(19)日本国特許庁(JP)　(12)公開特許公報(A)　(11)特許出願公開番号
特開2016-57969
(P2016-57969A)
(43)公開日　平成28年4月21日(2016.4.21)

(51)Int.Cl.		F I			テーマコード（参考）
G06F 11/28	(2006.01)	G06F	11/28	340A	5B042
G06F 11/36	(2006.01)	G06F	9/06	620M	5B376

審査請求 未請求　請求項の数 12　ＯＬ　（全 19 頁）

(21)出願番号　特願2014-185326 (P2014-185326)
(22)出願日　平成26年9月11日 (2014.9.11)

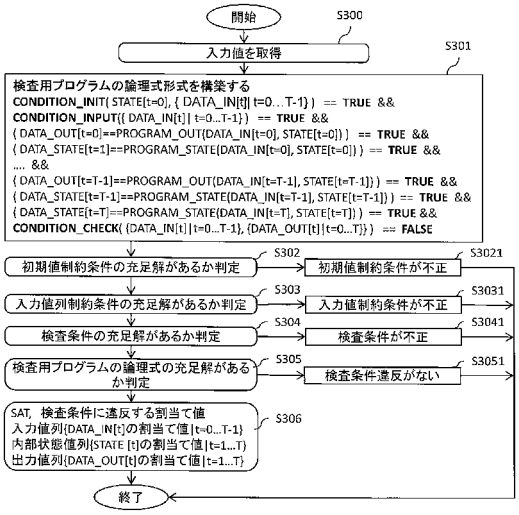
(71)出願人　509186579
日立オートモティブシステムズ株式会社
茨城県ひたちなか市高場２５２０番地
(74)代理人　100091096
弁理士　平木　祐輔
(74)代理人　100105463
弁理士　関谷　三男
(74)代理人　100102576
弁理士　渡辺　敏章
(72)発明者　西　昌能
東京都千代田区丸の内一丁目６番６号　株式会社日立製作所内
(72)発明者　成沢　文雄
東京都千代田区丸の内一丁目６番６号　株式会社日立製作所内

最終頁に続く

(54)【発明の名称】プログラム検査装置、ソフトウェア検査装置、ＳＡＴ制約条件データ、記憶媒体

(57)【要約】　（修正有）
【課題】検査対象ソフトウェアに対して入力する異常な入力値列をあらかじめ排除することにより、ソフトウェアの挙動を検査するために必要な計算リソースを抑制するプログラム検査装置、ソフトウェア検査装置、ＳＡＴ制約条件データ、記憶媒体を提供する。
【解決手段】プログラム検査装置は、検査対象である動的プログラムの入力値列、内部状態値列、出力値列、及びソフトウェアの要件を記述した検査条件を、充足可能性問題の制約条件として記述し、所望の検査条件を満たす解を探索することにより、検査に適した入力値列を出力値列から逆算的に求める。
【選択図】図３

図3



【特許請求の範囲】
【請求項１】
入力値にしたがって内部状態値を更新するとともに前記入力値に対応する出力値を出力する処理を実装した動的プログラムを検査するプログラム検査装置であって、
指定した制約条件を充足することができる解を求めることにより前記動的プログラムの前記入力値、前記内部状態値、および前記出力値を検査するＳＡＴソルバを備え、
前記ＳＡＴソルバは、
前記内部状態値を前記動的プログラムに保持させつつ入力値列内の入力値を前記動的プログラムに対して順次引き渡して実行することにより求められる前記動的プログラムの内部状態値列と出力値列を前記入力値列とともに記述した動的プログラム記述、
前記入力値列が満たすべき検査条件、前記内部状態値列が満たすべき検査条件、および前記出力値列が満たすべき検査条件の少なくともいずれかを記述した検査条件記述、
を前記制約条件として受け取り、前記制約条件に違反する前記入力値列、前記制約条件に違反する前記内部状態値列の初期値、および前記制約条件に違反する前記出力値列の少なくともいずれかが存在するか否かを前記解として探索し、前記解が存在しない場合はその旨の処理結果を出力し、前記解が存在する場合はその解に対応する前記入力値列、前記内部状態値列、および前記出力値列を出力する
ことを特徴とするプログラム検査装置。
【請求項２】
前記ＳＡＴソルバは、前記制約条件としてさらに、
前記内部状態値の初期値が満たすべき条件を記述した初期値制約条件記述、および
前記動的プログラムに対して引き渡す前記入力値列が満たすべき条件を記述した入力値列制約条件記述、
を受け取り、
前記ＳＡＴソルバは、
前記検査条件を満たす前記解が存在しない場合は前記検査条件自体が適正でない旨の処理結果を出力し、
前記初期値制約条件記述が記述する条件を満たす前記解が存在しない場合はその条件自体が適正でない旨の処理結果を出力し、
前記入力値列制約条件記述が記述する条件を満たす前記解が存在しない場合はその条件自体が適正でない旨の処理結果を出力する
ことを特徴とする請求項１記載のプログラム検査装置。
【請求項３】
前記検査条件、および前記入力値列制約条件記述が記述している条件は、前記入力値列内の入力値毎に定義されている
ことを特徴とする請求項２記載のプログラム検査装置。
【請求項４】
請求項１記載のプログラム検査装置を用いて前記動的プログラムを検査するソフトウェア装置であって、
前記内部状態値を前記動的プログラムに保持させつつ入力値列内の入力値を前記動的プログラムに対して順次引き渡して実行することにより前記内部状態値列と前記出力値列を求める検査用プログラム構築部、
前記入力値列が満たすべき検査条件、前記内部状態値列が満たすべき検査条件、および前記出力値列が満たすべき検査条件の少なくともいずれかを受け取り、前記検査条件記述に変換する、検査条件設定部、
を備え、
前記検査用プログラム構築部は、求めた前記内部状態値列と前記出力値列を前記動的プログラム記述として前記プログラム検査装置に対して引き渡し、
前記検査条件設定部は、前記変換した検査条件記述を前記プログラム検査装置に対して引き渡す

ことを特徴とするソフトウェア検査装置。
【請求項５】
請求項２記載のプログラム検査装置を用いて前記動的プログラムを検査するソフトウェア装置であって、
前記内部状態値を前記動的プログラムに保持させつつ入力値列内の入力値を前記動的プログラムに対して順次引き渡して実行することにより前記内部状態値列と前記出力値列を求める検査用プログラム構築部、
前記入力値列が満たすべき検査条件、前記内部状態値列が満たすべき検査条件、および前記出力値列が満たすべき検査条件の少なくともいずれかを受け取り、前記検査条件記述に変換する、検査条件設定部、
前記内部状態値の初期値が満たすべき条件を受け取り、前記初期値制約条件記述に変換する、初期値制約設定部、
前記動的プログラムに対して引き渡す前記入力値列が満たすべき条件を受け取り、前記入力値列制約条件記述に変換する、入力値列制約設定部、
を備え、
前記検査用プログラム構築部は、求めた前記内部状態値列と前記出力値列を前記動的プログラム記述として前記プログラム検査装置に対して引き渡し、
前記検査条件設定部は、前記変換した検査条件記述を前記プログラム検査装置に対して引き渡し、
前記初期値制約設定部は、前記変換した初期値制約条件記述を前記プログラム検査装置に対して引き渡し、
前記入力値制約設定部は、前記変換した入力値列制約条件記述を前記プログラム検査装置に対して引き渡す
ことを特徴とするソフトウェア検査装置。
【請求項６】
前記検査条件設定部は、前記入力値列、前記内部状態値列、および前記出力値列のうち少なくともいずれかが満たすべき前提条件および拘束条件を、論理演算式の形式で前記検査条件記述として生成する
ことを特徴とする請求項４記載のソフトウェア検査装置。
【請求項７】
前記入力値制約設定部は、前記入力値列が満たすべき前提条件および拘束条件を、論理演算式の形式で前記入力値列制約条件記述として生成し、
前記初期値制約設定部は、前記内部状態値の初期値が満たすべき前提条件および拘束条件を、論理演算式の形式で前記初期値制約条件記述として生成する
ことを特徴とする請求項５記載のソフトウェア検査装置。
【請求項８】
前記検査条件設定部は、ビット論理演算式の形式で前記検査条件記述を生成する
ことを特徴とする請求項６記載のソフトウェア検査装置。
【請求項９】
前記入力値制約設定部は、ビット論理演算式の形式で前記入力値列制約条件記述を生成し、
前記初期値制約設定部は、ビット論理演算式の形式で前記初期値制約条件記述を生成する
ことを特徴とする請求項７記載のソフトウェア検査装置。
【請求項１０】
指定した制約条件を充足することができる解を求めることにより、入力値にしたがって内部状態値を更新するとともに前記入力値に対応する出力値を出力する処理を実装した動的プログラムの前記入力値、前記内部状態値、および前記出力値を検査するＳＡＴソルバに対して入力する前記制約条件を記述したＳＡＴ制約条件データであって、
前記内部状態値を前記動的プログラムに保持させつつ入力値列内の入力値を前記動的プ

