[19] 中华人民共和国国家知识产权局

[51] Int. Cl.
G06F 17/30 (2006.01)
G06Q 99/00 (2006.01)
G06F 9/46 (2006.01)
G06N 3/04 (2006.01)

[12] 发明专利说明书

专利号 ZL 00810570.7

[45] 授权公告日 2008 年 3 月 12 日

[11] 授权公告号 CN 100375088C

[22] 申请日 2000.6.19 [21] 申请号 00810570.7

[30] 优先权

[32] 1999.6.18 [33] US [31] 60/140,005

[32] 2000.2.29 [33] US [31] 60/185,665

[86] 国际申请 PCT/US2000/016839 2000.6.19

[87] 国际公布 WO2000/079415 英 2000.12.28

[85] 进入国家阶段日期 2002.1.18

[73] 专利权人 国际商业机器公司

地址 美国纽约

[72] 发明人 劳伦斯·A·布克曼

戴维·阿尔伯特·布莱尔

史蒂文·M·罗森撒尔

罗伯特·路易斯·克威泽

迈克尔·J·贝克利

杰瑞·李·考伦 阿伦·拉泽道

莎亚姆·R·默达穆比

[56] 参考文献

CN1193849A 1998.9.23

US5909681A 1999.6.1

FAULT TOLERANT DISTRIBUTED COMPUTING USING ATOMIC SEND – RECEIVECHECKPOINT. WOJCIK, Z. M.; WOJCIK, B. E. PROCEEDINGS OF THE SECOND IEEE SYMPOSIUM ON PARALLEL AND DISRIBUTED PRO. 1990

审查员 李 楠

[74] 专利代理机构 中国国际贸易促进委员会专利商标事务所

代理人 陈 炜

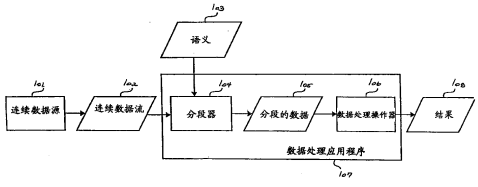
权利要求书 8 页 说明书 19 页 附图 7 页

[54] 发明名称

利用事务语义法分段和处理连续数据流

[57] 摘要

对于和事务相关的连续数据源而言，可以把数据分段并且以数据流程排列的方式处理数据，可选择的是以并行方式处理数据，可以不需要把数据存储在临时数据库就可以对数据进行处理。 可以以并行方式处理来自多个数据源的数据。 分段也可以定义一些点，在这些点可以提供聚合输出，进而在此处设立检查点。



1. 一种由计算机系统的处理元素在连续数据流上执行检查点操作的方法，包含:

接收事务语义的指示;

在数据上应用事务语义把连续数据流分割成段，由处理元素处理;

选择其中的一个段;

保存选择段的持久性指示;

由处理元素处理选择段，产生结果;

当检测到处理元素出现失败时，丢弃处理元素为选择段产生的结果并且根据保存的持久指示重新处理选择段;

当处理元素处理过程没有失败时，把处理元素产生的结果作为结果提供并且选择处理元素需要处理的下一个段。

2. 根据权利要求 1 的方法，其中应用步骤包括在连续数据流里插入数据指示数据段之间的边界。

3. 根据权利要求 1 的方法，其中连续数据流包括信息日志。

4. 根据权利要求 3 的方法，其中日志包括服务器日志。

5. 根据权利要求 3 的方法，其中日志包括 Web 服务器数据流。

6. 根据权利要求 3 的方法，其中日志包括预定系统的连续数据流。

7. 根据权利要求 3 的方法，其中日志包括销售终端系统的连续数据流。

8. 根据权利要求 3 的方法，其中日志包括自动柜员机的连续数据流。

9. 根据权利要求 3 的方法，其中日志包括帐务系统的连续数据流。

10. 根据权利要求 3 的方法，其中日志包括信用卡系统的连续数据流。

11. 根据权利要求 3 的方法，其中日志包括搜索引擎的连续数据

流。

12. 根据权利要求 3 的方法，其中日志包括视频系统的连续数据流。

13. 根据权利要求 3 的方法，其中日志包括视音频系统的连续数据流。

14. 根据权利要求 1 的方法，其中连续数据流包括多个纪录的多个连续数据流的第一连续数据流。

15. 根据权利要求 14 的方法，其中纪录包括一个或多个描述事务的字段。

16. 根据权利要求 15 的方法，其中纪录的格式包括可变的长度。

17. 根据权利要求 15 的方法，其中纪录的格式包括固定的长度。

18. 根据权利要求 15 的方法，其中纪录的格式包括标签过的纪录。

19. 根据权利要求 15 的方法，其中纪录的格式包括分界的纪录。

20. 根据权利要求 15 的方法，其中纪录的格式包括标记的语言。

21. 根据权利要求 20 的方法，其中标记的语言包括 SGML。

22. 根据权利要求 20 的方法，其中标记的语言包括 HTML。

23. 根据权利要求 20 的方法，其中标记的语言包括 XML。

24. 根据权利要求 15 的方法，其中纪录包括字符串。

25. 根据权利要求 15 的方法，其中纪录包括数组。

26. 根据权利要求 15 的方法，其中纪录包括存储在文件中的结构。

27. 根据权利要求 15 的方法，其中纪录包括数据库纪录。

28. 根据权利要求 15 的方法，其中纪录包括命名管道。

29. 根据权利要求 15 的方法，其中纪录包括网络数据包。

30. 根据权利要求 15 的方法，其中纪录包括帧。

31. 根据权利要求 15 的方法，其中纪录包括单元。

32. 根据权利要求 1 的方法，其中事务语义定义了连续数据流的字段的函数。

33. 根据权利要求 32 的方法，其中函数包括一段时间。

34. 根据权利要求 33 的方法，其中在一段时间内的连续数据流的所有数据都放在一个段里。

35. 根据权利要求 32 的方法，其中函数包括记录的集合函数。

36. 根据权利要求 35 的方法，其中纪录的集合函数包括数据总量。

37. 根据权利要求 36 的方法，其中数据包括销售数据。

38. 根据权利要求 32 的方法，其中函数包括商业规则信息。

39. 根据权利要求 32 的方法，其中函数包括系统需求。

40. 根据权利要求 1 的方法，其中连续数据流包括可识别的段。

41. 根据权利要求 40 的方法，其中分割连续数据流的步骤进一步包括根据可识别段分割连续数据流。

42. 根据权利要求 41 的方法，其中纪录插入到连续数据流里指示数据流中的两个段的边界。

43. 根据权利要求 42 的方法，其中纪录仅指示边界。

44. 根据权利要求 42 的方法，其中纪录是标签。

45. 根据权利要求 1 的方法，其中在数据上应用事务语义把连续数据流分割成段，由处理元素处理的步骤进一步包括接收应用于连续数据流将连续数据流分割成两段由处理元素处理的事务语义的第二指示。

