グラフ化による制御ループに関する反例の可視化手法の提案

青木 善貴[1,a)]　小形 真平[2,b)]　小林 一樹[2,c)]　中川 博之[3,d)]

概要：CPS(Cyber Physical System) では，多数の独立的に動くコンポーネントが連携し，複数の制御ループを構成してシステムを稼働している．制御ループ内のコンポーネントがうまく協調しないと，システムが想定外の振る舞いを起こし，事故につながる可能性が高い．このような事故を発見するために，先行研究において，CPS のアーキテクチャのモデリングから抽出した制御ループの安定性を，モデル検査を用いて検証する手法 [3] を提案してきた．本手法では，抽出した制御ループが停止するかどうかを検証できる時相論理式で，モデル検査を実行して得られた反例から不具合の要因を明らかにする．ただし，反例は膨大な状態遷移のトレースを出力したもので，それを解析して制御ループの振る舞いを理解する作業は煩雑であり，不具合の要因を特定するのには非常に手間がかかる．本稿では，反例を解析して理解するための煩雑な作業を軽減する目的で，制御ループの振る舞いをグラフ化する可視化手法を提案する．先行研究の事例 [3] への提案手法の適用により，同じ不具合の要因の特定ができた．

キーワード：モデル検査，反例，CPS

Proposal of Counterexample Visualization Method for Control Loop by Graphing

1. はじめに

システムが重大な事故を引き起こす要因は，制御とフィードバックの循環的構造である制御ループ内に潜みやすい．IoT や CPS(Cyber Physical System) では，多数の独立的に動くコンポーネントが存在し，それらが密な連係の中で協調して動作する．そのため，連携するコンポーネント間での振る舞いがうまく協調できないと，制御ループ内のコンポーネント間の相互作用の影響により，システムが想定外の振る舞いを起こし，事故につながる可能性が高い．

我々は，CPS のアーキテクチャのモデリング手法として TORTE を提案し [1]，さらにモデル検査器 NuSMV[2] を用いて CPS の制御ループの安定性を検証する手法を提案してきた [3]．本手法は制御ループが停止するかどうかを時相論理による検査式で問うことにより不具合の発見を行うものである．

システムの制御ループが停止するということは，何らかの不具合が発生しているといえる．この不具合の発生の要因を特定するためには，モデル検査器が出力する反例を解析する必要がある．反例とは，検査式に記述した性質をシステムのモデルが満たさない場合に，その満たされない状態に至る状態遷移のトレースのことである．ただし，多数の状態の値が変化していく反例から特定の制御ループの状態遷移を読み取り，制御ループが停止する要因を明確にする作業は煩雑で手間がかかる．検証はモデルを修正しながら何度も行うので，反例の解析を支援できれば大きな作業軽減につながる．

本稿では，反例の解析作業を支援する目的で，制御ループの振る舞いを二次元もしくは三次元のグラフにすることにより，可読性を上げて直感的に理解できるよう支援する

[1] 日本ユニシス株式会社
東京都 江東区豊洲 1-1-1
[2] 信州大学
長野県長野市若里 4-17-1
[3] 大阪大学
大阪府吹田市山田丘 1-5
a) yoshitaka.aoki@unisys.co.jp
b) ogata@cs.shinshu-u.ac.jp
c) kby@cs.shinshu-u.ac.jp
d) nakagawa@ist.osaka-u.ac.jp

可視化手法を提案する．そして先行研究において農園モニタリングシステム内の制御ループが停止するか検証した結果 [3] を適用事例として，提案手法の有効性を示す．提案手法を適用して作成したグラフより，先行研究と同じ不具合の要因を特定することができた．

2. 本研究の適用対象及び課題

2.1 CPS の検証について

CPS は設計思想が統一されていない多様なコンポーネントで構成され，かつそれらが相互作用を行うものなので予め事故を想定して，不具合の要因を分析することは難しい．このようなシステムでは，事故を想定したテストケースで検査することも困難である．

我々は，システムに普遍的に存在する振る舞いに基づいて検証を行えば，CPS における不具合の要因を特定できると考え，制御ループに着目している．制御ループは，システム上で常に回り続けるものであり，もし停止したならば重大な不具合が発生したといえる．CPS をモデル化して制御ループが停止するかを検証をすれば，重大な不具合の要因が特定できる．

制御ループの変化は複雑であり，その遷移の組合せの仕方も多岐に渡るので，検証にはシステム仕様の妥当性を数学的に検証できるモデル検査の利用が有効であるが，そのためにはコンポーネント間の作用の伝搬を状態遷移で記述できるモデル記法が必要である．

我々の先行研究において，CPS のアーキテクチャをモデリングする手法として TORTE[1] を提案している．TORTE は，コンポーネント間の六つの属性（要求/監視/制御/データ転送/エネルギー供給/使用）の受け渡しをモデリングするもので，制御ループ内のコンポーネント間の作用の伝搬を表すことに適している．我々は TORTE のモデルをモデル検査のモデルへ自動変換し，CPS の制御ループが停止するかどうかをモデル検査で問うことにより，システムの不具合の要因を特定する手法の提案を行った [3]．

本稿では，提案手法を上記の手法で得られたモデル検査の反例に対して適用する．

2.2 TORTE

我々は CPS のアーキテクチャをモデリングする手法として TORTE[1] を提案している．TORTE は，CPS のコンポーネント間の関係性を要求/監視/制御/データ転送/エネルギー供給/使用の六つに分類してモデルを記述するため，分離された関係ごとに複雑なシステムを整理することで，結果的に多種多様な制御ループを漏れなく捉えることにつながる．**図 1** は，データ転送の関係性のモデルであり，コンポーネント間のデータ転送の関係だけを図示している．TORTE のモデルの編集は，UML モデリングエディタ Astah のプラグインで作成したエディタで行う．このエディタは，記述した TORTE のモデルよりモデル検査器 NuSMV のモデルを生成する機能をもつ．

