基于模式匹配的大规模数据分析软件设计与实现①

尹志喜 甄国涌 (中北大学 电子测试技术国家重点实验室 山西 太原 030051)

摘 要: 在分析模式匹配算法的基础上，提出了一种改进的模式匹配算法，并将该算法应用于大规模数据分析软件设计之中。在数据分析的初始阶段，通过该模式匹配算法建立一张数据索引表，随后分析软件借助于索引表和帧结构分布表，对原始数据进行分析和处理。该算法的应用，有效的解决了大规模数据处理过程中的难题，提高了大规模数据处理软件的效率。

关键词: 模式匹配算法；大规模数据；索引；帧结构；数据分析

Design and Implementation of Large-Scale Data Analysis Software Based on Pattern Matching Algorithm

YIN Zhi-Xi, ZHEN Guo-Yong

(National Key Laboratory For Electronic Measurement Technology, North University of China, Taiyuan 030051, China)

Abstract: On the basis of analyzing pattern matching algorithm, this paper puts forward an improved algorithm and applies it to data analysis system that can deal with large-scale data. The system establishes an index table in the beginning of data analysis, then analyses data with index table and frame structure table. The algorithm can efficiently resolve large-scale data problem and improve system performance.

Keywords: pattern matching algorithm; large-scale data; index table; frame structure; data analysis

在数据采集系统中，随着问题复杂度和采集精度的提高，需要采集的数据越来越多，特别是高分辨率传感器的应用，使得采集数据量正在呈几何级数增长[1]。有效的处理这些大规模数据(数据量为 1G 以上)，对计算机的性能和应用软件的设计提出了更高的要求。

对于大规模数据的处理问题。日前有许多专家从不同侧面进行过研究。例如在数据压缩方面，已经有很多成熟的方法；在数据库以及数据仓库方面，也有一些完善的理论和方法。本文通过引入模式匹配算法来实现大规模数据的分析和处理，尤其对于实时采集数据的处理更佳有效。

1 模式匹配算法

1.1 KMP 算法[2]

对于给定的长度为 l 的模式串 T(模式串)。需要确定 T 在另一长度为 n (一般 l<< n)的串 S(也称其为母串)中是否出现以及出现的位置，这就是模式匹配，也称模式识别，又称串匹配或串查找，它是各种数据处理系统中最重要的操作之一。

假设母串 S 的长度为 n，模式串 P 长度为 m，i 指针指向母串中当前比较字符，j 指针指向模式串中当前比较字符。每趟匹配过程中出现字符比较不等时，不需回溯 i 指针，而是利用已经得到的“部分匹配”的结果将指针 j 向左移动一段距离后，继续进行匹配。该算法的关键在于预先根据模式串构造针对每个字符的左移匹配位置数组 NextPos[]，然后通过 NextPos[] 数组来得到字符不匹配时跳跃距离。NextPos[]数组的计算方法如下：

$$\text{NextPos}[j]=\begin{cases}0 & j=1\\ 1 & other\\ \text{Max}\{k \mid 1<k<j \text{ and } 'P_1\ldots P_{k-1}'='P_{j-k+1}\ldots P_{j-1}'\} & \end{cases}$$

① 基金项目:国家自然科学基金(50535030)
收稿时间:2009-05-21

1.2 改进的模式匹配算法(I_KMP 算法)

在数据采集系统中，为了便于数据分析处理，采集回来的数据通常按照某种固定的帧结构格式进行存储，然后交由数据分析模块进行分析处理[3-5]。比如在某个数据采集系统中，需要采集 15 路信号，每个信号采样点占用两个字节空间，按照高字节在前，低字节在后的格式存储，那么对 15 路信号的每次采样需要 30 个字节空间存储有效数据，为了记录采样时间和便于监测数据的完整性，通常在有效数据字节的后面增加帧计数和帧标志数据字节信息。这样分析软件就可以根据附加的信息对帧结构的完整性进行分析汇总，提取每路信号的有效数据。

针对 KMP 算法和数据分析中帧结构长度固定的特点，对 KMP 算法进行了改进。当模式串与母串匹配后，对母串剩余的字符进行模式匹配时，我们同时修改指针 i，让其向右移用一个固定距离(根据针长度决定距离长度)，从而减少比较的次数。假设帧长度为 36 字节，帧结构如图 1 所示，每个单元格为两个字节，匹配成功后 i 指针的右移距离 iPos 计算方法如下：

