SAF Data Model Implementation and Application

Mark C. Miller
Lawrence Livermore National Laboratory

Outline

An Abstract Math Model for Scientific Data

- *What is a Data Model?*
- *Introduction to the data model upon which SAF is based*
- *Numerous Examples and Discussion*

SAF: A Scalable/Parallel Implementation

- *limitations of current implementation*
- *strengths of current implementation*
- *future directions*

Applications In Real Physics/Viz Codes

- *Ale3d*
- *Visit*

Summary

What is a Data Model?

It is a formal, conceptual tool for describing data, data relationships, data semantics and consistency constraints[1]...

...in a representation independent manner

What do we mean by representation independence?

- *Free of implementation details such as...*
  bit and byte-order, floating point format, precision, coordinate systems, element-types, interpolation methods, processor decompositions, etc., etc.
- *That is not to say that these details aren't essential. They are!*
- *They are just not the focus of a data model*

In truth, representation independence falls on a spectrum

- *Whats one person's representation is another person's abstraction*

As a conceptual tool, a data model is always evolving!

1.Paraphrased from Silbershatz, Korth and Sudarshan, "Database System Concepts"

Do products like netCDF, HDF4, Exodus II, or Silo use a data model?

Not in the abstract sense I have defined the term here

- *The do not satisfy the degree of representation-independence I seek*
- *In fact, they are all, to varying degrees, representation dependent*

In lieu of a data model, these products employ a data menu

- *If you're lucky, your data matches exactly an object on the menu*
- *If you're unlucky, you have to "cast" your data into an object on the menu*
- *If you're really unlucky, all you can do is write an "opaque" blob of bytes*

Whats wrong with "cast"ing or writing an opaque blob?

- *A "cast" can be costly both in performance and loss of data integrity*
- *An opaque blob means that nothing but the writer understands the data*
- *Answer: Its the effect on your ability to utilize other tools to process your data*

Data Menu VS. Data Model

Data Menu		Data Model	
Strength	Shortcomings	Strengths	Shortcomings
More Development Experience	Having to "Cast" Data	No "casts"	Less Development Experience
Objects are Familiar	Fragile Extensibility	Robust Extensibility	**Abstract Math is Unfamiliar**
Simplicity of Design/Interface	Ever Expanding Menu	Small set of Primitive building blocks	Complexity of Design/Interface
	Partially Self-Describing	Wholly Self-Describing	
	Ad-hoc Interoperability	**Formal Interoperability**	
Each new and slightly different kind of data often requires a new object		**Each new and slightly different kind of data involves applying the same building blocks to form a slightly different assembly, to literally *model* data**	

Why is a Data Model So Important?

Because the manner in which data is described governs completely the generality of tools that can be developed to process it

In fact, a data model has far more compelling consequences for the software we write than the data we generate

- *Design of a data model is driven by its impact on the generality of software development it enables*

It Directly Effects Data Shareability & SW Interoperability

What Kind of Data Do We Need To Model?

Scientific Data: Simulated as well as Experimental

Two Broad Categories of Scientific Data[1]

- *Results Data*
- *Control Data*

Examples of Results Data

- *large arrays of floats/ints, etc, "nodal" and "zonal" variables, coordinates, time histories, load curves, material subsets, processor subsets, slide surface boundaries, etc.*

Examples of Control Data

- *version of code that generated the data, which materials are advecting, which algorithm to use for transport, name of user who ran the problem, etc.*

1. Borrowed this terminology from Doug Speck, LLNL. Don't know if its used in wider contexts

More on Results Data VS. Control Data

All sharing requires common conventions

- *Will those conventions be well known and formal or little known and ad-hoc?*

Results Data is shareable by formal models

- *Inherently mathematical in nature*
- *Mathematics is universal*
- *Can utilize well known, formal mathematical concepts as the "convention"*
- *Characterized as Discrete and Continuous Fields*

Control Data is shareable by ad-hoc conventions

- *No inherently unifying abstraction*
- *Just have to pick a convention*
- *Characterized as list of parameter/value pairs*

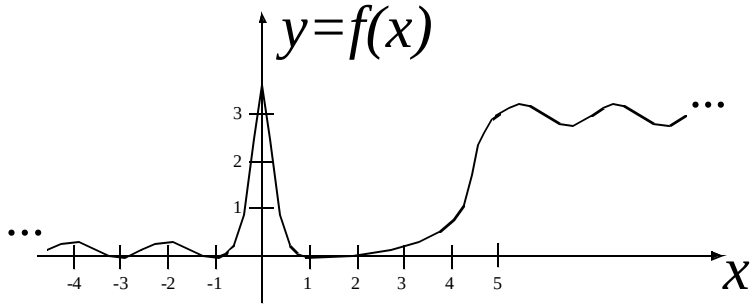
Will be focusing on Results Data

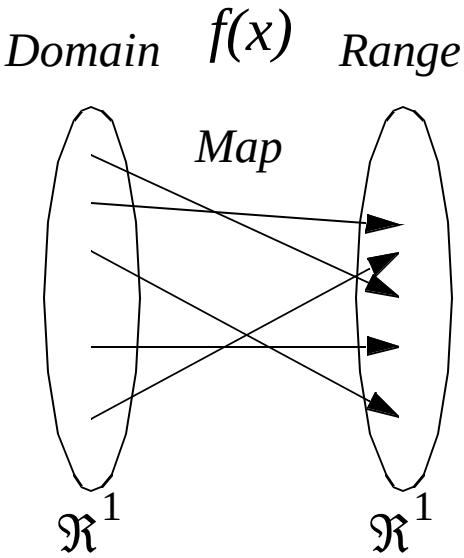
Results Data are *Fields*

The concept of a field serves as the standard for real (or imagined) observables in the real, continuous world

In terms of scientific computing, fields are the dependent (independent too) variables in a simulation

Fields Are a lot Like Functions





Three Logical Parts: Domain, Range, Map

We write

$$\Re^n \overset{f}{\Rightarrow} \Re^m$$

- *where n and m are the "dimensions" of the domain and range*

From Functions to Fiber-Bundles[1]
The Three Parts of a Field

Base-Space (~domain):

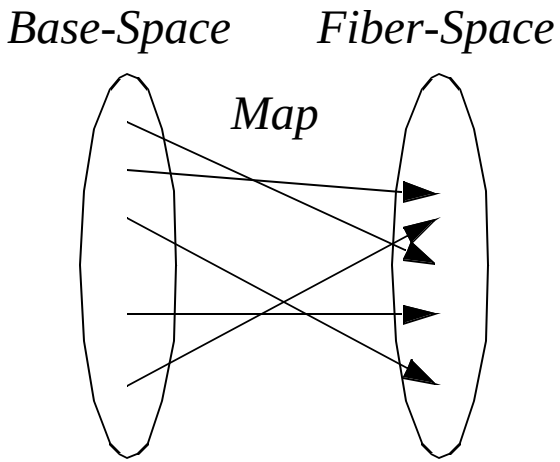
- *an infinite point set with a dimension called the "base-dimension"*

Fiber-Space (~range):

- *a linear vector space with a dimension called the "fiber-dimension"*

Map:

- *relates points in the base-space to points in the fiber-space*



1.Butler, D.M., Peldley, M.H., "A Visualization Model Based on the Mathematics of Fiber Bundles", Computers in Physics, V3 N5, pp. 45-51, 1989

Why the Abstract Math?

Our scientific computing applications employ it internally!

Also, work around limitations of functions

- *Single value at a point*
- *Inflexibility of domain topology*

Base-space and fiber-space are kept totally distinct

- *nodal connectivity, processor decomposition, element blocks, boundaries, slides are all describable in terms of the base-space, alone!*
- *Very powerful and useful*

Another "Benefit": Culture Shock

- *New terminology works to an advantage*
- *Eliminates biases and overloaded terminology in current culture*
- *Everyone learns a new language, together*

I am an Engineer
Not a Mathematician

Original notions put fourth in 1989 were visionary

- *But, there is a large gap between concept and application*

I Have a lot of experience applying the concepts

Also, rely heavily upon various references...

- *Baum, J. D., "Elements of Point-Set Topology", Dover Publications Inc., 1991.*
- *McCarty, G. "Topology: An introduction with Application to Topological Groups", McGraw Hill, 1967*
- *Berge, Claude, "Topological Spaces", Dover Publications Inc, 1997*
- *Butler, D. M., Pendley, M. H., "A Visualization Model Based on the Mathematics of Fiber Bundles", Computers in Physics, V3 N5, pp. 45-51 (1989).*
- *Butler, D. M., Bryson, S., "Vector Bundle Classes Form Powerful Tool for Scientific Visualization", Computers in Physics, V6 N6, pp. 576-584 (1992).*
- *Butler, D. M., "ASCI Data Models and Formats Committee Array API Specification" aka "VB-1.0", working draft, Limit Point Systems and Sandia National Laboratories, (1997).*
- *Butler, D. M., various consultations on the ASCI-DMF project (1997-1999)*
- *Cheney, E. W., "Introduction to Approximation Theory", McGraw-Hill, 1966.*

I won't be able to answer a lot of detailed math questions

Summary

A data model...

- *...is a conceptual tool for describing data in a representation independent manner*
- *...has more consequences for the software we write than the data we generate*

Two broad categories of scientific data

- *Results data: is inherently mathematical and can employ a formal math model*
- *Control data: is amorphous and must employ ad-hoc conventions*

We will be focusing on results data as *fields*

- *concept of a field serves as the standard for real (or imagined) observables*

Will use fiber-bundle data model to characterize fields

- *A fiber-bundle is a lot like a function*

Outline

Simple example

- *Introduce diagramming technique and some terminology*

Formalize the terminology with numerous examples

Will apply the model with some examples

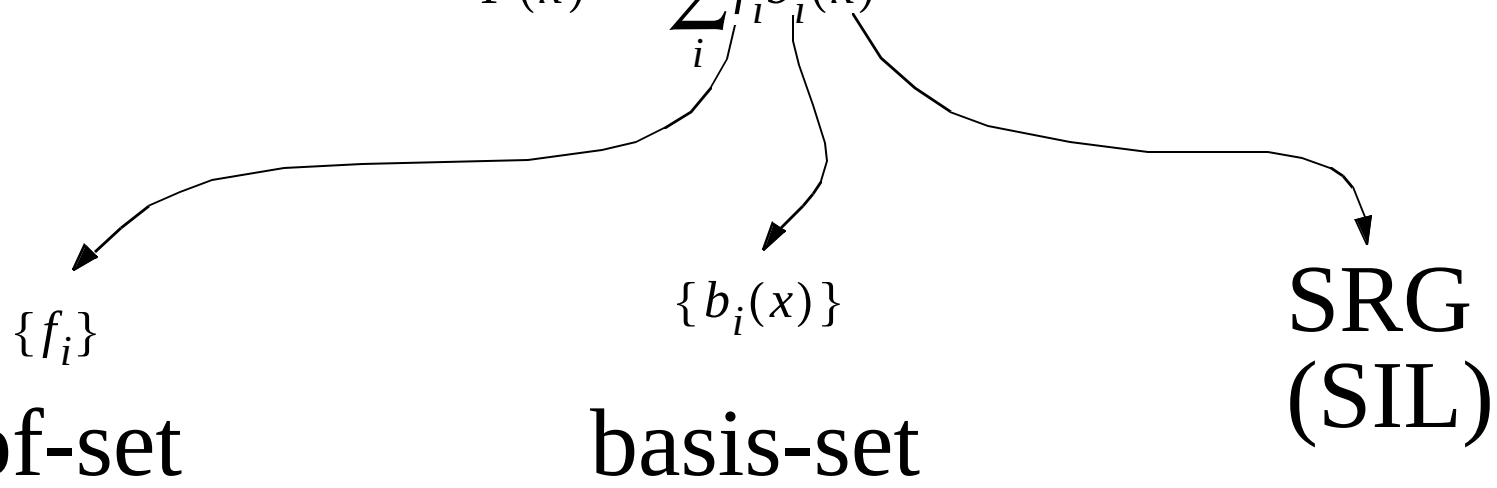
Will point out differences between the conceptual model and its current implementation in SAF

Most important key point

- *With a few basic principles, can apply and re-apply a handful of primitives to characterized (e.g. model) a wide variety of scientific data*

Continuous Fields

$$F(x) = \sum_i f_i b_i(x)$$



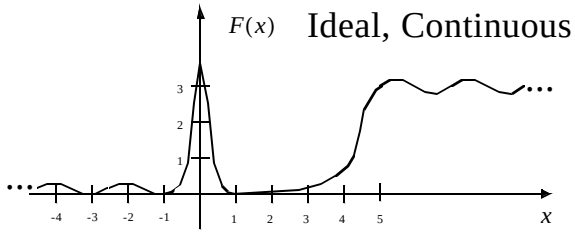
dof = "Degree of Freedom"

- *These are not "values" except when basis functions are interpolating*

SRG = "Set Relation Graph"

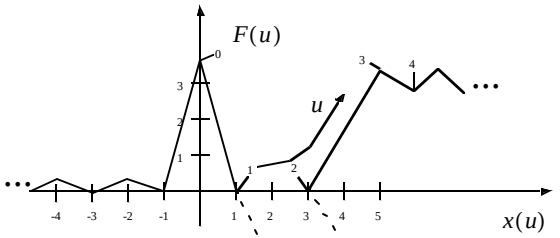
- *Will discuss more later*

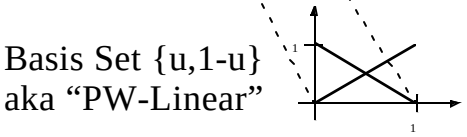
Detailed Introduction



Ideal, Continuous Field

reparameterize from *x* to *u*

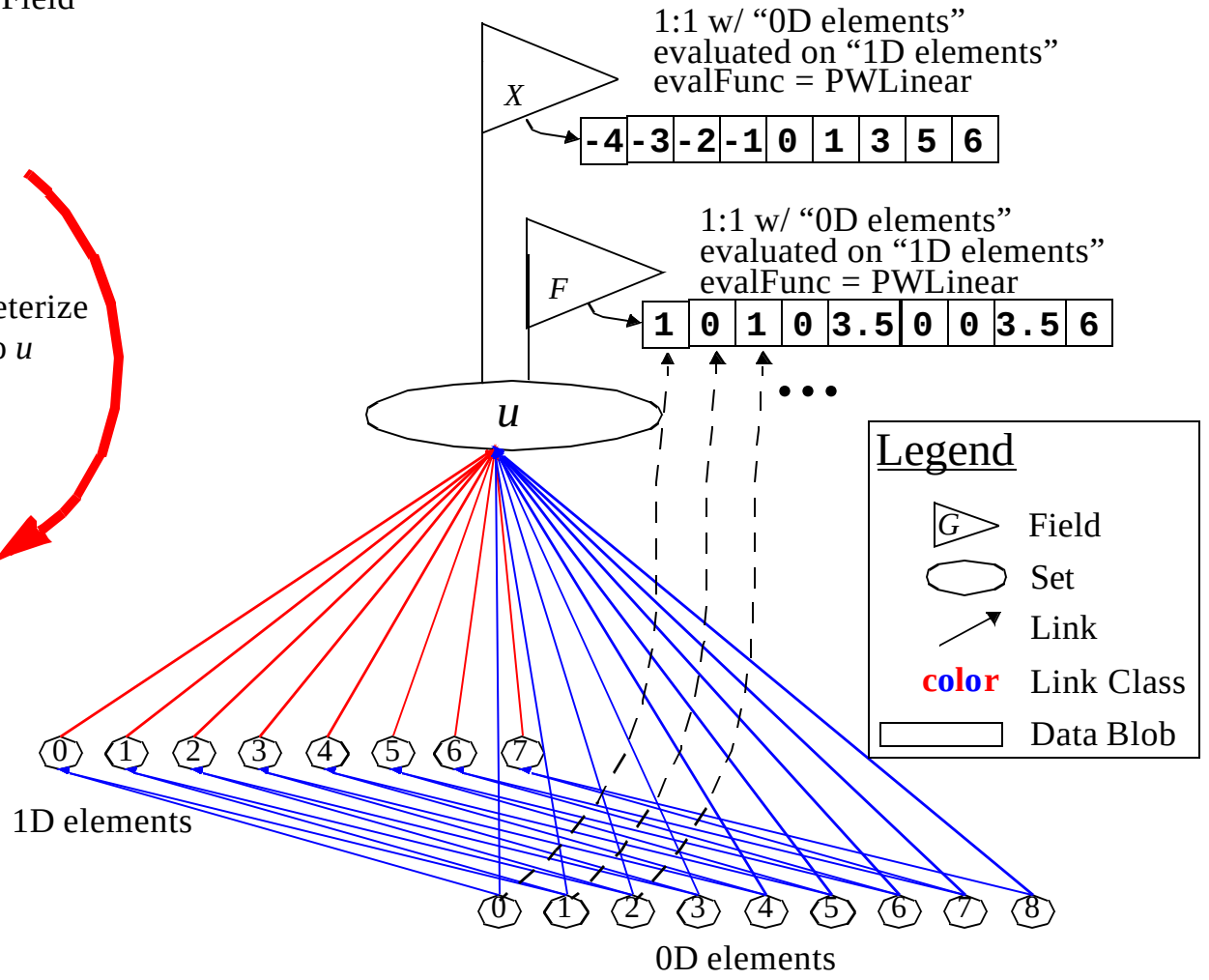




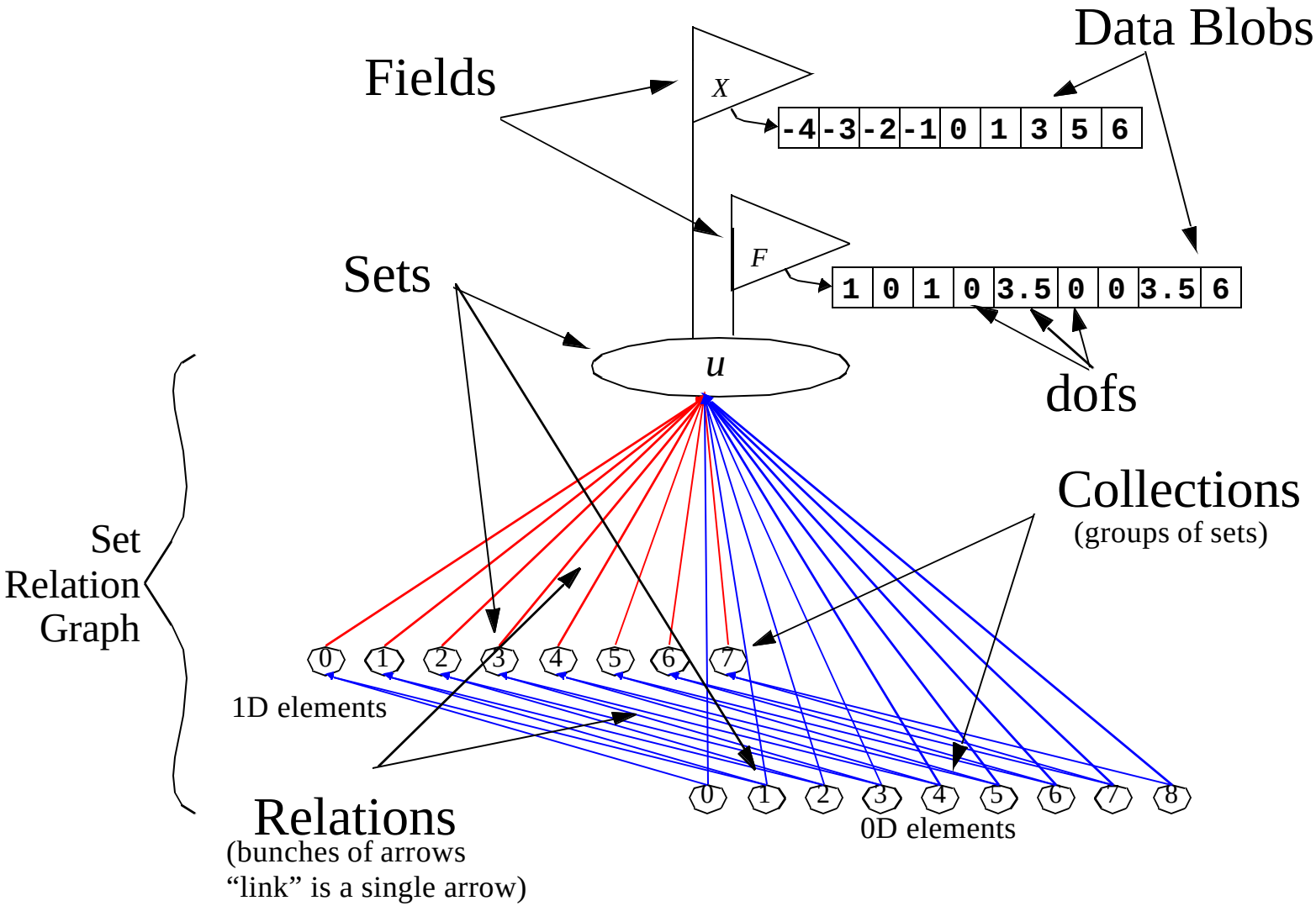
Basis Set {u,1-u}
aka "PW-Linear"

$\{f_i\}$ = {1,0,1,0,3.5,0,0,3.5,-4.5}

$\{x_i\}$ = {-4,-3,-2,-1,0,1,3,5,6}



Detailed Introduction



Some Questions...

Whats a more commonly used name for these kinds of fields?

Whats another name for the “1D” and “0D elements”?

There is something special about the field, “X”. What?

How many functions are in the basis set?

What set relationship do the links (each arrow) represent?

