一种基于测量构件变迁模型的可重构测量构件一致性检测方法

王 晶 汪斌强 申 涓
(国家数字交换系统工程技术研究中心 郑州 450002)

摘 要 可重构网络测量系统中,工作流测量构件间迁移的过程是否与规约描述一致,是检验测量构件一致性测试的重要内容。建立了一种基于工作流的构件变迁模型 MCTM(Measurement Component Transfer Model),详细阐述了MCTM模型的形式化定义,并基于MCTM模型给出了一种能自动生成遍历所有构件的测试用例生成算法 CTBMCTM。实验结果表明,CTBMCTM算法可以准确定位存在问题的构件,与 T&GS 算法相比,该算法在生成较短测试序列的同时显著缩短了算法的运行时间。

关键词 网络测量,可重构,测量构件,一致性测试
中图法分类号 TP393.1 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2015.9.032

Measurement Component Transfer Model-based Conformance Testing Approach of Reconfigurable Measurement Component

WANG Jing WANG Bin-qiang SHEN Juan
(National Digital Switching System Engineering and Technological R&D Center, Zhengzhou 450002, China)

Abstract In the reconfigurable network measurement system, whether the measurement component state transfer precess conforming to the criterions is the key content of measurement component conformance testing. The paper proposed a MCTM(measurement component transfer model) of multi-measurement components system based on workflow, and then specified some related definitions of MCTM model. A conformance test-case generating algorithm based on MCTM(CTBMCTM) was then proposed. The experimental results show that the approach can discover the abnormity measurement component exactly. At the same time, this approach has shorter testing stubs and running time than T&GS.

Keywords Network measurement, Reconfigurable, Measurement component, Conformance testing

1 引言

近年来,网络技术以及网络应用得到迅猛发展。为承载更加多样化的测量需求,提供更加准确、及时的测量结果,网络测量技术正朝着面向新结构、引入新技术的方向发展。国内外研究团队在网络测量新技术的研究方向上均开展了有益的探索[1-5],尝试从不同角度出发提高网络测量的灵活性和扩展性,使网络具有动态灵活部署并发网络测量任务的能力。在体系结构创新方面,软件定义网络测量 SDM(Software Defined Measurement)[2,3]提出了将测量与控制相分离的分层结构,借鉴软件定义网络 SDN(Software Defined Network)的思想,通过软件编程定义的方式来实现对测量的控制和管理。可重构的网络测量模型是结合可重构柔性网络体系结构提出的一种新型网络测量结构,它通过改变测量构件间的组合连接关系,实现系统测量功能的灵活动态变化,从而提高测量系统的灵活性和高效性。

测量构件是可重构系统中的一类特殊网络构件,它是用于组装网络测量功能模块的具有独立功能、可替换、有明确接口并且可以独立部署的软硬件实体[6]。在可重构的测量系统中,通过测量构件的组合能够实现丰富多样的测量功能,从而能够在一个测量系统中支持多样、并行测量任务的部署。目前关于网络测量构件的研究处于起步阶段,测量构件的管理、组装、安全等问题都有待深入思考。与传统构件相同,测量构件的开发是一个开放的过程,第三方开发的介入将带来构件可靠性降低的隐患。因此,对测量构件进行一致性测试是确保可重构测量系统正常工作的重要途径。当前一些构件测试方法[7-12]使用传统的软件测试方法,通过输入测试序列判断输出是否正确来进行一致性测试。但是在可重构测量系统中,由于测量构件类型和组合方式较多,且当测量构件的组合方式发生变化后,存在输入不变输出变化的情况,或者输入输出不变内部构件状态变化的情况。因此使用传统测试方法对重构测量构件进行一致性测试,将带来测试序列爆炸的问题。

到稿日期:2014-09-28 返修日期:2014-12-29 本文受国家重点基础研究发展计划("973"计划)基金资助项目(2012CB315901,2013CB329104),国家自然科学基金项目(61309019,61372121),国家高技术研究发展计划("863"计划)基金资助项目(2013AA013505)资助。
王 晶(1980-),女,博士生,讲师,主要研究方向为网络测量、可重构网络,E-mail:zhuwangzilz@163.com;汪斌强(1963-),男,博士,教授,博士生导师,主要研究方向为可重构柔性网络;申 涓(1976-),女,博士生,讲师,主要研究方向为可重构网络、网络路由技术。

本文要解决的可重构测量构件一致性测试问题，本质上是验证测量系统中按测量任务需求组合的测量构件是否能够按照规约工作。规约是指为实现某测量功能，测量构件间保持正确稳定的工作流传递关系，即使发生构件的重新组合——重构，构件也能按照新的工作流传递关系工作。可重构测量系统中的工作流在执行时通常涉及多个测量构件，外界输入触发工作流在构件间迁移，并使构件产生相应输出。所以可以通过检验输出序列验证工作流在构件间的迁移过程，从而检测出测量构件的异常情况。此外，如何根据构件输出唯一确定构件也是本文需要解决的关键问题。首先提出了一种可以描述工作流在测量构件间迁移的构件变迁模型MCTM(Measurement Component state Transfer Model)，并对模型进行了形式化定义，在此基础上提出了一种能自动生成遍历所有构件的测试用例生成算法。最后对MCTM模型进行了实例分析，并进行性能的仿真实验。实验结果表明，使用MCTM模型可对可重构测量系统中的测量构件进行一致性测试。

