江苏大学

硕士学位论文

基于多Agent医疗欺诈行为检测系统的研究与设计

姓名：田青华

申请学位级别：硕士

专业：计算机应用技术

指导教师：杨鹤标

20090531

摘　要

医疗保险作为整个社会保障体系的组成部分之一，在保障全体劳动者的健康方面发挥了重要作用。然而由于医疗保险中信息不对称而引发的违规和欺诈行为也与之相生相伴，违规和欺诈的手段也越来越隐蔽化和专业化，风险控制也成为理论和实践操作上的难题。在此背景下，探索将 Agent 技术引入该领域，利用 Agent 智能技术进行实时侦测，实时或准实时地检测出医疗保险体系中出现的违规和欺诈行为，对加强医疗保险监督工作，减少医疗保险基金的流失有着十分重要的现实意义。

保人就诊是在一个开放式的动态环境中，在此过程中保人、医生及保险机构都是具有智能性和理性的行为主体以及对问题的求解能力，并能够预测其行为的后果，这些都与 Agent 的特点十分相符。为此，本文提出了一个基于多 Agent 医疗欺诈行为检测系统(Medical Fraud Detection System based on Multi-Agent，简称 MAFDS)。系统中将医疗机构和保险机构分别视为对应的 Agent，将欺诈行为的检测过程视为一个多 Agent 决策过程，它们间通过黑板提供的公共工作区获得彼此间的信息和意图，利用 Agent 的智能性和多 Agent 的协作能力来检测违规欺诈的现象。

本文研究的主要工作：

1、对目前常用的异常检测方法及 Agent 的相关理论和技术进行了分析，并研究了 Agent 技术在异常检测中的应用。

2、对医疗保险领域的问题进行了分析，确定了异常检测的主题及检测流程。通过对医疗数据的分析，给出了需检测的医疗行为，进行了用例建模。

3、针对 Agent 的技术特点，给出了基于多 Agent 医疗欺诈行为检测系统模型，阐述了系统的工作机制，进行了体系结构及相关 Agent 的详细设计。

4、在 JADE 平台上，对系统原型进行了实现。针对门诊用药的效-效相似性，给出了异常检测策略，并通过实际医疗数据的测试，验证了系统的可用性。

关键词：多 Agent 系统，异常检测，医疗保险，欺诈，协作

ABSTRACT

Medicare as a part of the entire social security system plays a great role in health insurance of all the labors. However, the frauds and violations caused by the nonsymmetrical information come into being frequently, meanwhile, the means of frauds becomes more hidden and professional, resulting into lots of difficulties in risk management. Under this background, introducing the Agent technologies into the medicare domain to conduct real-time detection of medicare frauds and violations is important for medicare supervision and fund loss decrease.

Policyholders to see a doctor are in a open dynamic surrounding, during the process, doctors, policyholders and insurance companies are intelligent and rational agents and have the abilities to related problems solving, in addition, they can predict the consequences of their own behaviors. In a word, all of the above mentioned matches with Agent's features. Therefore, a medical fraud detection system based on multi-Agent is proposed in this dissertation. In the system, the medical institutes and insurance companies are taken as the corresponding Agents, frauds detection as the process of multi-Agent decision making. The blackboard among these Agents provides a public workspace to obtain information and intention from each Agents, the fraud detection actions are so identified through making use of Agent's intelligence and multi-Agent coordination.

The main contents are described as follows:

1、Analyzes the currently used abnormality detection methods and Agent related theories and technologies, applications of Agent technologies to abnormality detection is also explored.

2、On the basis of analysis on the medicare domain problems, the subject and flow of abnormality detection are decided. Defines the medical actions to be detected according to analyze the medical data.

3、In terms of features of Agent technologies, shows a model of medical fraud detection system based on multi-Agent, discusses its working mechanism and designs its architecture structure and inner functional Agents.

4、Implements the detection system prototype on the platform JADE. Furthermore, aiming at the effect-effect similarity in outpatient medication gives the abnormality detection strategies and verifies the system feasibility via test on a real-world data.

KEY WORDS: multi-agent system, anomaly detection, medicare, fraud, cooperation

学位论文版权使用授权书

本学位论文作者完全了解学校有关保留、使用学位论文的规定，同意学校保留并向国家有关部门或机构送交论文的复印件和电子版，允许论文被查阅和借阅。本人授权江苏大学可以将本学位论文的全部内容或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存和汇编本学位论文。

保　密 □，在　　年解密后适用本授权书。

本学位论文属于

不保密 ☑。

学位论文作者签名：田春华　　指导教师签名：杨敏官

2009年 6月 4日　　2009年 6月 4日

独创性声明

本人郑重声明：所呈交的学位论文，是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已注明引用的内容以外，本论文不包含任何其他个人或集体已经发表或撰写过的作品成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本人完全意识到本声明的法律结果由本人承担。

学位论文作者签名：田青华

日期：2009 年 6 月 4 日

第一章 绪论

1.1 研究背景

医疗保险是指当人们生病或受到伤害后，由国家或社会给予的一种物质帮助，即提供医疗服务或经济补偿的一种社会保障制度。医疗保险在保障公民健康方面发挥了重要作用，对于维护社会稳定、发展社会经济、保障人民生活具有重要影响。自 1998 年开始改革以来，城镇职工基本医疗保险制度已在全国范围内基本建立。

然而与此同时，医疗保险也面临着越来越多的挑战，频频见诸报端的医疗保险欺诈和违规现象却如蛀虫般悄然与之相生相伴，医疗保险欺诈和违规没完没了，欺诈手法也变的越来越隐秘，如冒名顶替；参保单位的经办人员利用职务之便，使用参保人年度内可享受的医疗费用余额冒名看病报销；将非医保支付的项目如美容、保健品、日用品等换成医保支付的项目，或以药换物；将非医保药开成医保药；将普通门诊药写成门诊特殊病用药，以提高报销比例；医疗明细作假；多申报诊疗费用，如用三天药申报成六天；凭空捏造；未进行的医疗检查项目、未使用的设施却开出检查费、治疗费和使用费等，进行申报结算；伪造住院和门诊病历、各种检查治疗记录、报告单等[1]。这些向医保基金管理机构骗取医保基金或医保待遇的行为，严重威胁到医疗保险制度的稳定发展和医疗保险基金的合理利用[2-3]。

虽然，医疗保险行业业务大部分已经信息化，但所开发的系统主要用于内部的业务处理，或是用于对保险数据进行一些决策所需的分析和预测，在对医疗数据存在的异常行为的检测方面尚未见相关的报道。

异常[4]是指在数据集合中显得与众不同的数据，这些数据往往超出了正常的范围，使人怀疑这些数据并非随机偏差，而是产生于完全不同的机制。异常检测[5-8]作为一种积极主动的系统安全防护技术，通过对出现的异常数据及时响应并对异常数据进行处理，来保证系统的正常运行。

本文从医疗保险的自身特点出发，试图结合医保基金的管理实践，借鉴异常检测理论，研究如何将 Agent 技术与医疗保险领域相结合，充分利用 Agent 具有的主动性、反应性、灵活性等优点，通过构建欺诈行为检测系统，为医疗欺诈行为的检测提供一个新的思路。

1.2 研究现状

1、异常检测研究现状

当前社会中许多领域都存在着异常，如信用卡欺诈、电信欺诈、网络入侵等等，对异常的研究具有十分重要的实际意义，因此异常检测研究也越来越受到研究者们的重视，并在实际应用中发挥了重要的作用。

美国在 2002 发生了一起涉及十万多笔虚假互联网交易的神秘信用卡诈骗案，Spitfire 公司总部在一个半小时内收到了多达十四万笔的信用卡业务，其中有六万多笔被确认有效。而正常情况下公司网站每天通常处理 5-30 笔交易，所以交易数量的异常激增使得管理人员提高了警惕性，在总部设在芝加哥的在线信用卡交易商 Online Data 公司的帮助下，联合联邦调查局，共同对此案展开调查。最后的调查结果表明原先被确认为有效的交易几乎都是虚假的，由于处理及时，因而使公司免受了巨大损失。

ClearCommere 公司开发的 ClearCommere Engine 产品主要是针对在线商户防范欺诈所用，这是一个比较完整的同时使用基于规则模型和基于神经网络模型的反欺诈系统。首先系统接收一个来自于 Web 的交易，先进行简单的规则判断，比如是否有错误格式，卡号是否过期等等，阻止了一些显而易见的错误。然后系统会进行卡号密码的认证。接着交易要经过 AVS(Address verification system)和 CVM(card Verification Method)两个基于规则的欺诈检测子系统的处理。AVS 主要进行地址验证，CVM 主要是卡身份验证。当交易从上面这两个侦测子系统顺利通过后，进入一个基于神经网络的风险评分子系统。神经网络要通过很多历史欺诈信息来建立。如果风险评分系统产生的分数比较高，则表明有可能是欺诈行为，业务人员会对这些交易进行进一步的审核。如果是成功的交易，则直接通知商户发放货物，否则会回退给用户，告知有欺诈交易发生。

Portnoy 等人采用单链接聚类算法来分离异常数据和正常数据[9]，但这种方法需要正常数据远远多于异常数据，无法检测突发性的大量异常情况。为了解决这个问题，Slobodan 等人提出一种新的模型[10]，采用 K-means 算法简单地把数据集分成两簇：数据多的簇当作正常簇、数据少的簇当作异常簇，再利用 DB 指数[11]评价每个簇内数据的相似程度，根据预先定义的阈值来判断是否需要给簇重新标记。

2、Agent 技术研究现状

目前，面向 Agent 技术作为一门设计和开发软件系统的新方法已经得到了学术界和企业界的广泛关注，基于 Agent 技术的应用也扩展到多个领域，如：个人信息管理、电子商务、接口设计、计算机游戏、商业和工业过程的控制、消息软

件、交通控制、金融业务管理开发工具、信息的筛选与搜集、用户界面软件、工作流管理和网络管理等方面。同时，Agent 的应用也涉及患者监控、医疗保健等医学应用领域中[12]。

目前对 Agent 的研究大致分为如下几个方面：

(1) 智能 Agent，即 Agent 的基础理论研究。

(2) 多 Agent 系统的研究。

(3) 面向 Agent 的程序设计。

(4) 面向 Agent 的软件工程。

近年来，Agent 技术的研究工作在美国和欧洲的大学和工业组织中不断取得新的进展。各种基于多 Agent 系统的原型甚至工业产品纷纷出现。如前台办公系统软件提供商 Siebel Co.采用 Agent 作为客户智能导引的呼叫中心产品、美国卡内基·梅隆大学研制的 Visitor Hosting 系统等。

有关 Agent 的理论跟技术为分布式开放系统的分析、设计和实现提供了一个崭新的途径，被称为软件开发的又一重大突破[13]。Agent 理论与技术研究最早源于分布式人工智能（Distributed Artificial Intelligence,DAI）[14]，但从 80 年代末开始，有关 Agent 的理论跟技术研究从 DAI 领域中拓展开来，并与许多其他领域相互借鉴和融合，在许多不同于最初 DAI 应用的领域得到了更为广泛的应用。

在系统监控和专家诊断方面，利用多 Agent 的联合意图机制可实现分布式预测与监控。Jennings 和 Draa 等分别利用 Agent 的联合意图实现了联合监控机制[15]。Hartvigsen 将 Agent 技术应用于暴风雨气象观测，将各区域观测站分别作为一个 Agent，各 Agent 对观测数据进行处理，做出局部预测，然后进行协调，构成一个多 Agent 系统。通过网络对整个地域进行分布式问题求解，最终形成一个可靠的一致解，从而实现全局预测。Russell[16]利用 Agent 技术建立了用于复杂问题实时诊断的分布式系统(MARVEC)，系统将复杂的诊断问题划分成多个子区域。单个 Agent 尽可能负责某个子区域，以便分别承担诊断任务，减少通讯量，提高实时性。Agent 通过元知识寻找超出其领域的合作系统中的多个 Agent 协调解决涉及多个领域的诊断问题。Wang 等介绍了原子能发电厂故障分析和监控系统(APACS)，该系统采用多 Agent 技术，由分布在不同位置的计算机上的 Agent 构成多 Agent 系统，从而完成分布式协调监控与诊断任务。各个 Agent 采用不同的表达方式和推理机制，以完成各自的任务。Lane[17]等采用黑板 Agnet 和实时知识库及实时推理等 Agnet 技术对机器人进行了设计，已成功应用于水下探险、测量等工作。

国内开始 Agent 和 MAS 的理论研究虽比国外落后近十年，但也取得了很大的进步，涌现出大量的科研成果，中科院研制的多主体环境 MAGE 更好的把

Agent 技术推向了应用方面，如信息处理、电子商务等。中国第一届 Agent 理论与应用学术会议的胜利召开，有力的促进了国内 Agent 技术与理论的迅速发展。

随着科技的不断发展，欺诈检测已经不再停留在手工和人脑的基础上，单纯依靠人的观察和经验已不能满足欺诈检测的需要。欺诈检测系统(Fraud Detection System，简称 FDS)作为检测领域信息系统的重要方面已成为计算机检测应用的热点和主要发展方向。

异常检测技术及 Agent 技术的发展为构建基于多 Agent 医疗欺诈行为检测系统提供了强有力的理论基础。

综上所述，借助现代信息技术，设计和开发一套医疗欺诈行为检测系统，对于保证社会保险基金的合理利用和安全，支持劳动保障部门的社会保障基金非现场监督工作就显得十分必要。

1.3 研究内容

本课题的研究内容是针对医疗保险领域特点，在异常检测技术和 Agent 技术的基础上，进行基于多 Agent 医疗欺诈行为检测系统的研究与设计，其主要研究内容以下几个方面：

1、对目前常用的异常检测技术、Agent 理论及 Agent 技术在异常检测中的应用进行分析，为论文的后续工作奠定基础。

2、对医疗保险领域中的违规欺诈现象进行研究，通过分析异常医疗行为的成因，抽取出医疗保险门诊的违规欺诈行为，确定系统的检测主题。通过分析医疗保险机构所采用的检测方法，针对检测主题，给出系统的检测流程，并对系统主要的功能进行用例建模。

3、在上述研究的基础上，结合 Agent 技术，构建基于多 Agent 医疗欺诈行为检测系统模型，并进行系统的体系结构、医疗数据预处理及各个功能模块的详细设计。针对医疗欺诈行为检测流程，确定系统中个 Agent 间的协同工作机制。

4、原型系统实现。在 JADE 平台上，进行基于多 Agent 医疗欺诈行为检测系统的原型系统的开发，完成对系统中关键技术的实现，并以具体的测试样例验证系统的可用性。

1.4 论文结构

本文分为六章，其主要内容概要如下：

第一章 提出课题的研究背景，分析领域的研究现状，确定本文的研究内容，

给出论文的结构。

第二章 介绍异常检测的情况和 Agent 相关技术，并分析 Agent 技术在异常检测中的应用情况。

第三章 研究对医疗违规欺诈行为的成因及特征，提出检测的主题和检测流程，最后进行用例建模。

第四章 对基于多 Agent 医疗欺诈行为检测系统的体系结构、数据预处理及各个 Agent 进行详细设计，给出 Agent 协同工作机制。

第五章 在 JADE 平台上对原型系统进行了开发，对系统中的关键技术进行实现，通过实际测试数据对系统进行验证。

第六章 进行全文总结，提出了需要进一步研究和改进的方向。

第二章 相关技术

本章主要介绍了目前常用的异常检测技术，并对各异常检测技术做了阐述。论述了 Agent 的相关理论和技术，最后介绍了 Agent 技术在异常检测中的应用情况，为论文的研究工作做了铺垫。

2.1 异常检测技术

2.1.1 异常检测

基于异常发现的异常检测[18-21]为每一个目标系统及其用户建立一个正常活动范围，通过比较系统及用户的审计数据和已建立的正常活动范围，根据两者的差异是否超出一定的阈值来判定是否有异常行为发生，它的优点是通用性强，可以检测出以前从未出现过的异常模式。

当然，异常的行为不一定就是欺诈行为，对于系统行为总共有如下四种情况：

(1) 行为具有欺诈性，但不是异常的。在这种情况下，由于欺诈行为不是异常的，所以欺诈检测检测不到这种行为，从而产生漏检情况。

(2) 行为不具有欺诈性，但却是异常的。在这种情况下，尽管行为不是欺诈性的，但会被检测出欺诈，从而产生误报。

(3) 行为是非欺诈性的，同时也不是异常的。这时系统会正确处理。

(4) 行为具有欺诈性，同时也是异常的。此时系统会正确处理。

异常检测的关键问题在于正常模式的建立以及如何利用该模式对当前的用户行为进行比较，从而判断出与正常模式的偏离程度。常用的异常检测方法如下所示：

1、基于统计的异常检测

基于统计方法假设所给定的数据集存在一个分布或概率模型，然后根据相应模型并通过不一致性测试来发现异常数据。基于统计方法的异常检测技术通过分析大量的系统参数生成正常行为库，并自适应地学习系统的正常行为模式。每个行为由一组检测阈值表示。系统采用统计学原理计算用户行为跟正常行为阈值之间的偏差，根据计算结果是否超出阈值来判断行为是否异常。统计分析方法的优势在于不必像误用检测系统那样对规则库不断地进行更新和维护，其主要缺陷是不能提供对异常行为的实时检测。统计方法在异常检测中主要有如下应用：

(1) 一个典型的系统是 SRI International 的(NIDES)[22]，NIDES 所使用的统

计分析技术支持对每一个系统用户和系统主体建立历史统计模式。所建立的模式被定期地更新，可以及时地反映出用户行为随时间推移而产生的变化。检测系统维护一个由行为模式组成的统计数据库，每个模式采用一系列系统度量来表示特定用户的正常行为。模式所包含的各个向量每天都以指数因子形式衰减，同时将新的用户行为所产生的审计数据嵌入到知识库中，计算出新的模式向量存储在知识库中。NIDES 使用 Agent 完成数据提取和格式化功能，并提交给分析模快。统计分析组件通过学习用户的行为，完成异常检测功能。

(2) Haystack 模型[23]是最早的基于统计的异常检测 IDS 之一，该模型采用了基于用户和组的异常检测策略，把系统参数建模成独立的高斯随机变量，并为每个属性定义正常值的范围。如果在一个会话期间，某个属性落在了正常范围之外，那么这个行为的异常度将被提高。假设各个属性之间相互独立，计算异常度的概率分布，如果异常度达到一定高度，Haystack 将产生一个报警。Haystack 也为每个用户组维护一个数据库，如果某个用户还没有被检测过，系统将利用基于用户组成员的限制策略为该用户建立一个具有最小权限的新用户轮廓。

