Array and Grid Databases

David Maier

Computer Science Department
Maseeh College of Computer Science and Engineering
Portland State University

Lots of Science Data is Arrays

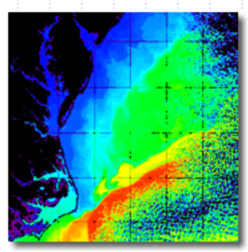
- Remote imaging (up and down)
- Tomographic reconstructions
- Computational simulation outputs
- In-situ sensing
- Next-Generation Sequencing

Implicit Information in the Structure

- Logical organization of an array can indicate order, adjacency, correlation
- However, meaning is different for different arrays

Example: Image Data

- Might have two dimensions corresponding to latitude and longitude
  - Neighboring entries adjoin in space
  - Lose information if you rearrange rows or columns
  - Operations – smoothing, edge detection, object extraction



NOAA CoastWatch

Example: Bi-gram Frequencies

- Entries are bi-gram frequencies
  - A(i, j) = number of times word i precedes word j in some corpus of text
  - Adjacency doesn't mean much: OK to permute rows and columns (in the same way)
  - Operations: row or column correlations; matrix multiplication

$$P = \begin{bmatrix} 5 & 0 & 8 & 0 & 0 & 7 & 0 & 4 & 0 \\ 4 & 0 & 2 & 9 & 0 & 5 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 8 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5 & 3 \\ 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 \\ 0 & 5 & 0 & 6 & 0 & 9 & 4 & 0 & 7 \\ 0 & 3 & 0 & 5 & 0 & 0 & 8 & 0 & 9 \end{bmatrix}$$

Example: Sequencing Data

- Have 2-D array, indexed by sample ID and DNA base position
  - Array element is a read call (A C G T N) and a confidence
  - Sample order could be shuffled, but not order of reads
  - Operations: aggregate (across base position or whole array); "array induction" – count values for x in every $b_1b_2xb_3b_4$, indexed by (b_1, b_2, b_3, b_4)

Support for Array Storage

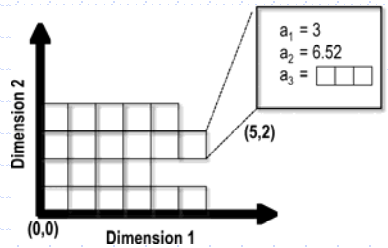
- netCDF, HDF, other interchange formats
- Rasdaman – rasters over DBMS
- SQL 1-D arrays
- RAM Layer on MonetDB
- SciDB – relatively new effort

Variations in Array Models

- Scalar or complex elements
  - Records
  - Nested arrays
- “Ragged” boundaries
- Special values
- Non-integer dimensions
- Updates vs. versions

SciDB Data Model

- Nested multi-dimensional arrays
  - Cells can be tuples or other arrays
  - Can have non-integer dimensions
- Additional "History" dimension on updatable arrays
- Ragged arrays allow each row or column to have a different legnth
- Support for multiple flavors of "null"
  - Array cells can be 'EMPTY'
  - User-definable treatment of special values



SciDB DDL

```
CREATE ARRAY Test_Array
      < A: integer NULLS,
        B: double,
        C: USER_DEFINED_TYPE >
       [I=0:99999,1000,10, J=0:99999,1000,10 ]
       PARTITION OVER ( Node1, Node2, Node3 )
       USING block_cyclic();
```

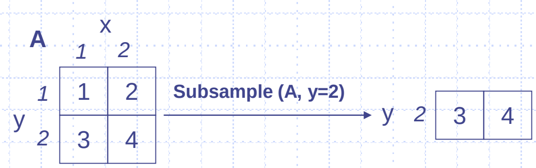
attribute names	dimension names	chunk size	overlap
A, B, C	I, J	1000	10

Operations on Arrays

- Need to preserve array structure
- Purely structural ops
- Content-based ops
- Linear algebra (if array represents a matrix)

