Multi-Scale Spatial Database and Map Generalisation

Sheng Zhou & Christopher B. Jones

Department. of Computer Science

Cardiff University, UK

4th August 2001

From Paper Map to GIS and Spatial Database (SDB)

- Limitations of paper maps
  - A single shared space for both storage and presentation
  - Limited information may be stored at a single location
  - Selection means loss of information
- Advantages of GIS and SDB
  - Separation of storage space and presentation space
  - In theory, infinite information at a single location may be stored in a GIS/SDB

The way to an integrated SDB

- Problem with the common layered structure
  - Conflicts between features at different layers
- Solution – integrated object-based SDB
  - Objects occupy locations and are aware of their environments
- Dominant and recessive objects at locations:
  - A location has one (or a few compatible) default dominant object(s) while others are made recessive by default
  - A non-default query may promote recessive object(s) to dominant position to be retrieved

From Single-Scale SDB (SS-SDB) to Multi-Scale SDB (MS-SDB)

- MS-SDB and MS object: bring the scale dimension into integrated SDBs
  - A MS object has its default dominant and recessive scale ranges
  - A MS object contains several geometric representations by referring to multi-scale geometries
- A snapshot of a MS-SDB at a certain scale is a complete integrated SS-SDB

Elements of a Multi-Scale Spatial Database (MS-SDB)

- 4Ms: four elements for design and implementation of MS-SDB
  - Multi-scale data models
  - Multi-scale data storage schemes for spatial DBMS
  - Methods to generate multi-scale dataset from source dataset
  - Methods to access MS-SDB and make legible presentations from query results
- The latter two elements are closely associated with map generalisation

Progressive Generalisation
– Generation of Multi-Scale Spatial Dataset (1)

- Conventional map generalisation
  - Generating representation at a fixed smaller scale from a source dataset at a fixed larger scale
  - Assuming the two representations are good approximations to the true representations at intermediate scales
  - The often-mentioned ***multi-version*** approach for MS-SDB is merely a digital analogue to traditional map series

Progressive Generalisation

– Generation of Multi-Scale Spatial Dataset (2)

- What is a true multi-scale spatial dataset
  - The MS dataset must support retrieval of true representations at any scales within the scale range it supports
  - The MS dataset must avoid the heavy data redundancy and the lack of connections between versions which are associated with the multi-version solution

Progressive Generalisation

– Generation of Multi-Scale Spatial Dataset (3)

- Effect of scale transition on spatial dataset
  - For two representations $rep_1(s_1)$ and $rep_2(s_2)$ (scales $s_1>s_2$) of a spatial dataset, the minimum difference between them is a single point:
    - $Rep_1(s_1) - rep_2(s_2) = \{Pt\}$
  - Therefore, it can be said that:
    - $Rep_1(s_1)$ is the true representation at $(s_2, s_1]$, $(s_1>s_2)$
    - Or alternatively, $rep_2(s_2)$ is the true representation at $[s_2, s_1)$, $(s_1>s_2)$
  - And for the point Pt, it is assigned the following scale range
    - $SR_{Pt} = [s_1, s_0)$, $(s_1 \leq s_0 < \infty)$

Progressive Generalisation

– Generation of Multi-Scale Spatial Dataset (4)

- From data alternation to scale transition
  - Q: at what scale (s_2) when a point should be removed from the original representation $rep_1(s_1)$ to create a new representation $rep_2(s_2)$?
  - A: when it makes no further (visual, etc.) contributions to the representation
- Criteria for point elimination:
  - Visual judgement / offset distance
  - Others

Progressive Generalisation

– Generation of Multi-Scale Spatial Dataset (5)

- From point elimination to feature modification
  - Point elimination will trigger geometry-level operations (elimination, merge, etc.). geometries and subsequently map features are classified towards scale
  - It is much easier to handle limited, progressive changes in dataset caused by small scale transitions than to deal with abrupt changed introduced by large scale transitions (say, from one scale to the next smaller scale in a map series)

Progressive Generalisation

– Generation of Multi-Scale Spatial Dataset (6)

- Decomposition of scale space
  - Key-space decomposition:
    - Partitioning the scale space into many predefined small intervals and then decide whether a point / geometry / feature is present at each scale
    - Providing (fine or adequate) approximations to true representations
    - Laying in many generalisation algorithms (e.g. D-P)
  - Object-space decomposition:
    - Using certain methods to calculate the exact scale range for each point / geometry / feature
    - Always providing true representations
    - The way we should follow (whenever possible)

