(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2021-124852
(P2021-124852A)
(43)公開日　令和3年8月30日(2021.8.30)

(51)Int.Cl.		F I			テーマコード（参考）
G06F 11/36	(2006.01)	G06F	11/36	124	5B042
		G06F	11/36	108	

審査請求 未請求　請求項の数 20　ＯＬ　外国語出願　（全 35 頁）

(21)出願番号　特願2020-16769(P2020-16769)
(22)出願日　令和2年2月4日(2020.2.4)

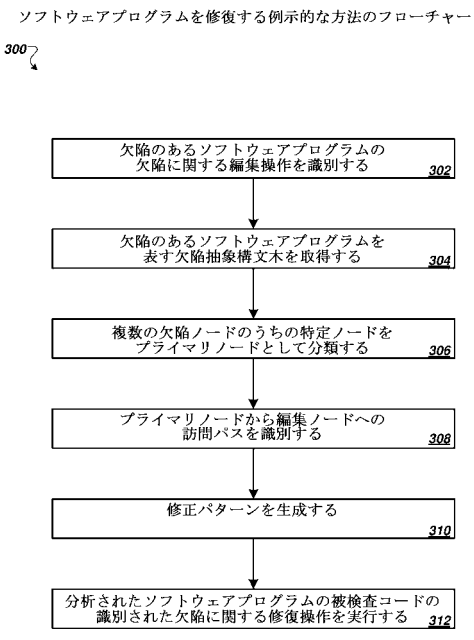
特許法第３０条第２項適用申請有り　（学会予稿集のウェブサイト）　掲載日　令和１年８月１２日　アドレス　ｈｔｔｐｓ：／／ｅｓｅｃ－ｆｓｅ１９．ｕｔ．ｅｅ／ｐｒｏｃｅｅｄｉｎｇｓ／　（学会）　開催日　令和１年８月２９日　集会名、開催場所　ＦＳＥ２０１９

(71)出願人　000005223
富士通株式会社
神奈川県川崎市中原区上小田中４丁目１番１号
(74)代理人　100107766
弁理士　伊東　忠重
(74)代理人　100070150
弁理士　伊東　忠彦
(72)発明者　バヴィシ・ロハン
アメリカ合衆国，カリフォルニア州　９４０８５，サニーヴェイル，イースト　アークス　アヴェニュ　１２４０番　フジツウ　ラボラトリーズ　アメリカ内

最終頁に続く

(54)【発明の名称】修正パターンのデータ駆動型合成

(57)【要約】　（修正有）
【課題】修正パターンのデータ駆動型合成によりソフトウェアプログラムを修正する方法を提供する。
【解決手段】ソフトウェアプログラムを修復するフローチャートは、以下のステップを含む。プログラムの欠陥に対応する編集を識別する３０２。編集は、プログラムと改善されたプログラムとの間の相違に基づき識別される。ＡＳＴ(抽象構文木)は、プログラムを表し、欠陥位置に対応する欠陥ノードを含み、編集ノードを含んだ、ＡＳＴを取得する３０４。編集の開始点として動作するプライマリノードとして特定ノードを分類する３０６。プライマリノードから編集ノードへのパスを識別する３０８。パス及び編集に基づき、修正パターンを生成する３１０。コードの識別した欠陥に関する修復を実行する３１２。修復は、修正パターンを使用し、コードの識別された欠陥がプログラムの欠陥と同じ種類であることに基づき、実行される。
【選択図】図３



【特許請求の範囲】
【請求項１】
欠陥のあるソフトウェアプログラムと欠陥の修復としての編集操作を含む改善されたソフトウェアプログラムとの間の１つ以上の相違に基づき、前記欠陥のあるソフトウェアプログラムの前記欠陥に関する前記編集操作を識別するステップと、
前記欠陥のあるソフトウェアプログラムを表す欠陥抽象構文木（AST）を取得するステップであって、前記欠陥ASTは、前記欠陥を含む前記欠陥のあるソフトウェアプログラムの欠陥位置に対応する複数の欠陥ノードを含み、前記複数の欠陥ノードは前記編集操作により変更される編集ノードを含む、ステップと、
前記複数の欠陥ノードのうちの特定ノードを、前記編集操作を実施する際の開始点として動作するプライマリノードとして分類するステップと、
前記プライマリノードから前記編集ノードへの訪問パスを識別するステップであって、前記訪問パスは前記欠陥ASTに対応する訪問ASTに含まれ、前記訪問パスは前記プライマリノードと前記編集ノードとの間のプログラム上の関係を示す１つ以上の訪問エッジのシーケンスを含む、ステップと、
前記訪問パス及び前記編集操作に基づき修正パターンを生成するステップであって、前記修正パターンは、前記欠陥のあるソフトウェアプログラムのソースコードに適合する形式で生成される、ステップと、
前記修正パターンを用いて、被検査コードの識別された欠陥が前記欠陥のあるソフトウェアプログラムの前記欠陥と同じ種類であることに基づき、被分析ソフトウェアプログラムの前記被検査コードの前記識別された欠陥に関する修復操作を実行するステップと、
を含む方法。
【請求項２】
前記編集ノードは、第１編集ノードであり、前記複数の欠陥ノードは前記編集操作により変更された第２編集ノードを含み、前記訪問パスは第１訪問パスであり、前記方法は、さらに、
前記プライマリノードから前記第２編集ノードへの第２訪問パスを識別するステップであって、前記修正パターンは前記第２訪問パスに更に基づく、ステップ、を含む請求項１に記載の方法。
【請求項３】
前記第１訪問パス及び前記第２訪問パスが同じ種類の訪問エッジの同じシーケンスを有することに基づき、前記第１訪問パス及び前記第２訪問パスを、互いに関して共通訪問パスであると識別するステップを更に含み、
前記修正パターンを生成するステップは、前記第１訪問パス及び前記第２訪問パスが共通訪問パスとして識別されることに基づく、請求項２に記載の方法。
【請求項４】
前記訪問パスが、前記欠陥又は前記欠陥のあるソフトウェアプログラムの他の欠陥に関連する他の訪問パスと比べて少数の訪問エッジを有することに基づき、前記修正パターンを生成する際に使用するために、前記訪問パスを選択するステップ、を更に含む請求項１に記載の方法。
【請求項５】
前記欠陥のあるソフトウェアプログラムは、複数の欠陥を含み、前記複数の欠陥ノードのうちの前記特定ノードを前記プライマリノードとして分類する前記ステップは、
前記欠陥のあるソフトウェアプログラムの中の前記複数の欠陥のうちの少なくとも一部の複数の共通欠陥属性を識別するステップであって、前記複数の欠陥ノードのうちの前記特定ノードは、前記特定ノードが前記欠陥のあるソフトウェアプログラムの中の前記複数の欠陥のうちの前記一部の前記複数の共通欠陥属性のうちの１つ以上を含むことに基づき、前記プライマリノードとして分類される、ステップを含む、請求項１に記載の方法。
【請求項６】
前記欠陥のあるソフトウェアプログラムの前記欠陥は第１欠陥であり、前記編集操作は

第１編集操作であり、前記プライマリノードは第１プライマリノードであり、前記訪問パスは第１訪問パスであり、前記編集ノードは第１編集ノードであり、前記複数の欠陥ノードは第２編集ノードを更に含み、前記方法は、さらに、
前記欠陥のあるソフトウェアプログラムと第２欠陥の修復としての第２編集操作を含む前記改善されたソフトウェアプログラムとの間の前記１つ以上の相違に基づき、前記欠陥のあるソフトウェアプログラムの前記第２欠陥に関する前記第２編集操作を識別するステップと、
前記複数の欠陥ノードのうちの別の特定ノードを、前記第２編集操作を実施する際の開始点として動作する第２プライマリノードとして分類するステップと、
前記第２プライマリノードから前記第２編集ノードへの第２訪問パスを識別するステップであって、前記第２訪問パスは、前記訪問ASTに含まれ、前記第２プライマリノードと前記第２編集ノードとの間のプログラム上の関係を示し、前記修正パターンは、前記第２訪問パス及び前記第２編集操作に更に基づく、ステップと、
を含む請求項１に記載の方法。
【請求項７】
前記第１訪問パス及び前記第２訪問パスが類似する訪問パス記述を含むことに基づき、前記第１訪問パス及び前記第２訪問パスを、互いに関して共通訪問パスであると識別するステップを更に含み、
前記修正パターンを生成するステップは、前記第１訪問パス及び前記第２訪問パスが共通訪問パスとして識別されることに基づく、請求項２に記載の方法。
【請求項８】
命令を格納するよう構成された１つ以上の非一時的コンピュータ可読記憶媒体であって、前記命令は、実行されることに応答して、システムに動作を実行させ、前記動作は、
欠陥のあるソフトウェアプログラムと欠陥の修復としての編集操作を含む改善されたソフトウェアプログラムとの間の１つ以上の相違に基づき、前記欠陥のあるソフトウェアプログラムの前記欠陥に関する前記編集操作を識別するステップと、
前記欠陥のあるソフトウェアプログラムを表す欠陥ASTを取得するステップであって、前記欠陥ASTは、前記欠陥を含む前記欠陥のあるソフトウェアプログラムの欠陥位置に対応する複数の欠陥ノードを含み、前記複数の欠陥ノードは前記編集操作により変更される編集ノードを含む、ステップと、
前記複数の欠陥ノードのうちの特定ノードを、前記編集操作を実施する際の開始点として動作するプライマリノードとして分類するステップと、
前記プライマリノードから前記編集ノードへの訪問パスを識別するステップであって、前記訪問パスは前記欠陥ASTに対応する訪問ASTに含まれ、前記訪問パスは前記プライマリノードと前記編集ノードとの間のプログラム上の関係を示す１つ以上の訪問エッジのシーケンスを含む、ステップと、
前記訪問パス及び前記編集操作に基づき修正パターンを生成するステップであって、前記修正パターンは、前記欠陥のあるソフトウェアプログラムのソースコードに適合する形式で生成される、ステップと、
前記修正パターンを用いて、被検査コードの識別された欠陥が前記欠陥のあるソフトウェアプログラムの前記欠陥と同じ種類であることに基づき、被分析ソフトウェアプログラムの前記被検査コードの前記識別された欠陥に関する修復操作を実行するステップと、
を含む、１つ以上の非一時的コンピュータ可読記憶媒体。
【請求項９】
前記編集ノードは、第１編集ノードであり、前記複数の欠陥ノードは前記編集操作により変更された第２編集ノードを含み、前記訪問パスは第１訪問パスであり、前記動作は、さらに、
前記プライマリノードから前記第２編集ノードへの第２訪問パスを識別するステップであって、前記修正パターンは前記第２訪問パスに更に基づく、ステップ、を含む請求項８に記載の１つ以上の非一時的コンピュータ可読記憶媒体。

【請求項１０】
前記動作は、前記第１訪問パス及び前記第２訪問パスが同じ種類の訪問エッジの同じシーケンスを有することに基づき、前記第１訪問パス及び前記第２訪問パスを、互いに関して共通訪問パスであると識別するステップを更に含み、
前記修正パターンを生成するステップは、前記第１訪問パス及び前記第２訪問パスが共通訪問パスとして識別されることに基づく、請求項９に記載の１つ以上の非一時的コンピュータ可読記憶媒体。
【請求項１１】
前記複数の欠陥ノードのうちの前記特定ノードを前記プライマリノードとして分類するステップは、
前記欠陥のあるソフトウェアプログラムの前記欠陥と同様の複数の欠陥を含む複数の検査用の欠陥のあるソフトウェアプログラムを取得するステップと、
前記複数の検査用の欠陥のあるソフトウェアプログラムの中の前記複数の欠陥のうちの少なくとも一部の複数の共通欠陥属性を識別するステップであって、前記複数の欠陥ノードのうちの前記特定ノードは、前記特定ノードが、前記複数の検査用の欠陥のあるソフトウェアプログラムの中の前記複数の欠陥のうちの前記一部の前記複数の共通欠陥属性のうちの１つ以上を含むことに基づき、前記プライマリノードとして分類される、ステップと、
を含む、請求項８に記載の１つ以上の非一時的コンピュータ可読記憶媒体。
【請求項１２】
前記欠陥のあるソフトウェアプログラムの前記欠陥は第１欠陥であり、前記編集操作は第１編集操作であり、前記プライマリノードは第１プライマリノードであり、前記訪問パスは第１訪問パスであり、前記編集ノードは第１編集ノードであり、前記複数の欠陥ノードは第２編集ノードを更に含み、前記動作は、さらに、
前記欠陥のあるソフトウェアプログラムと第２欠陥の修復としての第２編集操作を含む前記改善されたソフトウェアプログラムとの間の前記１つ以上の相違に基づき、前記欠陥のあるソフトウェアプログラムの前記第２欠陥に関する前記第２編集操作を識別するステップと、
前記複数の欠陥ノードのうちの別の特定ノードを、前記第２編集操作を実施する際の開始点として動作する第２プライマリノードとして分類するステップと、
前記第２プライマリノードから前記第２編集ノードへの第２訪問パスを識別するステップであって、前記第２訪問パスは、前記訪問ASTに含まれ、前記第２プライマリノードと前記第２編集ノードとの間のプログラム上の関係を示し、前記修正パターンは、前記第２訪問パス及び前記第２編集操作に更に基づく、ステップと、
を含む、請求項８に記載の１つ以上の非一時的コンピュータ可読記憶媒体。
【請求項１３】
前記動作は、前記第１訪問パス及び前記第２訪問パスが類似する訪問パス記述を含むことに基づき、前記第１訪問パス及び前記第２訪問パスを、互いに関して共通訪問パスであると識別するステップを更に含み、
前記修正パターンを生成するステップは、前記第１訪問パス及び前記第２訪問パスが共通訪問パスとして識別されることに基づく、請求項１２に記載の１つ以上の非一時的コンピュータ可読記憶媒体。
【請求項１４】
システムであって、
命令を格納するよう構成された１つ以上の非一時的コンピュータ可読記憶媒体と、
前記１つ以上のコンピュータ可読記憶媒体に通信可能に結合され、前記命令の実行に応答して、前記システムに動作を実行させるよう構成される１つ以上のプロセッサと、
を含み、前記動作は、
欠陥のあるソフトウェアプログラムと欠陥の修復としての編集操作を含む改善されたソフトウェアプログラムとの間の１つ以上の相違に基づき、前記欠陥のあるソフトウェアプ

