第 42 卷 第 6A 期
2015 年 6 月

计 算 机 科 学
Computer Science

Vol. 42 No. 6A
June 2015

基于关联矩阵的软件演化过程结构验证

刘金卓[1,2] 于 倩[1,2] 赵 娜[1,2] 谢仲文[1,2] 郁 湧[1] 杭菲璐[3] 金运志[1]

(云南大学软件学院 昆明 650091)[1] (云南大学云南省软件工程重点实验室 昆明 650091)[2]

(云南电网有限责任公司信息中心 昆明 650217)[3]

摘 要 软件演化过程领域将软件演化和软件过程领域联系起来,为了适应新的需求和新的环境,越来越多的软件演化过程模型被建模出来。针对软件演化过程模型结构合理性验证还没有有效实现的问题,对基于白盒建模的软件演化过程模型的结构性质进行了验证,其中包括结构有界性、可重复性和守恒性等,采用了关联矩阵的方法,证明了通过白盒建模得到的软件演化过程模型自身具备着非常优良的结构性质,提高了软件演化过程的质量。

关键词 软件演化过程,结构验证,关联矩阵,白盒建模,结构性质

中图法分类号 TP311 文献标识码 A

Structure Verification Method for Software Evolution Process Based on Incidence Matrix

LIU Jin-zhuo[1,2] YU Qian[1,2] ZHAO Na[1,2] XIE Zhong-wen[1,2] YU Yong[1] HANG Fei-Lu[3] JIN Yun-zhi[1]

(School of Software, Yunnan University, Kunming 650091, China)[1]

(Key Laboratory for Software Engineering of Yunnan Province, Yunnan University, Kunming 650091, China)[2]

(Information Center, Yunnan Power Grid Co., Ltd., Kunming 650217, China)[3]

Abstract The software evolution process, the inter-discipline of software process and software evolution, becomes a key area in software engineering. In the recent past, many researchers have paid more attention to and devoted great efforts in this area and have made great progress. As the structure of software evolution process has not been effectively verified, the incidence matrix approach is used to prove the properties such as the structural boundedness, repeatability, conservativeness and so on. Therefore, the structure properties of the model that modelling software processes by using the white box approach is proved. Consequently, the quality of software evolution processes is improved.

Keywords Software evolution process, Structure verification, Incidence matrix, White box modeling, Structure properties

1 概述

软件是客观事物的一种反映,客观世界的不断变化促使软件技术的不断发展,对于事物发展规律的探寻促使软件工程的产生和发展[1]。在实际生活当中,软件的应用也越来越广泛,Lehman 在文献[2]中认为,现实世界的软件如果不能持续不断地演化以适应环境的变化,就会变得越来越没有价值,这说明,软件演化已经成为软件生存周期的重要特性,软件演化过程也成为软件过程的一个重要方面。对软件演化过程进行建模能使软件演化按照定义好的软件演化过程执行,以便管理良好的软件演化过程能用于软件演化并模拟反应真实演化过程的行为模式。

文献[3]基于基本 Petri 网扩展了面向对象技术和 Hoare 逻辑,提出了软件演化过程元模型 EPMM(Software Evolution Process Meta Model),以及一系列建模方法、技术和工具,用以支持软件演化过程的建模。EPMM 分为 4 层:全局层、过程层、活动层和任务层。由 EPMM 建模产生的软件演化过程模型也应该包含同样的 4 层,其中,过程层的建模产生的软件过程本质上是基于基本 Petri 网的。在过程层,通过对活动用白盒方法和黑盒方法进行细化操作产生软件演化过程模型。

Petri 网的精细化设计思想一直为理论界和工业界所关注,已有大量的研究工作。Brauer[4]根据精细化后所保持的性质的不同,将精细化操作技术分为两类:一类是保持行为性质(如活性和有界性)的精细化操作;二是保持语义等价的精细化操作。文献中的精细化操作技术(如文献[5,6])大多属于第一类。文献[7]基于一般随机 Petri 网,提出了一种逐步建立可靠性模型的方法,先建立模型,然后根据有关条件再进

本文受国家自然科学基金(61262024,61462091,61462095,61462092),云南省自然科学基金(2014FD006,2012FB119,2013FB008),云南省教育厅科学研究基金(2013Z057,2014Y012,2011Y121),云南省软件工程重点实验室开放基金(2012SE401,2012SE308)资助。

刘金卓(1987—),女,博士,讲师,CCF 会员,主要研究方向为软件过程、软件工程、形式化验证,E-mail:jinzhuo.liu@hotmail.com;于 倩(1978—),女,博士,讲师,主要研究方向为软件工程;赵 娜(1982—),女,博士,讲师,主要研究方向为软件工程;谢仲文(1982—),男,博士,讲师,主要研究方向为软件工程;郁 湧(1981—),男,博士,副教授,主要研究方向为软件工程;杭菲璐(1984—),男,硕士,主要研究方向为软件工程;金运志(1989—),男,博士生,主要研究方向为软件工程。

行精细化，文献[8]按照逐步精细化的描述方法，模拟和实现整个制造系统(MS)，文献[9]对工作流网进行了保持正确性的精细化操作，在文献[3]中，EPMM 在过程层基于基本 Petri 网，提出了一种用白盒方法对软件演化过程模型进行逐步细化的方法。

本文的目的是研究通过白盒方法对软件演化过程模型进行精细化操作之后相应的结构性质保持问题。给出了精细化后的软件演化过程模型保持结构有界性、守恒性、可重复性等性质的明确易懂的证明。值得一提的是本文给出的结构性质证明能够让建模者和使用者放心地使用通过白盒建模得到的软件演化过程模型，确保其结构不发生异常。

2 相关概念

2.1 Petri 网相关概念

软件演化过程模型是基于 Petri 网的建模模型，而关联矩阵是 Petri 网的主要分析方法之一。

定义 1(关联矩阵[10]) 设 $\Sigma=(S,T;F,M_0)$ 是一个 Petri 网，$S=\{s_1,s_2,\cdots,s_m\}$，$T=[t_1,t_2,\cdots,t_n]$，则 Petri 网 Σ 的结构 $(S,T;F)$ 可以用一个 m 行 n 列矩阵

$$D=\lfloor d_{ij}\rfloor_{m\times n}$$

来表示，其中

$$\begin{cases} d_{ij}=1, & \text{若}(t_i,s_j)\in F \\ d_{ij}=-1, & \text{若}(s_i,t_j)\in F \\ d_{ij}=0, & \text{其他情况} \end{cases}$$

