Cluster Shadow

Brett Viren

July 23, 2024

Contents

1 Introduction **1**

2 Implementation **1**
2.1 Constructing 2
2.2 Using 2

1 Introduction

In the context of the `ClusterShadow` API, the unqualifed term *cluster* is short for *geometric cluster* (GC). A set of GCs is represented by a *blob connectivity graph* with vertices representing *blobs* and edges representing some type of *connection*. A connection is usually based on spatial nearness between the blob pair as determined by some metric which is defined over the longitudinal and transverse drift dimensions. The subgraph consisting of just b-b edges from an `ICluster` graph produced by the `BlobClustering` component in the `img` sub-package is one example of a GC.

A *blob shadow* (BS) is an overlap between two blobs in a given tomographic view (wire plane). The set of blob shadows for a set of blobs and over all views is represented as a *blob shadow graph*. See the presentation on Blob Shadows for more info on blob shadows.

A *cluster shadow* (CS) then represents the combination of these two relationships. A CS is formed from a b-b GC graph and a b-b BS graph and represented by a g-g graph ("g" for "geometric"). A vertex in the CS graph represents one connected component of the b-nodes from the original GC graph. The vertex object holds summary information about the cluster in the form of per-view "coverage" of the 2D time vs channel/wire index space. The g-g edges of a CS graph represent that the two clusters have some non-zero shadow (via their blobs). Each edge carries the `WirePlaneId` from which a view index may be derived. Note, a cluster may posses a *self-shadow* and this is represented by a loop/buckle edge connecting its vertex to itself.

2 Implementation

The main method of the `ClusterShadow` API produces a CS graph and a connected component property map. The vertex descriptor of the CS graph is merely the component index. Edges provide measures of the shadow.

2.1 Constructing

```
#include "WireCellAux/ClusterShadow.h"

using namespace WireCell;
using namespace WireCell::Aux;

cluster_graph_t& cgraph = ...;
BlobShadow::graph_t bsgraph = BlobShadow::shadow(cgraph, 'w'); // or 'c'

ClusterShadow::blob_cluster_map_t clusters;
auto cs_graph = ClusterShadow::shadow(cgraph, bsgraph, clusters);
```

If you don't care about the blob content of clusters you may call `ClusterShadow::shadow(csgraph,bsgraph)`.

2.2 Using

Each CS graph vertex has one `Rectangles` object for each view. You may iterate on the edges of the CS graph, retrieve the vertex object for each edge endpoint and then perform pair-wise set-theoretic operations on the `Rectangles` object from each.

Examples and/or pointers to tests t.b.d.