SciDAC Software
C++ Language Interface

Robert Edwards

Software Coordinating Committee

February 22, 2003

Software Hierarchy

Level 3: Full inverters,etc

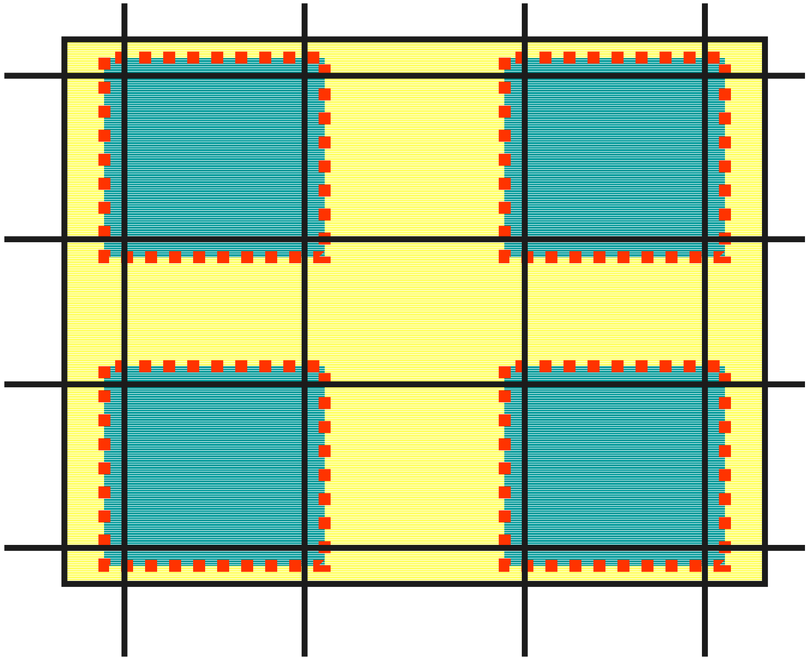
Level 2: QDP data parallel (C,C++)

Level 1: QLA (single node linear algebra) (C)

QMP (node-to-node messaging)

Data-Parallel Programming Model

Data layout over processors



- Lattice-wide operations: C(x) = A(x)*B(x)
- Like Fortran 90 but more complex types
- No compiler here - implement using an API
- API hides communications and all site looping

Data Parallel QDP/C++ API

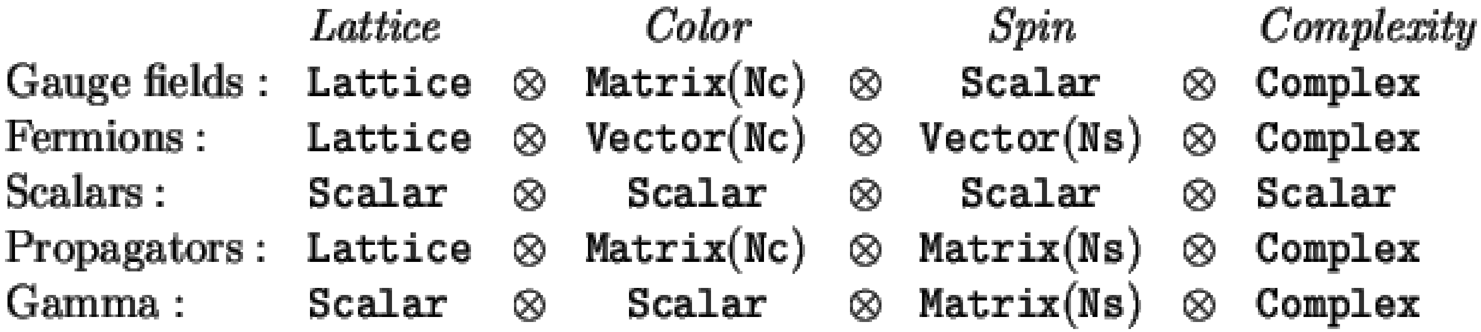
- Hides architecture and layout
- Operates on lattice fields across sites
- Linear algebra tailored for QCD
- Shifts and permutation maps across sites
- Reductions
- Subsets

Data Types

- Fields have various types (indices):

  Color: $U^{ij}(x)$, Spin: $\Gamma_{\alpha\beta}$, $\psi^{i}_{\alpha}(x)$, $Q^{ij}_{\alpha\beta}(x)$

