计算机网络基础研究论文集

李华天 刘积仁 编

东北工学院出版社

计算机网络基础研究论文集

李华天　刘积仁　　编

东北工学院出版社
中国·沈阳

内容简介

本论文集收集了“计算机网络基础研究”联合课题组第二次学术讨论年会上交流的论文。论文内容主要是关于计算机网络技术方面的基础研究，包括理论方面的论文二篇、网络体系结构方面的论文二篇、协议形式化描述技术与协议自动化生成工具和环境方面的论文五篇、网络一致性测试方面论文三篇、网络管理与控制方面的论文四篇、分组话音通信方面论文三篇、网络的设计与实现以及其它方面论文六篇共二十五篇另外还刊载了三篇摘要。该联合研究课题是国家自然科学基金委员会信息科学部资助的重点项目之一，由六所院校的计算机网络研究子课题小组联合组成。

计算机网络基础研究论文集

李华天　刘积仁　编

东北工学院出版社出版　　东北工学院出版社发行

（沈阳市·南湖）　　东北工学院印刷厂印刷

开本：787×1092　1/16　印张：11　字数：260　千字

1990 年 4 月第 1 版　1990 年 4 月第 1 次印刷

印数：　1～200

责任编辑：王金邦　　责任校对：孟莉

封面设计：于忠民

ISBN 7-81006-238-7/TP. 11　　定价：10.5元

国家自然科学基金委员会信息科学部资助
“计算机网络基础研究”联合研究课题组协调组成员

组长：李华天教授（东北工学院）

成员：（按姓氏笔划为顺序）

史美林副教授　（清华大学）

朱世伟副教授　（华中理工大学）

胡正家教授　（西安交通大学）

胡道元教授　（清华大学）

顾冠群教授　（东南大学）

谢希仁教授　（南京通信工程学院）

秘书：刘积仁教授　（东北工学院）

前　言

此论文集收纳了“计算机网络基础研究”联合课题组第二次年会上交流的学术论文。论文内容反映了参加该项目的各研究单位当前的工作和阶段成果。这里，有的论文因已被国内其它刊物所接受，故只刊登了文章的摘要。论文集涉及的面较广，粗略地可分成下列几类。

第一类是理论性的文章，都是理论联系实际，具有指导性意义的论文。通信协议中的“死锁”、“活锁”是最困扰设计人员的疏忽错误中的一种。《通信协议中的活锁及其检测》一文从概念开始用转移系统模型得到“活锁”的计算解，并讨论了检测“活锁”的通用双可达性分析技术，给出了检测“活锁”的算法。《令牌型局域网的最佳令牌轮转周期》一文，从理论上分析得到了使分组迟延最小的令牌最佳周期。

第二类是与计算机网络体系结构有关的论文。国际标准化组织（ISO）提出的开放系统互连七层模型（OSI）和各层次中的标准网络服务功能与协议，是协调实现未来全球范围内计算机网络互连的模式，并为大家提供了共同的概念和术语。这个模型已得到国际上广泛的承认和支持，除了这个网络体系结构之外，其它著名的公司和机构也各有自己的体系结构。其中影响较大的有美国国防部（DOD）的国防数据网 DDN，该网已有多年的运行经验，并有众多的用户。

前几年美国国防部发布了一套关于异种机联网通信协议结构模型或称为 DPA 模型。事实上到 1986 年末，以 DDN 为核心的 DARPA 网际网已发展到超过 150 个网络的网际网络，在美国及国际上很有影响。DPA 模型强调分级（Hierarchy）而不采用分层办法，强调网际互连和无连接服务，此外在管理功能手段上也和 ISO/OSI 模型有差异。近年来倾向于 DPA 模型的学者对 ISO/OSI 模型中其些具体问题提出非议，双方尚有不少争论。但事实上 ISO/OSI 模型是在 DPA 模型基础上建立起来的，已吸取了 DPA 模型中许多成功的设计和实施经验。虽然它由于采用分层结构而增加了一定的开销，但因它采用了系统工程方法，用结构级、服务级及协议级三个级别的抽象方法以及自顶向下功能划分等方法，优点还是很明显的。OSI 模型为研究开发和改进标准化工作明确了工作领域并为容纳已有的有关协议标准保持它们的一致性提供了协调的余地。因此 ISO/OSI 的体系结构成为大家所接受的国际标准这已经是大势所趋。我国计算机网络的发展道路，势必也将遵循这个方向。

论文《从 TCP/IP 过渡到 ISO/OSI 传送服务的研究》是在清华大学进行的一项有意义的工作，提出了 TCP/IP 到 OSI 传送服务的 TP0 转换策略。TCP/IP 是 DPA 结构中的传送层协议与网际协议，它提供了无连接的数据报服务。我国有些单位已进口了这种网络产品，采用这种转换策略就可以在现有网络基础上为 OSI 高层协议的开发研究及早提供一个工作环境。《DECnet 局域网与 IBM4381 终端网互连的研究与实现》一文介绍了西安交通大学用一台 IBM-PC 机作为 Gateway 把属于性能体系结构不同的网络互连通信的经验，用于实现网际文件的传

