录波分析软件中的图形状态记忆算法

桂 勋[1],姚 兰[2],钱清泉[1]
(1. 西南交通大学 电气工程学院,四川 成都 610031;
2. 成都信息工程学院 控制工程系,四川 成都 610225)

摘要:针对当前电力故障录波分析软件中由于缺少 undo(撤销)功能而导致的人机交互性能差和分析效率低的问题,通过把通道图形对象和通道录波数据间的关系由包含改造为相识,使得算法在设计上避免了内存使用量和效率上的瓶颈。阐述了算法中基于属性包的数据结构及其简化容器定义的技术,引入行为型设计模式 COMMAND 以降低程序设计中的耦合度和复杂度,其中专门论述了 COMMAND 模式中 Client(客户端触发器)和 Receiver(接收器)之间复杂的调用关系,并且给出了使用 2 个栈(RedoStack 和 UndoStack)实现 undo 算法的详细步骤及其类接口。使用该算法实现了各种波形控制操作的 undo 机制,用户可同时在多个显示状态间进行直接切换。

关键词:故障分析;故障录波;设计模式
中图分类号:TM 938 **文献标识码**:A **文章编号**:1006-6047(2007)09-0035-05

目前,国内录波器和继电保护信息系统厂家提供的各种电力故障录波分析软件[1-8]都无法实现图形记忆算法,用户对波形进行放大、缩小、标注或移动后,无法通过某种方式直接回到操作之前的某个图形状态。为了恢复某个图形状态,用户必须来回具体控制波形的显示和移动等操作,这种图形化分析模式降低了用户的工作效率,且不人性化。

另外,目前国内的绝大多数录波分析软件在设计上都采用把图形和数据绑定在一起的策略以简化波形显示的程序设计。在这种模式下进行图形状态记忆,就必须将图形状态和录波数据一起进行缓存,这将带来内存使用量和算法效率的缺陷。而且,图形状态记忆算法的实现具有一定难度,设计不合理往往会带来程序设计上的高复杂性和耦合度。现采用设计模式和动态栈调用实现了一种适用于电力系统录波分析环境的波形分析记忆算法,该算法在录波图形系统中可实现稳定的 undo(撤销)操作,提高了录波图形环境的分析效率和人机交互性。

收稿日期:2006-10-20;**修回日期**:2007-03-03

1 算法设计

1.1 算法基础设计

录波故障分析软件不同于其他的图形分析软件,其特点是图形以通道为单位。对于采样频率高且记录时间长的录波文件而言,绘制每个通道图形的数据容量非常大。各录波分析软件为了避免大容量数据的绘制工作,而提高工作效率,算法上的策略是只绘制当前用户可见区域的图形。为了简化此过程

中的程序设计,各厂家采用的方法是把通道对应的录波数据直接保存到图形对象,即每个图形对象本身就包含大量的录波数据,而这种包含大量绘制数据的通道图形对象就给图形记忆算法的实现设置了一个无法解决的难题——内存使用量。对于一个可能包含几万甚至是几十万个采样点数据的通道图形对象,假如要连续记录其100次图形状态,其内存使用可能就是几十兆,即使有足够的内存,在图形状态切换过程中所付出的性能代价也是无法容忍的。这种设计思路对于图形状态记忆而言并不可取。

为了把算法的内存消耗量降到最小,在设计上必须将通道图形对象和录波数据进行分离,把它们之间的关系由原来的包含关系转变为相识关系。一旦转变为相识关系以后,可实现多个通道图形对象对应于同一个录波数据,另外,将控制通道图形显示的各个变量提取出组成一个通道图形显示属性控制包,针对一个固定大小的属性包进行图形状态记忆,就使得算法可避开内存使用上的瓶颈,实现成为可能。改造后通道图形对象和录波数据间的关系见图1。