2 可重构测量模块的MCTM模型

2.1 MCTM模型的相关概念和定义

下面给出测量构件变迁模型MCTM的形式化定义。文中如不特别说明，所有的构件均指测量构件。

定义1 测量构件变迁模型MCTM是一个五元组 $M=\{MC,I,O,\delta,\lambda\}$，其中 $MC=\{MC_0,MC_1,MC_2,\cdots,MC_n\}$ 表示测量构件的集合；$I=\{I_0,I_1,I_2,\cdots,I_n\}$ 表示外界输入的集合；$O=\{O_0,O_1,O_2,\cdots,O_n\}$ 表示测量构件输出的集合；$\delta=MC_i\times I\rightarrow MC_j$ 是测量构件间的迁移函数，表示测量构件 MC_i 在外界输入 I 的作用下将工作流迁移给测量构件 MC_j；$\lambda=MC_i\times I\rightarrow O_i$ 是测量构件输出函数，表示构件 MC_i 在外界输入 I 的作用下产生输出 O_i。δ 的单个输入 I_i 使用@符号相连就构成了一个输入序列，同理把输入对应的输出连接起来就构成了一个输出序列。

当测量系统发生重构，改变测量构件间连接关系时，测量构件的输入输出顺序也随之改变，但原可重构测量模块的MCTM模型保持不变。用 R 和 T 分别表示可重构测量系统的测试需求集和测试用例集，假定两个集合都是非空有限集，对于测试需求集的任一子集 $R'(R'\subseteq R)$，在测试用例集都存在对应的子集 $T'(T'\subseteq T)$ 覆盖该测试需求集。设 n 为测试需求集的基数，m 为测试用例集的基数，则 $|R|=n,|T|=m$。

定义2 测试输入模型TIM是一个三元组 $F=\{R,T,\alpha\}$，其中 R 为测试需求集，T 为测试用例集，α 为使测试用例集覆盖测试需求集的输入集合，$\alpha\subseteq I$。

对于可重构测量系统，R 由可重构测量系统的测量构件文档描述抽象得出；T 为覆盖 R 的测试用例集合；α 为 T 的输入集合。

2.2 可重构测量构件的异常测试

可重构测量构件的异常测试是由人或机器在特定的环境下按照特定的程序对被测系统做实验，观察被测系统在实验中表现出的行为。实验可按多种方式进行，这取决于被测试的可重构测量模块的功能、性能、测试环境、测试者或测试机器，以及对测试结果的观察和解释能力等。对可重构测量构件测试时只关心被测构件的外部行为，而不关心它的内部操作。测试的对象被称为被测测量构件(Measurement Component Under Testing，MCUT)，测试的目是在不同的外部输入条件下测试MCUT是否按测量构件规约中描述的工作流进行构件迁移。

测量构件的一致性测试工作分为4步(见图1)：(1)按照构件组合规约、构件的文档描述确定测试需求集；(2)生成并描述测试集和测试用例；(3)按照测试集对被测MCUT进行测试；(4)根据测试系统的测试记录，参照M构件实现一致性描述(Measurement Component Implementation Conformance Statement，CICS)，对MCUT进行评估，给出测试结论。

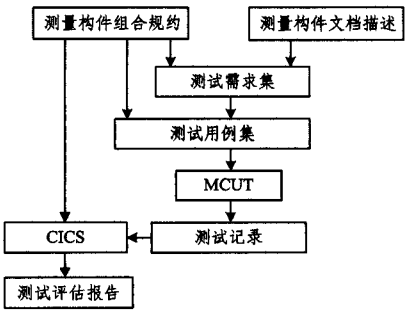

图1 可重构测量构件的一致性测试过程

3 测试用例自动生成算法

对于一个可重构测量模块的MCTM，接下来考虑的问题是如何生成测试集。根据测试需求，可以从MCTM中得到外界输入集合，向被测模块输入这个集合，被测模块产生相应的输出集合，工作流也在构件间发生迁移。每个重构测量模块的MCTM中存在一个处理外界输入的构件和一个向外界输出结果的构件，工作流从输入构件开始，最终迁移到输出构件。可以根据每个可重构测量模块的MCTM，直观地判断模块产生的输出事件是否正确。但是如何判断工作流中途是否迁移到MCTM中描述的应该达到的构件，是我们设计测试序列时需要解决的问题，也就是解决被测构件的可判定性。

