Lab03: Optimal Path

- build an ETL to obtain relevant data (DEM and Land Classifcation from MNGEO)
- Create Cost Surface Model
  - Distance Accumulation (from start/end point)
  - Rescale (above Distance accumulation output)
  - Weighted Sum (to produce a coast surface)
  - Optimal Region Connections (this is where I add the stream barrier 'No crossing water!')

Downloading data from MNGEO (DEM, Land Classification, Streams)

In [17]:
```
import requests
import json
import pprint
import zipfile
import arcpy
```

In [21]:
```
packages = "https://gisdata.mn.gov/api/3/action/package_list"
groups = "https://gisdata.mn.gov/api/3/action/group_list"
tags = "https://gisdata.mn.gov/api/3/action/tag_list"

#I believe this locates all the tags within the MN geo commons (I am looking for 'biota' and 'boundary')
response = requests.get(tags, auth = ('user','pass'), verify = False)
# locate bu 'imagery-basemaps'
response_1 = requests.get(groups, auth = ('user','pass'), verify = False)

#converting the response from unreadable bytes to json
tags_json = json.loads(response.content)

tags_json_1 = json.loads(response_1.content)
```

```
C:\Users\runac\AppData\Local\ESRI\conda\envs\arcgispro-py3-Lab2_clone\Lib\site-packages\urllib3\connectionpool.
py:1020: InsecureRequestWarning: Unverified HTTPS request is being made to host 'gisdata.mn.gov'. Adding certif
icate verification is strongly advised. See: https://urllib3.readthedocs.io/en/1.26.x/advanced-usage.html#ssl-w
arnings
  InsecureRequestWarning,
C:\Users\runac\AppData\Local\ESRI\conda\envs\arcgispro-py3-Lab2_clone\Lib\site-packages\urllib3\connectionpool.
py:1020: InsecureRequestWarning: Unverified HTTPS request is being made to host 'gisdata.mn.gov'. Adding certif
icate verification is strongly advised. See: https://urllib3.readthedocs.io/en/1.26.x/advanced-usage.html#ssl-w
arnings
  InsecureRequestWarning,
```

Extracting Land Cover file

In [ ]:
```
#I want "Impervious Surface Area by Landsat & lidar:2013-update -version 2" data labeled with the 'land cover'
groups = 'imagery-basemaps'
tag1 = "impervious surface area"
tag2 = 'land cover'
tag3 = 'landsat'
tag4 = 'lidar'
tag5 = 'object based image classification'
base_url = "http://gisdata.mn.gov/api/3/action/package_search?q="

package_information_url = base_url + groups + tag1 + tag2

#requesting all information associated with 'land cover' tag from MN Geo Commons
package_information = requests.get(package_information_url, auth = ('user', 'pass'), verify = False)

#converting all the information to a json dictionary
package_dict = json.loads(package_information.content)
#pprint.pprint(package_dict)

package_dict_result = package_dict["result"]['results']

r = requests.get(package_dict_result[0]['resources'][2]['url'])
open('tif_base_landcover_minnesota.zip', 'wb').write(r.content)
print('extracting the content...')

#unzipping the file and saving it to my desired local
with zipfile.ZipFile('tif_base_landcover_minnesota.zip', 'r') as zip_ref:
    zip_ref.extractall("E:/Fall 2021/ArcGIS1/Labs/Lab02/Lab02_CostSurface/MN_geo_data_pipeline")
```

Extracting DEM file

In [ ]:
```
groups = 'elevation'
tag1 = 'model'
tag2 = 'slope'
tag3 = 'elevation'
base_url = "http://gisdata.mn.gov/api/3/action/package_search?q="

package_information_url = base_url + groups

#requesting all information associated with 'elevation' tag from MN Geo Commons
package_information = requests.get(package_information_url, auth = ('user', 'pass'), verify = False)

#converting all the information to a json dictionary
package_dict = json.loads(package_information.content)
#pprint.pprint(package_dict)

package_dict_result = package_dict["result"]['results']

#pprint.pprint(package_dict_result[3]['resources'][1])

#ID to comfirm data
#My geodatabase ID: 1c2f17f6-f7df-43de-9d96-03b49b867f77

r = requests.get(package_dict_result[3]['resources'][1]['url'])
open('fgdb_elev_30m_digital_elevation_model.zip', 'wb').write(r.content)
print('extracting the content...')

#unzipping the file and saving it to my desired local
with zipfile.ZipFile('fgdb_elev_30m_digital_elevation_model.zip', 'r') as zip_ref:
    zip_ref.extractall("E:/Fall 2021/ArcGIS1/Labs/Lab02/Lab02_CostSurface/MN_geo_data_pipeline")
```