46. 根据权利要求 45 的方法，其中两段由第二处理元素处理。

47. 根据权利要求 1 的方法，其中处理选择段包括集合数据。

48. 根据权利要求 47 的方法，其中集合数据包括纪录累计。

49. 根据权利要求 47 的方法，其中集合数据包括纪录中的变量之和。

50. 根据权利要求 47 的方法，其中集合数据包括统计操作。

51. 根据权利要求 1 的方法，其中处理选择段的步骤包括建立多个用户。

52. 根据权利要求 1 的方法，其中处理元素包括并行处理元素。

53. 根据权利要求 52 的方法，其中将连续数据流分割成段的步骤进一步包括将连续数据流分割成多个区由并行处理元素处理。

54. 根据权利要求 52 的方法，其中每个并行处理元素为各分区提供临时结果。

55. 一种在计算机系统里的连续数据流上执行检查点操作的计算机系统，包括：

接收事务语义指示的装置；

在连续数据流上应用事务语义，把数据分割成段的装置；

选择其中的一个段的装置；

保存选择段的持久指示的装置；

处理选择段产生结果的处理元素；

在检测到处理元素失败之后，丢弃该处理元素为选择段产生的任何结果的有效装置，指导处理元素根据保存的持久指示重新处理选择段；

在处理元素正常结束处理之后，提供由处理元素产生的结果并且选择处理元素需要处理的下一个段的有效装置。

56. 根据权利要求 55 的计算机系统，其中应用的装置包括在连续数据流里插入数据以指示数据段之间的边界。

57. 一种在连续数据流上执行检查点操作的计算机系统，包括：

处理器；

与处理器相连的接收器，接收事务语义指示；

其中，处理器在连续数据流上应用事务语义，把数据分割成段；选择其中的一个段；保存选择段的持久指示；处理选择段产生结果；

在检测到处理元素失败之后，处理器丢弃该处理元素为选择段产生的任何结果的有效装置，指导处理元素根据保存的持久指示重新处理选择段；

在处理元素正常结束处理之后，处理器提供由处理元素产生的结果并且选择处理器需要处理的下一个段。

58. 根据权利要求 57 的计算机系统，其中在应用事务语义时，处

理器在连续数据流里插入数据指示数据段之间的边界。

59. 根据权利要求 57 的计算机系统，其中连续数据流包括信息日志。

60. 根据权利要求 59 的计算机系统，其中日志包括服务器日志。

61. 根据权利要求 59 的计算机系统，其中日志包括 Web 服务器数据流。

62. 根据权利要求 59 的计算机系统，其中日志包括预定系统的连续数据流。

63. 根据权利要求 59 的计算机系统，其中日志包括销售终端系统的连续数据流。

64. 根据权利要求 59 的计算机系统，其中日志包括自动柜员机的连续数据流。

65. 根据权利要求 59 的计算机系统，其中日志包括帐务系统的连续数据流。

66. 根据权利要求 59 的计算机系统，其中日志包括信用卡系统的连续数据流。

67. 根据权利要求 59 的计算机系统，其中日志包括搜索引擎的连续数据流。

68. 根据权利要求 59 的计算机系统，其中日志包括视频系统的连续数据流。

69. 根据权利要求 59 的计算机系统，其中日志包括视音频系统的连续数据流。

70. 根据权利要求 57 的计算机系统，其中连续数据流包括多个纪录的多个连续数据流的第一连续数据流。

71. 根据权利要求 70 的计算机系统，其中纪录包括一个或多个描述事务的字段。

72. 根据权利要求 71 的计算机系统，其中纪录的格式包括可变的长度。

73. 根据权利要求 71 的计算机系统，其中纪录的格式包括固定的

长度。

74. 根据权利要求 71 的计算机系统，其中纪录的格式包括标签过的纪录。

75. 根据权利要求 71 的计算机系统，其中纪录的格式包括分界的纪录。

76. 根据权利要求 71 的计算机系统，其中纪录的格式包括标记的语言。

77. 根据权利要求 76 的计算机系统，其中标记的语言包括 SGML。

78. 根据权利要求 76 的计算机系统，其中标记的语言包括 HTML。

79. 根据权利要求 76 的计算机系统，其中标记的语言包括 XML。

80. 根据权利要求 71 的计算机系统，其中纪录包括字符串。

81. 根据权利要求 71 的计算机系统，其中纪录包括数组。

82. 根据权利要求 71 的计算机系统，其中纪录包括存储在文件中的结构。

83. 根据权利要求 71 的计算机系统，其中纪录包括数据库纪录。

84. 根据权利要求 71 的计算机系统，其中纪录包括命名管道。

85. 根据权利要求 71 的计算机系统，其中纪录包括网络数据包。

86. 根据权利要求 71 的计算机系统，其中纪录包括帧。

87. 根据权利要求 71 的计算机系统，其中纪录包括单元。

88. 根据权利要求 57 的计算机系统，其中事务语义定义了连续数据流的字段的函数。

89. 根据权利要求 88 的计算机系统，其中函数包括一段时间。

90. 根据权利要求 89 的计算机系统，其中在一段时间内的连续数据流的所有数据都放在一个段里。

91. 根据权利要求 88 的计算机系统，其中函数包括记录的集合函数。

92. 根据权利要求 91 的计算机系统，其中纪录的集合函数包括数

据总量。

93. 根据权利要求 92 的计算机系统，其中数据包括销售数据。

94. 根据权利要求 88 的计算机系统，其中函数包括商业规则信息。

95. 根据权利要求 88 的计算机系统，其中函数包括系统需求。

96. 根据权利要求 57 的计算机系统，其中连续数据流包括可识别的段。

97. 根据权利要求 97 的计算机系统，其中分割连续数据流的步骤进一步包括根据可识别段分割连续数据流。

98. 根据权利要求 97 的计算机系统，其中纪录插入到连续数据流里指示数据流中的两个段的边界。

99. 根据权利要 98 的计算机系统，其中纪录仅指示边界。

100. 根据权利要求 98 的计算机系统，其中纪录是标签。

101. 根据权利要求 57 的计算机系统，其中处理器接收应用于连续数据流将连续数据流分割成两段由处理器处理的事务语义的第二指示。

102. 根据权利要求 101 的计算机系统，其中两段由第二处理器处理。

103. 根据权利要求 57 的计算机系统，其中处理器进一步通过集合数据处理选择段。

104. 根据权利要求 103 的计算机系统，其中集合数据包括纪录累计。

105. 根据权利要求 103 的计算机系统，其中集合数据包括纪录中的变量之和。

106.根据权利要求 103 的计算机系统，其中集合数据包括统计操作。

107. 根据权利要求 57 的计算机系统，其中处理器进一步处理选择段以产生包括建立多个用户的结果。

108. 根据权利要求 57 的计算机系统，其中处理器包括并行处理

器。

109. 根据权利要求 108 的计算机系统，其中将连续数据流分割成段包括将连续数据流分割成多个区由并行处理器处理。

110. 根据权利要求 108 的计算机系统，其中每个并行处理器为各分区提供临时结果。

利用事务语义法分段和处理连续数据流

相关申请

本申请要求共同待审的美国临时申请序号60/140,005的依据35 U.S.C 119(e)的权益，该申请1998年6月18日提交，标题为“利用事务语义法分段和处理连续数据流”，作者Lawrence A. Bookman等人，它的内容在此被引用。本申请也要求共同待审的美国临时申请序号60/185,665的依据35 U.S.C 119(e)的权益，该申请2000年2月2日提交，标题为“利用事务语义法分段和处理连续数据流”，作者Lawrence A. Bookman等人，它的内容在此被引用。

本发明的背景

基于计算机的事务系统产生与事务相关的数据，事务由这些系统执行。这些与事务相关的数据被分析以便于标示事务的特性。从这些特性中，可能暗示对这些事务和/或相关市场数据的修改，或者可能得出其它的商务决策。

为了分析与事务相关的数据，计算机系统通常访问存储在数据库里的数据。在经过一段时间的数据收集之后，这些收集到的数据以单个事务的方式写入数据库中。正如上面提到的，经过分析存储在数据库里的数据后产生相应的结果。从分析获得的结果典型地代表了数据库中的数据的变化。利用这些结果，例如，作为各种商务决策的基础；通常，这些结果也存储在数据库中。

在某些情况下，与事务相关的原始数据在经过处理之后并不保留在数据库中。这种处理与事务相关的数据的方式通常是批处理的形式。在批处理中，只有当所有的数据处理完之后结果才被输出。例如，如果与批量相关的每一个纪录以单独的事务存储在数据库中，那么该数据库的数据库管理系统将出现巨大的超负载情况。同样地，以单个事务的方式从数据库中读取大量的数据来允许对这些数据进行分析。在很多情况下，事务的发生以及利用有关该事务的数据产生结果之间的时间可能是许多天，甚至是许多周。

本发明的简述

如果事务系统连续产生与事务相关的数据，或者如果接收分析结果的期望时间少于需要执行批处理的时间，那么这样的批处理技术是不能使用的。通常不期望延迟获得分析结果，因为这些事务的用户行为可能频繁地发生变化。例如，在实时跟踪具有频繁变化的系统访问信息的数据库系统里，因为安全或性能原因，可能不能接受访问分析时有时无的情况。

给定与事务相关的连续数据源，这些事务数据可以被分段并且以数据流组织的形式被处理，可以选择并行方式，而且不需要把数据存储在临时数据库里就可以处理数据。因为数据分段而且单独操作，来自多个地方的数据可以被并行处理。分段也可以定义提供集合输出的点，在这里可以建立检查点。通过分割数据为段并且在分段之上定义检查点，在每一个定义的检查点处可以重新启动处理进程。在这种方式里，数据处理可能在特定的段上失败但不影响其它段的处理。因此，如果特定段上的数据处理失败，与此段相应的工作丢失，但是其它段上执行的工作不会丢失。这个检查点，例如，可以在关系数据库里实现。检查点使得关系数据库系统实现查询重递，由此数据库的性能得到增加。这对于依赖于系统性能获得成功

的数据库开发商和用户都是有好处的。总之，如果可以分割数据流，那么就可以实行检查点处理和恢复。

下面给出这些和其它的优势。

根据一个方面，提供了一种处理连续数据流的方法。这个方法包含的步骤有接收事务语义的指示，在连续数据流上应用事务语义以标示该连续数据流的段，处理该连续数据流的每一个段上的数据以便为段产生结果，并且在该连续数据流的每一个段上的数据经过处理之后，提供为该段产生的结果。

根据一个实施方案，数据包括一组纪录，每个纪录包括一组字段，事务语义是由该数据的一个或多个纪录的一个或多个字段的函数定义的。根据另外一个实施方案，本方法进一步包含根据标示的段分割连续数据流的步骤。根据另外一个实施方案，分割的步骤包括在里连续数据流里插入指示两个段边界的纪录。根据另外一个实施方案，此纪录是仅仅指示边界的标志纪录。根据另外一个实施方案，此纪录是包括与事务语义相关信息的语义纪录。

