(19)中华人民共和国国家知识产权局

(12)发明专利

(10)授权公告号 CN 1552032 B
(45)授权公告日 2010.04.28

(21)申请号 01822185.8
(22)申请日 2001.11.28
(30)优先权数据
0029238.3 2000.11.30 GB
(85)PCT申请进入国家阶段日
2003.07.21
(86)PCT申请的申请数据
PCT/GB2001/005242 2001.11.28
(87)PCT申请的公布数据
WO2002/044940 EN 2002.06.06
(73)专利权人 考珀瑞耶有限公司
地址 英国威尔特郡
(72)发明人 D·G·保利
(74)专利代理机构 上海专利商标事务所有限公司 31100
代理人 亓云
(51)Int.Cl.
G06F 17/30(2006.01)

(56)对比文件
CN 1271442 A,2000.10.25, 全文.
US 5899992 A,1999.05.04, 全文.

审查员 齐霁

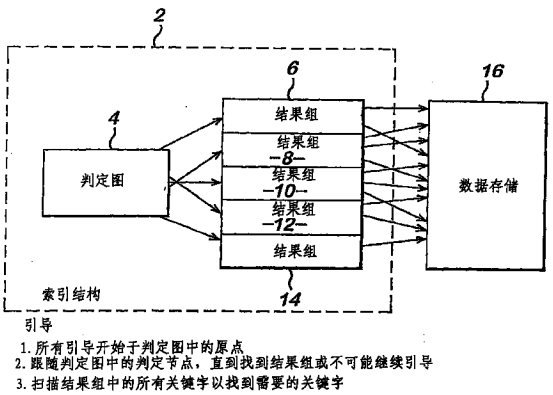
权利要求书 7 页 说明书 25 页 附图 27 页

(54)发明名称

数据库

(57)摘要

提供一数据库,其中提供数据库的索引(2)作为节点的层次结构,它在搜索过程中被引导,直到达到结果组。且该结构如此组织,使得对节点的关键字信息关系从该结构中节点的位置导出。



CN 1552032 B

1. 一种构建判定节点的分级结构的方法，所述判定节点用于使用时包含索引和数据的数据库，所述索引使用搜索关键字来查询以查找匹配一搜索标准的数据，所述索引是节点的分级结构，其在搜索期间被导航直至得到一结果组，且所述结构被组建使得所述搜索关键字与包括至少一比特的判定组比较，该判定组不保存在所述结构的一节点中，每一节点具有从其出发的最多两个出口路径，以及一些判定组包括多个比特。

2. 如权利要求 1 所述的方法，其特征在于，在一节点处检查的判定组与在之前的节点处检查的判定组是一样的。

3. 如权利要求 1 所述的方法，其特征在于，节点不保存与该节点相关的关键字的整体。

4. 如权利要求 1 所述的方法，其特征在于，节点不保存与节点的查询有关的判定规则。

5. 如权利要求 1 所述的方法，其特征在于，在一个节点用于查询的关键字或部分关键字与在该节点的判定准则比较，所应用的测试是查看该关键字或其部分是否大于或小于该判定准则。

6. 如权利要求 1 所述的方法，其特征在于，在数据库中的每条记录属于一结果组。

7. 如权利要求 1 所述的方法，其特征在于，使用中判定节点的分级结构形成判定图并保持在数据处理器的电子存储器中。

8. 如权利要求 6 所述的方法，其特征在于，每个结果组具有最大尺寸。

9. 如权利要求 8 所述的方法，其特征在于，该结果组的大小与在数据存储设备中使用的数据或地址块的大小相同，或是其整数倍。

10. 如权利要求 8 所述的方法，其特征在于，当结果组达到预定尺寸时，新的判定节点插入节点的分级结构，且在结果组中的数据重新索引到经新判定节点的出口路径达到的新结果组。

11. 如权利要求 10 所述的方法，其特征在于，新判定节点被插入到一个节点的指向达到预定大小的结果组的出口路径中。

12. 如权利要求 1 所述的方法，其特征在于，数据库的结构重组一次只影响 1 个或 2 个判定节点。

13. 如权利要求 1 所述的方法，其特征在于，判定节点与搜索关键字的部分相关。

14. 如权利要求 13 所述的方法，其特征在于，判定图很大程度上独立于关键字的结构。

15. 如权利要求 1 所述的方法，其特征在于，节点的分级结构构造成保持关键字序列的语义次序。

16. 如权利要求 6 所述的方法，其特征在于，结果组中的每个条目包含指向结果组中下一项的链接字段，保持与该结果组中的该条目有关的关键字的精确值的关键字字段，以及保持对应于该关键字的数据或指向对应于该关键字的数据的指针的属性字段。

17. 如权利要求 1 所述的方法，其特征在于，每个节点具有相关的判定值，它在使用时与关键字中的判定组比较，比较的结果用于选择该节点的出口路径。

18. 如权利要求 17 所述的方法，其特征在于，判定值选自下列之一：

该判定组的所有可能值的算术中值；

判定组的期望值的算术中值；

判定组的所有期望值的算术平均值；

在索引的准则处选择的预定值；以及

作为最近使用的判定值和前一个判定值的函数选择的值。

19. 如权利要求 17 或 18 所述的方法，其特征在于，在访问的判定节点处的对判定组的选择是选自下列之一：

对所有节点相同的判定组；

在每次对判定节点访问时都改变的判定组；

当前一节点的判定值达到当前判定组的判定范围的最大值或判定范围最小值时改变的判定组；以及

当连续节点之间的判定值改变小于预定阈值时改变的判定组。

20. 如权利要求 1 所述的方法，其特征在于，当加入数据时，索引被导航直到达到结果组，或发现需要创建结果组且随后数据被加到该结果组。

21. 如权利要求 1 所述的方法，其特征在于，一个或多个节点包含部分关键字序列。

22. 如权利要求 16 所述的方法，其特征在于，保存在结果组中的关键字以压缩方式存储。

23. 如权利要求 22 所述的方法，其特征在于，该压缩将该关键字映射到该关键字的一更小的表现形式。

24. 如权利要求 23 的方法，其特征在于，映射被执行为当映射应用到给定关键字时，它总能返回相同结果。

25. 如权利要求 22，23 或 24 所述的方法，其特征在于，使用至少二种不相同的算法压缩关键字。

26. 如权利要求 25 所述的方法，其特征在于，所述至少两种压缩的结果存入结果组。

27. 如权利要求 1 所述的方法，其特征在于，由运算子处理关键字，将关键字映射到有界范围。

28. 如权利要求 27 所述的方法，其特征在于，运算子在数据库中基本上均匀地分配关键字。

29. 如权利要求 27 或 28 所述的方法，其特征在于，从模数函数和散列函数之一选择运算子。

30. 如权利要求 1 所述的方法，其特征在于，保存在结果组中的关键字包括指出该关键字及其数据在数据库中保持期限的数据字段。

31. 如权利要求 30 所述的方法，其特征在于，过期的关键字和数据可用于被新数据改写。

32. 如权利要求 30 所述的方法，其特征在于，过期的关键字和数据不主动从数据库删除。

33. 如权利要求 30 所述的方法，其特征在于，当访问结果组时计算具有期限的每个条目的时效，计算结果被存储，或用于确定哪些条目能被改写。

34. 如权利要求 1 所述的方法，其特征在于，数据库包括关键字索引和判定索引，其中关键字索引中的关键字以压缩方式存储，且对关键字索引作出校验，查看在使用关键字查询判定索引之前该关键字是否存在。

35. 如权利要求 34 所述的方法，其特征在于，若在关键字索引中不存在该关键字，不搜索判定索引。

36. 如权利要求 34 所述的方法，其特征在于，在插入关键字的操作期间作出关键字索引的校验，若找到匹配的条目，该关键字作为重复而被拒绝。

37. 如权利要求 34 所述的方法，其特征在于，关键字索引中的关键字以压缩，编码或映射的形式存储。

38. 如权利要求 37 所述的方法，其特征在于，使用散列函数映射关键字。

39. 如权利要求 34 所述的方法，其特征在于，关键字索引包括一元素阵列，阵列中的元素数目至少和索引中独特的关键字元素的数目一样大。

40. 如权利要求 39 所述的方法，其特征在于，使用两个不同的函数，其中一个散列函数用作计算在关键字索引中目标元素 E(K) 的处理的部分，另一散列函数用于编码关键字，且结果 B(K) 存储在目标元素。

41. 如权利要求 40 所述的方法，其特征在于，在查询关键字索引期间，对目标关键字计算目标元素，且使用第二散列函数对目标关键字计算编码值，若编码值匹配在目标元素中已存在的值，该关键字被认为是重复的。

42. 如权利要求 1 所述的方法，其特征在于，该索引包括索引的分级结构，使得一个索引能委托其某些工作量给至少一个其他的索引。

43. 如权利要求 42 所述的方法，其特征在于，被委托工作的索引能委托工作给其它索引。

44. 如权利要求 42 所述的方法，其特征在于，由范围定义索引的关键字清单，且任何在该范围外的关键字上操作的请求被该索引拒绝。

45. 如权利要求 44 所述的方法，其特征在于，若关键字在数据库的索引范围内，数据库能在该关键字本身完成操作，或若该关键字在任何委托索引的清单范围内，委托该任务。

46. 如权利要求 42 所述的方法，其特征在于，每个索引与一物理存储设备相关，从而允许发生多个并发的盘存取。

47. 如权利要求 1 所述的方法，其特征在于，一节点可包括一关键字的子序列，将其与一候选关键字相比较以确定要采取的动作。

48. 一种数据库的判定节点的分级结构，所述数据库在使用时包含索引和数据，所述索引使用搜索关键字来查询以查找匹配一搜索标准的数据，所述索引是节点的分级结构，该结构在搜索期间被导航直至得到一结果组，且所述结构被组建使得在每个节点所述搜索关键字与包括至少一比特的判定组进行比较，该判定组不保存在所述结构的一节点中，每一节点具有从其出发的最多两个出口路径，以及一些判定组包括多个比特。

49. 如权利要求 48 所述的判定节点的分级结构，其特征在于，在一节点处检查的判定组与在之前的节点处检查的判定组是一样的。

50. 如权利要求 48 所述的判定节点的分级结构，其特征在于，节点不保存与该节点相关的关键字的整体。

51. 如权利要求 48 所述的判定节点的分级结构，其特征在于，节点不保存与节点的查询有关的判定规则。

52. 如权利要求 48 所述的判定节点的分级结构，其特征在于，在一个节点用于查询的关键字或部分关键字与在该节点的判定准则比较，所应用的测试是查看该关键字或其部分是否大于或小于该判定准则。

53. 如权利要求 48 所述的判定节点的分级结构，其特征在于，在数据库中的每条记录属于一结果组。

54. 如权利要求 48 所述的判定节点的分级结构，其特征在于，使用中判定节点的分级结构形成判定图并保持在数据处理器的电子存储器中。

55. 如权利要求 53 所述的判定节点的分级结构，其特征在于，每个结果组具有最大尺寸。

56. 如权利要求 55 所述的判定节点的分级结构，其特征在于，该结果组的大小与在数据存储设备中使用的数据或地址块的大小相同，或是其整数倍。

57. 如权利要求 55 所述的判定节点的分级结构，其特征在于，当结果组达到预定尺寸时，新的判定节点插入节点的分级结构，且在结果组中的数据重新索引到经新判定节点的出口路径达到的新结果组。

58. 如权利要求 57 所述的判定节点的分级结构，其特征在于，新判定节点被插入到一个节点的指向达到预定大小的结果组的出口路径中。

59. 如权利要求 48 所述的判定节点的分级结构，其特征在于，数据库的结构重组一次只影响 1 个或 2 个判定节点。

60. 如权利要求 48 所述的判定节点的分级结构，其特征在于，判定节点与搜索关键字的部分相关。

61. 如权利要求 60 所述的判定节点的分级结构，其特征在于，判定图很大程度上独立于关键字的结构。

62. 如权利要求 48 所述的判定节点的分级结构，其特征在于，节点的分级结构构造成保持关键字序列的语义次序。

63. 如权利要求 53 所述的判定节点的分级结构，其特征在于，结果组中的每个条目包含指向结果组中下一项的链接字段，保持与该结果组中的该条目有关的关键字的精确值的关键字字段，以及保持对应于该关键字的数据或指向对应于该关键字的数据的指针的属性字段。

64. 如权利要求 48 所述的判定节点的分级结构，其特征在于，每个节点具有相关的判定值，它在使用时与关键字中的判定组比较，比较的结果用于选择该节点的出口路径。

65. 如权利要求 64 所述的判定节点的分级结构，其特征在于，判定值选自下列之一：

该判定组的所有可能值的算术中值；

判定组的期望值的算术中值；

判定组的所有期望值的算术平均值；

在索引的准则处选择的预定值；以及

作为最近使用的判定值和前一个判定值的函数选择的值。

66. 如权利要求 64 所述的判定节点的分级结构，其特征在于，在访问的判定节点处的对判定组的选择是选自下列之一：

对所有节点相同的判定组；

在每次对判定节点访问时都改变的判定组；

当前一节点的判定值达到当前判定组的判定范围的最大值或判定范围最小值时改变的判定组；以及

当连续节点之间的判定值改变小于预定阈值时改变的判定组。

67. 如权利要求 48 所述的判定节点的分级结构，其特征在于，当加入数据时，索引被导航直到达到结果组，或发现需要创建结果组且随后数据被加到该结果组。

68. 如权利要求 48 所述的判定节点的分级结构，其特征在于，一个或多个节点包含部分关键字序列。

69. 如权利要求 63 所述的判定节点的分级结构，其特征在于，保存在结果组中的关键字以压缩方式存储。

70. 如权利要求 69 所述的判定节点的分级结构，其特征在于，该压缩将该关键字映射到该关键字的一更小的表现形式。

71. 如权利要求 70 的判定节点的分级结构，其特征在于，映射被执行为当映射应用到给定关键字时，它总能返回相同结果。

72. 如权利要求 69 所述的判定节点的分级结构，其特征在于，使用至少二种不相同的算法压缩关键字。

73. 如权利要求 72 所述的判定节点的分级结构，其特征在于，所述至少两种压缩的结果存入结果组。

74. 如权利要求 48 所述的判定节点的分级结构，其特征在于，由运算子处理关键字，将关键字映射到有界范围。

75. 如权利要求 74 所述的判定节点的分级结构，其特征在于，运算子在数据库中基本上均匀地分配关键字。

76. 如权利要求 74 所述的判定节点的分级结构，其特征在于，从模数函数和散列函数之一选择运算子。

77. 如权利要求 48 所述的判定节点的分级结构，其特征在于，保存在结果组中的关键字包括指出该关键字及其数据在数据库中保持期限的数据字段。

78. 如权利要求 77 所述的判定节点的分级结构，其特征在于，过期的关键字和数据可用于被新数据改写。

79. 如权利要求 77 所述的判定节点的分级结构，其特征在于，过期的关键字和数据不主动从数据库删除。

80. 如权利要求 78 所述的判定节点的分级结构，其特征在于，当访问结果组时计算具有期限的每个条目的时效，计算结果被存储，或用于确定哪些条目能被改写。

81. 如权利要求 48 所述的判定节点的分级结构，其特征在于，数据库包括关键字索引和判定索引，其中关键字索引中的关键字以压缩方式存储，且对关键字索引作出校验，查看在使用关键字查询判定索引之前该关键字是否存在。

82. 如权利要求 81 所述的判定节点的分级结构，其特征在于，若在关键字索引中不存在该关键字，不搜索判定索引。

83. 如权利要求 81 所述的判定节点的分级结构，其特征在于，在插入关键字的操作期间作出关键字索引的校验，若找到匹配的条目，该关键字作为重复而被拒绝。

84. 如权利要求 81 所述的判定节点的分级结构，其特征在于，关键字索引中的关键字以压缩，编码或映射的形式存储。

85. 如权利要求 84 所述的判定节点的分级结构，其特征在于，使用散列函数映射关键

字。

86. 如权利要求 81 所述的判定节点的分级结构，其特征在于，关键字索引包括一元素阵列，阵列中的元素数目至少和索引中独特的关键字元素的数目一样大。

87. 如权利要求 86 所述的判定节点的分级结构，其特征在于，使用两个不同的函数，其中一个散列函数用作计算在关键字索引中目标元素 E(K) 的处理的部分，另一散列函数用于编码关键字，且结果 B(K) 存储在目标元素。

88. 如权利要求 87 所述的判定节点的分级结构，其特征在于，在查询关键字索引期间，对目标关键字计算目标元素，且使用第二散列函数对目标关键字计算编码值，若编码值匹配在目标元素中已存在的值，该关键字被认为是重复的。

89. 如权利要求 48 所述的判定节点的分级结构，其特征在于，该索引包括索引的分级结构，使得一个索引能委托其某些工作量给至少一个其他的索引。

90. 如权利要求 89 所述的判定节点的分级结构，其特征在于，被委托工作的索引能委托工作给其它索引。

91. 如权利要求 89 所述的判定节点的分级结构，其特征在于，由范围定义索引的关键字清单，且任何在该范围外的关键字上操作的请求被该索引拒绝。

92. 如权利要求 91 所述的判定节点的分级结构，其特征在于，若关键字在数据库的索引范围内，数据库能在该关键字本身完成操作，或若该关键字在任何委托索引的清单范围内，委托该任务。

93. 如权利要求 89 所述的判定节点的分级结构，其特征在于，每个索引与一物理存储设备相关，从而允许发生多个并发的盘存取。

94. 如权利要求 48 所述的判定节点的分级结构，其特征在于，一节点可包括一关键字的子序列，将其与一候选关键字相比较以确定要采取的动作。

95. 一种导航判定节点的分级结构的方法，所述分级结构形成一数据库的索引，所述方法包含，使用搜索关键字导航索引以定位匹配该搜索关键字的数据，导航该索引直到到达一结果组，其中至少一个判定节点通过该搜索关键字相对于包含在该判定节点中的一关键字的子序列的模式匹配来导航，以确定该字序列是否被保存在该搜索关键字中从而确定要采取的动作。

96. 一种判定节点的分级结构，所述分级结构形成一数据库的索引，所述结构被导航直至到达一结果组，其中至少一个判定节点包含一关键字的子序列，且其中该索引被构建成该索引可通过该搜索关键字相对于包含在该判定节点中的一关键字的子序列的模式匹配来导航，以确定该字序列是否被保存在该搜索关键字中从而确定要采取的动作。

97. 如权利要求 96 所述的分级结构，其特征在于，至少一个判定节点包括一排斥指针和一包含指针，指向下一个节点或者结果组的地址的。

98. 如权利要求 97 所述的分级结构，其特征在于，至少一个判定节点包括一排斥指针类型和一包含指针类型，表示与它们相关的指针是否指向另一个判定节点或者是结果组。

99. 一个组织数据库的方法，其特征在于，用于引导索引的关键字是合成关键字，且其中一个或多个关键字以压缩形式表示。

100. 如权利要求 99 的方法，其特征在于，基本关键字是不压缩的，而合成关键字的其它关键字是压缩的。

101. 如权利要求99或100的方法，其特征在于，每个压缩的关键字作用如限定词，它用于细化使用数据库完成的搜索。

102. 如权利要求101的方法，其特征在于，使用主关键字引导索引直到达到结果组，且随后使用该限定词或每个限定词查询在结果组中的结果。

103. 如权利要求99的方法，其特征在于，这些关键字使用散列函数压缩。

数据库

技术领域

[0001] 本发明涉及索引，放入信息和查询数据库的方法。尤其是，本发明使能在数据存储系统中有效地搜索标量数据。

背景技术

[0002] 标量数据包括如逻辑数据，文本串，数字数据，和日期－时间数据。

[0003] 虽然数据处理器是快的，要查遍所有保存在存储系统中的数据，寻找具有指定性质或指定性质范围的特定项是低效且耗时的。而且，不可避免数据将存在海量存储介质中，由于盘存取与 CPU 速度及半导体存储器存取速度的比较，又增加了延迟。更有效的是建立索引，使得能使用搜索关键字检索。

[0004] 索引机制需要建立和维护内部索引结构。此结构的引导和维护本身引起处理开销。此开销能作为所使用的索引机制的函数而变化。当数据库的大小改变时，或当例如数据库中的数据随时间改变时，它也会改变。

[0005] 简单的例子是在存储如生日的编年信息的数据库中。例如，设计者可以那样构造数据，使得在公元 2000 年前出生的那些人具有相关的数据与在公元 2000 年后出生的那些数据分开。此策略可以在一段时间内有效，但可以看到，在公元 2000 年前出生的人的数目是有限止的组，而在此日子后出生的数目是无法约束的。因此，随时间经过，当公元 2000 年后出生的人的有关的实体部分增加时，索引关键字变成冗长的。

[0006] 针对若干属性判断索引方法的有效性能，如：

[0007] 1) 索引的大小。较大索引需要更多时间定位。它们也需要更多次数据交换或包括从如硬盘驱动器那样的海量存储介质和如半导体存储器那样更快的存储器传送的读操作。

[0008] 2) 索引结构上的约束。高度约束的结构提供初期的好处，但如果在数据库增大以适应更多数时需要重建索引，招致更大的计算开销。

[0009] 3) 由索引施加的关键字数据限止。因此，关键字或关键字的数据类型或由关键字表示的范围在索引增长时或在数据库随数据增殖时开始影响索引的性能。

[0010] 4) 由索引施加的关键字检索限止。某些索引只允许精确的关键字匹配，而另一些允许一定范围的匹配。

[0011] 5) 索引的增长极限。索引受最大尺寸限止。

[0012] 6) 并发限止。索引可以允许或禁止同时插入或检索关键字。

[0013] 由以前的技术已知。系统通常使用带有包括一个或多个关键字数据的项的节点的树形图。为查询数据库。通过将在每个访问的节点的关键字值与所需的搜索关键字比较来引导该树。在那样的设计中所固有的是，每个节点包括与建立该节点相关的关键字的整体，因此在数据存储项中该节点的“尺寸”是很大的。

[0014] 那样索引的有效性能根据关键字大小和数据被加到该索引的次序。

[0015] 此外，某些方法需要那些树在正确操作情况下总是“平衡的”。这些方法导致对关键字的插入和删除的很大的维护开销，使它们不适合于高数据通量的系统。

发明内容

[0016] 引导

[0017] 按本发明的第一方面，提供一个组织具有索引和数据的数据库的方法，其中使用包括至少一个符号的搜索关键字查询索引，至少一个符号由多个数据位表示，以便定位匹配搜索准则的数据，其特点是索引是判定节点的层次结构，在搜索期间节点的结构被引导，直到得到结果，且结构是那样组织，使得关键字符号不存储在结构中的节点上，并每个节点具有少于三个出口路径。

[0018] 因此有可能提供有效的索引结构，使能有效地搜索和更新数据库。该索引结构远小于以前技术的索引，因为节点不存储与该节点相关的整个关键字。实际上节点不需要存储任何与其相关的任何关键字。此外，节点不存储与查询节点有关的判定规则。这是因为规则是全局的，即它们被所有节点共享而不是节点专用的。

[0019] 每个判定节点作出二进制决策。即每个节点只具有最多两条出口路径。

[0020] 最好加到每个节点的判定是简单的查询。对此，这就意味着查询将关键字或关键字的部分与节点的判定准则比较，且通常（虽然不是必要的）以二进制测试的项目是被测试的关键字，或者其部分大于或小于判定准则。这就排除了在索引本身的结构中存贮搜索关键字的特征的需要。

[0021] 从一个节点的出口可能指向又一个判定节点，结果组或指向无效结果。

[0022] 若一个节点的出口指向另一个节点，则搜索继续。但若一个节点的出口指向结果组，则索引的引导一般结束。

[0023] 有可能没有匹配搜索项的结果，在此情况搜索以无效结果结束。

[0024] 在数据库的每个“记录”属于结果组。结果组本身指向满足特定搜索准则的数据，或者在某些情况可存储数据本身。

[0025] 判定节点的层次结构可看作判定图。判定图能是任意大小和内部结构。但是，就其在处理器的存储器中的尺寸而言，判定图通常是小的实体。

[0026] 判定图在使用时最好保持在数据处理器的电子存储器中。通过在 RAM 中保持相对小的判定图，数据处理器能迅速地找到查询结果。尤其是因为判定图的分析不需要存取磁的或其它物理介质，与那样介质的存取时间省去了，从而提高了性能。

[0027] 每个结果组最好只能由通过判定图的单一路径达到。这就保证在结果中的所有关键字符合共享的唯一的判定组。

[0028] 数据库索引能够，且实际上必须随时演变，而这能在数据库扩大时导致性能降低。因而需要允许在结果组中插入和删除数据，并还允许建立新的结果组。

[0029] 每个结果组最好具有最大尺寸。最大尺寸是由用户或设计者可控制的参数。但是，尺寸最好与由如包含一个或多个硬盘的磁存储器的存储设备使用的数据或编址数据块的大小有关。但是，可使用任何随机存取数据存储器，如光存储设备。

[0030] 当结果组接近或达到预定的尺寸时，新的判定节点插入到判定树，而曾是结果组的数据被重索引成路径新的判定节点的出口路径达到的新的结果组。新的判点节点被插入到曾指向超出其最大尺寸的结果组的节点的出口路径。

[0031] 因此结果组被构造成很好运作的组。

[0032] 判定图不包含任何搜索关键字数据本身。所有关键字数据是隐含的或从判定图的结构推导出。这就保持判定图紧凑。而且，这意味着判定图的尺寸无关于关键字的尺寸。

[0033] 本发明的又一个优点是能够对判定图的结构作出改变，且这些改变是局部的。即只有一个或少数判定节点受改变的影响。这意味着结构的重组没有很大的计算开销。

[0034] 判定节点最好有关关键字的部分，更好是有关它的相当小部分。这就得出结论，判定图很大程度上独立于关键字的结构。这也具有优点，判定图只对关键字的结构加上少量（若有的话）限止。此外，与节点相关的关键字的部分不必遵循判定图被引导的结构次序。因此所讨论的关键字的部分不必如图被引导那样的单调的方式沿着关键字进行。