extracting stream data

In [ ]:
```
groups = 'inland-waters'
tag1 = 'dnr fisheries'
tag2 = 'rivers'
tag3 = 'stream survey'
tag4 = 'streams'
base_url = "http://gisdata.mn.gov/api/3/action/package_search?q="

package_information_url = base_url + groups + tag1

#requesting all information associated with 'elevation' tag from MN Geo Commons
package_information = requests.get(package_information_url, auth = ('user', 'pass'), verify = False)

#converting all the information to a json dictionary
package_dict = json.loads(package_information.content)
#pprint.pprint(package_dict)

package_dict_result = package_dict["result"]['results']

#pprint.pprint(package_dict_result[7]['resources'][1])
#Shapefile ID: 0ad76fbd-452a-47b6-aa15-4a6cb49928ea

r = requests.get(package_dict_result[7]['resources'][1]['url'])
open('shp_water_measured_kittle_routes.zip', 'wb').write(r.content)
print('extracting the content...')

#unzipping the file and saving it to my desired local
with zipfile.ZipFile('shp_water_measured_kittle_routes.zip', 'r') as zip_ref:
    zip_ref.extractall("E:/Fall 2021/ArcGIS1/Labs/Lab02/Lab02_CostSurface/MN_geo_data_pipeline")

print('Done!')
```

extracting road data

In [44]:
```
groups = 'transportation'
tag1 = 'route direction'
tag2 = 'routes'
tag3 = 'route number'
tag4 = 'roads'
base_url = "http://gisdata.mn.gov/api/3/action/package_search?q="

package_information_url = base_url + tag3

#requesting all information associated with 'elevation' tag from MN Geo Commons
package_information = requests.get(package_information_url, auth = ('user', 'pass'), verify = False)

#converting all the information to a json dictionary
package_dict = json.loads(package_information.content)
#pprint.pprint(package_dict)

package_dict_result = package_dict["result"]['results']

#pprint.pprint(package_dict_result[9]['resources'][3])
# #Shapefile ID:91758b03-2f87-4a41-b0dc-555dcc8be279

r = requests.get(package_dict_result[7]['resources'][1]['url'])
open('shp_trans_roads_mndot_tis.zip', 'wb').write(r.content)
print('extracting the content...')

#unzipping the file and saving it to my desired local
with zipfile.ZipFile('shp_trans_roads_mndot_tis.zip', 'r') as zip_ref:
    zip_ref.extractall("D:/Fall 2021/ArcGIS1/Labs/Lab02/Lab02_CostSurface/MN_geo_data_pipeline")

print('Done!')
```

```
C:\Users\runac\AppData\Local\ESRI\conda\envs\arcgispro-py3-Lab2_clone\Lib\site-packages\urllib3\connectionpool.
py:1020: InsecureRequestWarning: Unverified HTTPS request is being made to host 'gisdata.mn.gov'. Adding certif
icate verification is strongly advised. See: https://urllib3.readthedocs.io/en/1.26.x/advanced-usage.html#ssl-w
arnings
  InsecureRequestWarning,
extracting the content...
Done!
```

Data Preprocessing:

In the GUI:

- cropped data to a smaller spatial extant (previously all statewide)
- created a shapefile for two points (the start and end locations for Dory's walk)

I made to new feature classes in the GUI interface of ArcPro, one creating the start and end points for Dory's walk, the other a rectangular extent used to cropped the statewide data down to a smaller size.

In the following code cells i will provide the code for buffering the road data (to 12ft wide) and then erasing where roads overlap with stream in order to build an optimal path that avoids stream corssing (this way Dory can find the bridges, since the stream will not register as stream at this location because we have altered the stream data)

In [15]:
```
###code for buffering road data

arcpy.analysis.Buffer("STREETS_LOAD_ClipLayer",
                      r"D:\Fall 2021\ArcGIS1\Labs\Lab02\Lab02_CostSurface\Lab02_CostSurface.gdb\STREETS_Buffere
                      "100 Feet",
                      "FULL",
                      "ROUND",
                      "NONE",
                      None,
                      "PLANAR")
```

Out[15]:

Output

D:\Fall 2021\ArcGIS1\Labs\Lab02\Lab02_CostSurface\Lab02_CostSurface.gdb\STREETS_Buffered_100ft

Messages

Start Time: Tuesday, November 30, 2021 2:50:36 PM
Succeeded at Tuesday, November 30, 2021 2:50:41 PM (Elapsed Time: 5.55 seconds)