根据另外一个实施方案，连续数据流是有关发送给服务器的请求的信息日志，应用的步骤包含从日志中读取与请求相关的信息；之后把事务语义应用到读取的信息上。根据另外一个实施方案，与每一个请求相关的信息包括一些字段，在此事务语义是由与一个或多个请求相关的信息的一个或多个字段的函数定义的。根据另外一个实施方案，此信息包括请求发送给服务器的时间，在此事务语义定义时间段。根据另外一个实施方案，本方法进一步包含过滤日志来排除与一个或多个请求相关信息的步骤。根据另外一个实施方案，过滤的步骤是在应用事务语义步骤之前执行的。根据另外一个实施方案，过滤步骤包括排除与蜘蛛关联的请求信息。根据另外一

个实施方案，本方法进一步包含过滤连续数据流以排除连续数据流中的数据的步骤。

根据另外一个实施方案，本方法进一步包含额外步骤，那就是处理连续数据流中的每个段的数据为段产生结果，并且在此额外处理步骤之中处理连续数据流中的数据之后，提供为段产生的结果。根据另外一个实施方案，处理步骤包含的步骤有：把段里的数据分割成一组并行分区；并行处理每个分区以提供每个分区的临时结果。根据另外一个实施方案，本方法进一步包含结合每个分区的临时结果为段产生结果。根据另外一个实施方案，连续数据流中的数据是有序的，而且有多个连续数据流的来源；本方法进一步包括判定连续数据流中的数据是否有序；如果判定处数据是无序的，中断处理步骤，根据事务语义在段里插入该数据，重新处理该段然后继续处理步骤。根据另外一个实施方案，发明进一步包含保存正在处理的数据的段的持久性指示；当检测到处理步骤失败时，放弃此选择段的处理步骤产生的结果并且根据保存的持久性指示重新处理选择段；当处理步骤成功结束时，提供产生的输入并且选择下一个段。

根据另外一个方面，提供了一种由计算机系统的处理元素在连续数据流上执行检查点操作的进程。该进程包含的步骤有：接收事务语义的指示；在数据上应用事务语义把连续数据流分割成段供处理元素来处理；选择其中一个段；保存选择的段的持久性指示。由处理元素处理选择段产生结果；当检测到处理元素的失败时，放弃处理元素为选择段产生的任何结果，根据保存的持久性指示重新处理选择的段；当处理元素处理过程成功时，提供由处理元素作为输出产生的输出，选择处理元素要处理的下一个段。根据另外一个实施方案，应用的步骤包括在连续数据流中插入数据指示数据段之间的边界。

根据另外一个方面，提供一种计算机系统来执行计算机系统里的连续数据流上的检查点操作。该计算机系统包含接收事务语义指示的装置；在连续数据流上应用事务语义把数据分割成段的装置；选择其中一个段的装置；保存选择段的持久性指示；用来处理选择段产生结果的处理元素；在检测到处理元素失败之后放弃处理元素为选择段产生的任何输出的装置；根据保存的持久性指示指导处理元素重新处理选择段的装置；在处理元素成功结束处理之后，提供处理元素产生的结果并且选择处理元素要处理的下一个段的装置。根据另外一个实施方案，应用的装置包括在连续数据流中插入数据指示数据段之间的边界。

根据另外一个方面，提供了一种处理连续数据流的方法。这个方法包含的步骤有接收事务语义的指示；在连续数据流上应用事务语义以标示该连续数据流的段；在连续数据流里插入数据指示连续数据流的标示段之间的边界。

下面引用所附上的图表详细描述了本发明的更多特征和优势以及本发明的各种实施方案的结构和操作。在图表中，相近的引用数字指示相近或功能上相似的元素。除此之外，引用数字的最左边的一个或两个数字标示该引用数字第一次出现的图表。

本发明图表的简要描述

在图表中，

图 1 是数据流程图，该图是根据本发明的一个实施方案表现系统处理连续数据。

图 2 是描述数据如何从连续数据源引入并行应用程序框架的操作。

图 3 是表现系统处理多个数据流的可选数据流程图。

图 4 是描述数据如何被多个管道系统处理的流程图。

图 5 是适合于实现本发明的各种实施方案的客户-服务器系统的框架图。

图 6 是处理数据的处理过程结构的框架图。

图 7 是以并行方式通讯的具有操作符的双节点系统的框架图。

详细描述

下面的详细描述应该和所附的图表一起阅读，图表中相似的引用数字指示相似的结构。所有在这里引证的参考文献都参考并入本篇。

图 1 中，连续数据源 101 提供连续数据流 102，该数据流是由数据处理应用程序 107 根据一些事务语义 103 处理后提供结果 108。这些事务语义 103 可能是决定流 102 如何被分段的信息。语义 103，例如，可能取决于系统操作流 102 的一些需求或者取决于分析数据的商务需求。在数据处理应用程序 107 中，节段器 104 根据事务语义 103 把数据分段提供分段数据 105。数据处理操作者 106 处理分段数据 105 中的每一个段里的数据来产生每一个段的结果 108。这些处理可能是，例如，读取或更新连续数据流 102 中的一个或多个数据部分。

连续数据源 101 一般从事务系统中提供与事务相关的数据。数据源是连续的，因为事务系统通常在一段时间内完成操作以便允许用户进行事务操作。例如，连续数据源可能是 WEB 服务器，它输出关于发送给此 WEB 服务器的请求的信息日志。这些请求可能被

WEB服务器当作日志纪录存储在服务器日志中。连续数据流源的其它例子包括来自预定系统的有关事务的数据源，销售终端系统，自动柜员机，帐务系统，信用卡系统，搜索引擎，视频或音频分布式系统，或产生连续数据流的其它形式的系统。也可能有一个或多个连续数据源提供一个或多个连续数据流，应用程序 107 可能会被配置成在这些数据流之上进行操作。

与事务相关的数据一般包括对应每个事务的纪录，该纪录包括一个或多个描述该事物的信息字段。该字段可能是几种不同格式的任何一种。与事务相关的数据，例如，可能具有可变或固定长度，可能是被标签过或未被标签过，也可能被分界或未被分界。与事务相关的数据可能是，例如，以标记语言格式诸如 SGML，HTML，XML，或其它标记语言的格式存在。有关数据从连续数据源 101 传送到数据处理引用程序 107 的样例构成包括字符型串，数组或其它存储在文件里的结构，数据库纪录，命名管道，网络数据包，帧，单元，或者其它格式。根据一个方面，连续数据流 101 是服务器日志，而且与事务相关的样例数据可能包括用户标示符，客户程序以及/或者系统标示符，时间戳，页面或广告标示符，指示页面或广告如何被存取的指示器，纪录类型，以及/或者其它涉及事务的信息。

事务语义 103 定义了连续数据流 102 的一个或多个纪录的一个或多个字段的函数。例如，事务语义可能定义一段时间，例如，一小时，这样在一小时之内的所有数据被放在一个段里。事务语义 103 也可能定义几个纪录的集合函数，例如，销售总额，而不是单一纪录的函数，诸如时间。这样的事务语义 103 也可能派生于指示从数据分析获得信息的商业规则。事务语义 103 也可能取决于一些系统需求。这种分析可能被执行，例如，基于每个段以便使能商业决策。

事务语义 103 由节段器 104 应用到连续数据流 102 上来标示连续数据流 102 上的段。连续数据流 102 可能被以许多方式根据这些标示段被分区。例如，插入到连续数据流 102 里纪录指示数据流中两个段的边界。该纪录可能是仅仅指示边界的标记。例如，所有纪录中都被放置了标签，结果时标志纪录有此标签的一个值，而数据纪录有此标签的另外一个值。可选的是，该纪录可能是包含与事务语义相关的信息，诸如事务语义自身或者是通过在数据上应用事务语义派生的信息，诸如指定时间段。而且，应用程序 107 可能允许多个数据处理操作器 106 根据存储在数据中的事务语义访问数据段。任何一种类型的信息都可以被用于指示数据流 102 里的分区。

多个分段器 104 也可以被用于产生不同段的连续数据流 105，有关该数据流 105 的处理执行方式可以不同。可选的是，多个数据处理操作器 106 可以以并行方式在分段连续数据流 105 上执行不同的分析。

数据处理操作器 106 可以执行许多种操作，例如，为每个数据段计算数据集合诸如纪录累计，纪录中的变量之和，以及统计值诸如平均数，各种数据字段的最大值和最小值。在一个连续数据流是服务器日志的应用程序中，计算用户的唯一编号是可能的，例如，服务器提供给这些用户在每一个段或段的组合里的每个信息项目。各种数据处理操作器 106 可能加入到数据处理应用程序 107 中或被从中删除以便提供各种不同的结果 108。