输、共享和文件目录的列表查询等功能。此外该文结合国情，利用现有的资源，采用高层互连的方法，在建立网际网方面所取得的经验与技术，也是有参考价值的。

第三类是关于形式化描述技术与协议生成自动化工具环境方面的论文，这是当前协议工程方法学最活跃的研究领域之一。计算机网络通信协议由协议的语义、语法和通信规则三部分组成。语义是通信双方所表达的内容，语法规定这些内容表示的形式，而通信规则是协议通信实体交换信息的应答变化关系。计算机网络通信协议在执行时具有非常复杂的动态行为，因而网络协议的研究、设计、实施是一项非常艰巨复杂的任务。协议工程学就是利用软件工程的理论，结合系统工程方法在分布式计算机系统通信协议中的具体应用。目前协议工程方法学中，形式化描述技术已经成为重要的基础。协议研究开发活动的生命周期，从系统分析、精确的形式描述、正确性验证、性能预测、协议代码生成到协议一致性测试等各阶段都已逐步形成一整套的方法和软件支撑工具。现在国际标准化组织已推出建议性的几种标准的形式化描述语言。如基于扩展有限状态机模型的 ISO 的 Estelle 语言，基于时序模型的 Lotos 语言及 CCITT 的 SDL 语言。对 Estelle 语言编译器的研究，国内已有几所院校进行了工作，《Estelle 编译器研究报告》一文就反映了清华大学在这方面工作的进展情况，其中代码生成选用 Pascal 作为目标语言。《形式描述技术：直观 Lotos 的提出和研究》一文，在标准 Lotos 的基础上，对该语言的语法表示方法进行了改进。

《CCITT LAPD 协议的形式化描述及其自动化生成》一文，是用 Estelle 语言把综合业务数字网（ISDN）I 系列用户—网络接口数据链路协议标准作为描述和实现的对象，介绍了从形式化描述开始到可执行代码自动生成为止，实现一个协议层开发的全部过程的工作。《协议形式语言翻译器的快速模型设计》这篇论文提出解决这样一个问题：因软件工具本身的发展已满足不了软件产品发展的需要，而必须加快对生产工具（或工作母机如翻译器之类）的设计和生产。文中提出面向客体的系统设计和软件工厂自动化方法，自动生产出“样机”并通过对“样机”的直接验证、抽象执行或实际运行来调整、精炼，进而改善原来的设计。因为这是软件工厂的自动化工具，所以对提高生产率和改善产品质量具有重要的价值。文中介绍了东北工学院采用了这种方法对 ASN·1 Macros 翻译器的设计和运行的经验证明它是十分有效的。它也是该院研究发展的网络协议支撑环境中的组成部分之一。

《ISO 工业制造信息通信协议标准（MMS）的服务定义，协议规范及实现策略》一文是介绍工业制造协议 MAP 应用层中最具有特征性的一个协议。MMS 可以将工厂中处于“孤岛”中的不同厂家生产制造的设备通过 MAP 互连起来实现综合自动化系统 CIMS。该文还探讨了根据东北工学院协议开发的支撑环境实现 MMS 的策略步骤。

第四类是关于协议一致性测试方面的论文。协议一致性测试在协议工程中占有非常重要的地位。妨碍连网的计算机的相互可操作性的一个原因是由协议实施时的疏忽和错误而造成实施与原协议规范定义不一致。另外一个更重要的原因是由于协议的复杂性，一个协议往往有几个子集而这些子集通常只有一个共同的交集，这些子集并不向上兼容，而且同一协议的同一子集也往往存在不同的任选参数，所以要做到联网的计算机之间相互通信，并完成各项服务的功能，决不是一项轻而易举的事。事实上经常发生这样的例子，几个厂家的协议，大家都声称按 ISO 标准协议实施的，但却不能很好地互连通信。协议一致性测试的目的是要看它在功能上和通信行为上是否与协议规范说明相一致。协议一致性测试的开销往往很大，有些情况下个别用户和厂家负担不了这种开销而需要由政府机构来负担。1987 年 ISO 推出一个

ISO 一致性测试方法学和工具系统的文本草案，以及推荐一个 TTCN 测试语言。TTCN 是一种描述抽象测试集的标记法。最初推出时是非形式化的表格形式，后来 ISO 又推出了 BNF 草案，这样就为机器的自动处理提供了可能。但 TTCN 的语义（及部分语法）还没有定义而正处于探索阶段。清华大学的《OSI 标准化测试系统的实现和研究》一文较全面地论述了标准化测试系统的设计、抽象测试方法和选择，并给出了一个实例。《TTCN 编译器及其与网络测试环境的接口》一文介绍了东北工学院在这方面的的研制工作：用编译器对描述抽象测试集的语言进行翻译即可自动地生成可执行测试集再以某种自动或半自动的方式产生测试选择和驱动部件就可以得到某一协议的测试系统。这样的测试环境可以大大简化一致性测试的过程。《MHS 系统的一致性测试》一文论述了对 MHS 测试系统中，所用的一致性测试方法和测试系统的结构。MHS 是 OSI 中一个提供信息存贮转发传送服务应用系统。