ログラムに対して順次引き渡して実行することにより求められる前記動的プログラムの内部状態値列と出力値列を前記入力値列とともに記述した動的プログラム記述、
前記入力値列が満たすべき検査条件、前記内部状態値列が満たすべき検査条件、および前記出力値列が満たすべき検査条件の少なくともいずれかを記述した検査条件記述、
を含み、
前記検査条件記述は、前記入力値列、前記内部状態値列、および前記出力値列のうち少なくともいずれかが満たすべき前提条件および拘束条件を、論理演算式の形式で記述している
ことを特徴とするＳＡＴ制約条件データ。
【請求項１１】
前記ＳＡＴ制約条件データはさらに、
前記内部状態値の初期値が満たすべき条件を記述した初期値制約条件記述、および
前記動的プログラムに対して引き渡す前記入力値列が満たすべき条件を記述した入力値列制約条件記述、
を含み、
前記入力値列制約条件記述は、前記入力値列が満たすべき前提条件および拘束条件を、論理演算式の形式で記述しており、
前記初期値制約条件記述は、前記内部状態値の初期値が満たすべき前提条件および拘束条件を、論理演算式の形式で記述している
ことを特徴とする請求項１０記載のＳＡＴ制約条件データ。
【請求項１２】
請求項１０記載のＳＡＴ制約条件データを格納したことを特徴とする記憶媒体。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、ソフトウェアを検査する技術に関する。
【背景技術】
【０００２】
近年、モデル検査手法と呼ばれる入出力関係を動的に解析する手法が効果的であることが知られるようになってきた。モデル検査手法は、ソフトウェアに対する入力値とソフトウェアの内部状態により出力値が動的に決定されるソフトウェアモデルを構築し、モデルから時系列に出力される出力値列として観察されるソフトウェアの動的挙動の妥当性を評価する手法である。これにより、機能要件および安全性要件に違反する不具合に相当する入力値または内部状態が存在することを探索する。
【０００３】
近年、ソースコードを対象としたモデル検査手法が、車載ソフトウェアの潜在的不具合を確実に検出する手段として有効であることが知られている。車載ソフトウェアは、時系列に沿って与えられる入力値列に対して内部状態値列を演算し、出力値列を出力するものであるため、上述のモデル検査手法の特性とよく合致するからである。以下このように時系列に変化する要素を有するソフトウェアまたはその挙動を模擬したソフトウェアのことを、動的プログラムと呼称する場合もある。ＩＳＯ２６２６２規格においてもソフトウェアの認証要件を満たすことを証明する手段としてモデル検査手法を要請するようになっている。
【０００４】
モデル検査手法は、ソフトウェアの内部状態を記述する内部状態値を定義し、ソフトウェアの入出力関係に対応する状態遷移規則を実装した状態遷移モデルを用いて、不具合に相当する状態遷移列を発見するという意味で、動的な入出力関係解析手法である。下記特許文献１～３は、モデル検査手法に係る従来技術を記載している。
【０００５】
モデル検査手法の他に検査手法としてよく用いられるシミュレーションは、モデル検査

手法における内部状態値の初期値に関する制約条件、およびソフトウェアに対する入力値列に関する制約条件を満たすような内部状態値および入力値列を、検査オペレータが逐一個別に設定してソフトウェアを実行し、その結果得られた出力値列を個別に評価することにより、出力値列が検査条件に違反するか否かを検査する手法である。

【０００６】

モデル検査手法とシミュレーションはともに、内部状態値と入力値列を網羅的に列挙することに起因する計算量が膨大となるため、計算リソースの制約がある前提の下では実用的な規模のソフトウェアに対して適用することが困難である。

【０００７】

この計算量増加の要因は、状態遷移経路を網羅するために設定する必要がある初期内部状態値の数が膨大にあり、個別に初期状態値を設定して状態遷移経路を構築していく必要があることに起因する。また、適当に選択した１つの初期状態値から入力値列に対応して得られる状態遷移経路の数は、探索したい時系列の入力値列の長さに対して少なくとも多項式的に、最悪で指数関数的に増加してしまう。このような探索方式に起因する計算量増加も、モデル検査手法の適用を妨げる要因となっている。

【先行技術文献】

【特許文献】

【０００８】

【特許文献１】特開２０１３－２００７８７号公報

【特許文献２】特開２０１２－１５５５１７号公報

【特許文献３】特表２００７－５２８０５９号公報

【発明の概要】

【発明が解決しようとする課題】

【０００９】

モデル検査手法をソフトウェアの検査に用いる際、旧来の方式においては、入力値列や初期内部状態値を一律にプログラム内変数とみなし、制約条件として変数毎に値域を設定するなどの制約を援用するに留まっていた。このため、検査に際して探索範囲として考慮する正常な入力値列として、値域の制約条件を満たす範囲内で取りうる乱数列まで含めてしまう。すなわち、ソフトウェアの本来の動作に鑑みると時系列に沿って変動し得る入力値列のみを探索範囲とすればよいにも関わらず、従来手法においては値域の制約条件さえ満たしている限りは時系列的に取りえない値であっても探索範囲に含まれてしまう。

【００１０】

例えば走行制御を目的として設計された動的プログラムが受け付ける入力値列は、エンジン温度、走行速度、車体加速度などであり、これら入力値列は、時系列における相関関係に対応する制約条件の下においてのみ、検査条件（すなわち機能要件や安全性要件）を満たしさえすればよい。

【００１１】

上記のような異常な入力値列が検査対象ソフトウェアに対して入力されると、検査対象ソフトウェアの出力値列も異常となるので、モデル検査装置はこれを異常とみなす。しかしこのような検査は本来不要である。したがってモデル検査手法（またはシミュレーション）を用いてソフトウェアを検査する場合、上記のように時系列上の制約条件を逸脱する異常な入力値列（例えば乱数列）は、計算リソースの消費を抑制する観点から、動的プログラムに対して入力する前に排除することが望ましい。

【００１２】

本発明は、上記のような課題に鑑みてなされたものであり、検査対象ソフトウェアに対して入力する異常な入力値列をあらかじめ排除することにより、ソフトウェアの挙動を検査するために必要な計算リソースを抑制することを目的とする。

【課題を解決するための手段】

【００１３】

本発明に係るプログラム検査装置は、検査対象である動的プログラムの入力値列、内部

状態値列、および出力値列と、ソフトウェアの要件を記述した検査条件とを、充足可能性問題の制約条件として記述し、所望の検査条件を満たす解を探索することにより、検査に適した入力値列を出力値列から逆算的に求める。

【発明の効果】

【００１４】

本発明に係るプログラム検査装置によれば、実用的な規模のソフトウェアを検査するための計算リソースを抑制することができる。また、異常な入力値列が入力されることによって異常な出力値列が出力された場合、これを異常とみなす誤判定を抑制することができる。

【図面の簡単な説明】

【００１５】

【図１Ａ】検査対象ソフトウェアの動作を記述した状態遷移モデルの概念図である。

【図１Ｂ】本発明に係るプログラム検査装置が用いる各パラメータを示す表である。

【図２】ソフトウェア１００を再帰的に繰り返し実行することにより各値列を求める様子を示す図である。

【図３】本発明に係るプログラム検査装置がソフトウェア１００に対して充足可能性判定を実施する処理を説明するフローチャートである。

【図４】実施形態２に係るソフトウェア検査装置４００の構成図である。

【図５】検査用プログラム構築部４１０がソフトウェア１００を用いて検査用プログラム４０５を構築する手順を説明するフローチャートである。

【図６】実施形態３における動的プログラム６０１の構成を示す図である。

【図７】ブレーキオーバーライド機能を実装した車両走行制御プログラム６０２のソースコード例である。

【図８】車両走行制御プログラム６０２の初期値制約条件の具体例である。

【図９Ａ】車両走行制御プログラム６０２の入力値列制約条件の具体例である。

【図９Ｂ】車両走行制御プログラム６０２の入力値列制約条件の具体例である。

【図１０】車両走行制御プログラム６０２の検査条件の具体例である。

【図１１】車両走行制御プログラム６０２の内部状態値、入力値列、および出力値列の演算例を示すタイムチャートである。

【図１２】プログラム検査装置４０６の構成図である。

【発明を実施するための形態】

【００１６】

＜本発明の基本的な考え方＞

本発明の理解を促進するため、以下ではまず本発明の基本的な考え方について説明し、その後に本発明の実施形態について説明する。

【００１７】

検査対象ソフトウェアに対して時系列的に不自然な異常入力値列を入力すると、ソフトウェアの内部状態値列や出力値列もその異常入力値列によって異常値となってしまう。検査を効率化する観点からは、時系列的に妥当な内部状態値列や出力値列のみが得られるような入力値列をソフトウェアに対して与えることが望ましい。しかし、いかなる入力値列を時系列的に与えれば妥当な内部状態値列や出力値列が得られるのかについては、必ずしも検査前の時点で明らかになっているわけではない。したがって従来のソフトウェア検査においては、入力値列を制限するとしてもせいぜい上下限値の範囲内に収めるようにする程度の制約しか課しておらず、結果として時系列的に異常な入力値列もテスト対象となるので、検査効率の観点からは望ましくなかった。

