New Applications of PAX in Physics Analyses at Hadron Colliders

Martin Erdmann[1], Ulrich Felzmann[2], Dominic Hirschbühl[2], Christopher Jung[2]
Steffen Kappler[1], Matthias Kirsch[1], Günter Quast[2], Klaus Rabbertz[2], Jens Rehn[3], Sven Schalla[2]
Patrick Schemitz[2], Alexander Schmidt[2] (c.a.), Thorsten Walter[2], Christian Weiser[2]
[1]RWTH Aachen, [2]University of Karlsruhe, [3]CERN

Abstract

The PAX (Physics Analysis eXpert) toolkit assists physicists in the analysis and interpretation step of a particle physics research project. Its aim is to provide a new level of abstraction beyond detector reconstruction which facilitates code reuse and unification. PAX makes use of fourvector arithmetics and offers sophisticated relation management and memory management functionality. This paper gives an overview of the toolkit and reports about its application in the hadron collider experiments CDF and CMS.

MOTIVATION

Physicists usually apply personalized analysis frameworks which certainly are well suited for individual use cases, but have not always been optimized with respect to modern software design principles. One of the main motivations for the development of the analysis toolkit PAX [1] is to provide physicists with well crafted design concepts which results in a gain of efficiency concerning development and performance of analysis code. Some keywords are memory management, relation management (to enable construction of decay trees) and persistency (save/load decay trees and different stages of the analysis to/from disk).

To facilitate efficient analysis development and to support team work, it is advisable to encapsulate the physics analysis code in a layer which ideally is independent of the data source. There have been earlier approaches e.g. in the H1 and ALEPH experiments (H1PHAN [2] and ALPHA [3] projects). The good experience with these projects motivated the development of a C++ toolkit which is capable to cope with the requirements of future collider experiments.

One of the advantages of a separate layer for the analysis code is the protection from changes in the user interface of the underlying detector reconstruction framework. Only the interface between data source and PAX objects has to be changed in this case. Another advantage is the possibility to apply the identical analysis code to various data sources (detector data, Monte Carlo generator data) and even share it among different experiments.

By providing an abstract language for the formulation of the physics analysis, the user is enabled to develop generalized analysis algorithms which may be applied in diverse contexts.

A simple example of such a generalized algorithm is the kinematics calculation of a W-boson in the decay mode $W \to l\nu$ by using a reconstructed charged lepton, missing transverse energy and a W-mass constraint. This algorithm gives two independent solutions (interpretation possibilities) for the boson. An implementation of this frequently encountered task is available in one of the example applications for the PAX toolkit [4].

THE STRUCTURE OF PAX

The PAX toolkit provides a collection of classes which provides containers and services necessary for the aforementioned requirements. In this section an overview of the basic design concepts of PAX is given.

The PaxFourVector Class

The *PaxFourVector* by default inherits the full functionality from the *HepLorentzVector* of the CLHEP [5] library (optionally, the user can choose the *TLorentzVector* of the ROOT [6] library instead). In addition to the fourvector arithmetics of the base classes, PAX provides some data members and methods that are proven to be useful by previous experiments (see Fig. 1). It also adds an implementation of local relation management (see *PaxRelationManager* section).

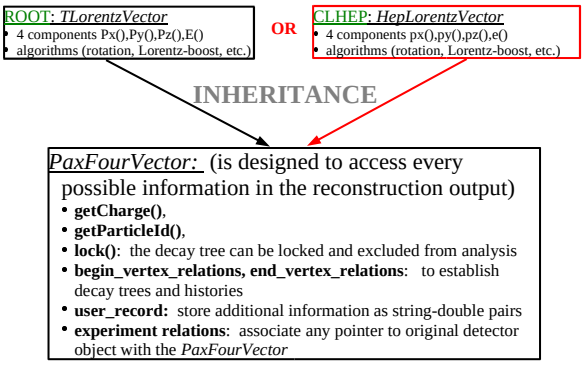

Figure 1: The *PaxFourVector* Class

As every PAX physics object, the *PaxFourVector* has a "user-record". This is a key-value map that enables the user to associate additional information to the fourvector such as cone sizes of jet algorithms, quality of track reconstruction etc.

Using the "experiment relations", it is possible to store

pointers to detector objects in the *PaxFourVector*, as described in more detail in the *PaxRelationManager* section.

The PaxVertex Class

The *PaxVertex* has similar properties as the *PaxFourVector*. The main difference is that *PaxVertex* is a threevector and therefore inherits from *Hep3Vector* of CLHEP, or *TVector3* of ROOT, respectively.

