Cluster Analysis: Efficiency and Purity

A Tool for Producing Efficiency Plots

Ron Cassell and Gary Bower, SLAC
LCD Simulation Workshop, May 22, 2003
EfficiencyProcessor

- Description
- Implementation
- Using it
- Examples
Description

• Given a list of clusters, use the MCParticle information to produce efficiency and purity plots.

• Efficiency plots produced for photons, electrons, neutral hadrons, charged hadrons and muons, using the “correct” identity of the clusters. (Purity = 100%)

• If clusters contain identification, actual efficiency and purity plots produced.

• (JAS2 + SIO)
Implementation

- Choose the final state particles
- Assign all calorimeter energy to final state particles
- Build clusters
- Analyze clusters
- Repeat for different configuration
- Overlay efficiency plots
Choosing the final state particles

• CreateFSParticleList(string s) creates the set of final state particles and stores them in Event. Currently 4 options are available:

• “generator”: Use the event generator list (all particles created by the simulator are traced back to a generator particle)

• “simulator”: The end of the simulation chain, with the caveat once a particle deposits energy in a calorimeter, that ends the chain.
Choosing (cont)

• “calorimeter”: default option. End chain at imaginary boundary 20cm in front of EM calorimeter.

• “calnobrem”: same as calorimeter except trace brem particles back to original electron.
Assigning calorimeter energy

• ReplaceCalorimeterHits uses the final state particle list in Event and replaces the CalorimeterHits assigning all the energy to final state MCParticles.(stored in Event)
Analyzing clusters: definitions

• Start with set of final state particles and a set of clusters
• Primary cluster – for each final state particle the cluster that contains the most energy from this particle
• Correct ID – for each cluster, the correct ID is the ID of the maximum energy contributor, only if the cluster is the primary cluster for that particle. Otherwise, the correct ID is fragment.
• Actual ID – the algorithm ID of the primary cluster of each final state particle.
• Fake – cluster which is not the primary cluster for any final state particle with the algorithm ID.
• Purity - # correct IDs/(# correct IDs + # fakes)
Examples

• All Examples use SDMar01, all cluster identification done with ClusterID
• Comparing event types (tt, zz, zh, udscb, zpole) 1000 event samples
• Comparing segmentation (.5x.5, 1x1, 2x2 segmentation in EM calorimeter) tt events
• Comparing algorithms (cheater, simple, simple with energy cut on hits) tt events
Event type
Event type
Event type

- **Zpole - gamma - Energy wted**
- **ttSD - gamma - Energy wted**
- **zzSD - gamma - Energy wted**
- **zhSD - gamma - Energy wted**
- **qqSD - gamma - Energy wted**
Segmentation
Segmentation
Algorithm
Using the tool

• All code in CVS, but not yet released.
• Instructions for running it are at:
  http://www.slac.stanford.edu/~cassell/clustering_efficiencies.htm
import hep.lcd.recon.cluster.simple.*;
import hep.lcd.recon.cluster.cheat.*;
import hep.lcd.recon.cluster.analysis.*;
import hep.lcd.mc.analysis.*;
import hep.lcd.recon.cluster.util.*;
import hep.lcd.util.driver.*;
import hep.lcd.recon.cluster.clusterID.*;
import cjnn.backprop.*;

final public class MyExampleEfficiencyDriver extends Driver
{
    public MyExampleEfficiencyDriver()
    {
        add(new CreateFSParticleList("calorimeter"));
        add(new ReplaceCalorimeterHits());
        add(new SimpleClusterBuilder());
        add(new ApplyClusterIDToSioFile());
        add(new EfficiencyProcessor("IdentifiedClusters",0,0.0,"qqSD"));
        add(new EfficiencyPlotProcessor());
    }
}
Summary

• A working tool exists for evaluating clustering algorithms, which can be used for comparison when varying a multitude of parameters.

• Documentation exists on how to run it.

• Please try it.