(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 104317773 A
(43)申请公布日 2015.01.28

(21)申请号 201410589530.4

(22)申请日 2014.10.28

(71)申请人 南京大学
地址 210093 江苏省南京市栖霞区仙林大道163号

(72)发明人 汤恩义 刘璐 方园 李宣东 冯世宁 张庆垒

(74)专利代理机构 南京瑞弘专利商标事务所（普通合伙） 32249
代理人 杨晓玲

(51)Int.Cl.
G06F 17/10(2006.01)

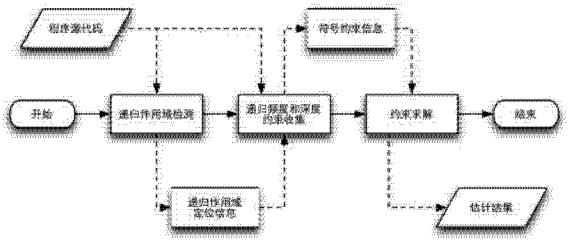
权利要求书2页 说明书5页 附图3页

(54)发明名称

一种递归最大执行频度与深度的静态估计方法

(57)摘要

本发明提出了一种递归最大执行频度与最大执行深度的静态估计方法，该方法通过静态扫描程序源代码以定位递归作用域并收集递归中函数调用与返回的执行条件，然后引入可满足性模求解器来求解这些收集到的符号条件约束，并最终直接获得估计结果。由于避免了反复执行程序，相比于传统的动态估计方法，本发明的方法具有更为高效、准确等优点。所分析的结果可以帮助程序设计、开发、维护人员用于性能优化、能耗估计、实时性分析等计算机应用技术领域。



CN 104317773 A

1. 一种递归最大执行频度与深度的静态估计方法，其特征在于通过静态程序分析的方法来收集程序的递归路径条件，并使用可满足性模理论来求解该递归路径条件，从而能高效准确的估计程序中递归作用域的最大执行深度，以及作用域中各递归函数的最大执行频度，具体步骤为：

1-1)、使用基于调用图的回路查找技术来检测程序中的递归作用域，所检测到的递归作用域为一个函数集合，集合中的各个函数相互调用，而构成递归，在检测过程中，使用强连通组件扩展技术来扩展调用图的回路，从而保证了所得到的函数集合包含当前递归作用域中的每一个函数；

1-2)、分别分析步骤 1-1) 所得到的各个递归作用域，收集作用域中各个递归函数调用和递归函数返回的分支条件与路径条件，并将相同递归函数的调用条件和返回条件合并，作为符号条件约束储存，以供步骤 1-3) 进行综合求解；

1-3)、将各递归作用域放回原始程序中进行综合分析与求解，即在原始程序中求得各个递归作用域的初始入口条件，并依据此入口条件与步骤 1-2) 给出的符号条件约束综合构建不同执行深度与执行频度的实际递归执行约束，最终使用可满足性模求解器对这些实际的递归执行约束求解。通过结合不同的递归深度与频度尝试策略，求解器能估计出递归实际可达的最大执行频度和深度。

2. 根据权利要求 1 所述的一种递归最大执行频度与深度的静态估计方法，其特征在于所述的步骤 1-1) 中使用强连通组件扩展技术来扩展调用图的回路的递归检测方法，使得所得到的函数集合必然包含当前递归作用域中的每一个函数，具体如下：

2-1)、由程序源代码按照传统方法构建调用图，调用图是一个用来表示程序调用关系的有向图，图中每一条有向边 (f, g) 代表程序中存在函数 f 到函数 g 的调用关系，因此，调用图上的任意回路表明在程序中存在函数递归关系；

2-2)、查找调用图中的回路，具体方法为按次序从调用图中的节点出发，尝试是否能够沿图中的有向边回到当前节点，通过不断的迭代，算法能够遍历调用图中所有的可能而找到回路，或者证明图中不存在回路；

2-3)、在步骤 2-2) 找到回路以后，使用强连通组件扩展算法扩展在调用图中的回路，从而保证所找到的递归作用域完整包含了所有的相关函数，即当递归作用域中存在子递归回路时，本算法保证找到的是整个递归作用域，在由步骤 2-1) 生成的调用图上反复迭代步骤 2-2) 和步骤 2-3)，即可得到程序中的所有递归作用域。

3. 根据权利要求 1 所述的一种递归最大执行频度与深度的静态估计方法，其特征在于通过静态程序分析的方法来收集程序的递归路径条件，以便后续的步骤来进一步求解递归的最大执行频度与深度，步骤 2-1) 到步骤 2-3) 所检测得到的递归作用域被标记为一个程序函数的集合 $\{f_1, f_2, \ldots, f_n\}$，这 n 个函数会通过相互调用而构成递归，特殊地，如果仅有单个函数参与递归，则 $n = 1$，在此基础上收集各递归条件，并构建符号约束的具体步骤如下：