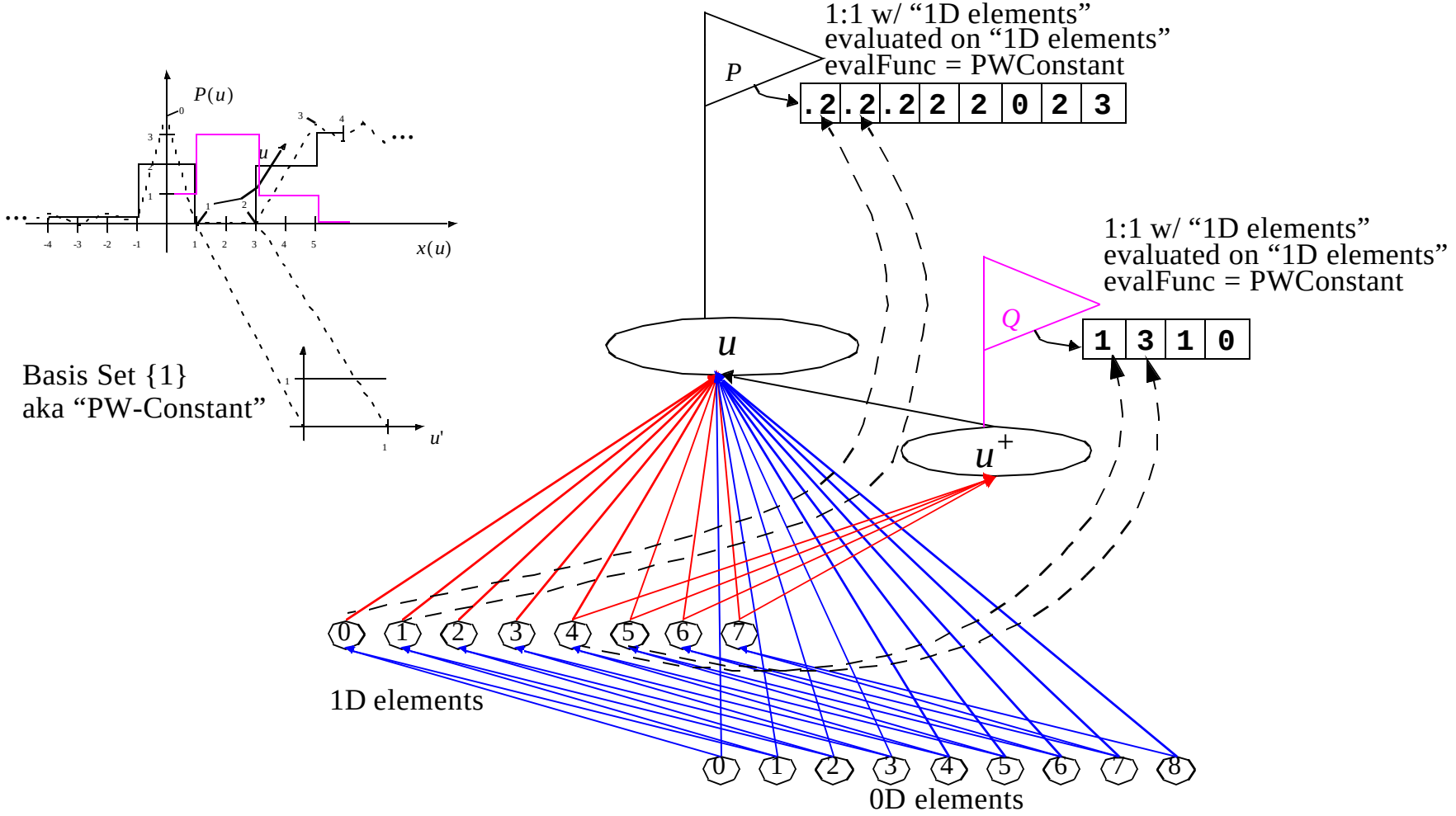
What would a “zone-centered” variable look like?

What if you wanted to define a field only for *u>0*?

What can be said about how 1D elements divide parent “u” up?

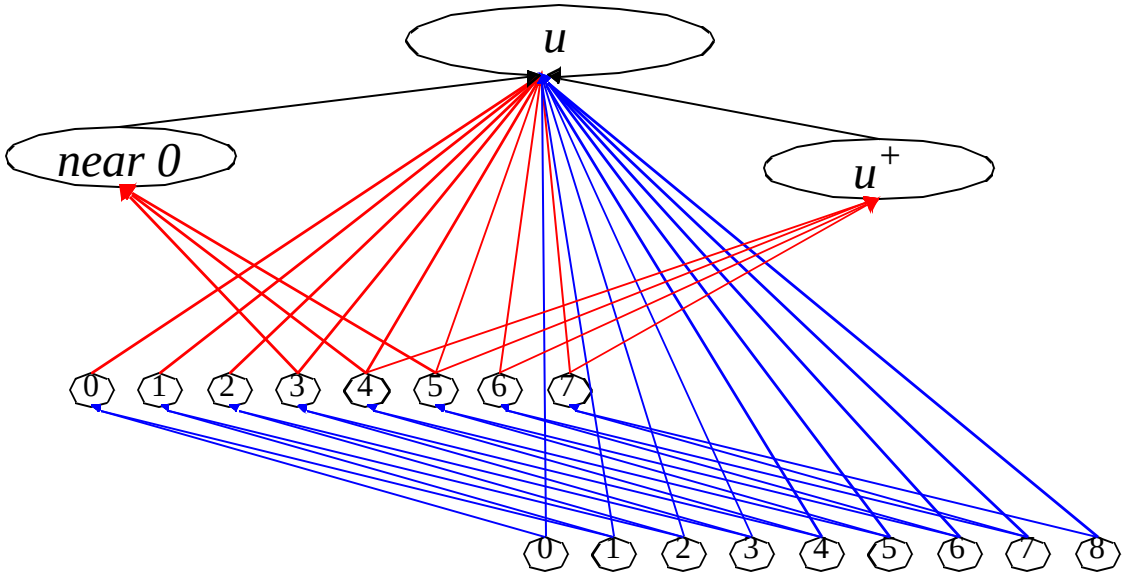
What if we strip the fields away and have only the sets (elipses) and their relationships (arrows). Is there anything useful left?

Other Fields



- *Why no "0D elements" under u^+? Do the 1D elements under u^+ decompose it?*

What if we strip away the fields

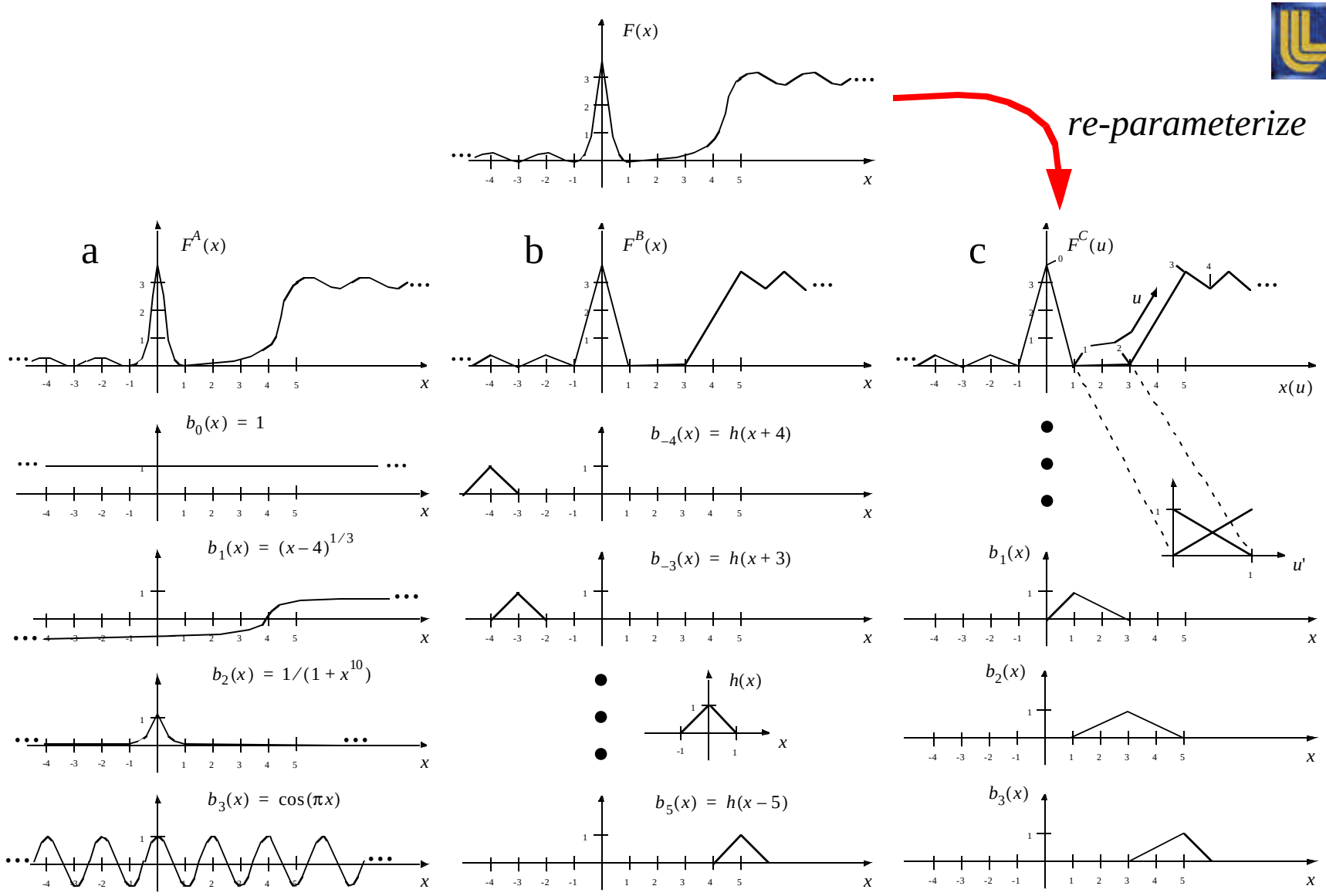


All we have is a bunch of sets/relationships. What good is that?

- *Processor decompositions, local to global maps*
- *subsetting for partial I/O and defining fields only where they're non-zero*
- *"nodal connectivities"*
- *Slide surfaces*
- *External facelists (boundaries)*

Plenty!!!

Different Basis Sets



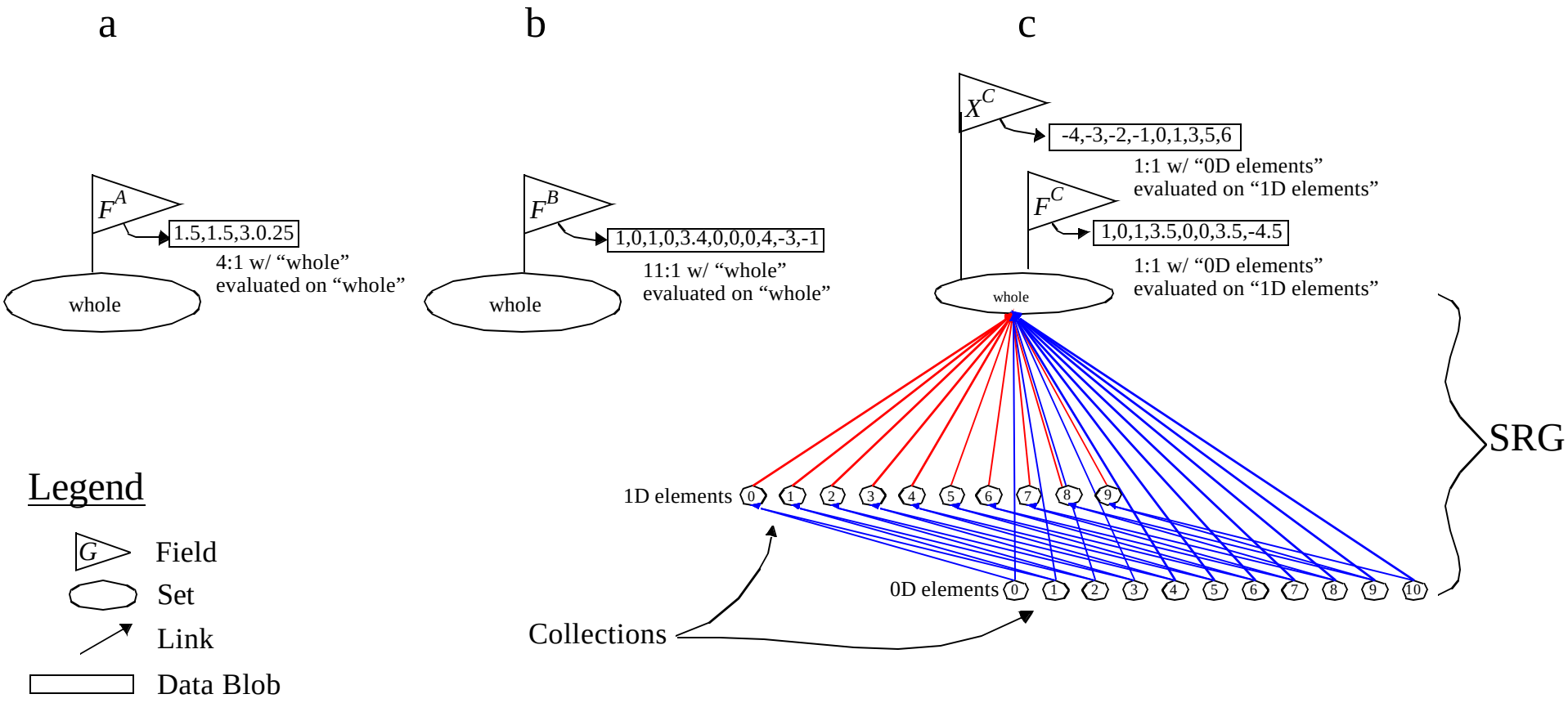
$\{f_i^B\}$ = {1.5,1.5,3,0.25}

$\{f_i^A\}$ = {1,0,1,0,3.5,0,0,0,4,-3,-1}

$\{f_i^C\}$ = {1,0,1,0,3.5,0,0,3.5,-4.5}

$\{x_i^C\}$ = {-4,-3,-2,-1,0,1,3,5,6}

Examples Diagramed

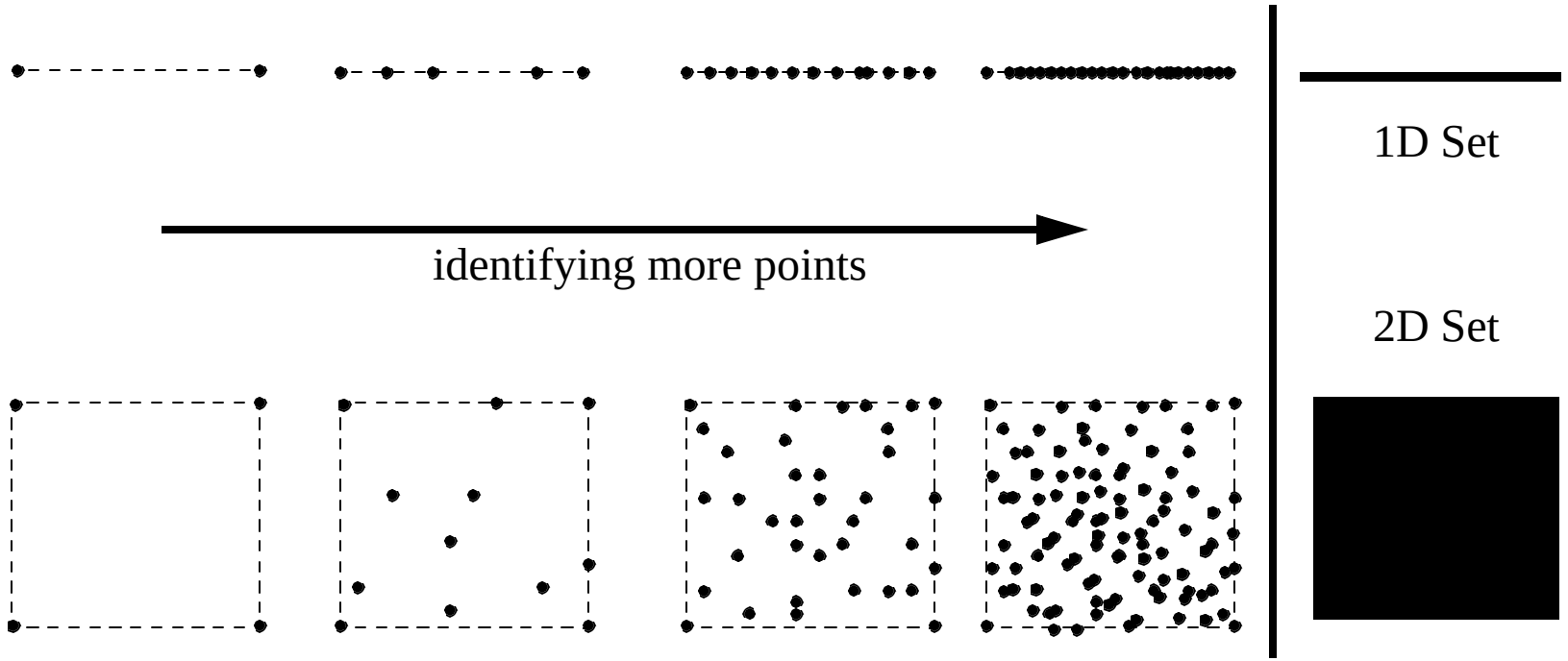


Terminology: Part 1
The Base-Space

Will be concerned solely with sets and information about relationships between sets

Set

Defn: An infinite point-set with a dimension called the base-dimension



Picture emphasizes the *infinite* nature of a set

- *Example: A quad element is NOT same as its 4 corner nodes*

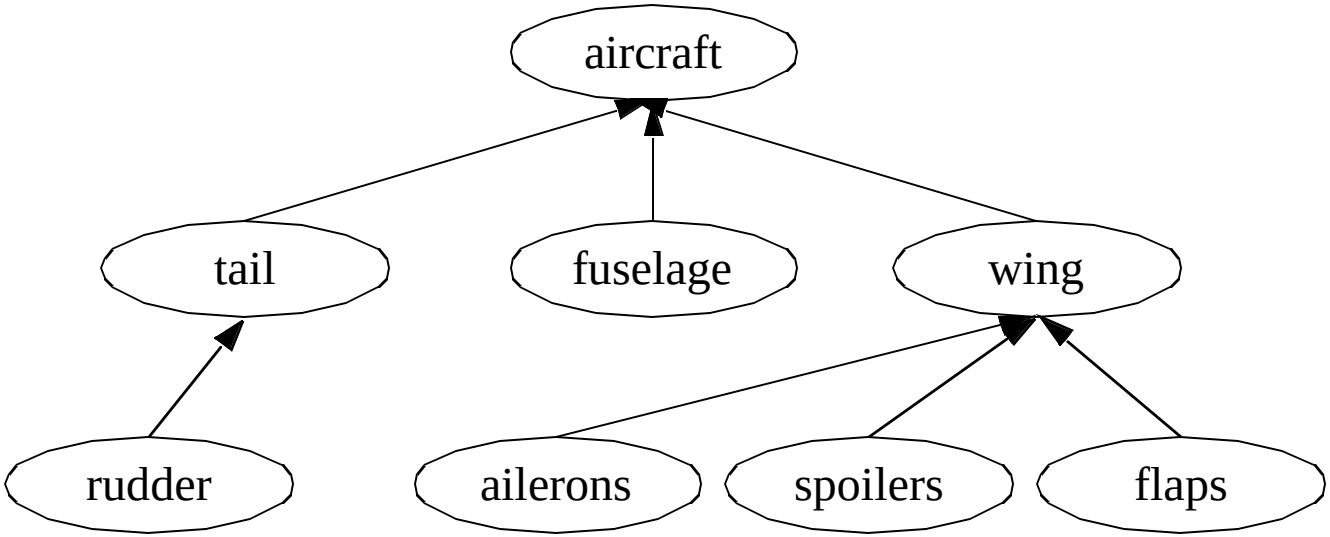
Set (cont'd)

entity	base dim
curve, line or edge	1
surface or face	2
solid or volume	3
a single point	?
a collection of discrete points	?
the empty set	-1

Diagrammatic Representation: An ellipse

Set Relation Graph (SRG)

Defn: A directed graph in which the nodes of the graph represent sets and the links represent relationships between sets



The most common relationship is "subset-of"

- *SRG's often look, more or less, like a tree*

Link Class (Stratum)

Defn: An arbitrary, sometimes client-defined, label to distinguish bunches of links used for a common purpose

- *"stratum" has more formal mathematical connotations (layers in the SRG)*

There are a number of pre-defined link classes (more later)

Diagrammatic representation (color of the arrows)

The Link Operation

Defn: The operative relationship between the sets (nodes in the SRG) at the tail and head of the arrow (link)

Operative Relationship

- *Set at the tail of the arrow is the left operand as in "L* $\longrightarrow$ *R"*
- *subset-of,* $\subseteq$
- *supset-of,* $\supseteq$
- *boundary-of, B*
- *copy-of,* $\Rightarrow$
- *neighbor-of,* $\cap \neq 0$
- *equal-to,* $\subseteq \supseteq$
- *Maybe possible to synthesize all of these from "subset-of"*

Diagrammatic representation (symbol on or near the arrows)

- *Only used when not obvious*

Subset Relation Graph (SubRG)

Defn: Portion of an SRG in which all links are subset-of

Also called a *Subset Inclusion Lattice (SIL)*

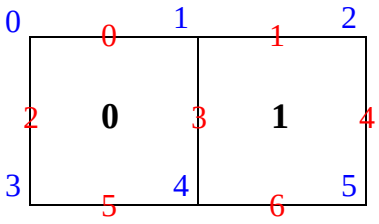
"Subset-of" tends to be the dominant link operation

SubRG looks a lot like a tree

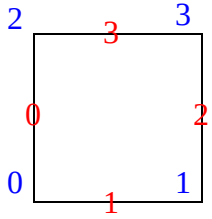
- *Use terms like "parent", "child", "traverse", "subtree" and "downset"*
- *Draw smaller sets below larger ones that include them*

Example

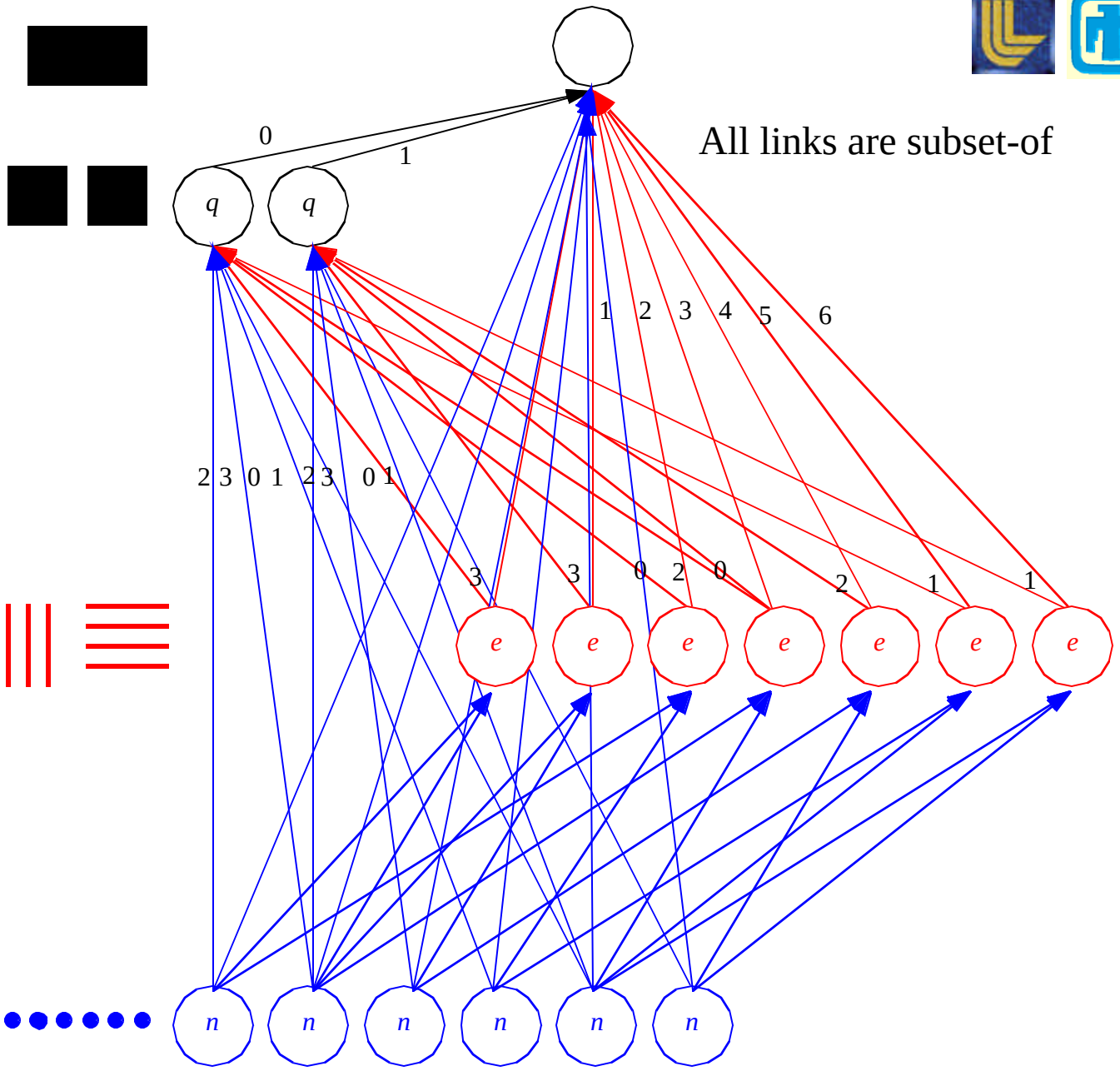
A Mesh of 2 Quads



q Template



e Template



Cover Relation Graph (CRG)[1]