Progressive Generalisation

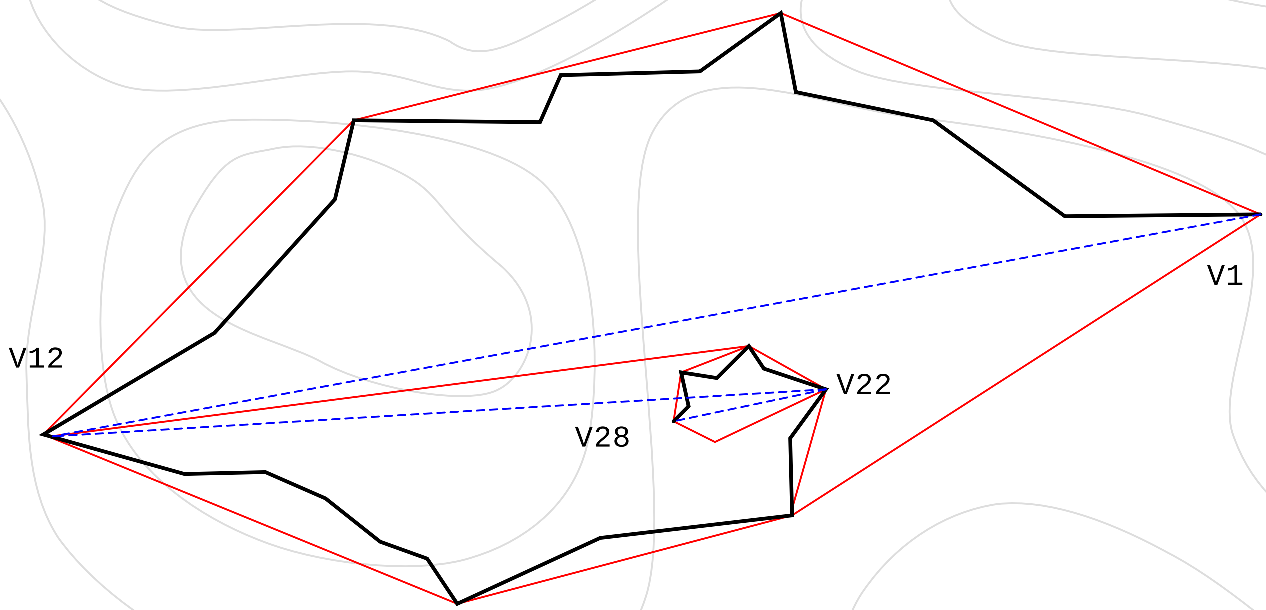
– Generation of Multi-Scale Spatial Dataset (7)

- BVT-tree: an object-space based scale classification method for linear/areal features
  - Step 1: partitioning geometry by recursive convex hulls
  - Step 2: for each section calculating tolerance value for each vertex in a D-P manner
  - Step 3: do tolerance promotion for vertices with reversed tolerance value to define a total-ordering for this section
  - Step 4: joining sections to define a total-ordering for the whole geometry
- At retrieval time, the total-ordering of tolerance (or other indicators) will be mapped to a total-ordering of scale

Progressive Generalisation

– Generation of Multi-Scale Spatial Dataset (8)

- BVT-tree: geometry partitioning

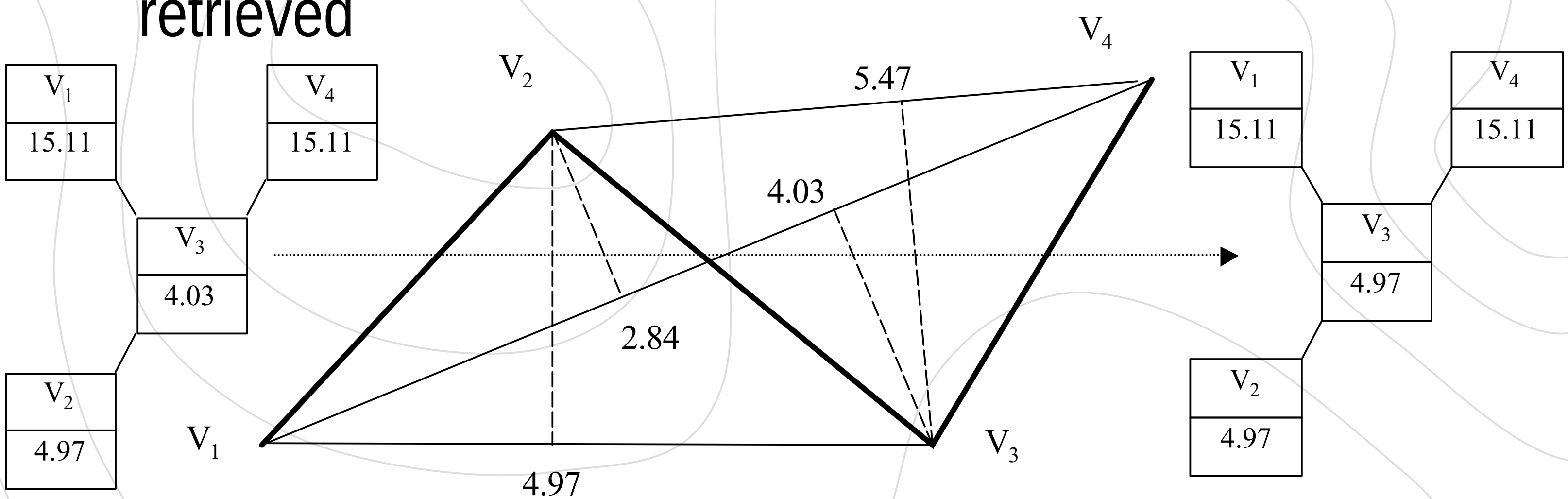


Geometry sectioning using convex hull. The object is divided to three sections: v_1–v_{12}, v_{12}–v_{22} and v_{22}–v_{28}

