(12) 特許協力条約に基づいて公開された国際出願

(19) 世界知的所有権機関
国際事務局

(43) 国際公開日
2012 年 5 月 3 日(03.05.2012)

PCT

(10) 国際公開番号
WO 2012/056569 A1

(51) 国際特許分類:
G06F 11/34 (2006.01)

(21) 国際出願番号: PCT/JP2010/069335

(22) 国際出願日: 2010 年 10 月 29 日(29.10.2010)

(25) 国際出願の言語: 日本語

(26) 国際公開の言語: 日本語

(71) 出願人 (米国を除く全ての指定国について): 株式会社日立製作所 **(HITACHI, LTD.)** [JP/JP]; 〒1008280 東京都千代田区丸の内一丁目６番６号 Tokyo (JP).

(72) 発明者; および

(75) 発明者/出願人 (米国についてのみ): 南谷　真哉 **(MINATANI, Shinya)** [JP/JP]; 〒2448555 神奈川県横浜市戸塚区戸塚町５０３０番地　株式会社日立製作所　ソフトウェア事業部内 Kanagawa (JP).

(74) 代理人: 後藤　政喜**(GOTO, Masaki)**; 〒1000013 東京都千代田区霞が関三丁目３番１号尚友会館 Tokyo (JP).

(81) 指定国 (表示のない限り、全ての種類の国内保護が可能): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) 指定国 (表示のない限り、全ての種類の広域保護が可能): ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), ユーラシア (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), ヨーロッパ (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

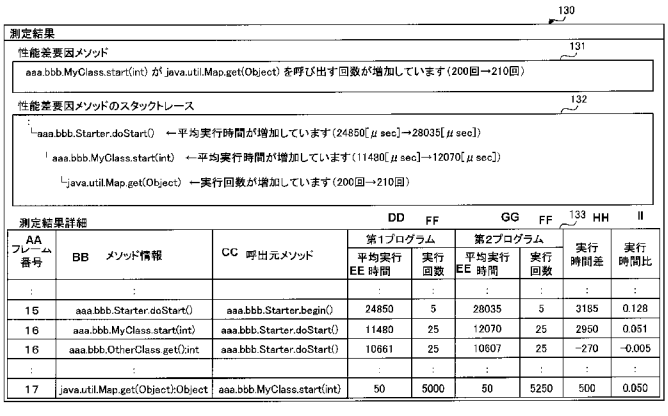
添付公開書類:

— 国際調査報告（条約第 21 条(3)）

(54) **Title**: PERFORMANCE MEASUREMENT METHOD, PERFORMANCE MEASUREMENT DEVICE, AND PERFORMANCE MEASUREMENT PROGRAM

(54) 発明の名称 : 性能測定方法、性能測定装置、及び、性能測定プログラム

[図6]



AA フレーム番号	BB メソッド情報	CC 呼出元メソッド	第1プログラム EE 平均実行時間	第1プログラム 実行回数	第2プログラム EE 平均実行時間	第2プログラム 実行回数	実行時間差	実行時間比
:	:		:	:	:	:	:	:
15	aaa.bbb.Starter.doStart()	aaa.bbb.Starter.begin()	24850	5	28035	5	3185	0.128
16	aaa.bbb.MyClass.start(int)	aaa.bbb.Starter.doStart()	11480	25	12070	25	2950	0.051
16	aaa.bbb.OtherClass.get():int	aaa.bbb.Starter.doStart()	10661	25	10607	25	-270	-0.005
:	:		:	:	:	:	:	:
17	java.util.Map.get(Object):Object	aaa.bbb.MyClass.start(int)	50	5000	50	5250	500	0.050

FIG. 6:
130 MEASUREMENT RESULTS
131 PERFORMANCE DIFFERENCE FACTOR METHOD
132 STACK TRACE OF PERFORMANCE DIFFERENCE FACTOR METHOD
133 MEASUREMENT RESULT DETAILS
AA FRAME NUMBER
BB METHOD INFORMATION
CC CALL ORIGIN METHOD
DD FIRST PROGRAM
EE AVERAGE EXECUTION TIME
FF NUMBER OF TIMES OF EXECUTION
GG SECOND PROGRAM
HH EXECUTION TIME DIFFERENCE
II EXECUTION TIME RATIO

(57) **Abstract**: A performance measurement device repeatedly executes a procedure of initially determining at least one first method among methods which are directly called by a program for which performance is to be measured, and by executing the program, measuring execution time of each of the first methods, and then, using the execution time of each of the measured first methods, assessing whether or not at least one second method matching a predetermined condition can be extracted from each of the first methods, and if at least one second method can be extracted, assessing whether or not at least one third method which is directly called from each of the extracted second methods can be extracted, and if at least one third method can be extracted, determining the extracted third method as a first method, measuring the execution time thereof, assessing whether or not a second method can be extracted, assessing whether or not a third method can be extracted, and determining the third method as a first method.

(57) 要約:

[続葉有]

WO 2012/056569 A1

性能測定装置は、性能が測定されるプログラムによって直接呼ばれるメソッドのうち、少なくとも一つの第１のメソッドを最初に決定し、プログラムを実行することによって、前記各第１のメソッドの実行時間を測定し、前記測定された各第１のメソッドの実行時間を用いて、所定の条件に一致する少なくとも一つの第２のメソッドが各第１のメソッドから抽出可能か否かを判定し、少なくとも一つの第２のメソッドが抽出可能である場合、抽出された各第２のメソッドから直接呼ばれる少なくとも一つの第３のメソッドが抽出可能か否かを判定し、少なくとも一つの第３のメソッドが抽出可能である場合、抽出された第３のメソッドを、第１のメソッドに決定し、実行時間を測定し、第２のメソッドが抽出可能か否かを判定し、第３のメソッドが抽出可能か否かを判定し、第３のメソッドを第１のメソッドに決定する手順を繰り返し実行する。

明　細　書

発明の名称 ：

性能測定方法、性能測定装置、及び、性能測定プログラム

技術分野

[0001] 本発明は、性能測定方法に関し、特に、プログラムの性能を測定する性能測定方法に関する。

背景技術

[0002] 複数のメソッドを含むプログラムを用いてシステムを稼動させる場合、期待した処理性能によってシステムが稼動するか否かを判定するため、プログラムの開発者等は、プログラムに特定の入力をし、それに対するシステムの処理性能を測定する。そして、測定の結果、システムが期待した処理性能によって稼動しない場合、開発者等には、プログラムのうち、処理性能に問題がある箇所を特定する必要が生じる。

[0003] 従来用いられる一般的な方法には、あらかじめプログラムに埋め込まれた機能を用いて、実行中のプログラムにログを出力させ、出力されたログによって、プログラムの全体の処理の開始から終了までの実行時刻を取得する方法などがある。しかし、このような方法は、荒い精度でしか実行時間を取得できなかった。

[0004] また、従来の方法には、ＯＳ又は仮想マシンが備えるプロファイリング機能を用いて、個々のメソッドの実行時間を測定するプロファイラと呼ばれるツールを用いる方法がある。この方法によれば、プロファイラを用いることによって、より詳細な精度に、プログラムの実行性能を分析することができる。そして、プログラムを最適化したり、処理性能に問題があった場合に問題を解決することができる。

[0005] しかし、一般的に、メソッドの個数はプログラムの規模にほぼ比例して増え、メソッドが実行される回数もプログラム全体の実行時間にほぼ比例して増える。このため、大規模なプログラム、及び、起動時間が長いプログラム

（例えば、サーバを稼動させるためのプログラムなど）の処理性能を測定する場合、プログラムに含まれるメソッドが大量になるため、プロファイラによる個々のメソッドの実行時間を記録することは、困難であった。

[0006] また、性能問題が発生した場合におけるボトルネックとなるメソッドの特定など、性能を測定したいメソッドが特定できていない場合、実行される全てのメソッドの実行時間を測定し、測定された結果を記録しておく必要がある。しかし、一般的にメソッドは、非常に短い時間の間隔で頻繁に呼び出されるため、頻繁に発生する実行時間の測定及び記録自体が、プログラムが稼動するシステムのリソースを消費した。このため、測定自体が性能に影響を及ぼし、正確な測定が困難であった。

[0007] これに対し、従来、１回目の測定において、荒い精度によって全てのメソッドの実行時間を測定し、所定の閾値以上の実行時間がかかったメソッドなど所定の条件に該当するメソッドを主要メソッドとして抽出する方法が開示されている（例えば、特許文献１参照）。そして、特許文献１は、２回目の測定において、前記主要メソッドのみを対象として詳細な精度で実行時間を測定する技術を開示する。

先行技術文献

特許文献

[0008] 特許文献1：特開２００８－２１７７２１号公報

発明の概要

発明が解決しようとする課題

[0009] 特許文献１に開示された技術は、特定の入力に対応する全てのメソッドを測定し、その統計情報（平均実行時間、実行回数）を生成する。このため、全体の実行時間が非常に短いプログラムを測定する場合、特許文献１に開示された技術は、正確な実行時間を取得できず、実際はボトルネックであるメソッドが、主要メソッドとして抽出されない場合がある。

[0010] さらに、特許文献１に開示された技術において用いられる所定の閾値は、

どのメソッドにも一定値である。このため、例えば、性能に影響のある修正が加えられたプログラムと、修正される前のプログラムとの二つのプログラム間で性能の比較をする場合において、プログラム全体の実行時間と比較して実行時間に大きな差はないが、修正前と修正後とのメソッドの性能に著しい差が生じているメソッドを特定する、といった用途に、特許文献１に開示された技術は用いることができない。

[0011] 本発明はこのような課題に鑑みてなされたものであり、処理性能に影響を与えるメソッドを精度よく特定し、かつ、処理性能を測定するプログラムの規模に関わらず、測定自体による処理性能への影響を低減する測定方法の提供を目的とする。

[0012] また、本発明は、複数のプログラム間で性能に差が生じているメソッドを特定するための性能の測定方法の提供を目的とする。

課題を解決するための手段

[0013] 本発明の代表的な一例を示せば以下の通りである。すなわち、計算機によってプログラムの性能を測定する性能測定方法であって、前記計算機は、プロセッサと、前記プロセッサによって実行されるプログラムを格納するメモリとを有し、前記方法は、前記計算機が、前記性能が測定されるプログラムによって直接呼ばれるメソッドのうち、少なくとも一つの第１のメソッドを最初に決定する手順と、前記計算機が、前記プログラムを実行することによって、前記決定された各第１のメソッドの実行時間を測定する手順と、前記計算機が、前記測定された各第１のメソッドの実行時間を用いて、所定の条件に一致する少なくとも一つの第２のメソッドが前記各第１のメソッドから抽出可能か否かを判定する手順と、前記少なくとも一つの第２のメソッドが抽出可能である場合、前記計算機が、前記抽出された各第２のメソッドから直接呼ばれる少なくとも一つの第３のメソッドが抽出可能か否かを判定する手順と、前記少なくとも一つの第３のメソッドが抽出可能である場合、前記計算機が、前記抽出された第３のメソッドを、第１のメソッドに決定する手順と、前記計算機が、前記実行時間を測定する手順と、前記第２のメソッド

が抽出可能か否かを判定する手順と、前記第３のメソッドが抽出可能か否かを判定する手順と、前記第３のメソッドを第１のメソッドに決定する手順とを繰り返し実行する手順とを含む。

発明の効果

[0014] 本発明の一実施形態によると、プログラムにおいて処理性能に影響を与えているメソッドを精度よく特定することができる。

図面の簡単な説明

[0015] [図1]本発明の第１の実施形態の複数のプログラムの性能差を分析する計算機の構成を示すブロック図である。

[図2]本発明の第１の実施形態の対象メソッドテーブルを示す説明図である。

[図3A]本発明の第１の実施形態の第１メソッド呼出テーブルを示す説明図である。

[図3B]本発明の第１の実施形態の第２メソッド呼出テーブルを示す説明図である。

[図4A]本発明の第１の実施形態の第１実行情報テーブルを示す説明図である。

[図4B]本発明の第１の実施形態の第２実行情報テーブルを示す説明図である。

[図5]本発明の第１の実施形態の測定結果テーブルを示す説明図である。

[図6]本発明の第１の実施形態の測定結果を表示した場合の画面を示す説明図である。

[図7]本発明の第１の実施形態の全体の処理を示すフローチャートである。

[図8]本発明の第１の実施形態の初期メソッドを示すシーケンス図である。

[図9]本発明の第１の実施形態の初期メソッドを抽出する処理におけるデータの流れを示すブロック図である。

[図10A]本発明の第１の実施形態の初期メソッドを抽出するためにテストを実行する処理を示すフローチャートである。

[図10B]本発明の第１の実施形態の測定対象メソッドを抽出する処理を示すフ

ローチャートである。

[図10C]本発明の第１の実施形態の初期メソッドを抽出する処理を示すフローチャートである。

[図11]本発明の第１の実施形態の測定対象メソッドの実行時間を測定する処理におけるデータの流れを示すブロック図である。

[図12A]本発明の第１の実施形態の実行時間を測定するためにテストを実行する処理を示すフローチャートである。

[図12B]本発明の第１の実施形態の測定対象メソッドの実行時間を測定する処理を示すフローチャートである。

[図13]本発明の第１の実施形態の性能差要因メソッドを抽出する処理を示すフローチャートである。

[図14]本発明の第１の実施形態の測定対象メソッドを再度設定する処理を示すフローチャートである。

[図15]本発明の第２の実施形態の単一のプログラムの性能差を分析する計算機の構成を示すブロック図である。

[図16]本発明の第２の実施形態のメソッド呼出テーブルを示す説明図である。

[図17]本発明の第２の実施形態の実行情報テーブルを示す説明図である。

[図18]本発明の第２の実施形態の測定結果テーブルを示す説明図である。

[図19]本発明の第２の実施形態の全体の処理を示すフローチャートである。

[図20]本発明の第２の実施形態の所定の条件を満たすメソッドを抽出する処理を示すフローチャートである。

[図21]本発明の第２の実施形態の初期メソッドを指定するための画面を示す説明図である。

発明を実施するための形態

[0016] 以下、本発明について、図面を参照して説明する。

[0017] 図１は、本発明の第１の実施形態の複数のプログラムの性能差を分析する計算機１００の構成を示すブロック図である。

[0018] 第１の実施形態の計算機１００は、演算装置１０１、メモリ１０２、外部記憶装置１０３、出力装置１０４、及び入力装置１０５を備える。計算機１００は、複数のプログラムの処理性能の差を分析する機能を備える装置である。

[0019] 演算装置１０１は、ＣＰＵなどのプロセッサである。演算装置１０１は、メモリ１０２においてプログラムを実行する。

[0020] メモリ１０２は、一時的にデータを記憶するための領域を備える記憶装置である。メモリ１０２は、プログラム及びテーブルを保持する。

[0021] 外部記憶装置１０３は、ハードディスクドライブなどの記憶装置である。演算装置１０１によって実行されるプログラムに入力される引数等を保持する。

[0022] 出力装置１０４は、ディスプレイ又はプリンタ等の装置である。計算機１００による処理の結果を、ユーザに提供するための装置である。入力装置１０５は、キーボード又はマウス等の装置である。ユーザは、演算装置１０１によって実行されるプログラムに入力される引数等を、入力装置１０５を介して計算機１００に入力する。

[0023] メモリ１０２は、制御部１０６、テスト実行部１０７、メソッド呼出照合部１０８、性能差要因メソッド判定部１０９、バージョン切替部１１０、第１メソッド呼出テーブル１１１、第２メソッド呼出テーブル１１２、対象メソッドテーブル１１３、第１実行情報テーブル１１４、第２実行情報テーブル１１５、測定結果テーブル１１６、初期メソッド記録部１１７、対象メソッド記録部１１８、コールバック受信部１１９、第１プログラム１２０、第２プログラム１２１、ＪａｖａＶＭ１２２（Ｊａｖａは登録商標、以下同じ）、及び、ＪＶＭＴＩ１２３を保持する。

[0024] 制御部１０６、テスト実行部１０７、メソッド呼出照合部１０８、性能差要因メソッド判定部１０９、バージョン切替部１１０、初期メソッド記録部１１７、対象メソッド記録部１１８、コールバック受信部１１９、第１プログラム１２０、第２プログラム１２１、ＪａｖａＶＭ１２２、及び、ＪＶＭ

ＴＩ１２３は、プログラムである。第１メソッド呼出テーブル１１１、第２メソッド呼出テーブル１１２、対象メソッドテーブル１１３、第１実行情報テーブル１１４、第２実行情報テーブル１１５、及び、測定結果テーブル１１６は、データが格納されるテーブルである。

[0025] 制御部１０６は、バージョン切替部１１０に第１プログラム１２０又は第２プログラム１２１を実行可能な状態に切り替えさせ、テスト実行部１０７に第１プログラム１２０又は第２プログラム１２１を実行（テスト）させる。また、第１プログラム１２０又は第２プログラム１２１を実行させることによって取得された測定結果の集計を同期させるなど、メモリ１０２において実行されるプログラムを制御する。

[0026] テスト実行部１０７は、第１プログラム１２０又は第２プログラム１２１のうち実行可能な状態になっているプログラムに、あらかじめユーザによって指定された所定の値を入力し、第１プログラム１２０又は第２プログラム１２１を実行する。そして、これによって、性能を測定するために必要な負荷をかけて、第１プログラム１２０又は第２プログラム１２１を実行する。

[0027] メソッド呼出照合部１０８は、初期メソッド又は性能差要因メソッドによって直接呼ばれたメソッドのうち、所定の条件に一致するメソッドを次の測定対象に設定するため、所定の条件に一致するメソッドを示す情報を対象メソッドテーブル１１３に格納する。

[0028] なお、第１の実施形態において初期メソッドとは、第１プログラム１２０又は第２プログラム１２１をテストしている期間中、各スレッドにおいて最初に呼ばれるメソッドである。また、第１の実施形態において性能差要因メソッドとは、第１プログラム１２０と第２プログラム１２１との性能に差があった場合、最も性能の差を発生させる要因となるメソッドである。

[0029] 性能差要因メソッド判定部１０９は、第１実行情報テーブル１１４及び第２実行情報テーブル１１５に格納されたメソッドの開始時刻とメソッドの終了時刻との差分に基づいて、性能を示す値を算出する。そして、測定結果テーブル１１６に算出された値を格納する。

[0030] 具体的には、性能差要因メソッド判定部１０９は、第１プログラム１２０及び第２プログラム１２１における測定対象メソッドの平均実行時間と実行回数とを、第１実行情報テーブル１１４及び第２実行情報テーブル１１５から抽出する。そして、抽出された値に基づいて、第１プログラム１２０と第２プログラム１２１との実行時間差と実行時間比とを算出する。そして、算出された値を測定結果テーブル１１６に格納する。さらに、測定結果テーブル１１６に出力された実行時間差と実行時間比とに基づいて、性能差要因メソッドを抽出する。

[0031] ここで、測定対象メソッドとは、本実施形態において性能を測定する対象のメソッドである。

[0032] バージョン切替部１１０は、第１プログラム１２０及び第２プログラム１２１のうち、制御部１０６から指定されたプログラムを起動し、第１プログラム１２０又は第２プログラム１２１を実行可能な状態にする。

[0033] 第１プログラム１２０及び第２プログラム１２１は、第１の実施形態の計算機１００によって、性能を測定されるプログラムである。第１の実施形態において、第１プログラム１２０はユーザによる修正前のプログラムであり、第２プログラム１２１はユーザによる修正後のプログラムである。

[0034] 本実施形態の第１プログラム１２０及び第２プログラム１２１は、Ｊａｖａ等によって作成されたプログラムであり、ＪａｖａＶＭ（仮想マシン上）１２２において実行される。ＪａｖａＶＭ１２２は、第１プログラム１２０及び第２プログラム１２１によって呼ばれるメソッドが、開始された時及び終了した時において、コールバックを発行するためのプロファイル機能を備える。ＪＶＭＴＩ１２３は、本実施形態のＪａｖａＶＭ１２２に備わり、コールバックを発行するためのプロファイル機能である。

[0035] しかし、本発明の第１プログラム１２０及び第２プログラム１２１は、Ｊａｖａ言語によって作成されたプログラムに限られるものではなく、第１プログラム１２０及び第２プログラム１２１によって呼ばれるメソッド（又は、メソッドに相当するサブルーチン）が開始された時及び終了した時におい

て、コールバック（又は、コールバックに相当する信号）を発行する手段を含む言語であれば、いかなる言語によって作成されてもよい。そして、ＪａｖａＶＭ１２２及びＪＶＭＴＩ１２３は、第１プログラム１２０及び第２プログラム１２１の言語に対応した、第１プログラム１２０及び第２プログラム１２１を実行するための環境であれば、いかなる環境でもよい。

[0036] コールバック受信部１１９は、ＪＶＭＴＩ１２３によって発行されるコールバックを受信し、コールバックが受信された時刻を、計算機１００が備える高精度タイマを用いて生成する。そして、生成された時刻と、コールバックに含まれるメソッドを示す情報（クラス名、メソッド名、メソッド識別子、呼び出し元のメソッドを示す情報、及び、コールスタックのフレーム番号など）と、メソッドが開始されたか終了したかを示す情報とを、初期メソッド記録部１１７又は対象メソッド記録部１１８に送信する。

[0037] 初期メソッド記録部１１７は、テスト実行部１０７によって第１プログラム１２０及び第２プログラム１２１に値が入力されている間（すなわち、第１プログラム１２０又は第２プログラム１２１がテストされている間）に、コールスタックに格納された最も上位のメソッドを抽出する。そして、抽出されたメソッドを示す情報を、各々第１メソッド呼出テーブル１１１及び第２メソッド呼出テーブル１１２に格納する。

[0038] 本実施形態において、第１プログラム１２０又は第２プログラム１２１に呼ばれたメソッドは、スレッド毎のコールスタックに格納され、開始される。そして、最も遅くコールスタックに格納され（ＰＵＳＨ）たメソッドは、最も早くコールスタックから取得される（ＰＯＰ）。すなわち、最も遅く開始されたメソッドが、最も早く終了する。

[0039] 本実施形態において、コールスタックに最も早く格納されたメソッドは、最も上位のメソッドと記載され、コールスタックに最も上位のメソッドより遅く格納されたメソッドは、下位のメソッドと記載される。

[0040] 対象メソッド記録部１１８は、テスト実行部１０７によって第１プログラム１２０又は第２プログラム１２１に値が入力されている間（すなわち、第

１プログラム１２０又は第２プログラム１２１がテストされている間）に、対象メソッドテーブル１１３に格納されている測定対象メソッドが開始された時刻及び終了した時刻と、メソッドを示す情報とを、コールバック受信部１１９から受信する。そして、コールバック受信部１１９から受信された時刻等の情報を、第１実行情報テーブル１１４又は第２実行情報テーブル１１５に格納する。

[0041] また、対象メソッド記録部１１８は、第１プログラム１２０又は第２プログラム１２１を測定している間に、測定対象メソッドによって直接呼ばれたメソッドを示す情報を抽出する。そして、抽出されたメソッドを示す情報を、第１メソッド呼出テーブル１１１又は第２メソッド呼出テーブル１１２に格納する。

[0042] 第１メソッド呼出テーブル１１１は、第１プログラム１２０によって呼ばれたメソッドを示す情報が格納されるテーブルである。第２メソッド呼出テーブル１１２は、第２プログラム１２１によって呼ばれたメソッドを示す情報が格納されるテーブルである。対象メソッドテーブル１１３は、第１プログラム１２０及び第２プログラム１２１によって呼ばれたメソッドのうち、第１プログラム１２０と第２プログラム１２１との間で共通のメソッドを示す情報が格納されるテーブルである。

[0043] 第１実行情報テーブル１１４は、第１プログラム１２０によって呼ばれたメソッドの開始時刻と終了時刻とが格納されるテーブルである。第２実行情報テーブル１１５は、第２プログラム１２１によって呼ばれたメソッドの開始時刻と終了時刻とが格納されるテーブルである。測定結果テーブル１１６は、第１実行情報テーブル１１４及び第２実行情報テーブル１１５に格納される各メソッドの開始時刻と終了時刻とに基づいて算出された、第１プログラム１２０と第２プログラム１２１との性能を比較するための値が格納されるテーブルである。