ログラムの前記欠陥に関する前記編集操作を識別するステップと、
前記欠陥のあるソフトウェアプログラムを表す欠陥ASTを取得するステップであって、前記欠陥ASTは、前記欠陥を含む前記欠陥のあるソフトウェアプログラムの欠陥位置に対応する複数の欠陥ノードを含み、前記複数の欠陥ノードは前記編集操作により変更される編集ノードを含む、ステップと、
前記複数の欠陥ノードのうちの特定ノードを、前記編集操作を実施する際の開始点として動作するプライマリノードとして分類するステップと、
前記プライマリノードから前記編集ノードへの訪問パスを識別するステップであって、前記訪問パスは前記欠陥ASTに対応する訪問ASTに含まれ、前記訪問パスは前記プライマリノードと前記編集ノードとの間のプログラム上の関係を示す１つ以上の訪問エッジのシーケンスを含む、ステップと、
前記訪問パス及び前記編集操作に基づき修正パターンを生成するステップと、
前記修正パターンを用いて、被検査コードの識別された欠陥が前記欠陥のあるソフトウェアプログラムの前記欠陥と同じ種類であることに基づき、被分析ソフトウェアプログラムの前記被検査コードの前記識別された欠陥に関する修復操作を実行するステップと、
を含む、システム。
【請求項１５】
前記編集ノードは、第１編集ノードであり、前記複数の欠陥ノードは前記編集操作により変更された第２編集ノードを含み、前記訪問パスは第１訪問パスであり、前記動作は、さらに、
前記プライマリノードから前記第２編集ノードへの第２訪問パスを識別するステップであって、前記修正パターンは前記第２訪問パスに更に基づく、ステップ、を含む、請求項１４に記載のシステム。
【請求項１６】
前記第１訪問パス及び前記第２訪問パスが同じ種類の訪問エッジの同じシーケンスを有することに基づき、前記第１訪問パス及び前記第２訪問パスを、互いに関して共通訪問パスであると識別するステップを更に含み、
前記修正パターンを生成するステップは、前記第１訪問パス及び前記第２訪問パスが共通訪問パスとして識別されることに基づく、請求項１５に記載のシステム。
【請求項１７】
前記動作は、前記訪問パスが、前記欠陥又は前記欠陥のあるソフトウェアプログラムの他の欠陥に関連する他の訪問パスと比べて少数の訪問エッジを有することに基づき、前記修正パターンを生成する際に使用するために、前記訪問パスを選択するステップ、を更に含む、請求項１４に記載のシステム。
【請求項１８】
前記複数の欠陥ノードのうちの前記特定ノードを前記プライマリノードとして分類するステップは、
前記欠陥のあるソフトウェアプログラムの前記欠陥と同様の複数の欠陥を含む複数の検査用の欠陥のあるソフトウェアプログラムを取得するステップと、
前記複数の検査用の欠陥のあるソフトウェアプログラムの中の前記複数の欠陥のうちの少なくとも一部の複数の共通欠陥属性を識別するステップであって、前記複数の欠陥ノードのうちの前記特定ノードは、前記特定ノードが、前記複数の検査用の欠陥のあるソフトウェアプログラムの中の前記複数の欠陥のうちの前記一部の前記複数の共通欠陥属性のうちの１つ以上を含むことに基づき、前記プライマリノードとして分類される、ステップと、
を含む、請求項１４に記載のシステム。
【請求項１９】
前記欠陥のあるソフトウェアプログラムの前記欠陥は第１欠陥であり、前記編集操作は第１編集操作であり、前記プライマリノードは第１プライマリノードであり、前記訪問パスは第１訪問パスであり、前記編集ノードは第１編集ノードであり、前記複数の欠陥ノー

ドは第２編集ノードを更に含み、前記動作は、さらに、
前記欠陥のあるソフトウェアプログラムと第２欠陥の修復としての第２編集操作を含む前記改善されたソフトウェアプログラムとの間の前記１つ以上の相違に基づき、前記欠陥のあるソフトウェアプログラムの前記第２欠陥に関する前記第２編集操作を識別するステップと、
前記複数の欠陥ノードのうちの別の特定ノードを、前記第２編集操作を実施する際の開始点として動作する第２プライマリノードとして分類するステップと、
前記第２プライマリノードから前記第２編集ノードへの第２訪問パスを識別するステップであって、前記第２訪問パスは、前記訪問ASTに含まれ、前記第２プライマリノードと前記第２編集ノードとの間のプログラム上の関係を示し、前記修正パターンは、前記第２訪問パス及び前記第２編集操作に更に基づく、ステップと、
を含む、請求項１４に記載のシステム。
【請求項２０】
前記修正パターンは、前記欠陥のあるソフトウェアプログラムのソースコードに適合する形式で生成される、請求項１４に記載のシステム。

【発明の詳細な説明】
【技術分野】
【０００１】
本開示で議論される実施形態は、修正パターンのデータ駆動型合成に関する。
【背景技術】
【０００２】
ソフトウェアプログラムは、意図したように動作できない障害（通常、「バグ」と呼ばれる）をそれらの中に含むことが多い。ときに、静的解析ツールが、ソフトウェアプログラムの中の欠陥を識別するために使用される。
【０００３】
本開示で請求される主題は、上述のような欠点を解決する実施形態や上述のような環境でのみ機能する実施形態に限定されない。むしろ、この背景技術は、単に、本開示に記載される複数の実施形態が実施される技術分野の一例を説明するために提供される。
【発明の概要】
【０００４】
一実施形態の一態様によると、方法は、編集操作を識別するステップを含んでよい。編集操作は、欠陥のあるソフトウェアプログラムの欠陥に対応してよい。編集操作は、欠陥のあるソフトウェアプログラムと改善されたソフトウェアプログラムとの間の１つ以上の相違に基づき識別されてよい。改善されたソフトウェアプログラムは、欠陥の修復としての編集操作を含んでよい。方法は、欠陥抽象構文木（abstract syntax tree：AST）を取得するステップを更に含んでよい。欠陥ASTは、欠陥のあるソフトウェアプログラムを表してよい。欠陥ASTは、複数の欠陥ノードを含んでよい。複数の欠陥ノードは、欠陥を含む欠陥のあるソフトウェアプログラムの欠陥位置に対応してよい。複数の欠陥ノードは、編集操作により変更される編集ノードを含んでよい。さらに、方法は、複数の欠陥ノードのうちの特定ノードをプライマリノードとして分類するステップを含んでよい。プライマリノードは、編集操作を実施する際の開始点として動作してよい。方法は、プライマリノードから編集ノードへの訪問（visitor、ビジター）パスを識別するステップを更に含んでよい。訪問パスは、訪問ASTに含まれてよい。訪問ASTは欠陥ASTに対応してよい。訪問パスは、プライマリノードと編集ノードとの間のプログラム上の関係を示す１つ以上の訪問エッジのシーケンスを含んでよい。方法は、修正パターンを生成するステップを含んでよい。修正パターンは、訪問パス及び編集操作に基づいてよい。さらに、修正パターンは、欠陥のあるソフトウェアプログラムのソースコードに適合する形式で生成されてよい。方法は、修復操作を実行するステップを更に含んでよい。修復操作は、被分析ソフトウェアプログラムの被検査コードの識別された欠陥に関して実行されてよい。修復操作は、

修正パターンを用いて実行されてもよい。さらに、修復操作は、被検査コードの識別された欠陥が欠陥のあるソフトウェアプログラムの欠陥と同じ種類であることに基づき、実行されてよい。

【０００５】

実施形態の目的及び利点が理解され、少なくとも特に特許請求の範囲で指摘された要素、特徴及び組合せを用いて達成されるだろう。

【０００６】

上述の全体的説明及び以下の詳細な説明の両方は、例示及び説明のためであり、本発明の範囲を限定しない。

【図面の簡単な説明】

【０００７】

例示的な実施形態は、添付の図面を用いて、更なる特異性及び詳細事項と共に記載され説明される。

【０００８】

【図１】ソフトウェアプログラムの修復操作を実行するために使用され得る修正パターンの生成に関連する例示的な環境を示す図である。

【０００９】

【図２】ソフトウェアプログラムの修復操作を実行するために使用され得る修正パターンを生成するよう構成され得る例示的なコンピューティングシステムを示す。

【００１０】

【図３】ソフトウェアプログラムを修復する例示的な方法のフローチャートを示す。

【００１１】

【図４】ソフトウェアプログラムの修復操作を実行するために使用され得る修正パターンを生成する例示的な方法のフローチャートを示す。

【００１２】

【図５】特定ノードをプライマリノードとして分類する例示的な方法のフローチャートを示す。

【００１３】

【図６】例示的な欠陥AST及び図４の環境で実施され得る例示的な改善されたASTを示す。

【００１４】

全ての図は本開示に記載の少なくとも１つの実施形態に従う。

【発明を実施するための形態】

【００１５】

本開示に記載される幾つかの実施形態は、修正パターンのデータ駆動型合成に関する。ソフトウェアプログラムは、多くの場合、ソフトウェアプログラムを意図しない方法で動作させ得る欠陥（通常、「バグ」として参照される）を含む。さらに、ソフトウェアプログラム内の欠陥を検出するために、静的分析システム及び技術が使用される場合がある。例示的な静的分析器は、限定ではないが、PMD、Coverity、Coverity Scan、Facebook Infer、Google error－prone、SonarQube、及びFindBugs(SpotBugs)を含み得る。

【００１６】

静的分析システム及び技術は、ソフトウェアプログラム内の欠陥を識別することをテストスーツに要求しないことがある。対照的に、動的分析システム及び技術は、ソフトウェアプログラム内の欠陥を識別することをテストスーツに要求し得る。静的分析システム及び技術は、被検査ソフトウェアプログラム内の様々なソフトウェア品質問題（例えば、違反及び／又はバグ）を識別し得る。例えば、静的分析システム及び技術は、ソフトウェアプログラムの中の、文体的違反、共通のソフトウェアの脆弱性、セキュリティ脆弱性、及び／又は他の様式ガイドライン違反を識別し得る。違反が識別されると、静的分析システム及び技術は、違反に関連付けられた欠陥のあるドメイン固有言語（domain specific language：DSL）（例えば、疑わしいコード）にフラグを立て得る。これらの報告された違反は、ソフトウェアプログラムの動作の性能及び／又は正確さに影響を与え得る。

【００１７】
報告された違反は、多数（例えば、大量の報告された違反）であることがある。さらに、報告された違反は、ソフトウェアプログラムの機能にとって重大でないことがある。標準的に膨大な報告される違反及び時にはソフトウェアプログラムの機能に重大な影響を与えないものにより、報告された違反は、時に無視され修正されない（例えば修復されない）ことがある。これは、違反に関連するDSLを手動で修復するための時間量がソフトウェアプログラムの機能における利益を上回ることがあるという事実に起因し得る。これは、使用中にソフトウェアプログラムの動作の意図しない結果及び／又は性能低下を引き起こすことがあり、ユーザの苦情を解決するためにソフトウェアプログラムの開発後に時間を要することがある。

【００１８】
例えば、１．１百万個を超える違反が４千６百個のオープンソースソフトウェア（OSS）プロジェクトに渡り報告されたとする。これらの１．１百万個の報告された違反のうち、約６０万個が解決される（例えば、対応するDSLに修復が行われる）。さらに、ソフトウェア開発者は、彼らの時間の約５０％を費やして、報告された違反を発見し修復する。これは、約６ヶ月をプロジェクトに追加し得る。

【００１９】
修正パターンを手動で生成することにより報告された違反を手動で解決することは、膨大な数のトレーニングサンプルを必要とし得る。手動で生成した修正パターンは、置換コードを含むが、既存のDSLの修正を記述するステップを含まない、DSLパターンであり得る。さらに、幾つかの手動で生成した修正パターンは、既存のDSLを修正するための単純なステップを含み得る（例えば、手動で生成した修正パターンの中のDSLは、単純な構造的パターン照合に基づいてよい）。さらに、手動で生成した修正パターンは、DSLに対する変更を、DSL内ではなく、ソフトウェアプログラム内に自然言語で記述することがある。さらに、手動で生成した修正パターンは、重要な時間を生成することを要求するソフトウェア開発者により、手動で予め定められることがある。

【００２０】
本開示に記載の１つ以上の実施形態は、手動で修正パターンを生成するのに比べてソフトウェアプログラムの開発における遅延を軽減し得る方法で膨大数の報告された違反を解決するために現実世界のDSLプロジェクトに適用され得る修正パターン（例えば、パッチ）を生成するための自動システム及び技術を提供し得る。修正パターンは、欠陥のあるソフトウェアプログラム内の１つ以上の識別された欠陥（例えば、違反）に関して１つ以上の編集操作（例えば、修復パターン）を識別してよい。さらに、修正パターンは、欠陥のあるソフトウェアプログラムに関連するDSLに対して行われ得る修復（例えば、修正）及び／又は改善として、編集操作を一般化し及び提示し得る。さらに、修正パターンは、DSL内で及び／又は欠陥のあるソフトウェアプログラムのDSLに適合する形式で生成されてよい。

【００２１】
本開示に記載の実施形態は、プログラム合成アルゴリズムを用いて、一般化された実行可能な修正パターンを学習してよい。さらに、本開示に記載の実施形態は、DSL内の新しい及び／又は未だ見ぬ違反に適用され得る修正パターンのデータベースを生成してよい。

【００２２】
本開示に記載の１つ以上の実施形態は、例えば修正パターン生成によりプログラミングを実行し得る修正パターンモジュールを含んでよい。修正パターンモジュールは、ビッグコードソースから１つ以上の欠陥のあるソフトウェアプログラムを受信してよい。修正パターンモジュールは、欠陥のあるソフトウェアプログラム内で識別された１つ以上の欠陥に関して１つ以上の編集操作を識別してよい。さらに、修正パターンモジュールは、欠陥のあるソフトウェアプログラムに対応し及び欠陥のあるソフトウェアプログラム内の欠陥の修復として編集操作を含む、１つ以上の改善されたソフトウェアプログラムを受信してよい。

【００２３】
幾つかの実施形態では、修正パターンモジュールは、１つ以上の欠陥抽象構文木（abstract syntax trees：AST）を取得してよい。欠陥ASTは、欠陥のあるソフトウェアプログラムを表してよい。欠陥ASTは、欠陥のあるソフトウェアプログラムの欠陥位置に対応する１つ以上の欠陥ノードを含んでよい。各欠陥AST内の欠陥ノードのうちの１つ以上は、編集操作により変更される編集ノードとして識別されてよい。さらに、各欠陥AST内の編集ノードのうちの１つ以上は、プライマリノードとして分類されてよい。各プライマリノードは、編集操作を実施する際の開始点として動作してよい。
【００２４】
上述の及び他の実施形態では、修正パターンモジュールは、欠陥ASTの各々に対応する訪問（visitor）ASTを取得し及び／又は生成してよい。修正パターンモジュールは、プライマリノードから編集ノードへの訪問パス（visitor path）を識別するために、訪問ASTを使用してよい。さらに、修正パターンモジュールは、訪問パス及び修正操作に基づき、DSL内の修正パターンを生成してよい。
【００２５】
幾つかの実施形態では、修復モジュールは、修正パターン及び被検査コードを受信してよい。修復モジュールは、修正パターンを使用して、及び被検査コードの中の１つ以上の欠陥が欠陥のあるソフトウェアプログラム内の１つ以上の欠陥と同じである又は同様であることに基づき、被検査コードに対して修復操作を実行してよい。
【００２６】
本開示の実施形態を、添付の図面を参照して以下に説明する。
【００２７】
図１は、ソフトウェアプログラムの修復操作を実行するために使用され得る修正パターン１１０の生成に関連する例示的な環境１００を示す図である。環境１００は、欠陥のあるソフトウェアプログラム１０４のソースコードに対して行われた修復（例えば、修正）を一般化し及び表して、ビッグコード１０２から取得される改善されたソフトウェアプログラム１０６を取得するよう構成される修正パターンモジュール１０８を含んでよい。修正パターンモジュール１０８は、また、修正パターン１１０を出力するよう構成されてよい。修正パターン１１０は、欠陥のあるソフトウェアプログラム内の同様の欠陥を修復し及び／又は改善するために使用され得る修復を示す、ソースコード内に表される修復パターンを含んでよい。修正パターン１１０は、欠陥のあるソフトウェアプログラム１０４内の１つ以上の識別された欠陥（例えば、違反）に関して１つ以上の編集操作（例えば、修復パターン）を識別してよい。さらに、修正パターン１１０は、欠陥のあるソフトウェアプログラム１０４に関連するDSLに対して行われ得る修復（例えば、修正）及び／又は改善として、編集操作を一般化し及び表してよい。さらに、修正パターン１１０は、DSL内で及び／又は欠陥のあるソフトウェアプログラム１０４のDSLに適合する形式で生成されてよい。
【００２８】
さらに、環境１００は、欠陥に対する被検査コード１１１（例えば、別の欠陥のあるソフトウェアプログラム）を分析するよう構成される修復モジュール１０９を含んでよい。修復モジュール１０９は、変更された被検査コード１１３を出力するよう更に構成されてよい。変更された被検査コード１０８は、修復モジュール１０９により修正パターン１１０に基づき被検査コード１１１に対して行われた１つ以上の変更を含んでよい。
【００２９】
被検査コード１１１は、例えばソフトウェアプログラム、ソフトウェアプログラムのコード、ライブラリ、アプリケーション、スクリプト、又は処理装置による実行のための他のロジック若しくは命令（例えば、修復モジュール１０９）のような、電子データを有してよい。幾つかの実施形態では、被検査コード１１１は、ソフトウェアプログラムの完全なインスタンスを含んでよい。追加又は代替で、被検査コード１１１は、ソフトウェアプログラムの一部を含んでよい。被検査コード１１１は、ソフトウェアプログラムのために

