An ESRI® Technical Paper • March 2010

Understanding Geometric Processing in ArcGIS®

ESRI 380 New York St., Redlands, CA 92373-8100 USA
TEL 909-793-2853 • FAX 909-793-5953 • E-MAIL info@esri.com • WEB www.esri.com

J-9880

Understanding Geometric Processing in ArcGIS

An ESRI Technical Paper

Contents **Page**

Understanding Geometric Processing in ArcGIS

Introduction

This is the second paper in a series that explains how ArcGIS® stores and processes the coordinates in your vector geographic information system (GIS) datasets. The first paper, *Understanding Coordinate Management in the Geodatabase,* explained that properties of a spatial reference affect the storage and resolution of your coordinate data.

This paper explains how ArcGIS interprets your coordinates when making spatial decisions (for example, selecting features near other features on a map), and how it generates new coordinates for new geometric objects (buffering, intersection, etc.).[1]

These and other operations are implemented in ArcGIS using a software component called the topology engine. The topology engine is used by the geometry, geodatabase and map topologies, editor, and geoprocessing systems. It is robust and efficient. Robustness is achieved by applying a process, *integration,* to the input data before any outputs or decisions are computed. This process prevents various kinds of computational errors that could generate invalid results or cause software failures. Integration has implications that you should be aware of if you are using ArcGIS to design geodatabases, make decisions based on spatial relationships between features, or edit features.

Three new spatial reference properties—the xy-tolerance, z-tolerance, and m-tolerance—were introduced at ArcGIS 9.2 to control integration in every context in which it occurs. The xy-tolerance is also referred to as the *cluster tolerance* in this paper and other ESRI documentation. You may already be familiar with this parameter in a more isolated context if you have worked with a geodatabase or map topology. At 9.2, this parameter is associated with every spatial reference. The cluster tolerance is used in two different ways by the integration process. The goals of this paper are to

- Describe integration.
- Show how integration uses the cluster tolerance.
- Recommend choices for an appropriate cluster tolerance.

This paper introduces integration through some basic examples. It describes how the coordinate grid resolution, cluster tolerance, and integration process affect the final integrated state of your geometric data.

Introduction to Integration

ArcGIS performs geometric computations on coordinates that have been snapped to a coordinate grid. The horizontal or vertical distance between each grid point is the *resolution* of the grid. Figure 1 shows two line segments stored on such a grid. The endpoints of the line segments can be placed exactly on grid points; thus, the endpoints all have the same resolution. Very few other points on the lines can be placed exactly on grid points. In particular, the intersection point (see inset) cannot be placed exactly on a grid point.

[1] The information in this document describes the behavior of ArcGIS 10 and prior releases. In future releases, this behavior may change.

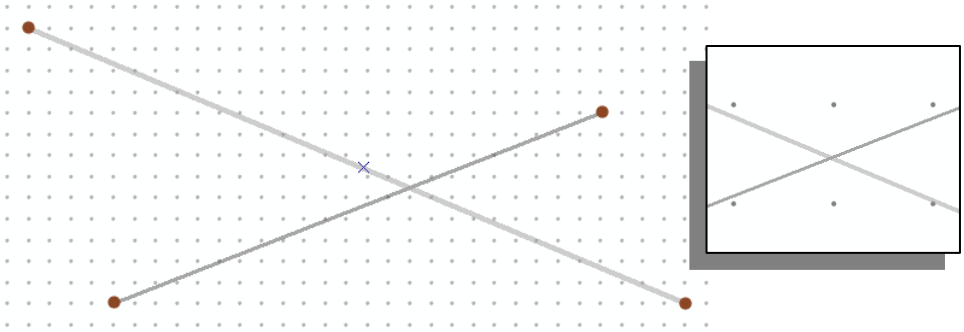

Figure 1: Two line segments stored on an integer grid. The coordinates of their endpoints (black dots) can be represented exactly on the grid. They are thus all represented at the same resolution. The intersection point (insert) cannot be represented at the same resolution.

Figure 2 shows the results of moving the intersection point to the nearest grid point and replacing the two input line segments with *four* new line segments that share the new point. The intersection of the input lines is now represented with the same resolution as their endpoints.

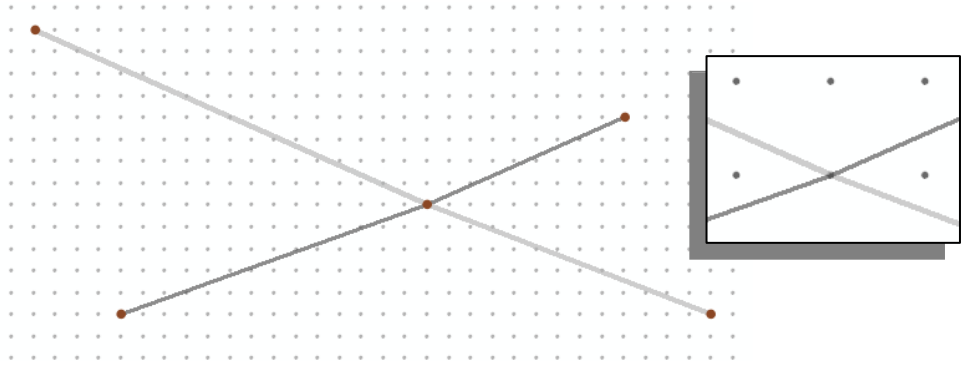

Figure 2: The two lines replaced with four lines coincident on a grid point closest to the original intersection point. The intersection point is now represented with the same resolution as other endpoints.

As a second example, in figure 3, the distance between these line segments is less than the grid resolution. The touch relational operator should report that these lines touch.

Figure 3: These two line segments should be classified as touching because the distance between the endpoint and the interior is less than the resolution.

This decision becomes clearer if the subresolution separation is removed, as shown in figure 4. The relational operator then uses the modified set of line segments as its input.

Figure 4: Two line segments that have been modified to remove a subresolution separation. The inset shows the position of the original line and the bent, snapped line.