数据处理应用程序 107 可以采用来自 Torrent 系统公司的 Orchestrate 并行结构来实现，这种技术在以下专利申请中有详细描述：美国专利申请序列号 08/627，801，1996 年 3 月 25 日提交，标题为“可编程并行计算机的设备和方法“，作者是 Michael J. Beckerle 等；美国专利申请序列号 08/807，040，1997 年 2 月 24 日提交，标

题为“监控并行计算性能的设备和方法”，作者是 M. Razdow,等；美国专利申请序列号 09/104,288，1998 年 6 月 24 日提交，标题为“在计算机系统里通过分割数据在数据上进行检查点操作的计算机系统和处理”，作者是 Michael J. Beckerle；美国专利号 5,909,681，1999 年 6 月 1 日发布，标题为“并行处理分割数据的计算机系统和计算机化的方法”，作者是 Anthony Passera，等。

在此系统中，并行数据源在多个操作器上以数据流程安排的方式被执行。特别地，在图表 1 中执行的每个操作，诸如分段或数据分析，可以作为操作器在 Orchestrate 并行处理结构中被执行。使用并行应用结构，由数据处理操作器处理的数据被分割成许多并行分区。这些并行分区的每一个分区都由不同的数据处理操作器实以并行方式处理，每个操作器为各自的分区提供临时结果。执行集合函数的操作器可能把这些临时结果进行组合以便提供集合段的结果。

再者,使用 Orchestrate 并行处理结构处理并行数据流，配置各种操作器来处理这些并行数据流，并且使用多项输入操作器组合两个数据流形成单一数据流。这个单一数据流可能被各种操作器操作，存储和传输，也可能在此数据流上执行其它操作。

数据处理操作器 106 被实现的方式很多。特别地，数据处理操作符 106 通常以批处理模式或连续模式处理数据。如果数据处理操作器 106 执行批处理，它直到所有与批处理条目相关的数据被处理之后才输出数据。操作器 106 可能被执行连续循环的程序控制，该循环在单个段的基础上给操作器提供数据。此程序向操作器标示数据已经到达每个段的边界，因此，这引起操作器 106 为段输入结果。可选的是，包括引起操作器 106 在每一个段的边界上输入接口的连续操作器可能被使用。

以各种形式的分段连续数据流 105 也可能被存储在 Orchestrate 并行数据集里。并行数据集通常包括名称，指向数据实际被存储的位置，模型以及元数据（描述数据的数据），元数据是指申明信息诸如硬件、磁盘、数据处理单元等的配置信息，指示数据存储的地方。一个数据集可能被用来标示多个段，或者每一个段使用独立的数据集。

如果象 Orchestrate 并行应用程序结构之类的系统被用于数据处理应用，那么可以从存储方式中把连续数据流 102 引入到此应用程序结构的数据集合里，连续数据源 101 在此存储方式中产生连续数据流 102。例如，连续数据源 101 可以是根据接收到的请求产生数据的 HTTPD 服务器，而且该服务器把这些数据保存在日志中。通常被称为日志管理器的单独应用程序定期创建 HTTPD 用于存储数据的日志文件。

例如，每日可以创建新的日志文件。有关日志管理器如何创建日志文件的信息被提供给象引入操作器之类的数据处理操作器 106，这样的引入操作器把日志文件集合作为连续数据流读取 Orchestrate 应用程序结构的数据集合里。可能有一个或多个并行操作日志文件的引入操作器，或者有同一个操作器的一个或多个实体在并行处理。日志文件也可能有许多来源，引入操作器的多个实例在并行处理这些日志文件。例如，多个 HTTPD 服务器可能并行写同一个日志文件。也就是说，多个 HTTPD 进程产生由一个或多个输入操作器处理的并行数据流。多个输入操作器可能被用于把这些数据流结合到单个数据流中，其它的操作器可能在此之上进行操作。

现在结合图 2 来描述由数据处理应用程序 107 执行的引入进程 200 的操作流程。引入进程 200 依赖于步骤 201 里接收到的数据标

示信息。此标示信息标示数据文件、命名管道或其它由连续数据源102使用的构造的命名规则。之后，在步骤202里根据接收到的源标示信息选择命名构造。接着，在步骤203里从这个命名构造中读取任何以后的数据纪录。如果此命名构造中包含标示信息，可能执行验证步骤来验证访问的是正确的命名构造。如果步骤203执行的操作返回数据，根据步骤204的判定，数据提供给步骤208的下一个操作器。下一个操作器可能是过滤操作器，把数据纪录变换成另外一种适合分段和处理的格式，或者可能是分段器。处理过程通过在步骤203里读取更多的数据。以这种方式，引入器不间断地从指定的连续数据源读取数据，在连续数据源和数据处理应用程序之间提供缓冲。

如果在执行读操作的时候没有可用数据，根据步骤204的判定，在步骤205首先判定服务器是否在运行。如果在步骤205里服务器没有运行，系统在步骤209里等待并且在等待之后在步骤203尝试再此读数据。等待的时间段可能是，例如，随机数，预先决定的数，或者它们的结合。如果服务器在运行并且还没有到达文件的尾部，根据步骤206的判定，事务系统可能被认为运行正常，而且还没有被用于产生与事务相关的数据。在步骤206之后，引入器进程200可能等待一段时间并且/或者可能给下一个操作器发送伪纪录，正如在步骤210里指示的那样，在步骤203里尝试再次读取数据之前。如果到达文件尾部，根据步骤206的判定，在步骤207里根据源标示信息选择下一个文件（或其它命名构造），在这之后，处理过程返回步骤203。这里的进程200可能被设计成没有中断地操作以便连续提供数据给数据处理应用程序107。

连续数据流102的分段也提供设备，通过此设备执行检查点操作。特别地，操作器106可能保存正在被处理的持久分段指示。当检测到操作器106执行处理的过程出现失败时，任何由操作器106

为选择的段产生的结果可能被丢弃。此段之后可能被重新处理使用保存的正在处理的段的持久指示。如果操作器 106 正常地结束处理过程，操作器 106 产生的输出可能在下一个段被处理之前输出。这种使用分段来进行检查点操作使得在连续数据流上的操作通过使用事务语义被实施检查点，该事务语义把连续数据流分割为段。分段可以被用于定义检查点，该检查点的执行行为在“利用数据流并行法装载数据库”，Sigmod Record，第 23 卷，4 号，72-83 页，1994 年 12 月，以及在美国专利申请序列号 09/104，288，1998 年 6 月 24 日提交，还有题为“计算机系统和在计算机系统内分割数据处理数据上的检查点操作”，作者是 Michael J.Beckerle。也可能使用不基于事务语义的分段，而是采用不同的分割来执行检查点。

在 Orchestrate 应用程序结构中，在上面结合图 2 一起描述的引入操作和分段器可能被作为复合操作器实现，使得从连续数据流的引入到结果的输出启用对整个数据处理应用程序的检查点。引入处理的检查点也可能根据事务语义被执行。例如，如果使用了时间字段，整个步骤可以被定期进行检查点处理，诸如 1 小时，30 分钟等。

在一些应用程序中，连续数据源可能遇到中断的情况，例如由于失败或其它原因，因而可能没有按照期望的顺序提供的数据。在一些应用程序中，无序的数据会被丢弃掉。然而，在某些分析中，无序的数据可能是有用的。在这些应用程序里，无序的数据被标示并且插入到合适的段里，这个段又被重新处理。无序数据是可以检测到的，例如，通过监控连续数据源 101 的状态。当数据源 101 在之前一直不可用之后变得可用时，就会中断对于其它段的处理，进而处理来自刚刚可用的数据源的无序数据。接着，把来自这个连续数源的数据附加到所属的数据集合的最后。在结束的时候，系统的连续操作重新启动。这种中断和数据的重新处理同样也可以应用在检查点操作上。

如上所述，可以配置数据处理应用程序 107 以并行方式处理多个连续数据流 102。图 3 描述了数据处理应用程序 308，在功能上和数据处理应用程序 107 相似，该应用并行接收来自许多不同数据源 302-304 的连续数据流 305-307。通过配置数据应用程序 308 在这些单个的流 305-307 上进行操作并且产生一个或多个结果 310。特别地，结果 310 可能是，例如，合并的数据流作为输入流 305-307 的函数。特别地，结果 310 可能是存储在数据库里的实时纪录流。根据一个实施方案，数据库是关系数据库，并且该关系数据库有能力执行并行访问数据库里的纪录。

在图 3 中描述的系统 301 是一个处理多个并行数据源的样例系统。特别地，这些数据源可以是产生日志文件数据流的 HTTPD 服务器。如果没有这样的结构 301，必须从多个数据源获得日志文件信息，然后以串行方式处理，或者多个进程必须独立处理单独的数据流。前一种情况下，吞吐量递减是因为引入了顺序瓶颈。后一种情况下，程序员直接管理独立的并行进程来处理单个数据流并且合并单个数据流数据。

系统 301 可能支持并行的多个唯独方面。特别地，系统 301 可能并行操作数据流的分区。再者，系统 301 可能使用并行管道操作一个或多个数据流。特别地，如图 1 所示，分段器 104 接收一个或多个连续数据流 102，并行操作这些数据流，也有许多数据处理操作器 24 对离散数据流进行操作。

