

DCN 地址配置冲突检测技术

陈 勇, 王 宏

(国防科学技术大学 计算机学院, 湖南 长沙 410071)

摘 要:数据中心网络(DCN)地址配置过程中可能存在配置冲突,利用图同构思想进行冲突检测是一种有效而且可行的方法。从分析冲突类型入手,提出了基于子图同构的冲突检测方法,并对 O_2 算法中基于SPLD产生的关键对的计算进行优化,减少潜在误报;在子图同构的 x 跳数选择上,探讨给出适当的跳数大小,利用二分法寻找同构子图;对不引起度改变的错误连线故障,通过记录两端节点的方式记录错误连线;改进后的方法一定程度上提升了冲突检测方法的性能,达到了预期效果。

关键词:数据中心网络;冲突检测;子图同构

中图分类号:TP31

文献标识码:A

文章编号:1673-629X(2011)11-0077-04

DCN Address Configuration Conflict Detection Technology

CHEN Yong, WANG Hong

(Computer Science School, National University of Defense Technology, Changsha 410071, China)

Abstract: Data center network address configuration may exist configuration conflicts, it is an effective and feasible method to use graph isomorphism ideological for conflict detection. Analysis of the type of conflict first, proposed a method of detection conflict based on subgraph isomorphism, and optimized the O_2 algorithm that key generated based on SPLD, reduced the potential for false positives; Subgraph isomorphism in the number of hops the choice of x , give the appropriate number of hops of size, use dichotomy looking for subgraph isomorphism; On the degree of change does not cause the error connection failed, the way by recording both ends of the nodes record the miswirings; Improved method to some extent enhances the performance of conflict detection method to achieve the desired effect.

Key words: data center network; conflict detection; subgraph isomorphism

0 引 言

配置冲突检测作为一种 NP-complete 问题,目前还没有得到完全解决。但是,过去几十年来,人们不断地探索新思路,改进方法,简化评价函数,取得了很大进步。配置冲突主要包括路由策略冲突、安全策略冲突和地址冲突等,这里着重讨论的是地址冲突。

文献[1]介绍了在 MANET 中常用的 2 种地址冲突检测方法:主动 DAD 与被动 DAD。在对经典的被动 DAD 检测方法 PACMAN 算法工作原理的分析后,针对 PACMAN 算法中的不足,提出了一种在 RREP 包中加入标识符时间集合的 RPS 算法。文献[2]介绍模型重构过程中重构应用规则的并行应用会导致重构应用规则之间产生冲突现象。关键对是一对有冲突的转换,模型转换规则之间的冲突关系可以通过图转换中

的关键对进行分析。

对于数据中心网络故障检测,希望通过图论来寻找解决办法。给定逻辑拓扑图 G_b 和物理拓扑图 G_p ,找到图 G_p 中所有出现故障的部件的问题相当于获得 G_b 和 G_p 的最大公共子图(MCS) G_{mcs} 。因此,通过比较发现 G_{mcs} 与 G_p 的不同之处,这些地方就是出现故障的部分。所有与这些部分相关的设备(即服务器或交换机),称之为故障设备。文献[3]提出了一种基于状态转换图同构求解的时序电路等价性验证算法。算法将两时序电路的等价性问题转化为验证相应状态转换图的同构性。然而,事实证明,MCS 的问题是 NP-complete^[4] 和 APX-hard^[5] 问题。也就是说没有一个有效的算法能解决这个问题,特别是对于大型的数据中心网络拓扑结构。

为了达到更好的性能和更便于管理,大型数据中心通常根据一定的模式或规则来设计和构建。这样的模式或规则意味着数据中心结构有两个属性:

1) 拓扑上的节点通常有规则的度。例如,在表 1^[6-8] 中显示了几个著名的数据中心网络模式的度;

2) 图是稀疏的。

收稿日期:2011-04-10;修回日期:2011-07-16

基金项目:国家自然科学基金(90604006)