As subresolution separations in the original data are removed and intersection points are moved to grid locations, other subresolution separations and overshoots may be created. Figure 5 shows a third vertical line segment added to the set of input segments in figure 3. Initially, this new segment is separated from the horizontal segment by a distance greater than the resolution. After the adjustments shown in figure 4, that is no longer the case. A second adjustment is required to remove the new subresolution separation.

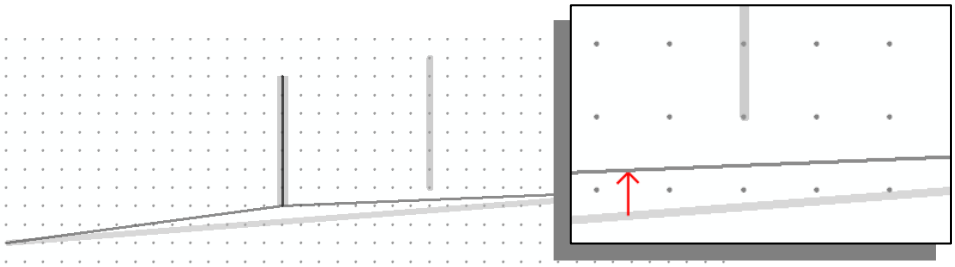

Figure 5: A subresolution separation created by removing another subresolution separation. As the lower line shifts upward to meet the left vertical line, it also approaches the right vertical line (inset).

Figure 6 shows the final result. The horizontal line segment has been bent in two places: on the left to remove a subresolution separation that was present in the input data, and on the right to remove a second separation caused by removal of the first one.

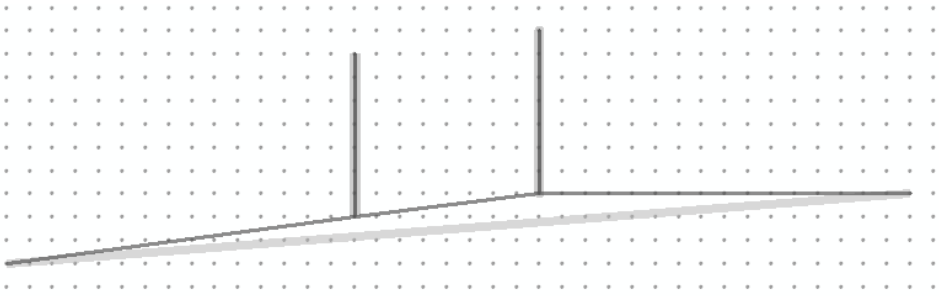

Figure 6: Multiple subresolution separations are removed.

The end result of these adjustments is an *integrated* representation of the input dataset in which *all* coordinates are explicitly represented with the same resolution, and all spatial relational facts about the dataset are explicitly represented *and* consistent with that resolution. The process of moving from input data with coordinates potentially expressed at a variety of resolutions, and with spatial relationships implicitly represented, to an integrated dataset is called *integration.*

In summary, there are several reasons why geometric processing in ArcGIS always integrates data before computing new geometric objects or making decisions based on existing geometric objects:

- Integration, along with a coordinate grid, helps prevent numerical problems arising from naive use of floating point numbers.
- Integration helps you control the resolution of your data. It helps prevent very small geometric objects (e.g., sliver polygons, small gaps, and overshoots) from affecting the answer to a construction operation or a relationship query.

Integration and the Cluster Tolerance Parameter

In practice, integration is not directly controlled by resolution, as illustrated in the introduction. Rather, for reasons that will be covered later, a separate distance parameter, called the *cluster tolerance,* is used to determine when pairs of points, or points and lines, are too close to each other or when line segments are too short.

You should keep in mind that the adjustments made to your data to integrate it, as illustrated in the previous section and upcoming examples, are of the same order of magnitude as the chosen cluster tolerance. The default cluster tolerance is 1 mm. Some statistics regarding average and worst-case point movement during the integration process are given in An Integration Example below.

In the following figures, assume that the coordinate grid resolution is much smaller than the default cluster tolerance. For example, the coordinate grid resolution sufficient to cover all of universal transverse Mercator (UTM) Zone 11N is on the order of micrometers. Therefore, when line endpoints or intersection points are shown, it will be understood that they are being snapped to the underlying grid, although the grid itself will not be shown.

This section gives a precise description of the algorithm used to integrate data. This algorithm is called *cracking/clustering*. As shown below in figure 7, cracking modifies intersecting line segments by calculating their intersection point, moving that intersection to a grid point, then updating the set of four new segments to be coincident on it. A segment can also be cracked by a nearby endpoint, called a projection crack point. *Nearby* in figure 7(b) means that segment B is less than $\sqrt{2}$ times the cluster tolerance units from the lower endpoint of segment A. The red circle is used to indicate the corresponding search area, with that radius, around A's endpoint.

This aspect of the integration process can cause confusion. The cluster tolerance that you specify, either in a geodatabase or map topology or as a property of a spatial reference associated with a feature class or a feature dataset, is a measure of how far a point *is allowed to move per cracking/clustering iteration.* Iteration will be explained shortly. In

addition, the distance actually used for clustering is *larger* than your specified cluster tolerance by a factor of $\sqrt{2}$.

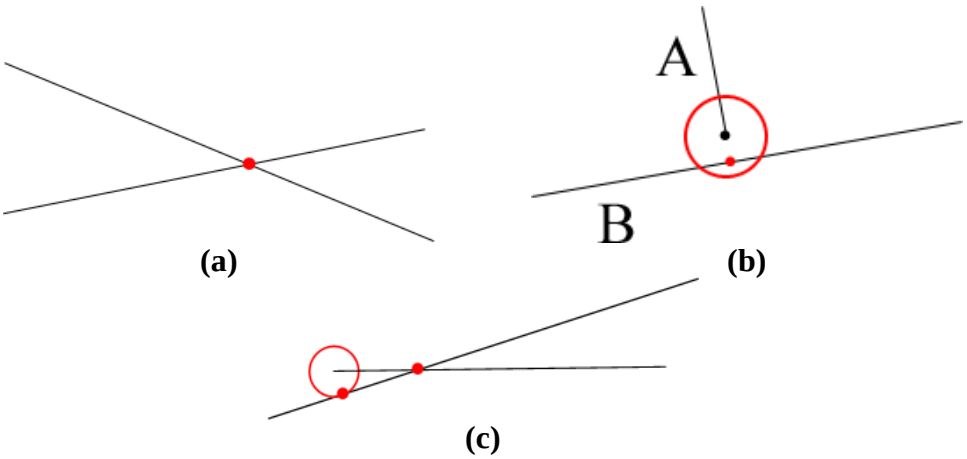

Figure 7: The different kinds of segment cracking. In illustration (a), the red dot shows a segment intersection crackpoint. In illustration (b), it identifies an endpoint projection crackpoint. The lower endpoint of line A can move anywhere in the region shown by the red circle, which has a radius of $\sqrt{2}$ times the cluster tolerance. Line B passes through this circle and will thus have an endpoint projection crackpoint inserted into it. Potentially, as shown in illustration (c), a pair of lines can receive both intersection and endpoint crackpoints during cracking.

After line segments have been cracked, *clustering* is performed. Clustering replaces groups of nearby points with a single representative point, then updates any affected segments incident on those points. As with cracking, nearby means a distance of $\sqrt{2}$ times your specified cluster tolerance. In figure 8(a), a pair of nearby endpoints is shown. The red circles represent the distances that each endpoint is allowed to move. Since the circles overlap, the points will be clustered. The chosen cluster point lies along the line joining the two endpoints and is exactly halfway between them, as shown in figure 8(b).

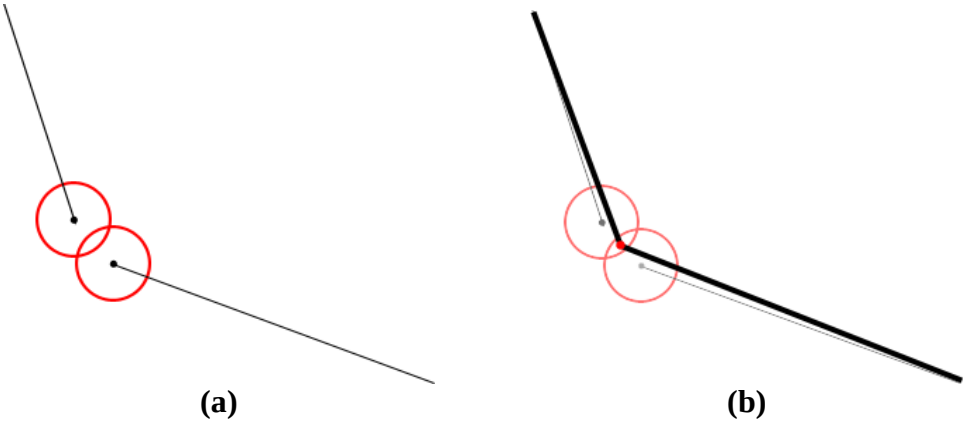

Figure 8: Two endpoints are close enough to be clustered together.

As a result of clustering, line segments whose length is less than $2\sqrt{2}$ times the cluster tolerance are deleted.

Figure 9 shows the effects of both intersection and projection endpoint cracking, followed by clustering. The inputs are shown as thick gray lines. The integrated outputs are shown as narrow black polylines that share a common segment in the upper right.

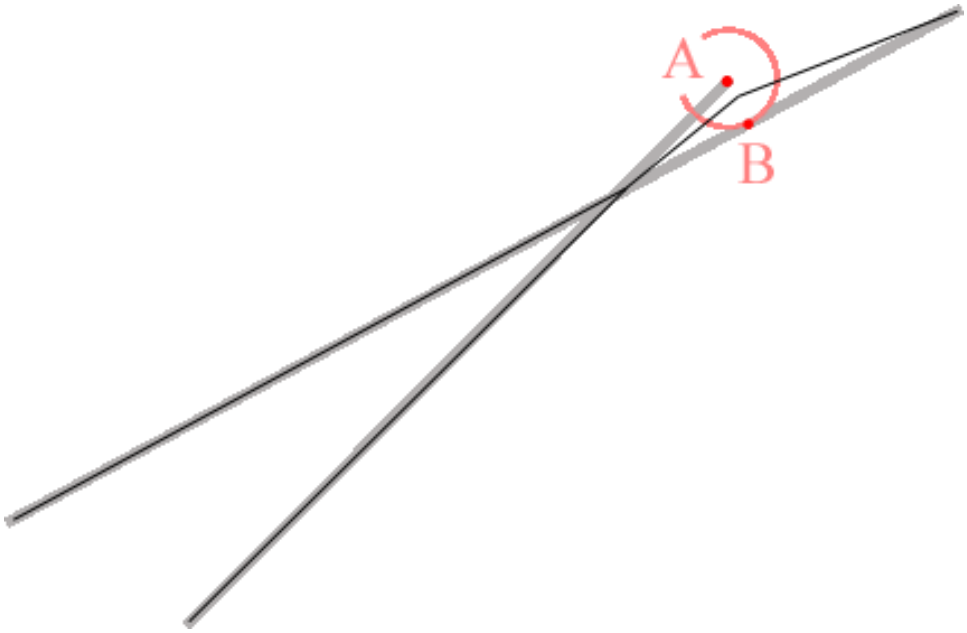

Figure 9: The results of clustering when both an intersection and projection crackpoint are involved. Endpoint A has caused crackpoint B. The two points were then clustered together.

The term *clustering step* describes the process of replacing exactly two points with a single representative point. Figure 8 shows a clustering step. A single *clustering operation* has one or more clustering steps. Figure 10 shows a clustering operation involving endpoints A, B, and C. They are combined because a cluster point can be chosen that is within ($\sqrt{2}$ x cluster tolerance) units of each endpoint. A *clustering pass* consists of multiple clustering operations applied to different geographic locations in the data.