Progressive Generalisation

– Generation of Multi-Scale Spatial Dataset (9)

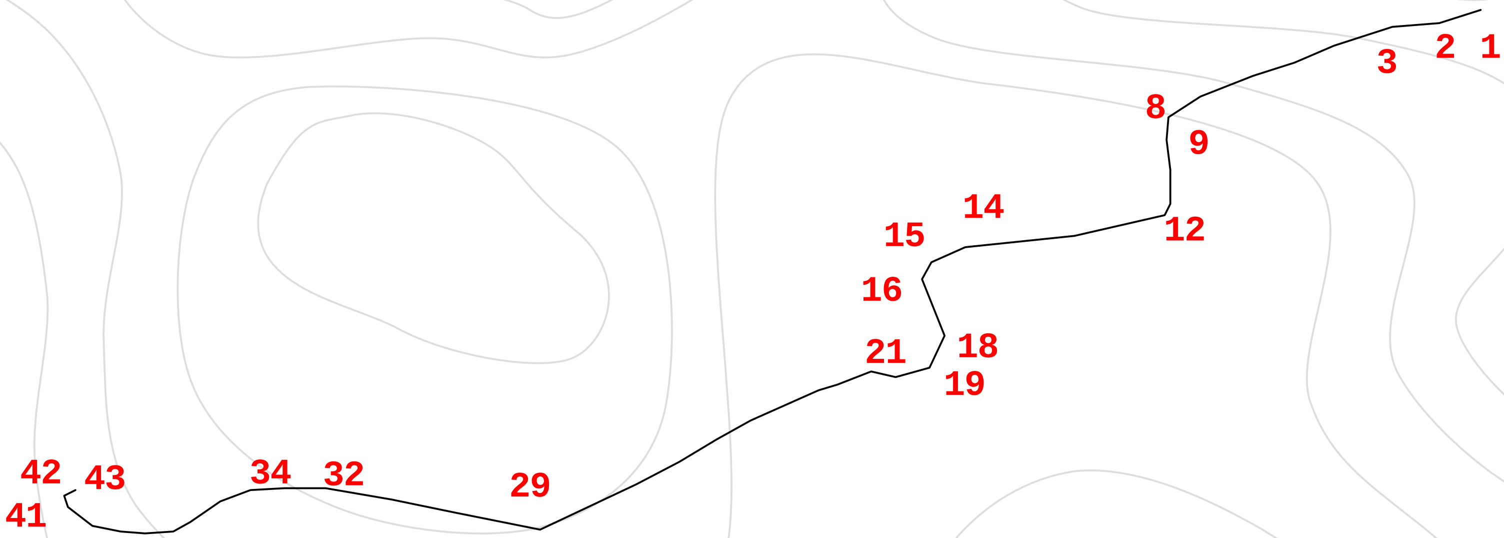
- BVT-tree: tolerance value promotion
  - Tolerance value of v_3 is promoted from 4.03 to 4.97 so that when v_2 at a lower level is retrieved, v_3 will also be retrieved



Progressive Generalisation

– Generation of Multi-Scale Spatial Dataset (10)