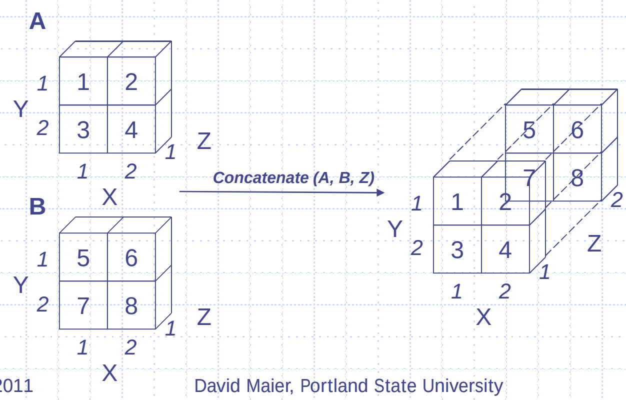
Subsample

Restrict an array by index ranges



Concatenate

Append arrays along specified dimension



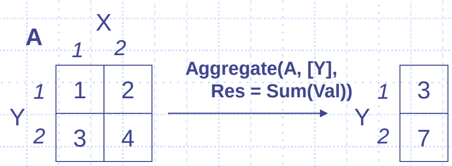
Filter

Apply predicate to array elements

Keeps array shape: Inserts empty elements

Aggregate

Reduce across one or more dimensions



Languages for Arrays

Many proposals, old and new

- APL: Falkoff, Iverson
- AML: Marathe, Salem
- NewS, R, Matlab
- rasql: Baumann
- SciQL: Kersten, Zhang, Ivanova, Nes

Array Comprehensions

Like MArray in rasql, Build in SciDB docs

- Supply a spatial domain S
  e.g. [I=0:999, J=0:4999]
- Have an expression g:S → ET (element type)

```
BUILD(S,(i,j) →
      <r=A[i,j+100].va,
   s=B[j].ba*5.0>
         )
```


SciDB: Array Query Language (AQL)

```
SELECT Geo-Mean ( T.B )
FROM Test_Array T
WHERE
    T.I BETWEEN :C1 AND :C2
AND T.J BETWEEN :C3 AND :C4
AND T.A = 10
GROUP BY T.I;
```

User-defined aggregate on an attribute B in array T

Subsample

Filter

Group-by

SciDB: Array Functional Language (AFL)

Lexical syntax for the algebra

```
A<va:int>[I=0:999,J=0:4999]
B<vb:int>[J=0:4999,K=0:2499]
aggregate(
  apply(
    sjoin(A,B,J=J),
    res=A.va*B.vb
        ),
  [I,K],vr=sum(res)
         )
```


Physical Representation

- Array of records → record of arrays

```
Array<va=int, fa=float>[I=0:99, J=0:499] →
<va=Array<int>[I=0:99, J=0:499],
 fa=Array<float>[I=0:99, J=0:499]>
```

- Nested array → merge dimensions

```
Array<va=int, fa=Array<r=float>[K=0:9]>
[I=0:99, J=0:499] →
<va=Array<int>[I=0:99, J=0:499],
 fa=Array<Array<r=float>[K=0:9]>[I=0:99, J=0:499]>
→
<va=Array<int>[I=0:99, J=0:499],
 fa=Array<float>[K=0:9, I=0:99, J=0:499]>
```

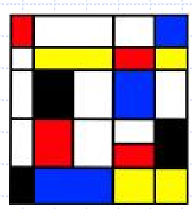
Physical Representation 2

- Non-integer indices → mapping array

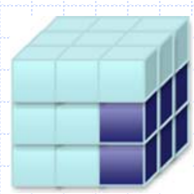
```
Array A<a1: int32, a2: double>
  [I(string)=100, J(double)=1000] →
Array BasicA<a1: int32, a2: double>
  [BI=0:99, BJ=0:999]
IMap<I=string>[BI=0:99]
JMap<J=double>[BJ=0:999]

A = Sjoin(BasicA, IMap, JMap,
          A.BI=IMap.BI, A.BJ=JMap.BJ)
```


Partitioning

- Rasdaman tiling of rasters
  - Many options, needn't be uniform
  - Can isolate regions of interest

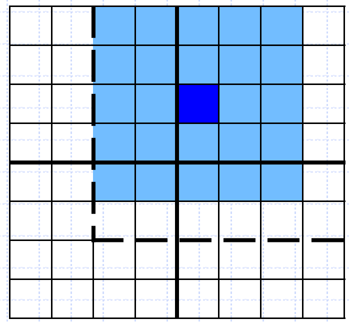


- SciDB chunking
  - Regular divisions along dimensions
  - Distribution pattern, e.g., block cyclic



Issue: Neighborhood Ops

- Doing a 5 x 5 stenciled average over a chunk requires up to 8 adjoining chunks
- Can specify an overlap (e.g., 2 elements)



Issue: Logical vs. Physical Size

- Dividing an array evenly in logical space can give unequal physical chunks after compression
- Equal physical chunks are easier for I/O, but makes it hard to align 2 arrays
- SciDB: Equal-sized logical chunks, but combine multiple physical chunks into an I/O segment

