Introduction to SNAP.py

CS224W Recitation Session

Alex Haigh

9/28/18

Slides based on 2017's review session: http://snap.stanford.edu/class/cs224w-2017/recitation/SNAP.PY_Recitation.pdf

A few notes before we start

1. These slides (and example code) will be available on Piazza and on cs224w.stanford.edu

2. **SNAP can do even more than what's described here!**
   a. The snap library has many functions not discussed in this deck
   b. Search the SNAP.py documentation before you reinvent the wheel

3. **SNAP is not omnipotent!**
   a. For many problems - both in and out of the homework - snap alone is not expressive enough to do exactly what you need to solve your problem
   b. Sometimes extra data structures are necessary to solve the problem (e.g. maintaining separate sets of disease IDs and node IDs in HW1 Q4)

Before we begin

- These slides are available at

  http://snap.stanford.edu/class/cs224w-2017/recitation/SNAP.PY_Recitation.pdf

- All examples used in these slides are available at

  http://snap.stanford.edu/class/cs224w-2017/recitation/examples.zip

What is SNAP?

- **S**tanford **N**etwork **A**nalysis **P**latform (SNAP) is a general purpose, high-performance system for analysis and manipulation of large networks.
    - http://snap.stanford.edu
        - Scales to massive networks with hundreds of millions of nodes and billions of edges
- **SNAP** Software: SNAP.PY for Python, SNAP C++
- **SNAP** Datasets: Over 70 datasets, available at http://snap.stanford.edu/data.

SNAP.PY Resources

- **Prebuilt packages** available for Mac OS X, Windows, Linux
    - http://snap.stanford.edu/snappy/index.html
- **Documentation** (including Tutorial & Reference Manual)
    - http://snap.stanford.edu/snappy/doc/index.html
- **User mailing list**
    - http://groups.google.com/group/snap-discuss

SNAP.PY Resources

- **Developer resources** (including Benchmarking tools)
  - https://github.com/snap-stanford/snap-python

SNAP Network Datasets

- Collection of over 70 network datasets
  - http://snap.stanford.edu/data

Installing SNAP.PY

- **Requires** Python 2.7
    - http://www.python.org/
- **Download** the SNAP.PY for your platform
    - http://snap.stanford.edu/snappy
- **Follow** instructions
    - http://snap.stanford.edu/snappy/index.html
    - `(sudo) python setup.py install`

Installing SNAP.PY

- **Problems?** Refer to our troubleshooting guide
    - https://docs.google.com/document/d/1iuFKw0mS5GsrVj7T7opXDY-qE8fbtd6HTJBZhDYeE3Q/edit
- Post or look at existing posts on Piazza.

Using SNAP.PY

- The most important step
  - `$ python`
    `>>> import snap`

SNAP.PY Tutorial

- **Available** on the website
    - http://snap.stanford.edu/snappy/doc/tutorial/index-tut.html
- **Today, we will cover**
    - Basic SNAP.PY data types
    - Vectors, hash tables and pairs
    - Basic graph types
    - Graph creation
    - Adding and traversing nodes/edges
    - Useful functions for HW0

Basic Types & Vector Types

- **Basic Types** in SNAP are `TInt`, `TFlt`, and `TStr`
    - Correspond to Python types **`int`**, **`float`** and **`str`**
- **Vector Types**
    - *Naming convention*: `T<value_type>V`
    - *Examples*: `TIntV, TFltV, TStrV`
    - *Operations*:
        - **`Add(<value>)`**: Append a value at the end
        - **`Len()`**: Vector size
        - **`[<index>]`**: Get or set a value of an existing element
        - **`for i in V`**: Iteration over the vector

Vector Example

```
import snap

v = snap.TIntV()                    # Create an empty vector

v.Add(1)                            # Add elements
v.Add(2)
v.Add(3)
v.Add(4)
v.Add(5)

print v.Len()                       # Print vector size

print v[3]                          # Get & Set elements
v[3] = 2*v[2]
print v[3]

for item in v:                      # Iterate over elements
  print item

for i in range(0, v.Len()):
  print i, v[i]
```