定义 2(结构有界性[10]) 设 $N=(S,T;F)$ 为一个网，如果对 N 赋予任意初始标识 M_0。网系统 (N,M_0) 都是有界的，则称 N 为结构有界网。

定理 1[10] 设 A 为网 $N=(S,T;F)$ 的关联矩阵，则 N 为结构有界网的充分必要条件是：存在 $m(m=|S|)$ 维正整数向量 Y，使得 $AY\leqslant 0$。

定义 3(守恒性[10]) 设 $N=(S,T;F)$ 为一个网。如果存在一个 $m(m=|S|)$ 维正整数权向量 $Y=[y(1),y(2),\cdots,y(m)]^{\mathrm{T}}$，使得对 N 的任一个初始标识 M_0 和任意 $M\in R(M_0)$ 都有

$$\sum_{j=1}^{m}M(s_j)Y(j)=\sum_{j=1}^{m}M_0(s_j)Y(j)$$

则称 N 为守恒的。

定理 2[10] 设 $N=(S,T;F)$ 为一个网，A 为 N 的关联矩阵。N 为守恒网的充分必要条件是：存在 $m(m=|S|)$ 维正整数向量 Y，使得 $AY=0$。

定义 4(可重复性[10]) 设 $N=(S,T;F)$ 为一个网。若存在 N 的一个初始标识 M_0 和一个无限的变迁序列 σ，使得 $M_0[\sigma>$，且 $\forall t\in T$ 在 σ 中无限多次地出现，则称 N 为一个可重复网。M_0 称为 N 的一个可重复标识。

定理 3[10] 设 $N=(S,T;F)$ 为一个网，A 为 N 的关联矩阵，则 N 为可重复网的充分必要条件是：存在 $n(n=|T|)$ 维正整数向量 X，使得 $A^{\mathrm{T}}X\geqslant 0$。

定义 5(协调性[10]) 设 $N=(S,T;F)$ 为一个网。若存在 N 的一个初始标识 M_0 和一个变迁序列 $\sigma\in T^*$，使得 $M_0[\sigma>M_0$，而且 $\forall t\in T:\#(t/\sigma)\geqslant 1$，则称 N 为一个协调网。

定理 4[10] 设 A 为网 $N=(S,T;F)$ 的关联矩阵，则 N 为协调网的充分必要条件是：存在 $n(n=|T|)$ 维正整数向量 X，使得 $A^{\mathrm{T}}X\geqslant 0$。

定义 6(死锁与陷阱[10]) 设 $N=(S,T;F)$ 为一个网，$S_1\subseteq S$。如果 $\cdot S_1\subseteq S_1^{\cdot}$，则称 S_1 为网 N 的一个死锁；如果 $S_1^{\cdot}\subseteq\cdot S_1$，则称 S_1 为网 N 的一个陷阱。

定理 5[10] 设 $N=(S,T;F)$ 为一个网，A 为 N 的关联矩阵。$S_1=\{S_{i_1},S_{i_2},\cdots,S_{i_k}\}$ 为网的一个死锁(或陷阱)充分必要条件是：A 关于 S_1 列的列生成子阵中，每个非全零行至少包含一个“-1”(或“1”)元素。

2.2 软件演化元模型

2008 年，文献[3]中第一次提出了软件演化过程元模型(EPMM)。EPMM 是一个形式化的元模型，用来建模软件演化过程模型。EPMM 元模型是在基本 Petri 网并扩展面向对象技术和 Hoare 逻辑的基础上提出的，它定义了软件演化过程模型的结构和语法。

自顶向下，EPMM 分为 4 层：全局层、过程层、活动层和任务层。在建模的过程中，为了支持自顶向下逐步求精的建模，文献[3]提出了两种建模方法：白盒建模和黑盒建模。在白盒建模的过程中，文献[3]提出了基本块；在黑盒建模的过程中，文献[3]提出了过程包。在细化的过程中，文献[3]中还证明了不论是白盒建模还是黑盒建模，抽象模型和细化模型间接口都是一致的。

定义 7(初始块[3]) 初始块 $I=(C,A;F,M_0)$ 是一个四元组，如图 1 所示，其中：

(1)C,A 和 F 分别称为条件集、活动集和弧集；

(2)M_0 是初始情态。

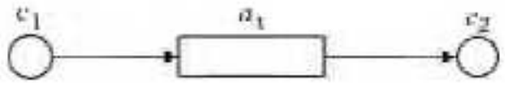

图 1 初始块

定义 8(基本块[3]) 基本块 $b=(C,A,F,A_e,A_x)$ 是一个五元组，其中：

(1)C,A 和 F 分别称为条件集、活动集和弧集；

(2)$A_e,A_x\subseteq A$ 分别称为基本块 b 的入口和出口。

文献[3]定义了 4 种基本块，分别是顺序块、选择块、迭代块和并发块(见定义 9—定义 12)。总的来说，基本块是一个单入口和单出口、没有初始情态、内部结构可见、结构化的 Petri 网模块。

定义 9(顺序块[3]) 顺序块描述了活动 e_i 和 e_j 的顺序执行，如图 2 虚线框所示。形式描述，$C=\{c\}$，$A=\{e_i,e_j\}$，$F=\{(e_i,c),(c,e_j)\}$，$A_e=\{e_i\}$，$A_x=\{e_j\}$。

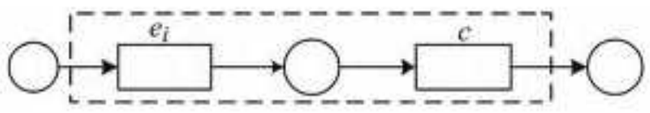

图 2 顺序块

定义 10(选择块[3]) 选择块描述了活动 e_i 和 e_j 的选择执行，如图 3 虚线框所示。形式描述，$C=\{\ \}$，$A=\{e_i,e_j\}$，$F=\{\ \}$，$A_e=\{e_i,e_j\}$，$A_x=\{e_i,e_j\}$。

