CIKM 2011 GLASGOW

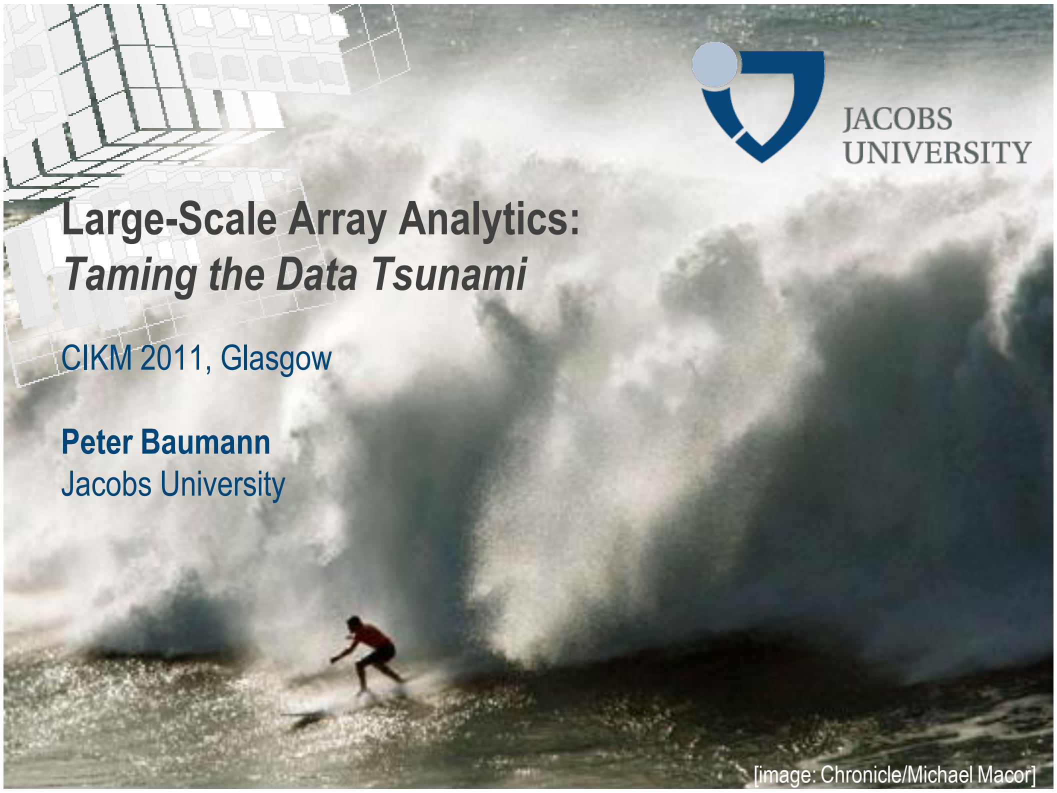
20TH ACM CONFERENCE ON INFORMATION AND KNOWLEDGE MANAGEMENT

Tutorial - AM2

Large-Scale Array Analytics: Taming the Data Tsunami

Peter Baumann

Crowne Plaza Hotel
Glasgow, Scotland
24-28 October 2011



Large-Scale Array Analytics: *Taming the Data Tsunami*

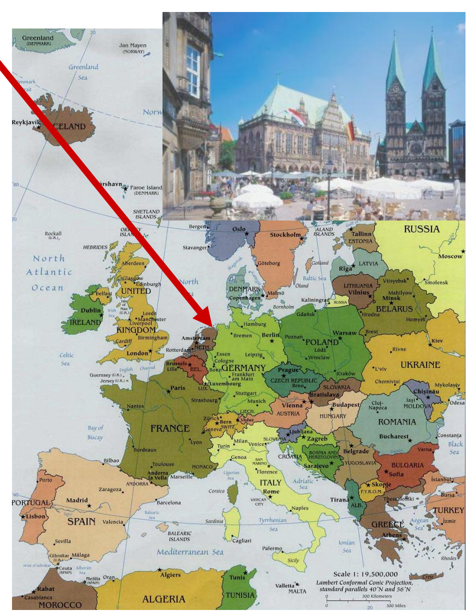
CIKM 2011, Glasgow

Peter Baumann
Jacobs University

[image: Chronicle/Michael Macor]

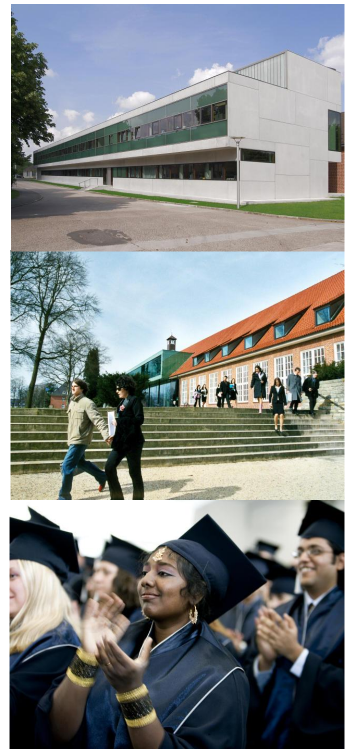
Jacobs University Bremen

- international, multi-cultural
  - 110 nations, English official language on campus



Array Research @ Jacobs U

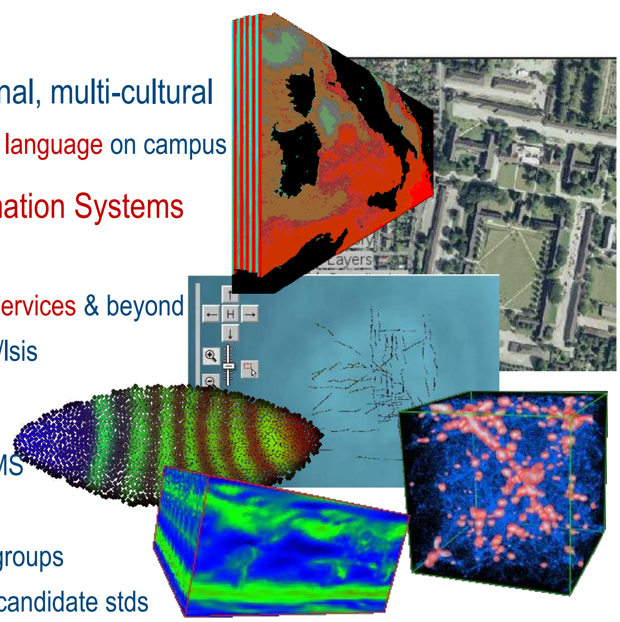
- Jacobs University: international, multi-cultural
  - 110 nations, English official language on campus
- Large-Scale Scientific Information Systems research group
  - focus: large-scale n-D raster services & beyond
  - See www.jacobs-university.de/lsis
- Results
  - rasdaman raster („array“) DBMS
  - OGC standardization
    - Chair, coverage working groups
    - editor of 8+ stds, several candidate stds



Array Research @ Jacobs U

JACOBS UNIVERSITY

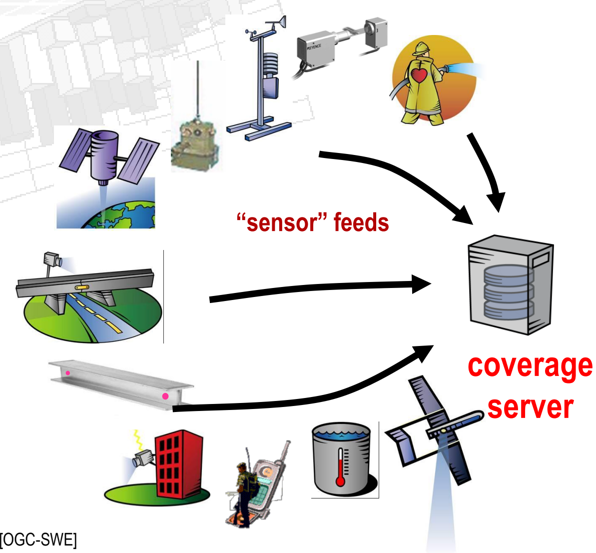
- Jacobs University: international, multi-cultural
  - 110 nations, English official language on campus
- Large-Scale Scientific Information Systems research group
  - focus: large-scale n-D raster services & beyond
  - See www.jacobs-university.de/lsis
- Results
  - rasdaman raster („array“) DBMS
  - OGC standardization
    - Chair, coverage working groups
    - editor of 8+ stds, several candidate stds



Roadmap

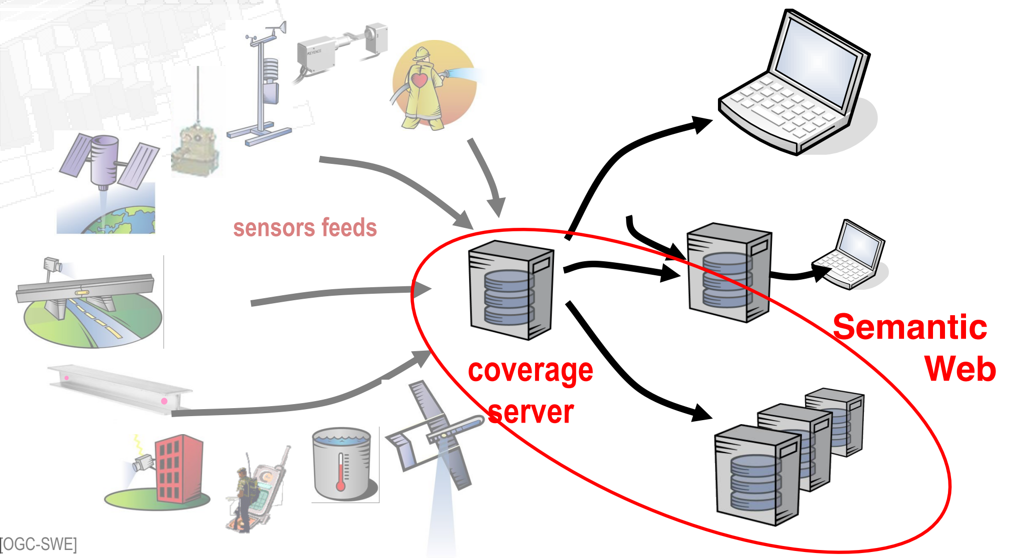
- Introduction
- Conceptual modelling
- Architecture
- Related Work
- Applications
- Wrap-up

Facing the Data Tsunami



[OGC-SWE]

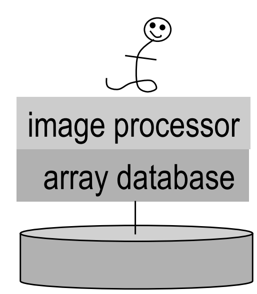
Taming the Data Tsunami



[OGC-SWE]