[0035] 最好判定图那样地构造，使保持关键字序列的语义次序。因此例如，该判定图能构造成使得第一节点或节点组测试在搜索关键字中的第一位或第一位组，并使得当判定图导向结果组时，在随后节点被测试的关键字位变来越来越不重要。因此判定图能通过精确匹配或范围匹配被引导以定位关键字。在搜索中范围可以部分地或完全被限定。

[0036] 判定图的结构，尤其是结构的的改变是局部（即只影响一、二个节点），而不是全局的事实，意味着维护判定图的计算任务一般是低的。重组的局部化特征方便了使用搜索存入，删除和检索数据。事实上，两个或更多个这些事件能同时地发生。

[0037] 最好对判定图的尺寸没有功能上的极限。索引的结构在判定图为适应数据增加而增长时在搜索时间方面产生良好表现。

[0038] 按本发明的第二方面，提供具有索引的数据库，并安排成保存数据，允许使用搜索关键字通过查询索引而定位数据，该关键字至少包括一个符号，该符号或每个符号用多个数据位表示，其中索引是判定节点的层次结构，节点的结构在搜索期间被引导，直到得到结果，每个节点的搜索准则在索引中由节点的位置编码，且每个节点具有直到两个出口路径。

[0039] 索引中中局部化结构改变方便了在索引中的并发的插入，删除和检索事件。

[0040] 最好在数据库中的每个数据项能只属于一个结果组。

[0041] 判定图分解

[0042] 虽然强烈希望判定图整体驻留在半导体存储器中，这不总是可能的。这可以是由于索引变得太大，或由于主数据处理器对数据库实现“虚拟机”环境，使得主机能多任务，且操作系统对数据库不能分配足够的存储器资源。

[0043] 在那样情况，判定图能保存在计算系统的海量存储设备的存储器块中。盘控制器借助离散的数据块管理盘。因此，例如若由盘控制器使用的最小块尺寸是 64k 字节，则 40k 的索引和 60k 索引均占据盘上同样的存储空间，即一个块，而类似地 65k 索引和 130k 索引也占据同样的存储空间，即 2 块。

[0044] 为适应这些系统约束并用其工作，需要控制索引的修改，因为当在索引中的关键字总数增加时，索引的尺寸也增加。因此，为增加索引，需要更多的存储器块。

[0045] 在第一方案中，在需要插入新的判定节点且块已满时，为了增加索引建立新的块，并将新节点放入新块。判定图被修改，使得若新的块不总是驻留在半导体存储器中，指向新节点的节点从磁存储器取出新的块。然后引导从新节点继续。

[0046] 此方法简单但是低效，因为看来导致从每个父辈的满块建立许多，可能是数以百计的几乎空的存储块。在占用的空间方面这可能不认为是问题，因为盘能保存非常大量的数据，但是它可能牺牲索引性能，因为为引导索引需要更多的盘 I/O 操作。

[0047] 较好的方法是建立新的块并随后将父块（即满的块）基本相等地在两个块之间分解。此方法需要查询原始块以找到将块最好地分成两块的节点。这可以通过块中节点的从头开始的分析或借助本质上更加统计化的方法实行。因此，例如可使用递归算法，对此分解重复地挑选测试节点，计算由于节点挑选引起的索引的划分，并根据该结果修改测试节点的挑选，直到达到基本均匀的分配。

[0048] 此第二方法具有下述好处，每个索引块包含主要的节点总数。这导致有效得多的存储器使用。

[0049] 模式配对

[0050] 如前所注意到，以前技术的系统存储与任何给定方式相关的搜索关键字的整体。

[0051] 本发明的第一和第二方面揭示了树形结构，其中节点不存储搜索关键字，但关键字数据为该树的结构所固有。

[0052] 按本发明的第三方面，提供查询具有索引及数据的数据库的方法，且其中为了定位匹配搜索准则的数据，使用搜索关键字查询索引，其特点是索引是判定节点的层次结构，结构被引导直到达到结果组，且其中节点可以包含关键字的子序列，候选的关键字可以针对它进行比较以确定要采取的行动，且其中存储在第 N+1 层索引的节点处的关键字部分与存储在第 N 层的先前节点中的部分没有关系。

[0053] 如在这里使用的，在关键字部分不是通过以名义上变化的方式选择关键字的顺序的部分来计算的意义上，在第 (N+1) 个节点的关键字部分与在第 N 个节点的关键字无关。

[0054] 因此有可能提供这样的数据库，其中判定节点能保持相对“小”，因为每个节只存储关键字的子序列而不是整个关键字。此结构便于通过模式匹配作数据库引导，插入和检索。

[0055] 按本发明的第四方面，提供具有索引和数据的数据库，且其中使用搜索关键字查询索引，以便定位匹配搜索准则的数据，其特点是该索引是判定节点的层次结构，该结构被引导直到达到一结果组，且其中一节点能包含关键字的子序列，候选的关键字能针对它比较，以便确定要采取的动作。

[0056] 增强的模式匹配

[0057] 在模式匹配过程中很明白，在模式匹配中的计算负载是匹配的模式的长度的函数，且尤其是测试对短的字符或符号串的匹配比匹配长的串要较少的计算花费。

[0058] 此外，数据库 / 索引的有效性也取决于关键字总体如何在每个节点划分，且在查询中一个序列看来如何相似。

[0059] 这最好通过考虑两个较极端的例子来解释。假设索引借助商业希望交往的客户或其它实体的名字组织。

[0060] 经常出现的模式匹配常不充分地减少在索引查询期间扫描的关键字的总数。因此，例如若在模式匹配中使用的序列是“E”，则如果这看来在大多数关键字中出现，它在划分关键字总体上只提供很少帮助。

[0061] 作为又一个极端例子，一个不经常出现的序列也将不减少在大多数查询中必须扫描的关键字总数。因此，例如“zz”不太在查询中出现，所以对模式匹配查询不是好的候选。

[0062] 如上所述，字符更多包含在序列中，它在关键字中序列出现得更少。因而单个字符在带很多字符的关键字中出现，而 4 个字符的特定序列较少出现。

[0063] 在理想情况，查询串将从一节点达到的关键字的全体基本均匀地划分。

[0064] 伴随 ASCII 字符集的问题是使用该集完全 255 字符寻找适当的序列常常是困难的，因为有许多字符，但是某些字符出现的或然率 (likelihood) 比许多其它字符的要大得多。

[0065] 但是双字符组合能是较少的。

[0066] 按本发明的第五方面，提供在数据库的索引中存储关键字或其部分的方法，其中关键字的单元从构成该单元能取的至少某些值的表的第一组映射到小于第一组的第二组。

[0067] 因此在关键字中能出现的不同字符的数目被减少。映射不是唯一的，但若映射被恰当地选择，第二组的任何单个字符出现的或然率有理由能是高的，且对序列能使用两个或多个字符的组合。

[0068] 最好在为插入，删除或查询操作引导判定图之前将映射应用到关键字。

[0069] 最好原始的关键字值或经转换的关键字值（在需要存储的空间方面是较小）能存入结果组。

[0070] 按本发明的第六方面，提供一数据库，其中在索引引导前，关键字被映射到较小范围的值。

[0071] 关键字压缩

[0072] 上面提到，某些以前技术的数据库在每个判定节点处存储完全判定 / 搜索关键字。假如那些关键字本身很大，这能产生相当大的索引结构。还注意到，在组成本发明的实施例的数据库中搜索关键字不需存储在每个判定节点，虽然能希望在判定节点存储一部分搜索关键字。存储搜索关键字的部分选项仍导致能被快速引导的相当紧凑的数据库判定树。然而，发明者认识到，关键字不必以其原始形式存入索引中。因此关键字可编码成占据缩减空间的形式。

[0073] 按本发明的第七方面，提供在数据库中编码关键字的方法，其中关键字数据以编码形式存入至少一个结果组，以致于减少了关键字的存储空间需要。

[0074] 因此可能提供更紧凑的结果组。数据库支持准确匹配。

[0075] 重要的是映射函数是表现良好，即当映射函数应用于特定关键字时，它总是返回同一结果。

[0076] 然而，映射不必要是唯一的。实际上在映射给出高度压缩时，更加不象映射是唯一的。

[0077] 最好至少使用两个迥然不同的映射算法。不同映射的使用能限于在结果组中的使用。

[0078] 如 CRC32 和 Adler32 那样已知和已发表的映射算法能用于实现映射功能。

[0079] 最好映射函数安排成从正整数（包括 0）范围返回值，其大小至少如保存在索引中不同关键字的总数那样大。因此，每个关键字具有潜在可能被算法映射到另外的值，而没有关键字太长。

[0080] 例如，映射能使用下述函数从长度为 N 字节的标量计算：

[0081] $[(K^0\times 31\hat{}(N-1))+(K^1\times 31\hat{}(N-2))\cdots+K^{N-1}]$modulo R

[0082] 其中 K^N 是关键字 K 的第 n 个字节值 (0…255)

[0083] R 是大于保持在索引中不同关键字的总数的大小的 2 的最小幂。

[0084] 假设 R 值是

[0085] 2^{16} 对包含小于 65000 个关键字的索引。

[0086] 2^{32} 对包含小于 4×10^{9} 个关键字的索引。

[0087] 2^{64} 对包含小于 1.8×10^{19} 个关键字的索引。

[0088] 最好存储在结果组的关键字信息能是如存储在以前技术系统中传统关键字，或更好是该关键字的编码结果，其中应用的映射函数不同于在判定图中用于编码关键字的函数。

[0089] 限定词 (Qualifier)

[0090] 索引查询能导致许多关键字选中。从查询返回的每个结果能导致从海量数据存储设备检索相关的数据。因为这通常是硬盘，因此对每个从查询返回的结果可能需要盘 I/O 操作。

[0091] 不是所有取出的数据是需要的。

[0092] 例如考虑多国集团的情况，它希望查询其人事数据库以确定在伦敦且在特定薪水范围的雇员数。对给定的查询，仅根据位置或薪水的索引可以返回许多雇员。只有产生 AND 逻辑与询问减少了查询的大小。

[0093] 已知对此类查询通过在索引中存储多个关键字来修改数据库索引。因此该关键字能称为组合的或合成关键字。与候选匹配一个关键字机会相比候选（即在数据库中的一个实体）匹配两个关键字的机会是缩减很多，因此由该查询返回的选中数减少且需要更少的盘 I/O 操作。

[0094] 已知通过将关键字串联一起作出这些合成关键字。简单推出的第二搜索准则在索引建立时加到第一准则。但是单个关键字可以很冗长，且产生串联的关键字的索引能导致搜索变得实在很长。

[0095] 专利申请人已认识到，关键字数据不需要以其原始形式存储，而能实际上被编码，或用大为减少长度的字或位序列表示。在索引中的第一关键字和在串联关键字中的每个后续关键字能以那样修改的方式表示。

[0096] 按本发明的第八方面，提供组织数据库的一个方法，其中索引关键字是至少包含一个关键字的合成关键字，且其中一个或多个关键字以压缩形式表示。

[0097] 多个关键字较好地以压缩方式提供。

[0098] 关键字较好地通过将它们从未压缩形式映射到压缩形式而被压缩。映射不需是唯一的，但必须表现良好。通过使用散列算法完成适当的映射。

[0099] 由于不同关键字能由同样的减缩的表示来表示的那个事实，存在一个危险，查询将返回两种结果，匹配查询的结果和不匹配查询的某些结果。但是，假设使用好的随机分布的散列编码，在一个关键字失配的概率是 1/255，但对 N 个限定词，失配的概率是 $(1/255)^{N}$。

[0100] 因此错误率通常是小的。

[0101] 最好所有返回的选中被审查，以致确认数据并去除任何错误返回的结果。

[0102] 在完成插入，删除或检索时，新的关键字必须使用建立索引时采用的同样关键字转换过程进行压缩。

[0103] 较好的是在串联关键字的第一单元之后的每个附加单元能表示成“限定词”。每个限定词最好只有一个字节长。

[0104] 因此，将本发明与以前技术比较，以前技术的串联关键字包括八个关键字单元，每个为八个字节，共占据 64 字节。在本发明中，主要关键字以未压缩格式表示，因而占据 8 字节，而 7 个附加关键字分别用一个字节的限定词表示。因而总的关键字长度现在只有 15 字节。

[0105] 限定词关键词是可选的，在索引查询期间可以不提供，或提供任意一个或所有的关键词，因为它们不影响索引结构的引导。当扫描结果组时完成针对限定词的选中对象的测试。

[0106] 因而可以看到，本发明的这方面能与如 B- 树那样传统数据库结构一起使用，或与本专利中其它地方揭示的结构一起使用。

[0107] 另外，限定词能结合索引使用，其中有关节点的信息从节点的位置导出，而不是直接由节点本身定义或包含在节点本身之中（如组成本发明的第一方面的一个实施例的索引）。

[0108] 按本发明的第九方面提供一个数据库，其中索引包含至少一个关键字，且其中一个或多个关键字以压缩形式表示。

[0109] 有界关键字

[0110] 如上提到，数据库索引的有效性高度依赖那里使用的关键字特征，在关键字值连续增加或减少的地方导致潜在的低效的索引系统。那样无界关键字的一个例子是日期 / 时间关键字，它随当前时间不断增加。当这些关键字被加到索引中，它们引起要建立新的索引区，且当老的关键字从索引中删除时，它们形成残片，空出的索引区不能重新使用于新的关键字值。若索引结构能以那样方式组织，使得无界关键字的使用不会使得索引成为不平衡，这是很好的。

[0111] 按本发明的第十方面提供在表示范围的区间的第一关键字和第二关键字之间的关键字范围内引导数据库的方法，且其中关键字被映射到缩减的范围，且若在映射期间关键字的次序不保持，搜索是对大于关键字的较大者小于关键字的较小者的映射空间作出。

[0112] 最好关键字由散列或取模函数操作，使得关键字在整个数据库上基本上均匀分布。因此，例如一个无界日期 / 时间 (K)（例如它可以是从 1970 年元月 1 日起的毫秒数）被转换成 (K modulo N)，其中 N 是对索引选择的模数。因此转换器的关键字值永远落入 0 到 (N-1) 的范围。

[0113] 较好的是应用于关键字的模数值大于或等于在索引中将保持的不同关键字值的最大数目。但是必须指出，此条件不是必须遵循的。

[0114] 最好结果组（树形索引端点的节点）以未映射的方式包含关键字。这就减少了错误的概率，因为搜索关键字随后能与结果组中的关键字以原始方式进行比较，使得能避免映射错误。

[0115] 本发明能应用于传统的 B- 树或其它索引。

[0116] 按本发明第十一方面，有包括编码在数据库中使用的无界关键字的方法的数据库，其特点是每个关键字由运算子处理，它将关键字映射到有界范围。

[0117] 瞬态关键字

[0118] 有时需要操纵运用数据，其有效性及有用性限于某个时间范围，或其它的值范围。在以前技术的数据库中，那样“瞬态”数据很像正常数据那样运用，其结果是通常需要直接

删除过时的瞬态关键字数据或去除过期关键字数据驻留的索引部分。那样的操作需大量处理器运算并招致在数据库内的大的性能损失或能强加设计约束。

[0119] 按本发明的第十二方面提供管理数据库内管理关键字的方法，其特点是关键字包括一数据字段，指出该关键字和与之有关的数据在数据库内保持的期限。

[0120] 因此，某些或整个结果组能修改，以包括指出关键字和与之相关的特性认为有效的期限的数据。

[0121] 较好的是过期的关键字不是实际从结论组除去，而是一旦它们过期就可用于被新的数据改写。因此没有必要直接删除瞬态关键字，因此在删除过程中不牵涉性能损失。而是一旦标记指出数据不再有效，所占空间能被重新使用。

[0122] 虽然可能对数据提供明确的日期和时间标记，在此时间信息不再在结果组中被检测，这样的编码方式占据不可接受的大量空间。在本发明的较佳实施例中，关键字关系到给出当前时段的时效的时效单位，和指出数据和关键字失效并可用于改写的实现限止。

[0123] 较好的是时效单位是变量，其长度表示在单个时效单位的秒数。时效限止表示时效单位数，在此之后关键字从索引中去除。

[0124] 最好每个结果组包含时效起点，它是在该组中最新存入／更新输入项的日期／时间标记。在访问结果组时，计算每个输入项的时效，并最好计算结果存入输入项时效字段。在输入项时效和时效限止之间作出比较，以确定该输入项何时能被改写。

[0125] 确定标量数据的重复

[0126] 上面提到，以前技术（B-树）的判定图被引导到保持关键字块和指针信息的叶子块。引导判定索引可能需要一个或多个盘输入／输出操作以找到所选的关键字。若使用该索引记录频繁重复组合的唯一关键字组合的存在，则对高性能的系统，索引的盘结构可以是不适合的。

[0127] 按本发明第十三方面，提供组织数据库的方法，使得它具有判定索引和关键字索引，其中关键字索引中的关键字以压缩方法存储，并且对关键字索引作出校验，查看关键字是否在使用该关键字查询判定索引之前存在。

[0128] 因此有可能提供有效的基于存储器的方法，用于确定带索引的关键字的存在，使得能避免对关键字直接搜索索引本身的需要。

[0129] 因此本发明通过在半导体存储区中保持编码的关键字有效地遵守了标准索引结构。在搜索索引之前对存储器校验关键字的存在。若在存储器找到匹配的输入项，关键字作为重复被拒绝，否则试图将输入项插入索引。

[0130] 为提供有效和紧凑的结构，关键字索引使用如散列函数那样单向映射函数，散列函数或其它映射函数的特征如下：

[0131] • 当映射函数不止一次地对同一关键字调用，若在该关键字中的信息未被修改，则映射函数必须一致地返回相同值；

[0132] • 若两个关键字认为相等，则对两个关键字的每一个调用映射函数必须产生相同结果；

[0133] • 两个不相等的关键字映射到不相似的值这不是必须的，但对不相等的关键字提供不同结果将改善索引的有效性。

[0134] 可使用 CRC 32 和 Adler32 那样发表的散列码算法，因为它们提供合适的功能。

[0135] 在关键字索引的一个实施中，存储器可安排成 4 字节单元的同构陈列。此陈列众知为等同陈列。较好的是在等同陈列中的单元数是保持在索引结构中唯一关键字单元的 1.1 倍或更多数目。但是，等同陈列至少象索引一样大这是足够了。两个散列函数，A 和 B，结合该陈列一起使用。函数可如下选择：

[0136] 对任何两个不等的关键字 K 和 L，其中 A(K) = A(L)，不象有 B(K) = B(L)。

[0137] 对任何两个不等的关键字 K 和 L，其中 B(K) = B(L)，不象有 A(K) = A(L)。

[0138] 使用下述等式，采用散列函数 A 对关键字 K 计算等同陈列中的偏置单元 (0…N)：

[0139] 单元数 = ABS(A(K))moduloN

[0140] 其中 ABS(..) 返回数的无符号幅值而 N 是在等同陈列中的单元数。

[0141] 此函数是众知的单元函数 E(K)。

[0142] 在等同陈列中的每个单元存储 B(K)，其中单元偏置由 E(K) 给出。若在偏置 E(K) 的单元包含 E(K)，关键字 (K) 认为是重复的。若单元包含任何其它值，必须搜索索引结构以判定其重复性。

[0143] 对要将关键字插入索引的试图发生下面事件序列。首先计算 E(K)。然后计算 B(K)。若在等同陈列中 E(k) 处的单元包含 B(K)，则因为是重复的 K，插入被拒绝。然而若在等同陈列中 E(K) 处的单元不包含 B(k)，则该单元用 B(K) 改写，且作出索引的搜索以确定该关键字是否重复。

[0144] 等同陈列的使用大为减少了需要保持关键字表所需的存储区。若数据库具有一百万输入项，且每个关键字是 32 字节长，则需要在存储器中超过 32Mb 的存储器来缓存索引，因而等价的等同陈列将只占 4.4Mb 存储器。

[0145] 用层次结构组织索引

[0146] 这里提到，以前技术的索引通常使用树形图，其节点包含一项或多项 4 个关键字的数据。树形图（判定图）向下引导到叶子块（结果组），它保存关键字值如指向存储的数据结构的指针信息。引导索引会需要一个或多个盘输入 / 输出操作以找到所选的关键字。盘存取操作通常避表示索引的最高的性能损失，因为通常盘存取慢于存储器存取，且能发生数据瓶颈。为减轻此损失，索引常常被划分，每部分指定到不同的物理盘，从而允许盘的 I-O 在索引操作期间能重叠进行以增加索引的整体通量。此细分索引的方法提出索引划分的平直的一维图。

[0147] 按本发明的第十四方面，提供以层次结构划分的数据库索引，使得一个索引能将其某些工作负担委托给一个或多个其它索引。

[0148] 最好委托的索引能自己委托其某些工作负担给一个或多个其它索引。

[0149] 参与那样索引的层次结构的一个索引负责一个范围的子关键字。以其它索引的行为操纵关键字的子范围的索引称之为委托索引，它的关键字子范围称为关键字清单 (manifest)。

[0150] 较好的是索引的关键字清单通过将该关键字中的连续字节子集限止到可能值的连续范围而确定。任何对不符合索引关键字清单的关键字的插入，删除或更新或搜索的试图被索引所拒绝，否则，关键字操作由该索引或其它委托索引之一操纵。层次结构能那样安排，使得每个索引能具有零个，一个或多个委托索引，但每个索引能只是对另外一个索引的委托。

[0151] 在索引具有一个或多个委托索引的地方，插入、更新或删除一关键字的操作委托给带有合适关键字清单的委托索引。若没有委托索引具有合适的关键字清单，则操作必须由索引本身操纵。

[0152] 在索引具有一个或多个委托索引时，搜索关键字范围的操作委托给带有合适关键字清单的所有委托索引，且搜索由索引本身作出。从搜索委托索引来的查询结果结合从索引本身搜索来的结果。所有查询能同时完成。因此能修改索引结构，使得工作能在各子索引之中细分。每个子索引能指向有关的物理存储设备，从而允许发生多个并发的盘存取。这就减轻了盘的瓶颈并允许数据库作为整体更快地操作。

附图说明

[0153] 参考附图，通过例如将进一步描述本发明，附图是：

[0154] 图 1 是本发明的示意概图；

[0155] 图 2 是判定图逻辑结构的示意性表示；

[0156] 图 3 概略地示出了判定节点的结构；

[0157] 图 4 概略地示出了结果组的结构；

[0158] 图 5 概略地示出结果组中输入项的结构；

[0159] 图 6 概略地示出精确的搜索过程；

[0160] 图 7 概略地示出范围搜索过程；

[0161] 图 8 概略地示出插入一关键字的过程；

[0162] 图 9 概略地示出删除一关键字的过程；

[0163] 图 10 概略地示出精确的关键字查询的过程；

[0164] 图 11 概略地示出查询的范围的过程；和

[0165] 图 12 概略地示出穿过带判定组 G 的节点，判定值 V，到最小和最大关键字范围并返回结果的过程。

[0166] 图 13 示出判定图的一部分的结构。

[0167] 图 14 示出修改判定节点的逻辑结构。

[0168] 图 15 示出插入一关键字的过程。

[0169] 图 16 示出删除一关键字的过程。

[0170] 图 17 示出精确的查询匹配过程。

[0171] 图 18 示出穿过带模式表的树并返回结果的过程。

[0172] 图 19 示出搜索模式表 (L) 的过程。

[0173] 图 20 是模式匹配过程如何帮助分类数据的例子。

[0174] 图 21 是具有有限索引的判定图的概略表示；

[0175] 图 22 示出修改的结果组的结构；

[0176] 图 23 概略地示出分割判定图的第一方法；

[0177] 图 24 概略地示出分割判定图的第二方法；

[0178] 图 25 概略地示出具有限定词的合成关键字的配置；和

[0179] （图 26 概略地示出按本发明的一个方面的搜索关键字的格式。）

具体实施方式

[0180] 图 1 概略地示出加入构成本发明的一个实施例的索引和数据存储的数据库。索引 2 包括判定图 4 和多个结果组 6、8、10、12 和 14。每个结果组由经过判定图的唯一条路径达到。因而每个结果组指向数据存储 16 中的对应输入项。

[0181] 图 2 概略地示出通常记作 20 的判定图的结构。判定图在起点 22 开始。通过判定图的所有引导必须在起点开始。起点可具有零个（如当数据库是新的），一个或二个判定指针，指向在判定图中的另外节点或结果组。每个其它判定节点可包含 0，1 或 2 个判定指针，每个判定指针指向另外判定节点或结果组。在判定节点上的判定指针这里称为“低指针”和“高指针”。

[0182] 在判定图中任何判定节点的判定指针能只指向同一判定图中一个其它判定节点或单个结果组。任何判定图节点必须由只同一判定图中的一个其它判定节点指向或由起点指向。类似地，任何结果组必须由判定图中仅一个判定节点指向。因此任何结果组只能跟随从起点通过判定图的单个和唯一的路径达到。此唯一的路径称为引导路径。

[0183] 图 3 概略地以更详细方式示出通常记作 40 的判定节点的逻辑结构。判定节点包括低指针类型，低指针，高指针类型和高指针。低指针类型 41 指出低指针 42 的目的。低指针类型 41 能指出低指针 42 是否指向判定节点或结果组。低指针 42 给出要指向的判定节点或结果组的地址。插入零值表示不存在指针。类似地高指针类型 44 指出高指针 45 是否指向判定节点或结果组。高指针指出要指向的判定节点或结果组的地址。零值也能用于指出不存在指针。这就使能了用数据处理器不其存储器表示和存储的判定图的“树形”结构。

[0184] 图 4 概略地示出结果组的结构。结果组包括多个关键字和属性输入项，60，62，64 和 68，每个通过给出在结果组中下一输入项的地址的直接链路互相链接。图 5 概略地示出结果组中每个输入项的逻辑结果。输入项包括三个字段，即链接字段 80，关键字字段 82 和属性字段 84。链接字段指向下一个结果组输入项。零值表示不再有输入项。关键字字段 82 保存关键字的精确值，而属性字段 84 保存与该关键字有关的属性，如对应于关键字字段 82 的数据的地址。定义了索列的分量，接着讨论如何使用和引导索引。

