

Regent I/O

CS315B Lecture 8

I/O in Parallel Programming

- I/O tends to be an afterthought in parallel programming systems
- Many papers ignore I/O time in reported results!
- But in real life, I/O time is ... time

Regent I/O

- The situation is better with Regent
- Already have the notion
 - There are distinct collections of data
 - regions
 - That can be in different places, have different layouts, etc.
 - And the details are kept abstract
 - Programmer doesn't need to know how data is accessed

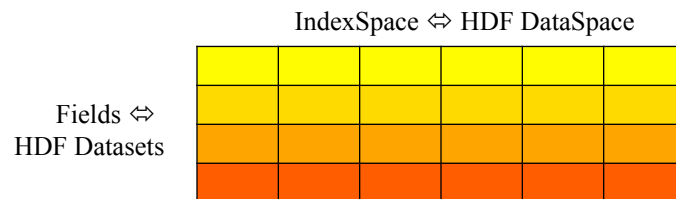
Regent I/O Outline

- Interpret files as regions
 - Integrate I/O into the programming model
- Why?
 - Want to overlap I/O with computation
 - Need to define consistency semantics
- Bottom line
 - I/O is (almost) like any other data movement

Attach Operation

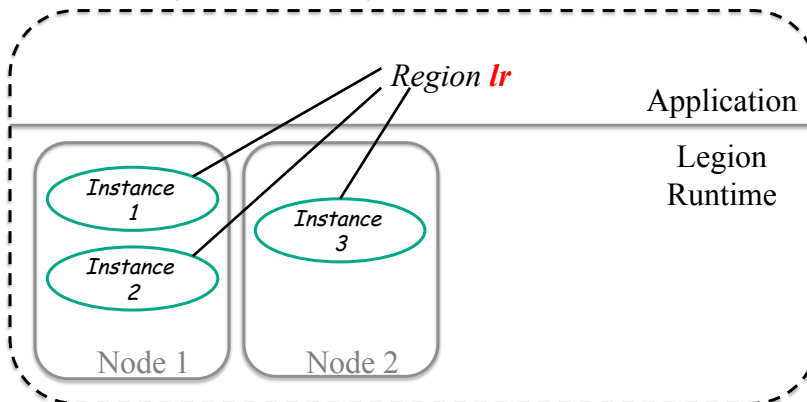
- Attach external resource to a region
 - Normal files, formatted files (HDF5), ...

```
PhysicalRegion attach_hdf(
    const char *filename,
    LogicalRegion lr,
    const std::map<FieldID,const char*> &fieldmap,
    AccessMode mode);
```



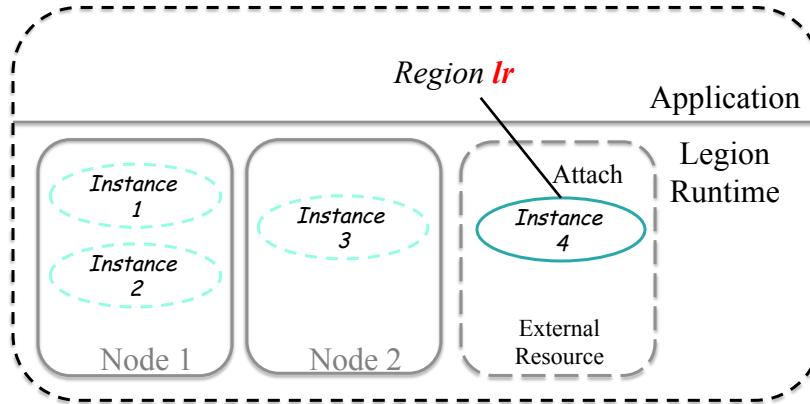
Attach Operation

- **Semantics**
 - Invalidate existing physical instance of *lr*
 - Maps *lr* to a new physical instance that represents external data (no external I/O)



Attach Operation

- **Semantics**
 - Invalidate existing physical instance of *lr*
 - Maps *lr* to a new physical instance that represents external data (no external I/O)



Digression: Task Coherence

Privileges

- Reads
- Reads/Writes
- Reduces (with operator)

Coherence

- Exclusive
- Atomic
- Simultaneous
- Relaxed

- Coherence declarations are wrt *sibling* tasks

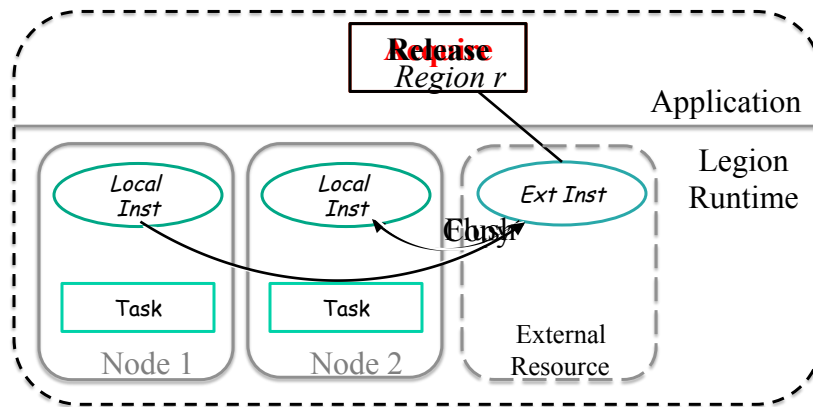
Attach Operation

- Attached region accessed using *simultaneous coherence*
 - Different tasks access the region simultaneously
 - Requires that all tasks must use the *only valid* physical instance
- *Copy restriction*
 - Simultaneous coherence implies tasks cannot create local copies
 - May result in inefficient memory accesses

Acquire/Release

- For regions with simultaneous coherence
- Acquire removes the copy restriction
 - Can create copies in any memory
 - Up to application to know this is OK!
- Release restores the copy restriction
 - Invalidates all existing local copies
 - Flushes dirty data back to the file

Acquire/Release Example

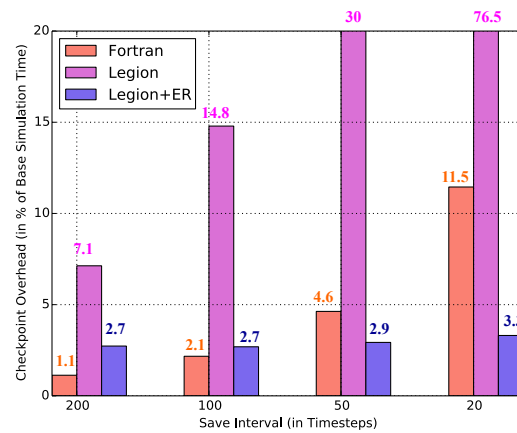


Opaque Data Sources

- Can also attach to sources that are other programs
 - E.g., read/write in-memory data structures from another process
- Done through a serialization/deserialization interface
 - Attach specifies the ser/des routines

S3D I/O Example

- A production combustion simulation
- Checkpoint after fixed # of time steps



Regent I/O Example

I/O Summary

- Definitely a useful feature!
- And less mature than other features
 - But simple cases will work fine
- Let us know if you need/want to use I/O