该模型是工作在离线状态下的，无法满足实时检测的要求，因为要达到此目标需要非常高效的性能。由于维护的各个用户的活动范围是相互独立的，对于系统管理员来说，一个普遍的问题就是如何确定哪个属性对于表示异常行为是有效的。

2、基于数据挖掘的技术

数据挖掘[24] (Data Mninig)就是从大量的、不完全的、模糊的、有噪声的数据中提取出隐含在其中的人们事先不知道的、有效的、可信的、并能最终被人理解的信息和知识的过程。数据挖掘方法是由人工智能、机器学习的方法[25]发展而来，结合传统的统计分析方法、模糊数学方法以及科学计算可视化技术，以数据库为研究对象，形成了数据挖掘方法和技术。

基于数据挖掘的异常检测系统利用数据挖掘中的关联分析、序列模式分析等算法提取与安全相关的系统特征属性，并根据系统特征属性生成安全事件的分类模型，用于对安全事件的自动鉴别。

数据挖掘方法能够对安全审计数据进行全面、高速和准确地分析，从包含大量冗余信息的数据中提取出尽可能多的隐藏的安全信息，抽象出有利于进行判断和比较的特征模型，这种特征模型可以是基于特征检测的特征向量模型，也可以是基于异常检测的行为描述模型。目前，数据挖掘技术被广泛地运用在异常检测中。

(1) 关联分析

关联分析的主要目标反映一个事件和其他事件之间的依赖和关联知识。若两

个或多个数据项的取值之间重复出现且概率很高时，那么其中一项的属性值就可以依据其他属性值进行预测。最为著名的关联规则发现方法是 R.Agrawal 提出的 Apriori 算法。通常，我们需要使用支持度和信任度两个阈值来筛选其中的强规则。关联规则的发现可以分为两步，第一步是迭代识别所有频繁项目集(frequent itemsets)，要求频繁项目集的支持度不低于用户设定的最小阈值，这一步是关联规则发现算法的核心；第二步是从频繁项目集中构造可信度不低于用户设定的最低值的规则。

(2) 聚类方法

聚类[26]是按照一定的规则将数据划分为不同的簇，目的是使得属于同一个簇中的对象之间具有较高的相似度，而不同簇中的对象差别较大，是一种无监督分类法。

其中较有特色的方法有：K-means 算法、CLIQUE 算法、BIRCH 算法、Clara 算法、Chameleon 算法。此外还有一些其它的聚类算法如 CURE 算法、ROCK 算法、层次聚集等几十种。

(3) 神经网络方法

神经网络为解决大复杂度问题提供了一种相对来说比较有效的简单方法，它模拟了人脑神经元结构，具有良好的鲁棒性、自适应性、并行处理、分布存储和高度容错等特性。神经网络可以很容易的解决具有上百个参数的问题。神经网络常用于两类问题：分类和回归。典型的神经网络模型以 MP 模型和 Hebb 学习规则为基础分为如下三类：以感知机、BP 反向传播模型、函数型网络为代表的，用于分类、预测和模式识别的前馈式神经网络模型；以 Hopfield 的离散模型和连续模型为代表的，分别用于联想记忆和优化计算的反馈式神经网络模型；以 Art 模型、Koholon 模型为代表的，用于聚类的自组织神经网络模型。

神经网络是一组连接的输入/输出单元，其中每个连接都与一个权相关联，在学习阶段，通过调整神经网络的权，使得能够预测输入样本的正确类标号来学习。

基于人工神经网络的异常检测系统的处理包括两个阶段：第一阶段的目的是构造异常分析模型的检测器，使用代表用户行为的历史数据进行训练，完成神经网络的构建和组装。第二阶段则是异常分析模型的实际运作阶段，神经网络接收输入的事件数据，与参考的历史行为相比较，判断出两者的相似度或偏离度。

Ghosh 等使用前馈网络和 Elman 递归网络[27]来对系统调用序列进行分类。但 Elman 网络的训练代价昂贵且需要大量的神经网络。Ramadas 等提出一个利用自组织映射进行网络异常流量检测的系统 ANDSOM[28]。该系统为每个检测服务建立一个二维的自组织映射。在训练阶段通过捕获正常目标数据的特征模式来

训练神经元。在检测阶段若待检数据通过神经元后计算的距离大于预先设定的阈值则认为该数据是异常的。

(4) 遗传算法

遗传算法是一种基于生物自然选择与遗传机理的随机搜索算法，它以自然选择和遗传理论为基础，将生物进化过程中"适者生存"规则与群体内部染色体的随机信息交换机制相结合的搜索算法。它主要包含三个基本操作：①选择：该操作从一个父代中挑选出适应性强的个体产生新的后代。②交叉：交叉操作选择两个不同个体的染色体的部分基因进行交换，形成新的个体。在很大程度上遗传算法的性能取决于所使用的交叉操作的性能。③变异：变异操作对某些个体的某些基因进行变异。通常情况下，变异操作就是将基因值取反(0 变 1，1 变 0)。

遗传算法的基本思想可归为如下两点：①将物种进化的理论用于求问题的解，物种的进化又可分为遗传和变异两个方面；②只有适合环境的物种才能保留下来，因而经反复求解后就可以得到最佳的解。遗传算法已在优化计算和分类机器学习方面发挥了比较好的效果，其商用产品有美国的 GeneHunter。将其用于金融及医疗等方面的应用可获得很好的效果。

在基于遗传算法的异常检测技术中，异常检测的过程可以抽象为：为检测记录定义一种向量表示形式，这种向量或者代表正常的行为，或者代表异常的行为。通过对所定义向量进行测试，提出改进的向量表示形式，不断重复这个过程，直到得到令人满意的结果为止。

(5) 粗糙集方法

粗糙集理论是一种研究不精确、不确定性知识的数学工具。 粗糙集理论在知识发现研究中有着许多具体应用，特别适合于数据之间依赖关系发现、评价某一属性的重要性、数据相似或差异发现、数据模式发现、从数据中产生一般决策规则、削减冗余对象与属性、寻求属性的最小子集以确保产生满意的近似分类等等。粗糙集方法有几个优点：不需要预先知道额外信息，简化了输入信息的表达空间；算法简单，易于操作。现在国际上已经研制出来了一些基于粗糙集的工具应用软件，如加拿大 Regina 大学开发的 kdd-r，美国 Kansas 大学开发的 lers 等。

(6) 决策树方法

决策树是一种常用于预测模型的算法，它提供了一种在什么条件下会得到什么值这类规则的方法，用于对离散和连续属性进行预测性建模，决策树通过将大量数据有目的分类，从中找到一些有价值的，潜在的信息。它的主要优点是描述简单，分类速度快，特别适合大规模的数据处理。最有影响和最早的决策树方法是由 Quinlan 提出的著名的基于信息熵的 ID3 算法。决策树是一个类似于流程图的树结构，其中每个内部节点表示在一个属性上的测试，每个分支表示一个测试

输出，而每个树叶节点代表类或类分布。树的最顶层节点是根节点。

对于离散属性，该算法根据数据集中输入列之间的关系进行预测，它使用这些列的值预测指定为可预测的列的状态。对于连续属性，该算法使用线性回归确定决策树的拆分位置。

(7) 序列模式发现

序列模式是指在多个数据序列中发现共同的行为模式。对序列数据库而言，序列模式发现问题就是在给数据库中寻找所有的频繁序列或所有的最长频繁序列。R.Agrawal 称在顾客序列数据库中满足用户指定最低支持度的最长频繁序列为序列模式。其发现方法与关联规则大致相似，但也存在区别，关联规则仅仅发现事务内部的模式，即频繁项集，而序列模式则是发现事务之间的模式，即频繁序列。另外还有一种在给定长度的时间区间内出现的时间的有序集合。而频繁情节指在时间序列中具有一定出现频率的情节。如果在事件序列中发现了频繁情节，就可以描述或预测该序列的行为。

3、基于机器学习的异常检测技术分析

机器学习是研究计算机怎样模拟或实现人类的学习行为，以获取新的知识或技能，重新组织已有的知识结构使之不断改善自身的性能。它是人工智能的核心，是使计算机具有智能的根本途径，其应用遍及人工智能的各个领域，它主要使用归纳、综合而不是演绎。机器学习是关于理解与研究学习的内在机制、建立能够通过学习自动提高自身水平的计算机程序的理论方法的学科。近年来机器学习理论在诸多应用领域得到成功的应用与发展，已成为计算机科学的基础及热点之一。采用机器学习方法的计算机程序被成功用于机器人下棋程序、语音识别、信用卡欺诈检测、自主车辆驾驶、智能机器人等应用领域，除此之外机器学习的理论方法还被用于大数据集的数据挖掘这一领域。实际上，在任何有经验可以积累的地方，机器学习方法均可发挥作用。

机器学习的目标与统计、数据挖掘是相同的，但是，与统计方法不一样的是它不注重理解数据的产生过程，而是注重构建一个能够根据先前的结果不断改进其性能的系统。也就是说，在最新获得的信息基础上，基于机器学习的系统具备改变执行策略的能力。目前，机器学习被广泛地应用到异常检测中。

(1) 系统调用序列

基于系统调用序列分析是通过分析服务进程执行时产生的系统调用序列，得到一组能够描述程序正常行为的系统调用序列模式集，并以此来进行异常检测。1996 年，Forrest 等人利用一组定长短序列模式来表示程序的正常行为模型。它们分析了一些 Unix 系统的程序，利用定长的系统调用序列来建立一个程序的正常行为轮廓。

一般来说，随着短序列模式长度的增加，IDS 倾向于在系统调用序列中捕捉到更多的异常，但同时序列模式集的规模也会增加，算法执行效率降低，并且误报率也会增加。由于程序的结构之间存在显著的差异，因此利用不定长序列模式来描述程序的正常行为比定长模式更为合理。由于堆栈中函数返回地址链信息反映了程序内部的函数依次调用的结构，使得每个模式分别代表了程序按某一路径执行的函数产生的完整系统调用序列，张诚、彭勤科提出了一种利用进程堆栈中的函数返回地址链来构造不定长序列模式的方法：根据产生系统调用的不同函数来分解系统调用序列从而得到不定长模式，并对得到的不定长模式进行合并从而精简模式集，并以不定长模式为基本单位构建了一个基于马尔可夫链的异常检测模型。

(2) 贝叶斯推理

基于贝叶斯推理的异常检测方法通过在数据中发现不同类别的数据集合，进而发现异常数据，借此对异常行为进行检测。该方法能对一个广泛的认知行为进行建模，具有概率推理能力。贝叶斯推理以其独特的不确定性知识表达形式、丰富的概率表达能力、综合先验知识的增量学习特性等成为当前众多分类方法中比较出色的方法之一。

2.1.2 特征检测

基于模式匹配的特征检测是对已知的异常违规行为进行分析，提取检测特征，构建异常模式，通过将系统当前数据与异常模式进行匹配，判断是否有异常行为发生。异常检测只能识别出那些与正常数据有较大偏差的行为，而无法知道具体的异常情况，很难获得精确的判定准则，异常检测经常会出现错报，也就是说将正常的数据误认为是异常违规数据。特征检测基于已知的异常规则，它能够比较准确地检测到某些特定的异常行为，它依赖于事先定义好的规则库，所以无法检测系统未知的异常行为，因而有时会产生疏漏。因而可以通过综合这两种方法来检测异常用以降低错报和漏报率。特征检测示意图如图 2.1 所示。

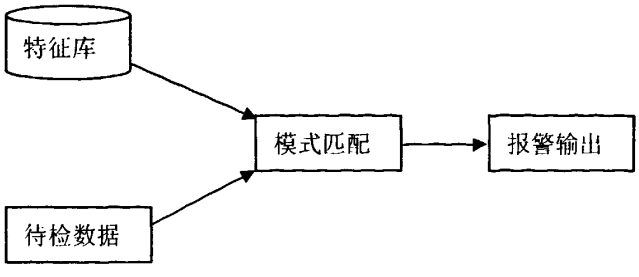

图 2.1 特征检测示意图

基于误用的异常检测技术的研究主要是从20世纪90年代中期开始，当时主要的研究组织有SRI，Purdue大学和California大学。

对于误用检测技术来说，最重要的技术问题是如何全面描述异常行为的特征，减少误报率。执行误用检测需要具备完备的检测规则库、可信的用户行为记录和可靠的行为记录分析技术。主要的误用异常检测类型如下：

1、专家系统

异常检测专家系统最显著的特征是采用一定的规则表示异常的行为，形成专家知识库，然后输入检测数据，系统根据知识库中的内容对检测数据进行评估，判断是否存在异常行为模式。专家系统将有关异常的只是转化为if-then结构的规则，其中if表示异常特征，then表示系统处理措施。专家系统的优点在于把系统的推理控制过程和问题的最终解答相分离，即用户不需要理解或干扰专家系统内部的推理过程，而只须把专家系统看作是一个自治的黑盒子。专家系统应用于异常检测时存在的问题包括：系统性能完全取决于专家知识、只能检测已知的异常模式。专家系统是最早的误用检测技术之一，被许多经典的检测模型所采用。

2、模式匹配

模式匹配主要是用一定的模式描述来提取异常行为的主要特征，通过匹配机制从目标中发现异常。模式匹配检测技术是由Kumar在1995年提出的，目前已成为异常检测领域中应用最为广泛的检测手段和机制之一，其特点是原理简单、扩展性好、检测效率高和可实时检测，但其局限性是只能适用于比较简单的异常检测，且误报率高。但由于系统的实现、配置和维护都非常方便，因此得到了广泛的应用。

综上所述，异常检测技术作为一种积极主动的安全保护技术，在保护系统数据方面发挥了重要作用，将其用在欺诈检测中可以有效的检测出系统中的异常行为和数据，从而保证了系统的正常运行。

2.2 智能Agent技术

2.2.1 Agent的定义

Agent又叫做智能体，是一个满足特定需求的计算机(硬件或软件)系统，是一个自治实体，它作用于某一特定环境，具有感知能力、问题求解能力和与外部环境进行通信的能力，并具有高度的灵活性和自治性，而且具有一定的生命周期，能持续自主地发挥作用，并能够影响和改变环境。它不仅封装了内部状态，而且也封装了内部行为[29]，它可以在尽可能不打搅用户的前提下依靠自身的能力，

采用各种可能的方法和技术来完成用户指定的各种任务。它可根据用户定义的准则自动搜索收集用户可能感兴趣的信息，并根据用户需求将目标传递至用户指定的目的地，成为用户与资源之间的中介。Agent 技术是分布式计算的一个分支，广泛应用于制造、金融、商业、电子商务等社会的各个领域。

目前为止，学术界对 Agent 还没有统一的定义，人们从不同角度对 Agent 提出了不同的定义。Shoham 认为一个 Agent 包含了信念（belief）、能力（capability）、决定（decision）和承诺（commitment）等精神状态[30]。

软件领域的研究者认为，软件 Agent[31]是具有自主性和协作性的，能完成某一特定任务的运行实体，它能根据对环境的感知，控制自身的策略和行为。

在分布式计算领域，人们认为 Agent 是指在协同计算环境中持续发挥作用且具有自主性、交互性、反应性和主动性的计算实体。

对于 Agent 特性的定义，目前被普遍引用并接受的是 Wooldridge 和 Jennings 提出的关于 Agent 的弱定义和强定义[32]：

1、弱定义：Agent 是处在某个环境中的计算机系统，它拥有以下几个特性：自主性、社交性、反应性和预动性。

自主性：Agent 能够在没有外界直接操纵的情况下持续运行，具有控制其自身行为和内部状态的能力。自主性是 Agent 区别于其他实体的主要方面。

社交性：Agent 可以通过通信语言与环境、人及其它 Agent 进行交互，从而完成某一特定的任务。

反应性：Agent 可以感知其所处的外部环境，并能够对所处环境的变化做出及时而适当的反应。其行为通过触发规则或执行定义好的计划来更新 Agent 的事实库，并发送消息给环境中的其他 Agent。

预动性：Agent 不仅能简单地对环境做出反应，而且可以主动发起可以基于目标的行为，或者说是自发的行为。

2、强定义：Agent 在弱定义的特性基础上，还要包括一些其它的，如人类所具有的某些特性。

协调性：Agent 能够处理复杂和高水平的任务，具有分配并调度任务以满足目标的能力。

连续性：Agent 具有连续运行的能力。

适应性：Agent 应能够调节自身的行为，以适应用户的需求。

友好性：若系统中各 Agent 的目标之间没有冲突，则每个 Agent 会尽力的帮其它的 Agent 完成所请求的要求。

合理性：Agent 能够在其信念允许的范围内采取合理的措施以完成它的任务。

协作性：Agent 在执行任务期间会注意到用户的一些不合理的要求。

当前比较常用的 Agent 模型是 Rao 等提出的 BDI(Belief(认知)，Desire(愿望)，Intention(意图))模型[33-35]，认知是 Agent 对于所处环境的认知和理解，愿望表示所希望达到的目标，意图则是描述了 Agent 为达到目标而采取的行为步骤。意图在 Agent 的行动中可能会随着环境的改变而采取新的动作步骤。

Agent 比较适合于将那些能根据功能进行分解和划分的应用问题。在这些应用系统中采用多 Agent 技术具有如下优点：系统的运行速度快；带宽要求低；可靠性高等。

2.2.2 Agent 分类

对于 Agent，目前还没有统一的分类[36]方法，现在普遍接受的是如下所示的几种分类方式：

1、按照功能划分

(1) 界面 Agent：协助用户完成乏味而重复性的工作，强调自治性和学习性，以完成所有者的任务。界面 Agent 最主要的性质是它是一个个人助理，它与同一工作环境中的用户进行合作。一个 Agent 与其他 Agent 之间如果有互操作的话，这种互操作仅限于请求建议。

(2) 信息 Agent：是一个具有主动性、适应性和相互操作性的万维网信息管理者，实现对分布式的信息进行搜索并管理网络资源。

(3) 任务 Agent：帮助人类进行复杂的决策和进行一些其它的知识处理工作。

(4) 移动 Agent：能在分布式环境中进行移动来完成任务的 Agent，移动的同时携带所积累的状态数据等。

2、按照属性划分

(1) 合作 Agent：系统中强调 Agent 的自治性和 Agent 间的相互合作性，在一个开放的多 Agent 环境中协调完成 Agent 的所有者的任务。

