[19] 中华人民共和国国家知识产权局

[51] Int. Cl.
G06F 9/44 (2006.01)

[12] 发明专利说明书

专利号 ZL 200610057382.7

[45] 授权公告日 2008 年 7 月 23 日

[11] 授权公告号 CN 100405294C

[22] 申请日 2006.3.14
[21] 申请号 200610057382.7
[30] 优先权
[32] 2005.3.15 [33] US [31] 11/081,054
[73] 专利权人 国际商业机器公司
地址 美国纽约
[72] 发明人 B·M·奥康奈尔
R·E·内斯比特 K·E·沃甘
[56] 参考文献
JP2003-280922A 2003.10.3
CN1374585A 2002.10.16
US6412108B1 2002.6.25
CN1461991A 2003.12.17
EP0778521A2 1997.6.11
CN1512334A 2004.7.14
US2002/0174418A1 2002.11.21
审查员 吉张媛
[74] 专利代理机构 北京市中咨律师事务所
代理人 于 静 李 峥

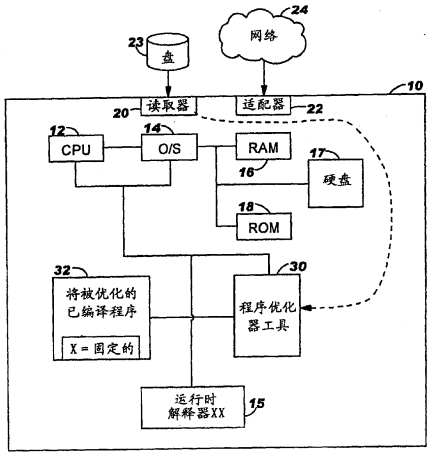
权利要求书 3 页 说明书 9 页 附图 2 页

[54] 发明名称

在运行时期间优化计算机程序的系统与方法

[57] 摘要

用于在运行时期间优化计算机程序的系统、方法与程序产品。 在运行时期间，确定计算机程序是否调用包括对固定变量的条件求值的方法，并且对固定变量的求值没有产生除返回执行计算机程序之外的任何成果。 如果是，则为计算机程序的后续迭代而从计算机程序中删除对包括所述条件求值的方法的调用。 从而，计算机程序的后续执行将产生如同所述条件求值已被执行的相同结果。 如果对固定变量的求值产生了某些成果，则不从计算机程序的后续执行中删除对包括所述条件求值的方法的调用。



1. 一种用于在运行期间优化计算机程序的方法，所述方法包括下述步骤：

在运行期间，确定所述计算机程序是否调用包括对固定变量的条件求值的第一方法，并且对所述固定变量的条件求值没有产生除返回执行所述计算机程序之外的任何成果，以及

如果是，则从所述计算机程序中删除对包括所述条件求值的所述第一方法的调用，从而所述计算机程序的后续执行不再调用所述第一方法，其中所述计算机程序的所述后续执行将产生如同所述条件求值已被执行的相同结果，以及

如果不是，则不从所述计算机程序的所述后续执行中删除对包括所述条件求值的所述第一方法的所述调用。

2. 如权利要求 1 所述的方法，其中对所述第一方法的调用被包括在所述计算机程序内的第二方法中，并且对所述固定变量的条件求值导致直接返回到所述第二方法。

3. 如权利要求 1 所述的方法，其中对所述固定变量的所述条件求值没有导致对所述计算机程序之外的另一方法的调用并且没有导致所述计算机程序所需的任何计算或其它求值。

4. 如权利要求 1 所述的方法，其中在所述计算机程序的第一次执行期间，所述变量的值被设置为固定值，并且与所述变量相关联的参量被计算。

5. 如权利要求 1 所述的方法，其中所述计算机程序包括对固定变量的条件求值，并且对所述固定变量的条件求值没有产生除返回执行所述计算机程序之外的任何成果，该方法还包括下述步骤：从所述计算机程序中删除用于计算调用所述第一方法所需的参量的指令。