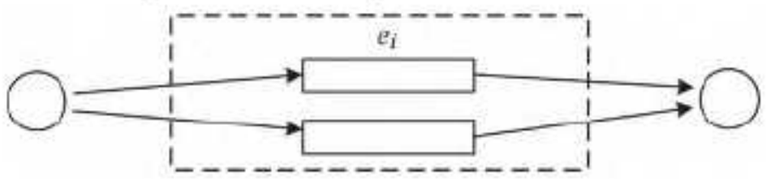

图 3 选择块

定义 11(并发块[3]) 并发块描述了活动 e_i 和 e_j 的并发执行，如图 4 虚线框所示。形式描述，$C=\{c_1,c_2,c_3,c_4\}$，$A=\{e_0,e_i,e_j,e_n\}$，$F=\{(e_0,c_1),(e_0,c_2),(c_1,e_i),(c_2,e_j),(e_i,c_3),$

$(e_j,c_4),(c_3,e_n),(c_4,e_n)\},A_e=\{e_0\},A_x=\{e_n\}$。

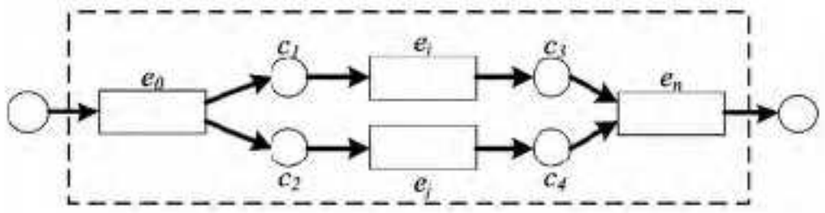

图 4　并发块

定义 12(迭代块[3])　迭代块描述了活动 e_i 和 e_j 的迭代执行，如图 5 虚线框所示。形式描述，$C=\{c_1,c_2\},A=\{e_0,e_i,e_j,e_n\},F=\{(e_0,c_1),(c_1,e_i),(e_i,c_2),(c_2,e_j),(e_j,c_1),(c_2,e_n)\},A_e=\{e_0\},A_x=\{e_n\}$。

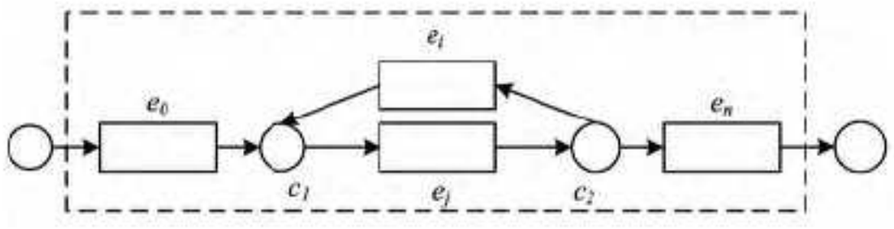

图 5　迭代块

3　软件演化过程模型关联矩阵

用 Petri 网对实际系统进行建模，主要目的就是借助系统模型来分析实际系统的性质和功能。如果一个 Petri 网模型确切地描述了一个系统的结构和运行，那么这个系统所具有的一些性质也会在其 Petri 网模型上得到体现。

使用 EPMM 建模软件演化过程模型的过程中，本文的作者之一提出了 4 种基本块：顺序块、选择块、并发块和迭代块，支持白盒建模[3]。从本质上来讲，软件演化过程模型是一个扩展了面向对象技术和霍尔逻辑的基本 Petri 网，因此，可以应用基本 Petri 网的一些方法来探讨软件演化过程模型的结构性质。本文将利用关联矩阵的方法证明软件演化过程模型的结构性质。

由于本文的内容经常涉及到向量大小的比较，在此先对有关记号做一些约定。

设

$$V_1=[V_1(1),V_1(2),\cdots,V_1(n)]$$

$$V_2=[V_2(1),V_2(2),\cdots,V_2(n)]$$

为两个 n 维向量。如果 $\forall i\in\{1,2,\cdots,n\},V_1(i)\leqslant V_2(i)$，则记为

$$V_1\leqslant V_2 \text{ 或 } V_2\geqslant V_1$$

如果 $V_1\leqslant V_2$，而且存在 $i\in\{1,2,\cdots,n\}$，使得 $V_1(i)<V_2(i)$，则记为

$$V_1<V_2 \text{ 或 } V_2>V_1$$

用 0 表示各个分量均为 0 的向量(全零向量)，因此 $V\geqslant 0$ 表示 V 是一个非负向量(在 Petri 网理论中，常表示非负整数向量)。

白盒建模的软件演化过程模型是由基本块不停地对行为进行细化而得到的，因此讨论软件演化过程模型的关联矩阵，核心问题有两个：一个是给出基本块的关联矩阵；另一个是给出细化操作后关联矩阵的变化。

3.1　基本块的关联矩阵

首先对基本块内的元素重新进行排序，对于顺序块和选择块，$a_i=a_1,a_j=a_2$；对于并发块和迭代块，$a_0=a_1,a_i=a_2,a_j=a_3,a_n=a_4$。根据定义 1，可以分别得到顺序块的关联矩阵：

$$\begin{array}{c} \\ a_1 \\ a_2 \end{array}\begin{array}{c} \begin{array}{c} c_1 \end{array} \\ \begin{bmatrix} 1 \\ -1 \end{bmatrix} \end{array}$$

选择块的情况比较特殊，模块内没有 c，因此选择块没有关联矩阵。用选择块对初始块中的行为进行细化后得到的模型的关联矩阵来代替选择块的关联矩阵：

$$\begin{array}{c} \\ a_1 \\ a_2 \end{array}\begin{array}{c} \begin{array}{cc} c_1 & c_2 \end{array} \\ \begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix} \end{array}$$

并发块的关联矩阵：

$$\begin{array}{c} \\ a_1 \\ a_2 \\ a_3 \\ a_4 \end{array}\begin{array}{c} \begin{array}{cccc} c_1 & c_2 & c_3 & c_4 \end{array} \\ \begin{bmatrix} 1 & 0 & 1 & 0 \\ -1 & 1 & 0 & 0 \\ 0 & 0 & -1 & 1 \\ 0 & -1 & 0 & -1 \end{bmatrix} \end{array}$$