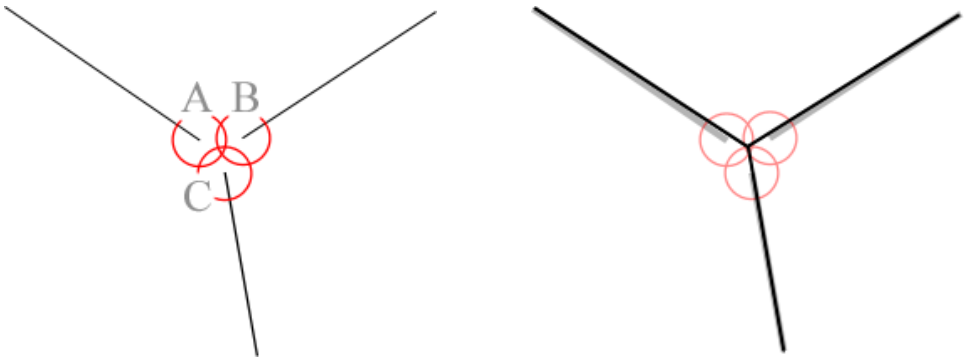

Figure 10: A clustering operation consisting of two clustering steps. Endpoints A and B are combined in the first step, and that cluster center is combined with endpoint C in the second step.[1]

[1] Since the distances between A, B, and C are equal, the steps of this clustering operation could have happened in several different orders.

Within a single clustering pass, any point can move by at most ($\sqrt{2}$ x cluster tolerance). In figure 11(a), the endpoints of lines A and B are clustered, producing the configuration shown in figure 11(b). A and B are clustered first because they are closer to each other than to the next closest endpoint, C. In figure 11(b), it might appear that another cluster could be formed, but that would cause A's endpoint to move farther than ($\sqrt{2}$ x cluster tolerance) from its original location; hence, the second cluster is *not* formed.

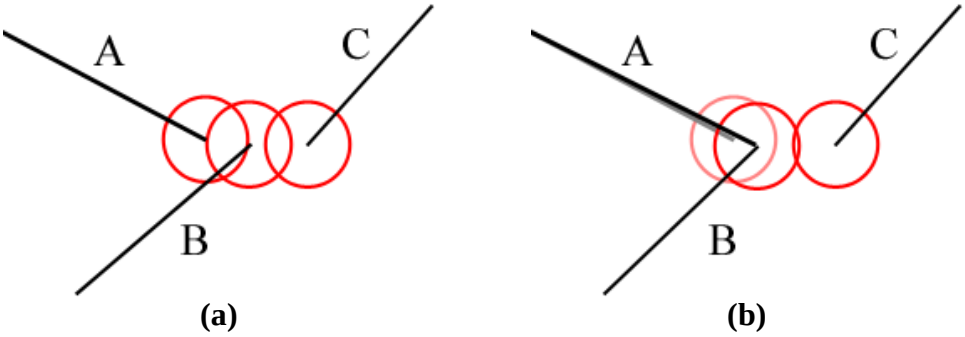

Figure 11: Rejection of a cluster center candidate. The clustering step being proposed in illustration (b) will not happen because A's original location would shift too far.

Since clustering moves endpoints, it may be necessary to perform additional cracking and clustering passes. The integration process terminates when the data passes a set of three tests. The tests are called the Milenkovic conditions,[1] or *M conditions* for short. They require that

- No two vertices are closer than the cluster tolerance.
- No vertex is closer than the cluster tolerance to another edge.
- No two edges intersect except at their endpoints.

Here are some important observations:

- The integration process uses a distance of $\sqrt{2}$ times the cluster tolerance, but the M condition check uses the cluster tolerance *directly.* The extra factor applied to the process ensures that the M conditions can always be tested without ambiguity.

- When presenting geometric inputs to the topology engine, an initial cracking pass is *always* performed.

- At least one clustering pass is performed after the initial cracking pass.

Point Weights

A point weight is a number attached to each input point. It is used in combination with the cluster tolerance to select the location for a computed cluster point. The point chosen as a cluster center during a clustering step will be on the line between the two input points, closer to the input point with the greater weight. The weight of the cluster center will be the sum of the input points' weights.[2]

[1] Milenkovic, Victor, "Verifiable Implementations of Geometric Algorithms Using Finite Precision Arithmetic," *Artificial Intelligence* 37:377–401, 1988.

[2] Point weights and ranks, as assigned to feature classes in a topology, are related but have different ways of controlling the integration process. This paper does not discuss ranks. The reader should assume that all input points have a weight of 1.

Point weights are maintained across clustering passes, whereas a point's *original input location is not.* It is therefore possible for a point to move a total distance *greater than* $\sqrt{2}$ times the cluster tolerance if it is involved in multiple clustering steps in different clustering passes. During this process, though, its weight will increase each time it is involved in a clustering step. Thus, it will tend to move less and less in the second and subsequent steps.

Point weights tie output point locations to input point locations while giving the integration process flexibility in creating and maintaining a constant level of resolution. Within a single clustering pass, the final location chosen for a cluster point is exactly the centroid (center of mass) of the several input points that were involved, two at a time, in the clustering operation.

In figure 12, four input lines—A through D—are shown in light red. The integrated results are shown in black. In this case, the coordinate resolution grid is also shown, with horizontal and vertical spacing between dots being one unit. A and B share a common endpoint. Assume that initially 1 is the input weight for the endpoints of the lines. The cluster tolerance is 5. If each endpoint were allowed to move up to 5 * $\sqrt{2}$ units, a single cluster would be formed and all segments would end up being connected. However, this is not what has happened. Line D has stayed separate.

Review each clustering step in sequence and determine how point weights affect the result.

1. The common endpoints of A and B are clustered first, producing a cluster point with weight 2.

2. The upper endpoint of C is then added to the cluster, with the resulting cluster point located beneath A and B but closer to them than to C. This point has a weight of 3.

3. In a third potential cluster step, the lower endpoint of D is considered. However, if the cluster center were chosen as the weighted affine combination of D and A,B,C, then D would have to move farther than 5 * $\sqrt{2}$ *to reach that center,* so that cluster is *not* formed and the cluster operation in that area is finished.

In the output, the final cluster point appears halfway between C and A and B, but that's due to the coordinate grid resolution.