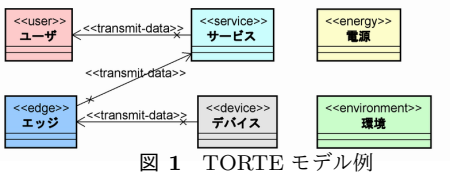

図 1 TORTE モデル例
Fig. 1 TORTE model example

2.3 モデル検査器 NuSMV

モデル検査は形式手法の一つで，システム仕様の妥当性を数学的に検証する．システムの振る舞いは数学的モデルによって定義され，満たすべき性質を表わす時相論理式（検査式）でその充足関係を検証する．性質が満たされない場合，モデル検査は反例を出力する．

モデル検査がもつ基本的な検証の特性は，「到達可能性」，「安全性」，「公平性」，「活性」である．これらの特性の意味は，p と q を基本命題とすると，「到達可能性」は「いつか p が成り立つ」であり，「安全性」は「p が成り立つことは決してない」，「公平性」は「p が無限回成り立つ」，「活性」は「p が成り立てばいつか q が成り立つ」である．これらの特性による検証は，時間軸を無限にとり，基本命題が成り立つかを判断する．

モデル検査器は，NuSMV [2] を使用した．時相論理式には，LTL(Linear Temporal Logic) と CTL(Computation Tree Logic) があり，NuSMV は，LTL と CTL の二種類の時間論理式を使用できる．

LTL は状態遷移を直線的な構造として捉え解釈するものである．ただし，“ある性質をもつパスが存在する”のような性質を LTL で記述するのは難しい．そのような場合は，状態遷移を木構造として捉えて“全ての経路について”または“ある経路において”という性質が付加できる CTL の方が適している．逆に，一つのパスだけに着目してそのパス上のみで成り立つ性質を検証するような場合は LTL の方が適している．検証したい性質に合わせてどちらの時相倫理を使うか決める必要がある．

2.4 制御ループが停止するかどうかの検証

制御ループが停止しないことを確認するには，時相論理式でループが永続的に続くことが検証できればよい．TORTE のモデルでは作用元コンポーネントから作用先コンポーネントへの属性の受け渡しが記述できる．この属性の受け渡しは，時相論理式では“作用元コンポーネントで p（例えばデータの送信）が成り立ったならいつか作用先コンポーネントで q（データの受信）が成り立つ”と記述できる．この記述で制御ループを構成するコンポーネント

を順次つなげていけば，“制御ループが一周する”を検証できる時相論理式となる．

制御ループが永続的に続くことは，“制御ループを一周回る”に「公平性」を追加して“制御ループが一周することが無限回続く”と記述することにより表せる．CTL は，ひとつのパスだけに着目して，そのパスだけ成り立つ性質の検証には適していないので，使用する時相論理式は LTL である．

検査式は例えばコンポーネント C1，C2 があり，C1 が C2 を制御して C2 が C1 へデータを転送する制御ループがあるとする．これらを TORTE で表すと**図 2** と**図 3** になる．この TORTE のモデルをモデル検査器 NuSMV のモデルに変換する．この NuSMV のモデルでは，送信側のコンポーネントが送信の状態になると，次に受信側のコンポーネントが受信の状態になることにより，属性の伝達を表す．

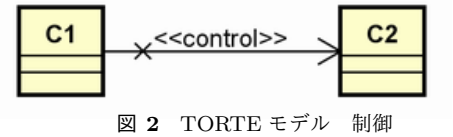

図 2　TORTE モデル　制御
Fig. 2　TORTE Model Control

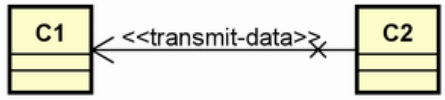

図 3　TORTE モデル　データ転送
Fig. 3　TORTE Model Transmit Data

コンポーネントの作用元（送信側）の状態をそれぞれ C1p，C2p，コンポーネントの作用先（受信側）の状態をそれぞれ C1q，C2q とした場合，これらの状態がもつ値を**表 1** に示す．

制御ループの状態遷移は，C1p=Controlling ⇒ C2q=Controlled ⇒ C2p=Transmitting_Data ⇒ C1q=Transmitted_Data となる．このループが永続的に周回することを検証する検査式は，式 (1) となる．

表 1　TORTE モデルの状態の値
Table 1　TORTE Model State Value

属性	位置	状態	状態の値
制御	送信側	C1p	Controlling,Ready
	受信側	C2q	Controlled,Ready
データ転送	送信側	C2p	Transmitting,Ready
	受信側	C1q	Transmitted,Ready

$$
\begin{aligned}
&G\ F\ (C1p = Controlling \wedge \\
&\quad F\ (C2q = Controlled \wedge \\
&\quad F\ (C2p = Transmitting\_Data \wedge \\
&\quad F\ (C1q = Transmitted\_Data)))) \qquad (1)
\end{aligned}
$$

式の意味は，“ある時点で C1p=Controlling が成り立ち，かつそれ以降の時点で C2q=Controllded が成り立ち，かつそれ以降の時点で C2p=Transmitting_Data が成り立ち，かつそれ以降の時点で C1q=Transmitted_Data が成り立つ”が無限回成り立つである．

2.5　反例解析の課題

モデル検査において，システムの振る舞いは数学的モデルによって定義され，満たすべき性質を表わす時相論理式（検査式）でその充足関係を検証する．与えられた性質が満たされなかった場合，モデル検査器はその満たされない状態に至る状態遷移のトレースを反例として出力する．トレースは，状態遷移の発生単位で記録されたレコードで構成されている．不具合は検査モデル上の特定の状態遷移で表されるため，検査者はこの反例を解析することにより，不具合の要因を特定することができる．