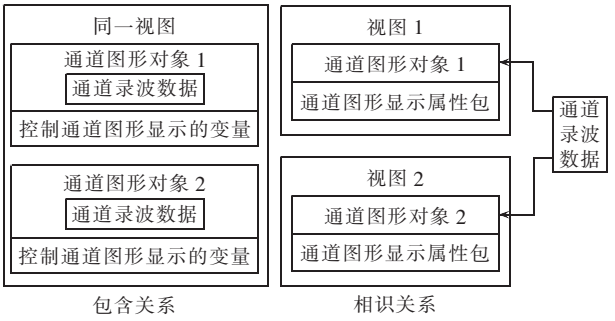

图1 改造后通道图形对象和录波数据间的关系

Fig.1 The reformed relationship between channel graph object and record data

由图1可见,通道图形对象和录波数据之间的关系由包容关系改变为相识关系以后,系统设计可更加灵活。同一个录波数据源的通道图形对象可同时存在于不同的视图中,这就使得用户可把不同录波文件中所感兴趣的通道集合到同一个视图中,便于实现联合分析的理想功能。另一方面,固定大小的通道图形显示属性包,使算法只用较小内存就可对图形状态进行缓冲,算法的执行效率大幅提高。属性包中包含的具体控制变量如表1所示。

由表1可见,属性表中包含了各种和通道显示相关的控制变量,通过修改这些变量就可完全控制通道图形对象的显示,其中通过修改通道绘制起始、结束数组下标,就可实现通道的截取范围显示,而以往的软件系统中必须通过拷贝截取范围的数据来完成此功能,导致消耗内存多且效率低。

由于图形系统中不同的命令负责记录不同的通道显示状态变量,而每个属性变量都和一个通道图形对象指针相关联,面对如此多的命令,算法在设计上采用了模板技术[9]及其STLPORT[10-11]中提供的容器类,简化了繁多的状态记录容器定义。

```
template<class T>
```

表1 属性包的内容

Tab.1 The contents of property package

属性说明	变量定义
是否绘制通道边框	m_bShowFrame
是否填充通道底色	m_bFill
通道数据是否取反显示	m_bInvertShow
绘制通道的逻辑画刷	m_logpen
绘制通道的逻辑画笔	m_logBrush
通道绘制起始、结束数组下标	m_nStartPoint,m_nEndPoint(此2变量组成个ShowArea结构体)
通道显示位置	m_rcPos
通道名称	m_strCh1Name
通道是否显示	m_bShow
通道是采用连线绘制还是点绘制	m_bUseLine
…	…

```
struct ChannelStatePair
{
    CDrawChannel* pChannel;
    T PropertyVar;
};
```

通道位置的属性变量为CRect类型,显示变量为BOOL,而显示范围为ShowArea,和其对应的状态记录向量容器被定义为

```
vector<ChannelStatePair<CRect> >
vector<ChannelStatePair<BOOL> >
vector<ChannelStatePair<ShowArea> >
```

1.2 采用的设计模式简化算法复杂度

在图形化的录波分析过程中,用户可通过各种方式对系统传达各种控制命令(如通过菜单操作、鼠标拖动等),这些命令有很多种。若要分开进行单独的图形状态记忆,会导致算法过于复杂。对此,算法在设计上引入了设计模式理论[12-15]中的对象行为模式——COMMAND来辅助算法设计。

采用COMMAND设计模式,用户输入的各种命令被抽象成一个个有相同接口的命令对象。为了进一步减少内存使用量,每个命令对象只保存其改变的属性值。命令对象被设计为以接口对象为基类的派生类,如图2所示。

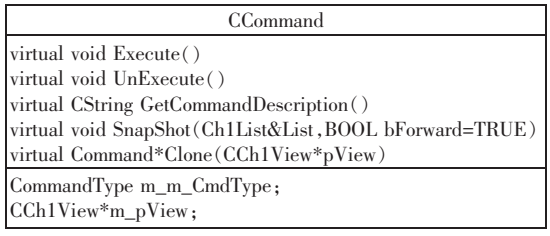

图2 命令对象的接口

Fig.2 The interface of COMMAND object

在接口类CCommand中提供的Execute函数用于具体执行各种命令操作,而UnExecute则用于通过记录的状态还原图形系统的状态,SnapShot函数用于对图形系统的显示状态进行快照,而GetCommand-

Description()返回命令的描述字符串,Clone 实现了基于原型设计的命令对象创建,各个命令对象通过继承的方法重载每个接口的虚函数,从而完成对不同命令对象的具体定制,命令对象间的关系如图 3 所示。