(2) 反应 Agent：由简单的行为模式构成，没有关于其环境的内部符号模型，以刺激-应答的方式对环境做出反应。

(3) 混合 Agent：它是由两种或两种以上 Agent 体系联合构成。

(4) 审慎 Agent：在目标指导下具备自主行动能力的 Agent，将人工智能领域的感知、学习、规则和方法等认知功能封装在一起。

3、按照行为划分

(1) 合作 Agent：能够跟环境中的其它 Agent 进行合作完成指定的任务。

(2) 自主 Agent：在复杂的动态环境中自主感知并采取行动。

(3) 助手 Agent：辅助人类完成某些工作，提供友好的交互方式。

2.2.3 多 Agent 系统

虽然单 Agent 具有很多优点，但是许多问题对于单个 Agent 来说比较复杂而不能解决，随着分布式人工智能的发展，多 Agent 系统[37-38](MAS)已经被应用到广泛的领域。

所谓 MAS 是指由多个分布的、相互作用的、相互联系的 Agent 组成的计算机系统，这些 Agent 成员之间通过相互协同，相互服务，共同完成一个任务，从而提高了系统效率。Agent 所具有的一些如自治、社会、学习和和反应等特性对构建复杂的分布式系统非常有用，可将基于多 Agent 的理论和技术应用于医保欺诈监控。

基于多 Agent 的系统将任务分配给多个 Agent，每个 Agent 承担一部分功能，彼此独立运行，单个 Agent 运行失败不会影响其它 Agent 的运行。MAS 一般是开放的，系统中根据需要可以增减 Agent，或对原有 Agent 进行重新配置来实现系统功能的扩展和重构。它侧重研究如何协调一组 Agent 的行为，即研究这组 Agent 为了联合采取行动或求解问题时，如何协调各自的知识、目标、策略和计划等。各个 Agent 是并行工作的。

多 Agent 系统是对社会的模拟，是针对不同的用途开发的不同功能的实体。多 Agent 系统主要特点有：(1)高内聚，指每个 Agent 能独立和自主地推理和规划并选择恰当的策略解决给定的子问题；(2)低耦合，指各 Agent 的目标和行为不受其它 Agent 成员的限制，它是独立自主的。

由于多 Agent 系统是由多个 Agent 基于一定的协调机制组成的自组织系统。Agent 的自主性和系统的协调机制使得多 Agent 系统在描述复杂系统方面具有优越的特点：

1、分布性：多 Agent 系统不仅在结构上是分布的，在逻辑上也是分布的，其中的 Agent 具有不完全的知识和分布的决策能力，计算也是异步进行的，因此多 Agent 系统非常适合于并行操作。多 Agent 系统可通过并行机制加速系统的运行。一个任务可以分解成若干个子任务，这些子任务可分别由不同的 Agent 完成。

2、适应性：多 Agent 可以在协调机制下通过交互和自学习适应环境的变化和不确定性。

3、开放性：Agent 无论从概念上还是从实现上都是一种封装模型，其内部结构和算法可以由不同人在不同时间和地点采取不同方法加以实现，通过标准的消息接口加入到多 Agent 系统中。

4、鲁棒性：对于外界的干扰，多 Agent 系统可通过 Agent 间的交互协调进行参数调整来保持系统的性能水平。

2.2.4 多 Agent 协作

多 Agent 系统中多个 Agent 之间相互协作[39]共同完成系统指定的工作，实现多 Agent 之间的协作有多种方法和机制，如结果共享模型、合同网和基于公告牌的合作规划系统等。

由 Smith 和 Davis 于 1981 年提出的结果共享模型对任务进行静态的分解与分配，在问题求解之前进行，各个单独的 Agent 不能独立完成任务，多个 Agent 之间通过交换结果实现协作的问题求解，并逐步获得最终的结果。

Smith 于 1980 年提出了合同网方法，在这个方法中，不需预先规定 Agent 的角色，当任意 Agent 无法独立完成其所负责的任务时，该 Agent 就将任务进行分解，并向其它 Agent 广播发送任务通知书，然后从返回的应答中选择最适合的 Agent，然后将子任务分配给它。

基于公告牌的合作规划系统[40]是在合同网协议的基础上发展而来的规划组织模型，其基本思想是某一 Agent 由于知识和能力限制不能生成相应规划以完成计划的任务时，通过公告牌发出规划生成请求，此时若环境中的空闲 Agent 则会根据自己的情况产生不完全的个体规划并将之发到公告牌，之后由发出请求的 Agent 将返回的个体规划融合生成比较适合的规划。

根据 Agent 之间相互依赖的关系，可将 Agent 之间的协作分为如下几类：

1、层次协作：高层 Agent 根据低层的 Agent 的处理结果对问题进行进一步的求解。

2、水平协作：分两种情况，(l)协作团队中的每个 Agent 不能独自求解全局问题，问题的解决需要通过多个 Agent 之间的协作来完成。(2)协作团队中的每个 Agent 都能独立解决问题，但多个 Agent 协作解决的话就能大大提高问题解的可信度。

3、循环协作：为了求得问题的解，各个 Agent 之间相互依赖，循环协作。

4、混杂协作：将层次协作、水平协作或循环协作类型结合起来进行协作。

在多 Agent 系统中，单个 Agent 不能解决整个系统的问题，因此协调多 Agent 系统中的各个 Agent 之间的行为非常重要，没有协调，多 Agent 系统分布解决问题的优势就无法实现。所以 Agent 之间协作的程度、系统的全局连贯性以及多 Agent 组织是否合理是衡量系统性能的重要指标[41]，关系到整个系统的智能性和灵活性。

2.2.5 多 Agent 间的通信方式

在多 Agent 系统中，多个 Agent 之间通过协作共同完成指定的任务，而各个 Agent 之间的协作是通过各种通信技术来实现的，Agent 之间通常采用以下几种方式进行通信：邮箱方式[42]、消息传递和黑板方式[43-46]。

1、邮箱方式

在邮箱方式中，参与通信的 Agent 之间需要建立一个邮件通道。一般情况下，邮件通道可以为多个 Agent 间的消息传输所共有。一个 Agent 欲向另一个 Agent 发送消息时，它可以将消息打包成邮件，并通过邮件通道发送到目标 Agent 的邮箱中。目标 Agent 可以按照一定的周期或者随机地访问它的邮箱，如果邮箱中收到了邮件，则它就可以取出邮件并对其进行处理。

2、消息传递

消息传递通信方式是参与交互的 Agent 之间建立通信通道，Agent 之间使用一组事先约好的格式和规则通过该通信通道进行消息传递。一旦在 Agent 之间建立了通信通道，那么处于通信通道中的双方 Agent 就可以进行双向、对等的通信。任何一方 Agent 既可充当消息通信的发送方，也可充当消息通信的接收方。由于通信通道归属于参与交互的双方 Agent，不能为其他 Agent 共享，一方 Agent 利用通信通道发出消息之后另一方 Agent 能很快地接收到，因而该通信方式具有较好的保密性和实时性。

3、黑板方式

在黑板通信方式中，多个人类专家或 Agent 专家协同求解一个问题，他们之间共享一个称作黑板的公共求解空间，都可以对该区域进行访问，如发布信息、处理结果和获取有用信息等。Agent 之间可以交换信息、数据和知识，Agent 可以在任何时候访问黑板，通过查看黑板寻找利用其知识经验求解问题的机会，当一个 Agent 发现黑板上的信息能够支持它求解时就开始求解并将结果记录在黑板上，新增加的信息有可能使其他 Agent 继续求解，重复这一过程直到问题解决。黑板模型是解决复杂知识处理系统的一种有效工具，模型结构如图 2.2 所示。

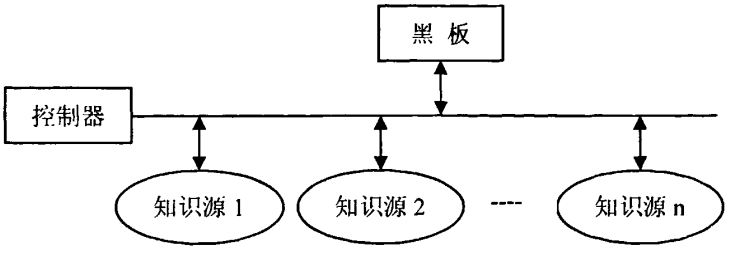

图 2.2 黑板模型结构

其中，①知识源：根据求解问题专门知识的不同将应用领域分为若干个相互独立的专家，这些专家称为知识源。每一知识源独立完成一种特定的任务或特定领域的任务。②黑板：即共享的问题求解工作区。在问题求解过程中，知识源不断地对黑板进行修改。知识源之间的通信和交互只能通过黑板进行。③控制器：根据黑板上的问题求解状态和各知识源的求解技能，依据某种控制策略，动态地选择和激活合适的知识源，使知识源能实时响应黑板变化。

2.3 Agent 技术在异常检测中的应用

目前，基于 Agent 的检测技术逐渐引起人们的注意。Agent 可以看作是在系统中执行某项特定检测任务的软件实体，通常以自治的方式在目标主机上运行，本身只受操作系统的控制。Agent 的独立性和自治性为系统提供了良好的扩展性和发展潜力。基于 Agent 的异常检测系统可以为保障系统的安全提供混合式的架构，综合运用误用和异常检测技术。

国际上关于 Agent 和 MAS 的研究非常广泛，基于 Agent 技术的异常检测系统[47]也获得了很大发展，在此期间，数据挖掘、免疫系统、基因算法等技术也渗透或融合到异常检测技术中，从而将异常检测技术的发展推向了一个新高度。

由 Purdue 大学 COAST 实验室开发的 AAFID 系统[48-49]通过多个相对独立的 Agent 检测异常活动。该系统是一种采用树形分层构造的 Agent 群体，最根部的是监视器 Agent，负责对全局的控制、管理和分析由上一层节点提供的信息，位于叶节点的 Agent 负责数据采集任务。位于中间层的 Agent 负责对底层 Agent 的控制，并对数据进行预处理操作，以提供给上层的监视器 Agent。

鉴于 Agent 具有很多良好的特性，AAFID 系统结构的扩展非常方便，多个 Agent 可以被分配成一个 Agent 群组，每个 Agent 执行各自相对独立的功能，通过协作共同完成复杂的检测任务。

EMERALD 系统借鉴了 IDES 和 NIDES[50]的经验，并将关注的重点从一系列主机转移到了整个网络。EMERALD 采用层次化的分布式 Agent 系统架构，以适应基于松散架构的大型网络环境。

基于统计学方法的异常检测技术是使用统计学的方法来学习和检测用户的行为。一个典型的系统是 NIDES 系统，NIDES 所使用的统计分析技术支持对每一个系统用户和系统主体建立历史统计模式。所建立的模式被定期地更新，可以及时地反映出用户行为随时间推移而产生的变化。检测系统维护一个由行为模式组成的统计数据库，每个模式采用一系列系统度量来表示特定用户的正常行为。模式所包含的各个向量每天都以指数因子形式衰减，同时将新的用户行为所产生

的审计数据嵌入到知识库中，计算出新的模式向量存储在知识库中。

NIDES 使用 Agent 完成数据提取和格式化功能，并提交给分析模快。统计分析组件通过学习用户的行为，完成异常检测功能。检测到的信息被 Resolver 过滤后，由 Archiver 组件存储或使用用户界面查看。NIDES 系统的维护比较方便，只要分析时所采用的度量能完成准确的鉴别，用户行为的改变相对应的度量能产生一致性的变化，保证行为模式的更新，那么对于那些异常行为，检测器就能够可靠并高效地完成检测任务。

美国 Columbia 大学提出的 JAM 系统利用 Agent 技术，将后向学习和分布式数据挖掘用于异常检测，在 JAM 的分布式 Agent 系统中，每个 Agent 运用数据挖掘，如分类，关联和序列分析等，对知识和行为进行建模和推理，系统设计包括两个核心组件：①本地检测 Agent，主要用来在一个单一的团体信息系统中学习怎样去检测异常；②后向学习系统，用来结合本地单个 Agent 学习的知识，进一步发掘有用信息。

美国 Lowa 州立大学发展的 MAIDS 系统是基于 Agent 的分布式异常检测系统。MAIDS 先采用软件故障树分析对一个异常建模，再使用有色 Petri 网将 SFT 模型转换为异常检测建模。接着用智能 Agent 技术来实现一个 CPN，包括一个从静态 Agent 到智能 Agent 的转换过程。

中国科学院提出一个基于多 Agent 的分布式异常检测系统的模型[51]。该模型采取了无控制中心的多 Agent 结构，每个检测单元都相对独立，该模型降低各检测单元间的耦合性，实现了分布式数据采集，并将异常检测和实时响应分别进行处理。

第三章 领域分析

本章阐述门诊医疗行为面临的问题，通过对检测主题的确定，给出系统的检测的流程，并进行用例建模。

3.1 问题分析

产生医保违规和欺诈行为的因素主要包括如下几个方面：①利益驱使。②监督管理不到位。③社会道德缺失。④法律制度建设滞后。

目前，医疗保险机构对定点医疗机构进行医疗行为的检测主要采取如下两种措施：

1、通过在信息系统中对支出数据进行筛选和分析，发现问题后再进行详细审查。检测手段主要是利用一些医疗常识、业务经验或相关政策对已发生的费用信息和医疗明细信息进行检测，从中发现异常信息，为行政执法提供依据。

2、医保部门采取一定的措施对医疗费用加以控制。如对医疗人次平均诊疗金额的限制，单病种诊疗金额的限制，增加病人自付费的次数和金额等。

这些措施都是比较粗放的控制措施，难以起到规范医疗行为、控制医疗欺诈的行为，甚至可能影响医院和病人的医疗质量和利益。究其原因：①由于参保人数的众多，就诊人次频繁，造成就诊信息呈几何级数增长。②通过靠人力来判断和分析就诊信息来实现及时、准确、无遗漏的费用监控几乎成为不可能的任务。③医保费用审核人员根据自己的经验对医疗数据进行检查，由于审核人员各自拥有不同的专业知识和经验，因而容易造成审核的标准不一致，审核的质量无法得到有力的保证。

3.1.1 问题

医疗门诊中经常出现的违规和欺诈行为可归结为如下几种：

1、超量配药。所谓超量配药是指违反门诊用药常规及联合用药规范而开出超剂量、超品种处方用药的行为。

2、分解处方。分解处方是指将就诊的大处方化整为零分解为若干个小处方，比如一种疾病本来只开一张处方，但却开成了三张处方，即分解成三张小处方。

3、重复门诊。重复门诊是指在一次门诊周期内多次享受门诊服务的行为，如对于普通感冒病例，一天内出现多次就诊的情况。

4、以药易药。以药易药是指违反规定，以可报销药品之名替换不可报销药品的行为。

5、多记多收。多记多收是指医生在开具处方时违反规定，将实际没有用到的药物也记录到处方中，多收取就诊费用。

3.1.2 医疗数据

医疗保险机构已经拥有保险业务数据库，其中包括了保人信息、就诊类别、就诊时间、就诊医疗机构、历次诊疗明细项目等信息。经分析，我们把医院和医保数据分为四类。

1、政策性数据

该类数据由医保机构传向医院，数据量小，很少变化，访问频繁。主要是各种数据划分线，报销比例等。

2、收费项目名目

该类数据由医院维护，医保机构审核，医院使用。数据量小，变化小，访问频繁。

3、参保人数据

此类数据由医保机构传向医院，数据量中等，有变化，但变化不大，访问频繁。主要是参保人员的相关信息。

4、参保人就诊数据

此类数据由医院传向医保机构，数据量大。主要是参保人的就诊明细情况。

3.1.3 检测主题

针对上述异常医疗行为及所拥有的医疗数据资源，确定系统需要对如下几个方面的用药时间进行检测：

1、门诊用药效-效相似关系检测

通过对医疗处方中药物的药效相似检测单方中存在的违规现象。

2、门诊用药周期与频次检测

在限定的用药周期内，对同类药物重复频次的检测。

3、门诊用药周期与数量检测

医疗中往往出现过量开药的情况，比如一次就诊，本来应该开一个疗程的药，但实际上却开了两个疗程的药。

3.2 系统分析

3.2.1 异常检测流程

根据异常检测确定的主题，系统的检测流程如图 3.1 所示。

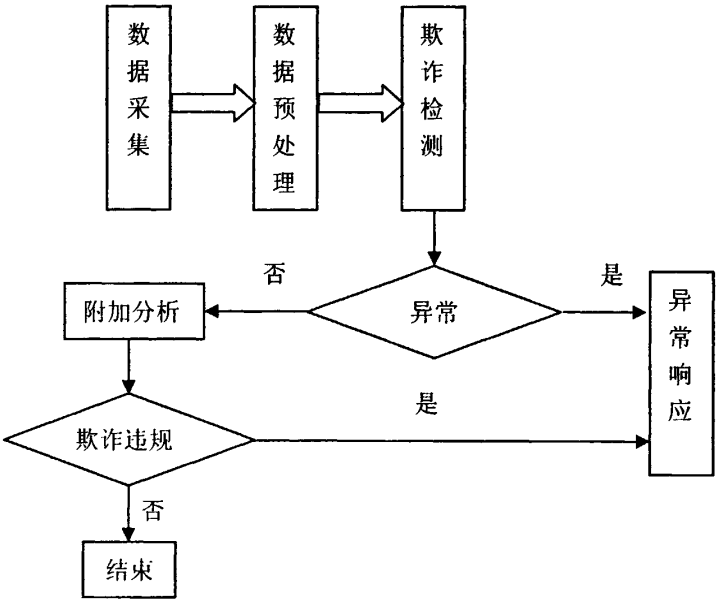

图 3.1 欺诈检测的检测流程

检测过程描述如下：

第一步：从医疗数据库中对医疗数据进行采集。

第二步：将采集到的数据进行预处理，如数据清洗等。

第三步：对预处理后的医疗数据采用相关的检测规则进行欺诈检测，如果数据检测结果是异常的，那么系统进行异常响应，若检测结果无异常，那么需要对数据进行进一步附加分析，以加强对医疗数据的审核工作。