3-1)、对于每一个递归作用域的每一个递归函数 f_i，扫描其源代码，定位其所有的函数出口即函数返回位置，和递归函数调用位置即调用当前递归作用域中的函数 f_j 的位置；

3-2)、分析递归函数 f_i 到其内部各个递归调用与函数出口的路径，并使用约束收集器收集路径上的分支条件，将每一条路径上的各个分支条件按逻辑合取式合并作为整个路径

条件；

3-3)、对于递归函数 f_i 中所有调用递归函数 f_j 的不同路径的路径条件，将按逻辑析取式合并，记为条件约束 $c_{i,j}$；而对于递归函数 f_i 到达所有出口的路径条件，也按逻辑析取式合并，记为条件约束 c'_i；这些条件约束信息将用于后续步骤的约束求解。

4. 根据权利要求 1 所述的一种递归最大执行频度与深度的静态估计方法，其特征在于使用可满足性模理论来求解该递归路径条件，具体过程如下：

4-1)、首先要从程序入口开始，沿程序控制流找到各递归作用域的入口，并由此路径分析得到当前递归作用域的初始条件，以便结合步骤 3-3) 的条件约束构建当前程序在具体执行到某一递归深度或频度的实际约束条件；

4-2)、按照递推式策略确定一个可满足性模理论的求解验证目标，即以递归深度 p 为目标，或者以某一递归函数的执行频度 q 为目标，并按照这一目标从递归作用域入口向内构建调用过程链，在结合步骤 4-1) 的递归初始条件和步骤 3-3) 的条件约束的基础上，按照调用过程链生成验证目标的实际组合约束条件；

4-3)、将步骤 4-2) 生成的实际组合约束条件转换成可满足性模的求解公式，并交由求解器求解，求解结果会有两种可能：当求解结果为当前公式真时，即当前条件可满足，表示当前递归最大深度 $\geq p$，或者某一递归函数的执行频度 $\geq q$，在这种情况下，如果不知道递归最大深度或者最大执行频度的上界，则以递推式策略尝试更大的 p 或者 q；当求解结果为假，当前条件不可满足，表示当前递归的最大深度一定达不到 p 或者执行频度 $< q$，此时 p 或 q 为验证目标的上界，本方法将继续以二分查找的方式尝试更小的 p 或者 q，迭代直到满足 $p-1 \leq$ 递归最大深度 $< p$ 或者 $q-1 \leq$ 递归函数最大执行频度 $< q$，由于目标值为整数，故而在此时我们能得到递归最大深度值 $= p-1$ 或递归函数最大执行频度值 $= q-1$。

一种递归最大执行频度与深度的静态估计方法

技术领域

[0001] 本发明属于计算机程序分析应用领域，涉及利用静态程序分析、可满足性模条件约束求解等技术，高效求解程序递归最大执行频度与最大执行深度，从而为设计开发人员进行程序优化，能耗估计，时间分析等提供依据，为一种程序递归最大执行频度与最大执行深度的静态估计方法。

背景技术

[0002] 递归作为一种算法设计思想，在程序设计与开发中被广泛使用。实际工业程序中的递归具有执行密度大，占用执行时间长等特点，对程序的执行效率，计算机的性能和吞吐量有重要影响。因此，递归作用域的最大执行频度和深度是程序优化，时间分析，系统能耗分析等领域所关注的重要指标。

[0003] 已有的递归执行频度和深度估计方法一般采用动态测试方法，通过生成测试用例对程序反复执行，来获得递归作用域可能达到的执行频率和执行深度。这样的方法有以下几个不足：1. 检测效率不高：由于动态方法需要反复执行程序，因此需要一定的执行量才能检测到所关心的递归作用域的相应指标；2. 检测结果准确度不够：由于动态检测方法依赖于测试用例，当测试用例的覆盖度不够时，未覆盖到的递归作用域是无法获得检测结果的；即使测试用例能覆盖到的递归，也很难保证测试用例能使得递归执行到最大的频度和深度。

[0004] 近年来，计算机自动化的符号约束求解能力大大提升，为能自动地静态检测递归执行频度和深度提供了可能。本发明基于可满足性模理论，通过静态收集与求解不同深度与频度的递归符号执行条件，来获得递归能在程序执行中达到的最大执行频度与深度信息，能够在较少的计算时间内达到较为准确的检测结果。