しかし，2.4 節の式 (1) で示した検査式で制御ループが停止に至る反例を出力したとしても，反例は検査式を満たさないある一つの状態への状態遷移の経緯を表しているに過ぎず，無関係な状態遷移も含まれているため，状態遷移を検査モデルと照らし合わせながら取捨選択して，状態遷移の振る舞いを再構築し，要因を特定する必要がある．

状態の種類及び反例のレコードが少なければ問題にならないが，複雑なモデルで状態の種類が多く，反例のレコードが膨大な場合は，不具合の要因の特定に非常に手間がかかる．このような手間がかかる作業をしなくても不具合の要因を特定できる手法が望まれる．

そのため反例に関する研究も進められている．Clarke ら [5] は，論理式を組み込んで反例の表現を拡張することを提案している．反例をより見やすい形にしようとしているが，解釈するには記述方法を学習する必要がある．

Lerda ら [6] はシミュレーションの技術を用いて反例を検索することを提案している．発見したい振る舞いを表す反例となるようにシミュレーションで状態探索を行って，反例をグラフ化している．シミュレーションを行うためにはあらかじめ目標が明確である必要があるため，想定外の事故の検証には向いていない．

モデル検査機 SPIN の可視化ツールとして iSPIN がある．iSPIN は反例をシーケンス図として表すことができる．また，同様にモデル検査機 UPPAAL[7] も反例をシーケンス図として表すことができる．これらは，複数あるプロセスがどのようなタイミグで連携をとっているかの確認には向いているが，制御ループに関係がないつながりも多く含

まれ，制御ループ特定は難しく，制御ループの振る舞いの理解の支援には向いていない．

3. 提案手法

3.1 基本的な考え方

提案手法は，反例より検査モデル内の制御ループに該当する状態遷移を抽出し，その振る舞いを理解が容易な形で可視化して反例解析を支援するものである．

通常，反例から制御ループが停止する要因を特定するには，検査者がモデル検査のモデルと反例を照らし合わせてその意味を読み取り，制御ループが停止するポイントとなる状態を特定し，再度反例を読みなおしてその状態に陥る理由を順序だてて明確する必要がある．

制御ループが停止するポイントの特定には手間がかかるため，直感的に停止するポイントが理解できる可視化が必要である．また，原因特定のために，再度反例を読み直すことをしなくてもすむように制御ループが停止する要因の特定ができることも必要である．

反例の解析はモデルの変更をしながら何度も行われるものである．したがって制御ループの停止する要因をある程度でも想定できる表示できれば，検証の手間を大幅に削減することができる．

複数の制御ループがある場合，制御ループの相互作用が停止の原因になることが想定されるため，相互作用を理解できる可視化も必要である．

提案手法は，以下の事項を満たす必要があると考える．

(1) 制御ループの停止を直感的に理解できること
(2) 制御ループの停止の要因を把握できること
(3) 複数の制御ループを図示できること

上記の事項を満たすために，X 軸に状態，Y 軸に時間（反例レコードのインデックス），Z 軸に制御ループの種類（制御ループの経路単位）をとる立体的なグラフで表す．グラフのノードは，コンポーネントの送受信の状態を表し，エッジはコンポーネントの属性の伝搬があったことを表す．

(1) については，制御ループを TORTE の状態遷移の定義に則ったグラフで表すことで対応する．状態遷移が停止すれば，グラフも途切れるので直感的に理解できる．（3.2 節に記述）

(2) については，制御ループを構成する各コンポーネント間の状態遷移の発生条件を時間（反例レコードのインデックス）を加味して表記することにより対応する．制御ループの状態遷移が発生すべき時に，条件が揃っているかどうかが分かるので，停止の要因が把握できる．（3.3 節に記述）

(3) については，経路が異なる制御ループを，Z 軸に沿って階層状に重ねて表記することにより対応する．（3.4 節に記述）

3.2 制御ループの状態遷移のグラフ表記

制御ループは，CPS を構成するコンポーネント間にある関係性の円である．しかし，制御ループは何度も周回するものであり，複数の制御ループを比較することも考慮すると，円形での表現は比較評価が行い難い．

図 4 の上図は，2.4 節で示した C1，C2 の二つのコンポーネントによる C1⇒C2⇒C1 と周回する制御ループを表す．この制御ループは，C1 が C2 を制御するものであり，C2 は C1 へデータをフィードバックしている．提案手法ではこれを図 4 のように，C1⇒C2⇒C1 の線分で構成される直線に展開することにより，制御ループを線形で捉える．

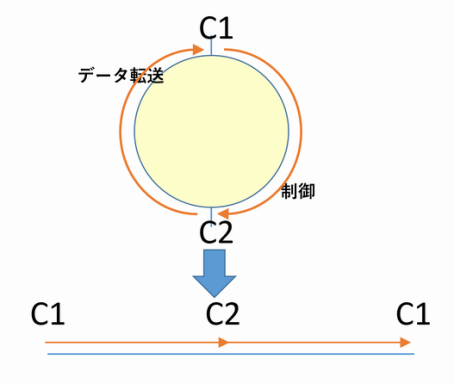

図 4 制御ループの直線への展開
Fig. 4 Expanding control loop to a straight line

モデル検査で検証を行うため，線形に表した制御ループは状態遷移に置き換える必要がある．TORTE における状態遷移の定義では，各状態は，“待機”，“送信”，“受信”の三つの値をもつ，コンポーネント間の属性（要求/監視/制御/データ転送/エネルギー供給/使用）の伝搬は，送り元の状態の値が“送信”となり，送り先の状態の値が“受信”になることで実現している．ただし，送り先の状態が“待機”の場合のみ，“送信”が可能となる．

