基于备份的可重构服务承载网可靠性映射方法

王志明* 汪斌强
(国家数字交换系统工程技术研究中心 郑州 450002)

摘 要：可重构柔性网络链路失效将严重影响其上承载的可重构服务承载网(RSCN)的可靠性。文章基于路径备份策略着重解决时延敏感类型RSCN的可靠性问题，并提出分阶段处理方式进一步优化备份资源消耗。在拓扑预处理阶段，根据RSCN是否支持路径分裂分别提出分裂的最小备份拓扑生成(S-MBT-Gen)算法和最小备份生成树(MBST- Gen)算法，减小备份拓扑带宽约束总量；在拓扑映射阶段，提出主备拓扑协同映射(RNM-PBT)算法，协调利用底层网络资源。仿真结果表明，本文提出的算法降低了RSCN平均资源消耗，且具有较高的请求接受率和较低的平均执行时间。

关键词：可重构柔性网络；可重构服务承载网；可靠性映射；最小备份拓扑；时延敏感
中图分类号：TP393 **文献标识码**：A **文章编号**：1009-5896(2013)01-0126-07
DOI: 10.3724/SP.J.1146.2012.00602

Reliable Mapping Method for Reconfigurable Service Carrying Network Based on Path Backup

Wang Zhi-ming Wang Bin-qiang
(*National Digital Switching System Engineering & Technology Research Center*, *Zhengzhou* 450002, *China*)

Abstract: The substrate link failures have made a great impact on the reliability of Reconfigurable Service Carrying Networks (RSCNs) over Reconfigurable Flexible Network (RFNet). In this paper, the reliability problem of delay-sensitive RSCNs is solved based on a path backup strategy, and a two-stage approach is presented to further reduce the backup resource cost. In the topology preprocessing stage, according to whether the path split is supported by RSCNs, the Splittable Minimum Backup Topology Generation (S-MBT-Gen) and Minimum Backup Spanning Tree Generation (MBST-Gen) algorithms are respectively proposed to reduce the total bandwidth constraints of backup topology. In the topology mapping stage, a Reconfigurable service carrying Network Mapping algorithm with coordinated Primary and Backup Topology (RNM-PBT) is proposed to make the best of the substrate network resource. The simulation experiments show that our proposed algorithms can reduce the average resource cost and execution time cost, while improving the request accepted ratio of RSCNs.

Key words: Reconfigurable flexible network; Reconfigurable Service Carry Network (RSCN); Reliable mapping; Minimum backup topology; Delay-sensitive

1 引言

当前互联网体系结构是僵化的[1]，为了以最小代价适应新业务，只能依靠设备升级、带宽扩展或改进网络协议等修补方式。针对这一痼疾，可重构柔性网络(Reconfigurable Flexible Network, RFNet)体系结构[2]被提出，该网络以可重构路由交换平台为支撑，通过节点重构技术在同一柔性网络之上构建多个可重构服务承载网(Reconfigurable Service Carrying Network, RSCN)，从而满足了用户多样化需求，并为解决网络僵化问题提供一种创新思路。

如何保证底层链路异常时的可靠性(即承载网的快速恢复与服务连续性)是当前研究亟待解决的问题。目前，可靠性研究主要分为迁移和备份两种策略，迁移策略的资源消耗小、恢复时间长，备份策略的资源消耗大、恢复时间短。考虑到大部分RSCN对时延特性较为敏感，如面向语音、视频类业务的RSCN，时延会造成语音变调和视频马赛克等现象，本文着重解决时延敏感RSCN的可靠性问题，并基于路径备份策略提出可靠性映射方法。

现有备份策略的研究工作主要集中在IP-over-WDM分层网络、MPLS网络以及虚拟网生存性等方面。针对IP-over-WDM分层网络，Lee等人[3]扩

2012-05-18收到，2012-10-22改回
国家973计划项目(2012CB315901, 2012CB315905)和国家863计划项目(2011AA01A103)资助课题
*通信作者：王志明 wangzm05@gmail.com

展了最大流最小割理论，提出了新的连通性度量指标，并利用整数线性规划求解技术最大化分层网络的连通性；Krishnaiyan 等人[4]利用环路和割集的对偶关系提出了 CIRCUIT-SMART 和 CUTSET-SMART 算法，从而有效缩短计算时间。针对 MPLS 网络，文献[5,6]对其进行了深入研究，并提出了带宽资源消耗最小算法。针对虚拟网生存性，Rahman 等人[7]提出了一种提供虚拟网生存性的映射算法，该算法在映射过程中为每条链路请求提供备份路径，从而达到提高服务连续性的目的；Chen 等人[8]提出了备份链路带宽智能共享的 Pardalis 算法，该算法既保证了虚拟网在单链路故障情况下的快速恢复，又能够降低备份方式的带宽资源消耗；Yu 等人[9]针对节点故障提出了 1 冗余和 k 冗余虚拟网拓扑增强算法，从而以较小的资源冗余保证虚拟网的可靠性；在文献[10]中，又针对虚拟网区域故障建立了最小资源消耗的混合整数线性规划模型，并提出了 SOUM 和 IOCM 启发式算法进行最优化求解。

上述研究存在以下不足：(1)只在解决节点故障问题[9,10]时对请求拓扑进行预处理，而未应用于链路故障问题，使得链路故障问题的备份拓扑等于主拓扑[7,8]，不仅增加备份拓扑映射的时间消耗，而且使备份拓扑的带宽共享缺乏稳定性，影响了资源消耗的优化程度；(2)主、备份拓扑映射过程相互独立，无法协调利用底层资源，降低了映射成功率。