[0185] 为沿索引引导，沿着引导路径访问的每个判定节点涉及关键字中的一个或一组位。此位组称为“判定组”，在判定组中的位数能设置以包括在关键字的单个位和所有位之间的任意位数。

[0186] 由判定组取的可能值的范围称为“判定范围”，且由判定范围最小值和判定范围最大值界定。判定范围最小值能是小于在判定组中所有位组的任意值，类似地判定范围最大值是大于判定范围的任何值。为方便起见，最小值和最大值能以无符号幅值标记系统表示。

[0187] 每个判定节点具有相关的判定值。判定值确定当经过判定节点引导时是否遵循低指针或高指针。当在关键字中审查的判定位组的值大于判定值时使用高指针，否则使用低指针。

[0188] 判定值的比较最好使用无符号幅值标记进行。判定值能设置成由系统管理员，设计员或操作员所选的任何值，但最好从下面之一选取：

[0189] 1、判定组所有可能值的数学中值。

[0190] 2、判定组所有预期值的数学中值。

[0191] 3、判定组所有预期值的数学平均值。

[0192] 4、在索引建立时指定的任意值。

[0193] 5、当前判定值（即最近使用的判定值）和前一次的判定值之间的中值。在访问的判定节点的判定组的选择最好能根据下列之一个或多个，但不是必需的：

[0194] i. 判定组对所有判定节点可以是相同的。

[0195] ii. 判定组能按每次对判定节的访问而改变。

[0196] iii. 当从以前的判定节点来的判定值达到或接近当前判定组的判定范围最大值或判定范围最小值时，能改变判定组。

[0197] Iv. 当在顺序节点之间的判定值改变小于一个单元或某预定阈值时，能改变判定组。

[0198] 在判定节点处判定组的大小能根据一个或多个参数确定。例如，判定组的大小能对所有判定节点是固定的。然而，当选择新的判定组时其大小能从以前的判定组增加，或相反，当选择新的判定组时，其大小能从以前判定组减少。

[0199] 相对于关键字在判定节点处判定组的位置能根据下面一个或多个原则。判定组的位置可以对所有判定节点是固定的。另外和 / 和另选地，在选择新判定组时，判定组的位置可以从以前的判定组有固定的偏置。

[0200] 若索引支持由范围匹配的关键字检索，最好施加更多约束。特别地，最高有效位必须包括在如原点那样的层次上最重要的节点。因此在整个关键字中，判定组的重要性以递减重要性的单调方式改变。

[0201] 最好考虑索引如何被引导的某些例子。图 6 示出 4 字节关键字如何能以精确匹配搜索的方式引导。在此例中，关键字以 ASCII 编码方式存储，且针对每个关键字示出的数以十六进制标记方式示出其值。在此例中，每个判定组具有 4 位的固定大小。在每个新的被访问节点上，判定组顺序向下移动到下面 4 个较低的有效位。在此例中，最高有效位被指定为 0 位，次高位是 1 位，再次高有效位是 2 位等。因此，若判定组具有值 0，1，2，3 或 4，穿过图形向左。但是若判定组具有 5 到 15 范围的值穿过图形向右。

[0202] 如图 6 所示，对数据库提出若干名字。这些名字或关键字值是 Fred，John，Bill，Zoe，Eric 和 Pete。每个关键字的十六进制等价值位于名字附近。将描述的引导过程适合于插入，删除或匹配。

[0203] 开始对第一判定节点 100 提出关键字，在那个节点测试位 0 到 3。因为每个十六进制字占据一个字节，很明显位 0 到 3 对应于关键字中第一字符。因此，因为 Fred，John，Bill 和 Eric 均以“4”开始，这些关键字向左沿着分支 102 传播到第二层节点 104。但是 Zoe 和 Pete 具有在 0 到 3 范围之外的初始 4 位，结果它们的关键字沿着高指针路径 106 传播到节点 108。

[0204] 节点 104 和 108 占据判定图的第二层，因而它们的判定是根据关键字中接下去的 4 位。这对应于考虑以十六进制表示的下一字符。因此节点 104Bill 沿低层指针 110 进到后续节点（未示出），因为在关键字中位 4 到 7 编码成值 2，而 Fred，John 和 Eric 沿高层指针 112 进到又一个节点（未示出），因为它们的关键字中的位 4 到 7 分别编码成十六进制的 6，A 和 5。类似地，在节点 108，关键字 Pete 沿低层指针 114 进行，因为位 4 到 7 编码成值 0，而 Zoe 沿高层指针进行，因为位 4 到 7 编码成值“A”。此过程能重复到已搜索到足够的关键字以达到结果组。

[0205] 图 7 概略地示出 4 位关键字如何能在范围匹配搜索中引导。如图 6，关键字以 ASCII 码存储，且针对每个关键字的 ASCII 值以十六进制标记示出其值。

[0206] 在此例中，判定组具有固定的 8 位长度。仅当判定范围用尽，即当判定值达到判定范围的两个边界时，判定组向下移到下 8 位次高有效位。在此例中，最高有效位指定为位 0，次高位为 1 位，等如图 6 示出例的情况。在每个节点处的判定范围是十六进制的 30 到 50，以便复盖以 ASCII 格式表示的所有数字和大写字符。在用于新判定组的每个节点的判定值是十六进制 48，然后改变成对那个同样的组在每个后续节点的同样判定组的以前判定值的平均值。在只知一个以前判定值的地方，使用判定知范围的最小值或最大值界限分别取决于是否我们正引导向左或右。

[0207] 使用与上例相同的关键字，该关键字提供给第一节点 118，它检查 0 到 7 的第一个 8 位，并将其与判定值 48（十六进制）比较。用示出的关键字，Fred，Bill 和 Eric 沿低指针进制到节点 122，而 John，Zoe 和 Pete 沿高指针 124 进到节点 126。

[0208] 在节点 122 必须改变判定值。对此新判定组使用的判定值改变到沿同样路径到达的以前的判定组的中值。既然在此情况只有一个以前的判定值是已知的，使用判定值范围的最小值和最大值界。假如针对低指针导向 4 个节点 122，则使用判定范围的低界限。

[0209] 因此，为计算新的判定值，必须直接找出在范围十六进制 30 到十六进制 48 中数的中值，此组数是 30，31，32，33，34，35，36，37，38，39，3A，3B，3C，3D，3E，3F，40，41，42，43，44，45，46，47 和 48。由此可见，中值是 3C。设此为判定值，Fred，Bill 和 Eric 均沿高指针 127 进行到下一节点（未示出），其判定值将是从十六进制 3C 到十六进制 48 范围的中值。

[0210] 类似地，沿高指针 124 从节点 118 达到的节点 126 的具有新的计算的判定值，在此情况是 4C（十六进制）。在此节点应用该关键字，John 将向左进行（低指针），因为它包含小于判定值 4C 的值 4A，而 Zoe 和 Pete 向右手（高指针）进行，因为它们分别包含大于判定值 4C 的值 5A 和 50。

[0211] 搜索的又一例子能在字符搜索的情况见到，因此使用下列准则完成搜索：

[0212] • 在 ASCII 字符关键字上的索引与占据 8 位并只在“A”到“Z”范围内的每个字符的精确匹配。

[0213] • 固定的 8 位判定组。

[0214] • 最高有效 8 位的判定图起点（第一字符）。

[0215] 对每个判定节点判定组按 8 位（一个字符）前进

[0216] 对“A”和“Z”的 ASCII 码分别是 65 和 90。因此对此组的合理的判定值是中值（即 78）。

[0217] 在此例中，判定组每次按一个字符 (8 位）前进，使达到新的判定节点。将在该关键字中的判定组中的位的值与固定的判定值 (78) 作出比较以确定跟随哪个指针。

[0218] 更完善的方法包括每次移到新的判定节点改变判定值。因此，对判定组的判定范围受判定范围最小值与判定范围最大值的限止，它们分别是 65 到 90。当移到下一判定节点，操作能通过使用当前的和以前的判定值的中值选择新的判定值。当这是新判定组而无以前判定值可用时，我们能用判定范围值的最小值或最大值（分别根据是否跟随低的或高的指针）。当判定值的改变小于 1 或某些其它预定阈值时，选择新判定组。下面给定判定值如何改变的例子。

[0219] 在该关键字中的判定组具有值 82。对新组的起始判定值是 78。判定范围是 65 到 90。当判定值的改变小于 1 时设置判定组使选择新的组。

[0220]

判定值	判定值的改变	操作
78		跟随高指针 (82 ＞ 78) 新判定值＝ (78+90)/2
84	6	跟随低指针 (82 ＜ 74) 新判定值＝ (84+78)/2
81	3	跟随高指针 (82 ＞ 81) 新判定值＝ (81+84)/2
82.5	1.5	跟随低指针 (82 ＜ 83) 新判定值＝ (82.5+81)/2
81.75	0.75	改变判定组

[0221] 应该注意，不需通过范围匹配支持关键字检索的索引不是象它们所做那样的厉害的约束。尤其是不需要支持范围匹配的索引能在其原点具有任何判定组，并当判定值接近（可由系统设计者定义或可由用户或某些管理功能选择）判定范围界限，即判定范围的最大和最小值时，或当判定值改变小于预定大小时，能任意改变判定组位置，且也能任意改变判定组。

[0222] 当为了插入，删除或检索候选关键字的目的穿过引导路径时，候选关键字的判定组在每个访问的判定节点上与判定节点的判定值比较，且沿着低指针或高指针到达下一判定节点或结果组。一旦到达结果组，沿引导路径的行程结束。当判定组值指出使用不存在的判定指针时，沿引导路径的行程也结束了。

[0223] 结果组包括一系列关键字和有关的属性。包含在结果组的关键字数目由下列限止的一个或多个的组合确定的上界所限止：

[0224] 1、固定的限止；

[0225] 2、在建立索引时指针的限止；

[0226] 3、能在整个索引的有效时间指定的限止；和

[0227] 4、作为索引增加的函数改变的限止。

[0228] 图 8 示出插入带属性 (Q) 的关键字 (K) 的过程。该过程在步骤 150 开始，其中索引带着由设计者设置的默认或初始判定组 DG，在起点处输入。实现一个算法来计算起始判定值 DV_0。然后控制进行步骤 152，测试在关键字中的判定组是否大于该判定值。若测试是满足的，控制进到步骤 154，审查高指针 HP 并建立新判定值 DV1。

[0229] 控制从步骤 154 进到步骤 156，完成校验，查看是否存在指针（是否为高指针或低指针）。类似地，若步骤 152 指出，在关键字上的判断组不大于判定值，则控制进到步骤 158，检查低指针并计算新的判断值 DV1。控制从步骤 158 进到步骤 156。若指针存在控制从步骤 156 进到步骤 160。步骤 160 测试，指针是否指向又一个判定节点。若指针指向判定节点，则控制进到步骤 162，它将测试推进到下一判定节点。控制从步骤 162 进到步骤 164，它完成测试，查看判定组是否应改变。若对此测试的回答是肯定，则控制进到步骤 166，它按照上面设定的准则计算新判定组的值。控制随后从步骤 164 或 166 返回到步骤 152。

[0230] 如步骤 156 确定指针不存在，控制进到步骤 170，导致建立新的结果组，并随后进到步骤 172，更新指针使指向新的结果组。控制随后进到步骤 174，将关键字及其属性加到结果组。

[0231] 而且，若步骤 160 不指向判定节点，则它必须指向结果组，且控制进到步骤 176，引起作出到对应结果组的跳转，且从那里的步骤 174，使得其关键字及其属性被加到结果组。

[0232] 在步骤 175 作出测试，查看结果组是否超过其大小限止。若结果组尚未增长得太大，控制进到步骤 178，它表示从关键字插入过程的出口，然而若结果组达到其大小极限，则控制进到步骤 180，建立没有指针的新判定节点。控制从步骤 180 进到步骤 182，它更新以前判定节点的指针使指向新的结果组而非老的结果组。然后控制进到步骤 184，它检查老的结果组 CS(S)，查看它是否空。若它不空，控制进到步骤 186，它进到结果组 CS(S) 中的第一关键字及属性的输入项并重新引导该索引以便找到此输入项应放置的新结果组。一旦输入项被重新定位到结果组，在步骤 188 它从其当前结果组 CS(S) 被去除。然后控制返回到步骤 184，过程重复直到结果组 CS(C) 变空。控制随后进到步骤 190，它表示关键字插入过程的出口点。

[0233] 因此通常，将关键字及其有关属性插入索引的过程需要沿着用于插入关键字的判定路径引导。若引导路径未到达结果组而终止，则新的结果组必须建立。判定指针必须更新，使得它指向新的结果组。新的判定指针必须放置在判定图上判定图的行程终止处的点上。然而若引导路径在结果组终止，则关键字及其属性插入此结果组。结果组的每次修改改变了该组的大小。因此若新的关键字插入结果组且引起或将引起该组超过其上限，则必须建立新的判定组。新的判定组和 / 或判定值必须与新的判定节点和判点指针相关，该以前指向满的结果组的指针必须修正以指向新的判定节点。然后从老的（现在满的）结果组的每个项必须在沿着索引的引导路径之后重新插入新的一个或多个结果组。

[0234] 数据库中的数据应随时更改，此更改包括删除关键字和数据，这是由于数据不再与数据库有关或由于为了归档的目的需要去除，这样使得“当前”版本的数据库不会变成由无关或过时数据所乱堆而成，图 9 概略地示出删除关键字的过程。某些步骤等同于关于图 8 描述的那些，这些步骤给以同样的参照号。因此步骤 150 到 166 等同于已描述的那些，不同处在于若在步骤 156 作的测试指出指针不存在，则控制进到步骤 200，它表示删除过程的出口。

[0235] 若步骤 160 的测试确定，指针未指向判定节点（如插入关键字的过程的情况），步骤 160 将控制转到步骤 176。控制从步骤 176 转到步骤 202，那里完成测试，查看结果组 CSCS) 是否为空。若结果组是空，控制进到步骤 204，它表示从过程的出口，否则控制进到步骤 206，它进行对结果组中输入项的扫描。扫描从结果组中第一个关键字 (L) 开始，然后控制进到步骤 208，那里作出测试，查看关键字 (L) 是否等于删除关键字。若是，控制进到步骤 210，从结果组删除关键字 (L) 及其属性。控制从步骤 210 进到步骤 214，在那里作出测试，查看关键字 (L) 不同于删除的最后关键字，控制也从步骤 208 进到步骤 214。然后控制进到步骤 216，在那里选择下一关键字，控制然后转到 208。否则控制从步骤 214 转到步骤 218，它表示此过程的出口。

[0236] 因此总而言之，为了从索引删除关键字及其相关的属性，需要沿着对要删除的关键字的引导路径进行引导。若引导路径未达到结果组而终止，则认为该关键字不存在于索

引中。然而，若引导路径终止于结果组，则对确实等于要删除的关键字的所有关键字，经过结果组开始扫描。然后从结果组去除这些关键字及其相关的属性。

[0237] 图 10 示出准确查询关键字并返回结果的过程。过程具有与删除关键字的过程很大相似性，且步骤 150 到 206 是如参照图 9 那样描述。然后在步骤 150 和 152 之间插入附加的步骤 151，其中结果组 (R) 被复位到空的状态。在步骤 202 引导索引到结果组之后，控制进到步骤 206，起动经过结果组的顺序扫描。控制从步骤 206 进到步骤 230，在那里作出测试，查看当前检查的关键字 (L) 是否等于搜索关键字。若是，控制进到步骤 232，否则控制进到步骤 234。在步骤 232，关键字及其属性加到搜索的查询结果表，控制随后进到步骤 234。步骤 234 测试，查看当前检查的关键字是否为结果组中最后的。若否，检查下一关键字，控制转到步骤 230。若关键字是最后关键字，控制从 234 进到步骤 236，它表示此例行程序的出口。

[0238] 图 11 概略地示出用于完成带着最小关键字值 (I) 和最大关键字值 (A) 的范围查询并返回结果的过程。过程在步骤 300 开始，在那里结果组 (R) 被复位成无效或空状态。然后控制进到步骤 302，在那里计算判定组和判定值，并控制从那里进到步骤 304，在那里穿过索引以寻找匹配范围查询的那些结果。在图 12 中更详细地示出穿过索引的过程。过程在步骤 350 处开始，在那里在其开始节点进入过程。然后控制进到步骤 352，在那里作出测试查看最小关键字是否小于或等于判定值（或忽略）且最大关键字是否大于其相关的判定值（或忽略）。若测试的结果是两个条件均满足，控制进到步骤 370，否则控制进到步骤 354。步骤 370 是校验，查看是否已设置低指针，若有控制进到步骤 372，在那里作出校验，查看低指针是否指向结果组。若是，控制进到步骤 400。若否，控制进到步骤 374，在那里重新计算判定组和判定值，且用于从达到的节点路径指针穿过索引。过程具有调用自己的能力，因而是递归的。控制随后进到步骤 376，在那里作出校验，查看是否在高指针，若否，控制进到步骤 378，它表示出口，若是，控制进到步骤 390，在那里作出测试，查看高指针是否指向结果组，若是，控制进到步骤 400，然而若高指针不指向结果组，控制进到步骤 392，在那里计算新的判定组和判定值，并取新节点为起始节点完成索引的又一个穿程。

[0239] 返回到接近过程的起点，在步骤 352 将控制进到步骤 354 的情况，步骤 354 完成测试，查看最小关键字值和最大关键字值是否均小于或等于判定值。若是，控制进到步骤 356，否则控制进到步骤 358。步骤 356 完成测试，查看是否存在低指针，若不存在，控制转到 357，表示过程的出口，否则控制转到步骤 360。步骤 360 进行测试，查看低指针是否指向结果组，若是，控制转到控制 400，否则控制进到步骤 402，它重新计算判定组和判定值并以递归方式调用穿程过程。返回到步骤 358，完成测试以查看是否存在高指针，若是，控制转到步骤 362，否则控制进到步骤 364，表示过程的出口。步骤 362 完成测试以查看高指针是否指向结果组，若是，控制进到步骤 400，否则控制进到步骤 364，它计算新的判定组和判定值并引起从这些新值开始要完成的索引的又一个穿程。

[0240] 步骤 400 完成测试，查看结果组 (S) 是否空，若是，控制进到步骤 404，表示过程的出口。若结果组未空，则在步骤 406 开始顺序扫描结果组中那些输入项，在那里选择结果组中的第一关键字。然后控制进到步骤 408，在那里完成测试，查看关键字 (L) 是否在由最小和最大关键字值定义的范围内。若测试结果为“yes”，则在步骤 410，关键字及其属性被加到结果组 (S)，而若步骤 408 的测试是否定的，控制进到步骤 412，其中作出测试，查看当前

考虑的关键字是否为结果组中的最后关键字，若是，控制进到步骤414，表示过程的出口。步骤410也将控制转到步骤412。若步骤412的测试确定，所考虑的关键字不是最后关键字，则在步骤416中选择结果组中下一关键字，控制返回到步骤408。

[0241] 因此总而言之，借助使用最优的最小搜索关键值和最优最大搜索关键值的范围匹配检索关键字及其属性，需要从原点上开始的判定图的递归穿程，那穿程连续地将在最小和最大搜索关键字中的判定组位与在每个判定节点的判定值比较。若在判定组中最小搜索关键字位的无符号幅值小于或等于判定值，且判定组中最大搜索关键字字位的无符号幅值大于判定值，则首先采取动作使用最小搜索关键字并忽略最大搜索关键字穿过从低指针达到的判定图；其次采取动作使用最大搜索关键字并忽略最小搜索关键字穿过从高指针达到的判定图。

[0242] 若判定组中的最小搜索关键字和最大搜索关键字位的无符号幅值均小于或等于判定值，则使用最小搜索关键字和最大搜索关键字完成从低指针达到的判定图的穿程。若判定组中的最小搜索关键字和最大搜索关键字字位的无符号幅值均大于判定值，则使用最小搜索关键字和最大搜索关键字完成从高指针达到的判定图穿程。

[0243] 很明显，需要判定图的多个及重复的穿程。若从穿程引出的所有引导路径终止而未达到结果组，则认为在索引中不存在匹配的关键字。对于在结果组终止的那些引导路径完成通过所有在结果中的关键字的扫描，且在搜索范围中匹配所需属性的所有关键字作出搜索结果返回。

[0244] 作为在数据处理器和存储系统中数据库的实现的步骤虽然并非必需，最好结果组的大小是存储设备块传输大小的整数倍，这提供物理存储器的最佳使用。虽然并非必需，还最好将判定图分成块，其大小是存储设备块传输大小的整数倍。这具有下述好处，当判定块满时，能建立新的块，且新的判定指针能恰当递指向新的块。

[0245] 为了达到数据库的快速操作，最好在使用中判定组全部或部分地保持在数据处理系统的半导体存储器中，从而避免与盘存取有关的时间开销。作为性能进一步增强，较好的是结果组存入半导体存储器，或者至少一旦结果组被识别，它被交换到半导体存储器，使得结果组的顺序扫描很快地完成。

[0246] 对所有关键字类型的范围匹配的最佳配置是使用带有固定的零判定值的一位的判定位组。在开始的判定组是最高有效位，对每个访问的判定节点判定组前进一位（朝最低有效位）。对数字或日期/时间精确关键字匹配，最优位配置看来是使用带有零固定判定值的一位的判定位组。在原点的判定组是最低有效位，且对每个访问的判定节点上判定组前进一位（朝最高有效位）。对基于ASCII的关键字的搜索，对精确匹配的最佳位配置是使用8位的判定组。最佳判定值将取决于预期的字符的范围。

[0247] 模式匹配

[0248] 参考前面的图描述的数据库结果涉及那样的结构，其中判定组绝不包含任何关键字数据。然而可能修改数据库结构，使得一个或多个节点包含部分关键字序列。这方便于借助模式匹配索引和检索标量数据。树形结构与以前技术的数据库比较仍是小的，因为每个判定节点只需要包含相当适度量的数据，而不是象以前技术的系统那样的整个关键字。

[0249] 数据库结构只需要在上述基础上作少量修改。图13示出判定图的结构，它在许多方面保持与图2所示相同的布局。然而，现在起点和每个判定节点包含位序列。位序列

表示候选关键字与其比较的子关键字的部分。图 14 示出每个判定节点的逻辑结构。此结构使人回忆起参考图 3 描述的结构。通常标明为 440 的节点具有用于判定位序列的存储区 450。判定位序列能是任意长。然而通常要注意的重点，判定位序列应大大短于向判定树提供的关键字的长度。节点包括排斥 (exclusion) 指针类型 452 和包含 (inclusion) 指针类型 456，它们指出其有关的排斥指针或包含指针是否指向又一个判定节点或结果组。因此，参考图 3 的描述这些指针分别实现与低指针类型及高指针类型 41 和 44 相同的功能。然而，术语的改变仅表明下述事实，若候选关键字包含判定位序列，跟随一个指针，而若候选关键字不包含判定位序列，跟随另一指针。

[0250] 图 15 示出插入关键字到树形结构的过程。此过程很类似于参考图 8 描述的过程，为简单起见对类似部分使用类似的参照号。然而，步骤 152 被修改成 152’，因为测试改变成查看在判定节点处的判定位序列是否包含在关键中，若是，控制进到步骤 154’，跟随包含指针。控制随后转到步骤 156。若步骤 152 发现，判定位序列不在候选关键字中，则控制进到步骤 159’，跟随排斥指针。对过程的进一步修改是步骤 162 能直接指向步骤 152’ 的输入，从而忽略步骤 164 和 166。然而在其它方面，关键字插入过程是如以前描述的。

[0251] 如图 16 所示删除关键字的过程也十分类似于以前所述，且对类似的部分再次使用类似的参照号。如对插入关键字的过程情况，如参考图 13 所示步骤 152，154 和 158 修改成步骤 152’，154’ 和 158’，使得步骤 152 测试以查看候选关键字是否包括判定位序列，若是，跟随包含指针，若否，跟随排斥指针。此外，步骤 162 现在直接指向步骤 152’的输入，从而忽略了步骤 164 和 166。

[0252] 当然，为了准确查询匹配关键字能引导索引。图 17 示出返回关键字的准确查询的引导过程。过程非常类似于参照图 10 所示，如图一样，为避免重复对类似的步骤使用类似的参考号，且不同处在于步骤 152，154，和 158 被修改，使得 152’ 现将候选关键字与判定位序列比较，若判定位序列在候选关键字中，控制进到步骤 154’，跟随包含指针，否则控制进到 158’，跟随排斥指针。此外，步骤 162 现在直接指向步骤 152’ 的输入。

[0253] 图 18 示出从带有模式表 (L) 的节点穿过并返回结果 (R) 的过程。过程在步骤 500 处开始，在那里进入其实节点。控制随后进到步骤 502，作出测试，查看判定位序列是否包含在模式表 (L) 中。若测试结果是否定，则控制进到步骤 504，在那里作出测试，查看从判定节点是否存在排斥指针。若存在排斥指针，则控制转到步骤 506，作出测试，查看该指针是否指向另外判定节点，若是，控制转向步骤 508，否则控制转向步骤 510。步骤 508 引起选择下一判定节点且在步骤 512 以递归方式调用穿过过程。控制从步骤 512 进到步骤 514。此外，若在步骤 502 的测试得出结论，判定位序列包含在模式表示中，则控制进到步骤 514，类似地若步骤 504 指出，排斥指针不存在，控制也进到步骤 514。

[0254] 回到步骤 510，引起作出到由在步骤 506 检查的指针指向的结果组的跳转，且随后在步骤 516 通过结果组作出搜索。搜索过程在后面参考图 19 描述。控制从步骤 516 进到步骤 514。步骤 514 完成测试，查看是否存在包含指针，若是，控制转到步骤 518，否则控制转到步骤 520，表示过程的出口。步骤 518 测试以查看指着那是否指向又一个判定节点，若是，控制转到步骤 522，再次以递归方式调用过程，否则控制转到步骤 524，引起到由该指针指向的结果组的跳转。从步骤 524 完成在步骤 526 的结果组的搜索，步骤 526 与 516 相同。步骤 526 和 522 均指向表示过程结束的出口步骤 528。

