工学硕士学位论文

基于Petri网的RFID复杂事件检测方法及优化策略研究

RESEARCH ON PETRI NET BASED RFID COMPLEX EVENT PROCESSING AND OPTIMIZATION METHOD

兰 钊

哈尔滨工业大学

2011年12月

国内图书分类号：TP391. 4　　学校代码：10213
国际图书分类号：621.3　　密级：公开

工学硕士学位论文

基于Petri网的RFID复杂事件检测方法及优化策略研究

硕士研究生：兰钊
导　　师：张春慨 副教授
申请学位：工学硕士
学　　科：计算机科学与技术
所在单位：深圳研究生院
答辩日期：2011 年 12 月
授予学位单位：哈尔滨工业大学

Classified Index: TP391. 3

U.D.C: 621.3

Thesis for the Master Degree in Engineering

RESEARCH ON PETRI NET BASED RFID COMPLEX EVENT PROCESSING AND OPTIMIZATION METHOD

Candidate: Zhao Lan

Supervisor: Associate Prof. Chunkai Zhang

Academic Degree Applied for: Master of Engineering

Specialty: Computer Science & Technology

Affiliation: Shenzhen Graduate School

Date of Defence: Dec, 2011

Degree-Conferring-Institution: Harbin Institute of Technology

摘 要

随着全球信息化进程的推进，无线射频识别（Radio Frequency Identification）技术已经在各个领域得到广泛应用，包括物流、交通运输、医疗保健、工业、商业、金融、海关及政府管理等。RFID 技术是一种非接触式的双向射频识别技术，利用它可以准确地对物体进行定位，识别，追踪。典型的 RFID 系统由 RFID 标签，RFID 阅读器，及上层 RFID 数据管理系统组成。随着 RFID 技术的普及和企业大规模的部署 RFID 设备，RFID 阅读器产生的数据也随之呈海量性增长，但是 RFID 阅读器产生的原始 RFID 数据对用户来说是毫无意义的，需要由 RFID 数据管理系统对这些海量数据进行处理，将 RFID 数据封装成 RFID 事件，然后再进行事件检测，最终提供给用户需要的结果。复杂事件处理技术便是 RFID 数据管理系统中检测事件的关键技术，它可以处理大量简单事件，并从中整理出有价值的事件，称为复杂事件，提供给用户或者上层其他的商业应用。

本文首先从复杂事件模型和复杂事件检测模型两方面回顾了复杂事件处理的研究现状，分析了目前研究存在的不足，然后深入介绍了 RFID 复杂事件处理的基本理论，在此基础上定义了一个基于时间点的复杂事件模型和一个基于 Petri 网的复杂事件检测模型。之后在 CESN 检测算法的基础上给出了一个改进的复杂事件检测算法，并通过对复杂事件检测过程的分析提出了相应的事件检测优化策略。最后通过实验验证了改进算法的正确性以及优化策略为复杂事件检测带来的效率上的提高。

本文的主要研究内容如下：

（1）回顾了 RFID 复杂事件处理技术的研究进展及存在的问题。

（2）深入分析了 RFID 数据的特点，RFID 事件的分类，建立了基于时间点的事件模型。并定义了该事件模型下的事件之间的关系，事件操作符及时间限制条件，用户可以根据它们建立满足不同场景需要的复杂事件。

（3）通过使用 Petri 网理论，给出了一个基于 Petri 网的复杂事件检测模型。之后在 CESN 检测算法的基础上给出了一个改进的复杂事件检测算法 PCED，并通过实验验证了该算法的正确性。

（4）分析了该检测算法在检测复杂事件流程中可以优化的部分，并给出了优化策略，最后通过对比实验验证了优化策略可以使算法在效率上得到提高。

（5）实现了一个可在不同场景下检测复杂事件的复杂事件检测系统。

关键词：RFID 事件；Petri 网模型；事件检测算法

Abstract

With the advancement of the global process of information, Radio Frequency Identification technology has been widely used in various fields, including defense, transportation, healthcare, public safety, customs and government management, etc. RFID technology is a non-contact two-way radio frequency identification technology, which can be used for location, identification and tracing. A typical RFID system consists of RFID tags, RFID readers, RFID data management system. With the popularity of RFID technology, enterprises have deployed a large number of RFID equipments, so the RFID data also has largely increased. But the raw RFID data is meaningless to users. So we need a RFID data management system to process the mass RFID raw data, change them to RFID primitive events, then detect RFID complex event from them, finally give users what they really need. Complex event processing technology is the key part of RFID data management system. It can get a lot of valuable information for users or upper application from primitive event.

This thesis first reviewed the recent research of complex event processing on complex event model and complex event detecting model. By analysing the shortcomings of them, we introduced the basic theory of RFID event processing, and gave a time point based RFID event model and a Petri net based complex event processing model. After that, it gave an improved complex event detecting algorithm based on the CESN algorithm, and two optimization strategies were given two. At last, an experiment was carried out and the result verified the correctness of the improved complex event detecting algorithm and efficient improvement of the optimization strategy.

The contributions of this thesis are as follows:

Firstly, this thesis reviewed the recent research of complex event processing and their shortcomings.

Secondly, the RFID data and the classification of RFID events were analyzed, and we carried out an event model based on time point. Then defined the relationship between events, event operator and event time limitation, users can make different complex events for different application scenarios with them.

Thirdly, by using the Petri net theory, this thesis gave a Petri net based RFID complex event detecting model, then gave an improved complex event detecting algorithm based on the CESN algorithm. And the experiment results verified the correctness of the improved complex event detecting algorithm.

Fourthly, this thesis analyzed the improved algorithm, and gave two

optimization strategies to improve the efficiency of the algorithm. The experiment result shows that two optimization strategies had made the algorithm more efficent.

At last, a RFID comlpex event detection system was carried out, it can be used for complex event detecting in many application fields.

Keywords: RFID event, Petri-net model, event detecting algorithm

目　录

第1章　绪论

1.1 研究的背景和意义

条码技术是一种目前广泛使用的自动识别技术。

条码技术在其诞生之初，具有识别速度快、识别成本低、识别准确性高、保密性强等一系列优点[1]，但是随着时代的发展，条形码技术也逐渐暴露出一些弊端，主要在以下几个方面：首先，条形码携带的信息容量小，一维条码的容量是仅仅为几个字节，二维条码的容量比一维码大很多，但也只可存储数千字符[2]，所以条码技术大多需要依赖数据库存在。在没有数据库或者无法联网的地方，其作用受到明显的影响。第二，识别速度慢，条形码读取一次至少需要 1 秒以上的时间，而一个条码扫描仪一次只能读取一条条形码信息，不可批量读取。这种识别速度虽然相对人工识别来说有很大提高，但是在很多场合如对于产品数量较为集中的仓储、大型超市等，仍满足不了快速识别的需求。第三，条形码技术只能近距离读取静止不动的物体，对于运动的物体识别能力有限。第四，环境抵抗力差，条形码的载体为纸张，因此容易受到环境污损，当条形码因为损伤或者被弄脏后，扫描仪就无法辨认目标。第五，条形码不可重复使用，印刷上去之后，条码信息便不可修改。

随着科技的发展，为了弥补传统条形码技术存在的种种不足，一种新型的识别技术——RFID 技术应运而生。

RFID 是英文 Radio Frequency Identification 的缩写，是上个世纪末逐渐开始研究的一种自动识别技术[3]。它利用无线射频信号进行通讯从而达到自动识别目标对象的目的 [4]。由于 RFID 技术是一种自动识别物品并获取数据的一种快速识别技术。相对于条码来说，RFID 技术具有以下几个突出优势：第一，RFID 可存储数据容量大，相对于条形码，可存储数 K 字节[5]，未来物品所需携带的资料量会越来越大，RFID 技术恰好可以满足这一需求。第二，RFID 可以快速扫描，RFID 标签一般的识别速度为毫秒级[6]，此外 RFID 读写器可非接触同时辨识多个 RFID 标签。这样便极大提高了物品识别速度。第三，RFID 可识别运动中的物体，RFID 超高频标签下可识别以 200 公里/小时的速度运动的物体[7]。第四，RFID 标签环境适应性强，不会像条形码一样易被化学物品腐蚀，RFID 标签的数据是存储在标签内部的芯片中的，因此即使标签表面被污损，也不会影响到标签的数据。第五，RFID 标签可重复使用，其内部的芯片可通

过阅读器反复读写，方便信息更新和重复利用。此外，RFID 技术还具有可穿透、安全性高等诸多优点[8]，因此 RFID 技术可以有效解决传统条码技术存在的种种不足之处，正逐步代替传统的条形码技术。RFID 技术目前已在各个领域得到应用，包括国防和军事[9]，生产和物流、交通、运输[10]、医疗[11]、防伪、跟踪[12]、设备和资产管理等[13]。

虽然与传统的条码技术相比，RFID 技术在实时监控和信息自动收集方面有着明显的优势，但随着 RFID 技术的普及和越来越多的应用场景大规模部署 RFID 设备，问题也随之产生。首先，RFID 标签信息数据量大，在采用 RFID 技术的应用中，RFID 设备将实时的产生大量 RFID 数据 [14] ，如何收集、管理、使用这些海量的数据已经成为 RFID 应用中急需解决的问题。其次，RFID 读写器的标签数据本身是无意义的，只包含阅读器 ID，标签信息及产生的时间。即使在特定场合下赋予其语义，也只能包含简单的语义信息[15]。而现实应用中人们需要的并非这些简单孤立的数据，而是希望从这些大量无意义的数据中提取出的具有更高级别语义的抽象信息。 所以，RFID 数据处理需要有新的机制，从大量的原始 RFID 数据中发现数据之间隐含的信息，使得上层应用能够及时获取需要的信息，以便系统及时做出反应，为决策者提供更有价值的信息。

RFID 复杂事件处理技术就是为了解决上述问题而产生的。

RFID 应用中将 RFID 数据与实际的商业逻辑结合在一起，形成具有现实语义的数据，这些数据可称为 RFID 事件[16]。RFID 事件可分为原始事件与复杂事件。原始事件是由阅读器与标签交互产生的，一般可将阅读器产生的一条原始数据看作一个原始事件。RFID 复杂事件处理的主要任务就是将大量无意义的实时 RFID 数据抽象成的原始事件，运用一些规则从中找出对企业有价值、有意义的商业信息，从而为决策的制定提供数据依据。例如，在一个部署了 RFID 设备的大型供仓储景中，通过 RFID 设备监控物品进入仓库、放入货柜、由仓库出库等原始事件，经过复杂事件处理，可做到检测某货物存货不足事件，发现物品被盗事件并报警、动态反馈仓库实时的库存量信息等管理者需要的信息。由于 RFID 复杂事件处理可以为企业提供更有价值的信息，从而为其决策提供更大的帮助 [17]，因此对 RFID 复杂事件处理技术的研究对推广 RFID 技术的应用和为企业提供更智能的商业信息具有重大的意义。

1.2 国内外相关研究综述

复杂事件处理技术的研究源于 20 世纪 90 年代初 [18]，这一阶段对主动数据库领域的研究促进了复杂事件处理技术的发展[19-23]。RFID 复杂事件处理技术

的主要目的就是从大规模的实时 RFID 数据中按照一定的规则抽取出对用户有用的信息。这种方法早期是以 RFID 数据为中心[24]。 以数据为中心的方法一般首先对 RFID 原始数据进行建模，并将原始数据保存到数据库中，在数据库管理系统的查询的基础上检测用户感兴趣的事件。由于以数据为中心的方法需要先收集原始数据，然后再将产生的历史数据保存在数据库中，最后进行处理，因此其检测的实时性不高，比较适用于处理低实时性的应用场合[25,26]。而现阶段的 RFID 系统一般都要求从高速产生的大量 RFID 数据中高实时发现对用户或企业有帮助的信息，如异常信息等，从而可以根据它们进行及时的决策或者提供一些警告信息，因此目前对系统的实时性要求很高。所以以数据为中心的方法目前已经很少用于 RFID 复杂事件检测。

以事件为中心的处理方法正是为了避免以数据为中心的不足之处发展而来的。以事件为中心的方法首先将复杂的商业逻辑表示成复杂事件，然后再形式化复杂事件的规则和语义，最后把搜集到的 RFID 数据按照这种规则和语义来进行检测，最后把检测到的复杂事件提供给系统或者用户。同时，在一些特定场合下，以事件为中心的处理方法还可以通过事件之间的组合产生语义级别更高的复杂事件[27]。本文主要研究以事件为中心的复杂事件处理方法。

1.2.1 RFID 事件模型

在以事件为中心的 RFID 复杂事件处理过程中，需要将数据以事件的形式展现。这就需要为 RFID 数据建立一个事件模型，用该模型将原始的 RFID 数据表示为 RFID 原始事件，这样才可以被复杂事件的规则和语义所检测。

由于 RFID 数据都包含时间属性，因此 RFID 事件模型中的一个重要元素即为事件的时间，那么如何定义事件的时间将会对事件之间的关系产生本质影响，因为 RFID 事件的关系中很重要一方面就是事件之间的时间关系[28]。根据如何定义事件的时间，目前，事件模型的定义主要有两种方向：基于时间点的事件模型和基于时间间隔的事件模型。

基于时间点的事件模型：用一个时间点来表示一个事件的发生时间 ，即某个事件在某个时间点发生。对于基于时间点的事件模型来说，原始事件（也有文献称简单事件）发生的事件就是该 RFID 数据产生的时间戳[29]，复杂事件（也有文献称复合事件）发生的时间时使该复合事件表达式满足条件的时间点[30]。原始事件与复杂事件都用一个时间点来表示。这样的定义的好处是可以简化复杂事件处理的检测工作。但是这样定义也存在的缺点就是在某些场合下不能够充分的反映实际事件的时序信息。

基于时间间隔的事件模型：用一个时间区间来表示一个事件的发生时间，

一个事件的发生是一个过程，分别是事件的开始时间和事件的结束时间[31]。这种模型下的原始事件其事件的开始时间等于结束时间；对于复杂事件来说，事件的开始时间时组成该复杂事件的所有子事件中最早发生的事件的发生时间，事件的结束时间时该复杂事件的子事件中最后结束的事件的结束时间。这样的定义的优点是可以使得 RFID 事件模型在某些场合更贴近实际事件，但是其缺点也是事件之间可能会发生重合，这样势必将增加事件处理的检测工作的复杂程度。

1.2.2 RFID 事件检测方法

前面提到，复杂事件的检测过程是先将复杂的商业逻辑表示成复杂事件，然后再形式化复杂事件的规则和语义，最后把搜集到的 RFID 数据按照这种规则和语义来进行检测，最后把检测的结果提供给系统或者用户。那么定义好了事件的模型，就可以把搜集到的 RFID 数据表示成事件，之后就应该寻找一种有效的方法来对其进行检测。

目前事件检测方法主要分四种，分别对应四种模型：自动机模型，Petri 网模型，匹配树模型和有向图模型。文献[32]中分别分析了几种方法的不同。

（1）基于自动机模型的复杂事件检测方法：自动机模型中包含有状态集和状态迁移函数。使用自动机模型来检测复杂事件时，首先对每个欲检测的复杂事件构造其相对应的自动机，当一个参与复杂事件的基本事件达到时，自动机便在系统定义的状态迁移函数作用下迁移至事先定义的状态，如此下去直到自动机进入摸个可接受的状态，此时说明相应的复杂事件已发生。基于自动机的复杂事件检测方法最早是由 Gehani 在主动数据库领域的 ODE 系统中使用的[33] 。最近几年的很多系统都是采用自动机模型进行事件检测的，典型的有 SASE[34,35]、Cayuga[36]等。

（2）基于 Petri 网模型的复杂事件检测方法：Petri 网是由库所、变迁及连接他们的有向弧组成的网状模型。使用 Petri 网来检测复杂事件的基本思想是用 Petri 网输入位置的库所表示组成该复杂事件的基本事件，输出位置的库所表示需检测的复杂事件。库所内的 token 数表示该库所代表的事件是否有实例产生。当一个基本事件到来时，代表该事件的库所内 token 数将被改变。通过计算变迁函数来判断是否满足变迁条件。若满足，则引发变迁并改变输入输出库所的状态，直到输出位置库所内的 token 状态被改变，此时产生对应的复杂事件。可供参考的系统有：主动数据库系统 SAMOS [37]。北京大学的 WenHui Hu 等人提出的一种三层结构的复杂事件处理中间件[38,39]，分别为逻辑结构层、时间限制层和事件处理层，其三层结构中的复杂事件处理层即是使用 Petri 网模型构造

的。此外 Xingyi Jin 等人对 Perti 网进行扩展的基础上提出了 TPN 检测算法[40]。

（3）基于匹配树模型的复杂事件检测方法：基于匹配树模型的检测方法其基本思想是为每一个复杂事件构建一个对应的匹配树，由匹配树的叶子节点表示基本事件，根节点表示欲检测的复杂事件。当一个基本事件到来时，代表该基本事件的叶子节点会通知其父节点，父节点在根据事件语义和其他子节点的状态来判定是否产生一个新的复杂事件并通知上层节点。如此直到根节点代表的复杂事件被检测出。使用匹配树来进行复杂事件检测的系统有 GEM[41],Yeast[42]。

