(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 105324964 B
(45)授权公告日 2019.06.28

(21)申请号 201480035570.2

(22)申请日 2014.07.18

(65)同一申请的已公布的文献号
申请公布号 CN 105324964 A

(43)申请公布日 2016.02.10

(30)优先权数据
61/859,650 2013.07.29 US
14/226,288 2014.03.26 US

(85)PCT国际申请进入国家阶段日
2015.12.22

(86)PCT国际申请的申请数据
PCT/US2014/047280 2014.07.18

(87)PCT国际申请的公布数据
WO2015/017145 EN 2015.02.05

(73)专利权人 甲骨文国际公司
地址 美国加利福尼亚

(72)发明人 B·博格丹斯基 B·D·约翰森

(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 11038
代理人 李晓芳

(51)Int.Cl.
H04L 12/735(2006.01)
H04L 12/931(2006.01)
H04L 12/707(2006.01)
H04L 12/933(2006.01)

(56)对比文件
US 2012027017 A1,2012.02.02,说明书第[0002],[0003],[0019]-[0021],[0023]-[0035]段、附图3-4B.
US 2012027017 A1,2012.02.02,说明书第[0002],[0003],[0019]-[0021],[0023]-[0035]段、附图3-4B.
US 2013114620 A1,2013.05.09,全文.
Laiquan Han等.A Novel Multipath Load Balancing Algorithm in Fat-Tree Data Center.《CLOUD COMPUTING》.2009,第3.1,3.2节.

审查员 颜悦

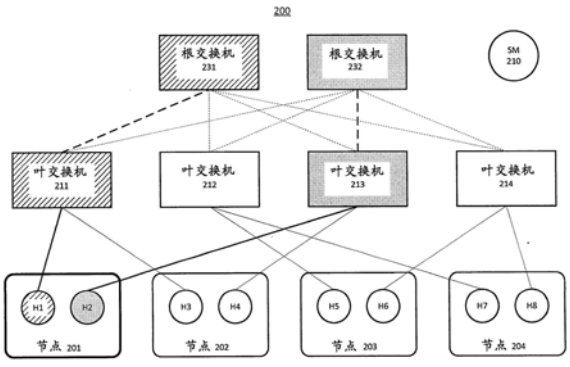
权利要求书4页 说明书10页 附图6页

(54)发明名称

用于支持在中间件机器环境中的多宿主胖树路由的系统和方法

(57)摘要

一种系统和方法可以支持网络环境中的多宿主路由,所述系统和方法可以基于使用胖树或类似的拓扑的无限宽带体系架构。系统可以提供与网络构架中的叶交换机上的交换端口相关联的末端节点。然后,系统可以针对末端节点上的多个端口中的每个端口执行路由,并且确保在末端节点上的多个端口采取相互独立的路径。



CN 105324964 B

1.一种用于支持网络环境中的多宿主路由的方法,包括:

提供与网络构架相关联的多个末端节点,其中所述网络构架包括连接到多个叶子交换机的多个根交换机,其中每个末端节点包括至少第一端口和第二端口,其中所述第一端口与所述网络构架中的第一叶子交换机上的交换机端口相关联,并且所述第二端口与所述网络构架中的第二叶子交换机上的交换机端口相关联;

通过执行以下操作针对所述多个末端节点中的每对末端节点执行路由:

自动计算通过所述多个根交换机和所述多个叶子交换机的所述每对末端节点之间的第一路径,

标记所述每对末端节点之间的所述第一路径上的每个叶子交换机和根交换机,以及

由子网管理器自动计算所述每对末端节点之间的第二路径,同时避免所述每对末端节点之间的所述第二路径使用所述第一路径上的被标记的每个叶子交换机和根交换机,

由此所述每对末端节点之间的第一路径和第二路径是通过所述多个根交换机和所述多个叶子交换机中的不同根交换机和叶子交换机的相互独立的路径。

2.根据权利要求1所述的方法,

其中所述网络构架基于胖树拓扑。

3.根据权利要求2所述的方法,

其中每个末端节点是通过包括所述第一端口和所述第二端口的多个端口与所述胖树拓扑中的两个或更多个部分相连接的多宿主节点。

4.根据权利要求1至3中的任何一个所述的方法,其中:

所述每对末端节点之间的每个第一路径连接所述每对末端节点的第一端口;以及

所述每对末端节点之间的每个第二路径连接所述每对末端节点的第二端口。

5.根据权利要求1至3中的任何一个所述的方法,还包括:

当不存在冗余交换机时,允许与所述末端节点上的另一个端口相关联的另一个路径使用一个或多个被标记的交换机;以及

当并行链路存在于所述一个或多个被标记的交换机上时,针对在相同的末端节点上的不同目标端口选择独立的链路。

6.根据权利要求1至3中的任何一个所述的方法,还包括:

在完成对每个末端节点上的所述第一端口和所述第二端口的路由之后,对每个被标记的交换机取消标记。

7.根据权利要求1至3中的任何一个所述的方法,还包括:

将在所述每对末端节点之间的每个相互独立的第一路径和第二路径与不同的主干交换机关联起来。

8.根据权利要求1至3中的任何一个所述的方法,还包括:

在完成对每个末端节点上的所述第一端口和所述第二端口的路由之后,将该末端节点标记为已路由的末端节点。

9.根据权利要求8所述的方法,还包括:

当每个末端节点遭遇另一个叶子交换机时,防止该末端节点再次被路由。

10.根据权利要求1至3中的任何一个所述的方法,还包括:

使用选取末端节点作为参数的路由算法。

11.一种子网管理器,包括:

提供单元,被配置为提供连接到网络构架的多个末端节点,其中所述网络构架包括连接到多个叶子交换机的多个根交换机,其中每个末端节点包括至少第一端口和第二端口,其中所述第一端口与所述网络构架中的第一叶子交换机上的交换机端口相关联,并且所述第二端口与所述网络构架中的第二叶子交换机上的交换机端口相关联;

执行单元,被配置为通过执行以下操作针对所述多个末端节点中的每对末端节点执行路由:

自动计算通过所述多个根交换机和所述多个叶子交换机的所述每对末端节点之间的第一路径,

标记所述每对末端节点之间的所述第一路径上的每个叶子交换机和根交换机,以及

由所述子网管理器自动计算所述每对末端节点之间的第二路径,同时避免所述每对末端节点之间的所述第二路径使用所述第一路径上的被标记的每个叶子交换机和根交换机,

由此所述每对末端节点之间的第一路径和第二路径是通过所述多个根交换机和所述多个叶子交换机中的不同根交换机和叶子交换机的相互独立的路径。

12.根据权利要求11所述的子网管理器,

其中所述网络构架基于胖树拓扑。

13.根据权利要求12所述的子网管理器,

其中每个末端节点是通过包括所述第一端口和所述第二端口的多个端口与所述胖树拓扑中的两个或更多个部分相连接的多宿主节点。

14.根据权利要求11至13中的任何一个所述的子网管理器,其中:

所述每对末端节点之间的每个第一路径连接所述每对末端节点的第一端口;以及

所述每对末端节点之间的每个第二路径连接所述每对末端节点的第二端口。

15.根据权利要求11至13中的任何一个所述的子网管理器,还包括:

第三允许单元,被配置为当不存在冗余交换机时,允许与所述末端节点上的另一个端口相关联的另一个路径使用一个或多个被标记的交换机;以及

选择单元,被配置为当并行链路存在于所述一个或多个被标记的交换机上时,针对在相同的末端节点上的不同目标端口选择独立的链路。

16.根据权利要求11至13中的任何一个所述的子网管理器,还包括:

取消标记单元,被配置为在完成对每个末端节点上的所述第一端口和所述第二端口的路由之后,对每个被标记的交换机取消标记。

17.根据权利要求11至13中的任何一个所述的子网管理器,还包括:

关联单元,被配置为将在所述每对末端节点之间的每个相互独立的第一路径和第二路径与不同的主干交换机关联起来。

18.根据权利要求11至13中的任何一个所述的子网管理器,还包括:

第二标记单元,被配置为在完成对每个末端节点上的所述第一端口和所述第二端口的路由之后,将该末端节点标记为已路由的末端节点。

19.根据权利要求18所述的子网管理器,还包括:

第二防止单元,被配置为当每个末端节点遭遇另一个叶子交换机时,防止该末端节点再次被路由。

20.根据权利要求11至13中的任何一个所述的子网管理器,还包括:

第四允许单元,被配置为允许路由算法选取末端节点作为参数。

21.一种用于支持网络环境中的多宿主路由的系统,包括:

一个或多个微处理器,

运行在所述一个或多个微处理器上的子网管理器,其中所述子网管理器操作以:

将多个末端节点与网络构架相关联,其中所述网络构架包括连接到多个叶子交换机的多个根交换机,其中每个末端节点包括至少第一端口和第二端口,其中所述第一端口与所述网络构架中的第一叶子交换机上的交换机端口相关联,并且所述第二端口与所述网络构架中的第二叶子交换机上的交换机端口相关联;

通过执行以下操作针对所述多个末端节点中的每对末端节点执行路由:

自动计算通过所述多个根交换机和所述多个叶子交换机的所述每对末端节点之间的第一路径,

标记所述每对末端节点之间的所述第一路径上的每个叶子交换机和根交换机,以及

由所述子网管理器自动计算所述每对末端节点之间的第二路径,同时避免所述每对末端节点之间的所述第二路径使用所述第一路径上的被标记的每个叶子交换机和根交换机,

由此所述每对末端节点之间的第一路径和第二路径是通过所述多个根交换机和所述多个叶子交换机中的不同根交换机和叶子交换机的相互独立的路径。

22.根据权利要求21所述的系统,其中:

所述网络构架基于胖树拓扑。

23.根据权利要求22所述的系统,其中:

每个末端节点是通过包括所述第一端口和所述第二端口的多个端口与所述胖树拓扑中的两个或更多个部分相连接的多宿主节点。

24.根据权利要求21-23中的任何一个所述的系统,其中:

所述每对末端节点之间的每个第一路径连接所述每对末端节点的第一端口;以及

所述每对末端节点之间的每个第二路径连接所述每对末端节点的第二端口。

25.根据权利要求21-23中的任何一个所述的系统,其中:

所述子网管理器操作以:

当不存在冗余交换机时,允许与所述末端节点上的另一个端口相关联的另一个路径使用一个或多个被标记的交换机;以及

当并行链路存在于所述一个或多个被标记的交换机上时,针对在相同的末端节点上的不同目标端口选择独立的链路。

26.根据权利要求21-23中的任何一个所述的系统,其中:

所述子网管理器操作以在完成对每个末端节点上的所述第一端口和所述第二端口的路由之后,对每个被标记的交换机取消标记。

27.根据权利要求21-23中的任何一个所述的系统,其中:

在所述每对末端节点之间的每个相互独立的第一路径和第二路径与不同的主干交换机相关联。

28.根据权利要求21-23中的任何一个所述的系统,其中:

所述子网管理器操作以在完成对每个末端节点上的所述第一端口和所述第二端口的

路由之后,将该末端节点标记为已路由的末端节点。

29.根据权利要求28所述的系统,其中:

所述子网管理器操作以在每个末端节点遭遇另一个叶子交换机时,防止该末端节点再次被路由。

30.根据权利要求21-23中的任何一个所述的系统,其中:

所述子网管理器操作以使用选取末端节点作为参数的路由算法。

用于支持在中间件机器环境中的多宿主胖树路由的系统和方法

[0001] 版权声明

[0002] 本专利文件的公开的一部分包含受到版权保护的材料。由于专利文件或专利公开出现在专利与商标局的专利文件或纪录中，因此版权所有者不反对任何人对专利文件或专利公开的传真复制，但在其他方面保留任何所有版权权利。

技术领域

[0003] 本发明通常涉及计算机系统，并且具体地涉及网络环境。

背景技术

[0004] 胖树(fat-tree)拓扑被用于高性能的计算(HPC)集群，并且被用于基于无限宽带(IB)技术的集群。例如，胖树拓扑被用在最快的超级计算机中，诸如MilkyWay-2。同样地，胖树IB系统包括诸如Stampede、TGCC Curie和SuperMUC之类的大型安装。

[0005] 这些是通常本发明的实施例意图解决的领域。

发明内容

[0006] 本文所描述的是可以支持在网络环境中的多宿主路由的系统和方法，可以基于使用胖树或类似的拓扑的无限宽带体系架构。系统可以提供与网络构架中的叶交换机上的交换端口相关联的末端节点。然后，系统可以针对末端节点上的多个端口中的每个端口执行路由，并且确保在末端节点上的多个端口采取相互独立的路径。

附图说明

[0007] 图1示出了支持网络环境中的胖树路由的说明；

[0008] 图2示出了根据本发明的实施例、支持在网络环境中的多宿主路由的说明；

[0009] 图3示出了根据本发明的实施例、提供用于支持在网络环境中的胖树路由的冗余的说明；

[0010] 图4说明根据本发明的实施例、支持在网络环境中的多宿主路由的示例性流程图；

[0011] 图5说明可以在其上实施本发明的实施例的计算机系统的框图；

[0012] 图6说明用于实施本发明的实施例的系统的框图。

具体实施方式

[0013] 在附图中以示例的方式而不以限制的方式说明本发明，在附图中相同的引用指示相似的元件。应当注意的是，在本公开中对“一”或“一个”或“一些”实施例的引用不一定是相同的实施例，并且这样的引用意味着至少一个。

[0014] 以下对本发明的描述使用无限带宽(IB)网络作为高性能网络的示例。对于本领域技术人员来说清楚的是，其它类型的高性能网络可以被使用，而不具有限制性。同样地，以

下对本发明的描述使用胖树拓扑作为构架拓扑的示例。对于本领域技术人员来说清楚的是，其他类型的构架拓扑可以被使用，而不具有限制性。

[0015] 本文所描述的是可以支持在网络环境中的多宿主路由的系统和方法。

[0016] 无限带宽架构

[0017] 无限带宽架构(IBA)支持两层拓扑分区。在低层处，IB网络被称作子网，其中子网可以包括一组使用交换机和点对点链路互联的主机。在高层级处，IB构架构成可以通过使用路由器而互联的一个或多个子网。

[0018] 另外，在子网内的主机和交换机可以通过使用局部标识符(LID)定址，而单个子网可以被限制到49151个LID。除了作为仅仅在子网内有效的局部地址的LID之外，每个IB设备可以具有被烧制到非易失性存储器中的64位的全局唯一标识符(GUID)。GUID可以被用于形成全局标识符(GID)，其是IB第三层(L3)地址。可以通过将64位子网标识符(ID)与64位GUID连结起来以形成类IPv6的128位地址，来创建GID。例如，不同的端口GUID可以被分配到与IB构架相连接的端口。

[0019] 另外，子网管理器(SM)可以负责在IB构架中执行路由表计算。这里，IB网络的路由旨在获得完全的连接性、无死锁性、以及在局部子网中的所有源和目的地对之间的合适的负载均衡。

[0020] 子网管理器可以在网络初始化的时间计算路由表。另外，每当拓扑改变时路由表可以被更新，以便确保最优的性能。在正常运行期间，子网络管理器可以执行对网络的定期光扫描以检查拓扑改变。如果在光扫描期间发现了改变，或如果发出网络改变的信号的消息(陷阱)被子网管理器接收，则根据被发现的改变，子网管理器可以重新配置网络。

[0021] 例如，当网络拓扑改变时，诸如当链路失灵时、当设备被添加时、或当链路被移除时，子网络管理器可以重新配置网络。重新配置的步骤可以包括在网络初始化期间执行的步骤。另外，重新配置可以具有被限制到子网络内的局部范围，在此范围中网络改变发生。另外，用路由器对大型构架的分段可以限制重新配置范围。