【００１８】

そこで本発明においては、時系列的に妥当な内部状態値列や出力値列が得られる入力値列を求めることを充足可能性問題（ＳＡＴｉｓｆｉａｂｉｌｉｔｙ　ｐｒｏｂｌｅｍ、以下ＳＡＴ問題と呼ぶ場合もある）として捉え直し、これを解くことにより検査に適した入力値列を求めることにした。

【００１９】
充足可能性問題とは、ある論理式を充足する変数群を求める問題である。これを本発明の課題に即して説明すると、時系列的に妥当な内部状態値列や出力値列を論理式として記述し、これを充足する入力値列を求めることに相当する。充足可能性問題を解くＳＡＴソルバは公知であるため、ＳＡＴソルバの制約条件として上記論理式を与えてその解をＳＡＴソルバに探索させることにより、所望の入力値列を得ることができる。その入力値列を用いてソフトウェア検査を実施することにより、限られた計算リソースの下で実用的な規模のソフトウェアを検査することができる。なおＳＡＴソルバは一般的にビット論理演算形式の論理式を充足する解を求めるように実装されているが、その他形式論理式によって表現されるＳＡＴ問題についても本発明を適用することができる。

【００２０】
ＳＡＴソルバに対して与える制約条件は、実施しようとしているソフトウェア検査の種類に応じて変えることができる。これにより、同じ仕組みを用いて複数種類のソフトウェア検査に対応することができる。制約条件としては例えば、（ａ）時系列の入力値列間の相関関係、（ｂ）時系列の出力値列間の相関関係、（ｃ）時系列の入力値列と出力値列との間の相関関係、（ｄ）これら相関関係と内部状態値の初期値との組み合わせ、などが考えられるが、これに限られるものではない。

【００２１】
＜実施の形態１＞
図１Ａは、検査対象ソフトウェアの動作を記述した状態遷移モデルの概念図である。図１Ａにおいて、ソフトウェア１００に対して時系列的に変化する入力値列ＤＡＴＡ＿ＩＮ［ｔ］を与え、ソフトウェア１００はこれに応じて時系列的に変化する内部状態値列ＳＴＡＴＥ［ｔ］を有するとともに、入力値列に対応する時系列の出力値列ＤＡＴＡ＿ＯＵＴ［ｔ］を出力する。ソフトウェア１００は時系列に沿って動的に内部状態が変化するので、動的プログラムと呼ぶ場合もある。

【００２２】
ソフトウェア１００を、入力値と内部状態から出力値を決定する関数とみなし、入力値列ＤＡＴＡ＿ＩＮ［ｔ］、内部状態値列ＳＴＡＴＥ［ｔ］、および出力値列ＤＡＴＡ＿ＯＵＴ［ｔ］と併せてその関数を定義することにより、ソフトウェア１００の動的挙動を表す状態遷移モデルを構築することができる。本発明に係るプログラム検査装置は、この状態遷移モデルと以下に説明する各パラメータを用いて、上述の充足可能性問題を解く。

【００２３】
図１Ｂは、本発明に係るプログラム検査装置が用いる各パラメータを示す表である。入力値列、出力値列、内部状態については図１Ａで説明した通りである。状態値Ｓ［ｔ］はこれら３変数を１つにまとめたものである。関数ＰＲＯＧＲＡＭ＿ＳＴＡＴＥと関数ＰＲＯＧＲＡＭ＿ＯＵＴはソフトウェア１００の状態遷移モデルを記述した関数である。初期値制約条件ＣＯＮＤＩＴＩＯＮ＿ＩＮＩＴは内部状態値の初期値として取り得る値を規制する条件である。入力値列制約条件ＣＯＮＤＩＴＩＯＮ＿ＩＮＰＵＴは、入力値列ＤＡＴＡ＿ＩＮ［ｔ］の値域など従来と同様の制約を記述したものである。検査条件ＣＯＮＤＩＴＩＯＮ＿ＣＨＥＣＫは、ソフトウェア１００に対して実施する検査を制約条件として記述したものである。

【００２４】
初期値制約条件ＣＯＮＤＩＴＩＯＮ＿ＩＮＩＴとしては、ソフトウェア１００が取りうる正常な初期内部状態値を列挙すればよい。検査条件ＣＯＮＤＩＴＩＯＮ＿ＣＨＥＣＫとしては、ソフトウェア１００の動作形態に即して定義した機能要件、動作制約、ハザード条件などを設定すればよい。

【００２５】
図２は、ソフトウェア１００を再帰的に繰り返し実行することにより各値列を求める様子を示す図である。初期内部状態値２０１とｔ＝０における入力値２０２をソフトウェア１００に対して入力して実行することにより、ｔ＝１における出力値２０３と内部状態値

２０４が求められる。ｔ＝１以降の入力値列ＤＡＴＡ＿ＩＮをソフトウェア１００に対して順次投入してソフトウェア１００を再帰的に実行することにより、ｔ＝１以降の各値列を求めることができる。

【００２６】

ソフトウェア１００に対する検査は、入力値列２０５、出力値列２０６、および内部状態値列ＳＴＡＴＥ［ｔ］のいずれかまたはこれらを組み合わせた値列が、所定の検査条件を満たしているか否かを判定することにより実施される。すなわち、ソフトウェア１００が検査条件に違反するか否かを検査することは、指定した検査条件および時系列における各値列の実際の値を制約条件として、検査条件を満たさない値列が存在するか否かを探索するＳＡＴ問題に帰結される。指定した検査条件および時系列における各値列の実際の値は、論理式として表現することにより、ＳＡＴ問題における制約条件として流用することができる。

【００２７】

図３は、本発明に係るプログラム検査装置がソフトウェア１００に対して充足可能性判定を実施する処理を説明するフローチャートである。以下図３の各ステップについて説明する。

【００２８】

（図３：ステップＳ３００）

プログラム検査装置は、図１Ｂで説明した初期値制約条件、入力値列制約条件、検査条件、入力値列ＤＡＴＡ＿ＩＮ［］、内部状態値列ＳＴＡＴＥ［］の初期値（ＳＴＡＴＥ［０］）を、プログラム検査装置に対する入力として取得する。これらの値は例えば適当なデータファイルなどに記載して入力すればよい。

【００２９】

（図３：ステップＳ３０１）

プログラム検査装置は、図１Ｂで説明した初期値制約条件、入力値列制約条件、および検査条件とともに、図２で説明した手法によって得られる入力値列ＤＡＴＡ＿ＩＮ［］、内部状態値列ＳＴＡＴＥ［］、および出力値列ＤＡＴＡ＿ＯＵＴ［］を、論理式形式で記述する。具体的には、時系列における各値列および各制約条件を論理積によって結合したものが、ＳＡＴソルバに対して与える制約条件となる。この制約条件を記述した論理式群のことを、以下では検査用プログラムと呼ぶ場合もある。なお本実施形態１においては、検査条件を満たさない値列が存在するか否かを探索することによって検査条件を充足するか否かを判定することとしたので、制約条件としてＣＯＮＤＩＴＩＯＮ＿ＣＨＥＣＫ＝＝ＦＡＬＳＥを指定した。検査用プログラムは、プログラム検査装置自身が構築してもよいし、外部装置がこれを構築してプログラム検査装置に対して与えてもよい。

【００３０】

（図３：ステップＳ３０２～Ｓ３０２１）

プログラム検査装置（のＳＡＴソルバ、以下同様）は、初期値制約条件を充足する解（すなわち各値列、以下同様）が存在するか否かを判定する（Ｓ３０２）。充足解がなければ、初期値制約条件そのものが適切に設定されていない旨を判定結果として報告し、本フローチャートを終了する（Ｓ３０２１）。

【００３１】

（図３：ステップＳ３０３～Ｓ３０３１）

プログラム検査装置は、入力値列制約条件を充足する解が存在するか否かを判定する（Ｓ３０３）。充足解がなければ、入力値列制約条件そのものが適切に設定されていない旨を判定結果として報告し、本フローチャートを終了する（Ｓ３０３１）。

【００３２】

（図３：ステップＳ３０４～Ｓ３０４１）

プログラム検査装置は、検査条件を充足する解が存在するか否かを判定する（Ｓ３０４）。充足解がなければ、検査条件そのものが適切に設定されていない旨を判定結果として報告し、本フローチャートを終了する（Ｓ３０４１）。