第五类是关于计算机网络管理控制方面的文章。计算机网络管理控制在计算机网络发展的早期是一个被忽略的领域。随着网络系统的日益庞大和复杂，网络资源的合理使用、日常维护、计费以及安全保密性等管理控制工作日益显得重要。ISO 在建立 OSI 网络体系结构的初期，对这方面工作也没有予以足够的重视。因此，直到目前为止有关网络管理的标准尚在演变发展之中。论文《OSI 管理的研究和实现》，《OSI 网络的管理控制》，《江苏通信子网网控中心的设计》和《江苏 JSnet 中管理系统的研究》，介绍了 OSI 网络管理和控制方面的功能和内容、管理模型和体系结构，并阐述了东南大学在研制江苏网 JSnet 管理系统所遇到的问题和经验，这些工作在国内是属于前驱性的。

第六类是有关分组话音通信问题的论文。我们知道分组交换网最早仅限于传送计算机的数据，而由于这种数据信息的突发性，应用持续连接线路来传送信息是很不经济的。对话音、图像信号，多少年来一直沿用固定比特率的线路连接。近年来由于 ISDN 的发展，数字编码的话音信号在分组交换网中传输的研究日益受到重视，因为 ISDN 是未来网络发展的方向而且向宽带网的方向发展。在分组交换网中话音通信有关的延迟、排队、包丢失和重发等问题都是很重要的基础性研究工作。论文集中有三篇论文和二篇摘要都是南京通信工程学院的研究成果：《分组话音通信对汉语话音质量的影响》，《分组话音通信对传输性能测试》，《NICENET—II 局域网分组话音同步的设计与实现》，《话音数据综合实验局域网 NICENET》和《分组话音通信中最佳分组长度的确定》。在这些论文中讨论叙述了分组通信对汉语话音质量的影响；分组话音通信中传输迟延及丢失等规律的实际测试统计分布规律以及在综合话音和数据业务局域网中分组话音通信的同步方法的分析研究和设计实现工作，这些在国内都属于前沿工作。

第七类关于网络设计实施及其它方面的论文共六篇。《JUNET 网络系统的设计及其它应用层软件的实现》一文介绍了西安交通大学利用现有公用电话网作通信介质的远程计算机网，文中阐述了系统的设计思想和系统构成等，重点介绍了应用层软件的设计与实现，该网已达到实用化程度。《综合业务数字 PABX 的设计与实现》一文介绍了在南京通信工程学院利用新一代的 PABX 来实现一个局域网，以便在局部范围内实现数据和话音的综合交换。关于这方面工作，虽然在国外已相当普遍，但在我国尽管目前引进的新型 PABX 不少，可是在发挥其话音和数据综合交换能力方面，大多数还没有发挥其潜在效能。因此，这方面工作的开展对促进我国 OA 技术的发展，对将来 ISDN 的发展将起到促进作用。《分组交换网通信子网内部协议的选择与设计》和《分组交换网 JSS 的端一端控制协议的设计与实现》这两篇文章是分别介绍在东南大学所做的江苏网通信子网（TSS）网内协议的选择设计方案和该子网端一端控制协议

TML 的设计与实现。此工作已取得了阶段成果，下一步即将进入运行联调的阶段。

文集的最后的两篇论文是华中理工大学所做的工作。《异种机联网应用数据抽象表示的研究》这篇论文对异种机联网应用层的数据表达作了有益的探讨，另一篇论文《异构环境中数据共享的研究》从另外一个角度探讨了在异构计算机网络环境中用远程调用的设施，建立顾客—服务者模型来达到数据共享。

“计算机网络基础研究”联合课题组的参加成员都是国内较早从事计算机网络研究的学术梯队，共有六所院校参加。建立联合研究课题组的目的是希望通过横向的联合，减少研究上的重复性工作，并通过学术讨论成果共享和分工协作，促使我国计算机网络的基础研究有较快的进展，以便尽快地赶上先进国家的水平。联合课题组从一开始就得到国家自然科学基金委员会信息科学部的支持、指导和资助。两年多来，所以课题组能取得很多可喜的成果，是和国家自然科学基金委的指导与资助分不开的，本人就此机会谨代表联合课题组的参加人员表示感谢。本论文集的出版得到了东北工学院出版社和印刷厂的大力支持，特别是王金邦同志在协助编审方面做了很多工作，另外还得到了孟莉同志、王向荣和文辉同志的帮助，在此一并表示感谢。由于时间仓促，限于编者水平，书中难免有疏漏错误不妥之处，请读者批评指正。

李华天

1990 年 3 月

目　　录

通信协议中的活锁及其检测

赵锦蓉

清华大学

摘要 本文研究通信协议中的活锁及其检测技术。我们采用转移系统（transition system）作为通信协议的形式模型。我们给出活锁的定义，将其与文献中所用到的各种活锁概念进行比较，并给出活锁的计算解。然后我们研究双可达性分析技术，并给出在两个限状态进程通信系统中检测活锁的算法。双可达性分析技术可以缓解直接计算的“状态空间爆炸”问题。

1. 引言

在对通信协议的分析中，一个重要任务是检测和消除设计的错误。通信协议设计错误之一就是所谓死锁 [1]。有大量文献对死锁现象进行了深入的研究。对死锁的的概念大家也比较熟悉。而与此相对，对活锁现象和它的检测技术的研究却不多。实际上活锁同死锁一样对于一个通信协议也是重大错误，但是由于系统不进入停顿，检测更为困难。

