Esri International User Conference | San Diego, CA

Technical Workshops |

Understanding Geometric Networks

Craig Gillgrass

Erik Hoel

Agenda

- Overview of the Model
- Editing and Analyzing
- Versioning
- Performance and Other Key Issues
- Network Model Comparison
- What’s coming in 10.1

Caveats

- Presumed knowledge of the Geodatabase
- Concentrate on key issues not emphasized elsewhere
  - Plenty of documentation (printed and online) that covers basic material
  - Finish up by talking about what we're working on for 10.1
- Lots of material, little time
  - solution: talk fast, ignore questions

Geometric Networks

- First introduced with ArcGIS 8.0 (Dec 1999)
- Connectivity relationships between network feature classes
  - Connectivity based upon geometric coincidence of vertices
  - Connectivity represented in a connectivity index
  - Connectivity relationships always maintained
- Feature classes must be in the same feature dataset

Network Feature Classes

- Network features only live in a geometric network
- Three types:
  - Simple junction
  - Simple edge
  - Complex edge
- Orphan junction feature class
  - Used to maintain integrity
    - Edges must always have a junction at their endpoints
  - System controlled – do not add attributes, etc.

Logical Network

- Physical representation of network connectivity – an index
- High performance graph engine
  - Allows fast network traversals
  - Very compact and optimized
  - Connectivity, weights (attributes) stored in BLOBs
- Analysis (e.g., tracing) is performed within logical network
- Also used with the Network Dataset

Simple Network Features

- Simple edges and junctions
- One-to-one mapping between features and elements
- Connecting a simple junctions to a simple edge at midspan results in a physical split of the edge

Simple Network Features

- Simple edges and junctions
- One-to-one mapping between features and elements
- Connecting a simple junctions to a simple edge at midspan results in a physical split of the edge

Simple Network Features

- Simple edges and junctions
- One-to-one mapping between features and elements
- Connecting a simple junctions to a simple edge at midspan results in a physical split of the edge

Simple Network Features

- Simple edges and junctions
- One-to-one mapping between features and elements
- Connecting a simple junctions to a simple edge at midspan results in a physical split of the edge

Complex Edges

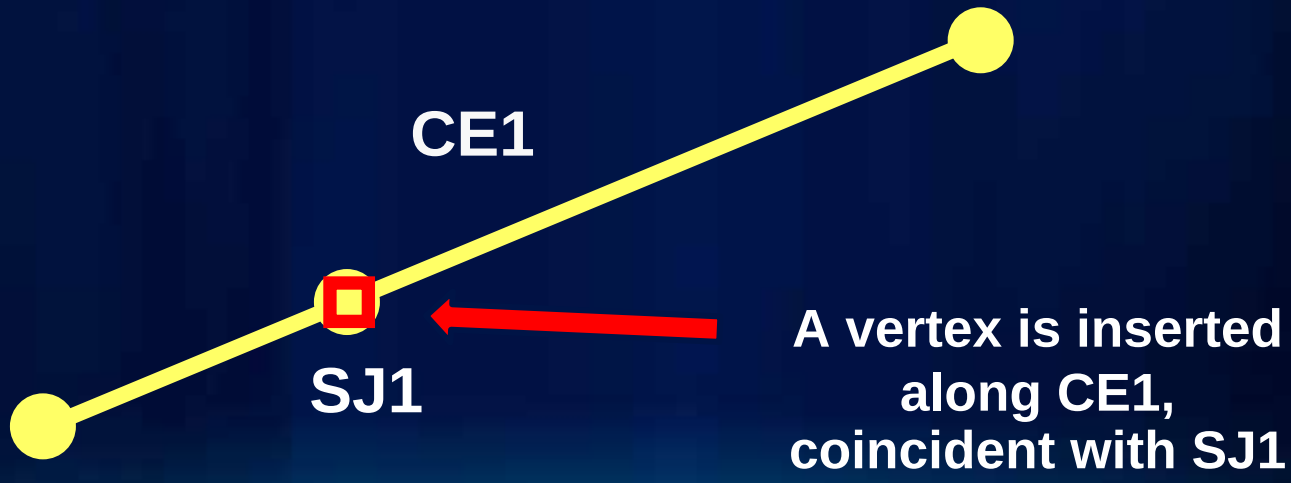
- One-to-many mapping between features and edge elements
- Connecting a junction at midspan does not cause a physical split
  - Causes a vertex to be created along the geometry of the edge
  - Causes a logical subdivision – i.e., a new edge element in the LN
  - Use the Split tool to physically split complex edges

Complex Edges

- One-to-many mapping between features and edge elements
- Connecting a junction at midspan does not cause a physical split
  - Causes a vertex to be created along the geometry of the edge
  - Causes a logical subdivision – i.e., a new edge element in the LN
  - Use the Split tool to physically split complex edges

Complex Edges

- One-to-many mapping between features and edge elements
- Connecting a junction at midspan does not cause a physical split
  - Causes a vertex to be created along the geometry of the edge
  - Causes a logical subdivision – i.e., a new edge element in the LN
  - Use the Split tool to physically split complex edges



Simple Versus Complex Edges

- Simple edges
  - No mid-span connectivity
  - Resources flow from endpoint to endpoint
    - e.g., service laterals, driveways, city streets
- Complex edges
  - Allow mid-span connectivity
  - Resources flow along, but may be siphoned off periodically
    - e.g., water mains, highways

Simple Versus Complex Edges

- Deciding whether a feature class should be simple or complex?
  - Ask yourself whether resources will be siphoned along the edge

Demo

Creating geometric network

Editing and Analyzing

Editing

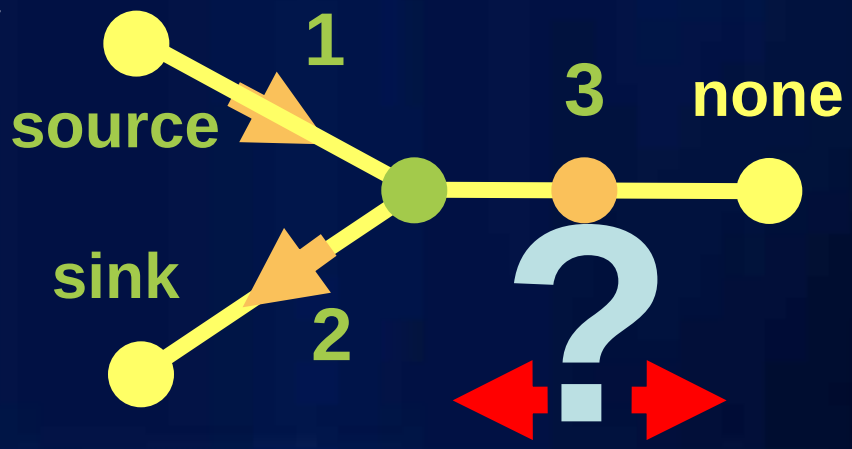
- Same workflow as editing simple features
  - Specific tools/commands on the Geometric Network Editing toolbar
- Connectivity maintained by the GN
  - Based on geometric coincidence of vertices
- Use Snapping and the Map Cache
- Junction subsumption
  - Snapping junctions to Orphan junctions
- Exhaustive network editing examples in the Help
  - See About editing geometric network features for more examples

Flow Direction

- Setting Flow Direction
  - Within an Edit Session
  - Must have at least one Simple Junction with an Ancillary Role field
  - Do this after:
    - Network creation
    - Feature creation or change in connectivity of existing features
    - Source/sink changes
- Does not follow digitized direction by default
- Arrows are drawn at mid-point on the edge features

