SmartSnapper Specifications

Introduction:

SmartSnapper is a utility for migrating point and line events from one version of NHD (source) to another version of NHD (target). The migration can go forward or backward in "NHD" time. The two versions of NHD need not be the same resolution, however, migrating from a high resolution to a low resolution will not be as accurate.

The basic steps in SmartSnapper are:

1. Events are snapper to flowlines where the Reachcode of the event is the same as the Reachcode of the target flowlines.
2. If there are no matching Reachcodes, events are snapped to flowlines where the GNIS_ID of the event is the same as the GNID_ID of the target flowlines.
3. If there are no matching Reachcodes and no matching GNIS_IDs, then events are snapped to the nearest flowline.

Generally, matching Reachcode snaps are the most accurate, followed by matching GNIS_ID snaps, with the least accurate being the simple spatial snap.

Line events are processed by snapping the endpoints to flowlines. Limitation: If the endpoints do not snap to the same Reachcode in the target NHD, SmartSnapper will not migrate the event. The endpoints many, however, snap to different flowlines as long as the flowlines have the same Reachcode.

SmartSnapper provides QAQC metrics for the migration. The primary metric is what type of migration was performed: R for a matching Reachcode snap, N for matching GNIS_ID snap, and S for a simple spatial snap. For Point events, the distance the point moved is provided. For Line events, the metrics are the distance that each endpoint moved and the change in length of the Line event. By sorting the resulting events by these metrics and examining the larger values, events that need to be manually re-indexed can be identified. This is especially important when the source NHD is high resolution and the target is medium resolution NHD. In this case, high resolution events on streams that are not in medium resolution will likely migrate to the wrong target flowlines.

SmartSnapper does not use the NHDReachCrossReference table. While this table was originally designed to track reachcode changes, the table has become corrupted over time. While there has been an attempt to fix the table, recent examination of the table indicates that it is still incorrect. Should the table become usable, it should be incorporated into SmartSnapper. The use of the NHDReachCrossReference table is the very best way to migrate events.

SmartSnapper takes advantage of the common reachcodes and GNISIDs in the source and target versions of the NHD. This produces a superior migration to a solely spatial process!! Note that, as of 2019, approximately 75% of Reachcodes in medium resolution NHD are still in high resolution NHD having been conflated from medium to high during the initial production of high resolution NHD.

The development environment for SmartSnapper can be Python or another language that can call ArcGIS functions. ArcGIS GP tools are shown in **bold type** throughout the specification. Comment lines are preceded by #.

Disclaimer: This specification has been carefully examined for mistakes and omissions. However, the specification was not thoroughly tested in ArcGIS. Caution should be used when implementing this specification. Results should be checked at each step. For example, after a select, confirm that there is a selected set and after steps that create an output table or feature class, confirm that the table or FC was created.

Inputs:

Source NHDFlowline (SFL)
Target NHDFlowline (TFL)
SFLEvents (point or line) table or feature class: events that are linked to reachcodes and measures in SFL.

Point Events - Required Fields:
- EventID,
- Reachcode,
- Measure

Line Events - Required Fields:
- EventID,
- Reachcode,
- FromMeasure,
- ToMeasure

*NOTE: EventID **must** be unique in the event table. If SFLEvents is a feature class, it should be in geographic coordinates.*

Output:

TFLEvents (point or line) feature class: events that are linked to reachcodes and measures in TFL. The feature class is in geographics. The attributes include the input event attributes plus QC information and metrics for each event

Point Events: QC metric is the distance the point moved during migration.

Line Events: QC metrics include distance each end point moved during migration and change in the event length before and after migration.

Note: Line events should have from and to measures along a single REACHCODE that represent the complete event even if the from and to measures are on different flowlines.

Note: At various points in time, the NHD update process did not apply the rules for Reachcode assignments properly when flowlines were being updated. One such rule was that if a part of a reach needed its Reachcode retired and replaced with a new Reachcode, then all flowlines with the retired Reachcode, were to be replaced with the new Reachcode(s). When this rule was violated, it disrupts any future event migration process and increases the QC that must be done on the final migration result.

Program Specifications:

If SFLEvents is a table with no geometry,

use **Make Route Event Layer** to compute geometry using SFL as the **Route Layer** and SFL.Reachcode as the **Route Identifier** and SFLEvents.ReachCode, and SFLEvents.Measure for point events, and SFLEvents.FromMeasure and SFLEvents.ToMeasure for line events.

Export the layer to SFLEvents feature class.