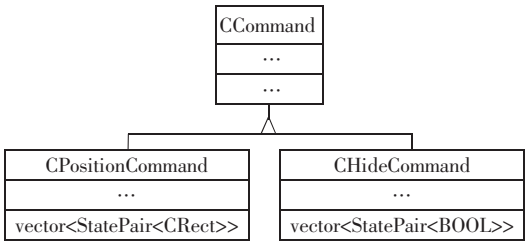

图 3 命令对象的继承树

Fig.3 The inheritance chart of COMMAND object

由于图形状态的记忆和回放是系统内各个对象间一个相互协作的复杂过程,关系相当的复杂。而通过设计模式可理顺其间的复杂关系。COMMAND 模式原理图如图 4 所示。图 4 中,Receiver(接收器)内包含了具体的 Execute 和 UnExecute 的执行函数体,Receiver 和命令对象是相知的关系,在软件设计上 Receiver 用于视图对象或者文档对象,而 Client(客户端触发器)负责根据用户的操作创建对应的命令对象,并让 Receiver 与其相关联,在关联以后 Client 需要调用 SnapShot 对系统的变化前后进行系统快照,并将前后变化的图形状态保存到对应的图形状态容器中,Client 在程序上对应于软件系统执行具体操作的响应函数。在进行具体的 undo 时,只是通过统一的接口(Execute 和 UnExecute)来回调用 Receiver 中的具体执行函数,而 Receiver 对此一无所知,可见采用 COMMAND 设计模式后,具体的命令对象类只是一个中间层,实际执行代码都集中在 Receiver 中实现,Receiver 自己的编译可完全独立于命令对象,大幅降低了系统间的代码耦合度和算法实现上的复杂度。

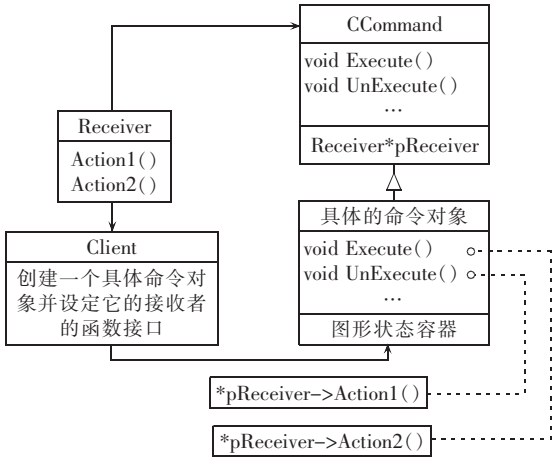

图 4 COMMAND 设计模式图

Fig.4 The design pattern chart of COMMAND

1.3 undo 操作的具体实现

undo 操作的整个逻辑过程如下:当用户通过命令对象对通道图形进行控制之前和之后,命令对象需通过快照接口 SnapShot 保存图形变化之前和之后全部状态数据,然后命令对象被一一缓存,当需要执行 undo 时,就从缓存中取出当前最新的命令对象而后执行 UnExecute 函数,再把此命令对象作为 undo 操作的待选对象进行缓存,当需要进行 undo 操作时从对应缓存中取出最新的命令对象执行 Execute,之后此对象又被当作最新命令对象被缓存起来。显然,缓存命令对象和取命令对象的过程是一个典型的 FILO 过程,此过程正是栈的特点,为此算法采用栈作为数据结构缓存命令对象,栈的实现采用了 STL中的 deque 容器,其接口定义如图 5 所示。

```
CUnDoReDoStack
UnDoStack(int nMaxSize);
virtual ~UnDoStack();
void GetCommandDescriptionList(CStringList& DescriptionList);
void UnExecute(int nStep,CUnDo_ReDo_Stack& redoStack);
void Execute(int nStep,CUnDo_ReDo_Stack& undoStack);
void PushCommand(Command*pCommand);
void Clear();
int Size();
bool IsEmpty();
deque<Command*>UnDoDeque;
int m_nMaxSize;
```