【００３３】
（図３：ステップＳ３０２～Ｓ３０４：補足）
これら３ステップは、制約条件を設定するオペレータが例えば人的操作ミスなどによって誤った制約条件を設定し、ＳＡＴソルバが妥当な解を導きだせなくなるようなエラーをあらかじめ抑制するための、予備的な処理である。
【００３４】
（図３：ステップＳ３０５～Ｓ３０５１）
プログラム検査装置は、ステップＳ３０１で構築した検査用プログラムが記述している論理式を充足する解が存在するか否かを判定する（Ｓ３０５）。ステップＳ３０５において解が存在しない（すなわちＣＯＮＤＩＴＩＯＮ＿ＣＨＥＣＫ＝＝ＦＡＬＳＥとなる値列が存在しない）場合は、検査条件に違反がない旨の判定結果を報告し、本フローチャートを終了する（Ｓ３０５１）。
【００３５】
（図３：ステップＳ３０６）
ステップＳ３０５において解が存在する場合は、その具体的な割り当て値（すなわちＤＡＴＡ＿ＩＮ［］、ＳＴＡＴＥ［］、およびＤＡＴＡ＿ＯＵＴ［］の各時系列値）を、検査条件に違反する具体的な事例として報告し、本フローチャートを終了する。
【００３６】
＜実施の形態１：制約条件の例＞
以下では、制約条件を適当に設定することにより、本発明に係るプログラム検査装置を様々な種別のソフトウェア検査に対して適用できることを説明する。具体的な適用例として、（ａ）ソフトウェア設計不具合の解析、（ｂ）ソフトウェア故障モード影響解析、（ｃ）ソフトウェア故障ツリー解析が挙げられる。
【００３７】
ソフトウェア設計不具合の解析において本発明を適用する場合、以下の制約条件を設定すればよい：
（ａ）ソフトウェア１００が取りうる正常な初期内部状態値の集合を初期値制約条件として設定する；
（ｂ）ソフトウェア１００の動作環境に依存する前提条件として機能要件内に明示されている、ソフトウェア１００に対する妥当な入力値列ＤＡＴＡ＿ＩＮ［］の定義を、入力値列制約条件として設定する；
（ｃ）ソフトウェアの動作形態に即して定義した機能要件を、正常な入力値列ＤＡＴＡ＿ＩＮ［］に対して得られる出力値列ＤＡＴＡ＿ＯＵＴ［］を検査条件として設定し、これに違反するような不都合な入力値列ＤＡＴＡ＿ＩＮ［］が存在するか否かを判定する。
【００３８】
ソフトウェア故障モード影響解析において本発明を適用する場合、以下の制約条件を設定すればよい：
（ａ）ソフトウェア１００が取りうる正常な初期内部状態値の集合を初期値制約条件として設定する；
（ｂ）ソフトウェア１００の動作時前提条件として機能要件内に明示されている、ソフトウェア１００に対する妥当な入力値列ＤＡＴＡ＿ＩＮ［］、またはソフトウェア１００が入力値を取得するデータソースにおいて起こりうる故障要因を明示的に指定して、これが招くいずれかのタイミングにおける異常な入力値列ＤＡＴＡ＿ＩＮ［］を、入力値列制約条件として設定する；
（ｃ）ソフトウェア１００の安全要件に即して定義したハザード条件を検査条件として設定し、検査条件に違反するような不都合な入力値列ＤＡＴＡ＿ＩＮ［］が存在するか否かを判定する。
【００３９】
ソフトウェア故障ツリー解析において本発明を適用する場合、以下の制約条件を設定すればよい：

（ａ）ソフトウェア１００が取りうる正常な初期内部状態値の集合を初期値制約条件として設定する；
（ｂ）ソフトウェア１００の動作時前提条件として機能要件内に明示されている、ソフトウェア１００に対する妥当な入力値列ＤＡＴＡ＿ＩＮ［］、またはソフトウェア１００が入力値を取得するデータソースにおいて起こりうる故障要因が招くいずれかのタイミングにおける異常な入力値列ＤＡＴＡ＿ＩＮ［］を、入力値列制約条件として設定する；
（ｃ）ソフトウェア１００の安全要件に即して定義したハザード条件を検査条件として設定し、検査条件に違反するような不都合な故障要因および入力値列ＤＡＴＡ＿ＩＮ［］の組み合わせが存在するか否かを判定する。

【００４０】

＜実施の形態１：まとめ＞

以上のように、本実施形態１に係るプログラム検査装置は、時系列に変化する値列、およびソフトウェア１００に対する検査条件を、ＳＡＴソルバに対する制約条件を記述した論理式として使用し、制約条件を充足する解を探索する。これにより、時系列で変化する入力値列ＤＡＴＡ＿ＩＮ［］に対して応答するソフトウェア１００が機能要件に違反するような不具合を有するか否かを判定することができる。

【００４１】

また、本実施形態１に係るプログラム検査装置によれば、時系列的に正常な入力値列ＤＡＴＡ＿ＩＮ［］、およびこれから出力される出力値列ＤＡＴＡ＿ＯＵＴ［］を導き出すことができる。これにより、検査範囲とする入力値列を時系列的に正常な値域のみに限定することができる。したがって、計算リソースが限られた状況下においても、実用的な規模のソフトウェアに対して課した検査項目を正しく判定することができる。具体的には、ソフトウェア１００の規模に対して解の探索範囲が指数関数的に増加するのに対し、本実施形態によれば計算量はソフトウェア１００の規模に対しておよそ多項式関数的に増加するに留めることができる。

【００４２】

また、本実施形態１に係るプログラム検査装置によれば、異常な入力値列ＤＡＴＡ＿ＩＮ［］に対応する異常な出力値列ＤＡＴＡ＿ＯＵＴ［］を誤って異常とみなすという誤検出を阻止できる。

【００４３】

＜実施の形態２＞

図４は、本発明の実施形態２に係るソフトウェア検査装置４００の構成図である。ソフトウェア検査装置４００は、実施形態１で説明したプログラム検査装置４０６を用いてソフトウェア１００を検査する装置であり、初期値制約設定部４０７、入力値列制約設定部４０８、検査条件設定部４０９、検査用プログラム構築部４１０、プログラム検査装置４０６を備える。

【００４４】

ソフトウェア検査装置４００は、初期値制約条件４０１、入力値列制約条件４０２、検査対象であるソフトウェア１００、検査条件４０３を入力として受け取り、図３のステップＳ３０２～Ｓ３０６で説明した各判定結果を出力する。具体的には、プログラム検査装置４０６が、初期値制約条件が不正である旨の判定結果４１６（Ｓ３０２１）、入力値制約条件が不正である旨の判定結果４１７（Ｓ３０３１）、検査条件が不正である旨の判定結果４１８（Ｓ３０４１）、検査条件違反がない旨の判定結果４１９（Ｓ３０５１）、検査条件に違反する初期値と時系列値４２０（Ｓ３０６）を出力する。

【００４５】

初期値制約条件４０１は、初期値制約記述単位の論理和、論理積、または論理否定を組み合わせることにより構成される。初期値制約記述単位は、（内部状態の初期値に対する前提条件）と（拘束条件）の対によって構成される。この前提条件および拘束条件は、内部状態の初期値に相当する変数、入力値列に相当する変数群、および固定値を用いて変数間の制約条件を記述した式を含む。

【００４６】
初期値制約設定部４０７は、適当なユーザインターフェースなどを介して初期値制約条件４０１を受け取り、これを初期値制約条件の形式で記述した初期値制約条件記述４１１に変換する。このとき、演算負荷の観点から、初期値制約条件４０１が指定する条件と等価かつ計算効率が良い論理式を構築してもよい。例えば初期値制約条件４０１が『（前提条件）が成立するならば（内部状態値の初期値の拘束条件）』という形式で指定されている場合、これと等価な論理式である『ＮＯＴ（前提条件）｜｜（内部状態値の初期値の拘束条件）』を初期値制約条件記述４１１としてもよい。入力値列制約設定部４０８と検査条件設定部４０９についても同様である。

【００４７】
入力値列制約条件４０２は、入力値列制約記述単位の論理和、論理積、または論理否定を組み合わせることにより構成される。入力値列制約記述単位は、（入力値列に対する前提条件）と（拘束条件）の対によって構成される。入力値列制約設定部４０８は、適当なユーザインターフェースなどを介して入力値列制約条件４０２を受け取り、これを入力値列制約条件の形式で記述した入力値列制約条件記述４１２に変換する。

【００４８】
検査条件４０３は、検査条件記述単位の論理和、論理積、および論理否定を組み合わせることにより構成される。検査条件記述単位は、（入力値列または内部状態初期値に対する前提条件）と（出力値列に対する拘束条件）の対によって構成される。この（前提条件）および（拘束条件）は、内部状態値の初期値に相当する変数、入力値列および出力値列に相当する変数群、および固定値を用いて変数間の制約条件を記述した式を含む。検査条件設定部４０９は、適当なユーザインターフェースなどを介して検査条件４０３を受け取り、これを検査条件の形式で記述した検査条件記述４１５に変換する。検査条件４０３はさらに、演算負荷を調整するための実行回数などを指定する再帰実行条件４１４を指定することもできる。