图 4 描述了多个连续数据源各自产生多个连续数据流的数据流程。在步骤 401 里，进程 400 开始运行。在步骤 402-403，系统 301 可能引入许多日志文件。这些引入进程可能并行发生，而且这些引入进程的结果可能避开一个或多个数据操作器 106，该操作器在步骤 405-407 执行处理日志文件。尽管描述了三个数据流，系统 301

可以处理任意数量的并行数据流，包含任意数量的并行管道。这些引入进程的结果可以对数据流进行重新分区，把数据流的不同部分重新分配给不同的数据处理操作器 106。

在步骤 405-407，这些日志文件以并行方式被处理，典型地，是由系统 301 的处理器的不同线程执行的。执行的处理可能包括对输入数据流元素进行排序和合并操作。这些排序和合并进程可以把相似数据联系在一起，或者根据语义 103 或预定以的规则重新组织数据。在步骤 408-410，每一个数据流各自地，例如，由数据处理操作器 106 处理。这些数据操作器可以执行包括数据删除、清洗和论证等功能。因为输入数据流可能包含坏数据，系统 301 有能力检测和拒绝这些数据。这种检测可能基于指示数据流里有效纪录开始的指定字节格式，或者其它业界周知的错误检测和更正方法。由于多达三分之一的流过 HTTPD 进程的互联网流量是由蜘蛛产生的，到来数据流的一个或多个部分可能被“清洗过”。特别地，可能存在在数据流里过滤和修改纪录的多用途组件。这些组件可能操作，例如用户通过管理系统 505 根据预定义的规则设置，下文结合图 5 解释管理系统 505。

再者，数据流里的项目可能被辅助于其它信息。例如，WEB 站点活动可能与来自其它事务源的实时数据相合并，诸如来自销售部门、发货部门和客户支持，以便形成一对一的市场应用。因此，系统 301 有能力辅助数据流基于，例如，内存表查找和数据库查找。例如，把与指定广告相关的所有广告客户信息辅助于数据流，这将允许用户对每个广告进行详细的收入分析。其它形式的数据辅助也可能出现。

在步骤 411-413，多个数据流的数据可能会被聚合。特别地，系统 301 可以提供几个分组操作器分析和组合来自多个数据流的数

据。这提供，例如，一种在几个相互独立的方面进行有效分组和分析数据的能力去分析 WEB 活动。更特别地是，为了提供对数据评估的准确性需要分析多个数据源的数据。在步骤 414-416，聚合的流数据保存在一个或多个地址。特别地，数据可能被聚合而且保存在关系数据库中。根据一个实事方案，系统 301 可能以并行的方式在关系数据库里保存信息。

系统 301 可能被，例如，作为执行在一个或多个计算机系统之上的程序来实现。这些计算机系统可能是，例如，业界周知的多用途计算机系统。更特别地，多用途计算机包括业界周知的处理器、内存、存储设备、以及输入/输出设备。多用途计算机可以在操作系统上执行，在此操作系统之上可以利用编程语言设计一个或多个计算机系统。样例操作系统包括微软公司的 Windows95，Windows 98 或 Windows NT 操作系统，来自于太阳微系统公司、惠普公司、红帽子公司以及许多供应商的 Solaris，HPUX，Linxu，或其它基于 UNIX 的操作系统或者其它未来熟悉的操作系统。

图 5 描述了一组功能上作为客户机 501 和服务器 503 的多用途计算机。在一个实施方案，数据处理应用程序 107 可以作为执行在服务器 503 上的一个或多个进程。特别地，服务器程序 510 在连续数据流 102 上执行一个或多个操作。在一个实施方案里，服务器 503 包括作为应用程序编程接口的对象结构 509，程序员可以通过应用程序接口控制服务器程序 501 的处理过程。客户机 501 可以包含管理应用程序 505，用户通过该管理程序执行输入和输出 502 以便执行服务器程序 510 的管理功能。管理应用程序 505 可以包括一般用户界面 506，该界面被配置用于显示和接收来自决定服务器程序 501 如何操作的配置信息。管理应用程序 505 也可能包含管理用户信息和给服务器程序 510 提供信息的基本客户机程序 507。客户机 501 和服务器 503 之间的通讯是通过客户机通讯程序 508 和服务器通讯

程序 511 在网络 504 上执行的。客户和服务器通讯程序 508、511 可能报刊，例如，网络协议诸如 TCP/IP，网络 504 可能是以太网、ISDN，ADSL，或其它形式的用于系统之间传递信息的网络。客户机-服务器和网络通讯在计算机和网络领域是众所周知的。

服务器 503 可能，例如，把结果 108 保存在和服务器 503 相关的一个或多个数据库 512。在一个实施方案里，数据库 512 是并行关系数据库。服务器 503 也可能存储许多描述服务器程序 510 如何操作的用户配置文件 513。

如上所述，数据处理应用程序 107 可以是基于客户-服务器的结构。这个结构可以使用包括 Java，C++以及其它编程语言的一个或多个编程语言来设计。根据一个实施方案，数据处理应用程序 107 是由 C++编写的，而且 C++结构被定义包括处理数据流数据的组件或对象。这些对象可能是对象结构 509 的一部分。例如，可能有用于分割、合并、连接、过滤和拷贝数据的组件。服务器程序 510 根据用户配置文件 513 管理数据处理应用程序 107 的执行。这个配置文件 513 描述了诸如处理节点的网络名的基本计算机系统资源以及诸如磁盘空间和内存等计算机系统资源。数据库 512 可以被用于存储相关的信息，诸如包含描述数据设计的元数据，以及用户定义的组件和程序。

图 6 描述了实现数据处理应用程序 107 的体系结构 601。体系结构 601 可能由多层组成。例如，体系结构 601 可以包括领头进程 602 负责创建单个程序行为。特别地，进程 602 建立数据处理应用程序 107 的实例。领头进程 602 可能生成区段带头进程 603 和 604。在一个实施方案里，领头进程 602 产生区段带头进程 603 和 604，使用周知的执行远程命令的 Unix 命令 “rsh”。在一个实施方案里，在每一个物理计算机系统上产生区段带头进程。每一个区段带头进

程 603-604 通过周知的 fork（）产生参与者进程，数据流程中的每个数据处理操作器 106 对应一个进程。领头进程，例如，可能和区段带头和/或参与者进程 605-610。

领头进程 602 通过连接 611，612 发送控制信息和接收状态消息来和区段带头进程 603-604 进行通讯。同样，区段带头进程 603-604 通过发送控制信息和接收状态和错误消息来和参与者进程 605-610 进行通讯。通常来讲，领头进程 602 组合消息流量并且确保程序操作的平滑操作。在参与者进程 605-610 失败的情况下，区段带头进程 603-604 帮助程序操作、结束他们控制的参与者进程，之后通知其它区段带头进程执行同样的动作。

数据应用程序 107 可能和整个结构里管理数据输入/输出的管理器有关。输入/输出管理器可能，例如，和领头进程（或者操作器）通讯以便处理结构里的数据流，可能和负责保存结果数据实现信息通讯。

输入/输出管理器可能提供一个或多个以下函数：

●提供结构里的数据移动的块缓冲。

●给数据管理器提供块输入/输出服务，例如，输入/输出管理器把块传递给数据管理者。

●为结构提供持久性存储服务，例如，由数据管理器指定将块保存在文件里。

●为避免死锁提供缓冲和流程控制。

在一个实施方案，输入/输出管理器可能给数据管理器提供端口接口。端口代表逻辑连接。例如，端口可能是输入端口（“输入”）

或者输出端口（“输出”），而且可能是虚拟的或物理存在的实体。输出代表单一的外出流，是为持久性数据集合的每个输出分区创建的。至于虚拟端口，进程管理器（领头进程）创建参与者进程之间的连接。根据一个实施方案，特定参与者进程的任何虚拟输出端口可以拥有到下流参与者进程的单一连接。类似地，输入代表单一的入站流，每一个入站数据流都会创建一个输入端口。输入虚拟端口的入站数据流可能不确定地被合并为单一数据块的流。数据块的排序可能被保留在特定的分区里，但是在分区之间没有暗示的排序。因为分区之间没有暗示的排序，因而死锁的现象就避免了。

图 7 描述了一系列建立于节点 1 和节点 2 之间的逻辑连接，每个节点具有操作器 A 和 B 的单独实体。特别地，节点 1 包括参与者操作器（或进程）A 701 和参与者操作器 B 702，在此操作器 A 以串形方式给操作器 B 提供处理用的数据。再者，节点 2 的操作器 A 703 也可能给节点 1 的操作器 B 702 以串形方式提供信息。类似地，参与者操作器 A 701 可能给节点 2 的参与者操作器 B 704 提供处理用的数据。操作器 701-704 之间的一个或多个连接可能使得这些数据传送更容易。在这种情形下，可能出现并行管道进程之间的通讯。

