可扩展路由器中 SPT 并行计算的实现

张小平,吴建平,周 兴,史 峰,赵有健,吴 鲲
(清华大学计算机科学与技术系,北京 100084)

摘 要: 随着互联网的飞速发展,集群结构的下一代核心路由器已经成为研究的重点.在可扩展路由器中(cluster router),并行路由算法是关键问题之一.对于广泛部署的 OSPF 协议,最短路径树(SPT)的并行计算是其并行化的核心难点.本文提出了一种计算最短路径树的算法 – 分区 Dijkstra 算法(D-D),分析了算法性能,并通过模拟实验验证了算法的性能.

关键词: 可扩展路由器; 路由节点; 最短路径树; Divisional-Dijkstra; 并行算法
中图分类号: TP393.09 **文献标识码:** A **文章编号:** 0372-2112 (2007) 11-2129-06

An Implementation for Parallel Computing SPT in Cluster Router

ZHANG Xiao-ping, WU Jian-ping, ZHOU Xing, SHI Feng, ZHAO You-jian, Wu Kun
(*Computer Science and Technology Department, Tsinghua University, Beijing* 100084, *China*)

Abstract: To keep up with the pace of fast development of Internet, cluster architecture has been proposed for next generation core routers. In a cluster router, parallel computation is expected. Computing shortest path tree (SPT) is a fundamental problem implementing OSPF, which is one of the most popular routing protocols. This paper presents a parallel algorithm D-D(Divisional-Dijkstra Algorithm) for computing SPT, analyzes the performance of D-D, and finally validates the D-D performance by experiments.

Key words: cluster router; routing node; Shortest Path Tree(SPT); Divisional-Dijkstra(D-D); parallel algorithm

1 引言

核心路由器是计算机网络中的关键设备,其主要功能是完成数据路由功能.目前的核心路由器都是单节点路由器(single-chassis router),其特点是单交换结构、单主处理器.随着互联网的飞速发展,单节点路由器已经体现出诸多缺陷:首先受交换结构发展的限制,总体性能提升非常困难;另一方面,随网络规模的急速膨胀,路由计算问题也成为路由器发展的瓶颈之一;此外,在产业界,单节点路由器的升级目前都是整体更换式更新,成本越来越高,也将成为阻碍产业发展的因素之一.在此情形下,结构可扩展的路由器(cluster router 简称 CR)将成为未来骨干网络的核心[1,2].

CR 是由路由节点(routing node 简称 RN)通过某种高性能内部互联网络级联构成,其定义可以为"将可独立工作的多个路由节点连接构成的集群单映像路由系统"[3].其特点是:(1)分布式多处理器结构,用于路由计算及应用服务的分布式处理;(2)采用可扩展互连结构,使得路由系统可灵活扩展;(3)单映像路由系统,使得 CR 对外表现为一台核心路由器.CR 模型如图 1[3]所示.图 2 为 AVICI 公司研制的 TSR,它是通过 3-Dtours 内部互连网络扩展级联而成的 CR[4].随 RN 数目的增加,CR 总体性能(如吞吐能力、服务承载能力、端口数目等)将随之提高.

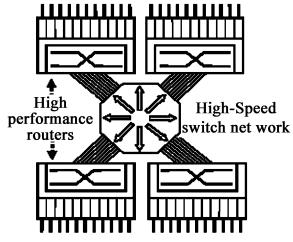

图 1 CR 体系结构参考模型

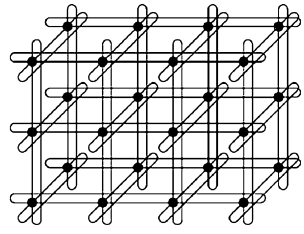
图 2 基于 3-Dtour互连的 CR

根据功能的侧重,路由器可以划分为转发平面和控制平面[5,6].转发平面完成数据流的路由查找与转发操作;控制平面对路由器进行控制与管理,主要包括与邻居路由器进行路由交互、路由计算以及系统的管理操作.转发平面的可扩展性方面已经有大量研究[7~10],并且已经有厂商推出相应的产品[4,11,12].而路由器控制平

收稿日期:2006-12-14;修回日期:2007-06-15
基金项目:国家自然科学基金(No.90604029);国家 973 重点基金研究发展规划(No.2003CB314801)

面的结构仍然以集中式为主.CR 控制平面特点之一是分布式计算.单节点路由器中,对路由协议维护、路由信息计算都通过单一 CPU 完成,可靠性低.可扩展路由器中,控制平面可以通过路由节点协同工作,对外部网络路由信息实施分布式计算,提高计算效率同时增强了路由器的可靠性.