本文针对单链路故障下的 RSCN 可靠性，提出采用分阶段处理方式进一步优化备份资源消耗：首先在拓扑预处理阶段，提出路径可分裂和不可分裂情况下的最小备份拓扑生成算法，实现 RSCN 内部的备份带宽优化共享，保证带宽共享的稳定性；然后在拓扑映射阶段，提出主备拓扑协同映射算法，保证底层网络资源的充分利用，提高映射成功率。

2　模型分析

2.1 网络模型

底层网络　无向图 $G^s=(N^s,L^s,A^n,A^l)$ 表示底层网络，其中 N^s 和 L^s 分别表示底层网络的节点和链路集合，$A^n(n_s)$ 和 $A^l(l_s)$ 分别表示底层网络节点 n_s 的交换资源和链路 l_s 的带宽资源。此外，定义 $B_L(e)$ 表示底层链路 e 为所有 RSCN 链路请求分配的备份带宽资源，$R_L(e)$ 表示底层链路 e 的剩余带宽资源，$R_N(u)$ 表示底层节点 u 的剩余资源。

RSCN 主拓扑　第 i 个到达的 RSCN 请求用 $\mathrm{Req}_i=(G_i^v,T_a,T_d)$ 表示，其中无向图 $G_i^v=(N_i^v,L_i^v,C_i^n,C_i^l,\mathrm{Loc}_i)$ 表示 RSCN 请求的拓扑结构，N_i^v 和 L_i^v 分别为节点和链路请求集合，C_i^n 和 C_i^l 分别为节点和链路请求的资源约束，Loc_i 为节点的地理位置约束，即每个节点请求允许映射的物理节点集合；T_a 和 T_d 表示 RSCN 运行的起止时间。

RSCN 备份拓扑　无向图 $G^b=(N^b,L^b,C^b)$ 表示备份链路请求所组成的备份拓扑，其中 L^b 和 N^b 分别表示备份链路请求集合和所涉及的节点请求集合，C^b 表示备份链路请求的带宽约束。L^b 并不一定是 RSCN 链路请求集合 L_i^v 的子集，但一定有 $N^b=N_i^v$ 成立。

主拓扑映射　底层网络为 RSCN 主拓扑 G_i^v 分配资源的过程称之为主拓扑映射，且需要满足约束条件 C_i^n，C_i^l 和 Loc_i，表达式如下：

$$f:G_i^v(N_i^v,L_i^v)\mapsto G^{\mathrm{sub}}(N^{\mathrm{sub}},P^{\mathrm{sub}}) \tag{1}$$

$$\left.\begin{aligned}&\forall n_v\in N_i^v,\forall n_s\in f_N(n_v),M_N(n_s,n_v)\ge C_i^n(n_v),\\&\quad n_s\in\mathrm{Loc}_i(n_v)\\&\forall l_v\in L_i^v,\forall l_s\in f_L(l_v),M_L(l_s,l_v)\ge C_i^l(l_v)\end{aligned}\right\} \tag{2}$$

G^{sub} 为 G^s 的子图，$N^{\mathrm{sub}}\subset N^s$，$P^{\mathrm{sub}}\subset P^s$，$P^{\mathrm{sub}}$ 和 P^s 分别为 G^{sub} 和 G^s 的路径集合，因此将 f 分为节点映射 $f_N:N_i^v\mapsto N^{\mathrm{sub}}$ 和链路映射 $f_L:L_i^v\mapsto P^{\mathrm{sub}}$。$M_N(n_s,n_v)$ 表示底层节点 n_s 为节点请求 n_v 提供的交换资源，$M_L(l_s,l_v)$ 表示底层链路 l_s 为链路请求 l_v 提供的带宽资源，通常取 $M_N(n_s,n_v)=C_i^n(n_v)$，$M_L(l_s,l_v)=C_i^l(l_v)$。如图 1(b)所示，节点请求 a, b, c 和 d 分别映射到物理节点 E, A, C 和 G，链路请求的映射路径如带箭头的实线所示。

备份拓扑映射　底层网络为 RSCN 备份拓扑 $G^b=(N^b,L^b,C^b)$ 分配资源的过程称之为备份拓扑映射，且需要满足备份带宽约束 C^b，表达式如下：

$$f^b:L^b\mapsto P^b \tag{3}$$

$$\forall l_b\in L^b,\forall l_s\in f^b(l_b),\mathrm{BM}(l_s,l_b)\ge C^b(l_b) \tag{4}$$

$P^b\subset P^s$ 表示底层网络为备份链路请求集合 L^b 提供的备份路径集合，$\mathrm{BM}(l_s,l_b)$ 表示底层链路 l_s 为

