

(19) 日本国特許庁(JP)

(12) 特 許 公 報(B2)

(11) 特許番号

特許第6079201号  
(P6079201)

(45) 発行日 平成29年2月15日(2017.2.15)

(24) 登録日 平成29年1月27日(2017.1.27)

(51) Int.Cl.

F I

G 0 6 F 11/36 (2006.01)

G 0 6 F 11/36 1 0 8

請求項の数 5 (全 12 頁)

(21) 出願番号	特願2012-273459 (P2012-273459)	(73) 特許権者	000005223
(22) 出願日	平成24年12月14日(2012.12.14)		富士通株式会社
(65) 公開番号	特開2014-119869 (P2014-119869A)		神奈川県川崎市中原区上小田中4丁目1番1号
(43) 公開日	平成26年6月30日(2014.6.30)		
審査請求日	平成27年8月4日(2015.8.4)	(74) 代理人	100103528
			弁理士 原田 一男
		(72) 発明者	片山 朝子
			神奈川県川崎市中原区上小田中4丁目1番1号 富士通株式会社内
		(72) 発明者	上原 忠弘
			神奈川県川崎市中原区上小田中4丁目1番1号 富士通株式会社内
		(72) 発明者	前田 芳晴
			神奈川県川崎市中原区上小田中4丁目1番1号 富士通株式会社内

最終頁に続く

(54) 【発明の名称】 シンボリック実行方法、シンボリック実行プログラム及びシンボリック実行装置

(57) 【特許請求の範囲】

【請求項 1】

シンボル変数の値を含む第1のデータをファイルに書き込むことを命令する書き込み命令を実行する場合、前記書き込み命令に従って前記第1のデータを前記ファイルに書き込むと共に、内部表現の前記シンボル変数の値を記憶装置に格納し、

前記第1のデータを前記ファイルから読み込むことを命令する読み込み命令を実行する場合、前記記憶装置から内部表現の前記シンボル変数の値を読み込み、内部表現の前記シンボル変数の値を含む第2のデータを前記第1のデータに基づき生成し、前記第2のデータを前記読み込み命令の返り値に設定する

処理をコンピュータが実行するシンボリック実行方法。

10

【請求項 2】

前記シンボル変数の値を含まない第3のデータを前記ファイル以外のファイルである第2のファイルから読み込むことを命令する第2の読み込み命令を実行する場合、前記第2の読み込み命令に従って前記第3のデータを前記第2のファイルから読み込み、前記第2の読み込み命令の返り値に設定する

処理をさらに前記コンピュータが実行する請求項1記載のシンボリック実行方法。

【請求項 3】

内部表現の前記シンボル変数の値を前記記憶装置に格納する処理において、

前記第1のデータが書き込まれる前記ファイルの情報を内部表現の前記シンボル変数の値に対応付けて前記記憶装置に格納し、

20

前記第 2 のデータを前記読み込み命令の返り値に設定する処理において、  
 前記読み込み命令において指定されている前記ファイルの情報に対応付けて格納されて  
 いる内部表現の前記シンボル変数の値を、前記記憶装置から読み込む  
 ことを特徴とする請求項 1 記載のシンボリック実行方法。

【請求項 4】

シンボル変数の値を含む第 1 のデータをファイルに書き込むことを命令する書き込み命令を実行する場合、前記書き込み命令に従って前記第 1 のデータを前記ファイルに書き込むと共に、内部表現の前記シンボル変数の値を記憶装置に格納し、

前記第 1 のデータを前記ファイルから読み込むことを命令する読み込み命令を実行する場合、前記記憶装置から内部表現の前記シンボル変数の値を読み込み、内部表現の前記シンボル変数の値を含む第 2 のデータを前記第 1 のデータに基づき生成し、前記第 2 のデータを前記読み込み命令の返り値に設定する

10

処理をコンピュータに実行させるためのシンボリック実行プログラム。

【請求項 5】

シンボル変数の値を含む第 1 のデータをファイルに書き込むことを命令する書き込み命令を実行する場合、前記書き込み命令に従って前記第 1 のデータを前記ファイルに書き込むと共に、内部表現の前記シンボル変数の値を記憶装置に格納する書き込み処理部と、

前記第 1 のデータを前記ファイルから読み込むことを命令する読み込み命令を実行する場合、前記記憶装置から内部表現の前記シンボル変数の値を読み込む読み込み処理部と、

前記読み込み処理部により読み込まれた内部表現の前記シンボル変数の値を含む第 2 のデータを前記第 1 のデータに基づき生成し、前記第 2 のデータを前記読み込み命令の返り値に設定する設定部と