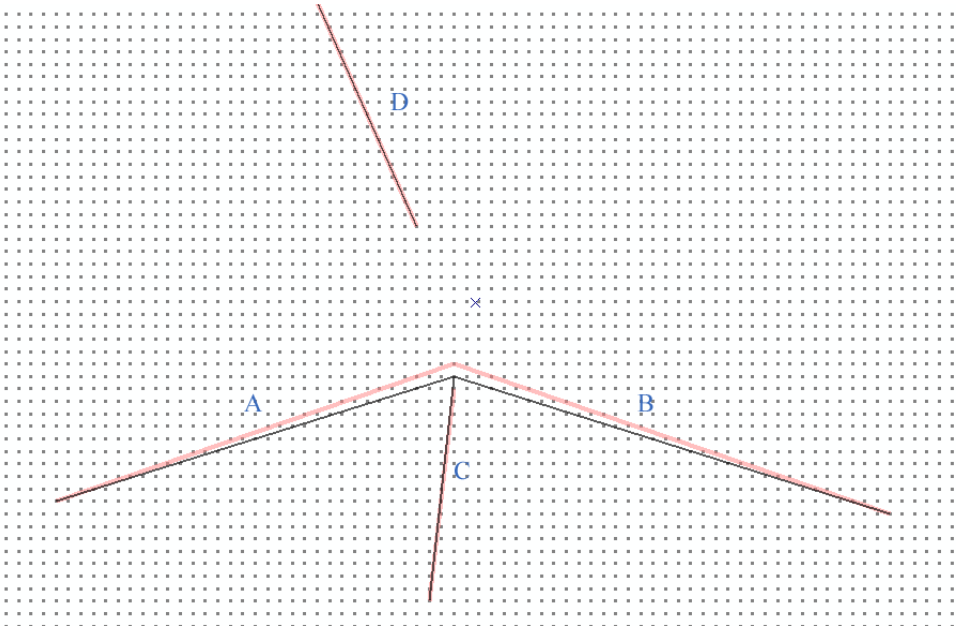

Figure 12: Effect of point weights on the integrated state of four lines, case 1. The cluster tolerance is $5\sqrt{2}$. The endpoints of A, B, and C have been clustered together, but the resultant higher weighted cluster point cannot be combined with D's endpoint, even though the distance between them is less than $2 * 5\sqrt{2}$.

In contrast, in figure 13, D initially has its lower endpoint closer to A and B, as marked by the red arrow. The distance it travels to reach the cluster center is less than $5\sqrt{2}$, and thus the third clustering step can be performed.

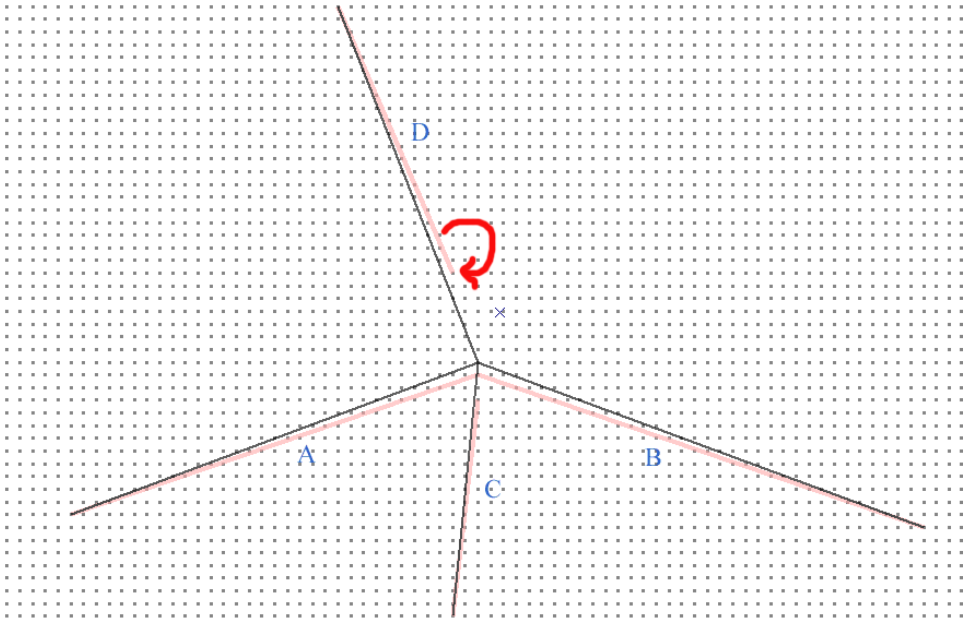

Figure 13: Effect of point weights on the integrated state of four lines. Contrast with figure 12. D's endpoint has been shifted down slightly from its position in figure 12. All four endpoints can now be clustered together.

The Cracking/Clustering Algorithm

As mentioned above, the cracking/clustering procedure is iterative. There is always an initial cracking pass, followed by an initial clustering pass. Additional passes may be required after that, as explained in the discussion relating to figure 5. The following pseudocode expresses the basic cracking/clustering loop:

```
Iteration = 0;
Assume M conditions are not satisfied
while (M conditions are not satisfied or Iteration == 1) {
  Iteration = Iteration + 1;
  if (Iteration <> 1)
    Cluster()

  Crack();
  Check M conditions;
}
```

In rare situations, all M conditions might not be satisfied. In that case, the maximum number of iterations is limited (currently to 10). If intersections still exist at this point, an error is reported. The details of how this is done are beyond the scope of this paper.

Selecting a Good Cluster Tolerance

During a clustering pass, a point cannot move farther than $\sqrt{2}$ * cluster tolerance. The worst case is that each additional clustering pass will cause a point to move in the same direction by the maximum amount. The input conditions required to cause this would be extremely rare, if even possible. It is more likely that a point without a stable integrated location will oscillate between other points during successive clustering operations. In most cases, the number of points that are altered in a given cracking/clustering pass is a small percentage of the ones that were affected in the previous pass.

How does this translate into a good value for the cluster tolerance parameter? **ESRI's advice is to use a value 10 times less than the minimum separation that you wish to preserve.** More specifically, if the worst-case movement per iteration is $\sqrt{2}$ * (cluster tolerance) and there are C iterations, then the minimum separation d between distinct points must be d > 2 * $\sqrt{2}$ * (cluster tolerance) * C, or cluster tolerance $\approx \frac{d}{10}$, assuming C = 3 iterations on average.