Array-Intensive Methods: Differentiation

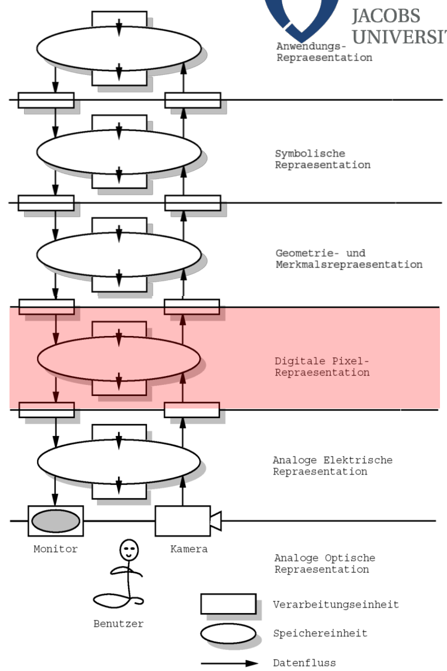
- multimedia databases
  - Analyse images, then drop them and work on auxiliary structure (ie, feature vector)
- image processing
  - Advanced processing of rasters, but on main memory size objects
- image understanding, computer vision
  - Aiming at feature extraction etc → specific task
  - Again, not significantly beyond main memory sizes
- visual analytics
  - Visual display/interaction of analysis results
  - Again, main memory size limits



Reference Model for Visualization

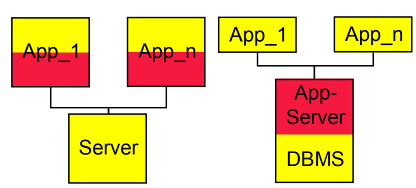
[Krömker 1991]

- Application specific. Parts represent an „image“ in app context
- Abstract description, no explicit geometry and features
- Geometric description, attributes, features, viewing parameters
- Space and color discretisation
- Images als analog signals
- Optical signals as visual stimuli



Why Array *Databases*?

- "classical“ database benefits for raster data:
  - data integration
  - flexibility
  - scalability
  - *...plus all further assets, like off-the-shelf tool support*



- Unfortunately database people are soooo conservative
  - "images are matrices [...] which are stored as byte strings, ie, BLOBs"
  - Array databases fill this gap

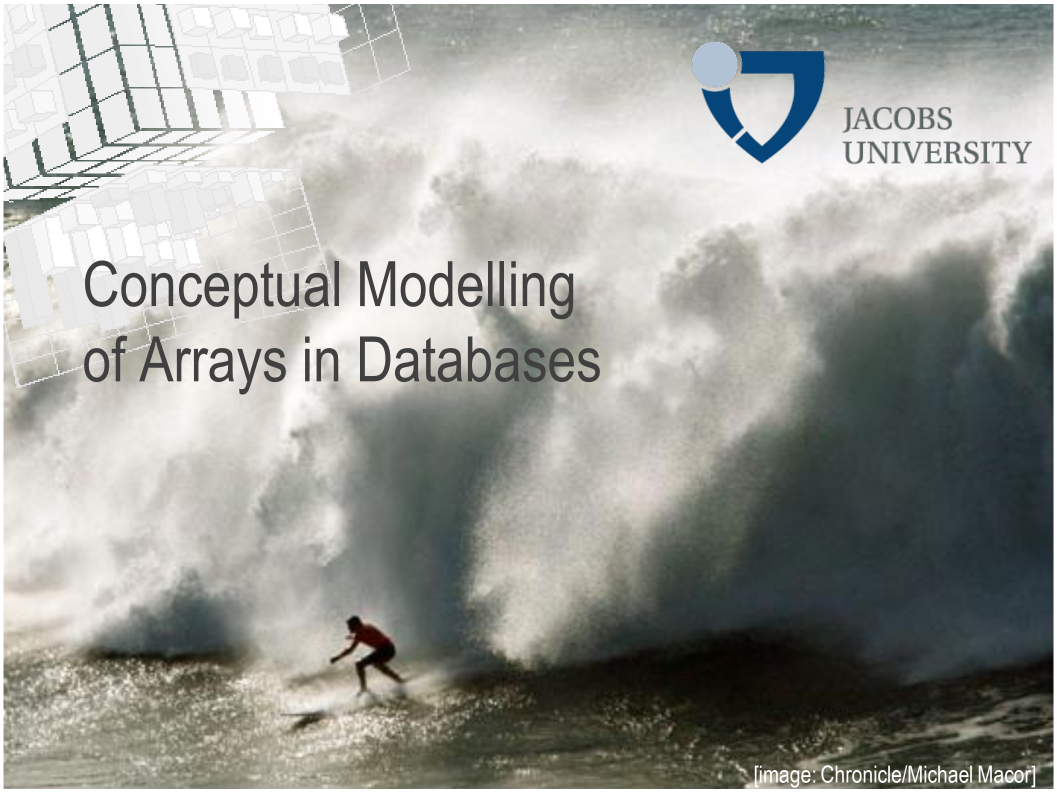
Array Analytics

- Array Analytics :=
  Efficient analysis on multi-dimensional arrays of a size several orders of magnitude above evaluation engine's main memory
  - Typically in client/server setup
  - „Big Science" on „Big Data", both ad-hoc and long-tail
  - For this talk: „array" = „raster"
- Issues:
  - Concepts: modeling, access interfaces (query languages)
  - Architecture: storage, processing, optimization
  - Scalability, usability, applications, standards
- *...obviously a typical database task (why didn't we realize this earlier?)*

Who Needs Array Databases?

- Sensor, image, statistics data
  - **Life Science:** Pharma/chem, healthcare / bio research, bio statistics, genetics
  - **Geo:** Geodesy, geology, hydrology, oceanography, meteorology, earth system research, ...
  - **Engineering & research:** Simulation & experimental data in automotive/shipbuilding/ aerospace industry, turbines, process industry, astronomy, experimental physics, high energy physics, ...
  - **Management/Controlling:** Decision Support, OLAP, Data Warehousing, census, statistics in industry and public administration, ...
  - **Multimedia:** e-learning, distance learning, prepress, ...

Conceptual Modelling of Arrays in Databases



[image: Chronicle/Michael Macor]

Array Models: History

- Image partitioning, API access library [Tamura 1980]
- Fixed set of imaging operators [Chang, Fu 1980; Stucky, Menzi 1989; Neumann et al 1992]
  - scaling, rotation, edge extraction, thresholding, ...
- PICDMS [Chock, Cardenas 1984]
  - stack of images (identical resolution); operations corresponding to rasql "induced" ops; no nesting; no architecture
- rasdaman [1991+], AQL [Libkin & Machlin, 1996+], AML [Marathe, Salem 1997], MonetDB [Zhang et al 2011]: formal array model for databases
- ESRI, Oracle; Google, Microsoft, …: ad-hoc solutions

The rasdaman Array Algebra

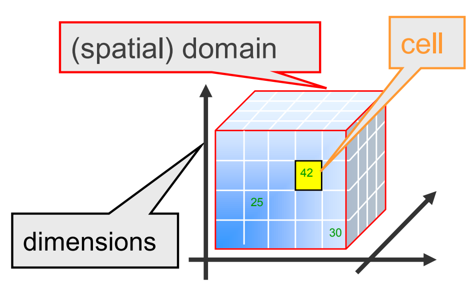
JACOBS UNIVERSITY

- Goal: enabling databases with support for massive n-D Sensor, Image, & Statistics Data [Baumann 1992+]
- Starting point was user study:
  how do imaging people model n-D array operations?
  - Most inspired by AFATL Image Algebra [Ritter et al 1990]
- Algebra basis for conceptual model, storage mapping, & optimization
  - Simplified: only arrays; reduced set of "pixel" types (atomic & nested records)
  - Database-adjusted: small, closed set of primitives, safe in evaluation

Array Algebra Overview

JACOBS UNIVERSITY

- array = function: $a: X \rightarrow F$
  (X n-D integer interval)
  $a = \{ (x,a(x)): x \in X, a(x) \in F \}$



- Core operations:
  - array constructor -- build array & initialize from cell expression
  - Condenser -- summarize over array, delivering a scalar (using some commutative & associative summarization op)
  - Sorter -- slice array along a dimension, sort slices
- All else just shorthands: image addition, overlaying, statistics, ...

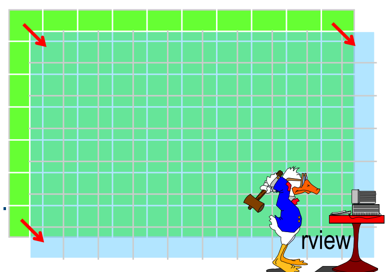
Array Operations: MARRAY

JACOBS UNIVERSITY

- Array constructor: MARRAY ($e|_x$, X, x) := { (x,f): f = $e|_x$, $x \in X$ }
  - for expression $e|_x$
    potentially containing occurrences of x, of result type F
- Example: image addition
  - a + b := MARRAY(a[x] + b[x], X, x) := { (x,f): f = a[x] + b[x], $x \in X$ }

addition of pixels!

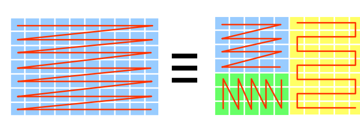
- $\rightarrow$ shorthands:
  unary and binary "induced" operations
  - *"whenever I have a pixel operation,*
    I automatically have the corresponding
    image operation"
  - Image addition, comparison, component access, ...
    a + b, a > b, a.green, ...



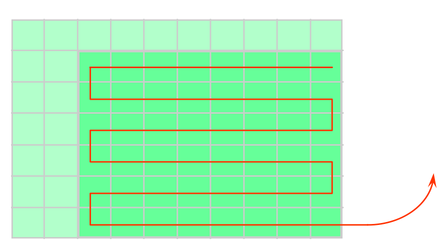
Array Operations: COND

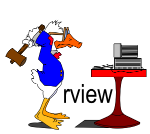
JACOBS UNIVERSITY

- Condenser: COND($e|_{a,x}$, o, X, x) := $e|_{a,p1}$ o $e|_{a,p2}$ o ... o $e|_{a,pn}$
  - x visits each coordinate in X = { p_1, ..., p_n }
  - $e|_{a,pi}$ expression potentially containing a and p_i
  - o commutative, associative



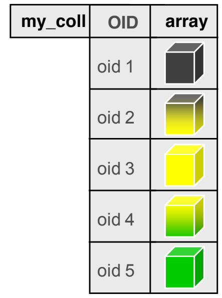
- Example: "*Sum over all cell values*"
  - add(a) = COND(a[x], +, sdom(a), x)
    = $a[p_1]$ + $a[p_2]$ + ... + $a[p_n]$





From Algebra To Query Language

- Data model:
  (multi-) sets („collections“) of typed arrays
- Data definition language rasdl [ODMG ODL]
  - Parametrised array constructor
- Retrieval and manipulation language rasql [ISO SQL92]
  - Set oriented, multidimensional operators
- Architecture streamlined towards piecewise processing of large objects
  - Tile streaming



