(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 103907322 B
(45)授权公告日 2017.01.18

(21)申请号 201280052888.2

(22)申请日 2012.11.09

(65)同一申请的已公布的文献号
申请公布号 CN 103907322 A

(43)申请公布日 2014.07.02

(30)优先权数据
61/557,722 2011.11.09 US
13/653,303 2012.10.16 US

(85)PCT国际申请进入国家阶段日
2014.04.28

(86)PCT国际申请的申请数据
PCT/US2012/064479 2012.11.09

(87)PCT国际申请的公布数据
WO2013/071130 EN 2013.05.16

(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

(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 冯玉清

(51)Int.Cl.
H04L 12/753(2006.01)

(56)对比文件
US 6931103 B1,2005.08.16,
US 2008239983 A1,2008.10.02,
Frank Olaf Sem-Jacobsen etc..Dynamic Fault Tolerance in Fat Trees.《IEEE TRANSACTIONS ON COMPUTERS》.2011,第60卷(第4期),508-525.
Frank Olaf Sem-Jacobsen etc..Dynamic Fault Tolerance in Fat Trees.《IEEE TRANSACTIONS ON COMPUTERS》.2011,第60卷(第4期),508-525.

审查员 尤一名

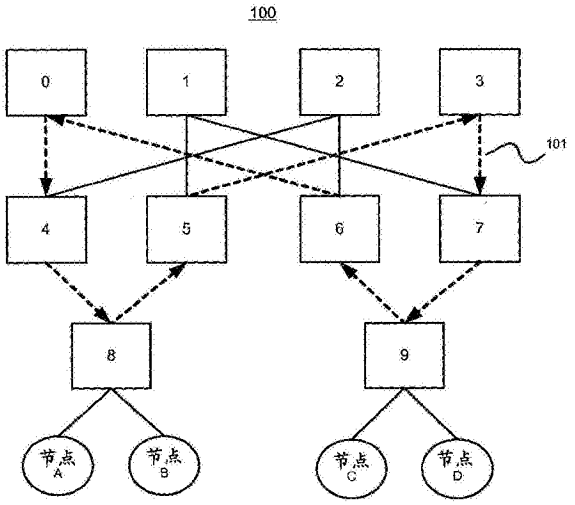
权利要求书3页 说明书10页 附图5页

(54)发明名称

用于提供胖树拓扑中的交换机之间的无死锁路由的系统和方法

(57)摘要

一种系统和方法可以支持在中间件机器环境中的多个交换机之间路由数据包,从而通过在中间件机器环境中启用IP over Infiniband(IPoIB)通信来支持基于互联网协议(IP)的管理业务。所述多个交换机可以使用路由算法来对中间件机器环境中的交换机间业务执行路由。然后,可以选择中间件环境中的交换机作为用于不能使用路由算法到达其目的地的交换机间业务的集线器交换机。此外,当经由集线器交换机在源交换机与目的地交换机之间存在路径时,可以更新与集线器交换机相关联的路由表。



CN 103907322 B

1.一种用于支持操作于一个或多个微处理器上的中间件机器环境中的多个交换机之间的路由的方法,包括:

使用路由算法对所述中间件机器环境中的交换机间业务执行路由,其中所述中间件机器环境包括多个叶交换机;

遍历所述多个叶交换机以确定特定叶交换机直接或者间接连接到所述中间件机器环境中的每个交换机;

选择所述特定叶交换机作为集线器交换机以用于不能使用所述路由算法到达其目的地的交换机间业务;以及

当经由所述集线器交换机在源交换机与目的地交换机之间存在路径时,更新与所述集线器交换机相关联的路由表。

2.根据权利要求1所述的方法,还包括:

使所述中间件机器环境中的所述多个交换机处于胖树拓扑中。

3.根据权利要求2所述的方法,还包括:

使用所述路由算法中的向上/向下拐弯模型来路由所述胖树拓扑中的交换机间业务。

4.根据权利要求3所述的方法,还包括:

当所述向上/向下拐弯模型失败时,使所述交换机间业务进行向下/向上拐弯,并且将所有的向下/向上拐弯局部化到其中所述集线器交换机作为子树根交换机的单个无死锁子树,以便防止所述胖树拓扑中的死锁。

5.根据权利要求1至4中的任何一个所述的方法,还包括:

当到所述目的地交换机的路径不存在时,检查所述集线器交换机是否具有兄弟交换机,并且检查与所述兄弟交换机相关联的路由表是否包含所述路径。

6.根据权利要求4所述的方法,还包括:

当到所述目的地交换机的路径不存在时,检查所述集线器交换机是否具有兄弟交换机,并且检查与所述兄弟交换机相关联的路由表是否包含所述路径;以及

当所述兄弟交换机存在并且与所述兄弟交换机相关联的路由表包含到所述目的地交换机的路径时,在所述源交换机和所述子树根交换机上设置通过所述兄弟交换机到所述目的地交换机的路径。

7.根据权利要求1至4中的任何一个所述的方法,还包括:

使用于所述目的地交换机的输出端口与用于所述集线器交换机的输出端口相同。

8.根据权利要求1至4中的任何一个所述的方法,还包括:

使用递推函数用于交换机间路由中的跳跃计算,其中所述递推函数在构成所述路径的一个或多个交换机上迭代。

9.一种用于支持中间件机器环境中的交换机间业务路由的系统,包括:

多个交换机,所述多个交换机运行于一个或多个微处理器上,其中所述多个交换机操作为执行以下步骤:

使用路由算法来对所述中间件机器环境中的交换机间业务执行路由,其中所述中间件机器环境包括多个叶交换机;

遍历所述多个叶交换机以确定特定叶交换机直接或者间接连接到所述中间件机器环境中的每个交换机;

选择所述特定叶交换机作为集线器交换机以用于不能使用所述路由算法到达其目的地的交换机间业务;以及

当经由所述集线器交换机在源交换机与目的地交换机之间存在路径时,更新与所述集线器交换机相关联的路由表。

10.一种用于支持中间件机器环境中的交换机间业务路由的系统,所述系统包括多个交换机,所述多个交换机中的至少一个交换机包括:

路由单元,配置为使用路由算法来对所述中间件机器环境中的交换机间业务执行路由,其中所述中间件机器环境包括多个叶交换机;

迭代单元,配置为遍历所述多个叶交换机以确定特定叶交换机直接或者间接连接到所述中间件机器环境中的每个交换机;

选择单元,配置为选择所述特定叶交换机作为集线器交换机以用于不能使用所述路由算法到达其目的地的交换机间业务;以及

更新单元,配置为当经由所述集线器交换机在源交换机与目的地交换机之间存在路径时,更新与所述集线器交换机相关联的路由表。

11.根据权利要求10所述的系统,其中,所述中间件机器环境中的所述多个交换机处于胖树拓扑中。

12.根据权利要求11所述的系统,其中,所述路由算法使用向上/向下拐弯模型来路由所述胖树拓扑中的交换机间业务。

13.根据权利要求12所述的系统,其中,当所述向上/向下拐弯模型失败时,所述交换机间业务进行向下/向上拐弯,并且所有的向下/向上拐弯都被局部化到其中所述集线器交换机作为子树根交换机的单个无死锁子树,以便防止所述胖树拓扑中的死锁。

14.根据权利要求10至13中的任何一个所述的系统,其中,所述至少一个交换机还包括:

检查单元,配置为当到所述目的地交换机的路径不存在时,检查所述集线器交换机是否具有兄弟交换机,并且检查与所述兄弟交换机相关联的路由表是否包含所述路径。

15.根据权利要求13所述的系统,其中,所述至少一个交换机还包括:

检查单元,配置为当到所述目的地交换机的路径不存在时,检查所述集线器交换机是否具有兄弟交换机,并且检查与所述兄弟交换机相关联的路由表是否包含所述路径;以及

路径设置单元,配置为当所述兄弟交换机存在并且与所述兄弟交换机相关联的路由表包含到所述目的地交换机的路径时,在所述源交换机和所述子树根交换机上设置通过所述兄弟交换机到所述目的地交换机的路径。

16.根据权利要求10至13中的任何一个所述的系统,其中,所述至少一个交换机还包括:

端口设置单元,配置为将用于所述目的地交换机的输出端口设置为与用于所述集线器交换机的输出端口相同。

17.一种用于支持操作于一个或多个微处理器上的中间件机器环境中的多个交换机之间的路由的设备,包括:

用于使用路由算法来对所述中间件机器环境中的交换机间业务执行路由的装置,其中所述中间件机器环境包括多个叶交换机;

用于遍历所述多个叶交换机以确定特定叶交换机直接或者间接连接到所述中间件机器环境中的每个交换机的装置；

用于选择所述特定叶交换机作为集线器交换机以用于不能使用所述路由算法到达其目的地的交换机间业务的装置；以及

用于当经由所述集线器交换机在源交换机与目的地交换机之间存在路径时，更新与所述集线器交换机相关联的路由表的装置。

18.根据权利要求17所述的设备，还包括：

用于使所述中间件机器环境中的所述多个交换机处于胖树拓扑中的装置。

19.根据权利要求18所述的设备，还包括：

用于使用所述路由算法中的向上/向下拐弯模型来路由所述胖树拓扑中的交换机间业务的装置。

20.根据权利要求19所述的设备，还包括：

用于当所述向上/向下拐弯模型失败时，使所述交换机间业务进行向下/向上拐弯的装置，以及用于将所有的向下/向上拐弯都局部化到其中所述集线器交换机作为子树根交换机的单个无死锁子树，以防止所述胖树拓扑中的死锁的装置。

21.根据权利要求17至20中的任何一个所述的设备，还包括：

用于当到所述目的地交换机的路径不存在时，检查所述集线器交换机是否具有兄弟交换机，并且检查与所述兄弟交换机相关联的路由表是否包含所述路径的装置。

22.根据权利要求20所述的设备，还包括：

用于当到所述目的地交换机的路径不存在时，检查所述集线器交换机是否具有兄弟交换机，并且检查与所述兄弟交换机相关联的路由表是否包含所述路径的装置；以及

用于当所述兄弟交换机存在并且与所述兄弟交换机相关联的路由表包含到所述目的地交换机的路径时，在所述源交换机和所述子树根交换机上设置通过所述兄弟交换机到所述目的地交换机的路径的装置。

23.根据权利要求17至20中的任何一个所述的设备，还包括：

用于使用于所述目的地交换机的输出端口与用于所述集线器交换机的输出端口相同的装置。

用于提供胖树拓扑中的交换机之间的无死锁路由的系统和方法

[0001] 版权声明

技术领域

[0003] 本发明总体上涉及计算机系统，更特别地，涉及支持中间件机器环境中的交换机对交换机通信。

背景技术

[0004] 胖树拓扑用于如今的高性能计算(HPC)集群，并且用于基于InfiniBand(IB)技术的集群。对于胖树，与大多数其他技术一样，路由算法对于网络资源的高效率使用是有益的。然而，当涉及交换机对交换机通信时，现有的路由算法具有限制。没有一种现有的路由算法支持对于高效率系统管理有益的无死锁、全连接的交换机对交换机通信。这些大致是本发明的实施例意图解决的领域。

发明内容

[0005] 本文中描述可以支持在中间件机器环境下的多个交换机之间路由数据包的系统和方法。中间件机器环境可以通过在中间件机器环境下启用IP over Infiniband(IPoIB)通信来支持基于互联网协议(IP)的管理业务。所述多个交换机可以使用路由算法来对中间件机器环境下的交换机间业务执行路由。然后，可以选择中间件机器环境下的交换机作为用于使用路由算法不能到达目的地的交换机间业务的集线器交换机。此外，当经由集线器交换机在源交换机与目的地交换机之间存在路径时，可以更新与集线器交换机相关联的路由表。

[0006] 在本发明的示例性实施例中，提供一种用于支持中间件机器环境下的交换机间业务路由的系统。该系统可以包括多个交换机，其中至少一个可以包括路由单元、选择单元和更新单元。路由单元可以使用第一路由算法来对中间件机器环境下的交换机间业务执行路由；选择单元可以选择中间件机器环境下的交换机作为用于使用第一路由算法不能到达目的地的交换机间业务的集线器交换机；当经由集线器交换机在源交换机与目的地交换机之间存在路径时，更新单元可以用于更新与集线器交换机相关联的路由表。

[0007] 在实施例中，所述多个交换机可以形成胖树拓扑。选择单元可以选择胖树拓扑中的叶交换机作为集线器交换机，并且该叶交换机可以到达胖树拓扑中的多个交换机目的地。在胖树拓扑中，第一路由算法可以使用向上/向下拐弯(turn)模型来路由交换机间业务。当向上/向下拐弯模型失败时，交换机间业务向下/向上拐弯，并且所有向下/向上拐弯都被局域化到其中集线器交换机作为子树根节点的单个无死锁子树，以便防止胖树拓扑中

的死锁。

[0008] 在另一实施例中，所述多个交换机中的所述至少一个还可以包括迭代单元，其迭代经过胖树拓扑中的多个叶交换机，以便找到可以到达胖树拓扑中的多个交换机目的地的一个或多个叶交换机。在例子中，所述多个交换机中的所述至少一个还可以包括检查单元。检查单元可以检查集线器交换机是否具有兄弟交换机，并且当至目的地交换机的路径不存在时，检查与所述兄弟相关联的路由表是否包含所述路径。在例子中，所述多个交换机中的所述至少一个还可以包括路径设置单元，其用于当所述兄弟交换机存在并且与所述兄弟交换机相关联的路由表包含到目的地交换机的路径时，在源交换机和子树根交换机上设置通过所述兄弟交换机到目的地交换机的路径。在另一个例子中，所述多个交换机中的所述至少一个还可以包括端口设置单元，其用于将用于目的地交换机的输出端口设置为与用于集线器交换机的输出端口相同。

附图说明

[0009] 图1示出当在中间件机器环境下发生死锁时的示例性场景的例示。

[0010] 图2示出根据本发明的实施例的提供单核胖树拓扑中的交换机之间的无死锁路由的例示。

[0011] 图3示出根据本发明的实施例的提供多核胖树拓扑中的交换机之间的无死锁路由的例示。

[0012] 图4例示根据本发明的实施例的用于提供胖树拓扑中的交换机之间的无死锁路由的示例性流程图。

[0013] 图5例示根据本发明的交换机的示例性实施例。

具体实施方式

[0014] 本文中描述可以支持在中间件机器环境下的多个开关之间路由数据包的系统和方法。中间件机器环境可以通过在中间件机器环境下启用IP over Infiniband(IPoIB)通信来支持基于互联网协议(IP)的管理业务。

[0015] InfiniBand(IB)网络和子网管理

[0016] 根据本发明的实施例，IB网络可以被称为子网，其中子网包括使用交换机和点对点链路互连的一组主机。IB交织结构(fabric)可以构成一个或多个子网，其中每个子网可以使用路由器互连。使用本地标识符(LID)来寻址子网内的主机和交换机，并且单个子网限于49151个LID。

