挖掘概念类之间的相互作用关系*)

张师超　陈　峰　尤晓芳

（悉尼科技大学信息技术学院　澳大利亚）（广西师范大学计算机系　桂林 541004）

摘　要　一条关联规则是有趣的如果它满足最小支持度和可信度的限制。这导致大量平凡的规则产生。设计一个算法挖掘这样的有趣规则，它的前件和后件分别属于不同的概念类，称这种规则为类间桥。类间桥在行销中的交叉销售，生物工程中的嫁接，化学中的合成等应用中有重要的应用价值。

关键词　数据挖掘，关联规则挖掘，聚类，类间桥

Mining Interactions within Conceptual Classes

ZHANG Shi-Chao　CHEN Feng　YOU Xiao-Fang

(Faculty of Information Technology, University of Technology Sydney, Australia)

(Department of Computer Science, Guangxi Normal University, Guilin 541004)

Abstract　An association rule is interesting if it satisfies two constraints: minimum support and minimum confidence. This can generate a great many trivial association rules. This paper designs an algorithm for mining a kind of association rules that the antecedent and action of such a rule belong to different conceptual classes, referred to class bridges. Class bridges are useful in applications, such as cross-sale in marketing, engraftation in bioengineering, and chemical synthesis.

Keywords　Data mining, Association rule, Clustering, Class bridge

1　引言

数据挖掘(Data Mining，又译作数据采掘)[1]是一个从大型数据库中提取人们感兴趣的知识的过程。这些感兴趣的知识是潜在的、事先不被人们所认知的、隐含的、有用的信息。所提取出的知识用概念(Concepts)、规则(Rules)、规律(Regularities)、模式(Patterns)等形式表示。

关联规则(Association Rules)[2]是数据挖掘中一个易于理解、研究最为广泛的课题。关联规则是指数据库中一组对象之间某种关联关系(例如"同时发生"或者"从一个对象可以推出另一个")。一条关联规则是有趣的如果它满足最小支持度和可信度的限制。这导致大量平凡的规则产生。本文将设计一个算法挖掘这样的有趣规则，它的前件和后件分别属于不同的概念类，称这种规则为类间桥。

类间桥在行销中的交叉销售，生物工程中的嫁接，化学中的合成等应用中有重要的应用价值。例如，"啤酒→尿不湿"是从市场营销数据中挖掘出的一条关联规则，也是引起数据挖掘被工业界的支持和学术界的深入研究的例证。这是因为，"啤酒"和"尿不湿"是属于两个不同概念类中的对象，并且规则是潜在的、事先不被人们所认知的、隐含的、有用的信息。它的用处在于通过交叉销售来刺激两种产品的销量。显然，"咖啡→方糖"作为一个例子就很难说服工业界对数据挖掘的投资。

欲寻找的数据间的相互关系的特征：它存在于两个不同概念类中的两个对象之间，就像一座"桥"一样，把这两个对象联系起来。通过发现这种"桥"，可以找到一些有趣的、有价值的模式("啤酒和尿不湿"的例子中表现为把啤酒和尿不湿放在一起增加了两种产品的销量)。

2　介绍算法

2.1　算法描述

本文的实现方法是在一种聚类算法——Chameleon 算法[3]的基础上通过改进得到的。已知的一些聚类算法通常只适用于静态模型，且都有各自的不足之处。K-means[4]、PAM[5]、CLARANS[6]算法，它们把分成的簇设想为一个个的椭球体或球体，且都具有相同或相近的尺寸，这显然在很多时候是不符合实际情况的。DBSCAN[7]是一种基于密度的聚类算法，它假设在同一个簇中的点都是密度可达的，而不在同一个簇中的点则是密度不可达的。CURE[8]和 ROCK[9]算法属于凝聚的层次聚类算法。CURE 算法度量两个簇间的相似度是以属于这两个簇的最近(这个最近可以指相似度的大小、距离的远近或其他标准)的代表点对的相似度为基础，没有从整体上考虑这两个簇的紧密程度；而 ROCK 算法的度量方法是比较两个簇的综合类间互连程度值和用户指定的互连程度值的大小，大于用户指定值的被认为是可以合并的簇，反之则不可合并。这种算法忽视了在同一数据集中不同簇间的互连程度的潜在变化情况。就是说，前者只注意了类内相似度对待合并簇的影响，忽视了互连性(两个簇间距离较近数据对的多少)所可能造成的影响；而后者则只注意合计相似度的结果，忽略了近似性(簇间数据对的最近距离)对合并簇的影响。它们的缺点在 Chameleon 算法中得到了较好的解决，Chameleon 是一种利用动态模型的层次聚类算法，它综合考虑了类内相