3.1 测试序列生成算法问题描述

我们的目的是为被测试的重构测量模块的MCTM寻找一个好的算法，来生成测试用的一个输入输出序列。通过输入输出序列来判断工作流的迁移是否符合MCTM中的描述，并检验是否得到匹配的输出，从而判断MCTM中的构件是否正常。对于MCTM中工作流在输入 I_i 的作用下从构件 MC_i 迁移到 MC_j 并且输出 O_i 的过程，可表示为 $\delta=MC_i\times I_i\rightarrow MC_j,\lambda=MC_i\times I_i\rightarrow O_i$，称 MC_i 为迁移的始构件，MC_j 为迁移的末构件。测试时要解决的问题就是如何判断工作流从 MC_i 迁移到了 MC_j，而不是其他构件。为了使工作流到达 MC_i，可以使用输入引导序列 I_m 使工作流从初始构件 MC_0 到达 C_i。对于 C_j，本文使用UIO(Uunique Input Output)序列[12]进行状态验证。UIO序列是指MCTM中每一个构件都存在一组唯一的输入输出序列，当对这个构件施加这组输入序列时，得到的输出序列与其他构件在这组输入序列影响下得到的输出序列不同，这样就能唯一确定这个构件。所以对构件 MC_i 加入输入 I_i 后继续施加UIO序列的输入序列 I_{UIO}，若构件组合系统在输出 O_i 之后能继续输出UIO序列的输出序列的 O_{UIO}，则表明工作流确实迁移到构件 MC_j。在对可重构测量模块测试时，要对其MCTM中的每一条迁移边进行测试，每条边的测试序列都由引导序列 I_m 和UIO输入序列 I_{UIO} 构成。这样最终构成的测试序列过长，并且存在冗

余测试序列。注意到如果一个迁移边测试的末构件正好是下一个迁移边测试的起始构件，就可以采取某种算法使构件间的工作流迁移连接起来，从而缩短测试序列。一个质量较高的测试集应该用尽可能少的测试序列覆盖尽可能多的测试路径，所以本文采用寻找欧拉路径的方法来减少测试序列的长度。

首先将 MCTM 模型转换成有向图。将可重构测量模块的 MCTM 看作是一个有向图 M，顶点的集合 $V=\{V_0,V_1,V_2,\cdots,V_n\}$ 表示 M 的测量构件集合 MC，每条边 $e_{ij}=(V_i,V_j;i_i/o_i)$，$e_{ij}\in E$，表示工作流从构件 MC_i 到 MC_j 的迁移，输入是 i_i，输出是 o_i。边集合 $E=\{(V_j,V_k;i_j/o_j)\mid V_j,V_k\in V\}$ 表示 M 的所有工作流的迁移。I_{UIO} 表示构件 MC_i 的 UIO 序列的输入序列，O_{UIO} 表示 UIO 序列的输出序列。构件 MC_k 表示构件 C_j 输入 I_{UIO} 后迁移到 MC_k 的构件。这样可以在图中添加一条生成边，始状态为 C_i，末状态为 C_k，输入是 $i_i@I_{UIO}$，输出是 $o_i@O_{UIO}$，这个迁移表示为 $e_{ik}=(V_i,V_k;i_i@I_{UIO}/o_i@O_{UIO})$，当遍历这个生成边的时候也就相应地测试了 C_i 到 C_j 的工作流迁移。对 M 中的每条边都添加这样一条生成边，M 变为 M^*（M 的生成子图），M^* 中既有原来的迁移边，又有新的生成边，且生成边可能是一个自环。

通过 M^* 中边上的输入和输出序列，可得到每个迁移边的测试序列。为了缩小测试集的规模，想尽可能地把测试序列从头到尾串起来，即寻找图中的欧拉路径。实际上仅靠 M^* 中的生成边不一定能构成欧拉图，因此在构造欧拉图时还需要选择原始边。但为了得到较短的测试序列，在构造欧拉图时要尽可能少地使用原始边。

3.2 基于最大流标记的 MCTM 欧拉图构造算法

根据图论中的概念，当一个有向图每个顶点的出度与入度相等并且是弱连通时，这个图就是欧拉图，且图中存在欧拉路径。为了得到较短的测量序列，需要在 M^* 的基础上构造欧拉图。依据欧拉图的定义，具体的构造方法为：若仅靠 M^* 中的生成边就能保证每个顶点的出度入度相等，则可把原来旧的迁移边删掉；若仅靠生成边不能使顶点的出度入度相等，则要在原有的迁移边中选择尽可能少的边来充当冗余边，使每个顶点的出度入度相等。下文将给出冗余边的选择算法——最大流标记算法。首先定义容量网络。

定义 3 设有向连通图 $G=(V,E)$，E 的每条边上有非负数 C_{ij}，C_{ij} 为边容量，找到一个入度为 0 的点 V_s 称为发点 U_s（源点），一个出度为 0 的点 V_t 称为收点 U_t（目的点），其余点为中间点，这样的网络 G 称为容量网络，记作 $G=(V,E,C)$。

定义 4 设任一容量网络 G 中的边 (V_i,V_j) 有流量 f_{ij}，称集合 $f=\{f_{ij}\mid(V_i,V_j)\in E\}$ 为 G 中的流。f 是可行流要同时满足下列条件：