到目前为止已经介绍了一些实施方案，接下来仅仅是阐述而且不局限于通过例子来表现，对于那些本领域内的普通技术人员这一点应该是显而易见的。

例如，在连续数据流 102 的分段之前，可以过滤数据以排除不符合或者偏离或者影响数据分析的纪录。例如，如果连续数据流是要排除发送给服务器的请求的信息日志，那么可能有一个或多个请求被过滤。这类被排除掉的信息包括与各种诸如计算机程序“蜘蛛”、“爬行者”、“机器人”等实体相关的请求。这种程序是由搜索

引擎执行的，用来在计算机网络上访问文件服务器获得文件，建立索引。这些由蜘蛛、爬行者和机器人发送的请求同样也被纪录在日志中。这些程序具有可能知晓的主机名和代理名。过滤操作可能过滤来自具有已知蜘蛛、爬行者和机器人的用户。服务器也可能具有预先定义好名称的文件，该文件指定服务器上的哪些文件可以被蜘蛛、爬行者和机器人访问。访问这些文件的信息可以用于标示蜘蛛、爬行者和机器人的主机名称或代理名称，继而可以把这些实体从访问其它文件中过滤掉。再者，排除重复的数据纪录或者其它数据清洗操作可能是合适的。这种过滤通常在应用事务语义给连续数据流分段之前被执行，但是也可能在数据分段之后执行。这些以及其它改动都被认为是属于本发明的范围之内。

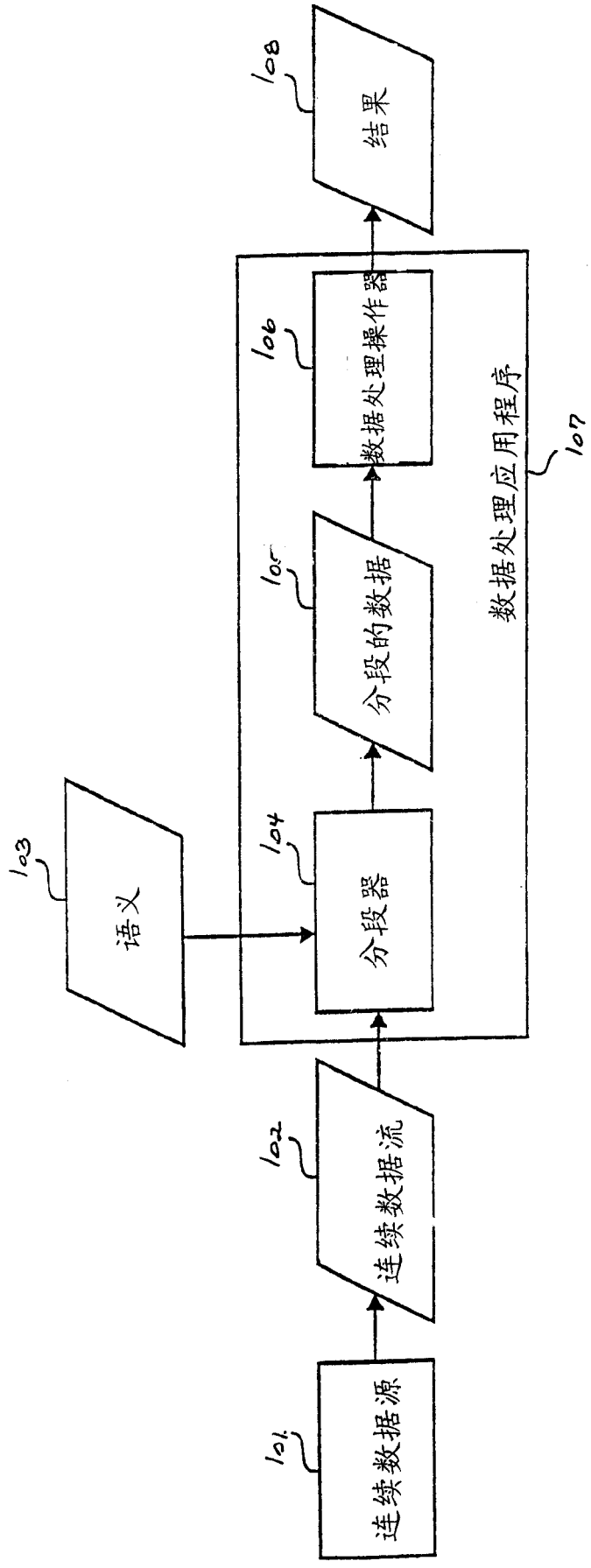

连续数据源
101
连续数据流
102
语义
103
分段器
104
分段的数据
105
数据处理操作器
106
数据处理应用程序
107
结果
108

图 1

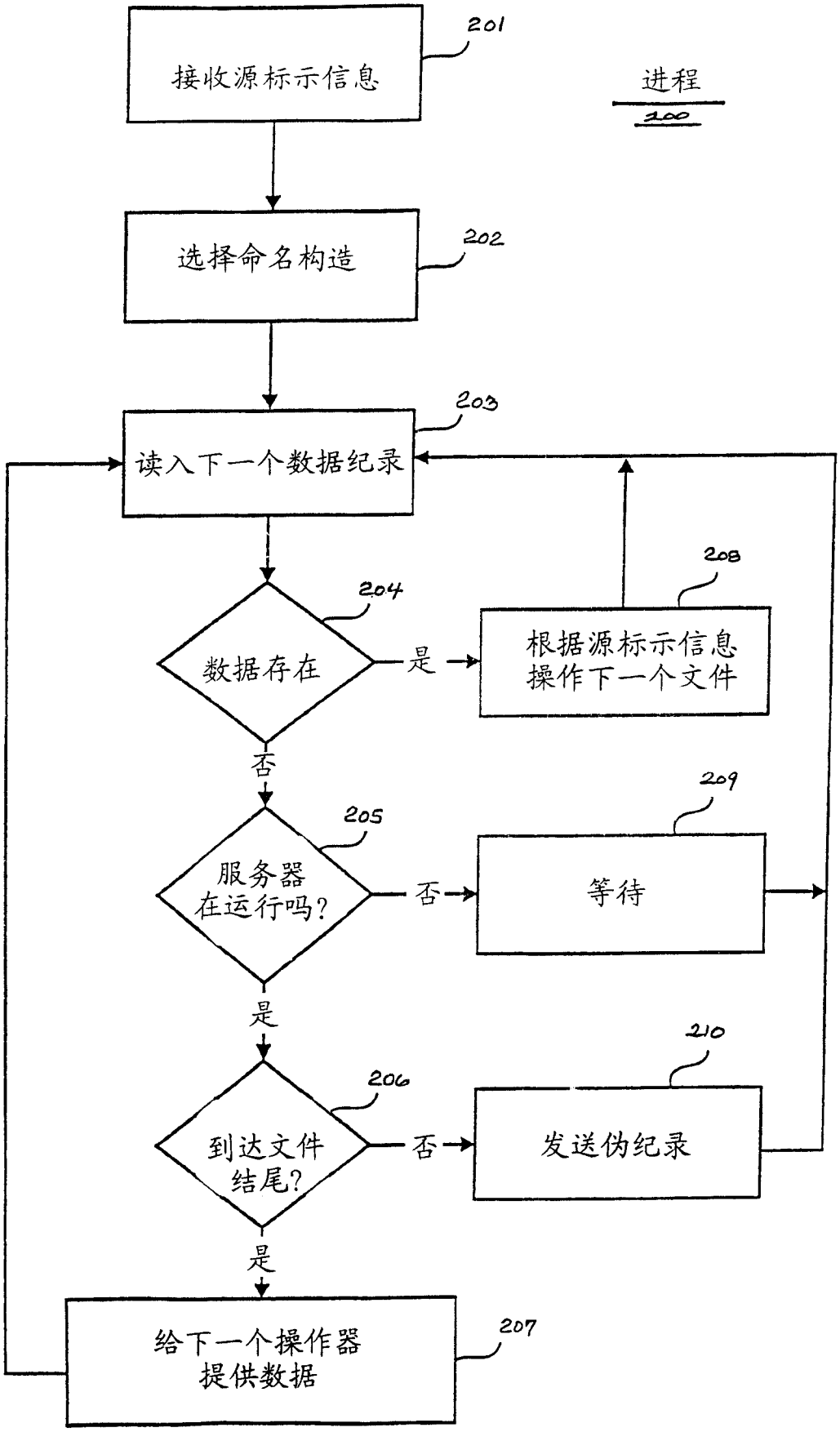

接收源标示信息
201
进程
200
选择命名构造
202
读入下一个数据纪录
203
数据存在
204
是
根据源标示信息
操作下一个文件
208
否
服务器
在运行吗?
205
否
等待
209
是
到达文件
结尾?
206
否
发送伪纪录
210
是
给下一个操作器
提供数据
207