制御ループは**図 5** のような状態遷移となる．この状態遷移の定義では，TORTE におけるコンポーネント間の関係性の伝搬の欠落，競合などによる不具合が検証ができる．

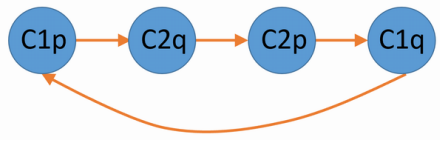

図 5 制御ループの状態遷移
Fig. 5 Control loop state transitions

図 5 のモデルにおける送信の状態の値は Controlling と Transmitting_Data であり，受信の状態の値は Controlled と Transmitted_Data である．C1p=Controlling ⇒ C2q=Controlled ⇒ C2p=Transmitting_Data ⇒

C1q= Transmitted_Data 遷移を表す反例がある場合，送信と受信の値だけを取り出して二次元グラフにすると図 6 となる．X 軸に状態 C1p，C2q，C2p，C1q を配置し，Y 軸に状態の値（Controlling,Controlled,Transmitting_Data,Transmitted_Data）を任意の位置に配置している．

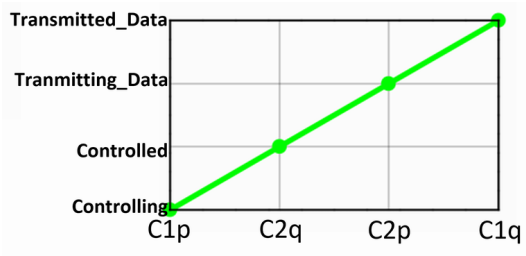

図 6 制御ループ グラフの例 I
Fig. 6 Control loop graph example I

制御ループが停止した場合には，“送信”⇒“受信”の関係が成立しない場所でグラフが途切れるのでどこで停止したのかが明確に分かる．図 7 は，コンポーネント C2 から C1 へのデータ転送が行われず，途切れた例である．

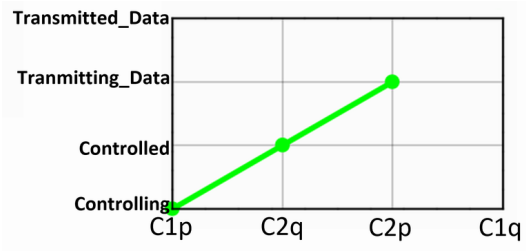

図 7 制御ループ グラフの例 II
Fig. 7 Control loop graph example II

3.3 反例レコードのインデックスを時間として加味したグラフ表記

図 6 の表記だけでは，制御ループが周回したことはわかるが，状態遷移の前後の関係まではわからない．反例は状態遷移が発生する単位でレコードを記録するので，レコードのインデックスを疑似的な時間と解釈できる．状態の値が，レコードのインデックスの単位でどのくらい保持されるのかを表せば，状態遷移の前後の関係を示せる．

TORTE における状態遷移の考え方は，送り元が送信の状態である間だけ送り先が受信の状態になることができる．送信の状態がどれだけ続くかは，状態遷移の非決定性やモデルの仕様により異なる．

送り先が受信になった後でも送信のままであることはあり得る．その場合，継続中の送信の状態のどこからでも受信の状態へエッジを引き得るため，グラフの線描は以下のように行うものとする．

- 送信終了ポイントの反例レコードのインデックスが，受信開始ポイントの反例レコードのインデックスより大きい場合は，受信開始了ポイントと送信開始ポイントをつなぐ（図 8）
- 送信の終了ポイントの反例レコードのインデックスが，受信開始ポイントの反例レコードのインデックス以下の場合は，受信開始ポイントと送信終了ポイントをつなぐ（図 9）

図 8 グラフの線描方法 I
Fig. 8 Graph drawing I

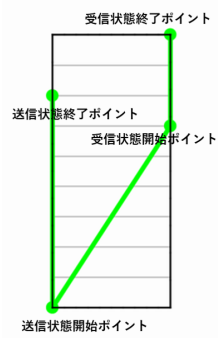

図 9 グラフの線描方法 II
Fig. 9 Graph drawing II

制御ループが一周することを示す反例の例を表 2 で示す．初期値状態 C1（始点）が送信から開始して，順次状態が伝搬して制御ループを一周する．

表 2 制御ループ 反例サンプル I
Table 2 Control loop counterexample sample I

インデックス	C1p	C2q	C2p	C1q
0 1	送信	待機	待機	待機
0 2	待機	受信	待機	待機
0 3	待機	受信	待機	待機
0 4	待機	待機	送信	待機
0 5	待機	待機	送信	待機
0 6	待機	待機	待機	受信
0 7	待機	待機	待機	受信
0 8	待機	待機	待機	受信

横軸（X 軸）に状態を配置し，縦軸（Y 軸）に反例レコードのインデックスを配置する．状態の値が“送信”もしくは“受信”の場合のみ二次元グラフ上にポイントし，その間を線でつないだものが図 10 である．

制御ループがどのような時系列で状態遷移するのかが明確にわかる．もし，状態 C1q が“待機”とならなければ送信が行えないため C2q⇒C2p⇒C1q への遷移は発生しない．その場合の反例は，例えば表 3 になる．

状態 C1p はレコード 06 から再度“送信”となり，以降も変わらないとすると，状態 C2p はレコード 07 以降で状態 C1q の“待機”待ちで“送信”にならない．この振る舞い