- Tensor product of indices forms type



	Lattice		*Color*		*Spin*		*Complexity*
Gauge fields :	Lattice	⊗	Matrix(Nc)	⊗	Scalar	⊗	Complex
Fermions :	Lattice	⊗	Vector(Nc)	⊗	Vector(Ns)	⊗	Complex
Scalars :	Scalar	⊗	Scalar	⊗	Scalar	⊗	Scalar
Propagators :	Lattice	⊗	Matrix(Nc)	⊗	Matrix(Ns)	⊗	Complex
Gamma :	Scalar	⊗	Scalar	⊗	Matrix(Ns)	⊗	Complex

- Some types
  - Real, Complex, ColorMatrix, LatticeReal, LatticeFermion
- Can add new subtypes to support other representations (e.g., supersymmetry)

Data-parallel Operations

- *Unary and binary:*

-a; a-b; …

- *Unary functions:*

adj(a), cos(a), sin(a), …

- *Random numbers:*

// platform independent

random(a), gaussian(a)

- *Comparisons (booleans)*

a <= b, …

- *Broadcasts:*

a = 0, …

- *Reductions:*

sum(a), …

Linear Algebra example

- Can create expressions

$$c_\alpha^{\ i}(r) = U^{ij}(r)\, b_\alpha^{\ j}(r) + 2\, d_\alpha^i(r) \qquad \forall r$$

```
LatticeColorMatrix  u;
LatticeDiracFermion  b, c, d;
c = u * b  +  2 * d;
```

Shift and Map examples

- Maps provide communications (mapping) of sites to sites
- Varieties (single,array, bidirectional). Can be optimized for hardware platform

```
Map knight(knight_func);
```

// where **knight_func(x)** maps x_μ to $[x_0,x_1+1,x_2+2,x_3]$

```
LatticeReal a,b,c;
c = a * knight(b);
```

Subset and Reduction examples

- Subset : a collection of sites
- Set : a collection of subsets

$$\forall_t \quad p(t) = \sum_{\vec{r}} a^*(\vec{r},t) \cdot b(\vec{r},t)$$

```
Set   ts(timeslice);                  // where  t = timeslice(r)
LatticeComplex   a,b;
multi1d<Complex>   p(nt);  // array of Complex
p = sum(conj(a)*b, ts); // inner product within each subset
```

QDP/C++ Implementation

- Built on QMP/C and some QLA/C
- Layout flexibility
- Communications overlapped with computations in an expression
- Site-wide operations use templates
- Built heavily on *Expression Templates* (PETE from LANL)

Linear Algebra Implementation

```
// Lattice operation
A = adj(B) + 2 * C;

// Lattice temporaries
t1 = 2 * C;
t2 = adj(B);
t3 = t2 + t1;
A = t3;

// Merged Lattice loop
for (i = ... ; ... ; ...) {
    A[i] = adj(B[i]) + 2 * C[i];
}
```

- Naïve ops involve lattice temps – inefficient
- Eliminate lattice temps -PETE
- Allows further combining of operations (*adj(x)*y*)

Binary File Formats

- Coordination with ILDG: file formats, metadata, middleware
- Example: NERSC gauge format: metadata+binary

```
BEGIN
<name1> = <value1>
<name2> = <value2>
END
<BINARY>…
```

- Metadata – data describing data; e.g., physics params, QDP type…
- Proposed I/O standard uses XML for metadata format
- File formats
  - Files mixed mode – ascii+binary
  - Using DIME (similar to e-mail MIME) to package
  - Use BinX (Edinburgh) to describe binary
- Benefits: extensibility, file archive database – leverage off market

Data Hierarchy

- Project built from
  datasets (e.g. gauge fields and propagators)
- Dataset built from
  files (e.g. gauge fields)
- File built from
  records (e.g. eigenvectors)
- Record = QDP field or metadata

- Separate metadata used by other programs

Status

- Release and documentation

http://www.lqcd.org

- QDP/C++ single node and parallel version working
- Performance improvements/testing underway
- Porting & development efforts of physics codes over QDP C++ on-going

- QIO design/development underway
  - Simple I/O API for reading/writing files
  - Coordination with ILDG - UKQCD, CPPACS, more