[0022] 另外，基于无损网络技术的IB网络，在特定条件下可能容易死锁。例如，在诸如缓冲区或信道之类的网络资源被分享并且分组丢弃不被允许的IB网络中，死锁可以发生。这里，死锁发生的一个必要条件是循环信用依赖性(cyclic credit dependency)的建立，这意味着循环信用依赖性可以使得死锁的发生成为可能。另一方面，这不意味着每当循环信用依赖性存在时总是会存在死锁。

[0023] 胖树路由

[0024] 胖树拓扑可以提供用于支持高性能互联的各种益处。这些益处可以包括无死锁性、自有的容错性、以及完全对分带宽。无死锁性表示使用树结构使得在没有对避免死锁的特别考虑的情况下路由胖树成为可能。自有的容错性表示在单独的源-目的地对之间的多个路径的存在使处理网络故障更容易。完全对分带宽表示网络可以保持在网络的两半之间的全速通信。

[0025] 另外，胖树路由算法可以被用于支持对底层胖树拓扑的高效使用。以下算法1是示例性胖树路由算法。

算法 1 *route_to_cns() function*

要求: 完成寻址
确保: 所有 *hca_ports* 被路由

[0026]
```
1: for swleaf = 0 to max_leaf_sw do
2:   for swleaf.port = 0 to max_ports do
3:       hca_lid = swleaf.port-> remote_lid
4:       swleaf.routing table[hca_lid] = swleaf.port
5:       route_downgoing_by_going_ up()
6:   end for
7: end for
```

[0027] 如上所示,路由函数route_to_cns(),可以在叶交换机的数组上进行迭代(行1-7)。对于每个被选择的叶交换机,路由函数可以路由到与被选择的叶交换机相连接的每个末端节点端口,例如按端口编号的顺序(行2-6)。

[0028] 另外,当路由与特定的LID相关联的末端节点端口时,路由函数可以在网络拓扑中向上一个层级以路由下行路径,并且当路由每个交换机端口时,路由函数可以向下以路由上行路径。这个过程可以被重复直到到达根交换机层级。在此之后,到所有节点的路径被路由并且被插入构架中的所有交换机的线性转发表(LFT)中。

[0029] 例如,route_downgoing_by_going_up()函数(行5)可以是可以均衡路径并且调用route_upgoing_by_going_down()函数的重现函数,route_upgoing_by_going_down()函数通过从中调用route_downgoing_by_going_up()函数的交换机将胖树中的上行路径路由到目的地。

[0030] 可以存在若干与route_to_cns()函数相关联的潜在的缺点。第一,route_to_cns()函数是不经意的,并且在不考虑末端端口属于哪个末端节点的情况下路由末端端口。第二,route_to_cns()函数依赖于用于路由的物理端口编号。

[0031] 图1示出了支持在网络环境中的胖树路由的说明。如图1所示,一个或多个末端节点101-104可以连接到网络构架100。网络构架100可以基于胖树拓扑,其包括多个叶交换机111-114,以及多个主干交换机或根交换机131-134。另外,网络构架100可以包括一个或多个中间交换机,诸如交换机121-124。

[0032] 仍然如图1所示,末端节点101-104中的每一个可以是多宿主节点,即通过多个端口连接到网络构架100的两个或多个部分的单个节点。例如,节点101可以包括端口H1和H2,节点102可以包括端口H3和H4,节点103可以包括端口H5和H6,而节点104可以包括端口H7和H8。

[0033] 另外,每个交换机可以具有多个交换端口。例如,根交换机S1131可以具有交换端口1-2,根交换机S2132可以具有交换端口3-4,根交换机S3133可以具有交换端口5-6,而根交换机S4134可以具有交换端口7-8。

[0034] 使用诸如算法1之类的、以叶交换机为基础进行路由的胖树路由算法,无法保证独立的路由将被分配到不同的两端口节点101-104。例如,端口H1、H2、H5和H6可以连接到每个交换机上的端口1(未示出),而端口H3、H4、H7和H8连接到每个交换机上的端口2(未示出)。这里,在通过四个末端端口路由以及遍历叶交换机113和交换机123之后,胖树路由算法可

以分配从节点101上的H1和H2末端端口对到相同的最左端的根交换机S1131的两个路径(分别经由交换端口1-2)。类似地,其他末端端口对,例如,节点102上的H3和H4、节点103上的H5和H6、以及节点104上的H7和H8,可以通过相同的根交换机(即,分别为S2132-S4134)被路由。

[0035] 这可以导致对用户来说不理想的行为。如图1所示,即使末端节点101可以具有内置的物理容错性(即,有连接到不同的叶交换机111和113的两个末端端口),但末端节点101在根交换机S1131处可以具有单个故障点。另外,依赖于物理布线,类似的问题可以出现,并且单个故障点可以发生在胖树拓扑中的其他交换机上。

[0036] 另外,在网络构架100内,到相同节点上的不同端口的流量可以通过单个链路被路由。因此,此单个链路可以表示针对末端端口的集合的附加的单个故障点,也可以表示性能瓶颈(因为以不同的末端端口为目标的流量可以仅仅能够有效地利用此单个共享链路的带宽)。

[0037] 多宿主胖树路由

[0038] 根据本发明的实施例,系统可以提供针对胖树中的多宿主节点的独立路由,因此单个故障点不会导致完全的停机。

[0039] 图2示出了根据本发明的实施例、对支持网络环境中的多宿主路由的说明。如图2所示,网络环境可以包括多个末端节点(例如,节点201-204),其中每个末端节点可以包括一个或多个端口(例如,端口H1-H8)。另外,多个末端节点201-204可以连接到可以是胖树拓扑的网络构架200。同样地,网络构架200可以包括多个交换机,例如叶交换机211-214以及根交换机S1231和S2232。

[0040] 根据本发明的实施例,诸如mFtree算法之类的多宿主的胖树路由算法可以被用于执行胖树路由,例如,通过子网管理器(SM)210执行。在图2所示的示例中,mFtree算法可以识别出,从节点201上的端口H1和端口H2出发的路径可能需要以相互冗余的方式被路由,因为两个端口均位于单个末端节点1上。

[0041] 另外,mFtree算法可以确保路径实际上是冗余的。例如,从节点201上的端口H1出发的路径可以经过叶交换机211并且最终通向根交换机231。如图2所示,系统可以标记出路径中的交换机(如深阴影所示)。然后,系统可以避免使用被标记的交换机来确定从节点201上的端口H2出发的路径。因此,从节点201上的端口H2出发的路径可以经过冗余路径(例如,经由叶片交换机213),并且最终通向不同的根交换机232(如淡阴影所示)。

[0042] 当针对节点201的路由步骤被完成时,算法可以标记此节点为已路由(如粗线所示),因此当算法遭遇节点201的另一个端口时,针对节点201,路由步骤不重复进行。因此,系统可以确保单个故障点不会导致多端口节点的完全停机。

[0043] 另外,胖树路由算法可以提供在无限带宽(IB)胖树拓扑的性能、可扩展性、有效性和可预测性方面的改善。

[0044] 以下的算法2是示例性的多宿主胖树路由算法。

算法 2 *route_multihomed_cns() function*

要求: 完成寻址

确保: 所有 *hca_ports* 通过独立的主干被路由

```
1: for swleaf = 0 to leaf_sw_num do
2:   for swleaf.port = 0 to max_ports do
3:       hca_node = swleaf.port -> remote_node
4:       if hca_node.routed ==   true then
5:           continue
6:       end if
7:       route hcas(hca_node)
8:   end for
9: end for
```

[0045]

[0046] 如上所示,作为多宿主路由算法的算法2可以在所有叶交换机上进行迭代,并且然后可以针对每个叶交换机在叶交换机的所有端口上进行迭代(行1-9)。因此,算法2可以是确定性的,类似于算法1。

[0047] 另外,算法2可以取在叶交换机上的交换端口以便找到与交换端口相关联的末端节点(行3)。与简单地取与叶交换机相连接的远程端口的LID的算法1不同,算法2可以将末端节点作为用于执行路由计算的参数(行7)。

