Transferring Hardware-related Information in sys-sage Across Processes and HPC Nodes

Finn Romanessen

School of Computation, Information, and Technology

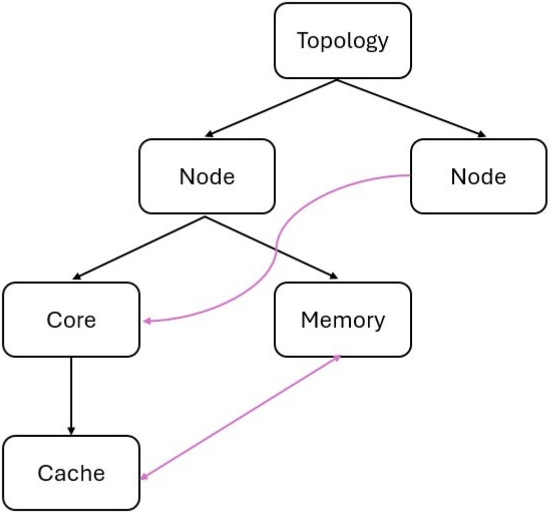
Technical University of Munich

29.05.2024

Sys-Sage

- Library to store and manage hardware topology information
- Designed to be highly dynamic and customizable
- Topologies consist of a component tree and DataPath graph
- Arbitrary data can be attached to components and DataPaths

Sys-Sage



Motivation

- Share sys-sage topologies between processes and nodes
- Hardware topologies within a HPC system are often very similar
- Share current information about the hardware state

Approach

- Exporting process copies topology into shared memory
- All parts are arranged sequentially in the memory region
- Any process can import the topology by recreating it in local memory

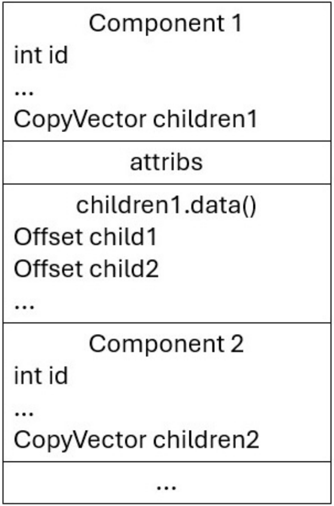
Shared Memory

- Shared memory regions created using memory-mapped files
- Virtual memory addresses different in each process
- Offset based pointers used within shared memory block

Components

- Represent specific parts of the hardware
- Vectors have to be transformed to offset based implementation `CopyVector`
- Pointers to children are replaced with offsets

Components



Component 1

int id

...

CopyVector children1

attribs

children1.data()

Offset child1

Offset child2

...

Component 2

int id

...

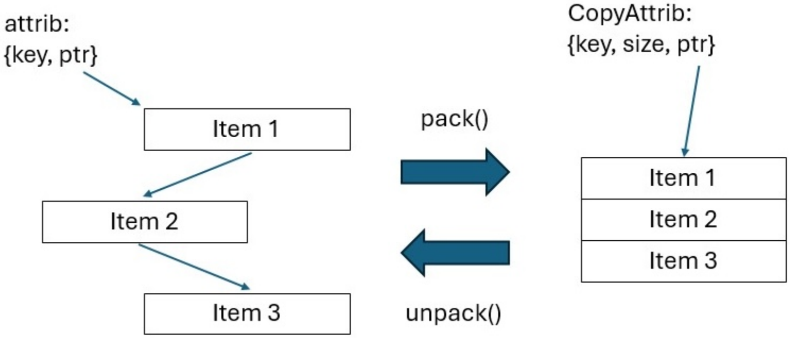
CopyVector children2

...

Attribs

- Store arbitrary data in key-value map
- Data needs to be contiguous
- Map items are copied into sequential memory block
- Size and data supplied using `pack()` function

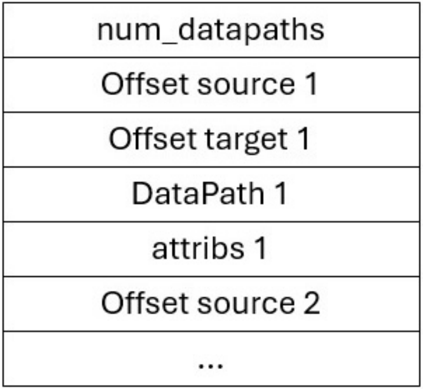
Attribs


attrib:
{key, ptr}
Item 1
Item 2
Item 3
pack()
unpack()
CopyAttrib:
{key, size, ptr}
Item 1
Item 2
Item 3

DataPaths

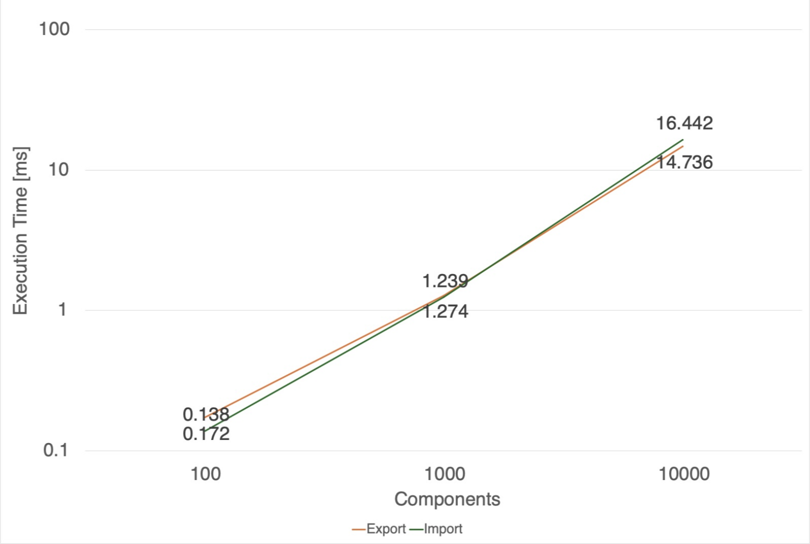
- Represent relationships between components
- DataPaths are only exported if both components exported
- Offsets of source and target component in shared memory stored

DataPaths



num_datapaths
Offset source 1
Offset target 1
DataPath 1
attribs 1
Offset source 2
...

Performance



Future Work

- Functionality to update existing shared topologies
- Make topologies usable within shared memory
- Changes possible from all processes

Conclusion

- Share sys-sage topologies between processes and nodes
- Implementation uses memory-mapped files
- Topologies are deconstructed and reconstructed by importing process