[0044] 図２は、本発明の第１の実施形態の対象メソッドテーブル１１３を示す説明図である。

[0045] 対象メソッドテーブル１１３は、開始時刻と終了時刻とが測定されるメソッドを示す情報が格納される。対象メソッドテーブル１１３に格納される値は、第１プログラム１２０又は第２プログラム１２１によって呼ばれたメソッドが、さらに別のメソッドを呼ぶ場合に更新される。すなわち、測定対象メソッドによって呼ばれたメソッドを、さらに次の測定対象メソッドとするため、対象メソッドテーブル１１３は更新される。

[0046] 対象メソッドテーブル１１３は、フレーム番号１１３１、メソッド情報１１３２、及び、呼出元メソッド情報１１３３を含む。フレーム番号１１３１は、コールスタックに格納されるメソッドの順番を示す数字である。フレーム番号１１３１の値が小さい程、コールスタックにおいて上位のメソッドであることを示す。

[0047] メソッド情報１１３２は、測定対象メソッドを示す情報が含まれる。メソッド情報１１３２に格納される値は、測定対象メソッドのクラス名とメソッド名とを含み、測定対象メソッドを一意に示す識別子である。

[0048] 呼出元メソッド情報１１３３は、メソッド情報１１３２に示す測定対象メソッドを呼んだメソッドを示す情報を含む。呼出元メソッド情報１１３３に格納される値は、メソッド情報１１３２に格納される値と同じ表記方法である。すなわち、呼出元メソッド情報１１３３に格納される値は、呼出元メソッドのクラス名とメソッド名とを含み、呼出元メソッドを一意に示す識別子である。

[0049] 図３Ａは、本発明の第１の実施形態の第１メソッド呼出テーブル１１１を示す説明図である。

[0050] 第１メソッド呼出テーブル１１１は、第１プログラム１２０が測定される際に、第１プログラム１２０によって呼ばれるメソッドを示す情報が格納される。第１メソッド呼出テーブル１１１は、フレーム番号１１１１、メソッド情報１１１２、及び、呼出元メソッド情報１１１３を含む。フレーム番号１１１１、メソッド情報１１１２、及び、呼出元メソッド情報１１１３は、対象メソッドテーブル１１３のフレーム番号１１３１、メソッド情報１１３

２、及び、呼出元メソッド情報１１３３に相当する。

[0051] 第１プログラム１２０に呼ばれた初期メソッドの開始時刻及び終了時刻を測定されている間、第１メソッド呼出テーブル１１１のメソッド情報１１１２には、初期メソッドによって直接呼ばれるメソッドを示す情報が、対象メソッド記録部１１８によって蓄積される。また、初期メソッドよりも下位の測定対象メソッドの開始時刻及び終了時刻が測定されている間、第１メソッド呼出テーブル１１１のメソッド情報１１１２には、測定対象メソッドによって直接呼ばれるメソッドを示す情報が、対象メソッド記録部１１８によって蓄積される。

[0052] 図３Ｂは、本発明の第１の実施形態の第２メソッド呼出テーブル１１２を示す説明図である。

[0053] 第２メソッド呼出テーブル１１２は、第２プログラム１２１が測定される際に、第２プログラム１２１によって呼ばれるメソッドを示す情報が格納される。第２メソッド呼出テーブル１１２は、フレーム番号１１２１、メソッド情報１１２２、及び、呼出元メソッド情報１１２３を含む。フレーム番号１１２１、メソッド情報１１２２、及び、呼出元メソッド情報１１２３は、対象メソッドテーブル１１３のフレーム番号１１３１、メソッド情報１１３２、及び、呼出元メソッド情報１１３３に相当する。

[0054] 第２メソッド呼出テーブル１１２に格納される情報は、第１メソッド呼出テーブル１１１に格納される情報と同様であり、第２プログラム１２１が実行されることによって抽出された値が格納される。

[0055] 図４Ａは、本発明の第１の実施形態の第１実行情報テーブル１１４を示す説明図である。

[0056] 第１実行情報テーブル１１４は、第１プログラム１２０によって呼ばれたメソッドの、開始時刻又は終了時刻を格納するテーブルである。第１実行情報テーブル１１４は、フレーム番号１１４１、メソッド情報１１４２、開始終了情報１１４３、及び時刻１１４４を含む。フレーム番号１１４１及びメソッド情報１１４２は、対象メソッドテーブル１１３のフレーム番号１１３

１及びメソッド情報１１３２に相当する。

[0057] 開始終了情報１１４３に格納される値は、各行に含まれる時刻１１４４が、メソッド情報１１４２が示すメソッドの開始時刻を示す場合″開始″であり、各行に含まれる時刻１１４４が、メソッド情報１１３２が示すメソッドの終了時刻を示す場合″終了″である。

[0058] 時刻１１４４は、メソッド情報１１４２が示すメソッドの開始時刻又は終了時刻を示す。図４Ａに示す時刻１１４４は、計算機１００において用いられるシステム時刻によって表記されるが、いかなる表記方法で時刻を示してもよい。

[0059] 図４Ｂは、本発明の第１の実施形態の第２実行情報テーブル１１５を示す説明図である。

[0060] 第２実行情報テーブル１１５は、第２プログラム１２１によって呼ばれたメソッドの、開始時刻又は終了時刻を格納するテーブルである。第２実行情報テーブル１１５に格納される情報は、第１実行情報テーブル１１４と同様の情報であり、第２プログラム１２１が実行されることによって抽出される値が格納される。

[0061] 前述の第１メソッド呼出テーブル１１１、第２メソッド呼出テーブル１１２、対象メソッドテーブル１１３、第１実行情報テーブル１１４及び第２実行情報テーブル１１５において、計算機１００による測定の対象とするメソッドが、次の測定対象メソッドに移行する際、各テーブルが更新され、各テーブルのフレーム番号が更新されてよい。すなわち、第１メソッド呼出テーブル１１１、第２メソッド呼出テーブル１１２、対象メソッドテーブル１１３、第１実行情報テーブル１１４及び第２実行情報テーブル１１５に含まれるフレーム番号は、各測定の時点においてすべて同じ値を示してよい。

[0062] ただし、測定対象メソッドが識別できれば、各テーブルはいずれの方法によって値を格納してもよく、例えば、測定対象メソッドを示すフレーム番号を変数によって保持し、変数が示すフレーム番号の測定対象メソッドを、計算機１００に測定させるなどの方法を用いてもよい。

[0063] 図５は、本発明の第１の実施形態の測定結果テーブル１１６を示す説明図である。

[0064] 測定結果テーブル１１６は、第１実行情報テーブル１１４及び第２実行情報テーブル１１５に含まれる開始時刻と終了時刻とに基づいて算出された値をメソッドごとに格納するテーブルである。

[0065] 測定結果テーブル１１６は、フレーム番号１１６１、メソッド情報１１６２、第１プログラムの平均実行時間１１６３１及び実行回数１１６３２、第２プログラムの平均実行時間１１６４１及び実行回数１１６４２、実行時間差１１６５、並びに、実行時間比１１６６を含む。

[0066] フレーム番号１１６１は、コールスタックに格納されるメソッドの順番を示す数字である。測定結果テーブル１１６の行は、メソッドが測定される都度追加されるため、フレーム番号１１６１は、少なくとも一つの値を示す。

[0067] メソッド情報１１６２は、測定されたメソッドを示す情報を格納する。

[0068] 第１プログラムの平均実行時間１１６３１は、各メソッドが呼出元メソッドから複数回呼ばれる場合において、複数回呼ばれた各メソッドの実行時間の平均値である。

[0069] 例えば、第１実行情報テーブル１１４のメソッド情報１１４２に含まれるメソッドを抽出し、抽出されたメソッドの各々の終了時刻から開始時刻を減算することによって、１回目の実行時間が算出される。そして、抽出されたメソッドと同じメソッドを示すメソッドを、さらにメソッド情報１１４２から抽出し、さらに抽出されたメソッドの実行時間を算出することによって２回目の実行時間が算出される。このように算出された複数回の実行時間の平均値を算出することによって、平均実行時間１１６３１が算出される。

[0070] 第１プログラムの実行回数１１６３２は、呼出元メソッドから呼ばれた回数を示す。第１実行情報テーブル１１４のうち、開始終了情報１１５３に開始及び終了の両方を含むメソッドが、第１実行情報テーブル１１４にいくつ含まれるかを算出することによって、求められる。

[0071] 第２プログラムの平均実行時間１１６４１及び第２プログラムの実行回数

１１６４２には、第１プログラムの平均実行時間１１６３１及び実行回数１１６３２と同様な値が格納され、第２実行情報テーブル１１５に基づいて算出された値が格納される。

[0072] 実行時間差１１６５は、呼出元メソッドが１回実行される毎の、メソッド情報１１６２が示すメソッドの実行時間の合計を、第１プログラム１２０と第２プログラム１２１とにおいて算出し、算出された各結果の差を示す。具体的には、第１プログラム１２０及び第２プログラム１２１の結果に基づいて、各々、メソッド情報１１６２が示すメソッドが実行された実行時間の合計を算出する。そして、実行時間の合計を、呼出元メソッドが実行された回数（呼出元メソッドが示す実行回数１１６３２に相当）によって除算する。

[0073] そして、前述の方法によって算出された値を、第１プログラム１２０と第２プログラム１２１とについて生成する。そして、第１プログラム１２０の算出された値によって、第２プログラム１２１の算出された値を減算することによって、実行時間差１１６５が算出される。

[0074] 本発明は、実行時間差１１６５によって、呼出元メソッドが１回実行される毎の、メソッドの実行時間の差を算出することができる。

[0075] 例えば、図５に示すメソッド情報１１６２″ａａａ．ｂｂｂ．ＭｙＣｌａｓｓ．ｓｔａｒｔ（ｉｎｔ）″の実行時間差１１６５の算出方法を以下に示す。図５は、メソッド″ａａａ．ｂｂｂ．ＭｙＣｌａｓｓ．ｓｔａｒｔ（ｉｎｔ）″が、″ａａａ．ｂｂｂ．Ｓｔａｒｔｅｒ．ｄｏＳｔａｒｔ（）″から呼ばれることによって実行されることを示す。

[0076] 第１プログラムにおいて実行される場合において、メソッド″ａａａ．ｂｂｂ．ＭｙＣｌａｓｓ．ｓｔａｒｔ（ｉｎｔ）″の呼出元メソッドによって呼ばれる回数は、図５の実行回数１１６３２が示す通り、２５回である。そして、呼出元メソッドである″ａａａ．ｂｂｂ．Ｓｔａｒｔｅｒ．ｄｏＳｔａｒｔ（）″の実行回数１１６３２は５回である。すなわち、図５は、メソッド″ａａａ．ｂｂｂ．Ｓｔａｒｔｅｒ．ｄｏＳｔａｒｔ（）″が１回実行される毎に、メソッド″ａａａ．ｂｂｂ．ＭｙＣｌａｓｓ．ｓｔａｒｔ（ｉｎｔ）″が５

回（２５／５＝５）呼ばれることを示す。

[0077] なお、図５において、メソッド″ａａａ．ｂｂｂ．ＭｙＣｌａｓｓ．ｓｔａｒｔ（ｉｎｔ）″及びメソッド″ａａａ．ｂｂｂ．Ｓｔａｒｔｅｒ．ｄｏＳｔａｒｔ（）″の、第２プログラムにおける実行回数１１６４２は、第１プログラムにおける実行回数１１６３２と同じである。

[0078] そして、実行時間差１１６５は、第１プログラムにおいて呼出元メソッドが１回実行される毎のメソッドの合計実行時間と、第２プログラムにおいて呼出元メソッドが１回実行される毎のメソッドの合計実行時間との差である。このため、メソッド″ａａａ．ｂｂｂ．ＭｙＣｌａｓｓ．ｓｔａｒｔ（ｉｎｔ）″の実行時間差１１６５は、第２プログラムの（平均実行時間１１６４１×（メソッドの実行回数１１６４２／呼出元メソッドの実行回数１１６４２）－第１プログラムの（平均実行時間１１６３１×（メソッドの実行回数１１６３２／呼出元メソッドの実行回数１１６３２）＝１２０７０×（２５／５）－１１４８０×（２５／５）＝２９５０と算出される。

[0079] ここで、呼出元メソッドがメソッドを呼ぶ回数が、第１プログラム１２０と第２プログラム１２１とにおいて相違する場合においても、実行時間差１１６５は、前述の式によって算出される。

[0080] 実行時間比１１６６は、メソッド情報１１６２が示すメソッドの１回あたりの実行時間の、第１のプログラム及び第２のプログラムとにおける差を算出し、算出された値を平均する。そして、平均された値を、第１プログラムの平均実行時間１１６３１によって除算することによって算出された値である。

[0081] すなわち、実行時間比１１６６は、メソッドが１回実行される毎の、新旧のメソッドにおける実行時間の差の比率である。本発明は、実行時間比１１６６によって、各メソッドの実行時間の変化率を算出することができる。

[0082] 具体的には、実行時間比１１６６は、実行時間差１１６５が示す値を、呼出元メソッドが１回実行される毎の、メソッド情報１１６２が示すメソッドの実行回数によって除算した値を、さらに、第１プログラムの平均実行時間

１１６３１によって除算した値である。

[0083] 例えば、図５において″ａａａ．ｂｂｂ．ＭｙＣｌａｓｓ．ｓｔａｒｔ（ｉｎｔ）″は、前述のとおり、呼出元メソッドが１回実行される毎に、５回呼ばれるため、実行時間比１１６６は、（２９５０／５）／１１４８０＝０．０５１と算出される。

[0084] なお、測定結果テーブル１１６において、初期メソッドの実行時間差１１６５及び実行時間比１１６６は、呼出元メソッド（すなわち、第１プログラム又は第２プログラム）の実行回数は、１として算出される。

[0085] 図６は、本発明の第１の実施形態の測定結果を表示した場合の画面１３０を示す説明図である。

[0086] 画面１３０は、本実施形態によって測定された結果を、出力装置１０４を介してユーザに提供するための画面である。画面１３０は、表示１３１、表示１３２、及び表示１３３を含む。

[0087] 表示１３１は、性能差要因メソッドであるメソッド名と、性能差要因メソッドである理由とを示す。図６に示す表示１３１は、呼出元メソッドである″ａａａ．ｂｂｂ．ＭｙＣｌａｓｓ．ｓｔａｒｔ（ｉｎｔ）″が、″ｊａｖａ．ｕｔｉｌ．Ｍａｐ．ｇｅｔ（Ｏｂｊｅｃｔ）″を呼ぶ回数が増加しており、第１プログラム１２０においては２００回呼び、第２プログラム１２１においては２１０回呼んでいることを示す。

[0088] なお、呼出元メソッドがメソッドを呼ぶ回数は、呼ばれたメソッドの実行回数（測定結果テーブル１１６の第１プログラムの実行回数１１６３２又は第２プログラムの実行回数１１６４２）を、呼出元メソッドの実行回数（測定結果テーブル１１６の第１プログラムの実行回数１１６３２又は第２プログラムの実行回数１１６４２）によって除算することによって算出される。

[0089] 表示１３２は、性能差要因メソッドが呼ばれるまでに呼ばれたメソッドを、階層構造によって示した表示である。すなわち、性能差要因メソッドを含むコールスタックをトレースした結果を示す。表示１３２には、性能差要因メソッドを含むコールスタックの各メソッドと、各メソッドにおける第１プ

ログラム１２０と第２プログラム１２１との性能の差を示す。

[0090] 表示１３３は、測定結果テーブル１１６を含む情報を表示する。図６に示す表示１３３は、測定結果テーブル１１６の情報と、対象メソッドテーブル１１３の呼出元メソッド情報１１３３とを含む。

[0091] ＜処理説明＞

図７は、本発明の第１の実施形態の全体の処理を示すフローチャートである。

[0092] 制御部１０６は、まず、第１プログラム１２０及び第２プログラム１２１の初期メソッドを抽出し、抽出された初期メソッドのうち、第１プログラム１２０及び第２プログラム１２１において共通するメソッドを対象メソッドテーブル１１３に格納する。すなわち、対象メソッドテーブル１１３は、第１プログラム１２０及び第２プログラム１２１において共通するメソッドを対象メソッドテーブル１１３に格納することによって、開始時刻及び終了時刻を測定するメソッド（測定対象メソッド）を、対象メソッドテーブル１１３に格納する（１４０）。

[0093] ステップ１４０の後、制御部１０６は、対象メソッドテーブル１１３に格納された測定対象メソッドの実行時間を測定する（１５０）。ステップ１５０の後、制御部１０６は、ステップ１５０の測定結果に基づいて、性能差要因メソッドを抽出する（１６０）。

[0094] そして制御部１０６は、ステップ１６０において性能差要因メソッドが抽出されたか否かを判定する（１７０）。性能差要因メソッドが抽出されない場合、第１プログラム１２０と第２プログラム１２１との間で性能の差がないため、制御部１０６は処理を終了する。

[0095] 性能差要因メソッドが抽出された場合、制御部１０６は、性能差要因メソッドによって直接呼ばれたメソッドを対象メソッドテーブル１１３に格納し、性能差要因メソッドによって直接呼ばれたメソッドをさらに測定させる（１８０）。これは、性能差要因メソッドによって直接呼ばれるメソッドの中から、最も顕著に性能の差が生じているメソッドを抽出するためである。

[0096] ステップ１８０の後、制御部１０６は、ステップ１５０に戻る。その後、ステップ１７０において性能差要因メソッドが抽出されないと判定された場合、測定対象メソッドを呼んだ呼出元メソッドにおいて、最も性能の差が生じていることを示すため、制御部１０６は、処理を終了する。

[0097] 図８は、本発明の第１の実施形態の初期メソッドを示すシーケンス図である。

[0098] 図８に示すシーケンス図は、スレッドＡ１０７１、スレッドＢ１０７２、スレッド１０７３、スレッドＤ１０７４、メソッド２０１～メソッド２１６の実行時間を示す。

[0099] ステップ１４０は、具体的には、テスト実行部１０７が第１プログラム１２０及び第２プログラム１２１へ特定の入力値を入力し始める時点から、特定の入力値を入力し終える時点までのテスト実行時間中に、第１プログラム１２０及び第２プログラム１２１に含まれる各スレッドのコールスタックにおいて、最も上位に格納されるメソッドを抽出する処理である。

[0100] 図８において、プログラム２００は、テスト実行部１０７が第１プログラム１２０又は第２プログラム１２１へ特定の入力値を入力し始める時から、特定の入力値を入力し終える時までに実行される第１プログラム１２０又は第２プログラム１２１を示す。

[0101] プログラム２００は、スレッドＡ１０７１において実行されるメソッド２０１を呼ぶ。図８において、プログラム２００によって呼ばれるメソッド２０１は、常に実行されるメソッドであり、プログラム２００が開始する前から実行されているメソッドである。このため、メソッド２０１は、性能の差が生じるメソッドではないため、スレッドＡ１０７１における初期メソッドではない。

[0102] プログラム２００によって呼ばれた後、メソッド２０１は、スレッドＢ１０７２において実行されるメソッド２０３を呼ぶ。図８において、メソッド２０１によって呼ばれるメソッド２０３は、プログラム２００が開始した後に実行され、プログラム２００が終了する前に終了するメソッドである。こ

のため、メソッド２０３は、スレッドＢ１０７２における初期メソッドである。ここで、メソッド２０３は、スレッドＢ１０７２におけるコールスタックの最も上位に格納される。

[0103] なお、スレッドBにおけるメソッド２０２及びメソッド２０９は、いずれもプログラム２００が実行される間に実行されるメソッドではなく、性能の差を生じる要因とならないため、初期メソッドではない。

[0104] メソッド２０３は、メソッド２０３が実行されている間に、メソッド２０４及びメソッド２０５を呼ぶ。図８において、メソッド２０３によって呼ばれるメソッド２０４及びメソッド２０５は、プログラム２００が開始した後に実行され、プログラム２００が終了する前に終了するメソッドである。このため、メソッド２０４及びメソッド２０５は、測定対象メソッドとなる可能性があるメソッドである。

[0105] 本実施形態において、初期メソッドであるメソッド２０３の、コールスタックにおけるフレーム番号が″１″である場合、メソッド２０４及びメソッド２０５のフレーム番号は″２″である。

[0106] さらに、メソッド２０５は、メソッド２０６及びメソッド２０７を呼ぶ。ここで、メソッド２０３のフレーム番号が″１″である場合、メソッド２０６及びメソッド２０７の、コールスタックにおけるフレーム番号は″３″である。

[0107] さらに、メソッド２０７はメソッド２０８を呼ぶ。ここで、メソッド２０３のフレーム番号が″１″である場合、コールスタックにおけるメソッド２０８のフレーム番号は″４″である。

[0108] 図８において、メソッド２０４、メソッド２０５、メソッド２０６、メソッド２０７及びメソッド２０８は、プログラム２００が開始した後に実行され、プログラム２００が終了する前に終了するメソッドである。このため、メソッド２０４、メソッド２０５、メソッド２０６、メソッド２０７及びメソッド２０８は、測定対象メソッドとなる可能性があるメソッドである。

[0109] メソッド２０４は、スレッドＣ１０７３におけるメソッド２１０を呼ぶ。

図８において、メソッド２０４によって呼ばれるメソッド２１０は、プログラム２００が終了した後も実行されるメソッドである。このため、メソッド２１０は、性能の差を生じる要因とならないため、スレッドＣ１０７３における初期メソッドではない。

[0110] メソッド２１０は、メソッド２１１を呼ぶ。図８において、メソッド２１０によって呼ばれるメソッド２１１は、プログラム２００が開始した後に実行され、プログラム２００が終了する前に終了するメソッドである。このため、メソッド２１１は、スレッドＣ１０７３における初期メソッドである。ここで、メソッド２１１は、スレッドＣ１０７３におけるコールスタックの上位に格納される。この時、スレッドＣ１０７３のコールスタックには、メソッド２１０が既に格納されているため、メソッド２１１のフレーム番号は″２″である。

[0111] さらに、メソッド２１１はメソッド２１２を呼ぶ。ここで、メソッド２１１のフレーム番号が″２″である場合、メソッド２１２のコールスタックにおけるフレーム番号は″３″である。

[0112] 図８において、メソッド２１２は、プログラム２００が開始した後に実行され、プログラム２００が終了する前に終了するメソッドである。このため、メソッド２１２は、測定対象メソッドとなる可能性があるメソッドである。

[0113] メソッド２１１は、スレッドＤ１０７４におけるメソッド２１４を呼ぶ。図８において、メソッド２１１によって呼ばれるメソッド２１４は、プログラム２００が開始する前から実行されているメソッドである。このため、メソッド２１４は、性能の差を生じる要因とならないため、スレッドＤ１０７４における初期メソッドではない。

[0114] メソッド２１４は、メソッド２１６を呼ぶ。図８において、メソッド２１４によって呼ばれるメソッド２１６は、プログラム２００が開始した後に実行され、プログラム２００が終了する前に終了するメソッドである。このためメソッド２１６は、初期メソッドである。ここで、スレッドＤ１０７４の

コールスタックには、メソッド２１３及びメソッド２１４が既に格納されているため、メソッド２１６のフレーム番号は″３″である。

[0115] スレッドＤ１０７４のメソッド２１５は、メソッド２１６及びメソッド２１４が終了した後に実行され、プログラム２００が終了する前に終了するメソッドである。このため、性能の差を生じる要因となるため、初期メソッドとなる。ここで、スレッドＤ１０７４のコールスタックには、メソッド２１３が既に格納されているため、メソッド２１６のフレーム番号は″２″である。

[0116] 本実施形態の制御部１０６は、前述の初期メソッドをステップ１４０において抽出する。

[0117] 図９は、本発明の第１の実施形態の初期メソッドを抽出する処理におけるデータの流れを示すブロック図である。

[0118] 図９に示す矢印は、図７に示すステップ１４０におけるデータの流れを示し、図１０Ａ、図１０Ｂ、及び、図１０Ｃに示す処理に対応する。

[0119] 図１０Ａは、本発明の第１の実施形態の初期メソッドを抽出するためにテストを実行する処理を示すフローチャートである。

[0120] 図１０Ａに示す処理は、図７に示すステップ１４０に含まれる処理である。まず、第１プログラム１２０をテストが可能な状態にする指示を制御部１０６から受信した後、バージョン切替部１１０は、第１プログラム１２０を実行可能な状態に切り替え、その後、切り替えの終了を制御部１０６に通知する（１４０１）。