[0048] 以下算法3是用于路由胖树中的单个末端节点的示例性算法。

算法 3 *route_hcas(hca) function*

要求: 要被路由的节点

确保: 所有属于具有 *hca_lid* 的节点的 *hca_ports* 被路由

```
1: for hca_node.port = 0 to port_num do
2:   hca_lid = hca_node.port -> lid
3:   swleaf = hca_node.port -> remote_node
4:   swleaf.port = hca_node.port -> remote_port_number
5:   swleaf.routing_table[hca_lid] = swleaf.port
6:   route_downgoing_by_going_up()
7: end for
8: hca_node.routed = true
9: clear_redundant_flag()
```

[0049]

[0050]

[0051] 如上所示,算法3可以在被选择的末端节点上的所有端口上进行迭代(行1-7)。例如,算法3可以通过使用修改的版本的route_downgoing_by_going_up()函数,对被选择的末端节点上的每个端口进行路由(行6)。当在被选择的末端节点上的所有端口被路由时,路由算法可以将被选择的末端节点标记为已路由(行8),因此当在另一个叶交换机上遭遇末端节点时,末端节点不被路由。同样地,算法3可以改善在各种状况下系统的性能(例如,算法3可以针对双端口节点节省一半的循环迭代)。

[0052] 另外,算法3可以被应用到在单个主机信道适配器(HCA)上具有多个端口的情形,也可以被应用到在两个或更多个HCA上有多个端口的情形。算法可以使用不同的方法以用于识别在相同的逻辑节点上的单个HCA或多个HCA上的端口(或任何末端端口)。这里,节点

可以是物理或虚拟的服务器、IO设备、或任何种类的经由一个或多个HCA端口与IB构架相连接的末端节点。

[0053] 另外，算法3可以路由在被选择的节点上的每个端口，并且使用标志(flag)对路径上的每个交换机进行标记。因此，算法3可以针对相同末端节点上的不同端口选择不同交换机。之后，算法可以翻转所有交换机上的标记，以使得算法可以前进到下一个节点。

[0054] 另外，可以通过在clear_redundant_flag()函数中清除交换机的冗余标志(行9)来优化系统。用于清除交换机的冗余标志的优化方式是建立在路径上的交换机的列表，并且确保clear_redundant_flag()函数仅仅在此列表上的那些交换机上迭代，而不是在函数中使用循环，循环使得不论特定的交换机是否在路径上都在所有交换机上进行迭代。

[0055] 以下的算法4是用于路由胖树中的单个末端节点的端口的示例性算法。

[0056]

算法4 *route_downgoing_by_going_up() function*

要求: 当前的跳交换机

确保: 进行尽力而为以找到向上的冗余交换机

确保: 在路径上的交换机被标记为冗余

[0057]

```
1: groupmin = 0
2: redundant_group = 0
3: for port_group = 0 to port_group_num do
4:     if groupmin == 0 then
5:         if groupmin    -> remote_node.redundant then
6:             groupmin = port_group
7:         end if
8:     else if port_group.cntdown < groupmin.cntdown then
9:         groupmin = port_group
10:            if groupmin -> remote_node.redundant then
11:            min redundant group = groupmin
12:            end if
13:        end if
14: end for
15: if groupmin == 0 then
16:        fallback_normal_routing(hca_lid)
17: else if groupmin -> remote node.redundant then
18:        groupmin = min_redundant_group
19:        groupmin -> remote_node.redundant = false
20: end if
```

[0058] 如上述，在算法4中修改版本的route_downgoing_by_going_up()函数将冗余视为首要考虑的问题。与算法1不同，在算法1中具有最小的向下计数器的端口组被选择，而如果不路由属于末端节点的任何其他端口(即，当冗余标志为真时)，则算法4可以仅仅选择末端节点中的向上节点作为下一跳。

[0059] 这里，冗余标志在下一个末端节点被路由之前被清除。如果没有可能出现在严重过度订购的构架中或在链路故障的情况下的冗余的节点，则mFtree退回到正常的胖树路由，在这种情况下，用户可以观察到此路由函数与在算法1中呈现的路由函数类似地执行。

[0060] 另外，当没有可替代的交换机和并行链路存在于两个交换机之间时，以上的算法4能够针对在相同末端节点上的不同目标端口选择不同的链路，以便支持性能/负载分布以及链路层级的冗余二者。另一方面，在性能具有优先级的情况下，仅仅在两个分开的链路具有相同层级的负载时，可以针对在相同的末端节点上不同的目标端口选择分开的链路。

[0061] 图3示出了对用于支持网络环境中的胖树路由提供冗余的说明。如图3所示，一个或多个多宿主末端节点301-304可以连接到网络构架300。网络构架300基于胖树拓扑，网络构架300可以包括多个叶交换机311-312，以及多个主干交换机或根交换机S1331-S4334。另外，网络构架300可以包括一个或多个中间交换机，诸如交换机321-324。

[0062] 同样如图3所示，节点301可以包括端口H1和H2，节点302可以包括端口H3和H4，节点303可以包括端口H5和H6，而节点304可以包括端口H7和H8。另外，根交换机S1331可以具有交换端口1-2，根交换机S2332可以具有交换端口3-4，根交换机S3333可以具有交换端口5-6，而根交换机S4334可以具有交换端口7-8。

[0063] 诸如mFtree算法之类的多宿主路由，可以以一种到达节点上的每个端口的路径是排他的方式路由每个多宿主节点301-304，即mFtree算法确保在多宿主节点上的每个端口是通过独立的路径可到达的。另外，mFtree算法可以改善网络性能。

[0064] 另外，在单个多宿主末端节点的情况下，mFtree算法可以确保没有单个链路被到达任何一对属于相同末端节点的端口的路径共享。同样地，当在网络构架300中存在从不同的源端口到相同的目的地节点上的不同端口的并发流量时，mFtree算法可以确保当可替代的路由存在时，并发流量不共享任何中间链路。

[0065] 因此，通过使用mFtree算法，诸如构架300中的主干交换机S1331之类的单个设备的故障，可以不造成节点301断开连接，因为到达不同端口的路径不在单个主干交换机S1331处聚集。

[0066] 另外，mFtree算法将相同节点上的每个端口视为分开并且独立的实体。因此，mFtree算法可以以节点为基础路由，而不是以端口为基础路由，并且mFtree算法可以考虑不同的末端节点可以具有的不同特性。

[0067] 图4说明了根据本发明的实施例、用于支持网络环境中的多宿主路由的示例性流程图。如图4所示，在步骤401处，系统可以提供与网络构架中的叶交换机上的交换端口相关联的末端节点，其中末端节点与多个端口相关联。然后，在步骤402处，系统可以针对末端节点上每个所述的端口执行路由。另外，在步骤403处，系统可以确保在末端节点上的多个端口取互相独立的路径。

[0068] 图5说明了在其上可以实施本发明的实施例的计算机系统的框图。计算机系统500包括总线或其他用于通信信息的通信机构，以及与总线耦接用于处理信息的硬件处理器。硬件处理器可以是例如通用微处理器。

[0069] 计算机系统500还包括主存储器，诸如随机存取存储器(RAM)，或其他动态的存储设备，主存储器耦接到总线以用于存储信息和要被处理器执行的指令。主存储器还可以被用于在执行要被处理器执行的指令的期间，存储临时变量或其他中间信息。当这样的指令被存储在处理器可执行的非暂时性存储介质时，这样的指令将计算机系统500呈现为被定制以执行指令中指定的操作的专用机器。

[0070] 计算机系统500还包括只读存储器(ROM)或其他耦接到总线以用于存储静态信息

和用于处理器的指令的静态存储设备。诸如磁盘或光盘之类的存储设备，被提供并且耦接到总线以用于存储信息和指令。

[0071] 计算机系统500可以包括诸如阴极射线管(CRT)之类的显示器或经由总线被耦接到所述显示器，显示器用于将信息显示给计算机用户。包括字母数字以及其他键或光标控制的输入设备，被耦接到总线以用于将信息和命令选择通信传送给处理器。