OSPF 是一个广泛部署的复杂路由协议,它包含邻居关系维护、数据库维护、最短路径树计算等多个组成模块.对于核心路由器上运行的 OSPF 协议任务,其主要瓶颈在于 SPT 计算.尽管有研究表明,在快速检测网络拓扑变化的要求下,邻居关系维护也会产生很大的系统开销,不过,由于这种开销可以通过邻居关系模块向线卡上的功能转移而得到很好的改善[13],最终不会成为系统的决定性瓶颈.以并行 SPT 算法为基础的分布式 OSPF 协议模型是可扩展路由器体系结构的重要构成部分之一.

目前对 SPT 的并行计算有一定的研究,基于 BTH (balanced-tree hierarchy)结构的分布式 SPT 算法[14,15]尽管可以实现 $O(\log(n))$的计算复杂度,但是是否所有的图都可以转化为 BTH 结构尚没有相关文献论述.文献[16]中提出了一种基于点断集的网络划分.对于单源最短路径树问题,该算法可以达到 $O(n)$的计算时间复杂度.但是该算法的划分不能解决任意多个平面上的拓扑网络的划分,这是一个 NP 问题,这是该算法最大的一个缺陷.文献[17]中也提出了一种多级划分的方法.这种划分是针对 mesh 结构的拓扑网络.该算法解决单源最短路径树问题的计算时间复杂度是 $O(N^{L-1/L})$,比一般的最短路径算法效率要高.但该算法适用性不广,它针对 mesh 拓扑结构的网络.对于可扩展路由器的具体应用,由于互联网拓扑的任意性,其分布式 SPT 算法应针对任意拓扑均适用.而上述的研究结果暂时或受限于网络拓扑,或带来更复杂的计算问题,都不能很好应用于可扩展路由器内部.

本文提出一种 SPT 的并行算法 – Divisional-Dijkstra (D-D)算法,该算法特点是针对任意拓扑的网络结构,对 SPT 并行计算效率较单 CPU 计算有显著提高.文章集中讨论 SPT 分布式计算模型、算法与性能分析,对 OSPF 整体模型结构的分布式设计不在本文的讨论范围之内.

2 算法描述

本文将外部网络拓扑抽象为一张带正权的无向图 $G=(V,E)$,其中 V 为图 G 的节点集合,E 为图 G 的边集合,$\forall e\in E, w(e)>0$,其 $w(e)$为边 e 的权值.求图 G 中一个节点 v_0(称为源点)到其他所有节点的最短路径,即为求图的最短路径树问题.传统单节点路由器中,OSPF 协议采用经典 Dijkstra 算法求解该问题,时间复杂度为 $O(N^2)$.文中算法期望通过多路由节点进行分布式最短路径计算,实现传统算法的改进,以适应可扩展路由器体系结构中 OSPF 协议的分布式实现.

2.1 定义

在本节中,我们将给出一些相关的定义.

定义 1 令 $D_i=\{V_i,E_i\}$(其中 $V_i\subseteq V, E_i\subseteq E$),则 D_i 是一些节点和边的集合.若满足如下条件:(1) $V_i\neq\varnothing$;(2) $\forall(v_j,v_k)\in E_i$,都有 $v_j,v_k\in V_i$.(3) $\forall v_j,v_k\in V_i$,如果 $e_{jk}=(v_j,v_k)\in E$,则 $e_{jk}\in E_i$.(4)集合$\{V_i,E_i\}$所形成的子图为连通图.

则称 D_i 为 G 的一个分区(Division). D_i 的所有元素可以形成 G 的子图,记为 $G_i=(V_i,E_i)$.

定义 2 对给定的正权无向图 $G=(V,E)$,从 v_l 到 v_m 的路径记为 $P_{lm}=(e_1,e_2,\cdots,e_n)$,其中 $e_1=(v_l,v_x)$, $e_n=(v_y,v_m)$.路径长度记为 w_{lm},其中 $w_{lm}=w(e_1)+w(e_2)+\cdots+w(e_n)$.

定义 3 图 G 存在 k 个分区 $D_i=\{V_i,E_i\}$,$(i=1,2,\cdots,k)$,如果 $\forall i,j\in\{1,2,\cdots,k\}$,满足 $V_i\cap V_j=\varnothing$,且 $\bigcup_{i=1}^{k}V_i=V$,则称$\{D_i,(i=1,2,\cdots,k)\}$为图 G 的一个 k-划分.