6. 一种用于在运行期间优化计算机程序的系统，所述系统包括：

用于在运行期间确定所述计算机程序是否调用包括对固定变量的条件求值的第一方法的装置，并且对所述固定变量的条件求值没有产生除返回

执行所述计算机程序之外的任何成果，以及

用于如下操作的装置：如果所述确定的结果为是则从所述计算机程序中删除对包括所述条件求值的所述第一方法的调用、从而所述计算机程序的后续执行不再调用所述第一方法，其中所述计算机程序的所述后续执行将产生如同所述条件求值已被执行的相同结果，并且如果所述确定的结果不是，则在所述计算机程序的所述后续执行中保留对包括所述条件求值的所述第一方法的所述调用。

7. 如权利要求 6 所述的系统，其中对所述第一方法的调用被包括在所述计算机程序内的第二方法中，并且对所述固定变量的条件求值导致直接返回到所述第二方法。

8. 如权利要求 6 所述的系统，其中对所述固定变量的所述条件求值没有导致对所述计算机程序之外的另一方法的调用并且没有导致所述计算机程序所需的任何计算或其它求值。

9. 如权利要求 6 所述的系统，其中在所述计算机程序的第一次执行期间，所述变量的值被设置为固定值，并且与所述变量相关联的参量被计算。

10. 如权利要求 6 所述的系统，其中所述计算机程序包括对固定变量的条件求值，并且对所述固定变量的条件求值没有产生除返回执行所述计算机程序之外的任何成果，该系统还包括用于从所述计算机程序中删除用于计算调用所述第一方法所需的参量的指令的装置。

11. 一种用于在运行期间优化计算机程序的方法，所述计算机程序包括对第一方法的调用，所述第一方法包括对第二方法的调用，所述方法包括下述步骤：

在运行期间，确定所述第二方法是否包括对固定变量的条件求值，使得所述第二方法的执行没有产生除返回到所述第一方法之外的任何成果，以及

如果是，则在所述第一方法中删除对所述第二方法的所述调用，以及

如果不是，则不在所述第一方法中删除对所述第二方法的所述调用。

12. 如权利要求 11 所述的方法，其中所述第一方法还包括用于计算所述第二方法的所述调用所需的参量的指令，并且所述第二方法包括对固定变量的条件求值，使得所述第二方法的执行不产生除返回到所述第一方法之外的任何成果，所述方法还包括下述步骤：从所述第一方法中删除用于计算所述第二方法的所述调用所需的所述参量的所述指令。

在运行时期间优化计算机程序的系统与方法

技术领域

本发明一般地涉及计算机系统，更具体地说，涉及优化用于执行的程序代码的程序工具。

背景技术

计算机程序常常包括基于变量值的条件求值，例如“如果 X = A，则进入步骤 10000；否则进入步骤 10010”。取决于变量“X”的当前值，条件可以是真或假。例如，当程序处于调试阶段时，变量“调试”被设置为“真”。存在这样的计算机程序，其对变量“调试”进行条件求值以确定是否应当记录对于调试目的有用的某些数据。换句话说，调试变量的状态被用在判定步骤中以确定是否记录数据。判定步骤可以是：“如果调试 = 真，则将 XYZ 数据写入日志；否则跳到下一步骤”。因此，如果调试变量被设置为“真”，则指定数据被写入日志。然而，如果程序没有处于调试阶段，例如程序已经完成了调试，则不需要将指定数据写入日志。因此，在调试阶段之后，程序开发者使用公知的程序工具将变量“调试”设置为“假”，从而 XYZ 数据将不被写入日志。并且，程序开发者将使用该工具来声明“调试”变量现在被固定（或不可变）为“假”。在 Sun Microsystems Java（tm）编程语言和其它运行时语言中，该声明将由程序员输入作为被写入源代码中的命令。该命令意味着，一旦在执行期间变量第一次被设置为预期的固定状态，它将永不改变。在编译期间，编译器记录该声明，监视该变量，并且如果该变量的值被试图从其初始值改变，就返回错误代码。

美国专利 6,728,952 公开了一种用于标识总是为真或总是为假（即，“空”）的断言的计算机系统，并且指明如果 IF 语句所依赖的表达式被标识为空断言，则用于该 IF 语句的代码可以被优化。这是因为 IF 语句所依