Choosing a Nondefault Cluster Tolerance

Transferring Data Created with ArcGIS 9.1 or Earlier into an ArcGIS 9.2 Geodatabase

Your data may have topological facts that are only implicitly represented. For example, in figure 14(a), a line is being snapped to another line using the ArcGIS editor (its edge snapping mode being in effect). When the new feature is saved, its coordinates are snapped to the coordinate resolution grid associated with the spatial reference of the feature class being edited. The result is shown in figure 14(b). Thus, the editor doesn't get the last word in when snapping. ArcGIS 9.1 geometric operations were designed to work with features such as this that are only *implicitly* integrated relative to the resolution. Relational decisions, such as touch, take resolution into account.

J-9880

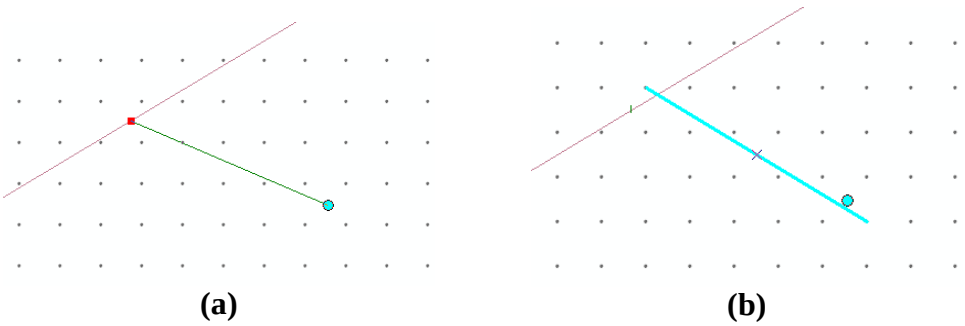

Figure 14: Edge snapping in the ArcGIS editor is shown in illustration (a); in illustration (b), the same edge is shown after being snapped to the underlying coordinate resolution grid.

Now assume that this implicitly integrated data is moved from its current low-resolution grid to a high-resolution grid, as shown in figure 15(a) and 15(b). Observe that the high-resolution grid is aligned with the low-resolution grid but is 10 times more detailed. The lines are no longer touching, based on this new resolution.

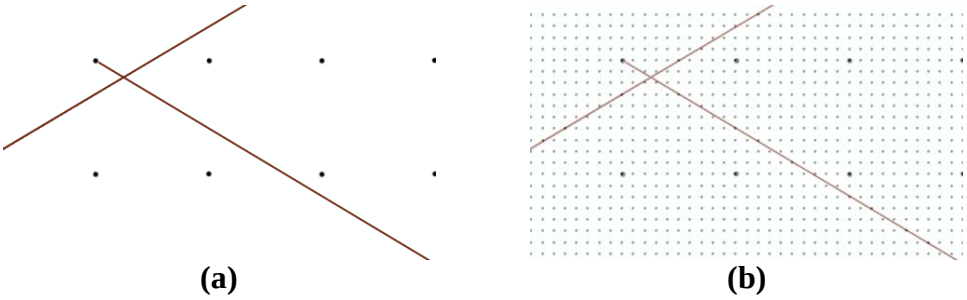

Figure 15: Two lines with an implicit touch relationship relative to an ArcGIS 9.1 coordinate resolution grid are shown in illustration (a); in illustration (b), the implicit touch relationship no longer exists after the data has been brought into a high-precision ArcGIS 9.2 database and upgraded to a high-resolution grid.

To preserve the ArcGIS 9.1 interpretation of the data, you should define the cluster tolerance of the ArcGIS 9.2 data to be at least as large as the ArcGIS 9.1 resolution. The Upgrade Spatial Reference geoprocessing tool defines the cluster tolerance of the dataset being upgraded as 2.0 * (9.1 resolution).

The Spatial Reference and Topology Cluster Tolerances Should Match

If you're upgrading a geodatabase topology, you may want the cluster tolerance of the ArcGIS 9.2 feature dataset to match the cluster tolerance of the ArcGIS 9.1 topology.

An Integration Example

The example below (figure 16) describes the results of integration as performed during a geodatabase topology validation operation, including how many points were shifted from their input locations and how far those shifts were. Repeated integration does not affect vertices that have no need of being modified. There are some aspects to topology validation that are not discussed in this paper.