20

を有するシンボリック実行装置。

【発明の詳細な説明】

【技術分野】

【0001】

本発明は、シンボリック実行技術に関する。

【背景技術】

【0002】

シンボリック実行とは、プログラム内の変数を具体化せず、変数をシンボル化して（すなわち、記号のまま）プログラムを実行する技術である。シンボル化される変数は、シンボル変数と呼ばれる。シンボリック実行中においては、具体値ではなくシンボル変数が満たすべき条件（以下、パス条件と呼ぶ）がシンボル変数の値としてメモリ等に保持される。そして、シンボリック実行が完了すると、プログラムにおける各パスについてパス条件が得られるため、パス条件を満たすシンボル変数の具体値を求めることで、プログラムをテストするためのテストデータを得ることができる。

30

【0003】

例えば、図 1 に示した制御構造を有するプログラムについてシンボリック実行を行うことを考える。この制御構造には、2 つの分岐が含まれる。1 つ目の分岐は、シンボル変数  $b$  が 0 であるか否かを判断する分岐であり、0 である場合にはパス 1 1 に進み、0 ではない場合にはパス 1 2 に進む。2 つ目の分岐は、シンボル変数  $b$  が 5 以下であるか否かを判断する分岐であり、5 以下である場合には変数  $a$  をデクリメントするパス 2 1 に進み、5 より大きい場合には変数  $a$  をインクリメントするパス 2 2 に進む。

40

【0004】

図 2 に、図 1 に示した制御構造を有するプログラムの一例を示す。このプログラムにおいては、1 行目の「@Symbolic("true")」がシンボリック実行を行うことを表している。そして、実際にこのプログラムについてシンボリック実行を行うと、 $b = 0$  というテストデータ（パス条件は（ $b = 0$ ）（ $b \leq 5$ ））と、 $b = 1$  というテストデータ（パス条件は（ $b = 0$ ）（ $b \leq 5$ ））と、 $b = 6$  というテストデータ（パス条件は（ $b = 0$ ）（ $b > 5$ ））とを得ることができる。

50

## 【 0 0 0 5 】

ここで、図 1 に示した制御構造を図 3 に示すように変更する。図 3 に示した制御構造において、1 つ目の分岐は図 1 と同じである。1 つ目の分岐と 2 つ目の分岐との間に、シンボル変数 b の値を「T e s t . t x t」という名前のファイルに出力する処理と、「T e s t . t x t」という名前のファイルから値を読み出す処理とが含まれる。2 つ目の分岐においては、ファイルから読み出した値が 5 以下である場合には変数 a をデクリメントするパス 2 1 に進み、5 より大きい場合には変数 a をインクリメントするパス 2 2 に進む。

## 【 0 0 0 6 】

図 4 に、図 3 に示した制御構造を有するプログラムの一例を示す。このプログラムは、図 2 に示したプログラムに、点線で囲った 5 行のコードを追加したプログラムである。そして、実際にこのプログラムについてシンボリック実行を行うと、b = 0 というテストデータ（パス条件は ( b = 0 ) ）と、b = 1 というテストデータ（パス条件は ( b = 0 ) ）とを得ることができる。このテストデータは、図 2 の例で得られるテストデータとは異なっている。

## 【 0 0 0 7 】

このように、プログラムの途中でシンボル変数の値を永続化すると、適切なテストデータを得られなくなることがある。これは、永続化によりシンボル変数が消えるためである。図 4 の例であれば、「d i s . r e a d I n t ( )」という命令によってファイルから読み込む値がシンボル変数 b の値であることが認識されず、シンボル変数 b が消えるため、シンボリック実行にシンボル変数 b を引き継ぐことができない。シンボリック実行に関する従来の技術においては、このような問題については検討されていない。

## 【 先行技術文献 】

## 【 特許文献 】

## 【 0 0 0 8 】

【 特許文献 1 】 特開 2 0 1 2 - 6 8 8 6 9 号公報

## 【 発明の概要 】

## 【 発明が解決しようとする課題 】

## 【 0 0 0 9 】

1 つの側面では、本発明の目的は、シンボル変数の値が永続化される場合であってもシンボリック実行を行えるようにするための技術を提供することである。

## 【 課題を解決するための手段 】

## 【 0 0 1 0 】

本発明に係るシンボリック実行方法は、シンボル変数の値を含む第 1 のデータをファイルに書き込むことを命令する書き込み命令を実行する場合、書き込み命令に従って第 1 のデータをファイルに書き込むと共に、内部表現のシンボル変数の値を記憶装置に格納し、第 1 のデータをファイルから読み込むことを命令する読み込み命令を実行する場合、記憶装置から内部表現のシンボル変数の値を読み込み、内部表現のシンボル変数の値を含む第 1 のデータを、読み込み命令の返り値に設定する処理を含む。