赖的表达式将总是为真或者将总是为假。该专利还指明已经存在专用计算机系统，其能够确定断言是否为空。

Auslander 等人的名为“Fast, Effective Dynamic Compilation”的出版物，SIGPLAN Notices，vol.31，no.5，第 149－159 页，1996 年 5 月，公开了多种运行时优化技术。例如，运行时常量可以成为指令立即数（instruction immediate）而非存储器负载，可以对它们应用常量传导与合并，基于它们的条件分支可以被消除，并且它们控制的循环可以被完全打开。

Calder 等人的名为“Value Profiling”的出版物，Proceedings of 13th Annual IEEE/ACM International Symposium on Micro architecture，第 259－269 页，1997 年，公开了在编译时将变量标识为不变量或常量，这允许编译器执行包括常量合并、代码专用和部分求值的优化。

本发明的目的是进一步优化对程序的执行。

发明内容

本发明蕴含于用于在运行时期间优化计算机程序的系统、方法和程序产品中。在运行时期间，确定计算机程序是否调用包括对固定变量的条件求值的方法，并且除了返回执行计算机程序外对固定变量的求值不产生任何成果。如果是，则为计算机程序的后续迭代从计算机程序删除对包括条件求值的方法的调用。从而，计算机程序的后续执行将产生如同条件求值已被执行的相同结果。如果对固定变量的求值产生某些成果，则不从计算机程序的后续执行中删除对包括条件求值的方法的调用。

根据本发明的特征，对第一所述方法的调用被包含在计算机程序内的第二方法中，并且对固定变量的求值导致直接返回到第二方法。对固定变量的条件求值不产生对计算机程序之外的另一方法的调用或者计算机程序所需的任何计算或其它求值。在计算机程序的第一次迭代期间，变量值被设置为固定值，并且与该变量相关联的参量被计算。

附图说明

图 1 是结合了本发明的计算机的框图.

图 2 是根据本发明的计算机程序工具的流程图.

具体实施方式

现在将参考附图详细说明本发明. 图 1 图示出结合并执行了本发明的计算机 10. 计算机 10 包括公知的 CPU 12、操作系统 14、运行时解释器 XX 15、RAM 16、硬盘 17、ROM 18、存储设备读取器 20 和网络适配器卡 22。读取器 20 可以是存储媒体 23 的 CD ROM 读取器、DVD 读取器、软盘读取器或其它读取器，在存储媒体 23 上，包含本发明的计算机程序产品可以被输入到计算机 10. 替代性地，网络适配器卡 22 可以从计算机可读网络 24（例如因特网或 LAN）读取包含本发明的计算机程序. 网络 24 还包括传播媒体，用以将计算机程序输送到计算机 10 以被读取到计算机 10 中. 一旦被输入到计算机 10，则包含本发明的计算机程序工具 30 被存储在 RAM 16 或硬盘 17 中，用于在 CPU 12 上执行. 图 1 还图示出根据本发明由 CPU 12 执行并由计算机程序工具 30 优化的另一计算机程序 32. 计算机程序 32 的功能或性质对本发明来说不重要，编写计算机程序 32 的计算机语言也不重要. 计算机程序 32 已被编译器（未示出）编译为运行时/目标代码格式，以在计算机 10 中执行. 作为示例，计算机程序 32 可以用 Sun Microsystems JAVA（tm）编程语言编写.

图 2 图示出根据本发明由程序工具 30 执行的处理. 在步骤 100 中，程序工具 30 获取要被优化并执行的程序 32 的“下一个”指令. 在步骤 100 的第一次迭代期间，这应当是程序 32 的第一个指令，并且判定 102 通向判定 104.（在程序 32 最后的指令被执行之后，则判定 102 将通向步骤 106，步骤 106 是程序 32 和程序工具 30 的执行的结束.）在判定 102 之后，在肯定分支处，程序工具 30 确定程序 32 的指令是否是调用/调入（call/invoke）诸如“方法”的函数的命令（判定 104）.“方法”是面向对象形式的函数，所述函数例如是记录数据、计算值、写入数据库、通过网络通信或链接数