Defn: A SubRG in which there are only Cover links

Cover has to do with the notion of "next largest"

Possibilities

- *minimal spanning "tree" for an SRG*
- *space optimization*
- *time optimization for operators such as union, intersection, difference*
- *a canonical representation for an SRG*

Won't discuss CRG further

- *If interested, contact David M. Butler*

1.Butler, David M., 2000, dmbutler@limitpt.com

Special Sets

Universe Set

- *Defines "scope" of a database*
- *Only one universe set in any given database*
- *All sets in the database are subsets of the Universe*

Empty Set

- *The set with no members*
- *Base dimension is -1*
- *Only one empty set in the database*
- *The empty set is a subset of every set in the database*

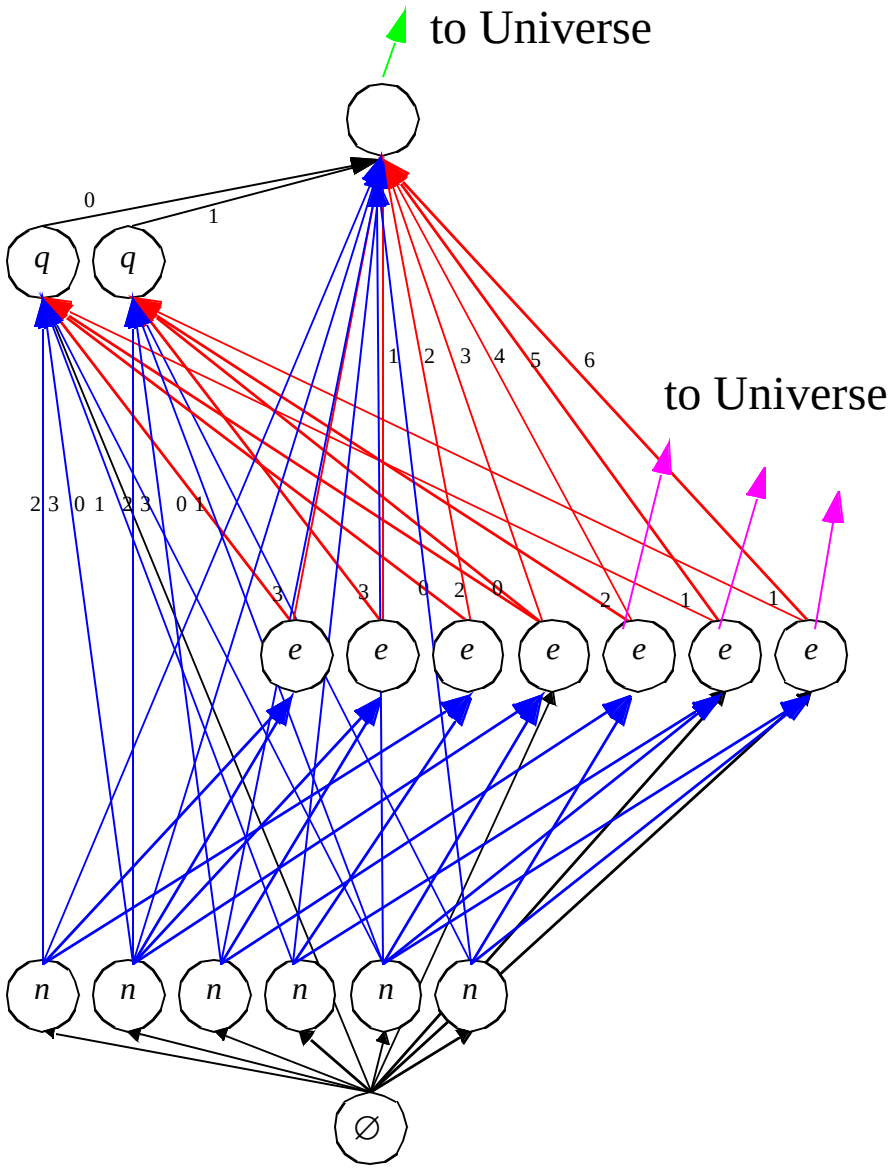
Typically, we don't bother to draw Universe or Empty Sets

- *But don't forget they are there*

Top Set

- *A set whose only parent is Universe*

Example


to Universe
0
1
q
q
1 2 3 4 5 6
to Universe
2 3 0 1 2 3 0 1
3
3
0 2 0
2
1
1
e
e
e
e
e
e
e
n
n
n
n
n
n
∅

Primitive and Aggregate Sets

Primitive Set

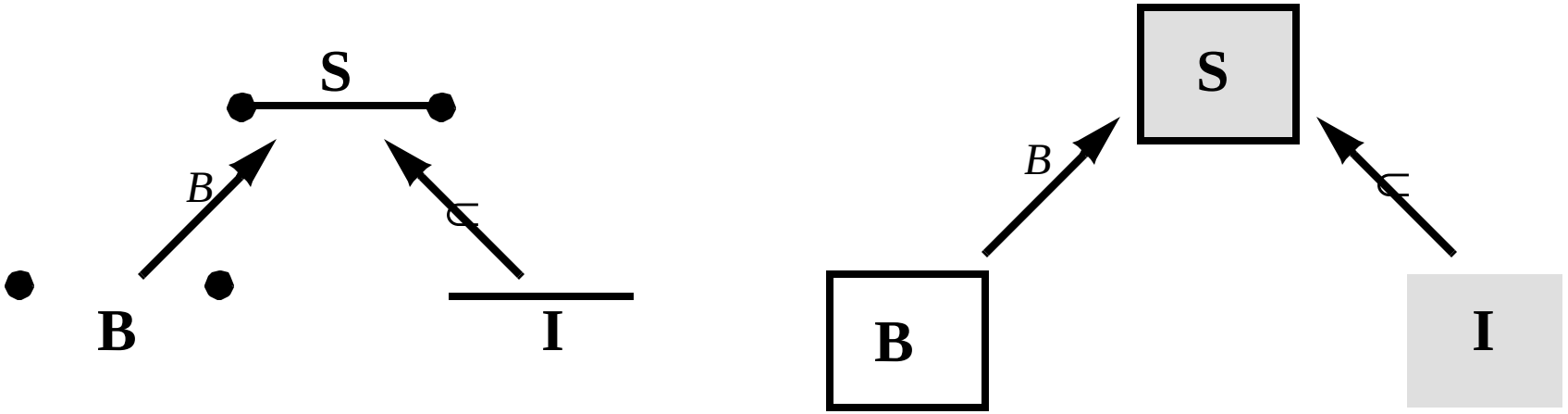
- *Defn: A set that cannot be formed by the union of other sets in the SRG*
- *Typically, these are your node, edge, quad, hex elements*

Aggregate Set

- *Defn: a set that can be formed by the union of other sets in the SRG*
- *not primitive*
- *these are processors, materials, side-sets, parts in an assembly, etc.*

Return to Example

Boundary and Inner Set



Boundary Set

- *Defn: A set which is known to be the boundary of another set*
- *Base-dimension is 1 less than that of the set it is the boundary of*

Inner Set

- *Defn: the set that results when you subtract its boundary*
- *Note:* $S = B \cup I$ *and* $B \cap I = \varnothing$
- *Also, of course, S is the closed set and I is the open set*

Summary, So Far

Set: infinity of points with a base-dimension

SRG: graph indicating relationships between sets

SubRG: SRG which has only “subset-of” relationships

Special Sets

- *Universe and Empty*
- *top*
- *primitive and aggregate*
- *Boundary and Inner*

Collections, Relations and Namespaces

Collection: Group of related sets in an SRG

Relation: Group of related links in an SRG

Namespace: Schema for indexing members of a collection

Collection

Defn: A group of sets, all related to a common parent by subset links of the same link-class. In other words, a bunch of sets, all subset of a common supset by a common link class

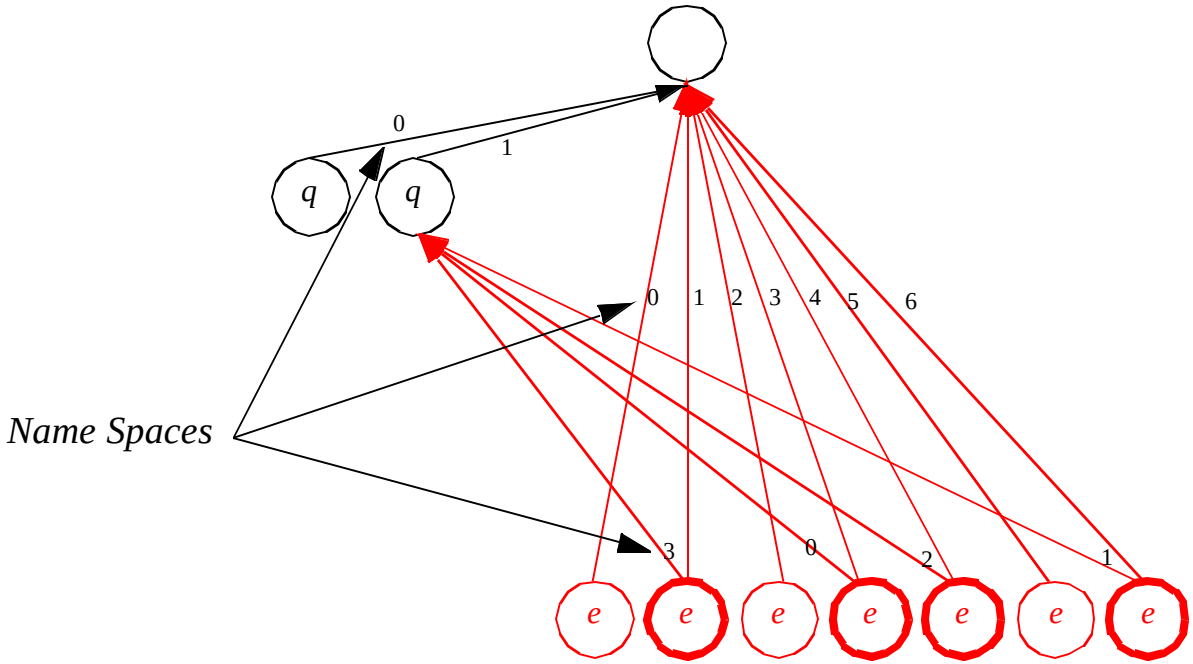
Back to the example

A collection is identified by a `<set, link-class>` pair

Name Space

Defn: Schema for indexing members of a collection

Namespaces are used to identify links involved in a collection



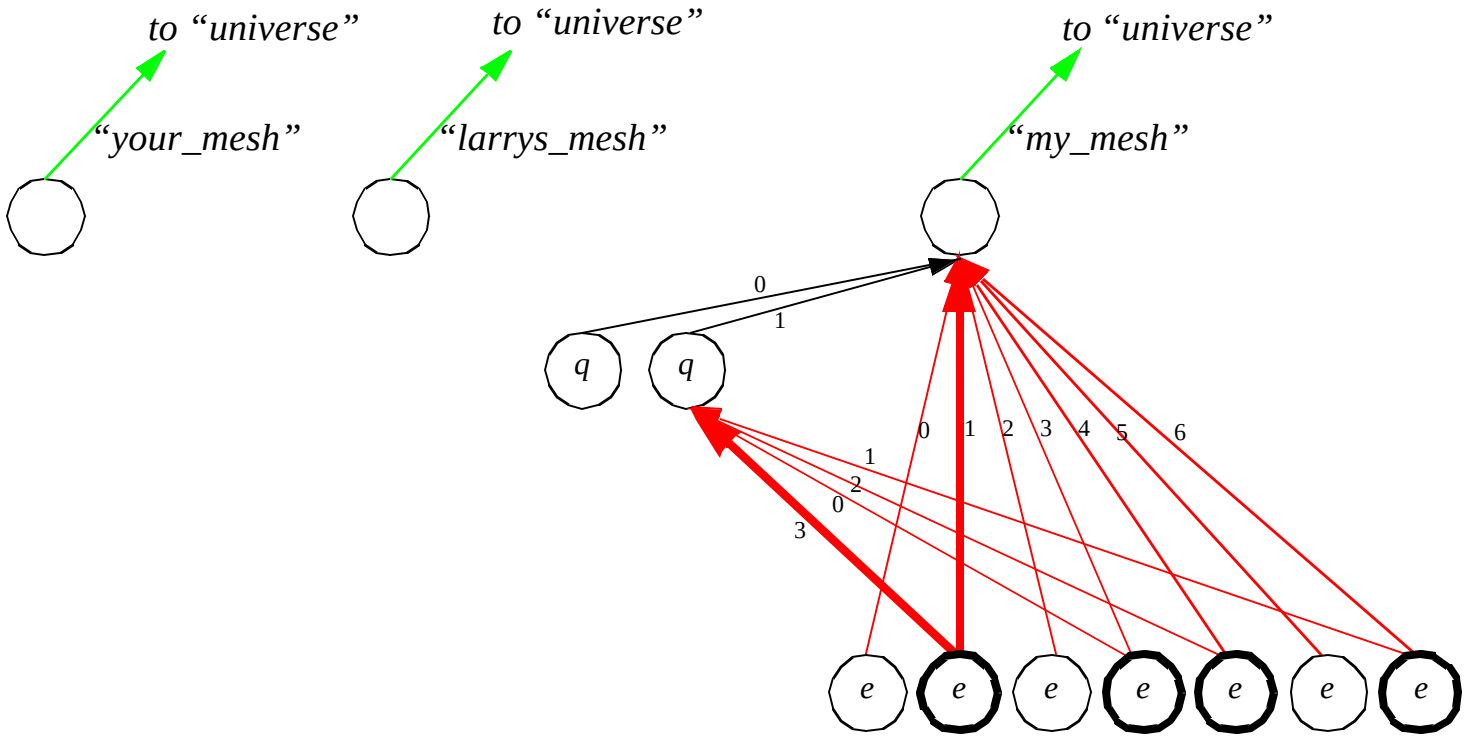
Attributes of a name-space

- *data-type (int, int-pair, int-triple, string, etc.)*
- *Sorted?, Compact?*

Pathnames for Sets

Schema: `/c-name[nsidx]/c-name[nsidx]/...`

- *like arrays of directories in a unix filesystem*
- `/tops["my_mesh"]/edges[1]`
- `/tops["my_mesh"]/quads[1]/edges[3]`
- *just as in C arrays, if you leave off brackets, you mean whole collection*



Decompositions and Partitions

Decomposition

- *Defn: A collection such that the union of the members forms their common supset*
- *Let P be the common supset and* C_i *be the ith member of the collection.*

$$P = \bigcup_i C_i$$

- *the collection is a decomposition of the supset, P*

Partition

- *Defn: a decomposition in which it is also true that*

$$I(C_i) \otimes I(C_j) = \varnothing \quad \forall (i \neq j)$$

- *where I(.) takes the "insides" of the set*

Back to example...

Relations

Defn: Groups of links in an SRG between common collections

A Relation is, in general, a list of ordered n-tuples

- *each n-tuple identifies and relates the members of n collections*
- *can be thought of as a table with n columns*

Diagrammatic Representation: an fat arrow (➡)

- *We simplify the SRG drawings a bit and use a single arrow to represent many arrows*

Binary Relation

Defn: a relation in which n is two

- *first column is called the "domain"*
- *second column is called the "range"*

Equivalence Relation: Special Binary Relation

- *Defn: A binary relation, in which each ordered pair identifies an equality relationship between the member of the domain collection and the member of the range collection*

Most aggregate sets in an SRG are defined by Binary Relations

- *Used in processor decompositions, material decompositions, etc.*

Partitioning Relation ("meshing" relation)

Defn: A relation that knits together the members of a partition

The cumulative effect is to form, logically, what we typically refer to as a *mesh*.

Every mesh always has at least one partitioning relation

- *Example: nodal connectivities*

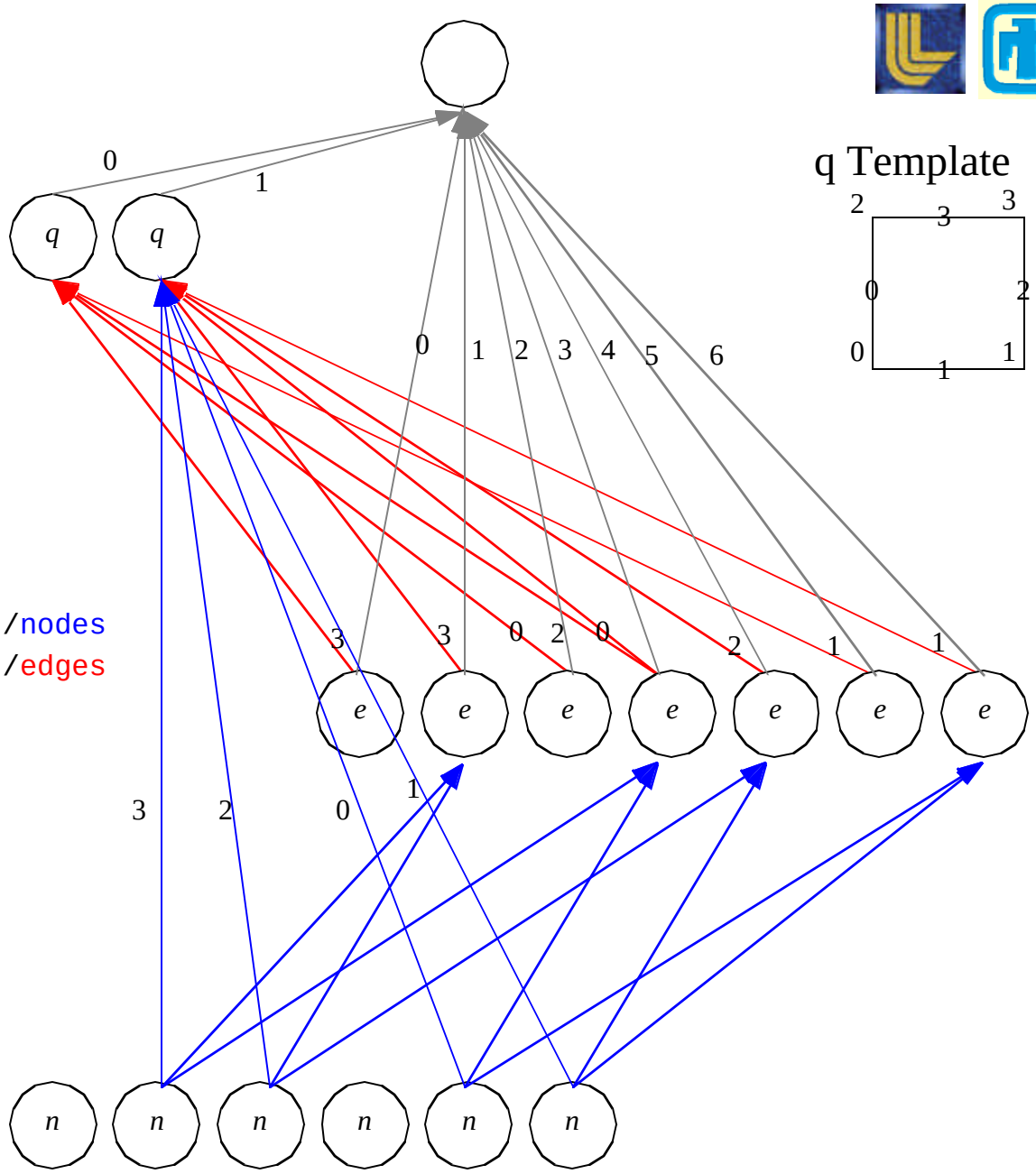
Some Relations

domain: `/tops["my_mesh"]/quads`
range: `/tops["my_mesh"]/edges`
op: supset-of

D	R	
0	2	0
0	5	1
0	3	2
0	0	3
1	3	0
1	6	1
1	4	2
1	1	3

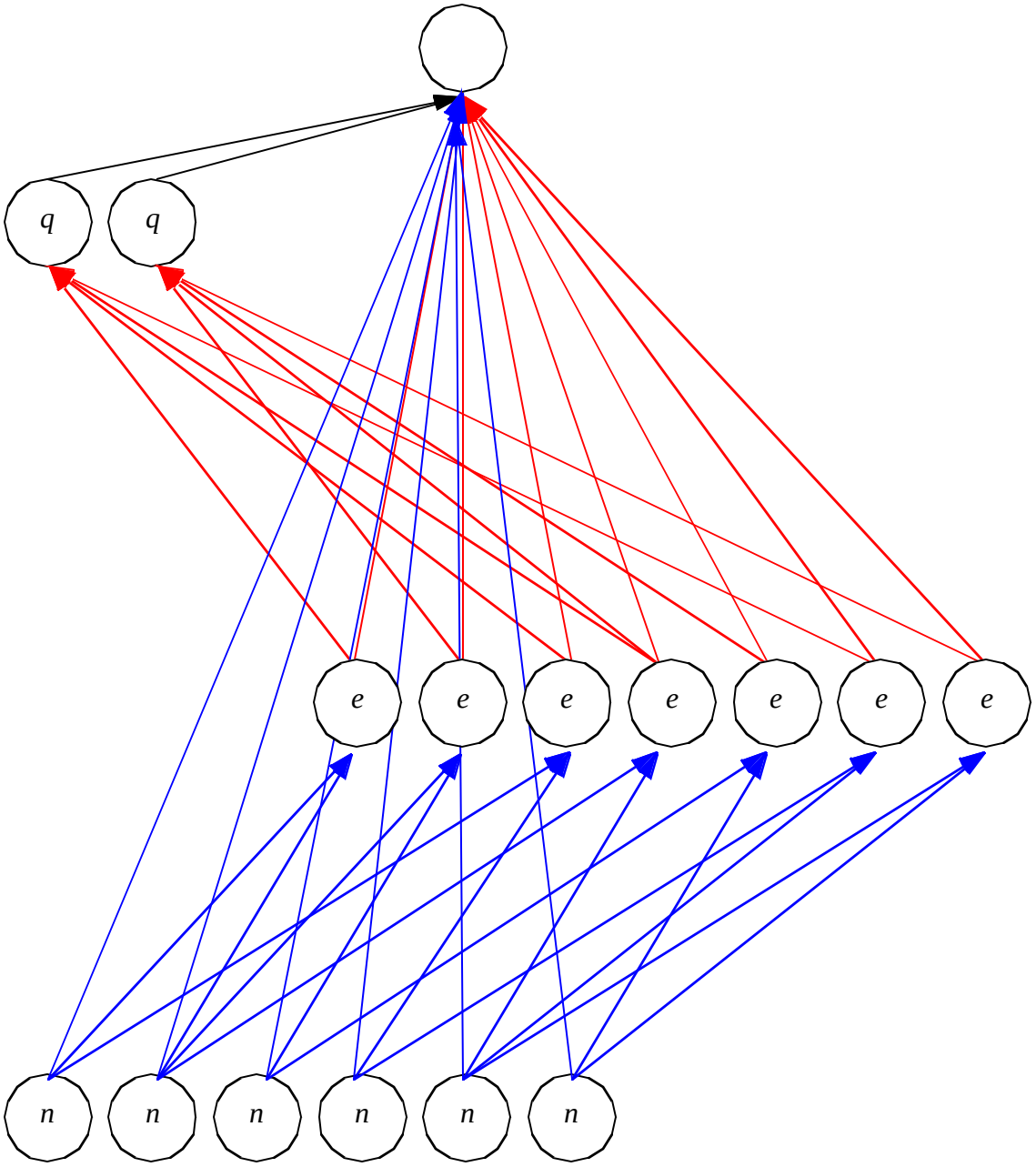
domain: `/tops["my_mesh"]/quads[1]/nodes`
range: `/tops["my_mesh"]/quads[1]/edges`
op: subset-of

