技术交底书格式

专利名称	一种基于聚类和关联分析的电信网络告警预测方法	所属技术领域	通信技术
专利发明人或设计人	林鹏，叶可江，须成忠	技术交底书撰写人	林鹏

技术问题联系人	姓名	电话	E-mail
	林鹏	13760148249	173213354@qq.com

1、本发明要解决的技术问题是什么？

随着电信技术的飞快发展，电信网络规模也越来越大、结构越来越复杂，每天产生的设备故障告警也越来越多。为了保证电信网络的良好运行，给用户带来良好的使用体验，需要在告警发生的时候迅速定位并排除故障。由于在电信网络中，软硬件组件间存在复杂的拓扑关系，使得告警之间有着很大的相关性。通过研究告警之间的相关性，可以有效地解决告警泛滥问题，更好地帮助网络管理人员找到故障的根源。更进一步地，通过告警相关性分析，人们可以预测那些还未发生的故障，并提前为此做好准备，做到“未雨绸缪”。但随着电信网络规模越来越大，传统的依赖专家知识建立的规则库已不能满足现实的需求，需要建立起更智能、更全面的相关性分析方法。本发明针对电信网络提供了一种基于聚类和关联分析的告警预测方法。

2、详细介绍技术背景，并描述已有的与本发明最相近似的实现方案。

随着电信网络规模的不断扩大，每天由电信设备故障产生的告警也越来越多，而且一个设备的故障经常会导致另一个设备的故障，从而给维护人员带来很大的麻烦。70年代有人提出利用专家知识建立告警关联规则库来处理电信网络产生的告警，但时至今日，这种依赖专家知识的告警关联工具显然不能很好地应付越来越复杂的网络结构。

由于近年来数据挖掘技术的发展，很多行业都开始使用这项技术来处理各自的业务，并取得了令人瞩目的效果，通信行业也不例外。在告警关联方面，运营商开始尝试使用数据挖掘技术对以往积累的大量告警历史信息进行分析，但复杂的业务逻辑和巨大的数据量给他们带来了很大的挑战。

在现有的挖掘技术中，WinEPI算法常被用于告警规则的关联。WinEPI是一种有效的序列模式挖掘算法，其基本思想是首先找到短的频繁情景，然后逐步递推找到大的频繁情景。在WinEPI算法的实现中，需要指定滑动时间窗口宽度和频繁度阈值。

3、现有技术的缺点是什么？针对这些缺点，说明本发明的目的。

目前电信网络中的告警关联规则一般是通过专家基于积累的相关经验进行总结提炼，然后讨论决定的。这种人工提取规则的方法，存在效率低、不完整、依赖性强等特点，无法很好适应当前复杂的电信网络结构。

已经有一些使用数据挖掘技术进行告警关联的案例，比较经典的算法是WinEPI。WinEPI通过设置固定的时间窗口宽度来提取告警序列，并发现告警在时间上的偏序关系。但由于告警序列通常是一组不均匀的数据，往往在一个短的时间段内密集产生，之后一段时间就恢复平静。且不同时间段内产生的告警事件频率、持续时间也各不相同，如果使用固定的时间窗口宽度来提取告警事务则可能存在很多无效的数据，导致最终提取的关联规则无效。本发明提出了一种基于时间密度聚类的告警事务提取方案，通过这种方案能实现动态的时间窗口宽度，从而能很好地处理具有突发性特点的告警数据；对于提取出的告警事务数据库，采用Apriori算法进行关联规则挖掘；最后通过规则匹配和多维属性概率相乘的方法预测告警。

4、本发明技术方案的基本内容。

针对上述，本发明提出了一种基于聚类和关联分析的电信网络告警预测方法，目的在于提高关联规则的有效性和对未发生的告警进行预测。

一种基于聚类和关联分析的电信网络告警预测方法，包括以下步骤：

（1） 按时间顺序分批读入告警数据并预处理。

（2） 对告警数据按照Unix时间戳进行DBSCAN聚类。

（3） 挖掘频繁项集和告警间的关联规则。

（4） 合并并保存所有批次告警数据挖掘出的规则。

（5） 利用告警关联规则预测未发生的告警。

5、本发明技术方案的详细阐述。

以下将结合附图对本发明的具体流程加以说明。

附图一描述了按时间顺序分批读入告警数据并预处理的流程。

第一步，从数据库中读取若干条告警记录，对每一条数据提取相应特征信息，组成以下数据格式：{网元、故障原因、故障类型、故障发生时间}。

第二步，检查该条告警故障原因是否存在故障表中，如是进入下一步；如否说明该告警可能是记录出错的脏数据，丢弃并回到步骤一。

第三步，检查数据库中该告警的告警类型字段是否属于“告警清除”类型，如是，丢弃并回到步骤一；如否，进入下一步。

第四步，检查内存中是否存在网元、故障原因、故障类型均相同，且故障发生时间绝对值之差小于给定阈值的告警记录，如是，说明该告警是重复告警，丢弃并回到步骤一；如否，将该告警添加进内存。

第五步，重复步骤一到四，直至该批告警数据全部被处理完。

附图二描述了对告警数据按照Unix时间戳进行DBSCAN聚类的流程。

第一步，将告警数据格式中的发生时间字符串转换成Unix时间戳，并将该时间列数据单独提取出来。

第二步，使用DBSCAN算法处理时间列数据，算法的参数eps需要调节，min_samples设为2，评价标准使用曼哈顿距离（Manhattan Distance）：$distance(a_i - a_j) = |t_i - t_j|$ ，其中t_i、t_j分别代表告警a_i，a_j发生的时间。

第三步，使用轮廓系数（Silhouette Coefficient）来评价聚类效果的好坏，对于样本i，轮廓系数的定义如下：$s(i) = \frac{b(i)-a(i)}{max\{a(i),b(i)\}}$ ，$a(i)$为样本i到同簇其他样本的平均距离，$b(i)$为样本i到其他簇所有样本的平均距离。$s(i)$越接近1，说明聚类越合理。

第四步，将聚类得出的簇标签结果添加到告警数据的最后一列：{网元、故障原因、故障类型、故障发生时间，簇类别}，根据所在类别划分告警事务，同一个类别的告警属于同一事务。

第五步，对事务中的告警按照时间顺序排序，并且去掉重复的告警（网元、故障原因、故障类型都相同的告警考虑为重复告警）。

附图三描述了挖掘频繁项集和告警间关联规则的流程。

第一步，设定最小支持度阈值min_sup，扫描所有聚类后得到的告警事务，对每个项进行计数，剔除出现小于min_sup的项集，得到频繁1-项集，记为L1。

第二步，对L1进行迭代，生成候选2-项集。根据Apriori的前提假设：如果一个项集是非频繁集，那么它的所有超集也是非频繁的，因此需要对候选2-项集进行剪枝处理，即剪掉候选2-项集中包含的不频繁1-项集，得到C2。

第三步，对C2进行支持度计数，将小于min_sup的项集剔除，得到L2。

第四步，重复步骤二和步骤三，生成L3，L4…Lk，直至Lk+1为空。

第五步，设定最小置信度阈值min_conf，输出提升度大于1的强关联规则。提升度的计算方式如下：$lift(X \to Y) = \frac{P(X,Y)}{P(X)\cdot P(Y)} = \frac{P(Y|X)}{P(Y)}$

第六步，用步骤一到步骤五的方法，分别单独挖掘告警中网元、故障原因的关联规则。

附图四描述了合并并保存所有批次告警数据挖掘出的规则的流程。

第一步，将第一批告警数据挖掘出的规则读进内存，格式为：{前件，后件，conf，samples}。

第二步，读进下一批告警数据挖掘出的规则，逐条对比规则是否存在内存当中，若否则添加；若已存在，更新内存中该规则的conf及samples：$conf = \frac{conf1*samples1+conf2*samples2}{samples1+samples}$，$samples = samples1 + samples2$

第三步，重复步骤二直至所有规则合并完成，将规则保存为可读写文件。

附图五描述了利用告警关联规则预测未发生的告警的流程。

第一步，读取存放规则的文件，将其转换为键值对，键为前件，值为后件及置信度：{key：前件，value：[后件，conf]}。

第二步，按照时间顺序读入当前发生的告警，判断该告警与上一条读进的告警发生时间之差是否小于规定的阈值min_time，如是，则将其添加到告警集合中去，进入步骤三；如否，执行步骤四。

第三步，遍历上述键值对规则表，检查规则表中是否存在一行数据属于当前告警集合的子集，若是，输出该规则的后件和概率并清空告警集合；若不是，继续执行步骤二，扩大告警集合。

第四步，查找单独的属性关联规则表（即网元→网元，故障→故障），判断是否存在当前网元和故障各自对应的后件，若有，记其置信度为p1和p2，将网元和故障所对应的后件组合成预测告警，并输出概率p=p1*p2；若无，即预测无故障发生。

第五步，清空告警集合，重复步骤二到步骤四，继续预测新的告警。

6、本发明的关键点和欲保护点是什么？

1、一种基于聚类和关联分析的电信网络告警预测方法，其特征在于：

S1: 按时间顺序分批读入告警数据并预处理。

S2: 对告警数据按照Unix时间戳进行DBSCAN聚类。

S3: 挖掘频繁项集和告警间的关联规则。

S4: 合并并保存所有批次告警数据挖掘出的规则。

S5: 利用告警关联规则预测未发生的告警。

2、根据权利1要求所述的基于聚类和关联分析的电信网络告警预测方法，其特征在于，S1包括：

S101：从数据库中读取若干条告警记录，将其重组成以下数据格式：{网元、故障原因、故障类型、故障发生时间}。

S102：检查告警的故障原因是否存在故障表中，如不存在则将该告警丢弃。

S103：过滤掉告警记录中属于“告警清除”的告警。

S104：过滤掉规定时间阈值内重复发生的告警。

S105：将告警写进内存，并处理下一条数据。

3、根据权利1要求所述的基于聚类和关联分析的电信网络告警预测方法，其特征在于，S2包括：

S201：将告警数据格式中的发生时间字符串转换成Unix时间戳，并将该时间列数据单独提取出来。

S202：使用DBSCAN算法对时间列数据进行聚类。

S203：将聚类得出的簇标签结果添加到告警数据的最后一列，根据所在类别划分告警事务，同一个类别的告警属于同一事务。

S204：对事务中的告警按照时间顺序排序，并过滤掉同一条事务中重复的告警。

4、根据权利1要求所述的基于聚类和关联分析的电信网络告警预测方法，其特征在于，S3包括：

S301：使用Apriori算法挖掘告警之间的关联规则。

S302：输出提升度大于1，且置信度大于给定阈值的规则。

S303：单独挖掘告警数据网元之间、故障愿意之间的关联规则。

5、根据权利要求4所述的方法，其特征在于，所述使用Apriori算法挖掘告警之间的关联规则具体为：

第一步，设定最小支持度阈值min_sup，扫描所有聚类后得到的告警事务，对每个项进行计数，剔除出现小于min_sup的项集，得到频繁1-项集，记为L1。

第二步，对L1进行迭代，生成候选2-项集。根据Apriori的前提假设：如果一个项集是非频繁集，那么它的所有超集也是非频繁的，因此需要对候选2-项集进行剪枝处理，即剪掉候选2-项集中包含的不频繁1-项集，得到C2。

第三步，对C2进行支持度计数，将小于min_sup的项集剔除，得到L2。

第四步，重复步骤二和步骤三，生成L3，L4…Lk，直至Lk+1为空。

6、根据权利1要求所述的基于聚类和关联分析的电信网络告警预测方法，其特征在于，S4包括：

S401：将第一批告警数据挖掘出的规则按以下格式读进内存：{前件，后件，conf，samples}。

S402：读进下一批告警数据挖掘出的规则，逐条对比规则是否存在内存当中，若否则添加；若已存在，更新内存中该规则的conf及samples：$conf = \frac{conf1*samples1+conf2*samples2}{samples1+samples}$，$samples = samples1 + samples2$。

S403：待所有规则合并完成，将规则保存为可读写文件。

7、根据权利1要求所述的基于聚类和关联分析的电信网络告警预测方法，其特征在于，S5包括：

S501：读取存放规则的文件，将其转换为键值对数据，键为前件，值为后件及置信度：{key：前件，value：[后件，conf]}。

S502：按照时间顺序读入当前发生的告警，判断该告警与上一条读进的告警发生时间之差是否小于规定的阈值min_time，如是，则将其添加到告警集合中去，执行S503；如否，执行S504。

S503：遍历上述键值对规则表，检查规则表中是否存在一行数据属于当前告警集合的子集，若是，输出该规则的后件和概率并清空告警集合；若不是，继续执行S502，扩大告警集合。

S504：查找单独的属性关联规则表（即网元→网元，故障→故障），判断是否存在当前网元和故障各自对应的后件，若有，记其置信度分别为p1和p2，然后将网元和故障所对应的后件组合成预测告警，并输出概率p=p1*p2；若无，即预测无故障发生。

S505：清空告警集合，重复S502，继续预测新的告警。

7、与第2条所属的最好的现有技术相比，本发明有何优点？

现有技术通过设置固定的时间窗口宽度来提取告警序列，并发现告警在时间上的偏序关系。但由于告警序列通常是一组不均匀的数据，且不同时间段内产生的告警事件频率、持续时间也各不相同，如果使用固定的时间窗口宽度来提取告警事务则可能存在很多无效的数据，导致最终提取的关联规则无效。

本发明提出了一种基于密度聚类的告警事务提取方案，通过这种方案能实现动态的时间窗口宽度，从而能很好地处理具有突发性特点的告警数据；对于提取出的告警事务数据库，采用Apriori算法进行关联规则挖掘；最后通过规则匹配和多维属性概率相乘的方法预测告警。

本发明使用时间聚类提高了关联规则的有效性，采用分批处理的思路提高了处理大批量数据的能力，并给出了一种预测告警的方案。

8、本发明是否经过实验、模拟、使用而证明可行，结果如何？

结果经试验模拟使用证明确实可行

9、本发明的变更设计（替代方案）及其它用途：

无

10、附图及说明

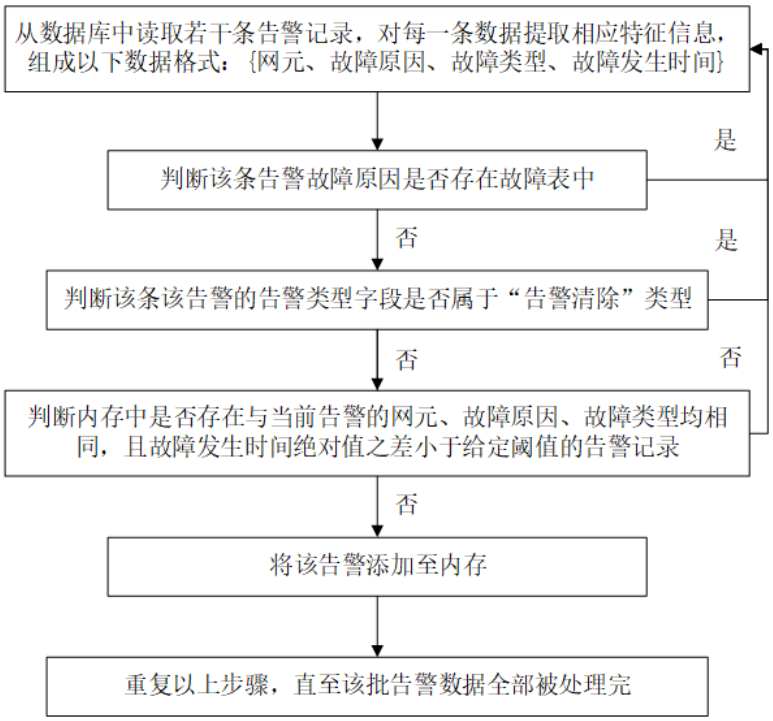
附图一描述了按时间顺序分批读入告警数据并预处理的流程。

附图二描述了对告警数据按照Unix时间戳进行DBSCAN聚类的流程。

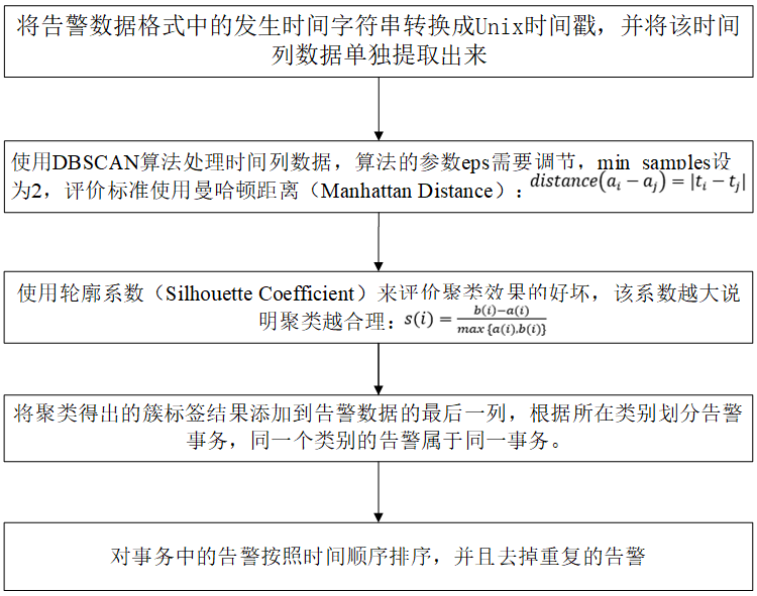
附图三描述了挖掘频繁项集和告警间关联规则的流程。

附图四描述了合并并保存所有批次告警数据挖掘出的规则的流程。

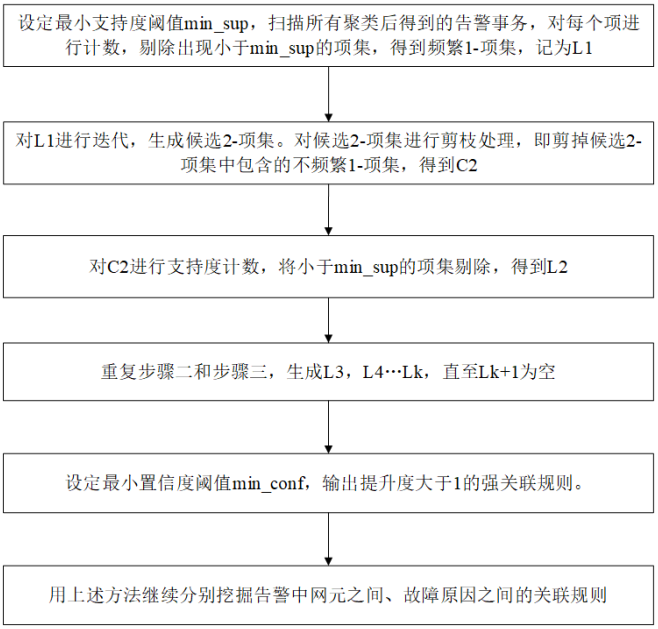
附图五描述了利用告警关联规则预测未发生的告警的流程。


从数据库中读取若干条告警记录，对每一条数据提取相应特征信息，组成以下数据格式：{网元、故障原因、故障类型、故障发生时间}
判断该条告警故障原因是否存在故障表中
是
否
判断该条该告警的告警类型字段是否属于“告警清除”类型
是
否
判断内存中是否存在与当前告警的网元、故障原因、故障类型均相同，且故障发生时间绝对值之差小于给定阈值的告警记录
否
将该告警添加至内存
重复以上步骤，直至该批告警数据全部被处理完

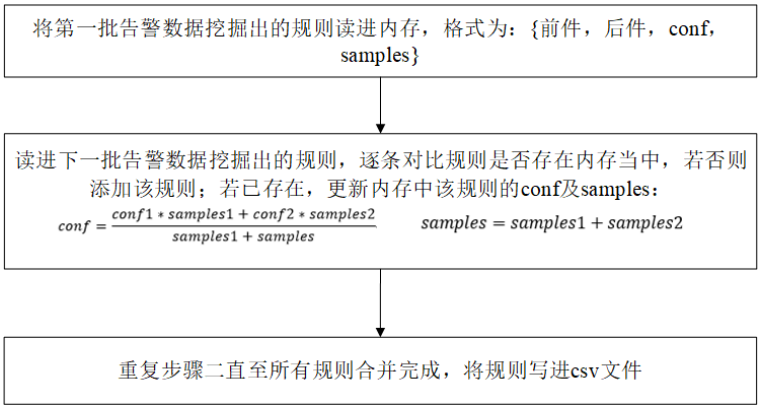
图一


将告警数据格式中的发生时间字符串转换成Unix时间戳，并将该时间列数据单独提取出来
使用DBSCAN算法处理时间列数据，算法的参数eps需要调节，min samples设为2，评价标准使用曼哈顿距离（Manhattan Distance）：$distance(a_i - a_j) = |t_i - t_j|$
使用轮廓系数（Silhouette Coefficient）来评价聚类效果的好坏，该系数越大说明聚类越合理：$s(i) = \frac{b(i)-a(i)}{max\{a(i),b(i)\}}$
将聚类得出的簇标签结果添加到告警数据的最后一列，根据所在类别划分告警事务，同一个类别的告警属于同一事务。
对事务中的告警按照时间顺序排序，并且去掉重复的告警