据串。如果指令不是调入函数的命令（判定 104，否定分支），则程序工具 30 将程序 32 的指令传送至运行时解释器 XX 15 用以执行（步骤 110）。然而，如果指令是调入方法的命令（判定 104，肯定分支），则程序工具 30 调入该方法，即调用该方法开始执行（步骤 112）。然后，被调用的方法开始执行。这可以包括被调用的方法准备随后将用于其自身执行或调用另一方法而需要的任何参量（步骤 116）。作为示例，参量可以包括程序 32 的使用者的身份、计算系数、用于数据库的数据、网络端点地址或数据串。当存在这样的参量时，参量的准备可以包括计算、字符串的连接、数据的获取等。在准备参量之后，所调用的函数执行其操作指令或语句，例如对变量进行条件求值、计算结果、写入数据库、连接到端点或链接数据串（步骤 120）。在被调用方法的每个指令或语句被传送到运行时解释器 XX 15 用于执行时，程序工具 30 监视程序指令或语句的性质（步骤 122）。在该监视期间，程序工具 30 确定被调用方法的当前程序指令或语句是否将对变量进行条件求值并且按照变量值行动，所述变量被预先声明为固定的（判定 130）。在该程序步骤中进行条件求值的所有值都是固定的。当程序 32 是以源代码编写的时候，这些声明将通过使用另一程序工具而出现，并且这些变量为固定的指示将已被编译到程序 32 的目标代码中。这在图 1 中由程序 32 中的参考数据象征性示出，即变量“X 是固定的并且 A 是固定的”。下面是这种条件求值的示例：“如果 X = A，则跳到步骤 10000；否则跳到步骤 10010C”、“如果 X = 真且 A = 假，则跳到步骤 10000，否则跳到步骤 10010C”、“当 X > A 时，则跳到步骤 10000；否则跳到步骤 10010C”。因此，通过寻找任何这些类型的条件求值，执行对判定 130 的确定。该确定是基于源代码中用于变量的修正关键字而做出的，所述修正关键字被编译在目标代码中。这种条件求值的形式列表被存储在存储器中。如果被调用方法的任何程序指令或语句都没有对固定的变量进行条件求值（判定 130，否定分支），则程序工具 30 循环回步骤 100 以取得并处理程序 32 中的下一个程序指令或语句。然而，如果被调用方法的任何程序指令或语句将要对固定变量进行条件求值（判定 130，肯定分支），则程序工具 30 确定固定

变量的条件求值是否总是导致立即/直接返回到被调用方法的调入者，而不存在得自该方法的任何成果，例如没有对另一方法的调用、没有有用的计算、没有表达式求值、以及没有语句求值（判定 132）. 程序工具 30 通过检查前述条件求值的最终指令来做出该确定，以确定是否是返回到调入者. 如果出现了某些成果（例如最终指令不是返回指令），则程序工具 30 进入步骤 100 以获取并处理程序 32 的下一个指令或语句. 然而，如果没有从固定变量的条件求值中获得任何成果（除了返回到调用者）（判定 132，肯定分支），则在步骤 134 中，程序工具 30 将删除调入方法中对被调入方法的调用（并从而避免执行条件求值）以及调入方法中计算用于对被调入方法的调用的参量所需的程序指令. 步骤 134 中的删除优化了程序 32 的后续执行，因为现在在程序 32 的后续执行期间将只处理更少的指令. 程序 30 通过重写目标代码来执行实际的指令删除. 运行时环境 XX 15 包含用于修改（删除）当前运行的目标代码的工具. 在上述步骤中由程序 30 标识出每个要被删除的语句. 然后，程序 30 从方法调入返回，即，将程序计数器设置为等于得自条件求值的程序指令（步骤 148）. 然后，程序 30 前进到步骤 100 以获取并处理程序 32 的这个指令.

尽管由程序 32 调用的方法或者由该方法调用的任何方法所执行的操作对本发明来说不重要，但是下面是一个示例. 在该示例中，程序 32 定义了含有两个函数方法的类 C. 下面是类 C 的伪代码:

类 C 的伪代码

```
ClassBody:
    ClassBodyDeclarations:
    FieldDeclaration:
        PCI Field X

    ConstructorDeclaration:
        SimpleTypeName:
            FormalParameterList:
```

Empty

ConstructorBody:

Assignment X=True

MethodDeclaration:

Method A

MethodDeclaration:

Method B

类 C 的前述定义在主体的前三行中指示出变量“X”是固定的或构造后不可变的(“PCI”). 类 C 的前述定义在主体的中间六行中指示出不存在用于类 C 的构造函数的参数，并且变量“X”的值为“真”. 类 C 的前述定义在主体的最后四行中指示出方法 A 和方法 B 被包含在类 C 中.