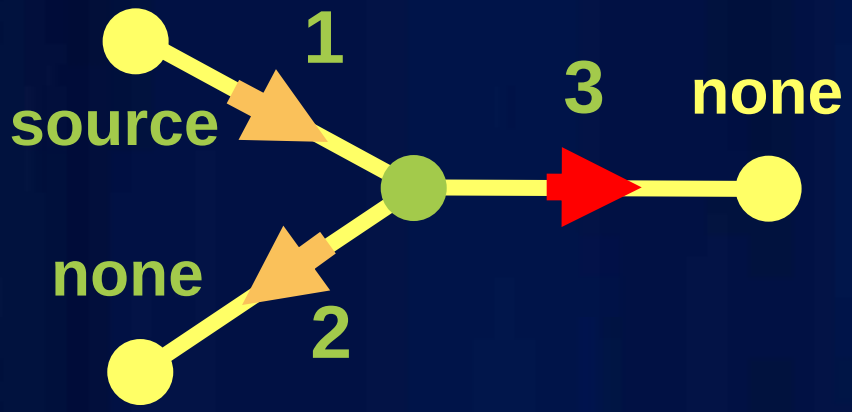
Flow Direction - Indeterminate Flow

- Multiple sources and sinks cause conflicting flow direction
  - Yields indeterminate flow direction
- Consider the following case where edge 3 has indeterminate flow

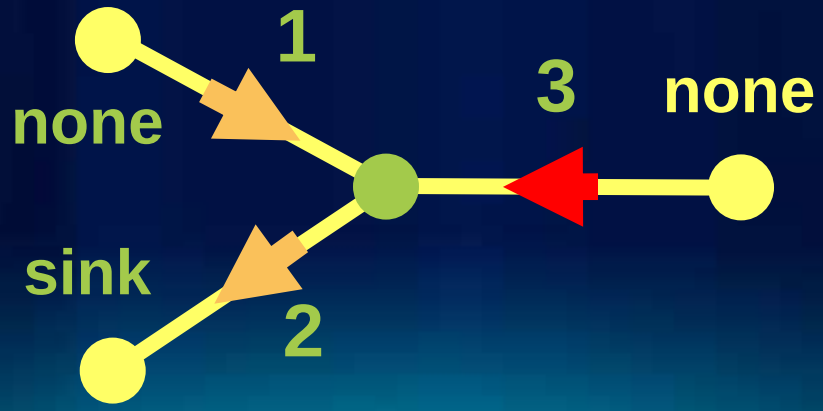


Flow Direction - Indeterminate Flow

- Consider flow direction when only the Source is set



- Consider flow direction when only the Sink is set



Flow Direction

- This results in a conflict
- Flow direction
  - If the flow direction is in agreement between both the source-only and sink-only cases, the flow direction is set to that direction
  - If the flow direction is in conflict between the source-only and sink-only cases, flow is set to indeterminate
- How to set flow direction manually?
  - Samples from ArcObjects Online
- Knowledge Base Article 20685

Network Connectivity and Verification Tools

- **Rebuild** connectivity tool
  - Selectively recreate all connectivity over an area
- **Repair** connectivity command (intended for larger areas)
  - Correct connectivity within a network
  - Does not require entire rebuild of network connectivity, only affects features with inconsistent connectivity
  - Warnings can be raised
  - Optional log file can be created
- Operate on network being edited for Personal/File geodatabases; entire version for ArcSDE geodatabases

Demo

Creating new network features, editing existing features, and performing tracing with flow direction

Versioning

Versioning

- Geodatabase uses an optimistic concurrency approach
- No locks applied when features/objects modified
  - Other editors may edit same features, at the same time
- Introduces the potential for feature conflicts
- A conflict may occur when
  - Two editors are editing the same data in the same version at the same time
  - The same feature is modified in two different versions
- How to manage this?
  - Use Workflow Management to prevent conflicts
  - Manage the conflicts once they occur

Versioning – Rules for Reconcile

1. GN editing rules apply to Reconcile / Conflict replacement
    a. i.e. Orphan junctions cannot subsume each other
2. New features are not created during Reconcile
3. Conflicts result if same features modified in two versions
    a. Update could be to connectivity and/or to geometry/attributes
    b. Feature may be propagated due to connectivity changes
        i. May be in conflict even though not directly edited
    c. Newly created features may be propagated to conflicts

Versioning – Rules for Reconcile

4. Changes only to the connectivity of a feature in two versions will not result in conflicts
    a. Reconcile will filter these features as Conflicts
5. Disconnected state of a features is not considered/maintained

Versioning – Rules for Conflict Management

1. GN editing rules apply to Reconcile / Conflict replacement
    a. i.e. Orphan junctions cannot subsume each other
    b. Restoring features:
        i. Restoring an edge restores the endpoint junctions
        ii. Restoring a junction will not restore connected edges
    c. Removing features:
        i. Removing an edge will not remove the junctions
        ii. Removing an endpoint junction will remove the edge
2. Conflict resolution can create new features
    a. Default junctions from connectivity rules are honored

Versioning - Scenarios

- Two versions, Edit and Target
- The current edit version, is a child of the target version (itself a child of the DEFAULT version)
- Edit Version is reconciled against Target Version
- The default behavior will be for the features in Target Version to take precedence over the features in Edit Version

Versioning - Scenarios

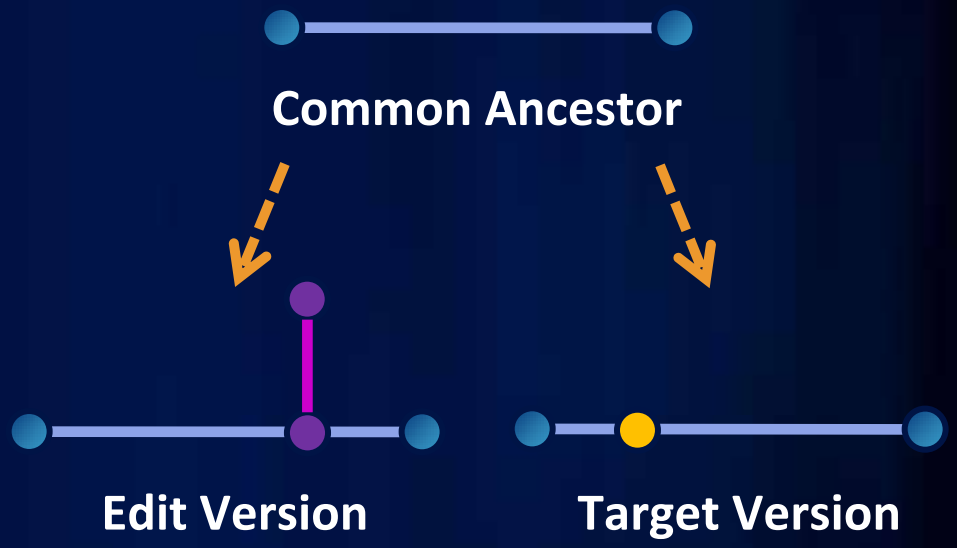
- **Scenarios:**
  1. **Change geometry of a complex edge in 2 version**
  2. **Change connectivity of a junction in 2 versions**
  3. **Delete a feature in the Target, change it in the Edit**
  4. **Update a network attribute in the Target, change the connectivity of the feature in the Edit**

Versioning – Scenario 1

- Target Version
  - A standard junction is added (vertex also added)