[0005] 可满足性模理论是计算机科学领域中在一阶谓词逻辑等式的基础上发展起来的逻辑公式判定理论，主要用于判断一阶逻辑公式是否可被满足。由于许多领域的实际问题最终都可以被归结为可满足性模理论问题（例如软硬件的形式化验证问题、电路的自动向量测试生成、人工智能中的规划与优化问题等），可满足性模理论近年来受到计算机理论界的广泛关注。目前广泛使用的可满足性模理论求解方法大约有三种：第一种方法将一阶谓词逻辑公式转换成更为简单的布尔公式，然后在此基础上求解布尔公式的可满足性；第二种方法将需要求解的内容根据其特性而划分成专门的理论域，并在各个理论域分别求解；第三种方法将该问题的求解归结到一种基于回溯的搜索算法框架中，并通过置换理论域，而达到高效可满足性求解的目的。经过研究者多年的努力，可满足性模理论求解算法得到了不断的改进与完善，已经能够解决实际应用中规模较大的问题。本发明利用可满足性模理论来静态求解递归作用域的执行条件约束，从而实现递归最大执行频度与最大执行深度的静态估计。

发明内容

[0006] 技术问题：本发明提出了一种递归最大执行频度与最大执行深度的静态估计方法，该方法不需要反复执行程序，即可以较为准确的估计出程序递归的最大执行频度和最大执行深度。

[0007] 技术方案：本发明的一种递归最大执行频度与深度的静态估计方法，通过静态程序分析的方法来收集程序的递归路径条件，并使用可满足性模理论来求解该递归路径条件，从而能高效准确的估计程序中递归作用域的最大执行深度，以及作用域中各递归函数的最大执行频度，具体步骤为：

[0008] 1-1)、使用基于调用图的回路查找技术来检测程序中的递归作用域，所检测到的递归作用域为一个函数集合，集合中的各个函数相互调用，而构成递归，在检测过程中，使用强连通组件扩展技术来扩展调用图的回路，从而保证了所得到的函数集合包含当前递归作用域中的每一个函数；

[0009] 1-2)、分别分析步骤 1-1) 所得到的各个递归作用域，收集作用域中各个递归函数调用和递归函数返回的分支条件与路径条件，并将相同递归函数的调用条件和返回条件合并，作为符号条件约束储存，以供步骤 1-3) 进行综合求解；

[0010] 1-3)、将各递归作用域放回原始程序中进行综合分析与求解，即在原始程序中求得各个递归作用域的初始入口条件，并依据此入口条件与步骤 1-2) 给出的符号条件约束综合构建不同执行深度与执行频度的实际递归执行约束，最终使用可满足性模求解器对这些实际的递归执行约束求解。通过结合不同的递归深度与频度尝试策略，求解器能估计出递归实际可达的最大执行频度和深度。

[0011] 所述的步骤 1-1) 中使用强连通组件扩展技术来扩展调用图的回路的递归检测方法，使得所得到的函数集合必然包含当前递归作用域中的每一个函数，具体如下：

[0012] 2-1)、由程序源代码按照传统方法构建调用图，调用图是一个用来表示程序调用关系的有向图，图中每一条有向边 (f, g) 代表程序中存在函数 f 到函数 g 的调用关系，因此，调用图上的任意回路表明在程序中存在函数递归关系；

[0013] 2-2)、查找调用图中的回路，具体方法为按次序从调用图中的节点出发，尝试是否能够沿图中的有向边回到当前节点，通过不断的迭代，算法能够遍历调用图中所有的可能而找到回路，或者证明图中不存在回路；

[0014] 2-3)、在步骤 2-2) 找到回路以后，使用强连通组件扩展算法扩展在调用图中的回路，从而保证所找到的递归作用域完整包含了所有的相关函数，即当递归作用域中存在子递归回路时，本算法保证找到的是整个递归作用域，在由步骤 2-1) 生成的调用图上反复迭代步骤 2-2) 和步骤 2-3)，即可得到程序中的所有递归作用域。

[0015] 所述通过静态程序分析的方法来收集程序的递归路径条件，以便后续的步骤来进一步求解递归的最大执行频度与深度，步骤 2-1) 到步骤 2-3) 所检测得到的递归作用域被标记为一个程序函数的集合 $f_1, f_2, \cdots, f_n$，这 n 个函数会通过相互调用而构成递归，特殊地，如果仅有单个函数参与递归，则 $n = 1$，在此基础上收集各递归条件，并构建符号约束的具体步骤如下：

[0016] 3-1)、对于每一个递归作用域的每一个递归函数 f_i，扫描其源代码，定位其所有的函数出口即函数返回位置，和递归函数调用位置即调用当前递归作用域中的函数 f_j 的位置；

[0017] 3-2)、分析递归函数 f_i 到其内部各个递归调用与函数出口的路径，并使用约束收集器收集路径上的分支条件，将每一条路径上的各个分支条件按逻辑合取式合并作为整个路径条件；