在该示例中，下面是方法 A 的伪代码:

方法 A 的伪代码

Method A:

MethodHeader:

MethodModifiers(opt) ResultType MethodDeclarator Throws(opt)

ResultType:

Void

MethodDeclarator:

Formal ParameterList:

Empty

MethodBody:

Block:

MethodInvocation (Method B)

FormalParameterList:

Expression:

String1+String2+String3

方法A的前述定义在方法头部段中指示出该方法没有返回值并且没有接受参量。方法 A 的前述定义在方法主体段中指示出方法 A 包含调入方法 B 的调用。方法A的前述定义在方法主体段中指示出调入方法B的调用需要通过连接字符串（定义程序 32 的使用者）而形成的参量，即，连接String1+String2+String3。在调用方法B之前，方法A在调入方法B之前连接前述的字符串。

在该示例中，下面是方法B的伪代码:

方法B的伪代码

```
Method B:
    MethodHeader:
        MethodModifiers(opt) ResultType MethodDeclarator Throws(opt)
            ResultType:
                Void
            MethodDeclarator:
                Formal ParameterList:
                    String1
    MethodBody:
        Block:
            IfThenElseStatement:
                If (X not equal True)
                    Then:
                        Expression involving String1
                    Else:
                        Return
```

方法B的前述定义在方法头部段中指示出方法B没有返回值，并且接受了被标为“String1”的串类型的一个参量。方法B的前述定义在方法主体段中指示出方法 B 执行对变量“X”的条件求值。即，“如果X不为真，则执行涉及String1的操作”。如果X为真，则立即/直接返回到调入者”（不

发生任何成果，例如，不调入另一方法、不执行有用的计算、没有表达式求值以及没有语句求值）.

在程序 32 的执行期间，运行时解释器 XX 产生类 C 的新实例. 在创建类 C 的该实例期间，运行时解释器 XX 记录变量“X”是固定的（如编译器所指出的那样），并且向程序 32 提供用于方法 A 的寻址信息. 当程序 32 随后被调入并执行时，它调入方法 A，方法 A 在步骤 112 中开始执行. 作为响应，方法 A 首先在步骤 116 中通过连接 String1 + String2 + String3 来计算用于调用方法 B 的参量. 在计算参量之后，方法 A 在步骤 120 中调入方法 B. 在方法 A 和方法 B 被执行的同时，程序 30 在步骤 122 中监视方法 A 和方法 B 的指令语句. 在该监视期间，程序 30 注意到，在方法 B 中，变量“X”是固定的并且正与另一固定变量或固定值进行比较，并且结果是立即/直接返回到调入者（方法 A）而不产生任何成果（判定 130，肯定分支）. 从而，通过从程序 32 的方法 A 中去除对方法 B 的调用以及相关联的计算用于对方法 B 的调用的参量的指令，程序 30 优化程序 32. 在程序 30 优化方法 A 之后，在步骤 134 中，包括在方括号[]中的下述步骤被从方法 A 中去除:

方法 A 的优化伪代码

```
Method A:
    MethodHeader:
        MethodModifiers(opt) ResultType MethodDeclarator Throws(opt)
            ResultType:
                Void
            MethodDeclarator:
                Formal ParameterList:
                    Empty
    MethodBody:
        Block:
            [MethodInvocation (Method B)]
```

[FormalParameterList:]

[X]

[Expression:]

[String1+String2+String3]

这样，在程序 32 的下一次迭代期间，当程序 32 调用方法 A 时，方括号[]中包括的前述步骤不会被执行。这减少了程序 32 的处理时间。

基于前述内容，已公开了一种在运行时期间优化计算机程序的系统、方法与程序产品。然而，在不脱离本发明的范围的条件下，可以做出多种修改和替换形式。因此，本发明是以示例性而非限制性的方式被公开的，并且应该参考以下权利要求来确定本发明的范围。