迭代块的关联矩阵：

$$\begin{array}{c} \\ a_1 \\ a_2 \\ a_3 \\ a_4 \end{array}\begin{array}{c} \begin{array}{cc} c_1 & c_2 \end{array} \\ \begin{bmatrix} 1 & 0 \\ -1 & 1 \\ 1 & -1 \\ 0 & -1 \end{bmatrix} \end{array}$$

3.2　细化操作后的关联矩阵

细化操作的定义为：设 $P=\langle C,A,F,M_0\rangle$ 是一个软件演化过程模型，$b=\langle C,A;F,A_e,A_x\rangle\in Block$，从 P 中细化活动 $a\in A$ 为 b 后，将产生一个新的软件演化过程模型 P'，且满足以下 4 个条件：

(1) $P'.A=(P.A-a\cup b.A)$；

(2) $P'.C=(P.C\cup b.C)$；

(3) $P'.F=(P.F-inflow(a)-outflow(a)\cup\{(x,y)|(x,a)\in inflow(a)\wedge y\in b.A_e\}\cup\{(x,y)|x\in b.A_x\wedge(a,y)\in outflow(a)\})\cup b.F$；

(4) $P'.M_0=P.M_0$。

其中，$Block$ 是基本块集(见定义 8)，$b.A$ 是基本块的活动集，$b.C$ 是基本块的条件集，$b.F$ 是基本块的流关系集，$b.A_e$ 是基本块的入口，$b.A_x$ 是基本块的出口，$inflow(a)$ 和 $outflow(a)$ 分别表示活动 a 的输入流和输出流，其中 D_B 是用于细化 a_j 的基本块 B 的关联矩阵，B 中有 $t+1$ 个行为，r 个条件；P 中 c 的个数为 n，a 的个数为 m，关联矩阵是 D_p，具体写作：

$$\begin{array}{c} \\ a_1 \\ a_2 \\ \vdots \\ a_m \end{array}\begin{array}{c} \begin{array}{cccc} c_1 & c_2 & \cdots & c_n \end{array} \\ \begin{bmatrix} d_{11} & d_{12} & \cdots & d_{1n} \\ d_{21} & d_{22} & \cdots & d_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ d_{m1} & d_{m2} & \cdots & d_{mn} \end{bmatrix} \end{array}$$

当用 b 替换掉 a 得到 P' 之后，为了便于讨论，要对 P' 内的元素重新排序。假设被替换掉的是 a_j，那么 a_j 之后的 a 的下标都减 1，b 中的 a 从 a_m 开始依次编号，b 中的 c 从 c_{n+1} 开始编号，假设接口为 c_k 和 c_{k+1}，P' 如图 6 所示。

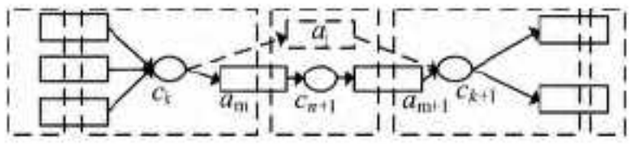

图 6　细化操作后元素的编号(以顺序块为例)

$$
\begin{array}{c}
\\
a_1\\
a_2\\
\vdots\\
a_j\\
a_{j+1}\\
\vdots\\
a_{m-1}\\
\hline
a_m\\
\vdots\\
a_{m+t}
\end{array}
\begin{array}{c}
\begin{array}{cccccccc}
c_1 & c_2 & \cdots & c_k & c_{k+1} & \cdots & c_n & c_{n+1}\cdots c_{n+r}
\end{array}\\
\left[\begin{array}{cccccccc}
d_{11} & d_{12} & \cdots & d_{1k} & d_{1(k+1)} & \cdots & d_{1n} & \\
d_{21} & d_{22} & \cdots & d_{2k} & d_{2(k+1)} & \cdots & d_{2n} & \\
\vdots & \vdots & \cdots & \vdots & \vdots & \cdots & \vdots & \\
d_{(j+1)1} & d_{(j+1)2} & \cdots & d_{(j+1)k} & d_{(j+1)(k+1)} & \cdots & d_{(j+1)n} & 0\\
d_{(j+2)1} & d_{(j+2)2} & \cdots & d_{(j+2)k} & d_{(j+2)(k+1)} & \cdots & d_{(j+2)n} & \\
\vdots & \vdots & \cdots & \vdots & \vdots & \cdots & \vdots & \\
d_{m1} & d_{m2} & \cdots & d_{mk} & d_{m(k+1)} & \cdots & d_{mn} & \\
\hline
 & & & -1 & 0 & & & \\
 & & & 0 & \vdots & & & D_B\\
 & & & \vdots & 1 & & &
\end{array}\right]
\end{array}
$$

上述矩阵为 D_p'，是 P' 的关联矩阵。

4 软件演化过程模型的结构性质

结构性质是指过程模型在结构和语法方面满足的某些限制条件，这些性质由网的结构确定，而同网的初始标识无关。

定理 6 白盒建模的软件演化过程模型具有结构有界性。

证明：假设白盒建模的软件演化过程模型是由 n 步细化操作构造成的，可以用数学归纳法证明。考虑 $k=1$ 的情况：$k=1$ 即用基本块对初始块进行细化，得到的模型的关联矩阵在 3.1 节已经给出。考虑要证明结构性质，对得到的新模型需添加一个变迁，以及两条有向弧，使新模型变成一个闭合系统，如图 7 所示。

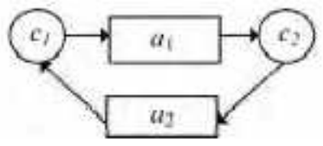

图 7 闭合系统（以顺序块为例）

用顺序块细化，其关联矩阵为：

$$\begin{bmatrix} -1 & 0 & 1 \\ 0 & 1 & -1 \\ 1 & -1 & 0 \end{bmatrix}$$

根据定理 1，存在 $y=[1\ 1\ 1]^{\mathrm{T}}$ 符合条件。

用选择块细化，其关联矩阵为：

$$\begin{bmatrix} -1 & 1 \\ -1 & 1 \\ 1 & -1 \end{bmatrix}$$

根据定理 1，存在 $y=[1\ 1]^{\mathrm{T}}$ 符合条件。

用并发块细化，其关联矩阵为：

$$\begin{bmatrix} -1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 1 \\ 0 & 1 & 0 & -1 & 0 & -1 \\ 1 & -1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