If SFLEvents are point events,

Copy Features SFLEvents to EventPoints feature class.

If SFLEvents are line events,

#Convert line events to endpoints

Add Field SFLEvent.EventLen Double and populate with length of event in meters using USGS Albers projection. SFLEvent geometry should remain in geographics.

Use **Feature Verticies to Points** with **Point Type = Both Ends** to create EventPoints feature class.

Add Field EventPoints.EndType Short Integer.

Calculate EventPoints.EndType = 0.

Select by Attributes mod("FID",2) = 1 (i.e. FID is odd),

Calculate EndType = 1 (i.e. these are downstream ends). Upstream ends have EndType = 0.

#Identify common Reachcodes between SFL and TFL

Summarize SFL on Reachcode creating SFLRch table.

Summarize TFL on Reachcode creating TFLRch table.

Join SFLRch.Reachcode with TFLRch.Reachcode.

Select records that received a join.

Export selected records to CommonRch table.

Remove Join and delete SFLRch and TFLRch.

Note: HR GNIS ID's have leading 0's, where MR GNIS_IDs may not. If the migration is from HR to MR or MR to HR and this difference exists, it must be reconciled before continuing (see code below) by removing the leading 0's from HR. The easiest way to do this is copy the GNIS ID to a numeric field and then back to the text field. This will remove the leading 0's.

Add Field to [TFL or SFl]Nam called GNIS_ID2 Long Integer

Calculate Field [TFL or SFl]Nam.GNIS_ID2 = [TFL or SFl]Nam.GNIS_ID

Calculate Field [TFL or SFl]Name.GNIS_ID = [TFL or SFl]Nam.GNIS_ID2.

Drop field GNIS_ID2.

Add Field EventPoints.GNIS_ID Text(10)

Join EventPoints.Reachcode with SFL.Reachcode.

Calculate EventPoints.GNIS_ID = SFL.GNIS_ID for joined records.

Note: Based on NHD rules, GNIS_ID is supposed to be the same for all flowlines with the same Reachcode.

#Identify common GNIS_IDs between SFL and TFL

Summarize SFL on GNIS_ID creating SFLNam table.

Summarize TFL on GNIS_ID creating TFLNam table.

Join SFLNam.GNIS_ID with TFLNam.GNIS_ID.

Select records that received a join. Un-select entry with GNIS_ID = ‘’ (or any other invalid GNIS_ID).

Export selected records to CommonNAM table.

Remove Join and delete SFLNam and TFLNam

#Derive EventPoints endpoint coordinates in meters

Project EventPoints to USGS Albers, creating Events_Project

AddField to EventPoints

GNIS_ID Text(10) default blank
DistMoved Double default 0
FromX Double default 0
FromY Double default 0
ToX Double default 0
ToY Double default 0
SnapType Text(1) default blank
NewReachCode Text(14) default blank
NewMeasure Double default 0
NewEventLen Double default 0

Use **Add XY** on Events_Project to add the original x,y coordinates (in meters) to the point attributes

Join EventPoints with Events_Project on Event_ID.

Calculate EventPoints.FromX to Events_Project.X_Coord

Calculate EventPoints.FromY to Events_Project.Y_Coord

Delete Events_Project

#Create an empty output event table

Create empty feature class called TFLPoints using EventPoints schema.

Add Field Snaptype Text(1) default blank.

#Perform matching Reachcode snapping
Loop through CommonRch and For each Reachcode (CurrentReach) in CommonRch

> **Select** TFL flowlines where TFL.Reachcode = CurrentReach, creating TargetRCH.
> If none selected, loop.
>
> **Select** EventPoints where EventPoints.Reachcode = CurrentReach, creating SnapPoints.
> If none selected, loop.
>
> **Snap** SnapPoints.shp to nearest line in TargetRCH.shp Type = EDGE No Distance limit.
>
> **Calculate** SnapPoints.SnapType = “R”
>
> **Append** SnapPoints into TFLPoints.
>
> **Delete** SnapPoints and TargetRch.

End Loop, Move to next CommonRch

#Perform matching GNIS_ID snapping
Loop through CommonNam and For each GNIS_ID (CurrentName) in CommonNam

> **Select** TFL flowlines where TFL.GNIS_ID = CurrentName, creating TargetRCH.
> If none selected, loop.
>
> **Select** EventPoints where EventPoints.GNIS_ID = CurrentName and where EventPoints.EventID is not in TFLPoints.EventID, creating SnapPoints.
> If none selected, loop.
>
> **Snap** SnapPoints.shp to nearest line in TargetRCH.shp Type = EDGE No Distance limit.
>
> **Calculate** SnapPoints.SnapType = “N”
>
> **Append** SnapPoints into TFLPoints.
>
> **Delete** SnapPoints and TargetRch.