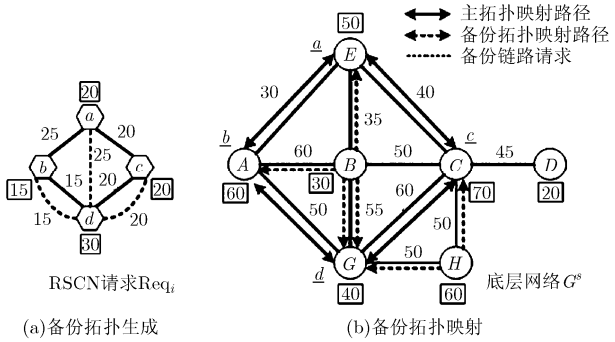


图 1 分阶段方式下的 RSCN 可靠性映射实例

备份链路请求 l_b 提供的备份带宽值，通常取 $\mathrm{BM}(l_s, l_b)=C^b(l_b)$。与主拓扑映射不同，备份拓扑映射中存在备份链路带宽的共享。如图 1(b)所示，备份链路请求 ad、bd 和 cd 分别映射到备份路径 EBG, ABG 和 CHG，且 EBG 和 ABG 在 BG 处存在备份带宽的共享。

2.2 问题分析

RSCN 可靠性映射所产生的资源消耗包括 3 部分：节点映射，主链路映射和备份链路映射。设每种资源的权重为 ω_i，则可以得到 RSCN 请求 Req_i 的资源消耗总量：

$$C(\mathrm{Req}_i)=\omega_1\sum_{n_v\in N_i^v}C_i^n(n_v)+\omega_2\sum_{l_v\in L_i^v}C_i^l(l_v)\cdot|f_L(l_v)|+\omega_3\sum_{l_s\in\mathrm{LS}}\mathrm{BA}(l_s) \tag{5}$$

$\mathrm{BA}(l_s)$ 表示底层链路 l_s 实际分配的备份带宽资源，LS 表示 P^b 所对应的底层链路集合：

$$\mathrm{LS}=\bigcup_{l_b\in L^b}f^b(l_b) \tag{6}$$

由于本文的备份拓扑映射算法允许共享备份链路带宽，所以有下式成立：

$$\sum_{l_b\in L^b}\mathrm{BM}(l_s,l_b)\ge\mathrm{BA}(l_s) \tag{7}$$

因此，除了关注主拓扑映射外，还可以通过减小 RSCN 备份拓扑规模和使用备份带宽共享方式来降低备份链路映射所产生的资源消耗，后文在第 3、第 4 节将分别提出具体算法。

3 最小备份拓扑生成算法

3.1 带路径分裂的最优化算法

将 RSCN 主拓扑 $G^v=(N^v,L^v,C^n,C^l,\mathrm{Loc})$ 的每条边转换为两条具有相同带宽约束的反向边，从而转换为有向图。同时，定义如下符号：$B(l_b,l_v)$ 表示备份链路请求 l_b 为主链路请求 l_v 提供的备份带宽总量；$B^h(l_b,l_v)$ 表示从主链路请求 $l_v=(s,t)$ 的起点 s 经过 h 跳到达备份链路请求 $l_b=(s',t')$ 的起点 s' 时，l_b 为 l_v 提供的备份带宽值；h 表示路径跳数，在节点请求数为 $N=|N^v|$ 的 RSCN 中，$h\in[0,N-1]$，且有 $\sum_{h=0}^{N-1}B^h(l_b,l_v)=B(l_b,l_v)$ 成立；$O(u)=\{\overrightarrow{uv}\,|\,\overrightarrow{uv}\in L^b\}$ 表示从 RSCN 节点请求 u 出发的所有备份链路请求集合；$I(v)=\{\overrightarrow{uv}\,|\,\overrightarrow{uv}\in L^b\}$ 表示到达 RSCN 节点请求 v 的所有备份链路请求集合。由于备份拓扑是任意的，所以初始化 $L^b\leftarrow N^v\times N^v$，若 $\exists l_b\in L^b, C^b(l_b)=0$，则表明 l_b 不是备份链路，将其从 L^b 中去除。

基于上述定义，建立带路径分裂的最小备份拓扑生成问题模型(S-MBT-LP)：

参数： $G^v=(N^v,L^v,C^n,C^l,\mathrm{Loc})$，$L^b$

变量： $C^b(l_b)$，$B^h(l_b,l_v)$（$l_b\in L^b$，$l_v\in L^v$，$h\in[0,N-1]$）

目标函数：

$$\min\sum_{l_b\in L^b}C^b(l_b) \tag{8}$$

约束条件：

$$\sum_{l_b\in O(u)}B^h(l_b,l_v)=\sum_{l_b\in I(u)}B^{h-1}(l_b,l_v),\ \forall l_v=(s,t)\in L^v,\ \forall u\in N^v\setminus\{s,t\},\forall h\in[0,N-1] \tag{9}$$

$$\sum_{l_b\in O(s)}B^0(l_b,l_v)\ge C^l(l_v),\ \forall l_v=(s,t)\in L^v \tag{10}$$

$$\sum_{l_b\in I(t)}\sum_{h=0}^{N-1}B^h(l_b,l_v)\ge C^l(l_v),\ \forall l_v=(s,t)\in L^v \tag{11}$$

$$\sum_{h=0}^{N-1}B^h(l_b,l_v)\le C^b(l_b),\ \forall l_v\in L^v,\forall l_b\in L^b \tag{12}$$

$$B^h(l_b,l_v)=B^h(l_b',l_v),\ \forall l_v\in L^v,\forall h\in[0,N-1],\ \forall l_b=(s,t)\in L^b,l_b'=(t,s) \tag{13}$$

$$C^b(l_b)\ge 0,\ \forall l_b\in L^b \tag{14}$$

$$B^h(l_b,l_v)\ge 0,\ \forall l_b\in L^b,\forall l_v\in L^v,\forall h\in[0,N-1] \tag{15}$$