[0017] IB子网可以具有至少一个子集管理器(SM)，其负责初始化和启动网络，包括驻留在子网中的交换机、路由器和主机信道适配器(HCA)上的所有IB端口的配置。在初始化时，SM从发现状态开始，在发现状态下，SM对网络进行扫描以发现所有交换机和主机。在发现状态期间，SM可以发现其他SM，并且可以就谁应是主控SM进行商议。当发现状态完成时，SM进入主控状态。在主控状态下，SM进行LID分配、交换机配置、路由表计算和部署以及端口配置。主控状态一完成，子网就启动并且准备好供使用，并且在子网已经被配置之后，SM负责监视网络变化。

[0018] 另外，SM可以负责计算保持所有的源和目的地对之间的全连接、无死锁和适当负

荷均衡的路由表。可以在网络初始化时计算路由表，并且每当拓扑改变时可以重复该处理以便更新路由表并且确保最佳性能。

[0019] 在正常操作期间，SM执行网络的周期性过光扫描(light sweep)，以检查拓扑改变事件，诸如当链路发生故障时、当装置被添加时、或者当链路被移除时。如果在过光扫描期间发现改变事件，或者如果SM接收到通知网络改变的消息(陷阱)，则SM可以根据所发现的改变来重新配置网络。该重新配置也可以包括在初始化期间使用的步骤。

[0020] IB是无损联网技术，其中可以对每一虚拟通道(VL)执行流控制(flow-control)。VL是同一物理链路上具有独立的缓冲、流控制和拥塞管理资源的逻辑信道。VL的概念使得可以在物理拓扑上构建虚拟网络。这些虚拟网络或层可以用于各种目的，诸如高效率路由、死锁规避、容错性和服务差异化。

[0021] 胖树路由

[0022] 根据本发明的实施例，胖树拓扑是分层网络拓扑，例如，均衡的胖树可以在每一层都具有相等的链路容量。此外，可以通过构建具有多个根的树(例如可以使用XGFT表示法描述的m端口n树定义或k-ary n树定义)来实现胖树拓扑。

[0023] 胖树路由算法可以利用可用网络资源。胖树路由算法可以包括两个阶段：从源转发数据包的向上阶段、以及当朝向目的地转发数据包时的向下阶段。这两个阶段之间的过渡发生在最近公共祖先处，最近公共祖先是可以通过其向下端口到达源和目的地两者的交换机。这样的路由实现确保无死锁，并且该实现还确保朝向同一目的地的每一个路径会聚在同一根(顶部)节点处，使得朝向该目的地的所有数据包在向下方向上遵循单个专用路径。通过具有用于每个目的地的专用向下路径，有效地去除了向下阶段中的争用(移至向上阶段)，使得用于不同目的地的数据包在其路径上仅争用一半交换机中的输出端口。另外，超额预订的胖树中的向下路径不是专用的，可以由几个目的地共享。

[0024] 交换机对交换机通信

[0025] 除了终端节点之间的通信之外，胖树拓扑允许交换机对交换机通信。当交换机对交换机通信发起很少的业务时，可以忽略交换机对交换机通信，或者换句话讲，可以认为IB交换机是从路由算法的观点来讲可以安全地忽略的透明装置。

[0026] 另一方面，交换机对交换机通信业务可能不总是可以忽略的。在这些情况下，可以使用前一章节中描述的胖树路由来带有局限性地路由交换机对交换机通信业务，因为可能需要找到最近公共祖先以发现任意两个交换机之间的路径，使得可以通过第一可用端口来转发业务。而且，胖树路由方案不提供不具有最近公共祖先的那些交换机之间的连接，即，前一章节中描述的胖树路由方案可能不能提供胖树拓扑中的全连接。

[0027] 由于各种原因，包括终端节点和交换机两者的胖树拓扑中的全连接可以是有益的。例如，IB诊断工具依赖于LID路由，并且需要全连接来进行基本的交织结构管理和监视。此外，诸如用于基准检查(benchmarking)的perftest之类的高级工具可以利用IPoIB，其依赖于底层LID路由并且需要全连接。IPoIB是使得可以通过IB网络运行TCP/IP业务的封装方法。此外，IPoIB也是非InfiniBand知悉的管理以及像SNMP或SSH之类的监视协议和应用所需要的。因此，交织结构中的所有交换机之间的全连接可以是有益的，因为交换机能够产生任意业务，可以像任何其他终端节点那样被访问，并且通常包含嵌入式SM。

[0028] 因为诸如缓冲区或信道的网络资源被共享，所以在胖树拓扑中可能发生死锁。

[0029] 图1示出当在中间件机器环境下发生死锁时的示例性场景的例示。如图1所示，中间件机器环境下的胖树拓扑100允许具有两个交换机对交换机通信对(交换机0对交换机3、以及交换机3对交换机6)和两个节点对节点通信对(节点B对节点D、以及节点D对节点A)的混合通信模式。这里，从交换机0到交换机3的业务经由交换机4、8和5引导，从交换机3到交换机6的业务经由交换机7和9引导。此外，从节点B到节点D的业务经由交换机8、5、3、7和9引导，从节点D到节点A的业务经由交换机9、6、0、4和8引导。

[0030] 如图1中的虚线所示，当以上混合通信模式存在于胖树拓扑100中时，创建了循环信用依赖(cycle credit dependency)或信用回路101。此外，当创建循环信用依赖时，死锁可能发生。这不意味着，每当存在循环信用依赖时，就将总是存在死锁。换句话讲，循环信用依赖的创建使得死锁发生成为可能。例如，当使用诸如minhop之类的路由算法时，在具有节点对节点和交换机对交换机业务的3级胖树中，可能发生死锁。这是因为minhop不能以无死锁的方式求解5个节点的环或更大的环，并且3级胖树可能包含6节点的环。因此，需要一种当存在所有交换机之间的交换机对交换机通信时可以提供胖树拓扑中的无死锁路由的算法。

[0031] 可以通过使用分割可用物理资源的VL来避免死锁。然而，使用VL来实现胖树中的死锁避免可能是无效率的，因为附加VL可以用于其他目的，比如，服务质量(QoS)。

[0032] 例如，用于IB的LASH算法可以支持全连接和无死锁。LASH算法使用VL来打破信用回路101，并且提供交织结构中的所有节点之间的全连接。LASH在分配路径时不利用胖树的性质，因此使得网络性能可能比普通胖树路由更糟，另外，LASH的路由时间长。DFSSSP是可以用于支持胖树中的所有交换机对交换机通信的另一路由协议。然而，DFSSSP不打破非HCA节点之间的信用回路，这意味着交换机对交换机通信可能不是无死锁的。

[0033] 因此，当在互连网络中发生死锁时，死锁可以阻止网络的部分或全部通信。因此，从互连网络的全系统角度来讲，胖树拓扑中所支持的全连接优选是无死锁的。

[0034] 单核胖树拓扑中的无死锁路由

[0035] 根据本发明的实施例，可以支持中间件机器环境下的节点之间的全连接，并且可以仅使用单个VL来实现整个交织结构的无死锁。

[0036] 图2示出根据本发明的实施例的提供单核胖树拓扑中的交换机之间的无死锁路由的例示。如图2所示，中间件机器环境下的单核胖树拓扑20可以包括多个交换机。此外，中间件机器环境可以包括连接到胖树拓扑200中的叶交换机202的多个终端节点220，诸如服务器节点。

[0037] 向上/向下拐弯模型可以用于提供中间件机器环境下的胖树路由。例如，交换机A1和A2可以通过借助于X1或X2上升而具有连接性。另一方面，对于不具有最近公共祖先的交换机(例如，A1和B1)之间的通信，向上/向下拐弯模型可能是无用的。

