软件开发中的活动模式挖掘

吉才盈[1] 代 飞[1,2] 李 彤[1,2] 蒋旭东[1]
(云南大学软件学院 昆明 650091)[1] (云南省软件工程重点实验室 昆明 650091)[2]

摘 要 如何获取实际运作的软件过程(而非经验过程)是一个关键问题。软件过程的事件数据呈现记录多、无明确活动与独具过程特性的特点,使得现有的业务过程挖掘方法难以被有效应用。首先获取事件日志的原子活动;然后采用折叠的方法从软件项目开发实例中挖掘活动模式,获取软件过程,对不同时期的软件开发行为进行分析;最后通过一个开源项目的实际案例来验证该方法。

关键词 过程挖掘,原子活动,活动模式,折叠,软件过程
中图法分类号 TP391 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2016.6.034

Activity Pattern Mining in Software Development

JI Cai-ying[1] DAI Fei[1,2] LI Tong[1,2] JIANG Xu-dong[1]
(School of Software, Yunnan University, Kunming 650091, China)[1]
(Key Laboratory of Software Engineering of Yunnan Province, Kunming 650091, China)[2]

Abstract It is a critical issue to mine the real enacted software process (not experiential process). Due to the characteristics of software process event data, i.e., many records, no specific activity and its unique features, the existing business process mining methods can't support them effectively. Firstly, the atomic activities of event log were gained. Then, activity patterns were mined from the software project development instances by folding method, the software process was got, and the software development activities in different periods were analyzed. Finally, this method was verfied through a case of open source projects.

Keywords Process mining, Atomic activity, Activity pattern, Folding, Software process

1 引言

软件过程对软件项目的开发具有重要意义,从软件项目产生的数据中挖掘软件过程成为既经济又可靠的方法[1]。在大数据时代的今天,软件开发中的事件数据[2,3]不断增长[4,5]。这些数据中蕴含着知识,而这些知识对发现、理解和改善软件过程至关重要[6,7]。进一步来说,过程挖掘[1]对软件开发有如下好处:1)从事件数据中挖掘实际运行的软件过程,而不只是利用依靠经验提前制定的软件过程模型来指导工作[8];2)挖掘的软件过程可以反映工程参与者与软件组织的工作特点;3)证明软件按照预期的方式设计、开发并验证(确认),出现的偏差能很快识别与纠正;4)随着软件开发的进行与事件数据的增加,不断地对软件过程进行改进[9]。

目前,在软件开发中,这些数据大量存在,却难以产生真正的价值,原因在于数据类型多、数据量大和数据难以被直接使用,从而导致企业不重视这些数据。而软件项目开发的事件数据具有以下特点:1)完整的软件项目历时周期长,发布版本众多,事件记录繁多;2)在项目日志中记录的是各种文件和划分功能的文件夹,而不是传统意义上的活动;3)软件具有可拷贝性,同一项目一个公司只做一次,每个项目都会因类别不同而具有其过程特点,因而不能直接使用它们的事件日志来提取同一个软件过程。如何从这些数据中挖掘有用的知识并指导软件开发,是亟待解决的问题。

针对上述问题,传统的数据挖掘和过程挖掘技术难以被直接应用。数据挖掘的目标是关联和规则,而我们挖掘的目标在于开发行为。现产生了许多针对业务过程挖掘的算法,如强调活动的原子性和瞬时完成性的 alpha 算法[10]、用于消除日志噪音影响的启发式算法[11],以及用聚类的理念设计的 fuzzy 算法[12]等,这些算法都是以活动与实例(instance)为基础设计的。然而因业务流程与软件项目的事件数据结构不一致,这些过程挖掘算法对软件过程的挖掘并不完全适用。

本文提出挖掘软件开发中的活动模式(重复出现的开发任务),以更好地理解各个阶段的软件开发行为,让企业与组织更好地分析各个开发项目之间的关系,以制定和不断地修改软件过程模型,提升软件企业 CMMI 等级[13]。使用折叠的方法挖掘软件过程活动模式。以具体项目的版本控制系统

到稿日期:2015-05-19 返修日期:2015-09-08 本文受国家自然科学基金资助项目(61262024,61379032,61462095),云南省自然科学基金资助项目(2012FD005),云南省教育厅科学研究基金项目(2013Y365),云南省软件工程重点实验室开放基金项目(2012SE307),第五批云南大学"中青年骨干教师培养计划"专项经费(XT412003)资助。