## 【 発明の効果 】

## 【 0 0 1 1 】

シンボル変数の値が永続化される場合であってもシンボリック実行を行えるようになる。

## 【 図面の簡単な説明 】

## 【 0 0 1 2 】

【 図 1 】 図 1 は、プログラムの制御構造を示す図である。

【 図 2 】 図 2 は、プログラムの一例を示す図である。

【 図 3 】 図 3 は、プログラムの制御構造を示す図である。

【 図 4 】 図 4 は、プログラムの一例を示す図である。

【 図 5 】 図 5 は、本実施の形態における情報処理装置の機能ブロック図である。

【 図 6 】 図 6 は、ファイルへの書き込みの際に実行する処理の処理フローを示す図である

10

20

30

40

50

。

【図 7】図 7 は、シンボル変数格納部に格納されるデータの一例を示す図である。

【図 8】図 8 は、シンボル値格納部に格納されるデータの一例を示す図である。

【図 9】図 9 は、ファイル格納部に格納されるデータの一例を示す図である。

【図 10】図 10 は、ファイルからの読み込みの際に実行する処理の処理フローを示す図である。

【図 11】図 11 は、コンピュータの機能ブロック図である。

【発明を実施するための形態】

【0013】

図 5 に、本実施の形態における情報処理装置 1 の機能ブロック図を示す。情報処理装置 1 は、シンボリック実行の対象となるクラスである対象クラスを格納する対象クラス格納部 110 と、モデルクラス群 101、判断部 102、シンボル変数格納部 104 及び実行エンジン 103 を含むシンボリック実行部 100 と、第 1 処理部 106 及び第 2 処理部 107 を含む入出力処理部 105 と、シンボル値格納部 109 と、ファイル格納部 108 とを含む。

【0014】

実行エンジン 103 は、シンボリック実行を行うモジュールであり、例えば Java (登録商標) Path Finder である。実行エンジン 103 は、シンボリック実行におけるシンボル変数をシンボル変数格納部 104 を用いて管理している。

【0015】

モデルクラス群 101 は、複数のモデルクラスを含む。モデルクラスは、例えばファイルへのデータの書き込み又はファイルからのデータの読み込み等の処理を実行するためのクラスであり、対象クラスから呼び出される。

【0016】

判断部 102 は、モデルクラス群 101 における、ファイルへのデータの書き込み又はファイルからのデータの読み込み等を実行するためのモデルクラスが呼び出されると、当該モデルクラスから呼び出される。判断部 102 は、シンボル変数格納部 104 に格納されているデータを用いて、出力データにシンボル変数が含まれているか否かを判断し、含まれている場合には書き込みメソッドに係るデータを入出力処理部 105 に出力する。また、判断部 102 は、読み込み対象のファイルの情報がシンボル値格納部 109 に格納されているか判断し、格納されている場合には読み込みメソッドに係るデータを入出力処理部 105 に出力する。

【0017】

入出力処理部 105 における第 1 処理部 106 は、判断部 102 から受け取った、書き込みメソッドに係るデータを用いて処理を行う。入出力処理部 105 における第 2 処理部 107 は、判断部 102 から受け取った、読み込みメソッドに係るデータを用いて処理を行う。

【0018】

次に、図 6 乃至図 10 を用いて、情報処理装置 1 の動作について説明する。まず、図 6 乃至図 9 を用いて、ファイルへの書き込みを行う際に実行する処理について説明する。

【0019】

シンボリック実行部 100 における実行エンジン 103 は、ユーザからの指示を受け付けると、対象クラス格納部 110 に格納されている対象クラスを用いて、シンボリック実行を開始する。シンボリック実行中、実行エンジン 103 は、ファイルへの書き込み命令 (例えば図 4 の例であれば、`dos.writeInt(b)`) を実行する (図 6 : ステップ S1)。ファイルへの書き込み命令とは、Java (登録商標) であれば、例えば `writeInt`、`writeChar` 又は `writeLong` 等を含む命令である。

【0020】

実行エンジン 103 がファイルへの書き込み命令を実行すると、書き込みメソッドを実行するモデルクラスがモデルクラス群 101 から呼び出される。また、呼び出されたモデ

10

20

30

40

50

ルクラスが、判断部 102 を呼び出し（ステップ S3）、判断部 102 に書き込みメソッドに係るデータを出力する。この処理は、例えば、モデルクラス「DataOutputStream」の中に、書き込みメソッドを実行する部分とは別に、判断部 102 を呼び出す部分を設けることにより実現する。なお、書き込みメソッドに係るデータには、出力データ、ファイル名、書き込み開始位置の情報、出力データ長の情報及び型の情報等が含まれる。