[0255] 图 19 概略地示出步骤 516 和 526 的搜索过程。过程在步骤 530 进入，控制从那里进到步骤 532。步骤 532 完成测试，查看结果组是否空，若是，过程在步骤 534 推出，否则控制转到步骤 536。步骤 536 选择结果组中第一关键字并随后控制进到步骤 538，测试以查看关键字是否包含模式表中每个模式。若是，控制转到步骤 540，否则控制进到步骤 542。步骤 540 将从结果组来的关键字及其属性加到结果表中。然后控制转到步骤 542。步骤 542 完成校验，查看当前关键字是否为结果组中最后关键字，若是，过程在步骤 544 退出，否则控制进到步骤 546。步骤 546 选择结果组的下一个关键字，并返回控制到步骤 538。

[0256] 图 20 概略地示出，使用组成本发明的实施例的数据库如何能完成范围搜索。假设节点 560 具有存储那里的位模式 RE，节点 562 具有位模式 OH。当用字 red，Fred，Bill 和 John 查询节点时，字 red 和 Fred 沿包含路径导向下一个节点 564，而字 Bill 和 John 沿排斥路径导向节点 562。节点 561 指导字 Bill 沿着排斥路径，而字 John 沿着包含路径，因为包括位模式“OH”。

[0257] 若数据库现使用通配符搜索查询，例如，使用搜索字符串“*red*”，这里 * 表示多字符通配符，则节点 560 能确定，导向 562 的排斥路径不需要搜索，因而搜索能限于包括节点 564 的路径。然而，若使用搜索项“*lap*”查询索引，则节点 560 不能确定位模式 RE 是否在搜索字符串（它能包括在通配符部分）中，并因此索引必须沿着包含路径及排斥路径搜索。

[0258] 注意到下述是有益的，在关键字中的位模式的位置在数据库中从一个节点引导到一个节点时不必须沿着关键字顺序移动。

[0259] 关键字压缩

[0260] 在传统的数据库系统中在每个判定节点处存储关键字整体。而在上述发明中在每个判定节点存储关键字的一部分，虽然该部分的长度没有约束。专利申请人认识到通过不以原始方式存储关键字整体关键字的一部分，而是从映射过程的结果导出的关键字的表示能获得进一步的节省空间。如 CRC32 和 Adler32 那样的散列码算法是众知的，并提供用于映射编码关键字值的函数。这些算法对长的位序列操作以导出短得多的位序列。映射算法在下面意义下是确定性的，即若输入位序列永远相同，导出的位序列也永远相同。但是由于所获得的高度压缩映射不能是唯一的，因而若干输入序列将映射到短得多的输出序列。然而通过接收此结果，索引的设计能大为减少。因此，当对候选的关键字 K 引导树形图时，遵循传统的穿程方法，区别在于使用 ACK) 替代 K，其中 A 是由于树形图的映射函数。

[0261] 因为二个或更多关键字能映射相同函数，结果组能包含不使用映射函数不能得到的信息。因此必须检查结果组中那些输入项以便排除不对应于未映射的关键字的那些。这也能通过保持未映射的关键字在结果组中，并将那些关键字与原始未映射的搜索关键字比较以排除不匹配的那些项来实现。然而，通过用不同于 A 的第二编码算法 B 编码结果组中的关键字也能达到空间节省。在两种映射算法下，映射到两个同样的映射值的不正确的搜索关键字的或然率是确实很小，因而极不象会获得虚假的映像。确实，虚假匹配的概率在 R^2 中接近 1，其中 R 是函数映射的范围大小。映射关键字 A 和 B 均存入结果组，因为若结果组达到其最大尺寸，这方便于结果组的重新映射。

[0262] 为插入一关键字到索引中，关键字使用第一散列函数码。然后使用编码的关键字经上述方法或传统的方法引导索引，直到达到结果组。然后使用第二散列函数，关键字能以

未编码或编码形式插入结果组。

[0263] 删除是类似的，关键字以其原始形式或使用编码值形式从结果组删除，编码值形式是作为使用第二散列函数在该关键字上操作的结果。

[0264] 因此有可能以下述形式提供编码数据库关键字的有效方法，其中尺寸缩小很大，且使用不相似的函数的双重编码将虚假匹配的危险降至可接受的小概率事件。

[0265] 有界关键字

[0266] 图 21 概略地示出组成本发明的实施例的索引。概念上，该索引类似传统的 B 树索引。然而，不是将整个原始关键字包括在判定树的每个节点中，而是使用编码的关键字，其中使用模数算术函数映射关键字。还可得出，在只有部分关键字无界的地方，按本发明只有部分关键字需要映射。因此，当为关键字提供用于搜索，删除或插入的索引时，在关键字使用前对其应用标准的模数函数。为了将模数应用于字符串的关键字类型，需要将字符串处理成多字节的很大的无符号整数，第一字符是该整数值中最高有效字节，然后应用模数。

[0267] 因而，为将关键字插入索引，使用为索引选择的模数编码该关键字。然后使用编码的关键字值经标准的索引方法或这里提出的方法，如上所述地引导索引结构到结果组（叶子块）。然后关键字以其原始方式或其编码方式插入结果组。删除和搜索以类似方式完成。因此，在搜索中一旦达到结果组，关键字以其原始方式或编码方式与结果组中所有关键字值比较。

[0268] 图 22 概略地示出操作员希望运作范围匹配而非如上所述的精确匹配的概率。对范围的最小和最大搜索关键字分别表示为 L（较低）和 U（较高）。如图 22 所示，这些关键字存在于无界或未映射的搜索空间 580。然后关键字通过模数过程 582 映射，以便将它们转换成有界范围的索引。在映射过程之后不能保证保持关键字的次序。这就引起两种可能性。首先，如概略地指定的映射组 584 所示，映射的低关键字具有小于映射的高关键字的值。若是这样情况，使用标准的索引引导方法引导索引到结果组。然而也如预期的和在映射组 586 概略地示出的那样，映射的低关键字 ML 能具有大于映射的高关键字 MU 的值。在此情况，搜索必须对所有具有大于或等于编码的最小关键字 ML 的编码值的结果组作出。此搜索使用标准的索引引导方法完成。此外，必须对具有小于或等于编码的最大关键字 MU 的编码值的所有结果组作出搜索。再次使用标准引导方法完成。

[0269] 应该注意，此技术只在关键字之间的差小于模数编码方案的域时有效。因此，作为普通的例子，若关键字是使用模数 10 映射，则在搜索范围内较低和较高关键字之间的差必须小于 10。

[0270] 瞬态关键字

[0271] 图 23 概略地示出组成本发明的实施例的结果组 / 叶子块 600 的数据结构。结果组包含时效基 602，它表示对结果中 600 最近更新的日期和时间。它还包含多个关键字输入项 604，606，和 608。为简单只考虑关键字输入项 608，它又分成三部分，即输入项时效 608A，输入项关键字信息 608B 和输入项属性信息 608C。

[0272] 当作出瞬态输入项时，它具有与其相关的输入项时效数据 608A。此数据涉及时效单位和时效限止，它们对索引作为整体设定，但可选地能对单独的结果组设定。时效单位指定在单个时效单位中的秒数。因而如值 60 意为 1 分钟，而 86400 意为 1 天。实现限止表示时效单位之数目，在此之后关键字能从索引中丢失。这通常在 1 到 254 的范围中，而使它能

放入 8 位的字中。因此，若关键字设定为 31 天后过期，时效单位设成 86400 秒，而实现限止设定为 31。对于一天后过期的关键字，时效单位和时效限止可分别设成 3600 秒和 24。

[0273] 每个输入项包含以一个字节有效地表示的输入项时效，且这能在 0 和 255 之间设定，在此例中可合适地选择其它值范围。零指出输入项是新的，或比时效基老小于一个时效单位。255 表示此项入项已过时。

[0274] 对结果组中的每次更新或插入，按照下面顺序。首先记上当前的日期和时间，随后计算在时效基和当前日期及时间之间经过的时效单位的数。然后，在结果组中的每个输入项具有其随经过的时效单位增加的输入项时效。任何带有大于或等于 255 的计算的输入项时效的输入项将其输入项年代设成 255。然后任意那样的输入项成为可再用。接着，时效基设成在过程开始记录的日期和时间。任何要插入的关键字放入结果组中可用的槽，并设成零输入项时效。可用的槽可以是当未分配的槽，或已经过时（因为其输入项时效已达到 255）的槽。此外，任何更新的输入项具有复位到零的输入项时效。

[0275] 在上面实施例中，输入项时效作为单个字节存储。这给出 254 时间周期的范围，其中数据类型变成过时。可能使用多个字节以给出更宽的时效限止范围，但增加了存储开销。另外，能使用小于 1 个字节作为输入项时效，从而减少了存储开销，但也减少了可单独定义的时效限止的范围。

[0276] 因此有可能提供改善的数据库。

[0277] 判定图划分

[0278] 如上讨论，较好的是判定图（索引）整体保持在半导体存储器中，因为允许快速存取。但是运行数据库的硬件平台的容量或数据库的大小可能不能够总是在半导体存储器中保存整个索引。例如这可能由于机器期望是多任务的并完成其它功能。

[0279] 众所周知，如硬盘驱动器那样的海量存储设备划分成预定大小的块，任何小于块大小的文件仍占据一块。因此，物理存储器以块大小的倍数使用。当对那样的存储器读写判定图时，较好的是判定图以那样方式构造，使得其整体或尽可能多地写入单个块。因此，判定图看来驻留在一个存储块中。然而，当索引变得越来越大时，判定图随关键字总数增长而增长。因此总有不可能将判定图容纳在单个块的时候。从而需要将判定图容纳在多个块之中的方法。

[0280] 参考图 24，假设存储器块 700 表示在计算机系统的硬盘上一块的物理尺寸，该块被细分成包含部分 702，704，706 和 708 及其他，其每一个与相对判定图中的有关节点的数据相关。此外假设该索引现已达到块 700 已完全充满的大小。

[0281] 现假设，作为数据库插入操作的结果需要将又一个节点插入判定图。节点插入能在任何点发生。上面注意到，每个判定节点（图 3）包括判定值和表示图中下一节点的低指针和高指针。结果，能作出图的局部修改，使新节点能插入到任何现有节点之后。因此假设，作为插入结果需要在其数据由数据项 706 表示的节点之后插入新节点。首先当新节点被插入时，节点 706 的输出节点之一被修改以指向新节点。作为插入操作的部分，还正确地设置新节点的输出指针，使得判定图不再分开。然而若不再可能将新节点 710 有关的数据存入存储器块 700 时，关于新节点 710 的细节必须存入下一块 712。关于新节点 710 的另外节点必须建立，且信息存入存储器块 712。

[0282] 若以后需要插入与以前存在节点 708 有关的新节点，则存储器块 700 再次变满，且

因此下一新节点 714 的细节必须存在存储器块之外。此过程允许判定图增大，但能导致从每个满的（父辈块滋生出大量几乎空的存储器块。

[0283] 图 25 示出增加索引的另选方法。再次认为存储器块 700 是满且现在需要插入新节点。如前，建立新存储器图块。但现在父辈判定图划分成相等的部分。一部分放置在老的项 700，而另一部分放置在新的块 712。

[0284] 参考图 25，开始假设，索引 720 存在存储器块 700 且块 700 已满。假设一个索引插入导致建立新节点 722，且在此过程使得索引超过存储器块 700 的大小。然而，在此控制索引增长的方法中，索引的结构被分解，使得划分成基本相等的索引块。因此取图 25 所示的索引部分，能够看到，从节点 724 开始由虚线 726 包围的 10 个节点能通过延伸到节点 724 左侧的路径达到，而由虚线 726 包围的 10 个节点能通过延伸到节点 724 左侧的路径达到，而由虚线 728 包围的 12 个节点能在延伸到节点 724 右侧的路径中找到。通过选择测试节点。再据此计算节点以便为划分索引找到合适的候选节点，用这样的算法递归地扫描索引结构。在图 25 示出的例中，节点 724 和由线 726 包围的节点放在老的存储器 700 块中，而由虚线 728 包围的节点放在新的存储器块 712 中。此过程保证每个存储块是最优且均衡地使用，不再滋生基本空的存储块。

[0285] 限定词

[0286] 经常希望使用两个或更多搜索准则完成索引中的搜索项。这些准则是组合的，且通常记成布尔代数的原理。

[0287] 可能将查询分成各单独的部分，每部分在下一部分完成前进行，且随后在终点组合结果。但是这很花费计算时间，并需要很多对如硬盘那样的成块数据存储的读写操作。例如，返回 100 个选中的查询可能需要另外 100 次慢的盘 I/O 操作，以便取出由索引查询指向的数据。

[0288] 不是所有由查询取出的数据都需要。以你从，若作出数据库的查询要找出所有 1997 年注册的 Mercedes 车，对 1997 年注册的车的查询返回数以千计的结果。但其中只有很小数量是 Mercedes 车。而且对许多查询，不可能以某种方式改变查询的次序以减少不需要的结果的数目。

[0289] 此问题能通过使用串联的关键字克服。然而至今，那样的串联关键字将基本搜索关键字和附属搜索关键字的整体保持在组合的串联关键字中，因而组合关键字变得特别长。

[0290] 专利申请人认识到，通过结合限定词，尤其是将限定词附加到基本搜索关键字，能作出搜索性能的改善。限定词不直接代表第二或次级搜索项，而是次级项的缩短形式。在本发明中，通过对该或每个次级搜索项进行散列函数产生限定词。对用于从数据库插入或删除数据的查询使用同样的散列函数。因此，在建立数据库时，用主关键字（如前）并用零个或多个限定词关键字建立索引。

[0291] 在使用时，通过使用主关键字引导索引，而数据以通常方式存入结果组。然而，每个限定词被散列编码成单字节，这是范围从 0 到 255 的数，且与关键字输入项一起存入结果组。

[0292] 在查询期间，为主关键字提供零或多个限定词关键字。然后如前，使用主关键字引导索引。最后，一旦达到正确的一个或多个结果组通过主关键字值及限定词关键字散列码

与结果组中的输入项的匹配，扫描这些组。只返回匹配所有主关键字及所有限定词关键字散列码的输入项。

[0293] 应该注意，多个限定词关键字能映射到同一单字节散列码。因而有小的但有限的概率返回不需要的数据。然而假设使用好的随机散列编码，错误匹配的概率是 255 分之一（或小于 0.5％），能够看出，本发明保证返回所有正确结果，但也伴随小百分比的不正确结果。因此所有返回的关键字的搜索结果必须在数据存储中审查以排除任何不需要的输入项。然而索引大小的减少和查询过程的简化更多于对附加分析返回的搜索结果的补偿。由于一个不正确返回结果的概率随限定词数目的增加而减少，当使用更多限定词时优点增加了。在理想的环境中，坏结果的概率是 $(1/255)^N$。因此，用 2 个限定词 (N = 2)，坏结果的概率是 65025 分之一或小于 0.002％。用 3 个限定词，返回错误数据项的机会减少到约一千四百万分之一。

[0294] 可以看到，搜索关键字的大小和在结果组中数据存储的需求大为减少。因此，若试图搜索 8 个项，每个能直到 8 字节长，传统上对每个搜索关键字需要存储 64 个数据字节。在本发明中此需要减少到只有 15 个数据字节。

[0295] 在查询数据库时，每个限定词关键字是可选的或没有，可提供任何个或所有那样的关键字，因为它们不影响索引结构的引导。限定词关键字只影响在扫描结果组时选中的结果数目。因此提供限定词越多，查询结果越精确。

[0296] 应该注意，此限定词技术能应用于任何索引结构，如象 B- 树那样众所周知的结构。图 26 概略地示出按本发的第一发明搜索关键字的格式，其中关键字的第一部分包含主关键字数据，而关键字其余部分包括限定词 Q1，Q2，等到 Q7，每一个是由对原始附加关键字的散列函数产生的一个字节长的字。

[0297] 模式匹配

[0298] 常希望在数据库中作模式匹配查询。因此不是对搜索关键字的整体搜索，而是使用关键字的辅助部分完成搜索。

[0299] 那样搜索的有效性取决于在判定图中的每个节点中关键字总体划分得如何好。

[0300] 对 ASCII 字符集的问题是它包括 255 个字符，但是某些字符出现的或然率很高，而双字符组合特别少。

[0301] 专利申请者认识到，若 ASCII 字符集（或等价的集）映射到更小的集，模式匹配能大为改善。

[0302] 专利申请人确定若干映射，它们将字符集减少到 16 字符，使得单字符装入 4 位 (0-15) 中，而 2 字符序列能装入单个字节中。

[0303] 这些修改的字符集能在单个索引中用于划分关键字的总数。

[0304] 字符集的映射在引导判定图作插入，删除或查询操作之前应用于关键字。用结果组存储原始关键字值还是转换后的值（较小的大小）取决于需要精确查询结果还是概率的查询结果。下面给出两个字符集的例子：

[0305] 语音集

[0306]

ASCII 字符	映射值
元音 (A,E,I,O,U)	删除
数字 (0,1,2,3,4,5,6,7,8,9)	删除
标点符号	删除
空格,Tab	0
B	1
C,K,Q	2
D	3
F	4
G,J	5
H	6
K	7
L	8
M,N	9
P	10
R	11
S,Z	12
T	13
V	14
W	15
X,Y	删除

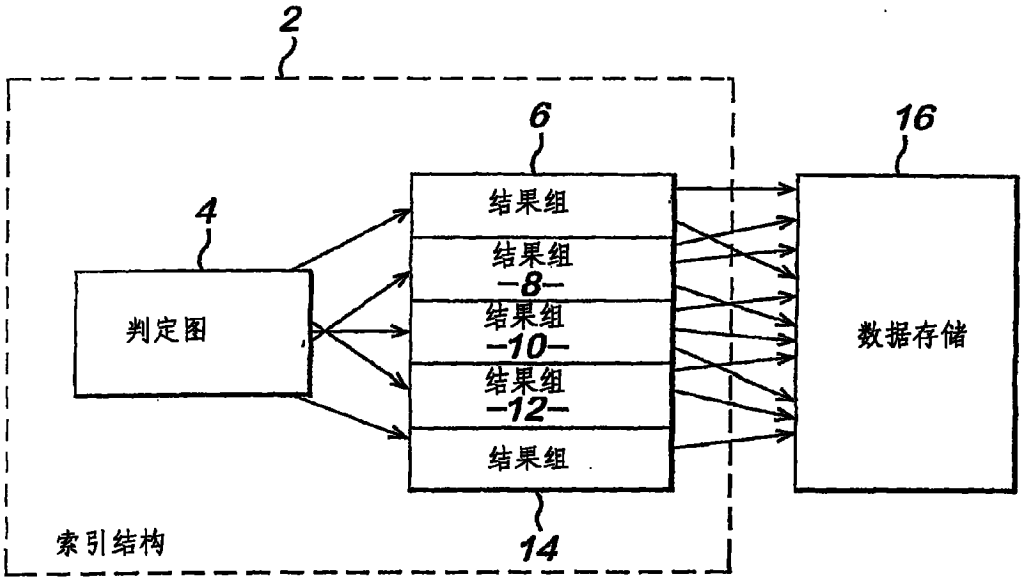
[0307] 类似的集

[0308]

ASCII 字符	映射值
空格，Tab	0
标点符号	1
数字 (0,1,2,3,4,5,6,7,8,9)	2
元音 (A,E,I,O,U)	3
B,D	4
C,K	5
F,P	6
G,J	7
H,Y	8
K,L	9
M,N	10
P,Q	11
R,S	12
T	13
V,W	14
X,Z	15

[0309] 在每个这些方案中字符转换成大写，使得查询无特定大小写。此外，如换行及回车那样非打印字符从映射中去除。

[0310] 通过使用此缩减的字符集，模式匹配的性能可以大为改善。在插入，查询或删除操作之前映射关键字，并随后使用映射的关键字引导索引。



引导

1. 所有引导开始于判定图中的原点
2. 跟随判定图中的判定节点，直到找到结果组或不可能继续引导
3. 扫描结果组中的所有关键字以找到需要的关键字

图 1

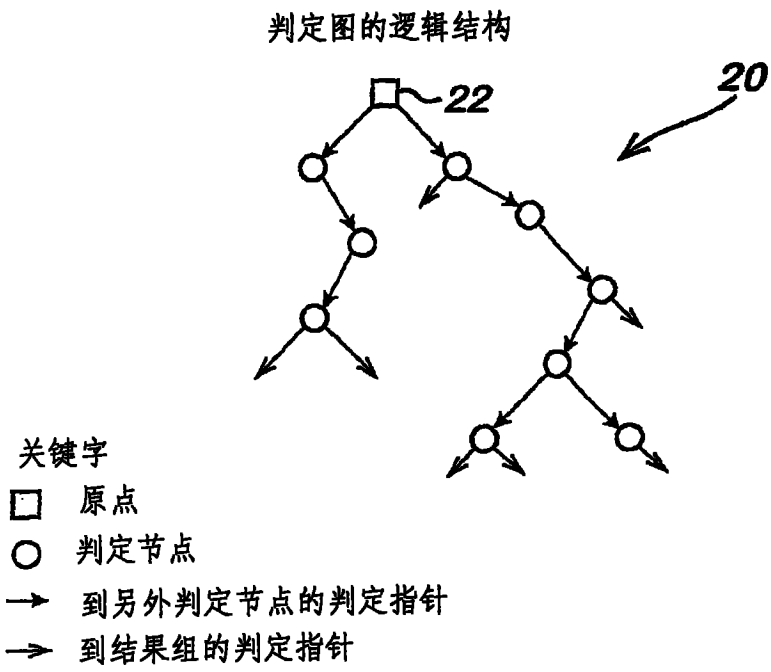
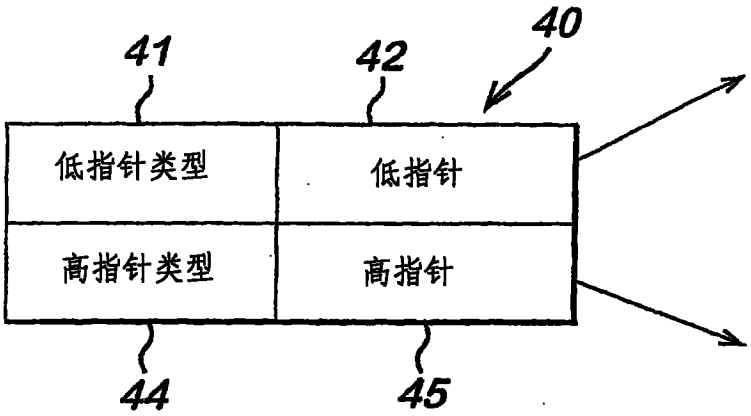


图 2

判定节点的逻辑结构



注

1、 低指针类型指出低指针的目的，它能指出该指针是否指向判定节点或结果组。

2、 低指针给出它指向的节点或结果组的地址，零值能表示不存在指针。

3、 高指针类型指出高指针的目的。它能指出给指针是否指向节点或结果组。

4、 该指针给出它指向的节点或结果组的地址，零值能表示不存在指针。

图 3

结果组的逻辑结构

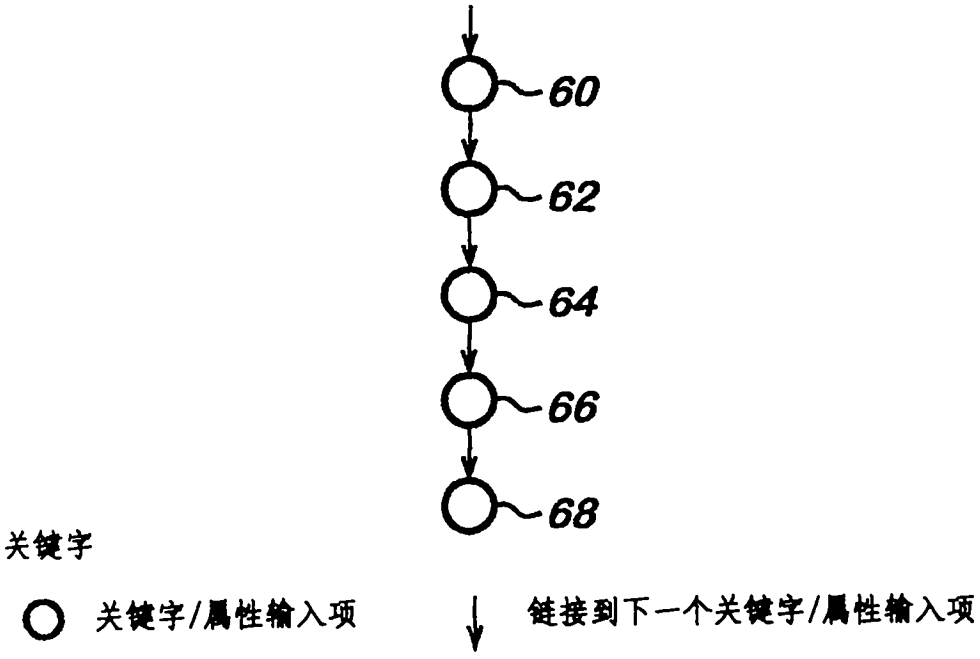
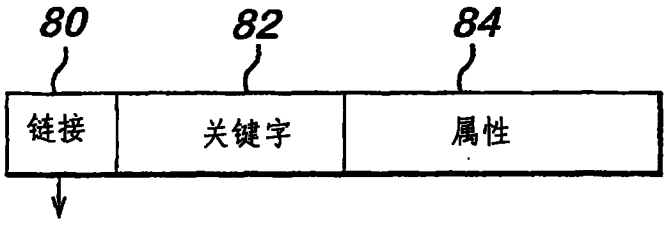