吉才盈 硕士生,主要研究方向为软件过程挖掘;代 飞 博士,副教授,主要研究方向为软件过程、业务过程;李 彤 博士,教授,主要研究方向为软件过程、软件演化;蒋旭东 硕士生,主要研究方向为软件动态演化。

Subversion 中产生的事件日志的格式类型为研究对象，对项目路径进行归类，得到关注的活动类别，识别出事件日志的原子活动。针对软件项目中出现的反复调试、修改与完善的行为，引入串联重复序列的概念和使用编辑距离[14]分析具有一定相似程度的活动片段，以获取可折叠活动片段。对可折叠片段进行分析与折叠，获得具有多种控制结构(顺序、选择、并行)的活动模式，从而达到从只有一个实例的项目日志中获取活动模式的目的。最后在与客户协商的基础上，对模式进行调整，使得模式尽可能简洁易懂，并且降低行为丢失的可能性。

本文第 2 节介绍相关的工作；第 3 节针对具体的项目进行原子活动识别与提取；第 4 节描述活动模式的挖掘过程，并通过模式调整进而简化过程模型；第 5 节以一个开源项目 ArgoUML 作为实际案例，阐明了本方法能够较好地应用于软件项目中，获得包含活动模式的过程模型；最后总结全文。

2 相关工作

本文的主要工作是提出软件过程活动模式的挖掘方法，因此本节将对软件过程挖掘与活动模式的相关领域工作进行简要介绍。

在进行过程挖掘之前，需要对数据进行处理。对软件过程来说，首先需要识别活动。文献[8]认为活动代表工作执行的单元，按照文档或软件制品的输入、输出来定义。在事件日志中没有记录“输入”，只记录了“输出”，则把先前发生的事件作为输入，使简单的工作变得复杂。文献[15]为了简化软件过程活动的定义，活动和事件的比例为 1:1，把文档名映射为抽象的活动名称，以获取原子活动。接下来从识别的活动中挖掘软件过程。Rubin 等人[15]从 CVS 版本控制日志中挖掘软件过程，使用 ArgoUML 开源项目进行软件过程的挖掘。从不同的子项目中获取多个实例，采用业务流程的挖掘方法获取软件过程。然而该方法只挖掘了软件项目初期的开发过程，若项目后期还使用同一种方法，则会使得过程模型复杂度过高而难以理解。文献[13]通过计算软件过程的马尔科夫链模型来确定功能点之间的关系，却忽略了项目在不同的开发时期的开发行为并不一致。

业务流程模型的复杂性导致其难以被理解，有多种方式可简化过程模型，在分析事件日志时，对频繁出现的模式加以替换，可以达到该目标。文献[16]采用后缀树在线性时间内查找串联重复序列[17]，以识别业务过程模式。而 Li J. 等人[18]对识别的模式进行缩放，以便查看全局的业务过程模型和局部的具体模式。还有一种简化过程模型的方法是对过程分支进行折叠[19]，本文将采用此方式来获取活动模式。

3 原子活动识别

本节讨论如何从软件项目的事件轨迹中提取原子活动。原子活动是基于事件的最初划分。在业务流程中，一个事件产生一个活动。然而在软件开发过程中，版本控制系统产生的事件日志提取的事件信息并不直接具有活动的特性，事件记录只有结束时间(提交时间)，没有开始时间；只有确切的项目路径，没有活动名称。因此，需要进行最初的原子活动识别。本节提出一种合并连续相同项的方法，使得活动既有开始时间，又有结束时间，以此来获取原子活动。以 ArgoUML 项目结构为例，从其日志文件中提取原子活动。活动类别的抽象程度依赖于客户的需要和项目的具体结构。

为了获取活动，把确切的项目路径映射为抽象的名称，称之为分类。文件的分类要根据客户的需要，而本文将以一般关注的几个软件项目的活动类型为研究，分别是编码、测试、网页、构建、库与配置。src 文件夹中的认为是编码，tests 文件夹中的认为是测试，在 www 文件夹中的则是网页，build 文件夹或含有 build 的文件名中的则是构建，在文件 Lib 中的则是库文件，以“·”开头的文件是配置。截取 ArgoUML 过程实例的一段事件序列，为了表示各个事件虽然名称一样却不是同一事件，对其编号，见表 1。

