(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 102867036 A
(43)申请公布日 2013.01.09

(21)申请号 201210312478.9

(22)申请日 2012.08.29

(71)申请人 北京工业大学
地址 100124 北京市朝阳区平乐园 100 号

(72)发明人 张正欣 张建

(74)专利代理机构 北京思海天达知识产权代理有限公司 11203
代理人 张慧

(51)Int.Cl.
*G06F 17/30*(2006.01)

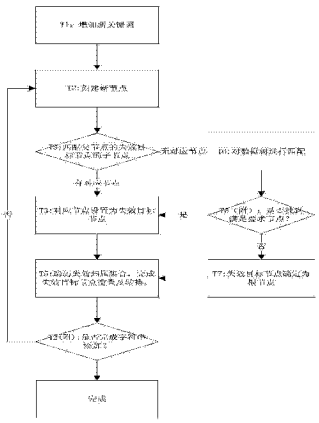
权利要求书 1 页 说明书 3 页 附图 2 页

(54)发明名称

实现 Aho-Corasick 算法所用数据结构动态生成的改进方法

(57)摘要

一种实现 Aho-Corasick 算法所用数据结构动态生成的改进方法，包括增加和删除特征字符串的操作；将特征字符串拆成单个字符，在 DFA 该位置上增加相应节点；在新节点设置相应的数据，检查父节点的失效目标；找到节点：踢出本节点指代字符串的第一个字符，用剩下的字符串对 DFA 进行匹配；找到失效目标的实现归属集合，遍历其中所有节点的引用，检查有无节点应该将本节点作为其失效目标节点；节点增加到 DFA 头部的字符集对象集合处；依次从后往前进行对字符串的减少工作；找到该对应节点。实现了对该数据结构的动态维护，方便实现了在较短的时间内对大量不断变动的字符串进行多模式匹配检索。



CN 102867036 A

1. 一种实现 Aho-Corasick 算法所用数据结构动态生成的改进方法，包括增加和删除特征字符串的操作；其特征在于：所述的增加特征字符串包括以下步骤：

步骤 1：将特征字符串拆成单个字符，经由 DFA 树进行单个匹配，当 DFA 中不存在相应字符时，在 DFA 该位置上增加相应节点；

步骤 2：在新节点设置相应的数据，检查父节点的失效目标，是否有对应于本节点指代字符的儿子节点。如有，设为本节点的失效目标，执行步骤 5；如无，执行步骤 3；

步骤 3：重复步骤 4，如找到节点，停止；如没有，再踢出一个字符串的头字符，重复步骤 4；如直到剩最后一个字符，都没有完成匹配，那么，执行步骤 6；

步骤 4：踢出本节点指代字符串的第一个字符，用剩下的字符串对 DFA 进行匹配，如找到符合的节点，将该节点作为失效目标，执行步骤 5，返回步骤 3；如没找到，也返回步骤 3；

步骤 5：找到失效目标的实现归属集合，遍历其中所有节点的引用，检查有无节点应该将本节点作为其失效目标节点，如有，设置；

步骤 6：节点增加到 DFA 头部的字符集对象集合处，如有该节点指代字符的字符集对象，添加该节点指针到对应字符集对象中，并遍历其中节点对象的引用，是否有对象应该讲本节点作为失效目标进行设置，如有，设置；

所述的减少特征字符串包括以下步骤：

步骤 7：依次从后往前进行对字符串的减少工作，单个字符进行操作，重复执行步骤 8，直到步骤 8 没有返回为止；

步骤 8：找到该对应节点，如本节点没有儿子节点，删除本节点，并返回步骤 7。

# 实现 Aho-Corasick 算法所用数据结构动态生成的改进方法

## 技术领域

[0001] 本发明属于计算机理论领域，用于为多模式字符串匹配的 Aho-Corasick 算法提供可动态加减的 Aho-Corasick-tree 数据结构。

## 背景技术

[0002] 随着信息技术的飞速发展，尤其在大数据处理的问题上，如何实现关键字段的快速检索是一个越来越突出的问题。尤其在 WEB2.0 时代，实时性的对大量数据进行遍历或搜索是一个常态化的操作。在如此大量数据量处理上，往往同时需要检索很多不同字符串，进行多模式匹配操作，这就需要用到 Aho-Corasick 算法。但此算法存在一个问题，作为一个自动机算法，它依靠的是来源于众多特征字符串而预先生成的树形数据结构，一旦在运行过程中需要增添或者删减特征串时，需要中断运行，删除以前的数据结构，重新生成新的数据结构。如果新的串集合较大，这样的步骤就需要相当的时间进行处理了。在此期间，数据的处理就无法及时反映出来，因此需要一种算法，即能保证实现多模式匹配，又能在有限时间内完成数据的重组操作。