D	R
0	0
0	1
1	2
1	1
2	3
2	2
3	3
3	0

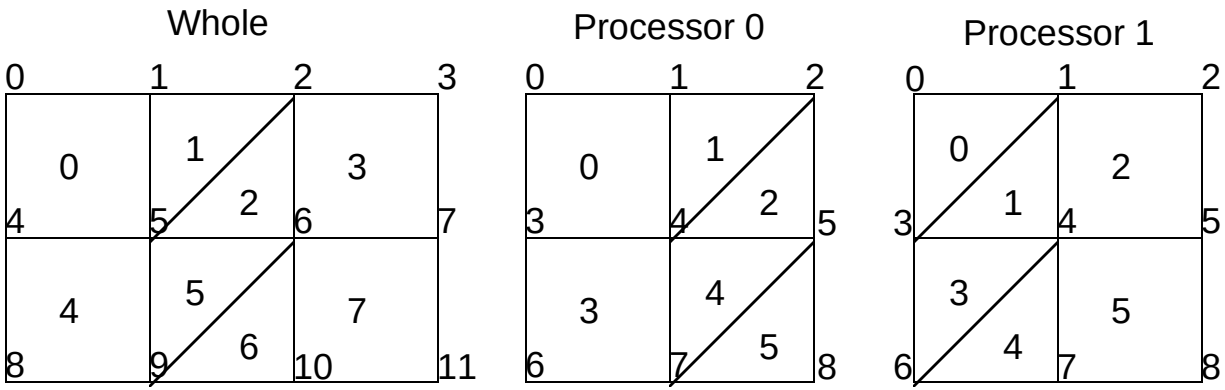


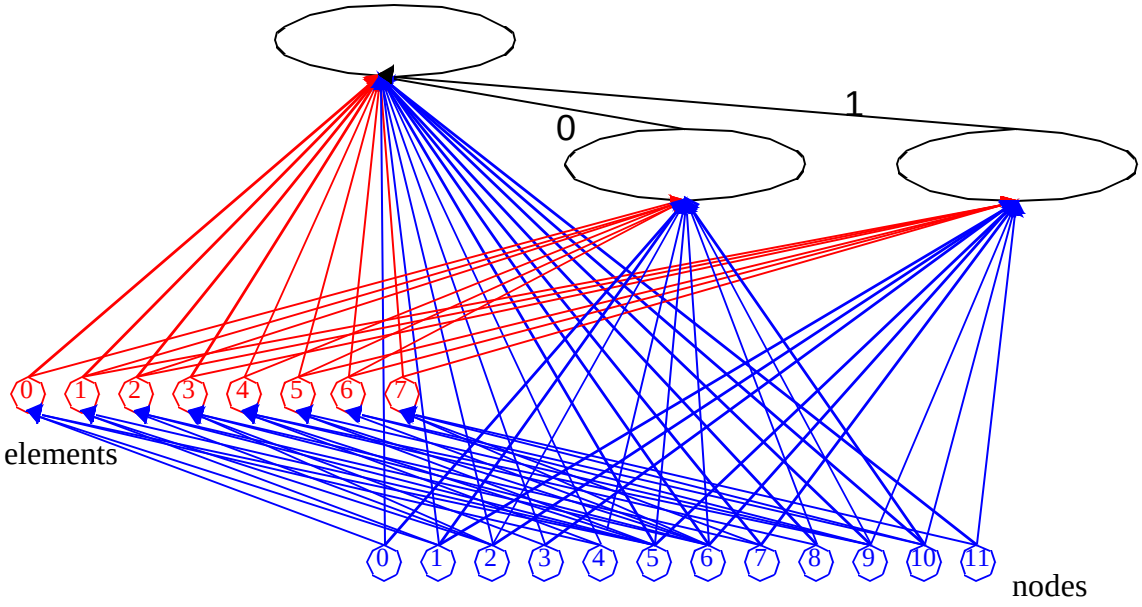
q Template

Where is the Partitioning Relation?

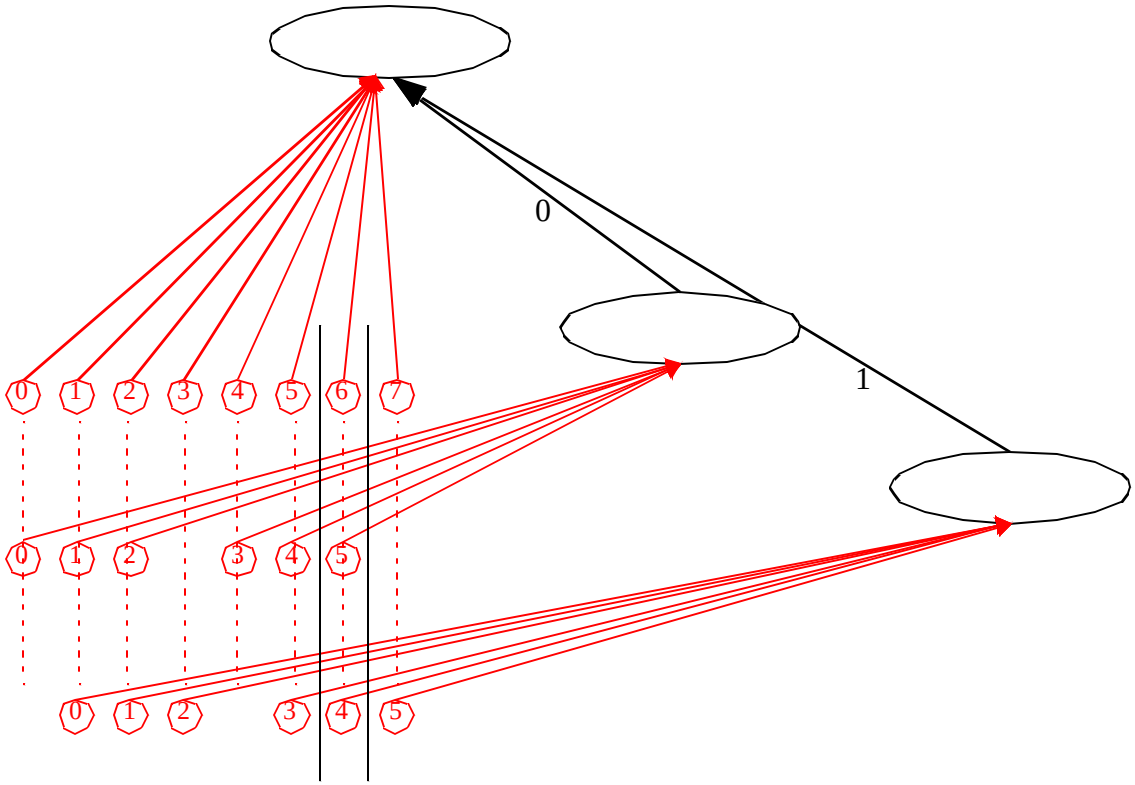


Parallel Example





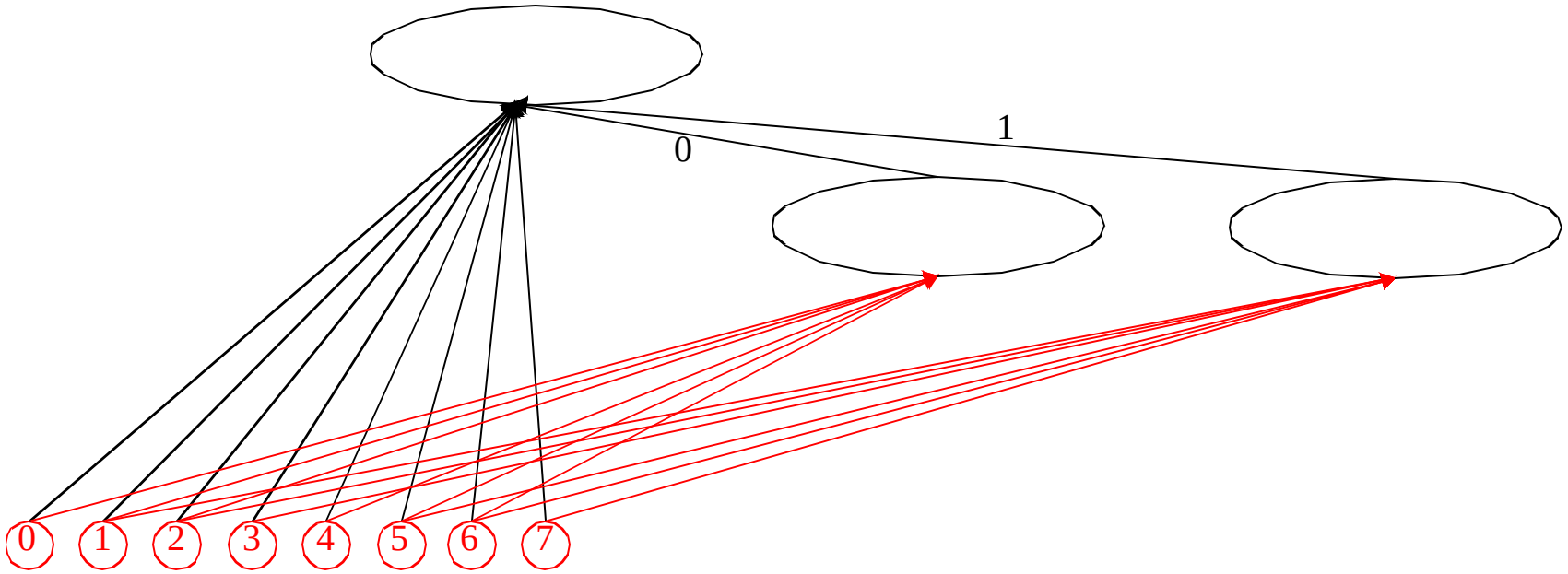
Pathnames



Pathnames for highlighted element

- `/tops["my_mesh"]/procs[0]/elems[5]`
- `/tops["my_mesh"]/procs[1]/elems[4]`
- `/tops["my_mesh"]/elems[6]`

Element-Based Processor Decomposition



domain: `/tops["my_mesh"]/procs[0]`
range: `/tops["my_mesh"]/elems`
op: subset-of

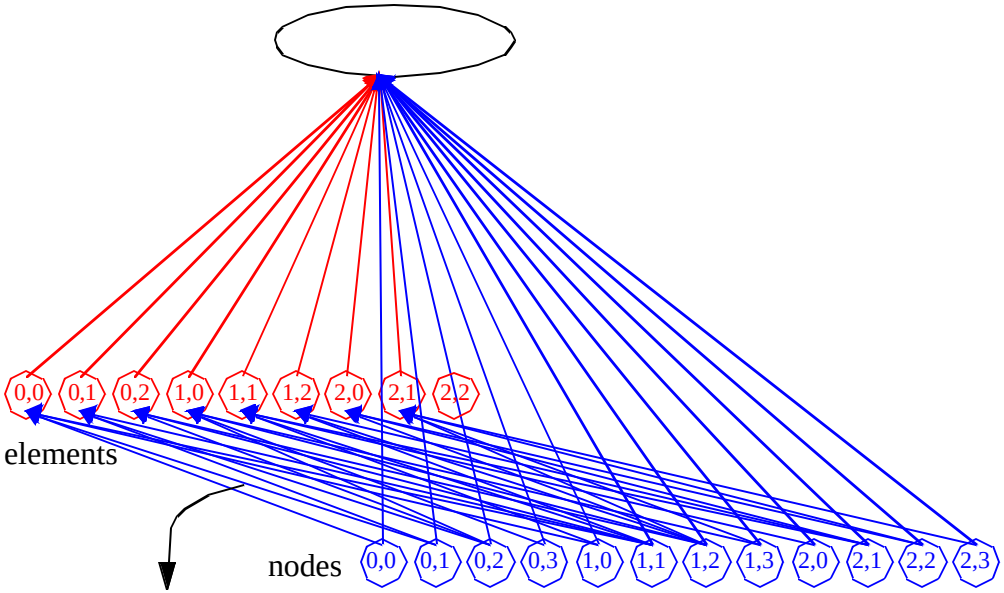
D	R
0	0
0	1
0	2
0	4
0	5
0	6

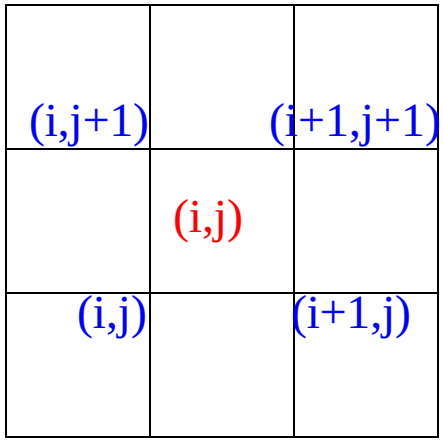
domain: `/tops["my_mesh"]/procs[1]`
range: `/tops["my_mesh"]/elems`
op: subset-of

D	R
0	1
0	2
0	3
0	5
0	6
0	7

Structure in Meshes

Rectangular Grid





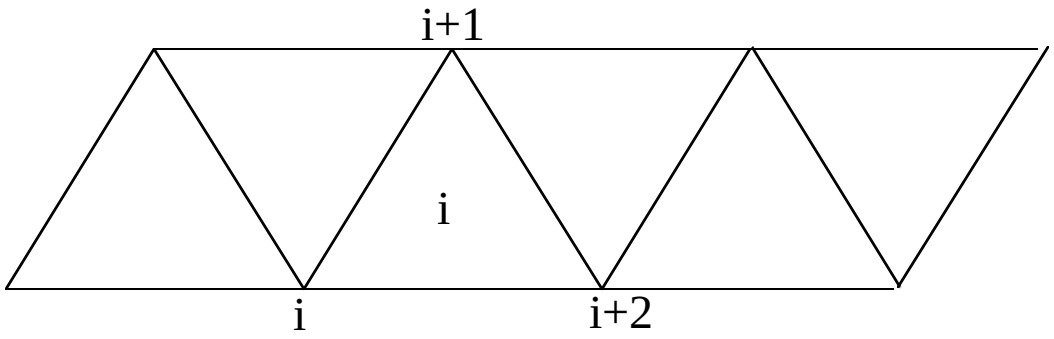
D (quads)	R (nodes)
0,0	0,0
0,0	0,1
0,0	1,0
0,0	1,1
0,1	0,1
0,1	0,2
0,1	1,1
0,1	1,2
⋮	⋮

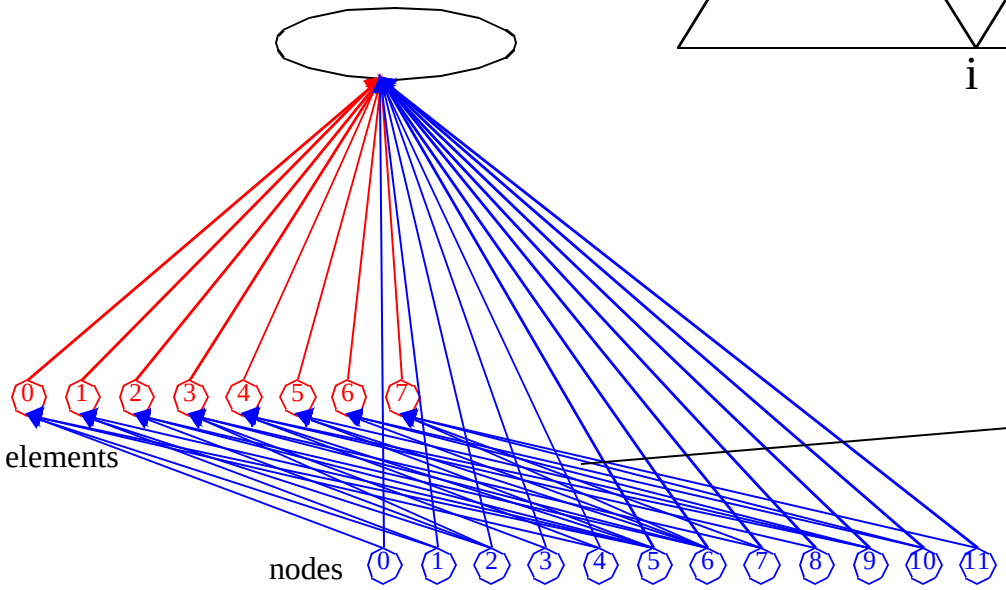
Structured Relation Template

D (quads)	R (nodes)
i,j	i,j
i,j	i,j+1
i,j	i+1,j
i,j	i+1,j+1

Structure in Meshes (Cont'd)