- BVT-tree: practical example – the contour

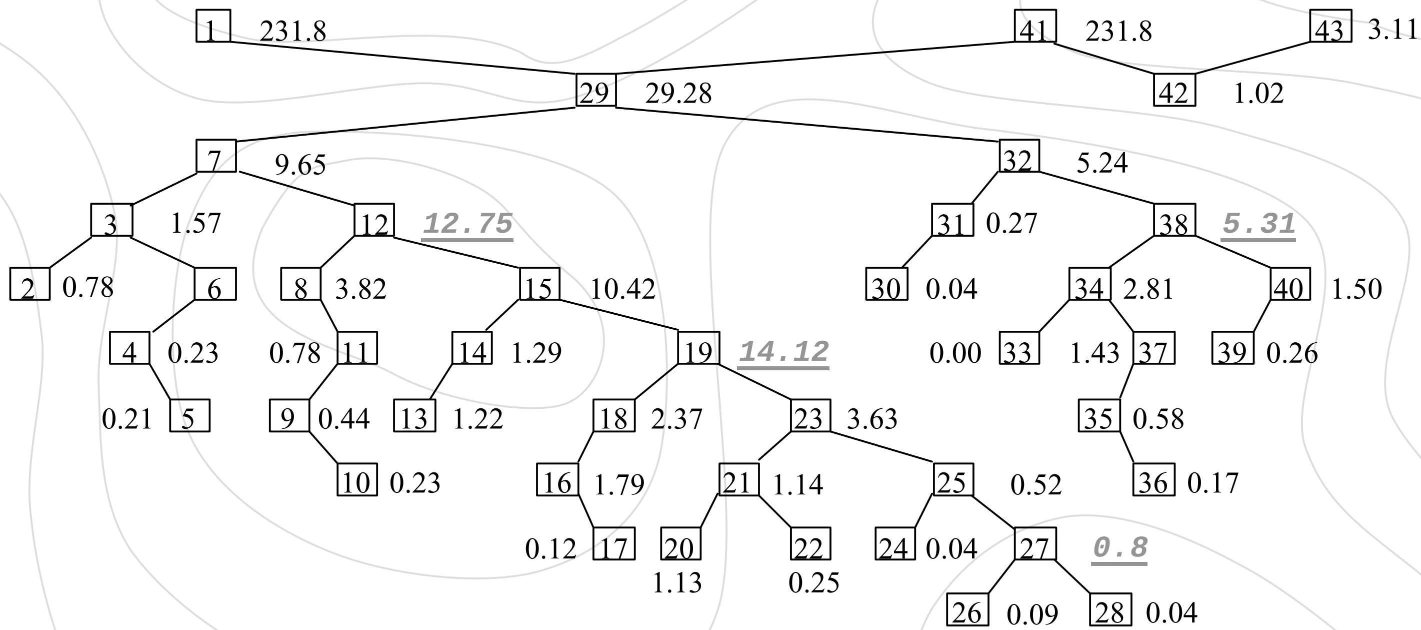


A contour with 43 vertices and two sections($v_{1}\_v_{41}$ & $v_{41}\_v_{43}$)

Progressive Generalisation

– Generation of Multi-Scale Spatial Dataset (11)

- BVT-tree: practical example – initial BVT-tree

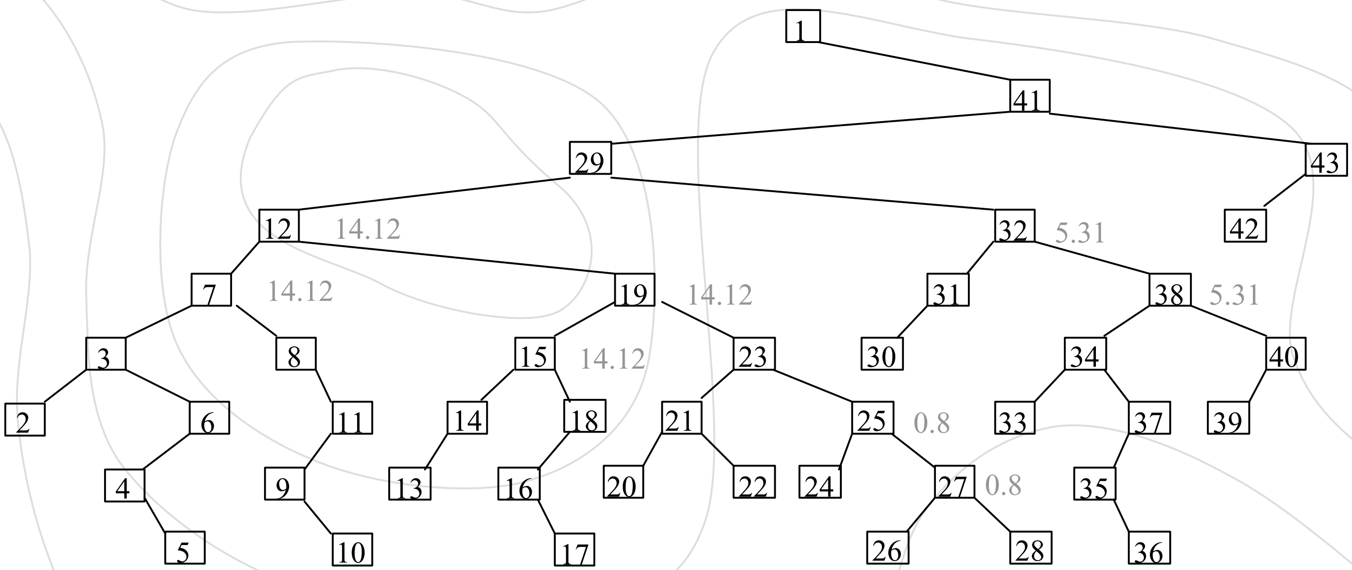


The initial forest-shape BVT-tree (abnormal tolerance values are underlined)

Progressive Generalisation

– Generation of Multi-Scale Spatial Dataset (12)

- BVT-tree: practical example – BVT-tree after tolerance promotion and re-shaping



Progressive Generalisation

– Generation of Multi-Scale Spatial Dataset (13)

- Issues to be addressed:
  - Intra-geometry conflicts (self-intersection) and inter-geometry conflicts (intersection, distance below tolerance, etc.)
- Potential solutions:
  - Applying other generalisation operators to adjust the scale ranges of vertices/geometries
- Future plan:
  - Using dynamic CDT to support progressive generalisation and generation of multi-scale spatial dataset

