2.7.6 システム設計の数理

（1）研究開発領域の定義

本領域は、各種のシステムを設計するための数理的な手法・技法およびその基盤となる数学理論の探求を行うことを目的とした領域である。システム設計のための数理的手法の要諦は、システムが望ましい性質を満たすことを証明する形式検証の営みであり、すなわち「証明を書く」営みである。これら手法の研究は主に以下の3点に主に注目する：（1）対象システムを数学的議論に載せるための「定義」＝モデリングの研究、（2）証明自体の正当性をソフトウェアによって検証するための形式化の研究、（3）証明構築の労力・コストを削減するための自動化の研究。対象となるシステムは拡大し続けており、コンピューターを中心とするシステム（ハードウェア、ソフトウェア、情報など）に加えて、IoTやCPSなど実世界の一部とともに構成されるシステムや、機械学習機能を有するシステム、さらに近年では量子コンピューターなども含まれるようになった。また、数理的手法が担う役割も広がっている。基本的にはシステムの「設計」を行うことが目標であるが、設計されたシステムを解析したり、所望の性質を有するかを検証したりすることも必要であり、実装、解析、検証のための数理的手法の探求も本領域に含まれる。ソフトウェア工学の観点では、数理的手法である「形式手法」を主に扱う。

（2）キーワード

システム設計、情報システム、情報セキュリティー、IoT、CPS、機械学習、自動運転、量子コンピューター、プログラム意味論、圏論、形式手法、ソフトウェア検証、自動証明

（3）研究開発領域の概要

［本領域の意義］

本領域の対象となるシステムとしては、コンピューターを構成するハードウェアおよびソフトウェアのシステム、コンピューターやネットワークを中核とする情報システムが典型的であるが、その範囲は拡大し続けている。IoT（Internet of Things）やCPS（Cyber-Physical System）は、実世界に対するセンサーやアクチュエータを含んでおり、実世界の一部とともにシステムを構成していると考えられる。これらのシステムの設計には、離散的な数理だけはなく、実世界をモデル化するための連続的な数理を必要とする。また近年では、古典物理に基づく現象だけでなく、量子力学に基づく現象を活用したシステムも登場している。量子コンピューターは量子現象を活用して計算を行うシステムであり、量子通信では量子現象を活用して通信ネットワークが構成される。当然ながら、量子力学をモデル化するには実空間だけでなく複素空間が必要となる。さらに近年では、実世界の状況を継続的に計測して学習を行ったり、強化学習の原理に基づいて実世界に働きかけながら学習を行ったりするシステム、すなわち、機械学習機能を有するシステムも一般的になりつつある。

以上のように「システム」の範囲は拡大し続けているが、そのために数理的手法が担う役割も広がっている。基本的にはシステムの「設計」を行うことが目標であるが、設計されたシステムを解析したり、所望の性質を有するかを検証したりすることも必要である。その結果は、システムの再設計に生かされる。また、設計されたシステムを、抽象度の意味でより低レベルのシステムによって「実装」するためにも数理的手法が活用される。実装されたシステムに対しても、解析や検証が必要になる。以上のように、実装、解析、検証のための数理的手法の探求も本領域に含まれる。

「システム」と「設計」の広がりにより、多彩な数理的手法が開発され、その基盤となる数学理論も深く豊かなものに発展している。システム設計における多彩な数理手法の開発とその基盤となる数学理論の深化こそが、本領域を発展させる原動力であり、将来への意義も大きい。

［研究開発の動向］

情報システムの設計と実装のための技術分野としてソフトウェア工学がある。近年、ソフトウェア工学の対象もIoTやCPS、さらに機械学習機能を有するシステムに広がってきており、特に後者を対象とする分野は「機械学習工学」と呼ばれている（「2.1.4 AIソフトウェア工学」も参照）。ソフトウェア工学は、各種の経験的手法も含み、開発チームの構成方法やプロジェクト管理方法などのソフトウェアの開発手法までをも扱っているが、本領域では数学理論に基づく数理的手法である「形式手法」を主に扱う。すなわち、プログラムの意味論および検証、契約によるソフトウェアの設計、検証のための記号論理、それを用いた自動証明などである。

また、コンピューターを構成するシステムの設計手法の中でも、コンピューターのハードウェアの設計手法は個別の技術分野を形成している。状態機械などの基本的な考え方はソフトウェアと共通しているが、古典的なハードウェアのモデル化の手法は確立しており、その設計手法は独自に発展・深化している。本領域にはハードウェアの一般的な設計手法は含めないが、ハードウェアの複雑なロジックに対して、モデル検査などの自動証明の技術が適用され成功を収めている事例も多くあり[1)]、システム設計の数理の応用分野にハードウェアを含めることは適切であろう。

一方、ソフトウェアの設計においてはモデル化自体が非自明であり、さらにIoTやCPSの設計においては、実世界の適切なモデル化も必要となる。モデル化のための各種の数理的手法、モデルの解析・検証のための手法は、対象となるシステムの特徴や目的に即して開発されなければならない。

❶ プログラムの意味論・圏論的意味論

ソフトウェアに対する数理的手法の研究は、プログラミング言語の意味論、プログラムの検証、それらを基礎とするソフトウェアの設計手法などから始まった。プログラミング言語の意味論の定式化をはかる際、その一つのスタイルとして、半順序集合などの数学的構造を用いた表示的意味論がある。表示的意味論はプログラムの意味を数学的に扱いやすい抽象的な形で定式化するのが強みである。一方で、実際のプログラムの実行過程の定式化に近い操作的意味論というスタイルもある。現在ソフトウェア科学の理論研究は、この2つのスタイルの意味論を組み合わせ行き来することで発展している。