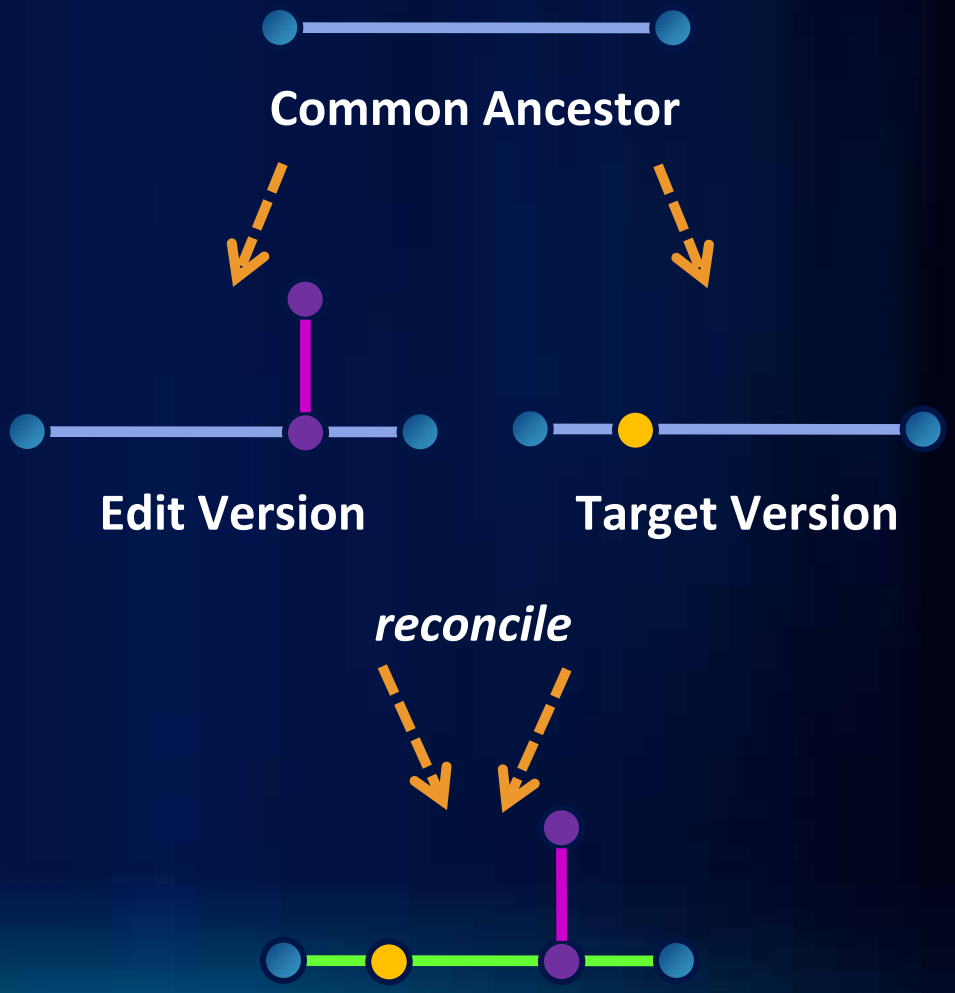
Versioning – Scenario 1

- Target Version
  - A standard junction is added (vertex also added)
- Edit Version
  - A simple edge is added to the same edge



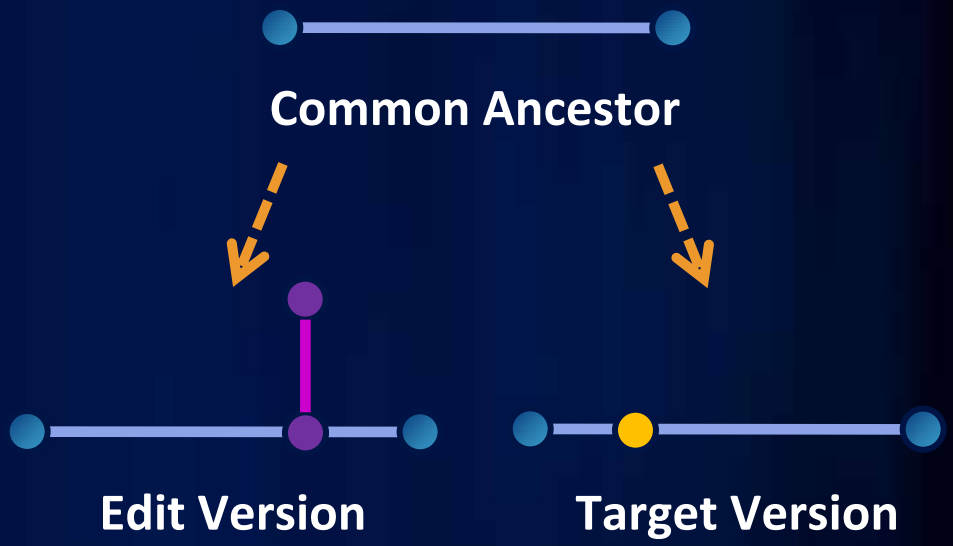
Versioning – Scenario 1

- Target Version
  - A standard junction is added (vertex also added)
- Edit Version
  - A simple edge is added to the same edge
- Reconcile
  - Update-update conflict on the horizontal edge



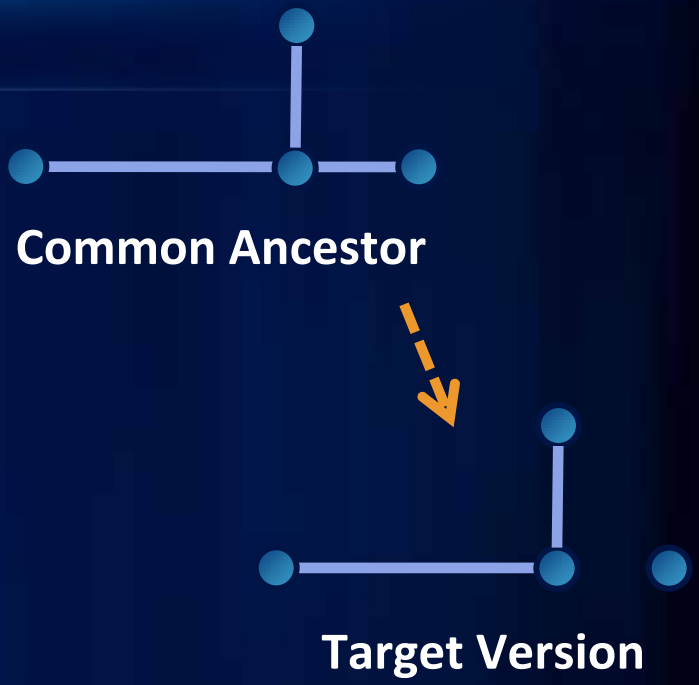
Versioning – Scenario 1

- Target Version
  - A standard junction is added (vertex also added)
- Edit Version
  - A simple edge is added to the same edge
- Reconcile
  - Update-update conflict on the horizontal edge
  - Due to the geometry as well as the connectivity being modified on each