## 发明内容

[0003] 本发明的目的在于针对上述算法的不足，，通过提供衣一种实现 Aho-Corasick 算法所用数据结构动态生成的改进方法，实现对该数据结构的动态维护，方便实现对大量不断变动的字符串进行多模式匹配检索。

[0004] 本发明是采用以下技术手段实现的：

[0005] 一种实现 Aho-Corasick 算法所用数据结构动态生成的改进方法，包括增加和删除特征字符串的操作；增加特征字符串包括以下步骤：

[0006] 步骤 1：将特征字符串拆成单个字符，经由 DFA 树进行单个匹配，当 DFA 中不存在相应字符时，在 DFA 该位置上增加相应节点；

[0007] 步骤 2：在新节点设置相应的数据，检查父节点的失效目标，是否有对应于本节点指代字符的儿子节点。如有，设为本节点的失效目标，执行步骤 5；如无，执行步骤 3；

[0008] 步骤 3：重复步骤 4，如找到节点，停止；如没有，再踢出一个字符串的头字符，重复步骤 4；如直到剩最后一个字符，都没有完成匹配，那么，执行步骤 6；

[0009] 步骤 4：踢出本节点指代字符串的第一个字符，用剩下的字符串对 DFA 进行匹配，如找到符合的节点，将该节点作为失效目标，执行步骤 5，返回步骤 3；如没找到，也返回步骤 3；

[0010] 步骤 5：找到失效目标的实现归属集合，遍历其中所有节点的引用，检查有无节点应该将本节点作为其失效目标节点，如有，设置；

[0011] 步骤 6：节点增加到 DFA 头部的字符集对象集合处，如有该节点指代字符的字符集对象，添加该节点指针到对应字符集对象中，并遍历其中节点对象的引用，是否有对象应该

讲本节点作为失效目标进行设置，如有，设置；

[0012] 减少特征字符串包括以下步骤：

[0013] 步骤 7：依次从后往前进行对字符串的减少工作，单个字符进行操作，重复执行步骤 8，直到步骤 8 没有返回为止；

[0014] 步骤 8：找到该对应节点，如本节点没有儿子节点，删除本节点，并返回步骤 7。

[0015] 本发明与现有技术相比，具有以下明显的优势和有益效果：

[0016] 本发明一种实现 Aho-Corasick 算法所用数据结构动态生成的改进方法，实现了对该数据结构的动态维护，方便实现了在较短的时间内对大量不断变动的字符串进行多模式匹配检索。即保证了实现多模式匹配，又能在有限时间内完成数据的重组操作。

## 附图说明

[0017] 图 1 为串的动态增加流程图；

[0018] 图 2 为串的动态删除流程图。

## 具体实施方式

[0019] 以下结合说明书附图对本发明的具体实施例加以说明。

[0020] 技术方案数据定义部分：需要用到的数据结构及组成部分的定义。

[0021] 定义 1：节点对象，包括：1）节点指代的字符；2）从根节点到达该节点时对应的字符串；3）节点的儿子节点引用集合；4）所有以该节点作为失效目标的节点引用集合（下文统称“失效归属集合”）；5）节点相对根节点的深度；6）节点的父节点引用；7）节点的失效目标节点引用；8）标记是否是字符串结束节点。

[0022] 定义 2：表头对象，包括：1）字符表对象集合。

[0023] 定义 3：根节点对象，包括：1）普通节点对象的一般信息；2）节点指代字符为空；3）节点失效目标指向自身。

[0024] 定义 4：字符表对象，包括：1）指代字符 2）封锁记号 3）所有节点字符与指代字符相同并且失效目标指向根节点的节点对象集合（下文统称称“失效对象归属集合”）。

[0025] 定义 5：Aho-Corasick 数据结构树对象，包括：1）根节点对象；2）表头对象；3）全体节点引用集合。

[0026] 算法步骤分为增加与删除两方面：

[0027] 请参阅图 1 所示，为串的动态增加流程图。

[0028] 动作 t1：增加新关键词：从根节点出发，单个字符地匹配需要增加的字符串，如有，则指向下一个；如没有，则创建一个新的节点 (t2)。

[0029] 动作 t2：创建新节点：1) 添加本节点的父节点引用；2) 在父节点的子节点引用集合中增加本节点引用；3) 设置本节点的指代字符；4) 设置本节点的指代字符串；5) 设置本节点的深度值；6) 将节点引用添加到数据树全体节点引用集合中；7) 如是新字符串的结束标识，请标记；8) 与本节点相关失效目标设置，执行动作 t3。