及び静的分析のために使用され得る任意の適切な種類のコンピュータ言語で書かれてよい。

【００３０】

修正モジュール１０９は、コンピューティング装置に、被検査コード１１１の１つ以上の変更を実行させて、変更被検査コード１１３を生成させるよう構成されるコード及びルーチンを含んでよい。追加又は代替で、修復モジュール１０９は、プロセッサ、（例えば、１つ以上の操作を実行する又はその実行を制御する）マイクロプロセッサ、FPGA（field－programmable gate array）又はASIC（application－specific integrated circuit）を含むハードウェアを用いて実装されてよい。幾つかの他の例では、修復モジュール１０９は、ハードウェア及びソフトウェアの組み合わせを用いて実装されてよい。本開示では、修復モジュール１０９により実行されるとして記載される操作は、修復モジュール１０９が対応するシステムに実行するよう指示し得る操作を有してよい。

【００３１】

修復モジュール１０９は、被検査コード１１１の中の１つ以上の欠陥を修復する（本開示では修正するとしても参照される）ために使用され得る被検査コード１１１に関する一連の編集操作を実行するよう構成されてよい。幾つかの実施形態では、修復モジュール１０９は、修正パターン１１０に基づく編集操作のうちの１つ以上を実行するよう構成されてよい。幾つかの実施形態では、修正パターン１１０は、被検査コード１１１の静的プログラム解析又は任意の他の適切な種類のプログラム解析により発見された被検査コード１１１内の未だ見ぬ複雑な及び／又は他の欠陥を修復するために、修復モジュール１０９により使用されてよい。

【００３２】

ビッグコード１０２は、欠陥のあるソフトウェアプログラム１０４、改善されたソフトウェアプログラム１０６、及び複数の検査用の欠陥のあるソフトウェアプログラム１０７を含んでよい。ビッグコード１０２は、例えば欠陥のあるソフトウェアプログラム１０４、改善されたソフトウェアプログラム１０６、検査用の欠陥のあるソフトウェアプログラム１０７、種々のソフトウェアプログラムのソースコード、ライブラリ、アプリケーション、スクリプト、又は処理装置による実行のための他のロジック若しくは命令のような、電子データを含んでよい。幾つかの実施形態では、ビッグコード１０２は、欠陥のあるソフトウェアプログラム１０４、改善されたソフトウェアプログラム１０６、及び／又は検査用の欠陥のあるソフトウェアプログラム１０７の完全なインスタンスを含んでよい。追加又は代替として、ビッグコード１０２は、欠陥のあるソフトウェアプログラム１０４、改善されたソフトウェアプログラム１０６、及び／又は検査用の欠陥のあるソフトウェアプログラム１０７の一部を含んでよい。被検査コード１１１は、ソフトウェアプログラムのために及び静的分析のために使用され得る任意の適切な種類のコンピュータ言語で書かれてよい。例えば、欠陥のあるソフトウェアプログラム１０４、改善されたソフトウェアプログラム１０６、及び／又は検査用の欠陥のあるソフトウェアプログラム１０７は、静的分析器により分析可能であってよいDSLで書かれてよい。幾つかの実施形態では、欠陥のあるソフトウェアプログラム１０４、改善されたソフトウェアプログラム１０６、及び検査用の欠陥のあるソフトウェアプログラム１０７は、DSLと同じ種類又は互換性のある種類で被検査コード１１１として書かれてよい。

【００３３】

幾つかの実施形態では、ビッグコード１０２は、任意の適切なソースから取得された既存のコードを含んでよい。例えば、ビッグコード１０２は、プログラム開発者によりアップロードされたコードを含んでよい。欠陥のあるソフトウェアプログラム１０４、改善されたソフトウェアプログラム１０６、及び検査用の欠陥のあるソフトウェアプログラム１０７は、異なるソフトウェアプログラム（例えば、被検査コード１１１）内の欠陥を修復するために使用されてよい。幾つかの実施形態では、欠陥のあるソフトウェアプログラム１０４、改善されたソフトウェアプログラム１０６、及び検査用の欠陥のあるソフトウェアプログラム１０７は、機械可読DSLで、ビッグコード１０２に格納されてよい。追加又

は代替として、欠陥のあるソフトウェアプログラム１０４、改善されたソフトウェアプログラム１０６、及び検査用の欠陥のあるソフトウェアプログラム１０７は、人間に可読な形式で、ビッグコード１０２に格納されてよい。

【００３４】

欠陥のあるソフトウェアプログラム１０４は、静的分析器（図示しない）により識別された１つ以上の欠陥を含むソフトウェアプログラムであってよい。幾つかの実施形態では、静的分析器は、欠陥のあるソフトウェアプログラム１０４を分析して、欠陥のあるソフトウェアプログラム１０４のDSL内の１つ以上の欠陥を識別してよい。幾つかの実施形態では、静的分析は、欠陥のあるソフトウェアプログラム１０４がビッグコード１０２に格納される前に実行されてよい。他の実施形態では、欠陥のあるソフトウェアプログラム１０４は、ビッグコード１０２に格納されるとき、欠陥があるとしてラベル付けされてよく、修正パターンモジュール１０８は、欠陥のあるソフトウェアプログラム１０４の静的分析を実行してよい。

【００３５】

幾つかの実施形態では、欠陥のあるソフトウェアプログラム１０４内の欠陥は、欠陥のあるソフトウェアプログラム１０４のDSL内の１つ以上の構文違反及び／又は意味的違反を含んでよい。上述の及び他の実施形態では、欠陥のあるソフトウェアプログラム１０４内の欠陥の識別は、それぞれの欠陥に対応する欠陥のあるソフトウェアプログラム１０４内のDSLの１つ以上の行の欠陥の欠陥位置を識別してよい。

【００３６】

改善されたソフトウェアプログラム１０６は、欠陥のあるソフトウェアプログラム１０４に関連するソフトウェアプログラムであってよい。例えば、改善されたソフトウェアプログラム１０６は、改善されたソフトウェアプログラム１０６内では１つ以上の欠陥が修復されていることを除き、欠陥のあるソフトウェアプログラム１０４と同じ又は同様の動作を実行してよい。例えば、改善されたソフトウェアプログラム１０６内のDSLの各行は、欠陥のあるソフトウェアプログラムのDSLの欠陥を含む１つ以上の行を除き、欠陥のあるソフトウェアプログラム１０４内のDSLの行と同じ又は同様であってよい。

【００３７】

幾つかの実施形態では、欠陥のあるソフトウェアプログラム１０４内の欠陥は、欠陥に関する１つ以上の編集操作を実行することにより、改善されたソフトウェアプログラム１０６において修正されてよい。編集操作は、それぞれの欠陥を修復すべき欠陥位置において、欠陥のあるソフトウェアプログラム１０４に対して行われ得る１つ以上の編集を含んでよい。その結果、改善されたソフトウェアプログラム１０６は、それぞれの欠陥の修復として対応する編集操作を含んでよい。特定の欠陥に対応する編集操作は、特定の欠陥を修正するために実行され得る１つ以上の編集に基づいてよい。編集操作は、新しいDSLの挿入、既存DSLの改訂、又は既存DSLの再構成、を含んでよい。欠陥のあるソフトウェアプログラム１０４及び改善されたソフトウェアプログラム１０６は、編集操作、及び改善されたソフトウェアプログラム１０６のDSLを取得するために欠陥のあるソフトウェアプログラム１０４のDSLに対して行われた対応する編集を識別するために使用されてよい。

【００３８】

修正パターンモジュール１０８は、コンピューティング装置に、改善されたソフトウェアプログラム１０６のDSLを取得するために欠陥のあるソフトウェアプログラム１０４のDSLに対して行われた編集操作を一般化し及び表すことを可能にするよう構成されるコード及びルーチンを含んでよい。追加又は代替として、修正パターンモジュール１０８は、（例えば、１又は複数の工程を実行する又はその性能を制御する）プロセッサ、マイクロプロセッサ、FPGA（field－programmable gate array）又はASIC（application－specific integrated circuit）を含むハードウェアを用いて実装できる。幾つかの他の例では、修正パターンモジュール１０８は、ハードウェア及びソフトウェアの組み合わせを用いて実装されても良い。本開示では、修正パターンモジュール１０８により実行されるとして記載される動作は、修正パターンモジュール１０８が対応するシステムに実行するよう

指示し得る動作を有してよい。

【００３９】

修正パターンモジュール１０８は、欠陥のあるソフトウェアプログラム１０４、改善されたソフトウェアプログラム１０６、及び／又は、改善されたソフトウェアプログラム１０６のDSLを取得するために欠陥のあるソフトウェアプログラム１０４のDSLに対して行われた編集操作を一般化し及び表すために使用され得る検査用の欠陥のあるソフトウェアプログラム１０７に関して一連の動作を実行するよう構成されてよい。さらに、修正パターンモジュール１０８は、以下に詳述するように、修正パターン１１０を生成してよい。幾つかの実施形態では、修正パターンモジュール１０８は、欠陥のあるソフトウェアプログラム１０４及び改善されたソフトウェアプログラム１０６の単一のペアに基づき、修正パターン１１０を生成してよい。他の実施形態では、修正パターンモジュール１０８は、欠陥のあるソフトウェアプログラム１０４及び改善されたソフトウェアプログラム１０６の複数のペアに基づき、修正パターン１１０を生成してよい。

【００４０】

例えば、修正パターンモジュール１０８は、文字列の（string）ソフトウェアルーチン（「文字列の例」とも呼ばれる）に対して行われる編集操作を表す修正パターン１１０を生成するために使用されてよい。文字列の例では、例示的なDSL内の欠陥のあるソースコードは次の通りであってよい：

【数１】

```
String message = "required artifacts missing:\n";
for (Artifact missingArtifact : missingArtifacts )
{
message += "  " + missingArtifact.getId() + "\n";
}
message += "\nfor the artifact:";
(throw new ArtifactResolutionException( message, project.getArtifact(),
project.getRemoteArtifactRepositories() );
```

さらに、文字列の例では、例示的なDSL内の改善されたソースコードは次の通りであってよい：

【数２】

```
StringBuffer message = new StringBuffer( "required artifacts missing:\n" );
for ( Artifact missingArtifact : missingArtifacts )
{
message.append( "  " ).append( missingArtifact.getId() ).append( '\n' );
}
message.append( "\nfor the artifact:" ); throw new
ArtifactResolutionException( message.toString(), project.getArtifact(),
project.getRemoteArtifactRepositories() );
```

【００４１】

幾つかの実施形態では、修正パターンモジュール１０８は、欠陥のあるソフトウェアプログラム１０４及び改善されたソフトウェアプログラム１０６をビッグコード１０２から取得してよい。上述の及び他の実施形態では、修正パターンモジュール１０８は、改善されたソフトウェアプログラム１０６を得るために行われた、欠陥のあるソフトウェアプログラム１０４内の特定の欠陥に関する編集操作を識別してよい。追加又は代替として、修正パターンモジュール１０８は、欠陥のあるソフトウェアプログラム１０４内の特定の欠陥に関する複数の編集操作を識別してよい。

【００４２】
幾つかの実施形態では、編集操作は、欠陥のあるソフトウェアプログラム１０４と改善されたソフトウェアプログラム１０６との間の１つ以上の相違に基づいてよい。他の実施形態では、編集操作は、プログラム解析（例えば、静的分析）により決定された欠陥のあるソフトウェアプログラム１０４内の欠陥の欠陥種類に基づいてよい。例えば、修正パターン１１０は、静的分析器から静的欠陥識別子を受信し、及び／又は静的分析器を用いて静的欠陥識別子を生成してよい。幾つかの実施形態では、静的欠陥識別子は、レポジトリ、及び欠陥に対応する欠陥のあるソフトウェアプログラム１０４のDSL内の行番号の記述を含んでよい。

【００４３】
文字列の例では、例示的なDSL内の強力なソフトウェアプログラムの静的欠陥識別子は、次の通りであってよい：

【数３】

SBSC_USE_STRINGBUFFER_CONCATENATION (performance issue) violation at PurgeLocalRepositoryMojo.java:632.