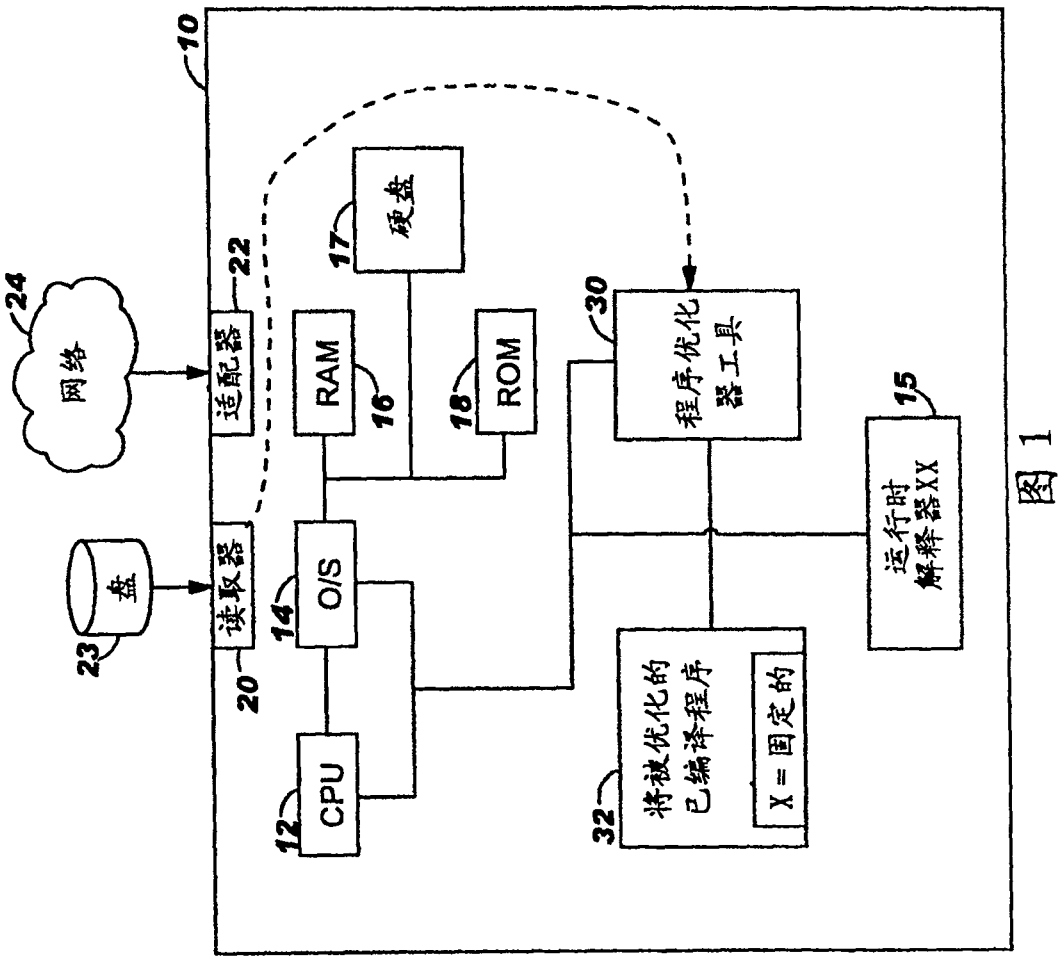

10
网络
24
适配器
22
硬盘
17
RAM
16
ROM
18
程序优化器工具
30
运行时解释器XX
15
盘
23
读取器
20
O/S
14
CPU
12
将被优化的已编译程序
32
X = 固定的