Chanel1	Chanel2	Chanel3	Chanel4
Chanel7	Chanel8	Chanel9	Chanel10
Chanel11	Chanel12	Chanel13	Chanel14
Chanel15	帧计数	帧标志 1	帧标志 2

图 1　帧结构图

$$iPos=\begin{cases}30 & \text{下一帧结构完整}\\ 1 & \text{下一帧结构不完整}\end{cases} \quad (1)$$

2　帧结构表和索引表构造方法

由于采集数据的复杂和数据量的巨大，为了加快对测量数据的访问速度，将索引技术引入到测量数据处理中。设计测量数据处理分两个阶段：数据预处理阶段和详细分析处理阶段。

在数据预处理阶段，优化软件对所有数据进行一次从头到尾的遍历，建立测试数据索引表，通过索引表再去访问测试数据将大大加快对测试数据的访问速度。有了测试数据索引，分析软件可以根据索引信息读取相应的测试数据进行详细分析。

在数据预处理阶段主要是对测试数据进行粗解析，一方面，通过预处理获取测试数据中有用信息，比如测试数据的位置和时间戳；另一方面，标识出数据中的无效信息。并不是所有的测试数据都是有效的，通过预处理可以验证测试数据的有效性，将无效的测试数据在索引中标识出来，保证处理软件能够跳过无效的测试数据。

在详细处理阶段，分析软件将根据索引表和帧结构表读取相关数据进行下一步的数据分析，比如曲线分析、信号谱分析、数据回放等。

2.1 帧结构表定义

为了定位帧结构中的每一路信号的准确位置，引入了三个一维数组 SFDotCnt[]，DotPos[]，DotOffset[]，其中 SFDotCnt[]数组用于存放一帧中每路信号的采样点数；DotPos[]数组用于依次存放一帧中每路信号的采样点数据存储位置；DotOffset[]数组用于存放一帧中每路信号的第一个采样点数据在 DotPos[]数组中的相对位置。在软件启动后根据具体的帧结构格式初始化三个数组的值，在数据分析的过程中，根据这三个数组就可以计算出每路信号的每个采样点数据的位置，计算公式如下：CPos= StartPos+DotPos (DotOffset(ChanelNum) + DotCode)，其中 StartPos 表示当前帧的起始地址(可以通过索引表获取)，ChanelNum 为当前信号的编号，DotCode 为当前信号中的采样点编号，在采样率不一样的数据采集系统中使用。

2.2 索引表构造方法

为了定位每个完整的帧在原始数据文件中的位置，通过对原始数据文件执行 I_KMP 算法建立数据文件的索引表，具体过程如算法 1 和算法 2 所示。

```
Status CreateIndex()
{
Openfile();//打开原始数据文件
BlockLen=100MB;//每次读取文件块大小
BlockNum=⌈Flen/BlockLen⌉;//块数量
for (i=0;i<blocknum;i++)
{
   Data[ ]=ReadData(i);//读取相应块的数据
   I_KMP(Data[ ],SubString[ ]);
}
Return OK;
}
```

算法 1　建立索引表

```
Void I_KMP(Data[ ],SubString[ ])
{  i=0;j=0;//母串和模式串的指针
   While (i<CurDataLen)// CurDataLen 为母串长
度
   {
     if Data[i]==SubString[j]
     {i++; j++;
      if j>SubSlen
       { AddIndexTable() ;//记录新的帧起始位
置
         i=i+iPos;// iPos 为匹配后 i 指针试探
逃过
         j=0;        //的距离
       }
     }
    else
    {j= NextPos[j];if j= =0 i++;}
   }
}
```