表 1 事件日志

文件名	作者	提交时间	抽象名称
cppProject/trunk/www/index.html	euluis	2006-04-21	www1
cppProject/trunk/www/core.html	euluis	2006-04-22	www2
cppProject/trunk/www/index.html	euluis	2006-04-23	www3
cppProject /trunk/build.xml	tfmorris	2006-04-25	build1
cppProject/trunk/src/org/manifest.mf	linus	2006-04-26	src1
cppProject/trunk/src/org/venge.cpp	linus	2006-04-27	src2
cppProject/trunk/tests/SimpleClass.cpp	euluis	2006-04-28	test1
cppProject/trunk/tests/TestSimple.cpp	euluis	2006-05-03	test2
cppProject/trunk/src/org/simple.cpp	linus	2006-05-04	src3
cppProject/trunk/src/org/venge.cpp	linus	2006-05-05	src4
cppProject/trunk/.classpath	tfmorris	2006-10-25	conf1

图 1 为事件与活动的映射方式，合并连续发生的事件。用最初事件的提交时间作为活动的开始时间最后的事件时间作为结束时间。有的事件并不连续出现，则该事件的发生时间既作为活动的开始时间又作为活动的结束时间，表示该活动是瞬时完成的，得原子活动表 2。

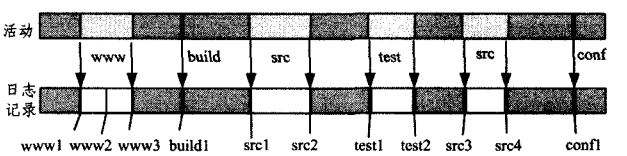

图 1 事件与活动映射

表 2 原子活动

活动名称	开始时间	结束时间
www	2006-04-21	2006-04-23
build	2006-04-25	2006-04-25
src	2006-04-26	2006-04-27
test	2006-04-28	2006-05-03
src	2006-05-04	2006-05-05
conf	2006-10-25	2006-10-25

4 挖掘软件开发中的活动模式

本文将采用折叠的方式对串联重复序列进行折叠，得到

最简序列，并分析能够提取顺序、并行与选择结构的序列特征，以此来获取活动模式。而文中所提的软件过程活动模式的挖掘基于一个项目，即只有一个实例。为了挖掘出活动模式，需要对实例序列进行分析，引入以下概念。

1)串联重复序列 TR：用三元组(i,α,k)来表示串联序列，i 为该串联重复序列在轨迹 T 的起始位置，α 为被重复的序列，k 为重复的次数。

2)最大串联序列 TM：假设轨迹 T 中存在串联序列 $T(i,j)$，i 表示开始位置，j 表示结束位置，该串联序列可以用 α^k $(k\geqslant 2)$来表示，表示 α 序列重复了 k 次。在最大串联序列的串联序列 $T(i,j)$的左右两边进行搜索，无序列 α 的存在。

3)最短重复单元 α：上述 α 为原始串联重复类型，并且 α 本身不是串联序列。

4)编辑距离函数(Levenshtein Distance，LD)：用于对比两个序列，计算把一个序列转换成另外一个序列所需要的操作数。

5)近似串联序列：两个编辑距离 $LD\leqslant M(M>0)$的序列。

接下来将对以下事件轨迹进行分析：实例 L=oabcrbcrb-crbcrmdehdhebcrmdehabcrmdhecrmbcuhbcghs，为了方便查找，表 3 对事件进行编号。

表 3　事件轨迹

0	1	2	3	4	5	6	7	8	9	10
o	a	b	c	r	b	c	r	b	c	r
11	12	13	14	15	16	17	18	19	20	21
b	c	r	m	d	e	h	d	h	e	b
22	23	24	25	26	27	28	29	30	31	32
c	r	m	d	e	h	a	b	c	r	m
33	34	35	36	37	38	39	40	41	42	43
d	h	e	c	r	m	b	c	u	h	b
44	45	46	47							
c	g	h	s							

4.1　活动模式

定义 1(可折叠片段)　可折叠片段是一个三元组 $T=(i,P,k)$，其中：

1)i 为该片段在事件轨迹中的开始编号；

2)P 表示模式，P 中的序列是不可折叠片段，是串联序列与近似串联序列的折叠结果；

3)k 为模式 P 序列的重复次数。

事件轨迹 L 的片段：T_1=oabcrbcrbcrbcrm，可以找到串联序列：(2，bcr，4)，(3，crb，3)，(2，bcrbcr，2)，(4，rbc，3)。则最大串联序列 TM 为 bcrbcrbcrbcr，最短重复单元 α 有：bcr，crb，rbc。