Raster Type Definition

All C/C++ types, except pointers

```
• typedef marray
  <    unsigned char, [ 1:1024, 1:768 ]
  > XGA_Grey_Image;

• typedef marray
  <    struct { unsigned char red, green, blue; }, [ *:*, *:*
  ]
  > RGB_Image;

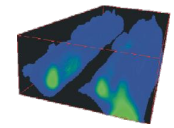
• typedef marray
  <    unsigned short, [ 1:1654, 1:* ]
  > G3_Fax;

• typedef marray
  <    struct { double vx, vy; }, [ 0:*, 0:127, 0:63, 0:16 ]
  > ECHAM_T42_Windspeed;
```

The rasql Query Language

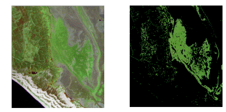
- selection & section

```
select c[ *:*, 100:200, *:*, 42 ]
from   ClimateSimulations as c
```



- result processing

```
select img * (img.green > 130)
from   LandsatArchive as img
```

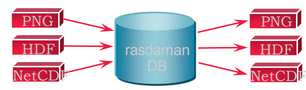


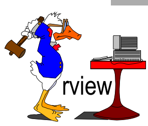
- search & aggregation

```
select mri
from   MRI as img, masks as am
where  some_cells( mri > 250 and m )
```

- data format conversion

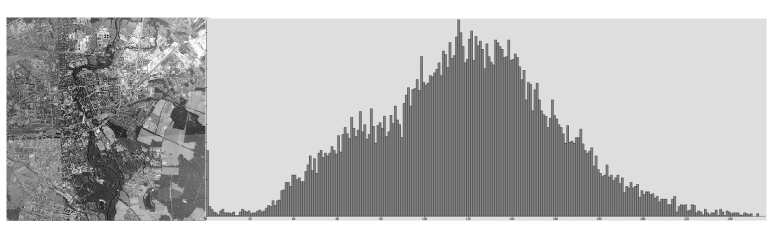
```
select png( c[ *:*, *:*, 100, 42 ] )
from   ClimateSimulations as c
```





Application Example: Histogram

- Histogram of an n-D array over 8-bit unsigned integer:
  - ```
    select marray n in [0:255]
           values count_cells( image = n )
    from   image
    ```
- changes cell type, dimension, domain


    ```

Architecture I: Storage Mapping

[image: Chronicle/Michael Macor]

Storage Mapping

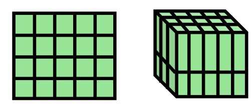
- Task:
  - materialise finite interval $X \subset \mathbf{Z}^n$, find suitable (disk) access structure
  - Core structural property: Euclidean neighbourhood in $\mathbf{Z}^n$
  - Secondary, contents/app based: data density („sparsity"), data pattern, access pattern
- Excursion: arrays in main memory
  - Ex: APL [Iverson 1968]
  - Assumption 1:
    access times independent from array position
    - *cost(* „`a[x]`" *) = const for all* „`x`"
  - Assumption 2:
    access times independent from access sequence
    - *cost(* „`a[x];a[y]`" *) = 2*cost(* „`a[x]`" *) = const for all* „`x`", „`y`"

Storage Mapping: Variants

JACOBS UNIVERSITY

- Coordinate-free sequence
  - BLOB (binary large object)
  - *Costs mainly position/dimension dependent*

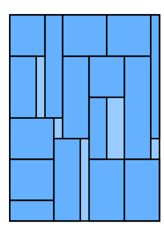
- Sequence independent, coordinates explicit $\{ (x_1,f_1), (x_2,f_2), \ldots, (x_n,f_n) \}$
  - ROLAP
  - *Costs not position correlated, but high*
- Imaging, multidimensional OLAP
  - Partitioning, sequence within partition
  - *Costs low for bulk access, usually not location correlated*

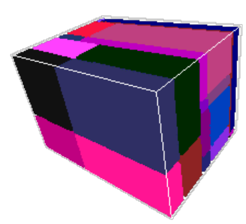


Tiled Array Storage

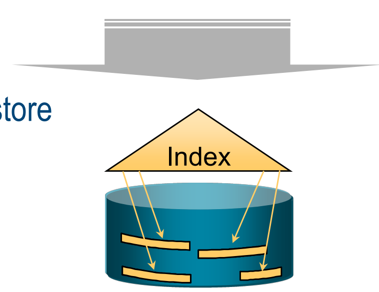
JACOBS UNIVERSITY

- partition multidimensional object
  → multidimensional tiles
  - Tile = subarray
    [Widmann 2001, Furtado 2002]
  - Regular tiling = mosaicking [imaging, geo],
    chunking [Sarawagi, DeWitt, Stonebraker]



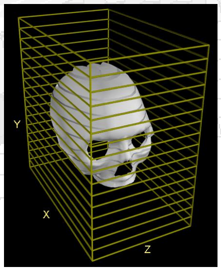


- Tiles form unit of access in persistent store
  - Ex: BLOB in relational database
  - Compression, geo index

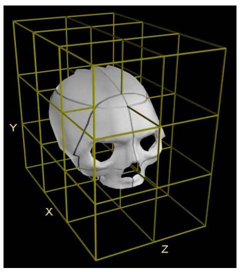


Benchmarks: Tiling Strategy

JACOBS UNIVERSITY



tomo_sliced 153x256
time:

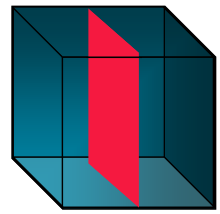


tomo_cubed 32x32x32
time:

Operand: 3-D MDD object

Operation: Z cut

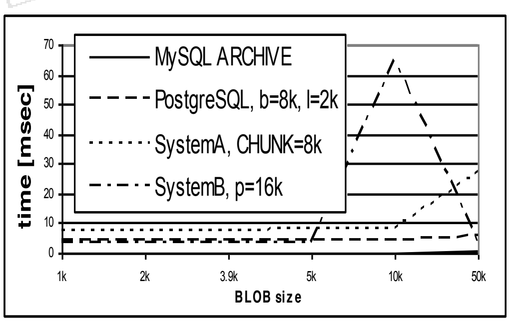
selectivity: 1.6 %

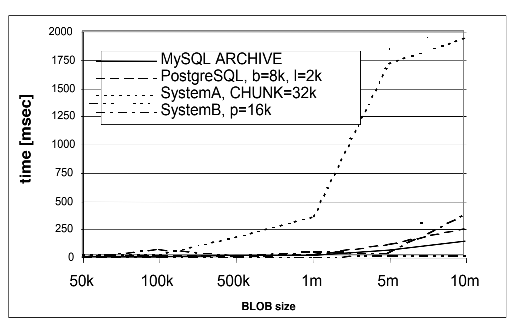


Comparison: BLOB Read Performance

JACOBS UNIVERSITY

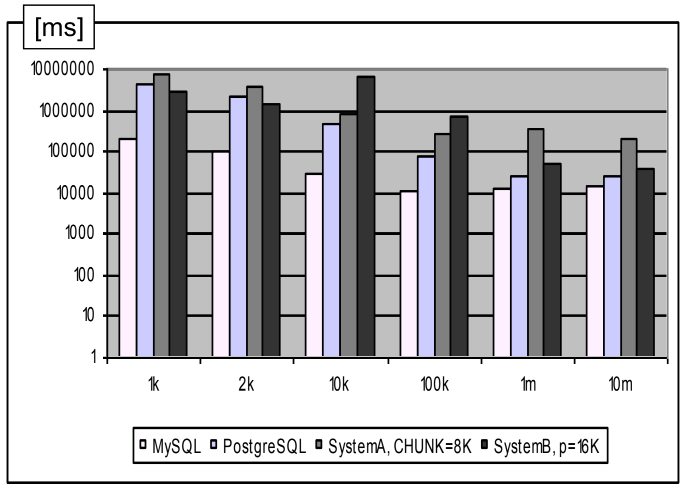
- Optimal tuning per system
- OS competitors often better!





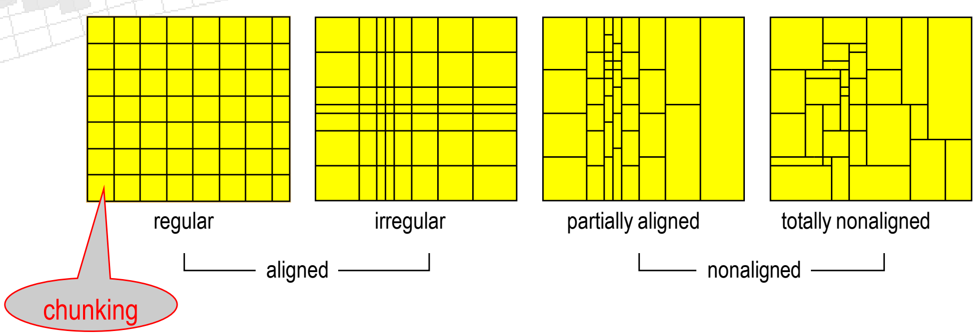
Comparison: Time to Read (Deduced)

- performance varies by two orders of magnitude!
  - @100K / MySQL vs @10K / SystemB



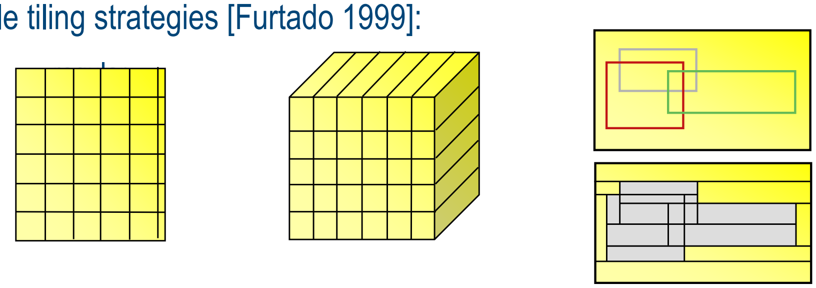
Tiling Strategies

- Goal: faster tile loading by adapting storage units to access pattern
- Tiling classification [Furtado+ 1999] based on degree of alignment



Tiling Strategies

- Goal: faster tile loading by adapting storage units to access pattern
- Tiling classification [Furtado+ 1999] based on degree of alignment
- Issues
  - When is tiling optimal? Tiling strategies?
- 3 sample tiling strategies [Furtado 1999]:



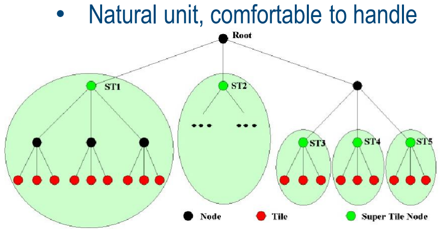
Storage Layout Language

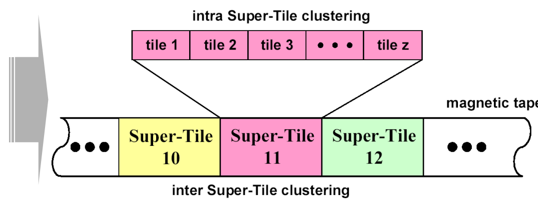
- Goal: Support ad-hoc storage tuning
- Approach: array storage layout sub-language extending *insert* statement [Baumann+ 2010]
- Ex:

```
insert into MyCollection
  values ...
  tiling area of interest [0:20,0:40], [45:80,80:85]
  tile size 1000000
  index d_index
  storage array
  compression zlib
```

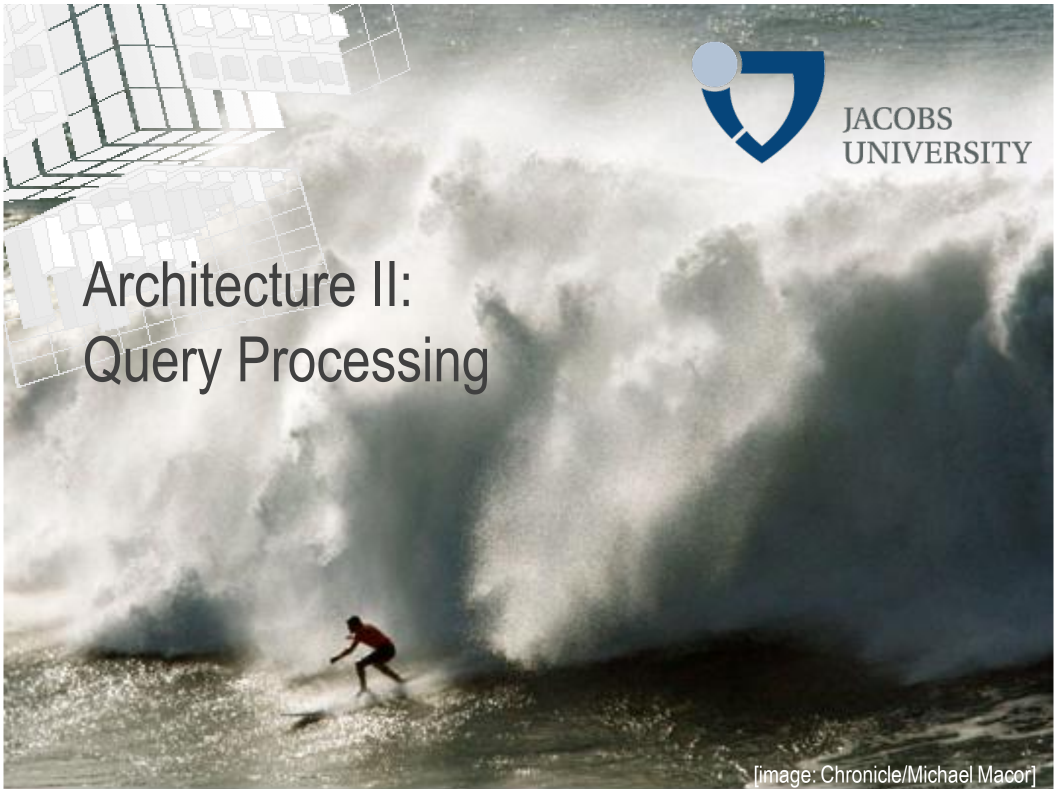
Adding Tertiary Storage

- tape archives for near-line access [Sarawagi, Stonebraker 1994]
- Problem: respect spatial clustering
  - Access locality (long positioning times!)
- Approach: super tiles = all tiles of particular index node [Reiner 2001]
  - Natural unit, comfortable to handle



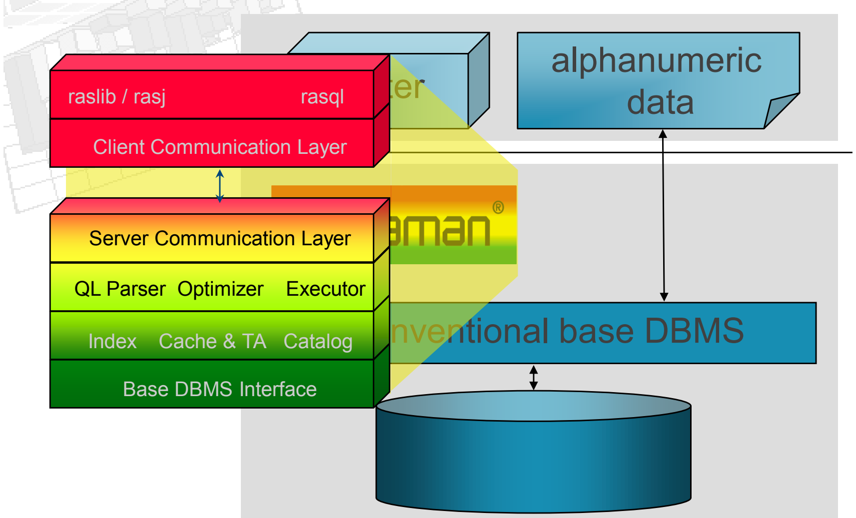


Coffee Break!



Architecture

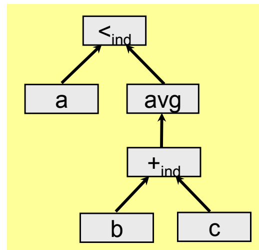
JACOBS UNIVERSITY



Query Processing: Overview

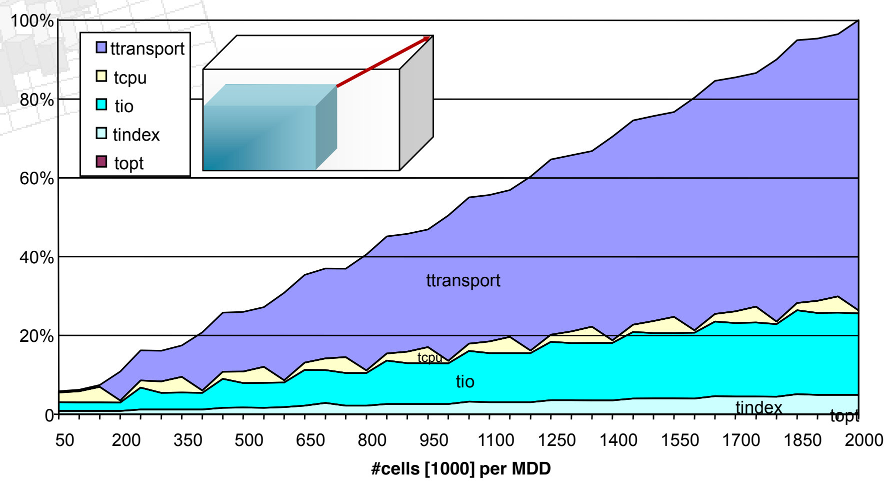
- Parsing
- Normalisation
- Optimization
  - Common subexpression elimination
- [Generate query plan]
- Tile-based evaluation

```
select a < avg_cells( b + c )
from   a, b, c
```



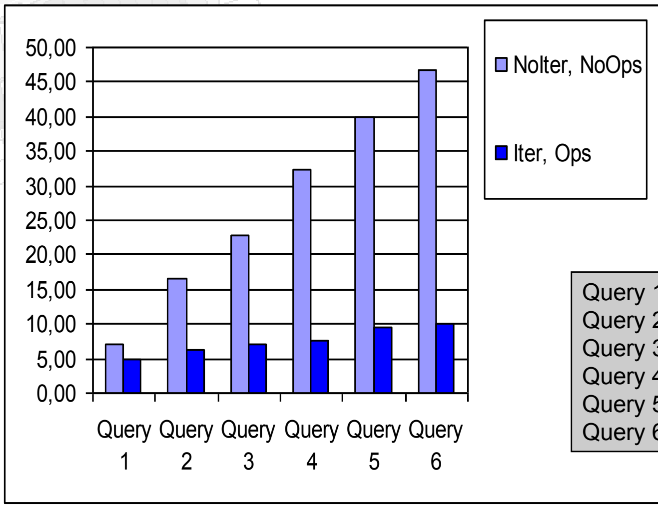
Benchmarks: Data Access

[Ritsch 2000, Widmann 2001]



Benchmarks: Data Processing

[Ritsch 2000, Widmann 2001]



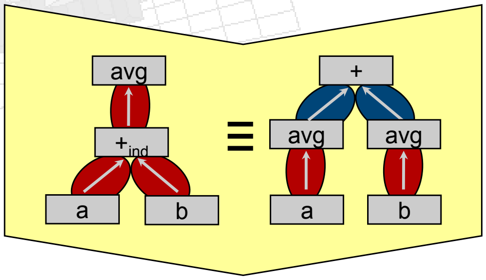
Query 1: access to 2-D object
Query 2: + 1 induced operation
Query 3: + 2 induced operations
Query 4: + 3 induced operations
Query 5: + 4 induced operations
Query 6: + 5 induced operations

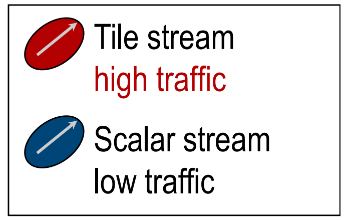
"Can't We Do That Object-Relationally?"

- Marray is not a *type*, but a type *constructor*
- Cf. Stack:
  - Stack<> is type constructor
  - Stack<int>, stack<float>, ... are concrete, instantiated types
- Relational model does not know type constructors → hard to integrate
  - does not even know user-defined attribute types
- Object-relational extensions allow user-defined data types, however not type constructors → no benefit
- Actually, whole engine stack needs reimplementation
  - Sub-page tuples vs multi-page (multi-disk!) arrays

Query Rewriting

```
select avg_cells( a + b )
from   a, b
```






```
select avg_cells( a )
     + avg_cells( b )
from   a, b
```

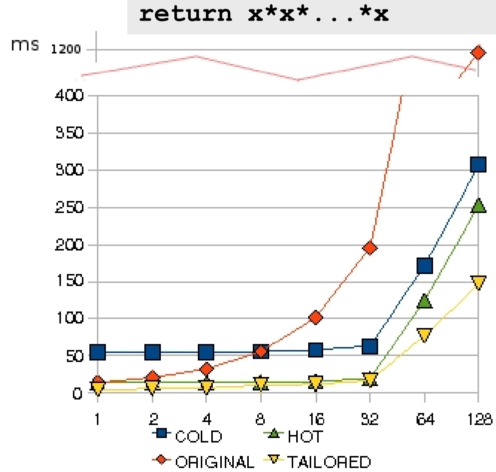
- *understood: heuristic optimization – 150 rules in rasdaman [Ritsch 2002]*
- *partially understood: cost-based optimization*

Just-In-Time Compilation

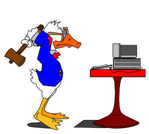
[Jucovschi, Stancu-Mara 2008]

- Observation: interpreted mode slows down
- Approach:
  - cluster suitable operations
  - compile & dynamically bind
- Benefit:
  - Speed up complex, repeated operations
- Variation:
  - compile code for GPU