[0018] 3-3)、对于递归函数 f_i 中所有调用递归函数 f_j 的不同路径的路径条件，将按逻辑析取式合并，记为条件约束 $c_{i,j}$；而对于递归函数 f_i 到达所有出口的路径条件，也按逻辑析取式合并，记为条件约束 c'_i；这些条件约束信息将用于后续步骤的约束求解。

[0019] 使用可满足性模理论来求解该递归路径条件，具体过程如下：

[0020] 4-1)、首先要从程序入口开始，沿程序控制流找到各递归作用域的入口，并由此路径分析得到当前递归作用域的初始条件，以便结合步骤 3-3) 的条件约束构建当前程序在具体执行到某一递归深度或频度的实际约束条件；

[0021] 4-2)、按照递推式策略确定一个可满足性模理论的求解验证目标，即以递归深度 p 为目标，或者以某一递归函数的执行频度 q 为目标，并按照这一目标从递归作用域入口向内构建调用过程链，在结合步骤 4-1) 的递归初始条件和步骤 3-3) 的条件约束的基础上，按照调用过程链生成验证目标的实际组合约束条件；

[0022] 4-3)、将步骤 4-2) 生成的实际组合约束条件转换成可满足性模的求解公式，并交由求解器求解，求解结果会有两种可能：当求解结果为当前公式真时，即当前条件可满足，表示当前递归最大深度 $\geq p$，或者某一递归函数的执行频度 $\geq q$，在这种情况下，如果不知道递归最大深度或者最大执行频度的上界，则以递推式策略尝试更大的 p 或者 q；当求解结果为假，当前条件不可满足，表示当前递归的最大深度一定达不到 p 或者执行频度 $< q$，此时 p 或 q 为验证目标的上界，本方法将继续以二分查找的方式尝试更小的 p 或者 q，迭代直到满足 $p-1 \leq$ 递归最大深度 $< p$ 或者 $q-1 \leq$ 递归函数最大执行频度 $< q$，由于目标值为整数，故而在此时我们能得到递归最大深度值 $= p-1$ 或递归函数最大执行频度值 $= q-1$。

[0023] 有益效果：本发明所描述的程序递归最大执行频度与最大执行深度的静态估计方法，能够在不执行程序的状况下通过静态扫描程序源代码，来求解程序递归中各递归函数的最大执行频度与递归本身的最大执行深度。具体来说有如下有益效果：

[0024] 1. 相对于通过生成测试用例对程序反复执行的传统动态测试方法，本发明所提供的静态估计方法更能高效、准确地获得程序递归的最大执行频度与深度。一方面本发明使用可满足性模求解器来直接求解程序递归的约束条件，避免了反复执行程序，从而使得本方法能更为快速的达到目标；另一方面，由于本方法并不依赖于测试用例，而以逻辑符号条件的求解来获得结果，因而所得到的结果比传统方法更为准确。

[0025] 2. 本发明所生成的递归最大执行频度与深度具有可达性，即必然至少存在一个程序输入，在以该输入执行程序时，程序中的递归能达到估计的深度或者执行频度。由于本发明使用可满足性模求解器来求解程序递归的约束条件，当条件可满足时，求解器会输出满足当前条件的具体输入值，因此，以这一输入值运行程序时能保证估计结果的可达性。

[0026] 3. 本发明获得的程序递归最大执行频度与深度具有重要的应用价值。由于递归作用域是实际工业程序中高密度执行区，分析递归的最大执行频度和深度有助于帮助开发人员定位与分析是程序热点，以便其进一步对程序进行优化及能耗分析。另外，本发明所提供的程序递归频度与深度估计方法可以分析程序中多个递归作用域的组合效果，这可以帮助工业控制程序的设计与开发人员进行实时性分析。

附图说明

[0027] 图 1 为递归最大执行频度与深度静态估计的总流程。

[0028] 图 2 为递归作用域的检测流程。

[0029] 图 3 为递归作用域内部的符号约束收集流程。

[0030] 图 4 为程序递归组合约束收集与求解流程。

具体实施方式

[0031] 本发明提出了一种静态分析程序递归作用域最大执行频度和深度的方法，该方法通过扫描程序源代码来检测与收集递归作用域的相关符号条件约束，并引入了可满足性模求解器来求解约束，从而获得目标结果。以下部分就实施过程中的一些具体细节作更进一步的描述：