Versions

- Conceptually, updates in SciDB are additions along a History dimension
- Implemented as reverse deltas at a chunk granularity

Application Programming Interface (API)

- Can do embedded queries in general-purpose programming languages, e.g., C++, Python
- Would like a more transparent interface from analysis environments such as R
  - Dynamically accumulate expressions (à la Ohkawa, RIOT)
  - Evaluate intelligently on demand, e.g., minimize data movement

Current R Support for Large Data Not Very Transparent

- Native R

```
result <- sum(array);
```

- Chunked access to netCDF

```
chunk.size <- 1000;
num.chunks <- ceiling(total.size/chunk.size);
for(i in num.chunks) {
  array.part <- get.var.ncdf(file.path,chunk.size);
   result <- result + sum(array.part);
   remove(array.part); gc(); }
```

- Call out to RDBMS

```
result <- sqlQuery(DBconn, "select sum(value)
                           from array_table");
```

- Specialized Libraries

Accumulate Expressions

Want to have as large of scope as possible before evaluating

```
A <- B + C;
…
D <- A[1:10];
…
print(A);
```

Accumulate to

```
print((B + C)[1:10]);
```

Minimize Data Transfer

- Reductive Transforms: less data to move (**bold** = op or arg in SciDB)

```
print((B + C)[1:10]); →
print((B + τ(C))[1:10]); →
print((B[1:10] + τ(C[1:10]));
```

- Consolidating Transforms: fewer transfers

```
print((B + C) + D); →
print((B + τ(C)) + τ(D)); →
print(B+ τ(C + D)); →
```


Additional Aspects

- Needs to be cost based

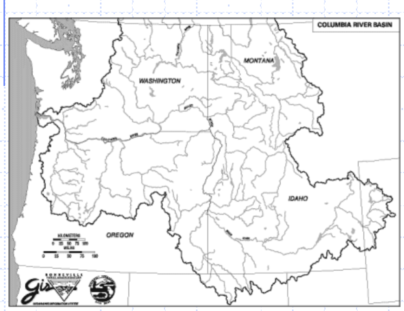
```
print((B%*%C)%*%D); →
print(B%*%(C%*%D));
B[20,500], C[500,1], D[1,300]
```

- Other considerations
  - Availability of operators in each engine
  - Data representation and distribution
  - Estimate execution time

Data on Grids

- Simulation data often bound to a grid (or mesh)
  Discrete model of continuous space
- Regular grids resemble arrays
  - gridded data often stored in arrays
  - but grids have richer structure
- Many grids not regular (*unstructured*)