根据定理 1，存在 $y=[2\ 2\ 1\ 1\ 1\ 1]^{\mathrm{T}}$ 符合条件。

用迭代块细化，其关联矩阵为：

$$\begin{bmatrix} -1 & 0 & 1 & 0 \\ 0 & 0 & -1 & 1 \\ 0 & 0 & 1 & -1 \\ 0 & 1 & 0 & -1 \\ 1 & -1 & 0 & 0 \end{bmatrix}$$

根据定理 1，存在 $y=[1\ 1\ 1\ 1]^{\mathrm{T}}$ 符合条件。

再考虑 $k=n$ 的情况，假设 $k=n-1$ 时，命题成立。$k=n-1$ 时，软件演化过程模型的闭合系统的关联矩阵 D_{n-1} 为：

$$
\begin{array}{c}
\\
a_1\\
a_2\\
\vdots\\
a_m\\
a_{m+1}
\end{array}
\begin{array}{c}
\begin{array}{cccc}
c_1 & c_2 & \cdots & c_n
\end{array}\\
\left[\begin{array}{cccc}
d_{11} & d_{12} & \cdots & d_{1n}\\
d_{21} & d_{22} & \cdots & d_{2n}\\
\vdots & \vdots & \cdots & \vdots\\
d_{m1} & d_{m2} & \cdots & d_{mn}\\
1 & 0 & \cdots & -1
\end{array}\right]
\end{array}
\tag{1}
$$

此时存在 $\vec{y}_{n-1}=[y_1\ y_2\cdots\ y_n]^{\mathrm{T}}$，使得 $D_{n-1}\vec{y}_{n-1}\leqslant 0$。当 $k=n$，利用基本块对 $k=n-1$ 时的闭合系统内的 a_j 做细化操作，根据细化操作的定义，被替换的行为只能有一个进入弧和一个输出弧，假定这两条弧另外一端的条件分别是 c_k、c_{k+1}，那么可以得到 $k=n$ 时的闭合系统的关联矩阵 D_n，如式(2)所示。

$$
\begin{array}{c}
\\
a_1\\
a_2\\
\vdots\\
a_j\\
a_{j+1}\\
\vdots\\
a_{m-1}\\
\hline
a_m\\
\vdots\\
a_{m+t}\\
\hline
a_{m+t+1}
\end{array}
\begin{array}{c}
\begin{array}{cccccccc}
c_1 & c_2 & \cdots & c_k & c_{k+1} & \cdots & c_n & c_{n+1}\cdots c_{n+r}
\end{array}\\
\left[\begin{array}{cccccccc}
d_{11} & d_{12} & \cdots & d_{1k} & d_{1(k+1)} & \cdots & d_{1n} & \\
d_{21} & d_{22} & \cdots & d_{2k} & d_{2(k+1)} & \cdots & d_{2n} & \\
\vdots & \vdots & \cdots & \vdots & \vdots & \cdots & \vdots & \\
d_{(j+1)1} & d_{(j+1)2} & \cdots & d_{(j+1)k} & d_{(j+1)(k+1)} & \cdots & d_{(j+1)n} & 0\\
d_{(j+2)1} & d_{(j+2)2} & \cdots & d_{(j+2)k} & d_{(j+2)(k+1)} & \cdots & d_{(j+2)n} & \\
\vdots & \vdots & \cdots & \vdots & \vdots & \cdots & \vdots & \\
d_{m1} & d_{m2} & \cdots & d_{mk} & d_{m(k+1)} & \cdots & d_{mn} & \\
\hline
 & & & -1 & 0 & & & \\
 & & & 0 & \vdots & & & D_B\\
 & & & \vdots & 1 & & & \\
\hline
1 & 0 & \vdots & 0 & 0 & \vdots & -1 & 0
\end{array}\right]
\end{array}
\tag{2}
$$

其中，D_B 是细化 a_j 的基本块 B 的关联矩阵，B 中有 $t+1$ 个行为，r 个条件，矩阵的第 m_k 个元素为 -1，是因为 c_k 与 B 的入口行为 a_m 之间有一条有向弧，矩阵的第 $(m+t)(k+1)$ 个元素为 1，是因为 B 的出口行为 a_{m+t} 与 c_{k+1} 之间有一条有向弧，矩阵右上角这片区域的元素都是 0，是因为 B 内的条件不与 D_{n-1} 的行为直接相连。并且由于 a_j 被细化，根据细化操作的定义，a_j 与且只与 c_k、c_{k+1} 相连，那么 D_{n-1} 的第 j 行只有第 k 个元素为 -1、第 $k+1$ 个元素为 1，其他的元素为 0，即

$$\vec{a}_j=[\underbrace{0\ 0\cdots 0}_{k-1\uparrow 0}\ -1\ 1\ 0\ 0\cdots 0]$$

根据假设，有 $D_{n-1}\vec{y}_{n-1}\leqslant 0$，那么 $\vec{a}_j\vec{y}_{n-1}\leqslant 0$，即 $y_{k+1}\leqslant y_k$，于是只需 $y_{k+1}=y_k=2$ 即可。

下面考虑求解 $\vec{y}_n$，使得 $D_n\vec{y}_n\leqslant 0$。

首先考虑 D_n 的前 $m-1$ 行、第 $m+t+1$ 行与删除掉第 j 行的 D_{n-1} 的 $m-1$ 行完全一样，$n+r$ 维向量 y_n 的前 n 个元素与 y_{n-1} 内的元素一样。

再考虑 y_n 的后 r 个元素：

考虑 B 是顺序块，只需 $y_{n+1}=y_k$，$y_{n+2}=y_{k+1}$ 即可，即

$$y_n=[y_1\ y_2\ \cdots\ y_{k-1}\ 2\ 2\ y_{k+1}\ \cdots\ y_n\ 2\ 2]^{\mathrm{T}}$$

考虑 B 是并发块，只需 $y_{n+1}=y_{n+2}=y_{n+3}=y_{n+4}=1$，即

$$y_n=[y_1\ y_2\ \cdots\ y_{k-1}\ 2\ 2\ y_{k+1}\ \cdots\ y_n\ 1\ 1\ 1\ 1]^{\mathrm{T}}$$

