

LHADA to XML converter & CutLang

Gokhan Unel / UCI
Nov 2016

Introduction

- Disclaimers:
 - These are mostly exploratory prototypings
 - If the only tool you know is ROOT, all problems seem like C++ codes.
- The programs, snippets, docs etc. are all publicly available in gitlab
 - <https://gitlab.com/Unel/deneme1.git>
 - there is a makefile to help with compilation
- the codes are mostly examples, look, copy, steal as you see fit.

LHDA-XML

- Started from a text file from twiki, tried to convert it into XML using ROOT and the TXMLEngine class

```
mac-unel-nbp2:LHADA ngu$ cat deneme.txt
info analysis
# Details about experiment
id SUSY-2013-15 # we dont know
publication JHEP11(2014)118
sqrtS 8.0
lumi 20.0
arXiv 1407.0583
hepdata https://atlas.web.cern.ch/Atlas/GROUPS/PHYSICS/PAPERS/SUSY-2013-15/
```

```
info units
energy GeV
length nm
xsec pb
```

```
function mT
### transverse mass
arg1 tlorentzvector # visible particle
arg2 tlorentzvector # missing energy vector
arg3 float # mass
return float # transverse mass
code http://google.de
doc http://google.de
```

```
function amT2
### asymmetric transverse mass
arg1 tlorentzvector # visible particle
arg2 tlorentzvector # missing energy vector
arg3 float # mass
return float # asymmetric transverse ma
code http://google.de
doc http://google.de
```

converter: 66 lines of c++ code including comments. very easy to learn and use

deneme.xml

```
<?xml version="1.0"?>
<main>
  <info name="analysis" id="SUSY-2013-15" publication="JHEP11(2014)118" sqrtS="8.0" lumi="20.0"
  arXiv="1407.0583" hepdata="https://atlas.web.cern.ch/Atlas/GROUPS/PHYSICS/PAPERS/SUSY-2013-15/" />
  <info name="units" energy="GeV" length="nm" xsec="pb" />
  <function name="mT" arg1="tlorentzvector" arg2="tlorentzvector" arg3="float" return="float"
  code="http://google.de" doc="http://google.de" />
  <function name="amT2" arg1="tlorentzvector" arg2="tlorentzvector" arg3="float" return="float"
  code="http://google.de" doc="http://google.de" />
  <function name="mT2tau" arg1="detectorobj" arg2="string" arg3="float" arg4="ptmin"
  arg5="bool" return="float" code="http://google.de" doc="http://google.de" />
  <function name="mHadTop" arg1="detectorobj" arg2="string" arg3="float" arg4="ptmin"
  arg5="bool" return="float" code="http://google.de" doc="http://google.de" />
  <function name="HTMissSig" arg1="detectorobj" arg2="string" arg3="float" arg4="ptmin"
  arg5="bool" return="float" code="http://google.de" doc="http://google.de" />
  <function name="HT" arg1="detectorobj-list" arg2="integer" return="float" />
  <function name="isBJet" arg1="detectorobj" arg2="float" return="bool" code=" doc" />
  <function name="isTauJet" arg1="detectorobj" arg2="string" return="bool" code=" doc" />
  <function name="isIdentifiedElec" arg1="detectorobj" arg2="string" return="bool" code=" doc" />
  <function name="isIdentifiedMu" arg1="detectorobj" arg2="string" return="bool" code=" doc" />
  <function name="isIsolated" arg1="detectorobj" arg2="string" arg3="float" arg4="float"
  arg5="bool" arg6="float" return="bool" code=" doc" />
  <function name="overlaps" arg1="detectorobj" arg2="detectorobj-list" arg3="float"
  return="bool" code="http://overlap.de" doc="http://overlap.de" />
  <detectorobj name="jets" take="external" algorithm="anti-kt" R="0.4" ptmin="20" etamax="2.5"
  code="http://jets.jet" doc="http://checkmate.jets.de" />
</main>
```

cutlang and past

- A cut language idea was born prior to LHADA.
- A working implementation to help the not-so-programmer-professor to get involved with the analysis was done in dbxA.
 - dbxA is a flat ntuple maker and analyser that our group was using in ATLAS exotic analyses. It can run multiple analyses and/or versions of the same analysis in parallel. (C++ & root AND TProof and GRID)

- analysis defined in text file like:

```
cut0 = "ALL "  
cut1 = "GRL "  
cut2 = "TilErr "  
cut3 = "LarErr "  
cut4 = "TilTri "  
cut5 = "Trig "  
cut6 = "VtxTrks >= 5 "  
cut7 = "NLEP GT 0 "  
cut8 = "NLEP == 1 "  
cut9 = "( NELE == 1 AND NMUO == 0 ) OR ( NELE == 0 AND NMUO == 1 ) " # NLEP >= 1  
cut10 = "TrigMatch "  
cut11 = "OverlapEle "  
cut12 = "BadgoodJets > 0 "  
cut13 = "NJET >= 2 " # jets  
cut14 = "NJET >= 3 " # jets  
cut15 = "NJET >= 4 " # jets  
cut16 = "( NELE == 1 AND MET > 30 ) OR ( NMUO == 1 AND MET > 20 ) " # jets  
cut17 = "BJET >= 1 " # b-jets
```

present and future

- The mechanism to read the text file and define cuts, especially interpret logical statement like

```
"( NELE == 1 AND MET > 30 ) OR ( NMUO == 1 AND MET > 20 ) "
```

```
initial problem = (( 1 + 2 ) * ( 3 / 4 )) - (1+6)
```

- is implemented in root /c++ and is in the git package
 - uses dbxCut class and Shunting-yard algorithm.
- dbxA is evolving to dbxA2 for run2 analyses
 - cutlang can/will be replaced by LHADA
- Could help researchers focus on physics rather than programming quirks and details.