图 4

结果组输入项的逻辑结构



注

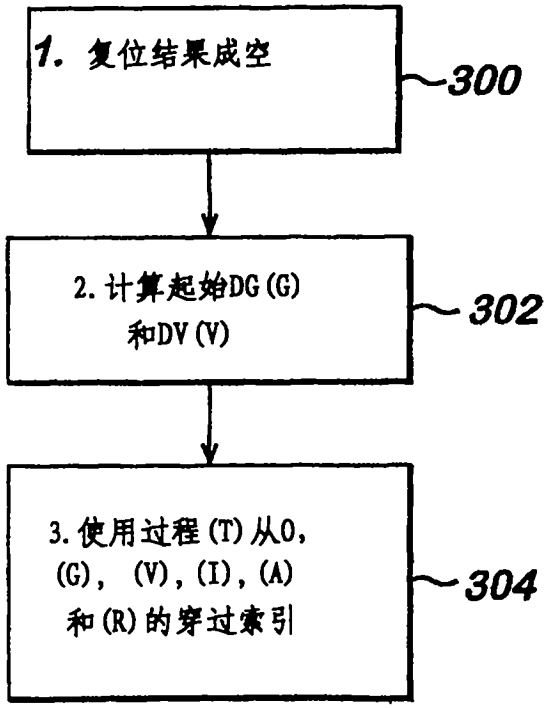
1、 链接字段指向下一个结果组输入项。零值表示不再有输入项。

2、 关键字字段保持精确的关键字值。

3、 属性字段保持与该关键字有关的属性。

图 5

对带Min Key(I)、Max Key(A)的范围查询过程(G)，
并返回结果(R)



缩写

CS－结果组；DG－判定组；DN－判定节点；DP－判定指针；
DV－判定值；HP－高指针；LP－低指针；O－原点；

图 11

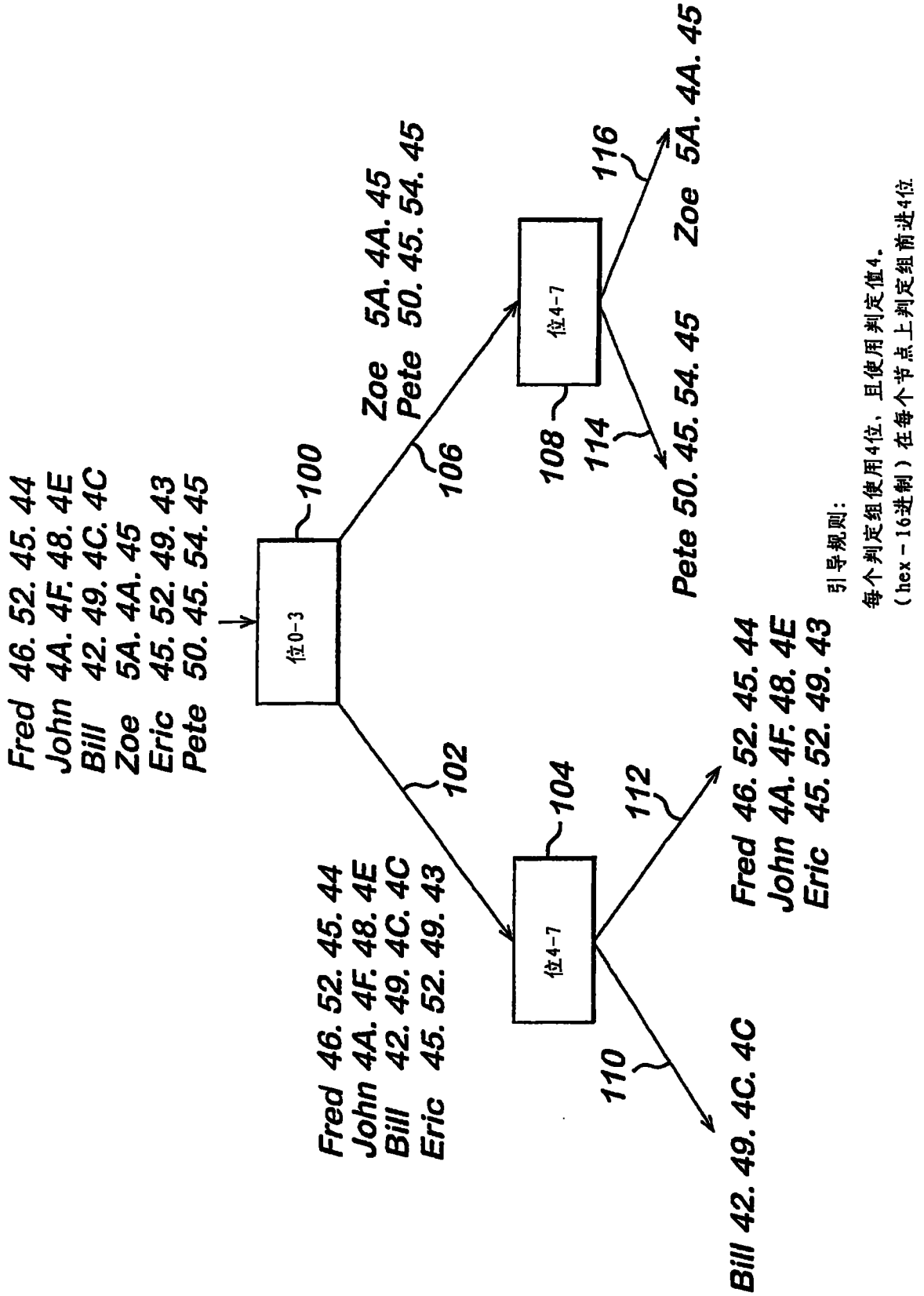

Fred 46. 52. 45. 44
John 4A. 4F. 48. 4E
Bill 42. 49. 4C. 4C
Zoe 5A. 4A. 45
Eric 45. 52. 49. 43
Pete 50. 45. 54. 45
100
位0-3
102
106
108
114
116
位4-7
Zoe 5A. 4A. 45
Pete 50. 45. 54. 45
Zoe 5A. 4A. 45
Pete 50. 45. 54. 45
Fred 46. 52. 45. 44
John 4A. 4F. 48. 4E
Bill 42. 49. 4C. 4C
Eric 45. 52. 49. 43
104
位4-7
110
112
Fred 46. 52. 45. 44
John 4A. 4F. 48. 4E
Eric 45. 52. 49. 43
Bill 42. 49. 4C. 4C
引导规则:
每个判定组使用4位，且使用判定值4。
(hex-16进制)在每个节点上判定组前进4位

图 6

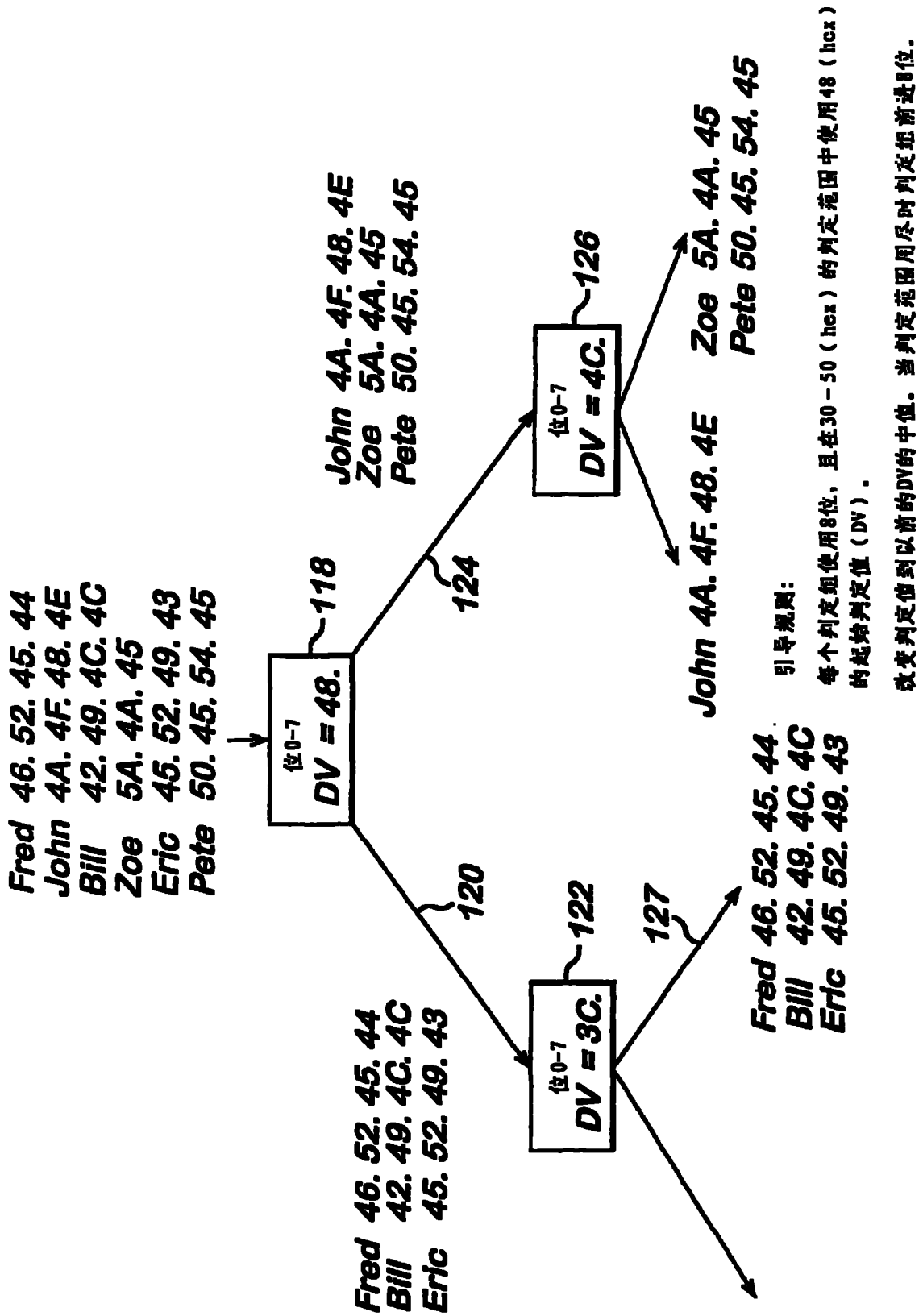

Fred 46. 52. 45. 44
John 4A. 4F. 48. 4E
Bill 42. 49. 4C. 4C
Zoe 5A. 4A. 45
Eric 45. 52. 49. 43
Pete 50. 45. 54. 45
118
位0-7
DV = 48.
124
120
位0-7
DV = 4C.
126
位0-7
DV = 3C.
122
127
John 4A. 4F. 48. 4E
Zoe 5A. 4A. 45
Pete 50. 45. 54. 45
Fred 46. 52. 45. 44
Bill 42. 49. 4C. 4C
Eric 45. 52. 49. 43
Fred 46. 52. 45. 44
Bill 42. 49. 4C. 4C
Eric 45. 52. 49. 43
引导规则:
每个判定组使用8位，且在30－50（hex）的判定范围中使用48（hex）的起始判定值（DV）。
改变判定值到以前的DV的中值。当判定范围用尽时判定组前进8位。

图 7

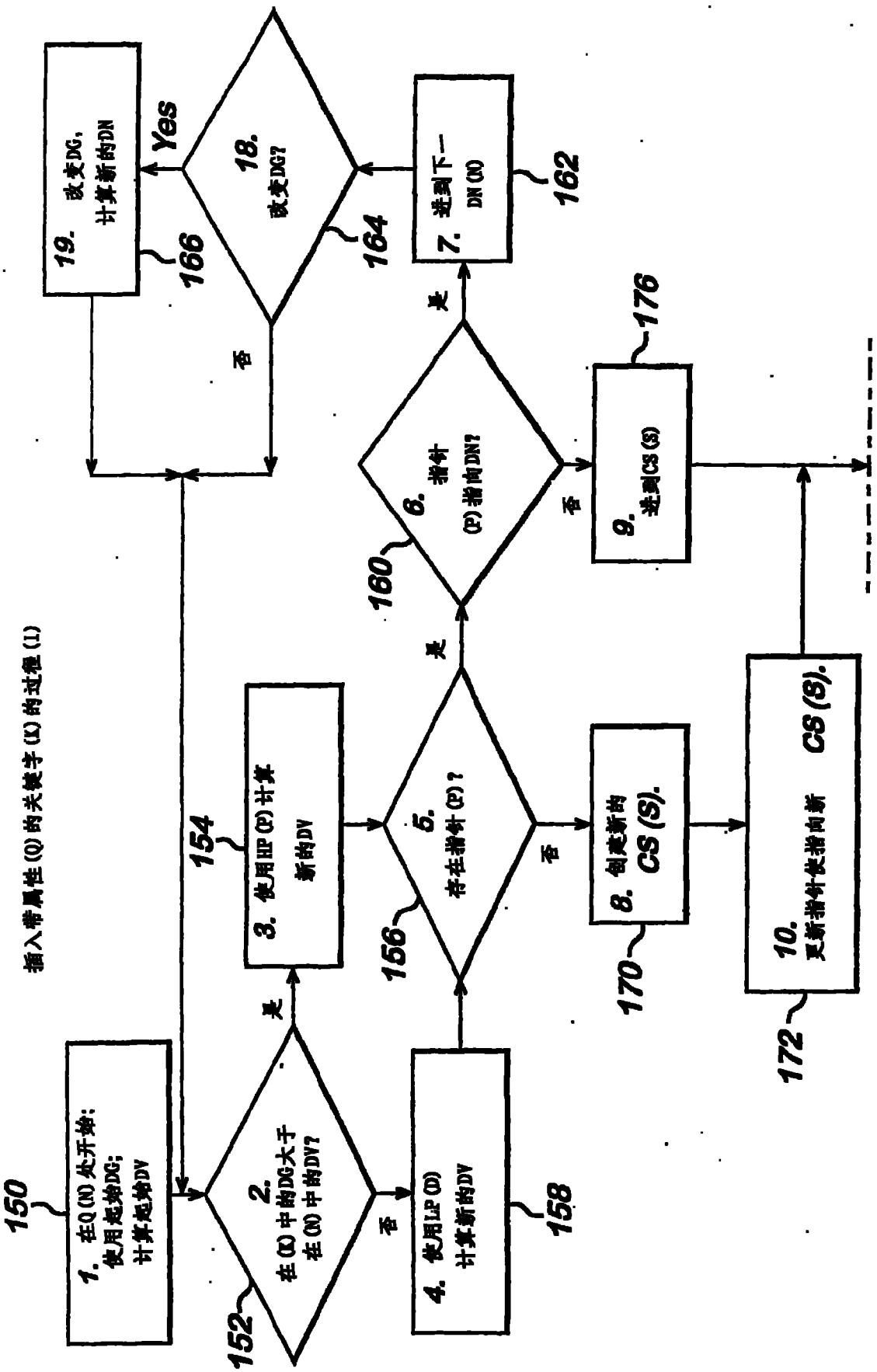

插入带属性(Q)的关键字(L)的过程(L)
150
1. 在Q(N)处开始;
使用起始DG;
计算起始DV
152
2.
在(L)中的DG大于
在(N)中的DV?
是
否
154
3. 使用LP(P)计算
新的DV
158
4. 使用LP(P)
计算新的DV
156
5.
存在指针(P)?
是
否
160
6.
指针
(P)指向DN?
是
否
162
7. 进到下一
DN(N)
164
18.
改变DG?
Yes
否
166
19. 改变DG,
计算新的DN
170
8. 创建新的
CS(S).
176
9.
进到CS(S)
172
10.
更新指针使指向新
CS(S).

图 8

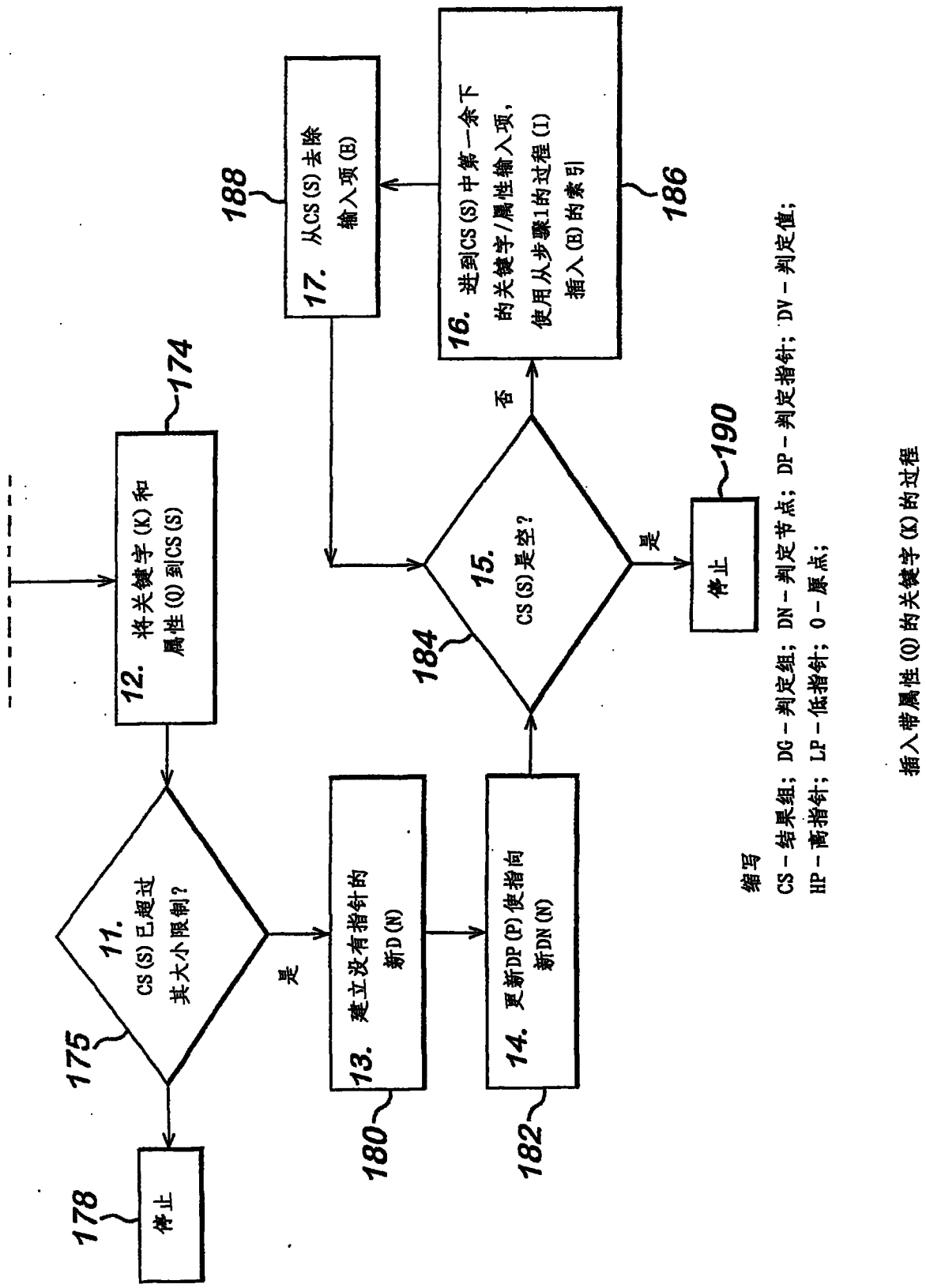

插入带属性(Q)的关键字(K)的过程
12. 将关键字(K)和属性(Q)到CS(S)
174
11. CS(S)已超过其大小限制?
175
是
否
178
停止
13. 建立没有指针的新D(N)
180
14. 更新DP(P)使指向新DN(N)
182
15. CS(S)是空?
184
是
190
停止
否
16. 进到CS(S)中第一余下的关键字/属性输入项,使用从步骤1的过程(I)插入(E)的索引
186
17. 从CS(S)去除输入项(E)
188
缩写
CS－结果组; DG－判定组; DN－判定节点; DP－判定指针; DV－判定值;
HP－高指针; LP－低指针; 0－原点;

图 8(续)

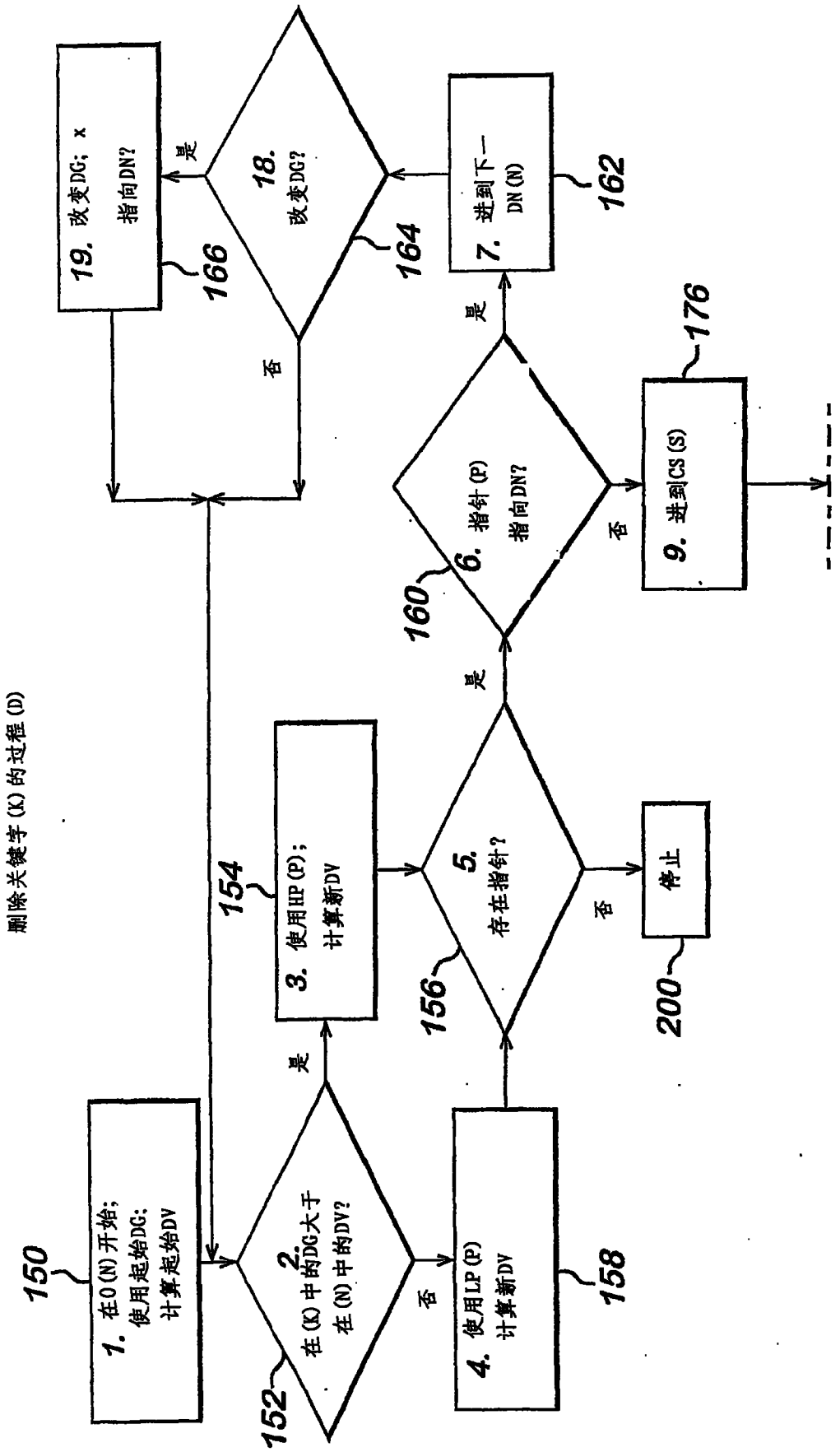

删除关键字(K)的过程(D)
150
1. 在O(N)开始;
使用起始DG;
计算起始DV
152
2.
在(K)中的DG大于
在(N)中的DV?
是
否
154
3. 使用HP(P);
计算新DV
156
5.
存在指针?
是
否
158
4. 使用LP(P)
计算新DV
160
6. 指针(P)
指向DN?
是
否
162
7. 进到下一
DN(N)
164
18.
改变DG?
是
否
166
19. 改变DG; x
指向DN?
176
9. 进到CS(S)
200
停止

图 9

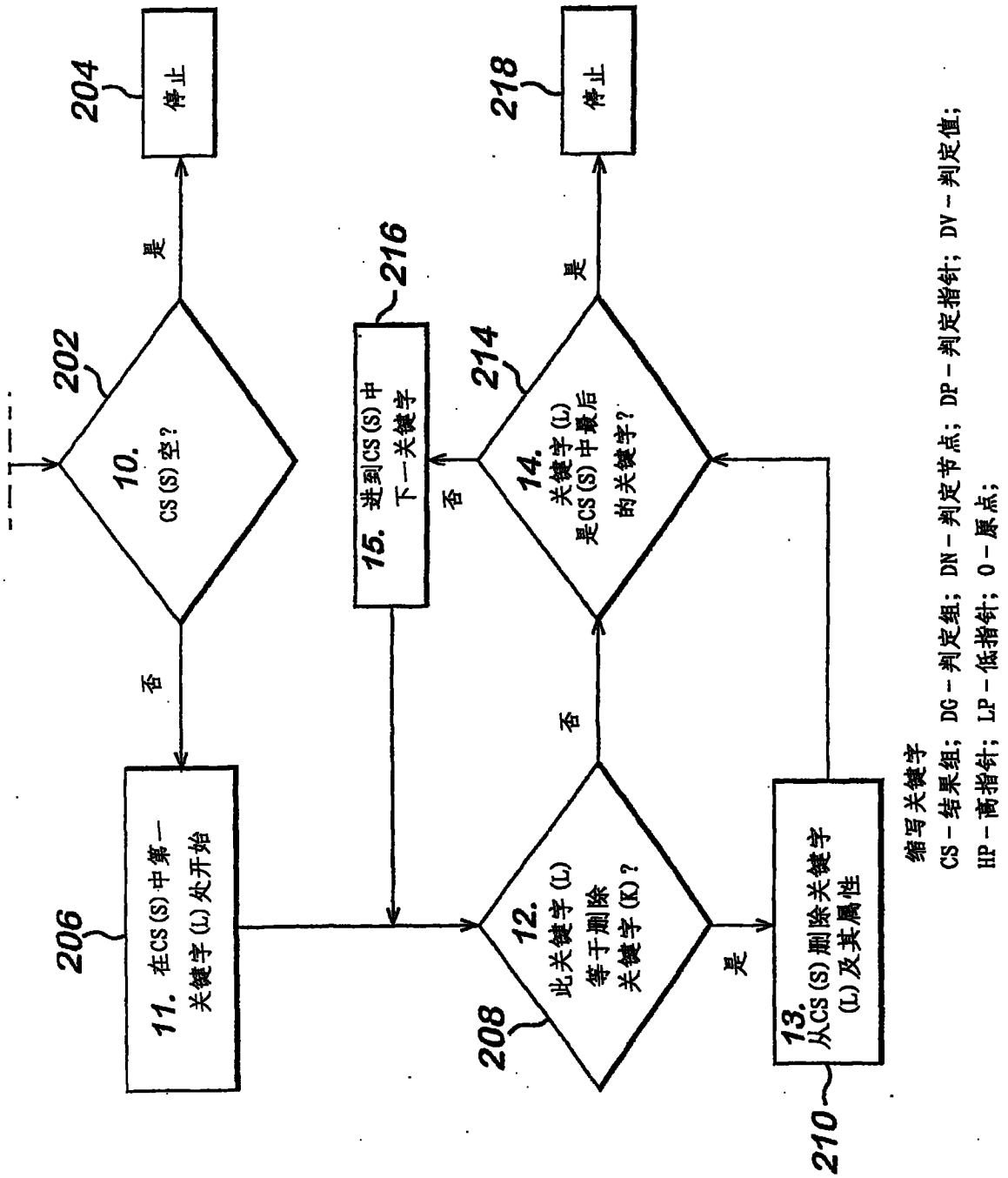

202
10. CS (S) 空?
是
204
停止
否
206
11. 在CS (S) 中第一关键字 (L) 处开始
208
12. 此关键字 (L) 等于删除关键字 (K)?
是
210
13. 从CS (S) 删除关键字 (L) 及其属性
否
214
14. 关键字 (L) 是CS (S) 中最后的关键字?
是
218
停止
否
216
15. 进到CS (S) 中下一关键字
缩写关键字
CS - 结果组; DG - 判定组; DN - 判定节点; DP - 判定指针; DV - 判定值;
HP - 高指针; LP - 低指针; 0 - 原点;