The PaxCollision Class

The PaxCollision provides a hook to handle multicollision events, as they occur at high-rate hadron colliders. It has no vector properties but it provides the relation management to associate physics objects and separate the collisions in the event.

The PaxEventInterpret Class

The *PaxEventInterpret* is a general container for the PAX physics objects (*PaxFourVector, PaxVertex, PaxCollision*). It is designed to represent one distinct interpretation of one event. This means that in case of many possible parallel interpretation possibilities, the user creates each interpretation in a separate *PaxEventInterpret*. This way, the analysis may be advanced into different directions starting from one *PaxEventInterpret* and making a copy for each hypothesis (e.g. connecting particles to the decay tree in different ways).

The copy of a *PaxEventInterpret* is effectively a deep copy. All the objects contained in a *PaxEventInterpret* are duplicated and the relations are set up correctly to stay within the copy.

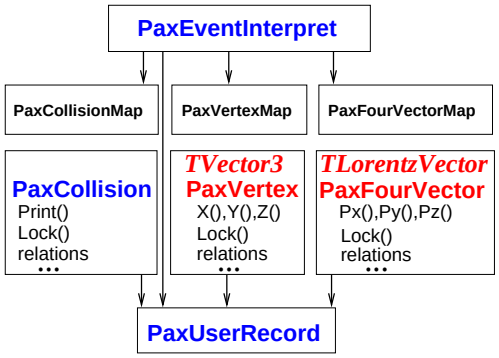

Figure 2: The *PaxEventInterpret* Class

The *PaxEventInterpret* takes over object ownership as soon as a new object is registered in it. It takes care of proper cleaning up of the memory as soon as the considered object is no longer needed. This adds some convenience for the user who does not have to worry about object ownership, or memory leaks.

The *PaxEventInterpret* is persistent. All the contained objects, including their relations, can be written to the storage device and read back to memory.

The PaxRelationManager

The primary functionality of the *PaxRelationManager* is the management of decay trees. The manager is based on the "Mediator" design pattern [7]. This means that the relations are local, in the sense that each object knows the related objects, but there is no global map or directory of the relationships. Each *PaxVertex* has outgoing and incoming fourvector relations, and each fourvector has begin- and endvertices. For an illustration see Figure 3. Here, two of

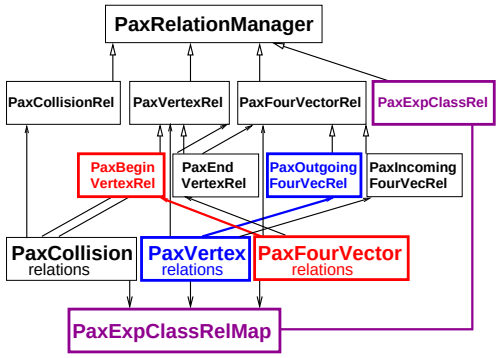

Figure 3: The *PaxRelationManager* Class

the components that constitute a decay tree are highlighted (a *PaxFourVector* has begin-vertex relations and a *PaxVertex* has outgoing fourvector relations).

The relation manager is also used to record an analysis "history". Each object which is copied keeps a pointer to its original instance. This way the user may always go back and ask for the original properties of an object which might have changed during the development of the analysis.

The *PaxRelationManager* also implements a so called locking mechanism. Using this mechanism it is possible to exclude parts of a decay tree from the analysis (i.e. excluding a lepton from a jet finding algorithm). If one fourvector or vertex is locked, all the objects down the decay tree will be locked, too.

Figure 3 also shows another essential element of the relation manager. The *PaxExperimentClassRelation* can store any pointer to an original detector object and associate it to a PAX physics object. This way, it is always possible to access the original detector information through a PAX object at a later stage of the analysis. In the current implementation of PAX this feature is not available when writing a *PaxEventInterpret* to disk.

Accessing Objects, the PaxIterator

To access the objects which are stored in the *PaxEventInterpret*, or which are related to the considered physics object, one can generally use the *PaxIterator*. It has a uniform syntax which works for almost all cases of object access.

For convenience, there are also some methods that allow direct access without using iterators, such as get mother/daughter particles.

EXPERIMENT INTEGRATION

As already mentioned before, it is advantageous to formulate the physics analysis itself independently from the underlying experiment environment to enable code reuse and to protect from changes in the experiment interface. To achieve this goal it is necessary to provide interface classes for the particular experiment. It is also possible to interface PAX to other data sources like Monte Carlo event generators or fast simulation programs. Some of these interfaces are provided and maintained by the PAX authors on the PAX webpage [4]. Further interfaces provided by PAX users are also made available on this webpage.