[0032] 一、关于递推式策略与二分回溯式策略等构建策略

[0033] 由于可满足性模求解器的求解结果为一个布尔值，即仅能得到当前条件是否能满足，因此，要得到递归最大频度和深度，我们还需要构建一个与频度与深度值相关的条件构造器，以辅助可满足性模求解器来进行工作。通过不断尝试不同的递归频度或者深度是否可满足，我们可以得到可以被当前条件满足的最大递归频度或深度值。本发明所设计的构建策略目标在于使得可满足性模求解器通过尽可能少的尝试次数来找到目标结果，该策略用于搜索递归最大频度和深度的过程是完全一样的，这里以递归最大深度 p 为例来描述构建策略。

[0034] 首先需要引入 p 的初始值 p_0，在实际工业上使用本发明方法的时候，可以根据经验估计一个递归最大深度的可能值来作为 p 的初始值，在完全没有经验指导的情况下，可以采用一个随机整数来作为 p 的初始值 p_0。

[0035] 当可满足性模求解器的求解结果为真时，表示 $p \geqslant p_0$，这时我们取到一个 p 的下界，而不知道 p 的上界，则下一次尝试则采用 $p_1 = 2 \times p_0$。事实上，当我们不知道 p 的上界的时候，则每一次尝试取前一次值的两倍，即 $p_{i+1} = 2 \times p_i$，以便能更快定位到 p 的上界。这样层层递推来寻找 p 的上界的过程在本发明中被称为递推式策略。

[0036] 当在某一次 p_i 的尝试中，可满足性模求解器的求解结果为假时，即表示 $p<p_i$，这时我们取到了 p 的上界，由于 p 的下界本来就已知（未知情况下取下界为 0），从而查找 p 的过程就被转化为二分搜索过程。即已知 p 的上界为 p_u，下界为 p_b，则下一次尝试值为上下界的中值 $p_{i+1} = \frac{1}{2} \times (p_u + p_b)$。如果求解器的结果为真，则用 p_{i+1} 代替 p_b，否则代替 p_u 进行反复迭代，直至 $p_u = p_b + 1$ 时算法终止，这时得到 $p = p_b$。

[0037] 这一过程在本发明中被称为二分回溯式策略。

[0038] 二、调用图的闭合回路搜索算法

[0039] 本发明采用了一种基于函数调用图的贪心算法来检测程序中递归作用域。具体来说，在一个调用图 g_c 中，每一个节点代表一个程序中的函数或过程，而每一条从节点 i 到节点 j 的有向边（记为 i, j）表示一个从过程 i 到过程 j 的调用。当我们使用 N_c 和 E_c 来表示调用图 g_c 的节点集和有向边集时，闭合回路搜索算法描述如下：

[0040] 2-2-1)、在调用图 g_c 中标示可达集 E'_c。将 E'_c 的初值设为 E_c，然后遍历 E'_c

中的每一条边，当存在首尾相连的两条边时，例如 i,k ∈ E′ $_c$ ∧ k,j ∈ E′ $_c$ 时，说明节点 i 到节点 j 可达，则将可达关系 i,j 也加入到 E′ $_c$ 中。

[0041] 2-2-2)、反复迭代步骤 2-2-1) 直至调用图 g_c 中所有的可达关系都被 E′ $_c$ 标定。然后对应于集合 N_c 中的每一个节点 k，查看可达关系 k,k 是否在可达集 E′ $_c$ 中，如果在可达集 E′ $_c$ 中，则表明调用图 g_c 中必然存在一条从节点 k 出发最终到达节点 k 的回路。从而可以将节点 k 输出至强连通组件扩展算法来获得当前回路对应的强连通组件。

[0042] 三、强连通组件扩展算法

[0043] 为了较为全面的处理更具有一般性的互递归，本发明在找到调用图中的回路以后，采用了强连通组件扩展算法来将回路扩展为调用图中的强连通组件，从而避免了直接使用调用图回路可能会取到嵌套递归作用域中的子回路的问题。有向图中的强连通组件是指该有向图中的子图，且要求子图中任意节点都可以沿有向边到达子图中的其他任意节点。

[0044] 本发明采用了一个可达遍历搜索的算法来扩展调用图中的回路成为一个强连通组件 S，具体算法如下：

[0045] 2-3-1)、首先将强连通组件 S 清空。

[0046] 2-3-2)、从步骤 2-2-2) 获得调用图 g_c 的回路节点 k，将节点 k 加入 S。沿此回路节点 k，从可达集 E′ $_c$ 中遍历所有能够被节点 k 到达的节点 h，即要求 (k,h) ∈ E′ $_c$。

[0047] 2-3-3)、通过可达集 E′ $_c$ 检查节点 h 是否能够到达节点 k，即如果 (h,k) ∈ E′ $_c$，则将节点 h 加入 S。

[0048] 2-3-4)、遍历所有 E′ $_c$ 中节点 k 出发的边，将满足条件的节点均加入 S，S 即所求的强连通组件。从而我们有：

[0049] $\forall h,(k,h) \in E'_c \wedge (h,k) \in E'_c \rightarrow S = S \cup \{h\}$