【００４４】
代替として、編集操作は、欠陥のあるソフトウェアプログラム１０４のDSL及び改善されたソフトウェアプログラム１０６のDSLの比較に基づき識別されてよい。追加又は代替として、編集操作は、欠陥のあるソフトウェアプログラム１０４を表す欠陥AST及び改善されたソフトウェアプログラム１０６を表す改善ASTの比較に基づき識別されてよい。上述の及び他の実施形態では、欠陥AST及び改善ASTは、欠陥のあるソフトウェアプログラム１０４と改善されたソフトウェアプログラム１０６との間の相違を、ソースコード及び／又は欠陥種類を用いて決定するよりも、該相違のより正確な表現を提供し得る。

【００４５】
幾つかの実施形態では、編集操作は、改善されたソフトウェアプログラム１０６を取得するために単一の変更（例えば、単一の編集）が欠陥のあるソフトウェアプログラム１０４に対して行われたことを示してよい。代替として、編集操作は、改善されたソフトウェアプログラム１０６を取得するために複数の変更が欠陥のあるソフトウェアプログラム１０４に対して行われたことを示してよい。例えば、編集操作は、第１編集が欠陥のあるソフトウェアプログラム１０４内の第１欠陥に適用されること、及び第２編集が欠陥のあるソフトウェアプログラム１０４内の第２欠陥に適用されること、を示してよい。

【００４６】
文字列の例では、編集操作は、４個の編集が文字列のソフトウェアルーチンのDSLに対して生じたことを示してよい。自然言語で表わされた４個の編集は次の通りであってよい：宣言型をStringBufferで置き換え、コンストラクタ呼び出しにより初期化し、全ての追加割り当てをappendで置き換え、全ての文字列の使用をtoStringの呼び出しで置き換える。さらに、例示的なDSL（例えば、ソースコード）内で表される文字列のソフトウェアルーチンのDSLに対する４個の編集は、以下を含んでよい：

【数４】

$$op_1 = \textbf{Update}(L, \text{ConstNode}(\textbf{VarDecl}, \text{“}\textbf{StringBuffer}\text{”}, L.var, \text{ConstNode}(\text{Constructor}, \text{Type}(\text{“}\textbf{StringBuffer}\text{”}), L.init)))$$
$$op_2 = \textbf{Update}(L, \text{ConstNode}(\textbf{MethodInvocation}, L.lhs, \text{“}\textbf{append}\text{”}, L.rhs))$$
$$op_3 = \textbf{Update}(L, \text{ConstNode}(\textbf{MethodInvocation}, L.lhs, \text{“}\textbf{append}\text{”}, L.rhs))$$
$$op_4 = \textbf{Update}(n, \text{ConstNode}(\textbf{MethodInvocation}, n.name, \text{“}\textbf{toString}\text{”}, null))$$

【００４７】
幾つかの実施形態では、修正パターンモジュール１０８は、欠陥AST及び改善ASTをビッ

グコード１０２から取得してよい。他の実施形態では、修正パターンモジュール１０８は、欠陥のあるソフトウェアプログラム１０４を用いて欠陥ASTを、及び改善されたソフトウェアプログラム１０６を用いて改善ASTを生成してよい。欠陥AST及び改善ASTは、任意の適切なAST生成技術を用いて生成されてよい。

【００４８】

欠陥AST及び改善ASTは、それぞれ、欠陥のあるソフトウェアプログラム１０４及び改善されたソフトウェアプログラム１０６の抽象構文構造を表してよい。さらに、欠陥AST及び改善ASTは、複数のノードを含んでよい。各ノードは、対応するソフトウェアプログラムの要素（例えば、対応するソフトウェアプログラムの構成）に対応してよい。

【００４９】

欠陥ASTは、複数の使用可能（operational）ノード、及び１つ以上の欠陥ノードを含んでよい。使用可能ノードは、いかなる欠陥にも関連付けられていない、欠陥のあるソフトウェアプログラム１０４のDSL内の要素に対応してよい。さらに、欠陥ノードは、欠陥に関連付けられている、欠陥のあるソフトウェアプログラム１０４のDSL内の要素に対応してよい。幾つかの実施形態では、欠陥ノードは、欠陥のあるソフトウェアプログラム１０４内の欠陥の欠陥位置に対応してよい。

【００５０】

欠陥ノードは、１つ以上の編集ノードを含んでよい。編集ノードは、改善されたソフトウェアプログラム１０６を取得するために編集される、欠陥のあるソフトウェアプログラム１０４のDSLの部分に対応してよい。

【００５１】

欠陥AST及び改善ASTは、複数のASTエッジも含んでよい。ASTエッジは、対応するソフトウェアプログラムの制御フロー内のジャンプを表してよい。さらに、ASTエッジは、対応するASTのノード間のホップを表してよい。

【００５２】

幾つかの実施形態では、修正パターンモジュール１０８は、欠陥のあるソフトウェアプログラム１０４内の欠陥の各々の欠陥属性を識別してよい。修正パターンモジュール１０８は、欠陥のあるソフトウェアプログラム１０４内の欠陥の少なくとも一部に共通である欠陥属性を識別してよい。幾つかの実施形態では、欠陥属性は、欠陥種類、欠陥位置、ノード名、他のノード属性、例えば、キーと対応する値とのペア（例えば、キーは「種類」であり、対応する値は「String（文字列型）」である）、ノード種類、及びノードラベル、を含んでよい。幾つかの実施形態では、欠陥属性は、欠陥のあるソフトウェアプログラム１０４のDSLの属性に対応してよい。他の実施形態では、欠陥属性は、欠陥のあるソフトウェアプログラム１０４を表すASTの属性に対応してよい。

【００５３】

幾つかの実施形態では、修正パターンモジュール１０８は、欠陥属性が欠陥のあるソフトウェアプログラム１０４内の欠陥の各々に関連付けられている場合、欠陥属性を共通欠陥属性として分類してよい。他の実施形態では、修正パターンモジュール１０８は、欠陥属性が欠陥のあるソフトウェアプログラム１０４内の欠陥の少なくとも一部に関連付けられている場合、欠陥属性を共通欠陥属性として分類してよい。幾つかの実施形態では、欠陥属性が欠陥のあるソフトウェアプログラム１０４内の欠陥の少なくとも大部分に関連付けられている場合、欠陥属性は共通欠陥属性として分類されてよい。

【００５４】

幾つかの実施形態では、ビッグコード１０２は、複数の欠陥のあるソフトウェアプログラム１０４を含んでよい。上述の及び他の実施形態では、修正パターンモジュール１０８は、単一の欠陥のあるソフトウェアプログラム１０４ではなく、複数の欠陥のあるソフトウェアプログラム１０４を用いて共通欠陥属性を識別してよい。

【００５５】

追加又は代替として、修正パターンモジュール１０８は、ビッグコード１０２から検査用の欠陥のあるソフトウェアプログラム１０７を取得してよい。幾つかの実施形態では、

検査用の欠陥のあるソフトウェアプログラム１０７は、欠陥を含む複数のソフトウェアプログラムを含んでよい。検査用の欠陥のあるソフトウェアプログラム１０７内の欠陥は、欠陥のあるソフトウェアプログラム１０４内の欠陥と同じ又は同様であってよい。例えば、検査用の欠陥のあるソフトウェアプログラム１０７内の欠陥は、同じ又は同様の欠陥種類であってよい。別の例では、検査用の欠陥のあるソフトウェアプログラム１０７内の欠陥は、同じ又は同様の欠陥位置を有してよい。

【００５６】

幾つかの実施形態では、修正パターンモジュール１０８は、検査用の欠陥のあるソフトウェアプログラム１０７内の欠陥の各々の欠陥属性を識別してよい。上述の及び他の実施形態では、修正パターンモジュール１０８は、欠陥のあるソフトウェアプログラム１０７内の欠陥の少なくとも一部に共通である欠陥属性を識別してよい。幾つかの実施形態では、欠陥属性は、欠陥種類、欠陥位置、ノード名、他のノード属性、例えば、キーと対応する値とのペア（例えば、キーは「種類」であり、対応する値は「String（文字列型）」である）、ノード種類、及びノードラベル、を含んでよい。幾つかの実施形態では、欠陥属性は、検査用の欠陥のあるソフトウェアプログラム１０７のDSLの属性に対応してよい。他の実施形態では、欠陥属性は、検査用の欠陥のあるソフトウェアプログラム１０７を表すASTの属性に対応してよい。

【００５７】

幾つかの実施形態では、修正パターンモジュール１０８は、欠陥属性が検査用の欠陥のあるソフトウェアプログラム１０７内の欠陥の各々に関連付けられている場合、欠陥属性を共通欠陥属性として分類してよい。他の実施形態では、修正パターンモジュール１０８は、欠陥属性が検査用の欠陥のあるソフトウェアプログラム１０７内の欠陥の少なくとも一部に関連付けられている場合、欠陥属性を共通欠陥属性として分類してよい。幾つかの実施形態では、欠陥属性が検査用の欠陥のあるソフトウェアプログラム１０７内の欠陥の少なくとも大部分に関連付けられている場合、欠陥属性は共通欠陥属性として分類されてよい。

【００５８】

修正パターンモジュール１０８は、（例えば、欠陥のあるソフトウェアプログラム１０４、他の欠陥のあるソフトウェアプログラム１０４、及び／又は検査用の欠陥のあるソフトウェアプログラム１０７から取得された）共通欠陥属性を使用して、欠陥のあるソフトウェアプログラム１０４に関連付けられた１つ以上の欠陥ノードをプライマリノードとして分類してよい。幾つかの実施形態では、修正パターンモジュール１０８は、任意の欠陥ノードが共通欠陥属性のうちのいずれかに関連付けられているか否かを決定してよい。幾つかの実施形態では、修正パターンモジュール１０８は、欠陥ノードが共通欠陥属性のうちの全部に関連付けられている場合、欠陥ノードをプライマリノードとして分類してよい。代替として、修正パターンモジュール１０８は、欠陥ノードが共通欠陥属性のうちの少なくとも１つに関連付けられている場合、欠陥ノードをプライマリノードとして分類してよい。

【００５９】

幾つかの実施形態では、修正パターンモジュール１０８は、欠陥のあるソフトウェアプログラム１０４の特定の欠陥に関連付けられた１つの欠陥ノードを、プライマリノードとして分類してよい。他の実施形態では、修正パターンモジュール１０８は、欠陥のあるソフトウェアプログラム１０４の特定の欠陥に関連付けられた１つより多くの欠陥ノードを、プライマリノードとして分類してよい。幾つかの実施形態では、修正パターンモジュール１０８は、欠陥AST内の単一の欠陥ノードをプライマリノードとして分類してよい。他の実施形態では、修正パターンモジュール１０８は、欠陥AST内の１つより多くの欠陥ノードをプライマリノードとして分類してよい。

【００６０】

他の実施形態では、プライマリノードは、編集操作を実施する際の開始点として動作してよい。他の実施形態では、プライマリノードは、編集操作の一部を実施する際の開始点

として動作してよい。さらに、プライマリノードは、複数の編集操作を実施する際の開始点として動作してよい。幾つかの実施形態では、編集操作は、改善されたソフトウェアプログラム１０６を取得するために欠陥のあるソフトウェアプログラム１０４に対して行われる変更の中心であってよい。幾つかの実施形態では、プライマリノードは、編集ノードとして分類されてもよい。

【００６１】

修正パターンモジュール１０８は、編集操作を実施する際に開始点として動作し得る、異なるソフトウェアプログラム内のコード（例えば、被検査コード１１１）の欠陥のある行を決定するための検索条件を生成するために、プライマリノードを使用してよい。編集操作を実施する際の開始点として動作する、欠陥のあるソフトウェアプログラム内のDSLの欠陥のある行に対応する欠陥ノードは、プライマリノードとして分類されてよい。追加又は代替として、修正パターンモジュール１０８は、編集操作を実施する際に開始点として動作し得る、異なるソフトウェアプログラム内のDSLの欠陥のある行に対応する欠陥ノードを直接決定するための検索条件を生成するために、プライマリノードを使用してよい。

【００６２】

文字列の例では、修正パターンモジュール１０８により生成された例示的なDSL内の検索条件は、次の通りであってよい：

【数５】

$$\text{pNode} = \text{MatchContext}(\text{id}(\text{nodeType:}\mathbf{\text{AssignStmt}}), \text{SAnalyzerReport}())$$

【００６３】

修正パターンモジュール１０８は、欠陥ASTに基づき、訪問ASTを生成してよい。訪問ASTは欠陥のあるソフトウェアプログラム１０４に対応してよい。さらに、訪問ASTは、欠陥のあるソフトウェアプログラム１０４のプログラム構造を表してよい。訪問ASTは、複数の訪問エッジを含んでよい。訪問エッジは、訪問AST内のノード間のプログラム上の関係を表してよい。プログラム上の関係は、親、子、decl、lhs、呼び出し元、又は任意の他の適切なプログラム上の関係を含んでよい。

【００６４】

幾つかの実施形態では、修正パターンモジュール１０８は、欠陥ASTの各ノード及び各ASTエッジを含む、重複（duplicate）欠陥ASTを生成してよい。幾つかの実施形態では、修正パターンモジュール１０８は、重複欠陥ASTから全部のASTエッジを削除してよい。修正パターンモジュール１０８は、重複欠陥AST内の種々のノード間の訪問エッジを追加してよい。訪問エッジが追加されると、重複欠陥ASTは、訪問ASTとして分類されてよい。

【００６５】

修正パターンモジュール１０８は、プライマリノードと各編集ノードとの間の、訪問AST内の各訪問パスを識別してよい。訪問パスは、プライマリノードから編集ノードへと辿られる訪問エッジの経路であってよい。複数の訪問パスは、プライマリノードから異なる編集ノードへの、訪問エッジの同じ又は同様のシーケンスを含んでよい。さらに、複数の訪問パスは、複数のプライマリノードから１つ以上の編集ノードへの、訪問エッジの同じ又は同様のシーケンスを含んでよい。例えば、第１プライマリノードから第１編集ノードへの訪問パスは、第２プライマリノードから第２編集ノードへの訪問パスと同じ又は同様の訪問エッジのシーケンスを含んでよい。

【００６６】

文字列の例では、プライマリノードから第１編集ノードへの訪問パスは以下を含んでよい：

【数６】

id.parent().parent().child(3)
id.lhs().decl().update()
id.decl().update()

さらに、プライマリノードから第２編集ノードへの訪問パスは以下を含んでよい：

【数７】

id.parent().parent().child(1)
id.lhs().decl().update()
id.decl().update()

【００６７】

修正パターンモジュール１０８は、編集ノードがプライマリノードから共通訪問パスを含むことに基づき、編集ノードをグループ化してよい。幾つかの実施形態では、編集ノードは、編集ノードが単一のプライマリノードからの共通訪問パスを含むことに基づき、グループ化（例えば、クラスタ化）されてよい。他の実施形態では、編集ノードは、編集ノードが複数のプライマリノードからの共通訪問パスを含むことに基づき、グループ化（例えば、クラスタ化）されてよい。修正パターンモジュール１０８は、異なる編集ノードに関連付けられた訪問パスが共通訪問パスであるか否かを決定してよい。幾つかの実施形態では、訪問パスは、訪問パスが同じ又は同様の訪問エッジ種類のシーケンスを含む場合、共通訪問パスであってよい。さらに、訪問パスは、訪問パスが同じ又は同様の訪問パスの記述を含む場合、共通訪問パスであってよい。代替として、訪問パスは、訪問パスが同じ又は同様の訪問エッジの記述のシーケンスを含む場合、共通訪問パスであってよい。

【００６８】

文字列の例では、共通訪問パスは、以下の訪問エッジの記述を含んでよい：

【数８】

id.lhs().decl().update()
id.decl().update()

【００６９】

幾つかの実施形態では、修正パターンモジュール１０８は、共通訪問パスのうちの１つを、望ましい訪問パスとして分類してよい。幾つかの実施形態では、望ましい訪問パスは、プライマリノードから編集ノードへの最も少ない数の訪問エッジ（例えば、最小ホップ数）を含む共通訪問パスに基づき決定されてよい。他の実施形態では、望ましい訪問パスは、他の共通訪問パスに比べて低く重み付けされたパス長を含む共通訪問パスに基づき決定されてよい。上述の及び他の実施形態では、共通訪問パスの中の各エッジは、３、２、１、又は任意の他の適切な値のような、重み付け値を含んでよい。例えば、第１ノードと第２エッジとの間のlhs()のプログラム上の関係に関連付けられた第１エッジは、３の重み付け値を有してよく、第２ノードと第３ノードとの間のupdate()のプログラム上の関係に関連付けられた第２エッジは、２の重み付け値を有してよい。共通訪問パスの重み付けされたパス長は、共通訪問パスの中のエッジに関連付けられた重み付け値の和であってよい。例えば、第２ノードをトラバースする、第１ノードと第２ノードとの間の共通訪問パスは、５の重み付けパス長を有してよい（例えば、３（第１エッジの重み付け値）＋２（第２エッジの重み付け値）＝５）。

【００７０】

２つ以上の編集ノードをグループ化することの利点は、修正パターン１１０を格納するためのメモリの使用量を削減することであってよい。別の利点は、修正パターン１１０が、グループ化（例えば、クラスタ化）を伴わない修正パターン１１０より、一般化された