一方、データやプログラムの具体的な領域を定めずに、各種の意味論に共通の構成法や共通に成り立つ性質を、圏論を用いて定義する手法が発展してきた。各種の意味論は、圏論的な意味論の具体例として捉えることができる。圏論的意味論は、関数プログラミングの基礎であるラムダ計算の意味論として定義され、さまざまなプログラミング言語の意味論に拡張された。さらに、システムの設計のために必要なさまざまな概念を、圏論を用いて定式化することが行われている[2)]。

❷ プログラムの検証

プログラムの検証の研究は公理的意味論から始まる。特に、この立場（公理的意味論）において、プログラムの正しさを表現するために、述語論理に対してホーアの三つ組を追加したホーア論理が提唱された[3)]。「意味論」と呼ばれているが、プログラムの各構文規則に対して公理が定められる。構文規則を組み合わせて構成されるプログラムが満たす性質は、構文規則の公理を組み合わせて検証することができる。もちろん、公理的意味論における公理は、各種の意味論に基づいて正当化することができる。なお、公理的意味論における公理は、プログラムの最弱事前条件もしくは最強事後条件という形で定式化することもできる。これらもプログラミング言語の意味論から導出される。

❸ 仕様記述・契約による設計

関数、手続き、メソッドなどのプログラムの構成単位を構築する際に、それぞれの構成単位が満たすべき性質を指定し、構成単位を利用するプログラムを、それらの性質のみに基づいて構築しておけば、指定された性質を変えずに構成単位を変更しても、それを利用するプログラムの方は変更する必要がない。構

成単位に対して指定された性質は、その仕様記述と呼ばれる。具体的には、構成単位を呼び出す際の事前条件と、呼び出した後で成り立つべき事後条件を指定する。これらを、構成単位が満たすべき契約と捉え、契約に基づいてソフトウェアを設計する手法は「契約による設計」と呼ばれ、典型的な形式手法として確立している[4]。

❹ 記号論理

プログラムの性質を意味論に基づいて直接的に検証するにせよ、公理を組み合わせて検証するにせよ、検証は数学的な証明を与える作業に他ならない。したがって、何らかの形式体系、すなわち、記号論理のもとで形式的に証明を構築することが可能である。記号論理としては、一階述語論理、様相論理、高階述語論理などが用いられる[5]。以下に述べる型理論から発展した高階型理論が用いられることも多い。また、特定の観点からプログラムの性質を簡潔に表現し検証するために、さまざまな記号論理が定義されている。例えば、時間経過を様相と捉えプログラムの時間的な性質を検証するための時間論理、メモリに関する性質を検証するための分離論理などが典型例である。

❺ 自動証明・静的解析

記号論理のもとでの形式的な証明をある程度自動的に構築することも可能である。一階述語論理、様相論理、高階述語論理、高階型理論などに対して、証明もしくはその一部を自動的に構成する技術が開発されてきた。もちろん、証明が存在するか存在しないかは一般には決定不能であり、証明が存在する場合でも、現実的な時間内に証明を構成できるとは限らない。表現力のより強力な論理ほどその傾向は強くなる。そこで、実用的に有効であって表現力はなるべく弱い論理に対する自動証明技術が発展してきた。具体的には、特定の対象領域に対する述語論理における充足可能性を自動的に判定する技術が発展し、さまざまな応用分野で活用されている（［新展開・技術トピックス］❶SMTソルバによる自動証明の項を参照）。また、並行計算のシステムから得られた状態遷移系の状態を網羅することによって、時間論理で記述された論理式の検証を自動的に行う技術はモデル検査と呼ばれ、実用的な自動証明技術として確立している[6, 7]。

自動証明のような汎用的な技術ではないが、プログラムを実行せずにプログラムの性質を解析する技術が発展している。このような技術は静的解析と呼ばれさまざまな手法が開発されている[8]。特にデータの領域を抽象化して、抽象的な領域においてプログラムを実行してプログラムの解析を行う手法は抽象解釈と呼ばれている。また、各種の型システムを用いて静的解析を行う研究も盛んに行われてきた。形式化された型システムは型理論とも呼ばれる。さらに、述語を用いて抽象領域を構成したり、型システムと述語を組み合わせたりして、上述の充足可能性の自動証明技術を用いて静的解析を行う手法も発展している。

❻ 応用分野

以上で述べた数理的手法は、さまざまな分野のシステムに対して応用されている。それぞれの応用分野に特化した数理的手法が探求された後、より汎用的な手法に一般化され、その結果が他の応用分野に適用され、というサイクルが繰り返されている。典型的な応用分野として、インターネットなどのネットワーク上で稼働する並列分散システムに対する数理的手法が盛んに研究されてきた。特に、並列分散システムを定式化するために各種の並行計算が提案された。また、そのような並列分散システムにおいてセキュリティーに関する性質は極めて重要であり、セキュリティーに特化した数理的手法が開発されてきている。

本領域の意義でも触れたように、IoTやCPS、量子コンピューターからなるシステムや量子コンピューターのソフトウェア、機械学習機能を有するシステムなどが、数理的手法の新たな応用分野となっており、以下で詳しく述べる。特に機械学習機能は多くのプロセスによって実現されるので、各プロセスの設計や自動化、機能全体の構成法などが課題となる。また、機械学習機能が満たす性質や性能を検証して保証することも大きな課題となっている。

最後に、数理的手法が依拠する数学理論自体も数理的手法における形式化や自動証明などの対象となることを付記する。数理的手法が依拠する数学理論に誤りがあれば、数理的手法によって設計され検証されたシステムの正しさは保証されない。数学理論を形式化することにより自動証明などの数理的手法によってその正しさを保証する研究も盛んに行われてきている[9)]。

❼ 諸外国の政策やベンチマーク

日本では、CRESTやERATOなどによる基礎研究が盛んになってきており、応用研究も自動運転やソフトウェアの分野で拡大している。米国はCPSに関する基礎分野で依然として優位であり、応用研究においてはAmazon Web Service（AWS）などのクラウドの検証が活発になってきている。欧州は伝統的に基礎研究に強いが、航空機に対する応用研究も顕著である。中国は中国科学院を中心に。基礎研究力が向上している。この他ではイスラエルが、基礎でも応用でも目立った研究を行っている。