#### 【0021】

判断部 102 は、シンボル変数格納部 104 に格納されているシンボル変数が出力データに含まれているか判断する（ステップ S5）。

#### 【0022】

図 7 に、シンボル変数格納部 104 に格納されるデータの一例を示す。図 7 の例では、シンボル変数と、変数の型とが格納される。例えば図 4 に示したプログラムの場合、冒頭の 2 行がシンボル変数を定義する部分にあたるため、この部分が実行されると図 7 に示したようなデータが格納される。

#### 【0023】

例えば、writeInt(b) という書き込みメソッドであれば b が出力データであるが、b はシンボル変数としてシンボル変数格納部 104 に登録されているため、シンボル変数が出力データに含まれるということになる。

#### 【0024】

シンボル変数が出力データに含まれない場合（ステップ S7：No ルート）、通常どおりの書き込みメソッドであるので、ステップ S9 の処理に移行する。そして、書き込みメソッドを実行するモデルクラスが、書き込みメソッドに係るデータに従って出力データをファイル格納部 108 におけるファイルに書き込む（ステップ S9）。

#### 【0025】

一方、シンボル変数が出力データに含まれる場合（ステップ S7：Yes ルート）、判断部 102 は、書き込みメソッドに係るデータを入出力処理部 105 に出力する。そして、入出力処理部 105 における第 1 処理部 106 は、実行エンジン 103 が使用するメモリ領域から、内部表現のシンボル変数の値を取得する（ステップ S11）。内部表現のシンボル変数の値には、ステップ S11 の時点においてメモリ領域に格納されているパス条件が含まれる。

#### 【0026】

第 1 処理部 106 は、書き込みメソッドに係るデータに含まれるファイルの情報と取得された内部表現のシンボル変数の値とをシンボル値格納部 109 に追加する（ステップ S13）。

#### 【0027】

図 8 に、シンボル値格納部 109 に格納されるデータの一例を示す。図 8 の例では、ファイル名と、書き込み開始位置の情報及び出力データ長の情報と、内部表現のシンボル変数の値とが格納される。

#### 【0028】

第 1 処理部 106 は、判断部 102 から受け取った、書き込みメソッドに係るデータに従って、ファイル格納部 108 におけるファイルに出力データを書き込む（ステップ S15）。そして処理を終了する。

#### 【0029】

図 9 に、ファイル格納部 108 に格納されるファイルの一例を示す。図 9 に示したファイルは、ファイル名が「Test.txt」であり、出力データは「0」である。内部表現のシンボル変数 b の値は図 8 における 2 行目の 3 列目に示された値であるが、ファイル上では整数型（Int）で表されているため「0」になっている。

#### 【0030】

以上のような処理を実行すれば、ファイルへの書き込みの時点における出力データを保存できるので、後にファイルから読み込みを行う際に利用できるようになる。

10

20

30

40

50

## 【 0 0 3 1 】

次に、図 1 0 を用いて、ファイルからの読み込みを行う際に実行する処理について説明する。

## 【 0 0 3 2 】

シンボリック実行部 1 0 0 における実行エンジン 1 0 3 は、ユーザからの指示を受け付けると、対象クラス格納部 1 1 0 に格納されている対象クラスを用いて、シンボリック実行を開始する。シンボリック実行中、実行エンジン 1 0 3 は、ファイルの読み込み命令（例えば図 4 の例であれば、`dis.readInt()`）を実行する（図 1 0：ステップ S 2 1）。ファイルの読み込み命令とは、Java（登録商標）であれば、例えば `readInt`、`readChar` 又は `readLong` 等を含む命令である。

10

## 【 0 0 3 3 】

実行エンジン 1 0 3 がファイルの読み込み命令を実行すると、読み込みメソッドを実行するモデルクラスがモデルクラス群 1 0 1 から呼び出される。また、呼び出されたモデルクラスが、判断部 1 0 2 を呼び出し（ステップ S 2 3）、読み込みメソッドに係るデータを判断部 1 0 2 に出力する。例えば、モデルクラス「`DataInputStream`」の中に、読み込みメソッドを実行する部分とは別に、判断部 1 0 2 を呼び出す部分を設けることにより実現する。なお、読み込みメソッドに係るデータには、ファイル名、読み込み開始位置の情報、データ長の情報及び型の情報等が含まれる。

## 【 0 0 3 4 】

判断部 1 0 2 は、読み込み対象のファイルの情報（ここでは、ファイル名、読み込み開始位置の情報及びデータ長の情報）がシンボル値格納部 1 0 9 に格納されているか判断する（ステップ S 2 5）。