与死锁不同，活锁的概念也比较复杂，有各种不同的理解。Sherman 和 Rudin [7] 把活锁描述为通信系统的全局状态图中的无效循环且无出口。Kwong [5] 在研究平行程序时提出两类活锁的概念。同时 Sifakis [8] 出给出了平行系统中活锁和弱活锁的定义。但 Kwong 的两类活锁与 Sifakis 的两类活锁并不一致。Gouda 等人在 [3] 中讨论了通信有限状态机系统中活锁检测问题的可判定性，他们定义的活锁是一个无进展循环，类似于 [8] 中的弱活锁。

我们将给出通信协议的活锁定义，并比较各种活锁的概念。我们的概念推广了 [7]，但与 [8] 中的活锁概念不同。Sherman 和 Rudin [7] 注意到了饥饿和活锁的关系，我们的定义明确了这种联系。

为了检测活锁我们将扩充 [9] 中的方法，引入双可达性变换规则 TR。这样所牵涉的全局状态的数目可大大地减少。Rudin 和 West [6] 在讨论死锁和未描述的接收的可达性分析时曾经注意到这个事实。

在本文的第 2 节我们研究通信协议的转移系统模型，并给出活锁，饥饿和死锁的概念。第 3 节讨论一个例子。第 4 节讨论活锁的双可达性分析。第 5 节给出算法。第 6 节是结论。

2. 通信协议中的活锁

2.1 转移系统

一个转移系统是一个三元组 $S=(Q, T, q_0)$，其中 Q 是状态集合；$T=\{t_i\}_{i=1}^{m}$ 是转移集合，即 Q 上的二元关系；$q_0\in Q$ 是初始状态。

一个转移系统可以用一个带标号的有向图来表示。图的结点对应 Q 中的状态，图的边是 T 中的转移。如果对 $t_i\in T$ 有 $t_i(q, q')$，则图中有一条边从结点 q 指向结点 q'，该边以 t_i 为标号。我们称 q' 为 q 的直接后继，q 为 q' 直接先行。无出边的结点称为终止结点。

对于状态 q 若存在一状态 $q'\in Q$ 有 $t_i(q, q')$，也即 q 与 q' 之间有二元关系 t_i，就说可以从状态 q 通过转移 t_i 到达 q'。也说转移 $t_i\in T$ 在状态 q 可以执行并到 q'。$t_i(q, q')$ 也可以表示为 $q\text{-}t_i\rightarrow q'$。

这个概念可以扩充到 T^*，T^* 表示 T 中符号的有限及无限连接的集合。令 $\tau=t_{i_1}\cdots t_{i_k}\in T^*$，$q\text{-}\tau\rightarrow q'$ 表示存在一有限状态序列 $q_1=q, q_2, \cdots, q_{k+1}=q'$，对 $1\leqslant j\leqslant k$ 有 $q_j\text{-}t_{i_j}\rightarrow q_{j+1}$。若 $q\text{-}\tau\rightarrow q'$ 就说可以从状态 q 通过转移 τ 到 q'，q' 是 q 的后继，从图上看从 q 到 q' 有一条路径。此外 $q\text{-}\tau\rightarrow$ 表示存在 $q'\in Q$，$q\text{-}\tau\rightarrow q'$。若 τ 是 T 上的一个无限序列，我们用 $q\text{-}\tau\rightarrow$ 表示对 τ 的每个前缀有限子序列 τ_1，有 $q\text{-}\tau_1\rightarrow$。关于转移系统的概念可以参考 [4]。

给定一转移系统 $S=(Q, T, q_0)$，我们可引入 pre 及 $\widetilde{pre}$ 的概念。令 $P\subseteq Q$，则 $pre(P)$ 是 P 的直接先行集合，$\widetilde{pre}(P)$ 是那些直接后继都在 P 的点集和终止点集。pre 与 $\widetilde{pre}$ 是不同的，$pre(P)$ 中的结点 q 必是 P 中某结点 q' 的直接先行，但 q 的直接后继不一定都在 P 中。而 $\widetilde{pre}(P)$ 中的结点 q 的直接后继必定都在 P 中。

2.2 通信协议的转移系统模型

一个通信系统可以用几个相互通信的有限状态进程来描述，它们通过先进先出的信息队列通信。有限状态进程也可以表示为带标号的有向图。它的结点对应进程的状态，它的边表示发送和接收信息。标号 $-m$ 表示发送信息，标号 $+m$ 表示接收信息。

我们在 [9] 中引入了一组方程，称为进程方程，来表示有限状态进程的转移函数，并详细讨论了利用进程方程的可达性分析。若有一条边从进程 A 的状态 A_i 指向状态 A_k，且带标号 a_{ik}，则我们在 A_i 有方程

$$A_i=\sum_{A_k\in succ(A_i)} a_{ik}\ A_k$$

其中 $succ(A_i)$ 表示 A_i 的所有直接后继的集合。若 $succ(A_i)=\varnothing$，则方程成为 $A_i=E$。我们把方程简写为

$$A_i=\sum_{k} a_{ik}\ A_k \qquad i=0, 1, 2, \cdots, n。$$

本文考虑一个包含两个通信有限状态进程 A 和 B 的系统 &。

设 A_i 和 B_j 分别为进程 A 和 B 的状态，a 和 b 分别为在 A 到 B 的通道和 B 到 A 的通道中的信息队列。我们用 $\langle a, A_i, b, B_j\rangle$ 来表示一全局状态。$\langle\lambda, A_0, \lambda, B_0\rangle$ 是初始全局状态，其中 λ 表示空队列。下面我们用 a_{ik}，b_{jl} 表示单一信息，有时我们用 $\pm c_i$，$\pm d_i$ 代替 a_{ik}，b_{jl}。