In [16]:
```
### code for erasing the intersection of road and stream data
arcpy.analysis.Erase("streams_with_measured_kittle",
                     "STREETS_Buffered_100ft",
                     r"D:\Fall 2021\ArcGIS1\Labs\Lab02\Lab02_CostSurface\Lab02_CostSurface.gdb\streams_roads_B
                     None)
```

Out[16]:

Output

D:\Fall 2021\ArcGIS1\Labs\Lab02\Lab02_CostSurface\Lab02_CostSurface.gdb\streams_roads_BIGGEST_Buff

Messages

Start Time: Tuesday, November 30, 2021 2:50:48 PM
Reading Features...
Cracking Features...
Assembling Features...
Succeeded at Tuesday, November 30, 2021 2:51:18 PM (Elapsed Time: 29.50 seconds)

Cost Surface Model

- first: calculate the 'Distance Accumulation' from both the Start-End locations
- second: rescale the outputs from the 'Distance Accumulation' Tool
  - low values-low cost, high values -high cost (scale 1-10)
- thrid: Weighted Sum (results in out cost raster)
  - will be borrowing the previously reclassified Land Use raster form lab02 (this designates Dory's prefernces for avoiding crop fields)

In [ ]:
```
#first output: has a strict avoidnace of slopes beyond -10 and 10 degrees
out_distance_accumulation_raster = arcpy.sa.DistanceAccumulation("Start_End_points",
                                                                  None,
                                                                  "DEM",
                                                                  None,
                                                                  "DEM",
                                                                  "BINARY 1 -10 10",
                                                                  None,
                                                                  "BINARY 1 45",
                                                                  r"D:\Fall 2021\ArcGIS1\Labs\Lab02\Lab02_CostSu
                                                                  None,
                                                                  None,
                                                                  None,
                                                                  None,
                                                                  None,
                                                                  '',
                                                                  "PLANAR"
                                                                 ); out_distance_accumulation_raster.save(r"D:\F
#second output, avoids slopes greater than 15 degrees and less than -15 degrees
out_distance_accumulation_raster = arcpy.sa.DistanceAccumulation("Start_End_points",
                                                                  None,
                                                                  "DEM",
                                                                  None,
                                                                  "DEM",
                                                                  "BINARY 1 -15 15",
                                                                  None,
                                                                  "BINARY 1 45",
                                                                  r"D:\Fall 2021\ArcGIS1\Labs\Lab02\Lab02_CostSu
                                                                  None,
                                                                  None,
                                                                  None,
                                                                  None,
                                                                  None,
                                                                  '',
                                                                  "PLANAR"
                                                                 ); out_distance_accumulation_raster.save(r"D:\F
#thrid output, avoids slpoes greater than 20 degrees and less than -20 degrees
out_distance_accumulation_raster = arcpy.sa.DistanceAccumulation("Start_End_points",
                                                                  None,
                                                                  "DEM",
                                                                  None,
                                                                  "DEM",
                                                                  "BINARY 1 -20 20",
                                                                  None,
                                                                  "BINARY 1 45",
                                                                  r"D:\Fall 2021\ArcGIS1\Labs\Lab02\Lab02_CostSu
                                                                  None,
                                                                  None,
                                                                  None,
                                                                  None,
                                                                  None,
                                                                  '',
                                                                  "PLANAR"
                                                                 ); out_distance_accumulation_raster.save(r"D:\F
```

In [ ]:
```
#rescaling the distance Accumulation rasters (from 1-10)
#made the choice to rescale using the "Logistic Decay" function, since it favors smaller values (aka closer dis

#-10 t- 10 degree slope
out_raster = arcpy.sa.RescaleByFunction("Dis_from_SE1",
                                        "LOGISTICDECAY # # 99 # # # #",
                                        1,
                                        10); out_raster.save(r"D:\Fall 2021\ArcGIS1\Labs\Lab02\Lab02_CostSurfac
#-15 tp 15 degree slope
out_raster = arcpy.sa.RescaleByFunction("Dis_from_SE2",
                                        "LOGISTICDECAY # # 99 # # # #",
                                        1,
                                        10); out_raster.save(r"D:\Fall 2021\ArcGIS1\Labs\Lab02\Lab02_CostSurfac

#-20 to 20 degree slope
out_raster = arcpy.sa.RescaleByFunction("Dis_from_SE3",
                                        "LOGISTICDECAY # # 99 # # # #",
                                        1,
                                        10); out_raster.save(r"D:\Fall 2021\ArcGIS1\Labs\Lab02\Lab02_CostSurfac
```