Triangle Strips

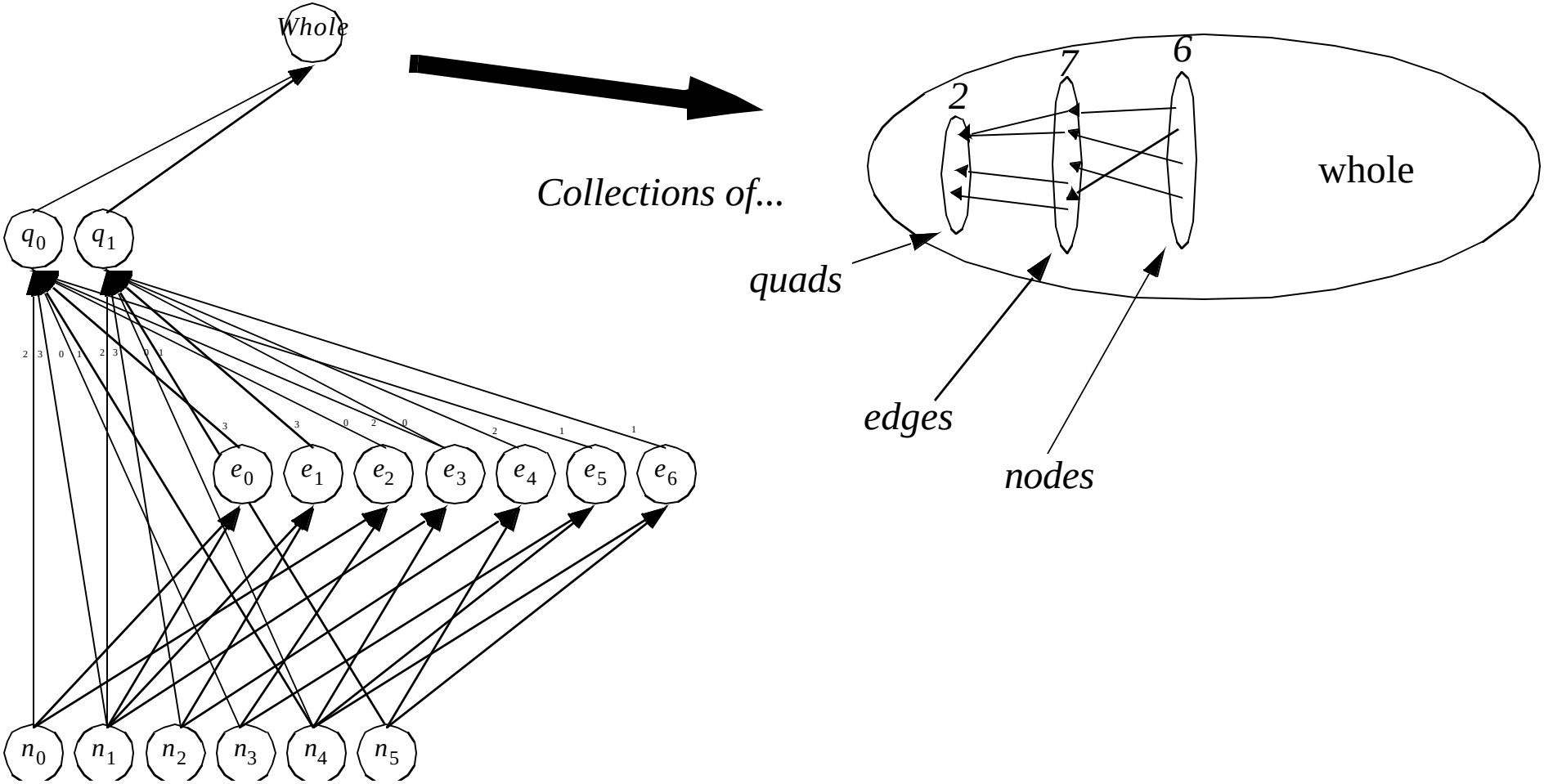




Structured Relation Template

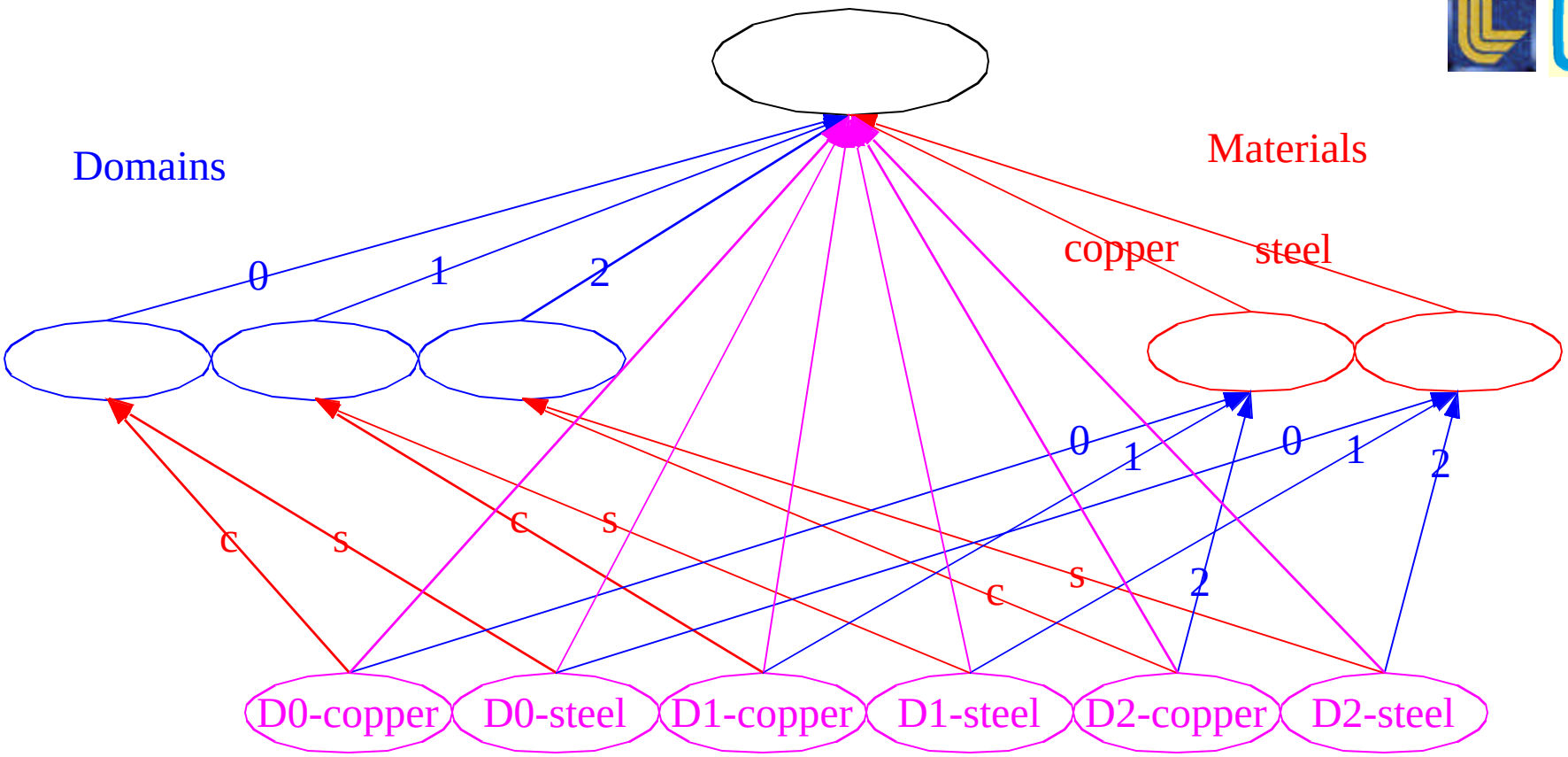
D (quads)	R (nodes)
i	i
i	i+1
i	i+2

Simplifying the SRG



- *Don't draw all the primitive sets!*
- *Arrows between aggregate sets are whole relations (e.g. many links in the SRG)*

Multiple Decompositions



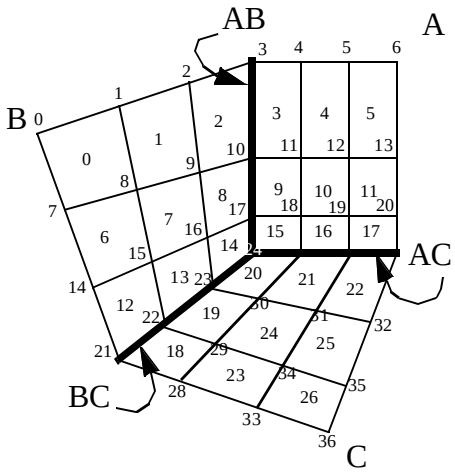
Example: Paths to D0-steel

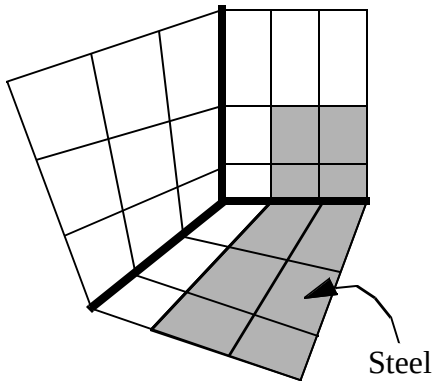
- `/tops["whole"]/domains[0]/mats["steel"]`
- `/tops["whole"]/mats["steel"]/domains[0]`

New link-class (color)

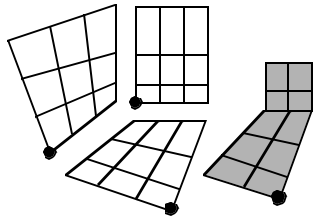
- *not a domain, not a material. Its domain AND material*

Unstructured Group of Rect. Pieces

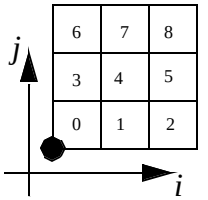


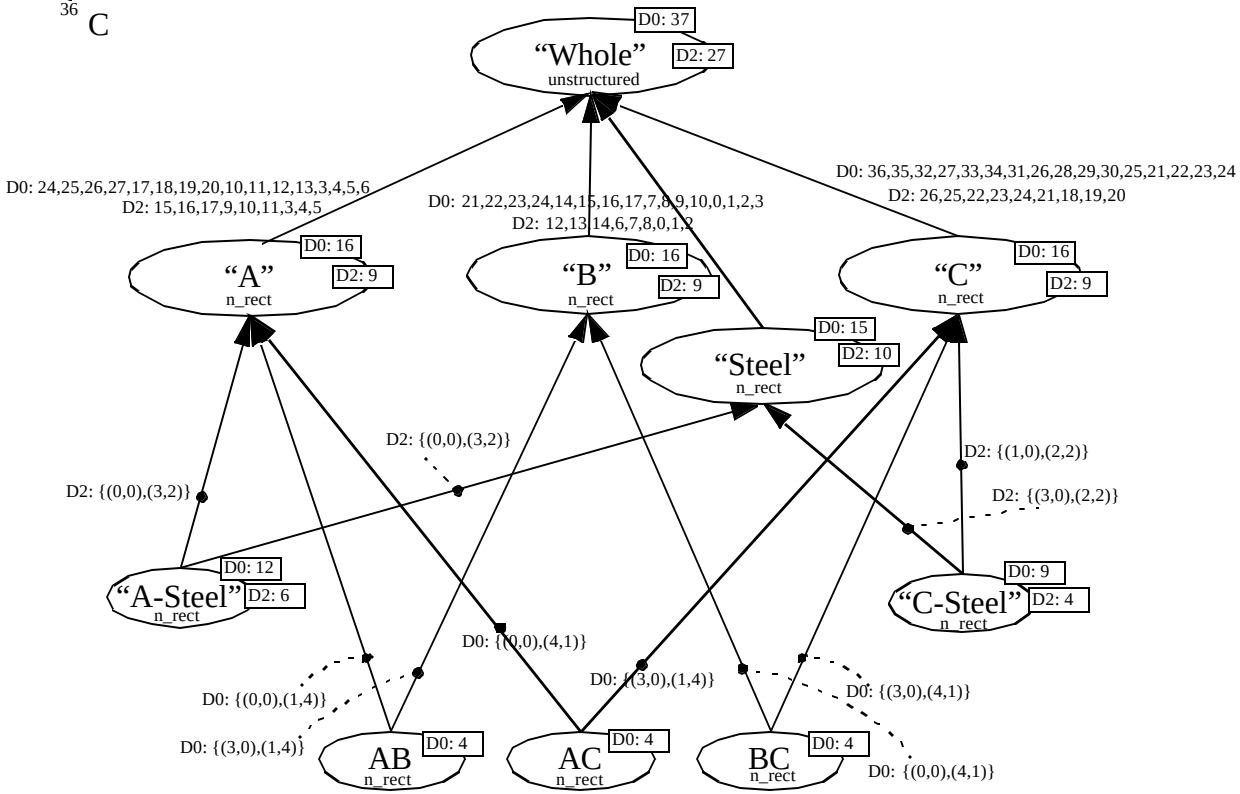


Desired Orientations



Canonical N_Rect Cell





Base-Space Summary

Set: infinity of points with a base-dimension

SRG: graph indicating relationships between sets

Subset-of is the dominant relationship

- *SRGs look more or less like trees*
- *Have a well-defined schema for identifying sets within such trees*

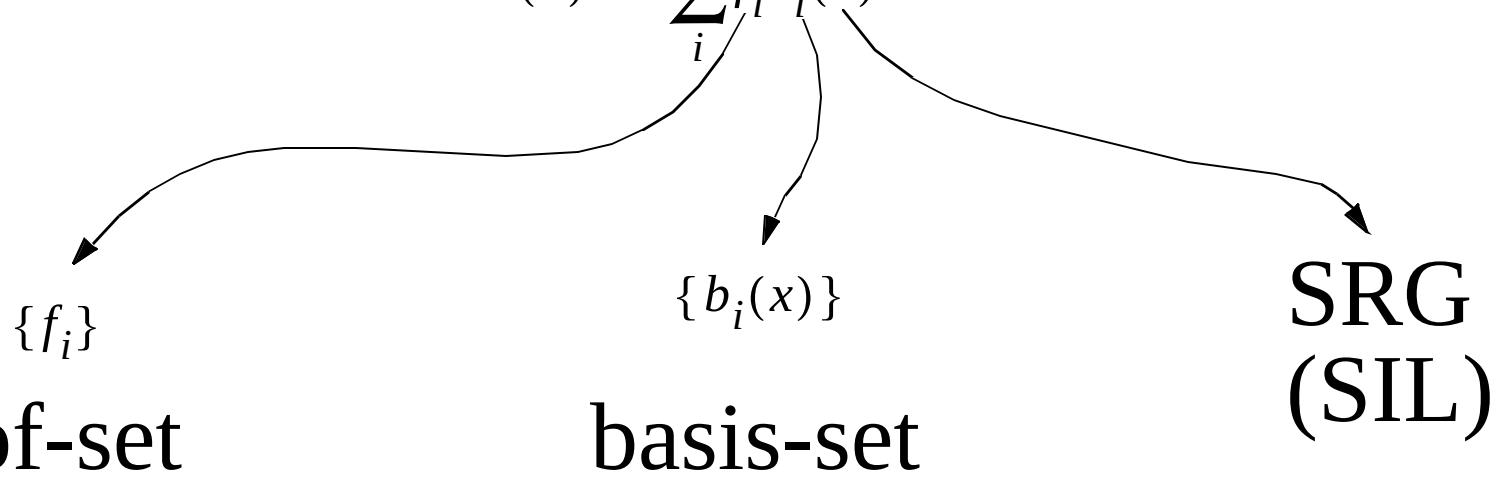
We have yet to talk at all about fields!!!

- *Can do a lot with just base-space information*

With base-spaces in hand, now we simply plug fields onto them

Fields

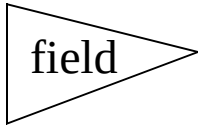
$$F(x) = \sum_i f_i b_i(x)$$



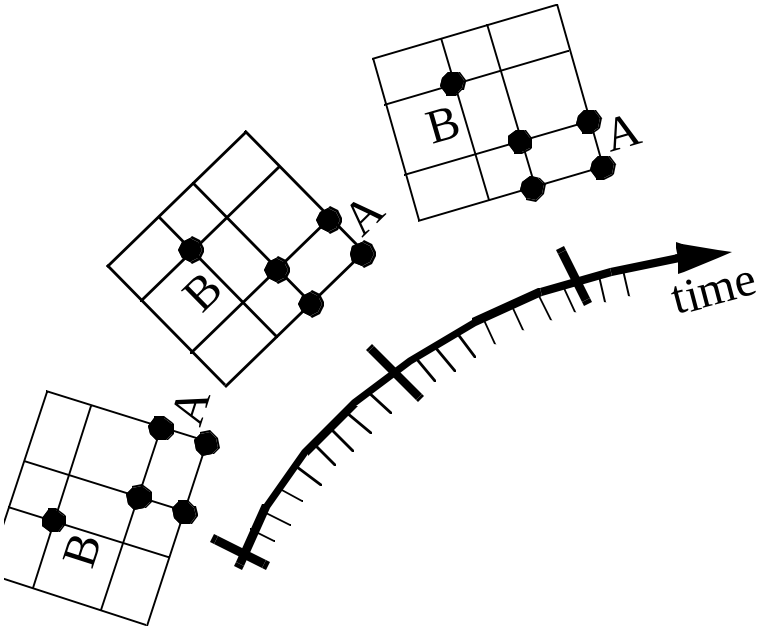
dof = "Degrees of Freedom"

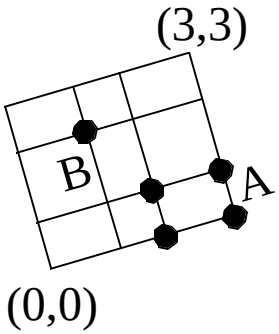
- *These are not "values" except when basis functions are interpolating*

Diagrammatic representation: Flag



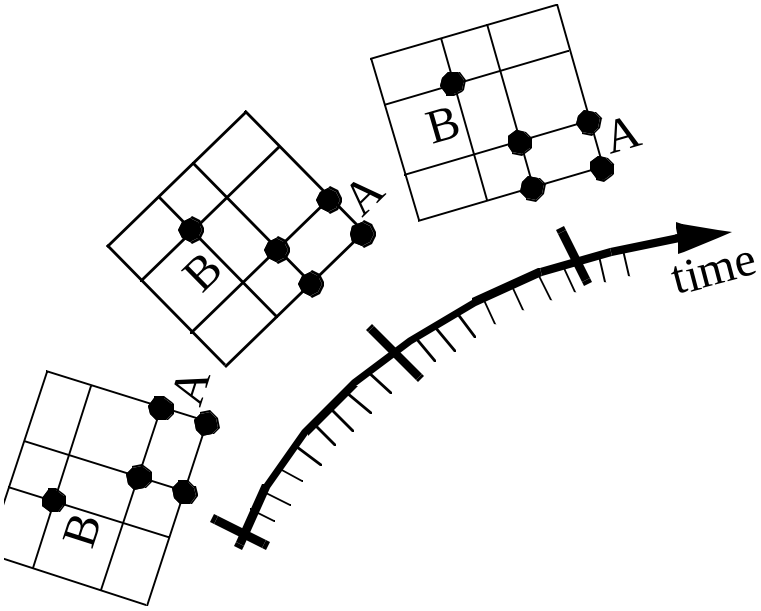
Return to First Example





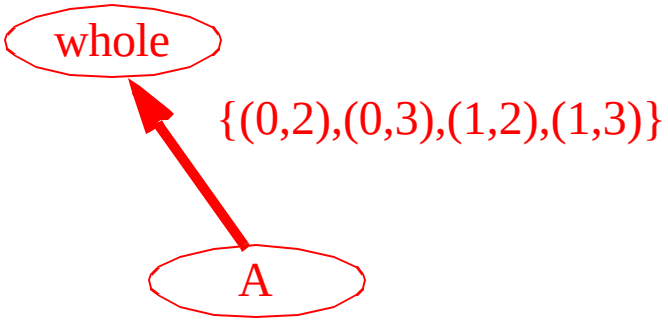
PROBLEM

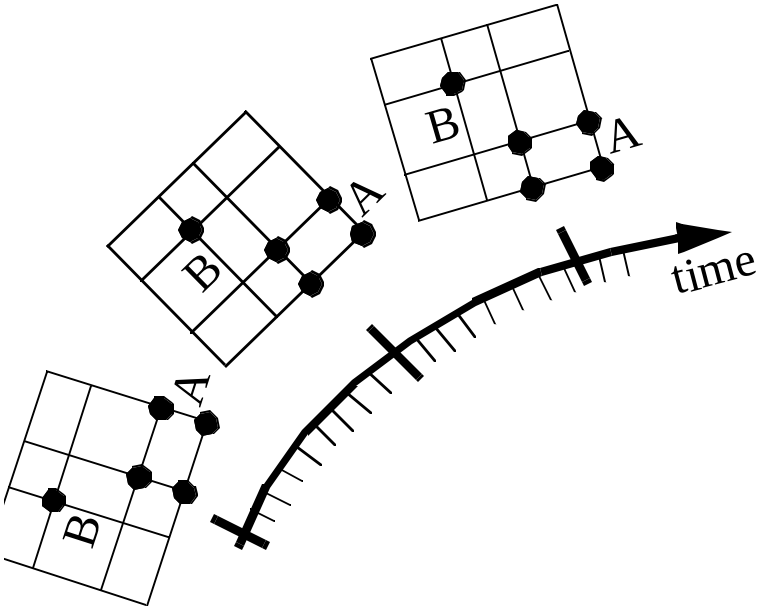
write “pressure” and “velocity” on whole mesh and stress tensor on nodeset A at times t0, t9, t17, and “pressure” time history on node B



PROBLEM

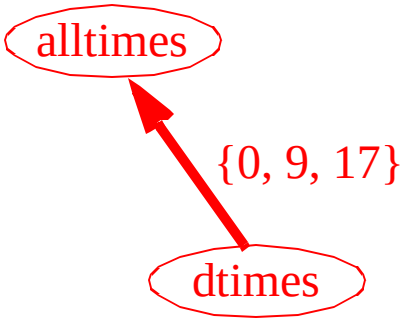
write “pressure” and “velocity” on whole mesh and stress tensor on nodeset A at times t0, t9, t17, and “pressure” time history on node B