设 $|N^v|=N$，$|L^v|=M$，则 S-MBT-LP 问题的变量数和约束条件数均为 $O(N^3M)$ 级别。由于求解线性规划问题的时间复杂度为变量数和约束条件数的多项式[11]，所以 S-MBT-LP 问题求解可以在多项式时间内完成。带路径分裂的最小备份拓扑生成算法 (Splittable Minimum Backup Topology Generation algorithm, S-MBT-Gen)如表 1 所示。

表 1 S-MBT-Gen 算法

算法 1 S-MBT-Gen

输入： $G^v(N^v,L^v,C^n,C^l,\mathrm{Loc})$

输出： $G^b(N^b,L^b,C^b)$, $B(*,*)$

(1) $N^b\leftarrow N^v$, $L^b\leftarrow N^v\times N^v$;

(2) 利用 CPLEX 工具[12]在多项式时间内求解 S-MBT-LP 问题，得到 $C^b(l_b)$ 和 $B^h(l_b,l_v)$（$\forall l_b\in L^b$，$\forall l_v\in L^v$，$\forall h\in[0,N-1]$）;

(3) 去除 L^b 中所有 $C^b(l_b)==0$ 的备份链路请求 l_b;

(4) 计算 $B(*,*):\forall l_b\in L^b,\forall l_v\in L^v, B(l_b,l_v)=\sum_{h=0}^{N-1}B^h(l_b,l_v)$;

(5) **return** $G^b(N^b,L^b,C^b)$, $B(*,*)$;

根据步骤(2)可知 S-MBT-Gen 算法的时间复杂度为多项式级别。另外考虑到 RSCN 请求的节点数 N 和链路数 M 均较小，S-MBT-Gen 算法的平均执行时间仍可接受。

3.2 无路径分裂的启发式算法

无路径分裂的最小备份拓扑生成问题模型(U-MBT-ILP)可以形式化为 0-1 整数线性规划问题，只需将式(15)替换为下式：

$$B^h(l_b,l_v)=i\cdot C^l(l_v), i=0,1, \forall l_b\in L^b, \forall l_v\in L^v, \forall h\in[0,N-1] \tag{16}$$

由于0-1整数线性规划问题隶属NP-hard范畴，所以 U-MBT-ILP 问题的最优化求解无法在多项式时间复杂度下完成。鉴于此，本节提出最小备份生成树算法(Minimum Backup Spanning Tree Generation algorithm, MBST-Gen)进行近似求解。

最小备份生成树算法的基本思想是：限定备份拓扑为主拓扑的生成树(Spanning Tree)；对于生成树上的备份链路请求，其带宽总量与对应的主链路请求保持一致；对于不在生成树中的主链路，根据生成树的特性可知，必然存在唯一一条备份路径对其进行备份；算法只需将每个回路中的最小带宽链路从生成树中排除，从而使得回路中其他链路请求所组成的路径能够为该链路请求提供足够的共享备份带宽。算法的具体流程如表 2 所示。

表 2 MBST-Gen 算法

算法 2　MBST-Gen
输入： $G^v(N^v,L^v,C^n,C^l,\mathrm{Loc})$
输出： $G^b(N^b,L^b,C^b)$, $B(*,*)$
(1) $N^b\leftarrow N^v$, $L^b\leftarrow\varnothing$, $i\leftarrow 1$;
(2) 对 L^v 中的链路请求按照带宽大小进行升序排列：$l_1,l_2,\cdots,l_M$;
(3) **while** $i\le M$ **do**
(4)　**if** l_i 与 L^b 中的链路未形成回路 **then**
(5)　$l_{\mathrm{new}}\leftarrow l_i, L^b\leftarrow L^b\cup\{l_{\mathrm{new}}\}, C^b(l_{\mathrm{new}})=C^l(l_i)$;
//添加备份链路请求
(6)　**else**
(7)　l_i 与 L^b 中的链路形成回路 S，得到 S 中带宽容量最小的备份链路 $l_{\min}$;
(8)　$l_{\mathrm{new}}\leftarrow l_i, L^b\leftarrow(L^b\setminus\{l_{\min}\})\cup\{l_{\mathrm{new}}\}, C^b(l_{\mathrm{new}})=C^l(l_i)$;
//将备份链路请求 $l_{\min}$ 替换为 l_i
(9)　**end if**
(10)　$i\leftarrow i+1$;
(11) **end while**
(12) $\forall l_b\in L^b, \forall l_v\in L^v, B(l_b,l_v)=0$;
(13) **for each** $l\in L^v$ **do**
(14)　**if** $l\in L^b$ **then** $B(l,l)=C^l(l)$;
(15)　**else** 得到 L^b 中与 l 构成回路的路径 P，且 $\forall l_b\in P$, $B(l_b,l)=C^l(l)$; **end if**
(16) **end for**
(17) **return** $G^b(N^b,L^b,C^b)$, $B(*,*)$;

上述算法中，步骤(1)–步骤(11)的时间复杂度与图论中求解最小生成树的 Kruskal 算法相同，即 $O(M\log M)$；步骤(15)的时间复杂度不超过 $O(N^2)$，所以步骤(13)–步骤(16)的时间复杂度为 $O((M-(N-1))\cdot N^2)=O(M\cdot N^2)$，其中生成树的规模 $|L^b|=N-1\le M$。综上所述，MBST-Gen 算法的时间复杂度为 $O(M\cdot N^2)$。

