Unordered STL Containers

Hüseyin Akcan
huseyin.akcan@ieu.edu.tr

* Slides are partially taken from the Working Draft of the C++ Standard (http://www.open-std.org/JTC1/SC22/WG21/docs/papers/2008/n2691.pdf)
Unordered associative containers

- Unordered associative containers provide an ability for fast retrieval of data based on keys. The worst-case complexity for most operations is linear, but the average case is much faster.

- The library provides four unordered associative containers: `unordered_set`, `unordered_map`, `unordered_multiset`, and `unordered_multimap`.

- Unordered associative containers conform to the requirements for Containers, except that the expressions in the table below are not required to be valid, where `a` and `b` denote values of a type `X`, and `X` is an unordered associative container class:

<table>
<thead>
<tr>
<th>unsupported expressions</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>a == b</code></td>
</tr>
<tr>
<td><code>a != b</code></td>
</tr>
<tr>
<td><code>a &lt; b</code></td>
</tr>
<tr>
<td><code>a &gt; b</code></td>
</tr>
<tr>
<td><code>a &lt;= b</code></td>
</tr>
<tr>
<td><code>a &gt;= b</code></td>
</tr>
</tbody>
</table>
Unordered associative containers

- Each unordered associative container is parameterized by Key, by a function object Hash that acts as a hash function for values of type Key, and by a binary predicate Pred that induces an equivalence relation on values of type Key.

- Additionally, unordered_map and unordered_multimap associate an arbitrary mapped type T with the Key.

- A hash function is a function object that takes a single argument of type Key and returns a value of type std::size_t.

- Two values k1 and k2 of type Key are considered equal if the container’s equality function object returns true when passed those values. If k1 and k2 are equal, the hash function shall return the same value for both.

- unordered_set and unordered_map support unique keys. unordered_multiset and unordered_multimap support equivalent keys. In containers that support equivalent keys, elements with equivalent keys are adjacent to each other.
Unordered associative containers

- The elements of an unordered associative container are organized into buckets. Keys with the same hash code appear in the same bucket.

- The number of buckets is automatically increased as elements are added to an unordered associative container, so that the average number of elements per bucket is kept below a bound.

- Rehashing invalidates iterators, changes ordering between elements, and changes which buckets elements appear in, but does not invalidate pointers or references to elements.

- The insert members shall not affect the validity of references to container elements, but may invalidate all iterators to the container.

- The erase members shall invalidate only iterators and references to the erased elements.
Unordered Set

• An unordered_set is an unordered associative container that supports unique keys (an unordered_set contains at most one of each key value) and in which the elements’ keys are the elements themselves.

template <class Value,
        class Hash = hash<Value>,
        class Pred = std::equal_to<Value>,
        class Alloc = std::allocator<Value> >

class unordered_set

See unordered.cpp for more...
Unordered Map

- An unordered_map is an unordered associative container that supports unique keys (an unordered_map contains at most one of each key value) and that associates values of another type mapped_type with the keys.

```cpp
template <class Key,
         class T,
         class Hash = hash<Key>,
         class Pred = std::equal_to<Key>,
         class Alloc = std::allocator<std::pair<const Key, T> > >

class unordered_map

See unordered_map.cpp for more...
**C++ Code Jam**

- Create map and unordered map containers for int key type
- Insert, and remove elements within these containers and measure the time
- Check if the behaviour matches with the asymptotic expected times.
- Do not forget to optimize your C++ code using -O2 flag!