修復パターンとして生成できることであってよい。

【００７１】

文字列の例では、望ましい訪問パスは次の通りであってよい：

【数９】

$$\text{id.decl().update()}$$

【００７２】

修正パターン１１０は、プライマリノードに対する、欠陥のあるソフトウェアプログラム１０４のDSLに対する変更の位置を指定してよい。幾つかの実施形態では、修正パターン１１０は、欠陥のあるソフトウェアプログラム１０４のDSLに適合する形式（例えば、ソースコード及び／又はコンピュータ言語）で生成されてよい。例えば、コアDSL内の修正パターン１１０は、次の通りであってよい：

【数１０】

$$\begin{gathered}
\text{fixpattern} ::= \text{FixPattern}(\text{name}, \text{rule}_1, \text{rule}_2, \ldots, \text{rule}_n) \\
\textit{rule} ::= \text{ApplyFixes}(\textit{primaryNode}, \textit{fix}_1, \textit{fix}_2, \ldots, \textit{fix}_n) \\
\textit{primaryNode} ::= \text{MatchCtx}(\textit{ctx}, \text{SAnalyzerReport}()) \\
\text{ctx} ::= \text{ctxpath}(\text{attrs}) \mid \text{ctx}_1 \wedge \text{ctx}_2 \\
\textit{ctxpath} ::= \textit{ctxpath}_1 \,.\, \textit{ctxpath}_2 \mid \text{id} \mid \text{child(i)} \mid \text{parent()} \mid \text{child(type, i)} \mid \ldots \\
\textit{attrs} ::= \text{true} \mid \{\text{key:value}\} \cup \textit{attrs} \\
\text{fix} ::= \text{Map}(\lambda\, \text{L} \rightarrow \text{Edit(L, operation)}, \text{editLocations}) \\
\text{editLocations} ::= \text{Visit}(\text{primaryNode}, \text{visitpattern}_1, \ldots, \text{visitpattern}_n) \\
\text{visitpattern} ::= \text{visitpath} \mid \text{visitpath} \oplus \text{ctx} \\
\textit{visitpath} ::= \textit{visitpath}_1 \,.\, \textit{visitpath}_2 \mid \text{id} \mid \text{child(i)} \mid \text{parent()} \mid \text{decl()} \mid \text{use()} \mid \ldots \\
\textit{operation} ::= \text{Insert(x, } \textit{ast}\text{, k)} \mid \text{Delete(x, } \textit{ref}\text{)} \mid \text{Update(x, } \textit{ast}\text{)} \\
\text{ast} ::= \text{const} \mid \text{ref} \\
\textit{const} ::= \text{ConstNode(kind, value, } \textit{ast}_1, \ldots, \textit{ast}_n\text{)} \\
\text{ref} ::= \text{Visit(n, visitpattern)}
\end{gathered}$$

【００７３】

幾つかの実施形態では、修正パターンモジュール１０８は、１つ以上の訪問パス及び編集操作に基づき、修正パターン１１０を生成してよい。例えば、欠陥ASTが、グループ化されていない４個の欠陥ノードを含み、そのうちの１つがプライマリノードである場合、修正パターン１１０は、４個の訪問パス（例えば、プライマリノードから他の３個の欠陥ノードの各々への望ましい訪問パス、及びプライマリノードへの訪問パス）及び編集操作に基づいてよい。別の例では、欠陥ASTが、４個の欠陥ノードを含み、そのうちの２つがグループ化され、グループ化されたノードのうちの１つがプライマリノードである場合、修正パターン１１０は、３個の訪問パス（例えば、プライマリノードから２つのグループ化されていない欠陥ノードへの望ましい訪問パス、及びプライマリノードへの訪問パス）及び編集操作に基づいてよい。幾つかの実施形態では、修正パターン１１０は、複数の編集操作に基づいてよい。

【００７４】

文字列の例では、コアDSL内の修正パターン１１０の一部は次の通りであってよい：

【数１１】

FixPattern(“SBSC_USE_STRINGBUFFER_CONCATENATION”, rule) where
rule = ApplyFixes(pNode, fix1, fix2, fix3)
pNode = MatchContext(id(nodeType:AssignStmt), SAnalyzerReport())
eLoc1 = Visit(n, id.decl().update())
eLoc2 = Visit(n, id.decl())
eLoc3 = Visit(n, id.decl().use())
fix1 = Map(λL → Edit(L, op1), eLoc1)
op1 = Update(L, ConstNode(MethodInvocation, L.lhs, “append”, L.rhs))
fix2 = Map(λL → Edit(L, op2), eLoc2)
op2 = Update(L, ConstNode(VarDecl, “StringBuffer”, L.var, ConstNode(Constructor, Type(“StringBuffer”), L.init)))
fix3 = Map(λL → Edit(L, op3), eLoc3)
op3 = Update(n, ConstNode(MethodInvocation, n.name, “toString”, null))

【００７５】
修復モジュール１０９は、修正パターン１１０を受信してよい。さらに、修復モジュール１０９は、被検査コード１１１を受信してよい。幾つかの実施形態では、修復モジュール１０９は、被検査コード１１１の静的分析を実行するよう構成されてよい。幾つかの実施形態では、修復モジュール１０９は、被検査コード１１１内の欠陥が、欠陥のあるソフトウェアプログラム１０４内の欠陥と同じ又は同様であるか否かを決定してよい。被検査コード１１１内の欠陥は、欠陥が、同じ又は同様の欠陥種類、欠陥位置、ノード名、キー及び対応する値のペア（例えば、キーが「種類」であり、対応する値が「String（文字列型）」である）のような他のノード属性、ノード種類、及びノードラベルを含む場合、欠陥のあるソフトウェアプログラム１０４内の欠陥と同じ又は同様であってよい。
【００７６】
被検査コード１１１内の欠陥が欠陥のあるソフトウェアプログラム１０４内の欠陥と同じ又は同様であることに応答して、修復モジュール１０９は、修正パターン１１０を用いて被検査コード１１１を変更して、変更被検査コード１１３を生成してよい。例えば、修復モジュール１０９は、欠陥のあるソフトウェアに対して行われたものと同じ又は同様の変更を被検査コード１１１に対して行って、改善されたソフトウェアプログラム１０６を得てよい。したがって、変更被検査コード１１３は、被検査コード１１１の改善されたバージョンであってよい。
【００７７】
本開示の範囲から逸脱することなく図１に対し変更、追加又は省略が行われてよい。例えば、環境１００は、本開示で示され説明されたものより多くの又は少ない要素を有してよい。さらに、幾つかの実施形態では、１つ以上のルーチン、１つ以上の命令、又は修復モジュール１０８、修正パターン１１０、及び修復モジュール１０９の少なくとも一部のコードは、結合され又は分離されてよい。幾つかの実施形態では、動作は、上述のものと異なる順序で実行されてよい。さらに、ビッグコード１０２は、単一の欠陥のあるソフトウェアプログラム１０４及び単一の改善されたソフトウェアプログラム１０６を含むとして示されるが、ビッグコード１０２は、任意の数の欠陥のあるソフトウェアプログラム１０４及び改善されたソフトウェアプログラム１０６、例えば２個の欠陥のあるソフトウェアプログラム１０４及び改善されたソフトウェアプログラム１０６、５個の欠陥のあるソフトウェアプログラム１０４及び改善されたソフトウェアプログラム１０６、又は１０００個の欠陥のあるソフトウェアプログラム１０４及び改善されたソフトウェアプログラム１０６、を含んでよい。
【００７８】
図２は、ソフトウェアプログラムの修復操作を実行するために使用され得る修正パター

ンを生成するよう構成され得る例示的なコンピューティングシステム２００を示す。コンピューティングシステム２００は、例えば図１の環境１００において実装されてよい。コンピューティングシステム２００は、１つ以上のプロセッサ２１４、メモリ２１６、及び修正パターンモジュール２０８と修復モジュール２０９とを含むデータ記憶装置２１２を含んでよい。修正パターンモジュール２０８は、図１の修正パターンモジュール１０８と同じ又は同様であってよい。修復モジュール２０９は、図１の修復モジュール１０９と同じ又は同様であってよい。

【００７９】

プロセッサ２１４は、任意の適切な特定用途向け又は汎用コンピュータ、コンピューティングエンティティ、又は種々のコンピュータハードウェア若しくはソフトウェアモジュールを有してよく、任意の適切なコンピュータ可読記録媒体に格納された命令を実行するよう構成されてよい。例えば、プロセッサ２１４は、マイクロプロセッサ、マイクロコントローラ、デジタシグナルプロセッサ（DSP）、ASIC、FPGA又はプログラム命令を解釈し及び／若しくは実行し並びに／又はデータを処理するよう構成された任意の他のデジタル若しくはアナログ回路を有してよい。

【００８０】

図２には単一のプロセッサを示したが、プロセッサ２１４は、より一般的には、本開示で説明される任意の数の工程を個々に又は共同で実行するよう構成される任意の数のプロセッサを有してよい。さらに、プロセッサ２１４のうちの１つ以上は、１つ以上の異なる電子装置又はコンピューティングシステムに存在してよい。幾つかの実施形態では、プロセッサ２１４は、プログラム命令を解釈し及び／又は実行し、及び／又はメモリ２１６、データ記憶装置２１２又はメモリ２１６及びデータ記憶装置２１２に格納されたデータを処理してよい。幾つかの実施形態では、プロセッサ２１４は、データ記憶装置２１２からプログラム命令をフェッチし、該プログラム命令をメモリ２１６にロードしてもよい。プログラム命令がメモリ２１６にロードされた後、プロセッサ２１４は該プログラム命令を実行してよい。

【００８１】

メモリ２１６及びデータ記憶装置２１２は、コンピュータ実行可能命令又はデータ構造を伝える又は格納しているコンピュータ可読記憶媒体を含み得る。このようなコンピュータ可読媒体は、プロセッサ２１４のような汎用又は特定目的コンピュータによりアクセスできる任意の利用可能な媒体を含み得る。例として且つ限定ではなく、このようなコンピュータ可読記憶媒体は、RAM、ROM、EEPROM、CD－ROM又は他の光ディスク記憶装置、磁気ディスク記憶装置又は他の磁気記憶装置、フラッシュメモリ装置（例えば、固体メモリ素子）を含む有形又は非一時的コンピュータ可読記憶媒体、又はコンピュータにより実行可能な命令若しくはデータ構造の形式でプログラムコード手段を伝える若しくは格納するために用いられ汎用若しくは特定目的コンピュータによりアクセス可能な他の非一時的記憶媒体を有し得る。上述の組合せも、コンピュータ可読記憶媒体の範囲に包含され得る。コンピュータ実行可能命令は、例えば、プロセッサ２１４に特定の工程又は工程のグループを実行させるよう構成される命令及びデータを含み得る。

【００８２】

修正パターンモジュール２０８及び／又は修復モジュール２０９は、データ記憶装置２１２に格納されたプログラム命令を含んでよい。プロセッサ２１４は、修正パターンモジュール２０８及び／又は修復モジュール２０９をメモリ２１６にロードし、ロードした修正パターンモジュール２０８及び／又は修復モジュール２０９を実行するよう構成されてよい。代替で、プロセッサ２１４は、メモリ２１６にロードしないで、データ記憶装置２１２から修正パターンモジュール２０８及び／又は修復モジュール２０９を１行毎に実行してよい。修正パターンモジュール２０８及び／又は修復モジュール２０９を実行すると、プロセッサ２１４は、本開示の他の場所に記載された修正パターンを生成するよう構成されてよい。

【００８３】

本開示の範囲から逸脱することなくコンピューティングシステム２００に対し変更、追加又は省略が行われてよい。例えば、幾つかの実施形態では、コンピューティングシステム２００は、明示的に示され又は記載されてない任意の数の他のコンポーネントを有してよい。本願明細書に記載した実施形態は、以下に更に詳細に議論するように、種々のコンピュータハードウェア又はソフトウェアモジュールを備えた特定用途又は汎用コンピュータの使用を含み得る。

【００８４】

図３は、ソフトウェアプログラムを修復する例示的な方法３００のフローチャートを示す。方法３００は、欠陥のあるソフトウェアプログラム及び改善されたソフトウェアプログラム（例えば、図１の欠陥のあるソフトウェアプログラム１０４及び改善されたソフトウェアプログラム１０６）に関して、任意の適切なシステム、機器、又は装置により実行されてよい。例えば、図１の修正パターンモジュール１０８及び／又は修復モジュール又は（例えば修正パターンモジュール及び／又は修復モジュールにより指示されるとき）図２のコンピューティングシステム２００は、欠陥のあるソフトウェアプログラム及び改善されたソフトウェアプログラムに関して方法３００に関連する工程のうちの１つ以上を実行し又は実行を指示してよい。別個のブロックとして示したが、特定の実装に依存して、方法３００のブロックのうちの１又は複数に関連するステップ及び工程は、更なるブロックに分割され、少ないブロックに結合され、又は除去されてよい。

【００８５】

方法３００は、ブロック３０２を含んでよく、ここで、欠陥のあるソフトウェアプログラムの欠陥に関する編集操作が識別されてよい。幾つかの実施形態では、編集操作は、欠陥のあるソフトウェアプログラムと、欠陥の修復として編集操作を含む改善されたソフトウェアプログラムとの間の１つ以上の相違に基づいて識別されてよい。編集操作は、欠陥のあるソフトウェアプログラムに対する、１つ以上の欠陥位置における、それぞれの欠陥を修復するための１つ以上の編集を含んでよい。その結果、改善されたソフトウェアプログラムが取得され得る。幾つかの実施形態では、欠陥のあるソフトウェアプログラム及び改善されたソフトウェアプログラムは、図１のビッグコード１０２のようなビッグコードから取得されてよい。

【００８６】

ブロック３０４で、欠陥のあるソフトウェアプログラムを表す欠陥ASTが取得されてよい。幾つかの実施形態では、欠陥ASTは、欠陥を含む欠陥のあるソフトウェアプログラムの欠陥位置に対応する複数の欠陥ノードを含んでよい。上述の及び他の実施形態では、欠陥ノードは、編集操作により変更される編集ノードを含んでよい。幾つかの実施形態では、欠陥ASTは、ビッグコード（例えば、図１のビッグコード）のような別個のソースから取得されてよい。他の実施形態では、欠陥ASTはローカルに生成されてよい。

【００８７】

ブロック３０６で、複数の欠陥ノードのうちの特定ノードは、プライマリノードとして分類されてよい。プライマリノードは、編集操作を実施する際の開始点として動作してよい。幾つかの実施形態では、特定ノードは、特定ノードが１つ以上の共通欠陥属性を有することに基づき、プライマリノードとして分類されてよい。幾つかの実施形態では、特定ノードは、以下に詳細に記載される図５の方法５００の１つ以上の動作に基づき、プライマリノードとして分類されてよい。

【００８８】

ブロック３０８で、プライマリノードから編集ノードへの訪問パスが識別されてよい。幾つかの実施形態では、訪問パスは、欠陥ASTに対応する訪問ASTに含まれてよい。上述の及び他の実施形態では、訪問パスは、プライマリノードと編集ノードとの間のプログラム上の関係を示す１つ以上の訪問エッジのシーケンスを含んでよい。訪問ASTは、図１に関連して上述したように生成されてよい。

【００８９】

ブロック３１０で、修正パターンが生成されてよい。幾つかの実施形態では、修正パタ

ーンは、欠陥のあるソフトウェアプログラムのソースコードに適合する形式で生成されてよい。例えば、修正パターンは、欠陥のあるソフトウェアプログラムに適合するDSL内で生成されてよい。

