

The Boost Property Maps

Martin, Mads & Kim

Generic Programming and Library Development

2nd of June 2006



Definition

- Property Maps came from Boost Graph Library (BGL)
- Graph algorithms should not have to deal with implementation details of properties.
- Separate lifetime of objects and external attributes (e.g. Color in BFS)
- Hide underlying data-structure using concepts (Generic access)



Property Map Concepts

- Readable `get()`
- Writable `put()`
- ReadWrite `get()` & `put()`
- Lvalue `get()` & operator `[]`



Types of Property Maps

- Identity
- Iterator
- Associative
- Const Associative
- Vector



Example

```
template <class Edge, class Graph, class WeightPropertyMap,
          class DistancePropertyMap>
bool relax(Edge e, const Graph& g, WeightPropertyMap weight,
           DistancePropertyMap distance)
{
    typedef typename graph_traits<Graph>::vertex_descriptor Vertex;
    Vertex u = source(e,g), v = target(e,g);
    if ( get(distance, u) + get(weight, e) < get(distance, v)) {
        put(distance, v, get(distance, u) + get(weight, e));
        return true;
    } else
        return false;
}
```



Summary

Pros

- Saves memory at runtime
- Saves maintenance-time (classes simpler to read)
- Saves space when saving the graph



Summary

Pros

- Saves memory at runtime
- Saves maintenance-time (classes simpler to read)
- Saves space when saving the graph

Cons

- Binaries a little larger
- Not easy to copy the underlying data-structure.