At the current time, interfaces are available for the experiments CDF (Tevatron) and CMS (LHC). In the CMS case PAX has been included in the CMS software environment as an external package. That means that the user does not have to take care of a proper installation because it is always available when the CMS environment is set up. To include the PAX libraries into the user analysis, only a single line has to be added to the so called "BuildFile" of SCRAM [8] which takes care of the CMS software configuration and build system. An extensive example analysis can be found in the "Examples" subsystem of the CMS detector reconstruction software ORCA [9].

The CDF implementation has been provided for the so called "Stntuple" [10] in a similar way as described for CMS.

APPLICATIONS

In this section two selected physics analyses, from the experiments CMS and CDF, in which the PAX toolkit has been applied, are presented

$t\bar{t}H$ Analysis in CMS

The PAX toolkit has been used in the study of the channel $t\bar{t}H$ (with $H \rightarrow b\bar{b}$) for CMS, using generator data and data from full detector simulation as input sources.

The challenge is the full reconstruction of the $t\bar{t}H$ partonic process as shown in the two Feynman Graphs in Figure 4. The difficulty resides in the large number of interpretation possibilities of the final state of this channel. In the case of perfect b-tagging and perfect jet reconstruction, the number of interpretation possibilities is 24 for graph a). For a visualisation of the concrete task see Figure 5.

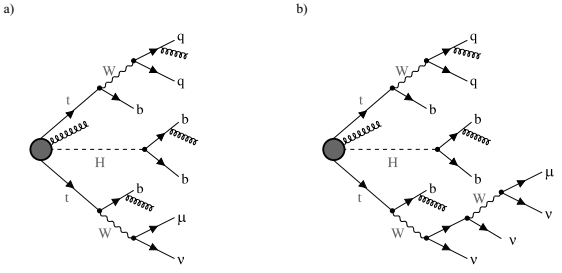

Figure 4: The $t\bar{t}H$ Feynman Graphs

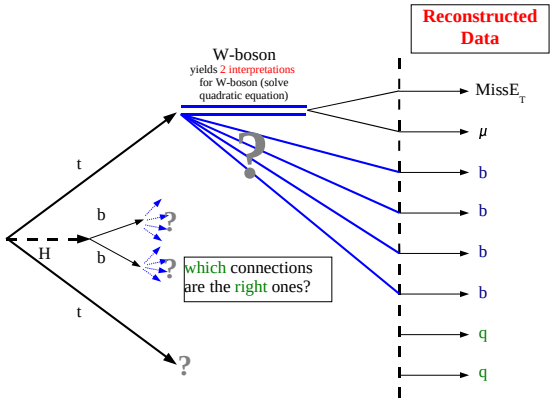

Figure 5: Visualisation of the interpretation possibilities of the $t\bar{t}H$ final state

The reconstruction of the W-boson yields two solutions: the longitudinal part of the neutrino's momentum is unknown and one is restricted to measuring missing transverse energy. The unavailable component has to be extracted by using the W-mass as constraint and solving the resulting quadratic equation.

The largest source of combinatorial multiplicities results from the number of possibilities to connect the b-jets to the decay tree. In case of realistic b-tagging or gluon radiation there are many more interpretation possibilities. The need for an efficient management of the combinatorics is obvious.

In so called "analysis factories" all the interpretation possibilities are realized and stored in separate *PaxEventInterpret*s. At a later stage of the analysis, the probability for each interpretation is calculated by Likelihood criteria and a "quality" is assigned to each *PaxEventInterpret*. The user then may decide to choose the interpretation with the best quality for the final result or even choose all the interpretations at the same time using a weight which can be calculated according to the quality.

The resulting mass plots [11] of this analysis are shown in figure 6 and 7. It is important to point out that these plots have been created with identical analysis code, but with different data interfaces.

$t\bar{t}$ Production at CDF

An analysis of $t\bar{t}$ reconstruction in the electron-plus-jet decay channel [12], shown in Figure 8, is similar to the previously discussed $t\bar{t}H$ channel with respect to the combinatorial task and reconstruction of the decay tree.

For this study, the Pythia Monte Carlo Generator and CDF detector simulation have been used. The resulting example top mass plots for the hadronically and for the leptonically decaying W-boson are shown in Figures 9 and 10 respectively.