20

## 【 0 0 3 5 】

格納されていない場合（ステップ S 2 7：No ルート）、通常どおりの読み込みメソッドであるので、読み込みメソッドを実行するモデルクラスが、読み込みメソッドに係るデータに従ってデータをファイル格納部 1 0 8 におけるファイルから読み込む。そして、読み込みメソッドを実行するモデルクラスは、ファイルから読み込んだデータを読み込みメソッドの返り値に設定する（ステップ S 2 9）。

## 【 0 0 3 6 】

一方、格納されている場合（ステップ S 2 7：Yes ルート）、判断部 1 0 2 は、読み込みメソッドに係るデータを入出力処理部 1 0 5 に出力する。そして、入出力処理部 1 0 5 における第 2 処理部 1 0 7 は、読み込み対象のファイルの情報に対応する内部表現のシンボル変数の値をシンボル値格納部 1 0 9 から取得する（ステップ S 3 1）。シンボル値格納部 1 0 9 に格納されている内部表現のシンボル変数の値には、書き込みの時点においてメモリ領域に格納されていたパス条件が含まれる。

30

## 【 0 0 3 7 】

第 2 処理部 1 0 7 は、取得された内部表現のシンボル変数の値を含む設定値を生成する（ステップ S 3 3）。例えば `writeInt(b)` という命令では出力データそのものがシンボル変数であるため、ステップ S 3 3 の処理をスキップしてもよい。これに対し、出力データの一部にシンボル変数が含まれるような場合には、ステップ S 3 3 の処理を実行する。そして、第 2 処理部 1 0 7 は、生成された設定値を判断部 1 0 2 に出力する。

40

## 【 0 0 3 8 】

判断部 1 0 2 は、第 2 処理部 1 0 7 から受け取った設定値を、読み込みメソッドの返り値に設定する（ステップ S 3 5）。例えば、第 2 処理部 1 0 7 は、返り値として `SymbolicExp[(s__b__3__SYMINT__STATIC__FIELD+CONST__1)]` を設定する。そして処理を終了する。これにより、実行エンジン 1 0 3 にシンボル変数の値を引き継ぐことができる。

## 【 0 0 3 9 】

以上のような処理を実行すれば、ファイル上の値ではなく内部表現のシンボル変数の値をシンボリック実行に引き継ぐことができるので、シンボル変数の値が永続化される場合

50

であってもシンボリック実行を行えるようになる。

【 0 0 4 0 】

以上本発明の一実施の形態を説明したが、本発明はこれに限定されるものではない。例えば、上で説明した情報処理装置 1 の機能ブロック構成は実際のプログラムモジュール構成に対応しない場合もある。

【 0 0 4 1 】

また、上で説明した各テーブルの構成は一例であって、上記のような構成でなければならないわけではない。さらに、処理フローにおいても、処理結果が変わらなければ処理の順番を入れ替えることも可能である。さらに、並列に実行させるようにしても良い。

【 0 0 4 2 】

例えば、上では J a v a (登録商標) のプログラムを例にして説明したが、その他のプログラミング言語のプログラムであってもよい。

【 0 0 4 3 】

なお、上で述べた情報処理装置 1 は、コンピュータ装置であって、図 1 1 に示すように、メモリ 2 5 0 1 と C P U (Central Processing Unit) 2 5 0 3 とハードディスク・ドライブ (H D D : Hard Disk Drive) 2 5 0 5 と表示装置 2 5 0 9 に接続される表示制御部 2 5 0 7 とリムーバブル・ディスク 2 5 1 1 用のドライブ装置 2 5 1 3 と入力装置 2 5 1 5 とネットワークに接続するための通信制御部 2 5 1 7 とがバス 2 5 1 9 で接続されている。オペレーティング・システム (O S : Operating System) 及び本実施例における処理を実施するためのアプリケーション・プログラムは、H D D 2 5 0 5 に格納されており、C P U 2 5 0 3 により実行される際には H D D 2 5 0 5 からメモリ 2 5 0 1 に読み出される。C P U 2 5 0 3 は、アプリケーション・プログラムの処理内容に応じて表示制御部 2 5 0 7 、通信制御部 2 5 1 7 、ドライブ装置 2 5 1 3 を制御して、所定の動作を行わせる。また、処理途中のデータについては、主としてメモリ 2 5 0 1 に格納されるが、H D D 2 5 0 5 に格納されるようにしてもよい。本発明の実施例では、上で述べた処理を実施するためのアプリケーション・プログラムはコンピュータ読み取り可能なリムーバブル・ディスク 2 5 1 1 に格納されて頒布され、ドライブ装置 2 5 1 3 から H D D 2 5 0 5 にインストールされる。インターネットなどのネットワーク及び通信制御部 2 5 1 7 を経由して、H D D 2 5 0 5 にインストールされる場合もある。このようなコンピュータ装置は、上で述べた C P U 2 5 0 3 、メモリ 2 5 0 1 などのハードウェアと O S 及びアプリケーション・プログラムなどのプログラムとが有機的に協働することにより、上で述べたような各種機能を実現する。