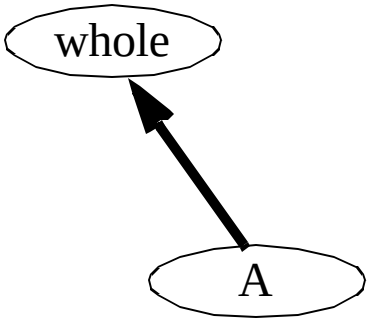


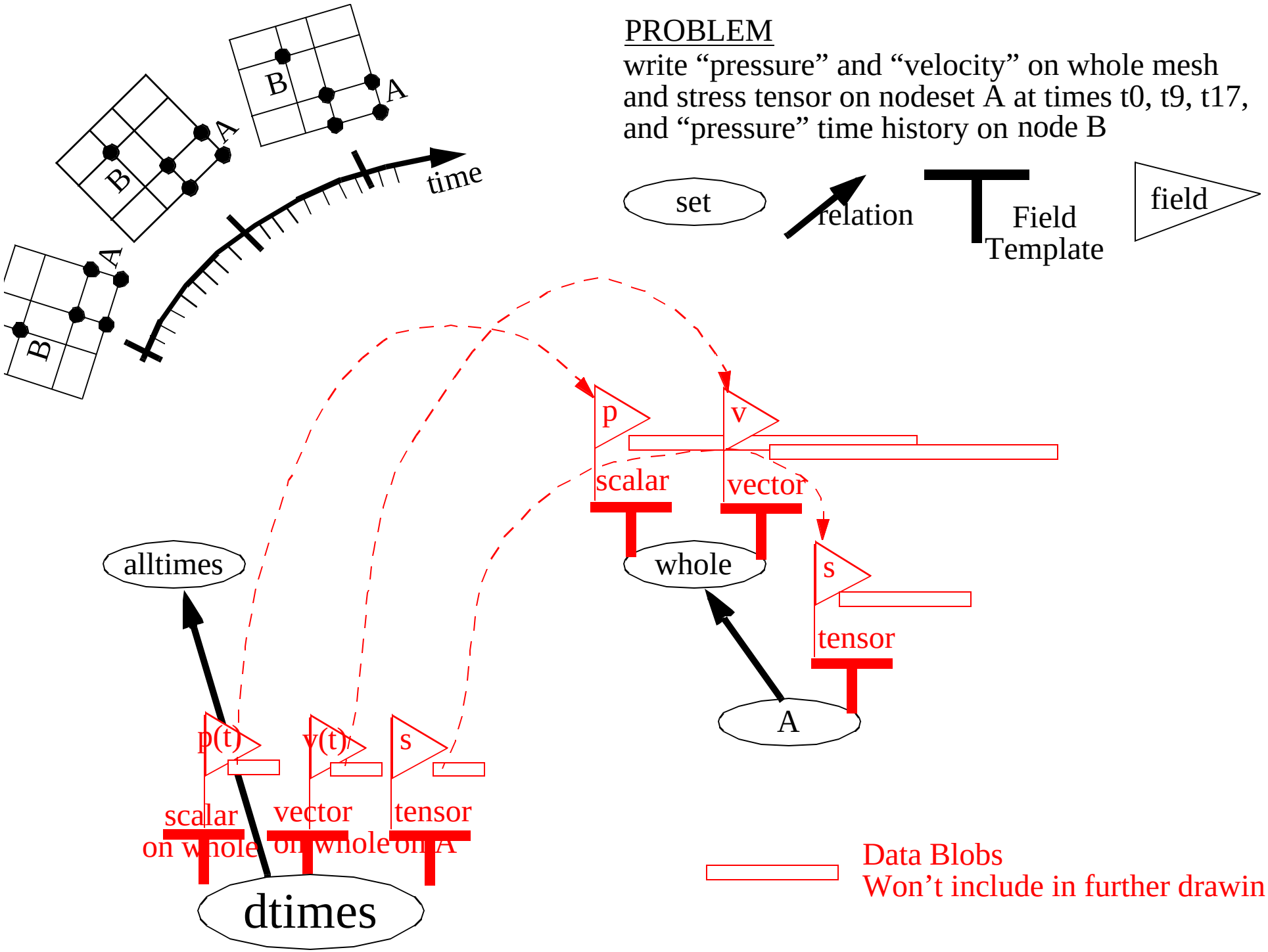


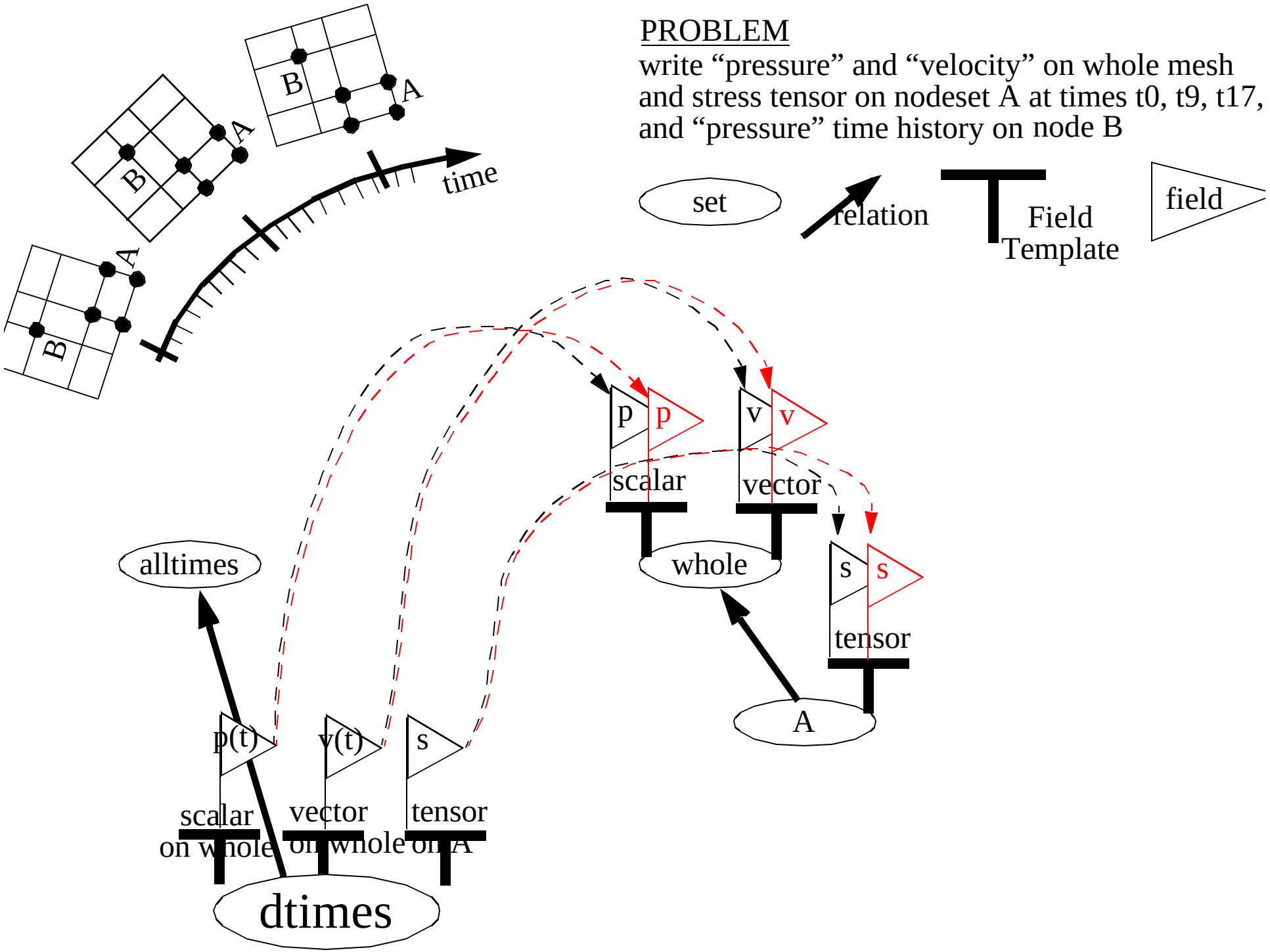
PROBLEM

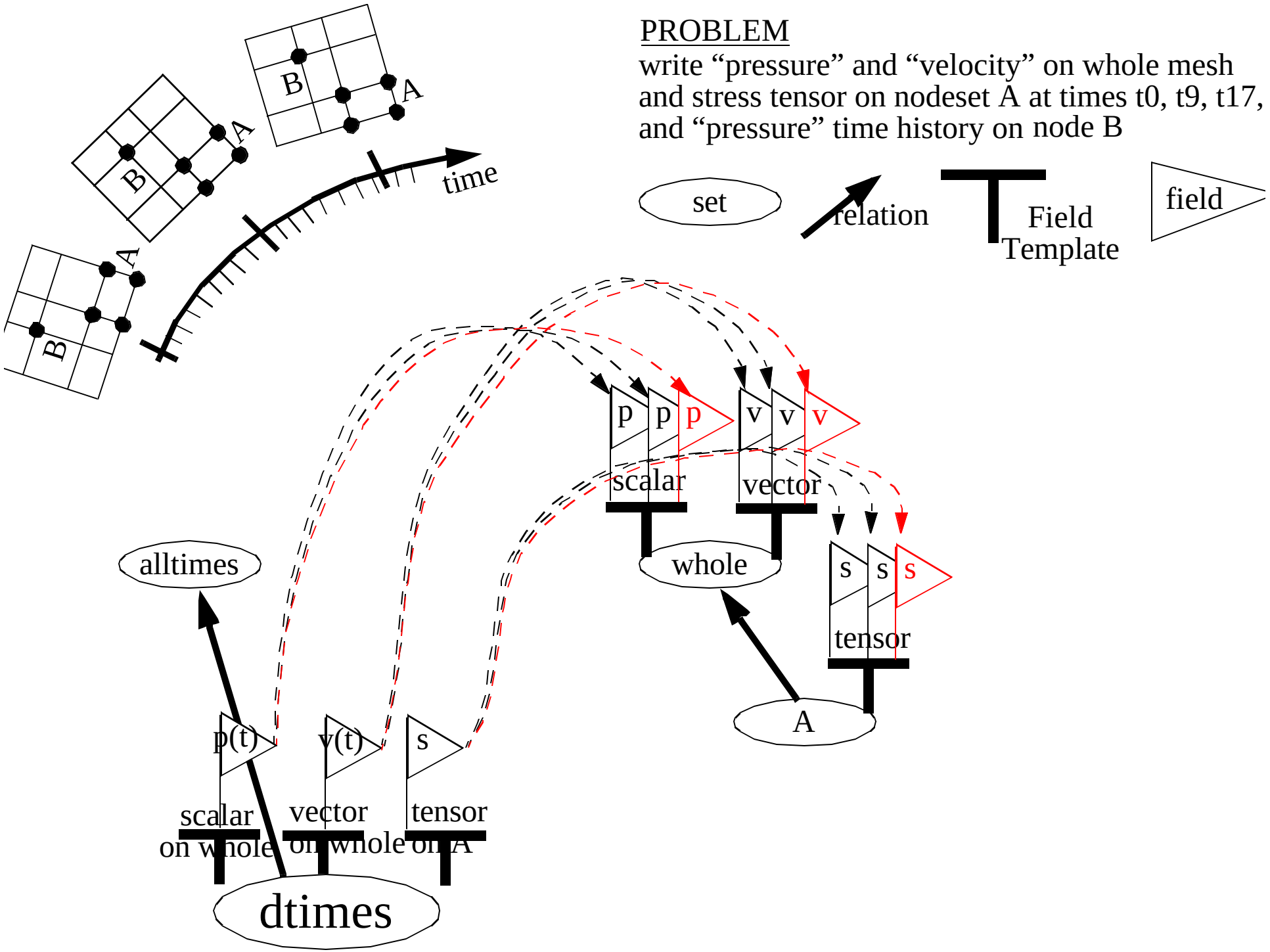
write “pressure” and “velocity” on whole mesh and stress tensor on nodeset A at times t0, t9, t17, and “pressure” time history on node B

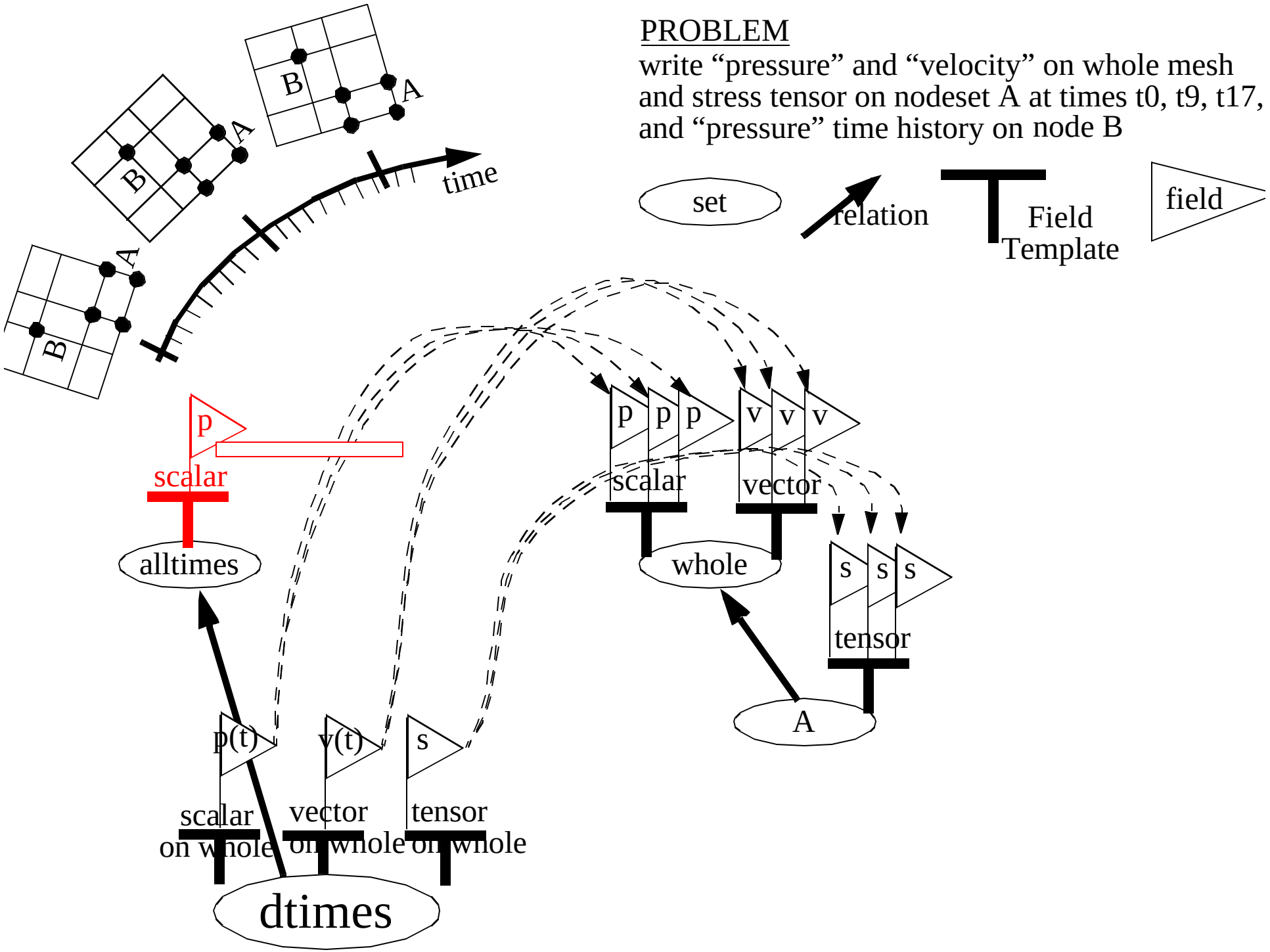




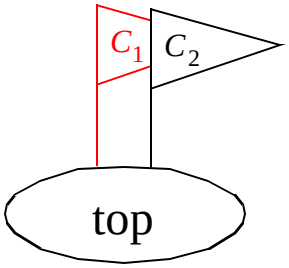

PROBLEM
write “pressure” and “velocity” on whole mesh
and stress tensor on nodeset A at times t0, t9, t17,
and “pressure” time history on node B
set
relation
Field
Template
field
B
A
B
A
B
A
time
p
v
scalar
vector
whole
s
tensor
A
alltimes
p(t)
v(t)
s
scalar
on whole
vector
on whole
tensor
on A
dtimes
Data Blobs
Won’t include in further drawin

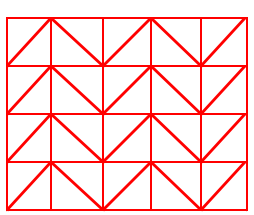

PROBLEM
write “pressure” and “velocity” on whole mesh
and stress tensor on nodeset A at times t0, t9, t17,
and “pressure” time history on node B
set
relation
Field
Template
field
B
A
B
A
B
A
time
p
p
v
v
scalar
vector
whole
s
s
tensor
A
alltimes
p(t)
v(t)
s
scalar
on whole
vector
on whole
tensor
on A
dtimes

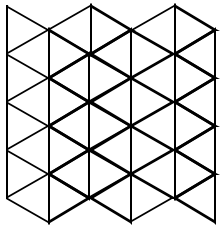

PROBLEM
write “pressure” and “velocity” on whole mesh
and stress tensor on nodeset A at times t0, t9, t17,
and “pressure” time history on node B
set
relation
Field
Template
field
B
A
B
A
B
A
time
p
p
p
v
v
v
scalar
vector
whole
s
s
s
tensor
A
alltimes
p(t)
v(t)
s
scalar
on whole
vector
on whole
tensor
on A
dtimes

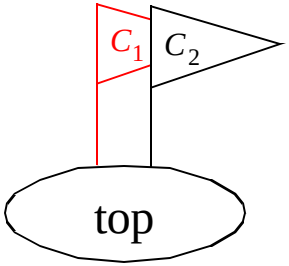

PROBLEM
write “pressure” and “velocity” on whole mesh
and stress tensor on nodeset A at times t0, t9, t17,
and “pressure” time history on node B
set
relation
Field
Template
field
B
A
time
p
scalar
alltimes
p p p
scalar
v v v
vector
whole
s s s
tensor
A
p(t)
v(t)
s
scalar
on whole
vector
on whole
tensor
on whole
dtimes

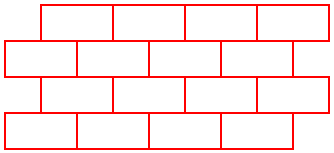
Multiple Coordinate Fields

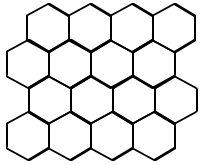




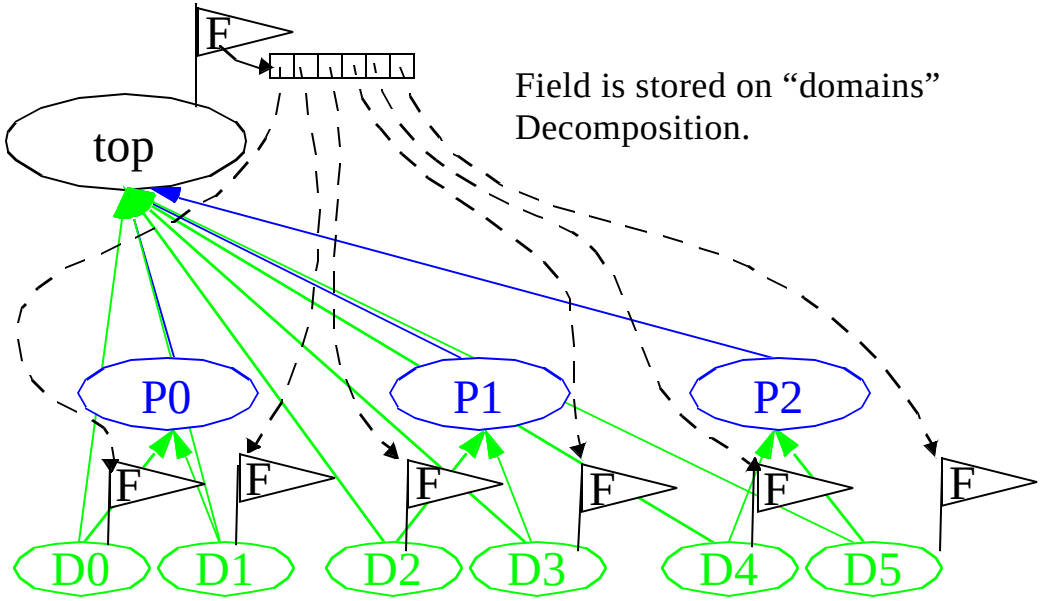








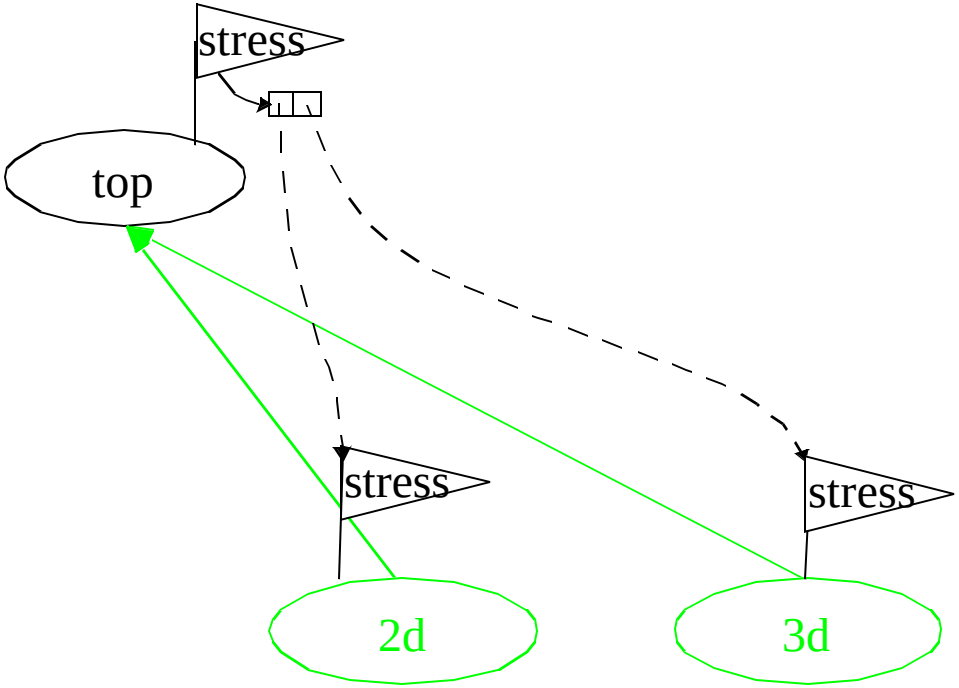
Inhomogeneous Fields
Decomposed Storage



Field on top is an “indirect” field

- *points to other fields*

Inhomogeneous Fields
Varying Fiber-Dimension



SAF: Sets and Field Parallel I/O And Data Modeling System

A first cut, scalable, parallel implementation of the data model

Key Goal of Implementation

Want to prove we can do with SAF...

...what we used to do with older technologies like Silo/Exodus

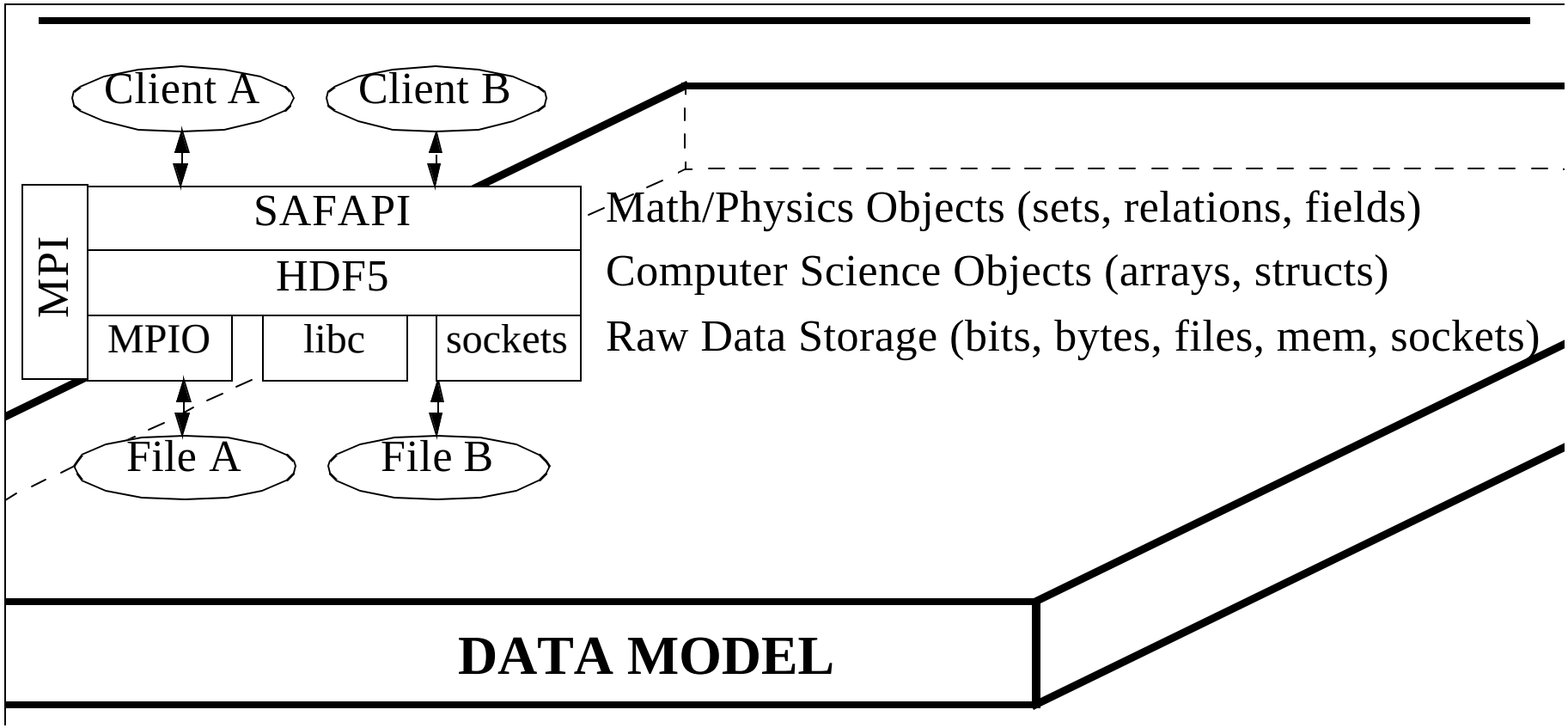
Once done, we intend to add many features

What is SAF?

SAF = Sets and Fields...

...Parallel I/O and Scientific Data Modeling System

SAF Software Architecture (Course View)



Data Model

- *Foundation for a majority of design*
- *Language for describing data*
- *Most significant part of whole effort*

What Does SAF Do?

Models all shareable scientific data (Results Data) in terms of...