第四步：结束。

3.2.2 用例分析

根据系统检测流程，下面给出医疗欺诈行为检测系统的主要用例图，以展现系统的主要功能。系统顶层用例图如图 3.2 所示。

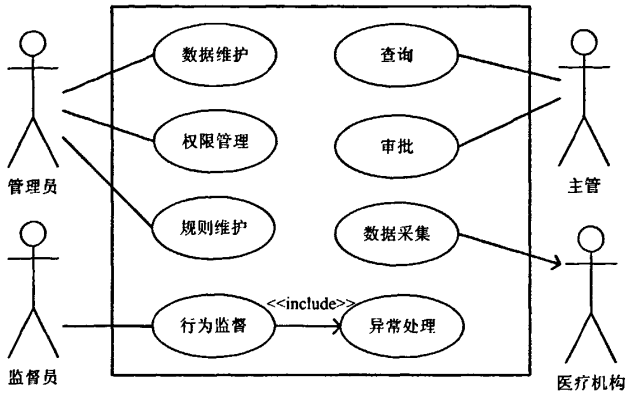

图 3.2 系统用例图

1、角色描述：

(1) 管理员：负责对系统数据和规则的维护，并对用户权限进行管理。

(2) 监督员：负责对医疗行为进行监督和检测，并对异常行为进行处理。

(3) 主管：对医疗行为进行查询和审批。

(4) 医疗机构：提供医疗数据。

2、用例描述：

(1) 数据维护：对医疗数据库进行管理和维护。

(2) 权限管理：对用户权限进行管理，不同的用户具有不同的权限。

(3) 规则维护：对系统的检测规则进行管理和维护。

(4) 行为监督：对医疗行为进行监督，如检测控制、结果展现等。

(5) 异常处理：对检测出的医疗异常行为进行处理。

(6) 查询：对系统信息进行查询，包括对异常信息、规则等的查询。

(7) 审批：对检测结果进行审批。

(8) 数据采集：从医疗数据库中提取医疗特征数据。

异常检测用例图如图 3.3 所示。

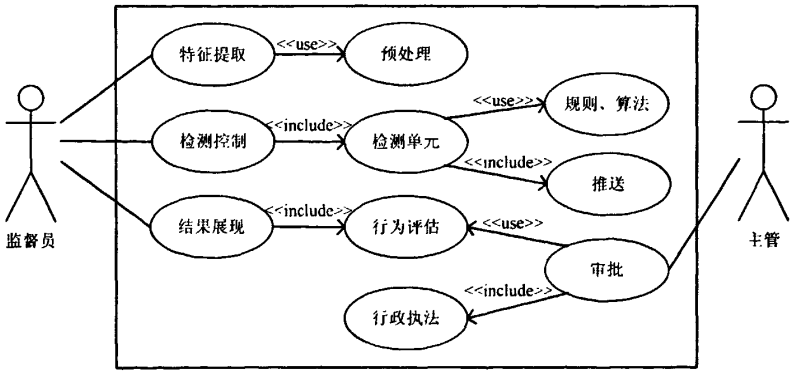

图 3.3 异常检测用例图

用例描述如下：

(1) 特征提取：提取医疗数据中的特征信息，便于对目标数据进行检测。

(2) 预处理：对从医疗机构采集到的医疗数据进行数据清理和归纳等操作，生成检测系统便于检测的数据。

(3) 检测控制：对欺诈检测进行管理和调度，对系统的检测过程和信息推送进行合理的控制。

(4) 结果展现：为用户提供良好的交互界面，便于用户对检测结果进行评估与管理。

(5) 审批：对检测出的异常信息进行行政执法等处理工作。

数据预处理用例图如图 3.4 所示。

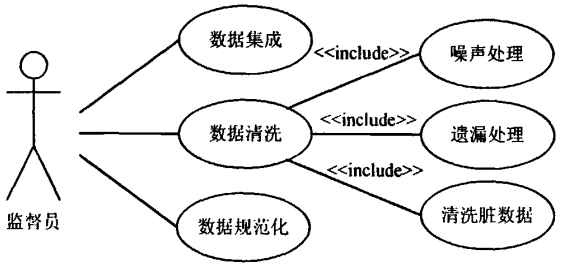

图 3.4 数据预处理用例图

用例描述如下：

(1) 数据集成：将多个数据源中的数据进行合并处理。

(2) 数据清洗：去除数据集合中的噪声数据和无关数据，处理遗漏的数据和清洗脏数据等。

(3) 数据规范化：对数据进行规范化处理，消除数据中存在的模糊关系。

3.3 本章小结

本章分析了异常医疗行为的成因及医疗保险机构所采用的检测方法，通过对问题域及其医疗数据资源的分析，总结了医疗保险门诊的违规欺诈行为，确定了检测系统的检测主题。围绕检测主题，给出了异常的检测流程以及系统主要的功能用例图。

第四章 基于多 Agent 医疗欺诈检测系统的设计

本章在前文对医疗欺诈行为及其检测流程分析的基础上，结合 Agent 技术，阐述基于多 Agent 医疗欺诈行为检测系统的体系结构，医疗数据预处理方法，进行了系统中各个组成模块的详细设计，并描述了系统的工作机制。

4.1 系统体系结构

根据前面章节对系统需求的分析和医疗欺诈行为问题域的特点，可将系统划分为如图 4.1 所示的层次结构。

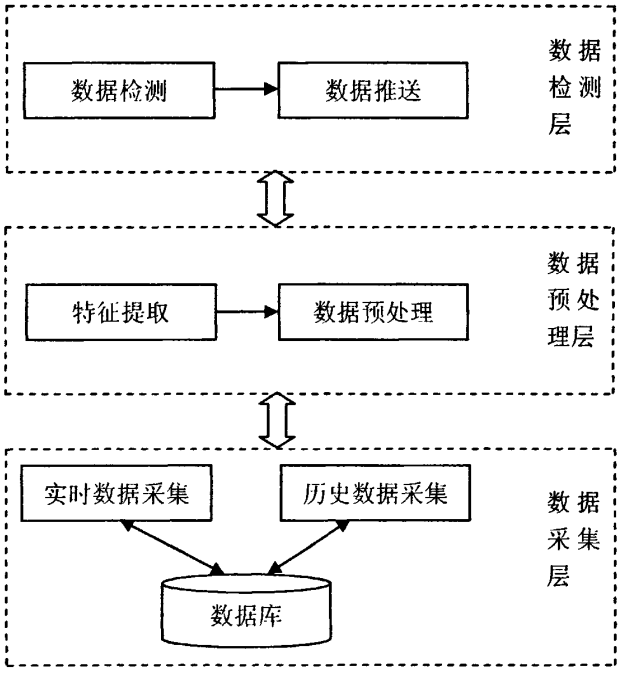

图 4.1 系统结构图

该结构分为三层：数据采集层、数据预处理层、数据检测层。

数据采集层通过前台 Agent，响应定点医疗机构的医疗事件，完成医疗数据采集和传送工作。

数据预处理层负责将数据采集层采集到的医疗数据进行预处理，完成医疗数据的特征提取，为异常检测工作做好准备。

数据检测层主要是对预处理后规范化的的医疗数据进行检测，并将检测结果推送给相关的监管人员。

4.2 数据预处理

数据预处理阶段主要是获取医疗数据，并根据背景知识中的约束规则对数据进行检查，通过清理和归纳等操作，生成检测系统便于检测的目标数据。数据预处理按如下步骤进行。

1、数据集成（Data Integration）

数据集成主要是将多个文件或多个数据库运行环境中的异构数据进行合并处理，解决语义的模型性。该部分主要涉及数据的选择、数据的冲突问题以及不一致数据的处理问题。在数据集成过程中，着重需要考虑解决字段的定义、数据类型的选择等。

医院的医疗费用数据，数据比较分散，分布在不同的数据表中，数据集成过程可以有效的将这些数据进行整合。

对医疗费用数据的集成，通过建立数据表，从医疗活动所产生的原始数据中抽取出相关的检测的数据。比如从患者表中提取患者的基本信息如性别、年龄、身份证号、职业等，从医疗记录中提取医疗时间、医疗用药，费用等，从疾病登记表中提取疾病名称、治疗情况、治疗结果等。通过对类似数据的抽取集成到统一的数据表中。

2、数据清洗（Data Cleaning）

数据清洗过程要去除原数据集中的噪声数据和无关数据，处理遗漏的数据和清洗脏数据，对重复数据和缺值数据进行处理，并完成对相关数据类型的转换工作。

针对门诊诊疗的特点，仅仅抽取检查、用药两个部分作为门诊行为。考虑到用药行为是通过机体发挥药理效应，所以数据处理过程中需要构建药物及药理效应之间的相互联系，以便于通过药-效角度，挖掘用药行为的相似性。处理后数据构成如下关系：

```
门    诊 = 病例 ID，日期，描述；
门诊诊治 = 病例 ID，0{行为}n；
行    为 = 项目，数量，金额。
```

药物及药理效应之间的相互联系如下：

```
药理 = 药理 ID，效名；
项目 = 项目 ID，0{药理 ID，权值}n。
```

3、门诊数据规范化

经过清理后的行为数据仍然存在药物数量与使用周期间的不完整的关系，如果对这含有不准确关系的数据进行后续过程的处理，会影响检测的准确性。针对这种现象，需要将数据规范化，消除模糊关系。

数据规范化按下述定义的规则处理：

```
用法 = 用法 ID，性质，用量；
性质 = 缺省 | 指定；
项目 = 项目 ID，用法 ID，周期。
```

每一项目的周期由用法性质确定，处理规则如下：

```
对每一项目
If 性质 = 缺省 then
  周期 = 用量
Else
  周期 = 数量/用量
Endif
```

4.3 Agent 设计

4.3.1 Agent 形式化描述

系统中 Agent 的基本结构如图 4.2 所示。

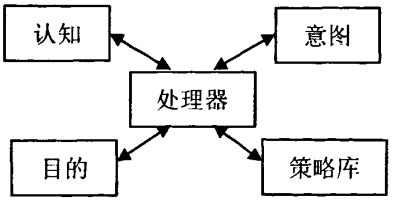

图 4.2 Agent 基本结构

认知是 Agent 对于所处环境的认知和理解，目的表示所希望达到的目标；意图则是描述了 Agent 为达到目标而采取的行为步骤；策略库存储了一些预先设计的计划，使 Agent 能方便地处理其意图；处理器则主要负责对 Agent 的控制。

在基于多 Agent 医疗欺诈行为检测系统的研究与设计中，参考文献[52]建立对系统中 Agent 功能和行为的定义和抽象，并进一步完成对各种 Agent 的结构、关系、行为、交互等的定义和描述。

根据对多 Agent 系统中 Agent 功能的分析，对有关的概念进行如下定义：

定义 1：动作（Action）：由 Agent 执行的能造成 Agent 自身或其所处环境的状态产生变化的最基本的执行单元。

定义 2：活动（Activity）：一个由 Agent 发起或由触发事件触发的多个动作及其相关约束组成的有序序列的集合。

定义 3：行为（Behavior）：由 Agent 的所有活动所组成的集合称为行为。

定义 4：序列（Sequence）：集合中所有元素的有序排列称为一个序列。序列可以由合式公式的形式来表示。在此做如下规定：设 a，b 为两个元素，则 a→b 表示 a 与 b 是先后关系，且 a 在 b 之前；a∨b 表示“a 或 b”；a∧b 表示“a 与 b 并列”。

定义 5：状态域（State Field）：对系统的某种特征属性的描述称为状态域，系统在某状态域上的所有可能的取值范围称为系统在该状态域上的值域。

定义 6：状态空间（State Space）：设一个系统有 N 个状态域，那么这些状态域及其值域构成了该系统的 N 维状态空间，系统的状态可以用状态空间上的 N 维向量来表示，如下所示：

$S_0=(r_1,r_2,\ldots,p_N)^T$，其中 $r_1,r_2,\ldots,r_N$ 是状态域变量。

定义 7：状态转移函数：系统的状态转移函数是如下所示的一个映射：

$\mathfrak{L}$：$S\rightarrow S_1$

其中，S、S_1 是系统的状态空间。它表明一个系统的状态发生变化的过程。设系统有 N 个状态域，则它也可以表示为一个 N 元方程：

$\mathfrak{L}=f(S_0)=f(r_1,r_2,\ldots,r_N)$，其中 $r_1,r_2,\ldots,r_N$ 是状态域变量。

对一个系统而言，状态域的取值有时是数值型的、有时却是一种描述性的，在此用一阶谓词公式来描述 Agent 的状态，将 Agent 的每个状态域定义一个值为真的谓词原子公式，描述如下：

谓词（项 1，[项 2]，[项 3]，…，[项 n]）

项的值可以是数值、名称或常量等。比如，可以设 AgentName(tqhagent)表示 Agent 名称的状态域，当前的状态值为 tqhagent。

1、动作的表达

根据前面所做的定义，可将动作表示为如下所示的一个三元组：

$Act=<S_s,\ O_p,\ S_e>$

$<S_s:\ \{r_1,r_2,\ldots,r_N\}^T>$

$<O_p:\ f(r_1,r_2,\ldots,r_N)>$

$<S_e:\ \{r_1,r_2,\ldots,r_N\}^T>$

其中，S_s 是起始状态，S_e 是终止状态，O_p 是状态转移函数（或称为动作函数），它的描述也可以采用其它的操作描述语言。对于顺序执行的多个动作，用

动作序列来表达。动作序列是由多个动作组成的有序序列，它可以表示如下：

$Acts=<A，Q>$

其中 A 是一个有限的动作集合，这里用 A_0 表示 Agent 内部所有动作的集合，则 A_0 包含 A；Q 是表示动作的有序排列的合式公式，用 A_s 表示 Agent 所有动作序列的集合，一个动作序列可以表示如下：

$Acts_1=<A_1，Q_1>$

$A_1=<Act_1，Act_2，Act_3，Act_4>$

$Q_1=Act_1\rightarrow Act_2\rightarrow (Act_3\vee Act_4)$

2、活动的表达

根据前面对活动的定义，活动可以表示为如下所示：

$Actvt=<Iexp，Evt，Acts>$

其中 Iexp 是对活动目的的描述。Evt 是事件的集合，它表示环境对 Agent 的触发事件，可以是空集，此时表明该活动是 Agent 自身发起的。Acts 是动作序列，Acts 属于 A_s。一个活动可以用如下式子来表示：

$Actvt_1=<Iexp_1，Evt_1，Acts_1>$

$Iexp_1=\{Target\ Description\}$

$Evt_1=<evt_1：Event\ Description>$

$Acts_1=<A_1，Q_1>$

$A_1=<Act_1，Act_2，Act_3，Act_4，Act_5>$

$Q_1=((Act_1\rightarrow Act_3)\vee Act_2)\rightarrow Act_4\rightarrow Act_5$

3、Agent 行为的描述

Agent 的行为特征可以由以下形式来表达：

$B=<I，Evt，A_0，V>$

其中，I 是 Agent 所有活动目的的集合，Evt 是 Agent 能够感知的所有事件的集合，A_0 表示 Agent 内部所有动作的集合，V 是所有活动的集合。

4.3.2 前台 Agent

前台 Agent 主要用来从环境中采集数据，并将采集到的医保数据由特征 Agent 处理后发送到共享黑板中。

前台 Agent（ForeAgent）主要的行为类（Behavior）由实时数据采集（RTimeDataGather）和历史数据采集（HistDataGather）组成，其关联关系如图 4.3 所示。

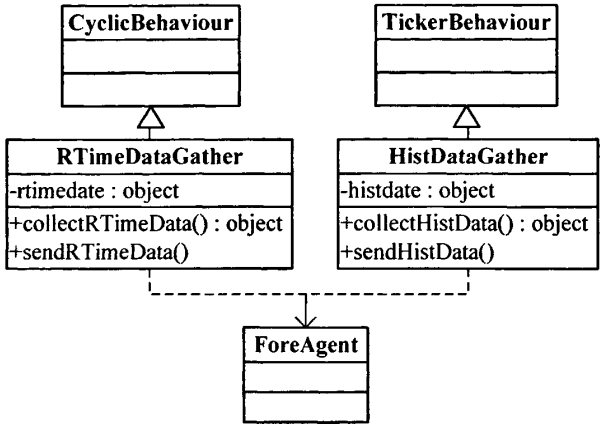

图 4.3 前台 Agent 的行为图

RTimeDataGather 由 CyclicBehaviour 继承而来，主要用来对实时医疗数据信息进行数据采集工作，collectRTimeData()函数循环执行对医疗实时数据进行采集，sendRTimeData()函数将采集到的数据发给特征 Agent。

HistDataGather 由 TickerBehaviour 继承而来，主要用来对本地数据库中的历史医保信息进行数据的采集工作，collectHistData()每隔一段时间间隔对历史数据进行数据采集，并由 sendHistData()将采集的数据提供给特征 Agent 进行处理。

4.3.3 特征 Agent

特征 Agent 主要是从前台 Agent 得到的数据中获取医保数据的特征信息，并将其进行预处理之后放入共享黑板中。特征 Agent 主要完成的工作如下：

1、从前台 Agent 获取特征信息。

2、对获取的特征信息进行数据清洗等预处理工作。

3、将预处理之后的特征信息放入共享黑板。

特征 Agent（CharacterAgent）的主要的行为类（Behaviour）为特征提取（CharacterGet），数据预处理（DataPretreat）。如图 4.4 所示。

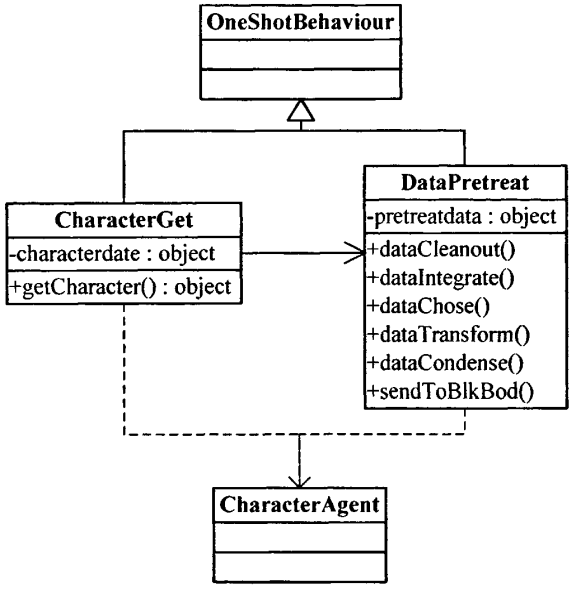

图 4.4 特征 Agent 的行为图

CharacterGet 跟 DataPretreat 都继承自 OneShotBehaviour，前者主要从前台 Agent 获取特征信息，后者主要将采集到的医保数据如姓名，性别，症状，处方，就诊金额，就诊时间等信息进行数据清洗、集成、抽取、变换等预处理操作，并由 sendToBlkBod()将预处理后的数据写到共享黑板中。

4.3.4 决策 Agent

决策 Agent 是系统功能实现的关键，是欺诈检测的裁断者，它根据系统的实时情况和相应规则进行欺诈检测并对异常情况进行相应的处理。其结构如图 4.5 所示。

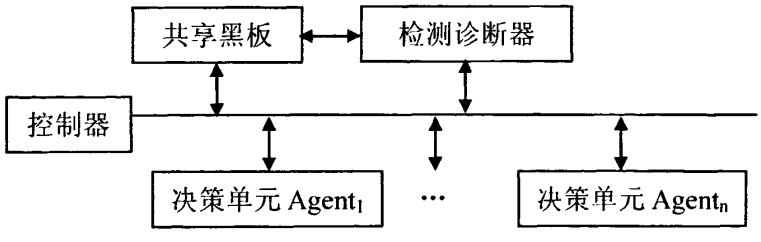

图 4.5 决策 Agent 结构

共享黑板是一种平行共享的数据结构，它可以存储任务、规则和部分结果等。它为决策单元提供资源共享和集体商讨的空间，是决策单元之间间接通信的方式。检测诊断器检测共享黑板中的信息，并将黑板中信息的检测任务分配给各个决策单元 Agent，由各决策单元 Agent 按照各自的检测规则对所负责的信息进行

检测，完成相应的检测任务。各决策单元 Agent 一旦发现异常行为，则将异常的信息保存到异常信息库中。对于很明显的异常信息，决策 Agent 可以直接将这些异常信息交由推送 Agent，由推送 Agent 将异常数据推送给用户。

在此应用黑板模型结构解决系统中多个 Agent 的协作问题，实现多信息源的交流与共享，为多 Agent 提供通信支持。

决策 Agent（DetectAgent）的行为说明如下：

DetectAgent 主要有如下几个行为类（Behaviour）：负载管理（LoadManage），状态检测（StateDetect），数据检测（DataDetect），如图 4.6 所示。

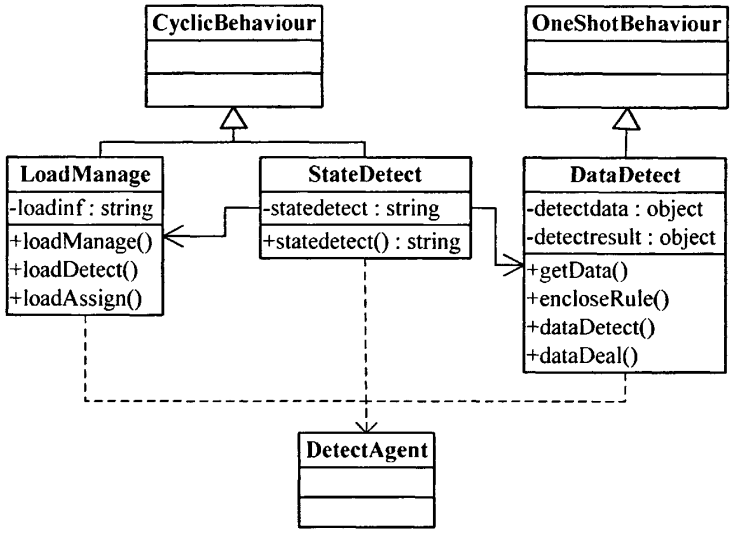

图 4.6 决策 Agent 的行为图

LoadManage 和 StateDetect 由 CyclicBehaviour 继承而来，主要用来对各决策单元 Agent 进行管理控制，保证负载平衡，由 loadAssign()将任务分配给决策单元 Agent，以便各个决策单元 Agent 能协作完成欺诈检测任务。DataDetect 继承自 OneShotBehaviour，主要对黑板中的数据进行初步检测，检测时，由 getData()从共享黑板中得到数据，encloseRule()获取检测规则，dataDetect()按照相应的检测规则对数据进行检测，并由 dataDeal()对检测结果进行处理。

根据前面对于 Agent 的形式化描述的介绍，简要说明决策 Agent 主要行为的描述方法，如下所示：

<Agent:name=DetectAgent>

<Intention>

I_1=<Name=DataDetect Expression=数据检测>

I_2=<Name=LoadManage Expression=负载管理>

I_3=<Name=StateDetect Expression=状态检测>

</Intention>

<Action>

Act_1=<Name= DateValidate Expression=验证数据有效性>

Act_2=<Name= Discard Expression=抛弃无效数据>

Act_3=<Name=Format Expression=格式化数据>

Act_4=<Name=DetectData Expression=检测数据>

Act_5=<Name=Skip Expression=略过正常数据>

Act_6=<Name=SaveData Expression=将数据存到异常信息库>

Act_7=<Name=SdToSendAgent Expression=向推送 Agent 报告异常信息>

Act_8=<Name=ValidateVacant Expression=验证决策单元 Agent 是否空闲>

Act_9=<Name=Assign Expression=分配检测任务>

Act_{10}=<Name=NewDetectUnitAgent Expression=新建一决策单元 Agent>

Act_{11}=<Name=AsnNewTask Expression=给新决策单元 Agent 分配任务>

</Action>

<Event>

evt_1=<Name=FindNoData Expression=发现未检测数据>

</Event>

<Activity>

$Actvt_1$=<Name=DataDetect Expression=数据检测>

$Actvt_2$=<Name=LoadManage Expression=负载管理>

</Activity>

<Behavior>

$B=<I,Evt,A_0,V>$

$I=<I_1;I_2;I_3>$

$Evt=<evt_1;evt_2>$

$A_0=<Act_1,Act_2,Act_3,Act_4,Act_5,Act_6,Act_7,Act_8,Act_9,Act_{10},Act_{11}>$

$V=<Actvt_1;Actvt_2>$

$Actvt_1=<Iexp_1,Evt_1,Acts_1>$

$Iexp_1=<I_1>$

$Evt_1=<evt_1>$

$Acts_1=<A_1,Q_1>$

$A_1=<Act_1,Act_2,Act_3,Act_4,Act_5,Act_6,Act_7>$

$Q_1=Act_1\rightarrow(Act_2\vee(Act_3\rightarrow Act_4\rightarrow(Act_5\vee(Act_6\wedge Act_7))))$

$Actvt_2=<Iexp_2,Evt_2,Acts_2>$

$Iexp_2$=<I_2,I_3>

Evt_2=<evt_1>

$Acts_2$=<A_2,Q_2>

A_2=<Act_8,Act_9,Act_{10},Act_{11}>

Q_2=Act_8→(Act_9∨(Act_{10}→Act_{11}))

</Behavior>

</Agent>

4.3.5 决策单元 Agent

决策单元 Agent 主要有两个功能模块组成：触发器，检测器，如图 4.7 所示。各个决策单元 Agent 按照自己的规则对决策 Agent 所分配的数据进行违规欺诈检测，在发现违规异常信息时则将异常信息保存到异常信息库中。

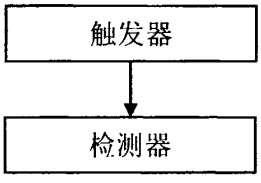

图 4.7 决策单元 Agent 结构

决策单元 Agent（DetectUnitAgent）的行为说明如下：

DetectUnitAgent 主要的行为类（Behaviour）是触发（UnitTrigger）和检测（UnitDetect），如图 4.8 所示。

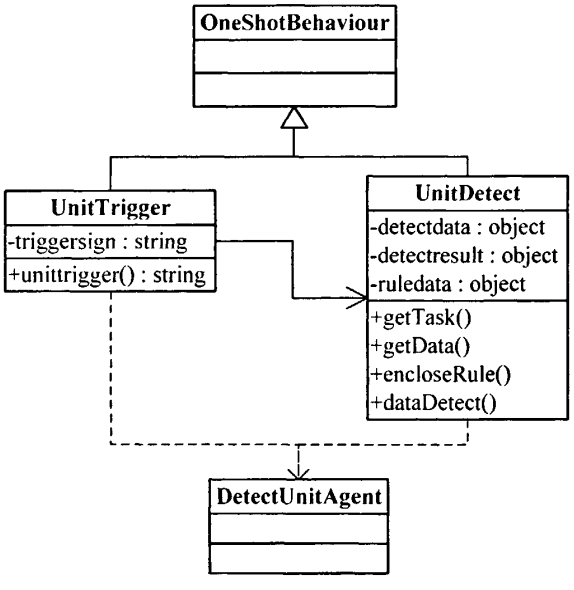

图 4.8 决策单元 Agent 的行为图

UnitTrigger和UnitDetect继承自OneShotBehaviour。它们随时等待决策Agent给自己分配检测任务，一旦 unittrigger()得到检测指令，则由 getTask()对检测指令进行响应，并由getData()从共享黑板中获得要处理的数据，通过encloseRule()获取检测规则，最后通过 dataDetect()对指定的数据进行检测，如果检测到有异常数据时，则对异常数据做相应处理，如做异常标记等，并将异常信息保存到异常信息库中。

4.3.6 推送 Agent

推送 Agent 主要是将异常信息推送给管理人员。推送 Agent 的内部结构如图4.9 所示。

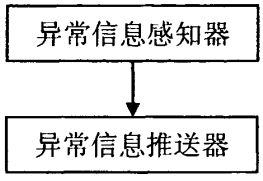

图 4.9 推送 Agent 结构

异常信息感知器周期性（如每周五上午八点半）的对异常信息库进行检测，如果发现还没有推送给用户的欺诈异常信息，则由异常信息推送器将该欺诈异常信息及时推送给相关人员，并对该信息做好已推送标记。相关人员按照自己所拥有的权限对异常信息完成后续的处理工作。

推送 Agent（SendAgent）的行为说明如下：

SendAgent 主要的行为类（Behaviour）是异常信息感知（FraudInfFeel）和异常信息推送（FraudInfSend），如图 4.10 所示。

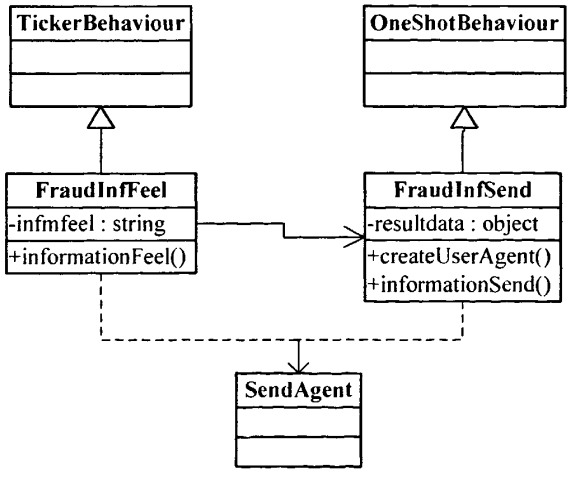

图 4.10 推送 Agent 的行为图

FraudInfFeel 继承自 TickerBehaviour，它周期性的检测异常信息库，一旦发现异常信息库中存在未推送的异常信息，则立即通知异常信息推送器将欺诈信息按照类别推送给相应的管理者。FraudInfSend 继承自 OneShotBehaviour，当收到来自 FraudInfFeel 的推送指令时，则将异常信息先保存至历史异常库中，历史异常库专门用来保存已经推送给用户的异常信息，然后再将异常信息推送给用户进行处理。

推送 Agent 主要行为的形式化描述如下：

<Agent:name=SendAgent>

<Intention>

I_1=<Name=FraudInfFeel Expression=异常信息感知>

I_2=<Name=FraudInfSend Expression=异常信息推送>

</Intention>

<Action>

Act_1=<Name=DateExist Expression=检测异常数据是否存在>

Act_2=<Name=BeginWait Expression=开始新的等待>

Act_3=<Name=GetDataFromDatabase Expression=从数据库提取数据>

Act_4=<Name=Format Expression=格式化数据>

Act_5=<Name=FraudInfSend Expression=将数据发给用户 Agent>

Act_6=<Name=DictateValidate Expression=检验指令的有效性>

Act_7=<Name=RefuseExec Expression=拒绝执行>

</Action>

<Event>

evt_1=<Name=GetForeAgentData Expression=时间周期已到>

evt_2=<Name=GetDetectAgentData Expression=收到决策单元 Agent 数据>

</Event>

<Activity>

$Actvt_1$=<Name=RuleCreate Expression=历史数据推送>

$Actvt_2$ =<Name=FraudInfSend Expression=实时数据推送>

</Activity>

<Behavior>

B=<I,Evt,A_0,V>

I=<I_1;I_2>

Evt=<evt_1;evt_2>

A_0=<Act_1;Act_2;Act_3;Act_4;Act_5;Act_6;Act_7>

```
V=<Actvt1;Actvt2>
  Actvt1=<Iexp1,Evt1,Acts1>
    Iexp1=<I1,I2>
    Evt1=<evt1>
    Acts1=<A1,Q1>
      A1=<Act1,Act2,Act3,Act4,Act5>
      Q1=Act1→(Act2∨(Act3→Act4→Act5))
  Actvt2=<Iexp2,Evt2,Acts2>
    Iexp2=<I2>
    Evt2=<evt2>
    Acts2=<A2,Q2>
      A2=<Act4,Act5,Act6,Act7>
      Q2=Act6→(Act7∨(Act4→Act5)
</Behavior>
</Agent>
```

4.3.7 共享黑板

共享黑板是为了多个 Agent 之间能够共享数据和交互而设立的一个公共数据区，例如一个 Agent 对共享黑板上的数据进行了修改，环境中的其它 Agent 可以直接访问这些数据，并对数据进行操作，也就是说各个 Agent 都能对共享黑板中的数据进行访问，黑板中的数据对各个 Agent 而言是共享的，从而便于多 Agent 之间的协作。其主要功能：

1、存放特征 Agent 获取的医疗特征信息。

2、存放检测过程中的中间结果，以便为各个 Agent 的检测提供方便。

3、存放各个功能 Agent 的信息，以及它们各自所能完成的检测任务等，如名称、能力、状态、位置等信息。

4、存放各个决策单元 Agent 的负载情况，以便于决策 Agent 为系统的稳定运行提供决策支持。

5、激活各个功能 Agent，以便进行医疗数据的检测工作。

```
public class BlkBod{
  public int informSignID;//黑板数据标记 ID，便于生成新的唯一 ID 号
  public ArrayList dataInfList;//存放黑板数据信息
  ……
```

```
    public bool addInf(BlkBodInf inf);//增加黑板数据信息
    public bool deleteInf(BlkBodInf inf); //删除黑板数据信息
     public bool modifyInf(BlkBodInf oldinf,BlkBodInf newinf);//修改数据信息
    public BlkBodInf getInf(int informID);//获取黑板数据信息
    public int getNewID();//获取新的唯一 ID，用于标识黑板数据信息
    public string getAgentName(String task);//获取能完成任务的 Agent 名称
    ……
}
public class BlkBodInf{
    public string dataSource;//数据来源
    public string dataDist; //数据目的地
    public int dataType; //数据类型
    public int informID;//标示黑板数据信息的 ID
    public string dataContent;//黑板数据信息的内容
}
```

基于多 Agent 医疗欺诈行为检测系统中各决策单元 Agent 之间相对独立，每个决策单元 Agent 表示各自欺诈检测任务所需的知识和规则，并专注于各自的检测任务。

4.4 系统工作机制

根据前面对各 Agent 的设计，基于多 Agent 医疗欺诈行为检测系统的工作机制如图 4.11 所示。

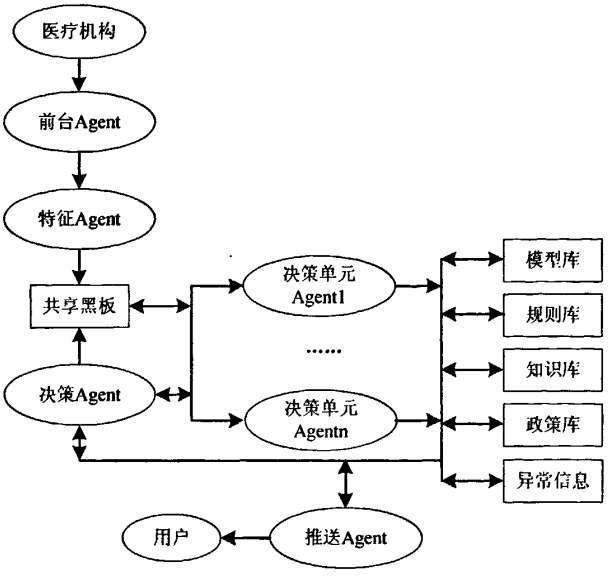

图 4.11 系统工作机制

各个 Agent 通过共享黑板协作完成医保欺诈检测任务，系统的工作机制如下所述：

1、前台 Agent 从医疗机构获得门诊医疗相关数据。

2、特征 Agent 从前台 Agent 获得门诊相关数据的特征信息，并存放到一个缓冲区（共享黑板）中。

3、决策 Agent 检查共享黑板中的信息，将相关数据根据政策库等里面的相关规定分类指派给各决策单元 Agent，并处理好负载平衡问题。由各决策单元 Agent 对共享黑板中的数据进行欺诈检测，每个决策单元 Agent 负责相关规则的检测任务，一旦发现异常信息，则将其保存到异常信息库中。

4、推送 Agent 定期检测异常信息库，当发现还没有推送给用户的异常信息记录时，则将其推送给相关用户。由用户决定对异常数据进行最后的处理，拥有不同权限的用户对异常信息具有不同的操作权限。

共享黑板保存着各个 Agent 的负载情况，决策 Agent 在分配给决策单元 Agent 检测任务前，首先查看黑板上各决策单元 Agent 的负载情况，如果所需的决策单元 Agent 相对空闲，则分配相关任务，否则就等待或由决策 Agent 新建一个决策单元 Agent 来执行新的检测任务。

4.5 本章小结

本章详细介绍了系统的体系结构和各个组成部分及其工作机制，对各个组成

部分的功能及其相互关系进行了形式化描述，并对系统中的各个功能模块进行了分析和设计。

第五章 原型系统的实现

本章在JADE平台上，进行了基于多Agent医疗欺诈行为检测系统的原型系统的开发，对系统中的各个功能Agent进行实现，并以效-效相似性检测为例，将改进的SimRank算法用在对医疗行为的欺诈检测中。