（4）基于有向图模型的复杂事件检测方法：基于有向图模型的检测方法是构建一个有向无环图去检测复杂事件。用该图的输入位置节点来表示基本事件，图的边来表示事件生成的规则。除此之外，某些中间节点也带有一些规则，主要是生成该节点表示的复杂事件的参数条件限制规则。当一个基本事件到来时，若有向图中有一个表示该事件的对应节点，节点的规则将被触发，该节点将被标记出来。Sentinel[43]系统和 EVE[44]系统采用的即为基于有向图的检测方法。

1.3 课题研究内容

因为本课题的目标是一个完整的 RFID 事件处理系统，所以研究内容涉及到上文提到的 RFID 事件模型，RFID 复杂事件表达式和 RFID 复杂事件检测方法及其优化策略。其中 RFID 事件检测模型及检测方法是本课题的研究重点。

1.3.1 RFID 事件模型

RFID 数据具有丰富的语义信息，而在各种语义信息中，最难处理的就是该数据的时序信息。要进行 RFID 的复杂事件检测，首先必须定义一个统一的事件模型来表示 RFID 数据、事件处理过程中产生的中间事件以及事件处理的结果事件。这个统一的事件模型应该尽可能地反映该数据或事件的实际信息，特别是时序信息。上文提到目前有两种事件模型，本课题研究的第一种，即基于时间点的事件模型。

1.3.2 RFID 事件检测方法

有了 RFID 的事件模型，我们需要的就是一个复杂事件的检测方法从 RFID 事件中检测出需要的复杂事件。上文提到，本文是使用以事件为中心的检测方法，而以事件为中心的检测方法主要基于四种模型，分别是自动机，Petri 网，匹配树，有向图。按照目前国内外的研究趋势，主要还是使用自动机与 Petri 网。但是这两种模型对于事件检测都有其局限性。例如 SASE[34]使用的是基于

自动机模型，但是其检测的复杂事件被限制于再原始事件之上构造；CESN[38]使用的是 Petri 网模型，但是在原始事件流上需要进行大量回溯，当数据流大时，无法保证处理速度。所以有必要对这些检测方法进行改进，使其符合事件检测的实际需要。

1.3.3 RFID 事件检测优化策略

同样的 RFID 数据流经过系统，系统可能需要同检测不同复杂事件，这些不同检测模型中有很多资源是可以共享的。若可以让不同的检测模型共享共用的部分，则可大大提高复杂事件检测引擎的性能。针对上文提到基于 Petri 网的复杂事件检测算法，在其基础上提出了一个检测优化策略。通过对比实验验证，证明利用检测优化策略进行复杂事件检测可以大大提高复杂事件检测引擎的性能。

1.4 本文的组织结构

本文主要研究的问题是： RFID 复杂事件模型、RFID 复杂事件检测方法，RFID 复杂事件检测算法的优化策略。对应与些问题的研究，本文的结构组织如下：

第一章绪论，主要介绍本论文的研究背景和意义，研究方向的国内外研究现状，包括 RFID 事件模型、RFID 复杂事件检测方法国内外研究现状，第一章结尾部分概述了本文的组织结构。

第二章介绍了 RFID 复杂事件检测的基本理论。介绍了两种 RFID 事件——RFID 原始事件及复杂事件的概念。在此基础上介绍了 RFID 复杂事件建模方法，分析定义了复杂事件的关系，各种复杂事件操作符。接着介绍了 Petri 网的相关知识及使用 Petri 网构建复杂事件模型的知识，本章的介绍的关于复杂事件的概念、定义及假设是后文研究的基础。

第三章首先给出了一个基于 Petri 网的 RFID 复杂事件检测模型。针对与 CESN 检测算法的不足，给出一个改进的复杂事件检测算法。最后，针对不同的复杂事件检模型，分析其可以共享的部分，在此基础上给出了检测共享优化算法。

第四章主要是对系统实现及实验分析部分。为了对复杂事件检测算法及优化策略进行验证，首先模拟了一个 RFID 场景实验，构建了该场景下的各种复杂事件，然后使用检测算法检测该场景下定义的复杂事件，通过实验结果来验证 RFID 事件检测算法和检测优化策略的正确性及效率的提高。最后，本文给

出了 RFID 事件处理系统，对其系统结构以及各个详细功能模块做了简要介绍。

第2章 RFID 复杂事件检测基本理论

2.1 引言

本章主要介绍 RFID 复杂事件检测的基本理论及相关概念。首先介绍 RFID 事件的概念，RFID 数据的特点，RFID 事件的分类；然后介绍了 RFID 复杂事件的建模方法，最后介绍基于 Petri 网的复杂事件检测模型。本章叙述的基本知识将为第三章论述 RFID 复杂事件检测算法及优化策略打好理论基础。

2.2 RFID 事件

在计算机系统中，一个事件可以定义为一个包含发生时间、发生地点、参与者等特定数据的一条记录。这些数据本身是没有意义的，但是当人们对数据进行逻辑抽象并与数据产生的上下文及应用背景信息相结合，按照某种逻辑组织起来，便可以赋予其丰富的语义信息，表达出有实际价值、有意义的信息。在一个 RFID 应用系统中，RFID 事件正是将 RFID 应用系统产生的 RFID 数据进行逻辑抽象并结合应用场景信息赋予其现实语义而产生的。可以说，RFID 数据是 RFID 事件的基础。下面首先介绍 RFID 数据的格式及其特点。

2.2.1 RFID 数据

一条 RFID 数据通常是由一个 RFID 阅读器读到一个标签后产生的。一个简单的 RFID 系统通常是由若干 RFID 标签、RFID 阅读器及 RFID 应用系统构成。

RFID 数据的一般格式为（EPC, ReaderID, Time）[45]。其中 EPC 为标签标示[46]，ReaderID 为阅读器标识，表示读到这个 RFID 标签并产生这条 RFID 数据的阅读器。Time 字段表示 RFID 阅读器读到该 RFID 标签的时间。随着 RFID 系统规模的扩大，一个 RFID 系统通常会包含很多 RFID 阅读器，同时由于 RFID 技术非接触性与快速识别的特点，RFID 标签可能会高速且源源不断经过阅读器，RFID 阅读器就会源源不断的产生大量识别到的 RFID 数据，因此一个 RFID 系统将会产生大量的 RFID 数据。这就为 RFID 复杂事件处理带来了新的挑战。同时经过分析发现，RFID 数据与普通的数据不同，它通常具有海量性、不准确性、语义丰富性、时态性、关联性等特征[24]。

有了 RFID 数据，便可以在其基础上定义 RFID 事件。表示一个 RFID 事件的一条 RFID 数据记录中，通常包含该事件产生的时间。上文曾提到，目前事

件模型的定义主要分两种方向：基于时间点的事件模型和基于时间间隔的事件模型，本文采用基于时间点的事件模型来定义事件。即一个事件发生的时间是一个时间点，没有持续的过程。这样的定义使得事件处理的后续工作得以简化，从而使本文专注于事件处理算法的研究。

2.2.2 RFID 原始事件

RFID 原始事件又被称作 RFID 原子事件，原始事件的产生是 RFID 阅读器检测到 RFID 标签后触发的，它是直接从原始 RFID 数据得到的瞬时发生的事件。RFID 原始事件具有较低的语义级别，逻辑结构简单。上面介绍过，RFID 数据的一般格式为（EPC, ReaderID, Time），用这个三元组便可以表示一个原始事件，在不同的场景下，赋予其不同的语义。例如，在图书馆 RFID 系统中，一条 RFID 原始数据（EPC, ReaderID, Time），EPC 表示图书 A 的唯一编号，ReaderID 表示放在某个书架 S 上的阅读器，Time 表示图书经过该书架的时间 T。这条原始数据就代表了一个原始事件的发生：图书 A 在 T 时刻经过了书架 S。

2.2.3 RFID 复杂事件

对于原始事件来说，它表示的是某个阅读器在某个事件读到了某个标签。然而在一个 RFID 应用系统中，原始事件通常表达的语义简单，往往无法满足应用中高层业务逻辑的需求，因此需要一定的规则将多个原始事件通过各种逻辑关系进行重新组合以表达更丰富的语义信息。我们将这些原始事件通过各种逻辑关系并经一定规则而产生的事件称作复杂事件。这些逻辑关系有包含关系、因果关系，时序关系等[24]。本文主要研究的复杂事件主要基于原始事件通过事件之间的时序关系进行组合。

仍然以图书馆应用场景为例：一条 RFID 原始事件 P_1（E_1,ReturnSet, t_1）表示图书 E_1 在 t_1 时刻放入图书馆还书箱；原始事件 P_2（E_1,BookTrolley, t_2）表示图书 E_1 在 t_2 时刻放在了上架车上；原始事件 P_3（E_1,Shelf, t_3）表示图书 E_1 在 t_3 时刻放在了分类的书架上。则由 P_1，P_2，P_3 顺序发生而组合产生的复杂事件 C_1 表示图书 E_1 重新上架完成。

从上文的分析可看出 RFID 原始事件与 RFID 复杂事件具有诸多方面的不同，他们分别解决了不同语义级别的问题。但 RFID 原始事件与 RFID 复杂事件也并不仅仅是为了解决不同问题而产生的，而这在内在本质特征上也有明显的不同跟区别，这些不同点包括：

（1）数据源不同，RFID 原始事件是由 RFID 原始数据封装而来的，而 RFID

复杂事件是由 RFID 原始事件和 RFID 复杂事件组合产生的，前者源于原始数据，后者源于事件。

（2）语义级别不同，RFID 原始事件语义级别较低，包含的信息量少。RFID 复杂事件语义级别较高，包含的信息量大，主要是该复杂事件的各个子事件间的逻辑关系，这些关系往往能表达出更加对应上层应用的更加丰富的语义。

（3）产生机制不同，上文提到过，原始事件的产生是 RFID 阅读器检测到 RFID 标签后触发的，它是直接从原始 RFID 数据得到的。而复杂事件是将多个原始事件通过各种逻辑关系并经一定规则而产生的，它通常涉及到多条原始数据，强调子事件之间的关系，需要经过一个复杂事件的检测模型判断所有的子事件都已经发生，且满足复杂事件产生规定的逻辑约束后才可产生。

2.3 RFID 复杂事件建模方法

上一节中介绍了 RFID 复杂事件的概念，可以看出，复杂事件实际上是由 RFID 原始事件通过一定关系组合而成的。本节中主要介绍 RFID 复杂事件的建模方法，即如何由 RFID 原始事件得到 RFID 复杂事件。下面依次介绍构成 RFID 复杂事件的主要元素，包括事件之间的关系、事件操作符及事件的时间限制属性等。

2.3.1 RFID 复杂事件关系定义

RFID 复杂事件之间的关系可分为两种：时序关系和逻辑关系。时序关系表示两个 RFID 事件的时间先后关系，逻辑关系主要表示两事件之间的“与”、“或”、“非”等逻辑关系。

首先介绍事件之间的时序关系。

上文提到，为了简化事件模型，从而专注于事件检测方法，本文采用基于时间点的事件模型来定义事件。这样的话事件之间的时序关系就只有三种：Before，After，Equal。

假设原始事件用大写字母表示，事件发生的事件用 T 加上事件小写字母作为下标表示。

如原始事件 A，发生的时间为 Ta，原始事件 B，发生的时间为 Tb。

A Before B：表示 A 事件先于 B 事件发生，有 Ta <Tb。

A After B：表示 A 事件后于 B 事件发生，有 Ta >Tb。

A Equal B：表示 A 事件于 B 事件同时发生，有 Ta =Tb。

基于时间点的事件模型时序关系如图 2-1 所示：

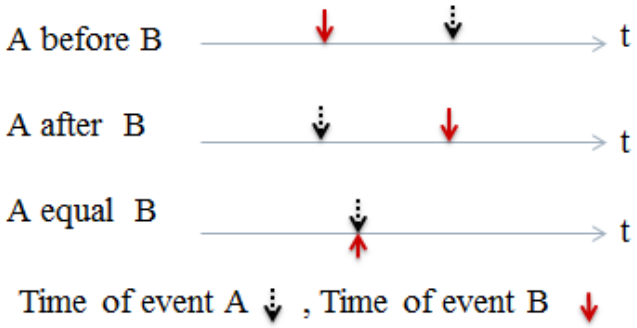

图 2-1 基于时间点的事件模型时序关系图

为了更精确地描述 A Before B，A After B 这两种时序关系，再引入另一个时序关系 During(A，B)：

During(A，B)=T ：表示 A Before B, A 且事件与 B 事件发生的时间间隔为 T，即 Tb - Ta =T 。

During(A，B)<T ：表示 A Before B, A 且事件与 B 事件发生的时间间隔小于 T，即 Tb - Ta < T 。

During(A，B)>T ：表示 A Before B, A 且事件与 B 事件发生的时间间隔大于 T，即 Tb - Ta > T 。

During(B，A)可参照以上定义给出事件 AB 之间的关系，这里不再赘述。

可以验证，基于时间点的事件模型下事件之间可能发生的时序关系都可通过上述分析得到的。

2.3.2 RFID 复杂事件操作符

上文中提到，所谓复杂事件是多个原始事件通过各种逻辑关系并经一定规则重新组合而产生的事件。常用的逻辑关系操作符“与”、“或”、“非”在之前的一些文献中已经被广泛使用[26,27,29]，用来描述原始事件之间的组合关系。由于 RFID 数据本身包含时间属性的特点，RFID 事件除了常用的“Before”，“After”，“equal”这些时序关系之外，还存在“与”、“或”、“非”等事件之间的逻辑关系时序关系[47]。但是目前仍然没有一个统一的标准规定 RFID 事件之间的关系种类，也没有一个统一的 RFID 复杂事件描述语言去描述复杂事件。

经过对 RFID 数据的深入分析，本文从一般的 RFID 系统商业逻辑出发，引入四种事件之间的关系操作符用以描述复杂事件:“与”、“或”、“非”、“顺序”。下面具体列出每种关系操作符的具体语义。

仍然假设原始事件用大写字母表示，事件发生的事件用 T 加上事件小写字母作为下标表示。如原始事件 A，发生的时间为 Ta ，原始事件 B，发生的时

间为 Tb。

“与”：表示两事件都发生。用符号“&”表示。A&B 表示 A 事件与 B 事件都发生。复杂事件 A&B 发生的时间为 Ta 和 Tb 中较大的一个。

“或”：表示两事件中有一个事件发生。用符号“|”表示。A|B 表示 A 事件发生或 B 事件发生。复杂事件 A|B 发生的时间为：若 A 事件发生，则为 Ta ，若 A 事件未发生而 B 事件发生，则为 Tb。

“非”：表示事件未发生。用符号“~”表示。~A 表示 A 事件未发生。

“顺序”：表示几个两事件顺序依次发生。用符号“SEQ”表示。SEQ(A,B) 表示 A 事件发生后 B 事件发生。

2.3.3 RFID 复杂事件时间属性限制

由于 RFID 数据的特殊性，定义好了复杂事件的操作符，还并不能完整表示一个复杂事件。如：检测复杂事件~A，即 A 事件未发生，若不定义一个检测的时间范围，则系统将无法进行检测。首先，系统无法从用户输入请求开始持续检测~A 事件，因为非事件并不会有具体的事件实例主动产生。只能是由用户提供一个检测的时间窗口，然后 RFID 复杂事件处理系统再去检测在该时间段内 A 事件是否发生，若该时间段内未检测到 A 事件发生，则产生~A 事件；其次， RFID 系统处理的对象一般都是海量的流数据，系统不能保存所有的 RFID 数据来进行事件检测，若系统持续检测某一事件而没有明确的事件限制，将必然导致内存溢出。因此在复杂事件操作符的基础上，还应该定义一个时间参数，用来表示复杂事件产生的时间窗口限制。下面引入一个时间属性关键字 WITHIN，用来描述 RFID 复杂事件检测的事件窗口限制。

A&B WITHIN（T_1，T_2）：表示 A 事件和 B 事件都在 T_1~T_2 时间内发生。即 T_1< Ta <T_2 且 T_1< T_b <T_2。

A| B WITHIN（T_1，T_2）：表示 A 事件或 B 事件在 T_1~T_2 时间内发生。即 T_1< Ta <T_2 或 T_1< T_b <T_2。

~A WITHIN（T_1，T_2）：表示 A 事件在 T_1~T_2 时间内未发生。

SEQ(A,B) WITHIN（T_1，T_2）：表示 A 事件和 B 事件都在 T_1~T_2 时间内发生，且 A 事件早于 B 事件发生。即 T_1< Ta <T_2 、T_1< T_b <T_2 且 Ta < Tb。

