Pesky find_if

Segment Network Produce Module

```
// match all SpacePoints with the sectors:
for (SpacePoint& aSP : storeArray) {

  const StaticSectorType* sectorFound = findSectorForSpacePoint(aSP);

  if (sectorFound == nullptr) {
    B2WARNING("matchSpacePointToSectors: SP in sensor " << aSP.getVxdID() << " no secto
    nSPsLost++;
    continue;
  }
  nSPsFound++;

  TrackNode* trackNode = new TrackNode( &aSP );
  trackNodes.push_back(trackNode);

  // sector for SpacePoint exists:
  FullSecID foundSecID = sectorFound->getFullSecID();
  B2DEBUG(5, "matchSpacePointToSectors: SP found!: " << foundSecID.getFullSecString());

  vector<RawSectorData>::iterator iter =
    std::find_if(
      collectedData.begin(),
      collectedData.end(),
      [&](const RawSectorData & entry) -> bool
{ return entry.secID == foundSecID; }
    );

  // if secID not in collectedData:
  if (iter == collectedData.end()) {
    collectedData.push_back({ foundSecID , false, nullptr, sectorFound, {trackNode}});
  } else {
    iter->hits.push_back(trackNode);
  }
  B2INFO("matchSpacePointToSectors: trackNode Found2: " << *trackNode); // TODO Jan8_20
```

Segment Network Produce Module

```
// loop over all raw sectors found so far:
for (RawSectorData& outerSectorData : collectedData) {
  ActiveSector<StaticSectorType, TrackNode>* outerSector = new ActiveSector<StaticSectorT
  (outerSectorData.staticSector);

  // skip double-adding of nodes into the network after first time found -> speeding up t
  bool wasAnythingFoundSoFar = false;
  // find innerSectors of outerSector and add them to the network:
  const std::vector<FullSecID>& innerSecIDs = outerSector->getInner2spSecIDs();

  for (const FullSecID innerSecID : innerSecIDs) {
    vector<RawSectorData>::iterator innerRawSecPos =
      std::find_if(
        collectedData.begin(),
        collectedData.end(),
        [&](const RawSectorData & entry) -> bool
/         { return (entry.wasCreated == false) and (entry.secID == innerSecID); }
      { return (entry.secID == innerSecID); }
        );

      // current inner sector has no SpacePoints in this event:
      if (innerRawSecPos == collectedData.end()) { B2DEBUG(5, "SegmentNetworkProducerModule
" " << outerSectorData.secID.getFullSecString() << " had inner sector " << innerSecID.getF
"); nNoSPinThisInnerSector++; continue; }

      // take care of inner sector first:
      ActiveSector<StaticSectorType, TrackNode>* innerSector = nullptr;
      if (innerRawSecPos->wasCreated) { // was already there
        innerSector = innerRawSecPos->sector;
      } else {
```

DirectedNodeNetwork

```
/** checks whether given 'toBeFound' is already in the network.
* returns iterator of toBeFound if found, returns m_nodes.rend() if not */
template<class AnyType>
typename std::vector< Node* >::reverse_iterator isInNetwork(const AnyType& toBeFound)
{
  return std::find_if(m_nodes.rbegin(),
                      m_nodes.rend(),
                      [&toBeFound](const Node * vecEntry) -> bool
    { return *vecEntry == toBeFound; }
                     );
}
```