(1)容量限制条件。对 G 中每条边，有 $0\leqslant f_{ij}\leqslant C_{ij}$。

(2)平衡条件。对中间点 V_i，有 $\sum_j f_{ij}=\sum_k f_{ki}$（即中间点 V_i 的流的输入量与输出量相等）；对收、发点 U_t、U_s，有 $\sum_j f_{ij}=\sum_k f_{ki}=W$（即从 U_s 点发出的流总量等于 U_t 点输入量），W 为网络流的总流量。

对于一个容量网络，可行流总是存在的。例如 $f=\Phi$，表示一个流量为 0 的可行流。所谓最大流问题，就是在容量网络中寻找流量最大的可行流。对于一个流 f，当 $f_{ij}=C_{ij}$ 时称流 f 对边 (V_i,V_j) 是饱和的，否则称 f 对边 (V_i,V_j) 不饱和。

定义 5 容量网络 G 中，若 μ 为网络中从 V_s 到 V_t 的一条链路，给定 μ 的方向为从 V_s 到 V_t，G 中的边凡与 μ 同向则称为前向边，凡与 μ 反向则称为反向边，其集合分别表示为 μ^+ 和 μ^-。f 是一个可行流，如果满足：

$$\forall f_{ij}\in f,\begin{cases}0\geqslant f_{ij}\geqslant -C_{ij}, & (V_i,V_j)\in\mu^-\\ 0\leqslant f_{ij}\leqslant C_{ij}, & (V_i,V_j)\in\mu^+\end{cases}$$

则称 μ 为从 V_s 到 V_t 的可增广链。

推论 可行流 f 是最大流的充要条件是不存在从 V_s 到 V_t 的可增广链。

证明：(充分性)因为若可行流 f 是最大流，那么在容量网络 G 中将不存在比 f 更大的可行流，即在 V_s 和 V_t 之间不存在一条链路，其流量满足 $0\geqslant f_{ij}\geqslant -C_{ij}$ 或 $0\leqslant f_{ij}\leqslant C_{ij}$，所以不存在从 V_s 到 V_t 的可增广链；(必要性)因为在容量网络 G 中不存在从 V_s 到 V_t 的可增广链，即在 V_s 和 V_t 之间不存在一条链路，其流量满足 $0\geqslant f_{ij}\geqslant -C_{ij}$ 或 $0\leqslant f_{ij}\leqslant C_{ij}$，那么将不存在一条可行流 f'，其流量满足 $f'\geqslant f$，所以 f 是最大流。推论得证。

可增广链的物理含义是：对从 V_s 到 V_t 的流，如果存在可增广链，则可以通过加入增广链的方法提高链路流量，调整后的流如果在各点仍满足平衡条件及容量限制条件，则仍为可行流。

这样就得到了寻求最大流的方法：从一个可行流开始，寻求关于这个可行流的可增广链，若存在，则可以经过调整得到一个流量比原来大的新可行流，重复这个过程，直到不存在关于该流的可增广链时就得到了最大流。

寻找容量网络中最大流的标记算法如算法 1 所示。

算法 1 最大流的标记算法

Begin:

1)找到图中的一条可行流 f。

2)发送点为 V_s，并给发送点标号$(\Delta,+\infty)$。

3)选择一个已标号的顶点 V_i，对于 V_i 的所有未标号的邻接点 V_j 按如下规则处理：

- if 边$(V_i,V_j)\in E$ 且(V_i,V_j)与(V_s,V_t)同向，$C_{ij}>f_{ij}>0$，则令$\delta=C_{ij}-f_{ij}$，并标号$(+V_i,\delta)$。
- if 边$(V_i,V_j)\in E$ 且(V_i,V_j)与(V_s,V_t)反向，$0>f_{ij}>-C_{ij}$，则令$\delta=C_{ij}-f_{ji}$，并标号$(-V_i,\delta)$。

4)重复 3)，直到收点 V_t 被标号或不再有顶点可标号时为止。

5)If V_t 得到标号，说明存在一条可增广链，按下述规则调整可行流的大小，去掉节点标号，回到 2)，继续为调整后的可行流寻找增广链路：

$$f'_{ij}=\begin{cases}f_{ij}+\delta, & \text{若}(V_i,V_j)\text{是可增广链的前向边}\\ f_{ij}-\delta, & \text{若}(V_i,V_j)\text{是可增广链的后向边}\\ f_{ij}, & \text{若}(V_i,V_j)\text{不在增广链上}\end{cases}$$

Else V_t 未被标号，标号过程已无法进行，说明 f 已是最大流，goto End。

End

现在，给出基于最大流标记的 MCTM 欧拉图构造算法。根据上述描述，在构造欧拉图之前，首先对 M^* 做如下变化，构造出 MCTM 的流量模型图 M^{**}。具体过程为：

(1)M^{**} 的顶点与 M^* 相同，用 M^* 的每个顶点所连接的生成边数目计算 M^{**} 顶点的出度和入度，入度记为负值，出度记为正值，出度与入度之和为通过该顶点的流量。

(2)M^{**} 的边为 M^* 中的原始边，边上的流量为 0。

(3)在 M^{**} 中增加一个源点 s 和一个目的点 t，增加 s 和 t 与其他顶点之间的边，若顶点流量为负，则添加一条从 s 到顶点的连边；若顶点流量为正，则添加一条从顶点到 t 的连边。

