Math 4997-3 Quiz 10: Due by Tuesday, November 12

Last exercise

Exercises

1. Adding components and actions to the partition-based 1D heat equation (10 credits):
   In this exercise you should understand the new version of the code of the 1D heat equation using components and actions. You can use the example code on Github[1] and there is no need to add the HPX features by your own. Look at the following HPX features

   (a)
   ```
   friend class hpx::serialization::access;

       template <typename Archive>
       void serialize(Archive& ar, const unsigned int version) const {}
   ```

   (b)
   ```
   struct partition_server
     : hpx::components::component_base<partition_server>{};
   ```

   (c)
   ```
   HPX_DEFINE_COMPONENT_DIRECT_ACTION(partition_server, get_data, get_data_action);
   ```

   (d)
   ```
   typedef hpx::components::component<partition_server> partition_server_type;
   HPX_REGISTER_COMPONENT(partition_server_type, partition_server);
   ```

   (e)
   ```
   typedef partition_server::get_data_action get_data_action;
   HPX_REGISTER_ACTION(get_data_action);
   ```

   (f)
   ```
   struct partition : hpx::components::client_base<partition, partition_server>{};
   ```

   (g)
   ```
       partition(hpx::id_type where, std::size_t size, double initial_value)
         : base_type(hpx::new_<partition_server>(where, size, initial_value))
       {}
   ```

   (h)

[1] https://github.com/diehlpkteaching/StencilLocaltoRemote/blob/master/Stencil5.cpp

```
hpx::future<partition_data> get_data() const
{
        return hpx::async(get_data_action(), get_id());
}
```

(i)

```
return dataflow(
        unwrapping(
                [middle](partition_data const& l, partition_data const& m,
                        partition_data const& r)
            {
                        // The new partition_data will be allocated on the same
                // locality as 'middle'.
                return partition(middle.get_id(), heat_part_data(l, m, r));
            }
            ),
            left.get_data(), middle.get_data(), right.get_data());
```

(j)

```
HPX_PLAIN_ACTION(stepper::heat_part, heat_part_action);
```

(k)

```
using hpx::util::placeholders::_1;
using hpx::util::placeholders::_2;
using hpx::util::placeholders::_3;

auto Op = hpx::util::bind(heat_part_action(), hpx::find_here(), _1, _2, _3);
```

and add a comment to the code why we need these features.