*)基金资助项目：澳大利亚 ARC 项目(DP0559536)；国家自然科学基金重大项目(60496321)；国家自然科学基金项目(60463003)。张师超　教授，博士，研究方向为数据挖掘，人工智能等；陈　峰　硕士研究生，研究方向为数据挖掘，数据库；尤晓芳　硕士研究生。

似度和类间相似度在合并簇时的影响，收到了较好的效果。

Chameleon 算法用 $RI(Ci,Cj)$ 函数表示相对互连性(Relative Inter-Connectivity)；$RC(Ci,Cj)$ 函数表示相对近似性(Relative Closeness)：

$$RI(Ci,Cj)=\frac{|EC_{\{G,G\}}|}{\frac{|EC_G|+|EC_G|}{2}}$$

其中，$EC_{\{G,G\}}$（又名绝对互连性）表示连接簇 Ci 和 Cj 的所有边的权重和；EC_G 表示把簇 Ci 分为两个大致相等部分的最小等分线的切断的所有边的权重和。相对互连性函数能处理簇间形状不同和互连程度不同的问题。

$$RC(Ci,Cj)=\frac{\bar{S}_{EC_{\{G,G\}}}}{\frac{|Ci|}{|Ci|+|Cj|}\bar{S}_{EC_G}+\frac{|Ci|}{|Ci|+|Cj|}\bar{S}_{EC_G}}$$

其中，$\bar{S}_{EC_{\{G,G\}}}$ 表示连接簇 Ci 和 Cj 的边的平均权重；$\bar{S}_{EC_G}$ 表示把簇 Ci 划分为两个大致相等部分的最小等分线切断的所有边的平均权重。

整个算法的步骤如图 1 所示。

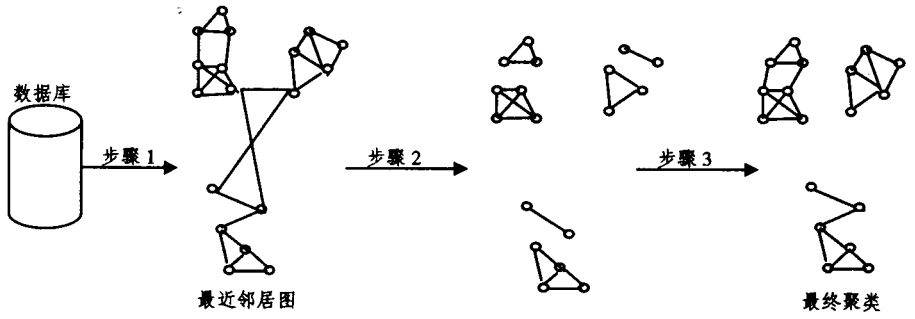

图 1

步骤一：构造最近邻居图(Construct a sparse graph)

从大数据集中随机抽取一部分数据对象(这里可以先进行一些数据预处理以去除孤立点和噪声，或其他一些不利影响)，两两计算相似度。度量相似度是用 $\frac{|A\cap B|}{|A\cup B|}$，即对象 A 和对象 B 中属性值相同的属性个数和所有属性个数的比值。相似度越大，认为两个对象越近似。

将每一个对象和其他对象的相似度从大到小进行排序，取前 k 个相似度值，k 个以外的认为相似度太小，可以忽略不加考虑。据此列出相似度矩阵，并将此矩阵转化成一个稀疏图，即 k 最近邻居图(k-nearest nerghbor graph)[10]。图中节点表示数据对象，边表示边的一个节点 u 在另一个节点 v 的 k 个最近似邻居节点中。使用 k 最近邻居图有如下好处：

- 距离很远的数据对象完全不相连。
- 边的权重代表了潜在的空间密度信息。
- 在密集和稀疏的数据区域都能同样建模。
- 表示的稀疏便于使用有效的算法。

步骤二：分割 k 最近邻居图形成待合并的子簇(Partiton the graph)