定义 4 在图 G 的一个 k-划分下,如 $\exists e_{lm}=(v_l,v_m)\in E$,有 $v_l\in V_i, v_m\in V_j(i\neq j)$,则称 e_{lm}为图 G 在此划分下的一条区间边.图 G 在此划分下的区间边集合记为 E_{IDC}(Inter-Connection of Divisions).

定义 5 在图 G 的一个 k-划分下,如果 $\exists v_l\in V_i$, $v_m\notin V_i, e_{lm}=(v_l,v_m)\in E$ 则称 v_l 是分区 D_i 的一个边界节点.分区 D_i 的边界节点集合记为 VB_i' .

定义 6 在图 G 的一个 k-划分下,分区 D_i 的扩展边界节点集合记为 VB_i,定义如下:

$$VB_i=\begin{cases}VB_i' & \notin V_i\\ VB_i'\cup\{v_0\}, v_0\in V_i\end{cases},\quad i=1,2,\cdots,k$$

记 $d_i=|VB_i|$, VB_i 中的元素表示为 $v_i^j(j=1,2,\cdots,d_i)$.在分区 D_i 中以 $v_i^j(j=1,2,\cdots,d)$为源点的最短路径树记为 T_i^j.

定义 7 在图 G 的一个 k-划分下,如果 $\exists v_l\in V_i$, $\forall e_{lm}=(v_l,v_m)$,都有 $v_m\in V_i$,则称 v_l 是分区 D_i 的一个内部节点.分区 D_i 的内部节点集合记为 VI_i.

定义 8 图 G 的一个 k-划分下,如果从源点 v_0 出发到节点 v 的路径 $p=((v_0,v_1),(v_1,v_2),\cdots,(v_{x-1},v_x),(v_x,v_{x+1}),\cdots,(v_{x+m},v_{y+1}),\cdots,(v_l,v))$满足条件: $v_{x-1},v_{y+1}\notin V_i$ 且 $v_x,v_{x+1},\cdots,v_{x+m},v\in V_i$,则称路径 p 穿越分区 D_i,称 v_x 为穿越点对.称路径$((v_x,v_{x+1}),\cdots,(v_{x+m},v))$为穿越路径.

定义 9 在图 G 的一个 k-划分下,如果从源点 v_0

出发到节点 v 的路径 $p=((v_0,v_1),(v_1,v_2),\cdots,(v_{x-1},v_x),(v_x,v_{x+1}),\cdots,(v_{x+m},v))$满足如下条件:$v_{x-1}\notin V_i$ 且 $v_x,v_{x+1},\cdots,v_{x+m},v\in V_i$,则称路径 p 进入 D_i,称 v_x 为进入点.称路径$((v_x,v_{x+1}),\cdots,(v_{x+m},v))$为进入路径.

2.2　算法步骤描述

由于问题的研究背景为可扩展路由器中 SPT 的分布式计算问题,因此我们假设可扩展路由器由 k 个路由节点组成.在此环境下,D-D 算法的算法步骤描述如下:

步骤 1　将网络拓扑图 G 进行 k-划分(如图 3 所示).

在此划分下,图 G 包含 $D_1,D_2,\cdots,D_k$ 这 k 个分区,各分区扩展边界节点集合为 $VB_1,VB_2,\cdots,VB_k$(如图 4 所示),内部节点集合为 $VI_1,VI_2,\cdots,VI_k$,图 G 的区间边集合为 E_{IDC}.

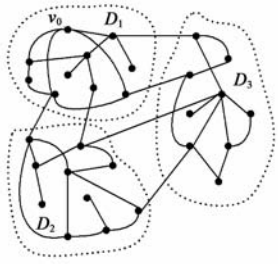

图 3　图 G 的 k-划分

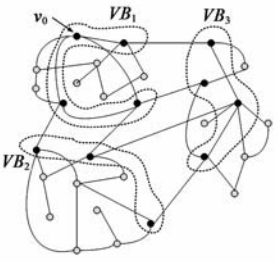

图 4　图 G 的各扩展边界节点集合

步骤 2　分区并行计算.

对于子图 G_i,分别以扩展边界节点集 VB_i 中的每一个节点为源点,采用 Dijkstra 算法计算子图 G_i 的 SPT.这样,每个子图 G_i 可以得到 d_i 个 T_i^j.每个路由节点完成一个分区内部的计算.分区中 $v_i^j(j=1,2,\cdots,d_i)$到分区 D_i 内部节点 v 的最短路径记为 $p_i(v_i^j,v)$.