【 0 0 4 4 】

以上述べた本発明の実施の形態をまとめると、以下のようになる。

【 0 0 4 5 】

本実施の形態に係るシンボリック実行方法は、( A ) シンボル変数の値を含む第 1 のデータをファイルに書き込むことを命令する書き込み命令を実行する場合、書き込み命令に従って第 1 のデータをファイルに書き込むと共に、内部表現のシンボル変数の値を記憶装置に格納し、( B ) 第 1 のデータをファイルから読み込むことを命令する読み込み命令を実行する場合、記憶装置から内部表現のシンボル変数の値を読み込み、内部表現のシンボル変数の値を含む第 1 のデータを、読み込み命令の戻り値に設定する処理を含む。

【 0 0 4 6 】

このようにすれば、ファイル上のシンボル変数の値ではなく内部表現のシンボル変数の値をシンボリック実行に引き継ぐことができるので、シンボル変数の値が永続化される場合であってもシンボリック実行を行えるようになる。

【 0 0 4 7 】

また、本シンボリック実行方法が、( C ) シンボル変数の値を含まない第 2 のデータを上記ファイル以外のファイルである第 2 のファイルから読み込むことを命令する第 2 の読み込み命令を実行する場合、第 2 の読み込み命令に従って第 2 のデータを第 2 のファイルから読み込み、第 2 の読み込み命令の戻り値に設定する処理をさらに含むようにしてもよ

10

20

30

40

50

い。このようにすれば、シンボル変数の値を含まないデータをファイルから読み込む場合には通常どおりの読み込みを行うことができるようになる。

【 0 0 4 8 】

また、上で述べた内部表現のシンボル変数の値を記憶装置に格納する処理において、( a 1 ) 第 1 のデータが書き込まれるファイルの情報を内部表現のシンボル変数に対応付けて記憶装置に格納し、上で述べた読み込み命令の戻り値に設定する処理において、( b 1 ) 読み込み命令において指定されているファイルの情報に対応付けて格納されている内部表現のシンボル変数の値を、記憶装置から読み込むようにしてもよい。このようにすれば、例えば複数のシンボル変数を扱うような場合であっても、適切な値を記憶装置から読み込めるようになる。

10

【 0 0 4 9 】

なお、上記方法による処理をコンピュータに行わせるためのプログラムを作成することができ、当該プログラムは、例えばフレキシブルディスク、C D - R O M、光磁気ディスク、半導体メモリ、ハードディスク等のコンピュータ読み取り可能な記憶媒体又は記憶装置に格納される。尚、中間的な処理結果はメインメモリ等の記憶装置に一時保管される。

【 0 0 5 0 】

以上の実施例を含む実施形態に関し、さらに以下の付記を開示する。

【 0 0 5 1 】

( 付記 1 )

シンボル変数の値を含む第 1 のデータをファイルに書き込むことを命令する書き込み命令を実行する場合、前記書き込み命令に従って前記第 1 のデータを前記ファイルに書き込むと共に、内部表現の前記シンボル変数の値を記憶装置に格納し、

20

前記第 1 のデータを前記ファイルから読み込むことを命令する読み込み命令を実行する場合、前記記憶装置から前記内部表現のシンボル変数の値を読み込み、前記内部表現のシンボル変数の値を含む前記第 1 のデータを、前記読み込み命令の戻り値に設定する

処理をコンピュータが実行するシンボリック実行方法。

【 0 0 5 2 】

( 付記 2 )

前記シンボル変数の値を含まない第 2 のデータを前記ファイル以外のファイルである第 2 のファイルから読み込むことを命令する第 2 の読み込み命令を実行する場合、前記第 2 の読み込み命令に従って前記第 2 のデータを前記第 2 のファイルから読み込み、前記第 2 の読み込み命令の戻り値に設定する

30

処理をさらにコンピュータが実行する付記 1 記載のシンボリック実行方法。

【 0 0 5 3 】

( 付記 3 )