图 5 命令对象栈的接口

Fig.5 The interface of CUnDoReDoStack

算法需要一个撤销栈(UndoStack)和重做栈(RedoStack)来分别缓存撤销操作和重做操作各自需要的命令对象,而算法的核心是 2 个栈间命令对象的动态调用过程,为此算法又设计了一个命令对象处理类 CCommandProcess 封装 2 个栈间命令对象的动态调用过程,其接口定义如图 6 所示。

```
CCommandProcess
CCommandProcessSystem(int nMaxSize);
virtual CCommandProcessSystem();
void Execute(int nStep);
void UnExecute(int nStep);
void PushCommand(Command*pCommand);
void ClearUnDoStack();
void ClearReDoStack();
void ReFreshUnDoCmdList(CStringList& strlist);
void ReFreshReDoCmdList(CStringList& strlist);
bool IsUnDoEmpty()
bool IsReDoEmpty()
int GetUnDoSize()
int GetReDoSize()
CUnDo_ReDo_Stack*m_pUnDoStack;
CUnDo_ReDo_Stack*m_pReDoStack;
int m_nMaxSize;
```

图 6 命令对象处理类的接口

Fig.6 The interface of CCommandProcess

接口中 PushCommand 用于向 UndoStack 中压入新的命令对象指针,而 UnExecute 和 Execute 则分别从 UndoStack 和 RedoStack 中弹出最新的命令对象序列完成撤销和重做操作(其函数参数 nStep 指定了撤销和重做一次执行的步骤数),并且在执行完毕后将命令对象压入相对执行反操作的栈中进行缓存。当栈中缓存的命令对象数超过最大值 m_nMaxSize 以后,在栈底的命令对象将自动从栈中被删除,即

m_nMaxSize 的数值就是图形系统进行撤销操作的最大步骤数。UndoStack 和 RedoStack 的动态调用关系见图 7,图中分 3 种情况讨论了算法的操作过程。

①表示在用户操作录波图形系统过程中不断地将新创建的命令对象压入 UndoStack 中进行缓存的情况,此时没有进行撤销操作,RedoStack 为空。

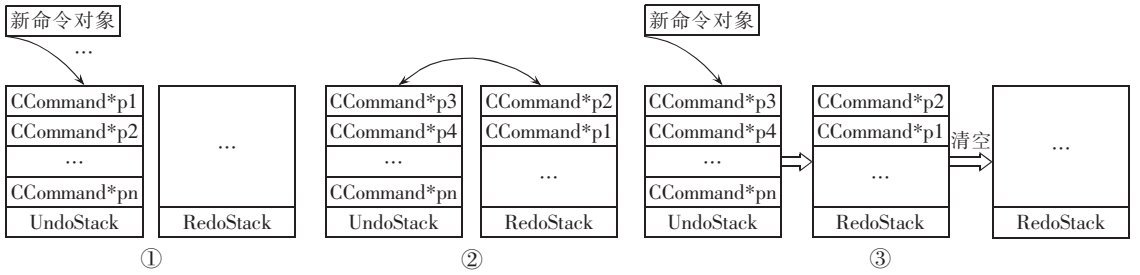

图 7 UndoStack 和 RedoStack 间的动态关系

Fig.7 The dynamic relationship between UndoStack and RedoStack

②表示用户在来回使用撤销和重做操作,这其间用户没有输入任何新的控制命令,命令对象只作为栈中的存储元素在 2 个栈间不断地根据操作被压入和弹出,图形显示可快速地在多个状态间进行切换。

③表示当用户在连续进行撤销操作后恢复到图形显示的某个状态,又接着向图形系统输入新的控制命令,此时最新的命令对象被压入 UndoStack 中,需要立刻清空 RedoStack。该情况下,UndoStack 和 RedoStack 中保存的命令对象在时间上执行顺序的已经被打乱,若 RedoStack 不清空,执行重做操作后所得到的图形状态就已经不是原有的状态了。

2 算法效果

该算法已经运用于晶控自由故障分析软件中,系统采用该算法实现了各种波形控制操作的 undo 机制,用户可同时在多个显示状态间进行直接切换,极大增大了系统的快捷性,使得全图形化的录波故障分析环境更加人性化,运行效果如图 8~11 所示。

图 9~11 是用户经过各种操作而得到图形状态,可见用户的全部操作都被记录下来,通过快速的撤销操作,图形可恢复到图 8 所示的图形原始状态,现在此系统已经可以轻松记录数百次的图形状态,电力故障分析用户可快速地在多个图形状态下来回切换进行波形比对分析,提高了故障分析的效率。