步骤 3　构造完全图 G_i'.

通过步骤 2 的计算,我们可知 VB_i 内部任意两个扩展边界节点 v_i^l,v_i^m 间的最短路径值,记为 w_i^{lm}.构造完全图 $G_i'=(VB_i,E_i)$,其中:$E_i'=\{(v_i^l,v_i^m)\mid v_i^l,v_i^m\in VB_i'\}$ 且 $w(v_i^l,v_i^m)=w_i^{lm}$;

步骤 4　构造图 G'(如图 5 所示).

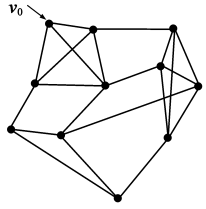

图 5　图 G'

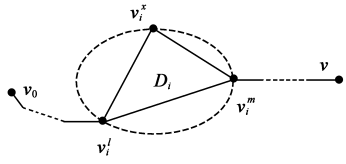

图 6　穿越路径示意图

构造图 $G'=(V',E')$,其中 $V'=(\bigcup_{i=1}^{k}VB_i)$,$E'=(\bigcup_{i=1}^{k}E_i')\cup E_{ICD}$.由 VB_i 定义可知,$v_0\in V'$.

步骤 5　计算图 G'中的以 v_0 为源点的 SPT.

该步骤可以采用经典 Dijkstra 算法计算以 v_0 为源点的最短路径树 T',将得到所有 $w'(v_0,v_i^l)$(其中 $i=1,2,\cdots,k;l=1,2,\cdots,d_i$).

步骤 6　合并计算图 G 中的以 v_0 为源点的最短路径树 T.

记图 G 中两点 v_i,v_j 间最短路径为 $p(v_i,v_j)$,记图 G'中两点 v_i,v_j 间最短路径为 $p'(v_i,v_j)$.在原图 G 中,$\forall v\in V$,如果 $v\in V'$,则 $p(v_0,v)=p'(v_0,v)$;

$\forall v\in V$,如果 $v\notin V'$,且 $v\in VI_i$,则 $w(p(v_0,v))=\min_{l=1}^{d_i}\{w(p'(v_0,v_i^l))+w(p_i(v_i^l,v))\}$.记上式取得最小值时 l 记为 $l_{\min}$,则 $p(v_0,v)=p'(v_0,v_i^{l_{\min}})+p_i(v_i^{l_{\min}},v)$.

经过以上 6 个步骤,我们完成了对图 G 中以 v_0 为源点的最短路径树计算.

该算法的并行计算部分主要在步骤 2.根据算法特点,各分区的内部计算互不相关,完全可以分布在 k 个路由节点中完成,且在此步骤中不存在相互通讯.我们在实际实现中,完全可以根据原图的规模平均分配给参与计算的各个路由节点.在并行计算结束后,合并计算的过程需要在单个路由节点中完成.

2.3　算法正确性证明

在本文前一部分的算法步骤基础上,我们有如下定理:

定理 1　在图 G 中,如果源点 v_0 到节点 $v\in VB_j(j=1,2,\cdots,k)$的最短路径 $p(v_0,v)$穿越分区 D_i,穿越点对为 v_i^l,v_i^m,其穿越路径记为 p_i^{lm}.则 $p(v_0,v)$由一系列区间边及穿越路径组成.对应图 G'中,将 $p(v_0,v)$中区间边保持不变,穿越路径 p_i^{lm} 以边(v_i^l,v_i^m)代替,则所得路径 $p'(v_0,v)$为 G'中 v_0 到节点 v 的最短路径.

证明:假设在 G'图中,$p'(v_0,v)$不是从源点 v_0 到 $v\in VB_j(j=1,2,\cdots,k)$的最短路径,则必定存在另外一条最短路径,不妨设为 $q'(v_0,v)$.则 $q'(v_0,v)$由一系列区间边及穿越路径$(v_i^l,v_j^m)(j=1,2,\cdots,k)$组成.将每个穿越路径$(v_i^l,v_j^m)(j=1,2,\cdots,k)$对应于图 G 中 T_j^l 中 v_j^l 到 v_j^m 的路径.经过替换后,路径 $q'(v_0,v)$对应于图 G 中的路径 $q(v_0,v)$.则由算法步骤 3 可知,$w(q(v_0,v))=w(q'(v_0,v))$.又由假设可知 $w(q'(v_0,v))<w(p'(v_0,v))=w(p(v_0,v))$,因此有 $w(q(v_0,v))<w(p(v_0,v))$,即 $p(v_0,v)$不是图 G 中的最短路径,矛盾.因此原命题成立.