End Loop, Move to next CommonNam

#If No Reachcode or GNISID for snap, perform a spatial snap with no target subset.

Clear selection in TFL flowlines.

Select EventPoints where EventPoints.EventID is not in TFLPoints.EventID, creating SnapPoints. If some were selected,

> **Snap** SnapPoints.shp to nearest line in TFL flowlines Type = EDGE No Distance limit.
>
> **Calculate** SnapPoints.SnapType = “S”
>
> **Append** SnapPoints into TFLPoints.
>
> **Delete** SnapPoints

#Derive new coordinates for snapped points.

Project TFLPoints to USGS Albers, creating Events_Project

Use **Add XY** on Events_Project to add the New x,y coordinates (in meters) to the point attributes

Join TFLPoints with Events_Project

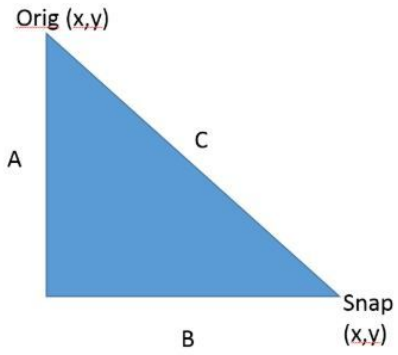
Calculate ToX = X_Coord

Calculate ToY = Y_Coord

Remove Join

Calculate Field: TFLPoints.Distmoved using the From x,y and To x,y and Pythagorean theorem

> Distmoved = ((FromX - ToX)**2 + (FromY - ToY)**2)** 0.5



A**2 + B**2 = C**2
(A**2 + B**2)**0.5 = C
A = Origx – Snapx
B = Origy – Snapy
((Origx – Snapx) **2 + (Origy – Snapy)**2)**0.5 = C

#Derive new Reachcode/Measure for snapped points

Locate Features along Routes using TFL flowlines and TFLPoints to create the new event table (TFLEventstbl with new reachcode (RCH)/measure (Meas). No tolerance is needed because the points have been snapped.

Join TFLPoints with TBLEventstbl on EventID

Calculate TFLPoints.NewReachcode to TFLEventstbl.RCH.

Calculate TFLPoints.NewMeasure to TFLEventstbl.Meas

Remove join.

Delete TFLEventstbl

#Create final output event table

If SFLEvents are point events, then
Copy TFLPoints to TFLEvents

If SFLEvent are Line Events, then ***#reconstitute TFLPoints as lines in TFLEvents and compute metrics:***

Select TFLPoints.PointType = 1 (i.e. the downstream point)

Export selected to dspoint

AddField dspoint.frommeas double

AddField dspoint.fromDist double

AddField dspoint.dsX double

AddField dspoint.dsY double

Calculate dspoint.frommeas = NewMeasure

Calculate dspoint.fromDist = DistMoved

Calculate dspoint.dsX = ToX

Calculate dspoint.dsY = ToY

Reverse selection (I.e. the upstream point)

Export selected to uspoint

AddField uspoint.toMeas double

AddField uspoint.ToDist double

AddField uspoint.usX double

AddField uspoint.usY double

Calculate uspoint.tomeas = NewMeasure

Calculate uspoint.ToDist = DistMoved

Calculate uspoint.usX = ToX

Calculate uspoint.usY = ToY

Join dspoint with uspoint

QAQC: If dspoint.newreachcode <> uspoint.newreachcode, ERROR maybe an error in the specification or execution of this process. Or it may be Reachcode changes that don't permit migration with SmartSnapper. Log event and delete event from joined table.

Export fields EventID, EventLen, NewReachcode, FromMeas, ToMeas, FromDist, ToDist, usX, usY, dsX, dsY to TFLLines

AddField TFLLines.NewEventLen double

AddField TFLLines.LenghtChg double

Make Route Event Layer using TFL flowlines and TFLLines to render the new line events with NewReachcode, FromMeas, ToMeas. No tolerance is needed because end points have been snapped.

Calculate geometry Length (TFLLines.NewEventLen) in USGS Albers.

Calculate TFLLines.LengthChg = ((Eventlen-NewEventlen)/Eventlen)*100

Copy TFLLines to TFLEvents