现在我们假定在 A_i 和 B_j 的进程方程分别为

$$A_i=\sum_{k} a_{ik}\ A_k \qquad 和 \qquad B_j=\sum_{l} b_{jl}\ B_l$$

设 α 表示任意的全局状态，ε 表示一特殊状态，它在 [9] 中被称为非发生状态。我们引入了下列全局状态的变换规则。

(1) 可达性变换规则

(*TR*1) $\langle a, A_i, b, B_j\rangle \rightarrow \langle a, \sum_k a_{ik}A_k, b, B_j\rangle$

(*TR*2) $\langle a, A_i, b, B_j\rangle \rightarrow \langle a, A_i, b, \sum_l b_{jl}B_l\rangle$

(2) 基本代数变换规则

(*TA*1) $\langle a, \sum_k a_{ik}A_k, b, B_j\rangle \rightarrow \sum_k \langle aa_{ik}, A_k, b, B_j\rangle$

(*TA*2) $\langle a, A_i, b, \sum_l b_{jl}B_l\rangle \rightarrow \sum_l \langle a, A_i, bb_{jl}, B_l\rangle$

(*TA*3) $\alpha+\varepsilon\rightarrow\alpha$

(*TA*4) $\varepsilon+\alpha\rightarrow\alpha$

(3) 归约变换规则

(*TD*1) $\langle a\ (+d_1), A_i, (-d_1)\ b, B_j\rangle \rightarrow \langle a, A_i, b, B_j\rangle$

(*TD*2) $\langle\ (-c_1)\ a, A_i, b\ (+c_1), B_j\rangle \rightarrow \langle a, A_i, b, B_j\rangle$

(*TD*3) $\langle a\ (+d_1), A_i, b, B_j\rangle \rightarrow \varepsilon$ *where* $b\neq (-d_1)\ b'$

(*TD*4) $\langle a, A_i, b\ (+c_1), B_j\rangle \rightarrow \varepsilon$ *where* $a\neq (-c_1)\ a'$

(*TD*5) $\langle a, E, b, B_j\rangle \rightarrow \varepsilon$

(*TD*6) $\langle a, A_i, b, E\rangle \rightarrow \varepsilon$

定义对全局状态应用一次可达性变换规则，即 *TR*1 或 *TR*2，接着是所有可能的基本代数变换规则和归约变换规则，称为对全局状态的一个变换步。

现在我们就可以用一个转换系统 $S=(Q, T, q_0)$ 作为系统 & 的描述模型。其中初始状态 $q_0=\langle\lambda, A_0, \lambda, B_0\rangle$。$Q$ 是从 q_0 经过有限次变换步得到的全局状态集合。至于 T 有两种情况，首先，若在变换步 $q=\langle a, A_i, b, B_j\rangle \rightarrow \langle aa_{ik}, A_k, b, B_j\rangle = q'$ 中应用了 (*TR*1) 和 (*TA*1) 且 a_{ik} 为负，则我们有 $q\text{-}a_{ik}\rightarrow q'$。另外若在变换步中 $q=\langle a, A_i, b, B_j\rangle \rightarrow \langle aa_{ik}, A_k, b, B_j\rangle$ 且 $a_{ik}=+d_1$，进一步 $\langle aa_{ik}, A_k, b, B_j\rangle = \langle a\ (+d_1), A_k, (-d_1)\ b', B_j\rangle \rightarrow \langle a, A_k, b', B_j\rangle = q'$ 应用了 (*TD*1)，则我们有 $q\text{-}+d_1\rightarrow q'$。类似地，若 b_{jl} 为负有 $q\text{-}b_{jl}\rightarrow q'$ 若 $b_{jl}=+c_1$，且 $a=(-c_1)\ a'$ 有 $q\text{-}c_1\rightarrow q'$。所以 T 包括两部分 $T(A)$ 和 $T(B)$。转移 $a_{ik}\in T(A)$ 只是进程 A 执行的动作，转移 $b_{jl}\in T(B)$ 只是进程 B 执行的动作，所以它们也可以表示为 $a_{ik}(A)$ 和 $b_{jl}(B)$，若我们并不关心转移是哪个进程的动作，也可以就用 a_{ik} 和 b_{jl} 表示。

2.3 死锁，饥饿和活锁

给定一个转移系统 $S=(Q, T, q_0)$，其中 $T=\{t_i\}_{i=1}^{m}$。我们用 W_i 表示 S 的有向图中有 t_i 出边的结点集合。

定义 (1) S 中的死锁状态是有向图中的终止结点。因此令 D 表示所有死锁状态的集合，则 $D=\bigcap_{i=1}^{m}\overline{W_i}$，其中 $\overline{W_i}$ 表示 W_i 的补集。