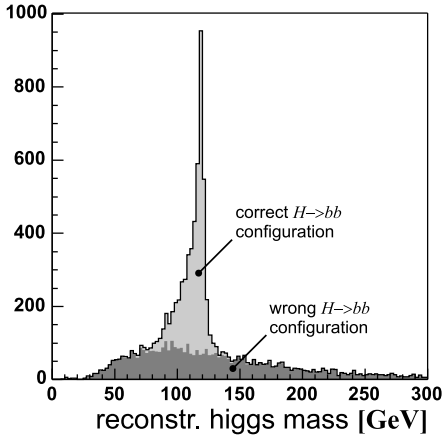

Figure 6: The reconstructed Higgs mass in the $t\bar{t}H$, $H \rightarrow b\bar{b}$ channel on generator level. Shown is the signal peak with the correct decay tree combination and the background resulting from wrong combinations.

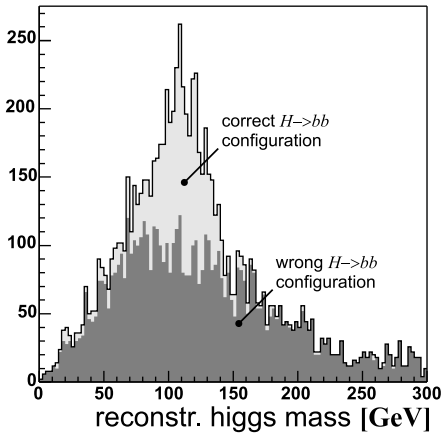

Figure 7: The same plot as in the previous figure but after full detector simulation and reconstruction [11]. The same analysis code has been used.

SUMMARY

The PAX library provides tools to enable an efficient physics analysis of hadron collider data. It focuses on reconstruction of complex decay trees and parallel processing of different interpretation possibilities of an event. It implements container, fourvector, vertex and collision

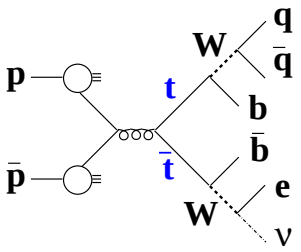

Figure 8: Example Feynman Graph of $t\bar{t}$ production.

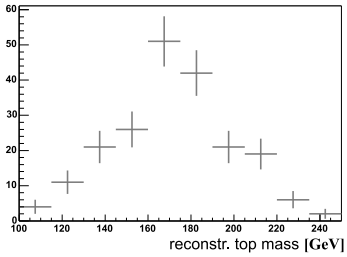

Figure 9: Reconstructed top mass from one b-jet, the electron and missing E_t using events generated with Pythia and full CDF detector simulation.

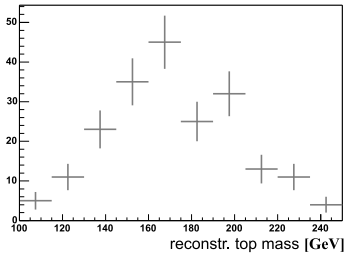

Figure 10: Reconstructed top mass from two jets resulting from W-decay, and one b-jet.

classes and provides a sophisticated relation and memory management. PAX has successfully been applied in various analyses including a $t\bar{t}H$ study in CMS and top analysis in CDF.

REFERENCES

[1] "Physics Analysis Expert (PAX): first applications", M. Erdmann et al., Proceedings of the International Conference on Computing in High Energy and Nuclear Physics, CHEP03, La Jolla, USA (2003), physics/0306085

[2] H1 Collaboration, internal software manual for H1PHAN

[3] ALEPH Collaboration, "ALPHA" internal note 99-087 SOFTWR 99-001

[4] `http://cern.ch/pax`

[5] CLHEP, A Class Library for High Energy Physics, `http://proj-clhep.web.cern.ch/proj-clhep`

[6] An Object-Oriented Data Analysis Framework, `http://root.cern.ch`

[7] E. Gamma et al., "Design Patterns", Addison Wesley, ISBN 0-201-63361-2 (1994)

[8] Software Configuration, Release And Management, `http://spi.cern.ch/cgi-bin/scrampage.cgi`

[9] Object-oriented Reconstruction for CMS Analysis, `http://cmsdoc.cern.ch/orca/`

[10] P.Murat et al., "Stntuple manual", CDF internal software manual (2004)

[11] S. Kappler, *"Higgs Search Studies in the Channel $t\bar{t}H$ with the CMS Detector at the LHC"*, PhD Thesis at University of Karlsruhe, **IEKP-KA 2004/17, part I, 2004.**

[12] D. Hirschbühl, PhD Thesis in preparation, University of Karlsruhe