利用 hMetis 算法[11]根据最小化截断的边的权重和来分割 k 最近邻居图，得到待合并的子簇。

首先是粗糙化阶段：把原始的 k 最近邻居图粗糙化，把相似度大的节点合并，用一个节点代表，从而减少以后的运算量同时保证运算结果的正确率不会有太大损失。

接下来是初始划分阶段：把上面的得到的粗糙图划分为两部分。在这两部分满足用户指定的平衡因子(两部分中节点的多少满足一定的比例关系)的前提下找到一个最小划分，使得截断的边的权重和最小。改进后的 hMetis 算法可以把粗糙图划分为多个部分[12]。

最后是反粗糙化和修正阶段：此时粗糙程度较高的图的划分被映射到上一级粗糙程度较低、节点较多的图上，并利用 Fiduccia-Mattheyses 算法[13]移动节点以便获得最好的划分。

之所以要先形成子簇，再有所选择地将其中的一部分合并，从而得到最终的聚类结果而不是直接进行划分形成聚类，是因为准确地计算簇内的互连性和近似性需要簇有足够的数据对象。

步骤三：合并子簇形成最终的聚类(Merge Partitions)

有两种合并标准用于子簇的合并，在此一一介绍如下：

1　用户指定阈值(T_{RI} 和 T_{RC})

i. 计算每个子簇与其临近子簇的 RI 和 RC。

ii. 合并 RI 和 RC 分别大于 T_{RI} 和 T_{RC} 的子簇对，若满足条件的子簇对多于一个，则合并具有最大绝对互连性的子簇对。

重复上面两步，直到没有可合并的簇。

2　定义度量函数 $RI(Ci,Cj)\times RC(Ci,Cj)^a$

$a>1$ 表示更重视相对近似性；$a<1$ 表示更重视相对互连性。每次选择使该函数值最大的子簇对合并。

本文用于试验的程序中采用的是第一种方法。

2.2　算法的改进

在 Chameleon 算法中，相似度被作为聚类的标准。通过把相似度大的数据对象聚为一类，把相似度小的数据对象分割开来不再考虑它们二者的联系，可以得到类内相似度最大、类间相似度最小的聚类结果。

注意到相似度表征的是两个数据对象间的相似程度。相似度越大，认为两个数据对象的特征越接近，两个对象越近似。在整个算法的三个步骤中，贯穿着这样一种思想：相似度小的数据项对被认为是近似度很低的，其联系可以被忽略。在步骤二分割 k 最近邻居图形成待合并的子簇和步骤三合并子簇形成最终的聚类中，都是把权重大的边(即相似度大的边)保留下来，而把权重小的边(即相似度小的边)舍弃不予考虑，从而得到最终聚类结果。

进一步考虑到：相似度小的数据项对并不是没有联系，只是联系较小，不足以使它们被分为一类而已。在前面提到的“啤酒和尿不湿”的例子中，如果按照酒类和婴儿用品类划分

的话，它们显然是属于不同的类。但实践已经证明它们二者是有联系的：啤酒和尿不湿放在一起销售同时增加了二者的销量。因此，保留下类间数据对象间的这种联系无疑是很有意义的。这种联系就是我们要找的“桥”。

由于本文中“桥”的度量标准和划分聚类时用的度量标准是一致的：相似度，因此利用 Chameleon 算法稍加改进就能找出所需要的联系。

在步骤三合并子簇形成最终的聚类中，定义了阈值（T_{RI} 和 T_{RC}）作为划分聚类的标准：凡 RI 和 RC 分别大于 T_{RI} 和 T_{RC} 的子簇对被认为是可以合并的对（若满足条件的子簇对多于一个，则合并具有最大绝对互连性的子簇对），反之则不予考虑。但事实上，RI 和 RC 都小于 T_{RI} 和 T_{RC} 子簇对或者有任意一个小于对应阈值的子簇对也是存在的，只是由于这种子簇对不满足合并条件，所以它们之间原本蕴含着的联系也被忽略了。这样考虑可以简化运算量，同时对于 RI 和 RC 确实很小（甚至等于 0）的子簇对，的确没有必要把它们考虑在一起，这样做也可以减少干扰。但令人遗憾的是，在下列情况下，这种划分方法显然遗漏了一些潜在的、可能很有价值的联系：