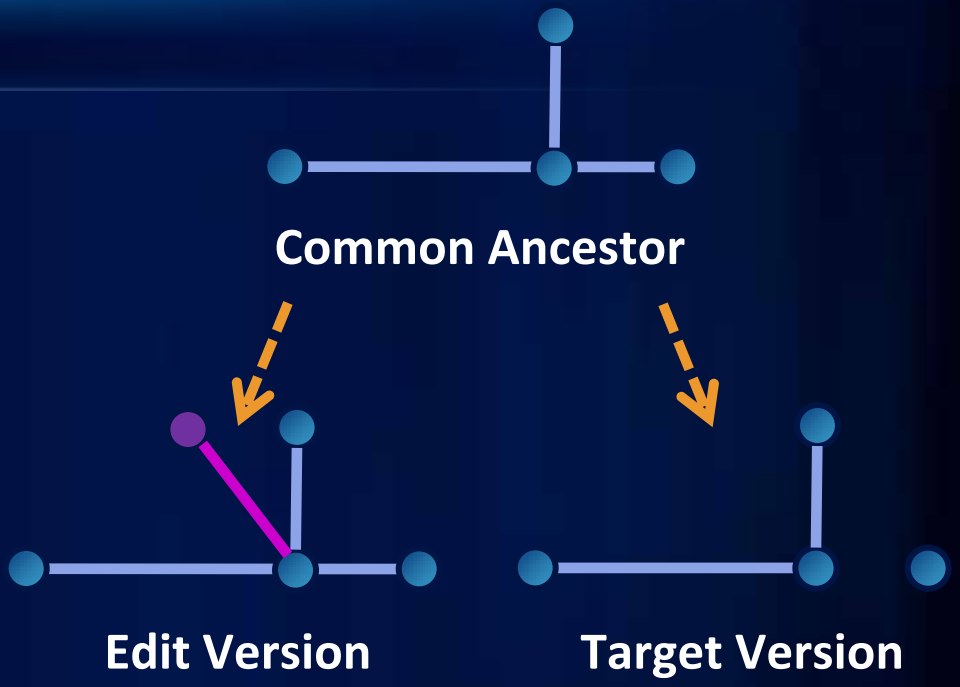
Versioning – Scenario 2

- Target Version
  - A simple edge is deleted



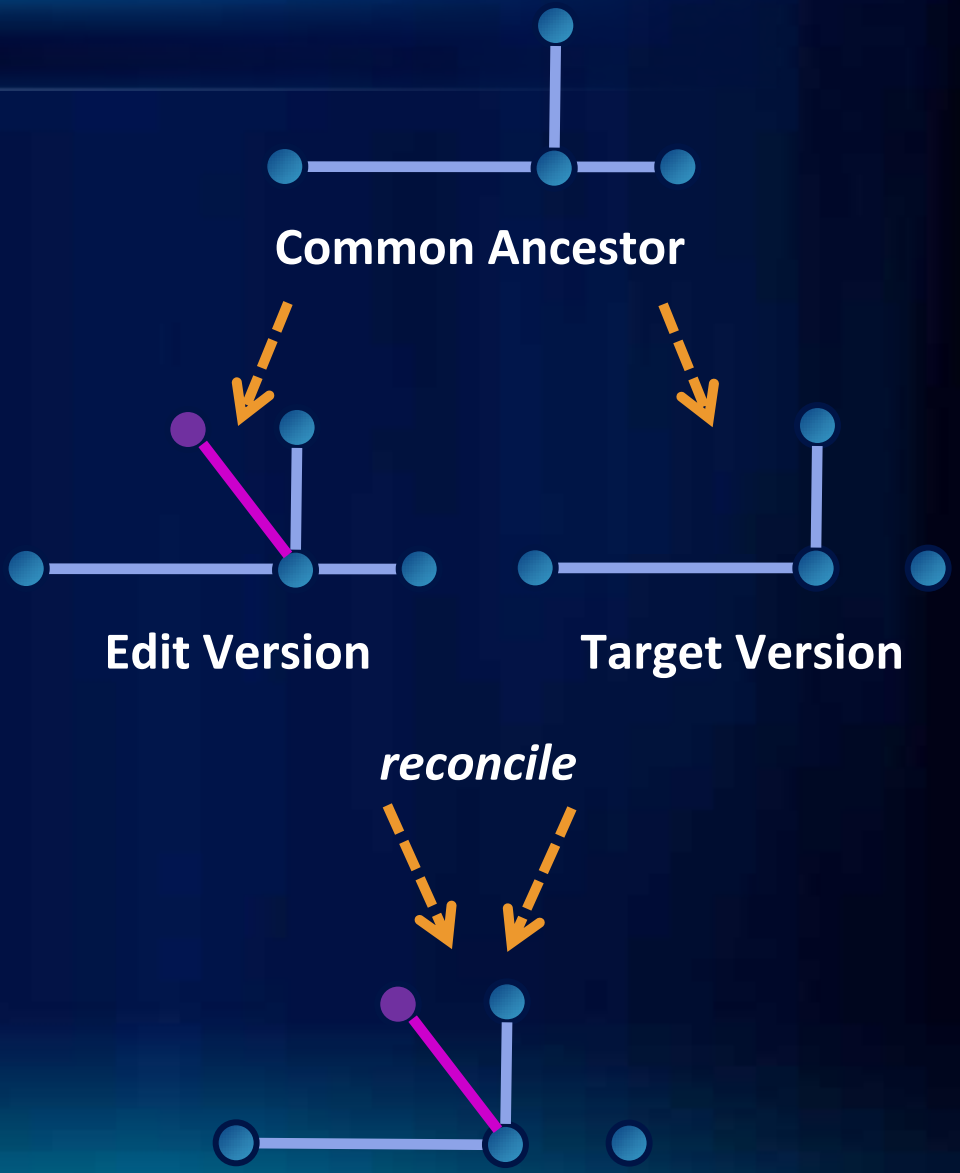
Versioning – Scenario 2

- Target Version
  - A simple edge is deleted
- Edit Version
  - An adjacent edge is added



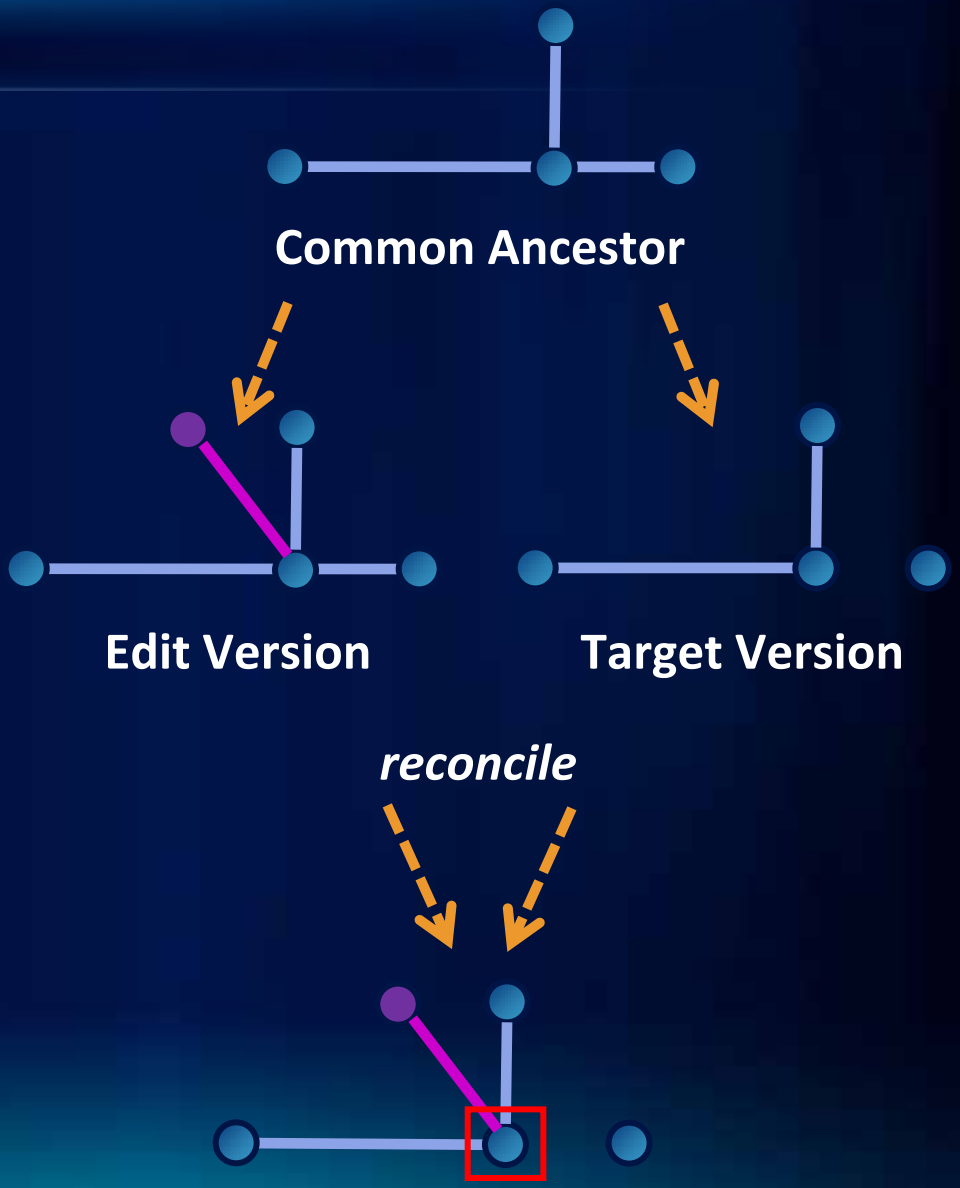
Versioning – Scenario 2

- Target Version
  - A simple edge is deleted
- Edit Version
  - An adjacent edge is added
- Reconcile
  - No conflicts are detected