...Sets, Relations and Fields

Examples...	**SAF Primitive**
• *The "copper" part (material)*	*Set*
• *A Slide surface*	*Set*
• *A processor piece*	*Set*
• *External facelist*	*Set*
• *local-to-global node map*	*Relation*
• *local-to-global element map*	*Relation*
• *nodal connectivity*	*Relation*
• *nodal (node-centered) "variable"*	*Field*
• *zonal (zone-centered) "variable"*	*Field*
• *coordinates (in time and space)*	*Field*
• *min/max of "pressure" over domains*	*Field*

Non-shareable data (Control Data) is described as attributes

- *simple parameter-value lists*
- *Examples: advection threshold, SQA info, configuration info*

What Does SAF Do (cont'd)?

Parallel (and Serial) I/O on Fields and Relations

- *Multifile and Single-file operation. A SAF "database" can have many files*
- *collective, parallel I/O from multiple processors into a file*

SAF Can Model Various Mesh Topologies

- *gridless meshes, structured, unstructured and arbitrarily connected meshes*

SAF Can Model Arbitrary Subsets and their Relationships

- *external facesets, slide surfaces, boundaries,*
- *processor decompositions, material decompositions*
- *nodesets, blocks, regions, etc.*

SAF Can Model Fields of Many Types Defined on Any Subset

- *Specification of fields defined on any subset*
- *Specification of the interpolation schemes needed to compute the value of a field*
- *Specification of field types, such as scalar, vector, tensor, and FIELD.*
- *Specification of abstract field quantities, such as length, time, force, energy*
- *Specification of specific units of measure such as meters, seconds, newtons or joules*

SAF Tools

saf-browser (Java-based)

- *Windows-95 Explorer-Like browsing of a SAF database*

safsh (Python-based)

- *a shell-like command-line tool for SAF databases*

safdiff (in development)

- *Differences representationally congruent SAF databases*
- *Will eventually be extended to deal with NON-representationally congruent databases*

MeshTV/Visit

- *integration of MeshTV/Visit's unstructured mesh object already*

safck (saf-check) (planned)

- *consistency checker for a SAF database*

Who has contributed to SAF?

Data Model

- *Limit Point Systems (LPS)*
- *SNL & LLNL*

Implementation Specifications

- *Vector Bundle Tables (VBT) Kernel - LPS*
- *SAF-API - SNL & LLNL*

Software Implementation

- *VBT - LLNL*
- *SAF-API - LLNL & SNL*

Portable, Parallel I/O via HDF5 and MPI-IO

- *NCSA & LLNL and ARL*

Tools

- *Java Browser - SNL*
- *safsh - SNL*
- *safdiff - LLNL*

Major SAF Releases

DMF-0.1	**Jan-1999**
DMF-0.8	**Jun-1999**
DMF-1.0 Beta Limited	**Nov-1999**
DMF -> SAF name change	**April-2000**
SAF-1.0 Production	**Jan-2001**
• *software is available, but not via web-site just yet*	
SAF-1.2	**Oct-2001**

SAF Software Project Practices

Portability Across All ASCI Platforms (Big Iron to Desktop)

Built On Top of Industry Standard Components; MPIO, HDF5

GNU Autoconf for Configuration/Installation

- *Autconf is a pain and hard to learn; but once you're using it, its *very* useful*

Incorporates Design By Contract Methodology

Incorporates CMM Practices

Reference Manual Generated Directly from SAF Sources

- *Encourages programmers to write documentation BEFORE writing code.*

Clear Separation of Raw Data and Meta Data I/O Streams

- *Client can "see" where raw data I/O is occurring; enables the publish/subscribe*

Growing Test Suite and Examples

Platforms SAF Runs On

Dec-Cluster, Blue, Snow, White, TFlops

O2k (1 box)

- *working on cross-box I/O*

Linux, Sgi, HP, Sun

Windows NT

Soon Expect to run on

- *Linux-cluster*
- *Sgi-Cray*

Some Implementation Details

Cells and Sets

Subset Relations and Topology Relations

Field Templates

All Collective All The Time

Cells and Sets

For performance reasons, we differentiate two kinds of sets...

...Primitive and Aggregate

Primitive sets are called “Cells”

- *Cells also have a Cell-Type (e.g. Quad, Hex, Tet, Arbitrary)*
- *Example: a Hex element*
- *never created with a call to saf_declare_set()*

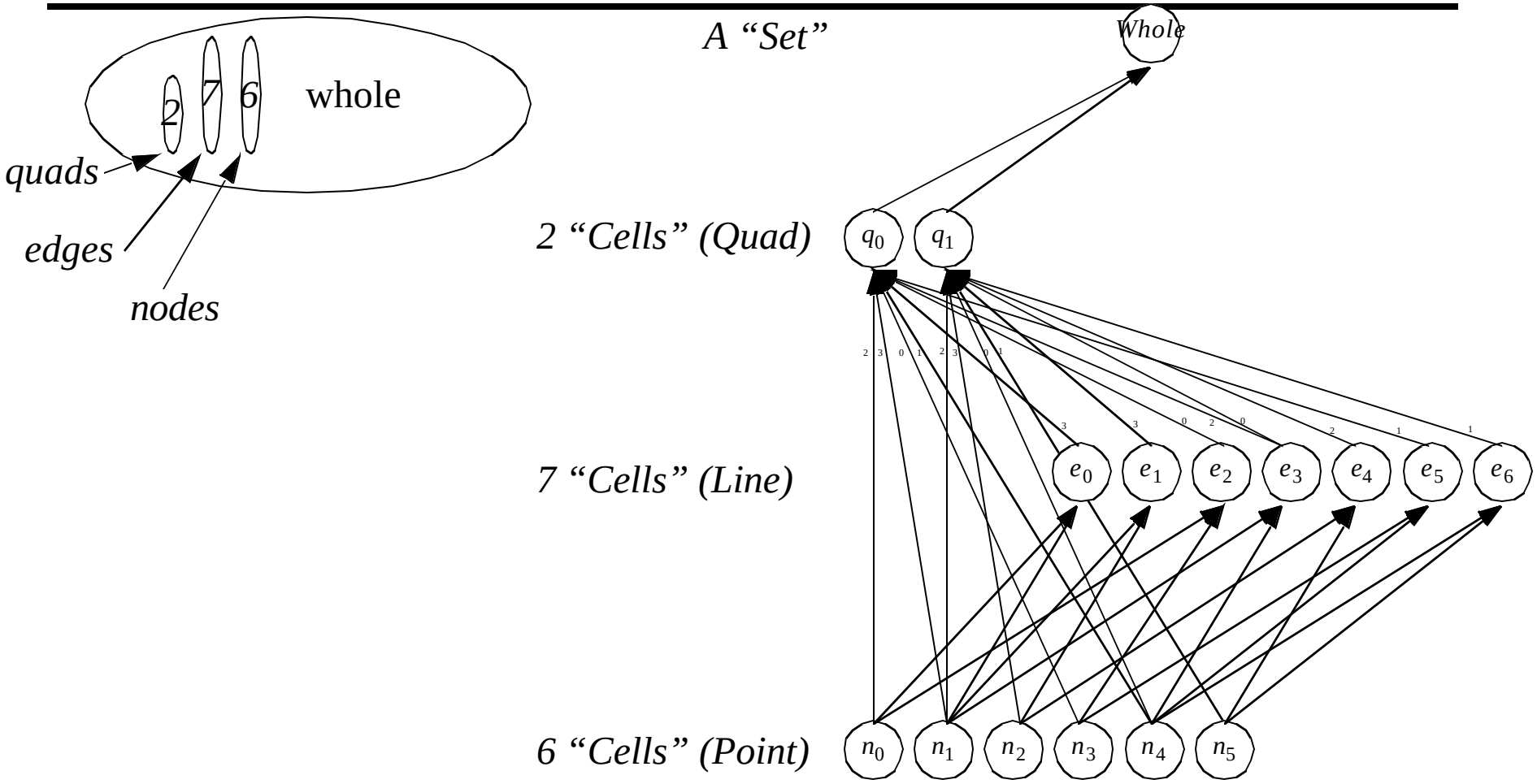
Aggregate sets are called “Sets”

- *Example: The Copper part*

The Difference between a Cell and a Set is...

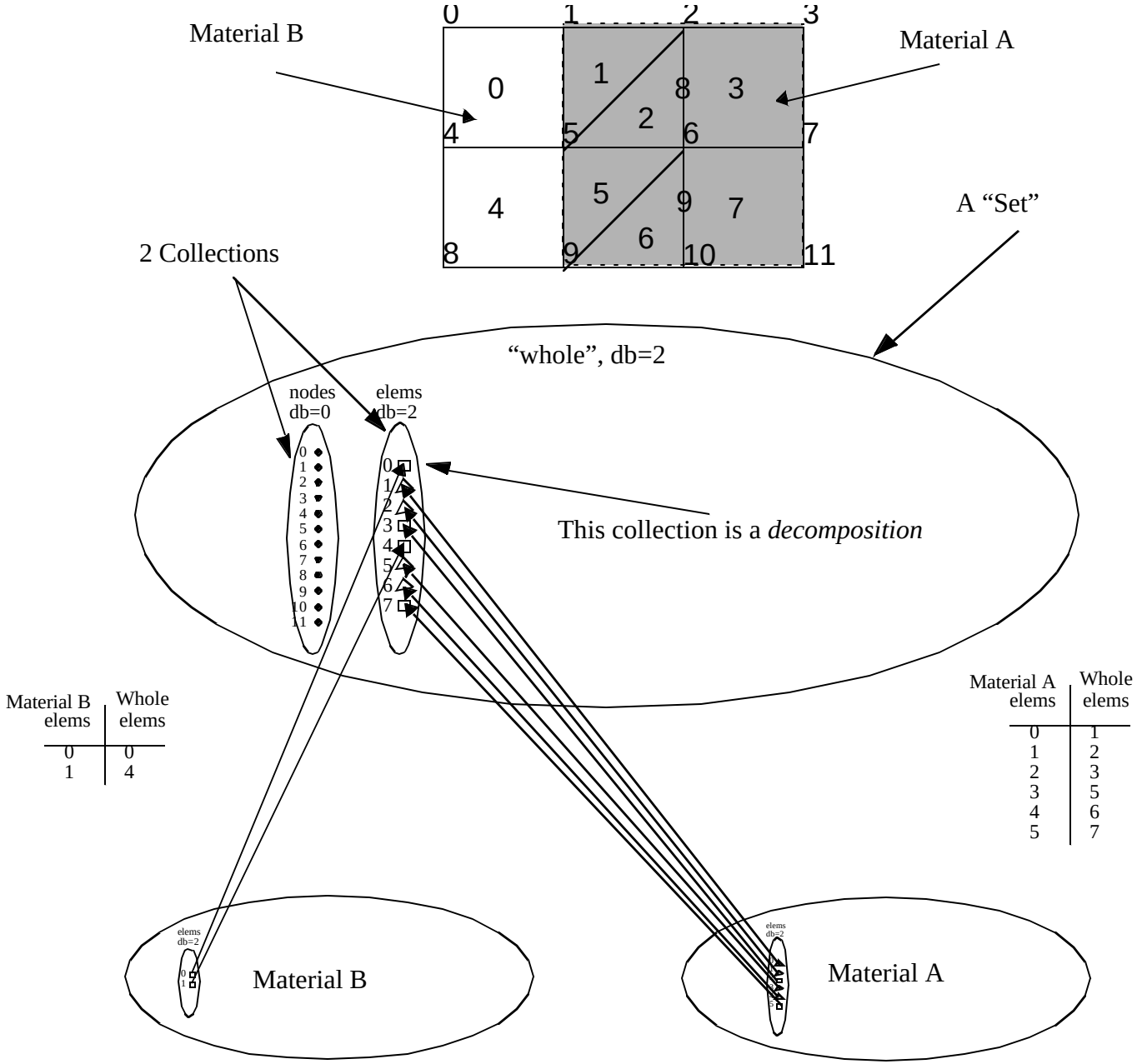
- *Sets may be equal to the union of other sets and/or cells.*
- *A cell is never equal to the union of any other cells or sets.*

Example of Cells and Sets



- *Cells never get instantiated as first class Sets.*
- *Instead they only ever appear as members of collections which are contained in Sets.*

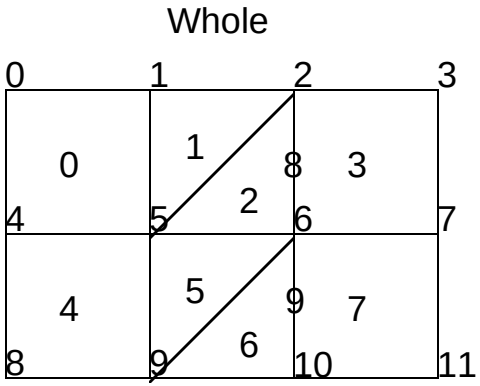
Subset Relations (op=equal-to)



Material B elems	Whole elems
0	0
1	4

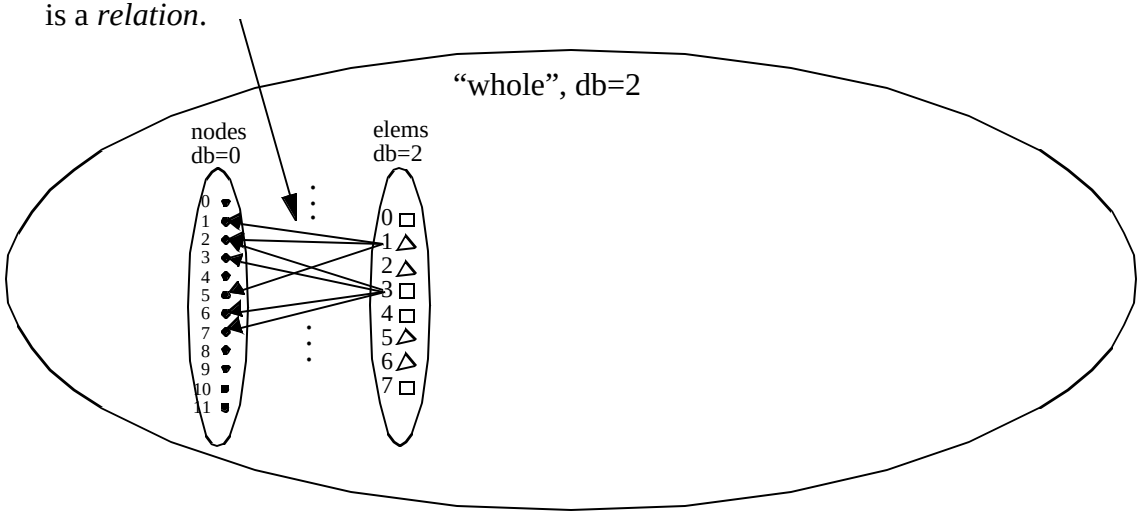
Material A elems	Whole elems
0	1
1	2
2	3
3	5
4	6
5	7

Topology Relations (op=supset-of)



This collection of arrows between elements and nodes is a *relation*.

The arrow heads indicate the direction of the relation



Whole elems	Whole nodes
0	4
0	5
0	1
0	0
1	5
1	2
1	1
2	5
2	6
2	2
3	6
3	7
3	3
3	2
4	8
4	9
4	5
4	4
5	9
5	6
5	5
6	6
6	9
6	10
7	10
7	11
7	7
7	6

Field Templates and Fields

Field Template: Representation Independent Features of a field

- *base space (set) its defined on (e.g. the wing)*
- *fiber-dimension (e.g. number of components)*
- *abstract quantity being represented (e.g. force or magnetic flux)*
- *algebraic type (e.g. scalar, vector, tensor, etc.)*
- *basis-vectors of fiber-space (if algebraic type is vector, basis might be cartesian)*

Field: Remaining representation dependent features

- *field-template*
- *collection dofs are associated with*
- *association ratio*
- *units of measure (e.g. Newtons or Pounds)*
- *evaluation scheme (interpolation functions, etc.)*
- *the dof data*

All Collective All The Time

Database/Client Interface winds up looking sort of SIMD

Causes

- *HDF5: opening/closing files, creating datasets*
- *SAF: internal database consistency*

Not as flexible as we'd like

- *Looking at "flexible-parallel" designs such as being proposed by HDF5*

SAF Integrations

Ale3d Physics Code

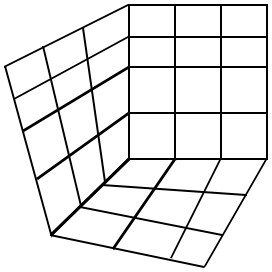
- *Domain Decomposition*
- *Mixing Materials*
- *Slide Surfaces*
- *Load Curves*
- *Time invariant and time varying data*

MeshTV/Vizit Visualization Tools

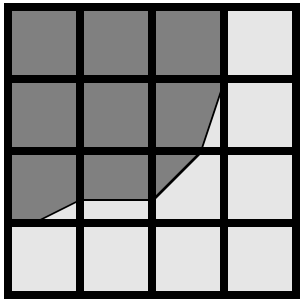
- *Will only discuss some viz-specific objects Ale3d generates to accelerate viz*
- *Min/Max Fields (Extents Fields)*
- *Wire-Frame Facsimile of Mesh*

Characterization of ALE3D's Data

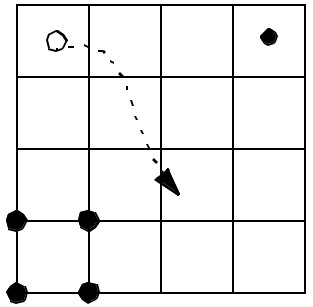
Multi-block, structured



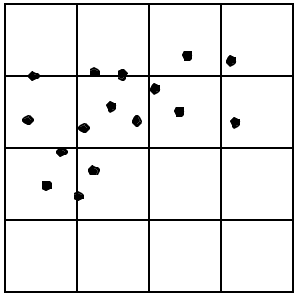
mixing materials within a zone



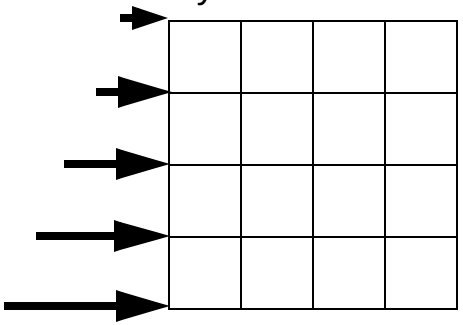
Time Histories (●), Tracers (○)



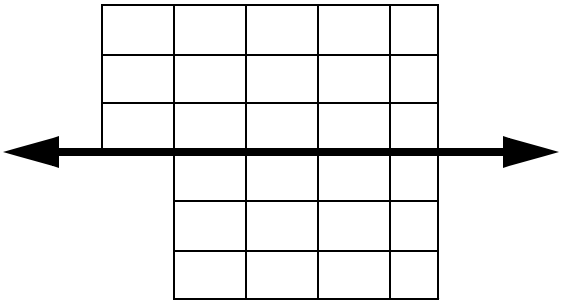
Tracer Particles



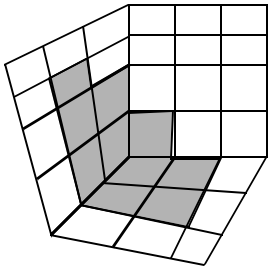
Boundary Conditions



Slide Surfaces



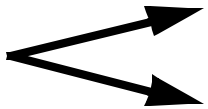
user-def'd subsets



Domain "overload"

D_0	D_1	D_2
D_3	D_4	D_5

Dynamic Load Balancing In Future



$\{D_0, D_1, D_2\}$ $\{D_3, D_4, D_5\}$

P_0 P_1

Processors

P_0 P_1 P_2

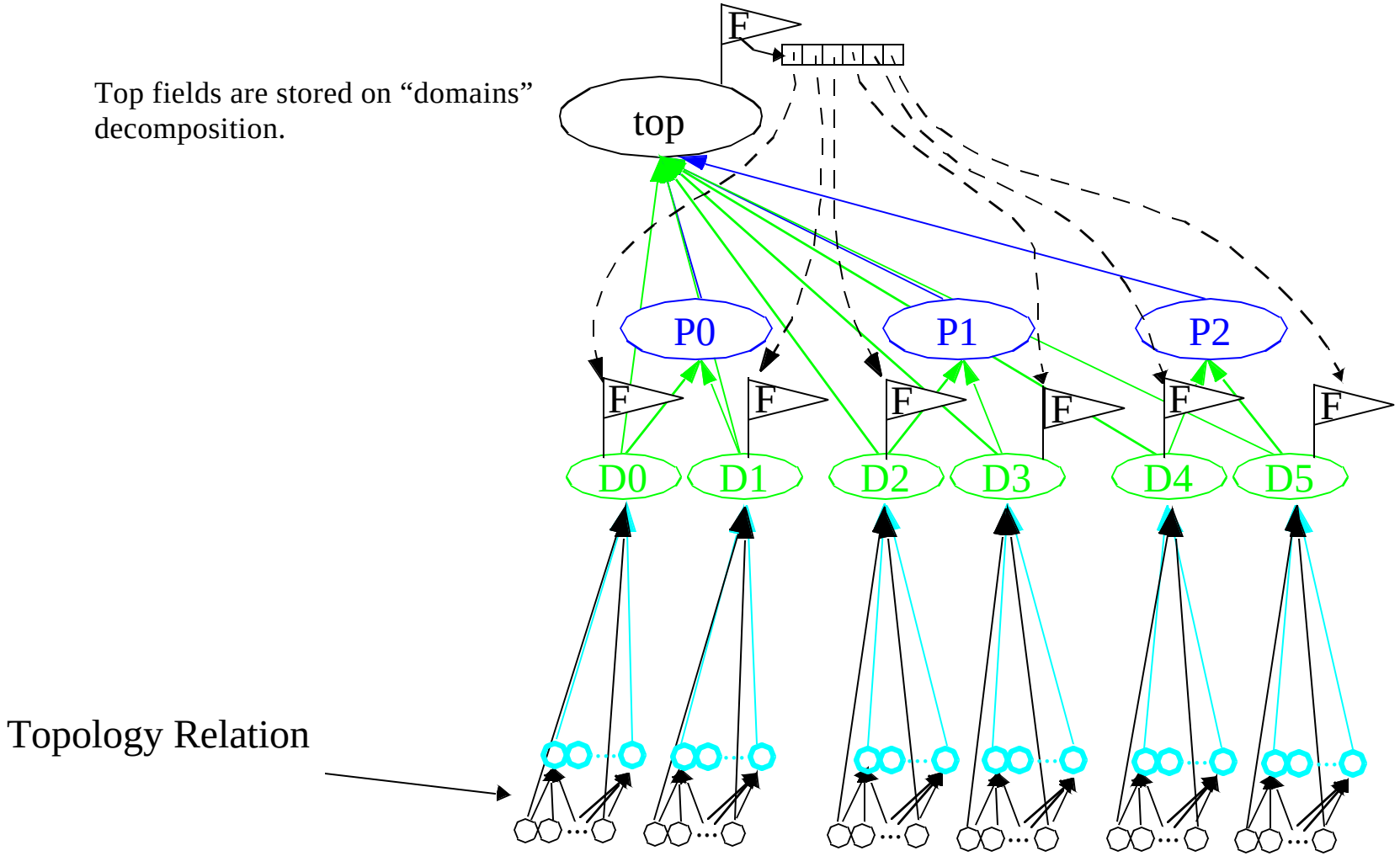
$\{D_0, D_3\}$ $\{D_1, D_4\}$ $\{D_2, D_5\}$

Field Types

$$[p] \quad \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} \quad \begin{bmatrix} s_{xx} & s_{xy} & s_{xz} \\ & s_{yy} & s_{yz} \\ & & s_{zz} \end{bmatrix}$$

Ale3d SRG: Domain Decomp part

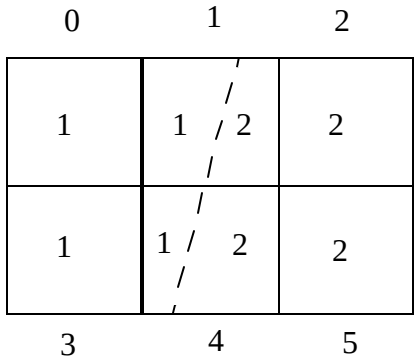
Top fields are stored on “domains” decomposition.