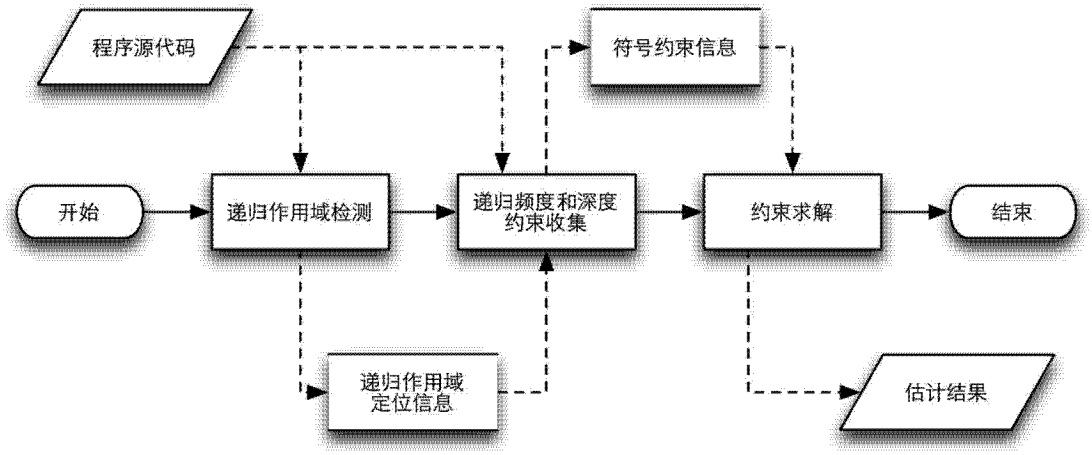

程序源代码
符号约束信息
开始
递归作用域检测
递归频度和深度
约束收集
约束求解
结束
递归作用域
定位信息
估计结果

图 1

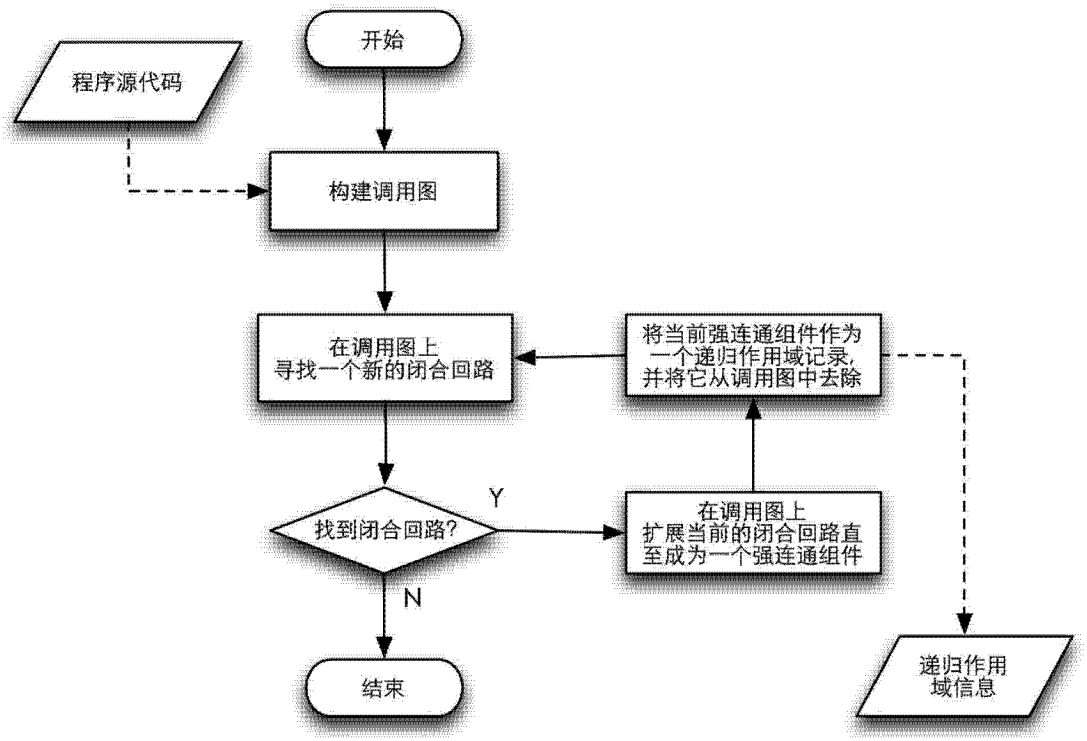

开始
程序源代码
构建调用图
在调用图上
寻找一个新的闭合回路
将当前强连通组件作为
一个递归作用域记录,
并将它从调用图中去除
找到闭合回路?
Y
N
在调用图上
扩展当前的闭合回路直
至成为一个强连通组件
结束
递归作用
域信息