定理 2　在图 G'中,如果源点 v_0 到节点 $v\in VB_j(j=1,2,\cdots,k)$的最短路径 $p'(v_0,v)$穿越分区 D_i,穿越点对为 v_i^l,v_i^m,则穿越路径必定为边(v_i^l,v_i^m).

证明:如图 6 所示,假如 $p'(v_0,v)$对分区 D_i 的穿越路径不是边(v_i^l,v_i^m),存在另外中间节点,假定为 v_i^x,则穿越路径为(v_i^l,v_i^x,v_i^m).由算法步骤 3 可知,$w(v_i^l,v_i^m)$

为 T_j^l 中 v_j^l 到 v_j^m 的路径权值,即为 v_j^l 到 v_j^m 的最短路径值,即 $w(v_i^l, v_i^m) < w(v_i^l, v_i^x, v_i^m)$. 显然该穿越路径不是最短路径 $p'(v_0, v)$的一部分,$p'(v_0, v)$对分区 D_i 的穿越路径只能是边(v_i^l, v_i^m).

定理 3 在图 G'中,如果源点 v_0 到节点 v 的最短路径$p'(v_0, v)$穿越分区 D_i,穿越路径为边(v_i^l, v_i^m). 则 $p'(v_0, v)$由一系列区间边及穿越路径组成. 对应图 G 中,将 $p'(v_0, v)$中区间边保持不变,穿越路径(v_i^l, v_i^m)以路径 p_i^{lm} 代替,则所得路径 $p(v_0, v)$为 G 中 v_0 到节点 v 的最短路径.

证明:假设图 G 中存在不同于路径 $p(v_0, v)$的 $q(v_0, v)$为最短路径,则 $q(v_0, v)$由一系列区间边及穿越路径组成,其中,存在与路径 $p(v_0, v)$不同的穿越路径. 对应图 G'中,将 $q(v_0, v)$中区间边保持不变,穿越路径 q_i^{lm} 以边(v_i^l, v_i^m)代替. 由于 $q(v_0, v)$存在与路径 $p(v_0, v)$不同的穿越路径,故 $q'(v_0, v)$是有异于 $p'(v_0, v)$的路径. 然而据定理 1,路径 $q'(v_0, v)$为 G'中 v_0 到节点 v 的最短路径. 矛盾.

定理 4 在图 G'中,如果源点 v_0 到节点 v 的最短路径$p'(v_0, v)$不穿越任何分区,则:

(1)若 v_0 与 v 在同一分区 D_i,图 G 中 v_0 到节点 v 的最短路径$p(v_0, v)$为分区 D_i 中 v_0 到 v 的最短路径.

(2)若 v_0 与 v 不在同一分区 D_i,$p'(v_0, v)$即为图 G 中 v_0 到节点 v 的最短路径$p(v_0, v)$.

证明:该定理为定理 3 的特殊情况,证明略.

定理 5 图 G 中,v_i^j 为分区 D_i 中边界节点,v 为分区 D_i 中内部节点. 如果 $p(v_i^j, v)$为 v_i^j 到 v 的最短路径不经过分区外节点,则 $p(v_i^j, v) = p_i(v_i^j, v)$,即 T_i^j 中 v_i^j 到 v 的路径.

证明:根据 T_i^j 的定义,$p_i(v_i^j, v)$为分区内 v_i^j 到 v 的最短路径,该命题显然成立.

以下证明本文 1.2 算法的正确性. 给定一个图 G 的 k-划分后,G 中以 v_0 为源节点的最短路径树中,除源节点 v_0 外,其他节点 v 即可以分为两类:边界节点和内部节点.

假如节点 $v \in V$ 属于边界节点,且在 T'中,v_0,v 间最短路径为 $p'(v_0, v)$. (1)如果 $p'(v_0, v)$不穿越任何一个分区,根据定理 4,$p(v_0, v) = p'(v_0, v)$. (2)如果 $p'(v_0, v)$穿越某个分区,根据定理 3,$p(v_0, v) = p'(v_0, v)$.

