様　式　Ｃ－１９、Ｆ－１９－１、Ｚ－１９　（共通）

科学研究費助成事業　　研究成果報告書

令和　２ 年　 ７ 月　 ２ 日現在

機関番号：１２４０１
研究種目：基盤研究(C)（一般）
研究期間：2016～2019
課題番号：１６Ｋ００００１０

研究課題名（和文）時間論理によるリアクティブシステム仕様のプログラム化可能性判定を行う証明システム

研究課題名（英文）Proof system for Realizability Decision of Reactive System Specification described by Temporal Logic

研究代表者

吉浦　紀晃（Yoshiura, Noriaki）

埼玉大学・理工学研究科・教授

研究者番号：００３０２９６９

交付決定額（研究期間全体）：（直接経費）　　3,500,000円

研究成果の概要（和文）：本研究の成果は、時間論理式の構文的な特徴からプログラム化可能性の性質の一種である強充足可能性や段階的充足可能性の判定が可能となるかを検討し、論理式の構造によりこれら２つの性質が一致することがあることを示した。また、これらの性質を効率よく判定することができることを明らかにした。しかし、構造が限定的であった。そこで、ネットワーク機器のソフトウェア、ロボット、自動車制御のネットワークシステムなどの検証すべき性質を記述してその構造を分析した。その結果、論理式の構造よりも、システムの動作を表す原子命題と利用者やシステムの外界の動作を表す原子命題の論理式の中の位置が重要であることを明らかにした。

研究成果の学術的意義や社会的意義
本研究の成果は、時間論理で記述されたリアクティブシステムの仕様の実現可能性の性質である強充足可能性と段階的充足可能性が、論理式の構造に特徴や制限がある場合に、同一になり、また、これらの性質が効率よく判定可能であることがわかった。このことは、プログラムの自動合成にとり重要な性質であり、安全なソフトウェアやシステムの開発に有効である。自動合成はバグのないソフトウェアの開発に重要であり、人手による開発によるバグの混入を防ぐことができる。

研究成果の概要（英文）：This research discusses whether the syntactic features of temporal logic formulas can be used for decision of strong satisfiability and stepwise satisfiability, which are properties of programability of program specification. The research finds that these two propoerties are the same under some restricted structure of formulas and that the restriction of logical formulas enables to determine the two properties effectively. The restriction, however, is too severe and the research investigates weak restriction of logical formulas for effective determination of the two properties.
The research describes several properties for practical systems or software such as software for network devices, robots, and network systems for automotive control to investigates the structure of formulas for practical systems. The investigation shows that the positions of atomic formulas which expresses behaviors of environment of the system are important for determination of the two properties.

研究分野：情報工学

キーワード：時間論理　プログラム合成　モデル検査

様　式　Ｃ－１９、Ｆ－１９－１、Ｚ－１９（共通）

１．研究開始当初の背景

社会インフラとして利用される IT システムの多くは、リアクティブシステムである。これはオペレーティングシステムや飛行機の操縦システムなどユーザとインタラクションをしながら稼動するシステムであり、不具合のないリアクティブシステムプログラムを構築することは社会的に重要である。プログラムの不具合にはバッファオーバーフローなどプログラミングによるもののほかに、設計に問題がある場合もある。例えば、DNS キャッシュポイズニングの原因はプログラミングではなくプロトコルやソフトウェアの設計の問題、つまり、仕様の問題である。不具合のないリアクティブシステムプログラムの構築や検証の方法として、仕様を形式的言語である時間論理で記述し、形式的手法によりプログラムの合成や検証を行う方法がある。SMV や SPIN などは時間論理に基づいた検証ツールがあり、プログラムが仕様を満たすかを検証できる。現在、これらの検証ツールは記述が簡単な仕様しか検証できないが、将来、記述が複雑な仕様を検証する場合、仕様自体に誤りがないかを判定することが重要となる。特に、仕様を満たすプログラムが存在しないような仕様の場合、プログラムが仕様を満たすかを検証すること自体が無駄になる。一方、時間論理により記述された仕様を満たすプログラムが存在するか(プログラム化可能性)の判定やプログラムの自動合成の研究があり、Lily などのツールが提案されている。リアクティブシステムは、ユーザとのインタラクションを伴って動作するが、ユーザの動作を制御できない。よって、仕様を安易に記述すると、仕様を満たすプログラムが存在しないことがある。例えば、リアクティブシステムであるエレベータの仕様が以下の要件を持つ場合、仕様を満たすプログラムは存在しない。