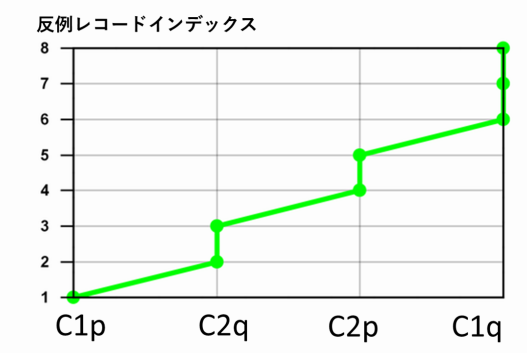

図 10　制御ループ　グラフの例 II
Fig. 10　Control loop graph example II

表 3　制御ループ　反例サンプル II
Table 3　Control loop graph example II

インデックス	C1p	C2q	C2p	C1q
0 1	送信	待機	待機	待機
0 2	待機	受信	待機	待機
0 3	待機	待機	送信	待機
0 4	待機	待機	待機	受信
0 5	送信	待機	待機	受信
0 6	待機	受信	待機	受信
0 7	待機	待機	待機	受信
0 8	待機	待機	待機	受信

を図 10 と同様に二次元グラフ化したものが**図 11** である．

状態 C2q⇒ 状態 C2p⇒ 状態 C1q へのエッジが引かれないため，制御ループが停止したことが明確にわかる．また，状態 C1q が“受信”となるレコード 06 以降で，図 11 の状態 C1p のレコード 06-08 間にエッジが引かれることから状態 C1q が“待機”にならないことが明白で，制御ループが停止する要因も“状態 C1q がレコード 06 以降で待機にならない”ことと把握できる．

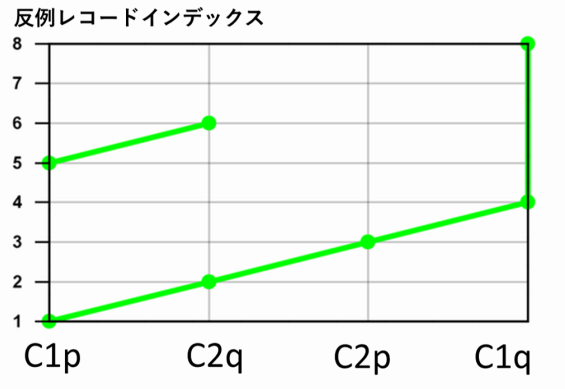

図 11　制御ループ　グラフの例 III
Fig. 11　Control loop graph example III

このような反例レコード数を時間として加味した二次元グラフにより制御ループが停止する場所が明確にわかり，停止する位置が分かれば，その停止した状態遷移が起きる条件をグラフ上に表記することにより，タイミングを含めて停止の要因が把握できる．

3.4　複数ループの表記

3.3 節の反例レコード数を時間として加味した二次元グラフでは，経路が異なる制御ループが複数重なると振る舞いが理解しづらくなる場合がある．例えば三つの制御ループが混在する例が**図 12** である．異なる経路の制御ループは色を変えてあるが，同じ状態を通る制御ループがあるとループの振る舞いの把握が困難である．

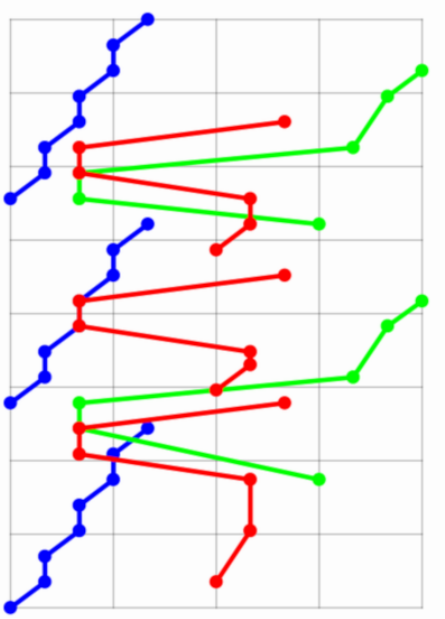

図 12　三つの制御ループを二次元で表すグラフ
Fig. 12　Graph showing three control loops in two dimensions

この表記だと，制御ループが混在して振舞いの理解が難しい．制御ループをループの経路毎にグルーピングして Z 軸方向に階層的に配置すると，**図 13** となる．

三種類の経路が異なる制御ループがあり，上からそれぞれ三回，二回，三回ループしていることがわかる．複数のループが相互作用を及ぼし合っている場合などに，階層的記法による三次元グラフにより制御ループの振る舞いの概要を把握できる．まず制御ループの振る舞いの概要を把握した後に，検証の対象を絞り反例レコード数を時間として加味した二次元グラフで詳細を検証することが有効と考える．今回線描に使用したソフトウェアは，RINEARN Graph 3D[8] である．Excel などの表計算ソフトのデータのファイルもしくはコピーを用いて 3D グラフを作成できる．作成した図形は自由に回転させることができるので，複数の制御ループの関係を見やすい角度から確認できる．

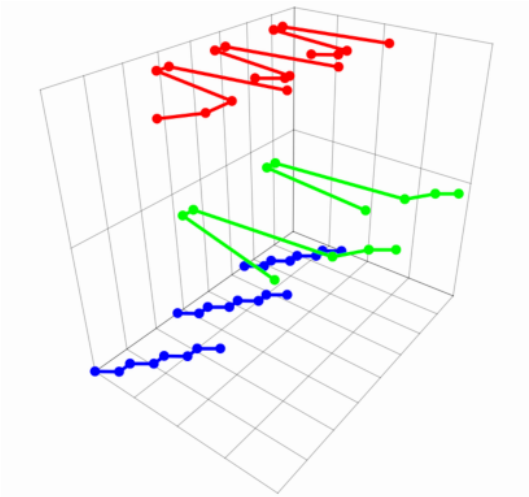

図 13　三つの制御ループを三次元で表すグラフ
Fig. 13　Graph showing three control loops in three dimensions