【００４９】
図５は、検査用プログラム構築部４１０がソフトウェア１００を用いて検査用プログラム４０５を構築する手順を説明するフローチャートである。本フローチャートは図３のステップＳ３０１～Ｓ３０４に相当する。検査用プログラム構築部４１０は初期化処理として論理式Ｐ＝ＴＲＵＥをあらかじめセットしておく（Ｓ５００）。以下図５の各ステップについて説明する。

【００５０】
（図５：ステップＳ５０１、Ｓ５０８）
検査用プログラム構築部４１０は、初期値制約条件記述４１１、入力値列制約条件記述４１２、および検査条件記述４１５がそれぞれ矛盾なく記述されており、これらを充足する解が存在するか否かを判定する。充足解がなければステップＳ５０８に進み、いずれかの制約条件記述が不正である旨の判定結果（Ｓ３０２～Ｓ３０４に相当）を出力して本フローチャートを終了する。充足解があればステップＳ５０２に進む。

【００５１】
（図５：ステップＳ５０２～Ｓ５０３）
検査用プログラム構築部４１０は、初期値制約設定部４０７により論理式形式に変換された初期値制約条件記述４１１と論理式Ｐの初期値ＴＲＵＥとの論理積をとる（Ｓ５０２）。検査用プログラム構築部４１０は、入力値列制約設定部４０８により論理式形式に変換された入力値列制約条件記述４１２と、ステップＳ５０２で更新した論理式Ｐとの論理積をとる（Ｓ５０３）。

【００５２】
（図５：ステップＳ５０４～Ｓ５０５）
検査用プログラム構築部４１０は、再帰実行条件４１４を取得する（Ｓ５０４）。再帰実行条件４１４は、例えば入力値列の最終時点Ｔを指定する。検査用プログラム構築部４１０は、以下のステップＳ５０６を再帰実行した回数のカウンタｔを初期化する（Ｓ５０

５）。
【００５３】
（図５：ステップＳ５０６）
検査用プログラム構築部４１０は、ソフトウェア１００に対して入力値列を順次入力して内部状態値と出力値を求め、これら値列（動的プログラム記述４１３に相当）と論理式Ｐとの論理積をとる。検査用プログラム構築部４１０は、この処理を最終時点Ｔに到るまで反復実行する。
【００５４】
（図５：ステップＳ５０７）
検査用プログラム構築部４１０は、検査条件設定部４０９により論理式形式に変換された検査条件記述４１５と、ステップＳ５０６で更新した論理式Ｐとの論理積をとる。この時点で得られた論理式Ｐを検査用プログラム４０５とする。プログラム検査装置４０６はこの検査用プログラム４０５を用いて実施形態１で説明した処理を実行する。
【００５５】
＜実施の形態３＞
本発明の実施形態３では、具体的な制約条件の記述例として、走行安定化機能ＥＳＣ（Ｅｌｅｃｔｒｏｎｉｃ　Ｓｔａｂｉｌｉｔｙ　Ｃｏｎｔｒｏｌ）とブレーキオーバーライド機能を同時に搭載した車両システムにおいて用いる車両走行制御プログラム６０２を検査対象とする例について説明する。各装置の構成は実施形態１～２と同様であるため、以下では制約条件の具体例について主に説明する。
【００５６】
本実施形態３では、動摩擦係数が一時的に低下する道路状態にある走行路を走行中にアクセルおよびブレーキを同時に強く踏み込んでしまったとき、車両走行制御プログラム６０２がＥＳＣによるブレーキ制御よりも強く踏み込まれたブレーキを優先するような実装となっていることを検査する。
【００５７】
ＥＳＣは通常、理想的な走行路との接触状況から逸脱して車両状態が不安定化したとき一時的に起動される走行補助機能である。ＥＳＣによるブレーキ制御とブレーキそのものの制御とが適切に実装されていない場合、車両システムの操縦者が急ブレーキを意図してブレーキを強く踏み込むと、ブレーキ自体の制御機能とＥＳＣによる制御機能との間で出力競合を招いてしまい、適切な加減速度を一意に決められなくなる。このような不具合を避けるため、車両走行制御プログラム６０２はこれら機能に関する適切な優先処理を実装している必要がある。
【００５８】
ブレーキオーバーライド機能は、このような出力競合時に、ブレーキの優先度を強制する機能である。例えば、車両走行制御プログラム６０２が意図しない加速指令を設定してしまう状況においても、操縦者によるブレーキ踏み込みを通じた減速指令を優先する機能が挙げられる。
【００５９】
図６は、本実施形態３における動的プログラム６０１の構成を示す図である。図６に示す動的プログラム６０１は、検査対象とする車両走行制御プログラム６０２、およびシミュレーションモデルを援用する等して適当に設計した車両の動特性モデル６０３によって構成される。
【００６０】
車両走行制御プログラム６０２に対する入力値列は、アクセル踏み込み値、ブレーキ踏み込み値、ステアリング角度値を含む車両操作値入力値列６０４、車両走行制御プログラム６０２が車両システムから受け取る計測状態値入力値列６０５、車両システムの動特性が依存する動作環境状態値入力値列６０６である。動作環境状態値入力値列６０６の１例として、走行路の状態に依存する動摩擦係数が挙げられる。
【００６１】

車両走行制御プログラム６０２は出力値列として、加速度力量値、減速度力量値、旋回トルク値、ストップランプ値を含む車両制御値出力値列６０７を出力する。車両走行制御プログラム６０２は内部状態値として、動摩擦係数推定値、車両速度、車両姿勢角速度を用いる。

【００６２】

車両の動特性モデル６０３は入力値列として、動作環境状態値入力値列６０６、車両制御値出力値列６０７を用いる。動特性モデル６０３は出力値列として、計測状態値入力値列６０５を出力する。動特性モデル６０３は内部状態値として、車両加速度、車両姿勢角速度を用いる。

【００６３】

図７は、ブレーキオーバーライド機能を実装した車両走行制御プログラム６０２のソースコード例である。図７（ａ）は同機能を適切に実装していない例を示し、図７（ｂ）は同機能を適切に修正した例を示す。

【００６４】

図７に示す例においては、姿勢不安定状態を判定する制約条件の例として、車両操作値入力値列６０４や計測状態値入力値列６０５を用いることを想定している。また急ブレーキ踏み込みを判定する制約条件の例として、車両操作値入力値列６０４の構成要素であるブレーキ踏み込み値を用いることを想定している。図７に示すソースコードは、これらの判定条件に相当する制約条件を用いて、ＢＲＡＫＥ＿ＯＮ、ＢＲＡＫＥ＿ＯＦＦ、ＥＳＣ＿ＯＮ、ＥＳＣ＿ＯＦＦ、ＦＯＬＬＯＷ＿ＣＯＭなどの適当な制御処理を選択し、車両制御値出力値列６０７である加速度力量値、減速度力量値、旋回トルク値、およびストップランプ値を出力する。

【００６５】

図７に示すソースコードを検査する場合、検査条件は、「急ブレーキ踏み込み時は必ずそのブレーキ踏み込み指令が優先される」となる。ソフトウェア検査装置４００は、この制約条件に違反する不都合な挙動が存在するか否かを探索する。この制約条件に違反するプログラム（図７（ａ））は、車両が不安定化した時のＥＳＣ起動を急ブレーキによるブレーキ制御よりも優先するように設計されているので、一時的に姿勢不安定化した時に踏み込まれた急ブレーキが有効に機能しない。ソフトウェア検査装置４００は、このような不都合な挙動を検出する。

【００６６】

図８は、車両走行制御プログラム６０２の初期値制約条件の具体例である。本例においては初期値の値域に関する制約条件のみを記述している。

【００６７】

図９Ａおよび図９Ｂは、車両走行制御プログラム６０２の入力値列制約条件の具体例である。前提条件の表現形式としては、前提条件の適用時点の制約や特定の条件付き制約条件を用いることができる。

【００６８】

動摩擦係数の制約条件として例えば、想定範囲とする動摩擦係数に関する拘束条件、動摩擦係数値の一次連続性に関する拘束条件、特定の期間中に動摩擦係数値が低下するという限定、などを用いることができる。

【００６９】

アクセル踏み込み値に関する制約条件として例えば、検査範囲とするアクセル踏み込み値の上下限制約、アクセル踏み込み値の一次連続性制約、アクセル踏み込み値の時系列値に関する制約、などを用いることができる。図９Ａおよび図９Ｂに示す例においては、踏み込み値初期値よりも最終値が小さくなり、上下限制約と一次連続性制約の下で、最終値が指定値となることを想定し、その旨の制約条件を記述している。すなわち検査範囲は、これら制約条件を満たす範囲内に限定される。