[0072] 计算机系统500可以通过使用与计算机系统结合使得计算机系统500、或对计算机系统500编程使其成为专用机器的、定制的硬接线逻辑、一个或多个ASIC或FPGA、固件和/或程序逻辑，来执行本文所描述的技术。根据一个实施例，响应于处理器执行被包含在主存储器中的一个或多个指令的一个或多个序列，本文中的技术被计算机系统500执行。这样的指令可以从诸如存储设备之类的另一个存储介质被读取到主存储器中。对被包含在主存储器中的指令的序列进行的执行使得处理器可以执行本文所描述的进程步骤。在替代实施例中，硬接线的电路可以被用于替代软件指令或与软件指令结合。

[0073] 计算机系统500还包括耦接到总线的通信接口。通信接口可以是综合服务数字网络(ISDN)卡、电缆调制解调器、卫星调制解调器、或用于提供与对应类型的电话线之间的数据通信连接的调制解调器。作为另一个示例，通信接口可以是局域网络(LAN)卡，以提供与兼容的LAN之间的数据通信连接。还可以实施无线链路。在任何这样的实施方式中，通信接口发送和接收携带代表各种类型的信息的数字数据流的电气、电磁或光学信号。

[0074] 如图5所示，存储设备可以包含子网管理器，其可以被配置为在网络构架中将末端节点与叶交换机上的交换端口相关联，其中末端节点与多个端口相关联。子网管理器可以进一步被配置为针对末端节点上的每个所述端口执行路由，并且确保在末端节点上的多个端口取相互独立的路径。

[0075] 在一些实施例中，子网管理器可以进一步被配置为在与末端节点上的多个端口中的一个端口相关联的路径上标记每个交换机，并且防止与末端节点上的多个端口中的另一个端口相关联的另一个路径使用被标记的交换机。

[0076] 在一些实施例中，子网管理器可以进一步被配置为，在不存在冗余交换机时允许与末端节点上的多个端口中的另一个端口相关联的另一个路径可以使用一个或多个被标记的交换机，并且当并行链路存在于一个或多个被标记的交换机时，针对相同的末端节点上的不同目标节点选择独立的链路。

[0077] 在一些实施例中，子网管理器可以进一步被配置为在完成对末端节点上的多个端口的路由之后，取消对每个被标记的交换机的标记。

[0078] 在一些实施例中，子网管理器可以进一步被配置为在完成对末端节点上的多个端口的路由之后，将末端节点标记为已路由的末端节点。

[0079] 在一些实施例中，子网管理器可以进一步被配置为当末端节点遭遇另一个叶交换机时，防止末端节点再次被路由。

[0080] 图6说明了用于实施本发明的实施例的系统600的框图。系统600中的块可以由硬件、软件或硬件和软件的组合实施，以实现本发明的原理。本领域的技术人员可以理解的是，图6中描述的块可以被组合或分离为子块，以如上所述实施本发明的原理。因此，本文中的描述可以支持本文描述的功能性块的任何可能的组合或分离或进一步的限定。

[0081] 如图6所示，系统600可以包括提供单元601，提供单元601被配置为提供与在网络

构架中的叶交换机上的交换端口相关联的末端节点，其中末端节点与多个端口相关联。系统600可以进一步包括执行单元602，执行单元602被配置为针对末端节点上的每个所述端口执行路由。另外，系统600可以进一步包括确保单元603，确保单元603被配置为确保在末端节点上的多个端口取相互独立的路径。

[0082] 虽然在图6中未示出，但在一些实施例中，系统600还可以包括被配置为允许网络构架可以基于胖树拓扑的第一允许单元。

[0083] 在一些实施例中，系统600还可以包括第二允许单元，第二允许单元被配置为允许末端节点可以是通过多端口连接到胖树拓扑的两个或更多个部分的多宿主节点。

[0084] 在一些实施例中，系统600还可以包括第一标记单元，其被配置为标记在与末端节点上的多个端口中的一个端口相关联的路径上的每个交换机，以及还可以包括第一防止单元，其被配置为防止与末端节点上的多个端口中的另一个端口相关联的另一个路径使用被标记的交换机。

[0085] 在一些实施例中，系统600还可以包括第三允许单元，其被配置为当不存在冗余交换机时，允许与末端节点上的多个端口中的另一个端口相关联的另一个路径可以使用一个或多个被标记的交换机，还可以包括选择单元，其被配置为当并行链路存在于一个或多个被标记的交换机时，针对在相同的末端节点上的不同目标端口选择独立的链路。

[0086] 在一些实施例中，系统600还可以包括取消标记单元，其被配置为在完成对末端节点上的多个端口的路由之后，取消对每个被标记的交换机的标记。

[0087] 在一些实施例中，系统600还可以包括关联单元，其被配置为将从末端节点出发的每个相互独立的路径与不同的主干交换机关联起来。

[0088] 在一些实施例中，系统600还包括第二标记单元，其被配置为在完成对末端节点上的多个端口的路由之后，将末端节点标记为已路由末端节点。

[0089] 在一些实施例中，系统600还可以包括第二防止单元，其被配置为当末端节点遭遇另一个叶交换机时，防止末端节点再次被路由。

[0090] 在一些实施例中，系统600还可以包括第四允许单元，其被配置为允许路由算法选取末端节点作为参数。

[0091] 用于支持网络环境中的多宿主路由的装置，包括提供与网络构架中的叶交换机上的交换端口相关联的末端节点，其中末端节点与多个端口相关联；针对末端节点上的每个所述端口执行路由；以及确保在末端节点上的多个端口取相互独立的路径以用于所述路由。

[0092] 用于支持网络环境中的多宿主路由的装置，其中网络构架基于胖树拓扑。

[0093] 用于支持网络环境中的多宿主路由的装置，其中末端节点是通过多个端口与胖树拓扑中的两个或更多个部分相连接的多宿主节点。

[0094] 用于支持网络环境中的多宿主路由的装置还包括，标记在与末端节点上的多个端口中的一个端口相关联的路径上的每个交换机，以及防止与末端节点上的多个端口中的另一个端口相关联的另一个路径使用被标记的交换机。

[0095] 用于支持网络环境中的多宿主路由的装置还包括，当不存在冗余交换机时，允许与末端节点上的多个端口中的另一个端口相关联的另一个路径可以使用一个或多个被标记的交换机，以及当并行链路存在于一个或多个被标记的交换机时，针对在相同的末端节

点上的不同目标端口选择独立的链路。

[0096] 用于支持网络环境中的多宿主路由的装置还包括，在完成对末端节点上的多个端口的路由之后，取消对每个被标记的交换机的标记。

[0097] 用于支持网络环境中的多宿主路由的装置还包括，将从末端节点出发的每个相互独立的路径与不同的主干交换机关联起来

[0098] 用于支持网络环境中的多宿主路由的装置还包括，在完成对末端节点上的多个端口的路由之后，将末端节点标记为已路由末端节点。

[0099] 用于支持网络环境中的多宿主路由的装置还包括，当末端节点遭遇另一个叶交换机时，防止末端节点再次被路由。

[0100] 用于支持网络环境中的多宿主路由的装置还包括，使用选取末端节点作为参数的路由算法。

[0101] 可以通过使用一个或多个常规的通用或专有的数字计算机、计算设备、机器、或微处理器，包括根据本公开的教导被编程的一个或多个处理器、存储器和/或计算机可读存储介质，方便地实施本发明。对于软件领域的技术人员来说清晰的是，基于本公开的教导，熟练的程序员可以容易地准备合适的软件编码。

[0102] 在一些实施例中，本发明包括计算机程序产品，其是在其上/其中存储有指令的存储介质或计算机可读介质，所述指令可以被用于对计算机编程以执行本发明的进程中的任何进程。存储介质可以包括但不限于任何类型的盘，包括软盘、光盘、DVD、CD-ROM、微驱动器、以及磁光盘、ROM、RAM、EPROM、EEPROM、DRAM、VRAM、闪速存储器设备、磁卡或光卡、纳米系统(包括分子存储IC)、或任何类型的适合于存储指令和/或数据的介质或设备。