Presentation of MS-SDB query result – Conflict resolution

- The issue of graphic conflicts in presentation
  - Returned query results should be legible by default, unless…
  - User specifies non-default symbol sizes for various feature types, or…
  - User supplies non-default query parameter values for scale-mapping indicators (e.g. BVT-tree tolerance) so that too many details are retrieved

Presentation of MS-SDB query result – Conflict resolution

- Previous methods to solve graphical conflicts
  - SA - Simulated Annealing (M. Ware & C.B. Jones 1998; M. Ware, C.B. Jones and N. Thomas in this workshop)
  - CA – "Conflict Aggregation" (M. Lonergan and C.B. Jones, to appear)
- Latest progress made by the authors (too late to enter the draft paper)
  - An Agent-style method based on communication and negotiation between objects

Presentation of MS-SDB query result – Conflict resolution

- Summary of the new Agent-style method (1):
  - For an object in conflict:
    - Evaluating conflicts and deciding the direction to move
    - If free space at the direction is insufficient, request the object on its way to move to free some space
    - If not success, return to the source of the conflict and request the source object (if not on boundary) to move

Presentation of MS-SDB query result – Conflict resolution

- Summary of the new Agent-style method (2):
  - For an object receiving request:
    - Evaluating free space on the requested direction
    - If there is sufficient free space, moves on and return its status to the requester
    - If free space is insufficient, passes an adjusted request to the object on the way and waits for its reply to decide next action

Presentation of MS-SDB query result – Conflict resolution

- Result comparison – the test dataset 1 & 2
  - Source: IGN-BDTopo
  - Stat: 367/366 objects (46/45 linear objects as region boundaries and 321 polygonal objects as buildings)
  - Conflict situation (the same as in SA test):
    - Building-Boundary tolerance: 7.5
    - Building-Building tolerance 7.5
    - Maximum Object Displacement: 7.5
    - Initial buildings in conflict: 193/191

Presentation of MS-SDB query result –

Conflict resolution – experiment results (1)

Method	Platform	OS	Dataset	Exec.Time(s)	Moves	Remaining Conflicts
Conflict-Aggregation	PII-233 64M	Win95	1	78	~400 (?)	38
Simulated Annealing	PIII-800 128M	Linux	2	1.55	37283.8 (s.d.1864.7)	26.5 (s.d.2.8)
The New Agent-style method	Mobile PIII-700 128M	NT5	2	0.799	1035	2

Presentation of MS-SDB query result – Conflict resolution – experiment results (2)

- Results WITHOUT Maximum Displacement Constraint

Platform	OS	Dataset	Exec.Time(s)	Moves	Remaining Conflicts
Pentium-133 / 24 M	Win98	1	4.844 (MGLIB)	1119	0
			3.801 (Triangle)	862	0
PIII-600 / 128 M	NT4	1	0.723 (MGLIB)	829	0
			0.847 (Triangle)	1119	0
Mobile PIII-700 / 128 M	NT5	1	0.805 (MGLIB)	951	0
			0.721 (Triangle)	868	0
		2	0.533 (MGLIB)	628	0

Presentation of MS-SDB query result – Conflict resolution

- Implementation illustrated (1)
  - Programme is written in C++
  - Exec. time shown are averages of 5~10 runs. As results are very stale, s.d. is not calculated
  - CDT is used for conflict and free space detection. CDT creation time (excluded from timing shown earlier) is 0.4 -1.0s on P133 and <0.1s on PIII-600/Mobile PIII-700. Alternatively, a dataset with pre-generated CDT may be used (as in M. Ware, etc.)
  - Both Triangle (a C programme by J.R. Shewchuk) and MGLIB (my own C++ dynamic CDT framework) have been used to generate CDT.

Presentation of MS-SDB query result – Conflict resolution

- Implementation illustrated (2)
  - Due to some internal implementation features and possibly round error of float calculations, CDT meshes generated by the two programmes (identical in geometry but with different internal edge orders) may result in different number of moves and hence different exec. time, although the final outputs of conflict resolution are virtually identical

Presentation of MS-SDB query result –
Conflict resolution: Original dataset 1

Presentation of MS-SDB query result – Conflict resolution: Original dataset 2

Presentation of MS-SDB query result –
Conflict resolution: output (ds2 without MDC)

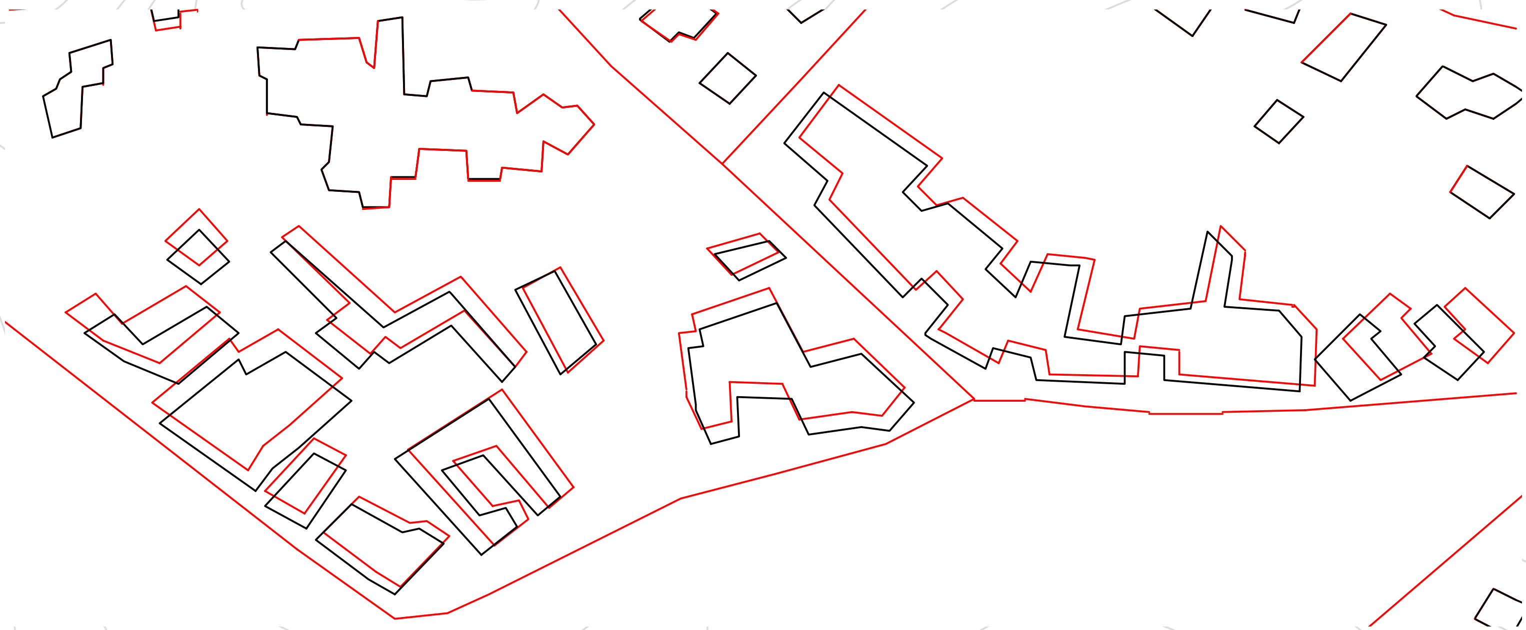
Presentation of MS-SDB query result –
Conflict resolution: output (ds2 with MDC)

Presentation of MS-SDB query result –
Conflict resolution: details-1 (without MDC)

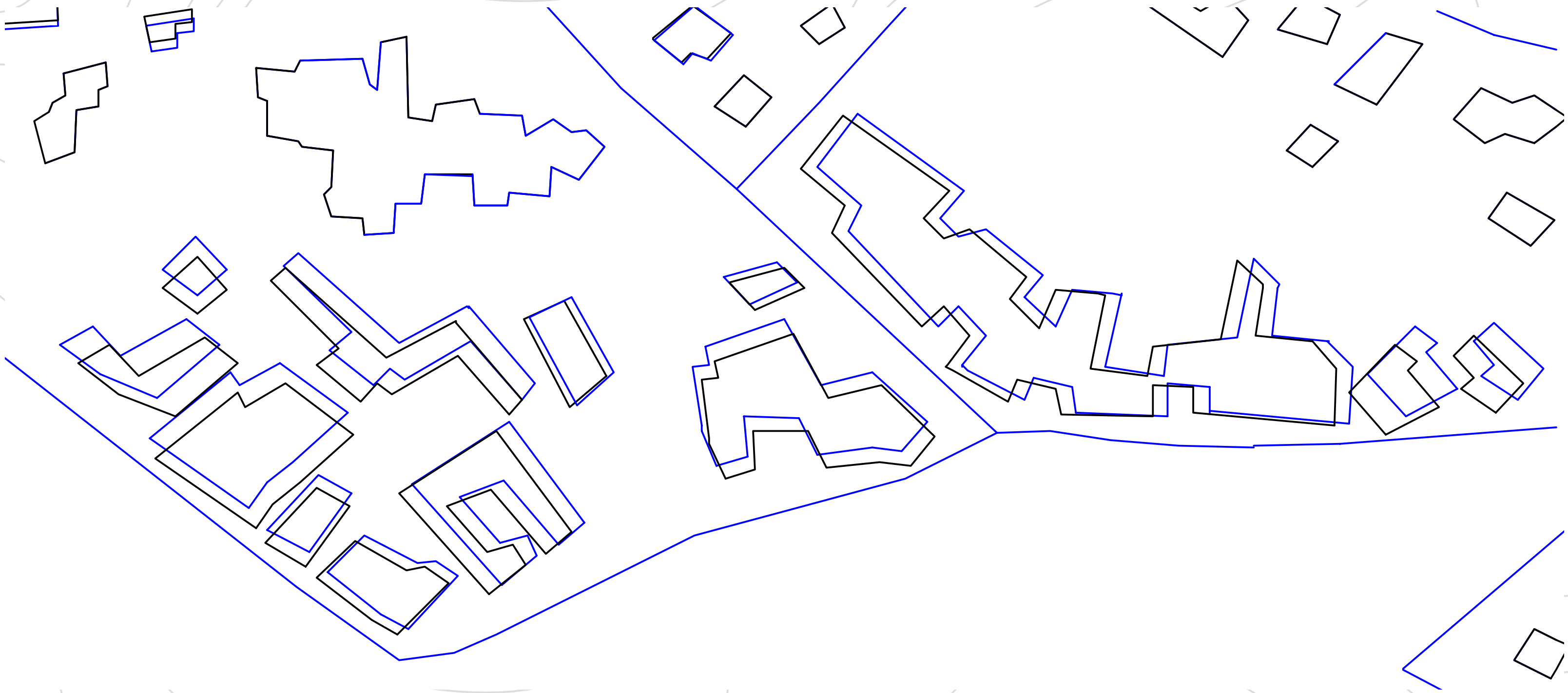
Presentation of MS-SDB query result –
Conflict resolution: details-1 (with MDC)