・　5 階で利用者が「下がるボタン」を押したならば、いつかエレベータが 5 階に到着する。

・　利用者がエレベータ内の「ドアを開くボタン」を押したならば、ドアは開く。

・　ドアが開いている間、エレベータは動かない。

この仕様では、「ドアを開くボタン」を利用者が押し続ける場合、エレベータは動けず、5 階にいる利用者が「下がるボタン」を押しても 5 階にエレベータが到着することはなく、結局仕様を満たすプログラムは存在しない。この仕様には「ドアを開くボタン」が押され続けることがないという暗黙の前提があり、仕様に問題があるとみなせる。このように仕様がプログラム化可能であるかを判定することは仕様の不備や暗黙の前提を見つけることでもある。

仕様のプログラム化可能性を判定する方法やプログラムを合成する方法は既に提案されている。しかし、これらの方法はオートマトン理論に基づいており、膨大な計算時間とメモリを必要とし、現状では実用的とは言えない。本研究申請者は、プログラム化不能性を判定する証明システムを構築し利用することで、プログラム化可能性の判定の高速化を目指してきた。申請者は証明システムとオートマトンによるプログラム化可能性の判定方法を提案し、仕様がプログラム化不能である場合に証明システムの利用により仕様のプログラム化不能性を高速に判定することが可能となった。しかし、この証明システムは完全ではなく、また、証明システムにより判定可能な場合が多くないため、より強力な証明システムが必要である。

２．研究の目的

前述の背景に基づき、本研究はリアクティブシステム仕様のプログラム化可能性を判定する完全な証明システムを構築し、プログラム化不能である場合、その原因を明らかにする証明システムを構築する。さらにこれらの実装を行う。具体的には以下を明らかにする。

・　時間論理で記述された仕様のプログラム化可能性または不能性を判定できる完全な証明システムを構築する。仕様が時間論理により記述されるとき、仕様はユーザの動作を示す原子命題とシステムの動作を示す原子命題から構成される。仕様のプログラム化可能性は、ユーザの動作を示す原子命題の真偽値に依らず、システムの動作を示す原子命題の真偽値によって仕様を充足させることが可能かどうかという問題に帰着できる。

・　仕様のプログラム化可能性の重要な必要条件として、強充足可能性と段階的充足可能性がある。仕様がプログラム化不能である場合、これら 2 つの必要条件が成り立たない場合が考えられる。そこで、これら 2 つの必要条件が成り立たないことを判定する完全な証明システムを構築する。強充足可能性は、ユーザの将来も含めた動作列が分かっている時に、仕様を満たすシステムの動作列が存在する性質である。段階的充足可能性は、あるプログラムが存在し、そのプログラムは任意のユーザの動作列に対して、仕様を満たす可能性を保持し続けることができ、ユーザが特定の動作を行った場合に仕様を満たすことができる性質である。

・ 仕様に証明システムを適用した時に、プログラム化不能、強充足不能、段階的充足不能である場合に、証明システムの作成した証明図からその原因を導出する方法を構築する。
・ プログラム化可能性、強充足可能性、段階的充足可能性に関する真偽を判定する完全な証明システムの構築を目指すが、完全な証明システムの存在が明らかではない。そこで、完全な証明システムが構築できない場合には、できるだけ強力な証明システムの構築を行うとともに、完全な証明システムの存在の有無を明らかにする。

・ 構築した証明システムを実装し、既存のプログラム化可能性の判定を行うことが可能なツールとの比較を行う。さらに、証明戦略を構築し証明システムの実装に導入するとともに、プログラム化不能等の原因の導出方法の実装も行う。

また、本研究の学術的な特色・独創的な点及び予想される結果と意義は次の通りである。

・ リアクティブシステム仕様のプログラム化可能性判定やプログラム合成に関する従来の研究はオートマトン理論を利用しており、ツールの実装も行われている。今後改良が期待されるが、現状では簡単な仕様しか取り扱えない。一方、本研究申請者は既に仕様のプログラム化可能性に関する証明システムを構築している。本研究はこの研究を発展させるものであり、論理における証明システムを利用する点が他の研究との違いである。

・ 仕様のプログラム化可能性に関する証明システムは、仕様がプログラム化不能な場合に非常に高速に判定できることが明らかになっており、既存のいくつかの判定ツールで判定できない仕様のプログラム化不能性を判定している。このように証明システムの構築は判定の高速化につながる。また、証明システムの並行利用や証明戦略の工夫によりさらなる高速化が可能となり、仕様がプログラム化可能ではない場合の原因を導出する方法など新たなツールの開発も目指している。