通过实际验证，使用这四种操作符加上事件属性关键字的限制来抽象大多数 RFID 系统或应用的商业逻辑已经足够。举个例子：供应链场景中，我们用事件 A 表示某物品必须经过定地点 A 去完成指定的业务流程。若一个货物需在 T 时间内，以此完成 A，B，C 三个业务流程才为合格，漏掉中间流程环节 B

则发出报警，则该物体的正常流程事件为

SEQ（A，B，C）, During(A，C)<T,

若需检查 3 小时内发生的非正常流程事件，则可用

SEQ（A，C）& ~B ,WITHIN (0，3 hour), During(A，C)<T。

2.3.4 RFID 复杂事件表达式

有了前面介绍的事件之间的关系、事件操作符及事件的时间限制属性，便可以根据它们来定义一个具体的 RFID 复杂事件。有了具体的 RFID 复杂事件表达式，便可以用 RFID 复杂事件模型将该复杂事件表达出来。

复杂事件表达式主要包括四部分，表示原始事件的字母、将原始事件组合起来的逻辑操作符、原始事件之间关系约束及时间限制属性。

下面将详细介绍各个部分。

（1）表示原始事件的字母以及将原始事件组合起来的逻辑操作符。原始事件的定义在第二章中已经详细陈述，事件操作符在上文中也做了定义。如：

A&B&C|D 表示 A、B、C 事件发生 或 D 事件发生。

A& SEQ(C，D) 表示 A 事件发生 且 C 事件与 D 事件顺序发生。

SEQ（A，C）& ~B 表示 A 事件与 C 事件顺序发生，但是 B 事件未发生。

（2）约束条件是复杂事件表达式中出现的原始事件之间关系的限制条件。关于复杂事件之间关系定义我们在 2.3.1 节中已经详细论述，包括 Before，After，Equal，During。给复杂事件表达式加上约束条件，可以更精确地表达用户需要的事件，如：

A&B&C|D WHERE A Before B 表示 A、B、C 事件发生且 A 事件先于 B 事件发生或 D 事件发生。

（3）WITHIN 关键字给出该表达式的时间限制，关于 RFID 复杂事件时间属性限制在 2.2.3 小节中已经论述。

最后，我们给出几个典型的 RFID 复杂事件表达式实例。

例 1： A&B During(A，B)<T WITHIN(0，1 Hour)

检测 1 小时内事件 A 先发生， 事件 B 在事件 A 发生之后的 T 时间段内发生的复杂事件。

例 2： A&B&SEQ（C，D） during（C, D）<T WITHIN(0，1 Hour)

检测 1 小时内事件 A 发生，事件 B 发生，事件 C、 D 顺序发生且事件 C 与 D 发生的事件间隔小于 T 的复杂事件。

2.4 基于 Petri 网的事件检测模型

在第一章中曾介绍过，目前的事件检测模型主要有自动机模型，petri 网模型，匹配树模型和有向图模型四种。本文中我们选择具有良好的并发处理和扩展性的 Petri 网作为事件检测的模型。下面介绍 Petri 网的基本概念及使用 Petri 网构建复杂事件模型的知识。

2.4.1 Petri 网的概念

Petri 网是离散并行系统的数学表示[48]。二十世纪六十年代由物理学家卡尔佩特里发明[49]，主要用于从物理学角度去描述并发现象。Petri 网既可以作为一种图形工具又可以作为一种数学工具，用它可以描述条件和事件之间的关系。经过四十多年的发展，目前 Petri 网已经被广泛用于离散事件系统的建模与仿真，并行程序设计，协议验证，数据分析等领域[50]。

一个五元组 N=（P，T，F，W，W_0）被称为一个 Petri 网，当且仅当：

（1）P 是库所的有限集合。

（2）T 是变迁的有限集合，且 $P \cup T \neq \Phi$，$P \cap T = \Phi$。

（3）$F \subseteq (P \times T) \cup (T \times P)$是有向弧的集合。

（4）M：P->{0,1,2…}是有库所的标识函数，M_0 为初始标识。

（5）W:F->{1,2,…}是有向弧的权函数。

由以上定义[51]可以看出，五元组 N=（P, T, F, M, W）的组成元素 P 为库所的有限集，T 为变迁的有限集，P 与 T 至少由一个为非空集合，且二者不相交。F 为从 P 到 T 或从 T 到 P 的有向弧的集合，其输入输出都为库所集合。有向弧连接的变迁的输入库所集合为 $t=\{p|(p,t) \in F\}$，输出库所为 $t=\{p|(t,p) \in F\}$；有向弧连接的库所的输入集合为 $p=\{t|(t,p) \in F\}$，输出库所集合为 $p=\{t|(p,t) \in F\}$。M 为库所的标识函数，初始标识 $M_0=[M_0(P_1), M_0(P_2), \cdots M_0(P_i), \cdots M_0(P_n)]$ $(2<i<n)$，$M_0(P_i)$表示在初始状态下库所 P_i 中的 token 数目。W 是权函数，是 F 中每条弧到该弧上的权值的一个映射集，本文中我们默认 F 中的所有弧权值为 1。

2.4.2 Petri 网实例

下面举一个 Petri 网的实例来分析，下图 2-2 为一个简单的 Petri 网示意图：

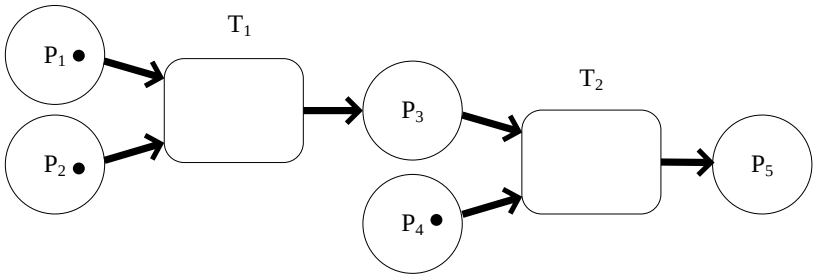

图 2-2 简单 Petri 网示意图

如图所示，黑色的圆圈表示库所，矩形表示变迁，库所内的黑点表示该库所的 token 数目。该图的初始 token 为 $P_1=P_2=P_4=1$，$P_3=P_5=0$。即 $M_0=[1,1,0,1,0]$。图中的箭头方向表示库所内 token 的流动方向，箭头上的数字代表需要该库所资源的数量，称为权值。本文默认权值为 1，故没有写出。Petri 网通过这些 token 的流动来描述一个动态过程。Token 的流动是通过变迁 T 来触发执行的，一个变迁被触发的条件是该变迁的每一个输入库所中的 token 数不少于对应有向弧的权值，在我们的模型中由于权值为 1，即可看作一个变迁被触发的条件时该变迁的每一个输入库所都至少有一个 token。当满足变迁条件时，变迁将会被触发，变迁触发的结果是 token 由变迁的输入库所按照权值流动到变迁的输出库所。

有了 Petri 网的基本理论做基础，便可分析 RFID 复杂事件检测，包括定义复杂事件之间的关系，复杂事件操作符等，然后便可以根据复杂事件检测的流程及要求结合 Petri 网的特点去建立一个基于 Petri 网的复杂事件检测模型。

2.4.3 四种基本逻辑关系的 Petri 网模型

上文已经分析过，复杂事件检测最重要的目的即从海量的原始事件中检测出用户需要的复杂事件，而复杂事件上文中定义为原始事件通过各种逻辑关系并经一定规则而产生的事件。对应 2.4.1 节中 Petri 网的介绍，不难发现，若用库所来代表事件，用变迁来表示事件之间的逻辑关系和组合规则，则使用 Petri 网便可以很好的去解决由原始事件生成复杂事件的检测过程。下面从上文定义的四种基本的逻辑关系入手，使用 Petri 网来表示复杂事件的生成。

（1）A&B，其 Petri 网表示如图 2-3 所示，输入库所为事件 A 与事件 B，变迁为操作符&，当满足变迁条件 A 事件和 B 事件都发生，即库所 A 和库所 B 都有 token 时，触发生成事件 A&B。

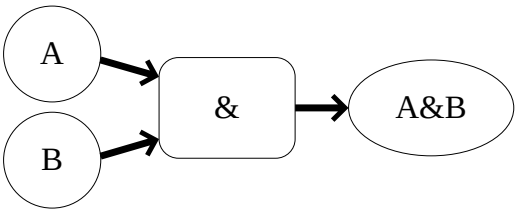

图 2-3 复杂事件 A&B 的 Petri 网表示

（2）A|B，其 Petri 网表示如图 2-4 所示，输入库所为事件 A 与事件 B，变迁为操作符|，当满足变迁条件 A 事件发生或 B 事件发生，即库所 A 内 token 数大于 1 或者库所 B 内 token 数大于 1 时，触发生成事件 A|B。

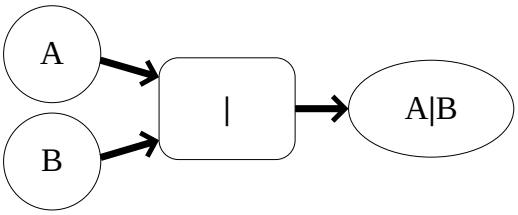

图 2-4 复杂事件 A|B 的 Petri 网表示

（3）~A，其 Petri 网表示如图 2-5 所示，输入库所为事件 A，变迁为操作符~，当满足变迁条件 A 事件在时间间隔 T 内未发生，即库所 A 在时间间隔 T 内的 token 数不大于 1 时，触发生成事件~A WITHIN（T）。

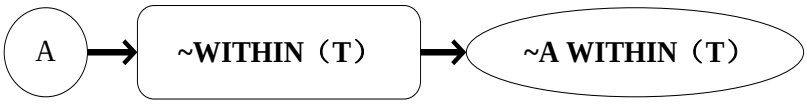

图 2-5 复杂事件~A 的 Petri 网表示

（4）SEQ(A,B)，其 Petri 网表示如图 2-6 所示，输入库所为事件 A 与事件 B，变迁操作符为 SEQ，当满足变迁条件 A 事件发生后 B 事件发生，即库所 A 内 token 数大于 1、库所 B 内 token 数大于 1 且 A Before B 时，触发生成事件 SEQ(A，B)。

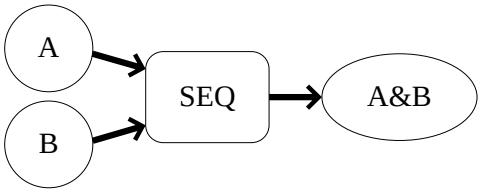

图 2-6 复杂事件 SEQ(A,B)的 Petri 网表示

有了以上四个基本操作符的 Petri 网，只需用用它们的输出库所作为下一个事件的输入库所，便可以通过它们组合出更多的复杂事件。如下图所示，复杂

事件 E=（A&B）|C 即是通过四个基本操作符的 Petri 网进一步组合得到的。

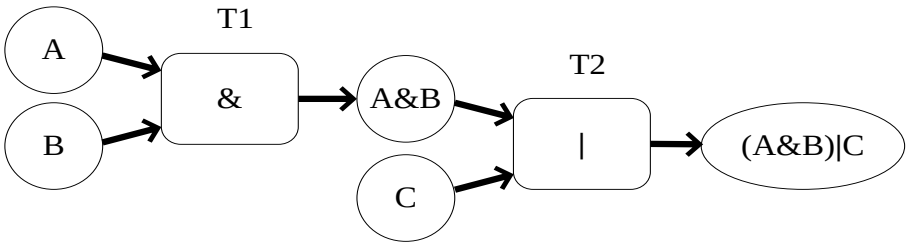

图 2-7 复杂事件 E=（A&B）|C

其 Petri 网表示如图 2-7 所示，若 A 事件与 B 事件都发生，则变迁 T_1 会触发产生复杂事件 A&B，即库所 A&B 的 token 数加 1。而库所 A&B 和库所 C 都未变迁 T_2 的输入库所，只要任何一个库所大于 1 则会触发变迁 T2，生成复杂事件 E=（A&B）|C。

2.5 本章小结

本章对 RFID 复杂事件检测的基本理论进行了介绍，首先介绍了 RFID 事件的生成，包括原始事件与复杂事件的定义等；其次介绍了到 RFID 复杂事件的建模方法，如何使用事件表达式去表示一个 RFID 复杂事件；最后介绍了 Petri 网的基本概念及使用 Petri 网构建复杂事件模型的知识。在本章介绍的基本理论知识的基础上，可以建立一个完整的复杂事件模型及基于 Petri 网的复杂事件检测模型。综上，本章为后续章节具体介绍复杂事件检测算法及基于该算法的复杂事件查询共享策略打下了良好的基础。

第3章 RFID 复杂事件检测算法及其优化策略

3.1 引言

上一章中，介绍了复杂事件处理的基本理论，使我们对复杂事件处理有了从概念到本质的理解。本章根据复杂事件处理的基本理论，首先定义了一个基于时间点的复杂事件模型，然后根据 Petri 网的知识建立了一个基于 Petri 网的复杂事件检测模型。在此基础上，针对于 CESN 检测算法[29]的不足，给出一个改进的复杂事件检测算法。最后，针对不同的复杂事件检模型，分析其可以共享的部分，并给出了两个复杂事件检测优化策略。

3.2 基于 Petri 网的 RFID 复杂事件检测模型

3.2.1 RFID 复杂事件模型

RFID 复杂事件是由 RFID 原始事件通过一定关系组合而成的，因此在定义 RFID 复杂事件模型之前，应当先给出 RFID 原始事件模型的定义。在第二章关于 RFID 复杂事件基本理论的介绍中，曾提到过，RFID 数据的一般格式为（EPC, ReaderID, Time），本文用这个三元组表示一个原始事件，在不同的场景下，赋予其不同的语义。因此，一个原始事件模型中包含以下三个组成部分：

（1）EpcCode：RFID 标签的 tag，表示每一个原始事件的 ID。

（2）ReaderID：读到 RFID 标签的阅读器 ID，表示原始事件的事件类型。

（3）Time：该条 RFID 数据的产生时间，表示每个原始事件的发生时间。

图 3-1 为根据以上定义的 RFID 原始事件的模型。

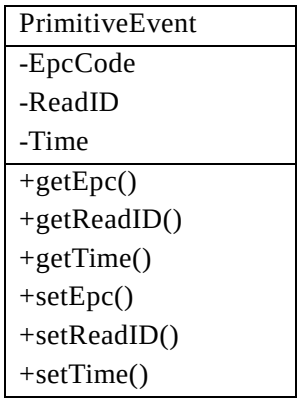

图 3-1 RFID 原始事件的模型

以一个部署 RFID 系统的智能图书馆为例，给图书馆中的每本书上贴一个

RFID 标签，由于每个 RFID 标签都有唯一的 Epc 编码，所以 RFID 原始事件模型中的 EpcCode 可唯一标识一本书籍；在图书馆不同地点如入口，书架，出口等处放置一个阅读器，则 RFID 原始事件模型中的 ReaderID 可标识图书经过的地点；加上 RFID 原始事件模型中的 Time 属性，当一个原始事件产生时，根据这个原始事件内的数据可确定某一本书在某个时间点经过了图书馆的某个确定位置。

有了原始事件模型，便可以在此基础上定义 RFID 复杂事件模型。上文中已经分析过了 RFID 复杂事件的各个组成部分，包括 RFID 事件之间的关系、RFID 复杂事件操作符，RFID 复杂事件的时间属性等。下面对 RFID 复杂事件模型进行整体定义。

一个复杂事件模型包含以下几个部分：

（1） Name：复杂事件的名称。

（2） Expression：复杂事件的表达式，由子事件与操作符加上时间关系属性组成，上文中已有对表达式的说明。

（3） PrimitiveEventList：子事件集合，包含组成该复杂事件的所有子事件。

（4） OperatorList：事件关系列表，包含所有的事件操作符。

（5） SameObject：组成该复杂事件的子事件所代表的物体是否相同，即 EPC 编码是否一样。本文中如非特别说明，默认为是。

（6） TimeLimitation：复杂事件产生的时间属性。

（7） Time：复杂事件的产生时间，表示该复杂事件满足条件的时间。

图 3-2 为根据以上定义的 RFID 复杂事件的模型：

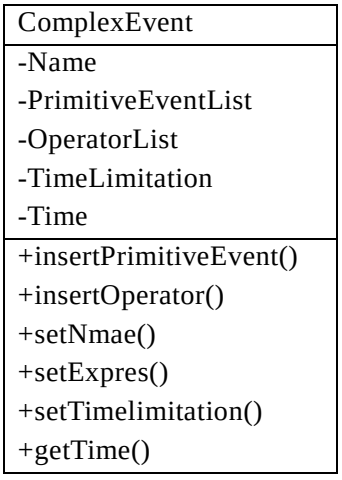

图 3-2 RFID 复杂事件的模型

仍以上文提到的一个部署 RFID 系统的智能图书馆为例，原始事件表示某

本书在某个时间点经过了图书馆的某个确定位置。给出一个表示图书重新进入流通的复杂事件，如下图 3-3 所示：

ComplexEvent：BookRecirculation
-Name：BookRecirculation -Expression：SEQ(Entrance,Return,Shelf) -PrimitiveEventList：Entrance，Return，Shelf -OperatorList：SEQ -TimeLimitation：4hours -Time：14:00:00

图 3-3 复杂事件实例：Book Recirculation

复杂事件名字为图书重新流通，表达式为 SEQ（Entrance，Return，Shelf），表示图书依次经过入口处，图书归还地点，放入书架上。组成该复杂事件的原始事件有 Entrance，Return，Shelf。事件操作符是 SEQ。检测的时间限制是 0 到 4Hour，复合事件发生的事件是 14:00:00。

例如：Entrance<book0001,Entrance,12:00:00>代表书籍 book0001 在 12:00:00 点经过了图书馆的入口处（Entrance）；Return：<book0001,Return,12:05:00>代表书籍 book0001 在 12:05:00 点被放置到了还书指定地点（Return）；Shelf：<book0001,Shelf,14:00:00>代表书籍 book0001 在 14:00:00 点被放置到该书应当存放的书架上（Return）；由于他们满足顺序发生的关系，则复杂事件 BookRecirculation 被触发，复杂事件发生的时间即组成它的子事件中最后发生的 Shelf 事件的时间。该复杂事件表示的语义为某本书被读者归还，且重新上架，重新进入流通环节。

3.2.2 用于检测复杂事件的 Petri 网模型

在 2.4.1 节中，本文已经介绍过一个 Petri 网是一个满足一定条件的五元组（P，T，F，M，W），Petri 网通过这些 token 的流动来描述一个动态过程。而 token 的流动是通过变迁来改变的，当满足变迁条件时，变迁将会被触发，变迁触发的结果是 token 由变迁的输入库所按照权值流动到变迁的输出库所，从而引起 Petri 网状态的改变。由此可知，Petri 网实际上是变迁和库所相互作用引起整个网状态变化的过程，弧集代表了变迁跟库所的组合形式与结构，权函数这里全部为 1，可以不予考虑，初始状态可以根据用户指定或者由系统默认。所以只需从中抽象出库所与变迁两个对象，定义它们的特征与操作，从而可以由它们来构建出一个完整的检测模型。

（1）每个代表事件的库所对象。以第二章中图 2-8 为例，一个库所应包含

的属性有库所名称，用它来表示事件，库所 ID，用它来与 RFID 数据中的 ID 匹配，token 数目，还有以它为输入、输出的变迁集。库所的操作应该有那些呢？首先，库所应当可以设定以其作为输入或输出的变迁；其次库所可以改变自己 token 的数目，当检测到某个它代表的原始事件发生时，给其 token 数加 1；当其作为输入的变迁满足条件触发后，应给予相应的 token 数减 1；再有，当 token 数目改变时，可以通知其连接的变迁检查变迁条件。

通过以上分析，我们给出代表事件的库所 Place 的 UML 设计图，如图 3-4 所示：

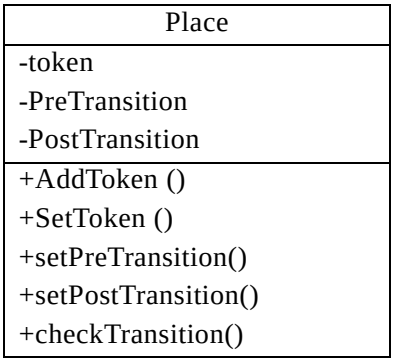

图 3-4 代表事件的库所对象的 UML 图

（2）每个代表事件之间关系的变迁对象。仍以第二章中的图 2-7 为例，来分析表示事件之间关系的变迁对象应包含的属性与操作。从图中可以看出，表示该复杂事件的 Petri 网中包含两个变迁：T_1 和 T_2 ，T_1 表示四种基本的逻辑关系中的“与”，T_2 表示四种基本逻辑关系的“或”。T_1 、T_2 都包含两个输入库所和一个输出库所。推广到表示其他复杂事件的 Petri 网中我们不难发现，不同的复杂事件只是作为输入的库所不同——代表检测的原始事件不同；作为组合事件的变迁不同——代表复杂事件的逻辑关系操作符不同。而我们已经规定好了四种表示事件之间关系的基本螺丝关系操作，因此在设计变迁对象时，我们先定义个表示变迁的基类，其中包含变迁的一些公共属性和必须的操作：输入库所集合和输出库所集合、变迁的约束条件、变迁的时间窗口限制、判断变迁条件是否满足的函数、当变迁条件满足触发变迁后对输入库所及输出库所 token 数目的改变。下面我们先给出表示变迁基类 Transition 的 UML 图，如图 3-5 所示。

其中，函数 Enable()的作用是判断变迁条件是否满足，它将遍历连接到该变迁上的所有库所，检查该库所的 token 数，若满足变迁条件，则变迁发生。函数 Fired()的作用是当变迁发生后改变相应输入输出库所内的 token 数。若该变迁发生，变迁的输入扩所 token 消耗，变迁的输出库所 token 产生。

Transition
-PrePlace -PostPlace
+Enable () +Fire () +insertPlacePre() +insertPlacePost ()

图 3-5 变迁对象基类的 UML 图

在此基础上，我们可以给出由基类 Transition 派生而来的表示四种基本逻辑关系的变迁类 Transition_And, Transition_Or, Transition_Not, Transition_SEQ。它们都拥有基类的属性跟方法，但是不同派生类根据当前变迁的性质需重写使能函数 Enable()及触发函数 Fire()。

3.2.3 根据复杂事件关系建立检测的 Petri 网

根据上文中定义的库所及变迁，便可由任意的复杂事件表达式建立 Petri 网。以复杂事件 E=（A&B）|C 为例，其 Petri 网模型如图 3-6 所示：

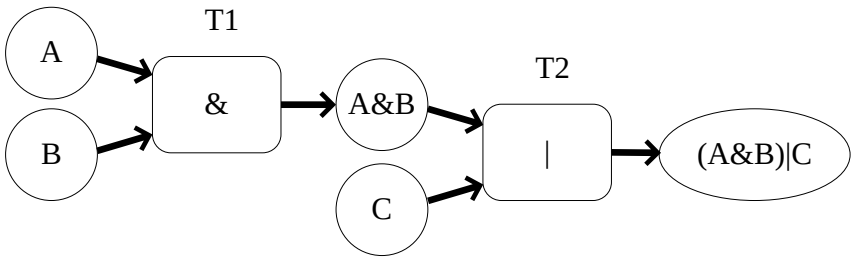

图 3-6 复杂事件 E=（A&B）|C

要建立该复杂事件的 Petri 网，首先需定义模型中出现的 5 个库所：

（1）Place　placeA ，表示原始事件 A

（2）Place　placeB ，表示原始事件 B

（3）Place　placeC ，表示原始事件 C

（4）Place　palceA&B，表示复杂事件 A&B

（5）Place　placeA&B|C，表示复杂事件 A&B|C

然后再定义表示操作符的两个变迁：

（1）Transition_And T1，表示操作符&

（2）Transition_Or　T2，表示操作符|

最后，根据 Petri 网的拓扑结构将相应的变迁与库所联系起来：

（1）库所A与库所B 作为变迁T1 的输入库所

placeA.setPostTransition(T1);

placeB.setPostTransition(T1);

（2）库所A&B与库所C作为变迁T2 的输入库所

placeA&B.setPostTransition(T2);

placeC.setPostTransition(T2);

（3）T1的输入库所为A和库所B，输出库所为palceA&B

T1.insertPlacetoPre(placeA);

T1.insertPlacetoPre(placeB);

T1.insertPlacetoPost（palceA&B）

（4）T2的输入库所为A&B与库所C，输出库所A&B

T2.insertPlacetoPre(palceA&B);

T2.insertPlacetoPre(placeC);

T2.insertPlacetoPost（placeA&B|C）

经过上述连接，就可以将库所及变迁组合成表示该复杂事件的 Petri 网。在原始事件经过该 Petri 网时，经过变迁内部的检测算法进行运算，决定是否发生变迁并传递信号给其输出库所，进而引起 Petri 网状态的改变，最终达到检测复杂事件的目的。

3.3 复杂事件检测算法

3.3.1 CESN 复杂事件检测算法

CESN 检测算法是 Wenhui Hu 于 2008 年提出的算法[29]，这是一种基于 Petri 网的算法。它是在 PRES 系统原形上开发的，基于关注分离的理念，将系统定义为由逻辑结构层、时间限制层和事件处理层三层结构的复杂事件处理中间件。其处理流程如下图 3-7 所示[38]。其核心部分是使用 Petri 网模型构造的复杂事件处理层，该层使用 CESN 检测算法来检测复杂事件。

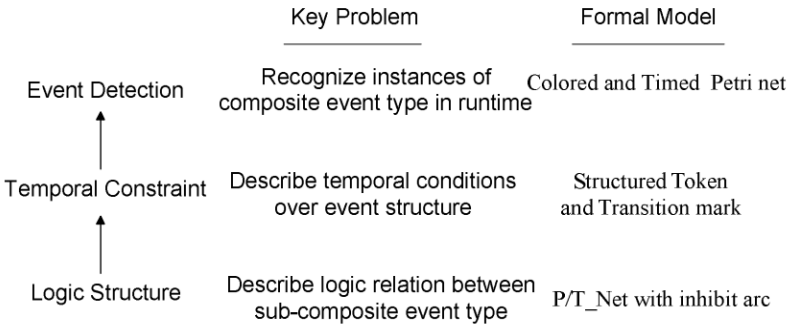

图 3-7 三层结构的复杂事件处理中间件

CESN 算法的处理流程如下：

首先为需检测的复杂事件构建逻辑表达式，并根据逻辑表达式新建变迁和库所，将逻辑表达式中的事件用库所表示，逻辑表达式中的操作符用变迁来表示。然后为该逻辑表达式加上时间限制。时间限制是在每一个变迁中设置 Guard 函数，由变迁内的 guard 函数去判定输入库所是否满足条件。变迁中除了要检查输入的库所是否满足变迁条件外，还需要设定输出库所的时间属性。因此算法在变迁中设置了一个 body 函数，用来根据输入库所的时间计算输出库所的时间属性。 最后根据复杂时间表达式构建 Petri 网，让原始事件经过 Petri 网，开始检测。具体做法是：对于不同的操作符，当变迁可以被激发时检查 guard 函数是否满足，若满足，则根据 body 函数输出库所，若不满足，则删除不满足条件的库所，继续检测。

我们以复杂事件 E=sequence(A&B)[$start_b$-end_a>0]为例，如图 3-8 所示，说明算法的执行过程。

首先，根据逻辑表达式，为其新建库所 A、B、E，代表事件 A、事件 B 和事件 E，新建变迁来表示 sequence 运算符。

其次，设置变迁内的时间限制 guard 函数，条件为 startb-enda ，body 函数，为 starte=min(starta,startb), ende=max(enda,endb)。

最后，让输入的事件流经过该复杂事件。当 A 事件产生时，库所 A 的 token 加一，并判断 B 是否有 token，若 B 内有 token，接着判断事件 A 与事件 B 是否满足 guard 内的时间限制条件，若满足则触发生成复杂时间 E，E 的时间属性根据 body 函数确定；若 B 没有库所，则继续检测下一个输入的原始事件。若 B 事件产生，库所 B 的 token 加 1，然后判断 A 事件内是否有 token，若 A 内有 token，则判断事件 A 与事件 B 是否满足 guard 内的时间限制条件，若满足则触发生成复杂时间 E，否则继续检测。

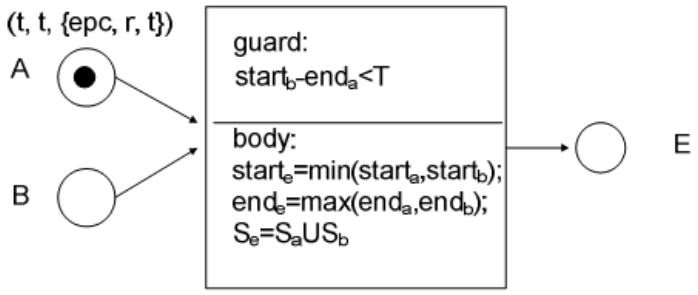

图 3-8 E=sequence(A&B)[startb-enda>0]

CESN 检测算法的核心是根据需检测的复杂事件逻辑表达式建立由库所和

变迁构成的 Petri 匹配网络结构，并在变迁内添加判定函数确保符合条件的模式被触发。它可以做到迅速在输入的简单事件流内检测到用户需要的复杂事件，但是算法中有以下几点尚需改进：

（1）Token 内存放了许多不必要的重复信息，当时间检测的事件窗口较长时，容易造成内存溢出。对于原始事件来说，RFID 数据{epc，r，t}中的 t 和 token 内的 t_s、t_e 是相等的，因此只存放一个用于标识时间就可以；同时，代表时间的库所 A 和 RFID 数据中阅读器表示应为一一对应的，故可以使用一个作为标识。

（2）存在大量回溯：在检测顺序事件时，算法的步骤是先判断组成该顺序事件的子事件是否都具有 token，当所有子事件都具有 token 后，再根据 guard 中的时间限制去判断是否满足顺序发生的时间限制。这样势必要保留很多子事件的 token，即使时间限制不满足也须等到所有子事件都到来时才重新回溯检测，进行判定。当数据量大时，无法保证处理速度，需优化。

3.3.2 改进的复杂事件检测算法

CESN 检测算法与其说是一种算法，倒不如说是一种方法，其核心的思想是将分离的逻辑表达式中的元素和时间限制条件，根据逻辑关系动态构造 Petri 网，以达到检测复杂事件的目的。因此我们根据 RFID 数据和事件的语义，设计相应的事件模型和 Petri 网，通过针对该算法存在的不足做出两点改进：

（1）优化了 Token 的数据结构，去掉了 CESN 算法中介绍的不必要的重复信息，对于原始事件来说，RFID 数据{epc，r，t}中的 t 和 token 内的 t_s、t_e 是相等的，改进算法中的 token 内只存放一个用于标识时间 t；同时，代表时间的库所 A 和 RFID 数据中阅读器表示应为一一对应的，故改进算法中的 token 内使用库所名称作为事件标识，去掉阅读器标识 r。

（2）针对存在大量回溯的情况：在检测顺序事件时，算法的步骤不是先判断是否所有子事件都具有 token，而是在一个原始事件到来的时候，就根据时间限制条件去判断该子事件之前的事件是否发生，若位于该子事件之前顺序序列的事件中有一个尚未发生，则直接删除该子事件；若位于该子事件之前的顺序序列中的所有子事件均发生，则保存该子事件，继续判断顺序序列中的后续事件。这样将会大大降低保留的子事件的 token 数目，在处理海量的原始事件时可以保证较高的处理速度，为事件检测引擎节省大量的内存空间。

根据以上对算法的改进，提出一个复杂事件检测算法 PCED（Petri-Net Based Complex Event Detecting Algorithm），以期可以减小所利用的内存空间，

大幅提高 RFID 复杂事件检测的处理效率。具体的改进算法将在下面 RFID 复杂事件检测算法 PCED 实现部分给予详细的说明。在给出该算法的详细描述之前，首先需要给出一些前提条件。

（1）事件流到来有序。PCED 算法处理的数据是 RFID 原始事件流，假设这些 RFID 原始事件是按照本身事件发生的时间顺利依次达到，即先发生的事件先到达，后发生的事件后到达，不存在乱序的情况。

（2）事件的消耗策略为 Chronicle。复杂事件检测的另一重要影响因素是事件的消耗策略，也有文献称为事件的组合方式。事件消耗策略定义了如何使用子事件去生成新的复杂事件，文献[31]研究了四种事件消耗策略，分别是 Recent、Continuous、Cumulative、Chronicle。大多数的 RFID 复杂事件检测方法使用 Chronicle 消耗策略，本文中使用的也是 Chronicle 消耗策略。在 Chronicle 消耗策略下，组成复杂事件的子事件以 Chronicle 方式组合：最早发生的起始事件与最早发生的结束事件组合，生成一个新的复杂事件。例如，欲检测复杂事件 A&B，当输入的复杂事件流为 a1，a2，b1，b2（小写字母代表该事件的一个实例）时，使用 Chronicle 消耗策略，事件 a1 将与 b1 匹配，生成复杂事件 A&B，事件 a2 将与 b2 配对，也生成复杂事件 A&B，但是事件 a2 不能与 b1 配对，生成复杂事件 A&B。

在不同的应用场景下，可能需要检测不同的复杂事件。因此检测某一场景下的复杂事件时，需要事先定义好该应用场景所需检测的复杂事件表达式和时间限制条件。然后再根据复杂事件表达式及时间限制给出基于 Petri 网的复杂事件检测模型。最终让 RFID 原始事件流依次经过这个检测模型，当满足了复杂事件产生的条件时，检测模型会触发生成相应的复杂事件。

在第三章中已经提到，复杂事件是原始事件通过各种逻辑关系并经一定规则而产生的事件，我们定义了代表复杂事件关系的四种变迁，用它来检测复杂事件。下面具体描述每种变迁内的检测算法。

3.3.3 复杂事件操作符

（1）“与”操作符关联的复杂事件检测

代表操作符&的变迁的语义为：变迁的所有输入库所都有 token 时，触发生成复杂事件。它的触发与时间属性无关，与输入库所的先后次序无关。变迁 Transition_And 的触发规则如下：

Input ：关联某一实体的 EPC 编码

Output：变迁是否可以触发，True or False

Transition_And::enable(string EPC)

1) *length = 变迁的输入库所个数*;
2) If *length==0* then
3) return *Enable*
4) Else
5) For *i=1 到 length*
6) *place 为 变迁的第 i 个输入库所*
7) *N 为 place 中关联该 EPC 实体的 token 数*
8) If *token <1*
9) Break;
10) If *i== length*
11) *Enable = true*
12) return *Enable*

从 Transition_And 的触发判定算法中可以看出，它将根据输入的 EPC 标识逐一检查它的输入 place（库所），一旦有一个 place 没有 token，即认为组成该复杂事件的子事件中还有事件尚未发生，则不能满足关系“与”的变迁条件，返回 false，退出。若当所有库所都检查过了，全都有 token，即组成该复杂事件的所有子事件均有实例发生，此时满足关系“与”，使能判断函数为真，返回 true，可以变迁。

（2）“或”操作符的复杂事件检测

变迁为操作符 | 的语义为变迁的所有输入库中任意有一个库所有 token 时，触发生成复杂事件。它的触发与输入库所的先后次序无关。变迁 Transition_ Or 的触发规则如下所描述：

Input ：关联某一实体的 EPC 编码

Output：变迁是否可以触发，True or False

Transition_Or::enable(string EPC)

1) *length = 变迁的输入库所个数;*
2) If *length==0 then*
3) *return Enable*
4) Else
5) For *i=1 到 length*
6) *place 为 变迁的第 i 个输入库所*
7) *N 为 place 中关联该 EPC 实体的 token 数*
8) If *token > =1*
9) *Enable = true*

10) *Break;*
11) If *i== length*
12) *Enable =false*
13) return *Enable*

从 Transition_Or 的触发判定函数中可以看出,它将根据输入的 EPC 标识逐一检查它的输入 place（库所），一旦有一个 place 有 token，即组成该复杂事件的子事件中有一个实例发生，则说明变迁条件 or 已经满足 ，可以触发变迁。此时设置变迁标志为真，然后退出。若所有库所都检查过了，全都没有 token，则设置变迁标识为假，使能判断函数返回假，不可以触发变迁。

（3）“非”操作符的复杂事件检测

变迁为操作符 ~ 的语义为变迁的输入库所在变迁的时间窗口限制内未检测到该输入库所有 token 时，触发生成复杂事件。

变迁 Transition_Not 的触发规则如下：

Input ：关联某一实体的 EPC 编码

Output：变迁是否可以触发，True or False

Transition_Not::enable(string EPC)

1) *length = 变迁的输入库所个数;*
2) If *length==0 then*
3) *return Enable*
4) Else
5) For *i = location-1 到 1*
6) *place 为 变迁的第 i 个输入库所*
7) *N 为 place 中关联该 EPC 实体的 token 数*
8) *If token >= 1*
9) *Break;*
10) *If Location == length and i==0*
11) *Enable =true*
12)return *Enable*

从 Transition_Not 的触发判定函数中可以看出，它将在时间窗口结束后检查它的输入 place（库所），若 place 有 token，则说明该事件有实例发生，不满足满足变迁条件 Not ；若该 place 没有 token，则说明该事件在时间窗口内未发生，则触发变迁，然后退出。

（4）顺序操作符的复杂事件检测

变迁为操作符 SEQ 的语义为变迁的所有输入库所都有 token 时，触发生成复杂事件，且它的触发与输入库所的先后次序有关：当且仅当所有库所按照先

后次序依次发生，该变迁才会发生。变迁 Transition_ SEQ 的触发规则如下描述：

Input ： 关联某一实体的 EPC 编码

Output：变迁是否可以触发，True or False

Transition_Seq::enable(string EPC)

1) *length = 变迁的输入库所个数;*
2) If *length==0 then*
3) *return Enable*
4) Else
5) *Location=当前输入的事件位于 SEQ 子事件列表中的位置;*
6) For *i = location-1 到 1*
7) *place 为 变迁的第 i 个输入库所*
8) *N 为 place 中关联该 EPC 实体的 token 数*
9) If *token <= 1*
10) *Break;*
11) If *Location == length and i==0*
12) *Enable =true*
13)return *Enable*

从 Transition_SEQ 的触发判定函数中可以看出，当某一子事件经过 SEQ 检测时，首先检测该子事件之前的子事件是否已经发生，即先检查时间限制是否满足，若位于该事件之前的子事件尚未发生，则之后到来的时间不可能与之组成顺序发生关系，直接将该事件丢弃。若该事件之前子事件已经发生，说明该事件与其之前的子事件满足时间上先后发生关系，将该事件保留。检测到某一事件为顺序序列中的最后一个子事件，则若其满足条件，则可直接出发生成该顺序关系的复杂事件。上文已经分析过，在 CESN 检测算法中，到来一个事件时，首先判断是否是组成该顺序关系复杂事件的子事件，若是则直接给相应的库所 token 数加 1，等到组成该顺序关系的最后一个子事件到来时，在根据时间限制去判断其之前的子事件是否均满足时间限制，若是则产生顺序事件，若否则删除该事件。通过与 CESN 算法的比较可以发现，CESN 检测算法在中间过程中保留了很多未必会产生顺序事件的子事件，等到最后再一次判定是否满足条件，这样势必增大了事件处理的计算量，耗费了更多的内存空间。由此分析也可发现，改进的算法将大大减少内存消耗，提高检测效率。

3.4 RFID 复杂事件优化策略

3.4.1 发掘 Petri 网模型相似性及可共享的资源

上一节中给出了改进的 RFID 复杂事件检测算法。可以看出，对于每一个需检测的复杂事件，需要根据组成该复杂事件的子事件及子事件之间的关系建立相应的事件库所和操作变迁，将它们组合起来，最后进行检测。

由于组成复杂事件的基本变迁只有四个，因此当需要检测的复杂事件很多时，两个以上复杂事件的 Petri 网中可能会出现公共部分，如图 3-9 与图 3-10 所示，一个变迁&及两个库所——库所 B 及库所 C，它们在两个复杂事件中都出现的情形。而一种变迁内进行的算法步骤都是相同的，那么能否能找出这些在多处出现的变迁及库所的特点，寻找他们可以共用的条件，若是满足一定条件下它们可以共用，怎必将大大降低系统的内存消耗及检测时间开支，从而提高系统的效率。基于这个出发点，结合如图 3-9 与图 3-10 所示的具体的复杂事件检测实例，来进行分析。

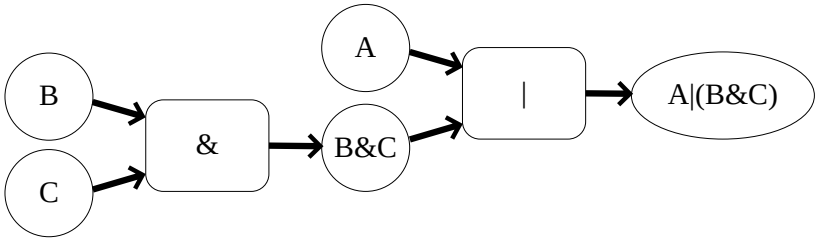

图 3-9 复杂事件 A|（B&C）

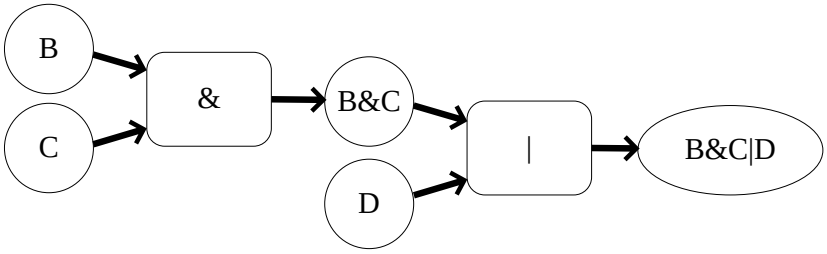

图 3-10 复杂事件（B&C）|D

从图上我们可以看出，两个复杂事件拥有公共的子事件 B&C。在二者的 Petri 网模型中，有以下元素是相同的：库所 B，库所 C，变迁&，库所 B&C。若两复杂事件的检测时间限制都为默认值，则可以二者的公共部分是可以共享的。

接下来再来分析另一种情况。有两个需要检测的复杂事件分别如下：

（1） E_1=A&B WITHIN（0, T_1）

（2） E_2=A&B WITHIN（0, T_2）

其中，$T_1 < T_2$ 。按照上文之前的叙述，若需要检测这两个复杂事件，应根据它们的事件表达式，分别建立 Petri 网，然后再去检测。但是这两个复杂事件除了在时间限制上不同之外，其余均相同。再来分析时间限制上的不同，有 $T_1 < T_2$ ，说明事件 E_1 检测的时间窗口是包含于事件 E_2 内的，因此在检测时间 E_2 的过程中，实际上又重复了一遍检测事件 E_1 的过程。因此若这两个事件可以共享检测结构，则可以省去检测时间 E_1 消耗的内存和时间资源，势必提高系统检测的效率。

3.4.2 优化策略

本文根据以上分析，将复杂事件检测中出现的两种典型情况总结如下，并给出两种情况下相应的优化策略。

（1）子事件共享策略。适用于两事件窗口限制相同的复杂事件，若其事件表达式的 Petri 网中含有公共部分，可以共享公共部分。此时，需要给共享部分的最终库所为两个输出变迁设置两组 token，两个输出变迁分别消耗对应组的 token，两组事件的生成彼此互不影响。

上文中分析的图 3-9 与图 3-10 所示的两个复杂事件共享子事件后的 Petri 网如下图 3-11 所示：

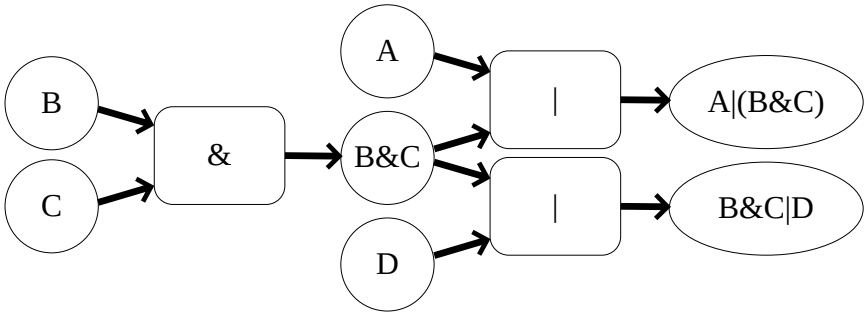

图 3-11 共享子事件后的 Petri 网

（2）时间窗口共享策略。该策略适用于两复杂事件的事件表达式相同，但是时间窗口限制不同的情况。假设两事件检测的时间窗口开始时间相同，都未 T0，结束时间分别为 T1 和 T2，则必有 T1<T2，或者 T2<T1。通过比较可得到时间窗口较大的一个复杂事件 E1，如图 3-12 所示。则时间窗口被其包含的另一事件 E2 可以共享 E1 在 E2 检测时间限制窗口内的检测结果。

通过在复杂事件 E_1 内设置一个时间共享标志，在共享时间区间内，将 E_1 的检测结果分别输出到 E_1 与 E_2 的检测结果记录内，当离开共享时间区间后，仅将检测结果输出到 E_1 本身的检测结果记录内。这样在二者的公共检测时间窗口内，只需检测一次即可获得两个复杂事件的结果。

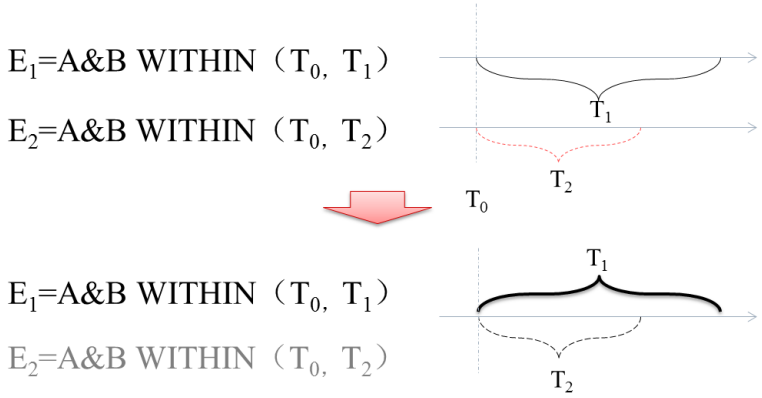

图 3-12 可共享时间窗口的两事件 E_1 与 E_2

3.5 本章小结

本章首先定义了一个基于时间点的事件模型，然后根据 Petri 网的知识建立了一个基于 Petri 网的复杂时间检测模型。之后介绍了 CESN 复杂事件检测算法，并针对与 CESN 检测算法的不足，给出一个改进的基于 Petri 网的复杂事件检测算法。在此基础上，针对需要检测的不同的复杂事件模型，分析其可以共享的部分，在此基础上给出了两种事件检测优化策略。

第4章　系统实现及实验分析

4.1 引言

为了比较改进后算法性能及优化策略对算法性能的影响，本文模拟了一个RFID场景实验，构建了该场景下的各种复杂事件，通过实验结果来验证RFID事件检测算法和检测优化策略的正确性及效率的提高。最后，本文给出了一个一般场景下的RFID复杂事件检测的系统实现，通过检测该场景下所需的复杂事件，然后生成事件报告及对事件的统计分析，从而可以实现对该场景的管理和监控。

4.2 RFID复杂事件检测实验分析

4.2.1 实验场景说明

本次实验中，我们模拟一个部署RFID设备的超市场景，如图4-1所示：

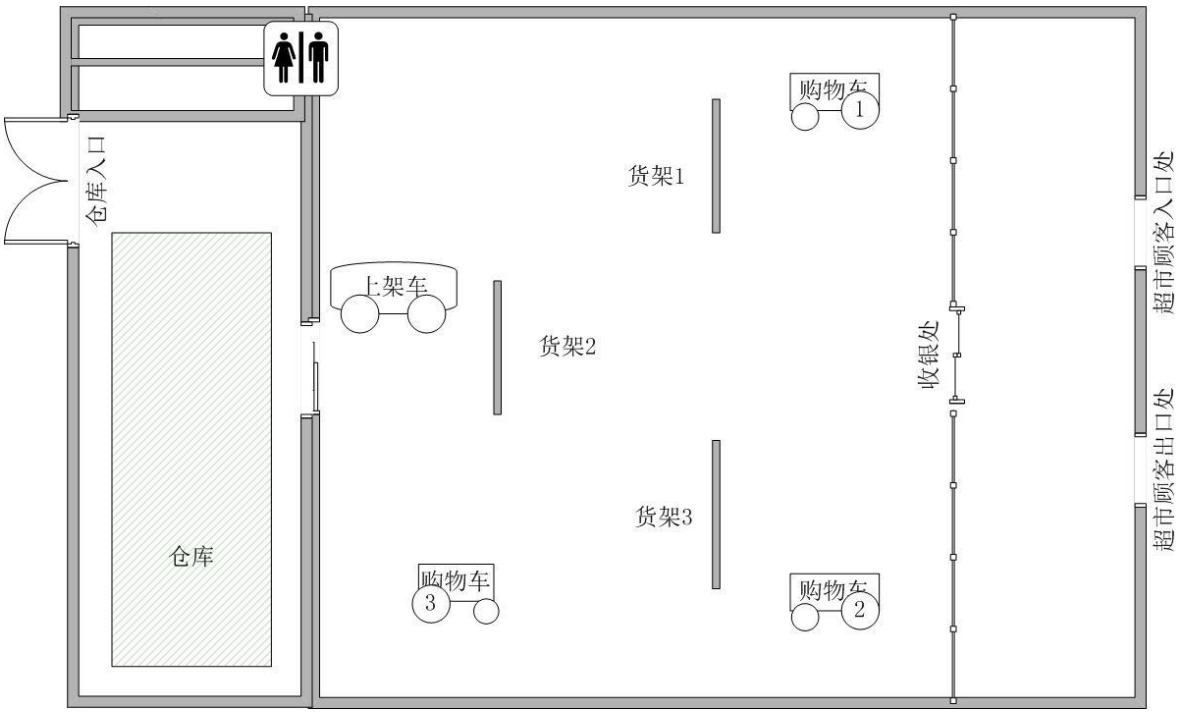

图4-1 部署RFID设备的超市场景

该场景下部署RFID阅读器的地点有超市仓库入口，超市仓库，超市上架车，超市顾客入口，货架1，货架2，货架3，购物车1，购物车2，购物车3，超市

收银处，超市顾客出口。当给每件商品都贴上一个 RFID 标签，则可以检测每一个商品在超市内的移动。通过建立复杂事件模型，可以更好的对超市的商品进行监控与管理。下面由分析事件开始，验证复杂事件检测过程。

4.2.2 实验数据

由于目前没用在超市场景下公开的 RFID 原始数据集，因此本文使用 Matlab 仿真生成所需的数据集。

在该场景下，每个部署 RFID 设备的地点都会产生 RFID 数据，我们在仿真数据生成程序内分别模拟不同的阅读器产生数据，每个阅读器产生的数据格式都是固定的<EPC,ReadID,Time>三元组。其中 EPC 字段代代表该场景下所有商品编号的数量，其编码范围是固定的；ReaderID 即为产生该数据的阅读器名称；Time 字段即为该条 RFID 数据产生的时间。

该场景下部署 RFID 阅读器的地点有超市仓库入口，超市仓库，超市上架车，超市顾客入口，货架 1，货架 2，货架 3，购物车 1，购物车 2，购物车 3，超市收银处，超市顾客出口，共 12 个阅读器。为了记录生成的复杂事件结果，从而方便对实验结果进行验证。对于不同的复杂事件，我们分别生成该类复杂事件的仿真数据。以商品上架事件为例，实验数据产生的过程如下：

产生 Warehouse 阅读器的数据 <EPC,Warehouse,Time1>。

产生 Truck 阅读器的数据 <EPC,Truck,Time2>，Time2 为 Time1 加上一个时间间隔 dt1,在加上一个时间随机数 R，随机数的范围可以根据场景由程序指定。

产生 ShelfOne 阅读器的数据 <EPC,ShelfOne,Time3>，Time3 也为 Time1 加上一个时间间隔 dt2，且 dt2> dt1 ，最后再在加上一个时间随机数 R，随机数的范围与由程序指定，且与上文中保持一致。

最后在生成的上架事件数据集内统计 EPC 相等，阅读器标识分别为 Warehouse、Truck、ShelfOne 的数据，若有 Time1> Time2> Time3，则统计入库复杂事件数量增加 1。

通过以上数据仿真方法，本文仿真产生了大量数据，具体的数据集介绍如下表 4-1 所示，数据集 1 中仿真生成了该场景下的 100,000 个商品的 RFID 数据。数据集 2 中仿真生成了该场景下的 50,000 个商品的 RFID 数据。数据集 3 中仿真生成了该场景下的 150,000 个商品的 RFID 数据。数据集 4 中仿真生成了该场景下的 60,000 个商品的 RFID 数据。

表 4-1 各数据集中包含的原始事件数量

原始事件	事件集 1	事件集 2	事件集 3	事件集 4
WarehouseEntrance	100,000	50,000	150,000	60,000
Warehouse	100,000	50,000	150,000	60,000
Truck	100,000	50,000	150,000	60,000
ShelfOne	100,000	50,000	150,000	60,000
ShoppingCartOne	100,000	50,000	150,000	60,000
CheckOut	100,000	50,000	150,000	60,000
Entrance	100,000	50,000	150,000	60,000
Exit	100,000	50,000	150,000	60,000

4.2.3 事件定义

首先定义该场景下的原始事件，由上文的分析可知，该场景下应定义以下原始事件：

WarehouseEntrance：表示商品经过超市仓库大门；

Warehouse：表示商品存放入超市仓库；

Truck：表示商品被放入超市上架车，准备上架；

ShelfOne：表示商品被放置到了货架 1；

ShelfTwo：表示商品被放置到了货架 2；

ShelfThree：表示商品被放置到了货架 3；

ShoppingCartOne：表示商品被放置到了购物车 1；

ShoppingCartTwo：表示商品被放置到了购物车 2；

ShoppingCartThree：表示商品被放置到了购物车 3；

CheckOut：表示商品经过了超市收银处；

Entrance：表示商品经过了超市顾客入口；

Exit：表示商品经过了市顾客出口。

有了原始事件定义，我们在根据该场景下的业务流程定义复杂事件。首先来分析该场景——超市的业务流程。

超市的业务流程主要有：

（1）进货入仓：商品经过采购，进入超市仓库中存放。此流程商品由超市仓库入口进入超市仓库。

（2）商品上架：商品通过超市上架车从超市仓库运送到商品指定的货架。

（3）商品下架：商品通过超市上架车从货架运回超市仓库。

（4）顾客购买商品：顾客将商品从货架取下，放入购物车，经过收银处付

账，最后将商品从超市顾客出口带出。

（5）商品失窃：商品从货架上离开，未经过收银点，直接由超市顾客出口离开超市。

（6）商品移位：商品被顾客从一个货架放到另一个货架。

除了超市固有的一些业务流程，还可以通过 RFID 系统来分析用户的购买习惯。如：

（7）超市欲将商品 A 与商品 B 绑定促销，为提前预计促销的效果，需统计某天内商品 A 与商品 B 同时被购买的事件。

（8）某商家共有两三种商品 C、D，E 在超市内销售，该商家为了促销，表示购买任一款该商家商品均有精美礼品赠送。

有了原始事件的定义，加上上文对业务流程和一些超市需要的用户行为分析，可以为每一个用该场景下要监控的业务建立复杂事件。系统建立如下复杂事件：

（1）进货入仓：

SEQ（WarehouseEntrance，Warehouse）

（2）商品上架：

SEQ（Warehouse，Truck，ShelfOne）

SEQ（Warehouse，Truck，ShelfTwo）

SEQ（Warehouse，Truck，ShelfThree）

（3）商品下架：

SEQ（ShelfOne， Truck，Warehouse）

SEQ（ShelfTwo， Truck，Warehouse）

SEQ（ShelfThree， Truck，Warehouse）

（4）顾客购买商品：

SEQ（ShelfOne，ShoppingCartOne，CheckOut，Exit）

SEQ（ShelfTwo，ShoppingCartOne，CheckOut，Exit）

SEQ（ShelfThree，ShoppingCartOne，CheckOut，Exit）

（5）商品失窃：

SEQ（ShelfOne，~CheckOut，Exit）

SEQ（ShelfOne，~CheckOut，Exit）

SEQ（ShelfOne，~CheckOut，Exit）

（6）商品移位：

SEQ（ShelfOne，ShelfTwo）

SEQ（ShelfOne，ShelfThree）

SEQ（ShelfTwo，ShelfThree）

SEQ（ShelfTwo，ShelfOne）

SEQ（ShelfThree，ShelfOne）

SEQ（ShelfThree，ShelfTwo）

（7）商品 A 与商品 B 同时被购买：

SEQ（ShelfOne，ShoppingCartOne，CheckOut，Exit）productType=’A’ And

SEQ（ShelfOne，ShoppingCartOne，CheckOut，Exit）productType=’B’

（8）商品 C、D，E 中任一款被购买

SEQ（ShelfOne，ShoppingCartOne，CheckOut，Exit）productType=’A’ Or

SEQ（ShelfTwo，ShoppingCartOne，CheckOut，Exit）productType=’B’ Or

SEQ（ShelfThree，ShoppingCartOne，CheckOut，Exit）productType=’C’

4.2.4 建立复杂事件检测模型

在上一节中，本文定义了在部署了 RFID 设备的超市场景下的 RFID 的原始事件和需要检测的复杂事件。下面介绍上文定义的复杂事件的检测模型。由于篇幅原因，不能对场景下每一个复杂事件的检测模型进行一一介绍，只选择具有代表性的几种事件检测模型进行分析。

（1） 商品上架：SEQ（Warehouse，Truck，ShelfOne）

该复杂事件的 Petri 网模型如下图 4-2 所示：

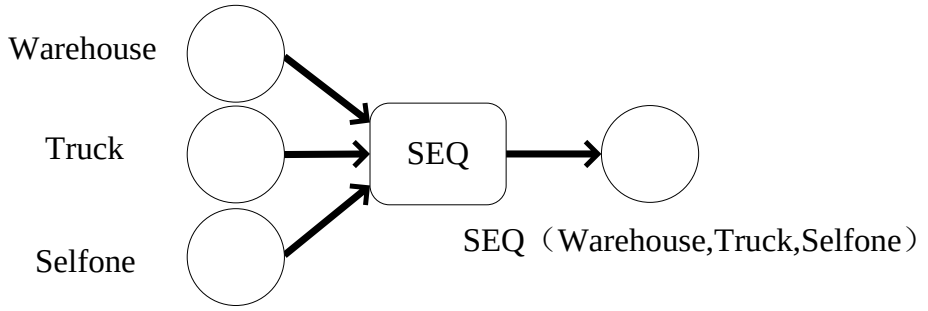

图 4-2 商品上架事件的 Petri 网

当商品依次由仓库，放入上架车，最后放到货架上，则认为该商品已经上架。

（2） 顾客购买商品： SEQ（ShelfOne，ShoppingCartOne，CheckOut，Exit）

该复杂事件的 Petri 网模型如下图 4-3 所示：

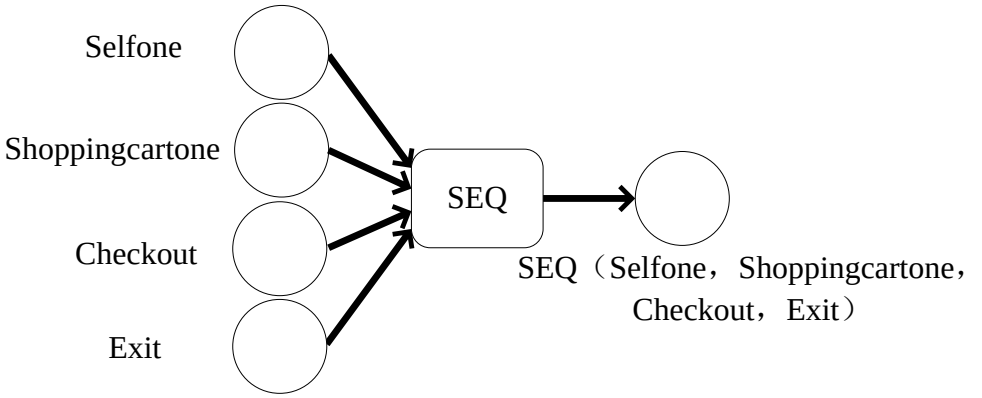

图 4-3 顾客购买商品事件的 Petri 网

上图可以看出，顾客购买商品事件实际上是顾客将商品从货架取下，放入购物车，经过收银处付账，最后将商品从超市顾客出口带出这四个基本事件依次发生。

（3） 商品失窃： SEQ（ShelfOne，~CheckOut，Exit）

该复杂事件的 Petri 网模型如下图 4-4 所示

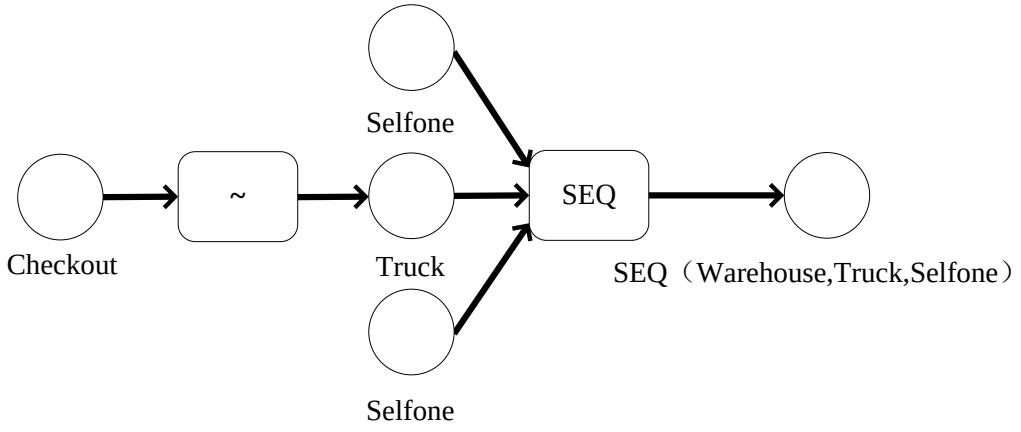

图 4-4 商品失窃事件的 Petri 网

商品从货架上离开，未经过收银点，直接由超市顾客出口离开超市。

（4） 商品 A 与商品 B 同时被购买：

SEQ（ShelfOne，ShoppingCartOne，CheckOut，Exit）productType=’A’ And

SEQ（ShelfOne，ShoppingCartOne，CheckOut，Exit）productType=’B’

该复杂事件的 Petri 网模型如下图 4-5 所示：

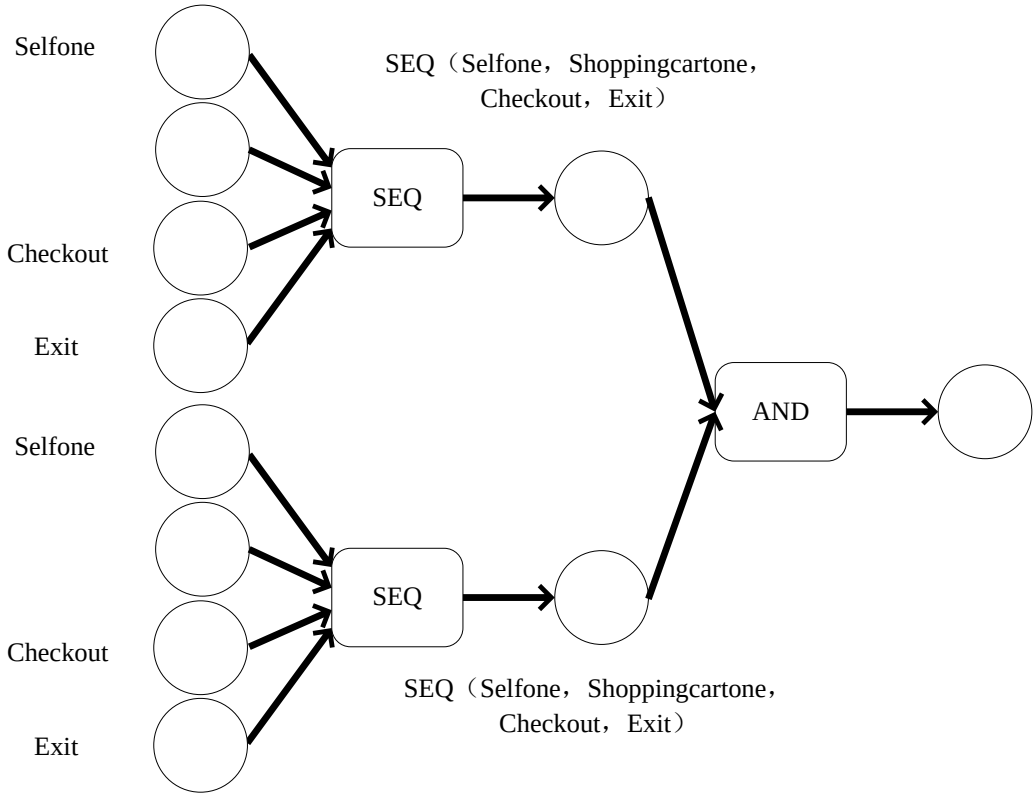

图 4-5 商品 A 与商品 B 同时被购买

（5） 商品 C、D，E 中任一款被购买

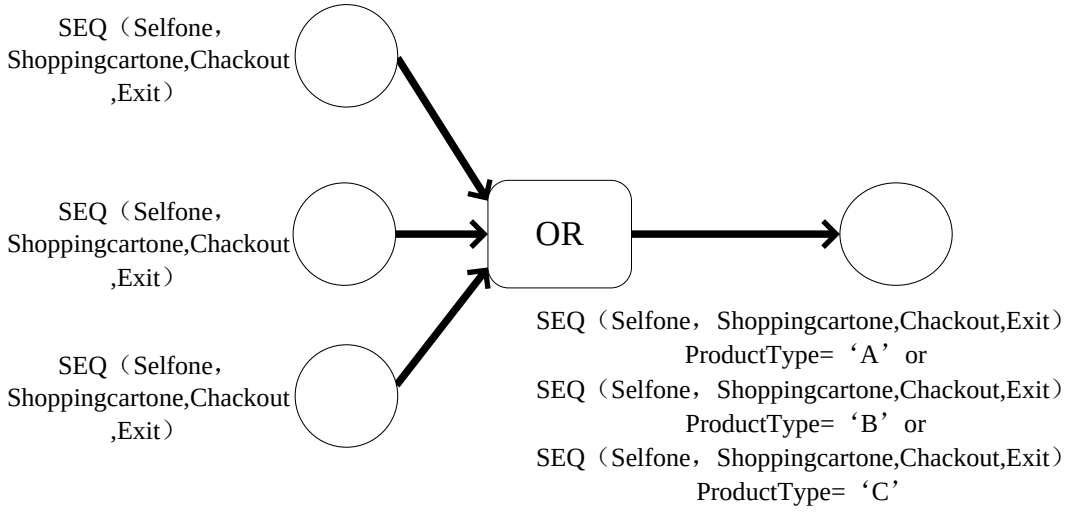

图 4-6 事件商品 C、D，E 中任一款被购买的 Petri 网

SEQ（ShelfOne，ShoppingCartOne，CheckOut，Exit）productType=’A’ Or
SEQ（ShelfTwo，ShoppingCartOne，CheckOut，Exit）productType=’B’ Or
SEQ（ShelfThree，ShoppingCartOne，CheckOut，Exit）productType=’C’

该复杂事件的 Petri 网模型如上图 4-6 所示。

4.2.5 验证改进后的复杂事件检测算法

（1）复杂事件检测率：首先验证改进算法的正确性，通过模拟数据分别产生不同数量的复杂事件，再经过改进算法进行检测，验证是否可以检测到所有预期的复杂事件。实验结果如图 4-7 所示：

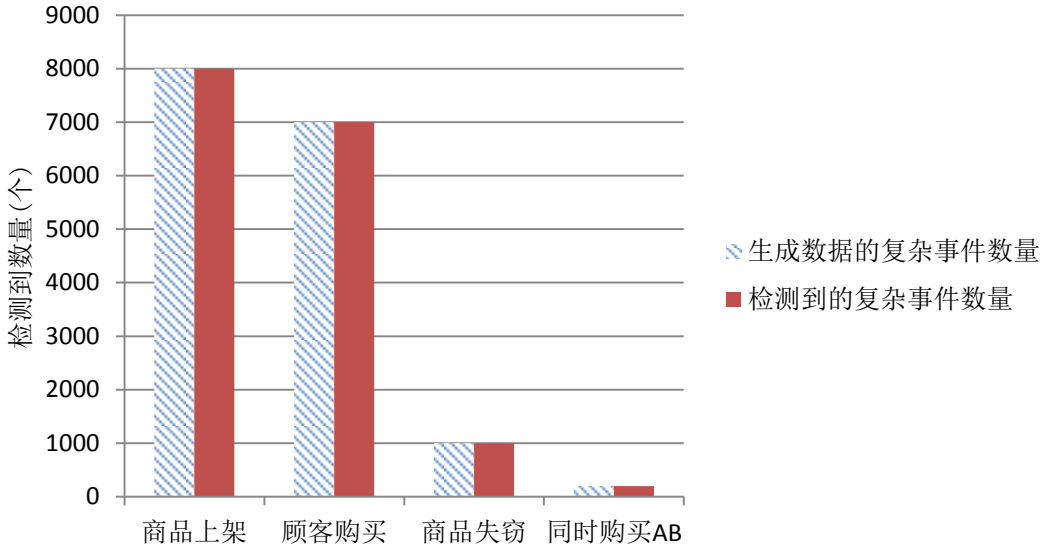

图 4-7 复杂事件检测结果验证

通过实验，将模拟生成的数据经过复杂时间检测算法检测后，生成的复杂事件数据量均等于检测到的复杂事件数，由此可知，我们的检测算法可以检测到生成的复杂事件，算法的是正确可行的。

（2）内存消耗：为了验证改进算法在处理海量 RFID 数据时可以有效节省内存空间，防止溢出，我们对比相同输入原始事件数据下 CESN 算法和改进算法的内存消耗。实验结果如图 4-8 所示：

其中，事件集 1 中含有复杂事件商品上架：SEQ（Warehouse，Truck，ShelfOne）10000 件；事件集 2 中含有复杂事件商品失窃： SEQ（ShelfOne，~CheckOut，Exit）5000 件；事件集 3 中含有复杂事件顾客购买商品：SEQ（ShelfOne，ShoppingCartOne，CheckOut，Exit）15000 件；事件集 4 中含有复杂事件 商品 A 与商品 B 同时被购买 2500 件。通过对比算法运行过程中内存消耗的峰值可以看出，在检测相同数据量大小的原始事件集下，改进算法所消耗的内存均小

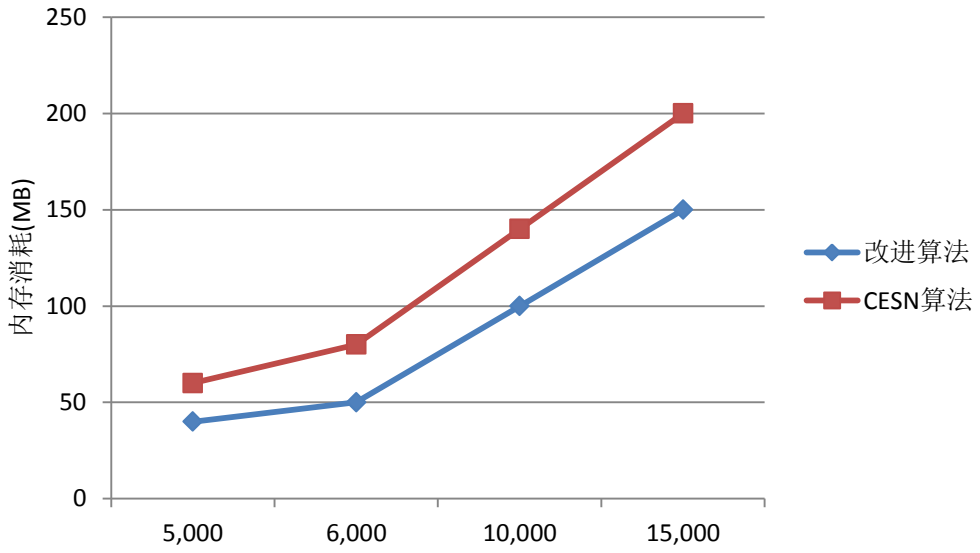

图 4-8 CESN 算法和改进算法的内存消耗

于原检测算法。由此可知，改进后的 token 结构及算法步骤确实有助于算法节省内存开支。

（3）检测时间分析：最后，根据输入同等的复杂事件数据集，通过对比两个算法消耗的计算时间的不同，来比较算法的效率。我们输入四个不同的实验数据集，数据集 1 为检测 and 事件，数据集 2 为检测 or 事件，数据集 3 为检测 seq 事件，数据集 4 为检测非事件。实验结果如图 4-9 所示。

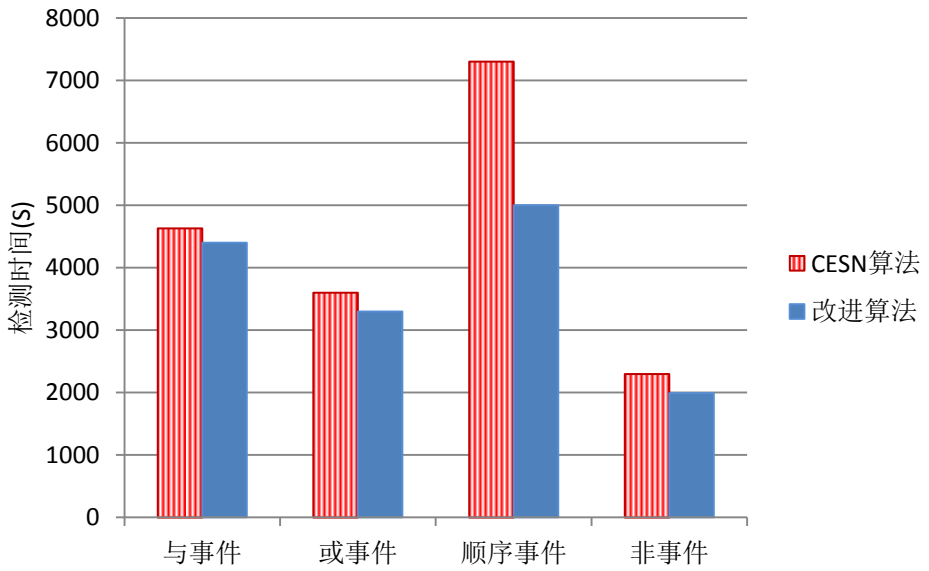

图 4-9 CESN 算法和改进算法的检测时间对比

从图中可看出，在检测“与”事件，“或”事件，“非”事件时，改进的检

测算法在消耗时间上稍低于 CESN 算法，但是在检测 SEQ 事件时，改进算法的消耗的时间比 CESN 算法小的多，效率明显高于后者。从算法对改进策略中我们可分析出原因，改进算法主要是针对 CESN 算法的顺序事件检测时多次回溯做了改进，而针对检测“与”事件，“或”事件，“非”事件，仅仅是检测是库所内的 token 结构变了，节省了运行时的内存开销，因此检测时间上与原算法差不多，但是在检测顺序事件时，由于先根据时间限制过滤掉了很多不必要的子事件，最后才判断 token 数目是否满足条件，因此较 CESN 算法，减少了大量不必要的回溯，提高了算法运行的效率。

4.2.6 验证复杂事件检测优化策略

（1）子事件共享策略。为了验证子事件共享策略，设置两个含可共享部分的复杂事件 E1：A|（B&C） E2：（B&C）|D。仍以输入相同的复杂事件数据集，通过对比两个算法消耗的计算时间的不同，来比较不使用优化策略与使用优化策略两种情况下算法的效率。实验结果如图 4-10 所示。

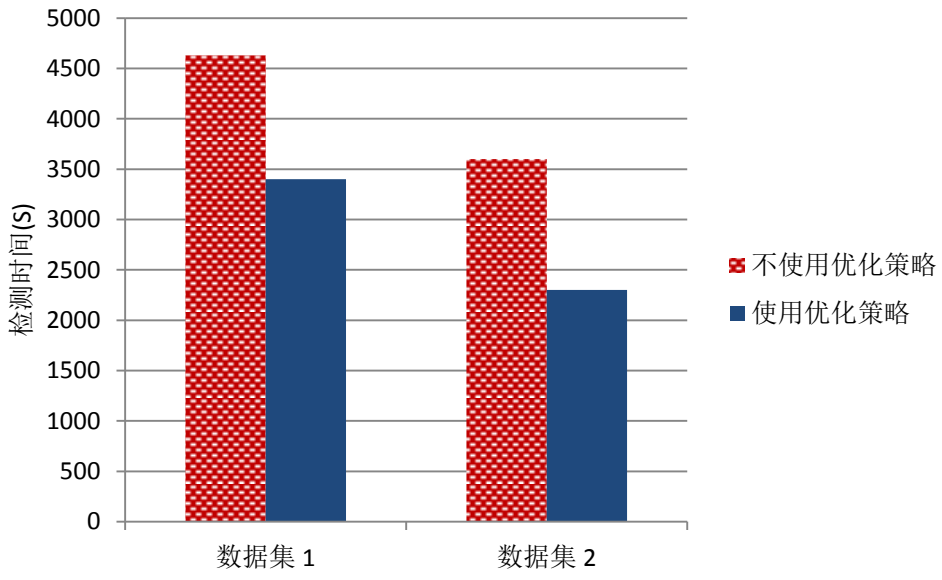

图 4-10 不使用优化策略与使用优化策略情况下算法消耗时间

从图 4-10 中可以看出，数据集 1 和数据集 2 中，使用优化策略之后检测算法检测两事件消耗的时间明显降低，说明优化策略可以起到提高检测效率的作用。

（2）时间窗口共享策略。为了验证时间窗口共享策略的正确性，本文设立两个可以共享事件窗口的复杂时间：

E1=A&B WITHIN（0，600）

E2=A&B WITHIN（0，300）

输入相同的复杂事件数据集，通过对比两个算法消耗的计算时间的不同，来比较不使用优化策略与使用优化策略两种情况下算法的效率。实验结果如图：

从图 4-11 中可以看出，在检测输入为数据集 1 和数据集 2 中，共享时间窗口之后检测算法检测两事件消耗的时间明显降低，说明优化策略可以起到提高检测效率的作用。

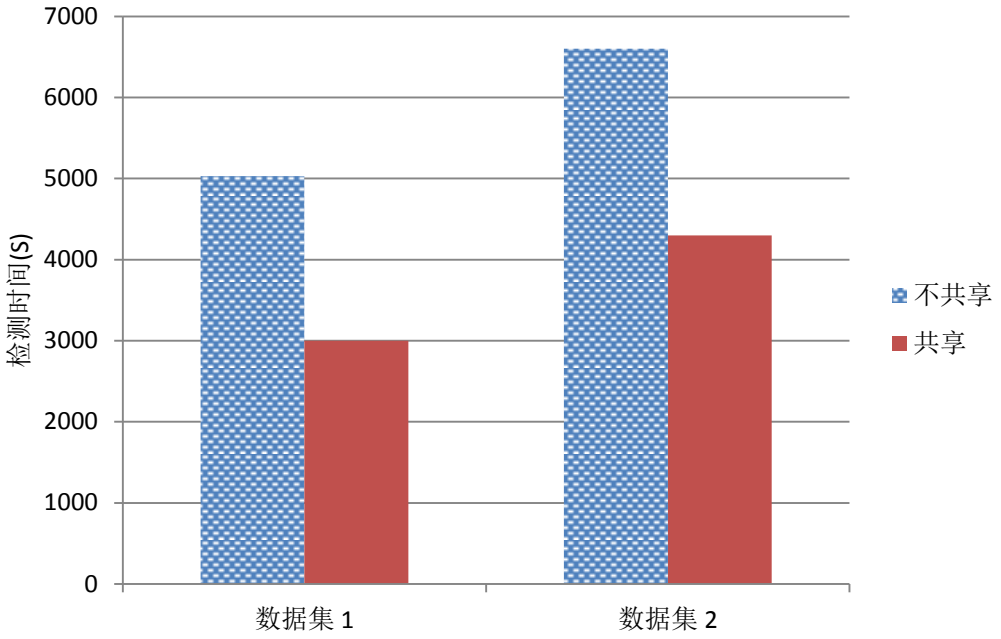

图 4-11 不共享时间窗口与共享时间窗口情况下算法消耗时间

4.3 系统实现

4.3.1 系统设计目标

系统的设计目标是：可以满足特定景下复杂事件检测的需求。即首先确定一个 RFID 系统的应用场景，在该场景下，根据场景的需求，定义复杂事件的表达式，然后根据相应的事件表达式建立该场景下的复杂事件模型，最终利用该模型检测用户关心的复杂事件。为了达到上述目的，系统应满足的功能需求如下：

（1）系统可以根据 RFID 原始事件模型对 RFID 原始数据进行建模，生成 RFID 原始事件。

（2）系统可以根据应用场景的复杂事件表达式定义相应的库所与变迁，根

据逻辑关系进行规则转化，最终生该复杂事件的模型。

（3）系统应该可以根据复杂事件模型对输入的原始事件进行检测。

（4）系统应可以对检测到的复杂事件进行统计和分析。

（5）系统应可以将检测到的复杂事件进行存储与管理。

4.3.2 系统总体设计及功能模块设计

RFID复杂事件处理系统的结构如图4-12所示，它是由原始事件建模模块，事件定义模块、复杂事件模块、规则转化模块、事件检测模型、事件处理模块、存储模块 7 个子功能模块构成，如图所示，系统输入为 RFID 原始数据，经复杂事件处理系统处理，输出为检测到的复杂事件报告。

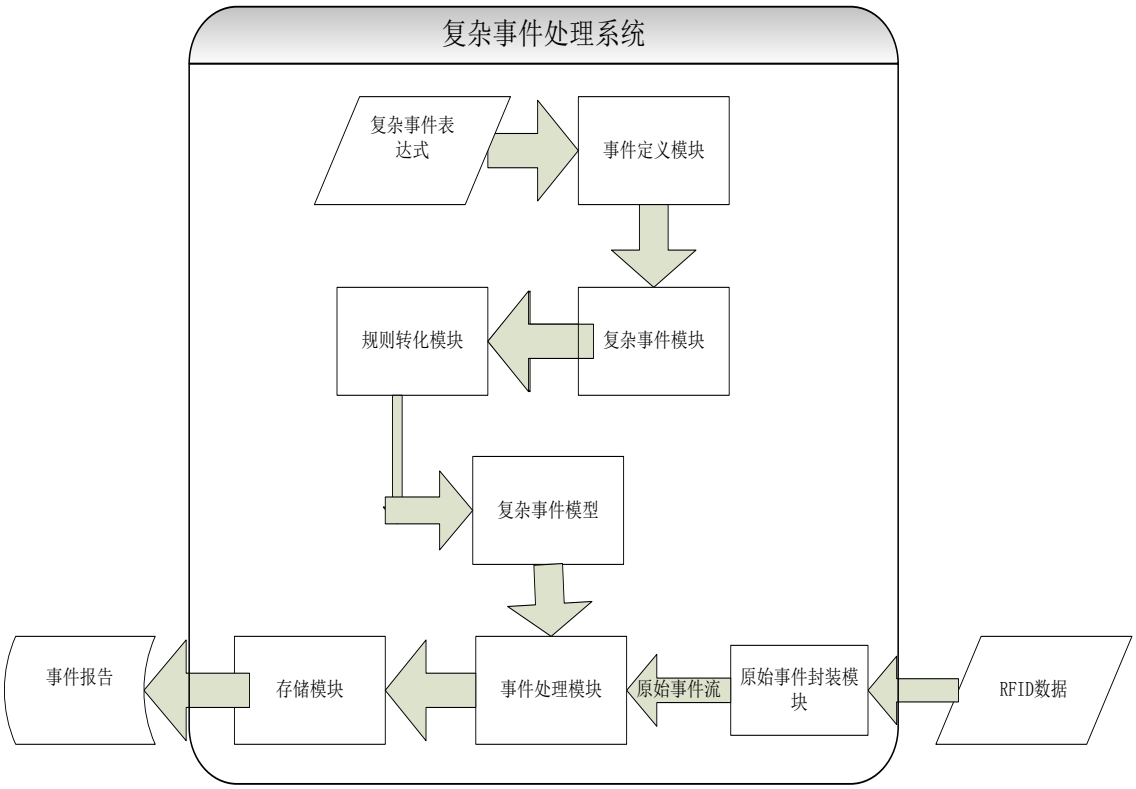

图 4-12 系统结构图

各模块功能介绍如下：

原始事件建模模块：RFID 原始事件建模主要是将系统读入的 RFID 原始数据根据建立的原始事件模型进行封装，给上层的检测模块输入统一的 RFID 原始事件，供上层检测模型使用。

复杂事件表达式是根据系统应用的场景下的业务流程，或者需要检测的事件来定义的。复杂事件表达式的规范在第三章已经介绍。

事件定义模块是根据每个复杂事件表达式中的子事件与操作符建立相应的库所和变迁。不同场景下需检测的复杂事件需再此模块内事先定义。本文通过定义四种不同的变迁，代表四种不同的事件关系，需要根据不同操作符来决定复杂事件的变迁。

复杂事件生成模块为每个事件表达式建立相应的复杂事件。并将事件定义模块内的库所和变迁封装起来。

规则转化模块按照复杂事件表达式的语义将复杂事件内的库所和变迁组合起来，生成一个复杂事件的 Petri 网，即复杂事件模型。

事件处理模块将接收到的原始事件流分发给复杂事件模型，进行事件检测。

存储模块：存储模块是对 RFID 复杂事件检测模型检测到的复杂事件进行存储，以报告的形式供用户查阅。存储生成模块还要负责对复杂事件进行统计。

4.4 本章小结

本章模拟了一个 RFID 场景，构建了该场景下的各种复杂事件，通过对改进检测算法检测该场景下的复杂事件结果分析，验证了 RFID 事件检测算法和检测优化策略的正确性及效率的提高。最后，本文给出了一个超市场景下的 RFID 复杂事件检测的系统实现，通过本事件检测系统对该场景下复杂事件的检测，然后生成事件报告及对事件的统计分析，可实现对该场景的管理和监控。

结　论

复杂事件处理技术是 RFID 事件检测的关键技术，它可以从大量简单事件中快速检测出有需要的复杂事件。用户或者上层其他的商业应用可以根据检测结果实现对某一场景下的物品实现定位，追踪和管理，对可能出现的异常进行判断和预警。本文深入研究了复杂事件检测技术，分析了 RFID 事件检测的基本理论，建立了相应的事件检测模型，并给出了相应复杂事件检测算法。

本文的主要研究成果如下：

（1）深入分析了 RFID 数据的特点、RFID 事件的分类，在此基础上建立了一个基于时间点的 RFID 事件模型。并定义了该事件模型下的事件之间的关系，事件操作符及时间限制条件。用户可以根据它们建立满足不同场景需要的复杂事件。

（2)通过借助 Petri 网模型,给出了一个基于 Petri 网的复杂事件检测模型。之后在已有的 CESN 检测算法的基础上给出了一个改进的复杂事件检测算法 PCED，实验证明改进算法的可有效节省复杂事件检测过程中的内存消耗，并且在检测顺序事件时可以有效提高检测效率。

（3）分析了该检测算法在检测复杂事件流程中可以优化的部分。针对复杂事件检测中出现的两种典型情况，分别给出了子事件共享策略和时间窗口共享策略两种对应的优化策略，最后通过对比实验验证了优化策略可以使检测算法在效率上的得到提高。

由于时间有限和客观条件限制，本文还存在以下几点不足之处，有待于进一步研究与探索：

（1)本文里提出了四种操作符虽然可满足复杂事件表达式去表达复杂事件的要求，但是实际上针对与某些场景下的应用，可能存在一些频繁出现的事件关系，对于它们，可以增加事件操作符，以简化事件表达式，同时也可提高事件检测效率。

（2）本文虽然对复杂事件检测算法进行了优化，给出了一些优化策略，但是对于与或非三种操作符优化的结果并不明显，还需要进一步研究。

（3)本文给出的复杂事件检测系统虽然可以针对不同的应用场景进行事件检测，但是需事先手动定义事件表达式及模型，若需要完全自由的切换与不同场景下，还需要定义一种复杂事件描述语义，使用户可以个根据语法规则去定义该场景下的事件表达式，系统根据语法规则去解析该表达式，自动地将其翻

译为检测模型。对于复杂事件描述语言，还可以进一步深入研究。

参考文献

[1] 中国物品编码中心. 条码技术与应用[M]. 北京：清华大学出版社，2003：17-21.

[2] 庞明. 物联网条码技术与射频识别技术[M]. 北京：中国物资出版社，2011：153-154.

[3] Nath B, Reynolds F, Want R. RFID Technology and Applications [J]. IEEE Pervasive Computing, 2006(5): 22 - 24.

[4] Want R. An Introduction to RFID Technology[J]. IEEE Pervasive Computing, 2006(5): 24 - 33.

[5] Mamei M, Zambonelli F. Pervasive Pheromone-based Interaction with RFID Tags[J]. ACM Transactions on Autonomous and Adaptive Systems, 2007,2(2):1556~1560.

[6] Weinstein R. RFID: A Technical Overview and Its Application to the Enterprise[J]. IT Professional, 2005(7):27~33.

[7] Sydanheimol L, Nummela J, Ukkonen L. Characterization of Passive UHF RFID tag Performance[J]. Antennas and Propagation Magazine, IEEE, 2008, 50(3): 207~12.

[8] Lisha Y. Application of RFID Technology[J]. Communications Technology, 2007(12):267~269.

[9] Collins J. DOD Tries Tags That Phone Home[J/OL]. RFID Journal, 2006. http://www.rfidjournal.com/article/pdf/1458.

[10] Collins J. Boeing Outlines Tagging Timetable[J/OL]. RFID Journal, 2006. http://www.rfidjournal.com/article/pdf/985.

[11] Swedberg C. Hospital Uses RFID for Surgical Patients[J/OL]. RFID Journal, 2005. http://www.rfidjournal.com/article/pdf/1714.

[12] Park J H, Lee B H. RFID Application System for Postal Logistics [J]., Portland International Center for Management of Engineering and Technology, 2007(8): 2345 - 2352.

[13] Yu M. Survey of Current Research State and Application of RFID[J]. Chinese Journal of Scientific Instrument, 2008:728~731.

[14] Palmer M. Seven Principles of Effective RFID Data Management[J/OL]. Progress Software-Real Time Division, 2004. http://cresent.progress.com/realtime/docs/articles/.

[15] Wu W, Cheng K, Zhang H. Semantics-Based Complex Event Processing for RFID Data Streams. Proc. Of ISDPE, 2007:32~34.

[16] Wang F, Liu P. Complex RFID Event Processing[C]. The VLDB Journal, 2009:914~931.

[17] Tzeng S F, Chen W H. Evaluating the Business Value of RFID: Evidence from Five Case Studies[J]. International Journal of Production Economics, 2008(112) 601-613.

[18] Luckham D. A Short History of Complex Event Processing[J/OL]. 2008, http://complexevents.com/wp-content/uploads/2008/02.

[19] Marcos K, Strom A R E, Stunnan D C, et al. Matching Events in a Content-based Subscription System[J]. Proceedings of the eighteenth annual ACM symposium on Principles of distributed computing, 1999:53-61.

[20] Arasu A, Babu S. CQL: A Language for Continuous Queries Over Streams and Relations[J]. Database Programming Language, 2003: 123-124.

[21] Dayal U, Blaustein B, Buchmann A. The HiPAC Project: Combining Active Databases and Timing Constraints[J]. SIGMOD, 1998(17):51-70.

[22] Bruno N, Koudas N, Srivastava D. Holistic Twig Joins: Optimal XML Pattern Matching[C]. Proceedings of the 2002 ACM SIGMOD international conference on Management of data, 2002(6):312-321.

[23] Hinze A. Efficient Filtering of Composite Events[J]. New Horizons in Information Management, 2003: 207-225.

[24] Gu Y, Yu G, Zhang T. RFID Complex Event Processing Teniques. Journal of Computer Science and Frontiers, 2007, 1(3):255-267.

[25] Rizvi S, Jeffrey S, Krishnamurthy S, et al. Events on The Edge[J]. Proc. of SIGMOD, 2005:885-887.

[26] Wang F, Liu P. Temporal Management of RFID Data[C]. 31st VLDB Conference, 2005:1128~1139.

[27] Agrawal J, Diao Y, Gyllstrom D, et al. Efficient Pattern Matching over Event Streams[C]. Proceedings of the 2008 ACM SIGMOD international conference on Management of data, 2008:147-159.

[28] White W, Riedewald M, Gehrke J, et al. What's "Next" in Event Processing?[C] Proc. of PODS, 2007.

[29] Yang H, Chen R, Liu Y. Research on RFID Composite Event Detection Methods[C]. ICSLP, 2010:141~143.

[30] Zhu Q, Wang H, Gao H. Real-Time Multiple Complex Event Queries Processing over RFID Streams[J]. Journal of Frontiers of Computer Science and Technology, 2011:257~259.

[31] Wang F, Liu S, Liu P, et al. Bridging Physical and Virtual Worlds: Complex Event Processing for RFID Data Streams[C]. EDBT, 2006:588~607.

[32] Zhang J F, Wei J. Composite Event Detection in Distributed Computing: Survey and Evaluation[J]. Application Research of Computers, 2005:1-5.

[33] Gehani N H, Jagadish H V, Shmueli O. Composite Event Specification in Active Database: Model and Implementation[C]. VLDB, 1992:327-338.

[34] Wu E, Diao Y and Rizvi S. High-Performance Complex Event Processing Over Streams[C]. ACM SIGMOD, 2006: 407–418.

[35] Gyllstrom D, Wu E, Chae H J, et al. SASE: Complex Event Processing over Streams[C]. 3rd Biennial Conference on Innovative Data Systems Research (CIDR) January, 2007:710-721.

[36] Demers A, Gehrke J, Panda B, et al. Cayuga: A General Purpose Event Monitoring System[C]. ACM SIGMOD international conference on Management of data, 2007(6):1100-1102.

[37] Gatziu S, Geppert A. The SAMOS Active DBMS Prototype[C]. Proceedings of the 1995 ACM international conference on Management of data, SIGMOD, 1995:480-491.

[38] Hu W, Ye W, Huang Y, et al. Complex Event Processing in RFID Middleware: A Three Layer Perspective[C]. ICCIT, 2008:1121-1125.

[39] Ye W, Zhao W, Huang Y, et al. Towards Passive RFID Event[C]. ICSAC, 2009:492-499.

[40] Jin X, Lee X, Kong N, et al. Efficient Complex Event Processing over RFID Data Stream[C]. Seventh IEEE/ACIS International Conference on Computer and Information Science, 2008(5):75-81.

[41] Samani M, Sloman M. GEM: A Generalised Event Monitoring Language for Distributed Systems[J]. IEE/IOP/BSC Distributed Engineering Journal, , 1997 (2):971-982.

[42] Gruber R E, Krishnamurthy B, Panagos E. The Achitecture of the READY Event Notification Service[C]. Proceedings of the 19th IEEE International Conference on Distributed Computing Systems Middleware Workshop, 1999.

[43] Chakravarthy S, Anwar E, Maugis L, et al. Design of Sentinel: an object-oriented DMBS with event-based rules[J]. Information and Software Technology.1994(36):555-568.

[44] Geppert A, Tombros D. Event-based Distributed Workflow Execution with EVE[C]. Middleware '98 Proceedings of the IFIP International Conference on Distributed Systems Platforms and Open Distributed Processing, 1998:427-442.

[45] Sudarshan S C, Venkat K, Sridhar R, et al. Managing RFID Data[C]. VLDB '04 Proceedings of the Thirtieth international conference on Very large data

bases, 2004:1189-1195.

[46] EPCglobal Inc™. EPC™ Generation 1 Tag Data Standards Version 1.1 Rev.1.27[S]. Technical Report, 2005.

[47] Jin X, Lee X, Kong N, et al. Efficient Complex Event Processing over RFID Data Stream[C]. Seventh IEEE/ACIS International Conference on Computer and Information Science, 2008(5):75-81.

[48] 袁崇义. Petri 网原理与应用[M]. 北京：电子工业出版社, 2005:12-15.

[49] Carl A P. Kommunikation Mit Automaten[D]. Darmstadt : Technische Hochschule Darmstadt, 1962.

[50] Wang Q, Han J. A Survey of Theories and Application of Synthesis Techniques for Petri Nets[J]. Computer simulation, 2008(12)8-11.

[51] 吴哲辉. Petri 网导论[M]. 北京：机械工业出版社, 2006:10-16.

哈尔滨工业大学学位论文原创性声明及使用授权说明

学位论文原创性声明

本人郑重声明：此处所提交的学位论文《基于 Petri 网的 RFID 复杂事件检测方法及优化策略研究》，是本人在导师指导下，在哈尔滨工业大学攻读学位期间独立进行研究工作所取得的成果。据本人所知，论文中除已注明部分外不包含他人已发表或撰写过的研究成果。对本文的研究工作做出重要贡献的个人和集体，均已在文中以明确方式注明。本声明的法律结果将完全由本人承担。

作者签名：[illegible]　日期：2012 年 1 月 4 日

学位论文使用授权说明

本人完全了解哈尔滨工业大学关于保存、使用学位论文的规定，即：

（1）已获学位的研究生必须按学校规定提交学位论文；（2）学校可以采用影印、缩印或其他复制手段保存研究生上交的学位论文；（3）为教学和科研目的，学校可以将学位论文作为资料在图书馆及校园网上提供目录检索与阅览服务；（4）根据相关要求，向国家图书馆报送学位论文。

保密论文在解密后遵守此规定。

本人保证遵守上述规定。

作者签名：[illegible]　日期：2012 年 1 月 4 日

导师签名：[illegible]　日期：2012 年 1 月 5 日

致　谢

值此论文完成之际，谨向给予我关心、鼓励和帮助的良师益友表示由衷的感谢。

首先感谢我的导师张春慨副教授，在研究生生涯的两年半里，无论是学习上还是生活上均给予了诸多关怀和帮助，在课题研究过程中，他带领我一起学习和实践，并给予了很多的指导，提出了许多建设性的建议。在他的亲切地指导了启发下，我解决了许多论文中的难题，开阔了视野，汲取了丰富的知识。他渊博的学识、耐心而细致的作风、对工作的满腔热情使我终生难忘。

感谢黎聪，赵霖，严学凯，付学文四位师兄，虽然跟你们一起交流的时间仅仅一年，但正是在这一年里了解了自己的研究方向，为自己的论文打好了基础。感谢陈岩，张蕾，刘亚男，李原，整个研究生期间我们都经常讨论问题，包括学习上和生活上的，从你们那里，我学习到了很多,感谢你们!

感谢 09 级计算机系和网络智能计算实验室的所有同学，感谢丁丁等“CS登山趴友”群友，与你们一起度过了研究生生活，一起学习，一起生活的日子里，你们为我带来了太多的快乐和美好的回忆。感谢小虎、凯瑞、维海、大飞等几位我寝室的兄弟，时常回忆我们住一起的日子，是你们伴我走过宝贵的研究生的生涯，咱们一起吃饭，自习，爬山，卧谈，那是我最美好的回忆。

感谢所有远方的朋友。工作的、读研的，虽然你们不在身边，但是我相信我们的友情不会因为距离的远近而改变!

感谢计算机学科部的领导、老师和教学秘书，你们在我日常学习和生活中给予的无私、默默的帮助才为我们每个学生增长了才干，为将来的的生活和工作打下了坚持的基础，感谢你们为所有学生的付出!

感谢女朋友于佳，从始至终你都一直默默的关心我，支持我，不管是找工作中艰难的等待，还是写论文时通宵的实验，你的关心和支持始终在身边。感谢生命中有你!

最后，我要感谢我的家人，感谢我的妹妹，这么多年来你始终那么懂事，那么活泼，你给全家带来了太多温馨！感谢我的父母，是你们的辛勤养育才有今天的我，我的每一次进步和成长都和你们的关心和鼓励分不开，你们用自己辛勤和努力将我培育成人，自己却渐渐年迈老去，谁言寸草心，报得三春晖。让你们过上幸福的日子是我最大的心愿!