[0103] 为了说明和描述的目的，以上已提供了对本发明的描述。不旨在穷尽或将发明限制到所公开的精确形式中。许多修改和变化对于本领域从业人员来说将是清晰的。修改和变化包括对所描述的特征的任何相关的组合。实施例被选择和描述，以便更好地解释本发明的原理和实际应用，从而使得本领域其他技术人员可以根据各种实施例和适合于所期望的特定使用的各种修改理解本发明。旨在由以下的权利要求和权利要求的等效物限定本发明的范围。

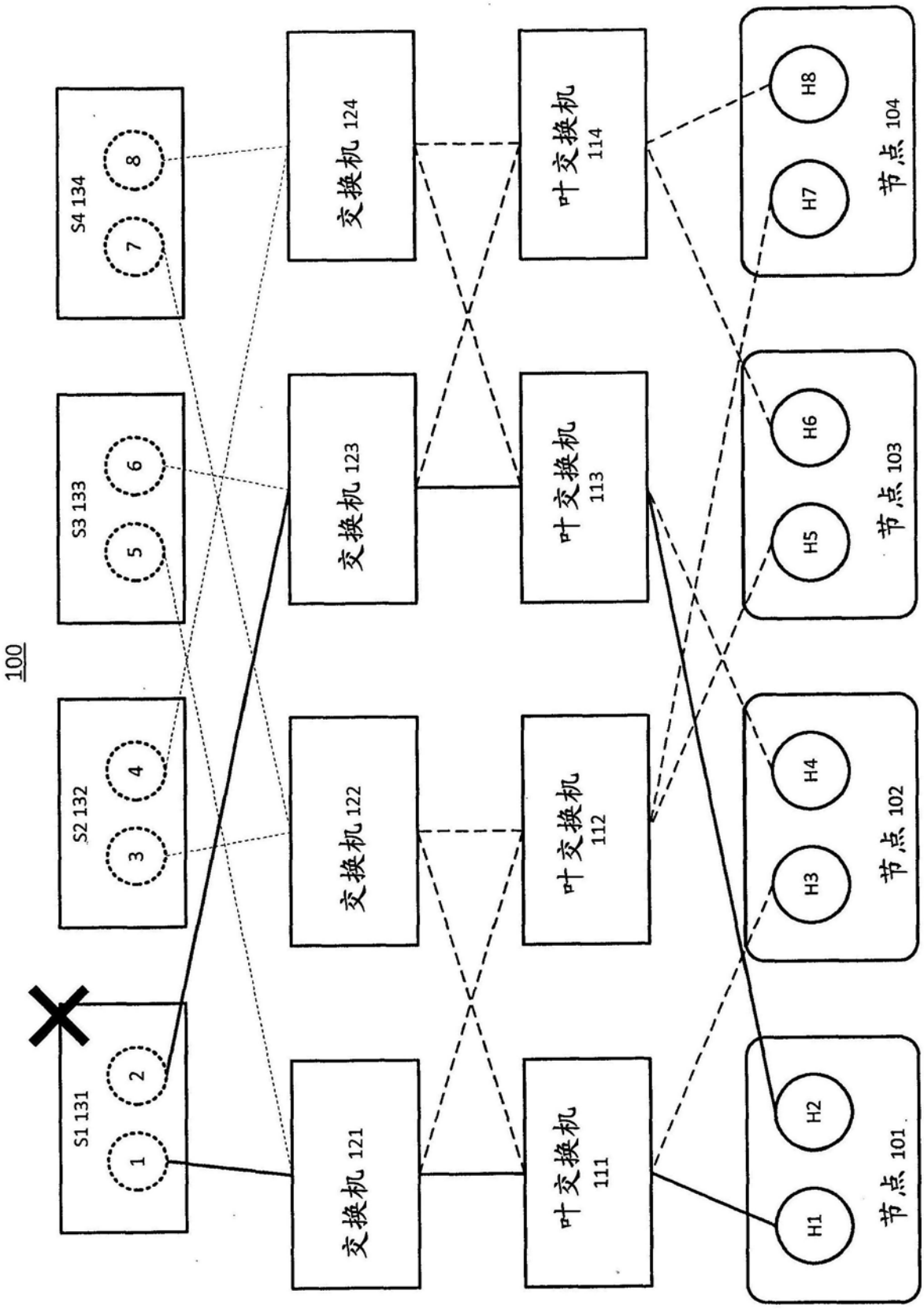

100
S1 131
S2 132
S3 133
S4 134
1
2
3
4
5
6
7
8
交换机 121
交换机 122
交换机 123
交换机 124
叶交换机 111
叶交换机 112
叶交换机 113
叶交换机 114
节点 101
节点 102
节点 103
节点 104
H1
H2
H3
H4
H5
H6
H7
H8