Validating U.S. Counties and Hydrogeographic Regions

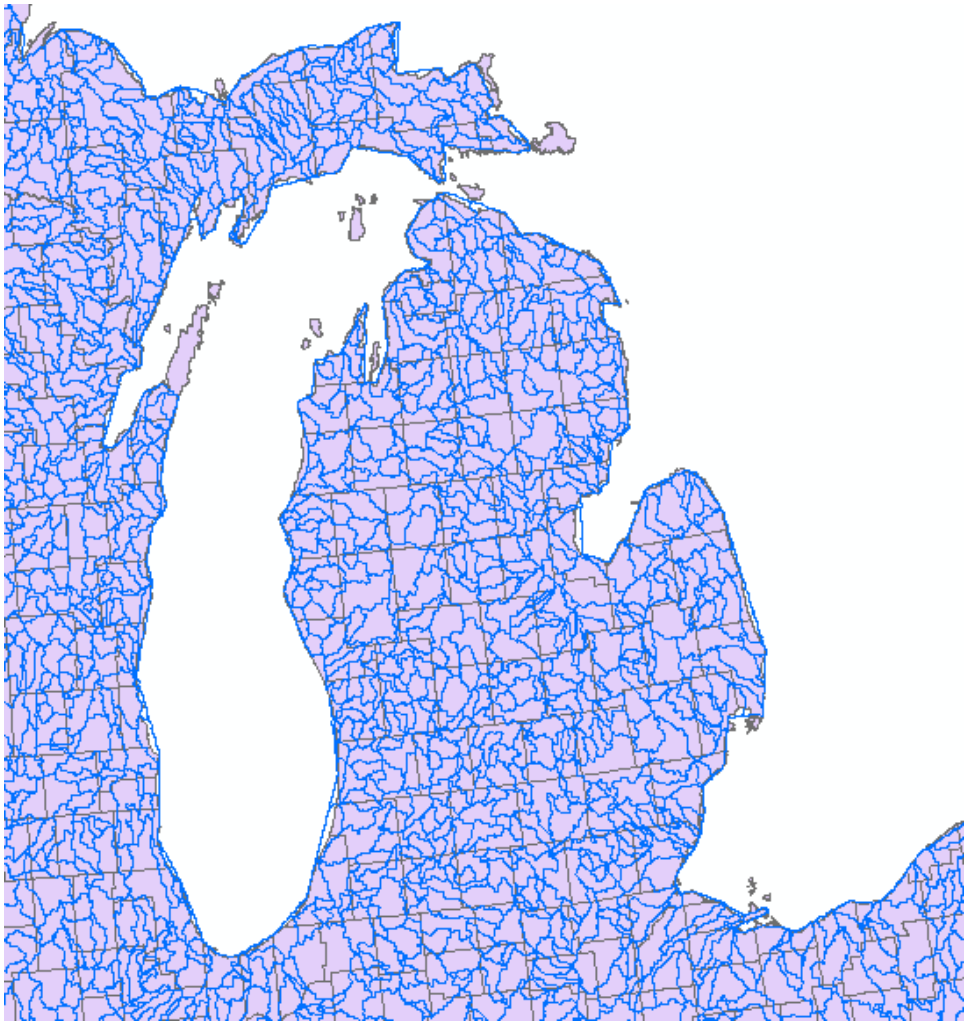

Figure 16: Detailed counties and hydrogeographic regions are shown in this example.

A geodatabase topology containing two national-scale feature classes was validated: a detailed representation of the counties of the United States and a classification of hydrogeographic regions.[1] The topology cluster tolerance was 1 m. This tolerance was appropriate for the hydrogeographic regions given that it was prepared from 1 km resolution digital elevation models (DEMs), but it was probably too large for the detailed counties. Changes made by the integration process to the counties feature class are a result of the validation.[2]

On input, the counties feature class contained 3,341,464 vertices and the hydrogeographic regions feature class contained 3,891,783 vertices.

[1] http://water.usgs.gov/GIS/metadata/usgswrd/XML/hlrus.xml.

[2] The feature classes were equally ranked (ranks are not discussed in this paper—for more information, see ArcGIS help documentation topic Topology in ArcGIS). An alternative would have been to use a higher rank (lower rank number) for counties along with a smaller cluster tolerance.

The data was validated twice. In the first validation, 159,559 intersection points between hydroregion and county polygons were created (table 1). In addition, 50,500 vertices from the counties class were clustered and therefore shifted some distance. The average shift was 0.721 m and the maximum shift was 1.946 m; 1,609 points shifted more than 1 m.

In the second validation, performed without any intervening edits, no additional changes occurred.

Table 1
Integration Statistics for a Topology Containing U.S. Detailed Counties and Hydrogeographic Regions

Vertex Operation	Number of Vertices Affected
Inserted into counties	159,559
Shifted in counties	50,500
Shifted in counties with distance larger than the cluster tolerance	1,609
Largest shift distance	1.946 meters
Average shift distance	0.721 meter

Figure 17 shows some locations where point clustering occurred, typically where hydrogeographic region and county polygons intersected. The distance across the figure is approximately 135 km. The circled location is detailed in figure 18.

J-9880

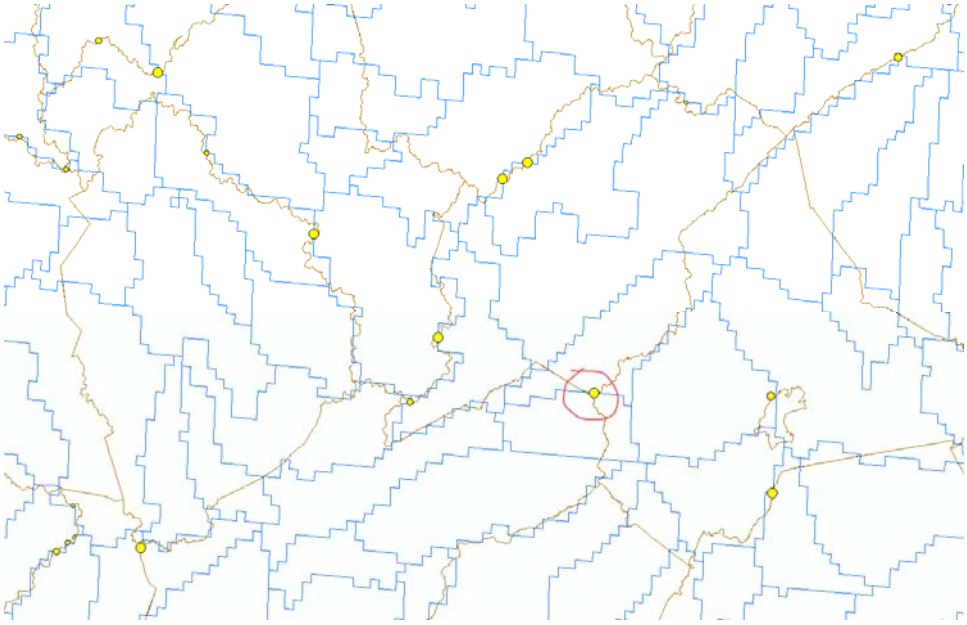

Figure 17: Some locations where clustering occurred. Blue lines are hydroregions, and light brown lines are county boundaries. The distance across the figure is approximately 135 kilometers. The circled location is shown in detail in figure 18.