新增边的流量取该顶点流量值的绝对值。

(4)M^{**} 中间点之间边上的容量为∞,中间点与 s 和 t 之间的边的容量与该边的流量相等。

通过上述 4 个步骤,可得到 MCTM 的流量模型图 M^{**}。然后通过最大流标号算法,在 M^{**} 中找最大可行流,那么这个最大可行流所流经的原始边就是在构造欧拉图时所选择的冗余边。具体而言,基于最大流标记的 MCTM 欧拉图构造算法如算法 2 所示。

算法 2 基于最大流标记的 MCTM 欧拉图构造算法

In:可重构测量模块的 MCTM

Out:MCTM 的欧拉图

Begin:

1)由 MCTM 构造有向图 M;

2)生成每个测量构件的 UIO 序列;

3)使用 UIO 序列构造 M 的生成图 M*;

4)将生成图 M* 变为流量模型图 M**;

5)利用算法 1 最大流标记算法,找到最大流和冗余边;

6)用 M* 中的生成边和找到的冗余边组成欧拉图 G。

End

3.3 测试用例生成算法

通过算法 2,可得到 MCTM 的欧拉图 G。G 中每条欧拉路径上的输入序列就组成了该重构测量模块 MCTM 的构件迁移的测试输入序列,欧拉路径上的输出序列则为测试输出序列。根据测量结果与输出序列的比对结果,参照每个构件的 UIO 序列,可对测试过程中发生错误的构件进行定位。

综上,基于测量构件变迁的测量构件一致性测试用例生成算法 CTBMCTM 如算法 3 所示。

算法 3 测试用例生成算法 CTBMCTM

In:可重构测量模块的 MCTM

Out:测试输入序列和输出序列

Begin:

1)按算法 2 生成 MCTM 的欧拉图 G;

2)在 G 中找到一条欧拉路径;

3)欧拉路径上对应的输入序列即为测试 MCTM 使用的测试输入序列;

4)欧拉路径上对应的输出序列即为测试 MCTM 使用的测试输出序列。

End

4 重构测量模块实例分析

本节以可重构测量中的流量抽样功能单元为例,如图 2 所示,通过建立工作流 1 中报文哈希构件、分类器构件和抽样计数器构件 3 个构件的 MCTM 模型,并利用第 3 节介绍的算法,生成基于 MCTM 工作流迁移的测试序列。

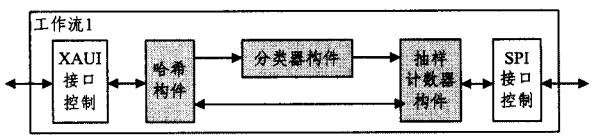

图 2 工作流 1 中测量构件分布图

(1)MCTM 模型的建立。可重构的流量抽样功能单元由报文哈希构件 C_1、分类器构件 C_2、抽样计数器构件 C_0 组合生成。报文哈希构件的主要功能是通过 HASH 函数减少测量报文数量;分类器构件对过滤的报文进行选择;抽样计数器构件对报文进行抽样计数。根据 3 个构件的迁移关系,建立其 MCTM 模型 $M=\{C,I,O,\delta,\lambda\}$(见图 3),其中 $\{C_0,C_1,C_2\}=C$,$\{a,d,e,T\}=I$,$\{b,e,f,g,n\}=O$。MCTM 中输入和输出序列的具体含义为:a,数据报文;b,丢弃报文;c,配置报文;d,哈希不匹配报文;e,分类报文;f,抽样报文;g,配置报文反馈;n,空操作;T,哈希匹配报文。

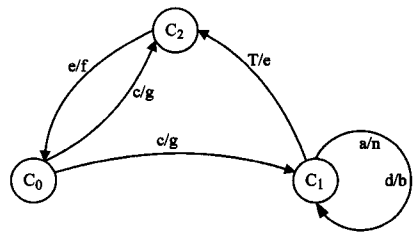

图 3 测量构件的 MCTM 模型

(2)找出 M 中每个测量构件的 UIO 序列。如表 1 所列,每个构件的 UIO 可以找到多组,我们选择一组较短的 UIO 序列。

表 1 3 个构件的 UIO 序列

构件	末构件	UIO 序列		UIO 输入序列		UIO 输出序列	
C_0	C_2	c/g	T/e	c	T	g	e
C_1	C_2	d/b	T/e	d	T	b	e
C_2	C_1	e/f	c/g	e	c	f	g

(3)利用构件的 UIO,基于 M 构造带生成边的图 M^*,如图 4 所示。图中的虚线边是生成边,每个生成边的输入输出序列都是由原始输入输出序列和构件的 UIO 序列组成,例如原 M 中的边 $e_{02}(C_0,C_2:c/g)$,因为 C_2 在 UIO 序列影响下的末状态为 C_2,所以在 M^* 中增加了一条新边 $e_{02}(C_0,C_2:c/g,d/b,T/e)$。