4. 適用事例

4.1 適用事例概要

3 章で提案した手法を農園監視モニタリングシステムに適用する（図 **14**）．このシステムは，農園に設置されたカメラから画像を定期的に取得し，クラウドサービス上のサーバに画像をアップロードする．そしてその画像を処理して消費者に提供する．太陽光発電による電力システムから電力が供給される．クラウドサービスには，Google ドライブ，Web サーバー，写真共有サイト Flikcr がある．

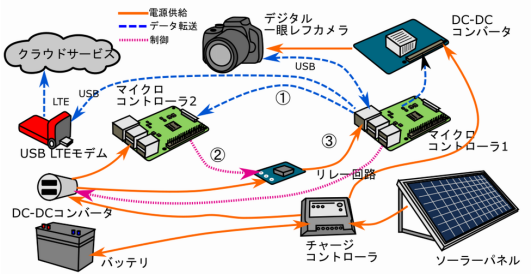

図 14　農園監視モニタリングシステム
Fig. 14　Farm monitoring system

図 14 にある① ⇒② ⇒③が制御ループである．マイクロコントローラ 1(MC1) は定期的に起動して撮影を行うため，次回の起動時間のデータをマイクロコントローラ 2(MC2) へデータ転送する (①)．MC2 はその起動時間に MC1 の電源が入るようにリレー回路を制御し (②)，リレー回路は MC1 に電源を供給する (③)．これらを制御ループとしてつなぐ関係性は，TORTE における制御，データ転送，エネルギー供給である．TORTE モデルによる制御ループの表現は次の図 **15**，図 **16**，図 **17** ようになる．

図 15 はデータ転送の関係性のモデルであり① が含まれ，図 16 は制御の関係性のモデルであり②が含まれ，図 17 はエネルギー供給の関係性のモデルであり③が含まれる．

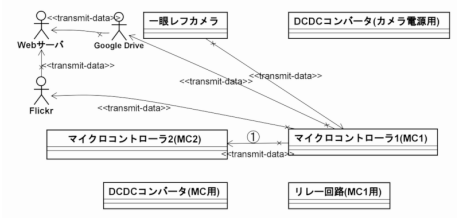

図 15　TORTE データ転送モデル
Fig. 15　TORTE Data Transfer Model

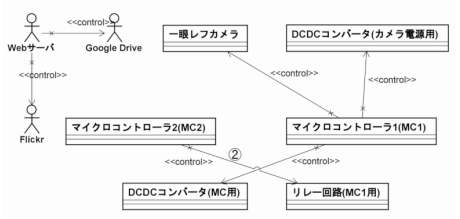

図 16　TORTE 制御モデル
Fig. 16　TORTE Control Model

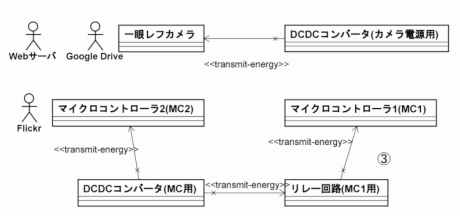

図 17　TORTE エネルギー供給モデル
Fig. 17　TORTE Energy Supply Model

4.2 制御ループが停止するかどうかの検証結果

先行研究 [3] において，2.4 節で記述した制御ループが停止するかどうかの性質を検証する検査式でモデル検査を実施したところ，性質は満たされない false となった．反例が出力され，制御ループが停止する可能性があることが分かった．

今回の検証の対象となるコンポーネントは，MC1，MC2 とリレー回路である．モデル検査のモデルにおける制御ループの状態は次の通りである．

(C1q)MC2 が MC1 よりデータを受信する（Transmitted_Data）
(C1p)MC2 がリレー回路を制御する（Controlling）
(C2q)リレー回路が MC2 に制御される（Controlled）
(C2p)リレー回路が MC1 へ電源供給をする（Transmitting_Energy）
(C3q)MC1 がリレー回路から電源供給を受ける（Transmitted_Energy）
(C3p)MC1 が MC2 へデータを送信する（Tranmitting_Data）

送信の状態の値は Controlling と Transmitting_Data，

Transmitting_Energy であり，受信の状態の値は Controlled と Transmitted_Data，Transmitted_Energy である．検査式は，以下の式 (2) のようになる．

$$
\begin{aligned}
G\ F\ (&C1q = Transmitted\_Data \wedge \\
&F\ (C1p = Controlling \wedge \\
&F\ (C2q = Controlled \wedge \\
&F\ (C2p = Transmitting\_Energy \wedge \\
&F\ (C3q = Transmitted\_Energy \wedge \\
&F\ (C3p = Transmitting\_Energy)))))
\end{aligned}
\tag{2}
$$

検査式に記述した式の意味は，“ある時点で MC2 が MC1 からデータを受信し，かつそれ以降のある時点で MC2 がリレー回路の制御を行い，かつそれ以降のある時点でリレー回路が制御される　かつそれ以降のある時点でリレー回路が MC1 へ電源供給を行い，かつそれ以降のある時点で MC1 がリレー回路から電源供給を受ける，かつそれ以降のある時点で MC1 が MC2 へデータを送信する”を無限回繰り返すかどうかである．

反例を解析したところ，MC1 がクラウドサービス側への送信のみ行っているため MC2 へのデータ転送が行われないことが要因で，制御ループが停止していることが分かった．

この結果に基づき，実際のシステムを検討したところ，MC2 へのデータ転送が行われないとリレー回路のタイマー起動の設定がされず，MC1 がダウンする可能性があることが判明した．安全を確保のために，一定間隔で MC2 へデータ転送を行う考慮の必要性が指摘できた．