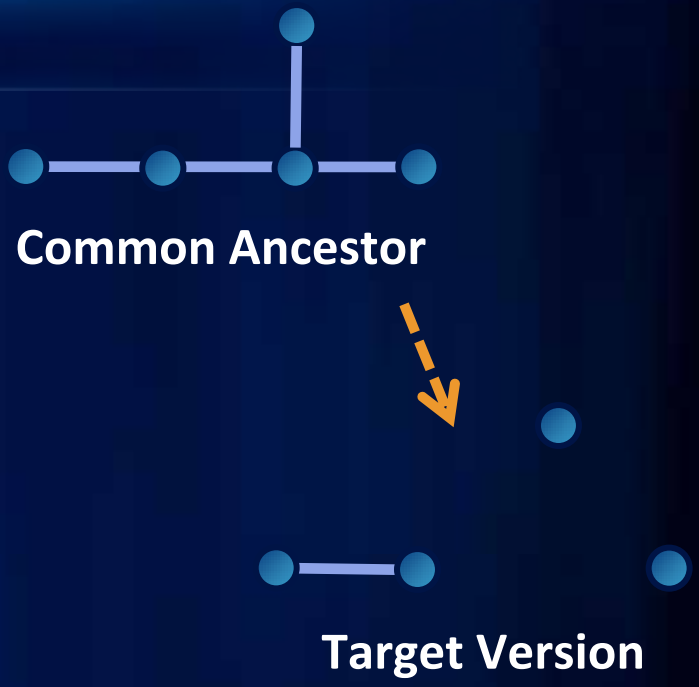
Versioning – Scenario 2

- Target Version
  - A simple edge is deleted
- Edit Version
  - An adjacent edge is added
- Reconcile
  - No conflicts are detected
  - Only the connectivity of the highlighted junction has changed; Reconcile filters any conflict



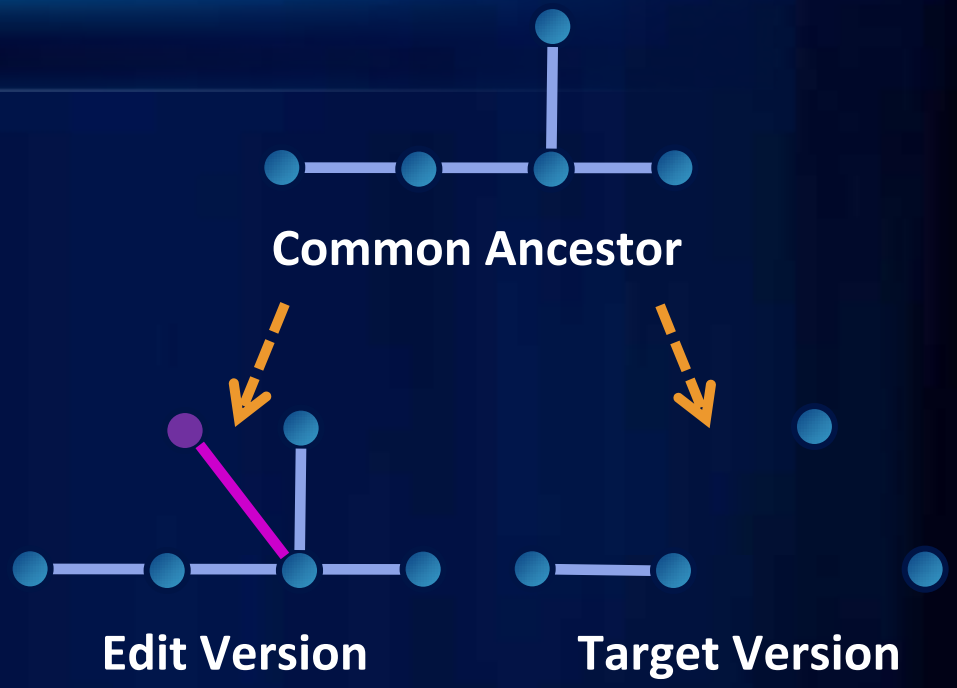
Versioning – Scenario 3

- Target Version
  - An orphan junction is deleted (along with simple edges)



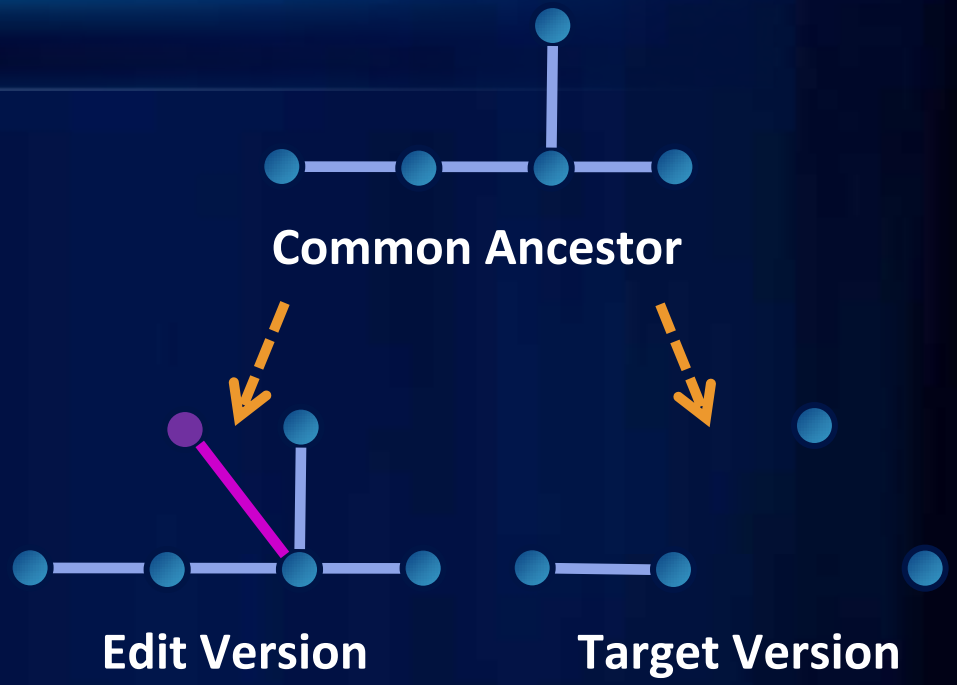
Versioning – Scenario 3

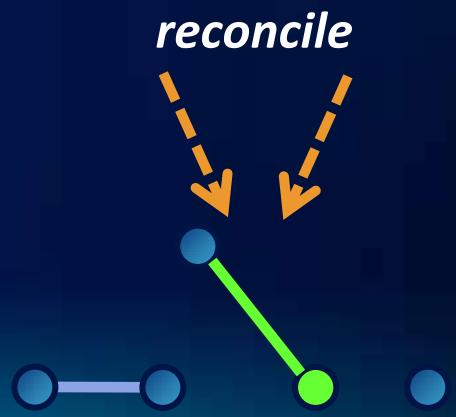
- Target Version
  - An orphan junction is deleted (along with simple edges)
- Edit Version
  - An adjacent edge is added