[0030] 动作 t3：首先找前节点的失效节点，查找其子节点集合中有无对应本节点指代字符的子节点，如有执行动作 t4；如没有，执行动作 t6。

[0031] 动作 t4：设置该节点为本节点的失效目标节点，并遍历该节点的失效归属集合，

对其内每个节点，执行动作 t5；遍历完毕后，将本节点，加入该失效归属集合，并设置本节点的失效目标节点为该节点。

[0032] 动作 t5：检查遍历节点的深度，如比本节点深度值低或等同，则遍历下一节点；如不是，检查该遍历节点指代的字符串的后面部分子串，与本节点指代字符串是否相同，若相同，则将该遍历节点从遍历集合中删除，加入本节点的失效归属集合，设置该节点的失效目标位本节点。

[0033] 动作 t6：将本节点的指代字符串的后面部分，长度为本节点深度小一的子串，放到数据树中进行匹配，如没有相应节点，再将子串长度减少一位再匹配；如在一子串上匹配成功，找到最终的匹配节点，执行动作 t4；如直到子字符串最后一位也没有，将失效目标指向根节点，执行动作 t7。

[0034] 动作 t7：失效目标指向根节点动作，检查数据树对象的字符表集合，有没有指代与本节点指代字符相同的字符表对象存在；如有，遍历该字符表对象的失效归属集合，对其内的每个节点，执行动作 t5。

[0035] 请参阅图 2 所示，为串的动态删除流程图。

[0036] 动作 t8：减少一个已有的串，实际上是减少一株数据节点；在数据树中，匹配该串，取得最后一个节点，执行动作 t9。

[0037] 动作 t9：设置变量“原节点”等于本节点，设定本节点指向本节点的父节点，执行动作 t10，直到本节点指向根节点。

[0038] 动作 t10：如本节点没有子节点，遍历本节点的失效归属集合，将其中的对象的失效目标改为本节点的失效目标，再将本节点的失效归属集合中的所有引用，添加到本节点的失效目标节点的失效归属集合中，如本节点的失效目标位根节点，则该添加到对应的字符表对象中的失效归属集合；在本节点的父节点的子节点集合中，删除本节点引用，删除本节点。