[0038] 根据本发明的实施例，可以利用替代方法来实现不能使用向上/向下拐弯模型建立通信的交换机之间的连接。如图2所示，可以在胖树拓扑200中创建倒置树或子树210(在阴影区域中示出)。子树210可以具有子树根节点Sn或集线器交换机210，其是胖树拓扑200中的叶交换机。可以通过胖树拓扑200中的集线器交换机201来建立全连接。

[0039] 在子树201内，交换机X1和Y1可以通过子树根Sn201来通信，X1和X2可以通过交换机A1来通信。此外，子树210外部的交换机可以通过子树210与胖树拓扑200中的其他交换机

通信，这导致非最短(或次优)路径的使用。所述交织结构中不能使用向上/向下拐弯模型到达另一个交换机的交换机可以首先将数据包发送到子树210。然后，可以在子树210中传递这些数据包，直到它们到达具有到目的地交换机的路径的交换机为止。换句话讲，数据包在集线器交换机201中做出U形拐弯，即，下到上拐弯，接着是朝向目的地的上到下的路径。中间件机器环境下的子树210可以将所述交织结构中的所有向下/向上拐弯都局部化到单个无死锁树。

[0040] 例如，当接收到在中间件机器环境下将一个或多个数据包从第一交换机(例如，A1)路由到第二交换机(例如，B2)的请求时，所述系统可以首先确定与子树根交换机或集线器交换机201相关联的路由表是否包含到第二交换机B2的路径。如图2中的虚线所示，集线器交换器201可以经由通过B1、Y1或Y2的路径连接到第二交换机B2。然后，可以将到第二交换机B2的路径插入到与第一交换机A1相关联的路由表。另外，在与A1相关联的路由表中，可以将用于交换机A2的输出端口设置为与用于集线器交换机210的输出端口相同。

[0041] 倒置子树210的根可以是叶交换机中的任何一个。此外，胖树拓扑200中的所有交换机可以共同选择用于通信的叶交换机，以实现中间件机器环境下的无死锁全连接。

[0042] 根据本发明的实施例，可以通过要求所有的U形拐弯仅发生在子树210内并且任何上到下拐弯都将引导业务离开子树210来在胖树拓扑200中实现无死锁。另外，离开子树210的业务可能不能循环回到子树210中，因为子树210外不允许U形拐弯。

[0043] 因此，在单核胖树200中，可以总是存在从源交换机连接到集线器交换机210、以及从集线器交换机201连接到每一个目的地交换机的路径。此外，用于通过拓扑中的叶交换机实现连接的通信方案不改变可以使用胖树路由到达彼此的交换机之间的向上到向下路径。

[0044] 附上了附录A，其提供关于支持中间件机器环境下的交换机之间的路由、以及整个本公开中所描述的平台的各个方面的更多信息。附录A中的信息是为了例示的目的而提供的，不应被解释为限制本发明的所有实施例。

[0045] 多核胖树拓扑中的无死锁路由

[0046] 根据本发明的实施例，在复杂的多核胖树或不规则胖树中的两个交换机之间不总是存在连接。在这些情况下可以利用尽力型(best-effort)方法来路由交换机对交换机通信。

[0047] 图3示出根据本发明的实施例的提供多核胖树拓扑中的交换机之间的无死锁路由的例示。如图3所示，中间件机器环境下的多核胖树拓扑300可以包括多个交换机。此外，中间件机器环境可以包括连接到胖树拓扑300中的叶交换机的多个终端节点320，诸如服务器节点。

[0048] 类似地，可以在胖树拓扑300中创建倒置树或子树310(在阴影区域中示出)。子树310可以具有子树根节点Sn或集线器交换机310，其是胖树拓扑300中的叶交换机。与图2不同，子树根交换机或集线器交换机301可能不具有到所有目的地的路径。

[0049] 另外，兄弟交换机302可以存在于中间件机器环境中，并且可以通过水平链路330与子树根交换机Sn或集线器交换机301连接。

[0050] 在一个例子中，第一交换机(例如，B2)可以请求在中间件机器环境下将一个或多个数据包路由到第二交换机(例如，B3)。子树根交换机301不具有连接到B3的路径，因为B3位于仅经由水平链路与其他节点连接的单独胖树中。

[0051] 所述系统可以首先检查子树根交换机301是否具有其路由表包含到B3的路径的兄弟交换机302。然后，所述系统可以在第一交换机B2和子树根交换机301上设置通过所述兄弟交换机302到第二交换机B3的路径。结果，所述系统可以被配置为例如经由Y1或Y2、B1、子树根节点301及其兄弟节点302来将从交换机B2发起的数据包路由到交换机B3(用虚线示出)。

[0052] SFTREE算法

[0053] 根据本发明的实施例，可以使用路由算法(即，SFTREE算法)来实现交换机之间的无死锁全连接。SFTREE算法可以包括两个步骤：第一步，发现子树的根Sn，其可以是叶交换机之一；以及第二步，执行交换机对交换机路由。

[0054] 下面的算法1包括用于找到子树的根并且建立子树的伪码。

[0055]

算法 1 选择子树根函数

要求：已经产生了路由表。

确保：子树根（sw_{root}）被选择。

```
1: found = false
2: for sw_leaf = 0 to max_leaf_sw do
3:   if found = = true then
4:     break
5:   end if
6:   sw_root = sw_leaf
7:   found = true
8:   for dst = 1 to max_dst_addr do
9:     if sw_root.routing_table[dst] = = no_path then
10:      found = false
11:      break
12:    end if
13:  end for
14: end for
15: if found = false then
16:   sw_root = get_leaf (0)
17: end if
```

[0056]

[0057] 算法1遍历拓扑中的所有叶交换机，并且对于每个叶交换机，检查在其路由表中是否将任何交换机目的地地址标记为不可到达。如果路由表中的所有交换机目的地都被标记为可到达，则将第一个遇到的叶交换机选择为子树根交换机。此外，在胖树中可以存在和具有全连接的叶交换机一样多的可能的子树。这些叶交换机中仅一个可以被选择为子树根，并且仅可以有一个根在这个特定叶交换机中的子树。

[0058] 下面的算法2包括用于执行胖树拓扑中的交换机对交换机路由的伪码。

算法 2 交换机对交换机路由函数

要求：子树根（sw_{root}）

确保：每个 sw_{src} 在最坏的情况下通过 sw_{root} 到达每个 sw_{dst}。

```
1: if found or sw_root.routing_table[dst] ≠ no_path then
2:   port = sw_src.routing_table[sw_root.addr]
3:   sw_src.routing_table[dst] = port
4:   get_path_length (sw_src, null, dst, sw_root , hops)
5:   set_hops (sw_src, hops)
6:   return true
7: else if (sw_sib = get_sibling_sw(sw_root)) ≠ null then
8:   if sw_src.routing_table[sw_sib.addr] ≠ no_path then
9:     if sw_sib.routing_table[dst] ≠ no_path then
10:      port = get_port_to_sibling (sw_sib)
11:      sw_root.routing_table[dst] = port
12:      hops = get_hops (sw_sib,dst)
13:      set_hops(sw_root, hops + 1)
14:      hops = 0
15:      port = sw_src.routing_table[sw_sib.addr]
16:      sw_src.routing_table[dst] = port
17:      get_path_length(sw_src, null, dst, sw_root, hops)
18:      set_hops(sw_src, hops)
19:      return true
20:    end if
21:  end if
22: end if
23: print 'SW2SW failed for sw_src and sw_dst.'
24: return false
```

[0059]

