Library 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

- **Standard Module Improvements**
  - Math
  - Path
  - DateTime
- **Package Module Improvements**
  - Crypto
  - LinearAlgebra
- **Other Library Improvements**

Standard Module Improvements

Math

Math

Background: Bessel functions were not supported

- Bessel functions are available in C standard library
- Requested by several Chapel users

This Effort: Implement Bessel functions available in standard C

- Includes *nth* order first kind (*j*) and second kind (*y*) Bessel functions
- Supports real(64) and real(32)
- Contributed by Nimit Bhardwaj

Status: Bessel functions now supported in Math module

```
j0(100);      // First kind, 0th order
j1(100);      // First kind, 1st order
jn(100, 9);   // First kind, nth order
y0(10);       // Second kind, 0th order
y1(10);       // Second kind, 1st order
yn(10, 5);    // Second kind, nth order
```

Path

Path

Background:

- Intended to contain 15 routines modeled after Python's os.path library
  - Proposed during 1.11 release cycle, prior to good string support
- As of last release, 9 routines left unimplemented

This Effort:

- Added support for joinPath(), isAbsPath(), and commonPath()
- Contributed by Surya Priy, Unnati Parekh, Prithvi Patel

Status:

- Path module now only missing 6 routines
- Windows paths not supported
  - (Chapel does not support Windows natively at present)

Next steps:

- Implement remaining features

DateTime

DateTime

Background: DateTime objects leaked timezones, lacked IO

- If the timezone field was used, it was not deleted
- Reading/writing DateTime objects used default record IO

This Effort: Fixed leak and improved I/O format

- Convert the timezone field into a Shared object
  - It is now reclaimed automatically
- Added IO routines to the DateTime types
  - Dates and times are read/written in ISO format

```
2018-03-29T14:41:37.747151
```

Impact:

- Timezone memory is handled automatically
- Output format for dates and times is significantly improved

Package Module Improvements

Crypto

Crypto

Background: The Blowfish Cipher is available in OpenSSL

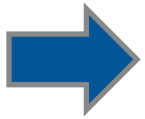
- But there was no Chapel interface to it

This Effort: Add Blowfish class to the Crypto package

- New class can encrypt/decrypt data using the Blowfish algorithm
- Contributed by Sarthak Munshi

Impact: Blowfish cipher can be used from Chapel programs

```
use Crypto;
const msg = new CryptoBuffer("secret”),
      bf = new Blowfish("cbc");
const ct = bf.encrypt(msg, key, iv),
      pt = bf.decrypt(ct, key, iv);
writeln("encrypted: ", toString(ct));
writeln("decrypted: ", toString(pt));
```



```
encrypted: □□□[2□0□
decrypted: secret
```

* Definitions of key, iv, and toString() are omitted for space

Next steps: Provide more functionality from OpenSSL

- ECC, DES, Twofish, additional key derivation functions, …

LinearAlgebra

LinearAlgebra: Background

- **Provides a high-level interface for linear algebra**
  - Design influenced by NumPy and MatLab
  - Provides helper functions for creating matrices and vectors as arrays
  - Supports many linear algebra operations on matrices and vectors
- **Implementations use both Chapel and BLAS library**
  - e.g. matrix-matrix multiplication uses BLAS
- **'Sparse' submodule supports sparse linear algebra**
  - Supports a subset of the features available in Linear Algebra module

LinearAlgebra: This Effort

- **Improved Performance**
  - Faster sparse matrix-matrix multiplication implemented
- **New Features**
  - Added Cholesky decomposition and eigensolvers
    - Using LAPACK
  - Added Kronecker product
    - Contributed by Nimit Bhardwaj
  - Added support for sparse identity matrices
    - Contributed by Nikhil Padmanabahn
- **Interface Changes**
  - Switched to 1-based indices as default instead of 0-based indices
  - New dependency on LAPACK library in addition to BLAS

LinearAlgebra: Impact

- **LinearAlgebra library contains more features:**

```
use LinearAlgebra, Random;

var A = Matrix(10, 10);
fillRandom(A);
// Eigenvalues
var eig = eigvals(A);
// Eigenvalues and eigenvectors
var (eig, right) = eigvals(A, right=true);
// Eigenvalues, eigenvectors, and left eigenvectors
var (eig, left, right) = eigvals(A, left=true, right=true);
// Obtain L & U with Cholesky decomposition
var L = cholesky(A);
var U = cholesky(A, lower=false);
// Compute Kronecker Product
var kronProduct = kron(A, A);
```

LinearAlgebra: Impact

- **LinearAlgebra module consistent with Chapel indices**

```
// Fewer surprises in user interface
var array = [1,2,3];
var vector = Vector(1,2,3);
assert(array.domain == vector.domain); // {1..3}
```

- **Sparse matrix-matrix multiplication is faster**
  - New algorithm performs better as density approaches 0
  - Measured performance of squaring NxN sparse matrix:

```
var AA = dot(A, A); // A is 2D CSR array
```

nonzero elements	density	1.16 (seconds)	1.17 (seconds)
1e4	0.01%	**1.9e-2**	**4.5e-5**
1e7	1e-8%	**38.75**	**0.02**

LinearAlgebra: Next Steps

- **Progress tracked in issue #5753**
- **Performance Improvements**
  - Enable non-sorted indices in CS layout for sparse matrix multiplication
  - Parallelize sparse matrix multiplication
- **Support other computation models**
  - Distributed array support
  - GPU support, using CuBLAS/CuLAPACK
- **More dense and sparse features**
- **Address promotion flattening issue, #5958**

Other Library Improvements

Other Library Improvements

- added channel.advancePastByte() to read until particular byte
- added versions of channel.mark(), commit(), revert() for locking==false
- array push* methods now use in intents and now support Owned
- added string.size overload for string.length
- made the Buffers module into a package module
- made UtilReplicatedVar module into package module ReplicatedVar
- added a new scalable barrier across all locales
- added binaryInsertionSort to the Sort module
- updated several modules to use error handling rather than try! / halt()
- added --debugTomlReader flag for TOML.TomlReader submodule

Legal Disclaimer

Information in this document is provided in connection with Cray Inc. products. No license, express or implied, to any intellectual property rights is granted by this document.

Cray Inc. may make changes to specifications and product descriptions at any time, without notice.

All products, dates and figures specified are preliminary based on current expectations, and are subject to change without notice.

Cray hardware and software products may contain design defects or errors known as errata, which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Cray uses codenames internally to identify products that are in development and not yet publically announced for release. Customers and other third parties are not authorized by Cray Inc. to use codenames in advertising, promotion or marketing and any use of Cray Inc. internal codenames is at the sole risk of the user.

Performance tests and ratings are measured using specific systems and/or components and reflect the approximate performance of Cray Inc. products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance.

The following are trademarks of Cray Inc. and are registered in the United States and other countries: CRAY and design, SONEXION, and URIKA. The following are trademarks of Cray Inc.: ACE, APPRENTICE2, CHAPEL, CLUSTER CONNECT, CRAYPAT, CRAYPORT, ECOPHLEX, LIBSCI, NODEKARE, THREADSTORM. The following system family marks, and associated model number marks, are trademarks of Cray Inc.: CS, CX, XC, XE, XK, XMT, and XT. The registered trademark LINUX is used pursuant to a sublicense from LMI, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis. Other trademarks used in this document are the property of their respective owners.

CRAY

THE SUPERCOMPUTER COMPANY