假如节点 $v \in V$ 属于内部节点,不失一般性,设 $v \in VI_i$. 在图 G 中,设源点 v_0 到节点 v 的最短路径为 $p(v_0, v)$,则 $p(v_0, v)$在进入 D_i 时必定存在某个进入点 v_i^l,故 $p(v_0, v) = p(v_0, v_i^l) + p(v_i^l, v)$. 显然,$p(v_0, v_i^l)$为图 G 中 v_0 到节点 v_i^l 的最短路径,$p(v_i^l, v)$为 v_i^l 到节点 v 的最短路径. 据定理 1,$p(v_0, v_i^l) = p'(v_0, v_i^l)$. 据定义 9 及定理 5,$p(v_i^l, v) = p_i(v_i^l, v)$. 因此,$w(p(v_0, v)) = \min_{l=1}^{d_i}\{w(p'(v_0, v_i^l)) + w(p_i(v_i^l, v))\}$,且上式取得最小值时 l 记为 $l_{\min}$,$p(v_0, v) = p'(v_0, v_i^{l_{\min}}) + p_i(v_i^{l_{\min}}, v)$.

3 算法性能分析

带正权的无向图 $G = (V, E)$,节点数设为 N,即 $N = |V|$. 当系统有 k 个路由节点时,需要将需要将图 G 进行 k-划分. 为使每一个处理器的计算任务基本相当,k-划分中,除最后一个分区外,其他路由节点分配相同节点规模的分区. 则每个分区的节点数 $N_i = |V_i| = \lfloor N/k \rfloor$(最后一个分区除外). 此外,我们设图 G 的 k-划分下,每个分区的平均边界节点数为 M. 则 $M \leqslant \lfloor N/k \rfloor$.

在算法步骤 1 中:将图 G 进行 k-划分的算法可以基于广度优先遍历. 具体方法是首先将一个未分配的节点加入到当前要分配的块中去,再从这个节点出发,广度优先遍历未分配的节点,按遍历次序将节点加入到当前要分配的块中去,直到块中的节点数目达到预先设定. 该步骤的时间复杂度为 $O(N)$.

在算法步骤 2 中:上一步骤中平均每个分区有 M 个边界节点. 由于对每个边界节点都要对子图用一次 Dijkstra 算法,每个分区的计算复杂度是 $O(M(N/k)^2)$.

在算法步骤 3 中:构造得到的图 G'中,节点数为 $k \times M$(所有块的边界节点). 对图 G'使用 Dijkstra 算法复杂度为 $O(k^2M^2)$.

三个步骤的总复杂度为 $O(N + N^2M/k^2 + k^2M^2)$.

考虑最坏情况,如果图 G 是完全图,则无论如何划分,每个分区的所有节点都是边界节点(任意两节点将均有边相连),此时 $M = N/k$,算法复杂度变为 $O(N + N^4/k^4 + N^2)$. 从上述分析可以看出,当 $k \geqslant \sqrt{N}$时,该并行算法的复杂度不会超过经典 Dijkstra 算法,此时算法复杂度为 $O(N^2)$,退化为单路由节点计算 SPT 情况. (事实上,如果图 G 是完全图,我们在计算中不需要进行步骤 1、2,因此算法仍然退化为单路由节点计算 SPT 情况.)

考虑最佳情况,在对图 G 进行 k-划分后,每个分区的边界节点只有一个,即 $M = 1$. 此时算法总复杂度为 $O(N + N^2M/k^2 + k^2M^2) = O(N + N^2/k^2 + k^2)$,此时,当 $k = \sqrt{N}$时,算法复杂度达到最优,为 $O(N)$.

因此,在采用上述算法进行分布式 *SPT* 计算时,算法复杂度可控制在 $O(N)$与 $O(N^2)$之间,算法将优于单路由节点计算.

4 实验验证

由于实验无法验证所有图的拓扑结构,简化起见,

本文图例将采用均匀图.定义均匀图 $G=(V,E)$,节点集 $V=\{v_i,0\leqslant i<N\}$,给定控制边密度的参数 d,定义边集 $E=\{(p_i,p_j)\mid N-\mid i-j\mid <d\times\sqrt{N}$或$\mid i-j\mid <d\times\sqrt{N}$,其中 $p_i,p_j\in V,(0\leqslant i,j<N)\}$,边权值随机给定均为值 $\omega_{ij}(0<\omega_{ij}<\infty)$.符合该定义的图从拓扑上看,为分布均匀的带弦环(但权值分布不均匀).如果补充定义当 ω_{ij}取值大于一定阈值时,p_i,p_j 之间为断路,则根据该定义可以生成随机图.因此,该定义所生成的图具有一定的代表性.