图 2

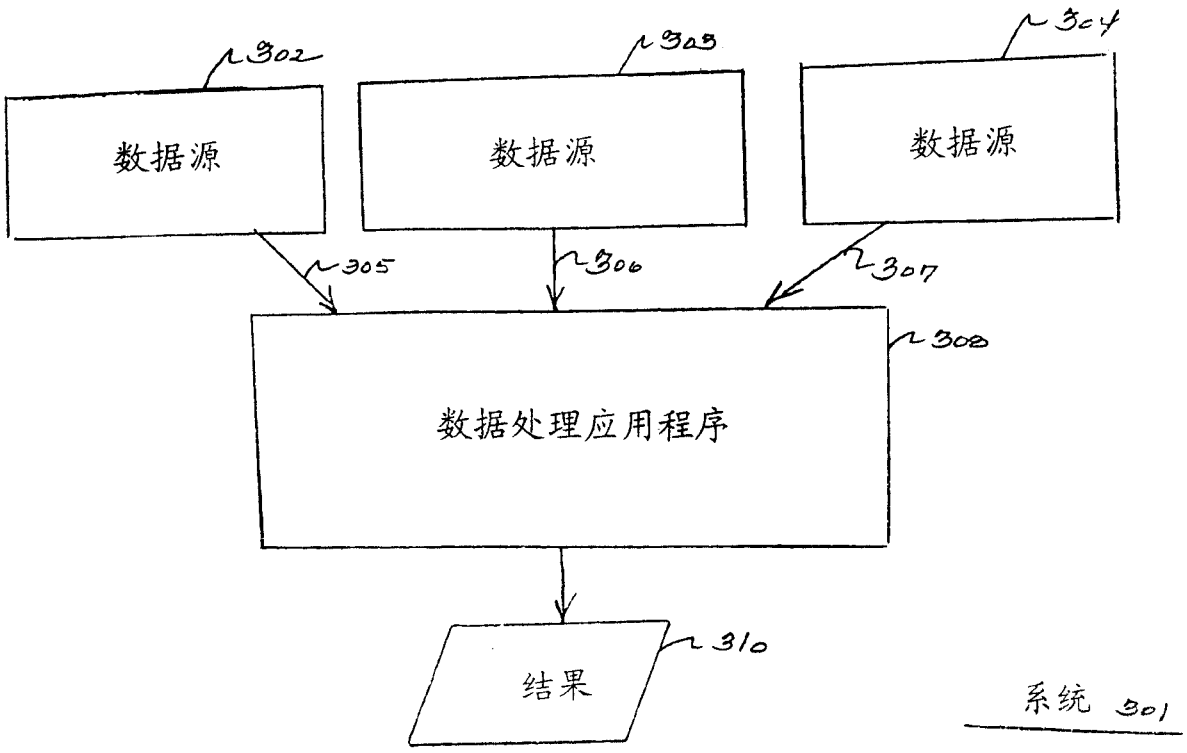


图 3

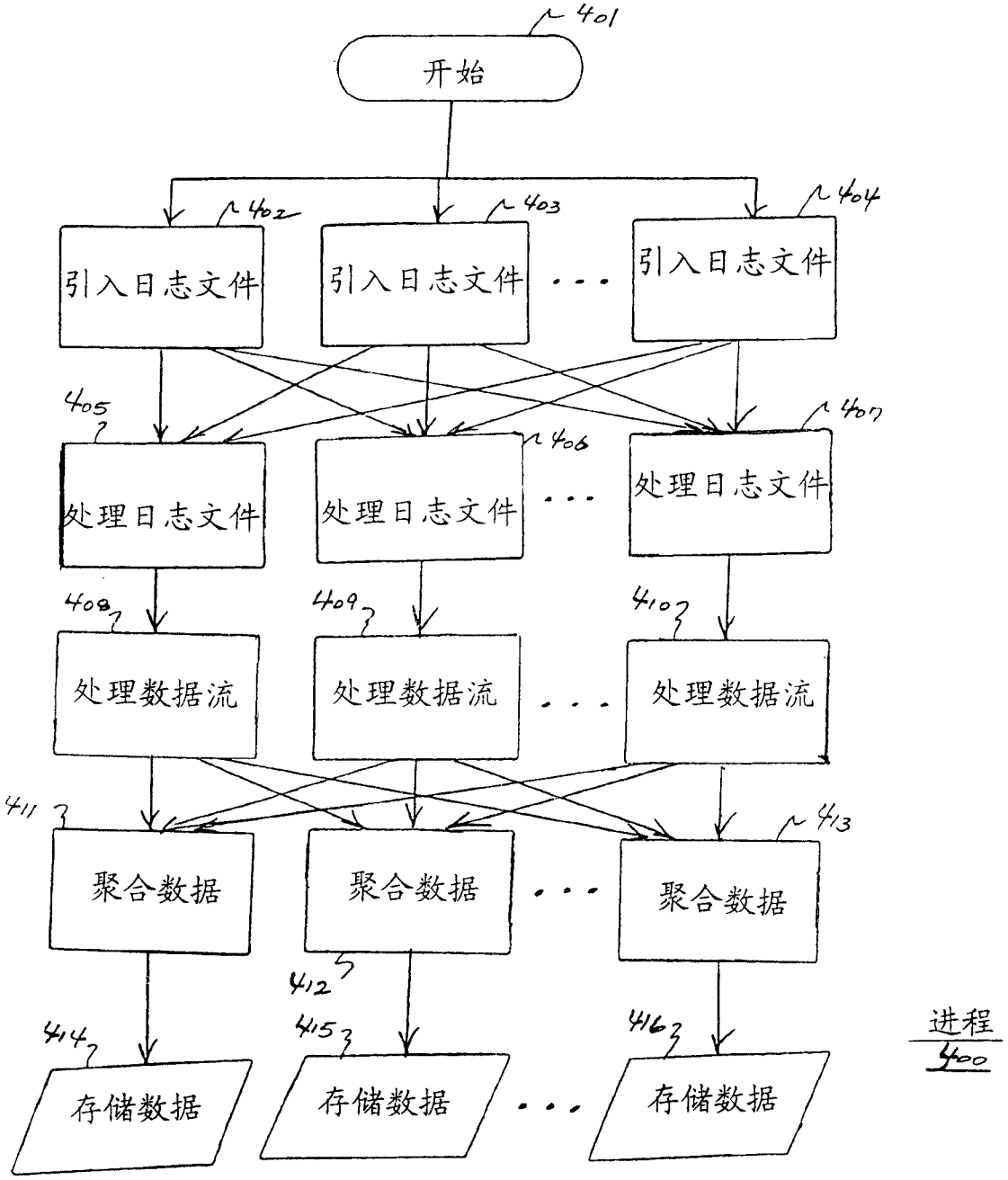

401
开始
402
403
404
引入日志文件
引入日志文件
引入日志文件
405
406
407
处理日志文件
处理日志文件
处理日志文件
408
409
410
处理数据流
处理数据流
处理数据流
411
413
聚合数据
聚合数据
聚合数据
412
414
415
416
存储数据
存储数据
存储数据
进程
400