[0060]

[0061] 当交换机对交换机路由函数被调用时，如算法2中的行1所示，第一步是确定子树根（sw_{root}）的路由表是否包含到目的地交换机（sw_{dst}）的路径。如果包含，则将到目的地交换机的路径插入到源交换机（sw_{src}）的路由表中，并且用于目的地交换机的输出端口与用于到子树根的路径的输出端口相同（行2-6）。因为对于每一个交换机调用交换机对交换机路由函数，所以最后，到每个不可到达的目的地交换机的所有路径都将会聚到子树。换句话讲，对于所有不可到达的目的地交换机，所选子树根是新目标。向下/向上拐弯将在位于子树中的第一可能交换机处发生，该交换机可以是具有到源交换机和目的地交换机两者的向上路径的交换机。

[0062] 算法2的第二部分（行7-24）解决复杂的多核或不规则胖树的情况，其中，利用尽力型方法，并且子树根不具有到所有目的地的路径。另一方面，在单核胖树中可以跳过算法2的这个第二部分，因为可以总是存在从源交换机到子树根的路径，并且子树根可以具有到每一个目的地交换机的路径。

[0063] 算法2检查子树根是否具有它们如图3中所示的那样通过水平链路连接到的直接邻居（行7）。接着，算法2检查该邻居（也称为兄弟）是否具有到目的地交换机的路径（行8）。如果兄弟存在并且该兄弟具有到目的地交换机的路径，则在子树根和始发源交换机上都设置通过兄弟交换机到目的地交换机的路径（行9-19）。换句话讲，用于不可到达的目的地交换机的目标仍可以是子树根，并且子树根可以经由其兄弟交换机将数据包转发给目的地交换机。

[0064] 如果前两个步骤都没有从函数返回真值，则SFTREE算法可能未能找到两个交换机之间的路径。例如，对于由于多个链路失败或节点之间的非常不规则的连接而不推荐运行胖树路由的拓扑，这可能发生。这样的拓扑的例子可以是以非对称方式彼此连接(例如，从较大的树的几个中间级的交换机到较小的树上的根)的不同大小的两个胖树。通过使用OpenSM中的各种命令行参数，可以迫使在这些拓扑上运行胖树路由。SFTREE算法可以给予不仅就性能而言、而且还就连接而言次优的路由。

[0065] 根据本发明的实施例，OpenSM使得每一个路径可以用到目的地的跳数标记。胖树路由算法可以使用树中的交换机等级来计算跳数。此外，可以使用主路由函数中的简单计数器来进行计数。当交换机对交换机路由完全独立于通过胖树算法进行的主路由时，这些计数方案对于可能的z字形路径(即，不遵循最短跳跃路径的路径)可能是不可靠的。

[0066] 在交换机对交换机路由期间可以使用用于跳跃计算的递推函数，例如get_path_length(行17)。该函数可以在构成路径的一系列交换机上迭代，并且当该函数到达具有朝向目的地的适当跳数的节点时，该函数可以将该跳数写入到路径上的交换机的路由表中。

[0067] 另外，可以以子网管理器节点(子网管理器运行于其上的终端节点)不连接到被标记为子树根的交换机这样的方式来选择子树根。因此，可以将交换机对交换机业务拉离子网管理器，因此不在关键位置造成瓶颈。

[0068] 图4例示根据本发明的实施例的用于提供胖树拓扑中的交换机之间的无死锁路由的示例性流程图。如图4中所示，在步骤401，中间件机器环境可以使用路由算法对中间件机器环境下的交换机间业务执行路由。然后，在步骤402，中间件机器环境可以选择中间件机器环境下的交换机作为集线器交换机以用于不能使用路由算法到达目的地的交换机间业务。此外，在步骤403，当经由集线器交换机在源交换机与目的地交换机之间存在路径时，中间件机器环境可以更新与集线器交换机相关联的路由表。

[0069] 图5例示根据本发明的交换机的示例性实施例，该交换机可以包括在上述支持中间件机器环境下的交换机间业务路由并且包括多个交换机的系统中的任何系统中。参照图5，示例性交换机10可以包括根据如上所述的本发明的原理配置的路由单元11、选择单元13和更新单元14。可以用硬件、软件或硬件和软件的组合来实现交换机10的功能单元(包括但不限于单元11、13和14)以实现本发明的原理。本领域的技术人员理解，图5中所示的功能块均可以组合或分离为实现如上所述的本发明的原理的子单元。因此，本文中的描述可以支持本文中描述的功能块的任何可能的组合或分离或进一步定义。

[0070] 路由单元11可以使用如上所述的第一路由算法来对中间件机器环境下的交换机间业务执行路由。选择单元13可以选择中间件机器环境下的交换机作为用于不能使用第一路由算法到达目的地的交换机间业务的集线器交换机。当经由集线器交换机在源交换机与目的地交换机之间存在路径时，更新单元14可以更新与集线器交换机相关联的路由表。

[0071] 在示例性实施例中，所述系统的所述多个交换机形成胖树拓扑。选择单元13可以选择胖树拓扑中的叶交换机作为集线器交换机，并且该叶交换机可以到达胖树拓扑中的多个交换机目的地。第一路由算法可以使用上/下拐弯模型来在胖树拓扑中路由交换机间业务。当上/下拐弯模型失败时，交换机间业务做出下/上拐弯，并且所有下/上拐弯都被局部化到其中集线器交换机作为子树根节点的单个无死锁子树，以便防止胖树拓扑中的死锁。

[0072] 在示例性实施例中，交换机10可以可选地包括迭代单元12，其用于迭代经过胖树

拓扑中的多个叶交换机，以便找到可以到达胖树拓扑中的多个交换机目的地的一个或多个叶交换机。因此，选择单元13可以从所述一个或多个叶交换机选择集线器交换机。

[0073] 在示例性实施例中，交换机10还可以包括检查单元15。检查单元15可以检查集线器交换机是否具有兄弟交换机，并且当到目的地交换机的路径不存在时，检查与兄弟相关联的路由表是否包含该路径。

[0074] 在示例性实施例中，交换机10还可以包括路径设置单元16。当兄弟交换机存在并且与兄弟交换机相关联的路由表包含到目的地交换机的路径时，路径设置单元16可以在源交换机和目的地交换机上设置通过兄弟交换机到目的地交换机的路径。

[0075] 在示例性实施例中，交换机10还可以包括端口设置单元17，其用于将用于目的地交换机的输出端口设置为与用于集线器交换机的输出端口相同。

[0076] 可以使用包括根据本公开的教导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的一个或多个常规的通用或专用数字计算机、计算装置、机器或微处理器来方便地实现本发明。如对于软件领域的技术人员将显而易见的，熟练的程序员可以基于本公开的教导容易地准备适当的软件编码。

[0077] 在一些实施例中，本发明包括一种计算机程序产品，其是其上/其中存储可以用于将计算机编程为执行本发明的任何处理的指令的存储介质或计算机可读介质。存储介质可以包括，但不限于，任何类型的盘(包括软盘、光学盘、DVD、CD-ROM、微驱动器和磁光盘)、ROM、EPROM、EEPROM、DRAM、VRAM、闪存装置、磁卡或光学卡、纳米系统(包括分子记忆IC)、或适合于存储指令和/或数据的任何类型的介质或装置。

[0078] 已经为了例示和描述的目的提供了前面对本发明的描述。并非意图是穷举的或者使本发明限于所公开的精确形式。许多修改和变型对于本领域的从业人员将是显而易见的。变型可以包括所公开的特征和/或元件的任何组合。为了最好地说明本发明的原理及其实际应用，选择并描述了实施例，从而使得本领域的技术人员能够针对各个实施例、在适合于所设想的特定用途的各种修改的情况下理解本发明。意图是本发明的范围由权利要求书及其等同形式限定。