In [ ]:
```
##creating the Cost Surface through Weighted Sums (all calulated using the -10-10 slope tolerance raster)

#first version: equal weights (1-1)
out_raster = arcpy.ia.WeightedSum("Reclass_land1 Value 1;Rescale_Dis_1 VALUE 1"); out_raster.save(r"D:\Fall 202

#second version: preferance for land cover (5-1)
out_raster = arcpy.ia.WeightedSum("Reclass_land1 Value 5;Rescale_Dis_1 VALUE 1"); out_raster.save(r"D:\Fall 202

#third version: prefernce for distance/slope (1-5)
out_raster = arcpy.ia.WeightedSum("Reclass_land1 Value 1;Rescale_Dis_1 VALUE 5"); out_raster.save(r"D:\Fall 202
```

In [7]:
```
##creating the Cost Surface through Weighted Sums (all calulated using the -15-15 slope tolerance raster)

#first version: equal weights (1-1)
out_raster2_11 = arcpy.ia.WeightedSum("Reclass_land1 Value 1;Rescale_Dis_2 VALUE 1"); out_raster.save(r"D:\Fal

#second version: preferance for land cover (5-1)
out_raster2_51 = arcpy.ia.WeightedSum("Reclass_land1 Value 5;Rescale_Dis_2 VALUE 1"); out_raster.save(r"D:\Fal

#third version: prefernce for distance/slope (1-5)
out_raster2_15 = arcpy.ia.WeightedSum("Reclass_land1 Value 1;Rescale_Dis_2 VALUE 5"); out_raster.save(r"D:\Fal
```

In [8]:
```
##creating the Cost Surface through Weighted Sums (all calulated using the -20-20 slope tolerance raster)

#first version: equal weights (1-1)
out_raster3_11 = arcpy.ia.WeightedSum("Reclass_land1 Value 1;Rescale_Dis_3 VALUE 1"); out_raster.save(r"D:\Fal

#second version: preferance for land cover (5-1)
out_raster3_51 = arcpy.ia.WeightedSum("Reclass_land1 Value 5;Rescale_Dis_3 VALUE 1"); out_raster.save(r"D:\Fal

#third version: prefernce for distance/slope (1-5)
out_raster3_15 = arcpy.ia.WeightedSum("Reclass_land1 Value 1;Rescale_Dis_3 VALUE 5"); out_raster.save(r"D:\Fal
```

Finally, Optimal Paths

This is where I can stipulate the avodance of stream crossings!

Note, I discovered as i wast calulating the optimal path, that as soon as i inputed the barrier (streams in this case since Dory doesn't want to cross them without a brigde), the results said it was impossible to connect my start and end points. AS soon as I removed the barrier, it sucessfully ran and found an 'OPtimal Path'. I surmise to possible reasons for this outcome:

- the buffered width of roads (12ft) was to small given our raster dat had measeure between 15 and 30 ft (after runnning Many simulations, I learned I had to bump up the buffer to a ridiculously large value in order to get the computer to notice the gaps between stream. It finally worked at a buffer of 100 feet)
- need to alter the function for discouraging slope so that it is more forgiving or inclusive of steep terrain (at the moment Bonary function returns NA values for cells with too steep of terrain).

In [ ]:
```
#using the 'Optimal Region connection tool'

#barrier for equal weight cost surface (FAILED)
arcpy.sa.OptimalRegionConnections("Start_End_points", r"D:\Fall 2021\ArcGIS1\Labs\Lab02\Lab02_CostSurface\Lab02
# the code when no barrier is specified runs just fine:
arcpy.sa.OptimalRegionConnections("Start_End_points", r"D:\Fall 2021\ArcGIS1\Labs\Lab02\Lab02_CostSurface\Lab02

#using the Cost surface derived from: slopes -20 to 20 and a Large buffered road (100feet) (SUCCESS!!)
#equal weights
arcpy.sa.OptimalRegionConnections("Start_End_points", r"D:\Fall 2021\ArcGIS1\Labs\Lab02\Lab02_CostSurface\Lab02

#weights favors land use (5-1)
arcpy.sa.OptimalRegionConnections("Start_End_points", r"D:\Fall 2021\ArcGIS1\Labs\Lab02\Lab02_CostSurface\Lab02

#weights favor slope/distance (1-5)
arcpy.sa.OptimalRegionConnections("Start_End_points", r"D:\Fall 2021\ArcGIS1\Labs\Lab02\Lab02_CostSurface\Lab02
```