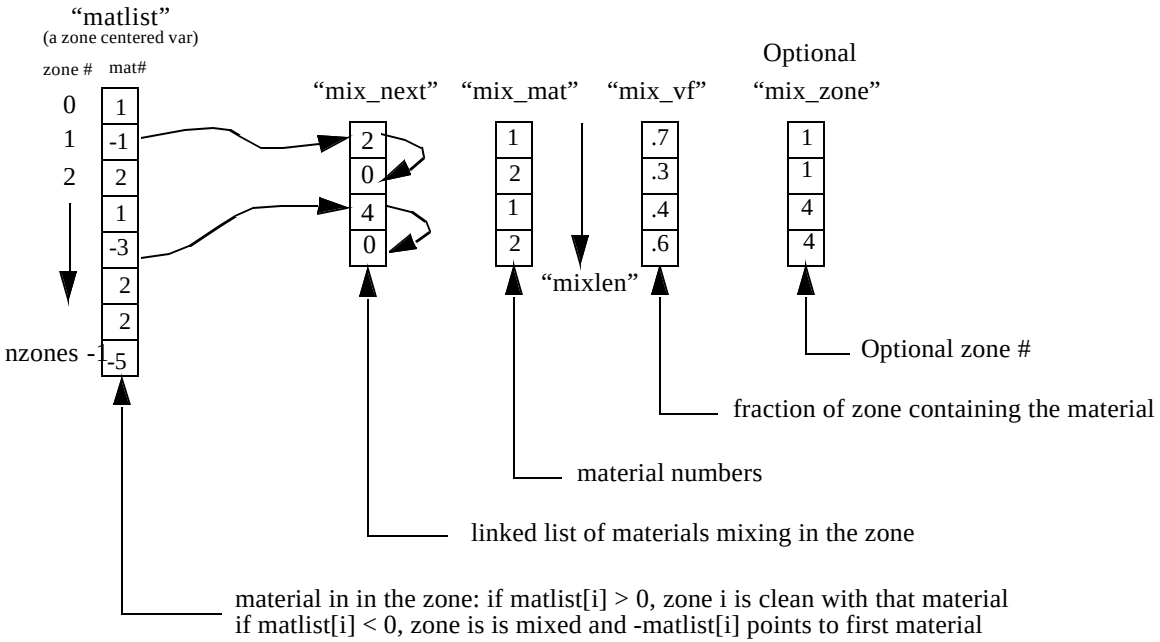
Topology Relation

- *Each Domain is an independent (except for “ghost” zones) chunk of mesh*
- *Every “group” of arrows is one of Ale3d’s internal arrays*

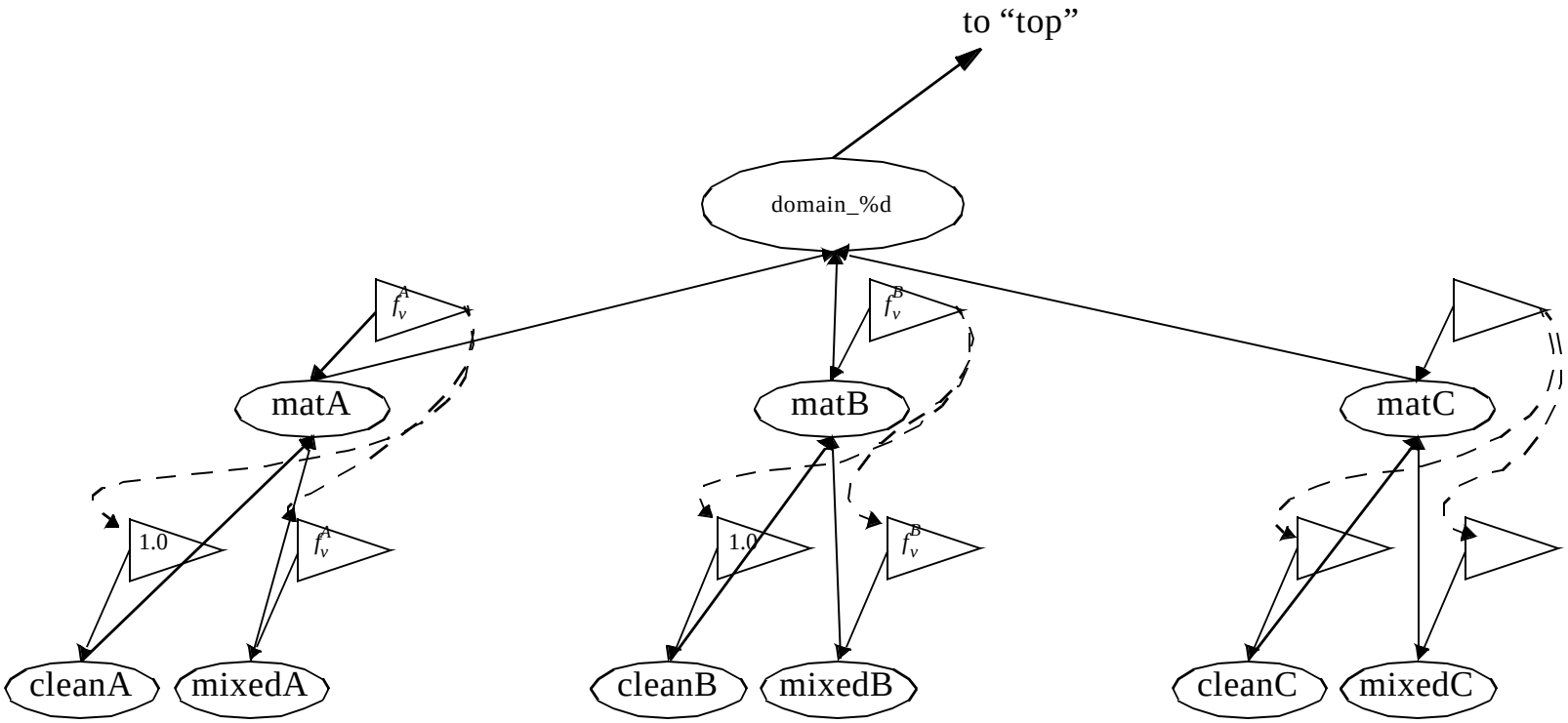
Mixed Materials
(The “old” way with Silo)



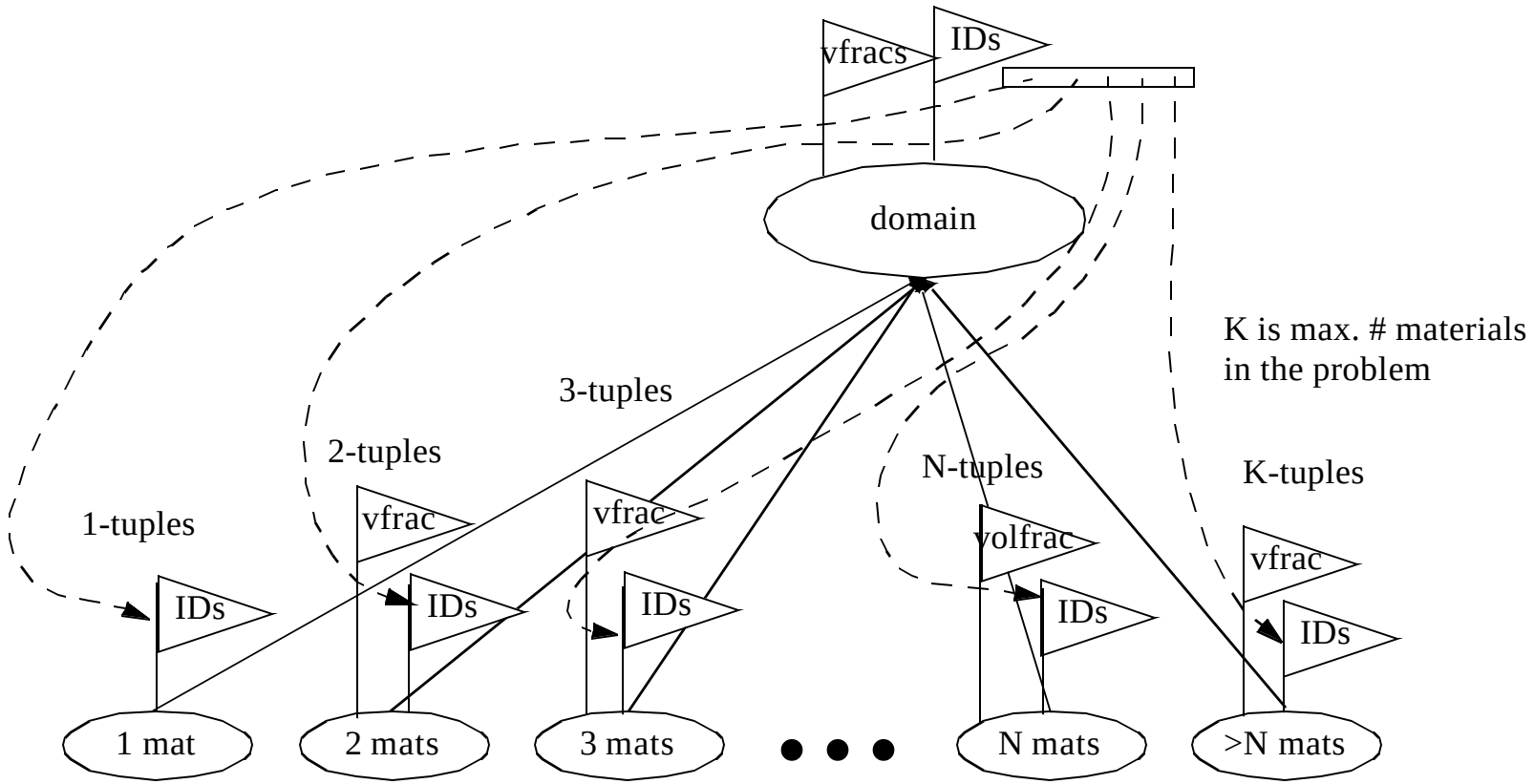
A simple 6 zone mesh
with mixing materials, 1 and 2



Mixed Material (materials as sets)

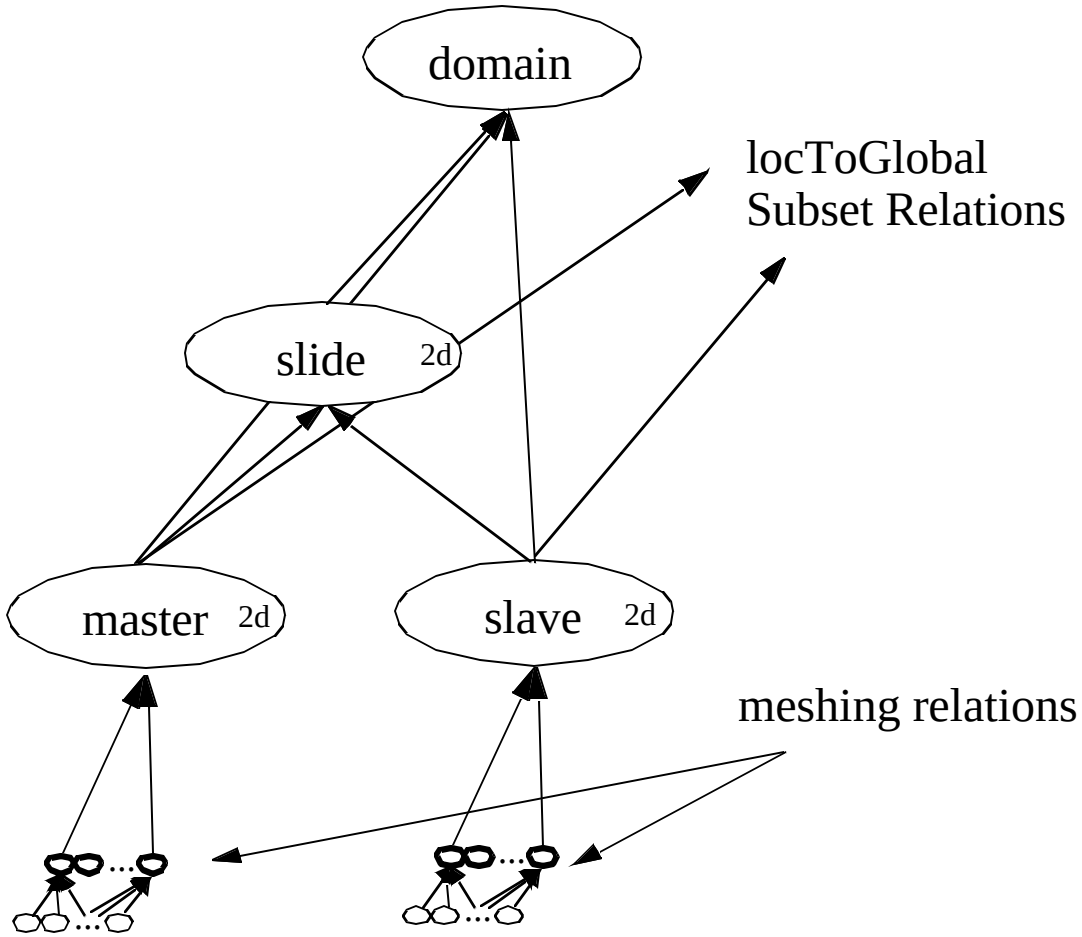


Mixed Materials (materials as fields)



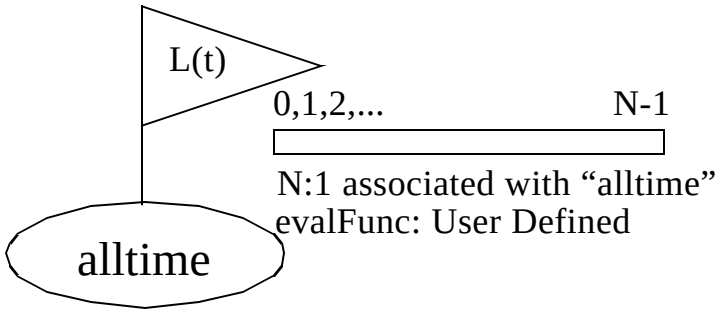
- *Can choose "N" to achieve certain optimization*
- *The base-spaces don't actually have any physical meaning. Thats Bad!*

Slide Surfaces



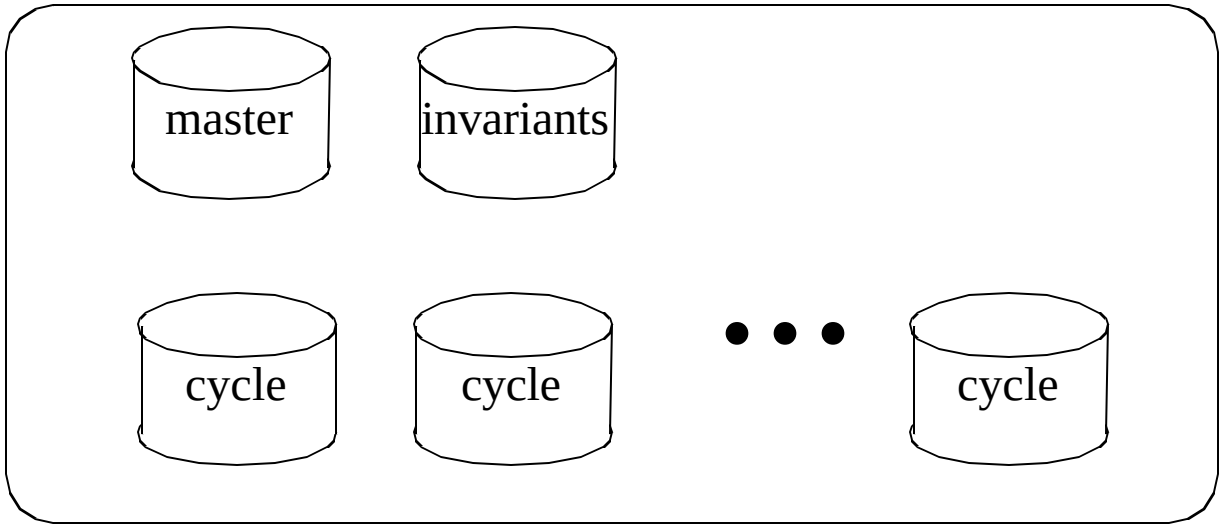
Load Curves

Time-variation in a boundary condition, typically applied to a set of nodes



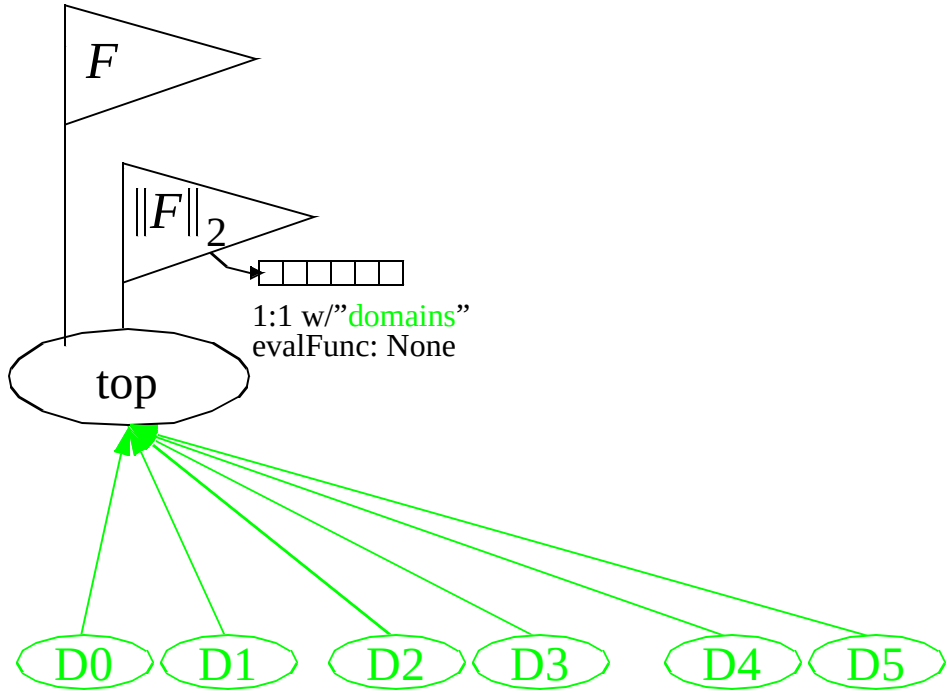
Time Invariant and Time Varying Data

One SAF Database for entire Run of Code



Taking advantage of invariants leads to a reduction in size of each dump by 20-25%

Min/Max Fields



Lazy Evaluation: Query for $\|F\|_2$. read if present/compute if not

- *SAF does not support this yet*

MeshTV/Vizit use Min/Max fields to accelerate operators

- *Only read domains of interest (within view, within iso-val, etc.)*

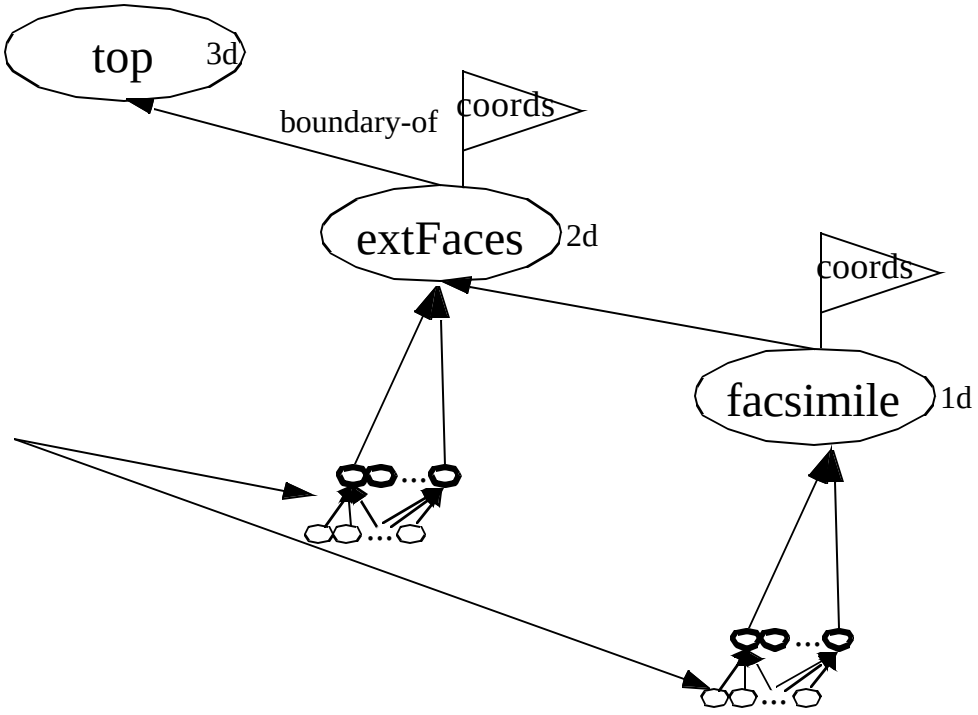
Wire Frame Facsimile

Viz tools need a small, easy to display/rotate facsimile for mesh

- *Answer: compute a wire-frame of the "important edges" of a mesh*
- *Important edge is an edge between external faces which meet at large dihedral angle*

Will borrow algorithm from MeshTV and put into Ale3d

Have their own meshing relations and their own coordinate fields



Summary and Future

We can describe a wide variety of scientific data...

...with a handful of primitives

"Describe" is the key word here

- *It means we can satisfy an existence condition*
- *Does the database system know what it needs to know about the data so that other software tools can do something useful with it? Yes.*

To complete an exchange of data...

...need to populate system with transform operators

- *between different processor decompositions*
- *between different node orders*
- *between different component interleave*
- *between different coordinate systems*
- *between different basis functions*
- *between different element types*
- *between different field evaluation methods*