[0121] ステップ１４０１の後、制御部１０６は、テスト実行中フラグに″ＴＲＵＥ″を格納する指示を初期メソッド記録部１１７に送信する。初期メソッド記録部１１７は、テスト実行中フラグに″ＴＲＵＥ″を格納する指示を制御部１０６から受信した後、テスト実行中フラグに″ＴＲＵＥ″を格納する（１４０２）。なお、初期メソッド記録部１１７は、メモリ１０２の中に、テスト実行中フラグを示す領域をあらかじめ保持する。

[0122] テスト実行中フラグに″ＴＲＵＥ″が格納される場合、テスト実行中フラグ

は、テスト実行部１０７がプログラムをテストしていることを示す。テスト実行中フラグに″ＦＡＬＳＥ″が格納される場合、テスト実行中フラグは、テスト実行部１０７がプログラムをテストしていないことを示す。

[0123] ステップ１４０２の後、制御部１０６は、第１プログラム１２０のテストを開始する指示をテスト実行部１０７に送信する。テスト実行部１０７は、第１プログラム１２０のテストを開始する指示を制御部１０６から受信すると、第１プログラム１２０のテストを開始する（１４０３）。

[0124] ステップ１４０３の後、テスト実行部１０７は、テストを実行するため、あらかじめユーザによって指定された所定の入力値を第１プログラム１２０に入力する（１４０４）。ステップ１４０４の後、テスト実行部１０７は、第１プログラム１２０のテストを終了し、その後、テストの終了を制御部１０６に通知する（１４０５）。

[0125] ステップ１４０５の後、制御部１０６は、テスト実行中フラグに″ＦＡＬＳＥ″を格納する指示を初期メソッド記録部１１７に送信する。初期メソッド記録部１１７は、テスト実行中フラグに″ＦＡＬＳＥ″を格納する指示を制御部１０６から受信した後、テスト実行中フラグに″ＦＡＬＳＥ″を格納する（１４０６）。

[0126] ステップ１４０６の後、制御部１０６は、第１プログラム１２０を終了させる指示をバージョン切替部１１０に送信する（１４０７）。

[0127] テスト実行中フラグに″ＦＡＬＳＥ″を格納した後、初期メソッド記録部１１７は、スレッドごとのコールスタック（以下、メソッド情報スタックと記載）のうち、終了済みフラグが付加されているメソッドを抽出し、抽出されたメソッドを示す情報を、第１メソッド呼出テーブル１１１に格納する（１４０９）。終了済みフラグは、後述する図１０Ｃに示す処理において更新される記憶領域である。

[0128] そして、初期メソッド記録部１１７は、すべてのスレッドのメソッド情報スタックについて、ステップ１４０９を繰り返す（１４０８）。

[0129] ステップ１４０８が終了した後、バージョン切替部１１０は、第２プログ

ラム１２１をテストが可能な状態にする指示を制御部１０６から受信する。その後、バージョン切替部１１０は、第２プログラム１２１を実行可能な状態に切り替え、その後、切り替えの終了を制御部１０６に通知する（１４１０）。

[0130] ステップ１４１０の後、制御部１０６は、テスト実行中フラグに″ＴＲＵＥ″を格納する指示を初期メソッド記録部１１７に送信する。初期メソッド記録部１１７は、テスト実行中フラグに″ＴＲＵＥ″を格納する指示を制御部１０６から受信した後、テスト実行中フラグに″ＴＲＵＥ″を格納する（１４１１）。

[0131] ステップ１４１１の後、制御部１０６は、第２プログラム１２１のテストを開始する指示をテスト実行部１０７に送信する。テスト実行部１０７は、第２プログラム１２１のテストを開始する指示を制御部１０６から受信すると、第２プログラム１２１のテストを開始する（１４１２）。

[0132] ステップ１４１２の後、テスト実行部１０７は、テストを実行するため、あらかじめユーザによって指定された所定の値を第２プログラム１２１に入力する（１４１３）。ステップ１４１３の後、テスト実行部１０７は、第２プログラム１２１のテストを終了し、その後、テストの終了を制御部１０６に通知する（１４１４）。

[0133] ステップ１４１４の後、制御部１０６は、テスト実行中フラグに″ＦＡＬＳＥ″を格納する指示を初期メソッド記録部１１７に送信する。初期メソッド記録部１１７は、テスト実行中フラグに″ＦＡＬＳＥ″を格納する指示を制御部１０６から受信した後、テスト実行中フラグに″ＦＡＬＳＥ″を格納する（１４１５）。

[0134] ステップ１４１５の後、制御部１０６は、第２プログラム１２１を終了させる指示をバージョン切替部１１０に送信する（１４１６）。

[0135] テスト実行中フラグに″ＦＡＬＳＥ″を格納した後、初期メソッド記録部１１７は、スレッドごとのメソッド情報スタックのうち、対応する終了済みフラグが生成されているメソッドを抽出し、抽出されたメソッドを示す情報を

、第１メソッド呼出テーブル１１１に格納する（１４１８）。

[0136] そして、初期メソッド記録部１１７は、すべてのスレッドのメソッド情報スタックについて、ステップ１４１８を繰り返す（１４１７）。

[0137] ステップ１４１７の後、制御部１０６は、測定対象メソッドを抽出し（１４１９）、図１０Ａに示す処理を終了する。

[0138] 図１０Ｂは、本発明の第１の実施形態の測定対象メソッドを抽出する処理を示すフローチャートである。

[0139] 図１０Ｂに示す処理は、図１０Ａに示すステップ１４１９に相当する。

[0140] 制御部１０６は、図１０Ａのステップ１４１７が終了した後、測定対象メソッドを抽出する指示をメソッド呼出照合部１０８に送信する（１４２０）。測定対象メソッドを抽出する指示をメソッド呼出照合部１０８から受信した後、メソッド呼出照合部１０８は、第１メソッド呼出テーブル１１１に格納される情報を読み込む（１４２１）。また、メソッド呼出照合部１０８は、第２メソッド呼出テーブル１１２に格納される情報を読み込む（１４２２）。

[0141] ステップ１４２１及びステップ１４２２の後、メソッド呼出照合部１０８は、新たな測定対象メソッドを対象メソッドテーブル１１３に格納するため、対象メソッドテーブル１１３に格納されていたデータを削除する（１４２３）。ステップ１４２３の後、メソッド呼出照合部１０８は、第１メソッド呼出テーブル１１１及び第２メソッド呼出テーブル１１２から読み込んだ情報に基づいて、第１メソッド呼出テーブル１１１と第２メソッド呼出テーブル１１２とに共通して格納されるメソッドを抽出し、抽出されたメソッドを対象メソッドテーブル１１３に格納する（１４２４）。

[0142] ステップ１４２４の後、メソッド呼出照合部１０８は、制御部に測定対象メソッドの抽出の終了を通知し、ステップ１４１９の処理を終了する（１４２５）。

[0143] ステップ１４１９の処理によって、第１メソッド呼出テーブル１１１及び第２メソッド呼出テーブル１１２に共通して格納されたメソッドが測定対象

メソッドとして抽出される。

[0144] 図１０Ｃは、本発明の第１の実施形態の初期メソッドを抽出する処理を示すフローチャートである。

[0145] 図１０Ｃに示す処理は、図１０Ａに示すテストの実行中に、各プログラムから呼ばれるメソッドの中から初期メソッドを抽出するための処理である。

[0146] コールバック受信部１１９は、ＪＶＭＴＩ１２３からコールバックを受信し、コールバックが受信された時刻と、コールバックに含まれる各情報を初期メソッド記録部１１７に送信する。コールバックには、前述の通り、メソッドを示す情報（クラス名、メソッド名、メソッド識別子、呼び出し元のメソッドを示す情報、及び、コールスタックにおけるフレーム番号など）と、メソッドが開始されたか終了したかを示す情報とが含まれる。

[0147] ステップ１４３０の後、初期メソッド記録部１１７は、テスト実行中フラグに″ＴＲＵＥ″が格納されているか否かを判定する（１４３１）。テスト実行中フラグに″ＦＡＬＳＥ″が格納されている場合、テストが実行されておらず、初期メソッドを抽出する必要がないため、初期メソッド記録部１１７はステップ１４３８に移行する。

[0148] テスト実行中フラグに″ＴＲＵＥ″が格納されている場合、初期メソッド記録部１１７は、受信されたコールバックに含まれる情報に基づいて、メソッドが開始されたか終了したかを判定する（１４３２）。

[0149] メソッドが開始されたと判定された場合、スレッド毎のメソッド情報スタックにコールバックに含まれるメソッドを示す情報を格納する（ＰＵＳＨ）（１４３３）。ステップ１４３３によって、メソッド情報スタックに格納されたメソッドは、初期メソッドの候補になる。

[0150] ステップ１４３２においてメソッドが終了したと判定された場合、初期メソッド記録部１１７は、コールバックに含まれるメソッドの情報が、既にメソッド情報スタックのうちのいずれかに格納されているか否かを判定する（１４３４）。ステップ１４３４の判定によって、コールバックが示すメソッドが、テストが実行されている間に開始されたメソッドであるか否かを判定

できる。なお、前述の通り、初期メソッドは、テストが実行されている間に、開始され、終了したメソッドである。

[0151] ステップ１４３４において、コールバックに含まれるメソッドの情報が、メソッド情報スタックのいずれにも格納されていないと判定された場合、コールバックが示すメソッドは、テストが実行されている間に開始されていないメソッドであり、初期メソッドにはならないため、初期メソッド記録部１１７は、ステップ１４３９に移行する。

[0152] ステップ１４３４において、コールバックに含まれるメソッドの情報が、メソッド情報スタックのうちのいずれかに格納されていると判定された場合、コールバックが示すメソッドは、テストが実行されている間に開始されたメソッドであり、初期メソッドになる可能性があるメソッドである。このため初期メソッド記録部１１７は、ステップ１４３４の後、ステップ１４３５に移行する。

[0153] 初期メソッド記録部１１７は、ステップ１４３４においてコールバックに含まれるメソッドの情報が格納されていると判定されたメソッド情報スタックの、最も下位に格納されたメソッドを取得する（ＰＯＰ）。そして、取得されたメソッドを示す情報を、先頭のメソッド情報に格納する（１４３５）。先頭のメソッド情報は、メソッドを示す情報を一時的に格納するための記憶領域であり、初期メソッド記録部１１７によってあらかじめ保持される。

[0154] ステップ１４３５の後、初期メソッド記録部１１７は、先頭のメソッド情報が、コールバックに含まれるメソッドを示す情報と同じであるか否かを判定する（１４３６）。先頭のメソッド情報が、コールバックに含まれるメソッドを示す情報と異なる場合、先頭のメソッド情報に格納されたメソッドは、コールバックに含まれるメソッドよりも下位のメソッドである。このため、初期メソッド記録部１１７は、さらに上位のメソッドを先頭のメソッド情報に格納するため、ステップ１４３５に戻る。

[0155] ステップ１４３６において、先頭のメソッド情報が、コールバックに含まれるメソッドを示す情報と同じである場合、初期メソッド記録部１１７は、

先頭のメソッド情報に対応する終了済みフラグを生成する（１４３７）。終了済みフラグが対応して生成されている場合、先頭のメソッド情報が示すメソッドは、終了していることを示す。

[0156] ステップ１４３７の後、初期メソッド記録部１１７は、先頭のメソッド情報に格納されるメソッド情報を、ステップ１４３５において取得されたメソッド情報スタックに再度格納（ＰＵＳＨ）する（１４３８）。

[0157] ステップ１４３８の処理によって、メソッド情報スタックのうち最も下位には、終了済みのメソッドが格納される。また、図１０Ｃに示す処理が、コールバックが発行された都度実行されることによって、テスト実行中に開始され、終了したメソッドのうち、最も上位のメソッドがメソッド情報スタックに残る。このため、テストが終了した後において、メソッド情報スタックに格納され、対応する終了済みフラグが生成されているメソッドは、初期メソッドである。

[0158] ステップ１４３８の後、初期メソッド記録部１１７は、図１０Ｃの処理を終了し、コールバック受信部１１９に処理の終了を通知する（１４３９）。

[0159] 図１０Ｃに示す処理によってメソッド情報スタックに初期メソッドが格納された後、図１０Ａに示すステップ１４０９及びステップ１４１８によって、初期メソッド記録部１１７は、スレッドごとのメソッド情報スタックのうち、終了済みフラグが生成されているメソッドを、第１メソッド呼出テーブル１１１又は第２メソッド呼出テーブル１１２に格納する。これによって、第１メソッド呼出テーブル１１１及び第２メソッド呼出テーブル１１２に初期メソッドが格納される。

[0160] 図１１は、本発明の第１の実施形態の測定対象メソッドの実行時間を測定する処理におけるデータの流れを示すブロック図である。

[0161] 図１１に示す矢印は、図７に示すステップ１５０におけるデータの流れを示し、図１２Ａ、及び、図１２Ｂに示す処理に対応する。

[0162] 図１２Ａは、本発明の第１の実施形態の実行時間を測定するためにテストを実行する処理を示すフローチャートである。

[0163] 図１２Ａに示す処理は、図７に示すステップ１５０に含まれる処理である。まず、第１プログラム１２０をテストが可能な状態にする指示を制御部１０６から受信した後、バージョン切替部１１０は、第１プログラム１２０を実行可能な状態に切り替え、その後、切り替えの終了を制御部１０６に通知する（１５０１）。

[0164] ステップ１５０１の後、制御部１０６は、テスト実行中フラグに″ＴＲＵＥ″を格納する指示を対象メソッド記録部１１８に送信する。対象メソッド記録部１１８は、テスト実行中フラグに″ＴＲＵＥ″を格納する指示を制御部１０６から受信した後、テスト実行中フラグに″ＴＲＵＥ″を格納する（１５０２）。

[0165] なお、対象メソッド記録部１１８は、メモリ１０２の中に、テスト実行中フラグを示す領域をあらかじめ保持する。初期メソッド記録部１１７が保持するテスト実行中フラグと対象メソッド記録部１１８が保持するテスト実行中フラグは、同じ領域に格納されてもよい。

[0166] ステップ１５０２の後、制御部１０６は、第１プログラム１２０のテストを開始する指示をテスト実行部１０７に送信する。テスト実行部１０７は、第１プログラム１２０のテストを開始する指示を制御部１０６から受信すると、第１プログラム１２０のテストを開始する（１５０３）。

[0167] ステップ１５０３の後、テスト実行部１０７は、テストを実行するため、あらかじめユーザによって指定された所定の値を第１プログラム１２０に入力する（１５０４）。ステップ１５０４の後、テスト実行部１０７は、第１プログラム１２０のテストを終了し、その後、テストの終了を制御部１０６に通知する（１５０５）。

[0168] ステップ１５０５の後、制御部１０６は、テスト実行中フラグに″ＦＡＬＳＥ″を格納する指示を対象メソッド記録部１１８に送信する。対象メソッド記録部１１８は、テスト実行中フラグに″ＦＡＬＳＥ″を格納する指示を制御部１０６から受信した後、テスト実行中フラグに″ＦＡＬＳＥ″を格納する（１５０６）。

[0169] ステップ１５０６の後、制御部１０６は、第１プログラム１２０を終了させる指示をバージョン切替部１１０に送信する（１５０７）。

[0170] ステップ１５０７の後、バージョン切替部１１０は、第２プログラム１２１をテストが可能な状態にする指示を制御部１０６から受信する。その後、バージョン切替部１１０は、第２プログラム１２１を実行可能な状態に切り替え、その後、切り替えの終了を制御部１０６に通知する（１５０８）。

[0171] ステップ１５０８の後、制御部１０６は、テスト実行中フラグに″ＴＲＵＥ″を格納する指示を対象メソッド記録部１１８に送信する。対象メソッド記録部１１８は、テスト実行中フラグに″ＴＲＵＥ″を格納する指示を制御部１０６から受信した後、テスト実行中フラグに″ＴＲＵＥ″を格納する（１５０９）。

[0172] ステップ１５０９の後、制御部１０６は、第２プログラム１２１のテストを開始する指示をテスト実行部１０７に送信する。テスト実行部１０７は、第２プログラム１２１のテストを開始する指示を制御部１０６から受信すると、第２プログラム１２１のテストを開始する（１５１０）。

[0173] ステップ１５１０の後、テスト実行部１０７は、テストを実行するため、あらかじめユーザによって指定された所定の値を第２プログラム１２１に入力する（１５１１）。ステップ１５１１の後、テスト実行部１０７は、第２プログラム１２１のテストを終了し、その後、テストの終了を制御部１０６に通知する（１５１２）。

[0174] ステップ１５１２の後、制御部１０６は、テスト実行中フラグに″ＦＡＬＳＥ″を格納する指示を対象メソッド記録部１１８に送信する。対象メソッド記録部１１８は、テスト実行中フラグに″ＦＡＬＳＥ″を格納する指示を制御部１０６から受信した後、テスト実行中フラグに″ＦＡＬＳＥ″を格納する（１５１３）。

[0175] ステップ１５１３の後、制御部１０６は、第２プログラム１２１を終了させる指示をバージョン切替部１１０に送信し（１５１４）、図１２Ａに示す処理を終了する。

[0176] 図１２Ｂは、本発明の第１の実施形態の測定対象メソッドの実行時間を測定する処理を示すフローチャートである。

[0177] 図１２Ｂに示す処理は、第１プログラム１２０が実行された際の測定対象メソッドの実行時間を測定する処理を示す。なお、第２プログラム１２１が実行された際の測定対象メソッドの実行時間を測定する処理は、図１２Ｂに示す処理に含まれる、第１プログラム１２０を第２プログラム１２１に変更し、第１メソッド呼出テーブル１１１を第２メソッド呼出テーブル１１２に変更し、第１実行情報テーブル１１４を第２実行情報テーブル１１５に変更した処理である。

[0178] まず、コールバック受信部１１９は、第１プログラム１２０のテストを実行中に、ＪＶＭＴＩ１２３からコールバックを受信し、コールバックが受信された時刻と、コールバックに含まれる各情報を対象メソッド記録部１１８に送信する（１５１４）。

[0179] 対象メソッド記録部１１８は、テスト実行中フラグに″ＴＲＵＥ″が格納されているか否かを判定する（１５１５）。テスト実行中フラグに″ＦＡＬＳＥ″が格納されている場合、テストが実行されていないため、実行時間を測定できないため、対象メソッド記録部１１８はステップ１５２６に移行し、コールバック受信部１１９に処理の終了を通知する。

[0180] なお、テスト実行中フラグには、第１プログラム１２０又は第２プログラム１２１のいずれがテストされているかを示す情報が付加されてもよく、対象メソッド記録部１１８は、テスト実行中フラグを参照することによって、第１実行情報テーブル１１４及び第１メソッド呼出テーブル１１１を用いて処理するか、第２実行情報テーブル１１５及び第２メソッド呼出テーブル１１２を用いて処理するかを判定してもよい。

[0181] テスト実行中フラグに″ＴＲＵＥ″が格納されている場合、対象メソッド記録部１１８は、コールバックが示すメソッドが実行されたスレッドのメソッド実行中フラグに″ＦＡＬＳＥ″が格納されているか否かを判定する（１５１６）。

[0182] メソッド実行中フラグは、対象メソッド記録部１１８によってあらかじめ保持される記憶領域である。あるスレッドのメソッド実行中フラグに″ＦＡＬＳＥ″が格納される場合、そのスレッドにおいて測定対象メソッドが実行されていないことを示し、あるスレッドのメソッド実行中フラグに″ＴＲＵＥ″が格納される場合、そのスレッドにおいて測定対象メソッドが実行されていることを示す。メソッド実行中フラグは、後述するステップ１５１９、及び、ステップ１５２２において更新される。

[0183] なお、対象メソッド記録部１１８は、メソッドが実行されるスレッドを、コールバックに含まれるメソッドを示す情報から取得してもよい。

[0184] ステップ１５１６において、メソッドが実行されるスレッドのメソッド実行中フラグに″ＦＡＬＳＥ″が格納されていると判定された場合、ステップ１５１４において受信されたコールバックはメソッドの開始を示すため、対象メソッド記録部１１８は、開始時刻を格納するためステップ１５１７に移行する。

[0185] ステップ１５１６の後、対象メソッド記録部１１８は、コールバックが示すメソッドが、対象メソッドテーブル１１３に格納されているか否かを判定する（１５１７）。コールバックが示すメソッドが、対象メソッドテーブル１１３に格納されていない場合、コールバックが示すメソッドは測定対象メソッドではないため、対象メソッド記録部１１８は、ステップ１５２６に移行する。

[0186] ステップ１５１７において、コールバックが示すメソッドが対象メソッドテーブル１１３に格納されていると判定された場合、コールバックが示すメソッドは測定対象メソッドであるため、対象メソッド記録部１１８は、コールバックに含まれるメソッドを示す情報と、コールバックに含まれる時刻（すなわち、メソッドの開始時刻）とを第１実行情報テーブル１１４に格納する（１５１８）。

[0187] ステップ１５１８の後、対象メソッド記録部１１８は、コールバックが示すメソッドが実行されるスレッドのメソッド実行中フラグに、″ＴＲＵＥ″を

格納する。また、対象メソッド記録部１１８は、コールバックに含まれる、メソッドのコールスタックにおけるフレーム番号を、メソッドが実行されるスレッドの測定対象フレームに格納する（１５１９）。

[0188] 測定対象フレームは、対象メソッド記録部１１８によって保持される、フレーム番号を一時的に格納するための記憶領域である。測定対象フレームに格納される値は、測定対象メソッドのフレーム番号を示し、スレッドごとに保持される。

[0189] ステップ１５１９の後、対象メソッド記録部１１８は、ステップ１５２６に移行し、コールバック受信部１１９に処理の終了を通知する。

[0190] ステップ１５１６において、コールバックが示すメソッドが実行されるスレッドのメソッド実行中フラグに″ＴＲＵＥ″が格納されていると判定された場合、ステップ１５１４において受信されたコールバックはメソッドの終了を示すため、対象メソッド記録部１１８は、終了時刻を格納するためステップ１５２０に移行する。

[0191] 対象メソッド記録部１１８は、ステップ１５１６の後、コールバックが示すメソッドが実行されるスレッドの測定対象フレームに格納される番号が、コールバックが示すメソッドのフレーム番号と同じか否かを判定する（１５２０）。

[0192] 測定対象フレームに格納される番号が、コールバックが示すメソッドのフレーム番号と同じである場合、コールバックが示すメソッドは、テスト実行中に開始され、ステップ１５１８において開始時刻を格納されたメソッドである。このため、対象メソッド記録部１１８は、ステップ１５２１に移行する。

[0193] ステップ１５２０の後、対象メソッド記録部１１８は、コールバックに含まれるメソッドを示す情報と、コールバックに含まれる時刻（すなわち、メソッドの終了時刻）とを第１実行情報テーブル１１４に格納する（１５２１）。ステップ１５２１の後、対象メソッド記録部１１８は、メソッド実行中フラグに″ＦＡＬＳＥ″を格納する（１５２２）。

[0194] ステップ１５２２の後、対象メソッド記録部１１８は、ステップ１５２６に移行し、コールバック受信部１１９に処理の終了を通知する。

[0195] ステップ１５２０において、コールバックが示すメソッドが実行されるスレッドの測定対象フレームに格納される番号が、コールバックが示すメソッドのフレーム番号と異なる場合、コールバックが示すメソッドは、まだ終了していない測定対象メソッドによって呼ばれたメソッド、又は、測定対象メソッドによって呼ばれたメソッドからさらに呼ばれたメソッドである。このため、対象メソッド記録部１１８は、ステップ１５２３に移行する。

[0196] ステップ１５２０の後、対象メソッド記録部１１８は、測定対象フレームに格納される値が、コールバックが示すメソッドのフレーム番号から１を減算した値か否かを判定する。すなわち、コールバックが示すメソッドが、測定対象メソッドによって直接呼ばれたメソッドであるか否かを判定する（１５２３）。