（4）注目動向

［新展開・技術トピックス］

❶ SMTソルバによる自動証明

上述の自動証明の流れの中で、2000年代中盤くらいから（表現能力が非常に限定された）命題論理の自動証明器の性能が飛躍的に向上してきた。これらの自動証明器は、命題論理式の充足可能性（satisfiability）を判定するものでありSATソルバと呼ばれる。さらに2010年代以降、表現能力を命題論理から少しずつ拡張し、応用上頻出する対象（実数、整数、ビットベクトル、リストなど）を表現する述語論理の理論（theory）を対象とする自動証明器が現れた。これらの自動証明器はSMTソルバ（satisfiability modulo theories）と呼ばれる[10)]。

SAT/SMTソルバの性能の向上は、システム検証（システムが所与の性質を満たすことを証明する）において「さまざまな問題をSAT/SMTに帰着させて証明する」というトレンドを引き起こした。

具体的には、システム検証の個々の応用分野ごとに自動証明器を開発することには、応用分野の特性を活かした最適化・チューニングが可能という利点がある。しかし現実には、開発・最適化リソースが分散してしまうので（各応用分野の開発チームは少数）、この利点が発揮されることは少ない。

一方で、SAT/SMTソルバは単純かつ基本的な論理を対象とした自動証明器であるので、多数の開発チームがコンペティションで最適化の技を競い合うことで性能が大きく向上してきた。この性能の優位性は、応用問題をSAT/SMTの問題に帰着する際のオーバーヘッドを補って余りあるものである。

このトレンドは近年多数の成功例を生んでいる。例えば集積回路の形式検証（1994年のPentium FDIVバグ以来Intelが力を入れている）や、ネットワーク設定の形式検証（設定ミスがあるとデータセンターやクラウドが落ちる）、クラウドサービスのセキュリティー設定の形式検証など、各応用分野の巨大な問題インスタンスがSAT/SMTソルバによって解決されている。プログラム検証においても、制約付きホーン節などを通じてSAT/SMTソルバが盛んに用いられている。

❷ 情報セキュリティー

情報セキュリティーの問題は社会的に重要である一方、攻撃者の能力を事前に規定することが難しく、数理的取り扱いに工夫を要する研究課題である。さまざまなシステム・応用分野に対し研究が進んでいる[11)]。

1990年から2000年代にかけては、暗号通信プロトコルの組合せ論的攻撃に対するセキュリティーを検証する研究が注目を集めた（Dolev–Yao攻撃者モデルのstrand space定式化など）。その後研究のトレンドは暗号の計算論的側面を包含し（Abadi–Rogawayなど）、さらに発展している。

各種情報システムのプライバシーも検証対象として注目を集めている。ここでは、差分プライバシーの概

念がプライバシーの数学的定式化として重要である。差分プライバシーの検証のためにプログラミング言語的アプローチが盛んに研究されており（所与のプログラムが差分プライバシーを満たすかをプログラムの文面に基づいて証明する）、関係ホーア論理と呼ばれる論理体系が主に用いられる。

ブロックチェーンの文脈で多く現れるスマートコントラクトの検証も最近盛んな研究トピックである。この問題は応用上の新規性に比べ、既存の（よく研究された）形式検証問題との技術的ギャップが大きくないため、他の問題に帰着させ既存手法で解くというのが一般的アプローチである。例えばプログラム検証のための一般的フレームワークとしてWhy3やKeYがあるが[12]、これらにスマートコントラクト検証問題を帰着させる手法が提案されている[13]。

❸ サイバーフィジカルシステム（CPS）

サイバーフィジカルシステム（CPS）とは、計算機によるデジタル制御と物理的ダイナミクスの融合を指す用語であり、2000年代中盤以降世界的に大きな学術的・産業的潮流となっている。 Industrie 4.0、Society 5.0、IoTなどのパラダイムは、CPSの流れにあると理解することができる[14]。

近年の工業製品のほとんどはCPSであり（自動車、飛行機、発電プラント、ロボットなど）、これらの安全性は、従来の情報システムの場合にも増して重要な課題である。

CPSの研究は当初ソフトウェア科学と制御理論の協働として米国国立科学財団（NSF）の主導で始まった（2000年代中頃）。ここで、ソフトウェア科学は情報システムの数学的解析手法を提供し、制御理論は物理システムの数学的解析手法を提供している。この2つの間に（離散・連続の違いはあるにしろ）明確な数学的類似が見られる、というのがCPS研究の当初の新規性および動機であった。この2分野のつながりはさらに発展し、精度保証付き数値計算の導入や機械学習コンポーネントの解析、制御目標の論理式による記述など、新たな展開が次々に生まれている。

❹ 量子プログラム・システム

量子計算・量子通信は量子力学の原理を用いた新しい計算・通信パラダイムである。多数の状態の重ね合わせを用いて計算複雑性を削減する量子アルゴリズム（Shorの素因数分解アルゴリズムが有名）や、量子もつれを用いて絶対のセキュリティーを保証する量子通信プロトコルが注目を集めている。これらの技術の物理的・アルゴリズム的側面の研究はもちろん重要だが、ソフトウェアおよびシステム設計の観点からも、以下の研究が進んでいる。

2000～2010年代においては、量子プログラミング言語の基礎的設計とその数理的意味論の研究が盛んに行われた。量子計算の最初のモデルである量子回路や量子チューリング機械は、実用上必要な高レベルプログラミングを可能にするものではないため、これらの研究では手続き型や関数型など複数のプログラミング言語が提案された。これら言語のうち多くに共有された指針が「古典制御・量子データ」であり、量子ビットをデータとして扱う一方で、プログラムの制御フローは古典的な実体とすることにより（制御フローの量子的重ね合わせは行わない）実装・解析を容易にするというものである。意味論においては、それ以前の意味論研究の数学的抽象化（特に圏論を用いた抽象化）により、古典プログラミング言語の意味論の抽象化がおおむね量子プログラミング言語のそれも包含し、統一的な枠組みを与えることが示された。