Hash Table Types

- A set of **(key, value)** pairs
    - Keys must be of the same type
    - Values must be of the same type
    - However, value type can be different from the key type
- *Naming convention*: `T<key_type><value_type>H`
- *Examples*: `TIntStrH, TIntFltH, TStrIntH`
- *Operations*:
    - **`[<key>]`**: Add a new value or get or set an existing value
    - **`Len()`**: Hash table size
    - **`for i in H`**: Iteration over keys

Hash Table Example

```
import snap

h = snap.TIntStrH()                    # Create an empty table

h[5] = 'apple'                         # Add elements
h[3] = 'tomato'
h[9] = 'orange'
h[6] = 'banana'
h[1] = 'apricot'

print h.Len()                          # Print table size

print 'h[3] = ', h[3]                  # Get element value

h[3] = 'peach'                         # Set element value
print 'h[3] =', h[3]

for key in h:                          # Iterate over keys
  print key, h[key]
```

Pair Types

- A pair **(value1, value2)**
  - Type of value1 can be different from type of value2
  - *Naming convention*: `T<type1><type2>Pr`
    - *Examples*: `TIntStrPr, TIntFltPr, TStrIntPr`
  - *Operations*:
    - **`GetVal1`**: Get value1
    - **`GetVal2`**: Get value2

Pair Example

```
import snap

p = snap.TIntStrPr(1, 'one')          # Create a new pair

print p.GetVal1()                     # Get values
print p.GetVal2()
```

Basic Graph Classes

- Graphs
  - **TUNGraph**: undirected graph
  - **TNGraph**: directed graph
  - **TNEANet**: multigraph with attributes on nodes and edges

Graph (Creation) Example

```
import snap

''' Graph (Creation) '''

G1 = snap.TNGraph.New()          # Create empty directed graph

G1.AddNode(1)                    # Important: Add nodes before adding edges
G1.AddNode(5)
G1.AddNode(12)

G1.AddEdge(1, 5)                 # Add edges
G1.AddEdge(5, 1)
G1.AddEdge(5, 12)

G2 = snap.TUNGraph.New()         # Create empty undirected graph

N1 = snap.TNEANet.New()          # Create empty multigraph with attributes
```

Graph (Traversal) Example

```
''' Graph (Traversal) '''

for NI in G1.Nodes():                # Node traversal
  print 'node id %d, out-degree %d, in-degree %d' % (NI.GetId(), NI.GetOutDeg(),
NI.GetInDeg())

for EI in G1.Edges():                # Edge traversal
  print '(%d, %d)' % (EI.GetSrcNId(), EI.GetDstNId())

for NI in G1.Nodes():                # Edge traversal by node
  for DstNId in NI.GetOutEdges():
    print '(%d, %d)' % (NI.GetId(), DstNId)
```

Graph (Saving & Loading) Example

```
''' Graph (Saving & Loading) '''

# Save graph to text file
snap.SaveEdgeList(G1, 'test.txt', 'List of Edges')

# Load graph from text file
G3 = snap.LoadEdgeList(snap.PNGraph, 'test.txt', 0, 1)
```

```
# Save graph to binary
FOut = snap.TFOut('test.graph')
G1.Save(FOut)
FOut.Flush()

# Load graph from binary
FIn = snap.TFIn('test.graph')
G4 = snap.TNGraph.Load(FIn)
```

Loading Text Files

Example file: wiki-Vote.txt

- Download from http://snap.stanford.edu/data

```
# Directed graph: wiki-Vote.txt
# Nodes: 7115 Edges: 103689
# FromNodeId    ToNodeId
0         1
0         2
0         3
0         4
0         5
2         6
…
```

```
LoadEdgeList(PGraph, InFNm, SrcColId, DstColId, Separator)
G = snap.LoadEdgeList(snap.PNGraph, "wiki-Vote.txt", 0, 1)
```

Useful Functions: `G.Nodes()` & `G.Edges()`

- Get a **generator** for all nodes in graph G
  - http://snap.stanford.edu/snappy/doc/reference/graphs.html?highlight=nodes()
- Get a **generator** for all edges in graph G
  - http://snap.stanford.edu/snappy/doc/reference/graphs.html?highlight=edges()
- Example
  - `for node in G.Nodes()`
    `for edge in G.Edges()`

Useful Functions: `G.GetNodes()` & `G.GetEdges()`

- Get the **total number** of nodes in G
  - http://snap.stanford.edu/snappy/doc/reference/graphs.html?highlight=getnodes
- