ARAna: Athena Root Access analysis

History

- Giacomo ha iniziato a sviluppare un esempio di analisi SUSY usando ARA
- Giacomo, Milano e Lecce hanno deciso di adottare ARAna come punto di partenza comune per le analisi SUSY
- E’ iniziato lo sviluppo di alcuni tool necessari per qualunque analisi (Selection Tool, Overlap Removal Tool)
- La comunità degli “interessati” ad ARAna si è allargata: SUSY + top + tau + …
- Siamo qui oggi a ricapitolare le esigenze comuni e/o individuali e capire come possono essere prese in considerazione

Main concepts

- Si sceglie il formato su cui fare analisi (AOD o DPD) e si crea la classe CollectionTree_trans (MakeClass())
- AnalysisSkeleton carica evento per evento tutte le variabili presenti in CollectionTree_trans e fornisce la base per l'analisi:
  - permette di scegliere che collezione usare per muoni, elettroni, getti, tau, fotoni, etmiss
  - chiama Selection Tool e riempie identifiedMuon con i muoni selezionati (e così via…)
  - chiama Overlap Removal Tool per i muoni selezionati e riempie nonOverlappingMuon con i muoni che non si sovrappongono ad altre particelle (e così via)
  - passa tutte le variabili lette alla classe UserAnalysis che fa i tagli, conta i getti, riempie gli istogrammi etc… (a totale discrezione dell'utente!)

Tool (Selection + Overlap Removal)

- Abbastanza generali da non essere modificati a meno di cambi radicali nell'analisi o nel formato dei dati (caso ideale...vedi slide 6)
- Utilizzabili anche all'interno di ATHENA
- Codice chiaramente leggibile anche da chi non lo scrive
- Configurabili all'interno uno script di ROOT per le opzioni più facilmente modificabili all'interno della stessa analisi (vedi slide 5)
- Configurabili modificando AnalysisSkeleton per le opzioni che non cambiano nel corso di una stessa analisi (es: ordine in cui si effettua Overlap Removal)

Tool configuration

```
sel.SetElectronPtCut(20.*GeV);

sel.SetElectronEtaCut(2.5);

sel.SetElectronCaloIsolationCut(10.*GeV);

sel.SetElectronIdCriteria("tight");

t.SetElectronContainerName("ElectronAODCollection");

ovl.SetElectronDeltaR(0.2);
```

to be addressed I

- policy nell'uso CVS:
  - tag per ogni cambiamento, solo alcuni tag consigliati per l'analisi
  - tag solo quando i cambiamenti sono considerati maturi e validati
  - …
- Quali sono le parti di codice che ci si aspetta che un utente voglia modificare?
  - solo AnalisysSkeleton?
  - anche parte dei Tool?
  - quale è il livello di configurazione che vogliamo avere a runtime (da uno script di ROOT)?

to be addressed II

- i tool agiscono su container o su singole particelle?
  - ```
    Select->Select(MuonCollection,identifiedMuon);
    Analysis::MuonCollection *swap = new MuonCollection();
    Overlap->DeltaR(identifiedMuon,swap);
    Overlap->TrackMatch(swap,nonOverlappingMuon);
    ```
  - ```
    Analysis::MuonContainer::const_iterator muoItr = MuonCollection->begin();
    Analysis::MuonContainer::const_iterator muoItrE = MuonCollection->end();
    for (; muoItr != muoItrE; ++muoItr) {
            if( ! Selection->selectMuon(**muoItr) ) continue;
            identifiedMuon->push_back(*muoItr);
            if( ! Overlap->DeltaR(*muoItr) ) continue;
            if( ! Overlap->TrackMatch(*muoItr) ) continue;
            nonOverlappingMuon->push_back(*muoItr);
    }
    ```

to be addressed III

- configurazione Overlap Removal:
  - diversi metodi (niente, DeltaR, TrackMatch…)
  - diversi parametri per ciascuna particella (DeltaR es...)
  - possibilità di cambiare l’ordine delle particelle
  - …
- al momento una matrice che a ciascuna coppia di particelle associa un metodo + vettore di DeltaR:
  - il codice è abbastanza oscuro if() su ogni riga!!
  - molte cose configurabili a runtime

Overlap Removal

- Black box (tutte le configurazioni in ROOT o editando il tool stesso)
  - `Overlap->Remove(identifiedMuon,nonOverlappingMuon);`
- Una funzione per ciascun “metodo” (rimane da definire con quali particelle uso quale metodo...)
  -
```
if (!Overlap->DeltaR(Muon)){
   if(!Overlap->TrackMatch(Muon))
      nonOverlappingMuon->push_back(Muon)
}
```
- Una funzione per ciascun “metodo” da chiamare per ciascun container:
  -
```
if (!Overlap->TrackMatch(Muon,nonOverlappingElectron)){
   if (!Overlap-> DeltaR(Muon,nonOverlappingTauJet))
      if (!Overlap-> DeltaR(Muon,nonOverlappingParticleJet))
         nonOverlappingMuon->push_back(Muon)
}
```

Conclusion

- Questa è difficile da prevedere…