Array, Domain, & Domain Map Improvements

Chapel Team, Cray Inc.
Chapel version 1.17
April 5, 2018

Safe Harbor Statement

This presentation may contain forward-looking statements that are based on our current expectations. Forward looking statements may include statements about our financial guidance and expected operating results, our opportunities and future potential, our product development and new product introduction plans, our ability to expand and penetrate our addressable markets and other statements that are not historical facts. These statements are only predictions and actual results may materially vary from those projected. Please refer to Cray's documents filed with the SEC from time to time concerning factors that could affect the Company and these forward-looking statements.

Outline

- Forwarding on Domain Maps
- Bulk Transfer Interface Redesign
- Subtype Queries on Distributions
- Other Array, Domain, Domain Map Improvements

Forwarding on Domain Maps

Forwarding on Domain Maps: Background

- **Some domain maps benefit from custom methods**
  - E.g., StencilDist.updateFluff() performs ghost cell exchanges

- **Exposing custom methods required working with internals**
  - Declaring a wrapper method on internal "_array" type

```
proc StencilArr.updateFluff() { ... }

proc _array.updateFluff() where isStencil(this._value) {
  this._value.updateFluff();
}
```

  - Or calling method on undocumented "_value" field

```
myArray._value.methodWithoutWrapper();
```

Forwarding on Domain Maps: Effort and Impact

This Effort:

- Implemented forwarding of methods on domain maps
  - Can now do away with wrapper methods

    ```
    proc StencilArr.updateFluff() { ... }
    ```

  - No longer need to access "_value" for wrapper-less methods

    ```
    myArray.methodWithoutWrapper();
    ```

Impact:

- Easier to write custom methods on domain maps
- Simplified and improved existing code
  - Removed existing _array wrappers around custom methods
  - Removed more uses of "_value" from tests
- Fulfilled “custom interface” concept from original domain map paper

Bulk Transfer Interface Redesign

Bulk Transfer Redesign: Background

- **Bulk transfer interface allows for optimized assignment**
  - Lets domain map authors perform assignments themselves
  - Allows for less overhead based on knowledge of memory layout

- **Original interface was overly complex**
  - Required an excessive number of methods to implement
  - Too much information baked into method names
    - "doiCanBulkTransfer"
    - "doiCanBulkTransferStride"
    - "doiBulkTransferToDR"

Bulk Transfer Redesign: Background (cont.)

- **Problem: How to pick between domain maps' methods?**
  - 'Dest.from(Source)', or 'Source.to(Dest)' ?
  - Problem for transfers between standard and package domain maps

```
// Block.to(Package) - Suboptimal transfer using local arrays
// Package.from(Block) - Optimal transfer using GETs/PUTs
packageArr = standardBlockArr;
```

- **Standard dists can't know about custom dists**
  - Custom dists do know about standard dists

Bulk Transfer Redesign: This Effort

- **Designed and implemented a new interface**
  - Two kinds of information encoded in method name:
    - Direction (To/From) and preferred method (Known/Any)
  - Support determined by attempting to resolve methods

```
proc doiBulkTransfer[To|From][Known|Any](myDom:domain,
                                          otherDMap,
                                          otherDom:domain) : bool;
```

- **Simpler interface that supports preferred methods**
  - 'Known' methods are attempted before 'Any' methods

Package = Block	Resolved?	Called?
Block.toKnown(Pkg)	False	False
Pkg.fromKnown(Block)	True	True
Block.toAny(Pkg)	*True*	*False*

Bulk Transfer Redesign: Impact and Next Steps

Impact: Less work for domain map authors

- Interface support determined by reflection in internal modules
- Return 'false' if the transfer cannot be completed
  - Caller is then responsible for completing the transfer

Next Steps:

- Re-examine bulk transfer of standard distributions
  - Contributed pre-2014, many things have since changed
- Provide helper functions for more advanced usage
  - Wrap reflection-testing of interface support

Subtype Queries on Distributions

Subtype Queries on Distributions

Background: Querying domain's distribution type was messy

- Users can access domain's distribution through "`domain.dist`"
- Querying type involved using undocumented internals

```
proc foo(D : domain) where D.dist._value: Block { ... }
```

This Effort: We now support subtype queries on "domain.dist"

```
proc foo(D : domain) where D.dist: Block { ... }
```

Impact: Can eliminate more uses of "_value" in tests/modules

Next Steps:

- Continue removing uses of "_value" from tests
- Retire special interpretation of ':' in where-clauses

Other Array, Domain, Domain Map Improvements

Other Array, Domain, Domain Map Changes

- Sparse CS domains can now have a sparse parent domain
- Support for querying the stridability of sparse domains
- Support for strided Block-sparse domains and arrays

Legal Disclaimer

CRAY CHAPEL

CRAY

THE SUPERCOMPUTER COMPANY