・ 論理の視点から見た場合、本来、証明システムは論理式が正しいことを判定することを目的としているが、本研究の証明システムはプログラム化可能性に関する性質を証明することを目的としており、従来の論理の証明システムとは考え方が異なる。また、完全な証明システムが構築可能であるか否かも時間論理の研究として非常に興味深い。

・ 本研究により仕様のプログラム化可能性の判定方法の高速化が行われれば、現在利用されつつあるモデル検査手法がより複雑な性質を検証する場合においても、性質自体に誤りがないかを検証可能となり、より複雑な仕様の検証を行うことが可能となる。さらに、仕様からのプログラムの自動合成にも有効であり、ソフトウェアの生産性と安全性の向上が期待できる。

３．研究の方法

本研究目的を達成するために以下を実施する。

1. 仕様のプログラム化可能性や不能性を判定する完全な証明システムの構築

2. プログラム化可能性の必要条件である強充足可能性と段階的充足可能性を仕様が満たさないことを判定する完全な証明システムの構築

3. プログラム化不能、強充足不能、段階的充足不能である場合の原因を証明システムの作成した証明図より導出する方法の構築

4. 各性質の判定のための証明システムの実装、証明戦略の構築、および原因の導出方法の実装

具体的な計画は以下のとおりである。

(1) 時間論理で記述された仕様のプログラム化可能性に関する完全な証明システムの構築

(ア) 仕様のプログラム化不能性を判定する証明システムの構築
時間論理で記述されたリアクティブシステム仕様から推論規則を適用して矛盾が導出される場合、仕様がプログラム化不能と判定できる完全な証明システムを構築する。既存の証明システムは健全性しか示されておらず、完全性が明らかではない。また、不足している推論規則があることも予想される。様々な仕様に対して証明システムを適用し、その証明力を検証し、完全性を満たすように証明力を向上させる。なお、プログラム化可能性の形式的定義は次のようになる。
時間iのユーザの動作をa_i、システムの動作をb_iとしたとき、ユーザの動作列を$a =$

$a_0a_1a_2\cdots$、システムの動作列を$b = b_0b_1b_2\cdots$とする。また、時間iのユーザとシステムの動作を合わせたものを(a_i, b_i)とし、その動作列を$(a_0, b_0)(a_1, b_1)\cdots$とする。仕様Ψがプログラム化可能であるとは、すべてのユーザの動作列aに対して、$(a_0, b_0)(a_1, b_1)\cdots$が仕様Ψを満たすようなプログラム$p$が存在することである。ただし、$b_i = p(a_0\cdots a_i)$である。

(イ) 仕様のプログラム化可能性を判定する証明システムの構築
仕様がプログラム化可能であることを示す証明システムを構築する。具体的には、結論が必ずプログラム化可能となるような完全な証明システムの構築を行う。この証明システムを利用することで、仕様がプログラム化可能であることを高速で判定する方法を構築する。この証明システムはそもそも構築できるか否かが明らかではない。初めに簡単な証明システムを構築し、徐々に証明力を持つように証明システムの構築を行う。

(2) プログラム化可能性の必要条件である強充足可能性と段階的充足可能性に関する性質を判定する証明システムの構築

(ア) 仕様のプログラム化可能性の必要条件である強充足可能性に関する判定手続きの構築
具体的には、仕様の強充足不能性を判定する完全な証明システムを構築する。仕様が強充足不能の場合、ユーザの動作を表す原子命題のみからなる論理式が仕様から導出される可能性がある。よって、仕様が強充足不能である場合に、原子命題のみからなる論理式が仕様から導出されることを証明し、さらに、その式を導出する証明システムを構築する。なお、強充足可能性の形式的定義は次のようになる。
仕様Ψが強充足可能性であるとは、任意のユーザの動作列$a = a_0a_1a_2\cdots$に対して、システムの動作列を$b = b_0b_1b_2\cdots$が存在し、その動作列$(a_0, b_0)(a_1, b_1)(a_2, b_2)\cdots$が仕様Ψを満たすことである。

(イ) 段階的充足不能性を判定する証明システムの構築
仕様のプログラム化可能性の必要条件である段階的充足可能性に関する判定手続きを構築する。具体的には仕様の段階的充足不能性を判定する完全な証明システムを構築する。仕様が段階的充足不能である場合、どのようなプログラムであっても、システムの動作がユーザの将来の動作を限定してしまう特徴を有する。この特徴を利用して証明システムを構築する。なお、段階的充足可能性の形式的定義は次のようになる。
仕様Ψが段階的充足可能性であるとは、あるプログラムpが存在し、任意のユーザの有限動作列を$a = a_0a_1\cdots a_n$と、$b_i = p(a_0a_1\cdots a_i)$となる動作列$(a_0, b_0)\cdots(a_n, b_n)$に対して、ユーザとシステムのある無限動作列$\sigma$が存在し、無限動作列$(a_0, b_0)\cdots(a_n, b_n)\sigma$が仕様Ψを満たすことである。