前記内部表現のシンボル変数の値を記憶装置に格納する処理において、

前記第 1 のデータが書き込まれる前記ファイルの情報を前記内部表現のシンボル変数に対応付けて前記記憶装置に格納し、

前記読み込み命令の戻り値に設定する処理において、

前記読み込み命令において指定されているファイルの情報に対応付けて格納されている前記内部表現のシンボル変数の値を、前記記憶装置から読み込む

40

ことを特徴とする付記 1 記載のシンボリック実行方法。

【 0 0 5 4 】

( 付記 4 )

シンボル変数の値を含む第 1 のデータをファイルに書き込むことを命令する書き込み命令を実行する場合、前記書き込み命令に従って前記第 1 のデータを前記ファイルに書き込むと共に、内部表現の前記シンボル変数の値を記憶装置に格納し、

前記第 1 のデータを前記ファイルから読み込むことを命令する読み込み命令を実行する場合、前記記憶装置から前記内部表現のシンボル変数の値を読み込み、前記内部表現のシンボル変数の値を含む前記第 1 のデータを、前記読み込み命令の戻り値に設定する

50



処理をコンピュータに実行させるためのシンボリック実行プログラム。

【 0 0 5 5 】

( 付 記 5 )

シンボル変数の値を含む第 1 のデータをファイルに書き込むことを命令する書き込み命令を実行する場合、前記書き込み命令に従って前記第 1 のデータを前記ファイルに書き込むと共に、内部表現の前記シンボル変数の値を記憶装置に格納する書き込み処理部と、

前記第 1 のデータを前記ファイルから読み込むことを命令する読み込み命令を実行する場合、前記記憶装置から前記内部表現のシンボル変数の値を読み込む読み込み処理部と、

前記読み込み処理部により読み込まれた前記内部表現のシンボル変数の値を含む前記第 1 のデータを、前記読み込み命令の返り値に設定する設定部と

を有するシンボリック実行装置。

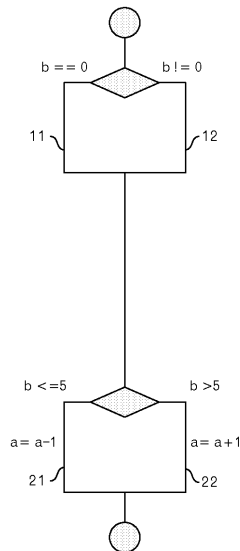
10

【符号の説明】

【 0 0 5 6 】

1	情報処理装置	1 0 0	シンボリック実行装置
1 0 1	モデルクラス群	1 0 2	判断部
1 0 3	実行エンジン	1 0 4	シンボル変数格納部
1 0 5	入出力処理部	1 0 6	第 1 処理部
1 0 7	第 2 処理部	1 0 8	ファイル格納部
1 0 9	シンボル値格納部	1 1 0	対象クラス格納部

【 図 1 】



【 図 2 】

```

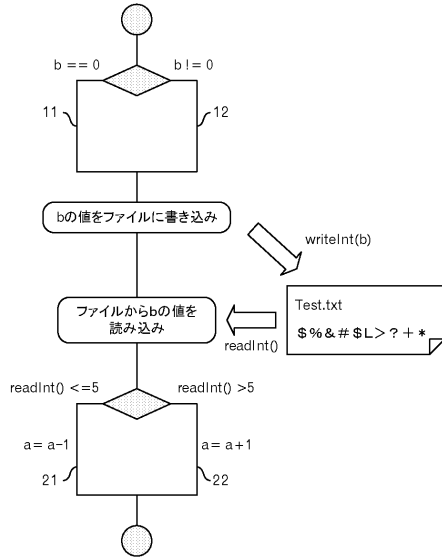
@Symbolic("true")
public static int b = 1134;

public static void testFile(){
try{
    if(b == 0)
        b = b - 1;
    else
        b = b + 1;

    if(b <= 5)
        a = a - 1;
    else
        a = a + 1;

} catch (Exception ex){
    ex.printStackTrace();
}
}
  