5.1 欺诈检测系统架构

基于上一章所进行的系统结构及主要Agent的设计工作，原型系统的技术方案如图5.1所示。

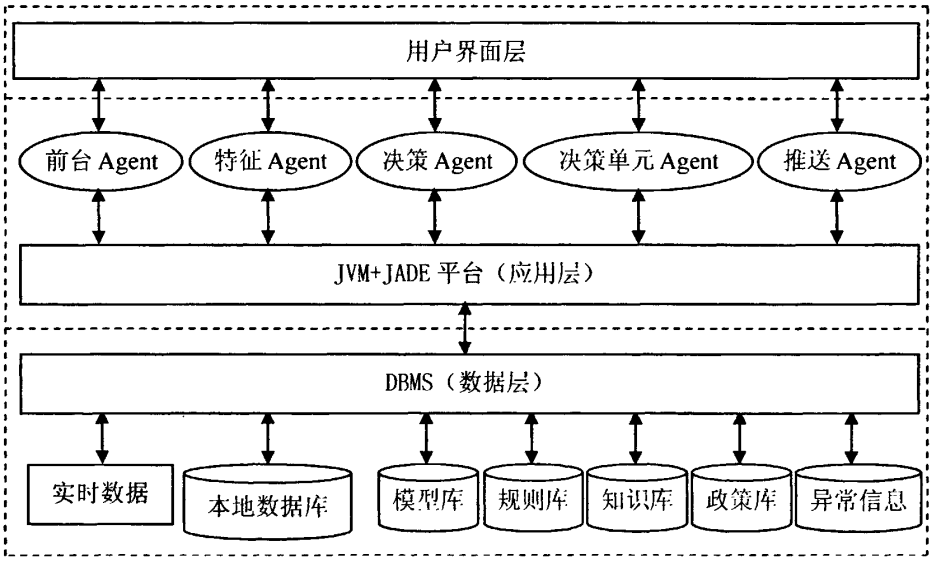

图 5.1 MAFDS 系统架构

该结构分为三层，由界面层、应用层及数据层组成。其中，用户界面层主要为用户提供良好的交互界面。

本地数据库与规则库等为数据层，本地数据库保存保险业务门诊数据，规则库保存了医疗欺诈检测的各种相关规则，本文所使用的数据系统为Sql Server2000。

JVM、JADE平台与各功能Agent组成了应用层，提供了对系统中各个Agent的创建、派遣，召回和销毁等管理功能，以及提供了多个Agent之间进行通信的功能。前台Agent、特征Agent、决策Agent、决策单元Agent和推送Agent等功能Agent通过协作共同完成系统的欺诈检测任务。

5.2 系统开发平台

1、JADE 平台

JADE[53]全称为 Java Agent DEvelopment Framework，是由 TILAB 开发的目前使用最为广泛的兼容 FIPA[54](Foundation for Intelligent Physical Agents)标准的 Agent 平台[55]。它是一个独立的 JAVA 的 API 包，通过确保 Agent 所必需的一些底层机制和服务，为开发基于 Agent 的系统提供便利的环境。它包含了两个主要的产品：一个是与 FIFA 兼容的 Agent 平台；另外一个是 Java Agent 的开发工具包[56]。作为一种 Agent 中间件，JADE 可以用来实现一个 Agent 平台和开发框架，它提供了很多基础类以及调试和配置 Agent 的工具，包括一套图形界面工具来管理和监控 Agent 的配置参数和状态[57]。JADE 是一个开放和分布式的软件，非常适合开发基于多 Agent 的系统。

由 FIPA 定义的标准的 Agent 平台模式由以下几个部分组成：

Agent 管理系统(AMS)是负责监督管理对 Agent 平台的访问和使用的 Agent，在一个单独的平台上只能有一个 AMS。AMS 提供白黄页服务以及生命周期服务，它保留了一个 Agent 标识符目录(AID)和 Agent 状态信息。每个 Agent 必须在 AMS 注册，以获得一个有效的 AID。

目录服务(DF)提供默认的黄页服务，Agent 可以凭此找到为自己提供服务的其它 Agent。

消息传输系统，又叫做 Agent 通信通道(ACC)，是控制平台内所有的信息交换，包括与远端平台进行信息交换的软件。

当一个 JADE 平台启动的时候，AMS 和 DF 就自动被建立了，同时 ACC 模块允许消息进行传输。

JADE 运行时的每一个实例都叫做 container，因为用它包含着各个 Agent。所有 container 的集合就是平台(Platform)。一个完整的 Agent 平台体系结构是由几个相关联的 Agent 容器组成。每台主机只运行一个 Java 虚拟机(JVM)，每个 JVM 都是 Agent 的容器，为 Agent 提供运行时的环境。在不同的 JVM 之间采用 Java RMI 通信，同一 JVM 内的通信采用事件调用的方式。每个平台中都有一个前端容器，该容器代表着本地平台与其它平台交互，容器内运行着管理 Agent(AMS)。

2、JADE 的特性

Agent 要执行一个任务就需要通过行为类来表示。在 JADE 中，一个行为必须从其父类 jade.core.behaviours.Behaviour 派生。在具体的执行过程中，利用 Agent 类的 addBehaviour 方法向 Agent 中加入实例化的行为。行为可以在需要的时候

随时加入到 Agent 中。每个从 Behaviour 派生的行为类必须实现 action()和 done()方法。action()方法定义了一系列 Agent 执行的操作，而 done()方法表明行为是否结束完毕。如果 done()方法返回的值为真，那表示这个行为已经执行完毕，可以从 Agent 的行为池中移除。一个 Agent 可以同时执行多个行为。一个行为的 action()一旦被调用，就会一直运行，直到返回。在 JADE 中 Agent 可以执行如下几种类型的行为：

(1) jade.core.behaviours.OneShotBehaviour：只执行一次，其 done()返回真。

(2) jade.core.behaviours.CyclicBehaviour：该行为将一直循环执行下去，其 done()返回假。

(3) 普通行为：满足条件时终止。

当一个 Agent 行为切换时，他的状态不包含任何堆栈信息，因此可以对它进行快照操作。这使得一些重要的高级特性得以实现，比如保存 Agent 的状态到永久存储器中以备以后使用或者把它传送到其他容器进行远程执行。

一个 Agent 线程的执行路径如图 5.2 所示。

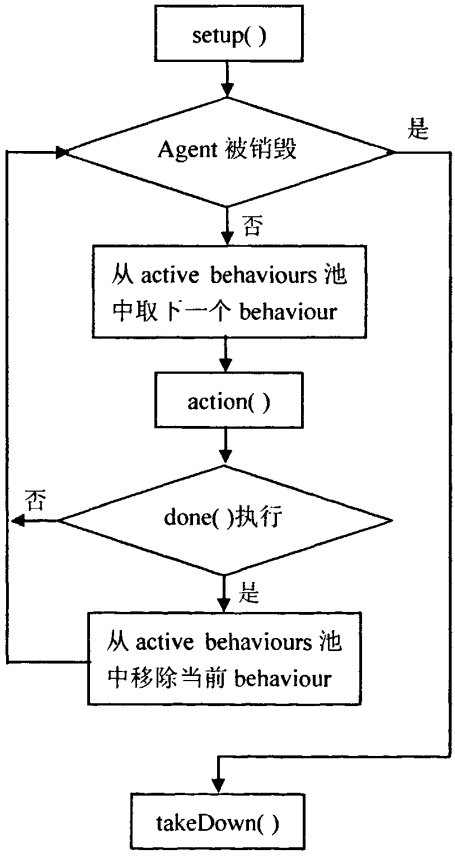

图 5.2 Agent 线程执行路径

首先，每个 Agent 都要执行 setup()方法进行初始化，一般在此通过调用

addBehaviour()函数添加 Agent 的各种行为到"行为池"（pool of active behaviours）中。Agent 通过内部的线程调度机制轮流执行行为池中的行为（Behaviour），即执行各 Behaviour 中的 action()方法。每次执行完 action()方法，再调用 done()方法判断这个行为是否结束：如果 done()方法的返回值为真，表明这个行为已经结束，则可以将这个行为从行为池中移除，否则将这个行为放回行为池中继续等待调度。在进行下一个行为的轮流调度前调用 doDelete()方法判断是否应该结束本 Agent，如果是，则调用 takeDown()方法销毁整个 Agent，完成最后的清除操作。

3、JADE 平台 Agent 间的通信

JADE 为 Agent 之间的通信提供了很大的便利，它采用异步消息传递方式来实现多个 Agent 之间的通信。每个 Agent 都有一个自己的消息队列，当有其它的 Agent 需要与其通信时，JADE 通信机制就把相应消息发送到其消息队列中，如图 5.3 所示。当消息队列中出现消息时，相应的 Agent 就会得到通知，此时就可以由编程人员决定是否对消息做出相应的操作。

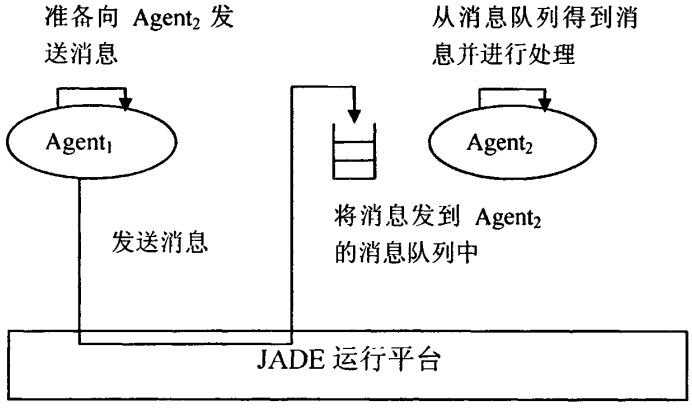

图 5.3 JADE 通信

目前，国际上比较流行的 Agent 通信语言是 KQML(Knowledge Query and Manipulation Language)。它定义了一组 Agent 之间进行信息传递的标准语法和动作，提供了一套标准的 Agent 通信原语，从而使得使用这种语言的 Agent 之间都可以进行交流、共享信息。

在概念上，KQML 是一种层次结构型语言，可分为三层：内容层、通信层和消息层。

内容层：是消息所包含的真正内容，用程序自己的表示语言来表示。KQML 可以采用任何形式的表示语言，值得指出的是，所有 KQML 语言的具体实现都不关心消息中的内容部分的具体含义。

通信层：通信层描述了与通信双方有关的一组属性参数，例如发送方和接受方的身份，与此次通信相关的惟一的标识等。

消息层：消息层构成了 KQML 语言的核心。它确定了可以与使用 KQML 语言的 Agent 进行交流的类型。消息层的基本功能是确定传送消息所使用的协议，并由发送方提供一个与内涵相关的行为原语，用于指明消息中的内涵为确认、询问、命令或是其它已知的原语类型。在消息层也包含对内涵进行描述的可选参数，例如所用的语言、采用的 ontology 等。这些属性参数可以让 KQML 语言的具体实现在内涵是透明的情况下，仍能够正确地分析和传送消息。

KQML 的语法非常简单，是基于平衡的括号表。表的开始为行为原语的名称，其余部分为一组以“：关键字值”形式出现的参数表。

KQML 中定义了一组含义明确的、预留的行为原语。这些预留的行为原语并非是 KQML 具体实现中必须实现的最小子集，它可以根据需要选择实现或添加新的原语，但是通常要求选择实现的预留原语应满足 KQML 标准的要求。习惯上，一条 KQML 行为原语也称为一条消息，一条典型的 KQML 消息如下：

(ask—about

: sender ForeAgent

: content(data)

: receiver SendAgent

: reply—with id

: language English

: ontology NYSE—TICKS)

其中，ask—about 为 KQML 的一条用于询问的预留的行为原语名字，sender、content、receiver、reply—with、language、ontology 为 KQML 预留的行为原语的参数。参数 content 的值是一个表达式，它必须符合参数 language 所指定的语言的语法，其中的常量必须在参数 ontology 指定的概念关系中有定义。由于 Agent 之间多采用异步方式通信，所以利用参数 in-reply-to 和参数 reply-with 来匹配发出的询问和收到的回答。ontology 是指存在于交互的 Agent 之间的或 Agent 中的概念与概念之间的关系的一种描述。在 KQML 中，ontology 参数指的是 content 参数使用的实体集的名称。

5.3 系统 Agent 部署与运行

部署运行欺诈检测系统各个 Agent 时，在运行窗口参数栏里输入参数“java jade.Boot -gui sendagent:dataBase.SendAgent foreagent:dataBase.ForeAgent characteragent:dataBase.CharacterAgent detectagent:dataBase.DetectAgent...”，运行后就启动了 JADE 的 GUI 图形界面，如图 5.4 所示，里面运行了欺诈检测系统

的七个主要的功能 Agent，分别为： foreagent@tqh:1099/JADE(前台 Agent)，characteragent@tqh:1099/JADE(特征 Agent)，detectagent@tqh:1099/JADE(决策 Agent)，三个 detectunitagent@ tqh:1099 /JADE(决策单元 Agent)和 sendagent@tqh:1099/JADE(推送 Agent)。

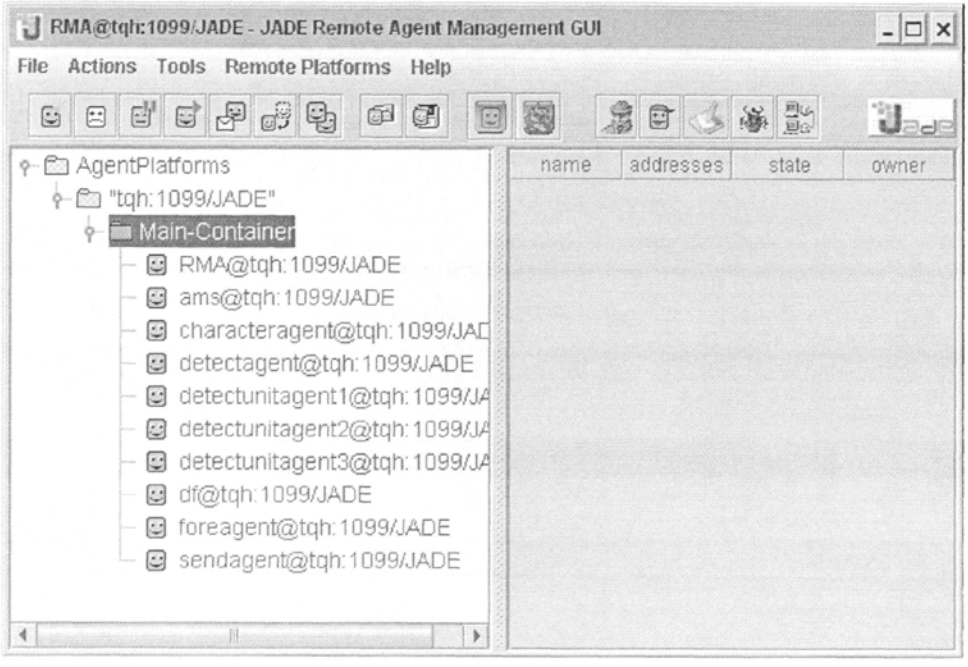

图 5.4 Agent 运行主界面

JADE 提供了一些工具用以方便操作多 Agent 系统。鼠标右击图 5.4 左边树状的 Main-Container 项，选择 Start New Agent，出现图 5.5 所示的界面，在 Agent Name 中输入 Agent 的名称，在 Class Name 中输入 Agent 的类名，点击 OK 按钮。此时 informationreceiveagent 已经正常启动，并且等待其它 Agent 的通知消息。

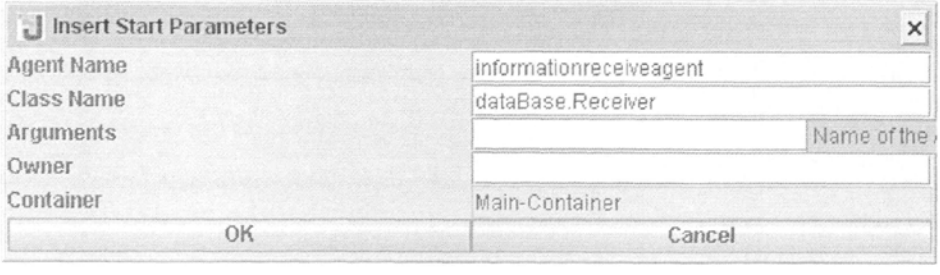

图 5.5 启动一个 Agent

在模拟 Agent 之间的通信时，我们利用 JADE 平台提供的 Dummy Agent。点击菜单 Tools，选择 Start Dummy Agent，按照图 5.6 所示输入相应字段信息，点击工具条当中的 Send the current ACL message 按钮，此时 sendagent 向 informationreceiveagent 发送一条类型为 INFORM-REF 的 ACL 消息。

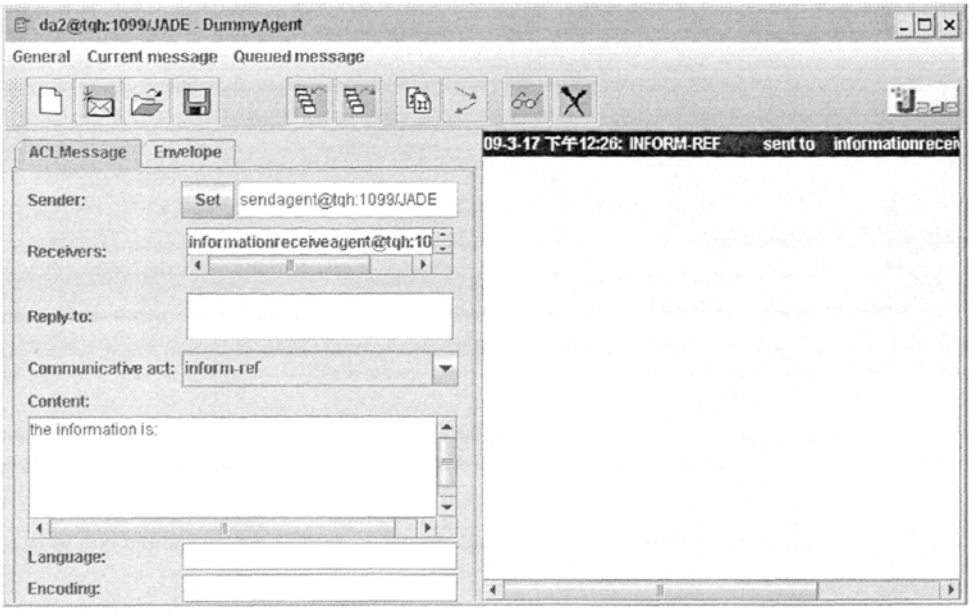

图 5.6 消息发送

5.4 信息推送

经过前面对 Agent 通信技术的说明，本欺诈检测系统的通信采用消息传递的方式，通过建立 socket 通道，使得 Agent 之间可以利用底层的 TCP/IP 协议进行消息传递，这种方式中信息的发送者和接收者都是已知的，该方式能减少不必要的信息冗余，降低通信负载。系统中消息的定义通常包括消息的发送者和接收者、消息的类型和内容，以及使用的语言等。本系统中定义的消息格式如下：

```
Message{
  SendAgent：String//消息的发送者
  ReceiveAgent：String//消息的接收者
  MessageType：String;//消息的类型
  Content：String;//消息的内容
  Language：String;//使用的语言
}
```

其中消息的发送者和接收者用 Agent 的 ID 表示；消息的类型包括注册、请求调用、请求应答、注销等。消息的内容包括具体请求调用的参数，传送数据等。使用的语言是指消息发送时所使用的语言。