(3) 仕様がプログラム化不能である原因を導出する方法の構築

(ア) 原因の導出方法の構築
仕様がプログラム化不能であることを判定する証明システムでは、仕様がプログラム化不能ではないことが証明された場合、その証明図は仕様がプログラム化不能である原因に関する情報を含むことが分かっている。プログラム化不能、強充足不能、段階的充足不能の各種性質に関する証明システムが構築した証明図から、その原因を導出する方法を構築する。

(イ) 最適な原因の選択方法の構築
仕様がプログラム化不能、強充足不能、段階的充足不能である場合、その原因は 1 つとは限らない。構築される導出方法が、仕様のプログラム化不能等の原因を複数個導出することもある。複数の中から原因として最適なものを選択する方法を構築する。最適な原因を決定するために、原因の間に順序関係を導入する。

(4) 証明システムと原因の導出方法の実装と証明戦略の構築

(ア) 証明システムの実装
プログラム化可能性に関する 3 つの性質の証明システムの実装を行う。さらに、様々な仕様を用いて、この実装の評価を行う。実装においては、マルチスレッドによる並列化など実装上の工夫により高速化を図る。

(イ) 証明戦略の構築
証明システムの実装に対して、証明システムの推論規則をどのように適用するかの方針、つまり証明戦略を構築し、実装に導入する。

(ウ)仕様がプログラム化不能である原因の導出方法の実装
証明システムの実装により証明図が得られるが、この証明図から仕様がプログラム化不能等の原因を導出する方法を実装する。

４．研究成果

本研究では以下の研究成果を得た。

(1) 仕様の構文的な性質と強充足可能性や段階的充足可能性との関係の分析
時間論理で記述された論理式の構文的な特徴から強充足可能性や段階的充足可能性の判定が可能となるかを検討した。論理式の構造により強充足可能性と段階的充足可能性が一致することがあることを示した。また、強充足可能性や段階的充足可能性を効率よく判定することができることがわかった。論理式の構造をあらかじめ調べることにより、強充足可能性や段階的充足可能性の判定をより高速に行えることを明らかにした。しかし、構造が限定的であり、このような性質を持つ構造をより多く発見する必要がある。

(2) 時間論理を用いた実際のシステムの仕様記述による考察
論理式の構造に特徴がある場合、強充足可能性や段階的充足可能性の判定が効率的、そして高速に行えることが明らかになったことから、実際のシステムやプログラムの検証において必要となる論理式の構造を分析した。その分析は、実際に様々なシステムやソフトウェアの検証において必要となる性質を実際に論理式で記述することにより行なった。具体的にはネットワーク機器のソフトウェア、ロボット、自動車制御のネットワークシステムなどの検証すべき性質を記述してその構造を分析した。その結果、論理式の構造よりも、論理式で利用される原子命題の役割が重要であることがわかった。つまり、システムの動作を表す原子命題と利用者やシステムの外界の動作を表す原子命題の論理式の中の位置が重要であることがわかった。

(3) 時間論理による実際のシステムの検証
実際のシステムやプログラムの検証すべき性質を記述した場合の時間論理式の構造の分析を行うのに合わせて、その記述した論理式を用いて実際のシステムやプログラムの検証を行い合わせて開発も行なった。検証を行うことでより安全なシステムやプログラムの開発を行うことができた。

(4) システムの分割による検証の効率化
実際のシステムの検証において、自動車の制御用ネットワークである TCAN の検証を行なった。この検証では、システム全体をモデル化して検証を行う場合、状態爆発を起こしてしまい簡単には検証できなかった。そこで、システムをモジュールに分割して検証を行うことでシステム全体では検証できなかった性質を検証可能にした。この検証方法では検証したい性質に関係があるモジュールとないモジュールに分割して、検証対象となるシステムを小さくすることで検証を可能にしている。本研究では、モジュールの分割を手動で行い、また、分割自体が難しくはなかったため、検証を実施することができた。しかし、モジュールの分割が難しい場合には分割方法を検討する必要がある。

５．主な発表論文等

〔雑誌論文〕　計9件（うち査読付論文　9件／うち国際共著　0件／うちオープンアクセス　0件）

１．著者名 Ishikawa Masato、Yoshiura Noriaki	４．巻 12034
２．論文標題 Privacy Protection in Location Based Service by Secure Computation	５．発行年 2020年
３．雑誌名 Lecture Notes in Computer Science	６．最初と最後の頁 493～504
掲載論文のDOI（デジタルオブジェクト識別子） https://doi.org/10.1007/978-3-030-42058-1_41	査読の有無 有
オープンアクセス オープンアクセスではない、又はオープンアクセスが困難	国際共著 －