Versioning – Scenario 3

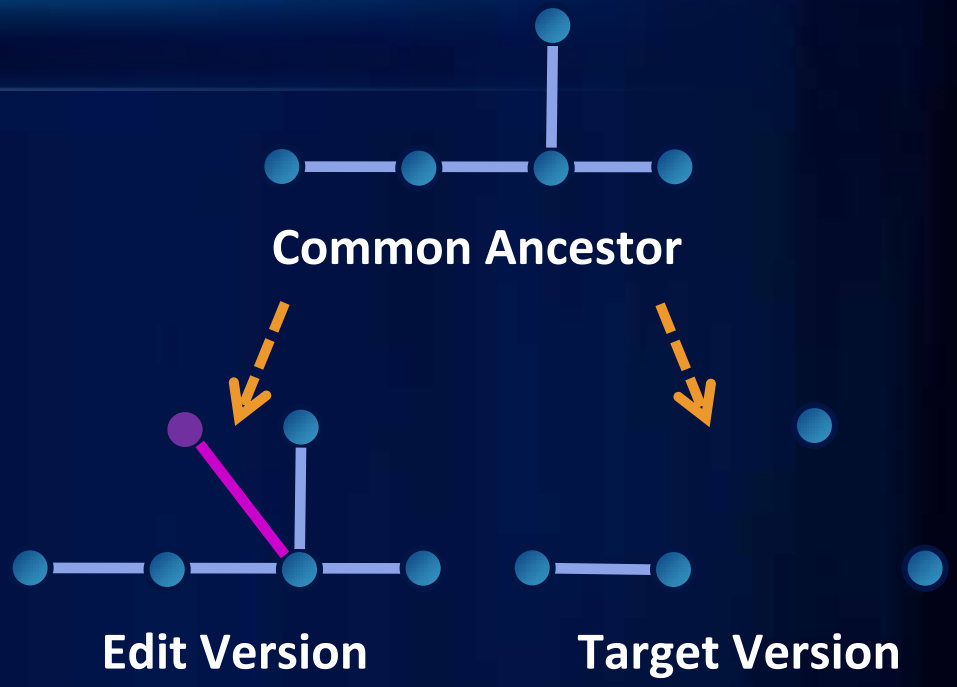
- Target Version
  - An orphan junction is deleted (along with simple edges)
- Edit Version
  - An adjacent edge is added
- Reconcile

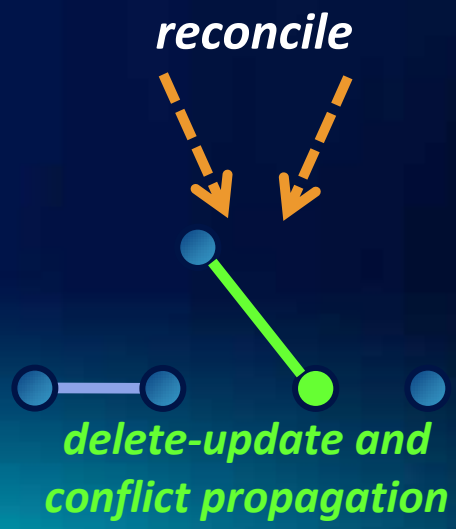




Versioning – Scenario 3

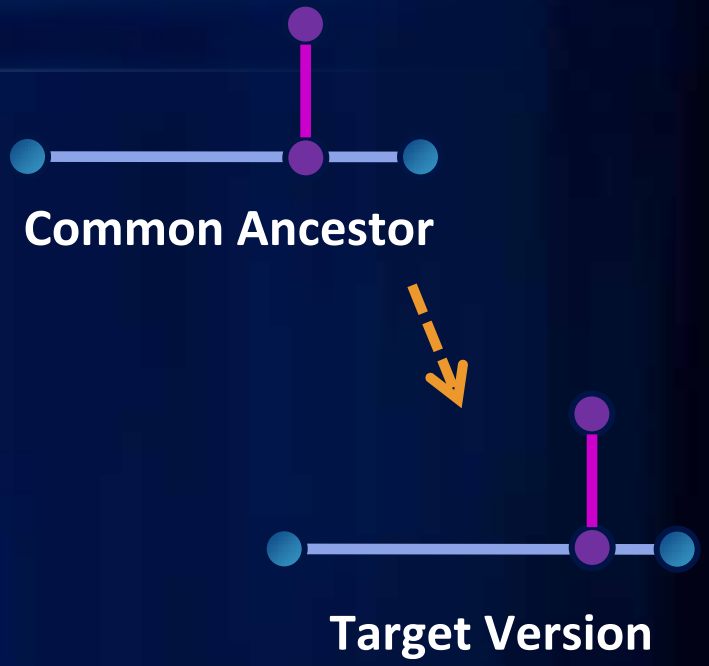
- **Target Version**
  - An orphan junction is deleted (along with simple edges)
- **Edit Version**
  - An adjacent edge is added
- **Reconcile**
  - Delete-update conflict on the junction
  - Conflict propagation on the new edge