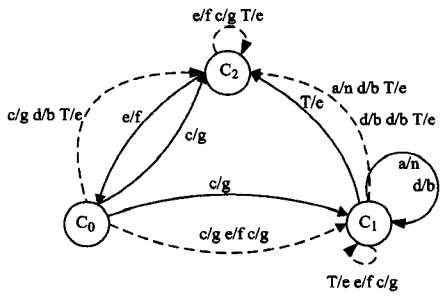

图 4 M 的生成图 M^*

(4)将生成图 M^* 变为流量模型图 M^{**},找到最大流和冗余边。每个节点的出度与入度的值实际就是每个节点生成边的个数。通过定义节点的流量来描述节点出入度之间的差异。图 4 中 C_0 的出度是 2,入度是 0;C_1 的出度是 3,入度是 2;C_2 的出度是 1,入度是 4。转换为流量模型图 M^{**} 后如图 5 所示,S 为源点,t 为目的点。图 5 中的虚线边为新增边,实线边为 M 中的原始边。采用算法 1 介绍的最大流标识算法,在 M 的原始边中选择图 6 中的粗黑线作为冗余边,与 M^* 中的生成边一起组成欧拉图。

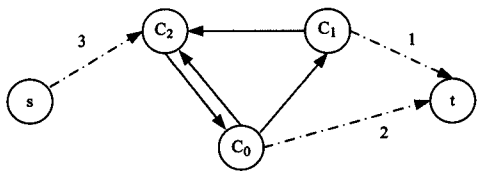

图 5 流量模型图 M^{**}

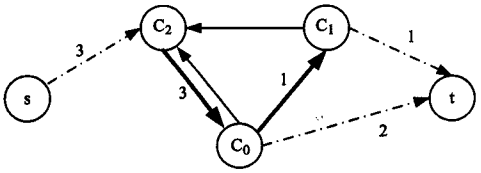

图 6　M 的生成图 M^*

(5)将图 6 所示的流量模型图还原成 MCTM，如图 7 所示。

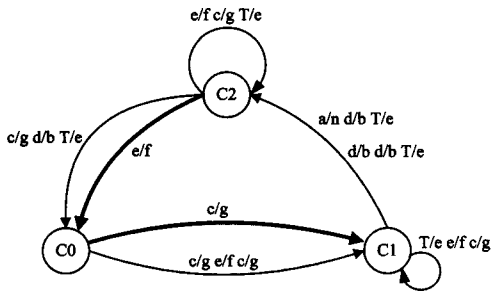

图 7　流量模型图 M^{**}

(6)最终在 MCTM 欧拉图中找到覆盖所有边的欧拉路径。图 7 中的一条欧拉路径为 $(C_0,C_1:c/g,e/f,c/g)(C_2,C_2:e/f,c/g,T/e)(C_1,C_2:d/b,d/b,T/e)(C_1,C_1:T/e,e/f,c/g)(C_0,C_1:c/g)(C_2,C_0:e/f)(C_1,C_2:a/n,d/b,T/e)(C_2,C_0:c/g,d/b,T/e)$。欧拉路径上边的输入序列就是测试的输入序列，输出序列就是对应的输出序列。图 7 得到的输入序列是：$e\ c\ T\ e\ c\ e\ c\ T\ e\ c\ a\ d\ T\ c\ d\ T\ c\ d\ d\ T$；对应的输出序列是：$f\ g\ e\ f\ g\ f\ g\ e\ f\ g\ n\ b\ e\ g\ b\ e\ g\ b\ b\ e$。使用输入序列测试可重构的流量抽样模块时，若输出序列在带有下划线的位置出现异常，便可根据表 1 所列的 UIO 输出序列定位是哪个构件发生了异常。

5　算法性能分析

为了测试算法的性能，本文将算法与文献[13]提出的基于图论的最少测试用例生成算法做比较，文献[13]中的 T&GS 算法能够生成较短的测试序列。

首先分析算法的时间复杂度。在 CTBMCTM 算法中寻找每个测量构件的 UIO 序列和寻找生成图的最大流是最费时间的步骤，在有 n 个构件的 MCTM 中，寻找 UIO 序列时，需要将每个构件的输出序列与其它 $n-1$ 个构件的输出序列相比较，计算复杂度约为 $O(n^2)$；寻找生成图的最大流时，根据标号算法，在有 n 个顶点(构件)的生成图中，每个顶点的标号最终会影响到其它 $n-1$ 个顶点，大约花费 $O(n^2)$ 的时间寻找最大流，所以 CTBMCTM 算法的时间复杂度约为 $O(n^2)$。根据文献[11]的分析，T&GS 算法每次迭代中的时间复杂度为 $O(n^2)$，最坏情况下需要进行 n 次迭代，所以 T&GS 算法的时间复杂度为 $O(n^3)$。

接着采用仿真实验的方法比较两者在不同拓扑及不同规模 MCTM 下的性能。仿真使用 IBM X3200M2，CPU 3.0 GHz，RAM 2GB，硬盘 250GB，两种算法均用 C 语言实现。为比较两个算法的运行时间，参考文献[12]中的模型，假设了一种含有 38 个构件的重构测量模块，模块的构件连接关系如图 8 所示。