```
for x in (float_matrix)
return x*x*...*x
```



Times [ms] for 512^2 * n ops

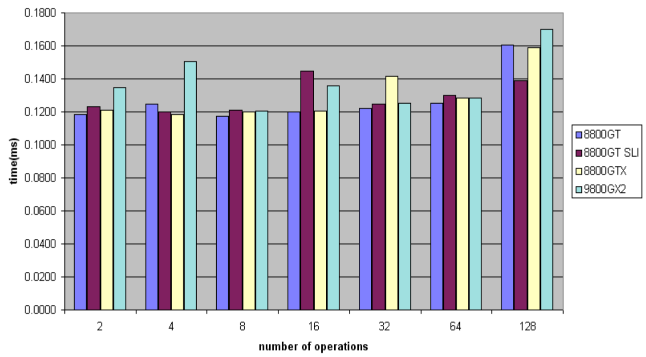


GPU Processing

JACOBS UNIVERSITY

[Stancu-Mara 2008]

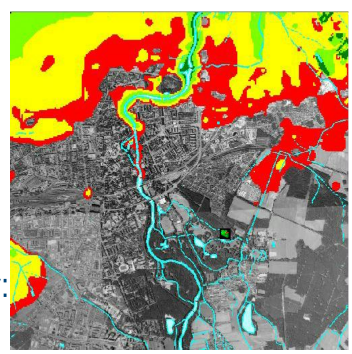
- Observation: pixelwise operations costly
- Approach (patented):
  - cluster suitable operations
  - Generate GPU code
  - Spawn GPU process
- Advantages:
  - keep CPU + GPU humming
  - # GPU cores >> # CPU cores
  - GPU driver schedules



- Preliminary observation:
  performance independent from #ops for up to ~100 ops

Optimisation Does Pay Off!

- Complex queries give more space to optimizer
- Example 1: Typical OGC Web Map Service query:



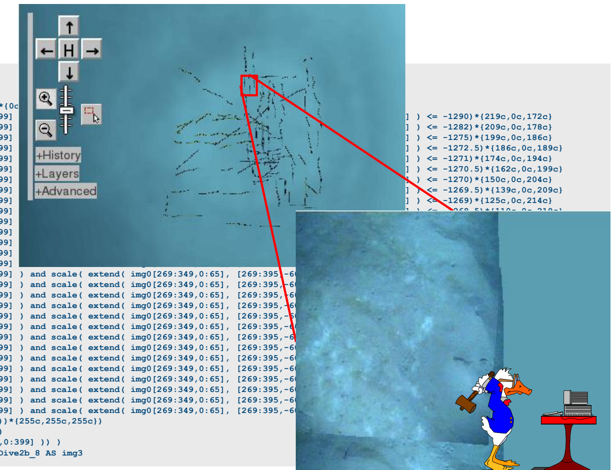
```
select jpeg(
              scale(bild0[...],[1:300,1:300])          * { 1c, 1c, 1c}
      overlay ((scale(bild1[...],[1:300,1:300])<71.0)) * {51c, 153c, 255c }
      overlay bit(scale(bild2[...],[1:300,1:300]), 2)  * {230c, 230c, 204c}
      overlay bit(scale(bild2[...],[1:300,1:300]), 5)  * {1c, 1c, 1c}
      overlay bit(scale(bild2[...],[1:300,1:300]), 7)  * {102c, 102c, 102c}
      overlay bit(scale(bild2[...],[1:300,1:300]), 6)  * {255c, 255c, 0c}
      overlay bit(scale(bild2[...],[1:300,1:300]), 3)  * {191c, 242c, 128c}
      overlay bit(scale(bild2[...],[1:300,1:300]), 4)  * {191c, 255c, 255c}
      overlay bit(scale(bild2[...],[1:300,1:300]), 1)  * {0c, 255c, 255c}
      overlay bit(scale(bild2[...],[1:300,1:300]), 0)  * {102c, 102c, 102c}
        )
from ...
```

Optimization Does Pay Off!

JACOBS UNIVERSITY

- Example 2: real-time WMS zoom/pan/styling
  - 1 background, 1 bathymetry, 3*RGB
  - www.earthlook.org

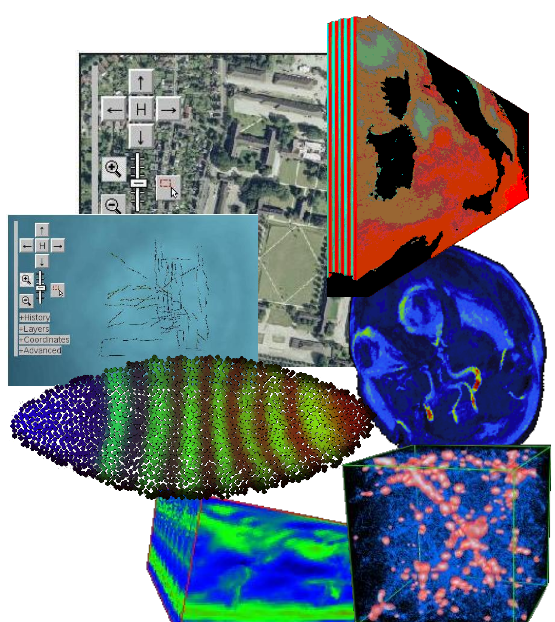
```
SELECT png(
(marray x in [0:399,0:399] values {255c,255c,255c})
overlay
((scale( extend( img0[269:349,0:65], [269:395,-60:65] ), [0:399,0:399] ) < -1300)*{0c
+(-1300.000000< scale( extend( img0[269:349,0:65], [269:395,-60:65] ), [0:399,0:399]   ] ) <= -1290)*{219c,0c,172c}
+(-1289.999999< scale( extend( img0[269:349,0:65], [269:395,-60:65] ), [0:399,0:399]   ] ) <= -1282)*{209c,0c,178c}
+(-1281.999999< scale( extend( img0[269:349,0:65], [269:395,-60:65] ), [0:399,0:399]   ] ) <= -1275)*{199c,0c,186c}
+(-1274.999999< scale( extend( img0[269:349,0:65], [269:395,-60:65] ), [0:399,0:399]   ] ) <= -1272.5)*{186c,0c,189c}
+(-1272.499999< scale( extend( img0[269:349,0:65], [269:395,-60:65] ), [0:399,0:399]   ] ) <= -1271)*{174c,0c,194c}
+(-1270.999999< scale( extend( img0[269:349,0:65], [269:395,-60:65] ), [0:399,0:399]   ] ) <= -1270.5)*{162c,0c,199c}
+(-1270.499999< scale( extend( img0[269:349,0:65], [269:395,-60:65] ), [0:399,0:399]   ] ) <= -1270)*{150c,0c,204c}
+(-1269.999999< scale( extend( img0[269:349,0:65], [269:395,-60:65] ), [0:399,0:399]   ] ) <= -1269.5)*{139c,0c,209c}
+(-1269.499999< scale( extend( img0[269:349,0:65], [269:395,-60:65] ), [0:399,0:399]   ] ) <= -1269)*{125c,0c,214c}
+(-1268.999999< scale( extend( img0[269:349,0:65], [269:395,-60:65] ), [0:399,0:399]
+(-1268.499999< scale( extend( img0[269:349,0:65], [269:395,-60:65] ), [0:399,0:399]
+(-1267.999999< scale( extend( img0[269:349,0:65], [269:395,-60:65] ), [0:399,0:399]
+(-1267.499999< scale( extend( img0[269:349,0:65], [269:395,-60:65] ), [0:399,0:399]
+(-1266.999999< scale( extend( img0[269:349,0:65], [269:395,-60:65] ), [0:399,0:399]
+(-1266.499999< scale( extend( img0[269:349,0:65], [269:395,-60:65] ), [0:399,0:399]
+(-1265.999999< scale( extend( img0[269:349,0:65], [269:395,-60:65] ), [0:399,0:399] ) and scale( extend( img0[269:349,0:65], [269:395,-6
+(-1265.499999< scale( extend( img0[269:349,0:65], [269:395,-60:65] ), [0:399,0:399] ) and scale( extend( img0[269:349,0:65], [269:395,-6
+(-1264.999999< scale( extend( img0[269:349,0:65], [269:395,-60:65] ), [0:399,0:399] ) and scale( extend( img0[269:349,0:65], [269:395,-6
+(-1264.499999< scale( extend( img0[269:349,0:65], [269:395,-60:65] ), [0:399,0:399] ) and scale( extend( img0[269:349,0:65], [269:395,-6
+(-1263.999999< scale( extend( img0[269:349,0:65], [269:395,-60:65] ), [0:399,0:399] ) and scale( extend( img0[269:349,0:65], [269:395,-6