图 2

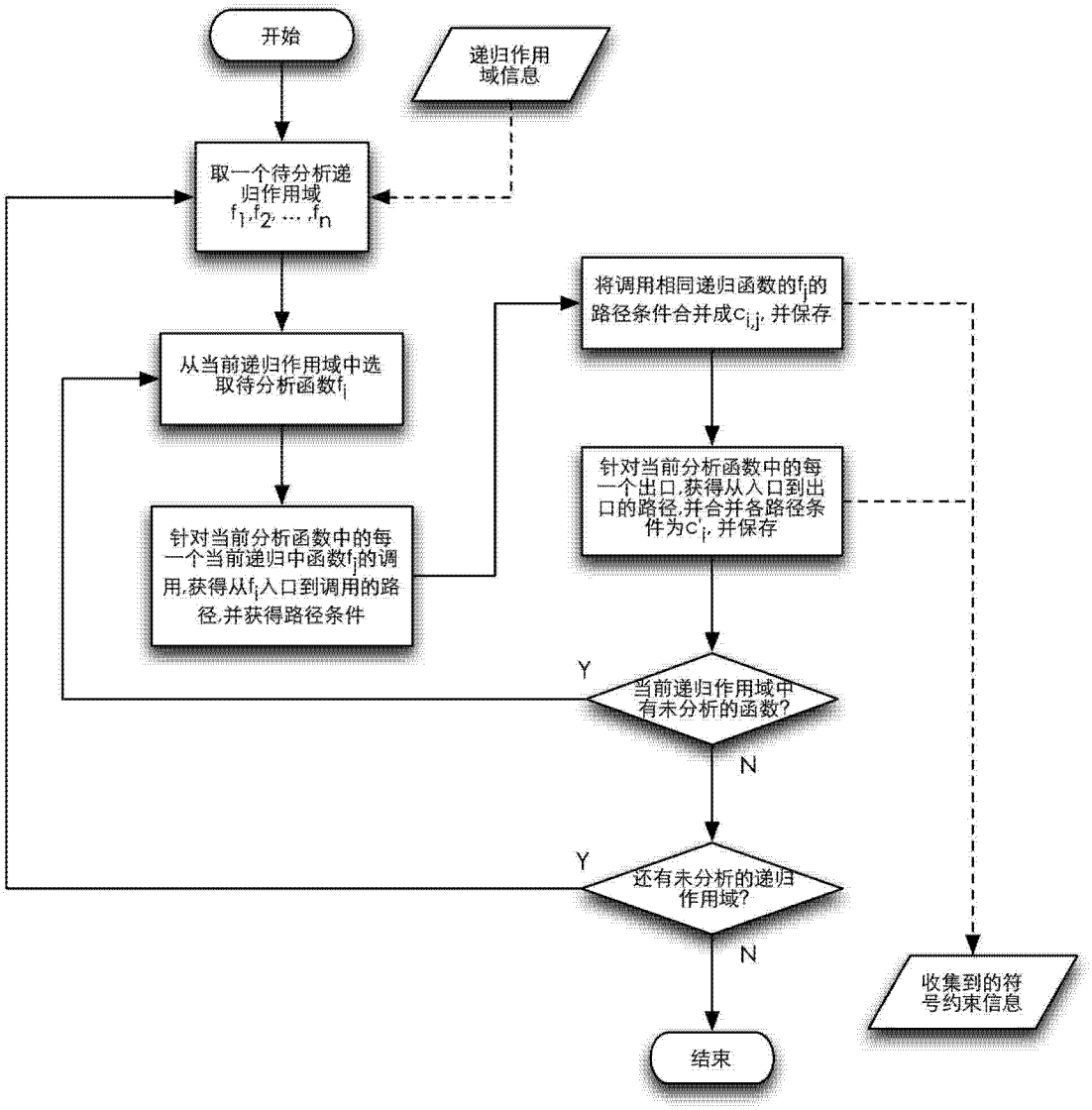

开始
递归作用域信息
取一个待分析递归作用域 $f_1, f_2, \ldots, f_n$
从当前递归作用域中选取待分析函数 f_i
针对当前分析函数中的每一个当前递归中函数 f_i 的调用,获得从 f_i 入口到调用的路径,并获得路径条件
将调用相同递归函数的 f_i 的路径条件合并成 $c_{i,j}$, 并保存
针对当前分析函数中的每一个出口,获得从入口到出口的路径,并合并各路径条件为 c'_i, 并保存
Y
当前递归作用域中有未分析的函数?
N
Y
还有未分析的递归作用域?
N
收集到的符号约束信息
结束

