02] Cimatti, A., Clarke, E., and Giunchiglia, E.: Nusmv 2: An opensource tool for symbolic model checking, *Computer Aided Verification*, Vol. 2404, pp. 359–364 (2002)

[Julia 16] Julia, J. and Tanaka, Y.: Proximity-based federation of smart objects, *Journal of Intelligent Information Systems*, Vol. 46, No. 1, pp. 147–178 (2016)

[Kauffman 02] Kauffman, S.: *Investigations*, Oxford University Press, Oxford New York (2002)

[Kripke 63] Kripke, S. A.: Semantical Analysis of Modal Logic I Normal Modal Propositional Calculi, *Zeitschrift für Mathematische Logik und Grundlagen der Mathematik*, Vol. 9, No. 5-6, pp. 67–96 (1963)

[Minoda 16] Minoda, R., Tanaka, Y., and Minato, S.-i.: Verifying Scenarios of Proximity-based Federation among Smart Objects through Model Checking, in *Proceedings of UBICOMM 2016 The Tenth International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies*, No. c, pp. 65–71 (2016)

[Tanaka 10] Tanaka, Y.: Proximity-based federation of smart objects: liberating ubiquitous computing from stereotyped application scenarios, in *Knowledge-Based and Intelligent Information and Engineering Systems*, pp. 14–30, Springer (2010)