[0079] 附录A

[0080] 下面是说明SFTREE算法无死锁的证明。该证明还包含相关术语的示例性定义。

[0081] 定义1.交换机s_n意指等级为n的交换机。根交换机具有等级n＝0。

[0082] 定义2.向下信道意指s_{n-1}与s_n之间的链路，其中业务从s_{n-1}流到s_n。向上信道意指s_n与s_{n-1}之间的链路，其中业务从s_n流到s_{n-1}。

[0083] 定义3.路径被定义为由信道连接的一系列交换机。它以源交换机开始，以目的地交换机结束。

[0084] 定义4.U形拐弯是向下信道之后为向上信道的拐弯。

[0085] 定义5.当占有信道c_i的数据包请求使用信道c_j时，信道c_i与c_j之间的信道依赖发生。

[0086] 定义6.信道依赖曲线图G＝(V,E)是有向图，其中顶点V是网络N的信道，边缘E是使得存在从c_i与c_j的(信道)依赖的信道对(c_i,c_j)。

[0087] 定义7.子树意指胖树内的会聚到单个叶交换机s的逻辑倒置树结构，所述单个叶交换机s是该子树的单个根。子树从其根扩展，并且其叶是胖树中的所有顶级交换机。任何

向上到向下的拐弯都将离开子树结构。

[0088] 以下定理声明路由函数的无死锁的充分条件。

[0089] 定理1.当且仅当在信道依赖曲线图G中不存在循环时，用于网络N的确定性路由函数R才是无死锁的。

[0090] 接下来，可以提供SFTREE算法的无死锁定理所遵循的引理。

[0091] 引理8.在胖树内存在至少一个子树，该子树仅使用在该子树内的U形拐弯来允许交换机对交换机的全连接。

[0092] 证明.在胖树中，从任何叶，可以使用向上/向下路径或水平兄弟路径到达所述交织结构中的任何其他节点(包括所有交换机)。这是因为每一个终端节点可以与每一个其他终端节点通信，并且因为终端节点直接连接到叶交换机，所以得出结论，叶交换机能够到达拓扑中的任何其他节点。因此，如果它不包含链路故障，则任何叶交换机可以充当子树的根。

[0093] 引理9.在不引入死锁的情况下，在单个子树内可以存在无限个U形拐弯。

[0094] 证明.子树是逻辑树结构，并且在它内可以发生两种类型的拐弯：U形拐弯(即，向下拐到向上的拐弯)、以及普通的向上拐到向下的拐弯。U形拐弯不能发生在子树(以及胖树)中的顶部交换机处，因为这些交换机不具有输出向上端口。根据定义7，任何向上拐到向下的拐弯都将离开子树。

[0095] 因为子树是不具有任何循环的连接图，所以它本身是无死锁的。任何死锁必须涉及子树外部的U形拐弯，因为U形拐弯之后为离开子树的向上拐到向下的拐弯。当做出起源于子树中的向上/向下拐弯并且仅跟随到达目的地的向下信道时，经过该拐弯的所有业务都将离开子树。这样的依赖性可能永不再次进入子树，并且可能永不到达任何其他U形拐弯，所以，不能形成任何循环。

[0096] 定理2.使用子树方法的SFTREE算法是无死锁的。

[0097] 证明.按照引理8和引理9，仅当在子树外部存在U形拐弯时，胖树中的死锁才可能发生。由于SFTREE算法的其中所有U形拐弯都在子树内发生的设计，这样的情况不能发生。离开子树的业务将不循环回到该子树，因为一旦它离开子树，它就仅通过向下信道被转发给目的地，并且它被消耗，因此，在拓扑中将不会发生循环信用依赖。