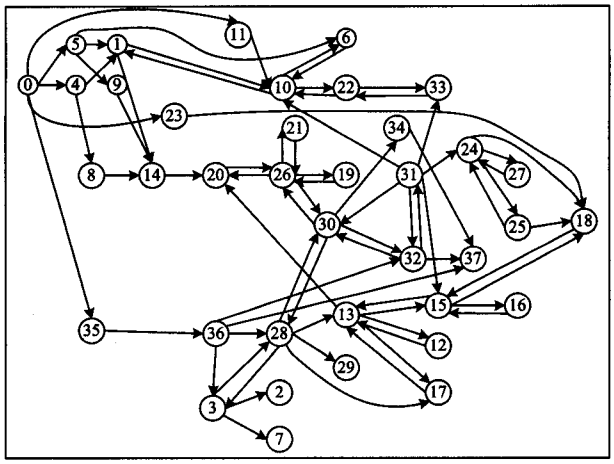

图 8　仿真用测量构件连接图

测试当构件数目不变时，构件间的拓扑关系变化(即迁移边的数目变化)对算法生成的测试序列和运行时间的影响。图 8 的 38 个构件(77 条边)构成 MCTM1，之后以 MCTM1 为原型，随机增加迁移边构成额外的 MCTM。其中 MCTM10 是全连接模型，代表了最坏的测试情况。每种模型运行 10 次，取测试结果的平均值得到表 2 所列结果。图 9 是对表 2 结果的图形化，以对比 T&GS 与 CTBMCTM 算法的性能。

表 2　构件数量相同拓扑关系不同时的测试结果

Input	Edge Numbers	Test sequence		Time(ms)	
		T&GS	CTBMCTM	T&GS	CTBMCTM
MCTM1	77	9.0	9.0	1.4	1.2
MCTM2	228	21.9	21.9	11.0	1.6
MCTM3	380	27	26.8	19.1	2.2
MCTM4	532	30.7	30.9	26.3	2.6
MCTM5	684	32.3	32.3	34.1	3.0
MCTM6	836	33.7	33.4	40.5	3.2
MCTM7	988	33.9	34.0	48.0	3.6
MCTM8	1140	34.9	34.5	54.7	3.8
MCTM9	1292	36.0	35	63.0	4.0
MCTM10	1406	37	36	70.5	4.1

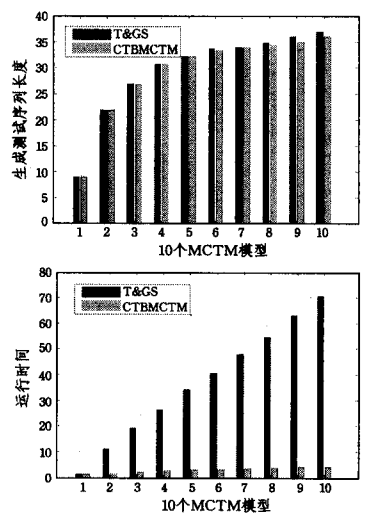

图 9　不同拓扑 MCTM 时不同算法生成的平均测试序列数、平均运行时间对比

从表 2 和图 9 的结果可以看出，在算法效果上，CTBMCTM 算法得到的测试序列长度与 T&GS 算法不相上下，10 个模型中有 5 个测试序列优于对方，3 个持平，2 个较差。T&GS 算法的目标是生成尽可能短的测试序列，仿真结果证

明所提算法可以得到较短的测试输入序列。在算法性能上，算法运行时间明显优于对方，CTBMCTM 比 T&GS 算法节省了大约 84.6%的运行时间；并且随着迁移边的增加，算法的运行时间增长比较平稳，说明 CTBMCTM 算法受迁移边增加的影响不大，比较适用于迁移边较多的 MCTM 模型。其原因是在迁移边较多的情况下，MCTM 只需使用较少的冗余边便可构成欧拉图并找到欧拉路径。

为了进一步验证 CTBMCTM 的性能和扩展性，本文通过改变模型的规模（构件数量）来测试算法的生成序列长度和运行时间，比较对象仍是 T&GS 算法。本文使模型规模从 50 到 1000 变化，随机生成构件间拓扑。考虑到构件规模的约束，本文使用了最为简单的 UIO 序列，每种情况运行 10 次，取测试结果的平均值得到表 3 所列结果。图 10 是根据表 3 的测试结构绘制的在不同规模下 T&GS 与 CTBMCTM 算法的性能对比图。

表 3　模型规模变化时的测试结果

Component	Test sequence		Time(ms)	
	T&GS	CTBMCTM	T&GS	CTBMCTM
50	26	25.8	32.0	6.7
100	50.9	49.2	79.7	8.4
200	97.2	95.9	455.5	30.1
300	139.4	137.5	1333.7	65.6
400	187.4	184.5	2960.7	115.7
500	235.3	236.5	5589.5	190.3
600	275.3	277.4	9526.7	256.9
700	322.7	322.0	14752.6	342.7
800	269.9	369.0	21905.6	449.6
900	416.3	414.0	30907.9	575.0
1000	419.7	416.5	42295.1	918.4