【００７０】

図１０は、車両走行制御プログラム６０２の検査条件の具体例である。１つ目の検査条

件は、再帰実行条件４１４として取得した検査範囲の任意の時点において、（ａ）ブレーキの踏み込み条件を満たすという前提条件と、（ｂ）ストップランプが点灯することに相当する拘束条件とによって構成されている。２つ目の検査条件は、再帰実行条件４１４として取得した検査範囲の任意の時点において、（ａ）ブレーキ踏み込み条件を満たすという前提条件と、（ｂ）減速度力量値に関する指定の拘束条件とによって構成されている。

【００７１】

図１１は、車両走行制御プログラム６０２の内部状態値、入力値列、および出力値列の演算例を示すタイムチャートである。このタイムチャートは、再帰実行条件４１４として取得した検査範囲の離散時間毎に、入力値列、内部状態値、出力値列を表示したものである。ここではソフトウェア検査装置４００が検査条件に違反する旨の判定結果を出力するタイムチャート例を示している。

【００７２】

走行途中で動摩擦係数（ＶＥＣ＿ｍｕ）が急激に低下し、操縦者が急ブレーキ（ＤＡＴＡ＿ＩＮ＿ＢＲＡＫＥ＿ＰＥＤＡＬ）を踏みこんだ時点から、姿勢角速度計測値（ＤＡＴＡ＿ＩＮ＿ＲＡＴＥ＿ＹＡＷ）が大きく負の方向に振れてしまい、図７（ａ）記載のプログラムにおける姿勢不安定を判定する条件を満たしたと仮定する。これによりＥＳＣ＿ＯＮ状態になり、急ブレーキによる原則よりもＥＳＣによる姿勢安定化を目的としたブレーキ制御が優先されることとなる。すなわち図７（ａ）に示すプログラムによれば、ストップランプ（ＤＡＴＡ＿ＯＵＴ＿ＳＴＯＰ＿ＬＡＭＰ）を点灯させる分岐条件に到達することができず、検査条件に違反する。ＥＳＣモードがＯＦＦになった時点から急ブレーキを有効にする制御を開始し、その後にストップランプはＯＮ状態に正しく移行している。

【００７３】

車両走行制御プログラム６０２の設計者は、図１１に示す検査条件違反に基づき、図７（ａ）に示すプログラムにおける違反原因を読み取り、図７（ｂ）に示す通り修正したプログラムを対象として再度検査を実行する。同様の検査条件違反が無くなった時点で、少なくとも、図８と図９で限定した範囲において、指定した検査条件に違反しないことを証明できたことになる。

【００７４】

＜実施の形態４＞

図１２は、プログラム検査装置４０６の構成図である。プログラム検査装置４０６は、ＳＡＴ問題を解くアルゴリズムを実装したソフトウェアであるＳＡＴソルバ４０６２、これを実行するＣＰＵ（Ｃｅｎｔｒａｌ　Ｐｒｏｃｅｓｓｉｎｇ　Ｕｎｉｔ）４０６１、検査用プログラム４０５や各制約条件記述を格納する記憶装置４０６３、データ入出力のためのインターフェース４０６４を備える。ＳＡＴソルバ４０６２は必ずしもソフトウェアとして実装する必要はなく、同様のアルゴリズムを実装した回路デバイスなどのハードウェアによって構成することもできるし、ＳＡＴ問題に特化した高速処理コンピュータなどの別デバイスとして構成することもできる。

【００７５】

本発明は上記した実施形態に限定されるものではなく、様々な変形例が含まれる。上記実施形態は本発明を分かりやすく説明するために詳細に説明したものであり、必ずしも説明した全ての構成を備えるものに限定されるものではない。また、ある実施形態の構成の一部を他の実施形態の構成に置き換えることもできる。また、ある実施形態の構成に他の実施形態の構成を加えることもできる。また、各実施形態の構成の一部について、他の構成を追加・削除・置換することもできる。

【００７６】

上記制御部、処理部、機能等は、それらの一部を、例えば集積回路で設計する等によりハードウェアで実現してもよい。また、上記制御部、処理部、機能等は、プロセッサがそれぞれの機能を実現するプログラムを解釈し、実行することによりソフトウェアで実現してもよい。各機能を実現するプログラム、テーブル、ファイル等の情報は、メモリ、ハードディスク、ＳＳＤ（Ｓｏｌｉｄ　Ｓｔａｔｅ　Ｄｒｉｖｅ）等の記録装置、ＩＣカード

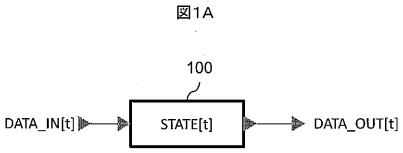
、ＳＤカード、ＤＶＤ等の記録媒体に格納することができる。

【符号の説明】

【００７７】

１００：ソフトウェア、４００：ソフトウェア検査装置、４０１：初期値制約条件、４０２：入力値列制約条件、４０３：検査条件、４０５：検査用プログラム、４０６：プログラム検査装置、４０７：初期値制約設定部、４０８：入力値列制約設定部、４０９：検査条件設定部、４１０：検査用プログラム構築部、４１１：初期値制約条件記述、４１２：入力値列制約条件記述、４１３：動的プログラム記述、４１４：再帰実行条件、４１５：検査条件記述。

【図１Ａ】



【図１Ｂ】

図1B

名称	シンボル	説明
入力値列	{DATA_IN[t=0], DATA_IN[t=1], ..., DATA_IN[t=T]};	動的プログラムに与える入力値DATA_IN[t]の時系列値
出力値列	{DATA_OUT[t=0], DATA_OUT[t=1],...,DATA_OUT[t=T]};	動的プログラムから得られる出力値DATA_OUT[t]の時系列値
内部状態	STATE[t]	動的プログラム内部変数値
状態値	S{t}={DATA_IN[t], STATE[t],出力:DATA_OUT[t]}	動的プログラム全体の動的状態を一意に定めるパラメータ
動的プログラム	STATE[t]を与える規則PROGRAM_STATE STATE[t+1]=PROGRAM_STATE(DATA_IN[t],STATE[t]);	内部状態値の処理用関数
	DATA[t]を与える規則PROGRAM_OUT DATA_OUT[t]=PROGRAM_OUT(DATA_IN[t],STATE[t]);	出力値の算出用関数
初期値	STATE[t=0]	前記動的プログラム内部状態値の初期値
初期値制約条件	CONDITION_INIT(STATE[t=0])	動的プログラムの初期値STATE[t=0]の取りうる値の範囲の内、検査対象として評価する値域を定義する論理式
入力値列制約条件	CONDITION_INPUT({DATA_IN[t]| t=0...T-1})	前記時系列の入力値列が取りうる値の内、検査範囲とする値域を指定する基準を与える論理式
検査条件	CONDITION_CHECK({DATA_IN[t], DATA_OUT[t]|t=0...T})	システムの状態値およびその状態遷移列に対して、検査内容の評価基準を与える論理式

【図２】

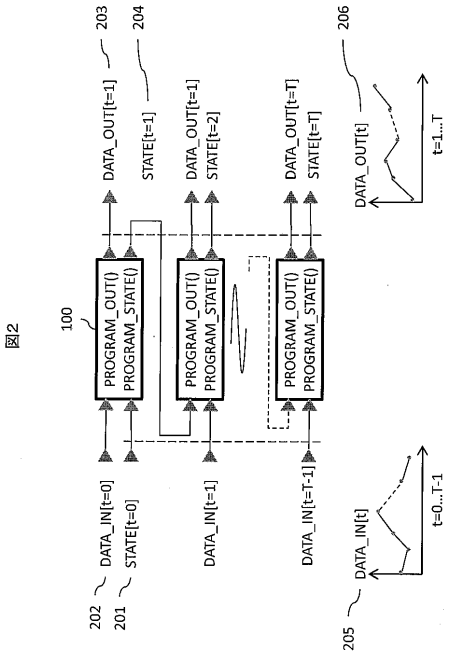


図2
100
PROGRAM_OUT()
PROGRAM_STATE()
DATA_IN[t=0]
STATE[t=0]
DATA_OUT[t=1]
STATE[t=1]
DATA_IN[t=1]
DATA_OUT[t=1]
STATE[t=2]
DATA_IN[t=T-1]
DATA_OUT[t=T]
STATE[t=T]
DATA_IN[t]
t=0...T-1
DATA_OUT[t]
t=1...T
201
202
203
204
205
206