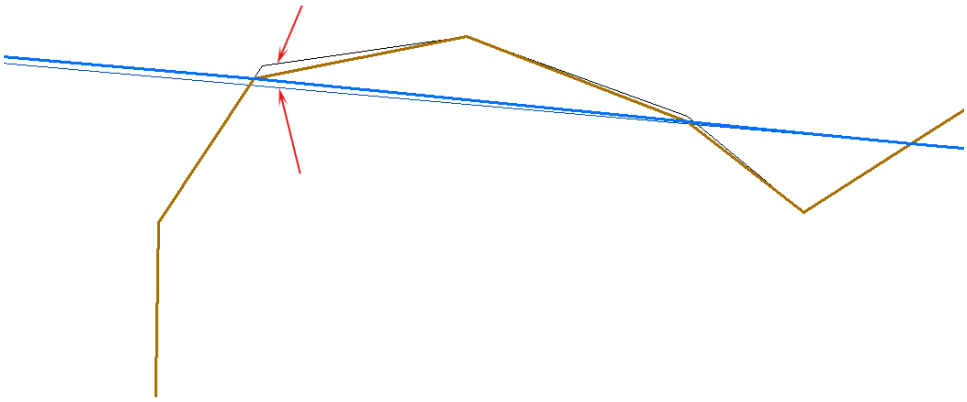

Figure 18: Detail of the circled area in figure 17 showing a location where a clustering operation has occurred. The operation is highlighted by the pair of arrows. The thick lines are integrated county (brown) and hydroregion (blue) polygons. The thin lines are the input versions of the same data. The distance across the figure is approximately 120 m.

This example illustrates the following points:

- Repeated integration of previously integrated data that has had no intervening edits does not change the data.[1]
- Repeated integration of data with intervening edits may change existing data in the vicinity of the edit.

Summary

This white paper introduced the integration process used by ArcGIS software when computing spatial relationships between geometric objects and when constructing new geometric objects. Integration is a multipass process with interleaved cracking and clustering parts in a pass. At least one pass is always performed. Integration is principally controlled by the cluster tolerance property of a spatial reference. During cracking, a point within $\sqrt{2}$ * tolerance of a line will cause a vertex to be inserted in the line. During clustering, each point is allowed to move $\sqrt{2}$ * tolerance to form a cluster. The cluster tolerance does specify an upper boundary on the distance that a point can move. Rather, it guarantees a minimum separation between points, and between points and lines. It is possible for a point to move more than $\sqrt{2}$ * tolerance if it participates in multiple cluster operations. However, since its weight increases in each cluster, larger movements becomes less and less likely.

ESRI recommends that you choose a cluster tolerance that is at least 10 times smaller than the minimum separation that you want to preserve between geometric objects (point-point, line-line, point-line). The coordinate grid resolution should be at least 10 times smaller than the chosen cluster tolerance.
Starting with ArcGIS 9.2, the cluster tolerance is a public property of every spatial reference. Also, as with previous releases, the cluster tolerance is a property of the geodatabase topology, the map topology, and feature construction operations (such as Auto Complete) and is also an environment setting for geoprocessing operations. All these public occurrences of this parameter are interpreted the same way, although not all of these contexts may actually delegate the operation to the topology engine.

Glossary

Clustering—Part of the integration process. It combines pairs of points.

Cluster tolerance—A property of a spatial reference that controls integration. Each spatial reference has three such tolerances: one to control integration in the x,y plane and two others to control the assignment of z- and m-values to integrated output points. These latter tolerances were not discussed in this paper.

Coordinate resolution grid—The set of representable coordinate values. Its location and extent are determined by a false origin and a resolution. There is one grid for x,y coordinates, another for z-values, and a third for m-values.

Cracking—Part of the integration process. It introduces new points into a dataset.

False origin—The smallest coordinate (x, y, z, or m) that can be represented in a dataset.

[1] This is true in the vast majority of situations. There may be cases where a group of points and lines within a few multiples of the cluster tolerance may oscillate between several different clustered positions. In no case will points be arbitrarily or systematically shifted on each application of integrate.

Integration—The process of introducing new points into a dataset and combining points so that all coordinates in the integrated dataset are explicitly represented with the same resolution.

Point weight—A property of a point that controls its clustering behavior. The weights of point pairs being clustered are used as the coefficients of the affine combination used to generate the clustered point. The clustered point's weight is the sum of the input points' weights.

Resolution—The distance that separates adjacent x- or y-coordinate values. There are two additional resolutions for controlling the minimum representable distance between z-coordinate values and m (measure)-values.

Spatial reference—An ArcGIS object that encapsulates a coordinate resolution grid; an earth model; and, optionally, a projection definition. Every dataset in ArcGIS needs to be associated with a spatial reference.

References

Güting, Ralf Hartmut, and Markus Schneider, 1993, "Realms: A Foundation for Spatial Data Types in Database Systems," in Symposium on Large Spatial Databases.

Law, Derek, 2007, *Understanding Coordinate Management in the Geodatabase,* ESRI technical paper.

Milenkovic, Victor, 1998, "Verifiable Implementations of Geometric Algorithms Using Finite Precision Arithmetic," *Artificial Intelligence* 37:377–401.

Zhang, G., and J. Tulip, 1990, "An Algorithm for the Avoidance of Sliver Polygons and Clusters of Points in Spatial Overlay," *Proceedings of the 4th International Symposium on Spatial Data Handling* 1, 141–150.

About ESRI

Since 1969, ESRI has been helping organizations map and model our world. ESRI's GIS software tools and methodologies enable these organizations to effectively analyze and manage their geographic information and make better decisions. They are supported by our experienced and knowledgeable staff and extensive network of business partners and international distributors.

A full-service GIS company, ESRI supports the implementation of GIS technology on desktops, servers, online services, and mobile devices. These GIS solutions are flexible, customizable, and easy to use.

Our Focus

ESRI software is used by hundreds of thousands of organizations that apply GIS to solve problems and make our world a better place to live. We pay close attention to our users to ensure they have the best tools possible to accomplish their missions. A comprehensive suite of training options offered worldwide helps our users fully leverage their GIS applications.

ESRI is a socially conscious business, actively supporting organizations involved in education, conservation, sustainable development, and humanitarian affairs.

Contact ESRI

1-800-GIS-XPRT (1-800-447-9778)
Phone: 909-793-2853
Fax: 909-793-5953
info@esri.com
www.esri.com

Offices worldwide
www.esri.com/locations

ESRI
380 New York Street
Redlands, California
92373-8100 USA