本文采用 Eclipse 开发工具跟 JADE 平台来开发医疗欺诈检测系统，在 JADE 平台中，欺诈检测系统各个 Agent 之间通过 KQML 语言进行通信时，消息由 jade.lang.acl.ACMessage 类对象表示。

1、发送消息

在 JADE 平台上运行的 Agent 通过 Agent 类的 send()方法将信息发往其它 Agent，下框为主要代码片段。

```
public class DetectAgent extends Agent{
  String suspectinform; //存放收到的欺诈信息
  public void setup() {
   addBehaviour(new CyclicBehaviour() {
    public void action() {
      ACLMessage msgRx=receive();//从其它 Agent 接收信息
     if (msgRx != null) {
       suspectinform=msgRx.getContent();
        //向管理者电脑 manager 上的推送 Agent 发送结果
       AID Rams= new AID("sendagent@manager:1099/JADE");
       Rams.addAddresses("http://manager:7778/acc");
       ACLMessage msg=new ACLMessage(ACLMessage.INFORM);
       msg.addReceiver(Rams);
       msg.setLanguage("English");
       String str1= suspectinform;
       try{
            msg.setContent(str1);
       }catch(Exception e){
       }
       send(msg);
        System.out.println("Message has sended:"+msg.getContent());
     }else block();
    }//action()
  });//addBehaviour()
}}
```

2、 Agent 接收消息

Agent 通过 receive()方法从其消息队列中接收来自其它 Agent 的消息，该方法返回消息队列中的第一条消息，然后将之从消息队列中移除。如果队列为空，则返回 null。主要代码如下所示。

```
public class SendAgent extends Agent {
  public void setup() {
    addBehaviour(new CyclicBehaviour() {
      public void action() {
        ACLMessage msgRx = receive();
        if(msgRx != null) {
          String strmsgrx=msgRx.getContent();
          System.out.println("The fraud information is:"+strmsgrx);
        ACLMessage msgReply = msgRx.createReply();
        msgReply.setContent("I have received the message.");
        send(msgReply);
        JOptionPane.showMessageDialog(null,strmsgrx,"警示：请注意！
                                ",JOptionPane.INFORMATION_MESSAGE);
       } else block();
      }
    });//addBehaviour()
 }}
```

5.5 系统数据访问

1、用 JDBC 访问数据库

由于本系统的 Agent 是用 JADE 平台作为开发和执行的平台，所以我们用 Java 提供的 JDBC API 这一支持基本 SQL 功能的应用程序编程接口来实现 Agent 对数据库的访问。JDBC API 提供的重要接口有：

(1) Java.sql.ResultSet：控制对于给定声明取得结果列的途径。

(2) Java.sql.Statement：在一个给定的连接中作为 SQL 执行声明的容器，它包括两个重要的子类型：

- ✧ Java.sql.PreparedStatement：用于执行预编译的 SQL 声明。
- ✧ Java.sql.CallableStatement：用于执行数据库中存储过程的调用。

(3) Java.sql.DriveManager：用于处理装载驱动管理程序，并且为创建新的数据库连接提供支持。

JDBC 通常有以下四种类型：①JDBC-ODBC 桥；②部分 java 部分本机驱动程序；③中间数据访问服务器；④纯 java 驱动程序。本系统中采用 Microsoft SQL server 2000 数据库，采用 Microsoft 提供的纯 java 的 JDBC 驱动程序。

2、数据对象的封装

数据对象封装的目的是进行数据库操作和对象操作的相互转换。本系统中对数据对象的封装根据对数据库的不同操作主要分为两类：

(1) 从数据库中查询、获取数据，此类特点是只从数据库读取数据，而不改变数据库内容和结构；我们定义了方法 getDataBySql(String sql,int col)来完成此类操作，具体代码如下：

```
public Vector getDataBySql(String sql,int col){
  Statement ps=null;
  System.out.println("start Sql"+sql);
  Vector data=new Vector();
  try{
    ps=myconn.prepareStatement(sql);
    Resultset rs=ps.executeQuery();
    while(rs.next()) //把数据放入 vector 中;
     {
      Vector childvector=new Vector();//定义一子 vector 存放每一行中的列值
      for(int i=0;i<col;i++){
        childvector.addElement(String.valueOf(rs.getObject(i+1)));
      }
      data.addElement(childvector);
     }
  }catch(Execption e){
      System.out.println("the problem is"+e.getMessage());
  }
  statement=null;
  return data;
}
```

(2) 对数据库中数据进行编辑操作，此类操作与第一类不同之处在于需要改变数据库内容或结构。我们可以定义方法 executeData(string sql)来完成此类操作。

```
public boolean executeData(String sql){
   boolean backP=false;
   Statement ps=null;
   try{
      ps=myconn.prepareStatement(sql);
      ps.execute();
      backP=true;
   }catch(Exception e){
      System.out.println("the error is:"+e.getMessage());
   }
      return backP;
}
```

5.6 关键技术实现

5.6.1 决策 Agent 实现

决策 Agent 主要负责对各决策单元 Agent 进行管理，根据黑板中医疗特征信息的情况通知相应的决策单元 Agent 对医疗数据进行欺诈检测，决策 Agent 的主要代码片段如下框所示。

```
public class DetectAgent extends Agent {
   protected void setup() {
    addBehaviour(new CyclicBehaviour() {//对各决策单元Agent进行管理
      protected void action() {
       if (blackboard has new information){ //黑板中有新的医疗信息
         for(int i=0;i<amount of DetectAgent;i++){
           detect state of DetectUnitAgenti; //查询第i个决策单元的状态
           if（accord detect condition）{ //第i个决策单元Agent空闲并满足检测条件
               assign detect task; //分配给该决策单元Agent检测任务
         }}
   }}); }
}
```

系统需要选择合适的决策单元 Agent 对数据进行检测，首先分析目标信息，然后再查看决策单元 Agent 的相关信息，并将任务交给相应的决策单元 Agent

来完成，如果没有合适的决策单元 Agent，则采用其它方式，如另外新建一决策单元 Agent 进行检测或等待其它决策单元 Agent。主要代码如下。

```
public void plan(DetectInf inf){
    ArrayList list=lookUp(inf);//查询规则库获取目标对应的检测规则
   BlkBod blkbod=new BlkBod();
   for(int i=0;i<list.Count;i++){
    String detectrule=(String)list.get(i);//取得检测规则
     String agentName=blkbod.getAgentName(detectrule);//查询相应的决策单元 Agent
     if(agentName!=null){
       loadAssign();//将任务分配给相应的决策单元 Agent
     }
       ……
}}
```

决策单元 Agent 接收到检测任务后，从共享黑板中获取相应的检测规则和检测目标，然后对医疗数据进行欺诈检测。

5.6.2 决策单元 Agent 实现

决策单元 Agent 在对目标数据进行欺诈检测时，需要以某种策略对数据进行分析和检测。本文以效-效相似性检测为例，实现决策单元 Agent 对医疗数据检测时用到的效-效相似性检测算法。

在进行效-效相似性检测时，决策单元 Agent 对医保数据中涉及到的药物信息进行相似性检测，其主要思想是将就诊时所开的药物进行效-效相似性检测，若发现存在相似性超过一定阈值的数据时，则规定这样的数据为异常的，并递交用户进行进一步的审核。

为了得到药物之间药效的相似程度，在此通过改进 SimRank 算法[58-60]，提出一个基于 SimRank 的带权值的效-效相似性分析算法，即 WSimRank 算法。通过该算法计算药物之间的相似度，用于对门诊中的一些药物进行检测，从中发现医疗中存在的欺诈行为。

1、效-效相似度的计算

有些药物具有相似的功效，如板蓝根有清热解毒、凉血利咽的功效，清热散结片具有清热解毒、散结止痛的功效，有时在药方中要尽量避免出现开出相似药性的药物。

SimRank 算法是 Jeh 和 Widom 在 2002 年提出的一种通过分析事物之间的连

接关系来计算事物之间相似度的一种方法。该算法的核心思想是：如果两个对象 a 和 b 连接在相似的对象上，那么 a 和 b 也具有相似性。

根据这种思想，药物跟其药性的关系也可以认为是一种连接关系。若两种药都跟某些药性连接，那么就认为这两种药具有某种程度的相似性，而且连接的共同药性越多，其相似性越大。

根据 SimRank 算法，设定三种药物 D_1，D_2，D_3，它们的药性 E 与药物的隶属关系如图 5.7 所示。

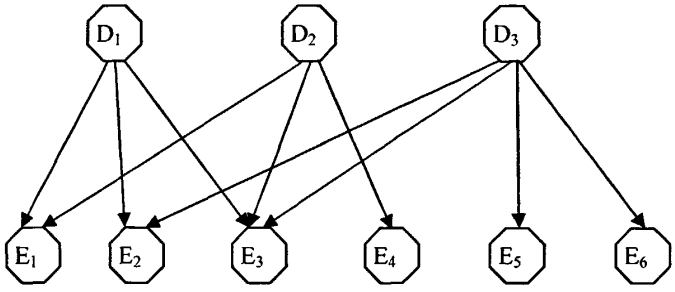

图 5.7 SimRank 有向图

从该图可以看出，对于药物 D_1 和 D_2，它们有两种共同的药性 E_1 和 E_3，对于药物 D_1 和 D_3 也有两种共同的药性 E_2 和 E_3，按照 SimRank 算法思想，可以得知药物 D_1 和 D_2 的相似度与 D_1 和 D_3 的相似度相同，即：

$$S(D_1,D_2)= S(D_1,D_3)$$

然而在现实中，有些药物的药性有好几种，而且药性的侧重度也不同，如药物 A 具有两种药性清热解毒和凉血利咽，但是主要对清热解毒更有效，对于凉血利咽只起辅助作用。这样通过 SimRank 算法算出的结果往往会由于忽视了药性在药物中所占比重的不同而出现偏差。

在此尝试按照药性在药物中所占比重的不同给药物的药性加一权值，在此同样取以上三种药物 D_1，D_2，D_3，并按照它们的药性在各自所在药物中的比重加一权值，该权值用一个数值 i 表示，$i\in[0,1]$，如图 5.8 所示。

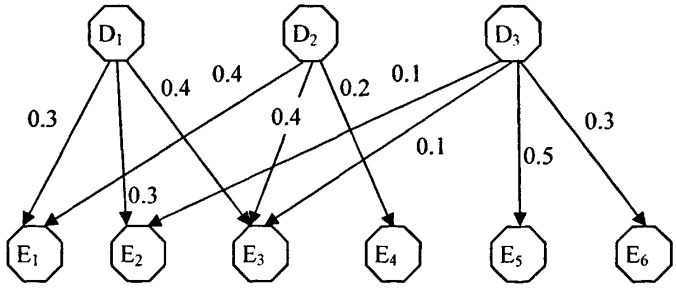

图 5.8 WSimRank 有向图

从该图可以看出，对于药物 D_1 和 D_2，它们有两种共同的药性 E_i 和 E_3，其中它们各自在 D_1 和 D_2 中所占的比重分别为 0.3、0.4 和 0.4、0.4；对于药物 D_1 和 D_3 也有两种共同的药性 E_2 和 E_3，这两种药性各自在 D_1 和 D_3 中所占的比重分别为 0.3、0.4 和 0.1、0.1。通过对比这两种结果，我们可以推测 D_1 和 D_2 的相似度要高于 D_1 和 D_3 的相似度，而并不是如 SimRank 算法算出的相似度相等的结果。

我们提出的 WSimRank 算法的基本思想就是按照药性在药物中所占比重的不同赋予其相应的权值，即基于药性的计算药物相似性的方法，在计算相似度时，我们规定如下：

(1) 对于同一种药名的药物 D，规定其跟自身的相似度为 1，即：

$$S(D,D)=1$$

(2) 对于不同药名的药物，规定如下：

设定两种药物 D_1 和 D_2，其各自都有不同的药性 E，按照各个药性在药物中所占比重的不同赋予各药性相应的权值 W，其中 $W\in[0,1]$，若 D_1、D_2 之间没有相同的药性，则规定它们之间相似度为 0。

在计算药物 D_1 和 D_2 的相似度 $S(D_1,D_2)$ 时，定义如下计算公式：

$$S(D_1,D_2)=C\sum_{i=1}^{|N(D1)|}\sum_{j=1}^{|N(D2)|}W_iW_jS(N_i(D_1),N_j(D_2)) \qquad (公式\ 1)$$

上式中 C 为一个常量，$C\in[0,1]$。W_i 为第 i 个药性在药物 D_1 中所占的权值，$Wi\in[0,1]$。表现在有向图中为相应边的一个介于 0 到 1 之间的值。当 W_i 为 1 时，表明药物 D_1 只有这一种药性，当为 0 时，则说明药物没有该药性，则在有向图中不需要画出。$N(D_1)$ 表示跟药物 D_1 相连的所有药性的集合$\{E_1, E_2, \ldots, E_n\}$。$N_i(D_1)$ 则表示 D_1 的第 i 个药性 E_i。

对于药性 E，同样适合如上对药物相似度的定义。如下所示：

$$S(E_1,E_2)=C\sum_{i=1}^{|N(E1)|}\sum_{j=1}^{|N(E2)|}W_iW_jS(N_i(E_1),N_j(E_2)) \qquad (公式\ 2)$$

根据以上的说明，对公式 1、2 交替进行迭代运算，对不同药物之间的相似度进行不断更新，最终达到收敛，此时即得到药物之间的相似度。该算法因为考虑到了药性在药物中所占比重的影响，从而在一定程度上提高了计算药物之间相似度的准确率，因而更有利于医疗欺诈行为的检测。

2、算法的应用

根据前面的规定，将药物跟其药性之间看作一种连接关系，我们用一个图例来表示这种关系，如图 5.9 所示。

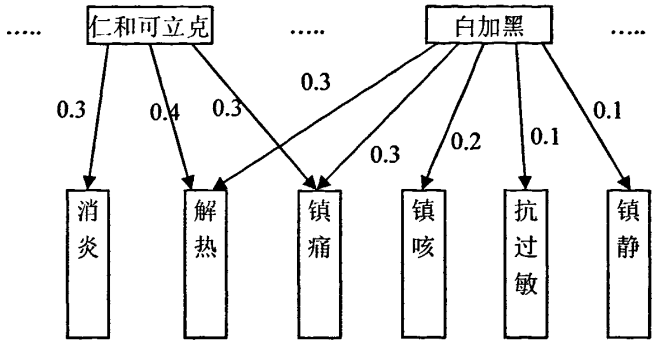

图 5.9 药效关联关系

其中边表示药性与药物之间的隶属关系，而边上的值则表示该药性在药物中的比重，即权重。

根据前面所定义的公式，根据公式 1、2，对“仁和可立克”和“白加黑”这两种药物，“解热”和“镇痛”这两种药性进行相似度计算，如下所示：

S(仁和可立克，白加黑)=C*{[0.3*0.3*S(消炎，解热)+...+0.3*0.1*S(消炎，镇静)+...+0.4*0.3*S(解热，解热)+...+0.4*0.1*S(解热，镇静)+...0.3*0.3*S(镇痛，解热)+...0.3*0.1*S(镇痛，镇静)]}

S(解热，镇痛)=C*{[...+0.4*0.3*S(仁和可立克，仁和可立克)+...+0.3*药效“镇痛”在药物 D_m 中所占的权值*S(白加黑，药物 D_m)+...]}

按照公式 1、2 交替迭代运算这两种药物之间的相似度，直到最后达到收敛为止，此时即得到这两种药的相似度 S(仁和可立克，白加黑)。

决策单元 Agent 对医疗数据中的药物信息进行分析，根据 WSimRank 方法计算出的药物间的相似度是否超过规定的阈值，得到医疗数据的异常与否，以此进行医疗欺诈行为检测的任务。

以下为决策单元 Agent 的主要功能代码：

```
public class DetectAgent extends Agent {
  protected void setup() {
   addBehaviour(new OneShotBehaviour() {
    protected void action() {
     ACLMessage msg=receive();
      if(msg!=null){
        if(msg.getSender().getLocalName().equalsIgnoreCase("detectagent")){
          unitdetect(msg); //收到决策 Agent 指令，进行欺诈检测，如果检测到异常
          ..... //信息，则将异常信息保存到异常信息库中
      }else block();
```

```
  }}); }
  protected void takeDown() {
   try {
     DFService.derester(this);
     doDispose();//停止Agent前做的清理工作。
    } catch (FIPAException e) {
       e.printStackTrace();
}}}
```

5.6.3 推送 Agent 实现

推送 Agent 定期检测异常信息库中的信息，当检测到有未推送给用户的异常数据时，将检测出的每一条异常数据封装在实体类 FraudInformation 的一个对象中，然后将该实体类对象加到一个 ArrayList 类型的变量 arraylistinf 中，并将之推送给用户。异常信息推送时的主要代码如下：

```
addBehaviour(new CyclicBehaviour() {
 public void action() {
    ......
   try{
    ACLMessage msg = new ACLMessage(ACLMessage.INFORM);
    AID useragent = new AID("useragent",AID.ISLOCALNAME );
    msg.addReceiver(useragent);  //设置接收者
    msg.setContentObject(arraylistinf);   //设置发送内容
    msg.setLanguage("JavaSerialization");
    send(msg);
   }
   catch (IOException e) {
    e.printStackTrace();
   }
}});
```

用户端接收到推送 Agent 推送来的信息时，通过使用 SWT 及 Jface 等控件实现数据信息的显示。

在SWT中使用的是表格控件table，SWT Designer插件里也只提供了table。通过使用一个在table基础上扩展而来的Jface控件TableViewer将数据显示出来。一般有如下几个主要步骤：