[0197] 測定対象フレームに格納される値が、コールバックが示すメソッドのフレーム番号から１を減算した値ではない場合、コールバックが示すメソッドは次の測定対象メソッドではないため、対象メソッド記録部１１８は、ステップ１５２６に移行する。

[0198] 測定対象フレームに格納される値が、コールバックが示すメソッドのフレーム番号から１を減算した値である場合、対象メソッド記録部１１８は、コールバックがメソッドの開始を示すか否かを判定する（１５２４）。ステップ１５２４において、コールバックがメソッドの開始を示さず、終了を示す場合、対象メソッド記録部１１８は、ステップ１６２６に移行する。

[0199] ステップ１５２４において、コールバックがメソッドの開始を示す場合、コールバックが示すメソッドは次の測定対象メソッドであるため、対象メソッド記録部１１８は、コールバックに含まれるメソッドを示す情報を、第１メソッド呼出テーブル１１１に格納する（１５２５）。

[0200] ここで、第１メソッド呼出テーブル１１１及び第２メソッド呼出テーブル１１２に格納された値は、図１２Ａに示すテストが実行される前に削除され

ていてもよい。すなわち、ステップ１５２５によって、次の測定対象メソッドになる可能性があるメソッドのみが第１メソッド呼出テーブル１１１及び第２メソッド呼出テーブル１１２に格納されるように、第１メソッド呼出テーブル１１１及び第２メソッド呼出テーブル１１２に含まれる前回の初期メソッド又は測定対象メソッドを示す値は、図１２Ａに示すテスト開始時に削除されてもよい。

[0201] その後、対象メソッド記録部１１８は、ステップ１５２６に移行する。

[0202] ステップ１５２６において、コールバック受信部１１９は、次のコールバックの受信まで待機する。

[0203] 図１２Ｂに示す処理によって、対象メソッド記録部１１８は、測定対象メソッドの実行時間を測定し、次の測定対象メソッドを抽出する。

[0204] 図１３は、本発明の第１の実施形態の性能差要因メソッドを抽出する処理を示すフローチャートである。

[0205] 図１３に示す処理は、図７に示すステップ１６０に相当する。ステップ１５０、すなわち、図１２Ａに示す処理が行われる毎に、ステップ１６０は実行される。

[0206] 制御部１０６は、性能差要因メソッドを抽出する指示を性能差要因メソッド判定部１０９に送信する（１６０１）。

[0207] 性能差要因メソッド判定部１０９は、性能差要因メソッドを抽出する指示を制御部１０６から受信した後、変数ｉと最大性能差要因率とを格納する一時的な記憶領域をメモリ１０２に保持する。そして、変数ｉに、１を格納し、最大性能差要因率に、０を格納する（１６０２）。

[0208] 変数ｉは、対象メソッドテーブル１１３の行を示すポインタであり、最大性能差要因率は、後述する性能差要因率の最大値を示す。

[0209] ステップ１６０２の後、性能差要因メソッド判定部１０９は、ステップ１６０４からステップ１６１３までの処理を、変数ｉに格納された値が対象メソッドテーブル１１３の行数を超えるまで、繰り返す。

[0210] ステップ１６０２の後、性能差要因メソッド判定部１０９は、第１プログ

ラム１２０における測定対象メソッドの平均実行時間１１６３１と実行回数１１６３２とを第１実行情報テーブル１１４に基づいて、算出する。

[0211] 具体的には、性能差要因メソッド判定部１０９は、対象メソッドテーブル１１３から測定対象メソッドを一つ抽出し、出力装置１０４のうち、メソッド情報１１４２の値が測定対象メソッドを示す行を抽出する。そして、抽出された行からさらに、開始終了情報１１４３が"開始"と"終了"とを各々示す連続した二つの行を抽出する。そして、抽出された二つの行の組の数を算出することによって、実行回数１１６３２を算出する。

[0212] また性能差要因メソッド判定部１０９は、ステップ１６０４において、抽出された二つの行の時刻１１４４の差を算出することによって、１回あたりのメソッドの実行時間を算出する。そして、抽出された二つの行の組の各々の実行時間を算出し、すべての組の実行時間の平均値を算出することによって、平均実行時間１１６３１を算出する。

[0213] ステップ１６０４の後、性能差要因メソッド判定部１０９は、第２プログラム１２１における測定対象メソッドの平均実行時間１１６４１と実行回数１１６４２とを第２実行情報テーブル１１５に基づいて、算出する（１６０５）。平均実行時間１１６４１と実行回数１１６４２の具体的な算出方法は、平均実行時間１１６３１と実行回数１１６３２の算出方法と同様であるが、平均実行時間１１６４１と実行回数１１６４２は、第１実行情報テーブル１１４の代わりに第２実行情報テーブル１１５に基づいて算出される。

[0214] ステップ１６０５の後、性能差要因メソッド判定部１０９は、実行時間差１１６５及び実行時間比１１６６を算出する（ステップ１６０６）。具体的には、性能差要因メソッド判定部１０９は、ステップ１６０５において算出された各組の実行時間の合計を実行回数１１６３２によって除算した値から、ステップ１６０４において算出された各組の実行時間の合計を実行回数１１６４２によって除算した値を減算することによって、実行時間差１１６５を算出する。

[0215] また、性能差要因メソッド判定部１０９は、ステップ１６０６において、

算出された実行時間差１１６５を、さらに実行回数１１６３２によって除算し、除算された結果を、さらに平均実行時間１１６３１によって除算することによって、実行時間比１１６６を算出する。

[0216] ステップ１６０６の後、性能差要因メソッド判定部１０９は、ステップ１６０４、ステップ１６０５、及び、ステップ１６０６において算出された平均実行時間１１６３１、実行回数１１６３２、平均実行時間１１６４１、実行回数１１６４２、実行時間差１１６５、及び、実行時間比１１６６を、測定結果テーブル１１６に格納する（１６０７）。

[0217] なお、ステップ１６０４からステップ１６０７の処理は、対象メソッドテーブル１１３に含まれる全てのメソッドに対して実施されてもよい。

[0218] ステップ１６０７の後、性能差要因メソッド判定部１０９は、対象メソッドテーブル１１３の変数ｉが示す行のメソッド情報１１３２に対応する測定結果テーブル１１６のメソッド情報１１６２と、実行時間差１１６５と、実行時間比１１６６とを抽出する（１６０８）。

[0219] ステップ１６０８の後、性能差要因メソッド判定部１０９は、ステップ１６０８において抽出された実行時間比１１６６が、あらかじめユーザによって指定された実行時間比閾値以上であるか否かを判定する（１６０９）。実行時間比１１６６が実行時間比閾値以上でない場合、第１プログラム１２０と第２プログラム１２１とにおける測定対象メソッドの性能差は、ユーザにとって問題ない範囲であるため、性能差要因メソッド判定部１０９は、ステップ１６１３に移行する。

[0220] 実行時間比１１６６が実行時間比閾値以上である場合、第１プログラム１２０と第２プログラム１２１とにおける測定対象メソッドの性能差は、ユーザにとって問題がある範囲であるため、性能差要因メソッド判定部１０９は、ステップ１６１０に移行する。

[0221] ステップ１６０９の後、性能差要因メソッド判定部１０９は、変数ｉが示す行の呼出元メソッド情報１１３３を対象メソッドテーブル１１３から抽出し、抽出された呼出元メソッド情報１１３３に対応するメソッド情報１１６

２の実行時間差１１６５を、測定結果テーブル１１６から抽出する。そして、性能差要因メソッド判定部１０９は、抽出された実行時間差１１６５を、呼出元実行時間差に格納する（１６１０）。呼出元実行時間差とは、性能差要因メソッド判定部１０９によって保持される、メモリ１０２の中の一部の記憶領域である。

[0222] ステップ１６１０の後、性能差要因メソッド判定部１０９は、ステップ１６０８において抽出された実行時間差１１６５を、呼出元実行時間差によって除算する。そして、除算された結果を、性能差要因率に格納する（１６１１）。

[0223] 性能差要因率とは、性能差要因メソッド判定部１０９によって保持される、メモリ１０２の中の一部の記憶領域である。性能差要因率に格納される値が大きい程、呼出元メソッドの性能差から変化が大きいことを示す。すなわち、呼出元メソッドの性能差よりも、性能差が大きいメソッドは、第１プログラム１２０と第２プログラム１２１との性能差を生じさせるメソッドである可能性が高いため、性能差要因率が大きいほど、性能差を生じさせるメソッドである可能性が高い。

[0224] ステップ１６１１の後、性能差要因メソッド判定部１０９は、ステップ１６１１において算出された性能差要因率が示す値よりも、最大性能差要因率が示す値のほうが大きいか否かを判定する（１６１２）。

[0225] 最大性能差要因率とは、性能差要因メソッド判定部１０９によって保持される、メモリ１０２の中の一部の記憶領域である。最大性能差要因率は、対象メソッドテーブル１１３の変数ｉが示す各行に、ステップ１６０８～ステップ１６１１を実行することによって算出される性能差要因率の中で、最も大きい値を示す性能差要因率の値を格納するための領域である。

[0226] ステップ１６１２において、ステップ１６１１において算出された性能差要因率が示す値より、最大性能差要因率が示す値は大きくない場合、最大性能差要因率が示す値は変更されないため、性能差要因メソッド判定部１０９は、ステップ１６１３に移行する。

[0227] ステップ１６１２において、ステップ１６１１において算出された性能差要因率が示す値よりも、最大性能差要因率が示す値が大きい場合、性能差要因メソッド判定部１０９は、ステップ１６１１において算出された性能差要因率によって最大性能差要因率を更新する。また、変数ｉが示す対象メソッドテーブル１１３の行のメソッド情報１１３２の値によって、性能差要因メソッドを更新する（１６１３）。ステップ１６１２及びステップ１６１３の処理によって、性能差要因メソッドが抽出される。

[0228] ステップ１６０９、ステップ１６１２、又は、ステップ１６１３の後、性能差要因メソッド判定部１０９は、変数ｉに格納される値に、１を加算する。すなわち、ステップ１６０４～ステップ１６１３に示す処理を行う対象メソッドテーブル１１３の行を、一つ移動させる（１６１３）。

[0229] 対象メソッドテーブル１１３のすべての行に、ステップ１６０４～ステップ１６１３の処理を行った後、性能差要因メソッド判定部１０９は、性能差要因メソッドの抽出処理が終了したことを制御部１０６に通知する（１６１４）。

[0230] なお、図１３のステップ１６０９に示す処理において、実行時間比１１６６を算出することによって、第１のプログラムと第２のプログラムとの性能差を比較したが、実行時間差１１６５を比較することによって性能差を比較してもよい。すなわち、呼出元メソッドからの呼出回数が一定している場合などは、実行時間差１１６５によって性能差を比較してもよい。

[0231] 図１３のに示す処理、すなわち、図７に示すステップ１６０が終了した後、性能差要因メソッドに値が格納されている場合、図１４に示す処理、すなわち、ステップ１８０が実行される。

[0232] 図１４は、本発明の第１の実施形態の測定対象メソッドを再度設定する処理を示すフローチャートである。

[0233] 図１４に示す処理は、図７のステップ１８０に示す処理に相当する。

[0234] ステップ１７０の後、制御部１０６は、測定対象メソッドを再度設定する指示を、メソッド呼出照合部１０８に送信する（１８０１）。

[0235] ステップ１８０１の後、メソッド呼出照合部１０８は、第１メソッド呼出テーブル１１１を読み込み（１８０２）、第２メソッド呼出テーブル１１２を読み込む（１８０３）。そして、対象メソッドテーブル１１３に格納された値を削除する（１８０４）。

[0236] ここで、第１メソッド呼出テーブル１１１及び第２メソッド呼出テーブル１１２には、図１２Ｂのステップ１５２５において格納されたメソッド情報１１３２が保持される。すなわち、次回の測定対象メソッドになる可能性があるメソッドを示す情報が保持される。

[0237] ステップ１８０４の後、メソッド呼出照合部１０８は、第１メソッド呼出テーブル１１１と第２メソッド呼出テーブル１１２とに共通して保持されるメソッド情報（１１１２及び１１２２）のうち、呼出元メソッド情報（１１１３及び１１２３）が示すメソッドが、図１３に示す処理によって抽出された性能差要因メソッドであるメソッド情報（１１１２及び１１２２）を、対象メソッドテーブル１１３に格納する（１８０５）。

[0238] ステップ１８０５の後、メソッド呼出照合部１０８は、測定対象メソッドの再設定の終了を制御部１０６に通知する（１８０６）。

[0239] 図１４に示す処理の後、図７に示すステップ１５０からステップ１８０までの処理が、性能差要因メソッドが抽出されなくなるまで、繰り返される。

[0240] 第１の実施形態によれば、例えば、性能に影響のある修正が加えられたプログラムと、修正前のプログラムとの性能を比較することができる。また、修正前のプログラムの実行時間と修正後のプログラムの実行時間とに大きな差がない場合も、メソッドごとに実行時間を比較することによって、性能に著しい差が生じているメソッドを抽出できる。

[0241] また第１の実施形態によれば、例えば、アプリケーションサーバの分野においては、アプリケーションサーバのバージョンを固定すれば、２つのユーザアプリケーション間の性能差要因を分析でき、ユーザアプリケーションを固定すれば、アプリケーションサーバのバージョン間の性能差要因を分析できる。

[0242] さらに、第１の実施形態によれば、実行時間を測定するメソッドを抽出するため、実行時間の測定による負荷への影響が軽減され、性能に問題のある処理が含まれるメソッドを高い精度で検出することができる。

[0243] 図１５は、本発明の第２の実施形態の単一のプログラムの性能差を分析する計算機１００の構成を示すブロック図である。

[0244] 第２の実施形態の計算機１００は、一つのプログラムのテストを実行し、一つのプログラムにおける性能を測定する。そして、ユーザによってあらかじめ指定された閾値と、測定された性能を比較し、性能差要因メソッドを抽出する。

[0245] 第２の実施形態の計算機１００は、第１の実施形態の計算機１００と同じ、メモリ１０２、演算装置１０１、外部記憶装置１０３、出力装置１０４、及び、入力装置１０５を備える。

[0246] また、第２の実施形態の制御部３００、テスト実行部３０１、メソッド呼出照合部３０２、性能差要因メソッド判定部３０３、初期メソッド記録部３０８、対象メソッド記録部３０９、コールバック受信部３１０、ＪａｖａＶＭ３１２、及び、ＪＶＭＴＩ３１３は、第１の実施形態の制御部１０６、テスト実行部１０７、メソッド呼出照合部１０８、性能差要因メソッド判定部１０９、バージョン切替部１１０、初期メソッド記録部１１７、対象メソッド記録部１１８、コールバック受信部１１９、ＪａｖａＶＭ１２２、及び、ＪＶＭＴＩ１２３と各々同様な機能を持つ。また、第２の実施形態の対象メソッドテーブル３０５及び測定結果テーブル３０７は、第１の実施形態の対象メソッドテーブル１１３、測定結果テーブル１１６と同じ情報を含む。

[0247] しかし、第２の実施形態の制御部３００は、一つの測定対象プログラム３１１のみをテストするため、第１メソッド呼出テーブル１１１に対応するメソッド呼出テーブル３０４と、第１実行情報テーブル１１４に対応する実行情報テーブル３０６は、各々一つである。

[0248] 図１６は、本発明の第２の実施形態のメソッド呼出テーブル３０４を示す説明図である。

[0249] メソッド呼出テーブル３０４は、第１の実施形態の第１メソッド呼出テーブル１１１及び第２メソッド呼出テーブル１１２と同じ情報を含み、フレーム番号３０４１、メソッド情報３０４２及び呼出元メソッド情報３０４３は、第１の実施形態のフレーム番号１１１１、メソッド情報１１１２及び呼出元メソッド情報１１１３に対応する。

[0250] 図１７は、本発明の第２の実施形態の実行情報テーブル３０６を示す説明図である。

[0251] 実行情報テーブル３０６は、第１の実施形態の第１実行情報テーブル１１４及び第２実行情報テーブル１１５と同じ情報を含み、フレーム番号３０６１、メソッド情報３０６２、開始終了情報３０６３及び時刻３０６４は、第１の実施形態のフレーム番号１１４１、メソッド情報１１４２、開始終了情報１１４３及び時刻１１４４に対応する。

[0252] 図１８は、本発明の第２の実施形態の測定結果テーブル３０７を示す説明図である。

[0253] 測定結果テーブル３０７は、フレーム番号３０７１、メソッド情報３０７２、平均実行時間３０７３及び実行回数３０７４を含む。フレーム番号３０７１、メソッド情報３０７２、平均実行時間３０７３及び実行回数３０７４は、第１の実施形態の測定結果テーブル１１６のフレーム番号１１６１、メソッド情報１１６２、平均実行時間１１６３１及び実行回数１１６３２に対応する。

[0254] 以下に、第１の実施形態の処理と異なる処理について説明する。

[0255] 図１９は、本発明の第２の実施形態の全体の処理を示すフローチャートである。

[0256] 制御部３００は、まず、測定対象プログラム３１１の初期メソッドを抽出し、抽出された初期メソッドを対象メソッドテーブル３０５に格納する。これによって、測定対象メソッドが対象メソッドテーブル３０５に格納される（３２０）。

[0257] 制御部３００は、ステップ３２０において、図１０Ａ～図１０Ｃに示す処

理を、測定対象プログラム３１１のみに実行する。具体的には、制御部３００は、図１０Ａに示すステップ１４０２からステップ１４０９までをテスト実行部３０１に実行させることによって、測定対象プログラム３１１のテストを実行する。図１０Ａに示す処理が実行される際、制御部３００は、コールバック受信部３１０及び初期メソッド記録部３０８によって、図１０Ｃに示す処理を測定対象プログラム３１１のみに実行する。

[0258] さらにステップ３２０において制御部３００は、メソッド呼出照合部３０２によって、図１０Ｂに示すステップ１４２１、ステップ１４２３及びステップ１４２４を測定対象プログラム３１１に実行する。ここで、ステップ１４２４において、メソッド呼出照合部３０２は、メソッド呼出テーブルに含まれるメソッド情報３０４２を、すべて対象メソッドテーブル３０５に格納する。

[0259] ステップ３２０の後、制御部３００は、測定対象メソッドの実行時間を測定する（３３０）。制御部３００は、ステップ３３０において、図１２Ａ及び図１２Ｂに示す処理を測定対象プログラム３１１にのみ実行する。

[0260] 具体的には、制御部３００は、ステップ３３０において、ステップ１５０２からステップ１５０７までをテスト実行部３０１に実行させることによって、測定対象プログラム３１１のテストを実行する。そして、測定対象プログラム３１１をテストしている間に、対象メソッド記録部３０９によって、図１２Ｂに示す処理を測定対象プログラム３１１に実行する。

[0261] ステップ３３０の後、制御部３００は、所定の条件を満たすメソッドを抽出する。

[0262] 図２０は、本発明の第２の実施形態の所定の条件を満たすメソッドを抽出する処理を示すフローチャートである。

[0263] 図２０に示す処理は、図１９に示すステップ３４０に相当する。

[0264] 制御部３００は、所定の条件を満たすメソッドを抽出する指示を性能差要因メソッド判定部３０３に送信する（３４０１）。

[0265] ステップ３４０１の後、性能差要因メソッド判定部３０３は、変数ｉをメ

モリ１０２に保持し、変数ｉに１を格納する（３４０２）。変数ｉは、対象メソッドテーブル３０５の行を示すポインタである。

[0266] ステップ３４０２の後、性能差要因メソッド判定部３０３は、ステップ３４０３からステップ３４０８までの処理を、変数ｉに格納された値が対象メソッドテーブル３０５の行数を超えるまで、繰り返す（３４０２）。

[0267] ステップ１６０２の後、性能差要因メソッド判定部３０３は、測定対象メソッドの平均実行時間３０７３と実行回数３０７４とを実行情報テーブル３０６に基づいて、算出する（３４０３）。平均実行時間３０７３と実行回数３０７４との算出方法は、第１の実施形態の平均実行時間１１６３１及び実行回数１１６３２の算出方法と同じである。

[0268] ステップ３４０３の後、性能差要因メソッド判定部３０３は、算出された平均実行時間３０７３と実行回数３０７４とを、測定結果テーブル３０７に格納する（３４０４）。

[0269] ステップ３４０４の後、性能差要因メソッド判定部３０３は、平均実行時間３０７３及び実行回数３０７４を乗じた値が、ユーザによってあらかじめ指定された閾値よりも大きいか否かを判定する（３４０５）。平均実行時間３０７３及び実行回数３０７４を乗じた値が、ユーザによってあらかじめ指定された閾値を超えない場合、変数ｉが示すメソッドには問題がないため、性能差要因メソッド判定部３０３は、ステップ３４０８に移行する。

[0270] ステップ３４０５において、平均実行時間３０７３及び実行回数３０７４を乗じた値が、ユーザによってあらかじめ指定された閾値よりも大きいと判定された場合、変数ｉが示すメソッドに問題がある可能性があるため、性能差要因メソッド判定部３０３は、ステップ３４０６に移行する。

[0271] ステップ３４０５の後、性能差要因メソッド判定部３０３は、平均実行時間３０７３及び実行回数３０７４を乗じた値が、最大実行時間メソッドの平均実行時間３０７３及び実行回数３０７４を乗じた値よりも大きいか否かを判定する（３４０７）。最大実行時間メソッドは、性能差要因メソッド判定部３０３によってメモリ１０２に保持される記憶領域であり、所定の条件を

満たすメソッドのうち、平均実行時間３０７３及び実行回数３０７４を乗じた値が最も大きいメソッドを示す情報を格納するための領域である。

[0272] 平均実行時間３０７３及び実行回数３０７４を乗じた値が、最大実行時間メソッドの平均実行時間３０７３及び実行回数３０７４を乗じた値を超えない場合、性能差要因メソッド判定部３０３は、ステップ３４０９に移行する。

[0273] 平均実行時間３０７３及び実行回数３０７４を乗じた値が、最大実行時間メソッドの平均実行時間３０７３及び実行回数３０７４を乗じた値よりも大きい場合、変数ｉが示すメソッドによって、最大実行時間メソッドを更新する（３４０８）。

[0274] ステップ３４０６、ステップ３４０７、又は、ステップ３４０８の後、性能差要因メソッド判定部３０３は、変数ｉに格納される値に、１を加算する（３４０９）。これによって、性能差要因メソッド判定部３０３は、対象メソッドテーブル３０５の次の行を取得することができる。

[0275] ステップ３４０３の繰り返し処理が終了した後、性能差要因メソッド判定部３０３は、所定の条件を満たすメソッドを抽出する処理の終了を、制御部３００に通知する（３４１０）。

[0276] なお、前述の第２の実施形態において、図１０Ａ～図１０Ｃ、図１２Ａ、図１２Ｂ及び図１４に示す処理のうち、第１メソッド呼出テーブル１１１及び第１実行情報テーブル１１４は、メソッド呼出テーブル３０４及び対象メソッド記録部３０９に置き換える。

[0277] また、本実施形態の計算機１００は、第１の実施形態と第２の実施形態との両方を実行することが可能である。すなわち、ユーザが、一つのプログラムの性能を測定するか、又は、二つのプログラムの性能を比較するかをあらかじめ指定し、制御部１０６又は制御部３００が、ユーザからの指定をに基づいて、第１の実施形態を実行するか、又は、第２の実施形態を実行するかを選択してよい。

[0278] また、前述の第１の実施形態及び第２の実施形態における初期メソッドは

、プログラムを実行することによって自動的に抽出されたが、ユーザによって指定されてもよい。

[0279] 図２１は、本発明の第２の実施形態の初期メソッドを指定するための画面を示す説明図である。

[0280] 図２１に示す画面４００は、第２の実施形態の初期メソッドを指定するための画面であるが、第１の実施形態において用いられてもよい。

[0281] 画面４００は、制御部３００（又は制御部１０６）が出力装置１０４を介してユーザに表示する画面である。画面４００は、選択領域４０１、選択領域４０２、指示領域４０３を含む。ユーザは、出力装置１０４に表示された画面４００を、入力装置１０５によって操作することによって、画面４００に示された情報を指定する。