考虑 B 是迭代块，只需 $y_{n+1}=y_k$，$y_{n+2}=y_{k+1}$，$y_{n+1}=y_{n+2}$，即

$$y_n=[y_1\ y_2\ \cdots\ y_{k-1}\ 2\ 2\ y_{k+1}\ \cdots\ y_n\ 2\ 2]^{\mathrm{T}}$$

由于选择块内并没有条件，因此利用选择块进行细化得到的关联矩阵 D_n，如式(3)所示。

$$
\begin{array}{c}
\\
a_1\\
a_2\\
\vdots\\
a_j\\
a_{j+1}\\
\vdots\\
a_{m-1}\\
\hline
a_m\\
a_{m+1}\\
\hline
a_{m+2}
\end{array}
\begin{array}{c}
\begin{array}{cccccccc}
c_1 & c_2 & \cdots & c_k & c_{k+1} & \cdots & c_n
\end{array}\\
\left[\begin{array}{cccccccc}
d_{11} & d_{12} & \cdots & d_{1k} & d_{1(k+1)} & \cdots & d_{1n}\\
d_{21} & d_{22} & \cdots & d_{2k} & d_{2(k+1)} & \cdots & d_{2n}\\
\vdots & \vdots & \cdots & \vdots & \vdots & \cdots & \vdots\\
d_{(j+1)1} & d_{(j+1)2} & \cdots & d_{(j+1)k} & d_{(j+1)(k+1)} & \cdots & d_{(j+1)n}\\
d_{(j+2)1} & d_{(j+2)2} & \cdots & d_{(j+2)k} & d_{(j+2)(k+1)} & \cdots & d_{(j+2)n}\\
\vdots & \vdots & \cdots & \vdots & \vdots & \cdots & \vdots\\
d_{m1} & d_{m2} & \cdots & d_{mk} & d_{m(k+1)} & \cdots & d_{mn}\\
\hline
 & & & -1 & 1 & & \\
 & 0 & & & & & 0\\
 & & & -1 & 1 & & \\
\hline
1 & 0 & \cdots & 0 & 0 & \cdots & -1
\end{array}\right]
\end{array}
\tag{3}
$$

选择块中的两个行为 a_m、a_{m+1} 都继承了 a_j 与条件之间的关系，即 D_n 的第 m、$m+1$ 行与 D_{n-1} 的第 j 行一样，即

$$\vec{a}_m=\vec{a}_{m+1}=[\underbrace{0\ 0\ \cdots\ 0}_{k-1\uparrow 0}\ -1\ 1\ 0\ 0\ \cdots\ 0]$$

那么显然 $\vec{y}_n=\vec{y}_{n-1}$ 即可使命题成立。

综上，命题成立。

证毕。

定理 7　白盒建模的软件演化过程模型具有结构守恒性。

证明：在证明定理 6 时，给出的 y 都可以使得 $Dy=0$。于是，命题成立。

证毕。

定理 8　白盒建模的软件演化过程模型具有结构重复性。

证明：假设白盒建模的软件演化过程模型是由 n 步细化操作构造成的，用数学归纳法证明命题。考虑 $k=1$ 的情况：$k=1$ 即用基本块对初始块进行细化，得到的模型的关联矩阵在 3.1 节已经给出。考虑要证明结构性质，对得到的新模型添加一个条件，以及两条有向弧，使新模型变成一个闭合系统。其关联矩阵已在证明定理 6 时给出。如果用顺序块进行细化：根据定理 3，存在 $x=[1\ 1\ 1]^{\mathrm{T}}$ 符合条件。

用选择块细化：根据定理 3，存在 $x=[1\ 1\ 2]^{\mathrm{T}}$ 符合条件。

用并发块细化：根据定理 3，存在 $x=[1\ 1\ 1\ 1\ 1]^{\mathrm{T}}$ 符合条件。

用迭代块细化：根据定理 3，存在 $x=[1\ 2\ 1\ 1\ 1]^{\mathrm{T}}$ 符合条件。

再考虑 $k=n$ 的情况，假设 $k=n-1$ 时，命题成立。$k=n-1$ 时，软件演化过程模型的闭合系统的关联矩阵 D_{n-1}，$\mathrm{D}_{n-1}^{\mathrm{T}}$ 如式(4)所示。

$$\begin{array}{c} \\ c_1 \\ c_2 \\ \vdots \\ c_n \end{array}\begin{array}{c} \begin{array}{ccccc} a_1 & a_2 & \cdots & a_m & a_{m+1} \end{array} \\ \begin{bmatrix} d_{11} & d_{12} & \cdots & d_{m1} & 1 \\ d_{12} & d_{22} & \cdots & d_{m2} & 0 \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ d_{1n} & d_{2n} & \cdots & d_{mn} & -1 \end{bmatrix} \end{array} \tag{4}$$

此时存在

$$\vec{x}_{n-1}=[x_1\ x_2\ \cdots\ x_k\ x_{k+1}\ \cdots\ x_m\ x_{m+1}]^{\mathrm{T}}$$

使得 $D_{n-1}^{\mathrm{T}}\vec{x}_{n-1}\geqslant 0$。

当 $k=n$，关联矩阵已在证明定理 6 时给出，那么 D_n^{T} 如式(5)、式(6)所示。其中式(5)的基本块可以是顺序块、并发块或者迭代块。式(6)的基本块是选择块。