【図３】

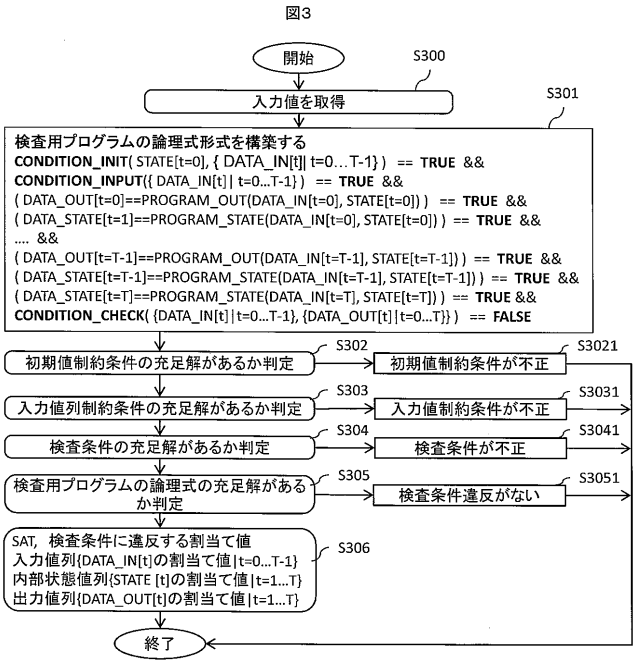


図3
開始
S300
入力値を取得
S301
検査用プログラムの論理式形式を構築する
CONDITION_INIT(STATE[t=0], { DATA_IN[t]| t=0…T-1}) == TRUE &&
CONDITION_INPUT({ DATA_IN[t]| t=0...T-1}) == TRUE &&
(DATA_OUT[t=0]==PROGRAM_OUT(DATA_IN[t=0], STATE[t=0])) == TRUE &&
(DATA_STATE[t=1]==PROGRAM_STATE(DATA_IN[t=0], STATE[t=0])) == TRUE &&
.... &&
(DATA_OUT[t=T-1]==PROGRAM_OUT(DATA_IN[t=T-1], STATE[t=T-1])) == TRUE &&
(DATA_STATE[t=T-1]==PROGRAM_STATE(DATA_IN[t=T-1], STATE[t=T-1])) == TRUE &&
(DATA_STATE[t=T]==PROGRAM_STATE(DATA_IN[t=T], STATE[t=T])) == TRUE &&
CONDITION_CHECK({DATA_IN[t]|t=0...T-1}, {DATA_OUT[t]|t=0...T}) == FALSE
初期値制約条件の充足解があるか判定
S302
初期値制約条件が不正
S3021
入力値列制約条件の充足解があるか判定
S303
入力値制約条件が不正
S3031
検査条件の充足解があるか判定
S304
検査条件が不正
S3041
検査用プログラムの論理式の充足解があるか判定
S305
検査条件違反がない
S3051
SAT, 検査条件に違反する割当て値
入力値列{DATA_IN[t]の割当て値|t=0...T-1}
内部状態値列{STATE [t]の割当て値|t=1...T}
出力値列{DATA_OUT[t]の割当て値|t=1...T}
S306
終了

【図４】

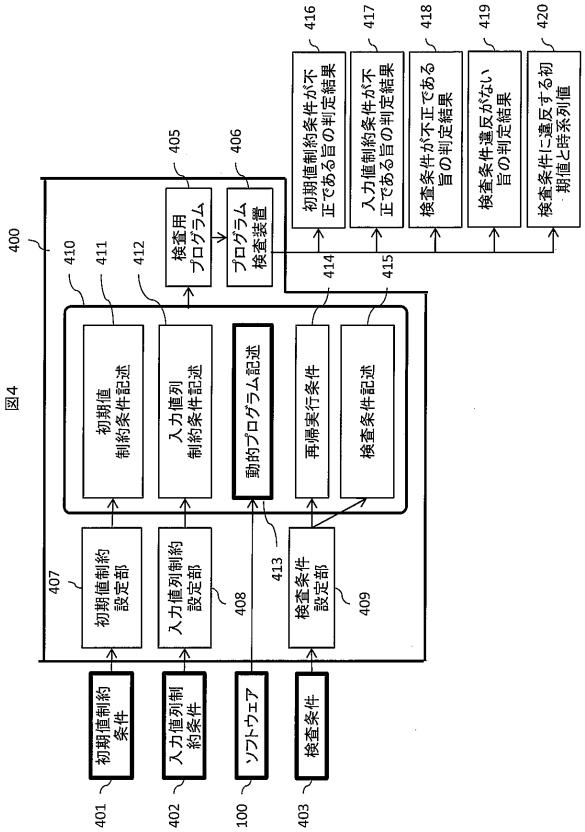


図4
400
401
402
403
100
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
初期値制約条件
入力値列制約条件
ソフトウェア
検査条件
初期値制約設定部
入力値列制約設定部
検査条件設定部
初期値制約条件記述
入力値列制約条件記述
動的プログラム記述
再帰実行条件
検査条件記述
検査用プログラム
プログラム検査装置
初期値制約条件が不正である旨の判定結果
入力値制約条件が不正である旨の判定結果
検査条件が不正である旨の判定結果
検査条件違反がない旨の判定結果
検査条件に違反する初期値と時系列値

【図５】

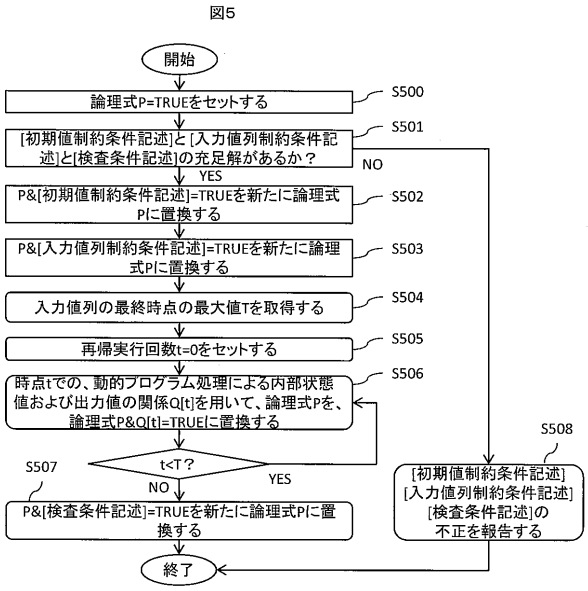
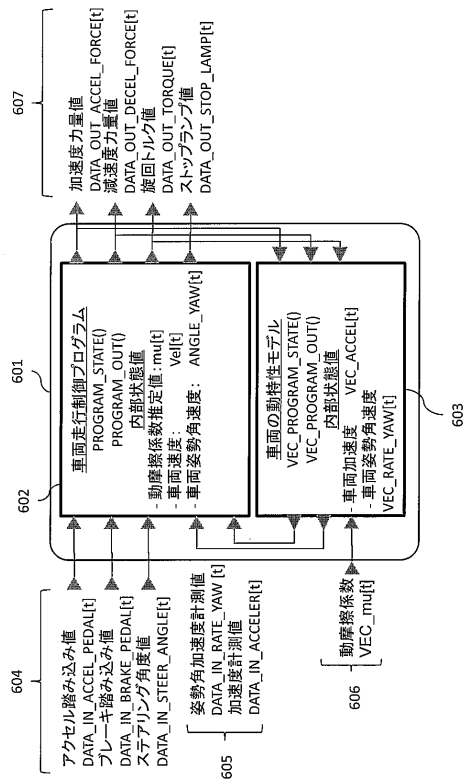


図5
開始
論理式P=TRUEをセットする
S500
[初期値制約条件記述]と[入力値列制約条件記述]と[検査条件記述]の充足解があるか？
S501
NO
YES
P&[初期値制約条件記述]=TRUEを新たに論理式Pに置換する
S502
P&[入力値列制約条件記述]=TRUEを新たに論理式Pに置換する
S503
入力値列の最終時点の最大値Tを取得する
S504
再帰実行回数t=0をセットする
S505
時点tでの、動的プログラム処理による内部状態値および出力値の関係Q[t]を用いて、論理式Pを、論理式P&Q[t]=TRUEに置換する
S506
t<T?
YES
NO
S507
P&[検査条件記述]=TRUEを新たに論理式Pに置換する
[初期値制約条件記述]
[入力値列制約条件記述]
[検査条件記述]の
不正を報告する
S508
終了

【図６】

図6



【図７】

図7

(a)

```
while(1){
  内部状態演算;
  if([姿勢不安定]=TRUE){
    BRAKE_OFF();
    ESC_ON();
  }else if([急ブレーキ]=TRUE){
    ESC_OFF();
    BRAKE_ON();
  }else {
    ESC_OFF();
    BRAKE_OFF();
    FOLLOW_COM();
  };
};
```

(b)

```
while(1){
  内部状態演算;
  if([急ブレーキ]=TRUE){
    ESC_OFF();
    BRAKE_ON();
  } else if([姿勢不安定]=TRUE){
    BRAKE_OFF();
    ESC_ON();
  }else {
    ESC_OFF();
    BRAKE_OFF();
    FOLLOW_COM();
  };
};
```

【図８】

図8

初期値制約条件		
前提条件	拘束条件	変換後の論理式
	ACCEL_LOW<=VEC_ACCEL[t=0]<=ACCEL_HIGH	左と同じ
	RATE_YAW_LOW<=VEC_RATE_YAW[t=0]<=RATE_YAW_HIGH	左と同じ
	mu_norm_low<=mu[t=0]<=mu_norm_high	左と同じ
	VEL_LOW<=Vel[t=0]<=VEL_HIGH	左と同じ
	YAW_LOW<=ANGLE_YAW[t=0]<=YAW_HIGH	左と同じ