Presentation of MS-SDB query result –

Conflict resolution: details-2 (without MDC)



Presentation of MS-SDB query result –
Conflict resolution: details-2 (with MDC)



Presentation of MS-SDB query result –
Conflict resolution: details-3 (without MDC)

Presentation of MS-SDB query result – Conflict resolution: details-3 (with MDC)

Presentation of MS-SDB query result – Conflict resolution – Discussion (1)

- Advantages of the new method
  - Most conflicts are removed – indeed, all conflicts can be removed for a reasonable tolerance value and without maximum displacement constraint
  - Efficient enough for on-line applications (at present, may require a PIII 450 or better to break the one sec barrier)
  - Objects are NOT moved unless in conflict or requested by other objects, so positional inaccuracy caused by conflict resolution is minimised
  - No need for partitioning – in principle, the gain from partitioning would be minimal for this method

Presentation of MS-SDB query result – Conflict resolution - Discussion (2)

- Further improvements
  - Re-engineering the programme
    - Currently it is an experimental programme completed in a very short period, containing many ad hoc patches made during its evolution
    - To get rid of all virtual function calls (a price paid for using the generic CDT framework) and use customised memory management scheme more comprehensively
  - Designing better free space detection procedures (which seems to be the bottle-neck at present) and test other alternatives of strategies for object interaction
  - Target set at: 0.5sec on my PIII 600-128M NT4 machine when maximum displacement constraint is applied

Presentation of MS-SDB query result – Conflict resolution - Discussion (3)

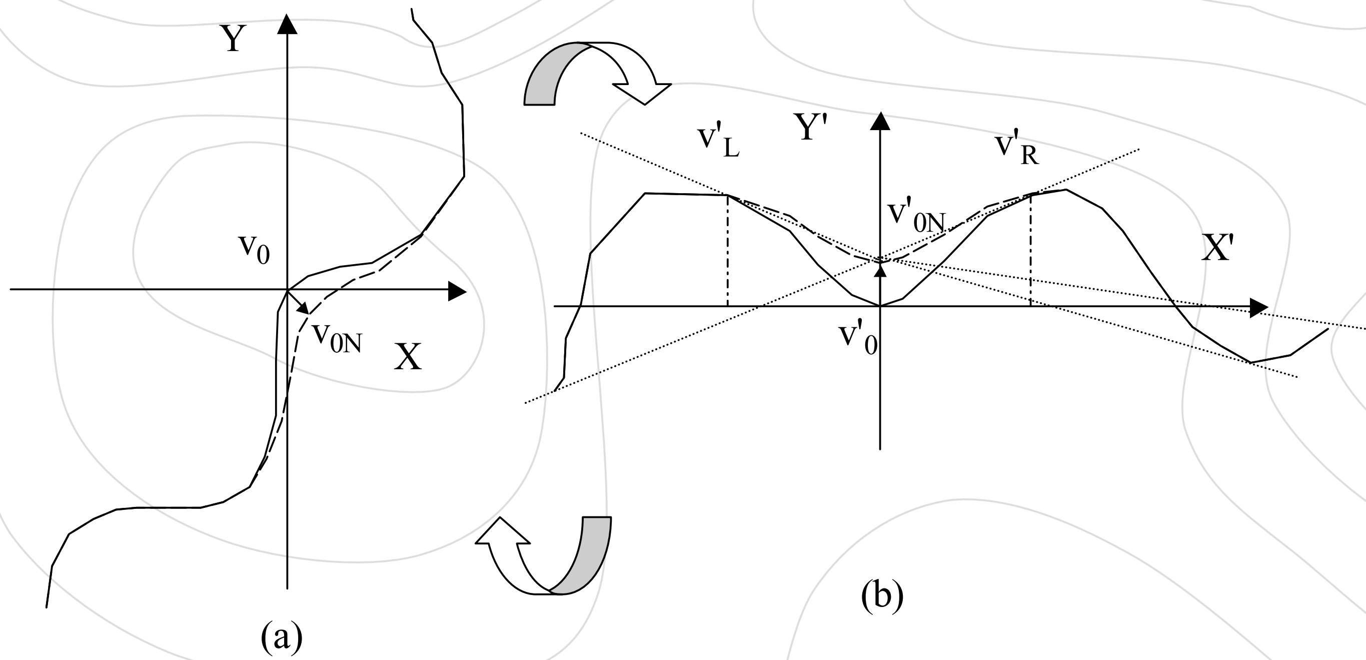
- What if it fails?
  - In theory it should be rare to find out that there is simply not enough free space; otherwise, either the query generator is terrible or the user is mad to have requested too many details!
  - To be user-friendly, under such circumstances, we may apply some methods introduced by (M. Ware, C.B.Jones and N. Thomas in this workshop)
  - Or, push the boundary…