实验环境为多个路由节点,它们通过内联网络形成路由节点群,协同实现 D-D 算法.当路由节点数目为 1 时,实验结果为单路由节点计算效果.实际实验条件采用配置完全相同的 PC 机模拟路由节点功能,采用百兆以太网实现内部互联网络.实验环境如图 7 所示:

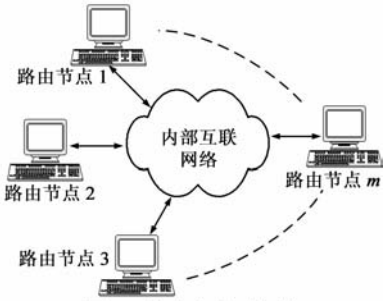

图 7　仿真实验示意图

实验 1　验证路由节点数递增的情况下,D-D 算法的计算效率变化曲线,同时验证不同节点规模下,计算效率变化曲线的区别.

图 8 是不同数目的路由节点参与计算的情况下,计算时间的变化曲线.其横轴为路由节点数,纵轴是计算时间.其中路由节点个数为 1 时采用经典 Dijkstra 算法计算.不同颜色的曲线分别是针对不同规模图(节点总数从 8000 到 12000)的测试结果.

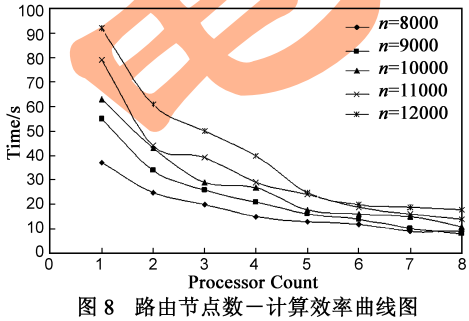

图 8　路由节点数－计算效率曲线图

从图 8 中可以看出,对于某特定规模的拓扑图,当路由节点增加时,采用 D-D 算法计算 SPT 的时间是逐步递减的,明显优于单路由节点计算.同时,图的规模上升,计算时间会相应有所增加,这一点比较容易理解.

实验 2　验证 D-D 算法随路由节点数增加,相比单路由节点计算效果的提高程度.

图 9 的横轴是 D-D 算法的计算时间,纵轴是单路由节点采用 Dijkstra 算法计算时间,这样纵轴与横轴的比值(或者是过原点直线的斜率)表示 D-D 算法计算效果的提高程度.每组曲线参与计算的路由节点个数相同.虚线是各组数据直线拟和的结果.

图 9 验证了 D-D 算法对单节点计算的提高程度.从实验曲线及拟合直线可以看出,在图的规模一定的情况下,当路由节点数目增加时,实验数据曲线基本可以拟合为直线.即随路由节点的增加,D-D 算法对 SPT 计算效率的提高是线性的.

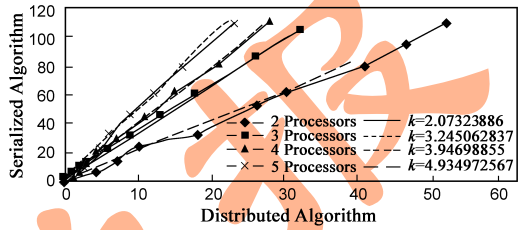

图 9　单路由节点－多路由节点计算效率比较曲线

实验 3　验证图中边的密度对算法性能的影响,对边密度不同的图例的计算时间进行比较.图 10 的横轴是图的规模(节点数),纵轴为计算时间.各条曲线为参数 d 不同的均匀图采用 D-D 算法的计算结果;所有数据均采用 3 个路由节点并行计算得到.图 10 中可以看出,当 d 比较小时,即边密度较小情况下,随图规模的增加,计算时间的增长比较缓慢;随着 d 增加、从而边密度增大,计算时间会增加,但是增长曲线逐渐变“陡”,但是其增长趋势不会超过最复杂的图－完全图的计算曲线(从图 10 中可以看出,完全图和边密度较大的图(如图中 $d=0.5$ 曲线的测试例)所用时间相差无几,时间曲线几乎重合).完全图曲线为 2 次曲线.即该算法即使在最坏情况下增长曲线也不会超过 2 次曲线.