同じ頃、量子通信プロトコルの数理的研究も進んだ。ここでもやはり、古典通信に対する意味論と検証手法の根本が量子通信にも応用可能であることが明らかになった。

しかし2010年代後半以降は以上の研究動向に大きな変化が見られる。この頃、量子ハードウェアが進化して相当数の量子ビットを有するNISQ計算機（Noisy Intermediate-Scale Quantum computer）が出現した。その結果、（量子コンピューターを概して想像上・理論上の存在としていた）従来の基礎的研究で取りあえず無視していたハードウェアの詳細が突然重要な課題となり、それらの対処のための数理的手法が強く求められている。具体的には、限られた量子ビットの効率的再利用や、量子ビットの物理的配

置に起因する量子もつれ生成の制約などが課題である[15]。その他にも量子回路設計の最適化という問題がある[16]。

❺ AutoML

機械学習が多くの場面で有用であることが広く知られる一方で、高い精度でこれらの機能を含むシステムを構築しようとした際には、技術的に精通した人間によるチューニングや管理が不可欠となり、この点が導入や普及において問題となり得る。 AutoML（自動化された機械学習）は、機械学習モデルの設計や構築を自動化すること、あるいはそれに必要な手法全般を指すもので、深層学習が注目される以前から関連分野では盛んに研究されてきた経緯がある[17]。特に最近では、個別の問題に対する深層学習の各プロセスの自動化に必要な手法の研究が盛んに行われ、またこれらを組み入れたサービスも多く見られるようになっている。

機械学習の機能を持つシステムは、データの収集や事前処理、モデルの訓練、推論の実行、モデルの更新など、多くのプロセスから構成される。AutoMLは、これらの各プロセスの自動化のための手法を統合し、技術的に精通した人間にかかるコストを回避可能なシステムの実現を目指すものである。各プロセスの自動化として主要なものとしては、訓練に用いるデータの精錬や拡張、特徴の抽出や選択の自動化や、ハイパーパラメーターの自動チューニング（最適化）などが挙げられる。特にデータ拡張については、敵対的生成ネットワーク（GAN）などの生成モデルを用いたアプローチが盛んに研究されている。またハイパーパラメーターの自動チューニングについては、いわゆるブラックボックス最適化（ベイズ最適化）を用いて、精度を最大化するハイパーパラメーターの探索を行う方法などが、代表的なアプローチとして研究されている。

❻ 機械学習のホワイトボックス化（解釈性向上）

深層学習を中心とした機械学習のさまざまな科学領域への応用や社会実装が進むに伴って、機械学習の解釈可能性、つまりその出力結果がどのような理由で出てくるのか、を人間が理解できるものにするための手法やモデルの研究は、近年ますます注目されるトピックの一つとなっている[18), 19)]。このような研究が注目される背景の一つとしては、機械学習機能を持つシステムを用いた意思決定が、より多様な領域へと広がっていることにもある。例えば、医療などの生命に関わる意思決定や、ビジネスにおいても大きな資金が動く場面において、機械学習が出力する結果だけではなく、その理由も同時に提示することで、より明確な理由をもって意思決定へとつながることが期待できる。しかし一般に、深層学習をはじめとして近年用いられる機械学習モデルは、非線形変換を伴うものを用いることが多い。特に深層学習においては、入力信号が複雑に変換される合成関数となっており、入力から出力に至る信号の変遷を追って解釈を行うことは原理的に困難である。

近年では、このような目的に資する多くの手法が提案されている。例えば、画像認識においては2017年にR.R. Selvarajuらが提案したGradCAMとその発展的手法を中心とした、入力空間における可視化が中心的話題として手法の開発が続けられている。画像に関わらず一般に、個別の出力結果に寄与した変数を重み付けする手法は重要なアプローチとして研究されており、代表的な手法としては、2016年にM.T. Ribeiroらにより提案された摂動に基づく手法であるLIME[20)]や、2017年にS.M. Lundbergらにより提案されたShapley Valueに基づくSHAP[21)]などが知られる。また、（線形モデルなどの）解釈可能な代理モデルを構築し、これにより、より複雑なモデルの出力結果を解釈する手法も重要なアプローチとして盛んに研究されている[22), 23)]。

❼（システム設計手法の応用分野としての）純粋数学

2000年代後半以来、システム設計のための数理的手法（定義で述べた（1）、（2）、（3））の進歩はそもそもの数学コミュニティー（「純粋数学」）の知るところとなり、純粋数学への逆輸入の試みが盛んに行

われている。すなわち、純粋数学における証明は現在非形式なものが主流であるが（紙に書く証明でありソフトウェア的確認ができない）、その正当性確認のため形式化を行おうという流れである。特に有名な試みとして、ホモトピー理論の形式化を目指すホモトピー型理論の研究が盛んである[24)]。

［注目すべき国内外のプロジェクト］

❶ SMTソルバによる自動証明：産業界での利用

「さまざまな問題をSAT/SMTに帰着させて、近年性能向上が著しいSAT/SMTソルバで解く」というトレンドは、その実用性の高さにより、学術研究のみならず産業界の現場で多数の実施例が見られる。例えば集積回路の形式検証はIntelが20年以上取り組んでいるトピックであり、ネットワーク設定やクラウドサービスセキュリティー設定の形式検証などはAmazon Web Service（AWS）の取り組みが近年目立っている。またMicrosoftは形式検証のさまざまなトピックに力を入れており、近年の代表的なSMTソルバであるZ3はMicrosoftによって開発されオープンソースになった。