(1) 实例化一个TableViewer对象并对其进行布局。

(2) 建立表格中的列。

(3) 设定内容器。一般写成内部类，可单独拿出来写成一个类，让整个系统共用。

(4) 设定标签器。

(5) 用setInput输入数据。这些数据会根据列设置、内容器、标签器的设定将自己显示在TableViewer的表格中。

用户界面给用户提供了一个与Agent交互的平台，用户查看从推送Agent推送来的异常数据并对这些异常数据做出相应的处理。在此只列出用户界面UserAgent处理来自推送Agent的部分代码。

```
import jade.core.*;
……
import org.eclipse.jface.viewers.*;
import org.eclipse.swt.*;
public class UserAgent extends Agent {
protected void setup() {
 addBehaviour(new OneShotBehaviour(this) {
  public void action() {
   try{
    ACLMessage msg = blockingReceive(); //收到来自推送 Agent 的数据包
     if ("JavaSerialization".equals(msg.getLanguage())) {
        ArrayList arlstinf= (ArrayList)msg.getContentObject();//获取 Java 对象
        //创建用户界面
        final Display display = new Display();
        final Shell shell = new Shell();
        shell.setLayout(new FillLayout());
        TableViewer tv=new TableViewer(shell, SWT.MULTI| SWT.H_SCROLL
|SWT.V_SCROLL|SWT.BORDER| SWT.FULL_SELECTION); //设置参数
        setInterface(); //对界面进行布局与设置
         ……..
        tv.setContentProvider(new FraudContentProvider()); //内容器
```

```
        tv.setLabelProvider(new FraudLabelProvider()); //标签器
        tv.setInput(arlstinf);  // 从 arlstinf 获得数据
        ..........      //添加其它组件，便于对异常记录进行处理
        shell.open();
        while (!shell.isDisposed()) {
         if (!display.readAndDispatch())
           display.sleep();
        }
 } }
 catch(UnreadableException e){
   System.err.println(getLocalName()+ " catched exception "+e.getMessage());
 }
 block();
}});}//setup()
}
```

其中 arlstinf 用来保存从推送 Agent 获得的异常数据集，是一个对象的形式，通过对其里面的数据进行读取并放到 TableViewer 控件中并显示给用户。在此设计一异常内容器和异常标签器，前者用来对所有记录集合进行处理，后者对单条的记录进行处理，如下所示：

异常内容器对所有记录集合进行处理：

```
    private static final class FraudContentProvider implements IStructuredContentProvider {
        public Object[] getElements(Object element) {
          if (element instanceof List)
            return ((List) element).toArray();//将 List 转化为数组
          else
            return new Object[0];//否则返回一个空数组
      }}
```

异常标签器对单条记录进行处理：

```
private static final class FraudLabelProvider implements ItableLabelProvider{ //返回各列的值
 public String getColumnText(Object element, int col){//参数：输入的对象、列号
  FraudInformation fi = (FraudInformation) element; //转换一下类型
   dealData();//处理数据
}}
```

为便于用户对异常信息进行处理，如添加欺诈标记，在此通过给表格中的列

添加一些其它组件如编辑组件 CellEditor、按钮等，对所选中的记录进行编辑处理，并将相关信息保存到相关的数据表中，具体代码不再详述。

用户在登陆欺诈检测系统前需要身份认证，不同的用户具有不同的权限，如低级用户只能浏览信息，而高级用户则可以对异常信息做相关欺诈处理。

系统部分运行效果界面如图 5.10 所示。

图 5.10 检测结果

异常列表页面中显示的是多 Agent 医疗欺诈检测系统检测到的有欺诈嫌疑的信息。对于选中的信息，用户在核定其异常程度之后可以进行添加违规标记或进行处罚等操作。当选中其中的某条记录时，通过查看异常情况页面，对该条记录的详细诊疗明细情况进行审核，并对审核结果进行处理，如图 5.11 所示。

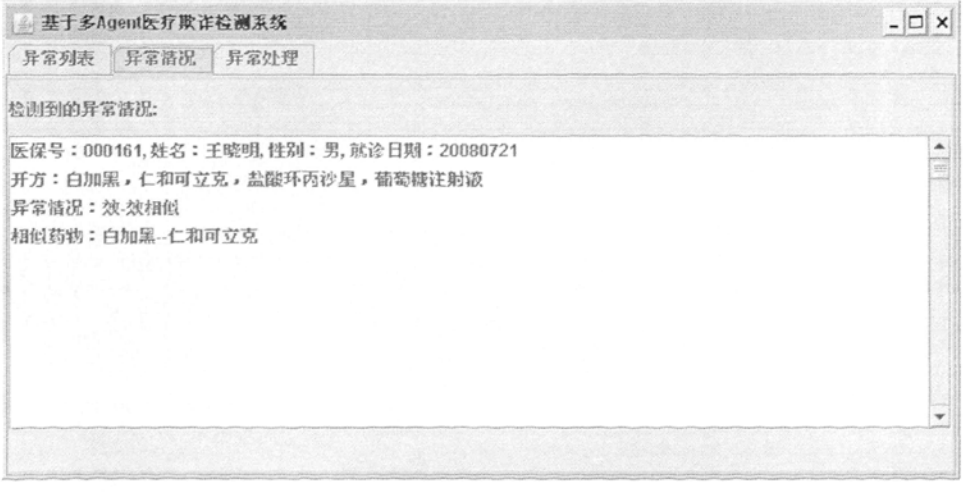

图 5.11 检测明细信息

5.7 本章小结

本章阐述了基于多 Agent 医疗欺诈行为检测系统的原型系统的开发过程，对系统中关键技术实现的方法作了详细的描述，并在 JADE 平台上进行了编码实现。最后，并以效-效相似性检测为例，将改进的 SimRank 算法用在对医疗行为的欺诈检测中。

第六章 结束语

本文利用 Agent 技术构建了一个基于多 Agent 医疗欺诈行为检测系统，用以对医疗欺诈行为进行检测，这为复杂的医疗欺诈行为检测提供了新的思维方法。本文所述系统中的 Agent 通过各自独立运行，专注于自己某一方面的任务，而使得系统具有可扩展性、灵活性、跨平台性、安全性和开放性等优点。各 Agent 之间通过相互协作完成检测任务，使它们表现为一个整体，提高了欺诈检测系统的性能。

尽管已经历了几十年的研究，但基于多 Agent 技术的应用还处在初始阶段，各种有关 Agent 的概念、结构、标准等的争论都还十分激烈，在应用方面还是局限于比较简单低级的应用，用其开发的整个系统还不是很稳定，缺乏良好的用户界面，缺乏一个系统的方法使设计者清楚地说明和构造多 Agent 系统。

由于客观原因，基于多 Agent 医疗欺诈行为检测系统的研究还只是一个初步的尝试，在研究过程中存在着不少困难，许多问题的解决还需要对 Agent 进行更深入的研究，这也是我们以后研究的重要内容。

今后需要进一步研究的工作主要集中在以下几个方面：

(1) 由于 Agent 及多 Agent 系统的体系结构跟真实系统之间存在千差万别的关系，因此 Agent 的结构势必多样化。Agent 个体是构成多 Agent 系统的基础，因此设计一种具有通用意义的 Agent 的结构具有重要意义。

(2) 系统的智能性有待进一步增强。多 Agent 系统中 Agent 之间的通讯和交互协作是多 Agent 系统的基本特征，其团队效应必须通过通讯和协作实现，可靠的通讯是多 Agent 之间协作的基础，有效的协作是保证系统整体性能的技术手段，因而 Agent 之间的协调和协作有待进一步研究。

(3) 系统的检测策略有待进一步的研究和改进。

基于多 Agent 医疗欺诈行为检测系统的进一步研究还有很多艰苦的工作要做。我们希望通过本文的工作，能够为该课题的后续研究提供一些新的思路。

参考文献

[1] 张晓燕.医疗保险中的道德风险分析与控制.江苏卫生事业管理.2004,15(77):10-13

[2] 马蔚姝,张再生.医疗保险市场中骗保行为产生的经济学分析[J].现代经济探讨.2008.11:46-49

[3] 夏波光.反欺诈"战役"悄然打响.中国社会保障.2006(6)

[4] 李炎,李皓,钱肖鲁等.异常检测算法分析[J].计算机工程.2002,28(6):5-6,32

[5] Nizar, A.H.; Zhao Yang Dong; Pei Zhang.Detection rules for Non Technical Losses analysis in power utilities[C].IEEE Power and Energy Society 2008 General Meeting:Conversion and Delivery of Electrical Energy in the 21st Century,PES.20-24 July 2008:1-8

[6] Lee W.A Framework for Constructing Features and Models for Intrusion Detection Systems[J].ACM Transactions on Information and System Security,2004,3(4): 227-261

[7] Peng,Yi;Kou,Gang;Sabatka,Alan;Chen,Zhengxin;Khazanchi,Deepak;Shi,Yong. Application of clustering methods to health insurance fraud detection[C]. Proceedings-ICSSSM'06:2006 International Conference on Service Systems and Service Management,v1.2007:116-120

[8] Cabral,Jose E;Pinto,Joao O.P;Linares,Kathya S.C;Pinto,Alexandra M.A.C. Methodology for fraud detection using rough sets[C].IEEE International Conference on Granular Computing. 2006:244-249

[9] L.Portnoy,E.Eskin,S.Stolfo.Intrusion detection with unlabeled data using clustering. Proceedings of ACM CSS Workshop on Data Mining Applied to Seeurity (DMSA-2001), Philadelphia,PA,November,2001

[10] P.Slobodan,A.Gonzalo,O.Agustin,et al.Labelling Clusters in an Intrusion Detection System Using a combination of Clustering Evaluation Techniques. Proceedings of the 39th Hawaii International Conference on System Seiences,IEEE,2006:129-136

[11] N.Bolshakova,F.Azuaje.Cluster Validation Techniques for Genome Expression Data.Signal Processing,2003:825-833

[12] 史忠植.智能 Agent 及其应用[M].北京:科学出版社.2000.12,7-11

[13] 刘大有等.Agent 研究现状与发展趋势[J].软件学报.2000.11(3):315-321

[14] 石慧,徐从富,刘勇等.Agent 通信语言 KQML 的实现及应用[J].计算机工程与应用.2005.13:94-97

[15] N.R.Jennings.Controlling Cooperative Problem Solving in Industrial Multiagent System Using Joint Intentions.Artificial intelligence.1995,75(2):195-240

[16] S.J.Russell.Provably Bounded optimall Agents.Proceedings of the 13th International Joint

Conference on Artificial Intelligence.1993:40-48

[17] D.M.Lane,A.G.Mcfadzean.Distributes Problem Solving and Real-Time Mechanism in Robot Architecture.Engineering Application of Artificial intelligence.1994,7(2):105-117

[18] S.Jia,YT.Qian;G.Dai.A Hybird Online Series Pattern Detection Algorithm,in Proc. of the Sth World Congress on Intelligent Control and Automation.Hangzhou, PR.China.2004: 1876-1879

[19] S.Kumar,E.Spafford.An application of pattern matching in intrusion detection. The COAST Project,Department of Computer Sciences,Purdue University,West Lafayette,Technical Report CSD-TR-94-013,June,1994

[20] 向尕,曹元大.一种面向检测的攻击分类方法及在 IDS 中的应用[J].计算机工程.2004, 30(11):49-95,173

[21] 徐明,陈纯,应晶.基于系统调用分类的异常检测[J].软件学报.2004,15(13): 391-403

[22] Anderson D,Valdes A.Next-generation Intrusion Detection Expert System. Computer Science Laboratory(SRI International Mello Park.CA).Technical Report SRI-CSL-95-07,1995

[23] S.E.Smaha.Haystack:An intrusion detection system.Proceedings of the IEEE Fourth Aerospace Computer Security Applications Conference,Orlando,FL.1988:37-44

[24] 朱明.数据挖掘.北京:中国科学技术大学出版社.2008

[25] 杨炳儒,刘发升.数据发掘与知识系统.北京科技大学学报.1999:20(6)

[26] HANDL J,KNOWLES J. An Evolutionary Approach to Multiobjective Clustering[C].IEEE Transactions on Evolutionary Computation. 2006

[27] A.K.Ghosh,C.Michael,M.Schatz.Areal-time intrusion detection system based on learning program behavior.Proceedings of the Third International Workshop on Advances in Intrusion Detection Toulouse,France.2000:93-109

[28] S.Ramaswamy,R.Rastogi,K.Shim.Efficient algorithms for mining outliers from large data sets.Proceedings of the ACM International Conference on Management of Data,USA. 2000:427-438

[29] F.Zambonelli, N.R.Jennings, and M.Wooldridge.Organizational Abstractions for the Analysis and Design of Multi-Agent Systems. P. Ciancarini and M. Wooldridge, editors, Agent-Oriented Software Engineering, Springer-Verlag Lecture Notes in AI, Vol.1957, January 2001

[30] Shoham Y.Agent oriented programming.Artificial Intelligence,1993,60(1):51-92.

[31] 黎建兴,毛新军,宋尧.软件 Agent 的一种面向对象设计模型[J].软件学报.2007,18(3): 582-591

[32] WOOLDBRIGE M J，J ENNINGS N R.Intelligent agents.theory and practice[J].Knowledge

Engineering Review.1995,10(2):115-152

[33] Rao,Georgeff Modeling Rational Agents within a BDI-Architecture.Proceeding of KR-91, San Mateo,CA,USA.1991:272-297

[34] Rao,Georgeff.An Abstract Architecture for Rational Agents.Proceeding of KR-92. 1992:423-441

[35] Rao,Georgeff A Model-Theoretic Approach to the Verification of Situated Reasoning systems.In IJCAI-93,Chamberey,France,1993

[36] 杨鲲,翟永顺,刘大有.Agent：特性与分类[J].计算机科学.1999,26(9):30-34

[37] Praca I,Ramos C,Vale Z,et al MASCEM:a multiagent system that simulates competitive electricity markets[J].IEEE IntellSyst.2003,18(6):54-60

[38] Nagata T,SasakiH.A multi-Agent approach to power system restoration[J].IEEE Trans Power Syst,2002,17(2):457-462

[39] C.Castelfranchi,M.Miceli,R.Conte,Limits and Levels of Cooperation: Disentangling Various Types of Prosocial Interaction,Decentralized AI-2,Yves Demazeau and Jean-Pierre Muller(Eds.),Elsevier Science Publisher B.V., Holland,1991

[40] Osawa.E.I.A Scherme for Agent Collaboration in Open Multiagent Environments[J]. IJCAI-93.1993,352-359

[41] 张维明,姚莉.智能协作信息技术[M].北京:电子工业出版社.2002:58-81

[42] 毛新军.面向 Agent 的软件开发[M].清华大学出版社.2005.6

[43] 韩伟,韩忠愿.基于黑板模型的多智能体合作学习[J].计算机工程. 2007,33(22):42-47

[44] 郭长伙.基于 Agent 的医院信息管理系统的研究[D]:硕士论文.吉林:吉林大学,2007

[45] Zhao,Yu;Li,Xiu;Lei,Lin;Liu,Wenhuang;Ren,Shouju.A blackboard based multi-agent distributed QFD System[C].IEEE International Conference on Systems, Man and Cybernetics.2004,6:5125-5129

[46] Kao,Hsing-Pei;Su,Eric;Wang,Brian.A blackboard-based multi-agent system for supporting concurrent engineering projects[J].International Journal of Production Research.2002 40,(5):1235-1262

[47] M.Asaka,T.Okazawa,A.Taguchi.An Intrusion Detection agent System[J].The Implementation of IDA.2002,10(22):65-70

[48] Spafford E.H.,Zamboni D.Intrusion detection using autonomous agents[J]. Computer Networks.October 2000,34(4):547-570

[49] Northcutt S,Novak J,McLanchlan D.Network Intrusion Detection An Analyst's Handbook[M].Indianapolis,Indiana:New Riders publishing,2000

[50] Anderson D,Valdes A.Next-generation Intrusion Detection Expert System. Computer Science

Laboratory(SRI International Mello Park.CA).Technical Report SRI-CSL-95-07,1995

[51] 史亮,李斌,庄镇泉.基于多主体技术的分布式入侵检测系统的研究与设计[J].计算机工程与科学.2005.27(2),5-8

[52] 曹军海,张和明,熊光楞.多 Agent 仿真中 Agent 行为的形式化描述方法[J].系统仿真学报.2004,16(11):2398-2400

[53] Fabio B,Giovanni Gaire,Tiziana T,et al.JADE Programmer's Guide.URL: Http://Agent.cs.bath.ac.uk/jade/programmersguide.pdf,20

[54] IEEE,FIPA 标准.http://www.fipa.org/,2006

[55] Eleni Mangina,Review of Software Produts for Multi-Agent Systems, http://www.AgentLink.org.2002.7

[56] Fabio Bellifemine,Giovanni Caire,Tiziana Trucco,Giovanni Rimassa.JADE PROGRAMMER'S GUIDE.2004.7,http://JADE.cselt.it

[57] Giovanni Caire.JADE TUTORIAL JADE PROGRAMMING FOR BEGINNERS, http://JADE.cselt.it.2003.10

[58] 田玲,曾涛,陈蓉等.基于 SimRank 的中药"效-效"相似关系挖掘[J].计算机工程.2007,34(12):242-244

[59] Jeh G,Widom J.SimRank:A Measure of Structural-context Similarity[C]//Proc.of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Edmonton,Canada:[s.n.].2002:538-543

[60] Yin Xiaoxin,Han Jiawei,Yu P S.LinkClus:Efficient Clustering via Heterogeneous Semantic Links[C]//Proc.of the 32nd Int'l Conf.on Very Large Data Bases'06. Seoul,Korea:[s.n.].2006:427-438

致　谢

岁月匆匆，三年的研究生生活即将结束，回想昨日的点点滴滴，而今尤历历在目。

首先，我要衷心地感谢我的恩师杨鹤标教授，在跟随杨老师的这三年期间，无论是治学还是为人，杨老师的认真指点，都给了我莫大的帮助与鼓励。杨老师总是在最关键的时候为我指明继续前行的方向，杨老师对我的栽培和帮助我此生难忘！其渊博的学术理论知识、严谨的治学态度、平易近人的处世风范和宽厚待人的高尚品质，为我树立了学习的榜样，是我永远学习的楷模！恩师的谆谆教诲，使自己在工作、学习和生活中取得了很大的进步。一日为师，终生为父，我会在今后的工作和学习中不断努力，谨记恩师的教诲，以不辜负恩师的殷切希望！

在此感谢所有同窗和同门师兄（姐）弟（妹）石春雷、王珏、李俊、刘志宏、郑甜、侯仁刚、乔亦民、陈玉坤、王健、陈凯、刘玲、刘志然，陈震，袁晓东等多年来所给予我的无私帮助和殷切关怀。

感谢所有上面提到的人在我完成硕士论文的过程中所给予的帮助和支持，感谢所有曾经以及正在帮助我的人们！

感谢我的父母，感谢我的舅妈和舅舅，感谢我的亲人，感谢你们的养育之恩和悉心照顾，正是因为你们多年来的无私奉献和鼓励，今天这一切才成为可能。

最后，谨向百忙之中审阅论文和参加答辩的每一位指导老师表示衷心的感谢！

攻读硕士学位期间发表的论文

[1] 基于模式识别临床病例异常检测算法研究.计算机工程与设计.2009 年 12 月发表.（第一作者）

[2] 一种支持界面自动生成的模型研究.计算机工程. 2010 年 04 期发表.（第三作者）