图 9(续)

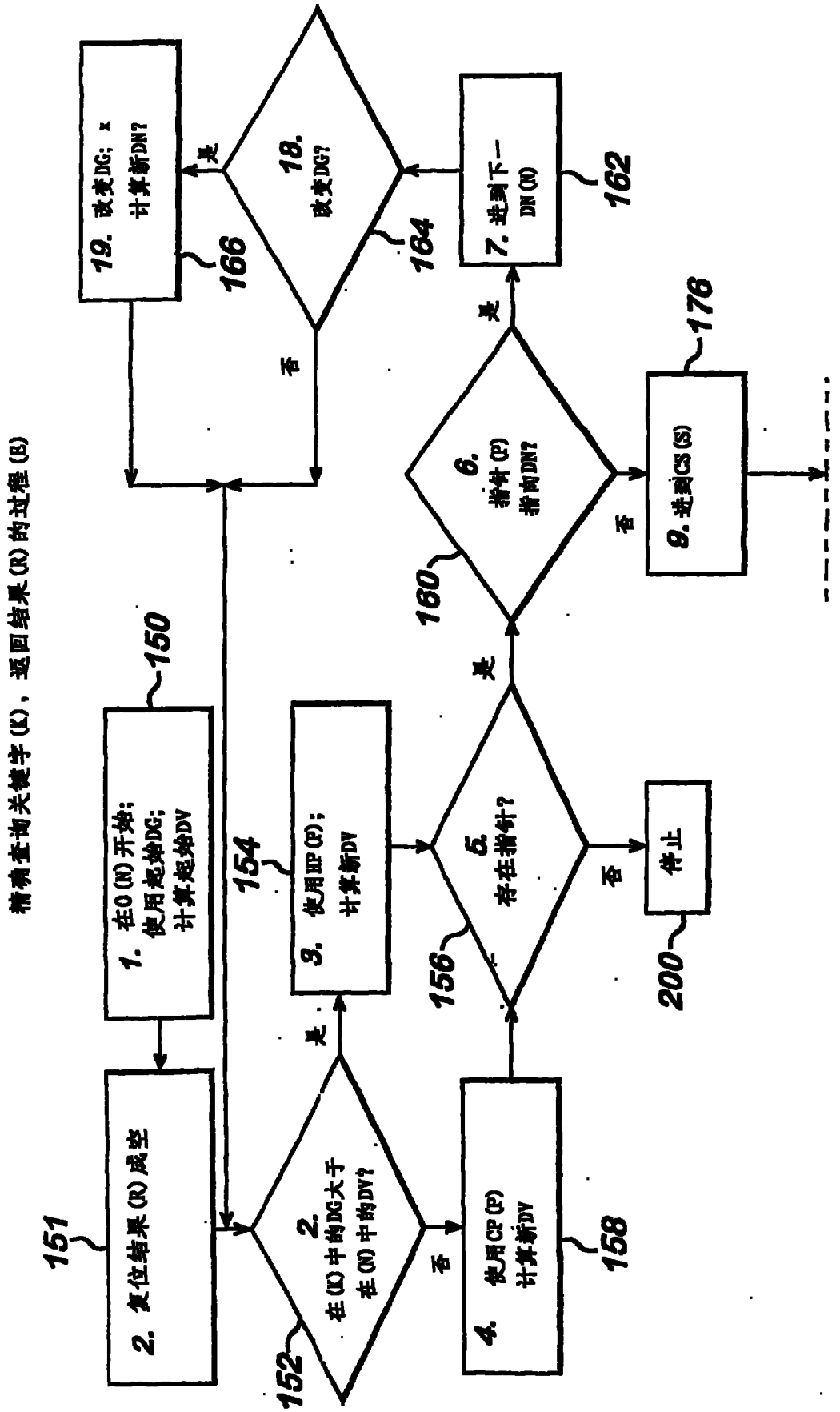

精确查询关键字(K)，返回结果(R)的过程(E)
1. 在O(N)开始；使用起始DG；计算起始DV
150
2. 复位结果(R)成空
151
2. 在(K)中的DG大于在(N)中的DV?
152
是
否
3. 使用DP(P)；计算新DV
154
4. 使用CP(P)计算新DV
158
5. 存在指针?
156
是
否
停止
200
6. 指针(P)指向DN?
160
是
否
7. 进到下一DN(N)
162
8. 进到CS(S)
176
18. 改变DG?
164
是
否
19. 改变DG；x计算新DN?
166

图 10

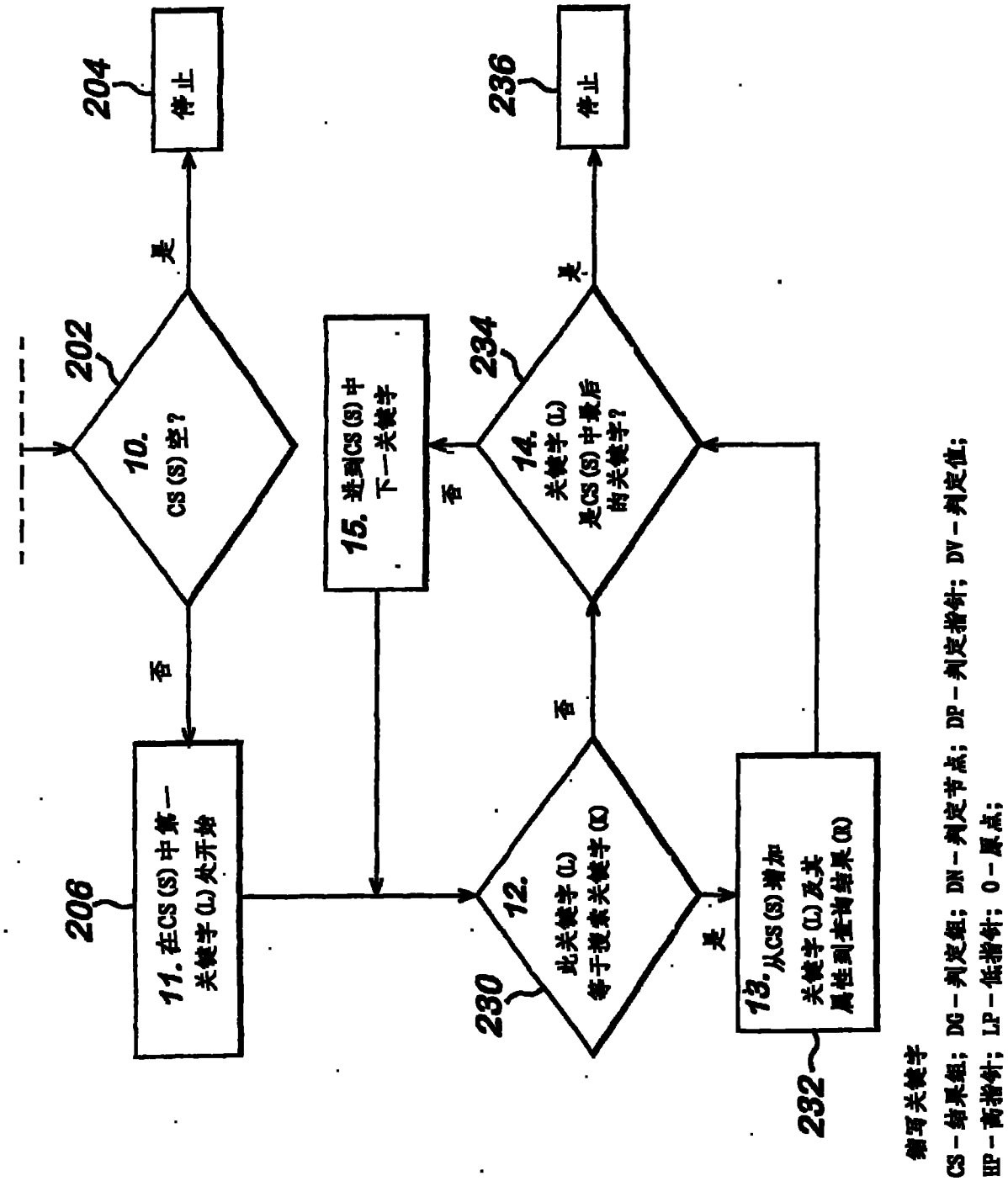

204
停止
202
10. CS(S)空?
是
否
206
11. 在CS(S)中第一关键字(L)处开始
15. 选到CS(S)中下一关键字
234
14. 关键字(L)是CS(S)中最后的关键字?
236
停止
是
否
12. 此关键字(L)等于搜索关键字(K)
230
是
13. 从CS(S)增加关键字(L)及其属性到查询结果(R)
232
编写关键字
CS-结果组; DG-判定组; DN-判定节点; DP-判定指针; DV-判定值;
HP-高指针; LP-低指针; 0-原点;

图 10(续)

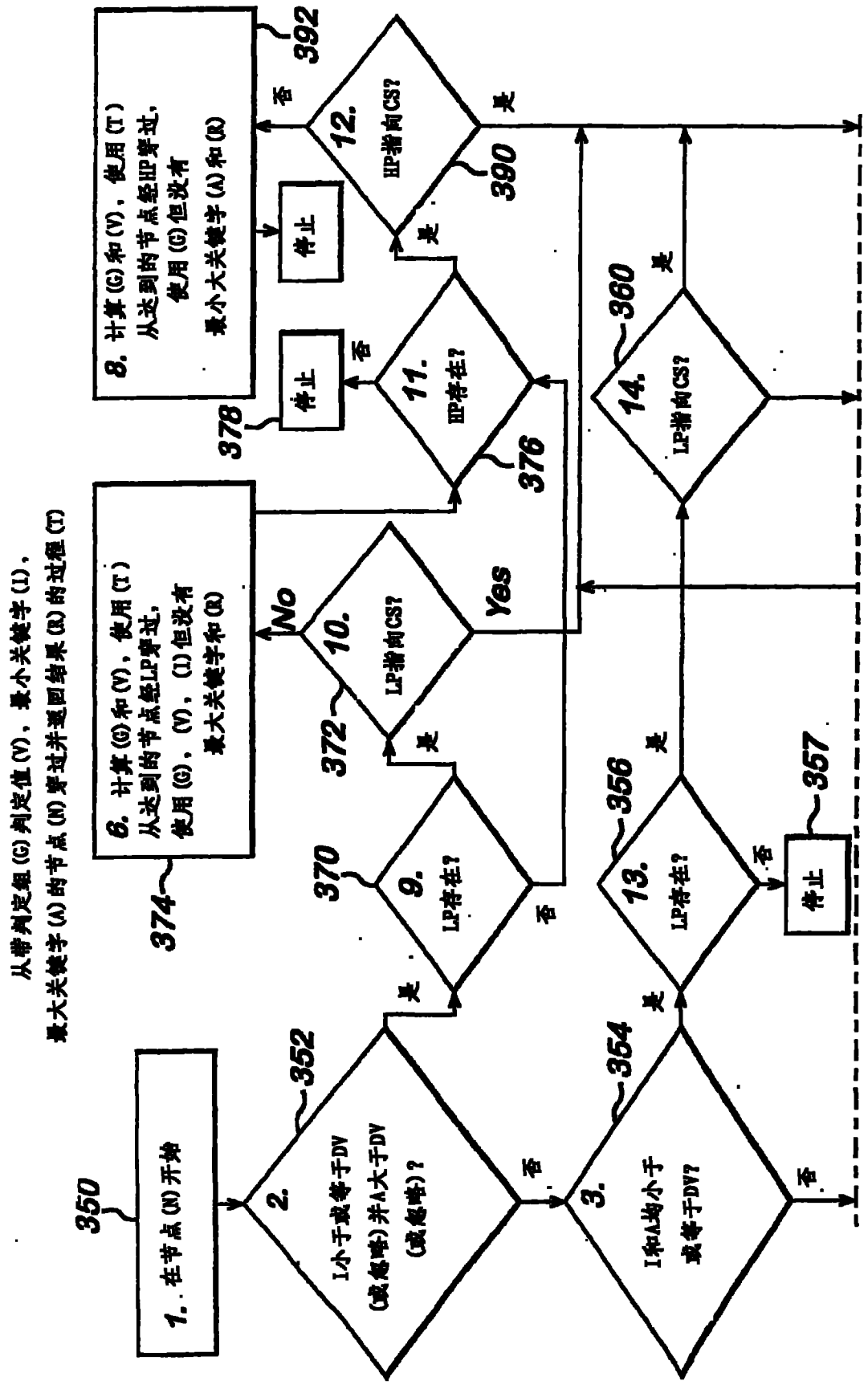

从带判定组(G)判定值(V)，最小关键字(L)，
最大关键字(A)的节点(N)穿过并返回结果(R)的过程(T)
350
1. 在节点(N)开始
352
2. L小于或等于DV (或虚拟) 并A大于DV (或虚拟)?
是
否
354
3. L和A均小于或等于DV?
是
否
356
13. LP存在?
是
否
357
停止
360
14. LP指向CS?
是
370
9. LP存在?
是
否
372
10. LP指向CS?
Yes
No
374
6. 计算(G)和(V)，使用(T)从达到的节点经LP穿过，使用(G)，(V)，(L)但没有最大关键字和(R)
376
11. HP存在?
是
否
378
停止
390
12. HP指向CS?
是
否
392
8. 计算(G)和(V)，使用(T)从达到的节点经HP穿过，使用(G)但没有最小大关键字(A)和(R)
停止

图 12

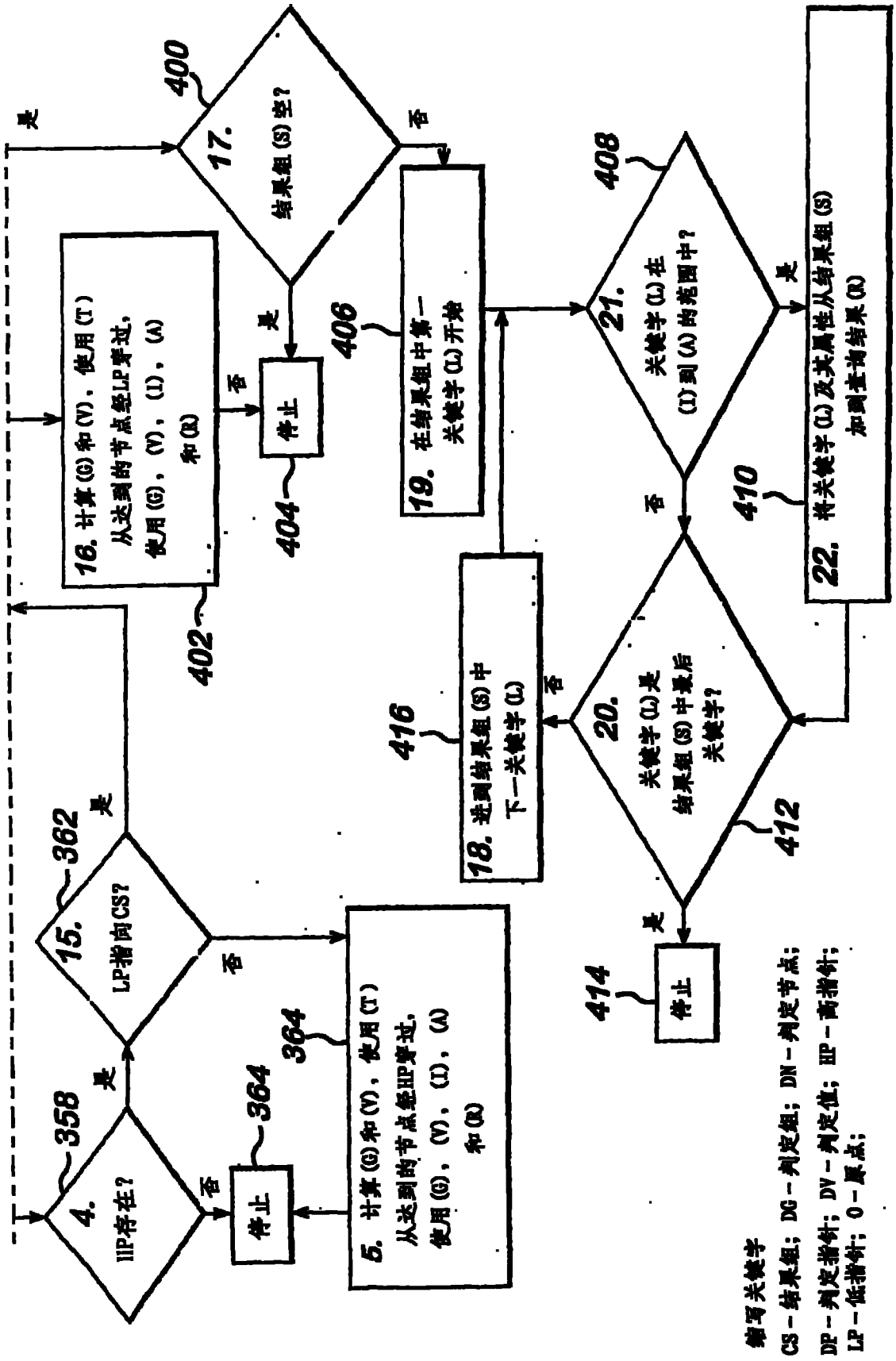

4. IIP存在?
是
否
358
5. 计算(G)和(V)，使用(T)从达到的节点经IIP算过，使用(G)，(V)，(I)，(A)和(R)
364
停止
364
15. LP指向CS?
是
否
362
16. 计算(G)和(V)，使用(T)从达到的节点经LP算过，使用(G)，(V)，(I)，(A)和(R)
402
17. 结果组(S)空?
是
否
400
停止
404
18. 达到结果组(S)中下一关键字(L)
416
19. 在结果组中第一关键字(L)开始
406
20. 关键字(L)是结果组(S)中最后关键字?
是
否
412
21. 关键字(L)在(I)到(A)的范围中?
是
否
408
22. 将关键字(L)及其属性从结果组(S)加到查询结果(R)
410
停止
414
编写关键字
CS－结果组；DG－判定组；DN－判定节点；
DP－判定指针；DV－判定值；HP－高指针；
LP－低指针；0－最点；

图 12(续)

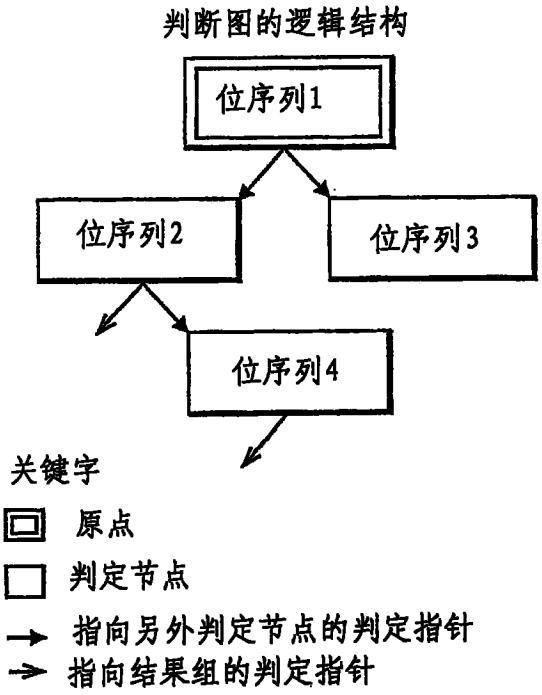
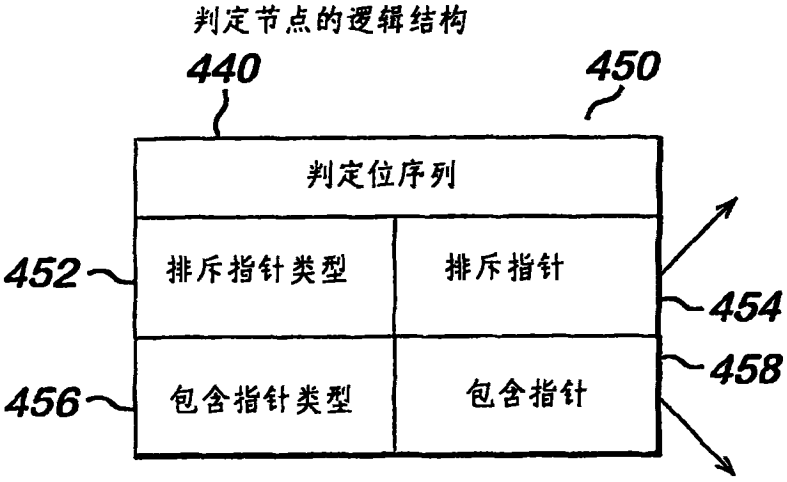


图 13



注

1. 判定位序到包含任意长位序列。
2. 排斥指针类型指出排斥指针的目的，它能指出指针指向判定节点或结果组。
3. 排斥指针给出它所指向的判定节点或结果组的地址。零值表示不存在指针。
4. 包含指针类型指出包含指针的目的，它能指出指针指向判定节点或结果组
5. 包含指针给出它所指向的判定节点或结果组的地址。零值表示不存在指针。

图 14

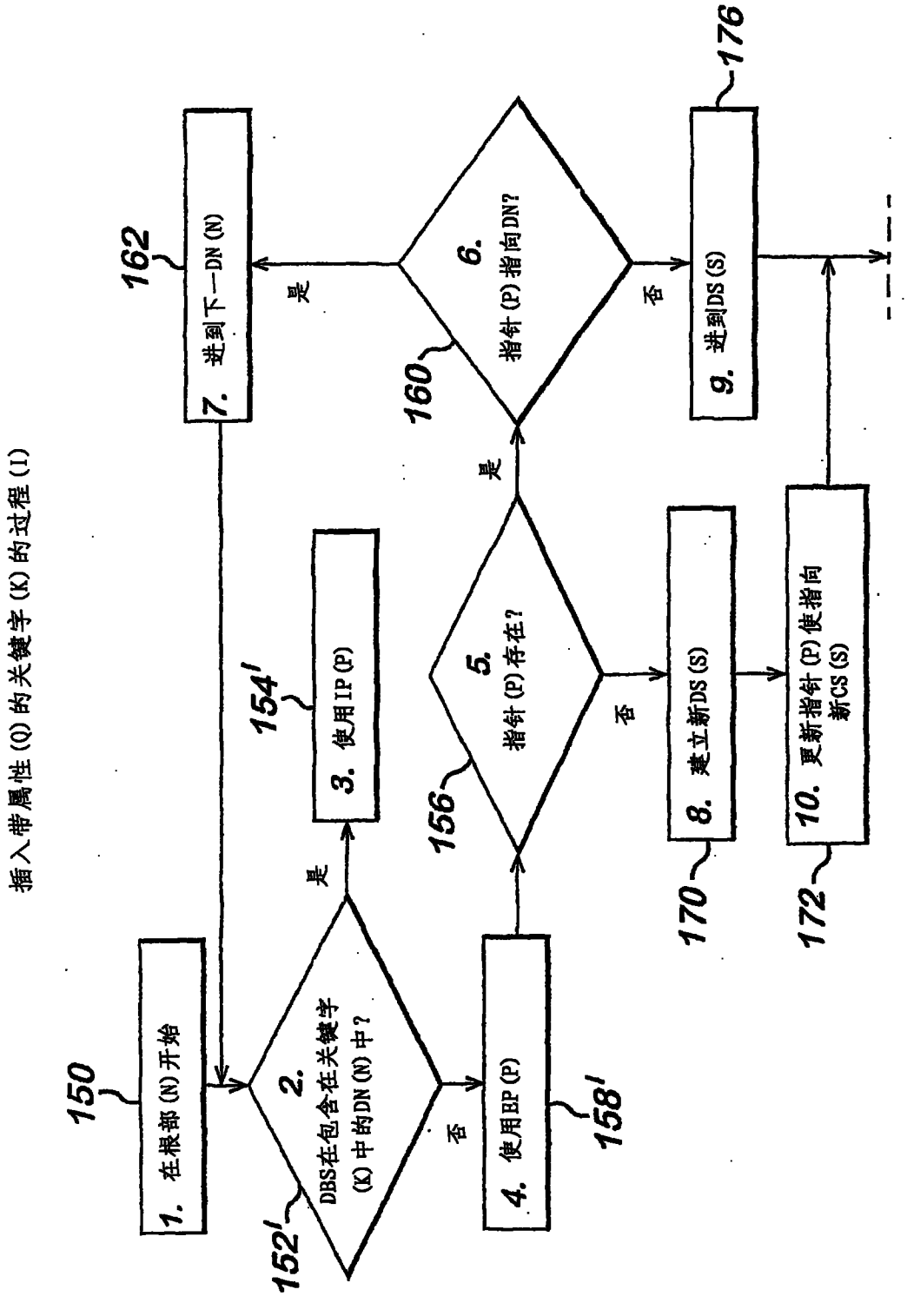

插入带属性 (Q) 的关键字 (K) 的过程 (I)
1. 在根部 (N) 开始
150
2. DBS在包含在关键字 (K) 中的DN (N) 中?
152'
是
否
3. 使用IP (P)
154'
4. 使用BP (P)
158'
5. 指针 (P) 存在?
156
是
否
6. 指针 (P) 指向DN?
160
是
否
7. 进到下一DN (N)
162
8. 建立新DS (S)
170
9. 进到DS (S)
176
10. 更新指针 (P) 使指向新CS (S)
172