❷ サイバーフィジカルシステム（CPS）

CPSの品質・安全性保証は、Industrie 4.0やIoTなどの関連パラダイムも含め大きな注目を集めている。特に米国ではNSFが主導してCyber–Physical Systems Virtual Organization（CPS–VO）を組織して産官学の研究活動を統括し、多数の大型研究プロジェクトが実施されている。欧州でも、EUが実施する研究・イノベーションプログラムであるHorizon 2020（2014～2020年）およびその後継のHorizon Europe（2021～2027年）を通じて、多数の研究プロジェクトへの助成が行われている。国内では、ERATO蓮尾プロジェクト（総括：蓮尾一郎（国立情報学研究所）、2016–2024年度）、CREST CyPhAIプロジェクト（代表：末永幸平（京都大学）、2020–2025年度）などを通じてCPSの数理的研究に助成が行われている。

産業界からの関心も当然高い。例えば当該分野の主要国際会議CAV、CPS–IoT Weekなどでは、トヨタ自動車、DENSO、Bosch、SIEMENSなどがしばしばスポンサーとなっている。

自動運転はCPSの重要なサブトピックであり、各国および多数の企業が研究を行っている。特に安全性保証の研究に注目すると、ドイツPegasus Projectと日本SAKURAプロジェクトが産官学の取り組みとして目立っている。これらの取り組みはテストによる統計的安全性保証のアプローチを主眼としている一方で、論理的形式検証のアプローチがIntel/Mobileyeによって提唱されており、IEEE 2846などの国際規格化の動きも見られる。

❸ 形式検証の純粋数学応用：ホモトピー型理論の大規模プロジェクト

米国防総省のMultidisciplinary University Research Initiative（MURI）programのプロジェクトの一つとして、総額750万ドルのホモトピー型理論の研究プロジェクトが2014～2019年に実施された。Carnegie Mellon University哲学科のSteve Awodey教授が代表者となった本プロジェクトには、形式検証・論理学・純粋数学という異なる分野から多数の研究者が参加し、トピックの包括的文献であるHomotopy Type Theory（“The HoTT Book”）の出版など多数の成果を上げた。ホモトピー型理論の研究は圏論的意味論の研究とも融合し、今日に至るまで多数の研究者によって追究されている。

（5）科学技術的課題

❶ SMTソルバ応用：さらなる対象領域拡大に向けて

今日のSMTソルバの性能向上は目を見張るものがあり、多数のシステム解析問題をSMTに帰着させて解くことの成功例の多さは上述の通りである。 SATソルバの用途は組合せ論的問題（ブール値真偽値の命題論理式で表現できる問題）に限られていたが、SMTソルバでは実数やリストなどの値に関する証明が可

能であり、潜在的な応用範囲はさらに広い。

その一方で、現状ではSMTソルバの活用が主に研究者の手に限られているのも事実である。 SMTソルバを経由して解ける問題は、集積回路やネットワークなど高度なIT応用領域にとどまらず、あらゆる産業領域やサービスの現場、日常生活などに遍在していると考えられる。よってこれらに対してSMTソルバを大規模活用し、さまざまな問題を高速・厳密に解くことは、大きな社会的インパクトを持つ。

そのためには、SMTソルバ自体のさらなる効率化の他に、SMTソルバ応用の敷居を下げる研究開発も必要である。この研究開発課題は、UI（ユーザーインターフェース）のさらなる改良から、適切な中間言語の設定、汎用的な問題帰着スキームの定義など多岐にわたる。

❷ 情報セキュリティー：ワンオフの検証から、逐次的改良によるロバストなセキュリティーへ

近年情報システムは急速に巨大化・多様化する現在、伝統的な情報セキュリティー検証の仮定の正当性が崩れつつある。すなわち、IoTシステムに代表される「計算能力が限られたデバイスが膨大な数接続されて一つのシステムを形成する」状況においては、サイドチャネルアタックやその他の脅威を通じて一定数のデバイスが乗っ取られている状況は自然であり、Dolev–Yao攻撃者モデルのように攻撃者の能力を明確に限定するのは非現実的である。よって今日の情報セキュリティー検証においては、当初想定した攻撃者モデルのもとでセキュリティーを証明するだけでなく、攻撃者モデルの誤り（＝想定しなかった脅威）に対処してシステム・証明を改訂する頑健性（ロバストネス）をどう実現するかも重要な課題となる。

❸ サイバーフィジカルシステム（CPS）：モデリングの課題

従来のCPSの検証手法はソフトウェア科学と制御理論の融合によるものであるが、これらはシステムのホワイトボックスモデリングを必要とするものが多い。すなわち、システムの内部動作原理の数学的記述に基づいて、その性質・安全性を数学的に議論する、というわけである。

しかし現実のCPSの多くではホワイトボックスモデリングは困難か不可能である。自動車では内燃機関（微分方程式による記述が非常に複雑）や外部購入部品（動作原理が開示されず）などが障壁になり、仮にこれらの障壁がなくてもモデリングには膨大な工数がかかる。ニューラルネットワークなどの統計的機械学習ユニットを論理的解析ができる形にモデリングすることも困難である。さらに、自動運転などのマルチエージェントシステムでは、周囲の物理環境や他車・歩行者の動作など、モデリングが困難な要素が非常に多い。

よってCPSの検証のさらなる発展と実システム応用には、モデリングの困難さの課題の対処が必要不可欠である。論理的検証の視点からはこれは非常に大きな課題であり（モデルがない ⇒ 定義がない ⇒ 証明が書けない）、基礎的研究が必要とされる。

この課題の対処法の具体例として、自動運転安全性の論理的証明のための方法論であるRSS（Responsibility–Sensitive Safety）を挙げる。 RSSは、自動運転という個別の応用領域に注力して「モデリングする/しない」の境界を注意深く設定することにより、実用上有効な論理的証明を得ることに成功している[25), 26)]。これは、ブラックボックスシステムを論理的安全エンベロープで包むことでシステム全体としての安全性を確保している、と言い換えることもできる。