$$\begin{array}{c} \\ c_1 \\ c_2 \\ \vdots \\ c_k \\ c_{k+1} \\ \vdots \\ c_n \\ \hline c_{n+1} \\ \vdots \\ c_{n+r} \end{array}\begin{array}{c} \begin{array}{ccccccccccc} a_1 & a_2 & \cdots & a_j & a_{j+1} & \cdots & a_{m-1} & a_m & \cdots & a_{m+t} & a_{m+t+1} \end{array} \\ \left[\begin{array}{ccccccc|ccc|c} d_{11} & d_{21} & \cdots & d_{(j+1)1} & d_{(j+2)1} & \cdots & d_{m1} & & & & 1 \\ d_{12} & d_{22} & \cdots & d_{(j+1)2} & d_{(j+2)2} & \cdots & d_{m2} & & & & 0 \\ \vdots & \vdots & \cdots & \vdots & \vdots & \cdots & \vdots & & & & \vdots \\ d_{1k} & d_{2k} & \cdots & d_{(j+1)k} & d_{(j+2)k} & \cdots & d_{mk} & -1 & 0 & 0 & 0 \\ d_{1(k+1)} & d_{2(k+1)} & \cdots & d_{(j+1)(k+1)} & d_{(j+2)(k+1)} & \cdots & d_{m(k+1)} & 0 & \cdots & 1 & 0 \\ \vdots & \vdots & \cdots & \vdots & \vdots & \cdots & \vdots & & & & \vdots \\ d_{1n} & d_{2n} & \cdots & d_{(j+1)n} & d_{(j+2)n} & \cdots & d_{mn} & & & & -1 \\ \hline & & & & 0 & & & & D_B^{\mathrm{T}} & & 0 \end{array}\right] \end{array} \tag{5}$$

$$\begin{array}{c} \\ c_1 \\ c_2 \\ \vdots \\ c_k \\ c_{k+1} \\ \vdots \\ c_n \end{array}\begin{array}{c} \begin{array}{ccccccccc} a_1 & a_2 & \cdots & a_j & a_{j+1} & \cdots & a_{m-1} & a_m\ a_{m+1} & a_{m+2} \end{array} \\ \left[\begin{array}{ccccccc|cc|c} d_{11} & d_{21} & \cdots & d_{(j+1)1} & d_{(j+2)1} & \cdots & d_{m1} & & & 1 \\ d_{12} & d_{22} & \cdots & d_{(j+1)2} & d_{(j+2)2} & \cdots & d_{m2} & 0 & & 0 \\ \vdots & \vdots & \cdots & \vdots & \vdots & \cdots & \vdots & & & \vdots \\ d_{1k} & d_{2k} & \cdots & d_{(j+1)k} & d_{(j+2)k} & \cdots & d_{mk} & -1 & -1 & 0 \\ d_{1(k+1)} & d_{2(k+1)} & \cdots & d_{(j+1)(k+1)} & d_{(j+2)(k+1)} & \cdots & d_{m(k+1)} & 1 & 1 & 0 \\ \vdots & \vdots & \cdots & \vdots & \vdots & \cdots & \vdots & & & \vdots \\ d_{1n} & d_{2n} & \cdots & d_{(j+1)n} & d_{(j+2)n} & \cdots & d_{mn} & 0 & & -1 \end{array}\right] \end{array} \tag{6}$$

下面考虑求解 $\vec{x}_n$，使得 $D_n^{\mathrm{T}}\vec{x}_n\geqslant 0$。先考虑式(5)的情况，在证明定理 6 时，已经说明过

$$\vec{a}_j=[\underbrace{0\ 0\ \cdots\ 0}_{k-1\uparrow 0}\ -1\ 1\ 0\ 0\ \cdots\ 0]$$

由于 $D_{n-1}^{\mathrm{T}}\vec{x}_{n-1}\geqslant 0$，那么对于 D_n^{T} 的前 n 行中除了第 k 行、第 $k+1$ 行都有 $D_n^{\mathrm{T}}\cdot\vec{c}_l\vec{x}_n\geqslant 0$，$l=1,2,\cdots,k-1,k+2,\cdots,n$ 只需令 $m+t$ 维向量

$$\vec{x}_n=[x_1\ x_2\ \cdots\ x_{m-1}\ x_m\ x_{m+2}\ \cdots\ x_{m+t}\ x_{m+1}]^{\mathrm{T}}$$

要使得 D_n^{T} 中的第 k 行、第 $k+1$ 行都有 $D_n^{\mathrm{T}}\cdot\vec{c}_l\vec{x}_n\geqslant 0$，$l=k,k+1$，那么只需令 $x_m=x_{m+t}=x_k$ 即可。

对于 B 是顺序块，只需

$$\vec{x}_n=[x_1\ x_2\ \cdots\ x_{m-1}\ x_k\ x_k\ x_{m+1}]^{\mathrm{T}};$$

考虑 B 是并发块，只需

$$\vec{x}_n=[x_1\ x_2\ \cdots\ x_{m-1}\ x_k\ x_k\ x_k\ x_k\ x_{m+1}]^{\mathrm{T}}\ \text{即可};$$

考虑 B 是迭代块，只需

$$\vec{x}_n=[x_1\ x_2\ \cdots\ x_{m-1}\ x_k\ 2x_k\ x_k\ x_k\ x_{m+1}]^{\mathrm{T}}$$

由于选择块内并没有条件，因此利用选择块细化得到的关联矩阵转置 D_n^{T} 如式(6)所示。那么只需 $\vec{x}_n=[2x_1\ 2x_2\cdots 2x_{m-1}\ x_k\ x_k\ 2x_{m+1}]^{\mathrm{T}}$ 即可使命题成立。

综上，命题成立。

证毕。

定理 9　白盒建模的软件演化过程模型具有结构守恒性。

证明：在证明定理 8 时，给出的 X 都可以使得 $DX=0$。于是命题成立。

证毕。

定理 10　白盒建模的软件演化过程模型不包含死锁和陷阱。

证明：用数学归纳法来证明，当 $k=1$ 时，通过观察定理 1 内的关联矩阵，可知 $k=1$ 时，命题成立。

考虑 $k=n$ 的情况。假设 $k=n-1$ 时命题成立，此时的关联矩阵如式(1)所示。对其进行细化操作可得新的闭合系统，其关联矩阵如式(2)所示。

由于 $k=n-1$ 时命题成立，可知，任意一个 D_{n-1} 的最大列生成子阵都不符合含有死锁和陷阱的条件，更进一步 D_{n-1} 的任意一个列生成子阵都不符合含有死锁和陷阱的条件。而 $k=n$ 时，D_n 的右上角部分是零矩阵，不含有 1 或者 -1，因此任意一个 D_n 的最大列生成子阵都不符合含有死锁和陷阱的条件，更进一步 D_n 的任意一个列生成子阵都不符合含有死锁和陷阱的条件。