【００９０】

ブロック３１２で、修復操作は、被分析ソフトウェアプログラムの被検査コードの識別された欠陥に関して実行されてよい。幾つかの実施形態では、修復操作は、修正パターンを用いて実行されてもよい。上述の及び他の実施形態では、修復操作は、被検査コードの識別された欠陥が欠陥のあるソフトウェアプログラムの欠陥と同じ種類であることに基づき、実行されてよい。

【００９１】

本開示の範囲から逸脱することなく方法３００に対し変更、追加又は省略が行われてよい。例えば、方法３００の工程は、異なる順序で実施されてよい。追加又は代替として、２以上の工程が同時に実行されてよい。さらに、概略のステップ及び動作は、単に例として提供され、幾つかのステップ及び動作は、開示の実施形態の本質から逸脱することなく、任意であり、より少ないステップ及び動作に組み合わされ、又は追加ステップ及び動作に拡張されてよい。

【００９２】

さらに、幾つかの実施形態では、方法３００は反復的に実行されて良く、ここで、１つ以上の工程が単一の障害位置に対して同時に実行されてよい。追加又は代替として、１つ以上のブロックに関連する１つ以上の工程は、複数の欠陥位置に関して一度に実行されてよい。

【００９３】

図４は、ソフトウェアプログラムの修復操作を実行するために使用され得る修正パターンを生成する例示的な方法４００のフローチャートを示す。方法４００は、欠陥のあるソフトウェアプログラム及び改善されたソフトウェアプログラム（例えば、図１の欠陥のあるソフトウェアプログラム１０４及び改善されたソフトウェアプログラム１０６）に関して、任意の適切なシステム、機器、又は装置により実行されてよい。例えば、図１の修正パターンモジュール１０８、修復モジュール１０９、又は（例えば修正パターンモジュール及び／又は修復モジュールにより指示されるとき）図２のコンピューティングシステム２００は、欠陥のあるソフトウェアプログラム及び改善されたソフトウェアプログラムに関して方法４００に関連する工程のうちの１つ以上を実行し又は実行を指示してよい。別個のブロックとして示したが、特定の実装に依存して、方法４００のブロックのうちの１又は複数に関連するステップ及び工程は、更なるブロックに分割され、少ないブロックに結合され、又は除去されてよい。さらに、方法３００の１つ以上の工程は、幾つかの実施形態において方法４００の一部として実行されてよい。

【００９４】

方法４００は、ブロック４０２で開始してよく、ここで、複数の欠陥のあるソフトウェアプログラム及び複数の改善されたソフトウェアプログラムが取得されてよい。欠陥のあるソフトウェアプログラム及び複数の改善されたソフトウェアプログラムは、図１のビッグコードのようなビッグコードから取得されてよい。

【００９５】

ブロック４０４で、１つ以上欠陥ASTが生成されてよい。幾つかの実施形態では、欠陥ASTは、欠陥のあるソフトウェアプログラムの各々について生成されてよい。幾つかの実施形態では、欠陥ASTは、１つ以上の欠陥を含む欠陥のあるソフトウェアプログラムの欠陥位置に対応する複数の欠陥ノードを含んでよい。上述の及び他の実施形態では、欠陥ノードは、編集操作により変更される編集ノードを含んでよい。

【００９６】

ブロック４０６で、編集操作は、欠陥のあるソフトウェアプログラム内の各欠陥について決定されてよい。幾つかの実施形態では、編集操作は、欠陥のあるソフトウェアプログラムと改善されたソフトウェアプログラムとの間の１つ以上の相違に基づき識別されてよ

い。編集操作は、欠陥のあるソフトウェアプログラムに対する、欠陥位置における、それぞれの欠陥を修復するための１つ以上の編集を含んでよい。その結果、改善されたソフトウェアプログラムが取得され得る。

【００９７】

幾つかの実施形態では、欠陥のあるソフトウェアプログラムの第２欠陥に関する第２編集操作が識別されてよい。幾つかの実施形態では、第２編集操作は、欠陥のあるソフトウェアプログラムと、第２欠陥の修復として第２編集操作を含む改善されたソフトウェアプログラムとの間の１つ以上の相違に基づいて識別されてもよい。第２編集操作は、欠陥のあるソフトウェアプログラムに対する、１つ以上の欠陥位置における、それぞれの第２欠陥を修復するための１つ以上の編集を含んでよい。その結果、改善されたソフトウェアプログラムが取得され得る。

【００９８】

ブロック４０８で、１つ以上のプライマリノードが各欠陥ASTの中で識別されてよい。プライマリノードは、編集操作を実施する際の開始点として動作してよい。幾つかの実施形態では、ノードは、ノードが１つ以上の共通欠陥属性を有することに基づき、プライマリノードとして分類されてよい。幾つかの実施形態では、ノードのうちの１つ以上は、以下に詳細に記載される図５の方法５００の１つ以上の動作に基づき、プライマリノードとして分類されてよい。

【００９９】

例えば、複数の欠陥ノードのうちの特定ノードは、第１プライマリノードとして分類されてよい。第１プライマリノードは、第１編集操作を実施する際の開始点として動作してよい。幾つかの実施形態では、特定ノードは、特定ノードが１つ以上の共通欠陥属性を有することに基づき、第１プライマリノードとして分類されてよい。上述の及び他の実施形態では、特定ノードは、図５の方法５００の１つ以上の動作に基づき、第１プライマリノードとして分類されてよい。

【０１００】

別の例として、複数の欠陥ノードのうちの別の特定ノードは、第２プライマリノードとして分類されてよい。第２プライマリノードは、第２編集操作を実施する際の開始点として動作してよい。幾つかの実施形態では、別の特定ノードは、別の特定ノードが１つ以上の共通欠陥属性を有することに基づき、第２プライマリノードとして分類されてよい。幾つかの実施形態では、別の特定ノードは、図５の方法５００の１つ以上の動作に基づき、第２プライマリノードとして分類されてよい。

【０１０１】

ブロック４１０で、訪問ASTが各欠陥ASTについて生成されてよい。幾つかの実施形態では、訪問ASTは、欠陥ASTに基づいてよい。上述の及び他の実施形態では、訪問ASTは、欠陥のあるソフトウェアプログラムに対応してよい。さらに、訪問ASTは、欠陥のあるソフトウェアプログラムのプログラム構造を表してよい。訪問ASTは、複数の訪問エッジを含んでよい。訪問エッジは、訪問AST内のノード間のプログラム上の関係を表してよい。訪問ASTは、図１に関連して上述したように生成されてよい。

【０１０２】

ブロック４１２で、各欠陥AST内の１つ以上の欠陥ノードがクラスタ化されてよい。幾つかの実施形態では、欠陥ノードは、欠陥ノードが編集ノードとして識別され及び訪問ASTの中のプライマリノードからの共通訪問パスを含むことに基づき、クラスタ化（例えば、グループ化）されてよい。上述の及び他の実施形態では、欠陥ノードは、欠陥ノードが訪問ASTの中の単一のプライマリノードからの共通訪問パスを含むことに基づき、クラスタ化されてよい。他の実施形態では、欠陥ノードは、欠陥ノードが訪問ASTの中の複数のプライマリノードからの共通訪問パスを含むことに基づき、クラスタ化されてよい。

【０１０３】

ブロック４１４で、各プライマリノードから各編集ノードへの１つ以上の訪問パスが識別されてよい。幾つかの実施形態では、訪問パスは、各訪問ASTについて識別されてよい

。上述の及び他の実施形態では、訪問パスは、プライマリノードと編集ノードとの間のプログラム上の関係を示す１つ以上の訪問エッジのシーケンスを含んでよい。

【０１０４】

例えば、第１プライマリノードから第１編集ノードへの第１訪問パスが識別されてよい。幾つかの実施形態では、第１訪問パスは、訪問ASTに含まれてよい。幾つかの実施形態では、第１訪問パスは、第１プライマリノードと第１編集ノードとの間のプログラム上の関係を示す１つ以上の訪問エッジのシーケンスを含んでよい。

【０１０５】

別の例として、第２プライマリノードから第２編集ノードへの第２訪問パスが識別されてよい。幾つかの実施形態では、第２訪問パスも、訪問ASTに含まれてよい。上述の及び他の実施形態では、第２訪問パスは、第２プライマリノードと第２編集ノードとの間のプログラム上の関係を示す１つ以上の訪問エッジのシーケンスを含んでよい。

【０１０６】

ブロック４１６で、各プライマリノードから各編集ノードへの望ましい訪問パスが識別されてよい。幾つかの実施形態では、望ましい訪問パスは、訪問AST内の各プライマリノード及び各編集ノードについて識別されてよい。上述の及び他の実施形態では、望ましい訪問パスは、共通訪問パスに基づき決定されてよい。幾つかの実施形態では、訪問パスは、訪問パスが同じ又は同様の訪問エッジ種類のシーケンスを含む場合、共通訪問パスであってよい。上述の及び他の実施形態では、訪問パスは、訪問パスが、プライマリノードから編集ノードへの最も少ない数の訪問エッジ（例えば、最小ホップ数）を含む場合に、望ましい訪問パスであってよい。

【０１０７】

ブロック４１８で、修正パターンは、それぞれの欠陥のあるソフトウェアプログラムのDSL内で生成されてよい。幾つかの実施形態では、修正パターンは、欠陥のあるソフトウェアプログラムのソースコードに適合する形式で生成されてよい。

【０１０８】

本開示の範囲から逸脱することなく方法４００に対し変更、追加又は省略が行われてよい。例えば、方法４００の工程は、異なる順序で実施されてよい。追加又は代替として、２以上の工程が同時に実行されてよい。さらに、概略のステップ及び動作は、単に例として提供され、幾つかのステップ及び動作は、開示の実施形態の本質から逸脱することなく、任意であり、より少ないステップ及び動作に組み合わされ、又は追加ステップ及び動作に拡張されてよい。

【０１０９】

さらに、幾つかの実施形態では、方法４００は反復的に実行されてよく、ここで、一度に、単一の欠陥のあるソフトウェアプログラム及び単一の改善されたソフトウェアプログラムが分析されてよい。

【０１１０】

図５は、特定ノードをプライマリノードとして分類する例示的な方法５００のフローチャートを示す。方法５００は、検査用の欠陥のあるソフトウェアプログラム及び欠陥のあるソフトウェアプログラム（例えば、図１の検査用の欠陥のあるソフトウェアプログラム１０７及び欠陥のあるソフトウェアプログラム１０４）に関して、任意の適切なシステム、機器、又は装置により実行されてよい。例えば、図１の修正パターンモジュール１０８又は（例えば修正パターンモジュールにより指示されるとき）図２のコンピューティングシステム２００は、特定ノードをプライマリノードとして分類するステップに関して、方法５００に関連する工程のうちの１つ以上を実行し又は実行を指示してよい。別個のブロックとして示したが、特定の実装に依存して、方法５００のブロックのうちの１又は複数に関連するステップ及び工程は、更なるブロックに分割され、少ないブロックに結合され、又は除去されてよい。上述のように、幾つかの例では、方法５００の工程のうちの１つ以上は、方法３００のブロック３０６及び方法４００のブロック４０８に関して実行されてよい。

【０１１１】
方法５００は、ブロック５０２で開始してよく、ここで、欠陥のあるソフトウェアプログラムの欠陥と同様の１つ以上の欠陥を含む複数の検査用の欠陥のあるソフトウェアプログラムが取得されてよい。検査用の欠陥のあるソフトウェアプログラムは、図１のビッグコードのようなビッグコードから取得されてよい。
【０１１２】
ブロック５０４で、検査用の欠陥のあるソフトウェアプログラムの中の欠陥のうちの少なくとも１つの１つ以上の共通欠陥属性が識別されてよい。幾つかの実施形態では、欠陥属性が検査用の欠陥のあるソフトウェアプログラム内の欠陥の各々に関連付けられている場合、欠陥属性は共通欠陥属性として分類されてよい。他の実施形態では、欠陥属性が検査用の欠陥のあるソフトウェアプログラム内の欠陥の少なくとも一部に関連付けられている場合、欠陥属性は共通欠陥属性として分類されてよい。さらに、欠陥属性が検査用の欠陥のあるソフトウェアプログラム内の欠陥の少なくとも大部分に関連付けられている場合、欠陥属性は共通欠陥属性として分類されてよい。幾つかの実施形態では、図１に関連して上述したように、欠陥属性が検査用の欠陥のあるソフトウェアプログラム内の欠陥の少なくとも閾百分率に関連付けられている場合、欠陥属性は共通欠陥属性として分類されてよい。
【０１１３】
ブロック５０６で、特定ノードはプライマリノードとして分類されてよい。幾つかの実施形態では、図１に関連して上述したように、プライマリノードは、編集操作を実施する際の開始点として動作してよい。幾つかの実施形態では、特定ノードは、特定ノードが共通欠陥属性のうちの１つ以上を有することに基づき、プライマリノードとして分類されてよい。
【０１１４】
幾つかの実施形態では、特定ノードは、特定ノードが共通欠陥属性のうちのいずれかに関連付けられていることに基づき、プライマリノードとして分類されてよい。他の実施形態では、特定ノードは、特定ノードが共通欠陥属性のうちの全部に関連付けられていることに基づき、プライマリノードとして分類されてよい。さらに、図１に関連して上述したように、特定ノードは、特定ノードが共通欠陥属性のうちのプライマリ閾百分率に関連付けられていることに基づき、プライマリノードとして分類されてよい。代替として、特定ノードは、特定ノードが共通欠陥属性のうちの少なくとも１つに関連付けられていることに基づき、プライマリノードとして分類されてよい。
【０１１５】
本開示の範囲から逸脱することなく方法５００に対し変更、追加又は省略が行われてよい。例えば、方法５００の工程は、異なる順序で実施されてよい。追加又は代替として、２以上の工程が同時に実行されてよい。さらに、概略のステップ及び動作は、単に例として提供され、幾つかのステップ及び動作は、開示の実施形態の本質から逸脱することなく、任意であり、より少ないステップ及び動作に組み合わされ、又は追加ステップ及び動作に拡張されてよい。
【０１１６】
図６は、図１の環境で実施され得る例示的な欠陥AST６２２及び例示的な改善AST６２４を示す。欠陥AST６２２及び改善AST６２４は、本開示で議論された文字列の例に対応してよい。欠陥AST６２２及び改善AST６２４は、それぞれ複数のノードを含んでよい。図６では、単一のノード６３８は、例示的なノードを示すために６３８として示される。欠陥AST６２２の中のノードの制御フローは、ASTエッジとして表され得る。図６では、単一のASTエッジは、例示的なASTエッジを示すために６４０として示される。ASTエッジは、文字列のソフトウェアルーチンの制御フロー内のジャンプを表してよい。
【０１１７】
欠陥AST６２２内のノードは、図１と関連して上述したように編集操作に関連して変更されるべき１つ以上の編集ノードを含んでよい。図６では、３個の編集ノードが、複数の

例示的な編集ノードを示すために６２８ａ、６２８ｂ、及び６２８ｃとして示される。さらに、欠陥AST６２２内のノードは、プライマリノード６４２を含んでよい。プライマリノード６４２は、図１と関連して上述したように、改善AST６２４及び改善されたソフトウェアプログラムを取得するために編集操作を実施するための開始点として動作してよい。幾つかの実施形態では、プライマリノード６４２は、編集ノードとして分類されてもよい。