Columbia River Estuary

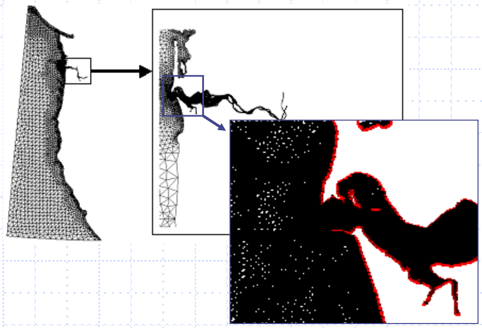


Hydrodynamic Models

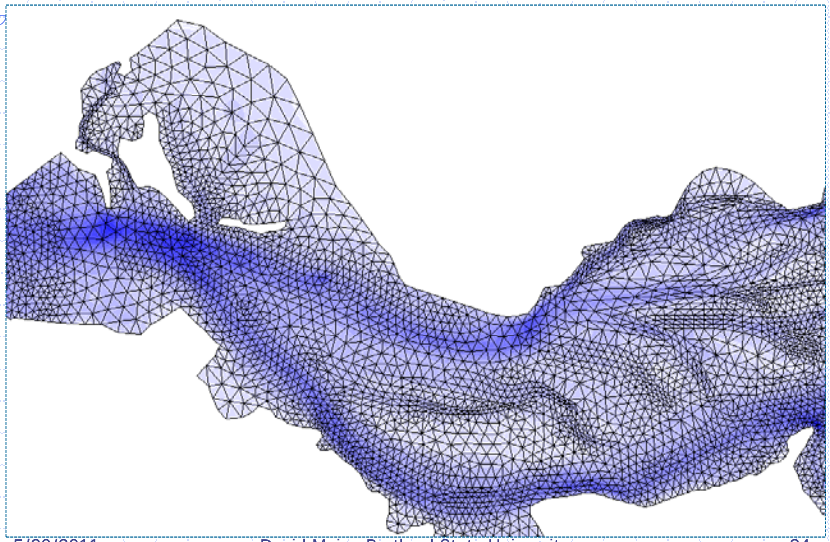
SELFE

- Finite-volume model on unstructured grid
- Large outputs

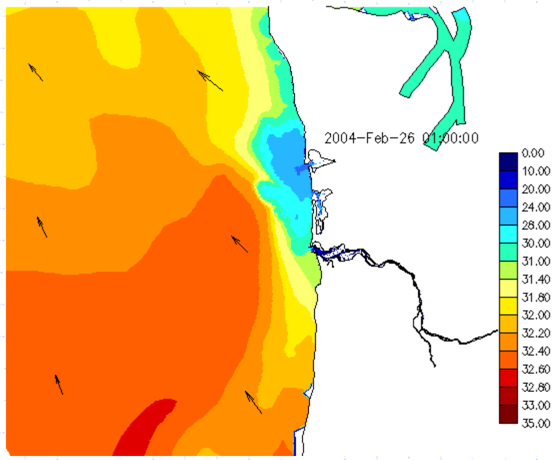
file	size
hvel	2.5 GB
salt vert temp	1.3 GB each



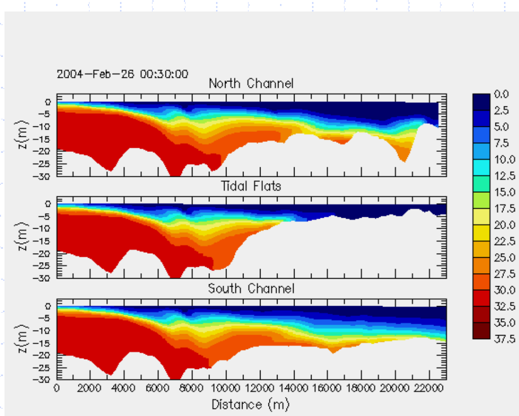
Mesh Around Channels



Isoline Data Product



Transect Data Products

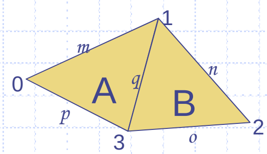


Would Like an Algebra for Grid Data Products

- A grid can have components at multiple dimensions
- A 2-D grid can have nodes, edges and faces (0-cells, 1-cells, 2-cells)
- Operations need to be cognizant of grid

Grid Topology

- Grid Topology
  - A collection of *cells* of various dimensions,
  - implicit or explicit incidence relationships



2-Cells = {A,B}
1-Cells = {m,n,o,p}
0-Cells = {0,1,2,3}

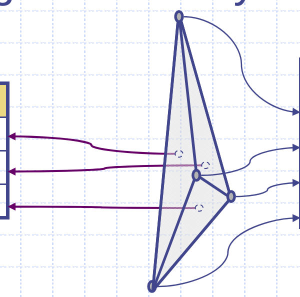
2-Cells	0-Cells
A	0
A	1
A	3
B	1
B	2
B	3

1-Cells	0-Cells
m	0
m	1
n	1
n	2
:	:

GridField: Grid with Bound Data

- Tuples of numeric primitives
- Total functions over cells of dimension k
- Two gridfields may share a grid

flux	area
11.5	3.3
13.9	5.5
13.1	4.5



x	y	salt	temp
x1	y1	29.4	12.1
x2	y2	29.8	12.5
x3	y3	28.0	12.0
x4	y4	30.1	13.2

Geometry

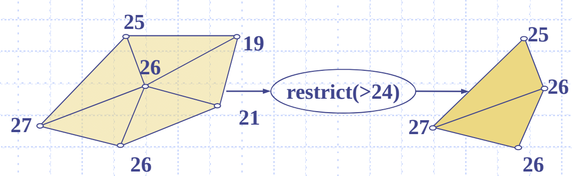
- Can derive cell incidence & adjacency from geometry in some cases
- Better to capture topology, have geometry as data
  - Many geometries for same topology
  - Geometry can change with time
  - Topology is often enough

Operators

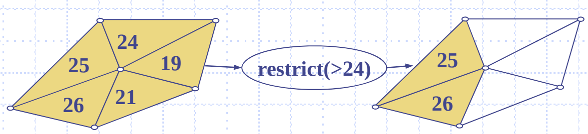
Task	Operator
associate grids with data	bind (b)
combine grids topologically	union, intersection, cross product (⊗)
reduce a grid using data values	restrict (r)
transform grids or data	aggregate (a)
map to new grid	regrid