[0282] 選択領域４０１は、初期メソッドを抽出する方法を選択するためのボタン等を含む。図２１に示す選択領域４０１において、″自動″ボタンが選択された場合、制御部３００（又は制御部１０６）は、ステップ３２０によって、初期メソッドを自動的に抽出する。″手動″ボタンが選択された場合、制御部３００（又は制御部１０６は、ステップ３２０（又は１４０）を実行せず、ユーザによって指定された初期メソッドを対象メソッドテーブル３０５（又は対象メソッドテーブル１１３）に格納する。

[0283] 選択領域４０２は、選択領域４０１において″手動″が選択された場合において、初期メソッドを指定するためのチェックボックス等を含む。図２１に示す選択領域４０２には、プログラムによって呼ばれるメソッドが階層構造によって表示される。選択領域４０２に表示される各メソッドは、チェックボックスによって選択可能である。

[0284] ユーザは、選択領域４０１及び選択領域４０２において、初期メソッドを抽出するための情報を指定すると、指示領域４０３に含まれるボタン等を操作する。その後、制御部３００（又は制御部１０６）は、画面４００によって指定された情報に基づいて、図１９（又は図７）に示す処理を開始する。

[0285] 第２の実施形態によれば、一つのプログラムにおいて、性能に影響がある

メソッド（最大実行時間メソッド）を自動的に抽出することができる。また、第１の実施形態と同じく、実行時間を測定するメソッドを抽出するため、プログラムの規模によらず、精度よく最大実行時間メソッドを抽出することができる。

[0286] さらに、本実施形態は、ユーザによって初期メソッドを指定することが可能であるため、実行時間を測定したいプログラムの一部を、限定的に測定することが可能である。例えば、プログラムの一部が修正され、すべてのプログラムの性能を測定する必要がない場合、ユーザは、性能を測定したいプログラムの一部を最初に実行するメソッドを、初期メソッドに指定する。これによって、短時間で性能を測定することが可能であり、消費電力を低減できる。

[0287] なお、本発明は、前記所定の条件に、呼び出し元メソッドの実行時間及び実行回数を判定条件に用いることもできるため、例えば、呼び出し元メソッドの実行時間に対する割合と所定の閾値とを比較することによって、性能差要因メソッド又は最大実行時間メソッドを抽出してもよい。

[0288] 以上、本発明の実施に関わる性能分析装置、方法、及びプログラムについて説明したが、本発明はこのような具体的構成に限定されるものではなく、添付した請求の範囲の趣旨内における様々な変更及び同等の構成を含むものである。

産業上の利用可能性

[0289] 本発明は、アプリケーションサーバにおいてプログラムの性能を測定するシステムに適用可能である。

請求の範囲

[請求項1] 計算機によってプログラムの性能を測定する性能測定方法であって、

前記計算機は、プロセッサと、前記プロセッサによって実行されるプログラムを格納するメモリとを有し、

前記方法は、

前記計算機が、前記性能が測定されるプログラムによって直接呼ばれるメソッドのうち、少なくとも一つの第１のメソッドを最初に決定する手順と、

前記計算機が、前記プログラムを実行することによって、前記決定された各第１のメソッドの実行時間を測定する手順と、

前記計算機が、前記測定された各第１のメソッドの実行時間を用いて、所定の条件に一致する少なくとも一つの第２のメソッドが前記各第１のメソッドから抽出可能か否かを判定する手順と、

前記少なくとも一つの第２のメソッドが抽出可能である場合、前記計算機が、前記抽出された各第２のメソッドから直接呼ばれる少なくとも一つの第３のメソッドが抽出可能か否かを判定する手順と、

前記少なくとも一つの第３のメソッドが抽出可能である場合、前記計算機が、前記抽出された第３のメソッドを、第１のメソッドに決定する手順と、

前記計算機が、前記実行時間を測定する手順と、前記第２のメソッドが抽出可能か否かを判定する手順と、前記第３のメソッドが抽出可能か否かを判定する手順と、前記第３のメソッドを第１のメソッドに決定する手順とを繰り返し実行する手順とを含むことを特徴とする性能測定方法。

[請求項2] 請求項１に記載の性能測定方法であって、

前記性能が測定されるプログラムは、第１のプログラム及び第２のプログラムを含み、

前記少なくとも一つの第１のメソッドを最初に決定する手順は、前記計算機が、前記第１のプログラム及び第２のプログラムの両方によって直接呼ばれるメソッドのうち、少なくとも一つの第１のメソッドを決定する手順を含み、

前記実行時間を測定する手順は、前記計算機が、前記各第１のメソッドの前記第１のプログラムにおける第１の実行時間と、前記各第１のメソッドの前記第２のプログラムにおける第２の実行時間とを測定する手順を含み、

前記第２のメソッドが抽出可能か否かを判定する手順は、前記計算機が、前記測定された各第１の実行時間と各第２の実行時間とを用いて、前記所定の条件に一致する第２のメソッドが前記第１のメソッドから抽出可能か否かを判定する手順を含むことを特徴とする性能測定方法。

[請求項3] 請求項２に記載の性能測定方法であって、

前記少なくとも一つの第１のメソッドを最初に決定する手順は、

前記計算機が、前記第１のプログラムの開始後に開始し、前記第１のプログラムの終了前に終了するメソッドのうち、呼び出し元が前記第１のプログラムの開始前に開始されたか、または前記第１のプログラムの終了後に終了されたメソッドである、少なくとも一つの第４のメソッドを抽出する手順と、

前記計算機が、前記第２のプログラムの開始後に開始し、前記第２のプログラムの終了前に終了するメソッドのうち、呼び出し元が前記第２のプログラムの開始前に開始されたか、または前記第２のプログラムの終了後に終了されたメソッドである、少なくとも一つの第５のメソッドを抽出する手順と、

前記計算機が、前記第４のメソッドと前記第５のメソッドとのうち、同じメソッドを各第１のメソッドとして決定する手順とを含むことを特徴とする性能測定方法。

[請求項4] 請求項２又は３に記載の性能測定方法であって、

前記少なくとも一つの第２のメソッドが抽出可能か否かを判定する手順は、

前記計算機が、前記第１のメソッドを直接呼んだ第６のメソッドが、前記第１のプログラムにおいて実行された第１の実行回数と、前記第１のメソッドを直接呼んだ前記各第６のメソッドが前記第２のプログラムにおいて実行された第２の実行回数とを保持する手順と、

前記計算機が、前記第６のメソッドから呼ばれる第１のメソッドの、前記第１のプログラムにおける第１の合計実行時間と、前記第２のプログラムにおける第２の合計実行時間とを保持する手順と、

前記計算機が、前記第２の合計実行時間を前記第２の実行回数によって除算した結果から、前記第１の合計実行時間を前記第１の実行回数によって除算した結果を、減算することによって、実行時間差を算出する手順と、

前記計算機が、前記算出された実行時間差が、所定の条件に一致する前記第１のメソッドが第２のメソッドとして抽出可能か否かを判定する手順とを、含むことを特徴とする性能測定方法。

[請求項5] 請求項４に記載の性能測定方法であって、

前記実行時間差によって第２のメソッドが抽出可能か否かを判定する手順は、

前記計算機が、前記第１のメソッドが、前記第１のプログラムにおいて実行された第３の実行回数と、１回あたりの第１の平均実行時間とを保持する手順と、

前記計算機が、前記算出された実行時間差を、前記第３の実行回数と、前記第１の平均実行時間とによって除算することによって、実行時間比を算出する手順と、

前記計算機が、前記算出された実行時間比と、所定の閾値とを比較する手順と、

前記比較の結果、前記計算機が、前記算出された実行時間比が、所定の閾値より大きい前記第１のメソッドが第２のメソッドとして抽出可能か否かを判定する手順とを、含むことを特徴とする性能測定方法。

[請求項6] 請求項１から５のいずれか一つに記載の性能測定方法であって、

前記計算機は、ユーザが少なくとも一つのメソッドを指定するための入力装置を備え、

前記少なくとも一つの第１のメソッドを最初に決定する手順は、前記入力装置を介して、前記ユーザから各メソッドが指定された場合、前記計算機が、前記指定されたメソッドを、前記各第１のメソッドに決定する手順を含むことを特徴とする性能測定方法。

[請求項7] 請求項１から６のいずれか一つに記載の性能測定方法であって、

前記各第１のメソッドの実行時間を測定する手順は、

前記計算機が、前記各第１のメソッドが開始した時刻と、前記各第１のメソッドが終了した時刻とを取得する手順と、

前記計算機が、前記取得された各第１のメソッドが終了した時刻から、前記取得された各第１のメソッドが開始した時刻を減算することによって、前記各実行時間を算出する手順とを含むことを特徴とする性能測定方法。

[請求項8] プログラムの性能を測定する性能測定装置であって、

前記性能測定装置は、

プロセッサと、前記プロセッサによって実行されるプログラムを格納するメモリとを有し、

前記性能が測定されるプログラムによって直接呼ばれるメソッドのうち、少なくとも一つの第１のメソッドを最初に決定し、

前記プログラムを実行することによって、前記決定された各第１のメソッドの実行時間を測定し、

前記測定された各第１のメソッドの実行時間を用いて、所定の条件

に一致する少なくとも一つの第２のメソッドが前記各第１のメソッドから抽出可能か否かを判定し、

前記少なくとも一つの第２のメソッドが抽出可能である場合、前記抽出された各第２のメソッドから直接呼ばれる少なくとも一つの第３のメソッドが抽出可能か否かを判定し、

前記少なくとも一つの第３のメソッドが抽出可能である場合、前記抽出された第３のメソッドを、第１のメソッドに決定し、

前記実行時間を測定し、前記第２のメソッドが抽出可能か否かを判定し、前記第３のメソッドが抽出可能か否かを判定し、前記第３のメソッドを第１のメソッドに決定する手順を繰り返し実行することを特徴とする性能測定装置。

[請求項9] 請求項８に記載の性能測定装置であって、

前記性能が測定されるプログラムは、第１のプログラム及び第２のプログラムを含み、

前記性能測定装置は、

前記第１のプログラム及び第２のプログラムの両方によって直接呼ばれるメソッドのうち、少なくとも一つの第１のメソッドを決定し、

前記各第１のメソッドの前記第１のプログラムにおける第１の実行時間と、前記各第１のメソッドの前記第２のプログラムにおける第２の実行時間とを測定し、

前記測定された各第１の実行時間と各第２の実行時間とを用いて、前記所定の条件に一致する第２のメソッドが前記第１のメソッドから抽出可能か否かを判定することを特徴とする性能測定装置。

[請求項10] 請求項９に記載の性能測定装置であって、

前記性能測定装置は、

前記第１のプログラムの開始後に開始し、前記第１のプログラムの終了前に終了するメソッドのうち、呼び出し元が前記第１のプログラムの開始前に開始されたか、または前記第１のプログラムの終了後に

終了されたメソッドである、少なくとも一つの第４のメソッドを抽出し、

前記第２のプログラムの開始後に開始し、前記第２のプログラムの終了前に終了するメソッドのうち、呼び出し元が前記第２のプログラムの開始前に開始されたか、または前記第２のプログラムの終了後に終了されたメソッドである、少なくとも一つの第５のメソッドを抽出し、

前記第４のメソッドと前記第５のメソッドとのうち、同じメソッドを各第１のメソッドとして最初に決定する手順とを含むことを特徴とする性能測定装置。

[請求項11] 請求項９又は１０に記載の性能測定装置であって、

前記性能測定装置は、

前記第１のメソッドを直接呼んだ第６のメソッドが、前記第１のプログラムにおいて実行された第１の実行回数と、前記第１のメソッドを直接呼んだ前記各第６のメソッドが前記第２のプログラムにおいて実行された第２の実行回数とを保持し、

前記第６のメソッドから呼ばれる第１のメソッドの、前記第１のプログラムにおける第１の合計実行時間と、前記第２のプログラムにおける第２の合計実行時間とを保持し、

前記第２の合計実行時間を前記第２の実行回数によって除算した結果から、前記第１の合計実行時間を前記第１の実行回数によって除算した結果を、減算することによって、実行時間差を算出し、

前記算出された実行時間差が、所定の条件に一致する前記第１のメソッドが第２のメソッドとして抽出可能か否かを判定することを特徴とする性能測定装置。

[請求項12] 請求項１１に記載の性能測定装置であって、

前記性能測定装置は、

前記第１のメソッドが、前記第１のプログラムにおいて実行された

第３の実行回数と、１回あたりの第１の平均実行時間とを保持し、

前記算出された実行時間差を、前記第３の実行回数と、前記第１の平均実行時間とによって除算することによって、実行時間比を算出し、

前記算出された実行時間比と、所定の閾値とを比較し、

前記比較の結果、前記算出された実行時間比が、所定の閾値より大きい前記第１のメソッドが第２のメソッドとして抽出可能か否かを判定することを特徴とする性能測定装置。

[請求項13] 請求項８から１２のいずれか一つに記載の性能測定装置であって、

前記性能測定装置は、

ユーザが少なくとも一つのメソッドを指定するための入力装置を備え、

前記入力装置を介して、前記ユーザから各メソッドが指定された場合、前記指定されたメソッドを、前記各第１のメソッドに決定することを特徴とする性能測定装置。

[請求項14] 請求項８から１３のいずれか一つに記載の性能測定装置であって、

前記性能測定装置は、

前記前記各第１のメソッドが開始した時刻と、前記各第１のメソッドが終了した時刻とを取得し、

前記取得された各第１のメソッドが終了した時刻から、前記取得された各第１のメソッドが開始した時刻を減算することによって、前記各実行時間を算出することを特徴とする性能測定装置。

[請求項15] 計算機にプログラムの性能を測定させる性能測定プログラムであって、

前記計算機は、プロセッサと、前記プロセッサによって実行されるプログラムを格納するメモリとを有し、

前記性能測定プログラムは、

前記計算機に、前記性能が測定されるプログラムによって直接呼ば

れるメソッドのうち、少なくとも一つの第１のメソッドを最初に決定させる手順と、

前記計算機に、前記プログラムを実行することによって、前記決定された各第１のメソッドの実行時間を測定させる手順と、

前記計算機に、前記測定された各第１のメソッドの実行時間を用いて、所定の条件に一致する少なくとも一つの第２のメソッドが前記各第１のメソッドから抽出可能か否かを判定させる手順と、

前記少なくとも一つの第２のメソッドが抽出可能である場合、前記計算機に、前記抽出された各第２のメソッドから直接呼ばれる少なくとも一つの第３のメソッドが抽出可能か否かを判定させる手順と、

前記少なくとも一つの第３のメソッドが抽出可能である場合、前記計算機に、前記抽出された第３のメソッドを、第１のメソッドに決定させる手順と、

前記計算機に、前記実行時間を測定させる手順と、前記第２のメソッドが抽出可能か否かを判定させる手順と、前記第３のメソッドが抽出可能か否かを判定させる手順と、前記第３のメソッドを第１のメソッドに決定させる手順とを繰り返し実行させるための性能測定プログラム。

[請求項16] 請求項１５に記載の性能測定プログラムであって、

前記性能が測定されるプログラムは、第１のプログラム及び第２のプログラムを含み、

前記少なくとも一つの第１のメソッドを最初に決定させる手順において、前記計算機に、前記第１のプログラム及び第２のプログラムの両方によって直接呼ばれるメソッドのうち、少なくとも一つの第１のメソッドを決定させる手順と、

前記実行時間を測定させる手順において、前記計算機に、前記各第１のメソッドの前記第１のプログラムにおける第１の実行時間と、前記各第１のメソッドの前記第２のプログラムにおける第２の実行時間

とを測定させる手順と、

前記第２のメソッドが抽出可能か否かを判定させる手順において、前記計算機に、前記測定された各第１の実行時間と各第２の実行時間とを用いて、前記所定の条件に一致する第２のメソッドが前記第１のメソッドから抽出可能か否かを判定させる手順とを実行させるための性能測定プログラム。

[請求項17] 請求項１６に記載の性能測定プログラムであって、

前記少なくとも一つの第１のメソッドを最初に決定させる手順において、

前記計算機に、前記第１のプログラムの開始後に開始し、前記第１のプログラムの終了前に終了するメソッドのうち、呼び出し元が前記第１のプログラムの開始前に開始されたか、または前記第１のプログラムの終了後に終了されたメソッドである、少なくとも一つの第４のメソッドを抽出させる手順と、

前記計算機に、前記第２のプログラムの開始後に開始し、前記第２のプログラムの終了前に終了するメソッドのうち、呼び出し元が前記第２のプログラムの開始前に開始されたか、または前記第２のプログラムの終了後に終了されたメソッドである、少なくとも一つの第５のメソッドを抽出させる手順と、

前記計算機に、前記第４のメソッドと前記第５のメソッドとのうち、同じメソッドを各第１のメソッドとして決定させる手順とを実行させるための性能測定プログラム。

[請求項18] 請求項１５から１７のいずれか一つに記載の性能測定プログラムであって、

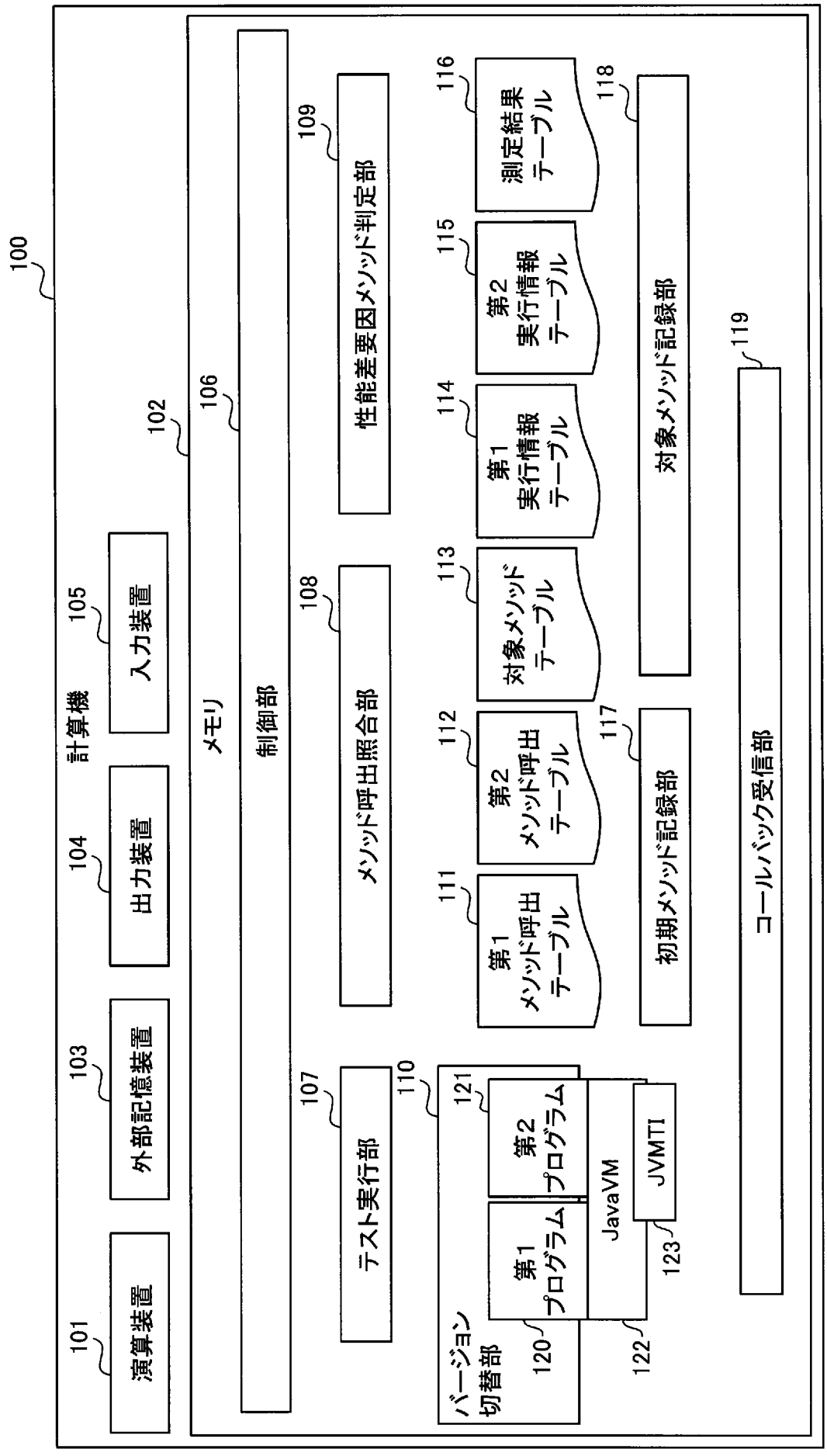
前記各第１のメソッドの実行時間を測定させる手順において、

前記計算機に、前記各第１のメソッドが開始した時刻と、前記各第１のメソッドが終了した時刻とを取得させる手順と、

前記計算機に、前記取得された各第１のメソッドが終了した時刻か

ら、前記取得された各第１のメソッドが開始した時刻を減算することによって、前記各実行時間を算出させる手順とを実行させるための性能測定プログラム。

[図1]


100
計算機
101
演算装置
103
外部記憶装置
104
出力装置
105
入力装置
102
メモリ
106
制御部
107
テスト実行部
108
メソッド呼出照合部
109
性能差要因メソッド判定部
110
バージョン切替部
120
第1プログラム
121
第2プログラム
122
JavaVM
123
JVMTI
111
第1メソッド呼出テーブル
112
第2メソッド呼出テーブル
113
対象メソッドテーブル
114
第1実行情報テーブル
115
第2実行情報テーブル
116
測定結果テーブル
117
初期メソッド記録部
118
対象メソッド記録部
119
コールバック受信部

[図2]

1131 1132 1133 113

フレーム番号	メソッド情報	呼出元メソッド情報
16	aaa.bbb.MyClass.start(int)	aaa.bbb.Starter.doStart()
16	aaa.bbb.OtherClass.get():int	aaa.bbb.Starter.doStart()
:	:	:

[図3A]

1111 1112 1113 111

フレーム番号	メソッド情報	呼出元メソッド情報
16	aaa.ccc.OldClass.setLen(int)	aaa.bbb.Starter.init()
16	aaa.Util.getName(String)	aaa.bbb.Starter.beforeStart()
16	aaa.bbb.MyClass.start(int)	aaa.bbb.Starter.doStart()
16	aaa.bbb.OtherClass.get():int	aaa.bbb.Starter.doStart()
:	:	:

[図3B]

1121 1122 1123 112

フレーム番号	メソッド情報	呼出元メソッド情報
16	aaa.ccc.OldClass.setLen(int)	aaa.bbb.Starter.init()
16	aaa.Util.getName(String)	aaa.bbb.Starter.beforeStart()
16	aaa.bbb.MyClass.start(int)	aaa.bbb.Starter.doStart()
16	aaa.bbb.MyClass.check(int)	aaa.bbb.Starter.doStart()
16	aaa.bbb.OtherClass.get():int	aaa.bbb.Starter.doStart()
:	:	:

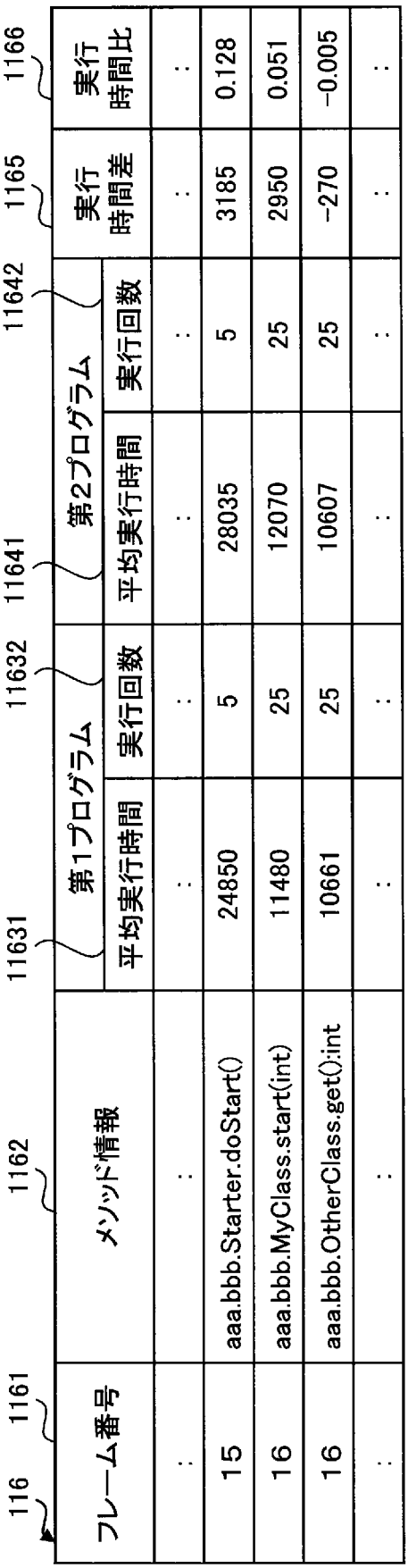
[図4A]