【０１１８】

幾つかの実施形態では、図１と関連して上述したように、プライマリノード６４２及び編集ノード６２８ｂは、プライマリノード及び編集ノード６２８ｂが共通訪問パスを含むことに基づき、グループ化（例えば、クラスタ化）されてよい。図６では、６０６のように示される楕円は、プライマリノード及び編集ノード６２８ｂのグループ化を表してよい。

【０１１９】

上述のように、編集操作は、文字列のソフトウェアルーチンに対する４個の編集を含んでよい。第１編集は、以下の例示的なDSLに対応してよい：

【数１２】

$$op1 = Update(L, ConstNode(VarDecl, \text{“StringBuffer”}, L.var, ConstNode(Constructor, Type(\text{“StringBuffer”}), L.init)))$$

A second edit may correspond to the following example DSL:

$$op2 = Update(L, ConstNode(MethodInvocation, L.lhs, \text{“append”}, L.rhs))$$

A third edit may correspond to the following example DSL:

$$op3 = Update(L, ConstNode(MethodInvocation, L.lhs, \text{“append”}, L.rhs))$$

A fourth edit may correspond to the following example DSL:

$$op4 = Update(n, ConstNode(MethodInvocation, n.name, \text{“toString”}, null))$$

【０１２０】

図６では、破線矢印は、様々な編集を表してよい。図６では、第１編集は破線矢印６３２により表されてよく、第２編集は破線矢印６３４により表されてよく、第３編集は破線矢印６３６により表されてよく、第４編集は破線矢印６３４により表されてよい。

【０１２１】

上述のように、本開示で記載した実施形態は、以下に更に詳細に議論するように、種々のコンピュータハードウェア又はソフトウェアモジュールを備えた特定用途又は汎用コンピュータ（例えば、図２のプロセッサ２１４）の使用を含み得る。さらに、上述のように、本開示に記載の実施形態は、コンピュータ実行可能命令又はデータ構造を伝える又はそれを格納されたコンピュータ可読媒体（例えば、図２のメモリ２５２）を用いて実施されてもよい。

【０１２２】

本開示で用いられるように、用語「モジュール」又は「コンポーネント」は、モジュール若しくはコンポーネントのアクションを実行するよう構成される特定ハードウェア実装、及び／又はコンピューティングシステムの汎用ハードウェア（例えばコンピュータ可読媒体、処理装置、等）に格納され及び／又はそれらにより実行され得るソフトウェアオブジェクト又はソフトウェアルーチンを表してよい。幾つかの実施形態では、本開示に記載されたのと異なるコンポーネント、モジュール、エンジン及びサービスは、（例えば、別

個のスレッドとして）コンピューティングシステムで実行されるオブジェクト又は処理として実施されてよい。本開示に記載のシステム及び方法の幾つかは概して（汎用ハードウェアに格納される及び／又はそれにより実行される）ソフトウェアで実装されるように記載されたが、専用ハードウェアの実装又はソフトウェアと専用ハードウェアの組み合わせの実装も可能であり考えられる。この説明では、「コンピュータエンティティ」は、本開示で先に定められたようにコンピューティングシステム、又はコンピューティングシステムで実行されるモジュール若しくはモジュールの組合せであってよい。

【０１２３】

本開示で及び特に添付の特許請求の範囲（例えば、添付の特許請求の範囲の本体）で使用される用語は、概して、広義の（open）用語と考えられる（例えば、用語「含む（including）」は「含むが、限定されない」と解釈されるべきであり、用語「有する（having）」は「少なくとも有する」と解釈されるべきであり、用語「含む（includes）」は「含むが、限定されない」と解釈されるべきである）。

【０１２４】

さらに、特定数の導入された請求項の引用が意図される場合、このような意図は、請求項の中に明示的に示され、このような引用が存在しない場合はこのような意図が存在しない。例えば、理解の助けとして、以下の添付の特許請求の範囲は、請求項の引用を導入するために、「少なくとも１つの」及び「１又は複数の」をいう前置語句の使用を含み得る。しかしながら、このような語句の使用は、同じ請求項が前置語句「１又は複数」又は「少なくとも１つの」及び「ａ又はａｎ」のような不定冠詞を含むときでも、不定冠詞「ａ、ａｎ」による請求項引用の導入がこのような導入された請求項引用を含む任意の特定の請求項をこのような引用を１つだけ含む実施形態に限定することを示すと考えられてはならない（例えば、「ａ」及び／又は「ａｎ」は「少なくとも１つの」又は「１又は複数の」を意味すると解釈されるべきである）。同様のことは、請求項引用を導入するために使用される定冠詞の使用についても該当する。

【０１２５】

さらに、特定数の導入された請求項引用が明示的に引用される場合、当業者は、このような引用が少なくとも引用された番号を意味することと解釈されるべきであることを認識するだろう（例えば、「２つの引用」はそのままで、他の変更が無ければ、少なくとも２つの引用、又は２以上の引用を意味する）。さらに、「Ａ、Ｂ、Ｃ、等のうちの少なくとも１つ」又は「Ａ、Ｂ、Ｃ、等のうちの１又は複数」に類似する慣例が用いられる例では、通常、このような構成は、Ａのみ、Ｂのみ、Ｃのみ、Ａ及びＢを一緒に、Ａ及びＣを一緒に、Ｂ及びＣを一緒に、又はＡ、Ｂ、Ｃを一緒に、等を含むと意図される。

【０１２６】

さらに、２以上の代替用語を表す任意の離接語又は語句は、説明、請求項、又は図面の中であるかに係わらず、用語のうちの１つ、用語のうちのいずれか、又は両方の用語を含む可能性を包含すると理解されるべきである。例えば、語句「Ａ又はＢ」は、「Ａ」又は「Ｂ」又は「Ａ及びＢ」の可能性を含むと理解されるべきである。

【０１２７】

本開示に記載された全ての例及び条件文は、教育上の目的で、読者が本開示の原理及び発明者により考案された概念を理解するのを助け、技術を促進させるためであり、これらの特に記載された例及び条件に限定されないものと考えられるべきである。本開示の実施形態が詳細に記載されたが、種々の変更、置換及び修正が本開示の精神及び範囲から逸脱することなく行われ得る。

【０１２８】

以上の実施形態に加えて、更に以下の付記を開示する。

（付記１）　欠陥のあるソフトウェアプログラムと欠陥の修復としての編集操作を含む改善されたソフトウェアプログラムとの間の１つ以上の相違に基づき、前記欠陥のあるソフトウェアプログラムの前記欠陥に関する前記編集操作を識別するステップと、

前記欠陥のあるソフトウェアプログラムを表す欠陥抽象構文木（AST）を取得するステ

ップであって、前記欠陥ASTは、前記欠陥を含む前記欠陥のあるソフトウェアプログラムの欠陥位置に対応する複数の欠陥ノードを含み、前記複数の欠陥ノードは前記編集操作により変更される編集ノードを含む、ステップと、
前記複数の欠陥ノードのうちの特定ノードを、前記編集操作を実施する際の開始点として動作するプライマリノードとして分類するステップと、
前記プライマリノードから前記編集ノードへの訪問パスを識別するステップであって、前記訪問パスは前記欠陥ASTに対応する訪問ASTに含まれ、前記訪問パスは前記プライマリノードと前記編集ノードとの間のプログラム上の関係を示す１つ以上の訪問エッジのシーケンスを含む、ステップと、
前記訪問パス及び前記編集操作に基づき修正パターンを生成するステップであって、前記修正パターンは、前記欠陥のあるソフトウェアプログラムのソースコードに適合する形式で生成される、ステップと、
前記修正パターンを用いて、被検査コードの識別された欠陥が前記欠陥のあるソフトウェアプログラムの前記欠陥と同じ種類であることに基づき、被分析ソフトウェアプログラムの前記被検査コードの前記識別された欠陥に関する修復操作を実行するステップと、
を含む方法。
（付記２） 前記編集ノードは、第１編集ノードであり、前記複数の欠陥ノードは前記編集操作により変更された第２編集ノードを含み、前記訪問パスは第１訪問パスであり、前記方法は、さらに、
前記プライマリノードから前記第２編集ノードへの第２訪問パスを識別するステップであって、前記修正パターンは前記第２訪問パスに更に基づく、ステップ、を含む付記１に記載の方法。
（付記３） 前記第１訪問パス及び前記第２訪問パスが同じ種類の訪問エッジの同じシーケンスを有することに基づき、前記第１訪問パス及び前記第２訪問パスを、互いに関して共通訪問パスであると識別するステップを更に含み、
前記修正パターンを生成するステップは、前記第１訪問パス及び前記第２訪問パスが共通訪問パスとして識別されることに基づく、付記２に記載の方法。
（付記４） 前記訪問パスが、前記欠陥又は前記欠陥のあるソフトウェアプログラムの他の欠陥に関連する他の訪問パスと比べて少数の訪問エッジを有することに基づき、前記修正パターンを生成する際に使用するために、前記訪問パスを選択するステップ、を更に含む付記１に記載の方法。
（付記５） 前記欠陥のあるソフトウェアプログラムは、複数の欠陥を含み、前記複数の欠陥ノードのうちの前記特定ノードを前記プライマリノードとして分類する前記ステップは、
前記欠陥のあるソフトウェアプログラムの中の前記複数の欠陥のうちの少なくとも一部の複数の共通欠陥属性を識別するステップであって、前記複数の欠陥ノードのうちの前記特定ノードは、前記特定ノードが前記欠陥のあるソフトウェアプログラムの中の前記複数の欠陥のうちの前記一部の前記複数の共通欠陥属性のうちの１つ以上を含むことに基づき、前記プライマリノードとして分類される、ステップを含む、付記１に記載の方法。
（付記６） 前記欠陥のあるソフトウェアプログラムの前記欠陥は第１欠陥であり、前記編集操作は第１編集操作であり、前記プライマリノードは第１プライマリノードであり、前記訪問パスは第１訪問パスであり、前記編集ノードは第１編集ノードであり、前記複数の欠陥ノードは第２編集ノードを更に含み、前記方法は、さらに、
前記欠陥のあるソフトウェアプログラムと第２欠陥の修復としての第２編集操作を含む前記改善されたソフトウェアプログラムとの間の前記１つ以上の相違に基づき、前記欠陥のあるソフトウェアプログラムの前記第２欠陥に関する前記第２編集操作を識別するステップと、
前記複数の欠陥ノードのうちの別の特定ノードを、前記第２編集操作を実施する際の開始点として動作する第２プライマリノードとして分類するステップと、
前記第２プライマリノードから前記第２編集ノードへの第２訪問パスを識別するステッ

プであって、前記第２訪問パスは、前記訪問ASTに含まれ、前記第２プライマリノードと前記第２編集ノードとの間のプログラム上の関係を示し、前記修正パターンは、前記第２訪問パス及び前記第２編集操作に更に基づく、ステップと、
を含む付記１に記載の方法。
（付記７） 前記第１訪問パス及び前記第２訪問パスが類似する訪問パス記述を含むことに基づき、前記第１訪問パス及び前記第２訪問パスを、互いに関して共通訪問パスであると識別するステップを更に含み、
前記修正パターンを生成するステップは、前記第１訪問パス及び前記第２訪問パスが共通訪問パスとして識別されることに基づく、付記２に記載の方法。
（付記８） 命令を格納するよう構成された１つ以上の非一時的コンピュータ可読記憶媒体であって、前記命令は、実行されることに応答して、システムに動作を実行させ、前記動作は、
欠陥のあるソフトウェアプログラムと欠陥の修復としての編集操作を含む改善されたソフトウェアプログラムとの間の１つ以上の相違に基づき、前記欠陥のあるソフトウェアプログラムの前記欠陥に関する前記編集操作を識別するステップと、
前記欠陥のあるソフトウェアプログラムを表す欠陥ASTを取得するステップであって、前記欠陥ASTは、前記欠陥を含む前記欠陥のあるソフトウェアプログラムの欠陥位置に対応する複数の欠陥ノードを含み、前記複数の欠陥ノードは前記編集操作により変更される編集ノードを含む、ステップと、
前記複数の欠陥ノードのうちの特定ノードを、前記編集操作を実施する際の開始点として動作するプライマリノードとして分類するステップと、
前記プライマリノードから前記編集ノードへの訪問パスを識別するステップであって、前記訪問パスは前記欠陥ASTに対応する訪問ASTに含まれ、前記訪問パスは前記プライマリノードと前記編集ノードとの間のプログラム上の関係を示す１つ以上の訪問エッジのシーケンスを含む、ステップと、
前記訪問パス及び前記編集操作に基づき修正パターンを生成するステップであって、前記修正パターンは、前記欠陥のあるソフトウェアプログラムのソースコードに適合する形式で生成される、ステップと、
前記修正パターンを用いて、被検査コードの識別された欠陥が前記欠陥のあるソフトウェアプログラムの前記欠陥と同じ種類であることに基づき、被分析ソフトウェアプログラムの前記被検査コードの前記識別された欠陥に関する修復操作を実行するステップと、
を含む、１つ以上の非一時的コンピュータ可読記憶媒体。
（付記９） 前記編集ノードは、第１編集ノードであり、前記複数の欠陥ノードは前記編集操作により変更された第２編集ノードを含み、前記訪問パスは第１訪問パスであり、前記動作は、さらに、
前記プライマリノードから前記第２編集ノードへの第２訪問パスを識別するステップであって、前記修正パターンは前記第２訪問パスに更に基づく、ステップ、を含む付記８に記載の１つ以上の非一時的コンピュータ可読記憶媒体。
（付記１０） 前記動作は、前記第１訪問パス及び前記第２訪問パスが同じ種類の訪問エッジの同じシーケンスを有することに基づき、前記第１訪問パス及び前記第２訪問パスを、互いに関して共通訪問パスであると識別するステップを更に含み、
前記修正パターンを生成するステップは、前記第１訪問パス及び前記第２訪問パスが共通訪問パスとして識別されることに基づく、付記９に記載の１つ以上の非一時的コンピュータ可読記憶媒体。
（付記１１） 前記複数の欠陥ノードのうちの前記特定ノードを前記プライマリノードとして分類するステップは、
前記欠陥のあるソフトウェアプログラムの前記欠陥と同様の複数の欠陥を含む複数の検査用の欠陥のあるソフトウェアプログラムを取得するステップと、
前記複数の検査用の欠陥のあるソフトウェアプログラムの中の前記複数の欠陥のうちの少なくとも一部の複数の共通欠陥属性を識別するステップであって、前記複数の欠陥ノー

ドのうちの前記特定ノードは、前記特定ノードが、前記複数の検査用の欠陥のあるソフトウェアプログラムの中の前記複数の欠陥のうちの前記一部の前記複数の共通欠陥属性のうちの１つ以上を含むことに基づき、前記プライマリノードとして分類される、ステップと、

を含む、付記８に記載の１つ以上の非一時的コンピュータ可読記憶媒体。

（付記１２）　前記欠陥のあるソフトウェアプログラムの前記欠陥は第１欠陥であり、前記編集操作は第１編集操作であり、前記プライマリノードは第１プライマリノードであり、前記訪問パスは第１訪問パスであり、前記編集ノードは第１編集ノードであり、前記複数の欠陥ノードは第２編集ノードを更に含み、前記動作は、さらに、

前記欠陥のあるソフトウェアプログラムと第２欠陥の修復としての第２編集操作を含む前記改善されたソフトウェアプログラムとの間の前記１つ以上の相違に基づき、前記欠陥のあるソフトウェアプログラムの前記第２欠陥に関する前記第２編集操作を識別するステップと、