4.3　得られた反例への提案手法の適用

先行研究において，反例の解析は，反例のテキストファイルを EXCEL へ取り込み必要なデータを抽出して振る舞いを理解することで行った．検証を実施する度に不規則に変化する状態の値の羅列から，制御ループを再構成するため非常に手間がかかる作業であった．

先行研究で得られた反例に本研究の提案手法を適用する．

4.3.1　検査モデルについて

検査モデルはモデル検査器 NuSMV で記述されており，コンポーネントごとにモジュールを作成した．そして各モジュールでは，JUSTICE(FAIRNESS) Running を宣言しており，これによりモジュールのすべてのインスタンスは非決定に選択されて，無限の時間軸において必ず周期的に実行されるため，全く実行されないモジュールは存在しない．

4.3.2　正常な制御ループのグラフ

正常に制御ループが一周回る反例を作成する．制御ループは全ての経路で常に周回する訳ではないが，ある経路においては周回するので CTL の式を用いて反例を出力する．

検査式は，4.2 節にある六つの状態，C1q，C1p，C2q，C2p，C3q，C3p からなる制御ループが対象である．検査式は，式 (3) のようになる．

$$
\begin{aligned}
\neg\ E\ F(&C1q = Transmitted\_Data \\
\wedge\ E\ F(&C1p = Controlling \\
\wedge\ E\ F(&C2q = Controlled \\
\wedge\ E\ F(&C2p = Transmitting\_Energy \\
\wedge\ E\ F(&C3q = Transmitted\_Energy \\
\wedge\ E\ F(&C3p = Transmitting\_Data))))))
\end{aligned}
\tag{3}
$$

式の意味は，“ある経路でいつか MC2 が MC1 からデータを受信し，かつそれ以降のある経路でいつか MC2 がリレー回路を制御し，かつそれ以降のある経路でいつかリレー回が MC2 に制御され，かつそれ以降のある経路でいつかリレー回路が MC1 へデータを送信し，かつそれ以降のある経路でいつか MC1 がリレー回路から電源供給を受け，かつそれ以降のある経路でいつか MC1 が MC2 へデータを送信する”を否定したものである．

モデル検査は検証結果が false となると反例を出力する．“制御ループがいつか周回する”は true なので，それを否定することにより制御ループが周回する反例が出力できる．式 (3) に先に示した状態 (C1q，C1p，C2q，C2p，C3q，C3p) を当てはめて検証した．検証した結果として得られた反例をグラフにしたものが，**図 18** である．

(C1q)⇒(C1p)⇒(C2q)⇒(C2p)⇒(C3q)⇒(C3p) の順で状態遷移して制御ループが周回していることがわかる．

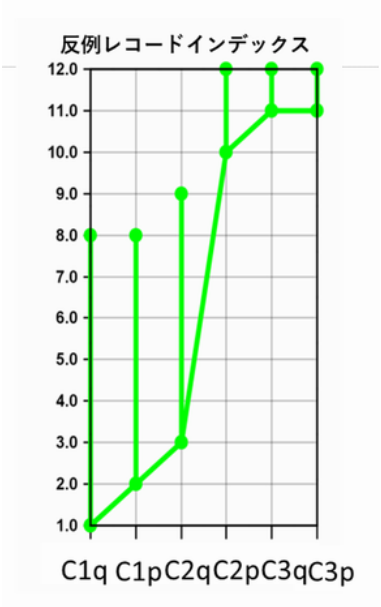

図 18　反例グラフ　正常制御ループ
Fig. 18　Counter example graph Normal control loop

4.3.3 停止した制御ループのグラフとの比較

制御ループが停止した反例をグラフ化する．この反例レコードのインデックスは 42 で，レコード 32 から 42 はループしている．正常な制御ループと合わせて表記したものが，**図 19** である．検査式は，2.4 節の式 (1) に 4.3.1 項に示した状態 C1q，C1p，C2q，C2p，C3q，C3p を当てはめたものである．グラフ①は停止した制御ループ，グラフ②は正常な制御ループである．

正常の制御ループと重ねて表記をすると，制御ループが停止した場所が明確に分かり，(f) の“MC1 が MC2 へデータを送信する”が発生しないことが特定できる．

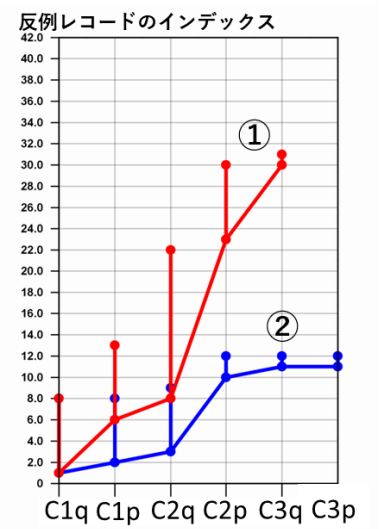

図 19 正常なループと停止するループの比較

Fig. 19 Counterexample graph Control loop to stop

停止する状態が判明したので，“MC1 が MC2 へデータを送信する”を発生させる条件の値をグラフ上に表記する．遷移の条件は，“MC2 がデータを受信中でない”，“MC1 が他のデータ送信を行っていない”及び“MC1 が待機中”の三つである．条件の表記は次のように行う．TORTE による状態遷移の条件にシステム固有の条件が付加されたものである．システム固有の条件の付加は NuSMV のモデルに対して手動で行った．システム固有の条件の付加の手法化は，本提案手法の課題である．