1　当 RI 和 RC 仅仅比 T_{RI} 和 T_{RC} 小很少时。

2　RI 和 RC 中有一个比对应阈值小而另一个却相等甚至远远大于其对应阈值时。

显然在这两种情况下，由于子簇对不满足合并条件，被认为是不能被合并的簇对，因而，簇对间数据对象的联系被一并认为是无价值的而丢弃。但实际上这些联系并不一定就是没有价值的。还是用“啤酒和尿不湿”做例子，啤酒和尿不湿显然不属于同一类商品，而且按照上述分类标准来看的话，它们之间也不会有什么联系（即使有也被丢弃了，因为啤酒和尿不湿之间的联系显然不会比啤酒和白酒，或尿不湿和奶瓶之间的联系更强）。本文正是要找到这种类间的联系，即“桥”。这些类间联系本来是有价值的、不应该被丢弃的。因此，根据以上考虑，在 Chameleon 算法原来的基础上对其做改进如下：

1　步骤三中的阈值（T_{RI} 和 T_{RC}）定义为两组，$T1_{RI}$，$T1_{RC}$ 和 $T2_{RI}$，$T2_{RC}$。并且规定 $T1_{RI} > T2_{RI}$，$T1_{RC} > T2_{RC}$；同时令 $T1_{RI}$ 和 $T1_{RC}$ 分别等于原来指定的阈值 T_{RI} 和 T_{RC}。

2　在合并子簇以形成最终聚类时，计算出的簇与邻近簇之间的 RI 和 RC 应分别和两组阈值比较。凡 RI 和 RC 分别大于 $T1_{RI}$ 和 $T1_{RC}$ 的簇对，被认为是可以合并的簇对，与原算法一样，把这些簇合并；凡 RI 和 RC 分别大于 $T2_{RI}$ 和 $T2_{RC}$ 同时又分别小于 $T1_{RI}$ 和 $T1_{RC}$ 的簇对，则被认为虽然属于不同的类，但类之间较之其他类对相比却又有比较紧密地联系。通过记录下这种子簇对，可以记录下簇对间数据对象的相似度，即数据对象之间的联系，这种联系，就是本文感兴趣的“桥”。

该算法是一个递归的过程，在不断地合并符合条件的子簇对的同时，记录下的“桥”也会根据所合并的簇对的变化情况被同步更新。当聚类过程结束，最终聚类结果形成时，寻找“桥”的工作也同时完成了。就是说，寻找“桥”与聚类是同步进行的，而不需要在聚类完成之后再找，这样做节省了运算量，保证了算法的效率，同时原算法的精度也不会受到影响。经过这样的改进，新算法中既保留了原有算法所生成的聚类结果，又找到了所感兴趣的类间“桥”的联系。同时，联系很小的簇对（RI 和 RC 分别小于 $T2_{RI}$ 和 $T2_{RC}$ 的簇对）被认为是无研究价值，不予考虑而舍弃。这样在得到预期结果的同时也保证了算法的质量和效率。

下面给出一个寻找类间“桥”的具体例子。

2.3　实例

依据上面所论述的算法思想，编程实现了改进后的 Chameleon 算法。所用的数据集是从网上下载的一个 zoo（动物园）数据集。共有 101 个数据，根据需要扩展成 150 个数据，每个数据有 18 个属性。为了试验需要，去掉了标志动物名称和类别的两个属性，将剩余的 16 个属性的属性值经过必要转换，设计成试验所需要的布尔项，并在此转换数据集上进行试验。

实验结果表明，依照改进后的 Chameleon 算法，不但得到了理想的聚类效果，而且找到了所希望找的“桥”，同时并没有牺牲算法的效率。根据实验结果，作出类间“桥”的示意图如图 2 所示。

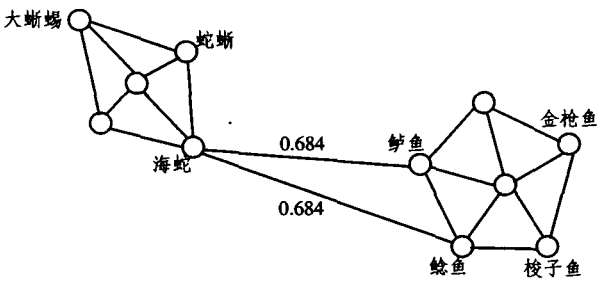

图 2

在本示意图中，节点代表动物，节点之间的连线表示两种动物间的相似程度。为了表示的明确与简便，以及突出所寻找的类间“桥”的关系，这里并没有给出所有节点所代表的动物名称和位于同一类内的动物之间的相似度，而是只给出了有代表性的节点所代表的动物名称和分别位于两个不同类中的动物的相似度度量值。图中，海蛇和鲈鱼，以及海蛇和鲶鱼的这种类间的联系被看作是“桥”加以分析。