MBST-Gen 算法得到的最小备份生成树为 RSCN 主拓扑的子图，根据文献[6]可知，子图约束下的备份带宽约束总量与最优解的比值在[2(1–1/N),2]区间内，N 为 RSCN 主拓扑的节点数，所以 MBST-Gen 算法能够得到近似解。

4　主备拓扑协同映射算法

定义 $\omega(l_s)$ 表示底层链路 $l_s\in L^s$ 为备份链路请求 $l_b\in L^b$ 提供备份的代价。$\omega(l_s)$ 的取值分 3 种情况：(1)若备份带宽总量 $B_L(l_s)$ 减掉冲突备份带宽 NS，剩余备份带宽仍然能够满足 l_b 的备份带宽约束 $C^b(l_b)$，则只需共享这部分带宽，此时 $\omega(l_s)$ 为 0; (2)若剩余备份带宽不能满足 l_b 的备份带宽约束 $C^b(l_b)$，则需要从空闲带宽 $R_L(l_s)$ 中分配额外带宽，此时 $\omega(l_s)$ 大于 0; (3)若空闲带宽也无法满足 l_b 的备份带宽约束 $C^b(l_b)$，则 $\omega(l_s)$ 为无穷大。公式如下：

$$\omega(l_s)=\begin{cases}0, & \text{if } C^b(l_b)+NS\le B_L(l_s)\\ C^b(l_b)+NS-B(l_s), & \text{if } C^b(l_b)+NS>B_L(l_s)\\ & \text{and } R_L(l_s)>C^b(l_b)+NS-B_L(l_s)\\ \infty, & \text{其它}\end{cases} \tag{17}$$

其中，冲突备份带宽 NS 的存在主要源于 l_s 上可能存在其他备份路径(包括属于其他 RSCN 备份拓扑 G_i^b)，如果这些备份路径的主路径集合与 l_b 的主路径集合 Ps 存在交集，那么交集链路的备份带宽不能被共享，否则在交集链路故障时会出现其他备份路径与 l_b 的备份路径均经过 l_s，从而产生带宽冲突。具体公式如下：

$$NS=\max_{l_a\in Ps}\{NS(l_s,l_a)\} \tag{18}$$

$$Ps=\bigcup_{l_v\in L^v, B(l_b,l_v)>0} f_L(l_v) \tag{19}$$

$$NS(l_s,l_a)=\max_i\{B(l_b',l_v')\}, l_b'\in G_i^b, l_v'\in G_i^v, l_s\in f^b(l_b'), l_a\in f_L(l_v') \tag{20}$$

根据上述设定，采用备份带宽共享策略的备份链路映射算法(Backup Link Mapping Algorithm, BLMA)流程如表 3 所示。

上述算法最复杂部分为公式(18)的计算，其最坏情况时间复杂度为 $O(K\cdot M)$，K 表示所有映射到 l_s 上的备份链路请求数，$M=|L^v|$。因此该算法的时间复杂度为 $O(m\cdot K\cdot M+n^2)$，其中 n 和 m 分别表示底层网络节点数和链路数。

基于 BLMA 得到 RSCN 主备拓扑协同映射算法(Reconfigurable service carrying Network Mapping with coordinated Primary and Backup

表 3 BLMA 流程

算法 3 BLMA
输入： $G^s(N^s, L^s, A^n, A^l)$, $l_b \in L^b$, $B(l_b, *)$
输出： $p_b \in P^b$
(1) 去除 L^s 中需要 l_b 提供备份的底层链路集合 Ps (见式(19))，得到新的底层拓扑 G^s_{new}；
(2) 计算 G^s_{new} 中所有底层链路 l_s 的备份代价 $\omega(l_s)$ (见式(17));
(3) 根据 l_b 两端点的映射位置 u、v，得到从 u 到 v 的备份代价和最小的路径 p_b；
(4) 若 p_b 的备份代价和非 ∞，则返回 p_b；否则返回 NULL;

Topology, RNM-PBT)，其基本思想为：(1)按照特定顺序进行节点请求映射，每次完成节点映射后，检查与该节点相邻的主链路请求或备份链路请求的另一端点是否已完成映射；(2)若完成，则对该链路请求进行映射：对于主链路请求，构建与其他主路径无交集的最短路径，对于备份链路请求，若所有与其相关的主链路均已完成映射，则利用 BLMA 算法构建备份路径；(3)若映射失败，则回溯至上一节点继续映射。具体流程如表 4 所示。

表 4 RNM-PBT 流程

算法 4 RNM-PBT
输入： $G^s(N^s, L^s, A^n, A^l), G^v(N^v, L^v, C^n, C^l, \text{Loc}), G^b(N^b, L^b, C^b), B(*, *)$
输出： f, f^b
(1) 构造 RSCN 主拓扑 G^v 的宽度优先搜索树 T：以资源约束最大的节点请求为根，且 T 中每层节点请求按照资源约束大小降序排列;
(2) 得到 T 的节点请求映射顺序：$n_v^i, i = 1, 2, \cdots, N$；
(3) $i \leftarrow 1$; backtrack_cnt $\leftarrow 0$;
(4) **while** $i \le N$ **do**
(5) **if** Match (n_v^i, f, f^b)==FAILED **then**
(6) **if** backtrack_cnt > MAX_CNT || i==0 **then return** NULL; **end if**
(7) backtrack_cnt++; $i \leftarrow i-1$; **goto** 4; // 若节点 n_v^i 映射失败，则回溯至节点 n_v^{i-1}
(8) **end if**
(9) $i \leftarrow i+1$
(10) **end while**
(11) **return** f, f^b;