图1

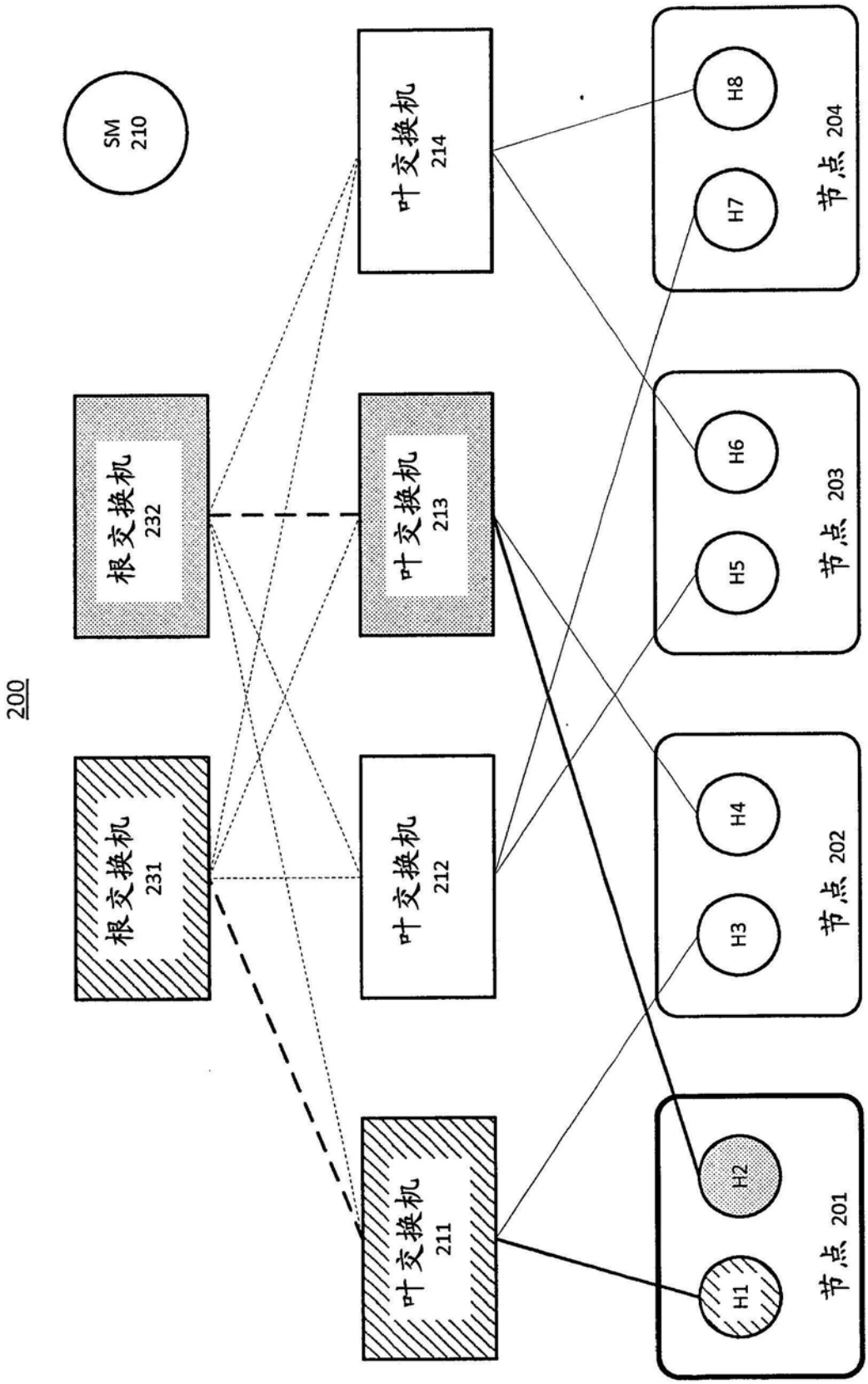

200
SM
210
根交换机
232
根交换机
231
叶交换机
214
叶交换机
213
叶交换机
212
叶交换机
211
H8
H7
节点 204
H6
H5
节点 203
H4
H3
节点 202
H2
H1
节点 201

图2

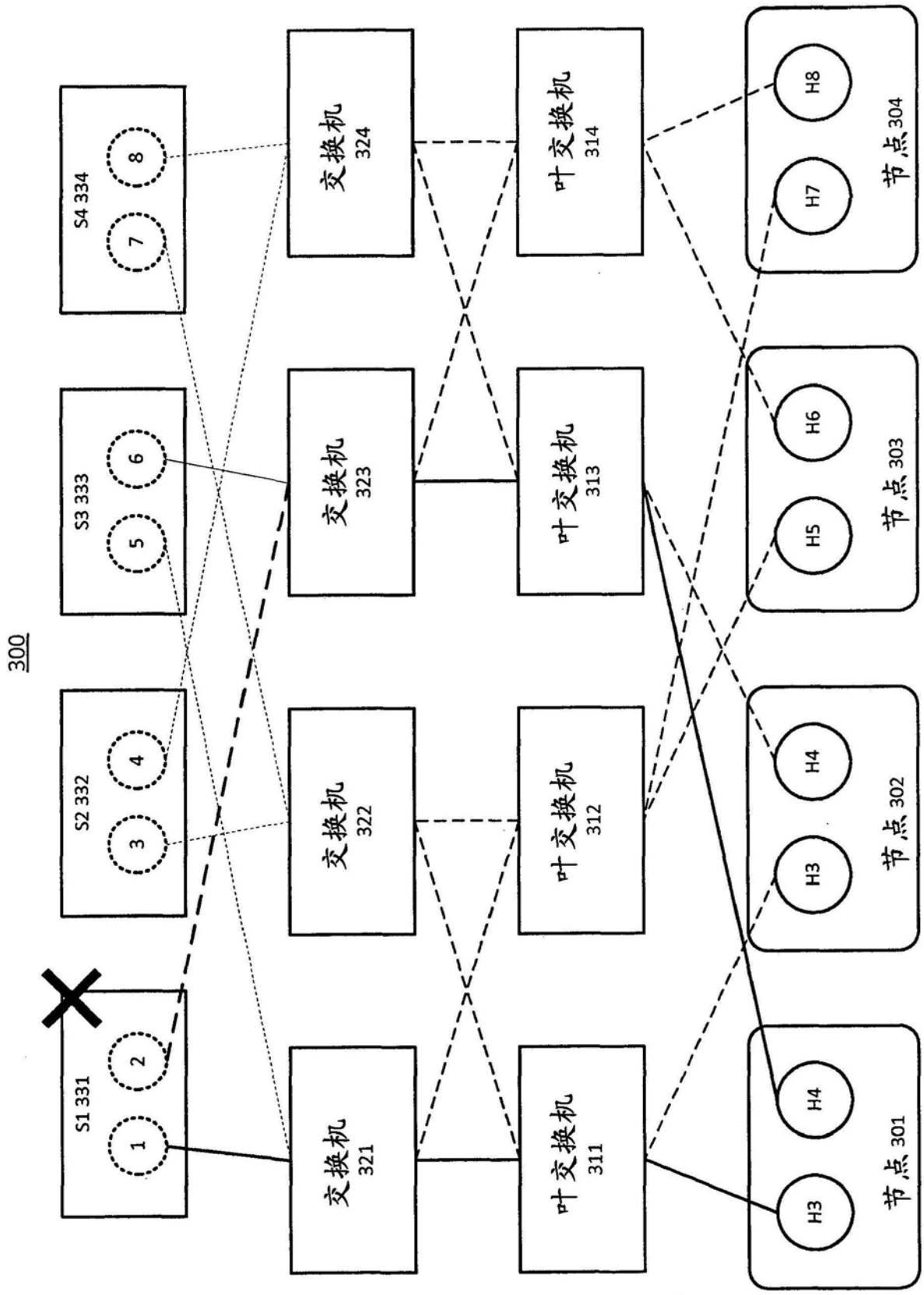

300
S1 331
1
2
S2 332
3
4
S3 333
5
6
S4 334
7
8
交换机 321
交换机 322
交换机 323
交换机 324
叶交换机 311
叶交换机 312
叶交换机 313
叶交换机 314
节点 301
H3
H4
节点 302
H3
H4
节点 303
H5
H6
节点 304
H7
H8

图3

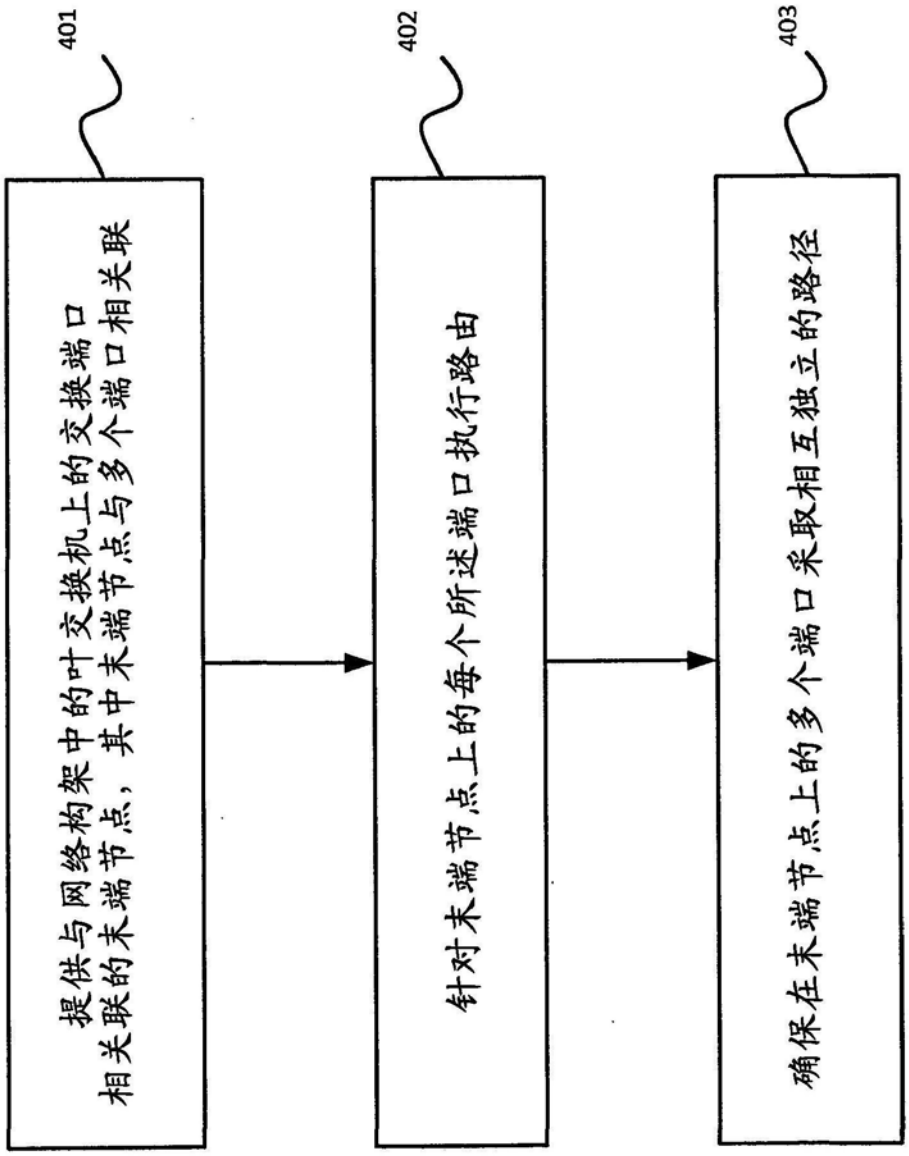

401
提供与网络构架中的叶交换机上的交换端口相关联的末端节点，其中末端节点与多个端口相关联
402
针对末端节点上的每个所述端口执行路由
403
确保在末端节点上的多个端口采取相互独立的路径