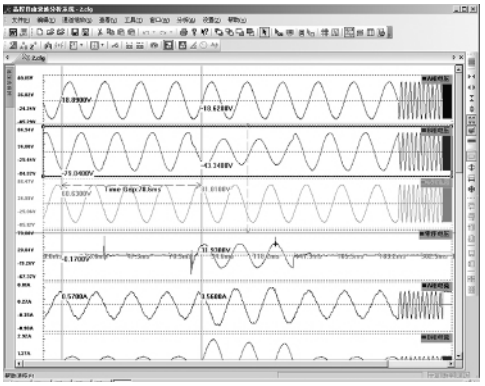

图 8 原始状态波形图

Fig.8 The original state of wave chart

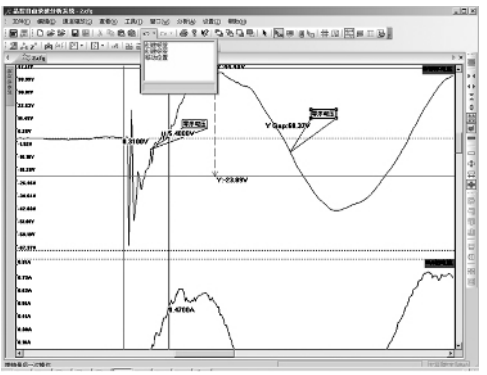

图 9 局部波形缩放

Fig.9 The zoom of partial curve

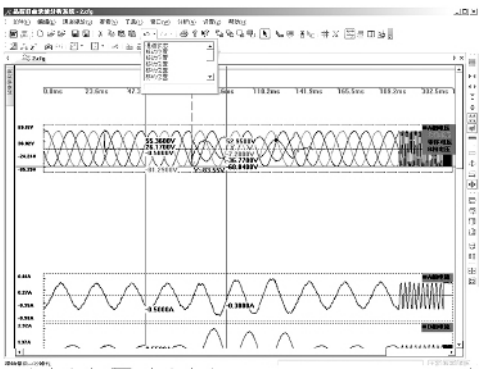

图 10 任意通道的坐标融合显示

Fig.10 The merge of curves

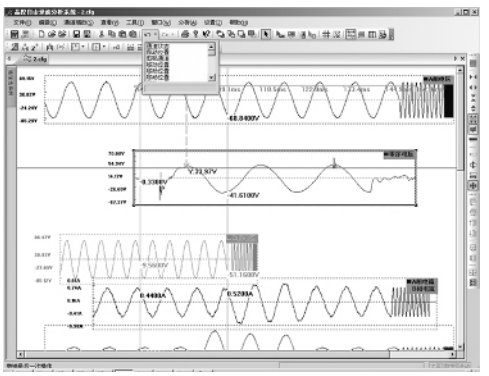

图 11 波形的自由移动和截取显示

Fig.11 The interception and movement of curve

3　结论

讨论了现有录波分析软件未能实现任意 undo 操作的原因,并给出了大容量数据显示环境中实现图形记忆的策略和算法,结合设计模式使算法的复杂度和代码间的耦合度大为降低。期望在不久的将来我国现有的电力故障分析软件都能结合设计模式,从而实现更快捷、高效和人性化的全图形化分析环境。

参考文献:

[1] 桂勋,郭凯,谭永东,等. 基于网络的全图形化故障录波分析软件系统[J]. 继电器,2004,32(24):44-50.
GUI Xun,GUO Kai,TAN Yong-dong,et al. All-graphic software system for fault record analysis based on network[J]. Relay,2004,32(24):44-50.

[2] 郑敏,黄华林,吕鹏,等. 故障录波数据通用分析与管理软件的设计[J]. 电网技术,2001,25(2):75-77.
ZHENG Min,HUANG Hua-lin,LÜ Peng,et al. General analysis and management software for transient data from protective relaying and fault recorder[J]. Power System Technology,2001,25(2):75-77.

[3] 宋墩文,蒋宜国,许勇. 波形数据通用分析系统的设计[J]. 电网技术,2002,26(11):77-79.
SONG Dun-wen,JIANG Yi-guo,XU Yong. Design of versatile analysis software for waveform data[J]. Power System Technology,2002,26(11):77-79.