1141 フレーム番号	1142 メソッド情報	1143 開始/終了	1144 時刻 (114)
16	aaa.bbb.MyClass.start(int)	開始	753227897
16	aaa.bbb.MyClass.start(int)	終了	753239377
16	aaa.bbb.OtherClass.get():int	開始	753239399
16	aaa.bbb.OtherClass.get():int	終了	753250060
16	aaa.bbb.MyClass.start(int)	開始	753250082
:	:	:	:

[図4B]

1151 フレーム番号	1152 メソッド情報	1153 開始/終了	1154 時刻 (115)
16	aaa.bbb.MyClass.start(int)	開始	134176904
16	aaa.bbb.MyClass.start(int)	終了	134188974
16	aaa.bbb.OtherClass.get():int	開始	134189995
16	aaa.bbb.OtherClass.get():int	終了	134199602
16	aaa.bbb.MyClass.start(int)	開始	134199623
:	:	:	:

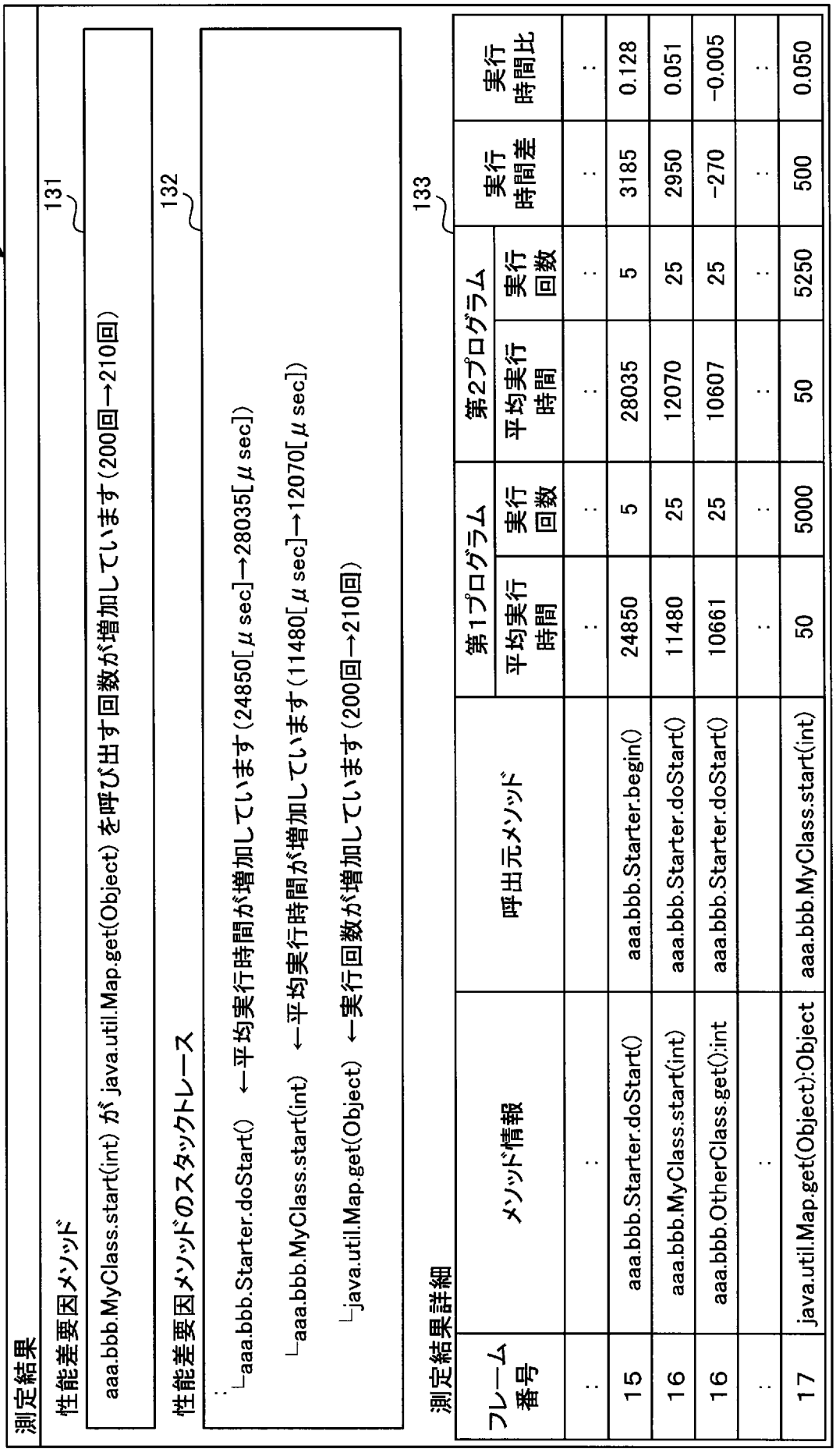
[図5]



フレーム番号 (116, 1161)	メソッド情報 (1162)	第1プログラム (11631)		第2プログラム (11641)		実行時間差 (1165)	実行時間比 (1166)
		平均実行時間	実行回数 (11632)	平均実行時間	実行回数 (11642)		
:	:	:	:	:	:	:	:
15	aaa.bbb.Starter.doStart()	24850	5	28035	5	3185	0.128
16	aaa.bbb.MyClass.start(int)	11480	25	12070	25	2950	0.051
16	aaa.bbb.OtherClass.get():int	10661	25	10607	25	-270	-0.005
:	:	:	:	:	:	:	:

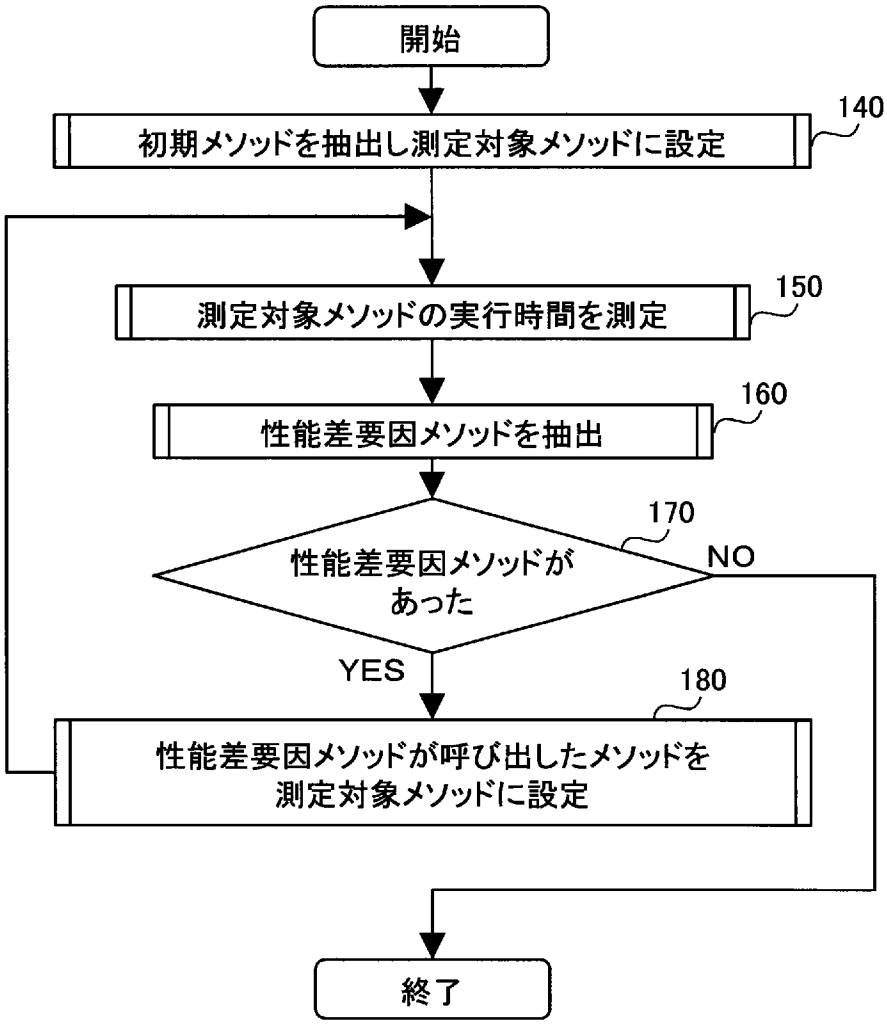
測定結果テーブル

[図6]

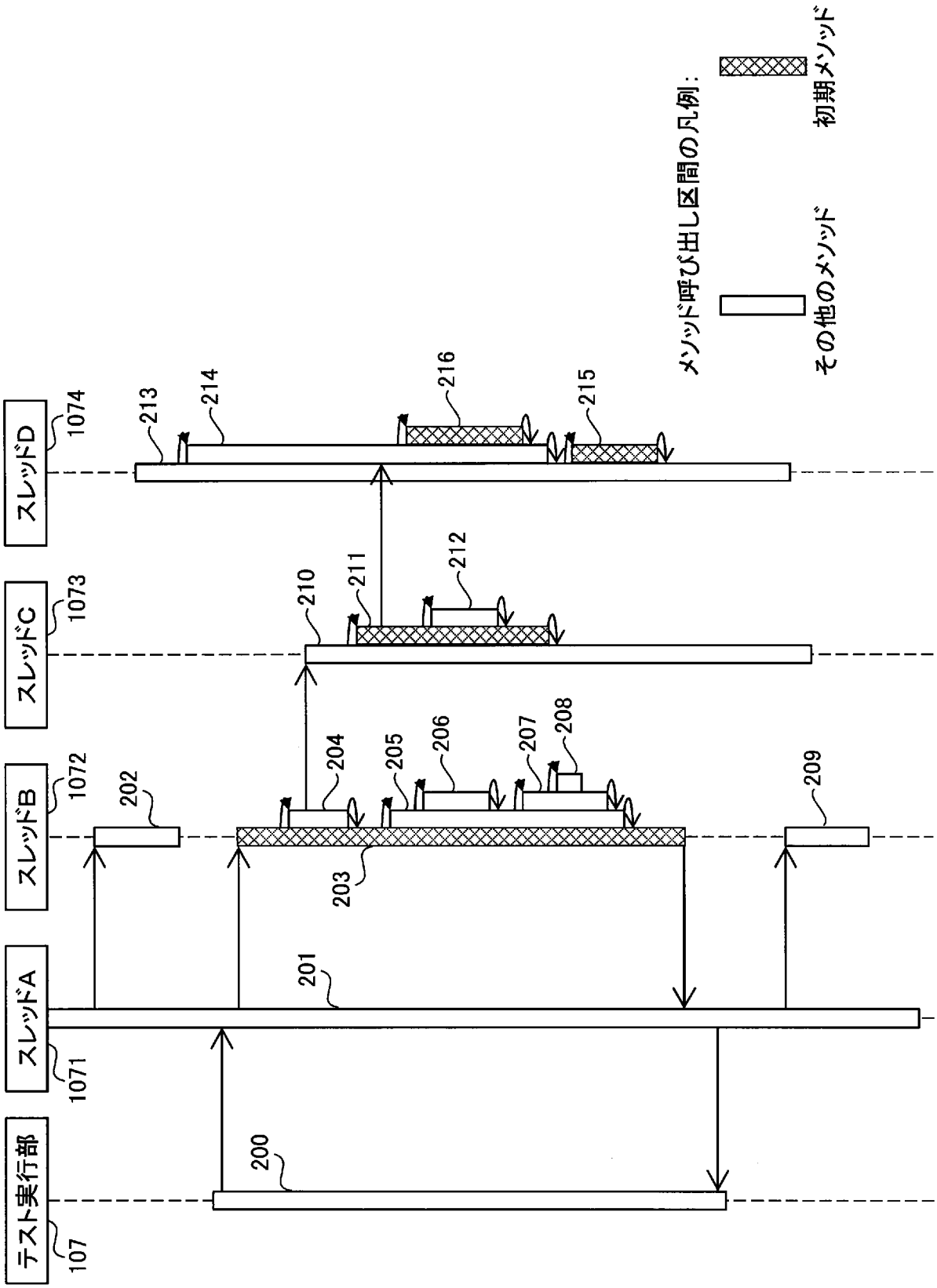


フレーム番号	メソッド情報	呼出元メソッド	第1プログラム		第2プログラム		実行時間差	実行時間比
			平均実行時間	実行回数	平均実行時間	実行回数		
:	:		:	:	:	:	:	:
15	aaa.bbb.Starter.doStart()	aaa.bbb.Starter.begin()	24850	5	28035	5	3185	0.128
16	aaa.bbb.MyClass.start(int)	aaa.bbb.Starter.doStart()	11480	25	12070	25	2950	0.051
16	aaa.bbb.OtherClass.get():int	aaa.bbb.Starter.doStart()	10661	25	10607	25	-270	-0.005
:	:		:	:	:	:	:	:
17	java.util.Map.get(Object):Object	aaa.bbb.MyClass.start(int)	50	5000	50	5250	500	0.050

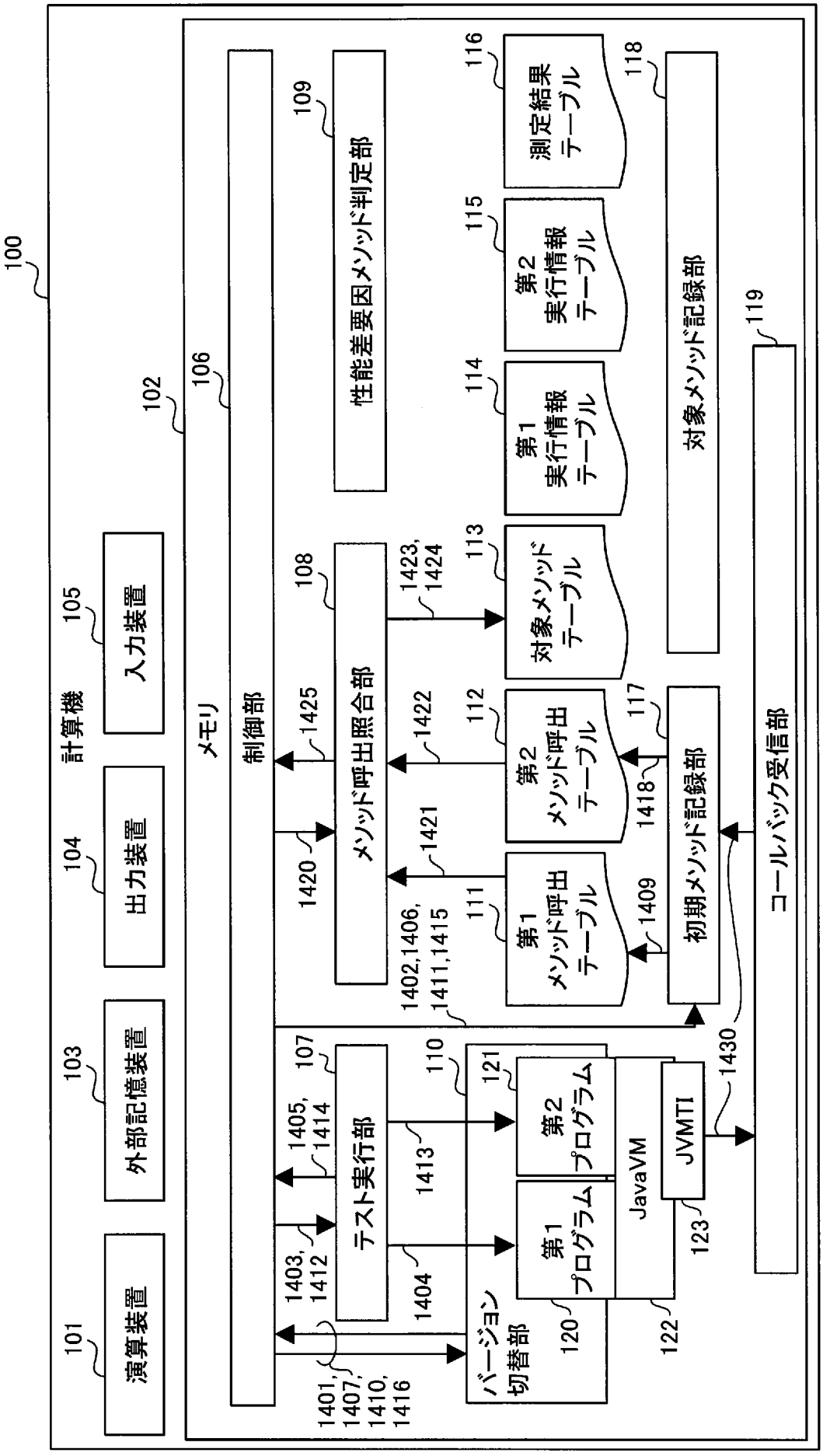
[図7]


開始
初期メソッドを抽出し測定対象メソッドに設定
140
測定対象メソッドの実行時間を測定
150
性能差要因メソッドを抽出
160
性能差要因メソッドが
あった
170
NO
YES
性能差要因メソッドが呼び出したメソッドを
測定対象メソッドに設定
180
終了

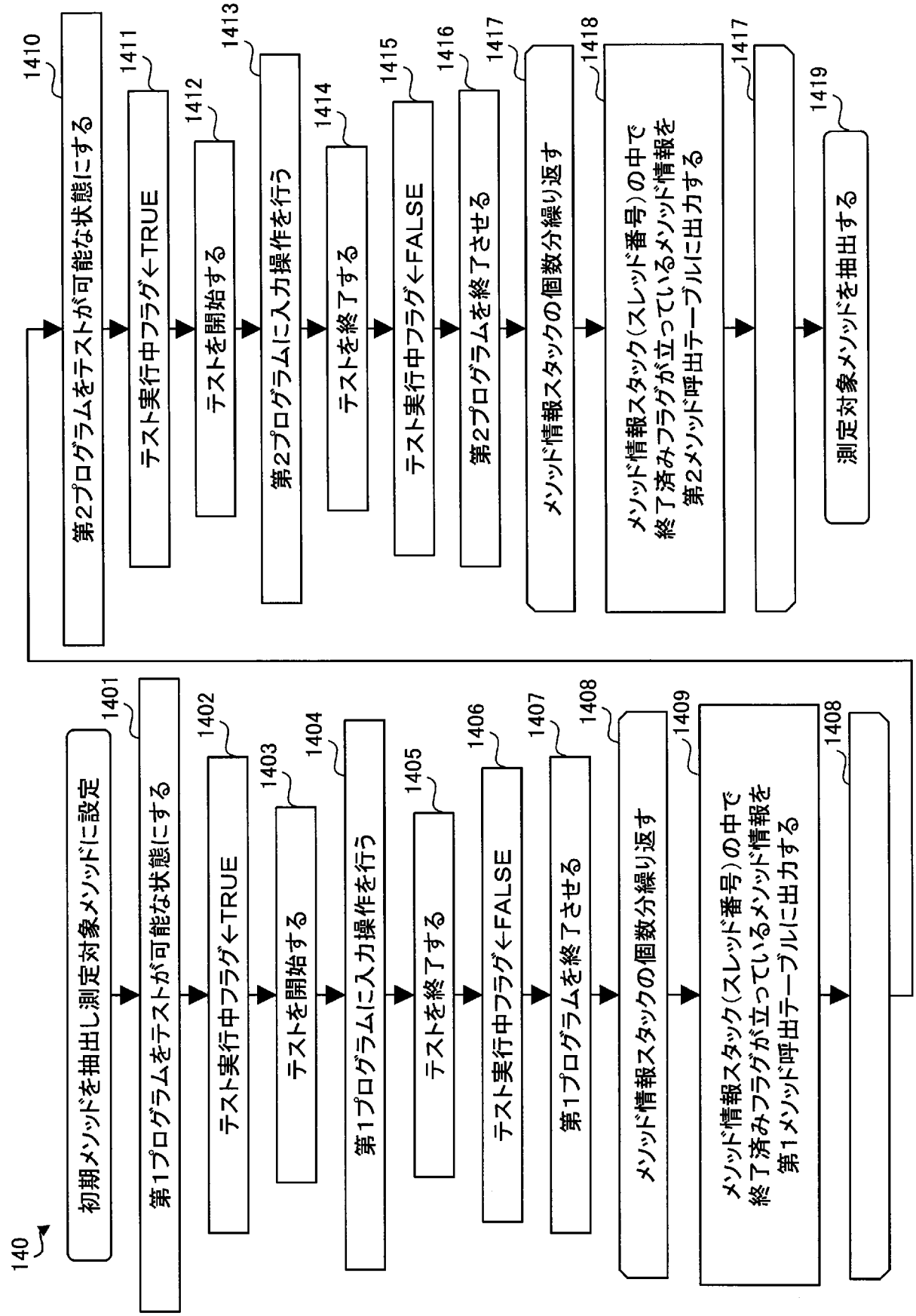
[図8]


メソッド呼び出し区間の凡例:
初期メソッド
その他のメソッド
スレッドD
1074
213
214
216
215
スレッドC
1073
210
211
212
スレッドB
1072
202
204
205
206
207
208
209
203
スレッドA
1071
201
テスト実行部
107
200

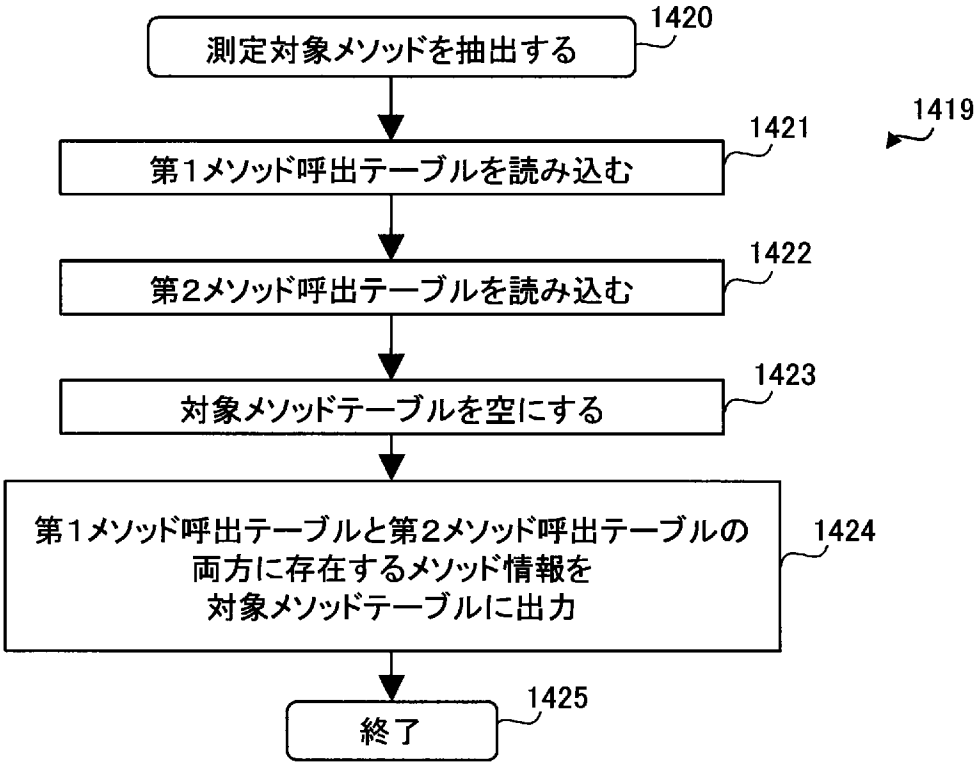
[図9]