１．著者名 Yoshiura Noriaki、Sugiyama Keigo	４．巻 -
２．論文標題 Packet Reachability Verification in OpenFlow Networks	５．発行年 2020年
３．雑誌名 Proceedings of the 2020 9th International Conference on Software and Computer Applications	６．最初と最後の頁 227-231
掲載論文のDOI（デジタルオブジェクト識別子） https://doi.org/10.1145/3384544.3384573	査読の有無 有
オープンアクセス オープンアクセスではない、又はオープンアクセスが困難	国際共著 －

１．著者名 Yoshiura Noriaki、Yano Hayata	４．巻 -
２．論文標題 IP Traceback method by OpenFlow	５．発行年 2020年
３．雑誌名 Proceedings of the 3rd International Conference on Software Engineering and Information Management	６．最初と最後の頁 194-198
掲載論文のDOI（デジタルオブジェクト識別子） https://doi.org/10.1145/3378936.3378965	査読の有無 有
オープンアクセス オープンアクセスではない、又はオープンアクセスが困難	国際共著 －

１．著者名 Kohichi Ogawa, Noriaki Yoshiura	４．巻 11226
２．論文標題 Development of a Support System to Resolve Network Troubles by Mobile Robots	５．発行年 2018年
３．雑誌名 Lecture Notes in Computer Science	６．最初と最後の頁 209-220
掲載論文のDOI（デジタルオブジェクト識別子） https://doi.org/10.1007/978-3-030-02738-4_18	査読の有無 有
オープンアクセス オープンアクセスではない、又はオープンアクセスが困難	国際共著 －

１．著者名 小川康一，吉浦紀晃	４．巻 60
２．論文標題 利用者のネットワーク機器を監視する監視装置との通信品質を改善する移動ロボット制御手法	５．発行年 2019年
３．雑誌名 情報処理学会論文誌	６．最初と最後の頁 779-790
掲載論文のDOI（デジタルオブジェクト識別子） なし	査読の有無 有
オープンアクセス オープンアクセスではない、又はオープンアクセスが困難	国際共著 －

１．著者名 小川康一，吉浦紀晃	４．巻 60
２．論文標題 移動ロボットと小型コンピュータを活用したネットワーク機器監視手法	５．発行年 2019年
３．雑誌名 情報処理学会論文誌	６．最初と最後の頁 668-679
掲載論文のDOI（デジタルオブジェクト識別子） なし	査読の有無 有
オープンアクセス オープンアクセスではない、又はオープンアクセスが困難	国際共著 －

１．著者名 Liu Shuxin, Noriaki Yoshiura	４．巻 10963
２．論文標題 Model Checking of TTCAN Protocol Using UPPAAL	５．発行年 2018年
３．雑誌名 Lecture Notes in Computer Science	６．最初と最後の頁 550-564
掲載論文のDOI（デジタルオブジェクト識別子） https://doi.org/10.1007/978-3-319-95171-3_43	査読の有無 有
オープンアクセス オープンアクセスではない、又はオープンアクセスが困難	国際共著 －

１．著者名 Noriaki Yoshiura	４．巻 10192
２．論文標題 The Relation Between Syntax Restriction of Temporal Logic and Properties of Reactive System Specification.	５．発行年 2017年
３．雑誌名 Lecture Notes in Computer Science	６．最初と最後の頁 105-114
掲載論文のDOI（デジタルオブジェクト識別子） https://doi.org/10.1007/978-3-319-54430-4_11	査読の有無 有
オープンアクセス オープンアクセスではない、又はオープンアクセスが困難	国際共著 －

１．著者名 Date Hiroaki、Yoshiura Noriaki	４．巻 9787
２．論文標題 Computational Verification of Network Programs for Several OpenFlow Switches in Coq	５．発行年 2016年
３．雑誌名 Lecture Notes in Computer Science	６．最初と最後の頁 223～238
掲載論文のDOI（デジタルオブジェクト識別子） https://doi.org/10.1007/978-3-319-42108-7_17	査読の有無 有
オープンアクセス オープンアクセスではない、又はオープンアクセスが困難	国際共著 －

〔学会発表〕　計0件

〔図書〕　計0件

〔産業財産権〕

〔その他〕

－

６．研究組織

	氏名 （ローマ字氏名） （研究者番号）	所属研究機関・部局・職 （機関番号）	備考