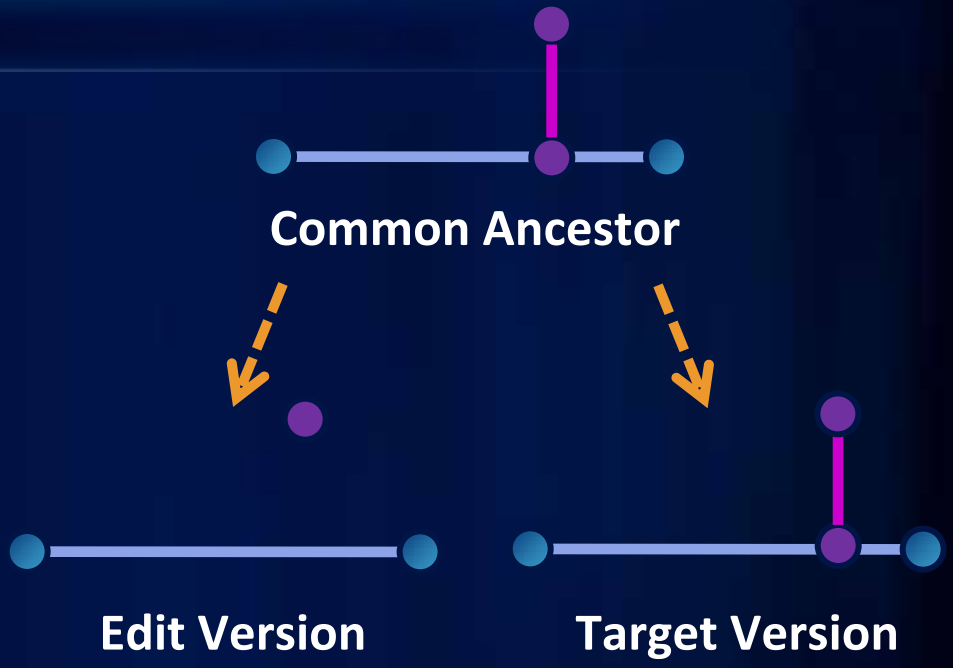
Versioning – Scenario 4

- Target Version
  - ENABLED value is updated on complex edge



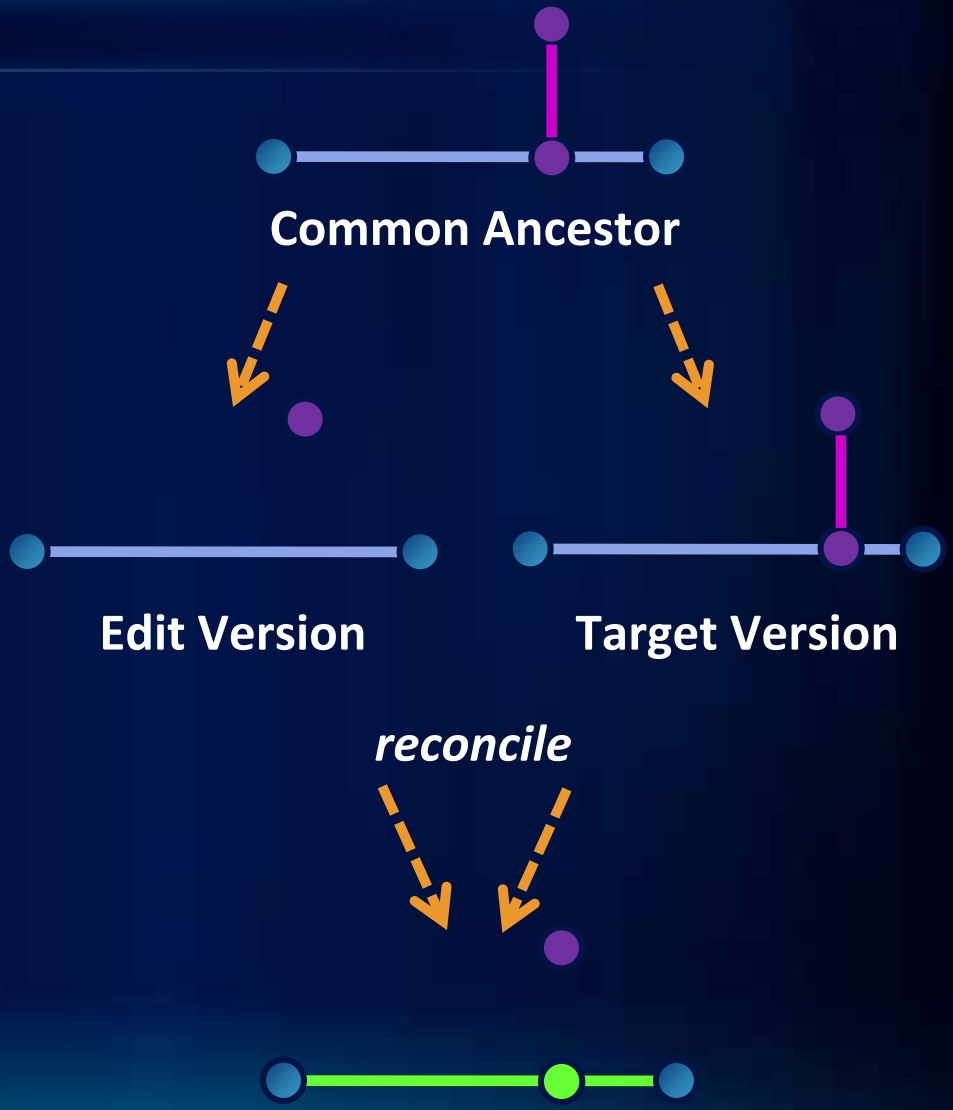
Versioning – Scenario 4

- Target Version
  - ENABLED value is updated on complex edge
- Edit Version
  - orphan junction on same complex edge is deleted



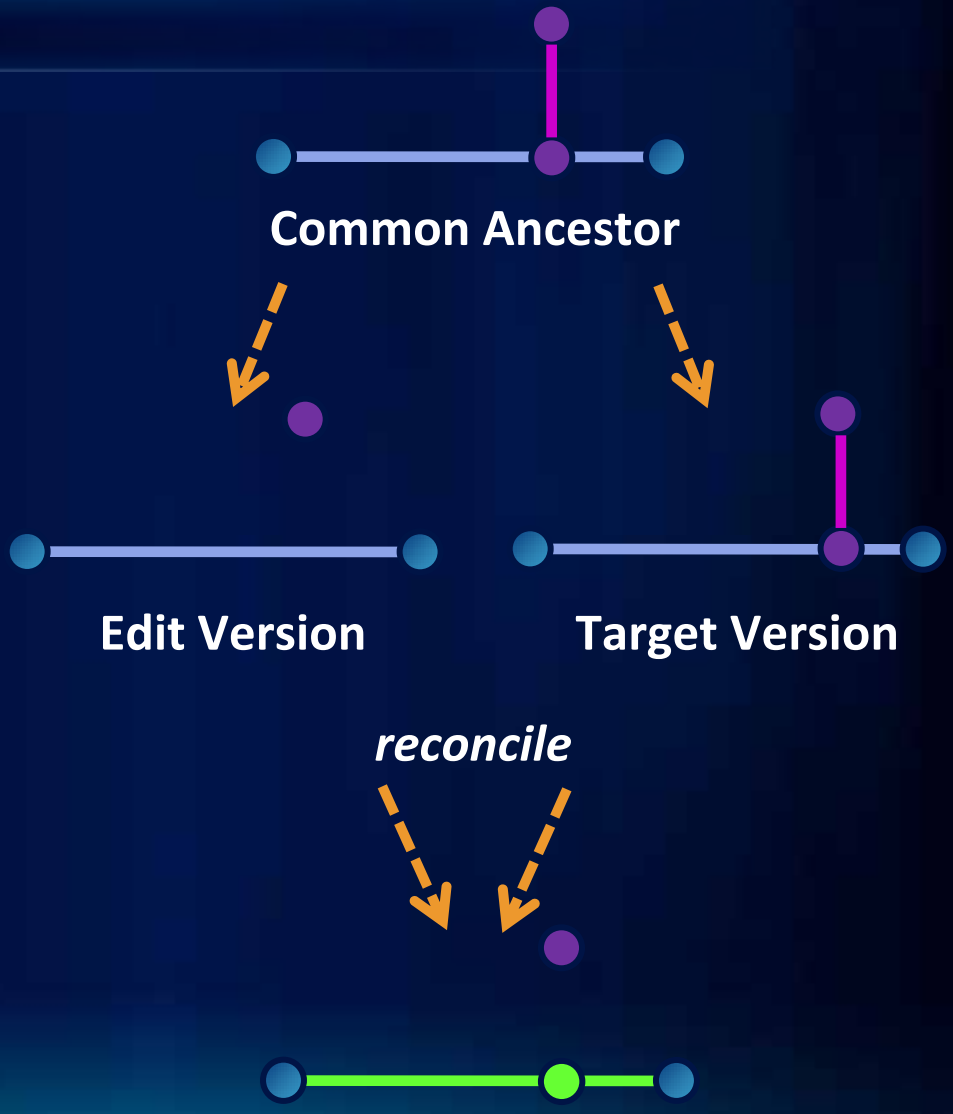
Versioning – Scenario 4

- Target Version
  - ENABLED value is updated on complex edge
- Edit Version
  - orphan junction on same complex edge is deleted
- Reconcile
  - Update-update conflict on the horizontal edge
  - Update-delete conflict on junction



Versioning – Scenario 4

- Target Version
  - ENABLED value is updated on complex edge
- Edit Version
  - orphan junction on same complex edge is deleted
- Reconcile
  - Update-update conflict on the horizontal edge
  - Update-delete conflict on junction



Versioning - Recommendations

- Use Workflow Management to prevent conflicts
  - Avoid editing features in multiple locations in same session
  - Avoid changing large/long features in different versions
  - Plan for bulk updates or edits
- Manage the conflicts once they occur
  - Use different Reconcile options
    - Define conflicts “By Attribute”
    - “In favor of the Edit Version”
  - Resolve conflicts at the top level or class level
    - Resolve junctions first to avoid errors
  - Still can’t resolve the conflict?
    - Consider moving on, and re-doing the edits in another version

Performance and Other Key Issues

API

- Use Logical Network API for navigation and tracing whenever possible
  - IForwardStar
- Navigational APIs available at the Geometric Network feature level
  - Intended for small tactical navigation
- Analysis algorithms should always consume the Logical Network APIs
  - Several orders of magnitude faster
  - INetwork, INetTopology, …

Performance

- Connectivity maintained on the fly
  - Connectivity based upon coincidence
  - When adding a new feature, all other network feature classes are searched
  - Use the map cache
- Minimize the number of network feature classes
  - Utilize subtypes
- Subtypes not for you? Consider lumping of classes
  - Handle unpopulated attributes
- Data model structure is critical
  - Empty classes as expensive as heavily populated
  - Relationship messaging and event handling