作者简介:陈 勇(1980-),男,湖北仙桃人,硕士,研究方向为计算机网络配置;王 宏,副研究员,博士,研究方向为计算机网络流量控制、计算机网络配置。

对于检测数据中心的故障,这些属性很重要。在判定冲突检测的思想中,第一个属性是用来在有节点度的变化时检测故障设备,而第二个属性为在没有发生度变化时检测故障设备提供了支持。

表 1 几种数据中心网络架构的度模式

结构	交换机的度	服务器的度
BCube(n, k)	n	$k + 1$
FatTree(n)	n	1
VL(n_r, n_p)	$n_p, (n_r + 2)$	1

1 配置冲突概述

数据中心网络的故障可能是由硬件和软件错误,或者简单的人为配置错误所导致。例如,坏的或不匹配的网卡和电缆故障比较常见,接线错误或不正确的连接电缆往往也是存在的。

将数据中心的故障分为三种类型:节点故障、链路故障和错误接线。节点故障发生在给定的服务器或交换机上,由于硬件或软件的原因,设备无法正常工作,从而导致网络无法到达;链路故障是电缆、网卡等损坏或存在错误地接入,使设备之间的链接中断;错误接线故障是指实际的电缆连接与网络蓝图不一致。这些故障可能会引发严重的问题,降低性能。

从物理拓扑结构上,很难明确区分某些故障种类。例如,网卡或者软件的故障都有可能将服务器处于故障状态,无法接入数据中心网络。因此,我们的目的是检测和定位所有相关的故障设备,并将设备信息报告给网络管理员,而不是确定故障类型。故障类型可以由网络管理员对故障设备进行相关操作来判断。

1.1 故障节点的度变化

基于上述三种故障类型,逐一讨论如下。通过观察,发现在大多数情况下故障会导致设备的度改变。

- 节点。如果有一个故障节点,其相邻所有节点的度都会下降 1,这样就可以通过检查它的邻居节点来确定故障位置。

- 链路。如果有一条故障链路,相关的两个节点的度会下降 1,这使得它可以被发现。

- 错误接线。错误接线稍微比上面两种故障复杂些。正如图 1 左所示,错误接线导致其相关的节点增加或减少其度,可以很容易被检测到。但是,在图 1 右所示,两条电缆同时发生错误接线,使得由一条电缆接线错误造成的度变化被另一个错误接线所掩盖,因此没有节点度变化的情况。这将是讨论的一种最困难的情况。

在故障检测中,对于以上三种类型的故障,无论是链路造成的故障,还是错误接线,都只报告相关的故障

节点(即故障设备),然后由网络管理员再对故障节点作下一步的处理。

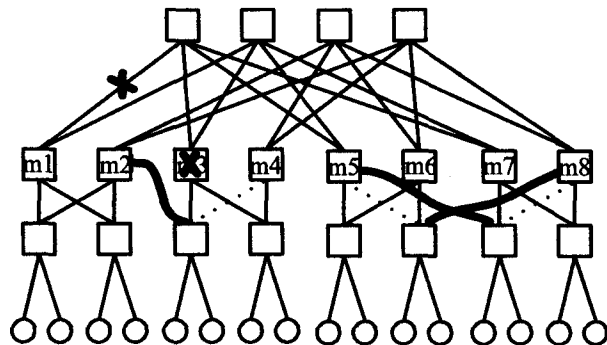


图 1 错误接线度变化与度不变化

1.2 故障节点的度不变化

大多数情况下,网络故障会使节点度发生可探测到的变化^[9],但是,错误接线导致节点度没有变化的情况仍然存在。这种情况发生后,管理员检查网络,首先将引起度变化的故障进行修复。这时,可以认为,节点故障和链路故障均已被排除,网络中只存在错误连线故障。

这里可以做出假设:

- 1) 相对于所有节点,涉及度不变化的故障节点的数量是很小的;

- 2) G_p 和 G_b 有相同数目的节点以及节点度模式。

基于数据中心网络的特点和网络的真实应用场景,提出自己的故障检测算法,通过识别设备 ID 检测故障设备。

尽管存在错误接线,但绝大部分 G_p 和 G_b 仍然是相同的。利用这个事实来检测这种错误接线。基本想法是,首先发现一些 G_p 和 G_b 之间的对应节点,然后用这些节点作为定位点来检测子图推测它们是否同构。通过这一点,得出了两个图之间的差异,对于不同的定位点,相互关联故障候选者,最终作出决定。所以,文中冲突检测算法包含两个部分:定位点的选择和故障检测。