【図９Ａ】

図9A

	入力値列制約条件	
1	前提条件	t=0..T-1のすべての時点
	拘束条件	//想定範囲とする動摩擦係数の制約 0<=VEC_mu[t]<=mu_norm_high
	論理式	(0<=VEC_mu[t=0]<=mu_norm_high) &&....&&(0<=VEC_mu[t=T-1]<=mu_norm_high)
2	前提条件	t=0..T-1のすべての時点
	拘束条件	//検査範囲とする動摩擦係数値の一次連続性制約 mu_diff_low<=VEC_mu[t+1]-VEC_mu[t]<=mu_diff_high
	論理式	(mu_diff_low<=VEC_mu[t=1]-VEC_mu[t=0]<=mu_diff_high) &&...&&(mu_diff_low<=VEC_mu[t=T]-VEC_mu[t=T-1]<=mu_diff_high)
3	前提条件	t=1..T-dtのいずれかの時点
	拘束条件	((0<=VEC_mu[t]<=mu_norm_low)&....&(0<=VEC_mu[t+dT]<=mu_norm_low)
	論理式	((0<=VEC_mu[t=1]<=mu_norm_low&...&0<=VEC_mu[t=1+dT]<=mu_norm_low)||.... ||(0<=VEC_mu[t=T-dt]<=mu_norm_low&...&0<=VEC_mu[t=1+T]<=mu_norm_low))
4	前提条件	t=0..T-1のすべての時点
	拘束条件	//検査範囲とするアクセル踏み込み値の上下限制約 (ACCEL_low<= DATA_IN_ACCEL_PEDAL[t]<=ACCEL_high)
	論理式	検査範囲とするアクセル踏み込み値の上下限制約 (ACCEL_low<=DATA_IN_ACCEL_PEDAL[t=0]<=ACCEL_high) &&...&&(ACCEL_low<=DATA_IN_ACCEL_PEDAL[t=T-1]<=ACCEL_high) &&
5	前提条件	t=0..T-1のすべての時点
	拘束条件	//検査範囲とするアクセル踏み込み値の一次連続性制約 (ACCEL_diff_low<= DATA_IN_ACCEL_PEDAL[t=1]-DATA_IN_ACCEL_PEDAL[t=0] <=ACCEL_diff_high)
	論理式	(ACCEL_diff_low<=DATA_IN_ACCEL_PEDAL[t=1]-DATA_IN_ACCEL_PEDAL[t=0]<=ACCEL_diff_high) &&... (ACCEL_diff_low<=DATA_IN_ACCEL_PEDAL[t=T]-DATA_IN_ACCEL_PEDAL[t=T-1]<=ACCEL_diff_high)

【図９Ｂ】

図9B

		入力値列制約条件
6	前提条件	無し
	拘束条件	最終的にアクセルが緩められて下限値をとる事//下式と同じ
	論理式	(ACCEL_DEV_LOW<=DATA_IN_ACCEL_PEDAL[t=0]-DATA_IN_ACCEL_PEDAL[t=T]) &&(DATA_IN_ACCEL_PEDAL[t=T]==ACCEL_low)
7	前提条件	t=0..T-1のすべての時点
	拘束条件	// 検査範囲とするブレーキ踏み込み値の上下限制約 (BRAKE_low<= DATA_IN_BRAKE_PEDAL[t=0]<=BRAKE_high)
	論理式	(BRAKE_low<=DATA_IN_BRAKE_PEDAL[t=0..T-1]<=BRAKE_high)
8	前提条件	t=0..T-1のすべての時点
	拘束条件	//検査範囲とする踏み込み値の一次連続性制約 (BRAKE_diff_low<= DATA_IN_BRAKE_PEDAL[t=1]-DATA_IN_BRAKE_PEDAL[t=0]<=BRAKE_diff_high)
	論理式	(BRAKE_diff_low<=DATA_IN_BRAKE_PEDAL[t=1]-DATA_IN_BRAKE_PEDAL[t=0]<=BRAKE_diff_high) && ... (BRAKE_diff_low<=DATA_IN_BRAKE_PEDAL[t=T]-DATA_IN_BRAKE_PEDAL[t=T-1]<=BRAKE_diff_high)
9	前提条件	無し
	拘束条件	最終的に時点t=Tでブレーキが踏み込まれて上限値をとる事//下式と同じ
	論理式	(BRAKE_DEV_LOW<=DATA_IN_BRAKE_PEDAL[t=T]-DATA_IN_BRAKE_PEDAL[t=0]) &&DATA_IN_BRAKE_PEDAL[t=T]==BRAKE_high

【図１０】

図10

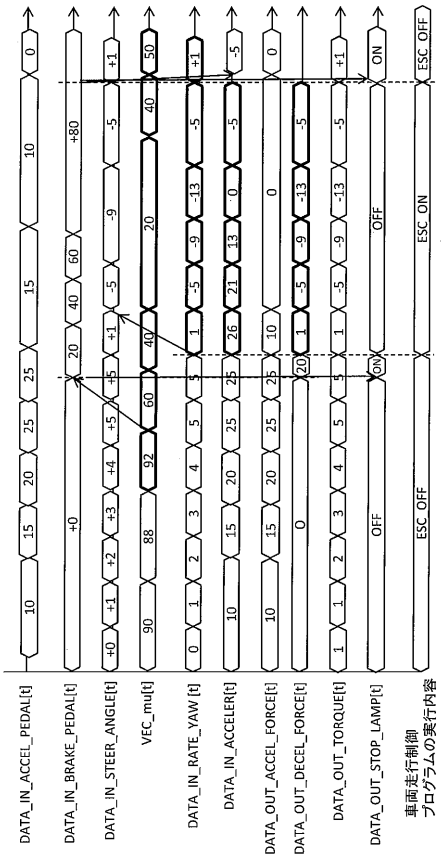
		検査条件
1	前提条件	t=0..T-1のすべての時点で、 ブレーキ踏み込み条件(BRAKE_ON_MIN<= DATA_IN_BRAKE_PEDAL[t]) が成立する場合に、
	拘束条件	//ストップランプの点灯 DATA_OUT_STOP_LAMP[t+1]=ON
	論理式	(! ((BRAKE_ON_MIN<= DATA_IN_BRAKE_PEDAL[t=0]))| |DATA_OUT_STOP_LAMP[t=1]=ON)&& ....&& (! ((BRAKE_ON_MIN<= DATA_IN_BRAKE_PEDAL[t=T-1]))| |DATA_OUT_STOP_LAMP[t=T]=ON)
2	前提条件	t=0..T-1のすべての時点で、 ブレーキ踏み込み値条件C0[t]= (BRAKE_ON_MIN<= DATA_IN_BRAKE_PEDAL[t]) が成立する場合に、

	拘束条件	// 減速度力量値が上限値DECEL_MAX未満であれば、ブレーキ踏みこみ値に依存する下記所定値範囲内に収まり、
		論理式 C1[t]=!((DATA_OUT_DECEL_FORCE[t])<=DECEL_MAX-DECEL_DIV_MIN)| |
		((BRAKE_DIV_MIN<=DATA_OUT_DECEL_FORCE[t+1]-DATA_OUT_DECEL_FORCE[t])&&
		(DATA_OUT_DECEL_FORCE[t+1]-
		DATA_OUT_DECEL_FORCE[t]<=BRAKE_DIV_MIN+CONST*(DATA_IN_BRAKE_PEDAL[t] -
		BRAKE_ON_MIN)));
		// 減速度力量値が上限値DECEL_MAXであれば、減速度力量値は最大値DECEL_MAXを保持する
		論理式 C2[t]=(!((DATA_OUT_DECEL_FORCE[t])>=DECEL_MAX-DECEL_DIV_MIN)| |
		(DATA_OUT_DECEL_FORCE[t+1]==DECEL_MAX));

		論理式 {!((C0[t=0])		(C1[t=0)&&C2[t=0]))&&....&&!((C0[t=T-1])		(C1[t=T-1]&&C2[t=T-1]))}

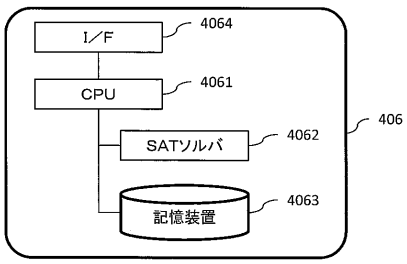
【図１１】

図11



【図１２】

図12



フロントページの続き

(72)発明者　大野　敦寛
茨城県ひたちなか市高場２５２０番地　日立オートモティブシステムズ株式会社内
(72)発明者　松原　正裕
東京都千代田区丸の内一丁目６番６号　株式会社日立製作所内

Ｆターム(参考) 5B042 GB08 HH07 HH17
5B376 BB05 BC38 BC69 BC70 BC71