定义 2(模式)　模式是一个三元组 $P=(A,i,Bool)$，存入模式库 PL 中，其中：

1)A 为数组，存储相似序列时，相同序列只需存储一个；

2)i 表示该模式在片段轨迹中的位置；

3)$Bool$ 表示 P 中是否含有模式，在模式包含模式的情况下值为 T，否则为 F。

定义 3(折叠)　给定一个可折叠片段 T，用'^'标识可折叠片段的折叠部位。用五元组 $tf=(i,j,\lambda,n,FR)$来存储折叠信息，其中：

1)i 与 j 表示该片段的开始位置与结束位置；

2)$\lambda(x)$表示片段 x 的元素，如：在日志 L 中的可折叠片段 α：dehdhe，$\lambda(\alpha)=\{d,e,h\}$；

3)$|x|=n$，n 表示片段 x 元素的个数；

4)折叠率(Folding Ratio) $FR=\dfrac{1-|\text{折叠后序列}|}{|\text{最初序列}|}\times 100\%$，用来计算生成模式的效率。

定义 4(活动关系)　给定一个最终折叠的活动片段 L，用三元组 $AR=(AT,P,F)$来表示活动片段的活动关系，其中：

1)AT 为活动片段 L 中模式展开后的活动序列；

2)用中括号'[ ]'区分模式 P 中的活动与普通的活动；

3)F 表示活动之间的二元关系，有顺序关系'<'、并行关系'‖'和选择关系'#'。

使用二元关系连接每一个活动，得到活动关系序列，使用该序列画出过程模型。

4.1.1　获取活动模式的折叠规则

折叠规则 1：顺序折叠，用'<'表示(为了简化，'<'省略)。

给定两个相连片段 x_1 和 x_2，编辑距离 $LD(x_1,x_2)=0$，表示两个序列符合顺序折叠规则。

例：对日志 L 中的可折叠片段$(2,P_1,4)$进行折叠，bcr^bcr^bcr^bcr=bcr，可得模式 $P_1(A_1,2,F)$，$A_1=\{bcr\}$，将序列 bcrbcrbcrbcr 替换成 P_1。

折叠规则 2：并行折叠，用'‖'表示。

给定两个相连片段 x_1 和 x_2，$\lambda(x_1)=\lambda(x_2)$，$|x_1|=|x_2|$，$LD\leqslant M$，$M>0$($M$ 可根据日志文件进行调整，在本文中并行折叠 $M=2$)，表示两个序列符合并行折叠规则。

例：对日志 L 中的可折叠片段$(15,P_2,2)$进行折叠，deh^dhe=d(e‖h)，得模式 $P_2(A_2,15,F)$，$A_2=\{deh,dhe\}$，将序列 dehdhe 替换成 P_2。

折叠规则 3：选择折叠，用'#'表示。

给定两个相连片段 x_1 和 x_2，$|x_1|$ and $|x_2|\geqslant 3$，$LD\leqslant M$，$M>0$(M 可根据日志文件进行调整，在本文中选择折叠 $M=1$)，表示两个序列符合选择折叠规则。

例：对日志 L 中的可折叠片段$(39,P_3,2)$进行折叠，bcuh^bcgh=bc(u#g)h 得模式 $P_3(A_3,37,F)$，$A_2=\{bcuh,bcgh\}$，将 bcuhbcgh 替换成 P_3。

折叠规则 4：

对以下特殊情况进行规定：

(1)模式中包含模式

如：实例 L 的片段 T_2=bcrmdehdhebcrmdeha。

1)一次折叠：$P_1=\{A_1,4,F\}$，$A_1=\{deh,dhe\}$。$T_2^1=bcrmP_1bcrmdeha$

2)二次折叠：对于模式 P_1 后面的序列 deh，对比 A_1 集合中的元素，若满足折叠规则，则将该序列放入 A_1 中；若 A_1 中已存在该序列，则不用放入。

$T_2^2=P_2a$，$P_2=bcrmP_1$，$P_2=\{A_2,1,T\}$，$A_2=\{bcrmP_1,bcrmdeh,bcrmdhe\}$。

(2)模式折叠模式

对比两个模式集合 A 中的所有序列，两两对比，若满足折叠规则，则把两个模式进行合并。

4.1.2 活动模式挖掘示例

使用示例日志 L 进行挖掘：

1)一次折叠：L=o a b c r^ b c r^ b c r^ b c r m d e h^ d h e b c r m d e h a b c r m d h e c r m b c u h^ b c g h s