如图 2 所示，海蛇、蛇蜥和大蜥蜴属于一类；鲈鱼、鲶鱼、金枪鱼和梭子鱼属于另一类。按照原 Chameleon 算法的划分标准，我们只能找到这两个聚类结果。现在用改进后的 Chameleon 算法，在得到聚类结果的同时，也找到了本来在原算法中被丢弃的类间联系，而这种类间联系也许是有一定意义，应该加以注意的联系。

在本例中可以看到，海蛇和鲈鱼、鲶鱼分属于不同的类，但它们之间却有着其他类间数据对象所没有的这种“桥”的联系。为什么会存在这种“桥”？同时，海蛇和金枪鱼、梭子鱼为什么没有这种“桥”？而大蜥蜴、蛇蜥和鲈鱼、梭子鱼等组成的这一类又为什么也没有这种“桥”？“桥”作为类间数据对象相互联系的一种特殊存在形式，对于进一步揭示两类的联系和区别，以及进一步探索“桥”两端的数据对象有别于其他数据对象的特殊意义有何帮助？找到这种“桥”后，应如何加以解释，挖掘其形成的深层次原因并加以利用？这些都是我们找到“桥”后所面临的问题和挑战。

3　实验分析

由于该算法是在原 Chameleon 算法的基础上改进而成的，因此它继承了原有算法的优点。在聚类过程中，将近似性和互连性都高的子簇合并，可以发现任意形状的高质量的聚类结果；同时对孤立点和噪声不甚敏感；聚类结果对参数选择的依赖程度也不是太强。但它同时也有聚类算法所共有的一

个通病:运算量太大,在最坏情况下高维数据的处理代价可能要 $O(n^2)$ 的时间。而且实验结果表明:相对而言,该算法在处理大数据量时,所得的聚类结果要优于处理小数据量时的结果。

改进后的 Chameleon 算法严格来说对原有算法的聚类效果没有什么提高,本文的重点是去发现类间"桥"并试图对其形成的原因做出解释,以便加以利用。这里能够保证的就是:在巩固原聚类结果,找出感兴趣的"桥"的同时,算法的质量和效率不会受到影响。

此外,在实验中还发现了一个与"桥"和聚类效果有关的问题。大家都知道,聚类,尤其是大数据量的聚类,要想得到百分之百正确的聚类结果基本上是不可能的。在找到的"桥"中,有这样一种"桥"引人注意:A 类中的某个对象 a 和 B 类中的很多对象 $b1$、$b2$、$b3$ 等等都有这种"桥"的联系,且相似度都很大。对照正确的聚类结果(实验所用数据集较小,有正确的聚类结果作为参考)发现,a 实际上是被错误地聚类了,它应该被归在 B 类中,和 $b1$、$b2$、$b3$ 等对象是属于同一类的。这种现象给出了一个启示:在对大数据集进行聚类时,对所有的数据对象都去验证它们是否被正确的聚类是不现实的,也是没有必要的。但在找到的"桥"中,留意上面提到的这种现象:A 类中的某个对象与 B 类中的很多对象都有联系且相似度都很大时,该对象很可能是被聚到了错误的类中,正确的聚类结果,该对象应该是属于 B 类的。这一特点可以作为衡量聚类算法效果是否理想的一个佐证。

本算法中有这样一点是值得注意的:划分聚类和寻找"桥"时使用的是同一种度量标准——相似度。而相似度的计算方法也是一致的:用到了数据集中所提供的所有属性,在聚类和找"桥"时都是如此。那么,这样的计算方法有什么优点?在实际中是否有广泛的应用领域和效果?可否加以改进,使得或提高算法效率,或改善结果?这些疑问在"啤酒和尿不湿"一例中或许可以得到一些启发。