图 15

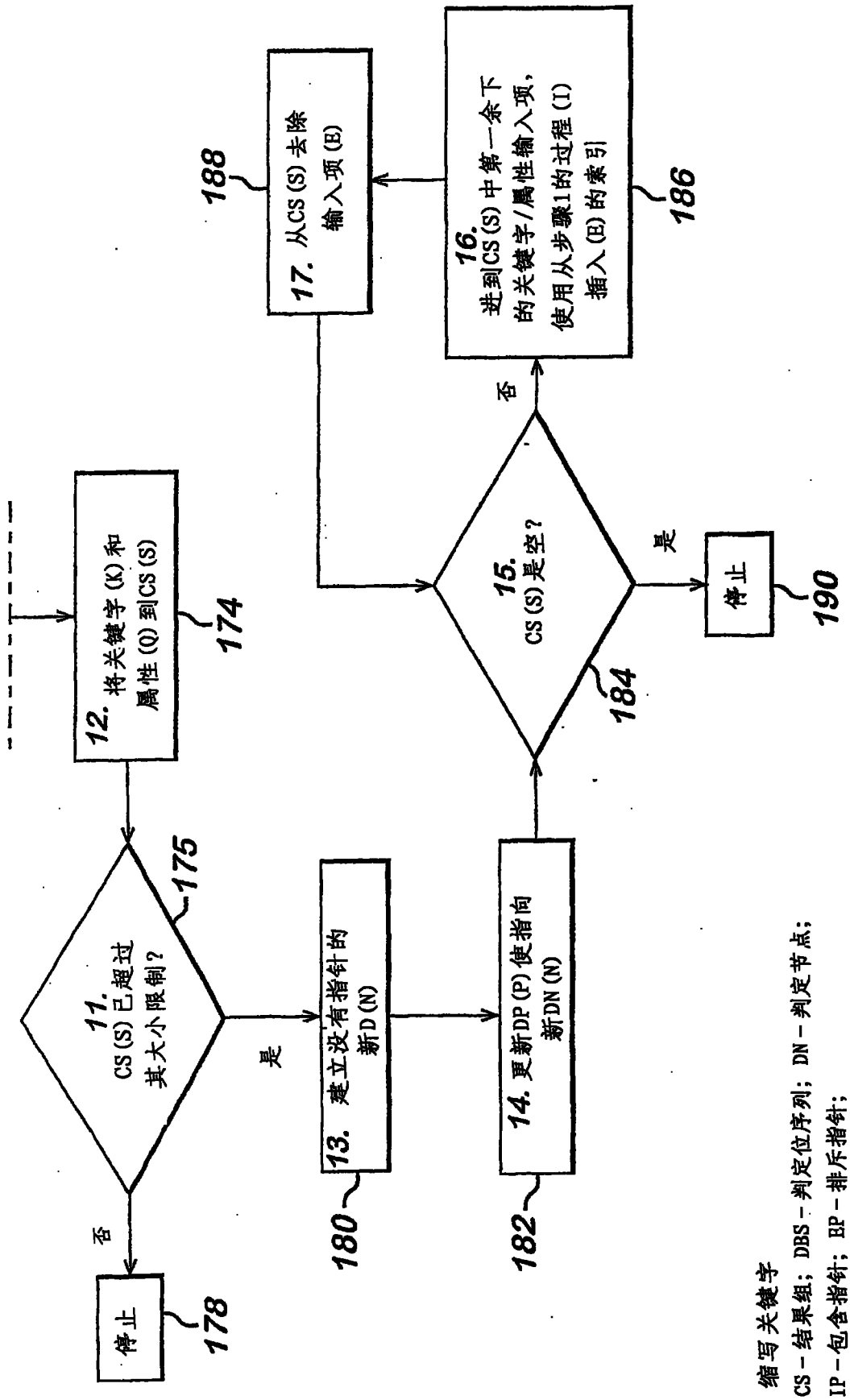

12. 将关键字(K)和属性(Q)到CS(S)
174
11. CS(S)已超过其大小限制?
175
否
停止
178
是
13. 建立没有指针的新D(N)
180
14. 更新DP(P)使指向新DN(N)
182
15. CS(S)是空?
184
是
停止
190
否
16. 进到CS(S)中第一余下的关键字/属性输入项,使用从步骤1的过程(I)插入(E)的索引
186
17. 从CS(S)去除输入项(E)
188
缩写关键字
CS-结果组; DBS-判定位序列; DN-判定节点;
IP-包含指针; EP-排斥指针;

图 15(续)

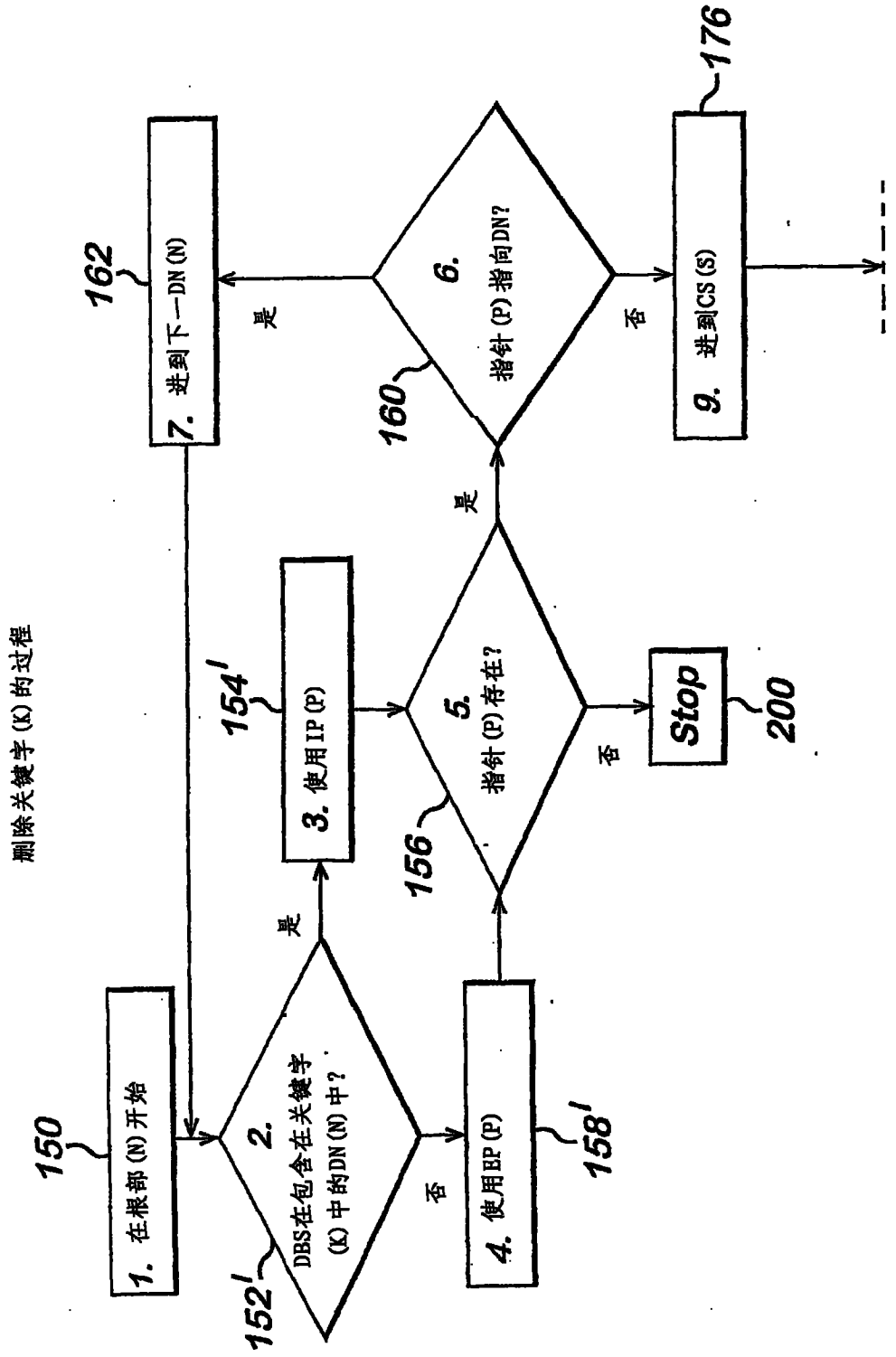

删除关键字(K)的过程
150
1. 在根部(N)开始
152'
2. DBS在包含在关键字(K)中的DN(N)中?
是
否
154'
3. 使用IP(P)
158'
4. 使用BP(P)
156
5. 指针(P)存在?
是
否
Stop
200
160
6. 指针(P)指向DN?
是
否
162
7. 进到下一DN(N)
176
9. 进到CS(S)

图 16

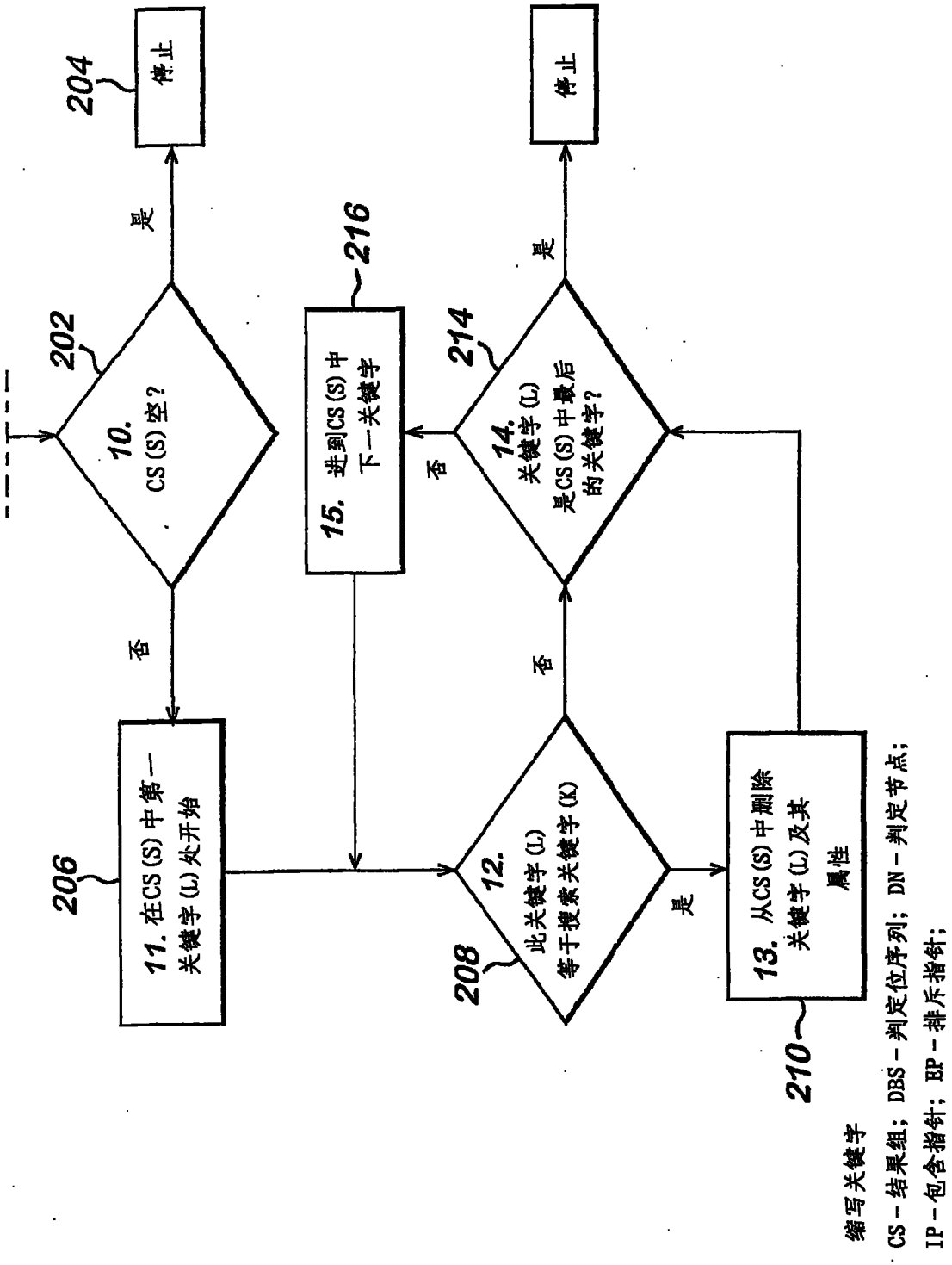

204
停止
是
202
10.
CS (S) 空?
否
206
11. 在CS (S) 中第一
关键字 (L) 处开始
208
12.
此关键字 (L)
等于搜索关键字 (K)
是
210
13. 从CS (S) 中删除
关键字 (L) 及其
属性
否
214
14.
关键字 (L)
是CS (S) 中最后
的关键字?
是
停止
否
216
15. 进到CS (S) 中
下一关键字
缩写关键字
CS - 结果组; DBS - 判定位序列; DN - 判定节点;
IP - 包含指针; EP - 排斥指针;

图 16(续)

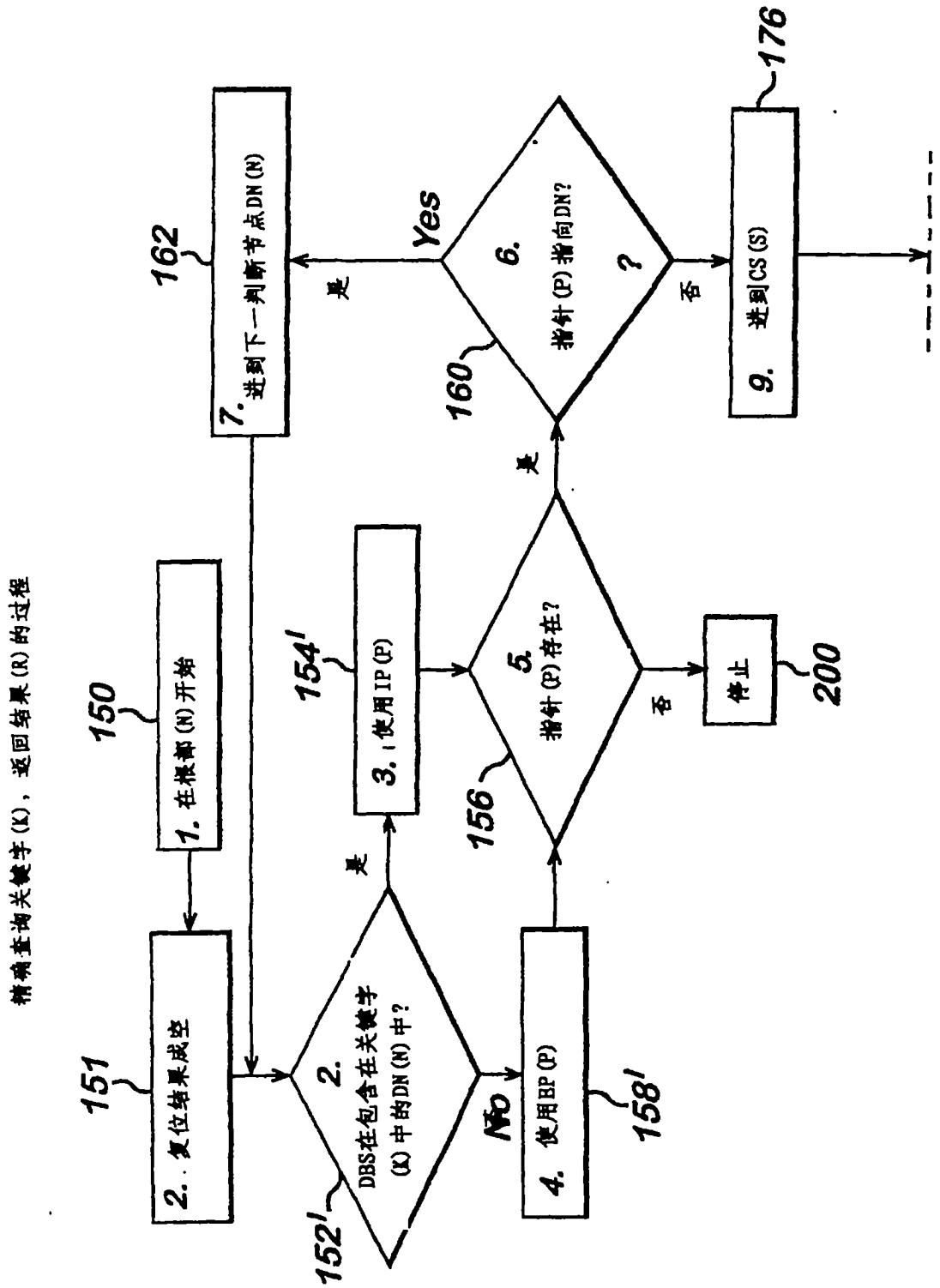

精确查询关键字(K)，返回结果(R)的过程
1. 在根部(N)开始
150
2. 复位结果成空
151
2. DBS在包含在关键字(K)中的DN(N)中?
152'
是
否
3. 使用IP(P)
154'
4. 使用BP(P)
158'
5. 指针(P)存在?
156
否
是
停止
200
6. 指针(P)指向DN?
160
Yes
是
否
7. 进到下一判断节点DN(N)
162
9. 进到CS(S)
176

图 17

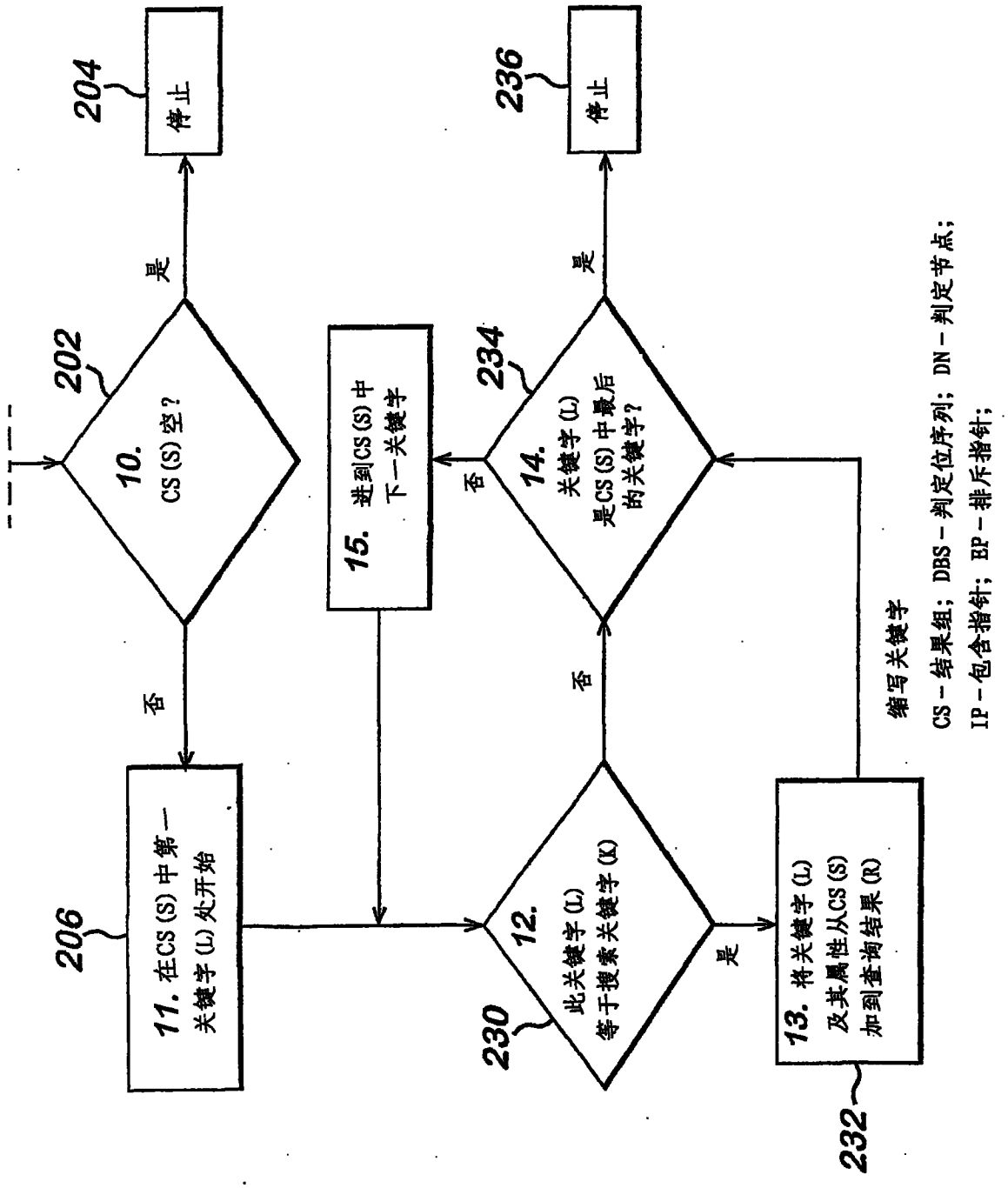

204
停止
202
10. CS (S) 空?
是
否
206
11. 在CS (S) 中第一关键字 (L) 处开始
15. 进到CS (S) 中下一关键字
234
14. 关键字 (L) 是CS (S) 中最后的关键字?
236
停止
是
否
230
12. 此关键字 (L) 等于搜索关键字 (K)
是
232
13. 将关键字 (L) 及其属性从CS (S) 加到查询结果 (R)
缩写关键字
CS－结果组; DBS－判定位序列; DN－判定节点;
IP－包含指针; EP－排斥指针;

图 17 (续)

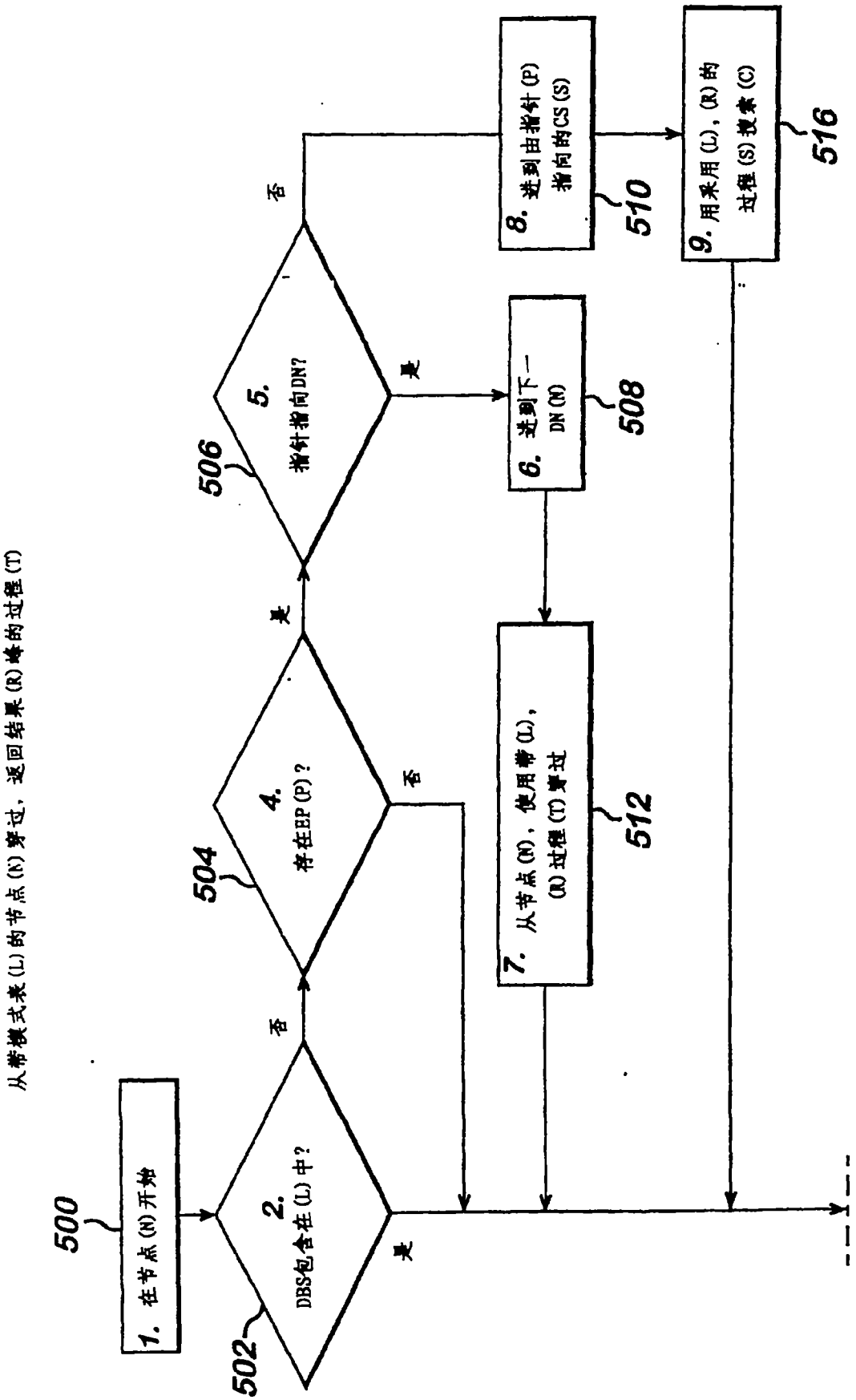

从带模式表(L)的节点(N)穿过，返回结果(R)峰的过程(T)
500
1. 在节点(N)开始
502
2. DBS包含在(L)中?
否
是
504
4. 存在EP(P)?
否
是
506
5. 指针指向DN?
否
是
508
6. 进到下一DN(N)
510
8. 进到由指针(P)指向的CS(S)
512
7. 从节点(N)，使用带(L)，(R)过程(T)算过
516
9. 用采用(L)，(R)的过程(S)搜索(C)