图 3

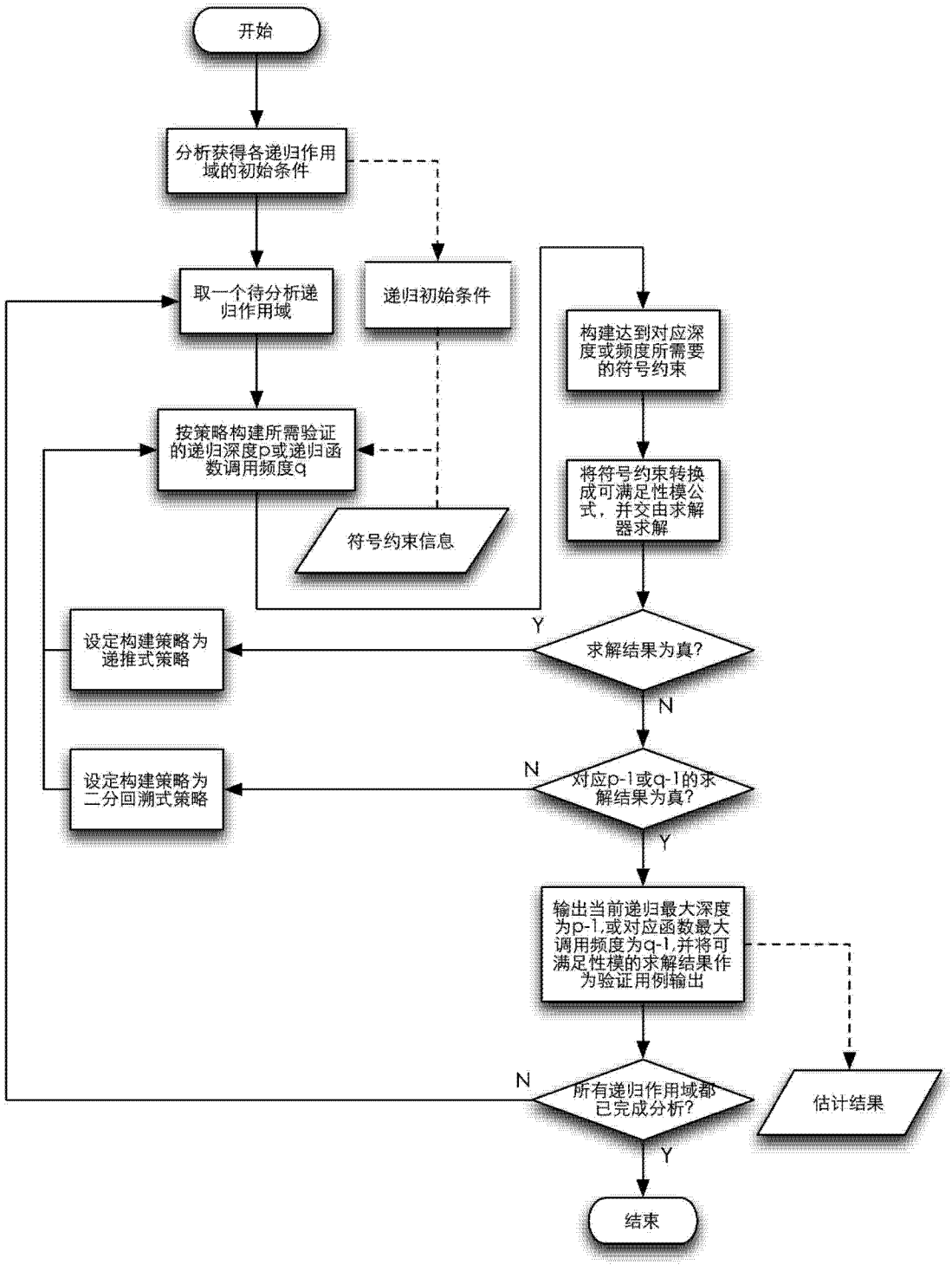

开始
分析获得各递归作用域的初始条件
取一个待分析递归作用域
递归初始条件
构建达到对应深度或频度所需要的符号约束
按策略构建所需验证的递归深度p或递归函数调用频度q
将符号约束转换成可满足性模公式，并交由求解器求解
符号约束信息
Y
求解结果为真?
设定构建策略为递推式策略
N
N
对应p-1或q-1的求解结果为真?
设定构建策略为二分回溯式策略
Y
输出当前递归最大深度为p-1,或对应函数最大调用频度为q-1,并将可满足性模的求解结果作为验证用例输出
N
所有递归作用域都已完成分析?
估计结果
Y
结束

图 4