图 1

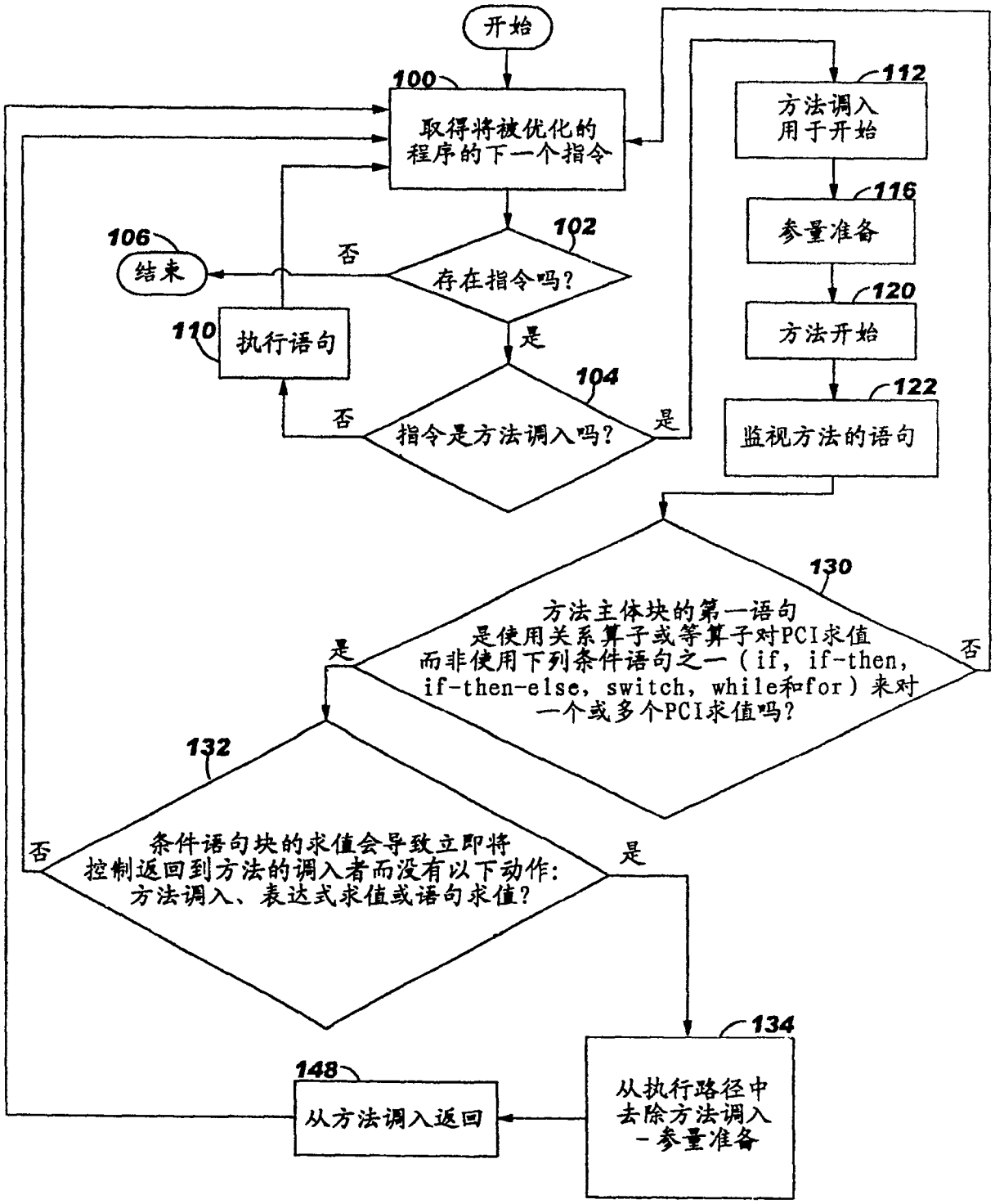

开始
100
取得将被优化的程序的下一个指令
102
存在指令吗?
否
106
结束
是
110
执行语句
104
指令是方法调入吗?
否
是
112
方法调入用于开始
116
参量准备
120
方法开始
122
监视方法的语句
130
方法主体块的第一语句是使用关系算子或等算子对PCI求值而非使用下列条件语句之一（if，if-then，if-then-else，switch，while和for）来对一个或多个PCI求值吗?
是
否
132
条件语句块的求值会导致立即将控制返回到方法的调入者而没有以下动作:方法调入、表达式求值或语句求值?
否
是
134
从执行路径中去除方法调入-参量准备
148
从方法调入返回

图 2