综上，命题得证。

证毕。

综合上面定理的证明，可以看出白盒建模的软件演化过程模型自身就具备着非常优良的结构性质，无须再对其进行结构验证。

结束语　本文利用关联矩阵对通过白盒建模方法建立的、基于 EPMM 的软件演化过程模型进行描述，对模型中的结构进行了验证，证明了其具备结构有界性、可重复性和守恒性等优良的结构性质，实现了对其结构性质的验证。采用关联矩阵方法表示出 EPMM 的基本块，包括初始块、顺序块、选择块、并发块、迭代块，进而得出通过各个模块细化后得到的

关联矩阵表达形式，通过数学方法证明通过白盒建模细化得到的软件演化过程模型自身就具备着非常优良的结构性质，无须再对其进行结构验证。

下一步的工作是对通过黑盒建模方法得到的软件演化过程模型进行研究，并在本文的数学基础上，更进一步地提升，从而能够实现对其模型的结构性质证明。

参 考 文 献

[1] 杨芙清. 软件工程技术发展思索[J]. 软件学报，2005，16(1)：1-7

[2] Lehman M M，Peryy D E，Ramil J F. Metrics and Laws of Software Evolution-the Nineties Views[C]// Proceeding of 4th International Symposium on Software Metrics. IEEE Computer Society Press，2000：20-32

[3] Li T. An Approach to Modelling Software Evolution Processes [M]. Springer-Verylage Berlin and Heidelberg GmbH & Co. K，2008

[4] Brauer W，Gold R，Vogler W. A survey of behavior and equivalence preserving refinement of Petri nets [J]. Lecture Notes in Computer Science. 1990，483：1-46

[5] Suzuki I，Murata T. A method for stepwise refinement and abstraction of Petri nets [J]. J. Comput. System Sci.，1983，27：51-76

[6] Valette R. Analysis of Petri nets by stepwise refinements [J]. J. Comput. System Sci.，1979，18：35-46

[7] Betous-Almeida C，Kanoun K. Construction and stepwise refinement of dependability models [J]. Performance Evaluation，2004，56：277-306

[8] Nketsa A，Valette R. Rapid and modular prototyping-based Petri nets and distributed simulation for manufacturing systems [J]. Applid Mathematics and Computation，2001，120：265-278

[9] van Hee K，Sidorova N，et al. Soundness and separability of workflow nets in the stepwise refinement approach [C]// Proc the 24th International Conference on Application and Theory of Petri Nets. Eindhoven，The Netherlands，2003，2679：337-356

[10] 吴哲辉. Petri 网导论[M]. 机械工业出版社，2006

(上接第 499 页)

最后介绍了常见的网格聚类算法，其中典型的算法有 STING、Wave Cluster、CLIQUE 等，并对这些算法进行了分析。在此基础上介绍了常见的处理多密度数据集的算法，其中包括 Chameleon、共享近邻 SNN 算法、多阶段等密度算法等，并对这些算法进行了分析。

结束语　基于网格聚类方法的优点是：它的处理速度快，因为其速度与数据对象的个数无关，而只依赖于数据空间中每个维的单元的个数；能发现任意形状和大小的簇；计算结果与数据输入顺序无关；计算时间与数据量无关；同时不要求向 k 均值意义预先指定簇个数等。但是，基于网格方法的聚类算法的输入参数对聚类结果影响较大，而且这些参数较难设置；当数据中有噪音时，如果不加特殊处理，算法的聚类质量会很差；而且，算法对于数据维度的可伸缩性较差。

基于网格的聚类方法目前还存在一些亟需解决的问题，主要有以下几点：(1)当簇具有不同的密度时，全局的密度参数不能有效发现这样的簇，需要开发具有可变密度参数的算法；(2)对于不同类型数据的聚类问题，比如对于高维数据，网格的数据将急剧增加，需要有效的技术发现近邻单元；(3)当数据集的规模巨大以及数据具有地理分布特性时，需要开发有效的并行技术算法来提高处理的速度；(4)对现有网格算法的优化，从不同方面提高网格算法的有效性，比如开发稀疏网格的压缩算法和密度相似网格的合并算法等。

本文对基于网格的聚类方法的已有研究进行了分析和总结，包括网格的定义与划分方法、网格单元密度的确定、由邻接网格单元形成聚簇的聚类过程；最后对网格聚类方法与局限性进行总结，在已有研究分析的基础上，提出后续需要重点解决的问题。

参 考 文 献

[1] 马刚，李志刚. 数据仓库与数据挖掘的原理及应用[M]. 北京：高等教育出版社，2012：20-42

[2] 陈志泊. 数据仓库与数据挖掘[M]. 北京：清华大学出版社，2011：8-37

[3] Tan Pang-ning，Steinbach M，Kumar V. 数据挖掘导论[M]. 范明，译. 北京：人民邮电出版社，2013：6-53

[4] Dunham M H. DATA MINING Introductory and Advanced Topics [M]. 北京：清华大学出版社，2010：23-60

[5] Ng R T，Han J. Efficient and effective clustering methods for spatial data mining[C]// Proc of the 20th VLDB Conference. Chile，Santia，2010：144-155

[6] Spivak G. Victory in Limbo：Imagism [C]. Nelson C，Grasberg L，eds. Urbana：University of Illinois Press，2010：271-313

[7] Zhang T，Rrmakrishnan R，Livny M. An efficient data clustering method for very large databases[C]// Proc of ACM SIGMOD International Conference on Management of Data. New York：ACM Press，2012：103-114

[8] Tan Pang-ning，Steinbach M. Introduction to Data Mining[M]. 2010：372-373

[9] Chen Y，Tu L. Density-Based Clustering for Real-Time Stream Data[C]// Proceedings of the 13th ACM SIGKDD International Conference of Knowledge Discovery and Data Mining. San Jose，California，USA，2009：133-142

[10] 曹洪其，余岚，孙志辉. 基于网格聚类技术的离群点挖掘算法[J]. 计算机工程，2006(11)：18-96

[11] 孙玉芬. 基于网格方法的聚类算法研究[D]. 武汉：华中科技大学，2011

[12] Han J，Kamber M. Data Mining：Concepts and Techniques [J]. Morgan Kaufmann Publishers，2011，2(9)：33-82

[13] Chen Ming-yan，Han Jia-wei，Philip S Y. Data mining：an overview from a database perspective [J]. IEEE Trans on Knowledge and Data Eng.，1996，8(6)：806-833