图 18

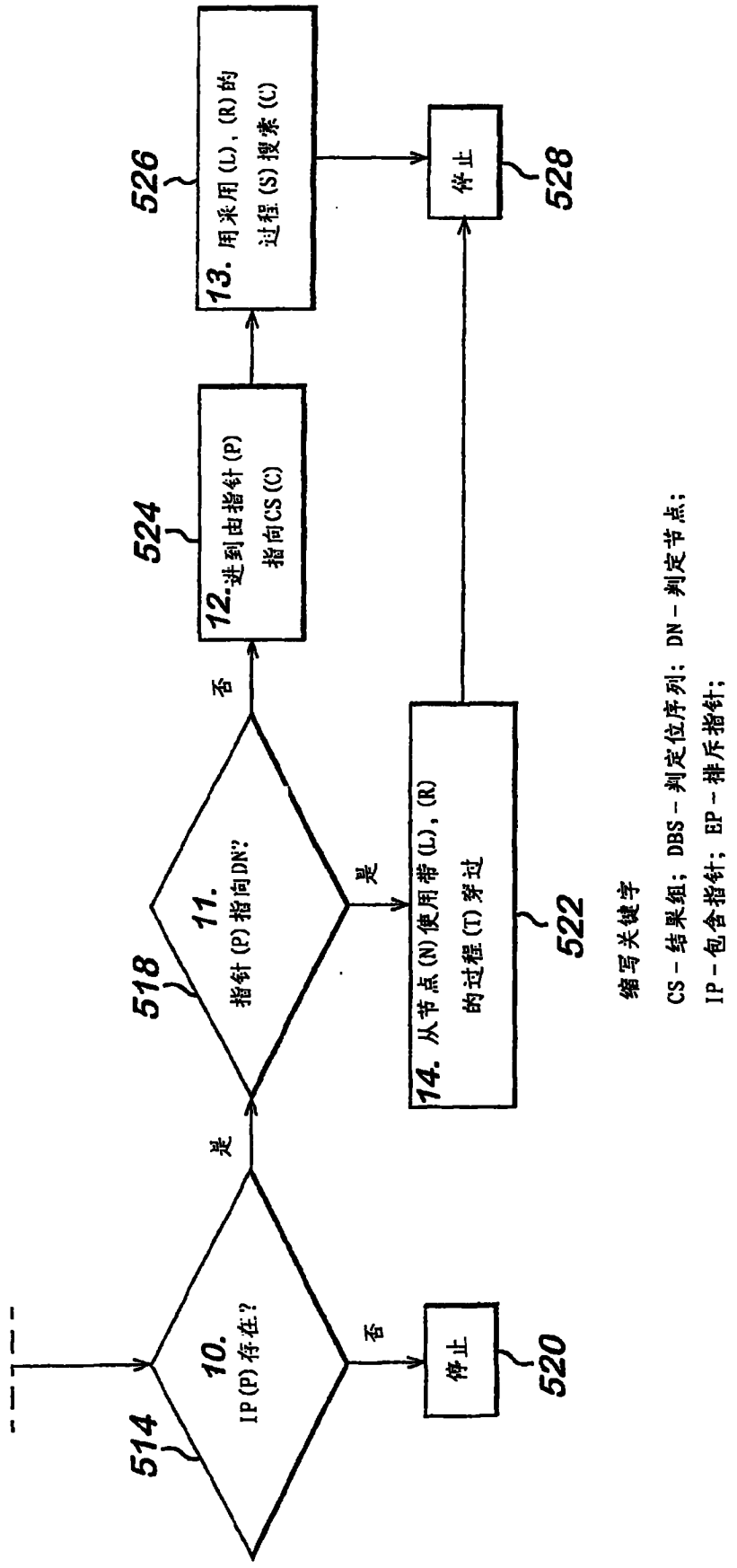

514
10. IP (P) 存在?
是
否
停止
520
518
11. 指针 (P) 指向DN'?
是
否
14. 从节点 (N) 使用带 (L), (R) 的过程 (T) 穿过
522
524
12. 进到由指针 (P) 指向CS (C)
526
13. 用来用 (L), (R) 的过程 (S) 搜索 (C)
停止
528
缩写关键字
CS - 结果组; DBS - 判定位序列; DN - 判定节点;
IP - 包含指针; EP - 排斥指针;

图 18(续)

搜索结果组（c）返回结果（R）中模式表（L）的过程（s）

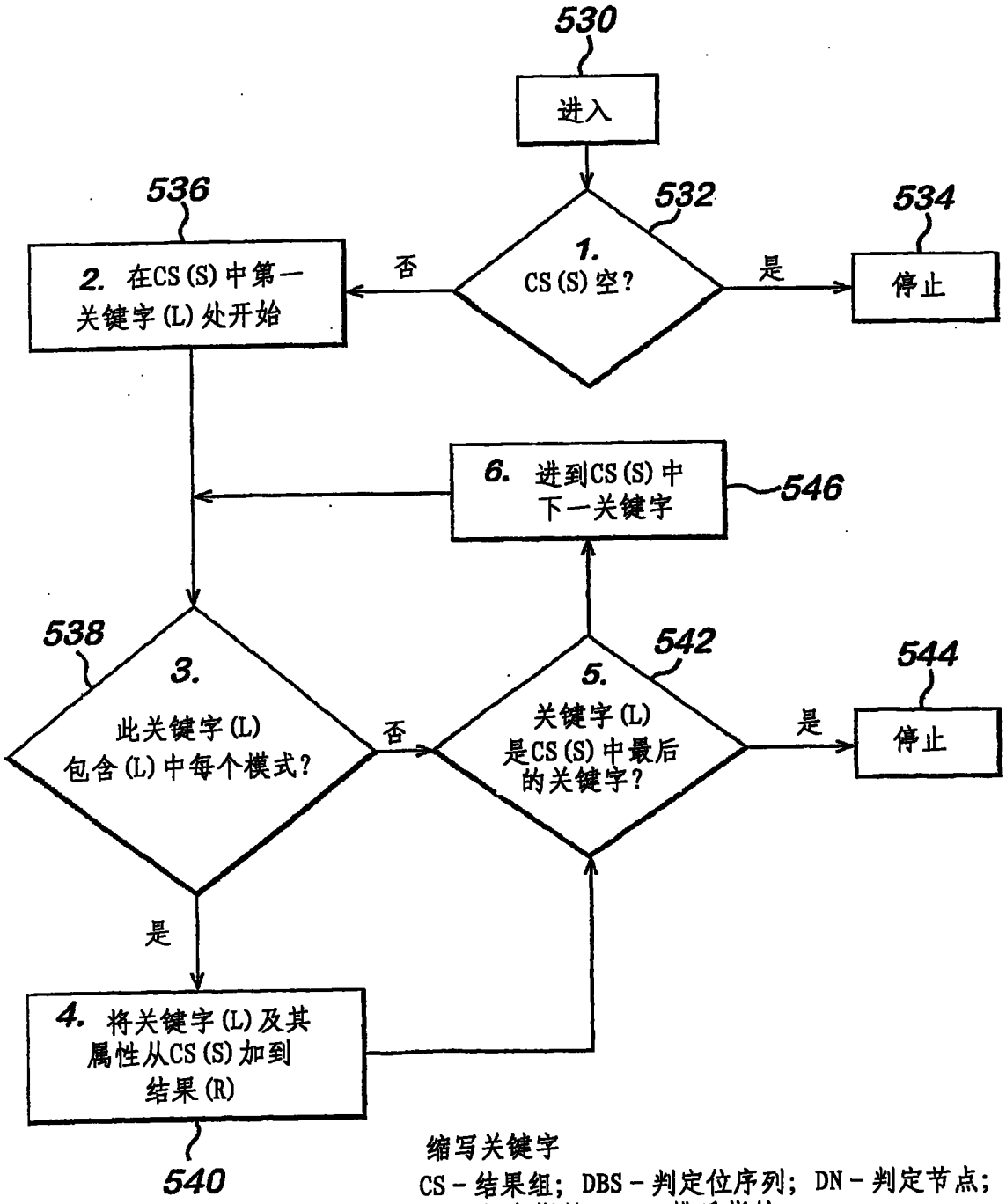


图 19

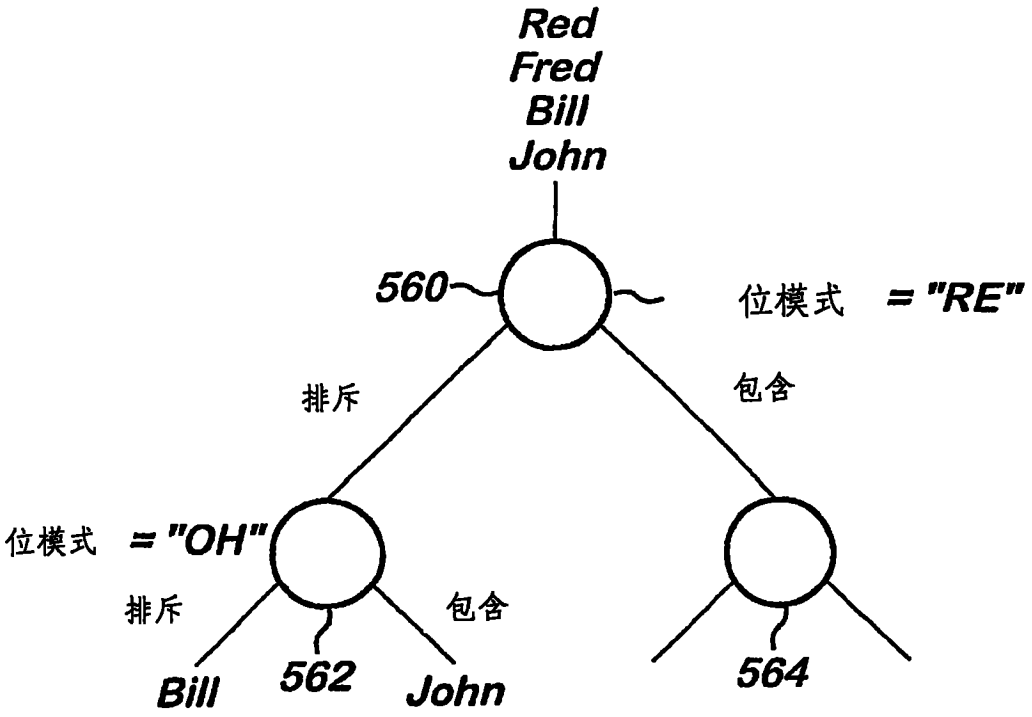

Red
Fred
Bill
John
560
位模式 = "RE"
排斥
包含
位模式 = "OH"
排斥
包含
Bill
562
John
564

图 20

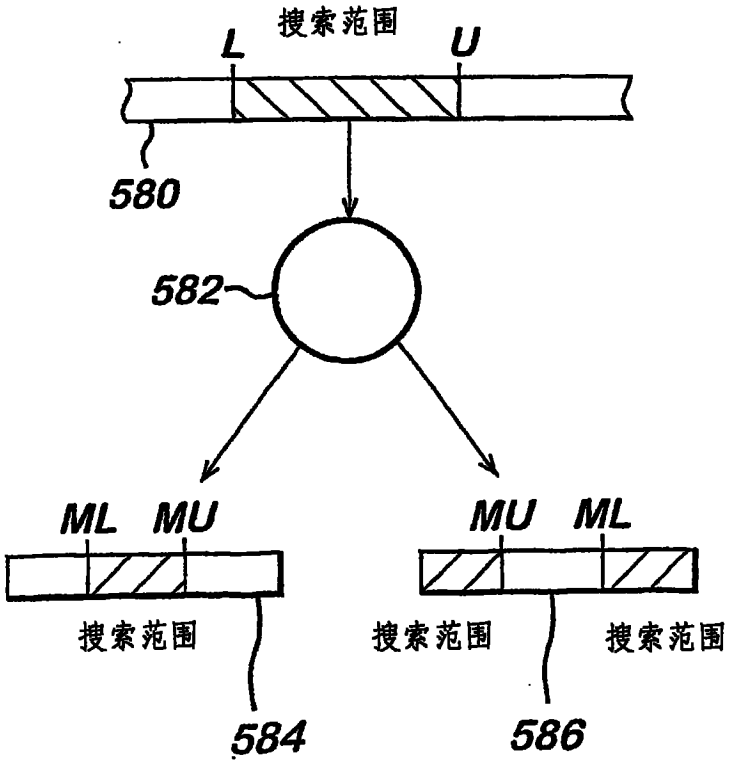

搜索范围
L
U
580
582
ML
MU
MU
ML
搜索范围
搜索范围
搜索范围
584
586

图 22

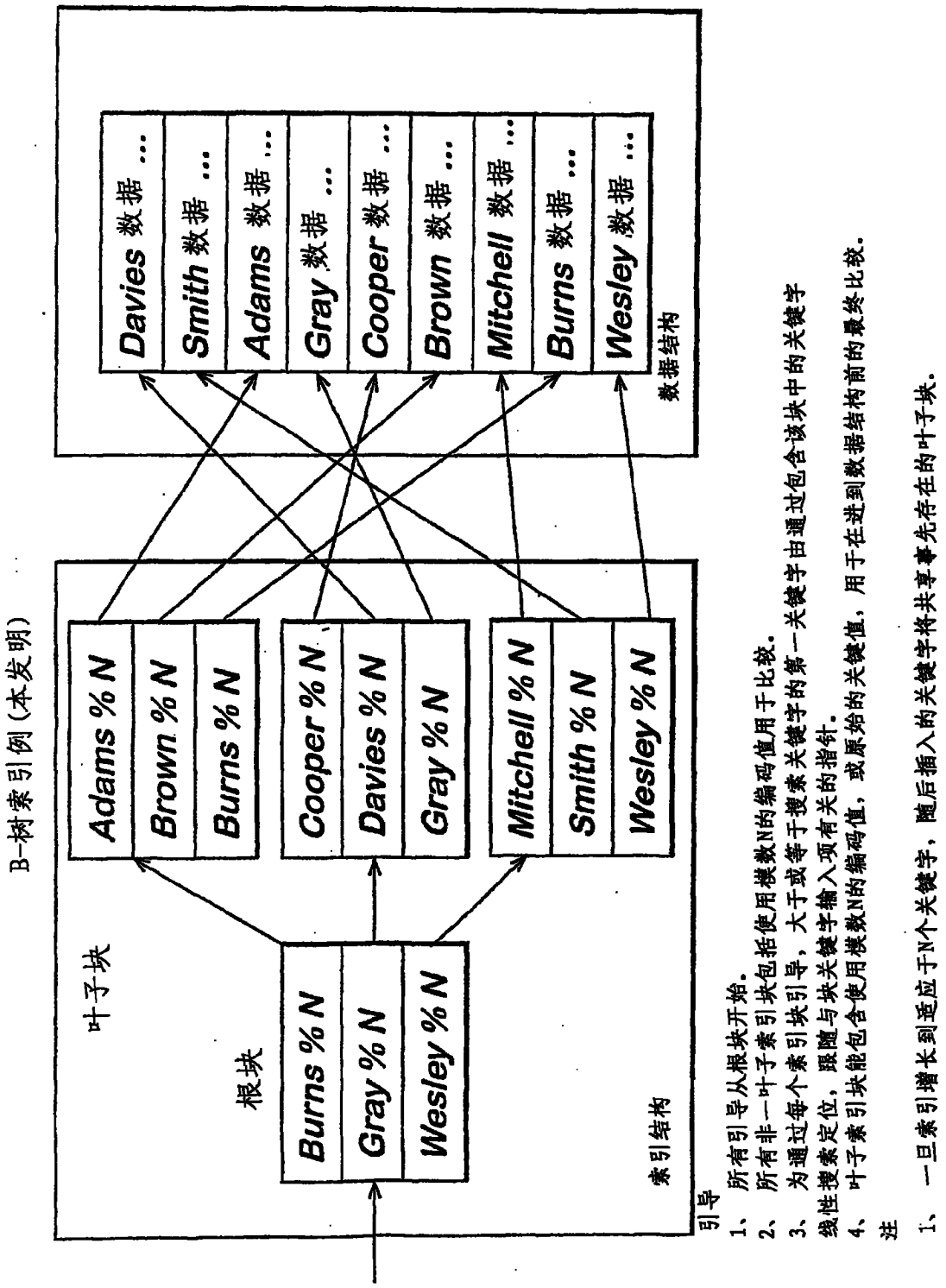
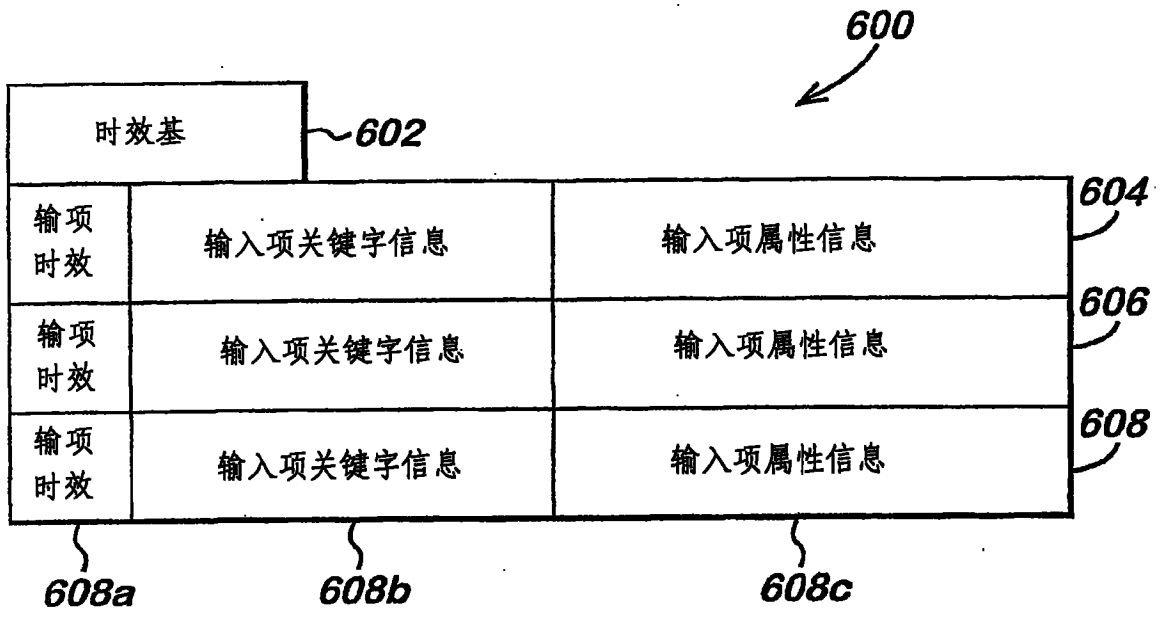

B-树索引例(本发明)
叶子块
根块
索引结构
Burns % N
Gray % N
Wesley % N
Adams % N
Brown % N
Burns % N
Cooper % N
Davies % N
Gray % N
Mitchell % N
Smith % N
Wesley % N
Davies 数据 ...
Smith 数据 ...
Adams 数据 ...
Gray 数据 ...
Cooper 数据 ...
Brown 数据 ...
Mitchell 数据 ...
Burns 数据 ...
Wesley 数据 ...
数据结构
引导
1、 所有引导从根块开始。
2、 所有非一叶子索引块包括使用模数N的编码值用于比较。
3、 为通过每个索引块引导，大于或等于搜索关键字的第一关键字由通过包含该块中的关键字线性搜索定位，跟随与块关键字输入项有关的指针。
4、 叶子索引块能包含使用模数N的编码值，或原始的关键值，用于在进到数据结构前的最终比较。
注
1、 一旦索引增长到适应于N个关键字，随后插入的关键字将共享事先存在的叶子块。

图 21



引导

1、 时效基记录最近插入或更新的日期/时间。

2、 数日向时效（0.255）以时效单位记录相对于时效基的输入项的时效，零值表示最近的输入项，255表示过时输入项，其空间能被新输入项再使用。

图 23

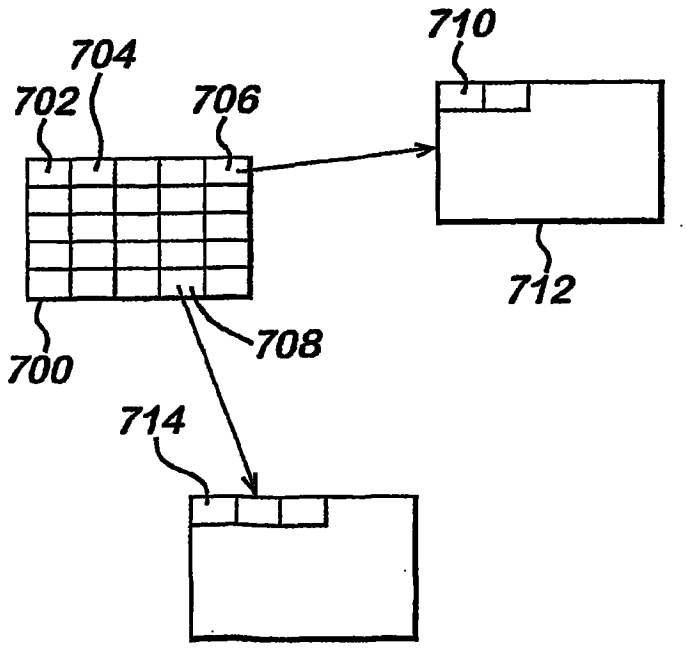


图 24

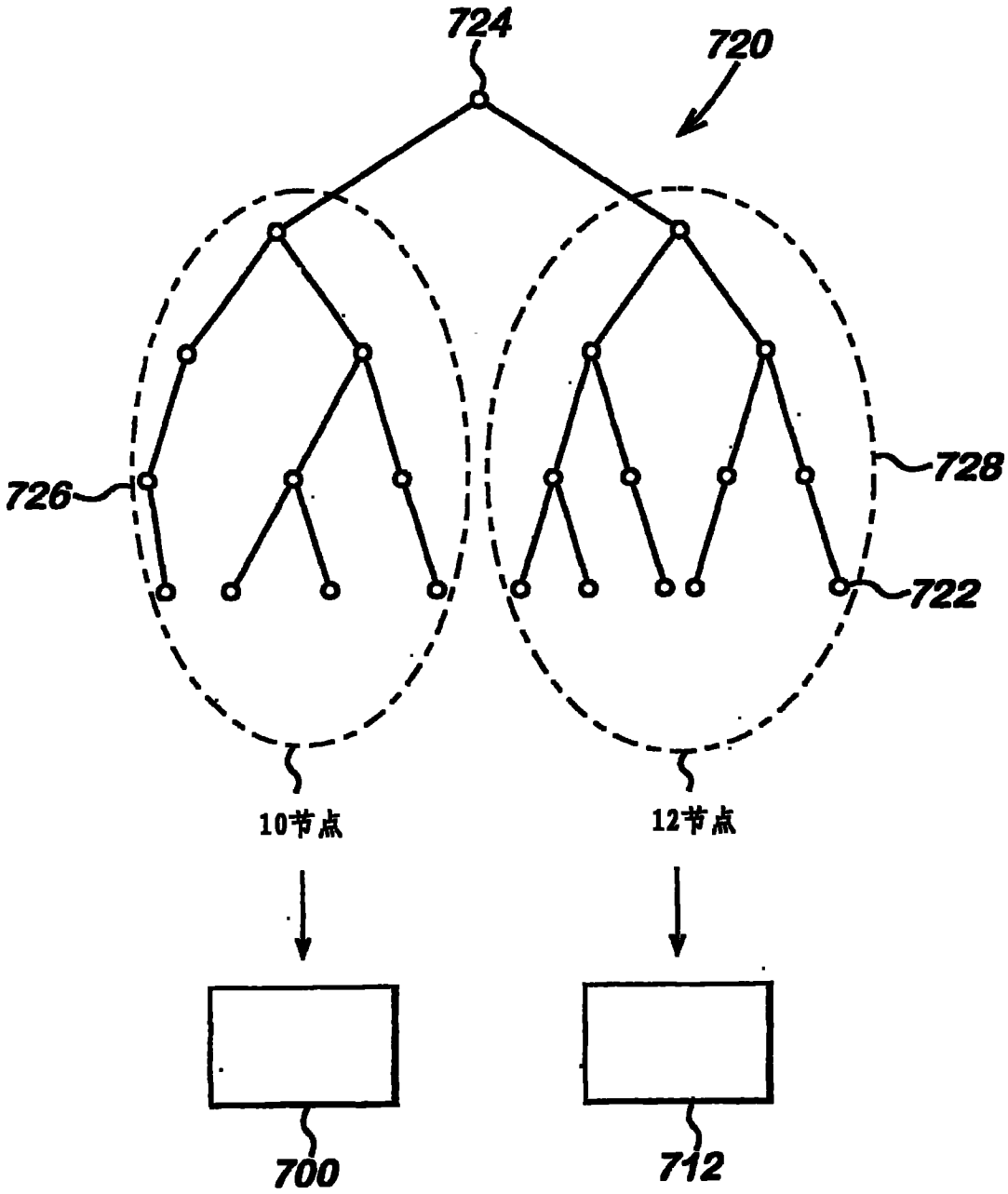


图 25

图 26