通过折叠可折叠片段：$(2,P_1,4),(15,P_2,2),(39,P_3,2)$

$\Rightarrow L^1$=o a P_1 m P_2 b c r d e h a b c r m d h e c r m P_3 s

$P_1=\{A_1,2,F\}$，$A_1=\{bcr\}$；$P_2=\{A_2,15,F\}$，$A_2=\{dhe,deh\}$；$P_3=\{A_3,39,F\}$，$A_3=\{bcuh,bcgh\}$。

2)二次折叠：L=o a P_1 m P_2^ b c r m d e h a b c r m d h e c r m b P_3 s

通过折叠可折叠片段：$(2,P_4,2)$

$\Rightarrow L^2$=o a P_4 a b c r m d h e c r m P_3 s

$P_4=\{A_4,2,T\}$，$A_4=\{P_1mP_2,bcrmdeh,bcrmdhe\}$。

3)三次折叠：L=o a P_4^ a b c r m d h e c r m P_3 s

通过折叠可折叠片段：$(2,10,a)$

$\Rightarrow L^3$=o P_5 c r m P_3 s

$P_5=\{A_5,1,T\}$，$A_5=\{aP_4,aP_1mP_2,abcrmdhe,abcrmdeh\}$。

从最终序列 oP_5crmP_3s 中遍历模式库，如表 6 所列；生成最终模式，与现有片段进行组合，得活动关系序列 o<[a<[[b<c<r]<m<[d<(e ‖ h)]]]<c<r<m<[b<c<(u # g)<h]s，画出过程模型，如图 2 所示，折叠率 FR=61.7%。

表 6 模式库

P_1{{bcr},2,F}	P_2={dhe,deh},15,F}
P_3={{bcuh,bcgh},37,F}	
P_4={{P_1mP_2,bcrmdeh,bcrmdhe},2,T}	
P_5={{aP_4,aP_1mP_2,abcrmdhe,abcrmdeh},1,T}	

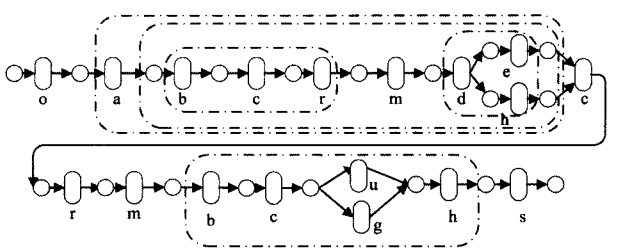

图 2 示例过程模型

4.2 模式调整

模式之间的关系可以分为包含关系和后继关系，模式经过模式调整可以最大程度折叠普通片段或其他模式。模式的调整需根据客户的意愿进行。

模式调整方式：

1)模式内存储最大串联序列中的最短重复单元，可以通过寻找所有适合的最短重复单元来替换模式。

例：对于 4.1.2 节中最后得到的过程模型，模式 $P_4=P_1mP_2$，调整模式得 $P_4'=mP_2P_1$。则可以合并序列 cr，产生一个新模式 P_6={{cr},7,F}，使得模式更简化，折叠率由原来的61.7%变为 66%。

2)模式(外模式)中包含一个模式(内模式)，无论内模式处于外模式的前端或后端，都可以调整为其反方向，即循环旋转。若模式中含包 $n(n>1)$个模式，也可使用该方法进行调整。如在活动挖掘示例中的模式 $P_5=aP_4$，调整为 $P_5'=P_4a$。

3)其他客户认为合理的调整方式。

4.3 算法

算法 1 展示了把事件日志转换成过程模型的活动关系序列的过程。处理事件日志，得到原子活动序列。查找活动序列的可折叠片段，可折叠片段进行折叠后可能形成新的可折叠片段，需要循环查找。模式集合 PL 存储所有识别的模式。对活动片段迭代处理 $i+1$ 次后，得到迭代 i 次的最终含有模式名称的活动片段，对该活动片段进行处理，识别活动间的二元关系。

算法 1 获取最终模式并组成过程模型

```
输入：事件日志 L
输出：活动关系序列 M
1. begin
2. A=g(L),L'=A,k=0
3. while f(L')!=null do
4.   FL=f(L')
5.   for i=1 to |FL| do
6.       k++
7.       Set pattern Pk=FL[i]
8.       append Pk to PL//把 Pk 添加到 PL 中
9.       L'=把识别的序列替换成 Pk
10. end for
11. end while
12. M=用'<'连接 L'中的每个活动
13. while M 存在模式 p do
14.   M=在模式前后插入'['与']',从 PL 中查找该模式
15.   if p.Bool=false //模式内不包含模式
16.     if |p.A|>1 //模式中序列超过 1 个
17.       M=用'<'、'‖'、'#'标识活动之间的关系
18.     else
19.       M=用'<'标识模式内的活动关系
20.     end if
21.   else //模式内包含模式
22.     选取模式中最短的序列,M=用'<'标识模式内的活动关系
23.   end if
24. end while
25. return M
```