+(-1263.499999< scale( extend( img0[269:349,0:65], [269:395,-60:65] ), [0:399,0:399] ) and scale( extend( img0[269:349,0:65], [269:395,-6
+(-1262.999999< scale( extend( img0[269:349,0:65], [269:395,-60:65] ), [0:399,0:399] ) and scale( extend( img0[269:349,0:65], [269:395,-6
+(-1261.999999< scale( extend( img0[269:349,0:65], [269:395,-60:65] ), [0:399,0:399] ) and scale( extend( img0[269:349,0:65], [269:395,-6
+(-1260.999999< scale( extend( img0[269:349,0:65], [269:395,-60:65] ), [0:399,0:399] ) and scale( extend( img0[269:349,0:65], [269:395,-6
+(-1259.999999< scale( extend( img0[269:349,0:65], [269:395,-60:65] ), [0:399,0:399] ) and scale( extend( img0[269:349,0:65], [269:395,-6
+(-1256.999999< scale( extend( img0[269:349,0:65], [269:395,-60:65] ), [0:399,0:399] ) and scale( extend( img0[269:349,0:65], [269:395,-6
+(-1249.999999< scale( extend( img0[269:349,0:65], [269:395,-60:65] ), [0:399,0:399] ) and scale( extend( img0[269:349,0:65], [269:395,-6
+(-1239.999999< scale( extend( img0[269:349,0:65], [269:395,-60:65] ), [0:399,0:399] ) and scale( extend( img0[269:349,0:65], [269:395,-6
+(-1229.999999< scale( extend( img0[269:349,0:65], [269:395,-60:65] ), [0:399,0:399] ) and scale( extend( img0[269:349,0:65], [269:395,-6
+ (-126.5 < scale( extend( img0[269:349,0:65], [269:395,-60:65] ), [0:399,0:399] ))*{255c,255c,255c})
overlay (scale( extend( img2[124:468,0:578], [124:717,-14:578] ), [0:399,0:399] ))
overlay (scale( extend( img3[11375:11578,0:120], [11375:11968,-473:120] ), [0:399,0:399] )) )
FROM Hakon_Bathy AS img0, Hakoon_Dive1_8 AS img1, Hakoon_Dive2_8 AS img2, Hakoon_Dive2b_8 AS img3
```



Optimization

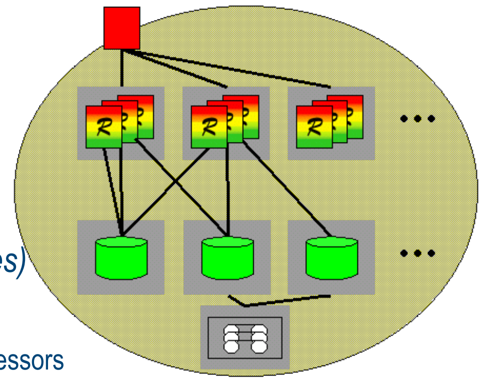
JACOBS UNIVERSITY

- Adaptive tiling
- Adaptive compression
- Multi-dimensional indexing
- Distributed query processing
- Query rewriting
- Pre-aggregation
- Physical operator clustering
- Transparent tape integration
- Just-in-time compilation
- GPU processing
- Tile caching
- ...



Query Parallelisation

- *easy: inter-query parallelization (one client – one server process)*
  - Long-runners don't block service
  - higher throughput
- *Non-trivial: intra-query parallelization (one client – several server processes) [Hahn 2003]*
  - Idea: tiles dynamically assigned to processors
  - *Non-trivial array index patterns?*



Array Database Applications

[image: Chronicle/Michael Macor]

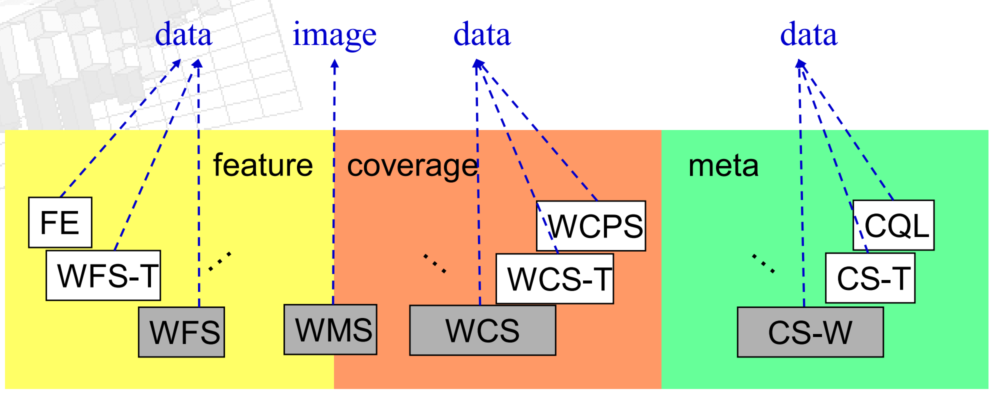
Geo Service Standardization

JACOBS UNIVERSITY

- OGC (Open GeoSpatial Consortium) driving geo service standards
  - Web-based modular, open, interoperable geo services
  - Liaisons with ISO TC 211, OASIS, CGI/IUGS; ...
  - consensus body, specs tested before released (eg, testbeds)
  - www.opengeospatial.org
- Array data special category of coverage in OGC / GIS speak
  - Web Coverage Service Standards Working Group (WCS.SWG)
  - Web Coverage Processing Service Group (WCPS)
  - Coverages WG
  - Metocean Domain Working Group
  - GALEON (Geo-interface to Atmosphere, Land, Earth, Ocean, NetCDF) OGCnetwork

(Part of) The OGC Quilt

JACOBS UNIVERSITY

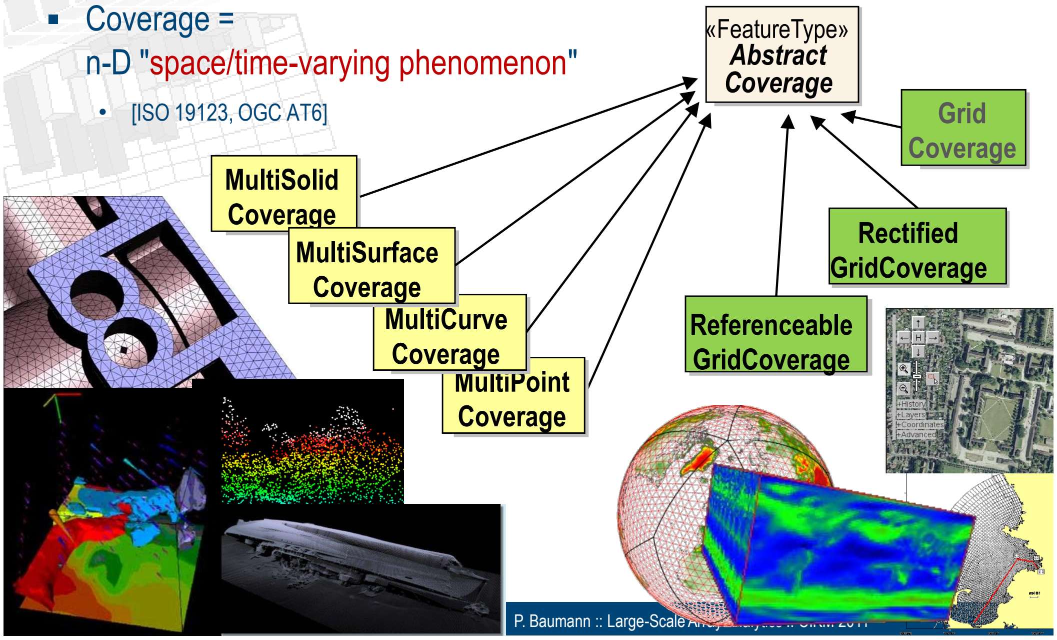


What Is a Coverage, After All?

JACOBS UNIVERSITY

[OGC 09-146r1]

- Coverage =
  n-D "space/time-varying phenomenon"
  - [ISO 19123, OGC AT6]

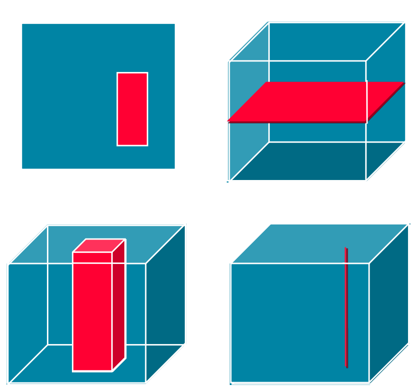


WCS Core Functionality

JACOBS UNIVERSITY

- In Core, simple data access (more in extension packages):

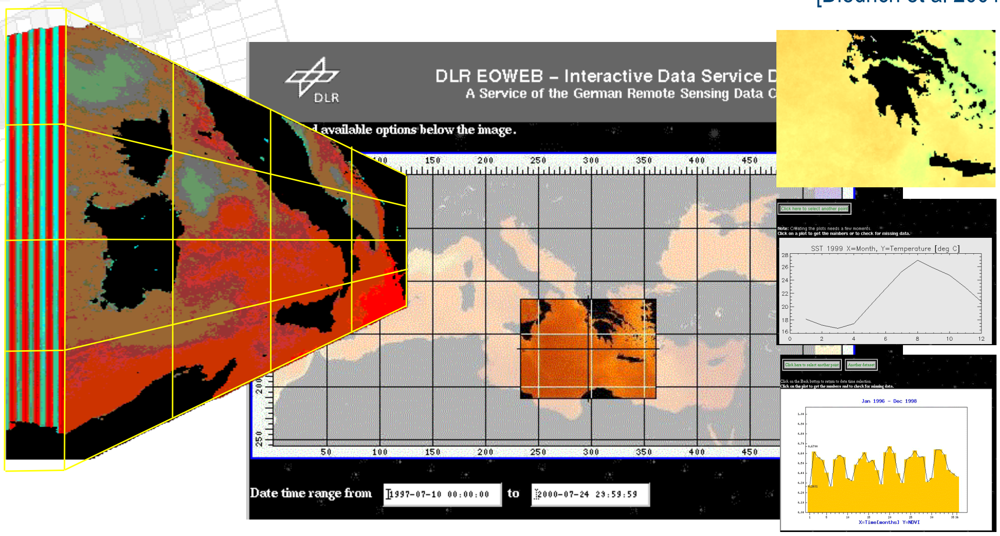
subset = trim | slice



Use Case: Satellite ImageTime Series

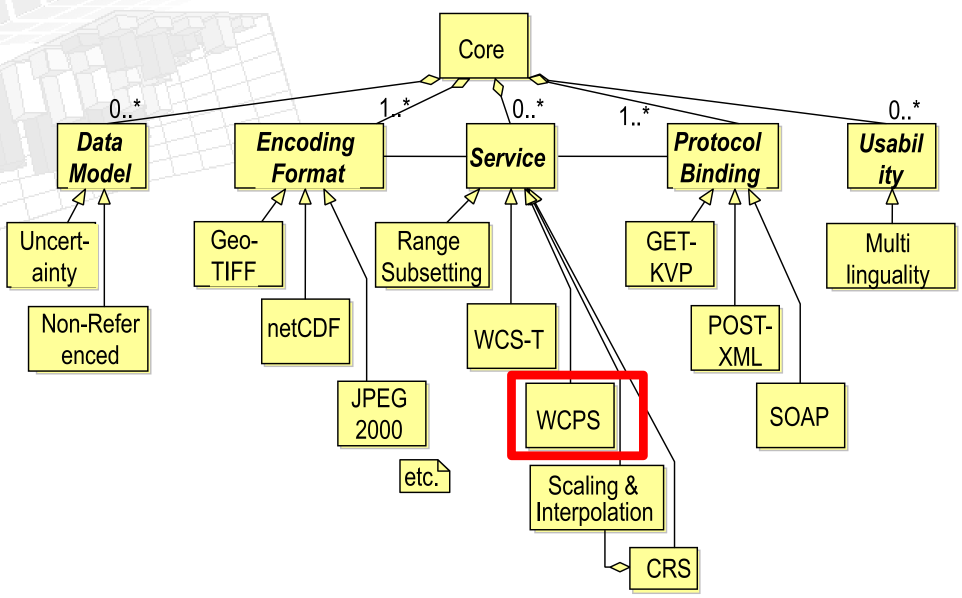
JACOBS UNIVERSITY

[Diedrich et al 2001]



WCS Suite: The Big Picture

JACOBS UNIVERSITY



Web Coverage Processing Service

- OGC WCPS standard, adopted 2008 [OGC 08-068r2]
  = aka "XQuery for multi-dimensional coverages"
  - image & signal processing, statistics
- (semi) formal algebraic semantics
- Safe in evaluation
- Expression nesting → unlimited complexity

WCPS By Example

- "From MODIS scenes **M1**, **M2**, and **M3**, the absolute of the difference between **red** and **nir**, in HDF-EOS"