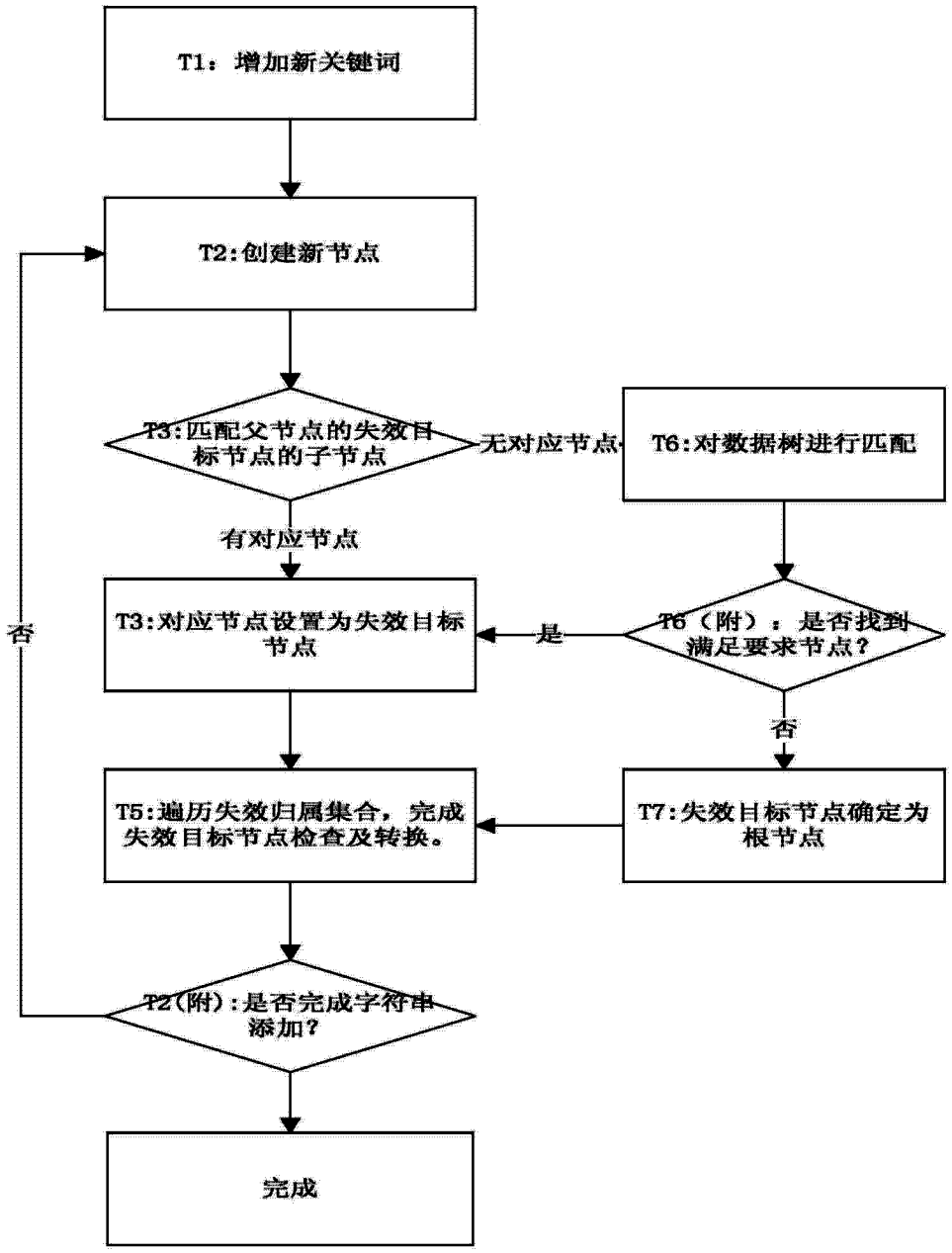

T1：增加新关键词
T2:创建新节点
T3:匹配父节点的失效目标节点的子节点
无对应节点
T6:对数据树进行匹配
有对应节点
T3:对应节点设置为失效目标节点
是
T6（附）：是否找到满足要求节点？
否
T5:遍历失效归属集合，完成失效目标节点检查及转换。
T7:失效目标节点确定为根节点
T2(附):是否完成字符串添加？
否
完成

图 1

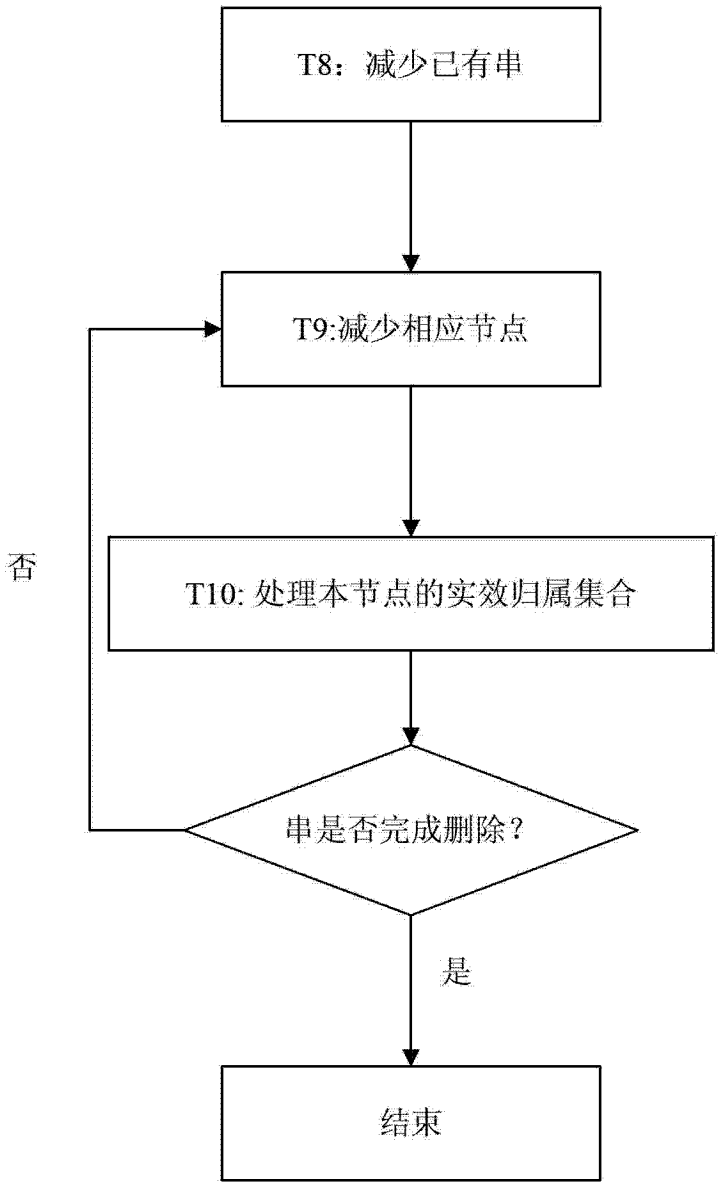



图 2