设子功能 $g:L->A$ 为把日志转换成活动序列。子功能 $f:A->PTR[]$为一次扫描从活动序列中寻找全部的模式 PTR，用 FL 链来存储。用 k 对所有模式进行编号并存储在模式集合 PL 中。L'存储折叠后形成的新活动序列。最后，对 L'与 PL 进行处理即可得活动关系序列。

5 案例分析

案例分析采用一个实际的开源软件 ArgoUML。ArgoUML 使用 Subversion 作为其版本控制管理的软件。选取 cpp 子项目提取日志文件进行分析，项目时间为 2005 年 11 月 30 日至 2014 年 8 月 31 日。该日志文件中有 996 条记录，识别出 352 个原子活动，将编码、测试、配置、构建、库和网页作为重点考虑的范围，过滤后剩下 251 个活动，将其作为一个实例，挖掘出包含模式的软件过程模型。

从图 3 可看出，挖掘出来的包含活动模式的过程模型只剩 60 个活动，看起来简洁明了，从而可以对在不同项目时期

团队合作的项目开发行为进行分析。挖掘出的过程模型包含了35个大大小小的模式，模式与模式折叠导致模式合并，最后剩下17个模式。表4列出了模式库中统计的出现次数最多的模式，出现最多的模式是测试、编码与构建，而被折叠次数最多的是测试和编码，该模式还频繁参与构建其他模式。分析不同项目阶段的活动模式发现，过程模型的前端和后端都使用了同一个模式 www<(test ‖ src)<build<(test ‖ src)<conf，构建网页之后，编码与测试并发执行，再进行构建活动，编码与测试并发执行，最后进行配置活动。经过分析不同阶段的活动模式，可找出项目在不同时期行为信息的异同。

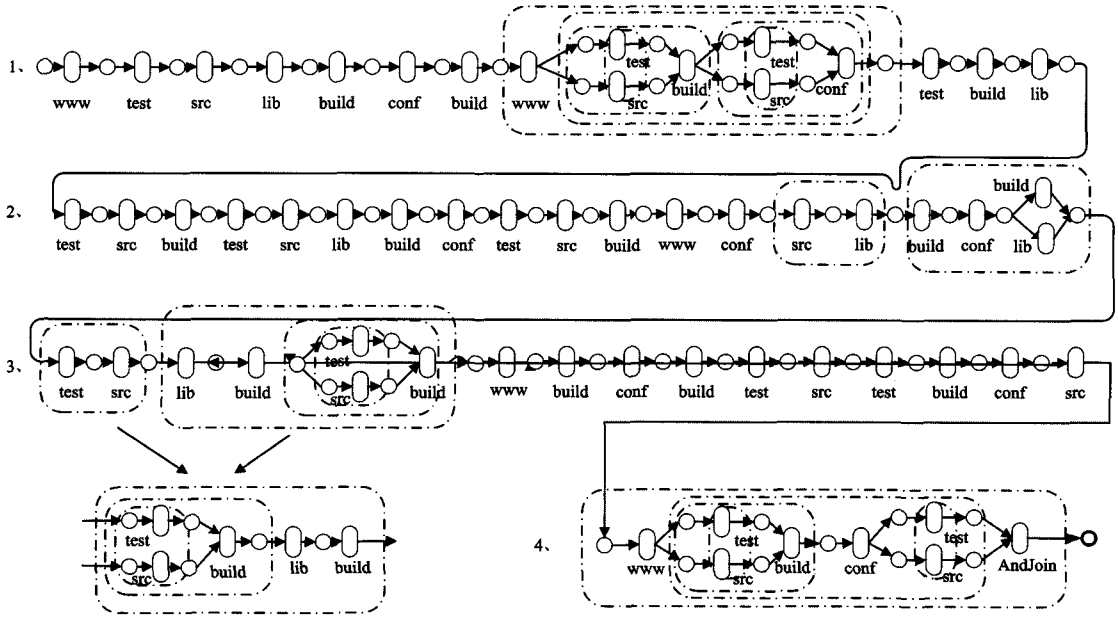

图3 包含活动模式的 ArgoUML 项目的过程模型

表4 活动模式统计

编号	模式	模式可能的调整	次数
1	test,src,build	{(test ‖ src)<build}, {build<(test ‖ src)}	8
2	test,src	{test<src}, {src<test}, {test ‖ src}	7
3	test,src,conf	{(test ‖ src)<conf}, {conf<(test ‖ src)}	6