```
for $c in ( M1, M2, M3 )
return
    encode(
       abs( $c.red - $c.nir ),
       "hdf"
    )
```

→ (hdf_A, hdf_B, hdf_C)

WCPS By Example

JACOBS UNIVERSITY

- "From MODIS scenes **M1**, **M2**, and **M3**, the absolute of the difference between **red** and **nir**, in HDF-EOS"
  - ...but only those where nir exceeds 127 somewhere

```
for $c in ( M1, M2, M3 )
where
    some( $c.nir > 127 )
return
    encode
        abs( $c.red - $c.nir ),
        "hdf"
    )
```

→ (hdf_A, hdf_C)

WCPS By Example

JACOBS UNIVERSITY

- "From MODIS scenes **M1**, **M2**, and **M3**, the absolute of the difference between **red** and **nir**, in HDF-EOS"
  - ...but only those where nir exceeds 127 somewhere
  - ...inside region R

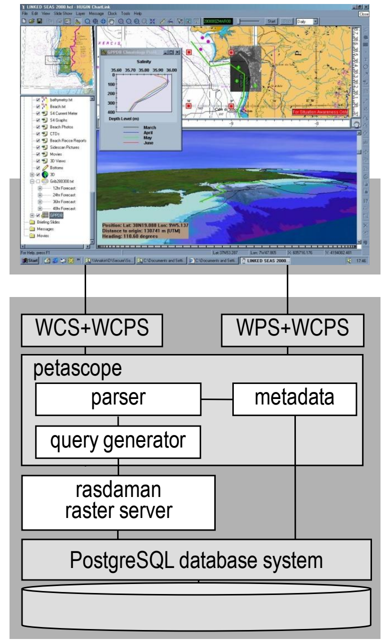
```
for $c in ( M1, M2, M3 ),
    $r in ( R )
where
    some( $c.nir > 127 and $r )
return
    encode
        abs( $c.red - $c.nir ),
        "hdf"
    )
```

→ (hdf_A)

Sample WCPS-Based C/S Architecture

JACOBS UNIVERSITY

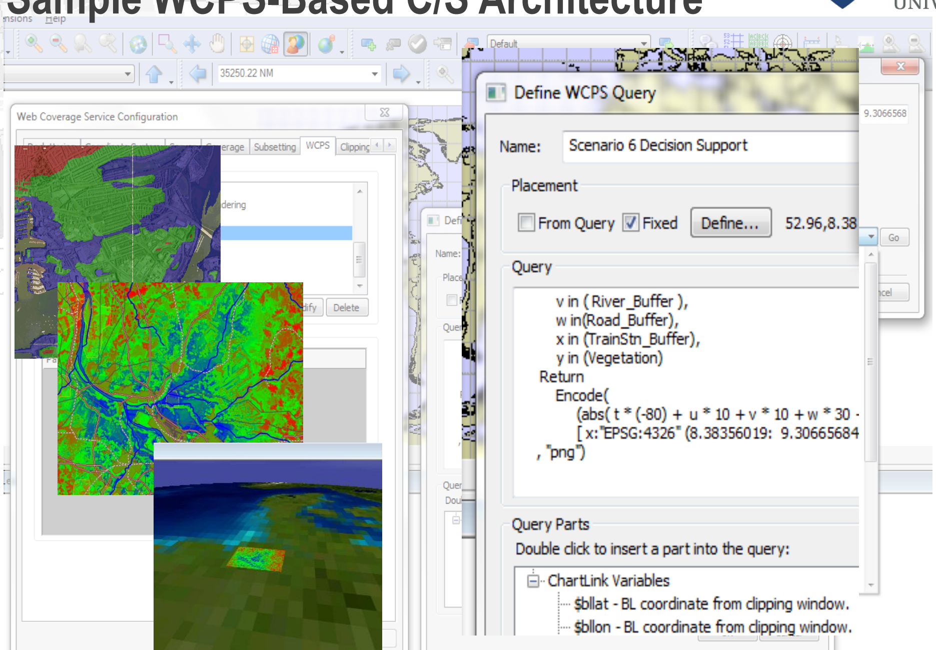
- ChartLink (Envitia Ltd, UK)
  - rich geo client
  - assembles WCPS queries, wrapping into visual functionality
- rasdaman (Jacobs U, rasdaman GmbH)
  - WCPS over various protocols
  - petascope for request translation and geo semantics resolution
  - rasdaman returns images (or scalars)



[ESA VAROS project, 2010]

Sample WCPS-Based C/S Architecture

JACOBS UNIVERSITY



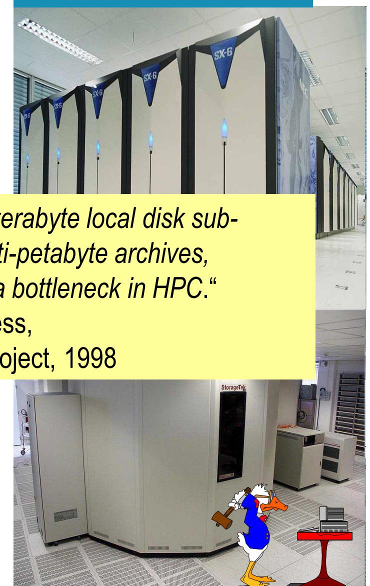
[ESA VAROS project, 2010]

Climate Modelling

- Example: ECHAM T42 (cf. video)
- 50+ physical parameters („variables“): temperature, wind speed x/y, humidity, pressure, CO_2, ...
- 2.5 TB per variable

dimension	extent
Longitude	128
Latitude	64
Elevation	17
time (24 min per time slice)	2,190,000 (200 years)

DKRZ: 24-node NEC SX-6

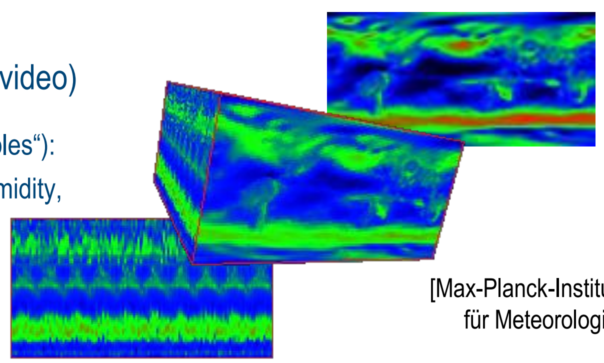


„Even with multi-terabyte local disk sub-systems and multi-petabyte archives, I/O can become a bottleneck in HPC.“
-- Jeanette Jenness,
LLNL, ASCI-Project, 1998

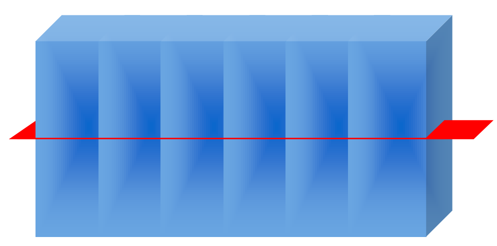
Climate Modelling

- Example: ECHAM T42 (cf. video)
- 50+ physical parameters („variables“): temperature, wind speed x/y, humidity, pressure, CO_2, ...
- 2.5 TB per variable

dimension	extent
Longitude	128
Latitude	64
Elevation	17
time (24 min per time slice)	2,190,000 (200 years)



[Max-Planck-Institut für Meteorologie]



EarthServer: *Big Analytics on Big Data*

JACOBS UNIVERSITY

www.earthserver.eu

- **Mission:** to enable standards-based ad-hoc analytics on the Web for Earth science data
  - scalable to Petabyte/Exabyte volumes
  - directly manipulate, analyze & remix any-size geospatial data
- **Core idea:** integrated query language for all spatio-temporal coverage data
- **Goal:** to establish OGC standards based client & server technology
- **Funded by** EU FP7-INFRA
  - **Started** Sep 1, runtime 3 years, 5.38m EUR budget, 11 partners

EarthServer Lighthouse Applications

JACOBS UNIVERSITY

www.earthserver.eu

- 100+ TB per site, accessible for direct analytics
- front-end to existing archives - no new archives

EO

snow & land ice

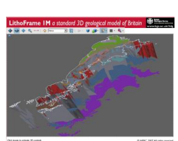
x/y + x/y/t

EOX, esa, NASA

Geology

3D geological models

x/y + x/y/z

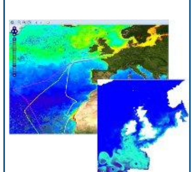


British Geological Survey, Natural Environment Research Council

Oceanography

EO + marine model runs + in-situ

x/y + x/y/z + x/y/z/t

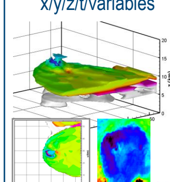


PML Plymouth Marine Laboratory

Meteorology

climate variables

x/y/z/t/variables

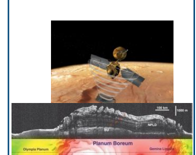


MEEO Meteorological Environmental Earth Observation

Planetary Sci

Mars geology

x/y + x/z + y/z



JACOBS UNIVERSITY

EarthServer: Main Innovations

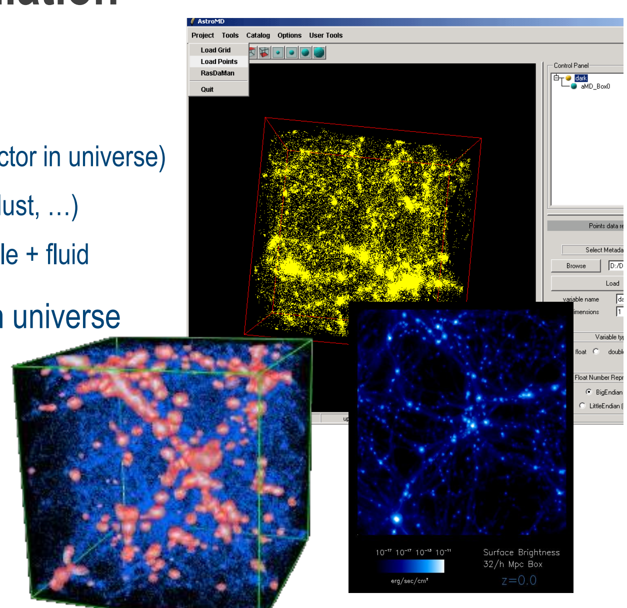
www.earthserver.eu

- Integrated coverage, feature, and metadata queries, including all OGC coverage types
- Transparent queries over heterogeneous file archives and databases
- Paving the way for Petabyte services:
  cloud distribution, parallelization, supercomputers
- Comprehensive OGC standards support for coverage data and services

- Vision: *barrier-free „mix & match" access to multi-source, any-size geo data*

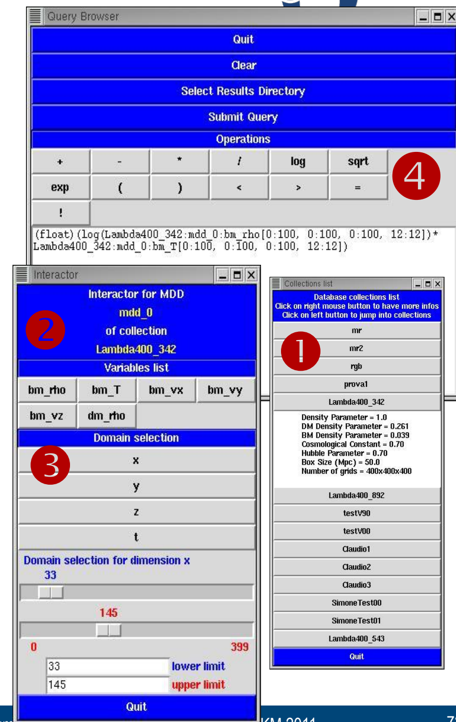
Cosmological Simulation

- Modelling domain: 4D
  - Dark matter (highest mass factor in universe)
  - Baryonic matter (stars, gas, dust, …)
  - → Coupled simulation: particle + fluid
- Results: 3D/4D cutouts from universe
  - Eg, 64 Mpc^3
    (1 pc = 3.27 light years)
- Screenshots: AstroMD
  [Gheller, Rossi 2001]



Cosmology (contd.)

- Guided retrieval:
  - Selection of objects ❶ and their attributes (cell components) ❷
  - interactive setting of trim operations per dimension ❸
  - Augmented with induced operations ❹
- Suitable for expert users
- Details: cosmolab.cineca.it



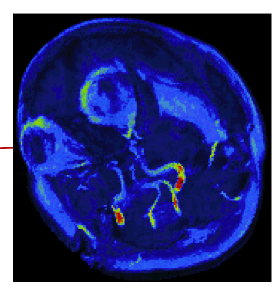
Human Brain Imaging

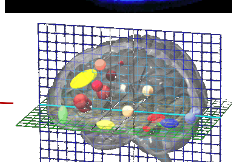
JACOBS UNIVERSITY

- Research goal: to understand structural-functional relations in human brain
- Experiments capture activity patterns (PET, fMRI)
  - Temperature, electrical, oxygen consumption, ...
  - → lots of computations → „activation maps“
- Example: “*a parasagittal view of all scans containing critical Hippocampus activations, TIFF-coded.*“