(2) S 中对转移 t_i 的饥饿是一个结点集 H，满足下列条件：

① 每个结点无 t_i 出边。

② 每个结点的直接后继都包括在 H 中。H 中的结点称作对 t_i 的饥饿状态。

(3) S 中对转移 t_i 的活锁 L 是有向图中一条或多条非终止路径，而这些路径上的结点构成对 t_i 的饥饿。

我们可以类似地定义对若干个转移的饥饿和活锁。

为叙述方便，下面我们将不区分路径与路径所包含的全体结点的集合。因此 L 既代表一组非终止路径，也代表这组路径上所有结点的集合。

按定义，一个对 t_i 的饥饿状态其所有后继也是对 t_i 的饥饿状态，所以转移 t_i 在进入这些状态后再也不能执行了。对 t_i 的活锁包含对 t_i 的某些饥饿状态和它们的所有后继且不含死锁。注意在文献中死锁一般指带空信息队列的终止状态，所以这里给出的定义略有不同，这里的死锁是指对所有 t_i（即对 T）的饥饿状态。

命题 (1) H 是 S 中对转移 t_i 的饥饿的充要条件是：

$$H\subseteq\overline{W_i},$$

$$H\subseteq\widetilde{pre}(H)。$$

(2) L 是 S 中对转移 t_i 的活锁的充要条件是：

$$L\subseteq\overline{W_i},$$

$$L\subseteq pre\ (L)$$

$$L\subseteq\widetilde{pre}(L)。$$

证明略。

定理 1 给定一转移系统 $S=(Q, T, q_0)$。

(1) 对转移 t_i 的最大饥饿集合是

$$H=\bigcap_j \widetilde{(pre)}^j\ (\overline{W_i}) =\overline{\bigcup_j\ (pre)^j\ (W_i)}。$$

(2) 对转移 t_i 的最大活锁集合是

$$L=\bigcap_j\ (I\cap pre\cap\widetilde{pre})^j\ (\overline{W_i}) =\overline{\bigcup_j\ (I\cup pre\cup\widetilde{pre})^j\ (W_i)},$$

其中 $I\ (P)=P$。

证明略。

现在我们讨论通信协议的转移系统。我们已注意到，转移集合 $T=T\ (A)\cup T\ (B)$。对 $T\ (A)$（或 $T\ (B)$）的饥饿对应于进程 A（或 B）永久锁住。在 [9] 中被称作接收锁住状态。对 $T\ (A)$ 或 $T\ (B)$ 的活锁对应某些只一个进程执行另一进程锁住的非终止路径。我们下面将不讨论这类活锁，因为接收锁住状态相对容易检测（可参考 [9]）。而排除了接收锁住状态就没有这类活锁了。

3. 一个例子

图 1 给出一个通信系统，在此系统中，每个进程只有一个缓冲区来存贮一个发送包或一个接收包。若缓冲区分配给发送包，则在此包的正确认被收到后释放缓冲区。若缓冲区被分配给接收包，则在正确认被发送后释放缓冲区。图上 p，a，n 分别代表数据包，正确认，和负

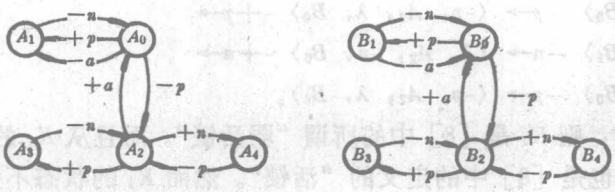

图 1

确认。

图 2 是全局状态图，也称可达图，从图可见无死锁状态。

现在我们考虑转移 $t_1=+a$ 并计算对转移 t_1 的最大饥饿和活锁。我们有：

$$W_1=\{\langle\lambda, A_2, -a, B_0\rangle, \langle\lambda, A_2, (-a)(-p), B_2\rangle, \langle -a, A_0, \lambda, B_2\rangle, \langle (-a)(-p), A_2, \lambda, B_2\rangle\},$$

$$pre\,[T](W_1)=\{\langle\lambda, A_2, \lambda, B_1\rangle, \langle\lambda, A_2, -a, B_0\rangle, \langle\lambda, A_1, \lambda, B_2\rangle, \langle -a, A_0, \lambda, B_2\rangle\},$$

$$pre^2\,[T](W_1)=\{\langle -p, A_2, \lambda, B_0\rangle, \langle\lambda, A_2, \lambda, B_1\rangle, \langle\lambda, A_0, -p, B_2\rangle, \langle\lambda, A_1, \lambda, B_2\rangle\},$$

……

$$\bigcup_j (pre\,[T])^j(W_1)$$

$$=W_1 \cup pre\,[T](W_1) \cup pre^2\,[T](W_1) \cup \cdots\cdots$$

$$=\{\langle\lambda, A_2, -a, B_0\rangle, \langle\lambda, A_2, (-a)(-p), B_2\rangle, \langle -a, A_0, \lambda, B_2\rangle, \langle (-a)(-p), A_2, \lambda, B_2\rangle, \langle\lambda, A_2, \lambda, B_1\rangle, \langle\lambda, A_1, \lambda, B_2\rangle, \langle -p, A_2, \lambda, B_0\rangle, \langle\lambda, A_0, -p, B_2\rangle, \langle\lambda, A_4, \lambda, B_0\rangle, \langle\lambda, A_0, \lambda, B_4\rangle, \langle\lambda, A_2, -n, B_0\rangle, \langle -n, A_0, \lambda, B_2\rangle, \langle\lambda, A_0, \lambda, B_0\rangle\}。$$