子程序 Match()完成当前节点请求以及与之相邻的主链路和备份链路请求的映射如表 5 所示。

在 RNM-PBT 算法中，Match()子程序的执行次数受 MAX_CNT 限制，而 Match()子程序的最坏情况时间复杂度为 $O(C \cdot M_b \cdot (m \cdot K \cdot M + n^2))$，其中 $C = \max_{n_v \in N^v} |\text{Loc}(n_v)|$，$M_b = |L^b|$。所以 RNM-PBT

表 5 子程序 Match( )的映射

PROCEDURE Match (n_v^i, f, f^b)：
(1) 得到满足资源和位置约束的候选节点集合 S，并对 S 中节点按空闲资源量进行降序排列;
(2) 若 n_v^i 已完成映射，则释放 $f(n_v^i)$ 的资源，并添加 $f(n_v^i)$ 到 S 中，S 只保留位于 $f(n_v^i)$ 之后的候选节点，以避免回溯时搜索到相同的节点;
(3) **if** i==0 || $S==\varnothing$ **then return** FAILED; **end if**
(4) $f_{\text{old}} \leftarrow f$; $f^b_{\text{old}} \leftarrow f^b$; // 保存当前映射方案
(5) **for each** $u \in S$ **do**
(6) $f_N(n_v^i) = u$; //完成主拓扑节点映射
(7) **for each** $l = (n_v^i, n_v^j) \in \{(n_v^i, n_v^j) \mid (n_v^i, n_v^j) \in L_v \cup L_b, j < i, n_v^j \in N_v\}$ **do**
(8) **if** $l \in L_v$ **then** // l 为主链路请求
(9) 得到 u 和 $f_N(n_v^j)$ 间满足 l 的带宽约束，且不经过其他主链路的最短路径 p_{best};
(10) **if** p_{best} 不存在 **then goto** 步骤(26); **end if**
(11) $f_L(l) = p_{\text{best}}$; //完成主拓扑链路映射
(12) **end if**
(13) **if** $l \in L_b$ **then** // l 为备份链路请求
(14) 得到所有满足 $B(l, l_v) > 0$ 的主拓扑链路请求 l_v 的集合 E;
(15) **if** E 中的链路请求均映射成功 **then**
(16) 利用算法 3(BLMA 算法)得到 l 的备份路径 p_b;
(17) **if** p_b 不存在 **then goto** 步骤(26); **end if**
(18) $f^b(l) = p_b$; //完成备份拓扑映射
(19) **else**
(20) 将 l 加入待处理队列 UnBackup 中;
(21) **end if**
(22) **end if**
(23) 检查 UnBackup 中的其他备份链路请求是否满足步骤(15)，若满足，则执行步骤(16)–步骤(18);
(24) **end for**
(25) **return** SUCCESS;
(26) $f \leftarrow f_{\text{old}}$; $f^b \leftarrow f^b_{\text{old}}$; //失败退出前，恢复原有映射方案
(27) **end for**
(28) **return** FAILED;

算法的时间复杂度同样为多项式级别。

5 实验评估

5.1 实验设置

实验主机为 Intel(R) Core(TM) i7 CPU 2.67 GHz, RAM 2G 的 PC，采用 C++语言编程实现相关算法。使用 GT-ITM[12]工具生成具有 100 个节点的底层网络初始拓扑，以及 2000 个 RSCN 请求。底层网络的节点和链路资源取值在[50,100]内均匀分布，节点连接概率为 0.5。RSCN 请求的节点个数 N 在[2,10)内均匀分布，节点和链路资源请求在[0,30]内均匀分布，节点连接概率 p 在实验过程中分别取 0.2, 0.3, 0.5 和 0.7。RSCN 请求的到达过程服从强

度为平均 100 个时间单位到达 20 个请求的泊松过程，每个 RSCN 的生命周期服从期望为 1000 个时间单位的指数分布。单链路故障发生过程服从强度为平均 100 个时间单位发生 1 次的泊松过程，单链路故障恢复时间服从期望为 50 个时间单位的指数分布。此外，设置 MAX_CNT 为 N(即 RSCN 节点请求个数)，设置公式(5)中的 $\omega_1=\omega_2=\omega_3=1$。

5.2 性能分析

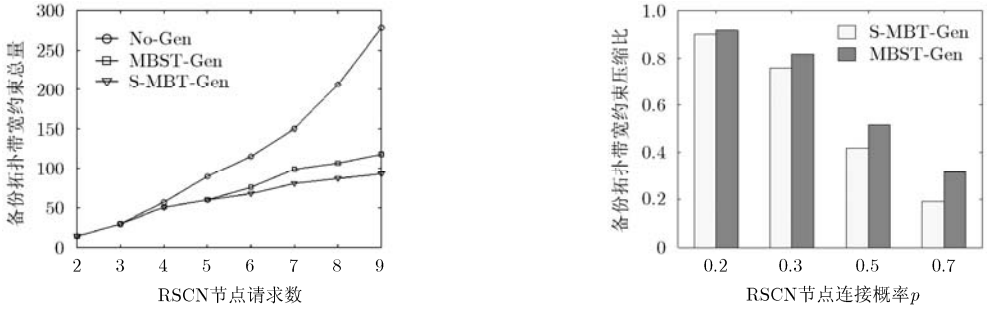
参与性能分析的算法包括：本文提出的 S-RNM-PBT 和 U-RNM-PBT 算法(其拓扑预处理阶段分别采用 S-MBT-Gen 和 MBST-Gen 算法，拓扑映射阶段采用 RNM-PBT 算法)；对所有主链路进行备份，但不共享备份带宽的 SVNE-Greedy 算法[7]；对所有主链路进行备份，并允许共享备份带宽的 PVNM-PRS 算法[8]；不采用备份方式的基本映射算法 No-Backup[13]。