```
select tiff( ht[ $1, *:*, *:* ] )
from   HeadTomograms as ht,
       Hippocampus as mask
where  count_cells( ht > $2 and mask )
       / count_cells( mask )
       > $3
```

$1 = slicing position, $2 = intensity threshold value, $3 = confidence

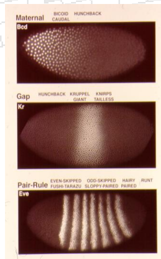


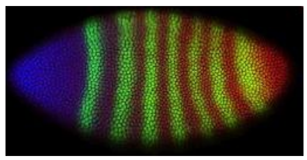


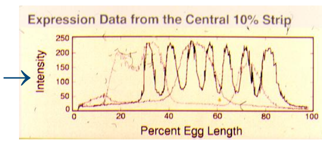
Gene Expression Analysis

http://urchin.spbcas.ru/Mooshka/ [Samsonova et al]

- Gene expression = reading out genes for reproduction
- Research goal: capture spatio-temporal expression patterns in Drosophila



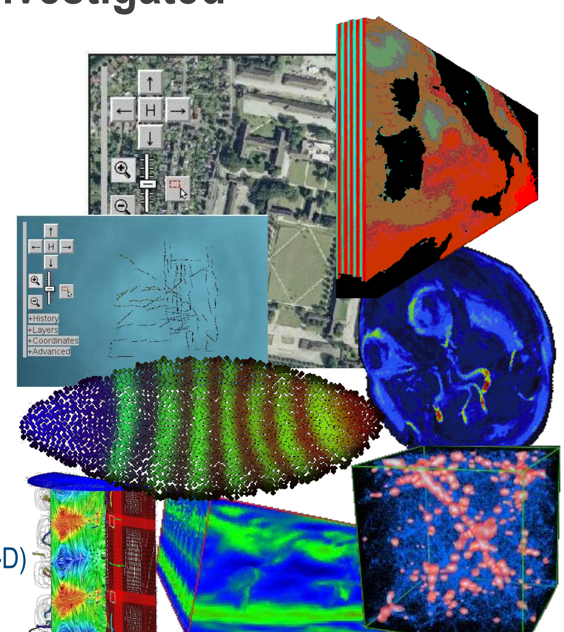





```
select jpeg( scale( {1c,0c,0c}*e[0,*:*,*:*]
                   +{0c,1c,0c}*e[1,*:*,*:*]
                   +{0c,0c,1c}*e[2,*:*,*:*], 0.2 ) )
from EmbryoImages as e
where oid(e)=193537
```


Summary: Domains Investigated

- Geo
  - Environmental sensor data, 1-D
  - Satellite / seafloor maps, 2-D
  - Geophysics (3-D x/y/z)
  - Climate modelling (4-D, x/y/z/t)
- Life science
  - Gene expression simulation (3-D)
  - Human brain imaging (3-D / 4-D)
- Other
  - Computational Fluid Dynamics (3-D)
  - Astrophysics (4-D)



Related Work

[image: Chronicle/Michael Macor]

Related Work: Brief History

JACOBS UNIVERSITY

- Image partitioning, API access library [Tamura 1980]
- Fixed set of imaging operators [Chang, Fu 1980; Stucky, Menzi 1989; Neumann et al 1992]
  - scaling, rotation, edge extraction, thresholding, ...
- PICDMS [Chock, Cardenas 1984]
  - stack of images (identical resolution); operations corresponding to rasql "induced" ops; no nesting; no architecture
- rasdaman [Baumann+ 1991+]: algebra, QL, architecture
- „Call to order“ [Maier 1993]
- AQL, AML, MQL: conceptual models
- Sarawagi/Stonebraker: tertiary storage
- ESRI, Oracle; Google, Microsoft, ...
  - Mostly Geo (Remote Sensing), some Space, practically no Life Science motivation
- TerraLib, MonetDB, SciDB, …

↳see next

Related Work: Systems

JACOBS UNIVERSITY

- Oracle GeoRaster
  - 2D, no QL integration

```
declare
   g sdo_georaster;
   b blob;
begin
   select raster into g
   from uk_rasters
   where id = 4;
   dbms_lob.createTemporary(b,true);
   sdo_geor.getRasterSubset(
       georaster => g,
       pyramidlevel => 0,
       window =>

   sdo_number_array(0,0,699,899),
       bandnumbers => '0',
       rasterBlob => b);
end;
```

```
select g.green[0:699,0:899]
from   uk_rasters as g
where  oid(g) = 4
```

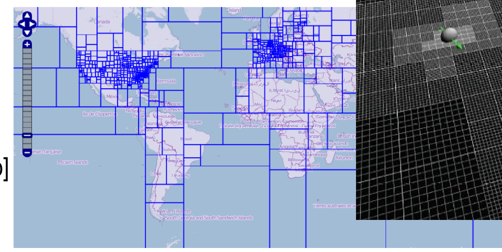

Related Work: Systems

JACOBS UNIVERSITY

- Oracle GeoRaster
  - 2D, no QL integration
- PostGIS Raster
  - Excellent QL integration
  - 2D, no tile management, no storage layout tuning, no adaptive tile streaming, no raster query optimization; utilizes small tiles, ... scalability?
- MonetDB (column store DBMS)
  - n-D arrays under development
  - Arrays as first-class citizens – array similar to table
- SciDB
  - n-D arrays announced, components demoed, under development
  - Mingles logical with physical aspects on QL level

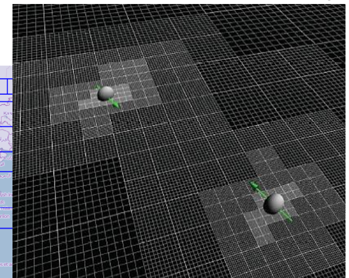
Related Work: Tiling

- Partitioning common in imaging & geo data
  - Tiling, mosaicking, ...
  - e-Science often uses irregular partioning



[OpenStreetMap]

[Centrella et al: scidacreviews.org]



- Array databases
  - Regular „chunks“ [Stonebraker, Sarawagi 1996], refined by [Rotem et al 2008]
  - Also regular: TerraLib [Vinhas+ 2007], MonetDB [Ballegooj+ 2005], PostGIS Raster [Racine 2010], ESRI ArcSDE, Oracle 11g
  - SciDB [Cudre-Maroux et al 2009]: 2-level approach, regular chunking, redundancy
  - rasdaman [Baumann 1994, Furtado+ 1999]: arbitrary partitioning

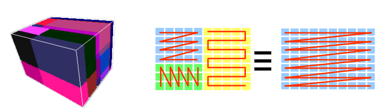
Related Work: Applications

- MS SQL Server / SDSS SkyServer [Gray et al,]
  - Recently: MonetDB / SDSS SkyServer [Ivanova et al, DBDBD 2007]
  - Emphasis on point objects and proximity queries, no arrays in “top 20 queries”

The Big Picture

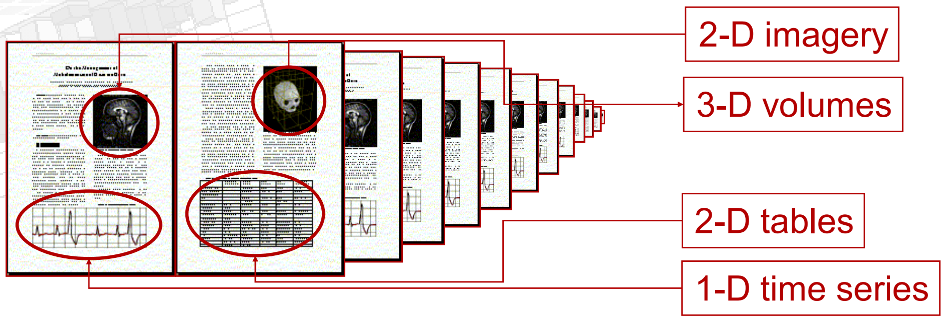
JACOBS UNIVERSITY

- Large-scale array services important + growing field
  - Currently driven by geo services
  - Largely neglected challenge to databases
  - largest single DB objects ever!
- Service providers & users demand it
  - "2D, 3D imagery next great challenge in geo databases" [Xavier Lopez, Oracle]
- Can translate most features from alphanumeric databases (and benefit):
  - Declarative, optimizable query language
  - formal semantics definition
  - Suitable storage architecture



- Many open issues, such as:
  - what expressive power? Primitives?
  - architecture
  - optimization
  - standardized benchmarks

Use Case: Reverse Lookup



*„all clinical trials of drug X
where patient temperature > 40º C within the first 48 hours.“*

Conclusion

- Array databases form nucleus for large-scale scientific data analytics
  - n-D arrays found in earth, space, life sciences, business, ...
  - Emerging „next wave“ – cf XLDB, Array Databases workshop @ EDBT/ICDT (www.rasdaman.com/ArrayDatabases_Workshop)
  - Our research: flexible, scalable raster services & beyond
    - *www.rasdaman.org, www.earthlook.org*
- DB technology can contribute significantly,
  - Flexibility, scalability, information integration, ...
- ...but must transcend traditional (table-driven) viewpoints
  - QL primitives, architectures, ...