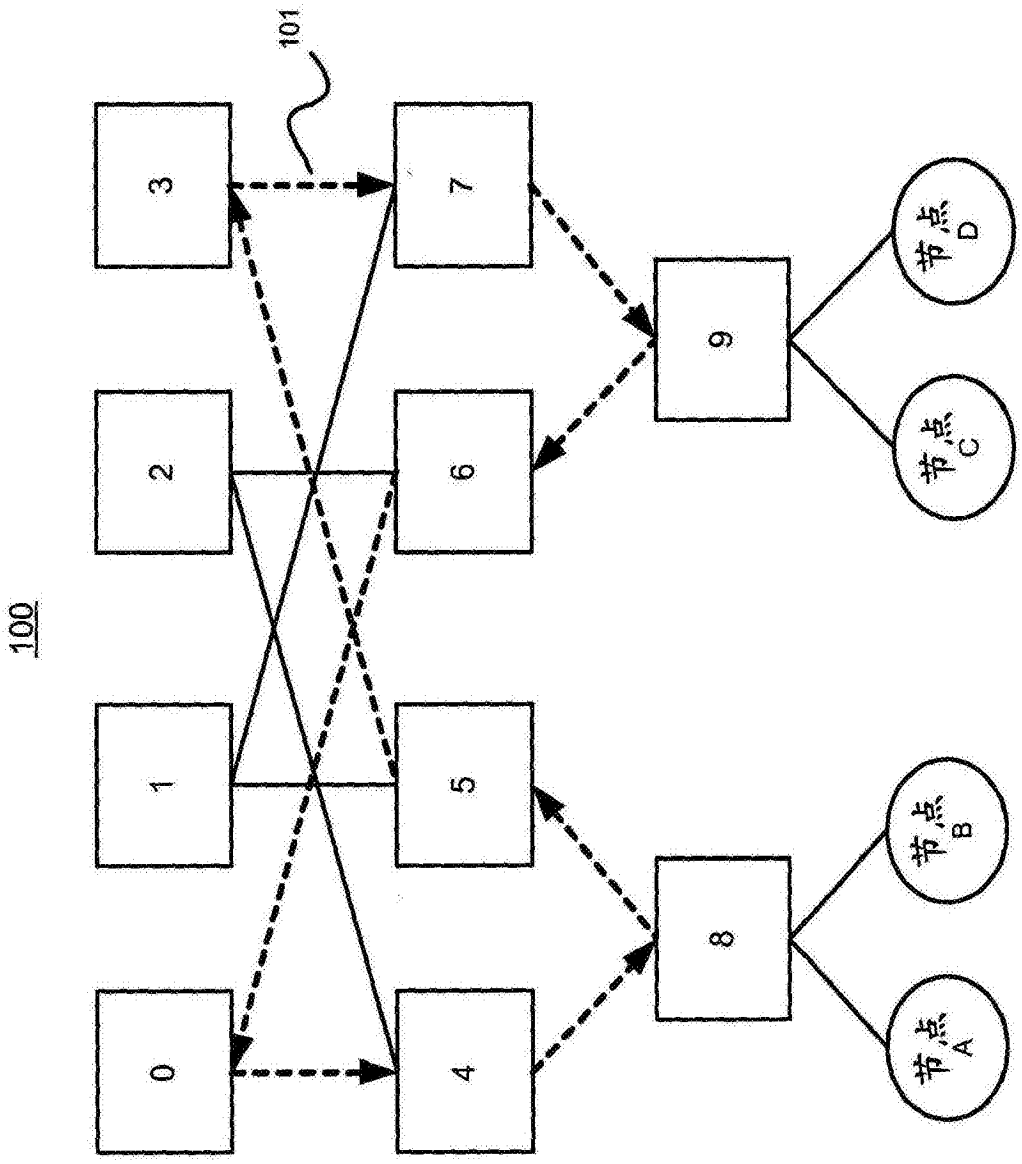


图1

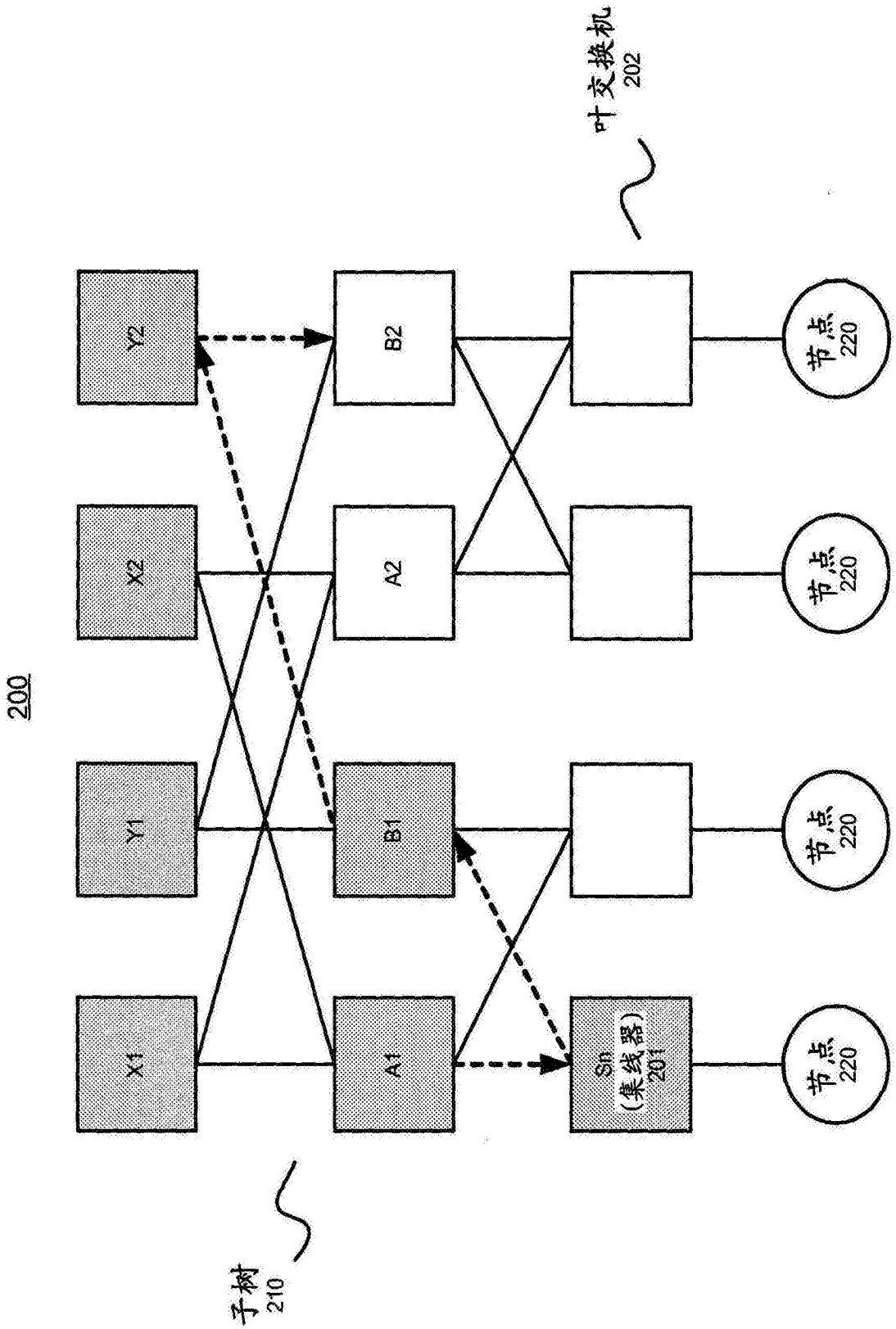

叶交换机
202
Y2
B2
节点
220
X2
A2
节点
220
200
Y1
B1
节点
220
X1
A1
Sn
(集线器)
201
节点
220
子树
210

图2

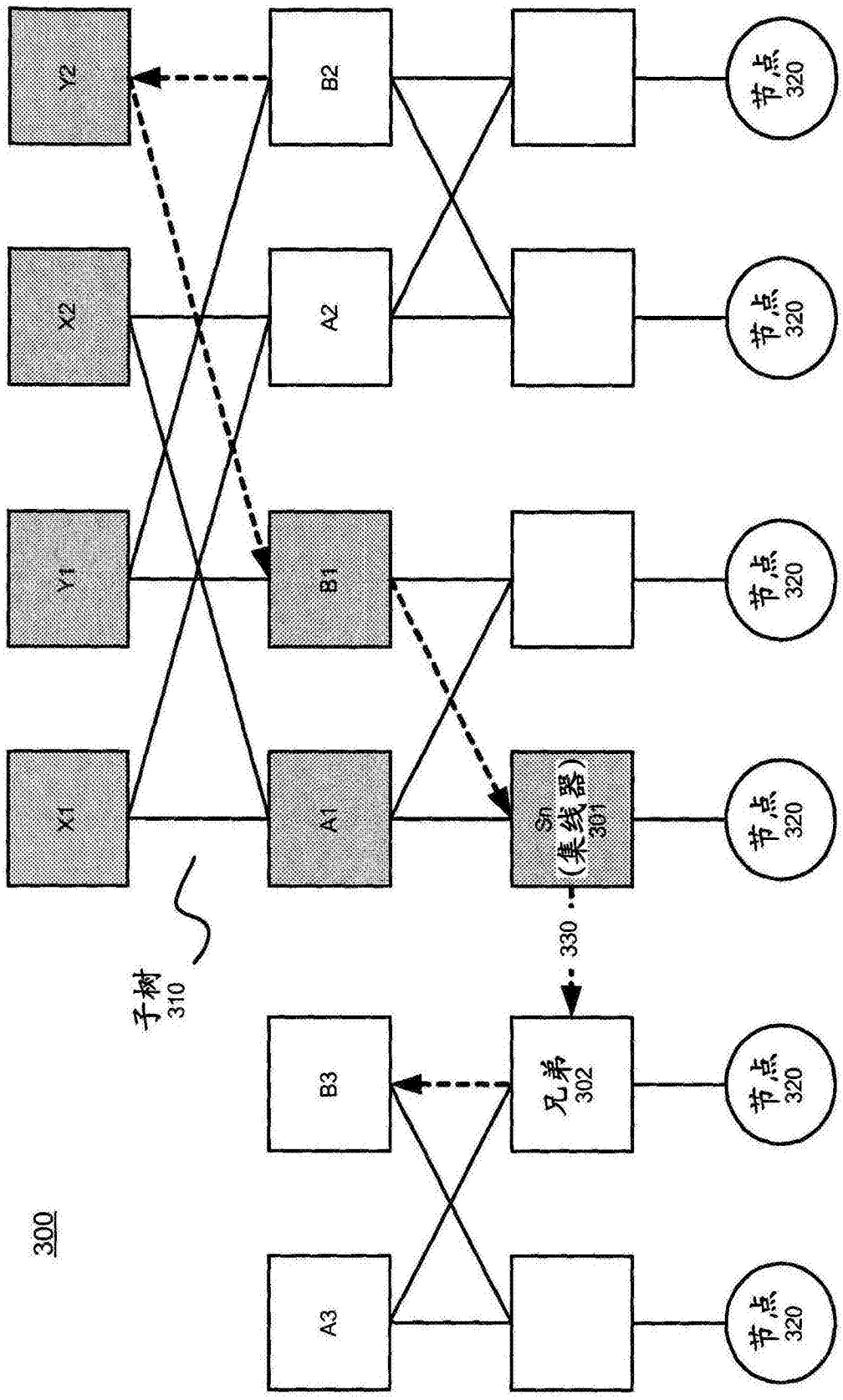

Y2
B2
节点
320
X2
A2
节点
320
Y1
B1
节点
320
X1
A1
Sn
(集线器)
301
节点
320
子树
310
330
B3
兄弟
302
节点
320
300
A3
节点
320