Restrict Semantics

Values bound to 0-cells (nodes)

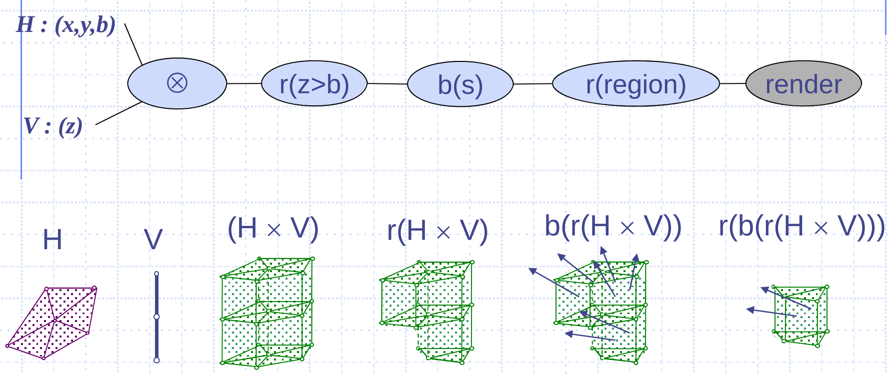


Values bound to 2-cells (triangles)

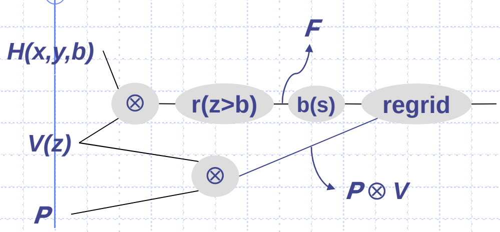


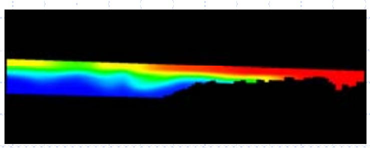
GridField Algebra

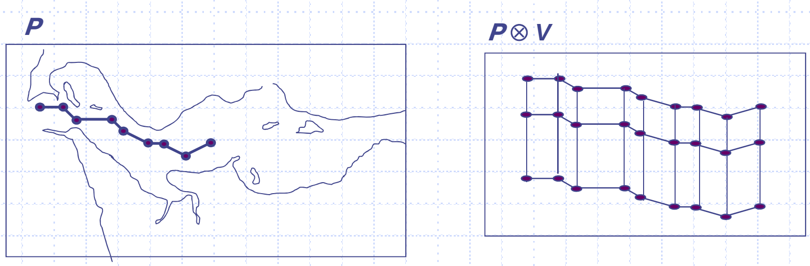
Build up a "recipe" of operators



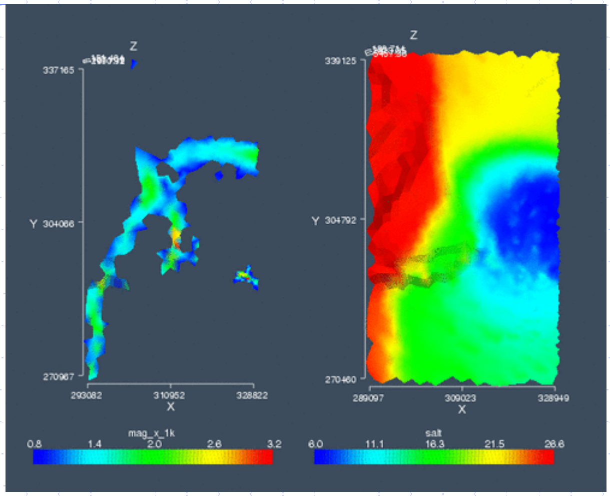
Transect (Vertical Slice)







Defining New Products: Plume Front



Whence GridFields?

- Initial version was in-memory
- Some work on exchange standards
  - Earth Systems Modeling Framework
- Want to investigate layering over SciDB

Thanks to

- SciDB (Version 11.6 coming soon!)
  - Marilyn Matz, Suchi Raman, Paul Brown, Paradigm4
  - www.scidb.org
- R-SciDB Interface
  - Patrick Leyshock, PSU
- Oceanographic Examples
  - Center for Coastal Margin Observation and Prediction (CMOP)
  - www.stccmop.org
- GridFields
  - Bill Howe, University of Washington eScience Institute
- Support from
  - NSF OCE-0424602