前記複数の欠陥ノードのうちの別の特定ノードを、前記第２編集操作を実施する際の開始点として動作する第２プライマリノードとして分類するステップと、

前記第２プライマリノードから前記第２編集ノードへの第２訪問パスを識別するステップであって、前記第２訪問パスは、前記訪問ASTに含まれ、前記第２プライマリノードと前記第２編集ノードとの間のプログラム上の関係を示し、前記修正パターンは、前記第２訪問パス及び前記第２編集操作に更に基づく、ステップと、

を含む、付記８に記載の１つ以上の非一時的コンピュータ可読記憶媒体。

（付記１３）　前記動作は、前記第１訪問パス及び前記第２訪問パスが類似する訪問パス記述を含むことに基づき、前記第１訪問パス及び前記第２訪問パスを、互いに関して共通訪問パスであると識別するステップを更に含み、

前記修正パターンを生成するステップは、前記第１訪問パス及び前記第２訪問パスが共通訪問パスとして識別されることに基づく、付記１２に記載の１つ以上の非一時的コンピュータ可読記憶媒体。

（付記１４）　システムであって、

命令を格納するよう構成された１つ以上の非一時的コンピュータ可読記憶媒体と、

前記１つ以上のコンピュータ可読記憶媒体に通信可能に結合され、前記命令の実行に応答して、前記システムに動作を実行させるよう構成される１つ以上のプロセッサと、

を含み、前記動作は、

欠陥のあるソフトウェアプログラムと欠陥の修復としての編集操作を含む改善されたソフトウェアプログラムとの間の１つ以上の相違に基づき、前記欠陥のあるソフトウェアプログラムの前記欠陥に関する前記編集操作を識別するステップと、

前記欠陥のあるソフトウェアプログラムを表す欠陥ASTを取得するステップであって、前記欠陥ASTは、前記欠陥を含む前記欠陥のあるソフトウェアプログラムの欠陥位置に対応する複数の欠陥ノードを含み、前記複数の欠陥ノードは前記編集操作により変更される編集ノードを含む、ステップと、

前記複数の欠陥ノードのうちの特定ノードを、前記編集操作を実施する際の開始点として動作するプライマリノードとして分類するステップと、

前記プライマリノードから前記編集ノードへの訪問パスを識別するステップであって、前記訪問パスは前記欠陥ASTに対応する訪問ASTに含まれ、前記訪問パスは前記プライマリノードと前記編集ノードとの間のプログラム上の関係を示す１つ以上の訪問エッジのシーケンスを含む、ステップと、

前記訪問パス及び前記編集操作に基づき修正パターンを生成するステップと、

前記修正パターンを用いて、被検査コードの識別された欠陥が前記欠陥のあるソフトウェアプログラムの前記欠陥と同じ種類であることに基づき、被分析ソフトウェアプログラムの前記被検査コードの前記識別された欠陥に関する修復操作を実行するステップと、

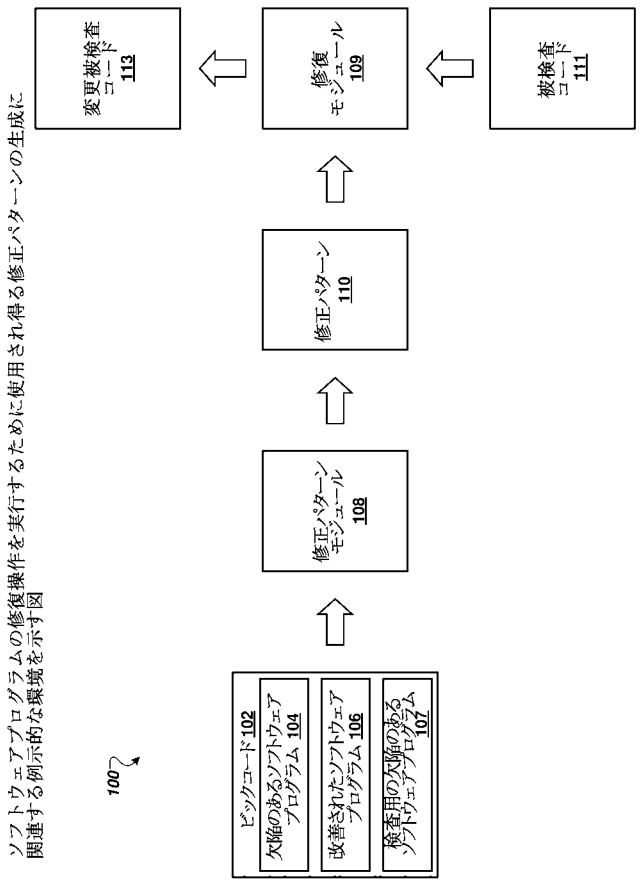
を含む、システム。

（付記１５）　前記編集ノードは、第１編集ノードであり、前記複数の欠陥ノードは前記

編集操作により変更された第２編集ノードを含み、前記訪問パスは第１訪問パスであり、前記動作は、さらに、
前記プライマリノードから前記第２編集ノードへの第２訪問パスを識別するステップであって、前記修正パターンは前記第２訪問パスに更に基づく、ステップ、を含む、付記１４に記載のシステム。
（付記１６）　前記第１訪問パス及び前記第２訪問パスが同じ種類の訪問エッジの同じシーケンスを有することに基づき、前記第１訪問パス及び前記第２訪問パスを、互いに関して共通訪問パスであると識別するステップを更に含み、
前記修正パターンを生成するステップは、前記第１訪問パス及び前記第２訪問パスが共通訪問パスとして識別されることに基づく、付記１５に記載のシステム。
（付記１７）　前記動作は、前記訪問パスが、前記欠陥又は前記欠陥のあるソフトウェアプログラムの他の欠陥に関連する他の訪問パスと比べて少数の訪問エッジを有することに基づき、前記修正パターンを生成する際に使用するために、前記訪問パスを選択するステップ、を更に含む、付記１４に記載のシステム。
（付記１８）　前記複数の欠陥ノードのうちの前記特定ノードを前記プライマリノードとして分類するステップは、
前記欠陥のあるソフトウェアプログラムの前記欠陥と同様の複数の欠陥を含む複数の検査用の欠陥のあるソフトウェアプログラムを取得するステップと、
前記複数の検査用の欠陥のあるソフトウェアプログラムの中の前記複数の欠陥のうちの少なくとも一部の複数の共通欠陥属性を識別するステップであって、前記複数の欠陥ノードのうちの前記特定ノードは、前記特定ノードが、前記複数の検査用の欠陥のあるソフトウェアプログラムの中の前記複数の欠陥のうちの前記一部の前記複数の共通欠陥属性のうちの１つ以上を含むことに基づき、前記プライマリノードとして分類される、ステップと、
を含む、付記１４に記載のシステム。
（付記１９）　前記欠陥のあるソフトウェアプログラムの前記欠陥は第１欠陥であり、前記編集操作は第１編集操作であり、前記プライマリノードは第１プライマリノードであり、前記訪問パスは第１訪問パスであり、前記編集ノードは第１編集ノードであり、前記複数の欠陥ノードは第２編集ノードを更に含み、前記動作は、さらに、
前記欠陥のあるソフトウェアプログラムと第２欠陥の修復としての第２編集操作を含む前記改善されたソフトウェアプログラムとの間の前記１つ以上の相違に基づき、前記欠陥のあるソフトウェアプログラムの前記第２欠陥に関する前記第２編集操作を識別するステップと、
前記複数の欠陥ノードのうちの別の特定ノードを、前記第２編集操作を実施する際の開始点として動作する第２プライマリノードとして分類するステップと、
前記第２プライマリノードから前記第２編集ノードへの第２訪問パスを識別するステップであって、前記第２訪問パスは、前記訪問ASTに含まれ、前記第２プライマリノードと前記第２編集ノードとの間のプログラム上の関係を示し、前記修正パターンは、前記第２訪問パス及び前記第２編集操作に更に基づく、ステップと、
を含む、付記１４に記載のシステム。
（付記２０）　前記修正パターンは、前記欠陥のあるソフトウェアプログラムのソースコードに適合する形式で生成される、付記１４に記載のシステム。
【符号の説明】
【０１２９】
１００　環境
１０２　ビッグコード
１０４　欠陥のあるソフトウェアプログラム
１０６　改善されたソフトウェアプログラム
１０７　検査用の欠陥のあるソフトウェアプログラム
１０８　修正パターンモジュール

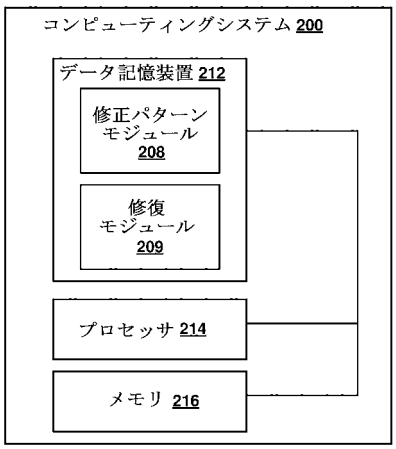
１０９　修復モジュール
１１０　修正パターン
１１１　被検査コード
１１３　変更被検査コード

【図１】



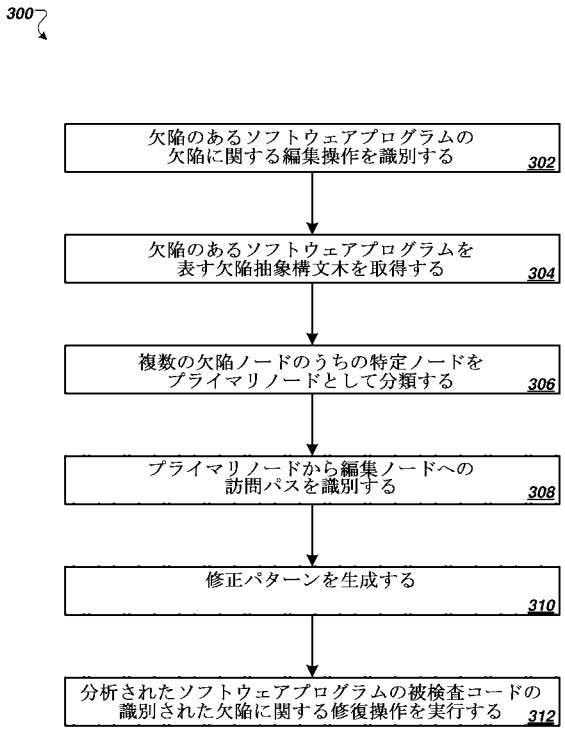
【図２】

ソフトウェアプログラムの修復操作を実行するために使用され得る修正パターンを生成するよう構成され得る例示的なコンピューティングシステムを示す図



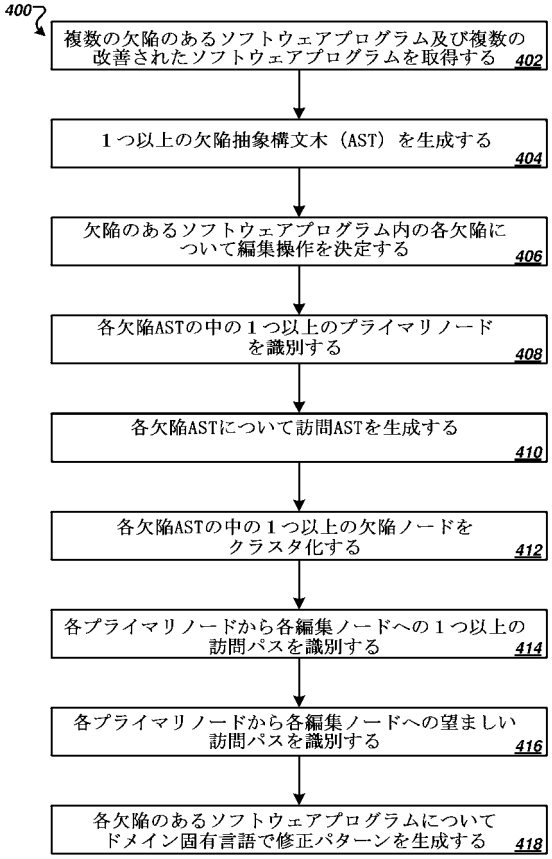
【図３】

ソフトウェアプログラムを修復する例示的な方法のフローチャート



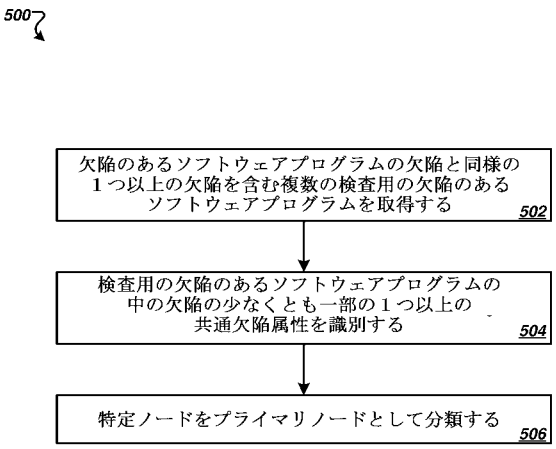
【図４】

ソフトウェアプログラムの修復操作を実行するために使用され得る修正パターンを生成する例示的な方法のフローチャート

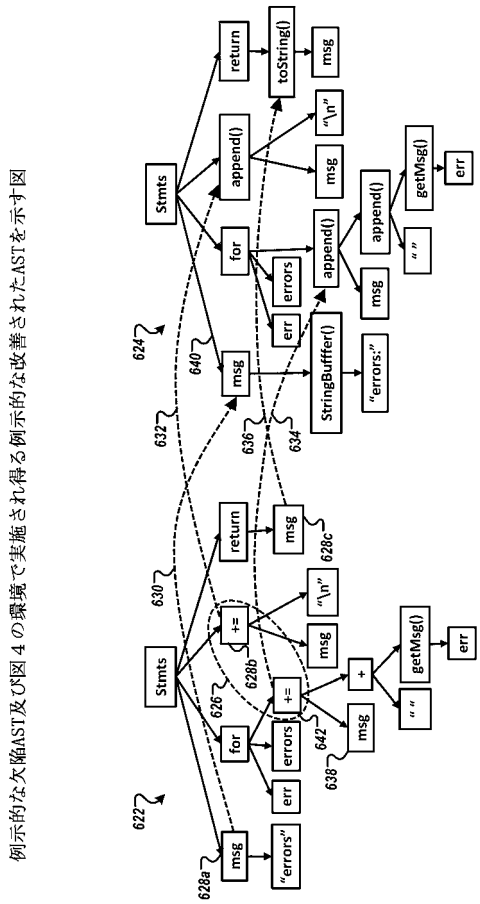


【図５】

特定ノードをプライマリノードとして分類する例示的な方法のフローチャート



【図６】



フロントページの続き

(72)発明者　吉田　浩章
アメリカ合衆国, カリフォルニア州　９４０８５, サニーヴェイル, イースト　アークス　アヴェニュ　１２４０番　フジツウ　ラボラトリーズ　アメリカ内
(72)発明者　プラサド・ムクル　アール
アメリカ合衆国, カリフォルニア州　９４０８５, サニーヴェイル, イースト　アークス　アヴェニュ　１２４０番　フジツウ　ラボラトリーズ　アメリカ内

Ｆターム(参考) 5B042 GA02　HH10　HH40　HH49

【外国語明細書】
2021124852000001.pdf