算法 2 I_KMP 模式匹配算法

3　帧结构表和索引表构造方法

该软件主要包括以下几个模块：硬件指令接口、数据预处理、数据信号曲线分析、数据回放、实时监测、数据打印输出、数据导出和图形输出。具体功能如下：

硬件指令接口：完成软件系统和硬件系统的指令通信和数据传输任务；

数据预处理：对原始数据执行模式匹配算法，建立索引表；

数据信号曲线分析：对选定的信号进行曲线分析、实时跟踪和谱分析等；

数据回放：对原始数据进行动态回放，观察数据变化情况；

实时监测：监测硬件传输回来的数据，分析当前的硬件工作状态；

数据打印输出：对选定的信号数据通过打印机输出；

数据导出：对选定信号数据以文本格式或 Excel 格式输出到相应文件中；

图形输出：将信号分析曲线保存为图片格式。

软件主界面和菜单效果图见图 2 和图 3。

4　软件测试

软件测试环境为 Pentium IV,1.6GHz 主频，512MB 内存，硬盘 80G，WindowsXP 操作系统。

图 2　软件主界面图

图 3　软件菜单图

4.1 功能测试

图 4　信号选择界面

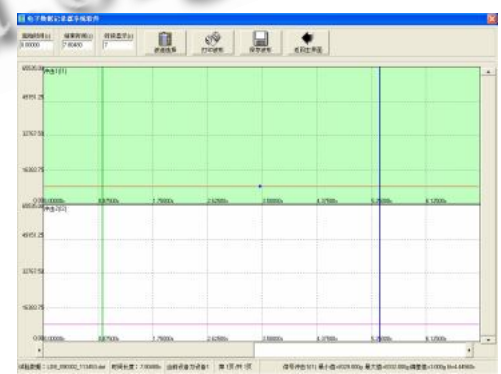

图 5　数据分析界面

通过软件的数据分析菜单，选择打开文件菜单项，该菜单项的功能是对原始数据文件执行模式匹配算法(I_KMP 算法)的同时建立索引表，并对保存到索引表文件，如果索引表文件已经存在，则直接读取索引表

文件，将索引表调入内存。随后选择数据分析菜单项，在选择好分析的信号数量之后，就可以对相关信号进行下一步分析处理，信号选择界面如图 4 所示，数据分析界面如图 5 所示。功能测试结果表明该软件满足实际应用的需求。

4.2 性能测试

利用系统提供的计时器功能对软件的响应速度进行了时间统计分析，由于该软件主要在建立索引和曲线显示时最好时间，所以主要记录了建立索引表的时间和曲线显示时间。统计结果如表 1 所示。

表 1　建索引时间统计结果

数据容量(GB)	建索引时间(s)
1.8	10.325
1.0	4.982
500	2.269

表 2　显示时间统计结果

显示数据量(GB)	单路信号显示时间(s)
0.1	1.534
0.2	2.863
0.5	7.379

由于在软件中建立索引表只是在第一次分析数据时使用，所以我们 10 几秒的时间对于用户来说可以忍受。但是对于显示时间问题，如果同时显示多路的话，显示时间将变得不可忍受，针对这个问题，我们在绘制曲线之前，首先对绘制区域的数据提取特征点，然后根据特征点绘制曲线，可以大大缩短显示时间，满足用户的需求。表 3 为采用特征点算法后，显示时间统计结果。

表 3　特征点显示时间统计结果

显示数据量(GB)	单路信号显示时间(s)
0.1	0.528
0.2	0.963
0.5	1.439

5 小节

文中提出了一种基于改进的模式匹配算法的大规模数据处理方案。在模式匹配过程中建立原始数据的索引表，为后续的数据分析处理等操作奠定了基础。在对数据分析的过程中为了解决显示时间过长的问题，采用了特征点提取算法对显示数据进行处理，随后再对提取的特征点绘制曲线，从而有效地解决了显示时间过长的问题。对软件进行的功能和性能测试结果表明，该方案对于大规模数据的处理效果是非常理想的，在数据处理软件中得到了广泛的应用。

参考文献

1 吕京国,黄国满,杨明辉.用 Visual C++实现大数据量的快速存取.测绘学报, 2002,27(3):29－32.
2 严蔚敏,吴伟民.数据结构.C 语言版,北京:清华大学出版社, 2001.79－80.
3 巫喜红,凌捷.单模式匹配算法研究.网络与通信, 2006,22(8):202－204.
4 闵联营,赵婷婷.BM 算法的研究与改进.武汉理工大学学报, 2006,30(3):528－530.
5 万国根.改进的 AC_ BM 字符巾匹配算法.电子科技大学学报, 2006,35(4):531－533.