Licensing

- ArcEditor or ArcInfo license required to create or edit Geometric Networks
- Geometric Networks are read-only with an ArcView license
  - Can still use Utility Network Analysis with ArcView

Dropping Networks

- Why?
  - Add a new populated class
  - Snapping tolerance too small on previous build
- What happens?
  - Network classes revert to simple classes
  - Network index (logical network) deleted
  - Orphan junction class will be deleted
  - Re-specify connectivity rules and weights
  - Enabled and ancillary role fields retained
  - If snapped during first build, may not need to specify snapping again

Preparing your data for the Geometric Network

- Ideally, your data is clean before you build a network
  - Features that should be connected are geometrically coincident
    - no overshoots or undershoots
- If your data is not clean or you are not sure, you can use one of the following workflows:
  1. Enable snapping during the network creation
  2. Use Topology to find and correct errors
- May still encounter invalid geometries if either method is used

Preparing your data for the Geometric Network

1. **Enable snapping during the network creation**
   - **Good option if:**
     - **You're confident with your overall data quality**
     - **Minor corrections are needed in your data**
2. **Use Topology to find and correct errors**
   - **Good option if:**
     - **You're unsure of your overall data quality**
     - **Know that major edits and corrections are needed in your data to ensure geometric coincidence**
   - **More rules available at 10 that help to discover common data errors for geometric networks**

Coincident Features

- How?
  - Leftover from original data
  - Loaded or created coincident features
- Why is this an issue?
  - Connectivity is based on geometric coincidence
  - Coincident features result in indeterminate connectivity
- What to do?
  - Remove coincident features
  - Offset from each other
  - Use Relationship Classes

Adding Bulk Data

- Two workflows depending upon whether network is versioned
  - Non-versioned
    - drop the network
    - load the data
    - redefine and build network
  - Versioned; several options
    - Consider unversioning the network
    - Use the ObjectLoader
      - Can use in conjunction with the Map Cache
    - Use Disconnected Editing with Replication
    - Geometric Network Incremental Loader at 10.0

Prototyping

- **Largest mistake made with the Geodatabase**
- **Structure is critical – data quantity is not**
- **Prototype as soon as a first pass model is available**
  - **General structure; small details unimportant**
  - **Load a modest amount of data (on versioned SDE)**
  - **Empty classes are OK**
- **Try editing, observe system performance**
- **Repeat this process as necessary**

Network Comparison

Geometric Networks

- Motivated by utility and natural resources industries
- Contain edges and junctions
- Connectivity is continually maintained
- No support for turns, coincident geometries
- All participating features are custom (i.e., not simple features)
- Clients must utilize logical network when implementing analysis algorithms

Network Datasets

- Motivated by transportation industry
- Contain edges, junctions, and turns
- Connectivity re-established at user-controlled times
- Multi-modal connectivity models
- Richer attribute model
- Features may participate in a topology
- Native Shapefile support
- Requires network analyst extension

Comparison

Network Dataset	Geometric Network
transportation	utilities/natural resources
pathfinding and allocation operations	network tracing functionality
turns supported	turns not supported
uses simple features: points and lines	uses custom features: simple/complex edge features and junctions
more robust attribute (weight) model	weights based on feature attributes
user controls when connectivity is built	system automatically maintains connectivity

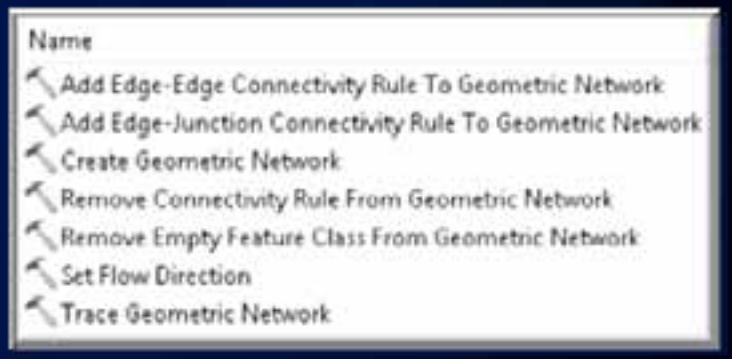
What’s coming in 10.1

What’s new at 10

- Updated Geometric Network wizard
- More scalable and robust geometric network creation algorithm
  - Able to create geometric networks from 10s of millions of features
- Geometric Network Incremental Loader
  - Command to load large amounts of features into a geometric network in a timely manner
- Support network features with the Editor Merge command

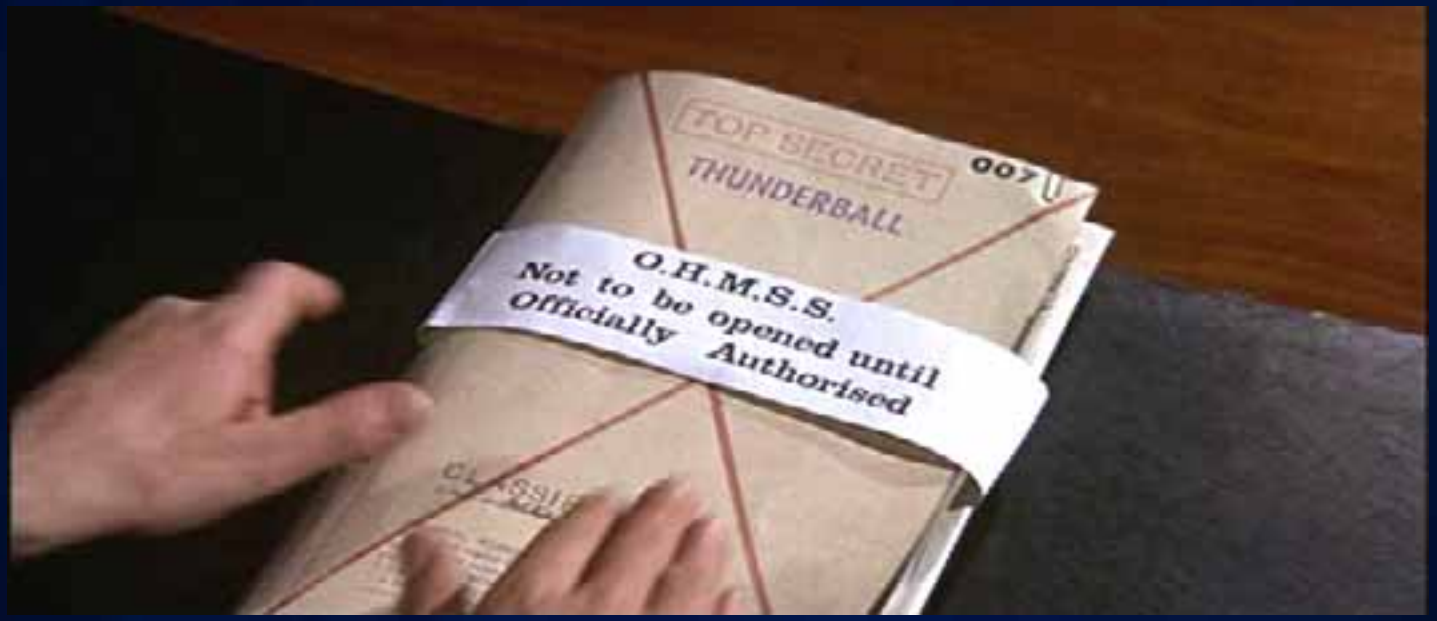
What's new at 10.1

- Geometric Network functionality available through geoprocessing
- Geometric Network creation and management
  - Creation of network and ability to remove empty feature classes
  - Connectivity rule management
- Network Tracing
  - Trace and Set Flow Direction



- Persist settings made to the

Utility Network Analyst toolbar in map documents.


TOP SECRET
007
THUNDERBALL
O.H.M.S.S.
Not to be opened until
Officially Authorised

Questions?