❹ 実応用を見据えた圏論的研究の深化

圏論は現代数学の多くの分野で用いられる抽象言語であり、種々の数学的概念を（その構成でなく）他の対象との関連性を用いて記述することで、数学的理論の本質的構造を明らかにする。数学では特に、異なる分野に現れる概念の間の類似性を定式化する際に強力な道具となる。

情報学およびシステム設計の技法においても、圏論は圏論的意味論において盛んに用いられ、多くの成果を上げてきた。例えば抽象言語としての圏論はHaskellなどのプログラミング言語の設計にインスピレー

ションを与え、プログラムの抽象化と生産性の向上に貢献してきた。また圏論の抽象性・一般性により、一つの検証手法を他の種類のシステムに横展開することが可能になる。これは（［新展開・技術トピックス］❹量子プログラム・システムなど）新たな計算パラダイムが現れた際に、既存の解析手法を適用する有力な手助けとなる。

上記の利点などの理由で、システム設計の数理一般において圏論への関心は（多数ではないにしろ）根強い。一方、圏論応用の試みは単なる抽象的・一般的記述に終わることも少なくなく、そこから応用上の価値を実際に引き出すためには、抽象論を応用上の現実にマッチさせる労力が必要となる。量子や機械学習など新たな計算パラダイムが次々現れて、圏論の一般化力が期待される今こそ、逆に応用上の具体的課題を深く理解して抽象論とつなげる泥臭い努力が必要とされている。

❺ 機械学習の数学的理解

機械学習の一部のモデル、特に深層ニューラルネットについては、高い経験的な精度が実現されることが知られる一方、その理由についてはまだ分かっていない部分も多い。上述のように、今後ますます機械学習の機能を組み込んだシステムが社会へ普及し利用場面が広がっていくことが予想され、その中で、機械学習の出力結果の解釈可能性の付与や、機械学習プロセスのホワイトボックス化は技術的要素としてより重要な課題となっていくと言える。このような課題において、機械学習モデルが、どのような特徴量の学習を行っているのか、なぜ高い汎化性能（新しいデータに対する予測性能）を得られるのか、また、理論的にはどのような場合に古典的な機械学習モデルに対して優位であるのかなど、多くの未解決な課題を解決していく必要がある。

例えば、古典的な学習の理論では、学習モデルの複雑さを大きくしていくと、いわゆる過学習（データ中の雑音情報もモデルに組み込んでしまう）が起こるために汎化性能が低下することが知られている。しかし近年では、モデルの複雑さをより大きく上げることで、二重降下という現象が起きて、さらに汎化性能が向上することが確認されている[8), 2)]。その数理的な理解については、まだ十分に解明されていないものの、このような現象が深層学習の高い性能を生み出す原因であることが少しずつ分かってきている。また、深層学習は入力から出力への変換を繰り返す合成関数であることは上述の通りである。しかしその変換は一様というわけではなく、入力に近い層では特徴学習、そして出力に向けてその特徴量を用いた識別関数を学習していると言われている。最近では、この仕組みに関する数学的理解も進んできており、これにより深層学習が得意なデータやタスクについても少しずつ分かってきている。

（6）その他の課題

❶ ファンディング

システム設計の数理的研究は、（1）数理的手法をシステム設計の実課題に適合させ応用する「フロントエンド研究」と（2）抽象的・一般的で多数の実課題に展開可能なポテンシャルを持つ数理的手法自体の「バックエンド研究」の2つのフェーズからなる。後者のバックエンド研究の重要性は非常に大きく、研究エフォート分担の面でも横展開によるインパクトの面でも、少なくとも（フロントエンドの研究者も含む）当該分野の研究者はこれらの重要性を強く認識している。研究エフォートという点では、高度に数学的な研究には専従エフォートが必要である。また、横展開の点では、一般理論の横展開による理論研究エフォートの節約および新応用・新領域の創出というインパクトがある。

しかし、ファンディング審査に代表されるような分野外へのアピールという点では、バックエンド研究はフロントエンド研究という1ステップを経てのみ実応用に到達できるため、その必要性が理解されにくいことが多い。直接役に立たない研究、理解の難しい研究をなぜ助成する必要があるのか？が問われてしまうためである。このような理論的バックエンドの助成におけるハードルは、数学応用一般における課題でもある。解決策としては、ファンディング主体の積極的働きかけを行うことがある。すなわち、長期的な科学技

術振興を担う学術研究だからこそバックエンド研究にも力を入れるべきだと主張することである。加えて、バックエンド研究の直接のユーザーたるフロントエンド研究者がフロントエンド研究とのマッチングによる重要性のアピールすることなどが考えられる。

❷ 研究組織

システム設計の数理的研究は数学の社会応用の一つの形であり、いわゆる応用数学の諸分野との交流・協働を推進することで、理論自体のみならず数学応用の方法論の共有と発展をはかることが望まれる。しかし現状では「いわゆる応用数学」の主流は数学的には解析や統計であり、システム設計で用いる数理論理学や数学基礎論のみならず、コミュニティー的に距離がある。研究者個人レベルでのさらなる交流と相互理解が望まれる。また同時に、ファンディングや研究組織などを通じた組織的取り組みの推進も待たれる。

❸ 産学連携

実社会のICTシステムや物理システムが急速に複雑化する現在、システム設計の数理的研究の成果に対する産業界のニーズは非常に高い。しかし、学術研究における価値を産業界のニーズの解決に振り向ける際にはギャップが多い。例えば、産学の協議による課題の洗い出しと定式化、契約事務、成果・知財の切り分け、数学的定式化による実世界ノイズへの対処などがギャップの具体例である。これらギャップへの対処には研究者側に相当のノウハウが必要とされ、ただでさえ研究時間の少なくなっている大学教員にとって、新たに産学連携に乗り出すための時間を捻出するのは難しいのが現状である。