从图3中阶段2的前半段和阶段3的后半段可以看出，并不是所有的序列都能采用折叠规则来获取活动模式，也有可能一个项目的所有阶段都不符合折叠规则，特别是当活动种类繁多的时候。此时可以通过增加相似序列的 M 值，使得活动模式能容纳多个序列片段，当然也会使模型变得更加复杂。分析该过程模型，对部分模式进行调整，在阶段2的第二个和第三个模式，test<src，lib<build<(test ‖ src)<build，把后一个模式通过模式调整方法2调整为(test ‖ src)<build<lib<build，则可以把这两个模式进行合并，如图3的箭头转换部分所示。然而，模式的调整虽然使得模型变得简洁，却也会丢失许多行为信息，因此模式调整需在征得客户意见后进行。得到的过程模型折叠率 $FR=76\%$，折叠效果良好。

结束语 本文提出一种基于一个软件项目实例的活动模式的挖掘方法。识别原子活动后，通过折叠规则得到具有控制结构的模式，并与原序列进行组合与替换得到过程模型。通过该模型可分析软件项目在不同时期的开发行为，以便更好地了解该项目。今后将对不同项目的开发行为进行对比，分析异同，得到适用范围更广的过程模型，以有针对性地指导软件项目的开发。

参 考 文 献

[1] Cook J E, Wolf A L. Discovering models of software processes from event-based data[J]. ACM Transactions on Software Engineering and Methodology (TOSEM), 1998, 7(3): 215-249

[2] Baier T, Mendling J. Bridging abstraction layers in process mining: Event to activity mapping[M] // Enterprise, Business-Process and Information Systems Modeling. 2013, Springer, 2013: 109-123

[3] Rubin V, Lomazova I, van der Aalst W M. Agile development with software process mining[C] // Proceedings of the 2014 International Conference on Software and System Process. ACM, 2014

[4] Hilbert M, López P. The world's technological capacity to store, communicate, and compute information[J]. Sscience, 2011, 332(6025): 60-65

[5] Manyika J, et al. Big data: The next frontier for innovation, competition, and productivity[M]. 2011

[6] Van Der Aalst W, et al. Process mining manifesto[M] // Business Process Management Workshops. Springer, 2012

[7] Van Der Aalst W. Process mining: discovery, conformance and enhancement of business processes[M]. Springer Science & Business Media, 2011

[8] Rubin V, Kindler E, Schöfer W. Activity Mining for Discovering Software Process Models[C] // Proc. of the Software Engineering. 2006

[9] Kindler E, Rubin V, Schöfer W. Incremental workflow mining based on document versioning information[C] // International Software process Workshop(ICSP). 2005: 287-301

[10] Van der Aalst W, Weijters T, Maruster L. Workflow mining: Discovering process models from event logs[J]. IEEE Transactions on Knowledge and Data Engineering, 2004, 16(9): 1128-1142

[11] Weijters A J, Van der Aalst W M. Rediscovering workflow mo-

dels from event-based data using little thumb[J]. Integrated Computer-Aided Engineering, 2003, 10(2): 151-162

[12] Günther C W, Van Der Aalst W M. Fuzzy mining-adaptive process simplification based on multi-perspective metrics[M]// Business Process Management. Springer, 2007: 328-343

[13] Lemos A M, et al. Using process mining in software development process management: A case study[C]// 2011 IEEE International Conference on Systems, Man, and Cybernetics (SMC). IEEE, 2011

[14] Ristad E S, Yianilos P N. Learning string-edit distance[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 1998, 20(5): 522-532

[15] Rubin V, et al. Process Mining Framework for Software Processes[M]// Software Process Dynamics and Agility. Springer, 2007: 169-181

[16] Bose R J C, van der Aalst W M. Abstractions in process mining: A taxonomy of patterns[M]// Business Process Management. Springer, 2009: 159-175

[17] Ukkonen E. On-line construction of suffix trees[J]. Algorithmica, 1995, 14(3): 249-260

[18] Li J, Bose R J C, Van Der Aalst W M. Mining context-dependent and interactive business process maps using execution patterns [M]// Business Process Management. Springer, 2011