首先评估最小备份拓扑生成算法的备份带宽压缩效果。如图 2(a)所示，S-MBT-Gen 和 MBST-Gen 算法在 RSCN 节点数较大时能够明显减少备份拓扑的带宽约束总量，而不对拓扑进行预处理(No-Gen 曲线)的带宽约束总量增长明显。如图 2(b)所示，通过比较节点数为 8 的 RSCN 备份拓扑带宽约束压缩比(即最小备份拓扑的带宽约束总量与所有主链路均备份的带宽约束总量的比值)可以看出，S-MBT-Gen 算法在节点连接概率 p 较大时优势较为明显，而 MBST-Gen 算法则更适用于节点连接概率 p 较小的情况。

根据公式(5)得到不同算法下的 RSCN 平均资源消耗(如图 3 所示)，此时 $p=0.5$，$N\in[2,10)$。由实验结果可知，PVNM-PRS 算法相比 SVNE-Greedy 算法节省了 20.3%的资源消耗，而 S-RNM-PBT 和 U-RNM-PBT 算法分别在 PVNM-PRS 算法基础上又节省了 18.7%和 12.5%的资源消耗。S-RNM-PBT 和 U-RNM-PBT 算法首先通过生成最小备份拓扑的方式规划了单个 RSCN 内部的备份带宽共享方案，然后在拓扑映射阶段进一步开展 RSCN 间的备份带宽共享，相比于 PVNM-PRS 算法只考虑拓扑映射阶段的备份带宽共享，本文提出的算法保证了带宽共享的稳定性，从而降低了平均资源消耗。由图 2(b)可知，S-RNM-PBT 算法的备份拓扑带宽约束压缩比优于 U-RNM-PBT 算法，所以在平均资源消耗上更具有优势。

不同算法下的 RSCN 请求接受率(映射成功的 RSCN 数与到达的 RSCN 数之间的比值)如图 4 所示，此时 $p=0.5$，$N\in[2,10)$。由实验结果可知，No-Backup 算法请求接受率最高，SVNE-Greedy 算法的请求接受率最低，S-RNM-PBT, U-RNM-PBT 和 PVNM-PRS 算法的请求接受率居于二者之间，并依次降低。由于参与比较的算法均不关注负载均衡，所以影响 RSCN 请求接受率的因素主要为剩余资源总量和映射算法的策略。No-Backup 算法不存在备份带宽的消耗，使得底层网络的剩余资源总量高于其他算法，所以 RSCN 请求接受率最高；而其他 4 种算法的平均资源消耗(如图 3 所示)关系已经明确，且 S-RNM-PBT 和 U-RNM-PBT 算法通过主备拓扑协同映射算法协调利用了底层网络资源，所以形成了图 4 的实验结果。

当 MAX_CNT 为 N 时，对于节点数为 8 的 RSCN，不同算法得到的平均执行时间如图 5 所示。由实验结果可知，SVNE-Greedy 算法的平均执行时间最小，U-RNM-PBT 算法的平均执行时间小于 S-RNM-PBT 算法，而 S-RNM-PBT 和 U-RNM-PBT 算法在 $p\geq 0.3$ 时的平均执行时间小于 PVNM-PRS 算法。原因在于，相比于拓扑预处理阶段的最小备份拓扑生成算法，拓扑映射阶段的备份链路映射算法时间消耗更大。SVNE-Greedy 算法并不采用备份带宽共享方式，所以平均执行时间最小；PVNM-PRS 算法需要对所有主链路进行备份带宽共享映射，而本文提出的算法只需对部分链路进行备份带



(a)备份拓扑带宽约束总量(以p为例)　　(b)备份拓扑带宽约束压缩比(以N=8为例)

图 2 最小备份拓扑生成算法的备份带宽压缩效果

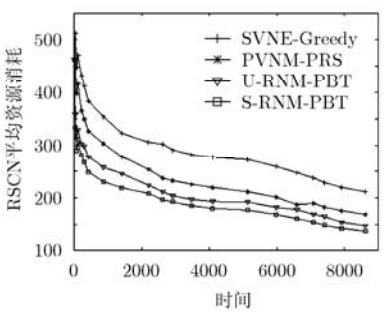


图 3 RSCN 平均资源消耗

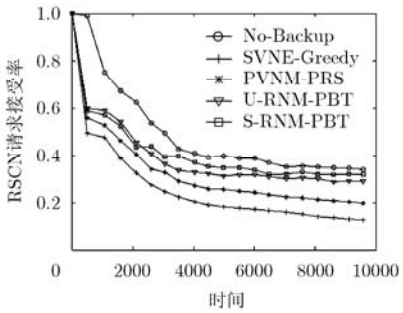


图 4 RSCN 请求接受率

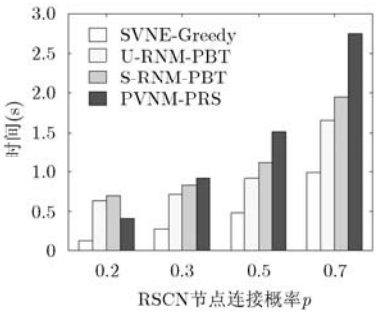


图 5 平均执行时间(以 N=8 为例)