关于把啤酒和尿不湿放在一起同时销售,从而增加了二者的销量,在市场营销学中是这样解释的:青年夫妇刚有了一个婴儿,如果此时由于妻子有事,需要丈夫去买尿不湿时,做丈夫的通常不会只买尿不湿,而会同时买一两件男人需要的东西,比如啤酒。这就是啤酒和尿不湿销量同时增加的秘密所在。事实上,沃尔玛公司在实际操作中并不是只把啤酒和尿不湿放在一起,同时放在一起的还有香烟和剃须刀片等男士用品,也都收到了不错的效果。

那么得到的启示又在哪里呢?有一点不要忘记,商业运作中对商品的分类或聚类最终都是为了将相应的客户群区别开来,以便制定相应的对策提高利润。如果按照本文介绍的算法思想,以属性不同或相同属性的不同属性值来划分顾客类,就会引发以下的问题:

通常情况下,在规划购买啤酒的顾客特征时,是否结婚应该不是一个特别重要的属性,但在考虑购买尿不湿的顾客特征时,该属性就显得尤为重要了。所以,在考虑啤酒和尿不湿的联系的时候,这个属性也是不可或缺的。这就是说,当利用本文介绍的算法找"桥"时,由于事先不知道所找的"桥"到底和那种属性联系紧密,因此可能需要考虑一些在单纯进行聚类操作时不需要考虑的属性。在大数据集中,增加属性所引起的运算代价可能是很大的。鉴于此,设想引入粗糙集中属性约简的概念,试图找到一种新的方法,在考虑聚类时用一部分属性;考虑桥时用另一部分属性。这两部分属性可能相同,也可能不同。希望借此提高算法的效率。但如此一来,又引发了另外的问题:究竟属性该如何选择?聚类与找"桥"是否还可以同步进行?这是下一步的研究方向。

总结 针对当前聚类研究中的现状,提出了一个新的研究方向:寻找数据中不同概念类的对象间的联系。并从理论和实际中的例子两方面论述了寻找这种关系的意义和价值,然后对一个原有算法作了改进,进行了找"桥"的初步尝试,并得到了预期的结果。同时在此基础上,总结了现有算法的优缺点、适用范围以及下一步工作可能面临的困难,为今后的研究工作指出了研究的思路和方向。

参考文献

1 Egghe L, Michel C. Construction of weak and strong similarity measures for ordered sets of documents using fuzzy set techniques. Information Processing and Management, 139:771～807

2 胡侃,夏绍玮. 基于大型数据库的数据采掘:研究综述. 软件学报,1998,9(1)

3 Karypis G, et al. CHAMELEON: A Hierarchical Clustering Algorithm Using Dynamic Modeling. IEEE Computer, 1999. 68～75

4 Jain A K, Patrick E A. Algorithms for Clustering Data. Prentice Hall, 1998

5 Kaufman L, Rousseeuw P J. Finding Groups in Data: an Introduction to Cluster Analysis. John Wiley & Sons, 1990

6 Ng R, Han J. Efficient and effective clustering method for spatial data mining. In: Proc. of the 20th VLDB Conf. Santiago, Chile, 1994. 144～155

7 Ester M, Kriegel H P, Sander J, Xu X. A density-based algorithm for discovery clusters in large spatial databases with noise. In: Proc. of the Second Int'l Conf. on Knowledge Discovery and Data Mining, Portland, OR, 1996

8 Guha S, Rastogi R, Shim K. CURE: An efficient clustering algorithm for large databases. In: Proc. of 1998 ACM-SIGMOD Int. Conf. on Management of Data, 1998

9 Guha S, Rastogi R, Shim K. ROCK: a robust clustering algorithm for categorical attributes. In: Proc. of the 15th Int'l Conf. on Data Eng., 1999. 345～366

10 Eppstein D, Paterson M S, Yao F F. On Nearst-Neighbor Graphs

11 Karypis G, Aggarwal R, Kumar V, Shekhar S. Multilevel Hypergraph Partitioning: Applications. In: VLSI Domain 34th Design Automation Conf.

12 Karypis G, Kumar V. Multilevel k-way Hypergraph Partitoning. In: Proc. 36th Design Automation Conf.

13 Fiduccia C M, Mattheyses R M. A linear time heuristic for improving nerwork partitions. In: Proc. 19th IEEE Design Automation Conf. 1982. 57～62