100
計算機
101
演算装置
103
外部記憶装置
104
出力装置
105
入力装置
102
メモリ
106
制御部
109
性能差要因メソッド判定部
116
測定結果テーブル
118
対象メソッド記録部
115
第2実行情報テーブル
119
コールバック受信部
114
第1実行情報テーブル
108
メソッド呼出照合部
1423,1424
113
対象メソッドテーブル
1425
1422
112
第2メソッド呼出テーブル
117
1418
初期メソッド記録部
1420
1421
111
第1メソッド呼出テーブル
1409
1402,1406,1411,1415
1403,1412
1405,1414
107
テスト実行部
1413
110
1404
121
第2プログラム
第1プログラム
120
JavaVM
JVMTI
123
122
1430
バージョン切替部
1401,1407,1410,1416

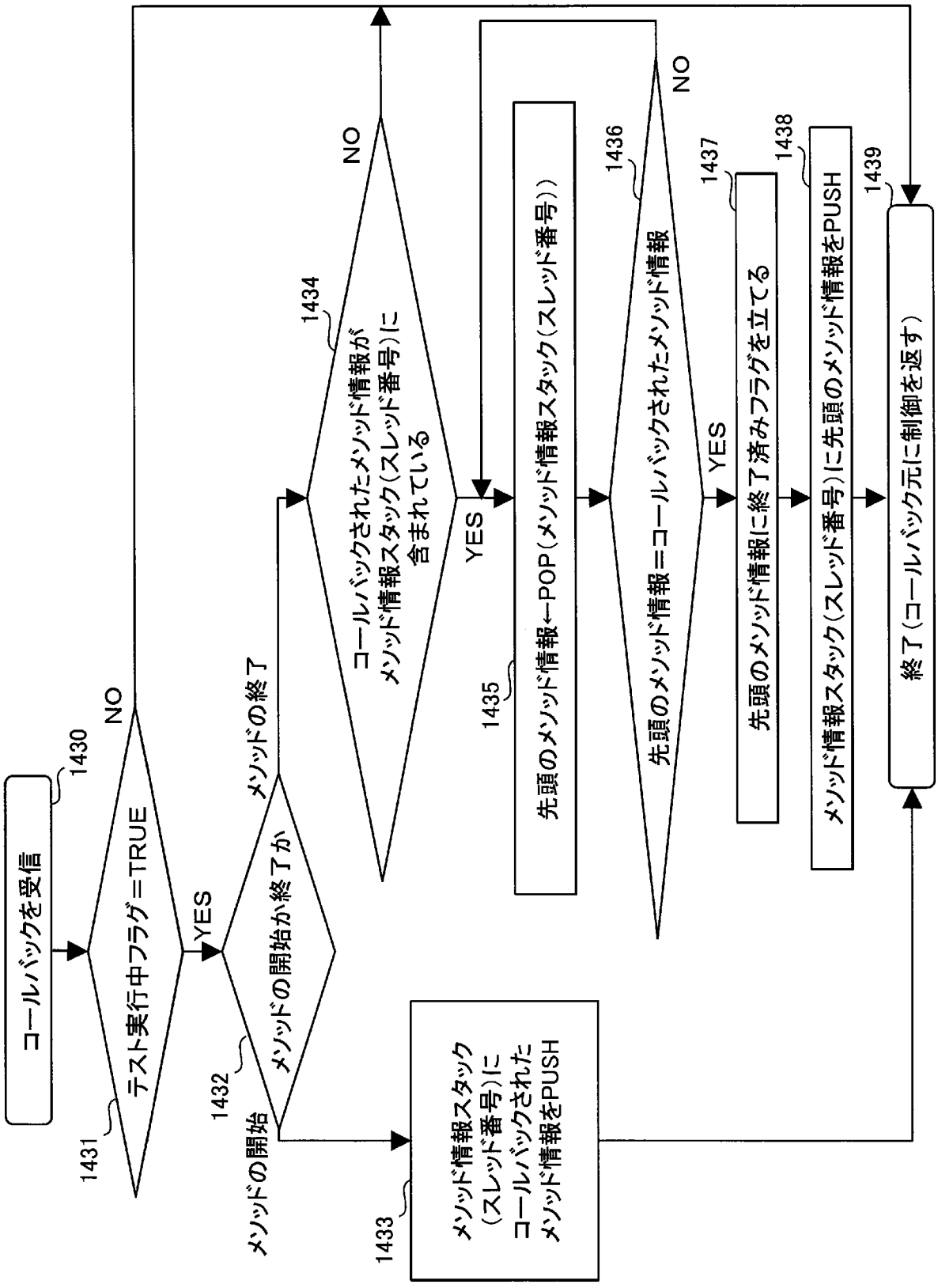
[図10A]


140
初期メソッドを抽出し測定対象メソッドに設定
1401
第1プログラムをテストが可能な状態にする
1402
テスト実行中フラグ←TRUE
1403
テストを開始する
1404
第1プログラムに入力操作を行う
1405
テストを終了する
1406
テスト実行中フラグ←FALSE
1407
第1プログラムを終了させる
1408
メソッド情報スタックの個数分繰り返す
1409
メソッド情報スタック(スレッド番号)の中で
終了済みフラグが立っているメソッド情報を
第1メソッド呼出テーブルに出力する
1408
第2プログラムをテストが可能な状態にする
1410
テスト実行中フラグ←TRUE
1411
テストを開始する
1412
第2プログラムに入力操作を行う
1413
テストを終了する
1414
テスト実行中フラグ←FALSE
1415
第2プログラムを終了させる
1416
メソッド情報スタックの個数分繰り返す
1417
メソッド情報スタック(スレッド番号)の中で
終了済みフラグが立っているメソッド情報を
第2メソッド呼出テーブルに出力する
1417
測定対象メソッドを抽出する
1419

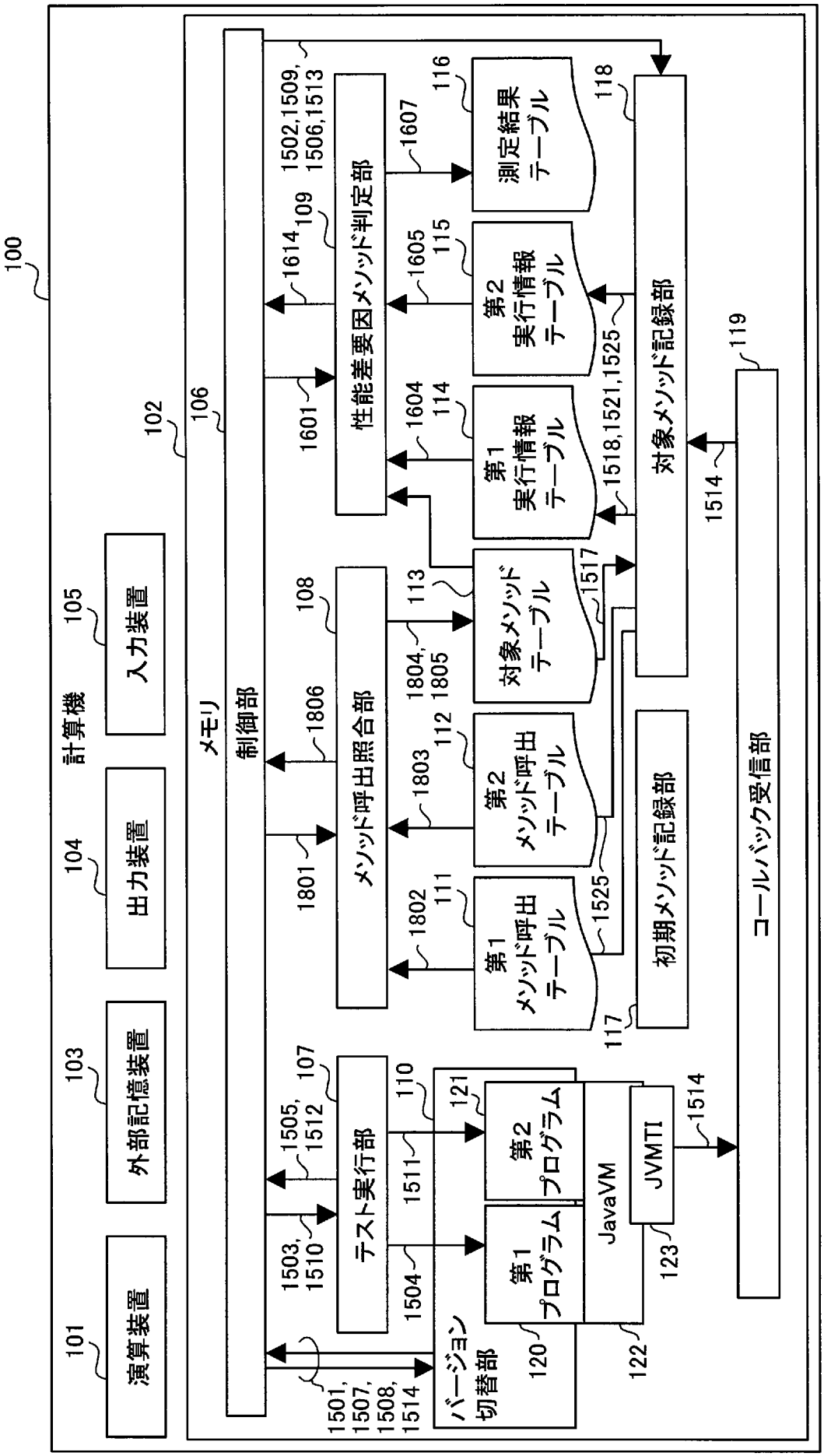
[図10B]


測定対象メソッドを抽出する
1420
1419
第1メソッド呼出テーブルを読み込む
1421
第2メソッド呼出テーブルを読み込む
1422
対象メソッドテーブルを空にする
1423
第1メソッド呼出テーブルと第2メソッド呼出テーブルの
両方に存在するメソッド情報を
対象メソッドテーブルに出力
1424
終了
1425

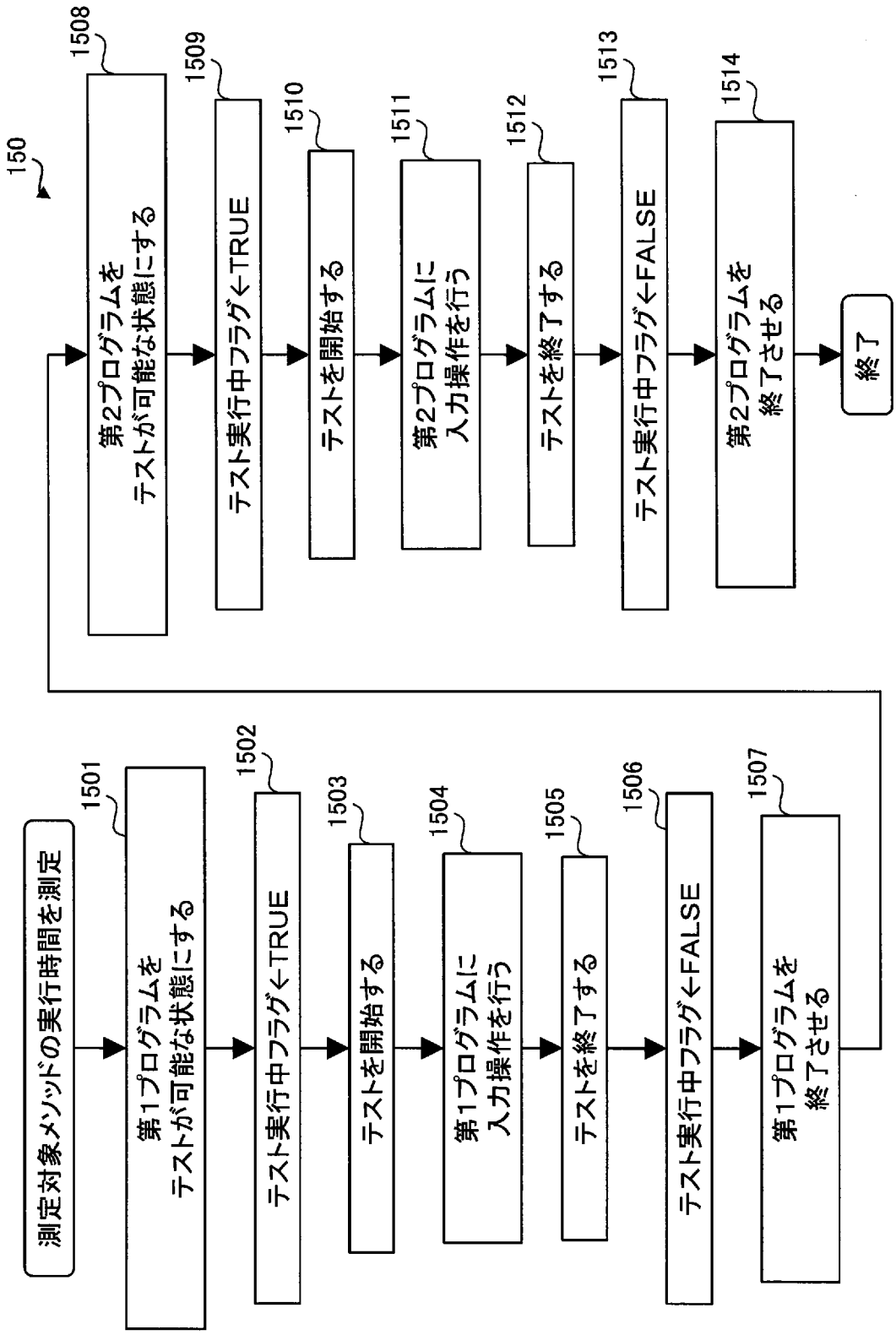
[図10C]


コールバックを受信
1430
テスト実行中フラグ=TRUE
1431
NO
YES
メソッドの開始か終了か
メソッドの開始
1432
メソッドの終了
メソッド情報スタック(スレッド番号)にコールバックされたメソッド情報をPUSH
1433
コールバックされたメソッド情報がメソッド情報スタック(スレッド番号)に含まれている
1434
NO
YES
先頭のメソッド情報←POP(メソッド情報スタック(スレッド番号))
1435
先頭のメソッド情報=コールバックされたメソッド情報
1436
NO
YES
先頭のメソッド情報に終了済みフラグを立てる
1437
メソッド情報スタック(スレッド番号)に先頭のメソッド情報をPUSH
1438
終了(コールバック元に制御を返す)
1439

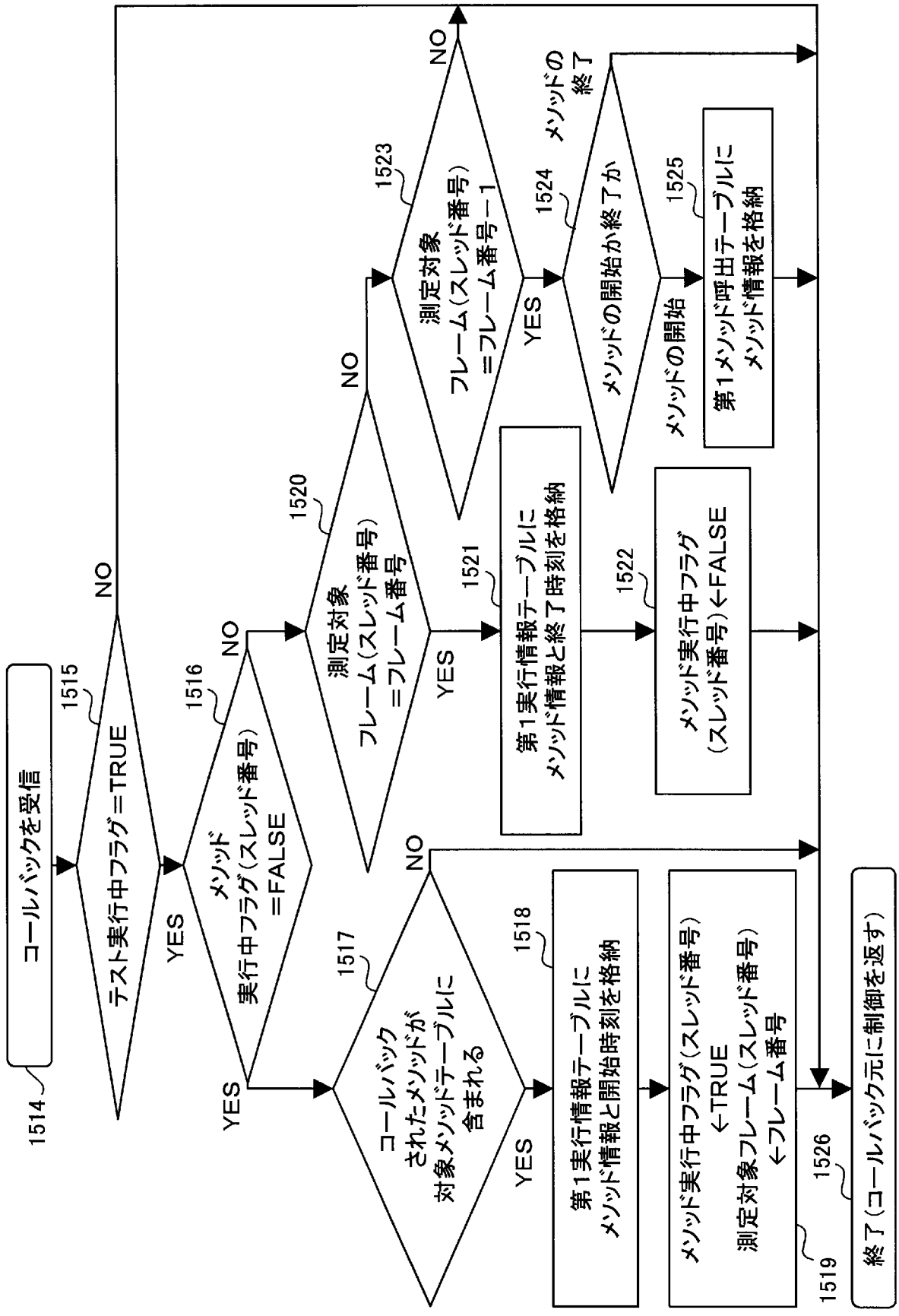
[図11]


100
101
演算装置
103
外部記憶装置
104
出力装置
105
入力装置
計算機
102
メモリ
106
制御部
1501,
1507,
1508,
1514
1503,
1510
1505,
1512
107
テスト実行部
1504
1511
110
バージョン
切替部
120
第1
プログラム
121
第2
プログラム
122
JavaVM
123
JVMTI
1514
1801
1802
111
第1
メソッド呼出
テーブル
1525
1803
112
第2
メソッド呼出
テーブル
1806
108
メソッド呼出照合部
1804,
1805
113
対象メソッド
テーブル
1517
117
初期メソッド記録部
1601
1614
109
性能差要因メソッド判定部
1502,1509,
1506,1513
1604
114
第1
実行情報
テーブル
1518,1521,1525
1605
115
第2
実行情報
テーブル
1607
116
測定結果
テーブル
118
対象メソッド記録部
119
コールバック受信部

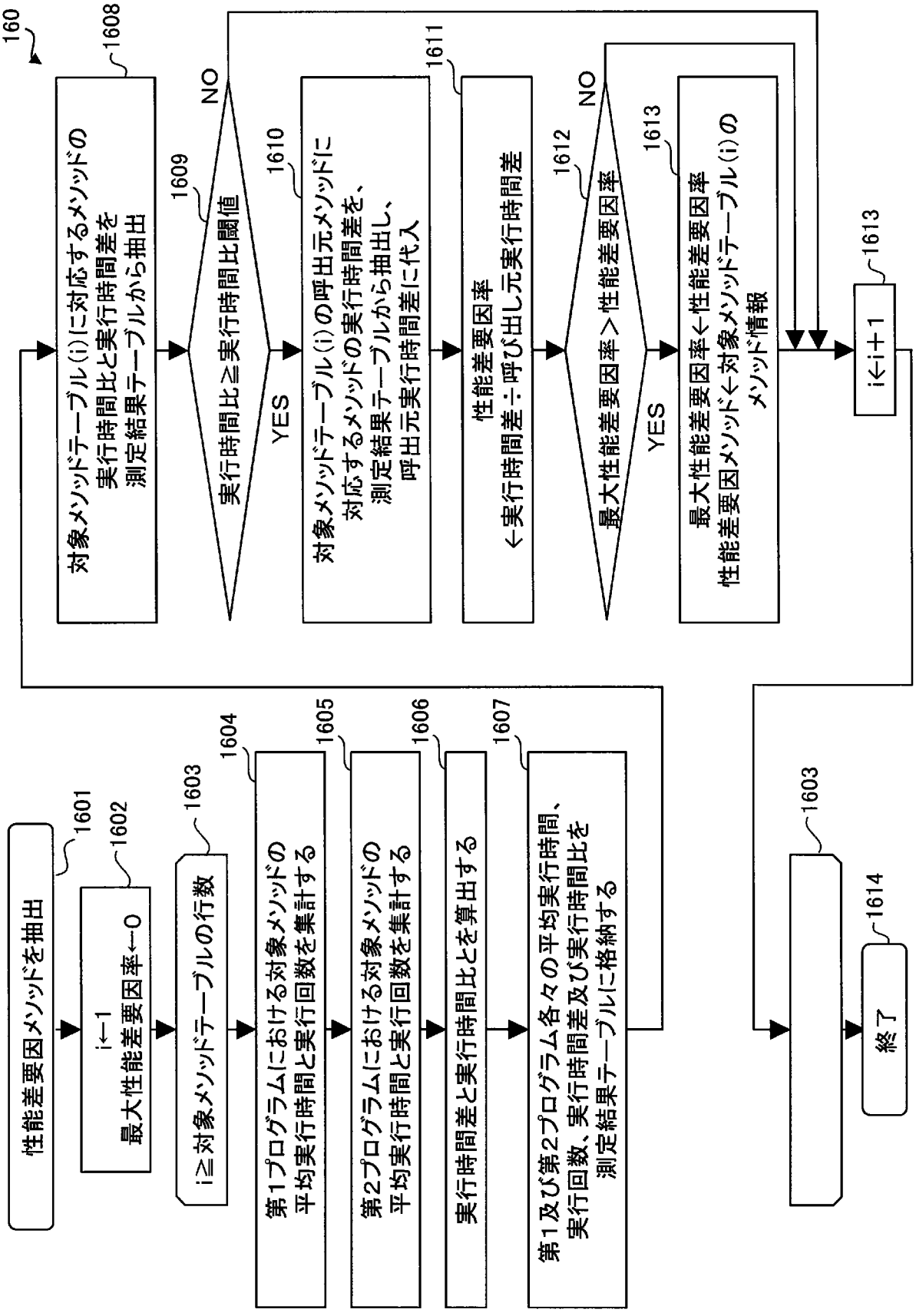
[図12A]


150
測定対象メソッドの実行時間を測定
1501
第1プログラムを
テストが可能な状態にする
1502
テスト実行中フラグ←TRUE
1503
テストを開始する
1504
第1プログラムに
入力操作を行う
1505
テストを終了する
1506
テスト実行中フラグ←FALSE
1507
第1プログラムを
終了させる
1508
第2プログラムを
テストが可能な状態にする
1509
テスト実行中フラグ←TRUE
1510
テストを開始する
1511
第2プログラムに
入力操作を行う
1512
テストを終了する
1513
テスト実行中フラグ←FALSE
1514
第2プログラムを
終了させる
終了