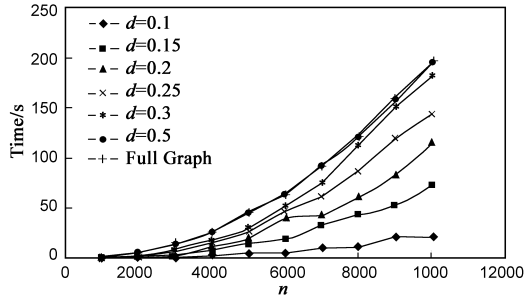

图 10　规模－计算效率曲线图

5　小结

本文研究背景为可扩展路由器中的分布式 OSPF 路由问题,提出了可实现 SPT 分布计算的新算法 D-D 算法,论证了算法正确性,并通过实验验证了算法性能.实验表明该分布式算法在合适的条件下,计算效率高于传统 Dijkstra 算法,突破了可扩展路由器体系结构

中分布式 OSPF 研究的难点.后续研究将集中于图 G 的 k-划分算法、D-D 算法对拓扑更新情况下的处理.

参考文献:

[1] H J Chao. Next generation routers[J]. Proceedings of the IEEE,2002,90(9):1518 – 1558.

[2] Lijun Fan, Chengbin Quan, Guixing Luan. Scalable distributed architecture of the terabit router[A]. PDCAT'2003[C]. Chengdu: IEEE Press,2003.472 – 475.

[3] 张晓哲,卢锡城,朱培栋,等.一种集群路由器转发表同步框架及关键算法[J].软件学报,2006,17(3):445 – 453.
Zhang Xiaozhe, Lu Xicheng, Zhu Peidong, et al. A synchronization framework and critical algorithm maintaining single image of IP forwarding tables among cluster router's nodes [J]. Journal of Software,2006,17(3):445 – 453. (in Chinese)

[4] William J Dally. Scalable switching fabrics for Internet routers [R]. USA: Computer Systems Laboratory, Stanford University and Avici Systems Inc,1999.

[5] RFC3746. ForCES[S].

[6] Manasi Deval, Hormuzd Khosravi. Distributed control plane architecture for network elements[J]. Intel Technology Journal, 2003,7(4):51 – 63.

[7] Henry C B Chan, Hussein M Alnuweiri, Victor C M Leung. A framework for optimizing the cost and performance of next-generation IP routers[J]. IEEE Joural on Selected Areas in Communications,1999,17(6):1013 – 1029.

[8] Sundar Iyer, Nick McKeown. Analysis of the parallel packet switch architecture[J]. IEEE ACM Transactions on Networking,2003,11(2):314 – 324.

[9] H Jonathan Chao, Kungli Deng, Zhigang Jing. PetaStar: a petabit photonic packet switch[J]. IEEE Journal on Selected Areas in Communications,2003,21(7):1096 – 1112.

[10] Jan Cheyns, Chris Develder, Didier Colle, et al. Clos lives on in optical packet switchin[J]. IEEE communications magazine,2004,114 – 121.

[11] Cisco carrier routing system[EB/OL]. http://www.cisco.com/en/US/products/ps5763/,2006-09-25.

[12] T640 routing node and TX Matrix? platform: architecture [EB/OL]. http://www.juniper.net/solutions/literature/white-papers/200089.pdf,2006-09-25.

[13] Anindya Basu, Jon G Riecke. Stability issues in OSPF routing [A]. SIGCOMM'01[C]. San Diego, California: IEEE Inc, 2001.225 – 236.

[14] Shan Zhu, Garng M Huang. A new parallel and distributed shortest path algorithm for hierarchically clustered data networks[J]. IEEE Transactions on Parallel and Distributed Systems,1998,9(9):841 – 855.

[15] John K Antonio, Garng M Huang, Wei K Tsai. A fast distributed shortest path algorithm for a class of hierarchically clustered data networks[J]. IEEE Transactions on Computers, 1992,41(6):710 – 724.

[16] Edith Cohen. Efficient parallel shortest-paths in digraphs with a separator decomposition[A]. ACM SPAA[C]. Velen Germany: ACM Press,1993.57 – 67.

[17] Romeijn H E Smith R L. Parallel algorithms for solving aggregated shortest-path problems[J]. Computers and Operations Research,1999,26(10 – 11):941 – 953.

作者简介:

张小平 男,1975 年生于内蒙古.清华大学计算机科学与技术系在职博士.研究方向为路由器体系结构、分布式路由.
E-mail:zhxp@tsinghua.edu.cn

吴建平 男,1953 年生于山东巨野,博士,教授,博士生导师.主要研究领域为计算机网络体系结构、协议工程学、互联网络.