2 DAC 地址冲突检测方法

DAC^[10] (Data center Address Configuration system)

是一个通用的数据中心和自动地址配置系统。对于所有现有和未来的数据中心网络,DAC 给出了一个很有意义的解决方案。DAC 系统设计了一个实用的方案,巧妙地利用了所有数据中心结构度的规律性,检测故障导致设备的度的变化。对于最难的没有度的变化,系统提出了一个方案,从多个定位点比较蓝图和物理拓扑图并通过多数表决的相关故障。评价表明,DAC 的解决方案,能够快速检测出所有的最难以检测的故障。

三点不足:

(1) 考虑到在错误接线所涉及的节点数量很少, 为了尽量减少人为干预, DAC 算法采用 SPLD 作为评价指标自动选择定位对, 但是鉴于 SPLD 计算量较大, 而且两个类似 SPLDs 节点不一定是真正的对称的一对, 故障检测方案将考虑到潜在的误报。

(2) 搜索最大的同构子图的过程中, x -hop 中 x 的值是随机选择的, 并不利于提高效率。

(3) 在 DAC 的故障检测中, 对于以上三种类型的故障, 无论是链路造成的故障, 还是错误接线, 都只报告相关的故障节点 (即故障设备), 然后由网络管理员再对故障节点作下一步的处理。

3 改进算法

在蓝图 G_b 和物理拓扑图 G_p 中, 自动地选择定位节点对, 利用电压模拟法^[8]来进行节点对的选择。在实际的网络中, 错误接线所涉及的节点数量是很少的, 因此图 G_b 和 G_p 中的两个“对称”节点仍会有近似的节点电压值, 相比于 SPLD, 准确性更高。

基于此, 设计定位点对选择算法, 即 Anchor_Pair_Selection(), 见图 2。在该算法中, $||$ 节点电压 (v) - 节点电压 (v') $||$ 是简单的欧氏距离。一旦定位节点对已被选中, 比较这些 G_b 和 G_p 的定位节点对, 通过多数投票的方式关联故障。

```

/* pre-compute the 节点电压 of all nodes in  $G_b$ , select one node
from each group of nodes with same 节点电压 and store it in  $G_b$ 
*/
( $A_p; A_b$ ) = Anchor_Pair_Selection( $G_p; G_b$ )
1  $A_p$  = a group of selected anchor points in  $G_p$ ;
2 foreach  $v \in A_p$ 
3   select a  $v' \in G_b$  that minimizes  $||$  节点电压 ( $v$ ) - 节点电压
( $v'$ )  $||$ ;
4   store  $v=v'$  in  $A_p=A_b$ ;

```

图 2 定位点对选择算法的伪代码

如图 3 所示, 故障检测算法是 Malfunction_Detection()。具体来说, 给定图 G_p/G_b , 节点对集合 A_p/A_b , 定义最大的子图 $S_p^x(v)$, 对于每个定位点对 $v/v' \in A_p/A_b$, 在图 G_p/G_b 中搜索最大的同构子图, 分别从节点 v/v' 拓展 x -hop, 获得一个子图^[11-13]。 x 的值通常参照为定位点的最大跳数的 $2/5$, 然后通过二分查找来加速检索过程, 如果发现 $S_p^x(v)$ 和 $S_b^x(v')$ 是同构的, 而 $S_p^{x+1}(v)$ 和 $S_b^{x+1}(v')$ 不是同构的, 则认定有错误连接发生在距离节点 v 的 x -hop 到 $(x+1)$ -hop 之间的这两层上, 这两层上每个节点都有可能是故障设备。如果发现 $S_p^x(v)$ 和 $S_b^x(v')$ 不是同构的, 而 $S_p^{x-1}(v)$ 和