图 4

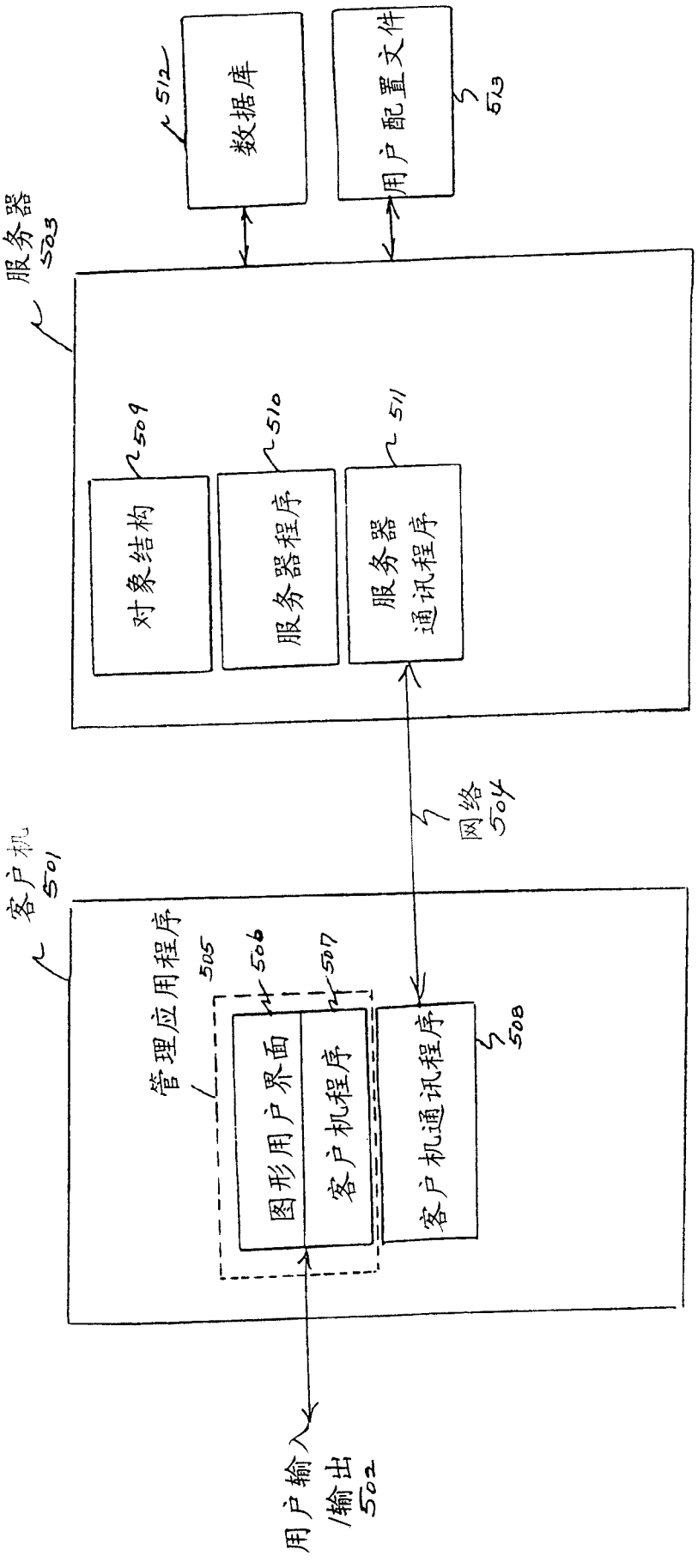

数据库
512
用户配置文件
513
服务器
503
对象结构
509
服务器程序
510
服务器
通讯程序
511
网络
504
客户机
501
管理应用程序
505
图形用户界面
506
客户机程序
507
客户机通讯程序
508
用户输入
/输出
502

图 5

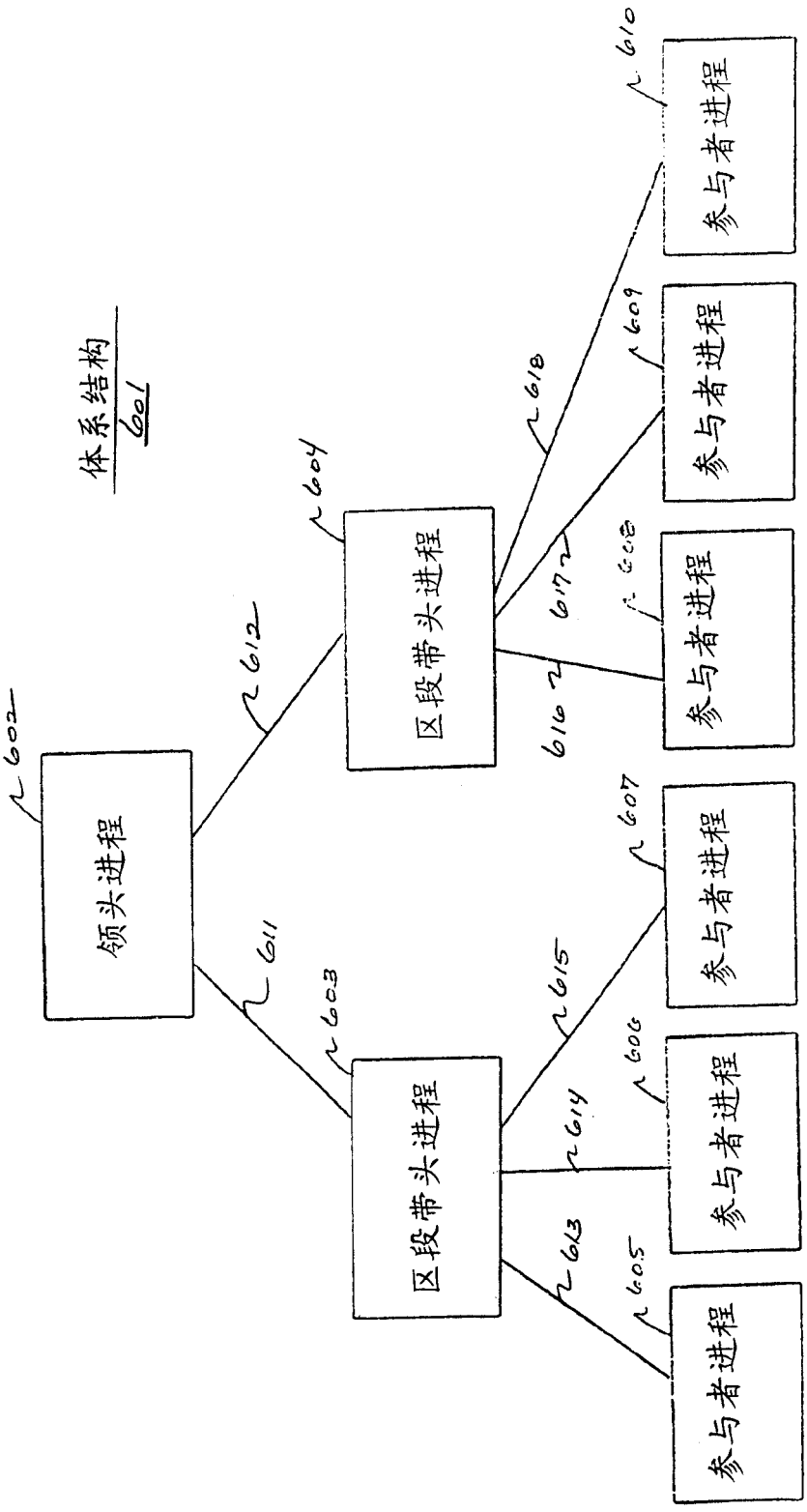

体系结构
601
领头进程
602
区段带头进程
603
区段带头进程
604
参与者进程
605
参与者进程
606
参与者进程
607
参与者进程
608
参与者进程
609
参与者进程
610
611
612
613
614
615
616
617
618

图 6

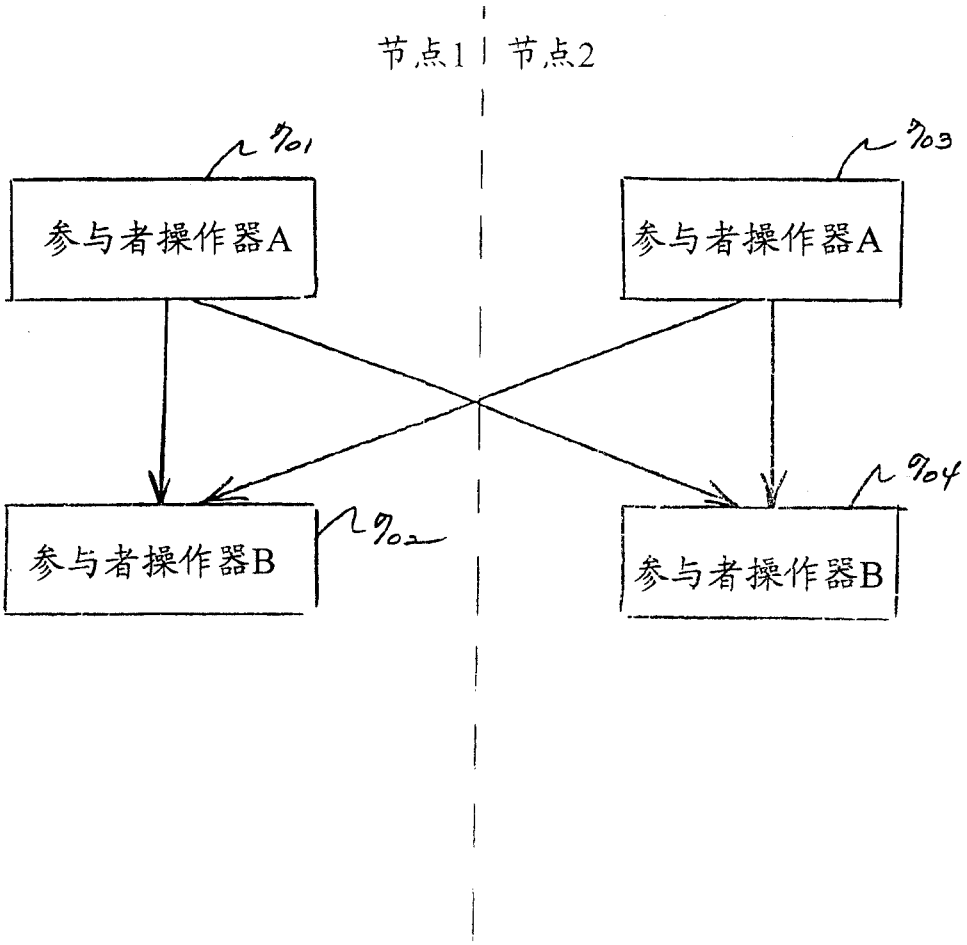


图 7