宽共享映射，所以在RSCN拓扑较为稠密时，PVNM-PRS 算法的平均执行时间最大；S-RNM- PBT 和 U-RNM-PBT 算法生成的最小备份拓扑链路数差别不大，但 S-RNM-PBT 算法求解最优解的时间高于 U-RNM-PBT 算法求解近似解的时间，所以 S-RNM-PBT 算法的平均执行时间高于 U-RNM- PBT 算法。

此外，路径备份策略下的 RSCN 通过主备路径切换保证服务的连续性，请求接受率越高，意味着更多的 RSCN 完成了映射和备份，从而在故障时具有更高的全网可靠性。S-RNM-PBT 和 U-RNM-PBT 算法通过降低平均资源消耗(图 3)提高了请求接受率(图 4)；相反，SVNE-Greedy 和 PVNM-PRS 算法对所有主链路进行备份，造成平均资源消耗较大(图 3)，降低了请求接受率(图 4)。因此，S-RNM-PBT 和 U-RNM-PBT 算法在全网可靠性上优于 SVNE-Greedy 和 PVNM-PRS 算法。

综上所述，本文提出的 S-RNM-PBT 和 U-RNM-PBT 算法在平均资源消耗、请求接受率和平均执行时间 3 项指标上具有优势，进一步优化了路径备份策略的资源消耗，并保证了 RSCN 服务提供的可靠性。

6 结束语

本文针对时延敏感类型 RSCN 提出分阶段处理方式的可靠性映射方法。该方法将 RSCN 可靠性映射分为拓扑预处理和拓扑映射两个阶段，利用最小备份拓扑生成算法减小了备份拓扑带宽约束总量，利用主备拓扑协同映射算法协调分配了底层网络资源。通过比较，该方法降低了平均资源消耗，且具有较高的 RSCN 请求接受率和较低的平均执行时间。

参 考 文 献

[1] Turner J and Taylor D. Diversifying the internet[C]. Proceedings of IEEE Conference on Global Telecommunications, St. Louis, USA, 2005: 755–760.

[2] Hu Yu-xiang, Lan Ju-long, and Wu Jiang-xing. Providing personalized converged services based on flexible network reconfiguration[J]. *Science China Information Sciences*, 2011, 54(2): 334–347.

[3] Lee K and Modiano E. Cross-layer survivability in wdm-based networks[C]. Proceedings of IEEE INFOCOM 2009, Rio de Janeiro, Brazil, 2009, 1017–1025.

[4] Krishnaiyan T, Muhamad S, and Larry X. Circuits/cutsets duality and a unified algorithmic framework for survivable logical topology design in ip-over-wdm optical networks[C]. Proceedings of IEEE INFOCOM 2009, Rio de Janeiro, Brazil, 2009: 1026–1034.

[5] Liu Y, Tipper D, and Siripongwutikom P. Approximationg optimal spare capacity allocaton by successive survivable routing[J]. *IEEE/ACM Transactions on Networking*, 2005, 13(1): 198–211.

[6] Banner R and Orda A. Designing low-capacity backup networks for fast restoration[C]. Processings of IEEE INFOCOM'10, San Diego, California, USA, 2010: 1–9.

[7] Rahman M, Issam A, and Boutaba R. Survivable virtual network embedding[C]. Proceedings of the 9th IFIP TC 6 International Conference on Networking, Chennai, India, 2010: 40–52.

[8] Chen Yang, Li Jian-xin, Wo Tian-yu, *et al.*. Resilient virtual network service provision in network virtualization environments[C]. Proceedings of 2010 IEEE 16th International Conference on Parallel and Distributed Systems, Shanghai, China, 2010: 51–58.

[9] Yu Hong-fang, Anand V, Qiao Chun-ming, *et al.*. Enhancing virtual infrastructure to survive facility node failures[C]. Processings of Optical Fiber Communication Conference (OFC), Los Angeles, California, USA, 2011: 1–3.

[10] Yu Hong-fang, Qiao Chun-ming, Anand V, *et al.*. Survivable virtual infrastructure mapping in a federated computing and networking system under single regional failures[C]. Proceedings of IEEE Global Telecommunications Conference (Globecom), Miami, Florida, USA, 2010: 1–6.

[11] Karmarkar N. A new polynomial-time algorithm for linear programming[J]. *Combinatorica*, 1984, 4(4): 373–395.

[12] Zegura E W, Calvert K L, and Bhattacharjee S. How to model an internetwork[C]. Proceedings of IEEE INFOCOM, San Francisco, California, USA, 1996: 594–602.

[13] Yu M, Yi Y, Rexford J, *et al.*. Rethinking virtual network embedding: substrate support for path splitting and migration[J]. *ACM SIGCOMM Computer Communication Review*, 2008, 38(2): 17–29.

王志明： 男，1986 年生，博士生，研究方向为网络虚拟化、新型网络体系结构.

汪斌强： 男，1963 年生，教授，博士生导师，主要研究方向为宽带信息网络、核心路由器关键技术.