$S_b^{x-1}(v')$ 同构, 则发现存在故障设备所在的层; 否则就继续收敛 1 -hop, 进行同构判定。如果节点 v 的 x -hop 到 $(x+1)$ -hop 之间可能出现错误连线, 通过记录 G_p 中连线两端的节点的方式来记录一条连线, 将连线信息保存在一个列表中。在第 5 行, 增加一个计数器, 对可能的故障连线出现的次数进行记录, 作为确定故障连线的评价参数。

```

Malfunction_Detection( $G_p; G_b; A_p; A_b$ )
1 Define  $S_p^x(v)$  as maximal subgraph of  $G_p$  with maximal hop length  $x$ 
from node  $v$  where  $v \in G_p$ , and the same as  $S_b^x(v')$ ;
2 foreach pair of nodes  $v/v' \in A_p/A_b$ 
   value  $x = 2/5$  for vertex  $v$  maximal hop
3 if (mDLI_Mapping( $S_p^x(v), S_b^x(v')$ ) = true)
   if (mDLI_Mapping( $S_p^{x-1}(v), S_b^{x-1}(v')$ ) = false)
4   foreach wiring  $e \in E_p$  that is  $x$ -hop or  $(x+1)$ -hop away
from  $v$ 
5     counter( $i$ ) = counter( $i$ ) + 1;
6   return a wiring list sorted by their counter values;
7   else  $x = x + 1$ , return 3;
8   else  $x = x - 1$ , return 3;

```

图 3 故障检测算法的伪代码

完成所有定位点的检测后, 向管理员报告这个列表。该列表包含连线两端节点的设备 ID 和每个连线的计数器的值, 依据计算器的值降序排列。从本质上讲, 连线计数器的值越大, 该项所代表的连线是错误接线的可能性也就越大。然后, 网络管理员通过列表查找设备和连线并纠正接线错误。但是, 如果发现检测到的节点不是真正的接线错误节点时, 进程停止并忽略列表上的其余节点, 重新进行检测。

本方案故障检测的准确性很大程度上取决于网络中节点数量与检测定位点数量的比值。实验表明, 用足够数量的定位点, 文中的算法能够找到所有的故障 (即将接线错误设备放在输出到计数器列表的顶部)。根据实验结果, 最多选择 1.4% 的定位点, 可以发现评估结构上的所有错误连接节点。为了更加可靠, 可以谨慎地选择一个较大比例的定位点来开始检测, 最有可能检测所有错误连接 (即它们都位于列表的顶部)。其实, 通过并行计算可以更容易, 因为文中的故障检测, 从不同的定位点的计算是相互独立的, 因此可以被并行执行。

修理错误连接后, 将再运行地址配置算法来获取设备到逻辑 ID 映射。即使不是所有的错误连接都在列表顶部的情况下, 仍然可以在下一轮映射配置中很快发现它们。然后, 重新运行故障检测方法, 直到检测到所有的故障设备并纠正错误连接, 最后, 可以得到正确的设备到逻辑 ID 映射。

4 算法的性能评估

由于度变化的故障能很容易地被检测,因此实验重点在于模拟不发生度变化的错误接线。实验针对图1右中所示的错误,通过3种数据中心网络结构来检测、评估算法的准确性。对每种结构,实验选择适中的大小,分别是BCube(6,4), FatTree(40)和VL2(20,100)。通常情况下,没有度变化的错误接线是很稀少的,每个这样的案例需要至少4个错误接线的设备。所以在实验模拟中,可以随意地创造5组这样的错误连线,这样就会产生20个错误连线的节点。在算法的输出中,会得到检测错误节点所需要的定位点的数量。错误节点数量越多,所需要的定位点的数量也相应越多。一个接线错误节点被检测到表示该节点处于计数器列表的顶端。管理员将根据计数器列表的顺序纠正错误接线,而如果列表中不再存在错误节点,程序将停止。