一方でシステム設計の数理的研究においては、産業界のニーズこそが数学的理論の発展を促す発想のタネの主要なものである。よって、産業応用を通じた社会貢献の点でも、さらなる学術的発展の点でも、産学連携の積極的推進が望まれる。具体策としては（1）大学教員の研究時間一般の確保、（2）産学連携のノウハウの体系的共有、（3）機関やファンディングを通じた契約事務や知財活動のサポート（知財に関しては独立行政法人 工業所有権情報・研修館の知財プロデューサー派遣事業などの例がある）、（4）ケーススタディー論文を学術的に評価するようにする、などが考えられる。

❹ 分野連携

研究組織の項で述べたように、応用数学の他分野との数学的交流や方法論的交流が望まれる。また、実社会のシステムの複雑性を鑑みるに、これらの解析や品質保証を数理的側面のみで語ることは不可能であり、より経験的・実践的な研究分野との密接な協働が望まれる（例えばソフトウェア工学やシステム工学など）。

❺ 人材育成

システムが複雑化し、またセーフティクリティカルさの度合いが上がるに従い、システム安全性の社会的重要性が年々増している。またここでは、自動運転の例でも明らかなように、安全・高品質なシステムを作るだけでは十分でなく、安全・高品質であることを顧客や社会に説明し信頼を勝ち取ることが、新技術の社会受容のために必須である。

システム設計の数理はこのようなシステムと社会のインターフェースの基礎理論であり、システム設計に関わる多くの技術者――まずはソフトウェア技術者および制御系技術者――がシステム設計の数理の基礎的理解を持つことが望まれる。そのためには、大学などの講義カリキュラムが当該内容をカバーすることが必要である。また、企業に就職する学生の研究指導においては、数理的理論と実社会ニーズのすり合わせの経験を積ませることが有益である。

（7）国際比較

国・地域	フェーズ	現状	トレンド	各国の状況、評価の際に参考にした根拠など
日本	基礎研究	○	↗	数学のソフトウェア研究分野における応用については、ERATOやCRESTなどのプロジェクトを通じたJSTの支援がここ数年目立っている。制御理論やソフトウェア工学との協働も盛んになっているが、特に制御理論との協働の源流は、FIRST合原プロジェクトを始めとする先行の取り組みに求められる。システムセキュリティー・ICT基盤への応用にも注目が集まっており、2021年に文科省の戦略目標が設定された。
	応用研究・開発	○	↗	純粋な情報処理システムに比べて、工業製品・ロボティクスなどのCPSへの応用が目立っている。自動運転の安全性の研究が好例。システムセキュリティー・ICT基盤への応用についても、物理コンポーネントを含むIoTシステムが強調されることが多い。
米国	基礎研究	○	→	数学のシステム設計への応用のうち、こと論理学的な理論については、伝統的に米国よりも欧州・日本の方が盛んである。一方、SMTや対話型定理証明など、応用に直結する理論的研究は非常に盛ん。2000年代中頃に始まったCPS研究の流れは、当初の方法論がホワイトボックス必須であったことから、再検討の段階に来ている。
	応用研究・開発	◎	↗	スケーラブルな自動検証手法としてのSMTソルバの活用が進んでいる。AWSなどの大規模ウェブサービスの安全性やセキュリティー検証のため実際に応用されており、今後さらに広範な展開が予想される。一方、CPSの安全性（自動運転など）や情報プライバシーにおいては、論理的な保証・説明への社会的ニーズが欧州・日本ほど高くなく、産業応用の事例もそう多くない。
欧州	基礎研究	◎	→	ICTシステムの研究においては、伝統的に米国よりも数学的理論に重みを置く。 University of Oxford、ENS Paris、ETH Zürichなどの有名大学のみならず、RWTH Aachen University、Aalborg Universityなど、強みを持つ大学が分散している。ドイツ Max Planck Institute for Software SystemsやIST Austriaなどの研究所も強い。企業ではMicrosoft Researchが伝統的に基礎研究に力を入れている。
	応用研究・開発	○	↗	数学的理論の研究成果の応用事例がAirbus、Boschなどの製造業を始め多く見られる。英国にスタートアップが多く、数学的理論を用いてソフトウェアの品質保証を行う大学発スタートアップ2社がFacebookやGithubに買収されExitした。自動運転安全性保証技術のスタートアップも英国に多い。
中国	基礎研究	○	↗	中国科学院を中心に近年プレゼンスの向上が著しい。米国流の応用駆動型研究ももちろん、数学的理論においても、欧州で博士号をとった研究者が中国で成果を上げている。
	応用研究・開発	△	→	企業のケーススタディーなどが表に出てくることは少なく、活動実態が見えづらい。
韓国	基礎研究	△	→	ソウル大学校、KAISTなどに目立つ研究グループがあるが、コミュニティー全体におけるプレゼンスは高くない。
	応用研究・開発	△	→	ソフトウェアシステムや製造業への応用の事例は多くない。
台湾	基礎研究	△	→	数学的理論の研究においてのプレゼンスは高くない一方、主要産業たる集積回路に関連するSAT/SMTソルバの研究は盛んである。
	応用研究・開発	○	→	集積回路設計への応用が進んでいる。
イスラエル	基礎研究	○	→	伝統的に論理学研究が強く、Hebrew University of Jerusalem、Technion–Israel Institute of Technologyなどのグループが世界的に目立っている。
	応用研究・開発	◎	→	集積回路応用（Intel）、自動運転（Intel/Mobileye）など、いくつかの分野で世界トップのvisibilityを有する。