图4

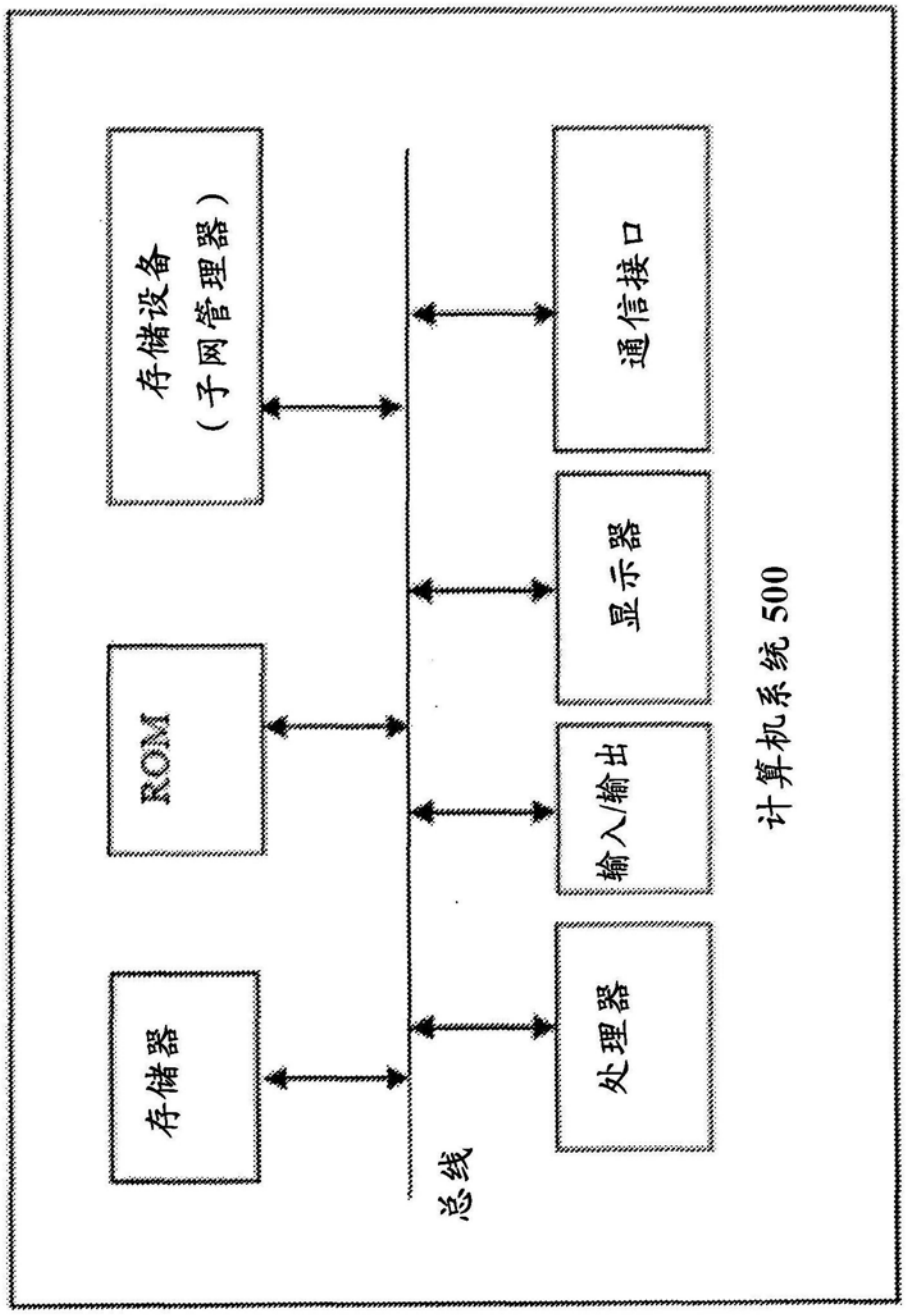


图5

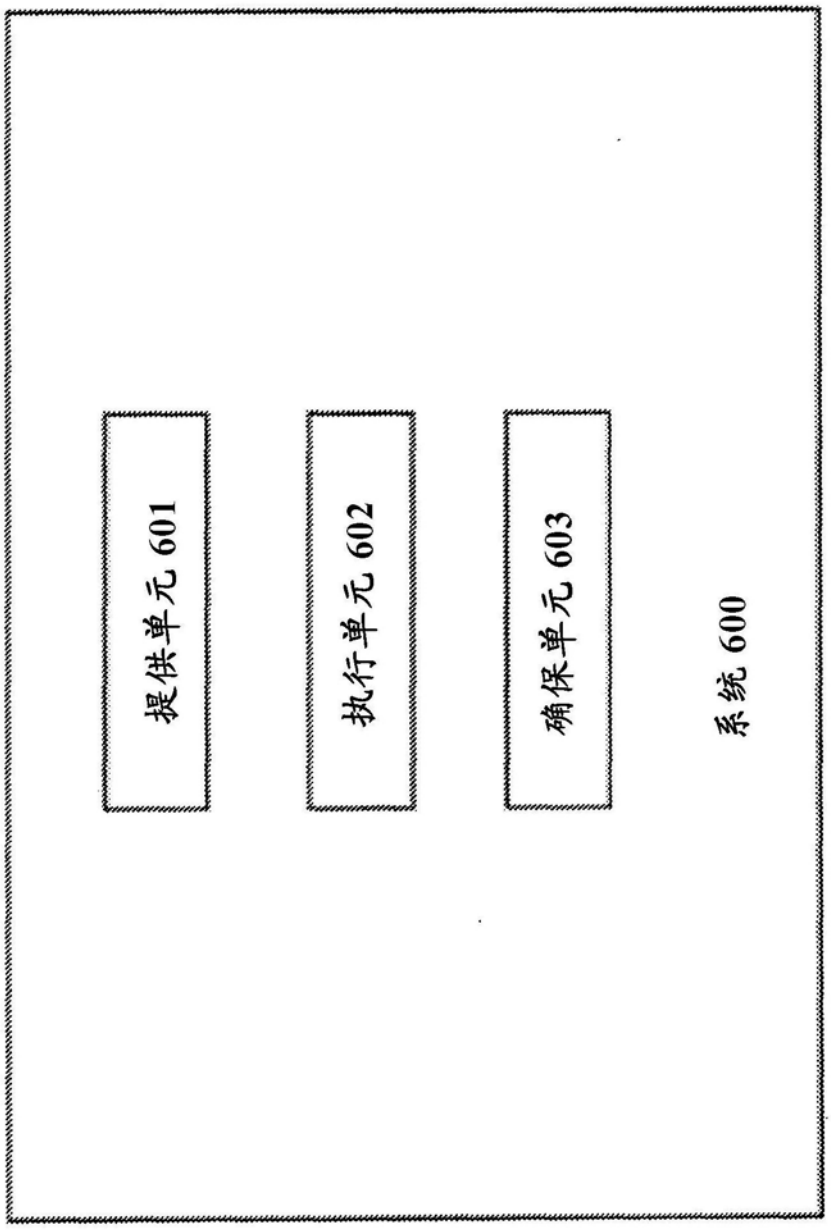


图6