Presentation of MS-SDB query result – Conflict resolution – Line displacement

- A shape-preservation method for linear feature displacement
  - Agent-style (again!)
  - Vertices communicate and negotiate with each other to distribute displacement gradually over the whole feature and at the same time preserve all basic curves on the feature
  - Detection of potential new conflicts is not addressed at present but will be handled in future implementations

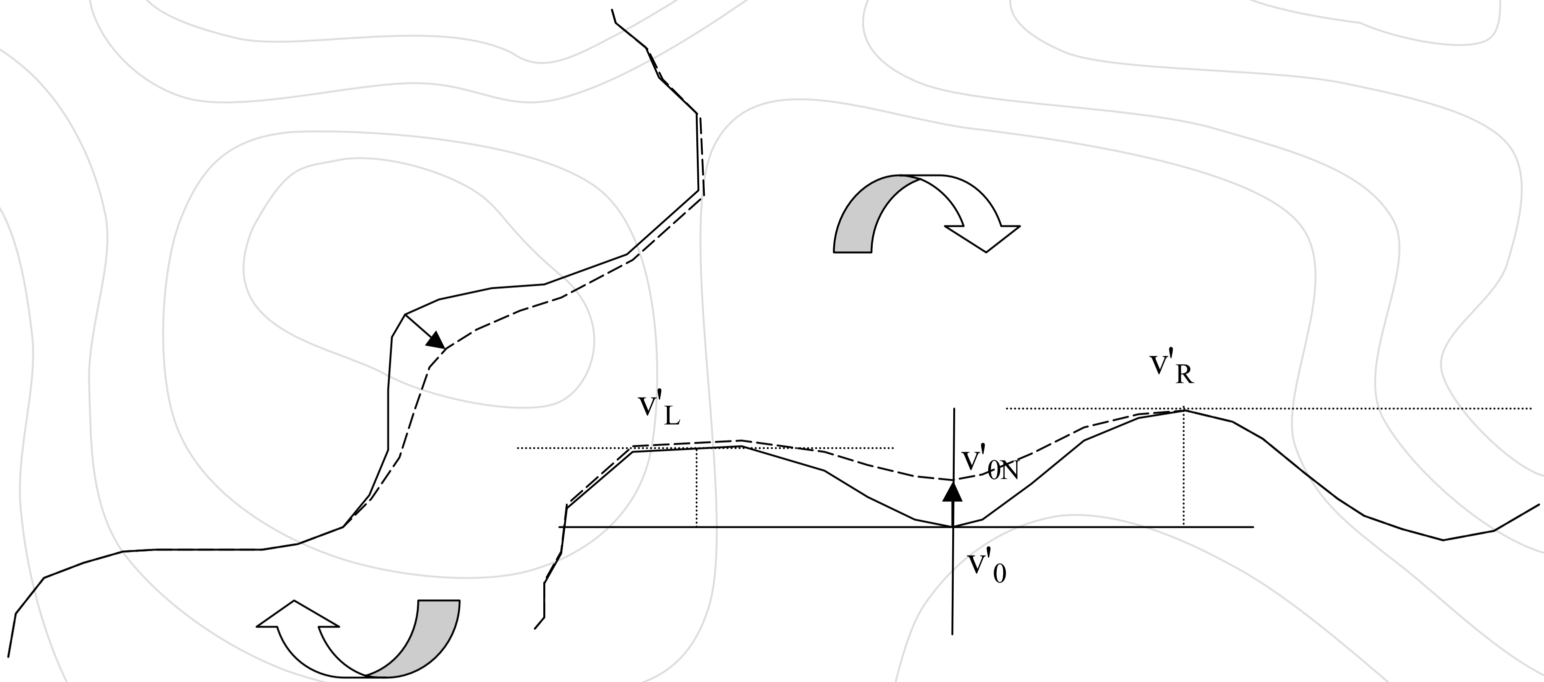
Presentation of MS-SDB query result – Conflict resolution – Line displacement

- Scheme 1: sectioning at tangent points



Presentation of MS-SDB query result – Conflict resolution – Line displacement

- Scheme 2: sectioning at extreme points



Presentation of MS-SDB query result –

Conflict resolution – Line displacement

- Result comparison – better result may be achieved by combining the two schemes

- That's all!
- Thanks for your time and attention!!!