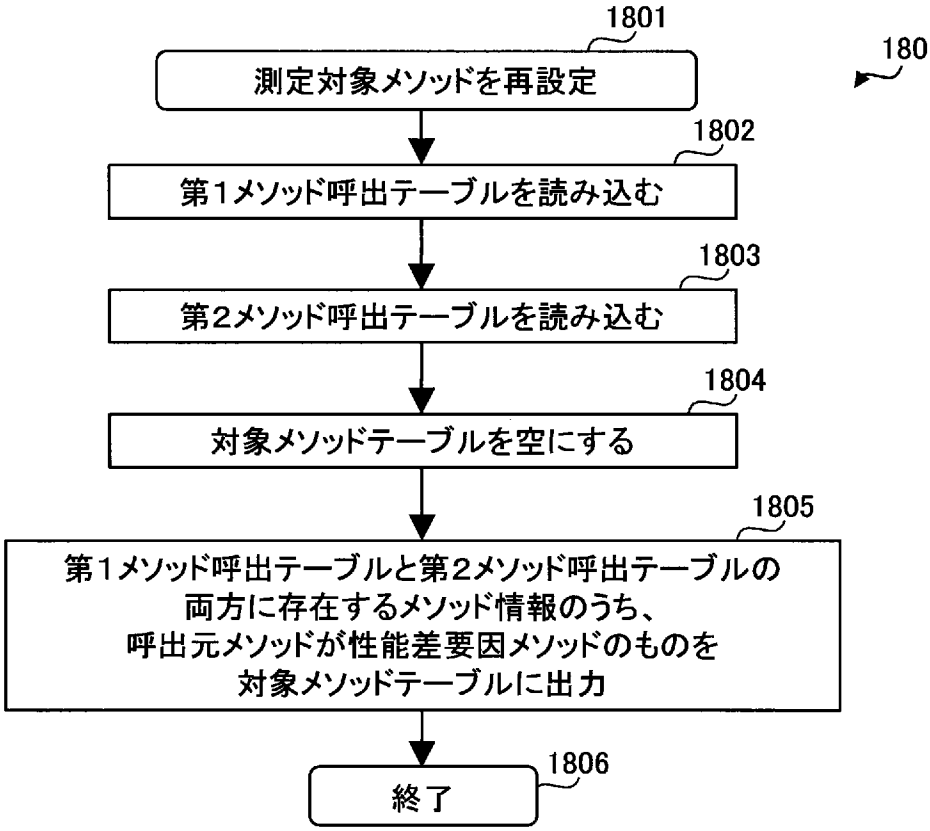
[図12B]


コールバックを受信
1514
テスト実行中フラグ＝TRUE
1515
NO
YES
メソッド
実行中フラグ(スレッド番号)
＝FALSE
1516
NO
YES
測定対象
フレーム(スレッド番号)
＝フレーム番号
1520
NO
YES
第1実行情報テーブルに
メソッド情報と終了時刻を格納
1521
メソッド実行中フラグ
(スレッド番号)←FALSE
1522
測定対象
フレーム(スレッド番号)
＝フレーム番号－1
1523
NO
YES
メソッドの開始か終了か
1524
メソッドの開始
メソッドの終了
第1メソッド呼出テーブルに
メソッド情報を格納
1525
コールバック
されたメソッドが
対象メソッドテーブルに
含まれる
1517
NO
YES
第1実行情報テーブルに
メソッド情報と開始時刻を格納
1518
メソッド実行中フラグ(スレッド番号)
←TRUE
測定対象フレーム(スレッド番号)
←フレーム番号
1519
終了(コールバック元に制御を返す)
1526

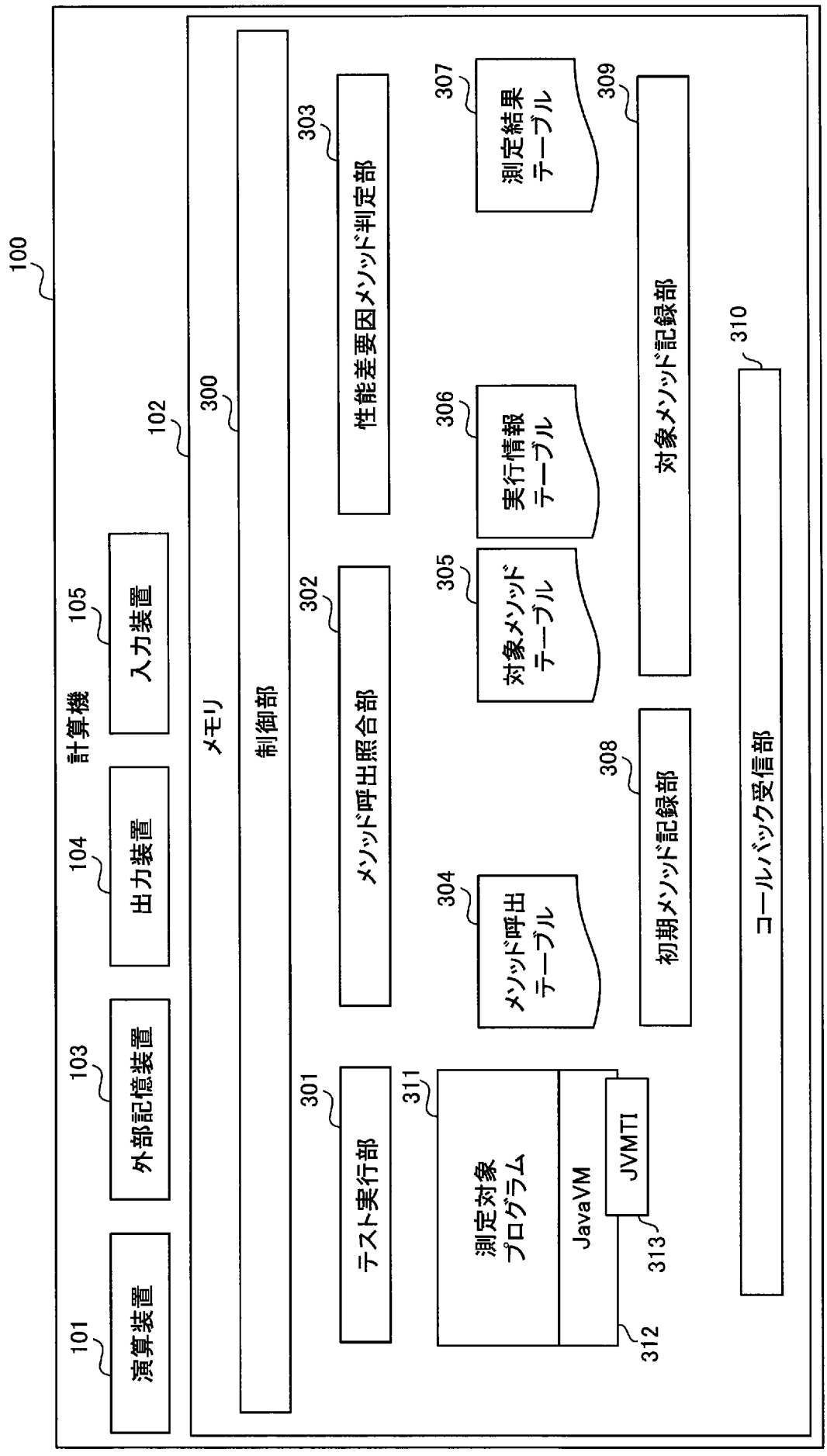
[図13]


160
性能差要因メソッドを抽出
1601
i←1
最大性能差要因率←0
1602
i≧対象メソッドテーブルの行数
1603
第1プログラムにおける対象メソッドの
平均実行時間と実行回数を集計する
1604
第2プログラムにおける対象メソッドの
平均実行時間と実行回数を集計する
1605
実行時間差と実行時間比とを算出する
1606
第1及び第2プログラム各々の平均実行時間、
実行回数、実行時間差及び実行時間比を
測定結果テーブルに格納する
1607
対象メソッドテーブル(i)に対応するメソッドの
実行時間比と実行時間差を
測定結果テーブルから抽出
1608
実行時間比≧実行時間比閾値
1609
NO
YES
対象メソッドテーブル(i)の呼出元メソッドに
対応するメソッドの実行時間差を、
測定結果テーブルから抽出し、
呼出元実行時間差に代入
1610
性能差要因率
←実行時間差÷呼び出し元実行時間差
1611
最大性能差要因率＞性能差要因率
1612
NO
YES
最大性能差要因率←性能差要因率
性能差要因メソッド←対象メソッドテーブル(i)の
メソッド情報
1613
i←i+1
1613
1603
終了
1614

[図14]


1801
測定対象メソッドを再設定
180
1802
第1メソッド呼出テーブルを読み込む
1803
第2メソッド呼出テーブルを読み込む
1804
対象メソッドテーブルを空にする
1805
第1メソッド呼出テーブルと第2メソッド呼出テーブルの
両方に存在するメソッド情報のうち、
呼出元メソッドが性能差要因メソッドのものを
対象メソッドテーブルに出力
1806
終了

[図15]


100
計算機
101
演算装置
103
外部記憶装置
104
出力装置
105
入力装置
102
メモリ
300
制御部
301
テスト実行部
311
測定対象プログラム
JavaVM
JVMTI
312
313
302
メソッド呼出照合部
303
性能差要因メソッド判定部
304
メソッド呼出テーブル
305
対象メソッドテーブル
306
実行情報テーブル
307
測定結果テーブル
308
初期メソッド記録部
309
対象メソッド記録部
310
コールバック受信部

[図16]

3041 フレーム番号	3042 メソッド情報	3043 呼出元メソッド情報 (304)
16	aaa.ccc.OldClass.setLen(int)	aaa.bbb.Starter.init()
16	aaa.Util.getName(String)	aaa.bbb.Starter.beforeStart()
16	aaa.bbb.MyClass.start(int)	aaa.bbb.Starter.doStart()
16	aaa.bbb.OtherClass.get():int	aaa.bbb.Starter.doStart()
:	:	:

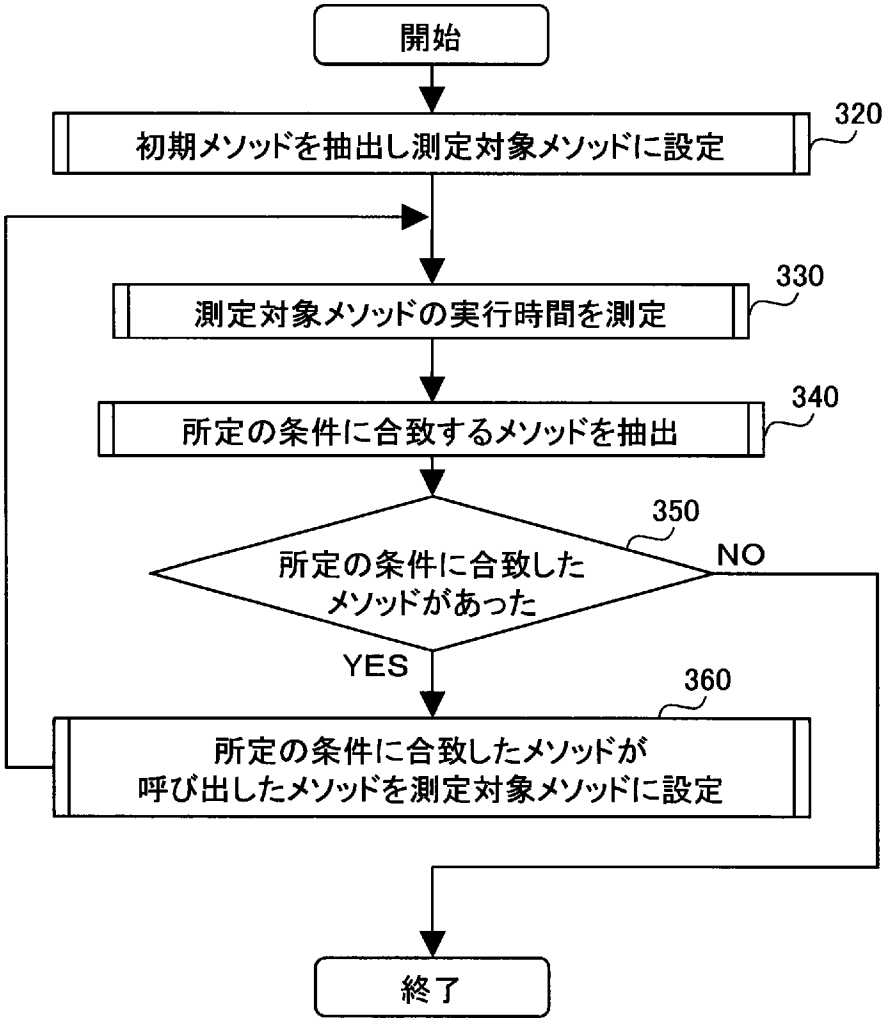
[図17]

3061 フレーム番号	3062 メソッド情報	3063 開始/終了	3064 時刻 (306)
16	aaa.bbb.MyClass.start(int)	開始	753227897
16	aaa.bbb.MyClass.start(int)	終了	753239377
16	aaa.bbb.OtherClass.get():int	開始	753239399
16	aaa.bbb.OtherClass.get():int	終了	753250060
16	aaa.bbb.MyClass.start(int)	開始	753250082
:	:	:	:

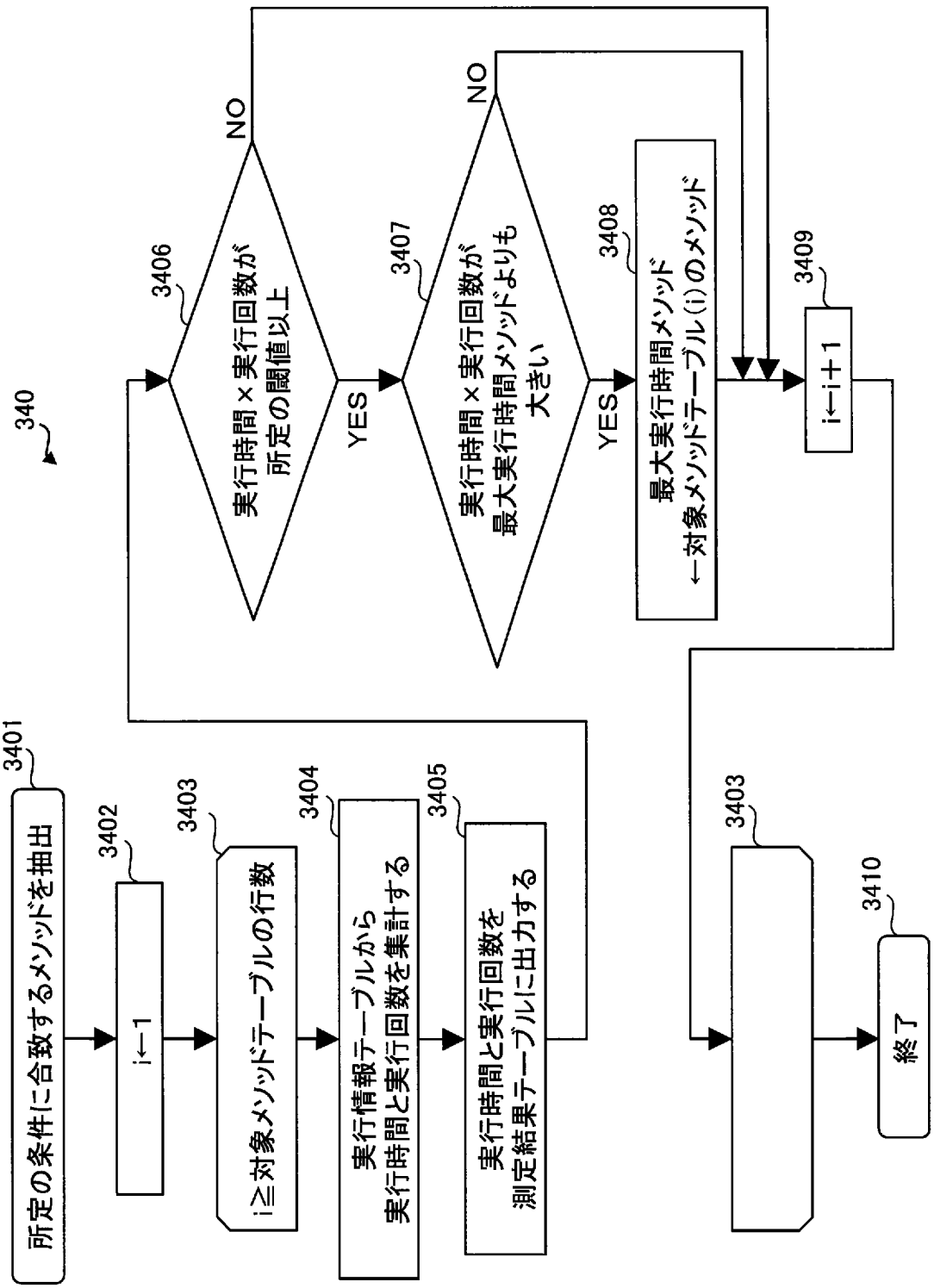
[図18]

3071 フレーム番号	3072 メソッド情報	3073 平均実行時間	3074 実行回数 (307)
:	:	:	:
15	aaa.bbb.Starter.doStart()	24850	5
16	aaa.bbb.MyClass.start(int)	11480	25
16	aaa.bbb.OtherClass.get():int	10661	25
:	:	:	:

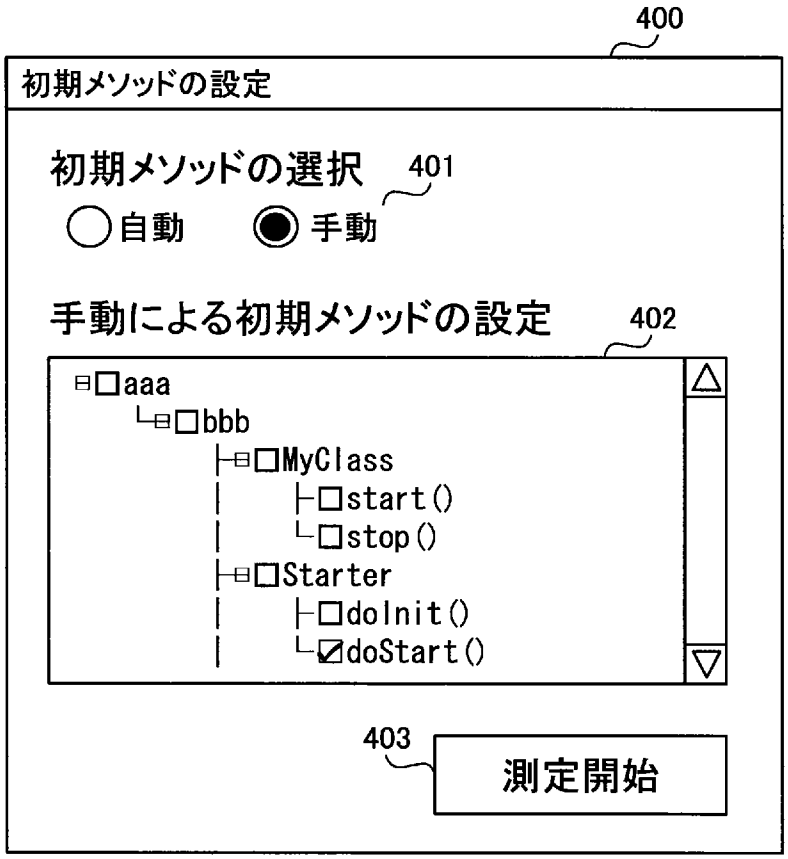
[図19]


開始
初期メソッドを抽出し測定対象メソッドに設定
320
測定対象メソッドの実行時間を測定
330
所定の条件に合致するメソッドを抽出
340
所定の条件に合致した
メソッドがあった
350
NO
YES
所定の条件に合致したメソッドが
呼び出したメソッドを測定対象メソッドに設定
360
終了

[図20]


340
所定の条件に合致するメソッドを抽出
3401
i←1
3402
i≧対象メソッドテーブルの行数
3403
実行情報テーブルから
実行時間と実行回数を集計する
3404
実行時間と実行回数を
測定結果テーブルに出力する
3405
実行時間 × 実行回数が
所定の閾値以上
3406
NO
YES
実行時間 × 実行回数が
最大実行時間メソッドよりも
大きい
3407
NO
YES
最大実行時間メソッド
←対象メソッドテーブル(i)のメソッド
3408
i←i+1
3409
3403
終了
3410

[図21]


400
初期メソッドの設定
初期メソッドの選択
401
自動
手動
手動による初期メソッドの設定
402
aaa
bbb
MyClass
start()
stop()
Starter
doInit()
doStart()
403
測定開始

INTERNATIONAL SEARCH REPORT

International application No.
PCT/JP2010/069335

A. CLASSIFICATION OF SUBJECT MATTER
G06F11/34(2006.01)i

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
G06F11/34

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Jitsuyo Shinan Koho	1922-1996	Jitsuyo Shinan Toroku Koho	1996-2011
Kokai Jitsuyo Shinan Koho	1971-2011	Toroku Jitsuyo Shinan Koho	1994-2011

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	JP 2008-217721 A (NTT Data Corp.), 18 September 2008 (18.09.2008), paragraphs [0014] to [0080] (Family: none)	1,6-8,13-15, 18
Y	JP 2010-198133 A (International Business Machines Corp.), 09 September 2010 (09.09.2010), paragraphs [0040] to [0050] (Family: none)	1,6-8,13-15, 18
Y	JP 2007-249495 A (NEC Corp.), 27 September 2007 (27.09.2007), paragraphs [0007] to [0029] (Family: none)	1,6-8,13-15, 18

☐ Further documents are listed in the continuation of Box C.

☐ See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
17 January, 2011 (17.01.11)	25 January, 2011 (25.01.11)
Name and mailing address of the ISA/ Japanese Patent Office Facsimile No.	Authorized officer Telephone No.

国際調査報告

国際出願番号　ＰＣＴ／ＪＰ２０１０／０６９３３５

Ａ．　発明の属する分野の分類（国際特許分類（ＩＰＣ））

Int.Cl.　G06F11/34(2006.01)i

Ｂ．　調査を行った分野

調査を行った最小限資料（国際特許分類（ＩＰＣ））

Int.Cl.　G06F11/34

最小限資料以外の資料で調査を行った分野に含まれるもの

日本国実用新案公報	１９２２－１９９６年
日本国公開実用新案公報	１９７１－２０１１年
日本国実用新案登録公報	１９９６－２０１１年
日本国登録実用新案公報	１９９４－２０１１年

国際調査で使用した電子データベース（データベースの名称、調査に使用した用語）

Ｃ．　関連すると認められる文献

引用文献のカテゴリー＊	引用文献名　及び一部の箇所が関連するときは、その関連する箇所の表示	関連する請求項の番号
Y	JP 2008-217721 A（株式会社エヌ・ティ・ティ・データ）2008.09.18, 段落[0014]-[0080]（ファミリーなし）	1, 6-8, 13-15, 18
Y	JP 2010-198133 A（インターナショナル・ビジネス・マシーンズ・コーポレーション）2010.09.09, 段落[0040]-[0050]（ファミリーなし）	1, 6-8, 13-15, 18
Y	JP 2007-249495 A（日本電気株式会社）2007.09.27, 段落[0007]-[0029]（ファミリーなし）	1, 6-8, 13-15, 18

☐　Ｃ欄の続きにも文献が列挙されている。

☐　パテントファミリーに関する別紙を参照。

＊　引用文献のカテゴリー

「Ａ」特に関連のある文献ではなく、一般的技術水準を示すもの

「Ｅ」国際出願日前の出願または特許であるが、国際出願日以後に公表されたもの

「Ｌ」優先権主張に疑義を提起する文献又は他の文献の発行日若しくは他の特別な理由を確立するために引用する文献（理由を付す）

「Ｏ」口頭による開示、使用、展示等に言及する文献

「Ｐ」国際出願日前で、かつ優先権の主張の基礎となる出願の日の後に公表された文献

「Ｔ」国際出願日又は優先日後に公表された文献であって出願と矛盾するものではなく、発明の原理又は理論の理解のために引用するもの

「Ｘ」特に関連のある文献であって、当該文献のみで発明の新規性又は進歩性がないと考えられるもの

「Ｙ」特に関連のある文献であって、当該文献と他の１以上の文献との、当業者にとって自明である組合せによって進歩性がないと考えられるもの

「＆」同一パテントファミリー文献

国際調査を完了した日 １７．０１．２０１１	国際調査報告の発送日 ２５．０１．２０１１
国際調査機関の名称及びあて先 日本国特許庁（ＩＳＡ／ＪＰ） 郵便番号１００－８９１５ 東京都千代田区霞が関三丁目４番３号	特許庁審査官（権限のある職員）　５Ｂ　３５６２ 多胡　滋 電話番号　０３－３５８１－１１０１　内線　３５４５