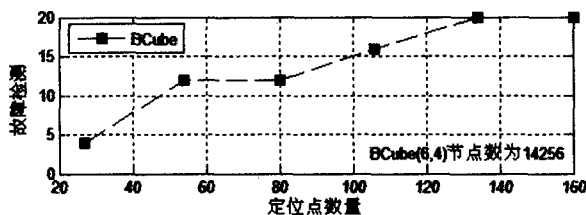


图4 BCube(6,4)故障检测分析图

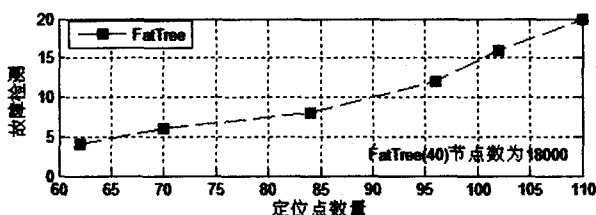


图5 FatTree(40)故障检测分析图

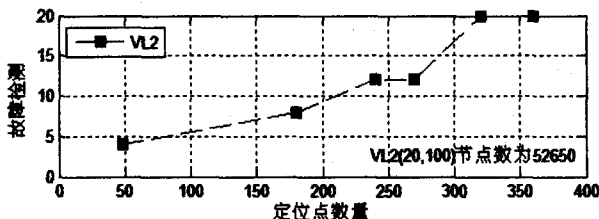


图6 VL2(20,100)故障检测分析图

由以上三图显示的结果清楚地表明,检测到的故障数量随着选定的定位点数量增加。在所有结构的实验上,最多选定为1.4%的定位点,可以发现所有的故障节点。对于不同的结构,需要不同数量的定位点,以检测所有20个接线错误的设备。例如,在BCube和VL2,分别只需要130和320个定位点来检测所有故障;在FatTree,需要110个定位点来检测它们。通过与DAC系统的比较,可以发现在BCube和VL2结构

上,本算法优于DAC,但对于FatTree结构,DAC系统效果更好。

5 结束语

文中分析了数据中心网络错误的类型。提出配置冲突检测和处理的方法,对于故障节点度变化和不变化分别进行了讨论,特别是度不发生变化时的处理方法;在分析DAC冲突检测方法的基础上,提出了优化改进算法。最后在三种数据中心网络上对算法的性能进行了模拟分析,优化效果得到了进一步的分析证明。

参考文献:

- [1] 陶洋. MANET中的高精度地址冲突检测算法[J]. 计算机工程, 2009, 35(5): 94-96.
- [2] 陈军冰. 基于图转换的模型重构冲突检测方法[J]. 计算机工程, 2010, 36: 40-42.
- [3] 卢英, 李炜. 基于状态转换图同构求解的等价性验证算法[J]. 计算机技术与发展, 2009, 19(3): 74-75.
- [4] Maximum common subgraph problem[EB/OL]. 2011. http://en.wikipedia.org/wiki/Maximum_common_subgraph_isomorphism_problem.
- [5] Kann V. On the approximability of the maximum common subgraph problem[C]// Annual Symposium on Theoretical Aspects of Computer Science. [s.l.]: [s.n.], 1992.
- [6] Guo C, Lu G, Li D, et al. BCube: A High Performance, Server-centric Network Architecture for Modular Data Centers[C]//SIGCOMM. [s.l.]: [s.n.], 2009.
- [7] Mysore R N, Pamboris A, Farrington N, et al. PortLand: A Scalable Fault-Tolerant Layer 2 Data Center Network Fabric[C]//SIGCOMM. [s.l.]: [s.n.], 2009.
- [8] Greenberg A, Jain N, Kandula S, et al. VL2: A Scalable and Flexible Data Center Network[C]//SIGCOMM. [s.l.]: [s.n.], 2009.
- [9] Rodeheffer T, Thekkath C, Anderson D. SmartBridge: A scalable bridge architecture[C]//SIGCOMM. [s.l.]: [s.n.], 2000.
- [10] Chen Kai, Guo Chuanxiong, Wu Haitao, et al. Generic and Automatic Address Configuration for Data Center Networks[C]//SIGCOMM. [s.l.]: [s.n.], 2010.
- [11] 李琳琳. 图同构判定算法研究[D]. 上海: 复旦大学, 2005.
- [12] 孙惠泉. 图论及其应用[M]. 北京: 科学出版社, 2004.
- [13] 臧威. 任意图的同构判定及应用研究[D]. 上海: 复旦大学, 2007.