图3

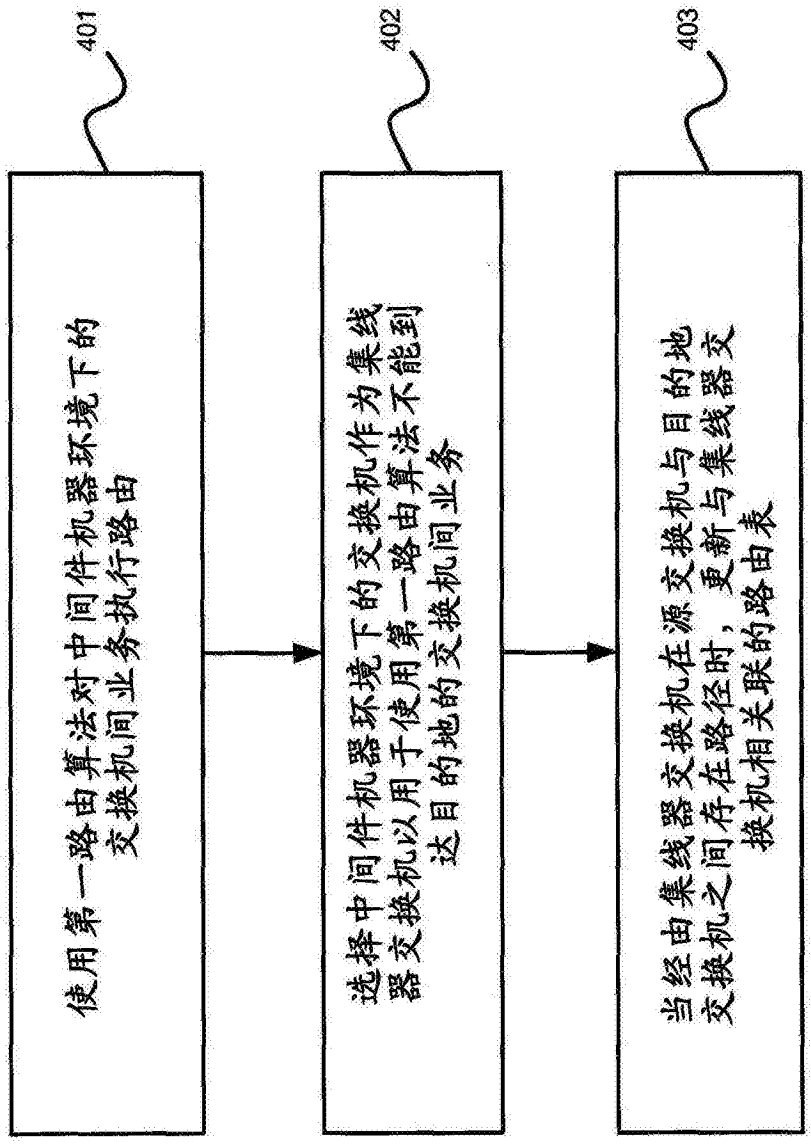

401
使用第一路由算法对中间件机器环境下的交换机间业务执行路由
402
选择中间件机器环境下的交换机作为集线器交换机以用于使用第一路由算法不能到达目的地的交换机间业务
403
当经由集线器交换机在源交换机与目的地交换机之间存在路径时，更新与集线器交换机相关联的路由表

图4

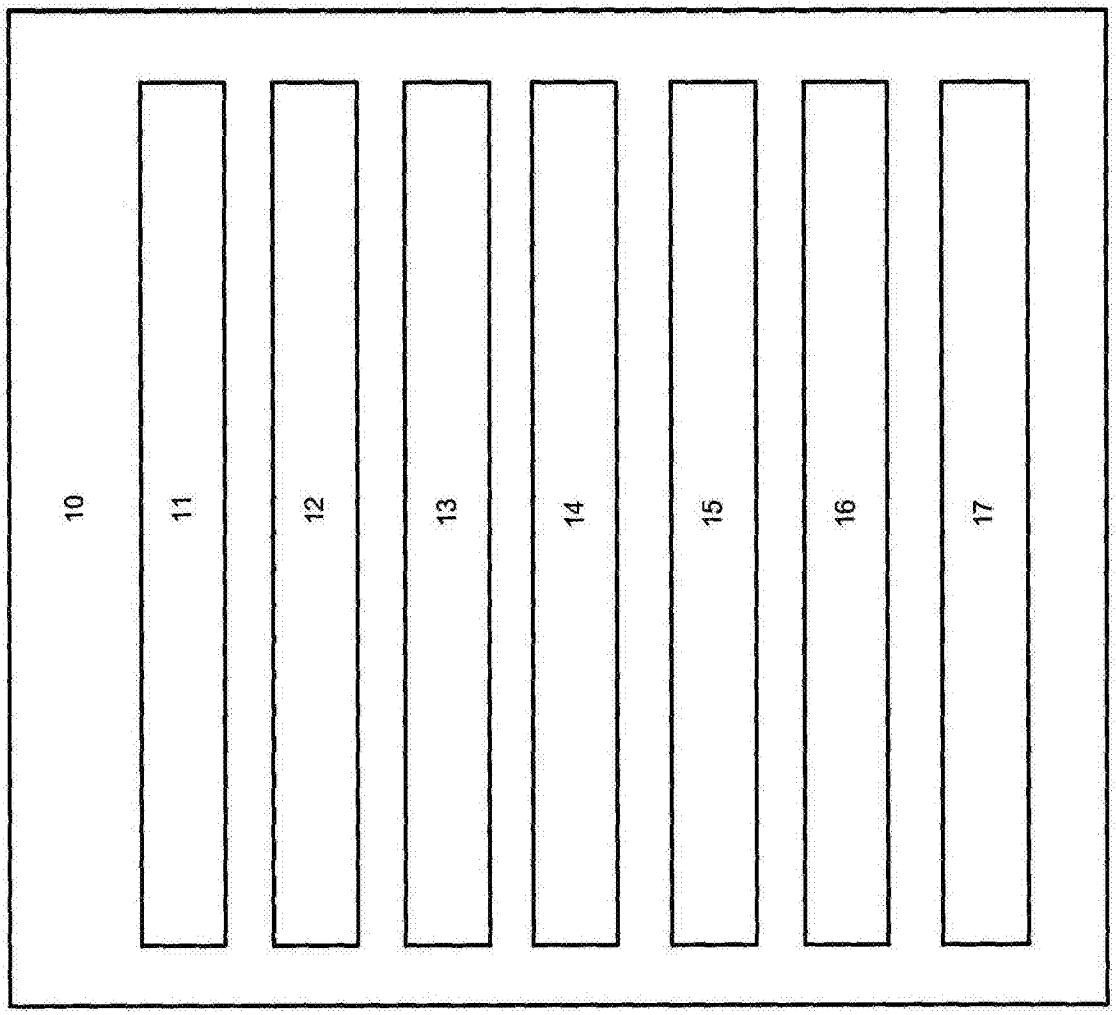


图5