所以对转移 t_1 的最大饥饿是

$$H=\overline{\bigcup_j (pre\,[T])^j(W_1)}$$

$$=\{\langle -p, A_2, -p, B_2\rangle, \langle\lambda, A_2, -p, B_3\rangle, \langle -p, A_3, \lambda, B_2\rangle, \langle\lambda, A_2, (-p)(-n), B_2\rangle, \langle (-p)(-n), A_2, \lambda, B_2\rangle, \langle\lambda, A_3, \lambda, B_3\rangle, \langle\lambda, A_3, -n, B_2\rangle, \langle -n, A_2, \lambda, B_3\rangle, \langle -n, A_2, -n, B_2\rangle, \langle\lambda, A_2, -n, B_4\rangle, \langle -n, A_4, \lambda, B_2\rangle, \langle\lambda, A_2, (-n)(-p), B_2\rangle, \langle (-n)(-p), A_2, \lambda, B_2\rangle, \langle\lambda, A_4, \lambda, B_4\rangle, \langle\lambda, A_4, -p, B_2\rangle, \langle -p, A_2, \lambda, B_4\rangle\}。$$

因为 $\widetilde{pre}(W_1)=\varnothing$，对 t_1 的最大活锁 L 与 H 一样。L 是非终止路程，它只是包含对转移 t_1 的饥饿。L 中的状态反映了这样的事实，两个进程同时发送数据包，它们各自的唯一缓冲区被发送包占有，因此回答负确认，拒绝接收到来的数据包，并且重发各自的未被确认的数据包。从 L 中的任何状态都不能执行 t_1，且跳不出 L。若我们考虑只有 t_1 是有进展转移，则进入 L 后系统就再也不能有进展了。

现在我们用此例子来解释不同的活锁概念。考虑路径 K_1：

$\langle \lambda, A_0, \lambda, B_0\rangle \xrightarrow{p} \langle -p, A_2, \lambda, B_0\rangle \xrightarrow{+p}$

$\langle \lambda, A_2, \lambda, B_1\rangle \xrightarrow{n} \langle \lambda, A_2, -n, B_0\rangle \xrightarrow{+n}$

$\langle \lambda, A_4, \lambda, B_0\rangle \xrightarrow{p} \langle -p, A_2, \lambda, B_0\rangle$。

转移 t_1 不包含在路径中，则 K_1 是 [8] 中的所谓“弱活锁”。而且从 K_1 的任何状态都无 t_1 出边，即 $K_1 \subseteq \overline{W_1}$，则 K_1 也是 [8] 中的定义的“活锁”。然而 K_1 的状态不是对 t_1 的饥饿状态。K_1 的状态 $\langle \lambda, A_2, \lambda, B_1\rangle$ 有后继 $\langle \lambda, A_2, -a, B_0\rangle$，此状态有出边 t_1，即 t_1 是可执行的，因此 K_1 不是本文中定义的活锁。在我们的定义中对转移 t_1 的活锁不仅是非终止路径，而且也是对 t_1 的饥饿，它包括路径结点的所有后继。这保证了从这种路径没有出口。如果我们考虑转移 t_1 为进展转移，其它的都是非进展转移，则我们可以把路径 $K \subseteq \overline{W_1}$ 称作非进展路径。从这个例子中我们可以得到非进展、非终止路径，如 K_1 在通信协议中可以不带来问题。而且如果我们采用重传作为对付差错的策略，则非进展、非终止路径甚至不可避免。只有当系统永久陷入非进展，非终止的路径才是协议的问题。

Sherman 和 *Rudin* 在 [1] 中描述活锁为全局状态图中的一个无效的循环且无出口。循环必是一条非终止路径，所以他们的概念包含在我们的概念中。

那末怎样计算对转移 t_i 的活锁呢？正如上面的例子给出的那样，我们可以用可达性分析来计算它，即计算 W_i，$pre\ (W_i)$，…。然而“状态空间爆炸”是一严重问题 [1]。在下一节我们将给出可达性分析一个简约方法。

4. 对活锁的双可达性分析方法

现在我们给出检测活锁的双可达性分析方法。双可达性的想法在 [6] 中已提到。我们考虑包含两个通信有限状态进程 A 和 B 的系统。假定在 A_i 和 B_j 的进程方程分别为：

$$A_i = \sum_k a_{ik} A_k \qquad 和 \qquad B_j = \sum_l b_{jl} B_l$$

我们引入下面的双可达性变换规则来代替规则 *TR*1 和 *TR*2。

双可达性变换规则：

(*TR*) $\langle a, A_i, b, B_j\rangle \rightarrow \langle a, (\sum_k a_{ik} A_k), b, (\sum_l b_{jl} B_l)\rangle$