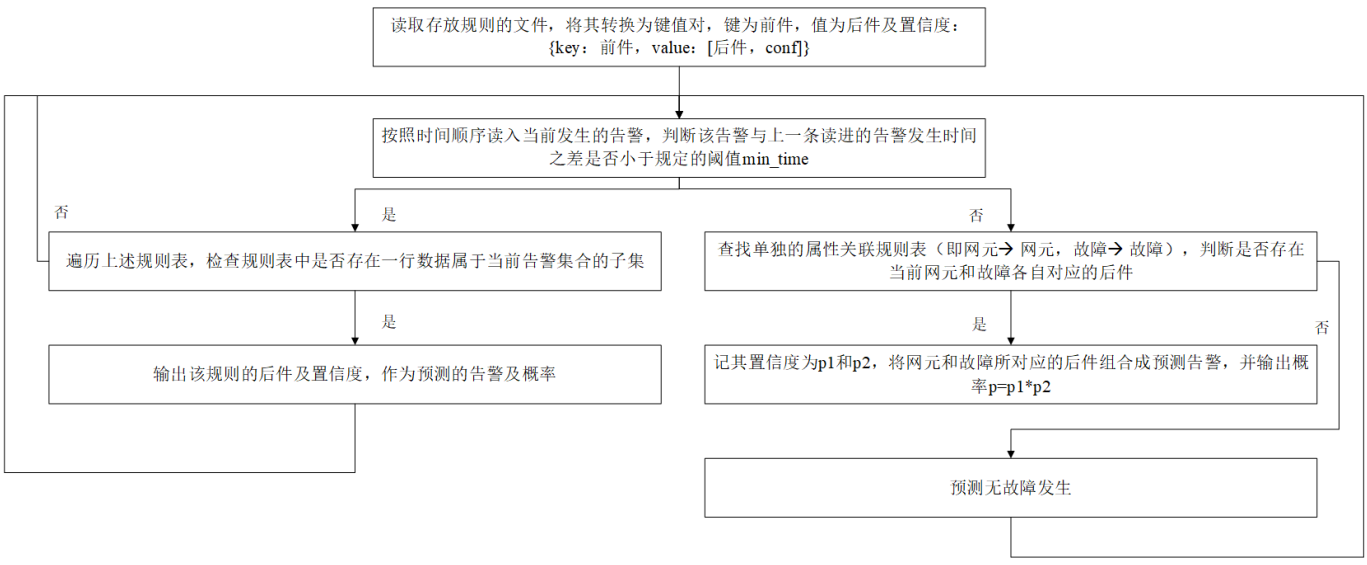
图二


设定最小支持度阈值min_sup，扫描所有聚类后得到的告警事务，对每个项进行计数，剔除出现小于min_sup的项集，得到频繁1-项集，记为L1
对L1进行迭代，生成候选2-项集。对候选2-项集进行剪枝处理，即剪掉候选2-项集中包含的不频繁1-项集，得到C2
对C2进行支持度计数，将小于min_sup的项集剔除，得到L2
重复步骤二和步骤三，生成L3，L4…Lk，直至Lk+1为空
设定最小置信度阈值min_conf，输出提升度大于1的强关联规则。
用上述方法继续分别挖掘告警中网元之间、故障原因之间的关联规则

图三


将第一批告警数据挖掘出的规则读进内存，格式为：{前件，后件，conf，samples}
读进下一批告警数据挖掘出的规则，逐条对比规则是否存在内存当中，若否则添加该规则；若已存在，更新内存中该规则的conf及samples：
$conf = \frac{conf1 * samples1 + conf2 * samples2}{samples1 + samples}$ $samples = samples1 + samples2$
重复步骤二直至所有规则合并完成，将规则写进csv文件

图四


读取存放规则的文件，将其转换为键值对，键为前件，值为后件及置信度：{key：前件，value：[后件，conf]}
按照时间顺序读入当前发生的告警，判断该告警与上一条读进的告警发生时间之差是否小于规定的阈值min_time
否
是
否
遍历上述规则表，检查规则表中是否存在一行数据属于当前告警集合的子集
查找单独的属性关联规则表（即网元→ 网元，故障→ 故障），判断是否存在当前网元和故障各自对应的后件
是
否
是
输出该规则的后件及置信度，作为预测的告警及概率
记其置信度为p1和p2，将网元和故障所对应的后件组合成预测告警，并输出概率p=p1*p2
预测无故障发生

图五