```

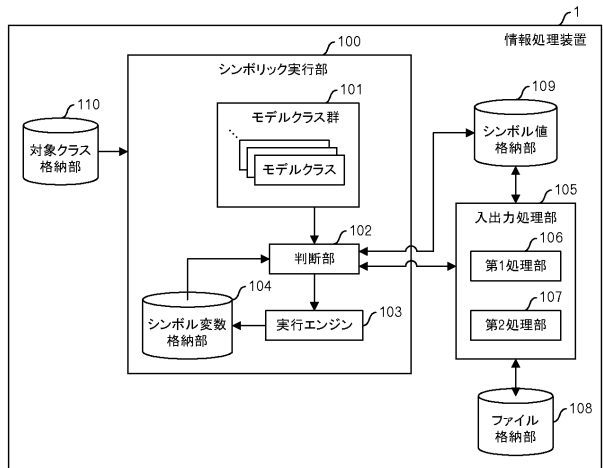
【図 3】



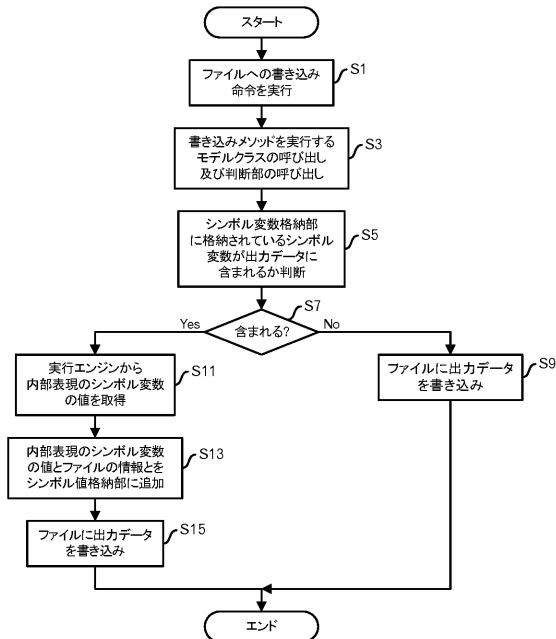
【図 4】

```
@Symbolic("true")
public static int b = 1134;
public static void testFile()
{
    try{
        if(b == 0){
            b = b - 1;
        }else{
            b = b + 1;
        }
        FileOutputStream fos = new FileOutputStream("test.txt");
        DataOutputStream dos = new DataOutputStream(fos);
        dos.writeInt(b);
        FileInputStream fis = new FileInputStream("test.txt");
        DataInputStream dis = new DataInputStream(fis);
        if(dis.readInt() <= 5){
            a = a - 1;
        }else{
            a = a + 1;
        }
    }catch(Exception ex){
        ex.printStackTrace();
    }
}
```

【図 5】



【図 6】



【図 8】

ファイル名	位置と長さ	内部表現のシンボル変数の値
Test.property	0バイトから長さ256バイト	(SR_STATIC_FIELD != CONST_0) && (SR_SYMINT_STATIC_FIELD == CONST_0)
Test.txt	0バイトから長さ4バイト	s_b_3_SYMINT_STATIC_FIELD + CONST_1

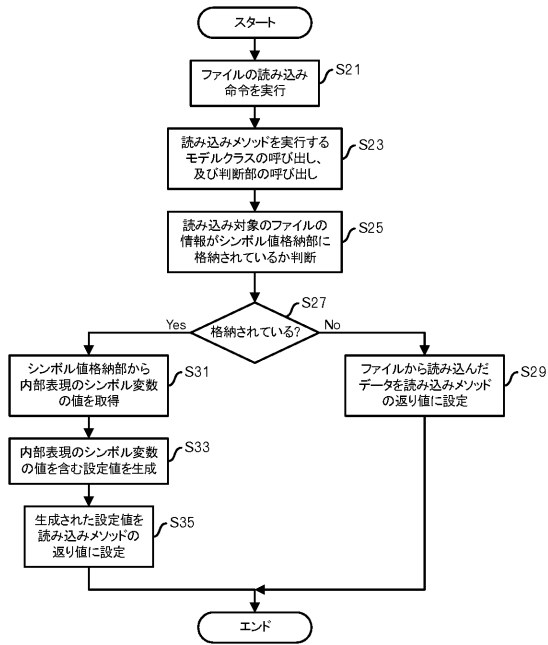
【図 9】

Test.txt  
0

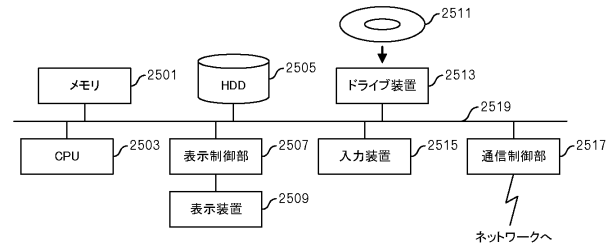
【図 7】

シンボル変数	型
b	int

【図 10】



【図 11】



---

フロントページの続き

(72)発明者 徳本 晋

神奈川県川崎市中原区上小田中4丁目1番1号 富士通株式会社内

(72)発明者 藤原 翔一郎

神奈川県川崎市中原区上小田中4丁目1番1号 富士通株式会社内

審査官 多胡 滋

(56)参考文献 特開2012-018675(JP,A)

特開2007-122207(JP,A)

(58)調査した分野(Int.Cl., DB名)

G06F 11/36