定义. 对全局状态的一个双变换步包括应用一次双可达性变换规则 *TR*，接着是所有可能的基本代数变换和归约变换。

现在我们可以引入双可达性的概念。一个全局状态可以从 $\langle \lambda, A_0, \lambda, B_0\rangle$ 经过有限次双变换步得到，则称为双可达的。所有的双可达状态组成双可达图。

定理 2. $\langle a, A_i, b, B_j\rangle$ 是一可达全局状态且 $|a| = |b|$，则它是双可达的，其中 $|a|$ 表示队列 a 的长度。

证明略。

定理 3. 假定作为两个通信有限状态进程的系统模型的转移系统 S 是无接收锁住状态的，则存在对 t_i 活锁 L 的充要条件是对 t_i 的活锁是双可达的。

证明略。

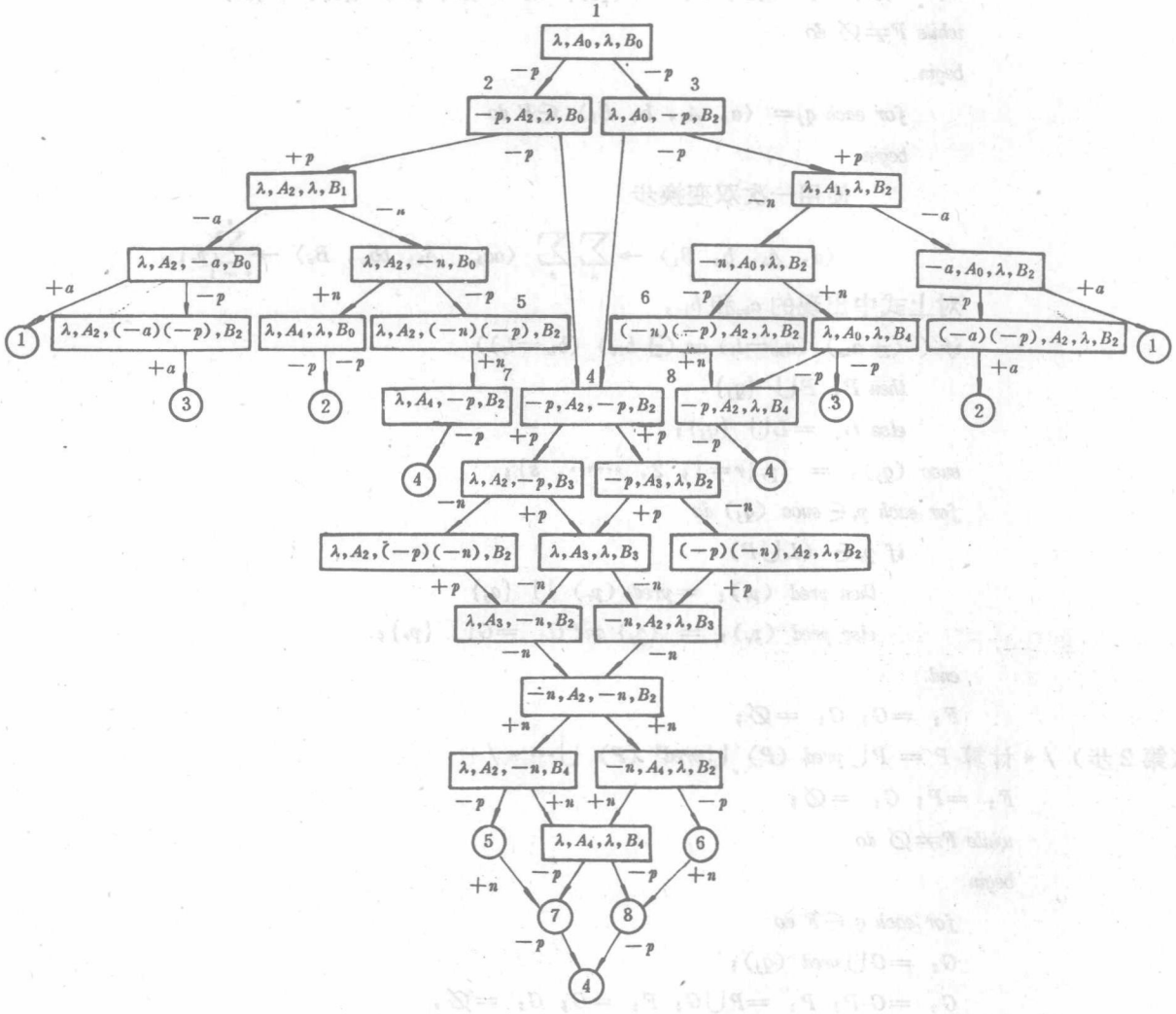

图 2

5. 检测活锁的算法

本节我们给出一个双可达性分析算法，用来检测对转移 t_i 的活锁。算法只适用于两个进程的协议类。我们假定通道之一是有界的，且系统无接收锁住状态。

所使用的变量有：

P ：双可达的且对 t_i 非饥饿的状态集合，

L ：双可达的且对 t_i 饥饿的状态集合，

$pred\ (q)$ ：双可达图中状态 q 的直接先行集合，

$succ\ (q)$ ：双可达图中状态 q 的直接后继集合，

F，G ：状态集合，

p，q ：状态。