[4] 张杰,涂东明,张克元. 基于 COMTRADE 标准的故障录波的分析与再现[J]. 继电器,2000,28(11):20-23.
ZHANG Jie,TU Dong-ming,ZHANG Ke-yuan. Analysis and representation of the recorded fault based on standard COMTRADE[J]. Relay,2000,28(11):20-23.

[5] 刘天斌,王永业,柳焕章,等. 基于 COMTRADE 格式的故障分析管理系统[J]. 继电器,2001,29(11):47-49.
LIU Tian-bin,WANG Yong-ye,LIU Huan-zhang,et al. The fault data management & analysis system based on COMTRADE[J]. Relay,2001,29(11):47-49.

[6] 陈佳胜,曾克娥,孙扬声,等. 大型发变机组故障录波装置分析软件的研制[J]. 电网技术,2002,26(5):76-80.
CHEN Jia-sheng,ZENG Ke-e,SUN Yang-sheng,et al. Development of analysis software of fault recorder for high power generator-transfromer bank[J]. Power System Technology,2002,26(5):76-80.

[7] 刘志超,黄俊,承文新,等. 电网继电保护及故障信息管理系统的实现[J]. 电力系统自动化,2003,27(1):72-75.
LIU Zhi-chao,HUANG Jun,CHENG Wen-xin,et al. The implementation of management information system for protective relaying and fault recorder[J]. Automation of Electric Power Systems,2003,27(1):72-75.

[8] 杜新伟,李媛,刘涤尘. 电力故障录波数据综合处理系统[J]. 电力系统自动化,2006,30(12):75-80.
DU Xin-wei,LI Yuan,LIU Di-chen. Integrated processing system for power fault recording data [J]. Automation of Electric Power Systems,2006,30(12):75-80.

[9] JOHNSTON B. 现代 C++程序设计[M]. 曾葆青,译. 北京:清华大学出版社,2005.

[10] JOSUTTIS N M. C++标准模板库[M]. 侯捷,译. 武汉:华中科技大学出版社,2002.

[11] STEPANOV A,AUSTERN M. Background mainpage[EB/OL]. [2006-10-17]. http://www.stlport.org.

[12] GAMMA E,HELM R,JOHNSON R,等. 设计模式:可复用面向对象软件的基础[M]. 李英军,马晓星,蔡敏,等,译. 北京:机械工业出版社,2002.

[13] FREEMAN E,FREEMAN E,SIERRA K,et al. Head first design partterns[M]. 南京:东南大学出版社,2005.

[14] SHALLOWAY A,TROTT J R. Design patterns explained:a new perspective on object-oriented design[M]. 2nd ed. 北京:中国电力出版社,2005.

[15] ALEXANDRESCU A. C++设计新思维:泛型编程与设计模式之应用[M]. 侯捷,於春景,译. 武汉:华中科技大学出版社,2002.

(**责任编辑:**李育燕)

作者简介:

桂　勋(1978-),男,贵州安顺人,博士研究生,从事电力系统自动化研究(**E-mail**:guinh3@263.net)。

Memory algorithm of wave chart state in fault record analysis software

GUI Xun[1],YAO Lan[2],QIAN Qing-quan[1]

(1. Southwest Jiaotong University,Chengdu 610031,China;
2. Chengdu University of Information Technology,Chengdu 610225,China)

Abstract: Aiming at the low analysis efficiency and bad MMI in current fault record analysis software without undo function,the relationship between channel graphic object and channel record data is changed from CONTAINMENT to ACQUAINTANCE to avoid the efficiency bottleneck of memory usage in software design. The data structure based on property package and the technique of simplified container definition are expatiated. By introducing the behavioral design pattern COMMMAND,the complexity and coupling depth in program design are lowered. The complicated CALL relationship between Client and Receiver in COMMAND mode is especially discussed,and the detailed implementation steps of undo algorithm with two stacks(RedoStack and UndoStack) and its CLASS interfaces are given. By using the proposed algorithm,the undo mechanism for all wave controls are realized,and different display states can thus be directly switched over.

Key words: fault analysis; fault record; design patterns