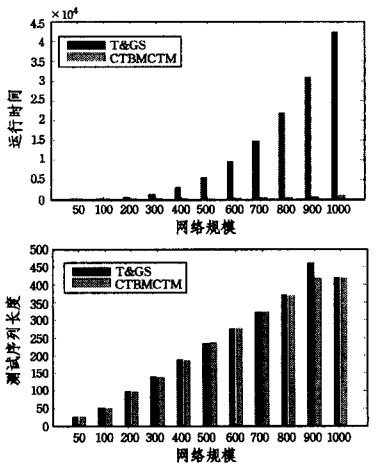

图 10　不同规模 MCTM 时不同算法生成的平均测试序列数、平均运行时间对比

从测试结果可以看出，在算法效果上，本文算法生成的测试序列仅仅比 T&GS 生成的测试序列短约 0.84%，但是在运行时间上本文算法比 T&GS 缩短了将近 96.01%。表 3 中可以看到，构件规模为 100、300、500、1000 的 4 个结果，在构件规模分别扩大了 3 倍、5 倍、10 倍，它们对应的 CTBMCTM 算法时间分别是 8.4、65.6、190.3、918.4，分别增长了 7.8 倍、22.6 倍、109.3 倍，大约是 3^2、5^2、10^2，证明构件规模的增长趋势与时间复杂度的增长趋势符合上文分析所得出的 CTBMCTM 算法时间复杂度约为 $O(n^2)$ 的结论。

结束语　测量构件的灵活组合构成可重构的测量系统，可以为网络提供更为灵活、多样的测量服务，但同时由于测量构件的开发是一个开放的过程，从而为整个测量系统留下安全隐患。在研究测量构件的功能、组合关系等问题的同时，必须对测量构件的一致性测试问题进行深入探讨。本文提出了一种重构测量构件的变迁模型 MCTM，该模型可以描述工作流在构件间的迁移过程。本文对模型进行了形式化定义，并基于 MCTM 提出了一种能自动生成遍历所有构件迁移边的测试用例生成算法 CTBMCTM，最后对算法进行了仿真实验。结果表明，使用 CTBMCTM 算法不仅能够对测量构件进行一致性测试，并且可以准确定位存在问题的构件。同时，对生成的测试序列长度和算法运行时间这两项性能指标与 T&GS 算法进行比较，结果表明 CTBMCTM 算法产生的测试序列长度基本与 T&GS 算法持平，但在测试用例生成时间上要明显优于 T&GS 算法。未来，在测量构件一致性测试研究方面，还将关注 UIO 序列的自动生成、重构测量系统安全性评估等方面的工作。

参 考 文 献

[1] Yuan Li-hua, Chuah Ch-N, Mohapatra P. ProgME: Towards programmable network measurement[J]. IEEE/ACM Transactions on Networking, 2011, 19(1): 115-128

[2] Masoud M, Minlan Y, Ramesh G. Resource/Accuracy tradeoffs in Software-defined measurement [C] // HotSDN (2013). Hongkong, China, August 2013

[3] Minlan Y, Jose L, Rui M. Software Defined Traffic Measurement with OpenSketch[C]//NSDI'2013. Chicago, USA, 2013

[4] Zhang Xiao-dan, Li Jun. Network measurement and analysis architecture of cloud service [J]. Application Research of Computer, 2012, 29(2): 725-729

[5] Cao Zheng, He Jian-bing. Virtualization based network measurement platform [J]. Journal on Communication, 2013, 34(Z2): 84-89

[6] Liu Qiang, Wang Bing-qiang, Xu Ke. Construction and Reconfiguration Scheme of the Hierarchical Reconfiguration Network Based on the Components [J]. Chinese Journal of Computer, 2010, 33(9): 1557-1568

[7] Mao Cheng-ying, Lu Yan-sheng. Research progress in testing techniques of component based software[J]. Journal of Computer Research and Development, 2006, 43(8): 1375-1382

[8] Gallagher L, Offutt J. Automatically testing interacting software components[C]// Proceedings of the 2006 International Workshop on Automation of Software Test. Shanghai, China, May, 2006

[9] Gallagher L, Offutt J. Test sequence generation for integration testing of component software[J]. Computer Journal, 2009, 52(5): 514-529

[10] Gallagher L, Offutt J, Cincotta A. Integration testing of object-oriented components using finite state machines [J]. Software Testing Verification and Reliability, 2006, 16(4): 215-266

[11] Ma Y, Lu J, Zhao R. A test path generation approach for component testing based on UML state diagram[C]// SEA(2006). Dallas, TX, USA, 2006: 13-15

[12] Le Traon. Efficient Object-Oriented Integration Testing and Regression Testing[J]. IEEE Transaction on Reliability, 2000, 49(1): 12-25

[13] Ural H, Wu Xiao-lin, Zhang Fan. On minimizing the lengths of checking sequences[J]. IEEE Transaction on Computers, 1997, 46(1): 93-99