（註1）フェーズ

基礎研究：大学・国研などでの基礎研究の範囲

応用研究・開発：技術開発（プロトタイプの開発含む）の範囲

（註2）現状　※日本の現状を基準にした評価ではなく、CRDS の調査・見解による評価

◎：特に顕著な活動・成果が見えている　　○：顕著な活動・成果が見えている

△：顕著な活動・成果が見えていない　　×：特筆すべき活動・成果が見えていない

（註3）トレンド　※ここ1～2年の研究開発水準の変化

↗：上昇傾向、→：現状維持、↘：下降傾向

参考文献

1) Aarti Gupta, Malay K. Ganai, and Chao Wang, “SAT-Based Verification Methods and Applications in Hardware Verification,” in *Formal Methods for Hardware Verification: SFM 2006*, eds. Marco Bernardo and Alessandro Cimatti, Lecture Notes in Computer Science 3965（Berlin, Heidelberg: Springer, 2006）, 108-143., https://doi.org/10.1007/11757283_5.
2) Bart Jacobs, *Introduction to Coalgebra: Towards Mathematics of States and Observation*, Cambridge Tracts in Theoretical Computer Science 59（Cambridge: Cambridge University Press, 2016）., https://doi.org/10.1017/CBO9781316823187.
3) Glynn Winskel, *The Formal Semantics of Programming Languages: An Introduction*（MIT Press, 1993）.
4) Bertrand Meyer, “Applying 'design by contract',” *Computer* 25, no. 10（1992）: 40-51., https://doi.org/10.1109/2.161279.
5) 萩谷昌己, 西崎真也『論理と計算のしくみ』（東京: 岩波書店, 2007）.
6) 中島震『SPIN モデル検査：検証モデリング技法』（東京: 近代科学社, 2008）.
7) Christel Baier and Joost-Pieter Katoen, *Principles of Model Checking*（MIT Press, 2008）.
8) Vijay D’Silva, Daniel Kroening, and Georg Weissenbacher, “A Survey of Automated Techniques for Formal Software Verification,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 27, no. 7（2008）: 1165-1178., https://doi.org/10.1109/TCAD.2008.923410.
9) 萩原学, アフェルト・レナルド『Coq/SSReflect/MathCompによる定理証明：フリーソフトではじめる数学の形式化』（東京: 森北出版, 2018）.
10) Neha Rungta, “A Billion SMT Queries a Day（Invited Paper）,” in *Computer Aided Verification: CAV 2022*, eds. Sharon Shoham and Yakir Vizel, Lecture Notes in Computer Science 13371（Springer Cham, 2022）, https://doi.org/10.1007/978-3-031-13185-1_1.
11) David Basin, et al., “Tamarin: Verification of Large-Scale, Real-World, Cryptographic Protocols,” *IEEE Security & Privacy* 20, no. 3（2022）: 24-32., https://doi.org/10.1109/MSEC.2022.3154689.
12) François Bobot, et al., “Let’s verify this with Why3,” *International Journal on Software Tools for Technology Transfer* 17（2015）: 709-727., https://doi.org/10.1007/s10009-014-0314-5.
13) Luís Pedro Arrojado da Horta, et al., “A tool for proving Michelson Smart Contracts in WHY3,” in *2020 IEEE International Conference on Blockchain (Blockchain)*（IEEE, 2020）, 409-414., https://doi.org/10.1109/Blockchain50366.2020.00059.
14) 奥村洋「CPS研究の世界的潮流と日本の現状」『研究 技術 計画』32 巻 3 号（2017）: 251-265., https://doi.org/10.20801/jsrpim.32.3_251.

15) Anouk Paradis, et al., “Unqomp: synthesizing uncomputation in Quantum circuits,” in *Proceedings of the 42nd ACM SIGPLAN International Conference on Programming Language Design and Implementation*（New York: Association for Computing Machinery, 2021）, 222-236., https://doi.org/10.1145/3453483.3454040.
16) 山下茂「量子回路設計における最適化問題」『電子情報通信学会 基礎・境界ソサイエティ：Fundamentals Review』14 巻 4 号（2021）：337-346., https://doi.org/10.1587/essfr.14.4_337.
17) Xin He, Kaiyong Zhao, and Xiaowen Chu, “AutoML: A survey of the state-of-the-art,” *Knowledge-Based Systems* 212（2021）：106622., https://doi.org/10.1016/j.knosys.2020.106622.
18) Christoph Molnar, *Interpretable Machine Learning: A Guide for Making Black Box Models Explainable*（2nd ed.）. https://christophm.github.io/interpretable-ml-book/
19) Cynthia Rudin, et al., “Interpretable machine learning: Fundamental principles and 10 grand challenges,” *Statistics Surveys* 16（2022）：1-85., https://doi.org/10.1214/21-SS133.
20) Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin, ““Why Should I Trust You?”: Explaining the Predictions of Any Classifier,” in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*（New York: Association for Computing Machinery, 2016）, 1135-1144., https://doi.org/10.1145/2939672.2939778.
21) Scott M. Lundberg and Su-In Lee, “A Unified Approach to Interpreting Model Predictions,” Advances in Neural Information Processing Systems 30（NIPS 2017）, https://papers.nips.cc/paper/2017/hash/8a20a8621978632d76c43dfd28b67767-Abstract.html,（2023年3月8日アクセス）.
22) Mikhail Belkin, et al., “Reconciling modern machine-learning practice and the classical bias-variance trade-off,” *PNAS* 116, no. 32（2019）：15849-15854., https://doi.org/10.1073/pnas.1903070116.
23) Preetum Nakkiran, et al., “Deep double descent: where bigger models and more data hurt,” *Journal of Statistical Mechanics: Theory and Experiment* 2021（2021）：124003., https://doi.org/10.1088/1742-5468/ac3a74.
24) The Univalent Foundations Program, Institute for Advanced Study, *Homotopy Type Theory: Univalent Foundations of Mathematics*（The Univalent Foundations Program, 2013）.
25) Shai Shalev-Shwartz, Shaked Shammah, and Amnon Shashua, “On a Formal Model of Safe and Scalable Self-driving Cars,” arXiv, https://doi.org/10.48550/arXiv.1708.06374,（2023年3月8日アクセス）.
26) Ichiro Hasuo, et al., “Goal-Aware RSS for Complex Scenarios Via Program Logic,” *IEEE Transactions on Intelligent Vehicles*（2022）., https://doi.org/10.1109/TIV.2022.3169762.