- “MC2 がデータを受信中でない”は (C1q) に該当するので，(C1q) の軸上に表記する．
- “MC1 が他のデータ送信を行っていない”は新たに (C4) を設けて表記する．
- “MC1 が待機中”は自明なのでグラフ上に表記しない．

二次元のグラフで表したものが，**図 20** であり，三次元のグラフで表したものが，**図 21** である．

①は，反例のレコード 32 から 42 は状態遷移がループしている．(C4) においてこれに該当する部分がグラフ上の点線の楕円である．(C4) は，(C3p) の成立以降において，

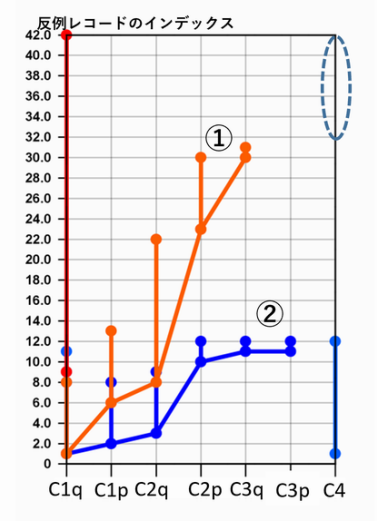

図 20 反例解析　二次元グラフ

Fig. 20 CCounterexample analysis two-dimensional graph

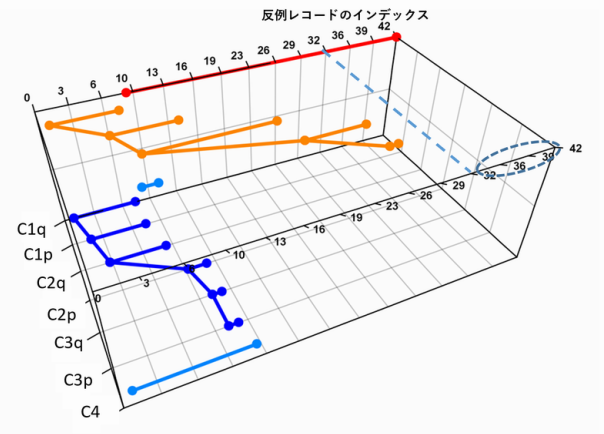

図 21 反例解析　三次元グラフ

Fig. 21 Counterexample analysis three-dimensional graph

永久に満たされることはないことが明確にわかる．(C1q) については，図 20 では制御ループと条件が見分けづらいが，図 21 では，経路が異なる制御ループを Z 軸方向にずらして配置しているので見分けがつき，満たされていることがわかる．

二つの図より，制御ループの停止の要因が (C4)“MC1 が他のデータ送信を行っていない”が満たされないこと，すなわち“MC1 が他のデータ送信ばかり行っている”と特定できる．この結果は，先行研究において導き出した検証の結果と同じである．手動により導き出した結果を反例を可視化したグラフより読み取ることができた．

5. まとめ

本稿では，制御ループの反例をグラフ化することにより，その振る舞い及び，不具合の要因をグラフより理解する可視化手法を提案した．本稿の適用事例で用いた制御ループは比較的シンプルなものであったが，反例を解析して理解

することは難しい．二次元と三次元のグラフを併用することで，直感的に理解できる表記であることが示せた．

3 章で示した提案手法は，基本的な可視化の方法である．実際には 4 章のの適用事例のように，システムの特性に合わせて修正を入れる必要があることが今後の課題である．

ただし，TORTE は NuSMV をモデル化するためのテンプレートをもっており，これを修正することにより状態遷移の条件や状態の値のパターンについては変更対応が可能である．

本稿では提案手法の事例への適用は手動で行ったが，手順は明確であるため今後ツールに実装して自動化することにより検証作業の大幅な軽減ができ，不具合の要因の想定が難しい複雑なモデルにも適用できると考える．

今回は，TORTE のモデルに合わせた制御ループの反例の可視化であったが，複数の制御ループの交点における同期の状態の状態を可視化するなど，検査モデルでも検証の目的に合わせて，抽出項目を変えることで可視化対応できると考える．

また，CPS ではコンポーネント間の作用の質（正確さ等）についても考慮が必要であるが，TORTE のモデルでは対象外であるため別途モデルを考案する必要がある．

参考文献

[1] S. Ogata, H. Nakagawa, Y. Aoki, K. Kobayashi, Y. Fukushima: A Tool to Edit and Verify IoT System Architecture Model, MODELS 2017, pp.571-575, 2017.
[2] A.Cimatti, E. M. Clarke, F. Giunchiglia，M. Roveri: NuSMV: A new symbolic model checker, STTT, vol.2, no.4, pp410-425, 2000.
[3] Y. Aoki, S. Ogata, K. Kobayashi, H. Nakagawa: Verification of CPS Based on Control Loop Using Model Checking, 25th Asia-Pacific Software Engineering Conference (APSEC) p678-682 ,2018.
[4] G. Schirner, D. Erdogmus, K. Chowdhury, T. Padir: The Future of Human-in-the-Loop Cyber-Physical Systems, Computer, vol.46, no.1, pp.36-45, 2013.
[5] Clarke, E.,Jha, S.,Yuan Lu ,Veith, H.,”Tree-like counterexamples in model checking” , Logic in Computer Science, 2002. Proceedings. 17th Annual IEEE Symposium on ,pp.19 - 29 ,2002.
[6] Lerda, F. ,Kapinski, J., Maka, H. ,Clarke, E.M., Krogh, B.H.,”Model Checking In-The-Loop:Finding Counterexamples by Systematic Simulation”, American Control Conference, pp.2734-2740,2008.
[7] UPPAAL, http://www.uppaal.org/, 2020.
[8] RINEARN Graph 3D，https://www.rinearn.com/graph3d/, 2020.