[19] Fahland D, Van Der Aalst W M. Simplifying mined process models: an approach based on unfoldings[M]// Business Process Management. Springer, 2011: 362-378

(上接第 130 页)

的波动，这是因为 θ 取值太小时会增加干扰链路数量，对目标位置产生干扰；θ 取值太大时，只使用了少量有效链路，同样会造成较大误差。图 5(d)表示椭圆宽度 λ 对定位精度的影响，由图可知椭圆宽度 λ 取值很小时，定位误差变化很小，因此在定位过程中，λ 取一个很小的值即可。

结束语 本文提出的基于贝叶斯背景模型的免携带设备目标定位算法有效排除了多径环境中的冗余链路，减小了图像重建计算量，降低了多径效应的影响，并通过均值修正法对目标位置进行实时修正，保证了定位的精度。与传统 RTI 和 cdRTI 相比，本方法在多径环境中具有更高的定位精度。从实验可以得到位置估计的均方根误差达到 0.25m，满足室内定位系统的高精度要求。

参 考 文 献

[1] Saeed A, Kosba A E, Youssef M. Ichnaea A Low overhead Robust WLAN Device free Passive Localization System[J]. IEEE J. Sel. Topics Signal Process, 2013, 8(1): 5-15

[2] Jiang X, Kaishun W, Youwen Y, et al. Pilot: Passive device-free indoor localization using channel state information[C]// Proceedings of ICDCS. 2013: 236-245

[3] Patwari N, Agrawal P. Effects of Correlated Shadowing: Connectivity, Localization, and RF Tomography[C]// Information Processing in Sensor Networks. 2008: 82-93

[4] Wilson J, Patwari N. Radio tomographic imaging with wireless networks[J]. IEEE Trans. Mobile Comput., 2010, 9(5): 621-632

[5] Martin R K, Anderson C, Thomas R W, et al. Modelling and analysis of radio tomography[C]// 2011 4th IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP). Dec. 2011

[6] Chen X, Edelstein A, Li Y, et al. Sequential Monte Carlo for simultaneous passive device-free tracking and sensor localization using received signal strength measurements[C]// ACM/IEEE Information Processing in Sensor Networks (IPSN). April 2011

[7] Wilson J, Patwari N. See through walls: Motion tracking using variance-based radio tomography networks[J]. IEEE Trans. Mobile Comput., 2011, 10(5): 612-621

[8] Zhao Y, Patwari N. Noise reduction for variance-based device-free localization and tracking[C]// Proc. 8th IEEE Conf. SECON. Salt Lake City, UT, USA, Jun. 2011

[9] Zhao Y, Patwari N, Phillips J M, et al. Radio tomographic imaging and tracking of stationary and moving people via kernel distance[C]// Proc. 12th Int. Conf. IPSN. New York, NY, USA, 2013: 229-240

[10] Maas D C. Radio frequency sensing measurements and methods for location classification in wireless networks[D]. Utah Univ., 2014

[11] Kaltiokallio O, Bocca M, Patwari N. Enhancing the accuracy of radio tomographic imaging using channel diversity[C]// Proc. 9th IEEE Int. Conf. Mobile Ad Hoc Sensor Systems. Oct. 2012

[12] Hao Xiao-xi, Yang Zhi-yong, Guo Xue-mei. A method of link selection for radio frequency tomography with Bayesian compressive sensing[J]. Chinese Journal of Electronics, 2013, 41(12): 2507-2512(in Chinese)
郝晓曦，杨志勇，郭雪梅. BCS 实现的射频层析成像链路选择方法[J]. 电子学报，2013，41(12)：2507-2512

[13] Wilson J, Patwari N. A fade-level skew-Laplace signal strength model for device-free localization with wireless networks[J]. IEEE Trans. Mobile Comput., 2012, 11(6): 947-958

[14] Yang Zhi-yong, Huang Kai-de, Wang Guo-li. A new sparse reconstruction algorithm for device-free localization with sensor network[C]// Proceedings of the 9th Asian Control Conference. 2013: 1-6

[15] Haines T S F, Xiang T. background subtraction with dirichlet process mixture models[J]. IEEE PAMI, 2014, 36(4): 670-683

[16] Haines T S F, Xiang T. Background Subtraction with Dirichlet Processes[C]// Proc. 12th European Conf. Computer Vision (ECCV). 2012

[17] Kaltiokallio O, Bocca M, Patwari N. A fade level-based spatial model for radio tomographic imaging[J]. IEEE Trans. Mobile Comput., 2014, 13(6): 1159-1172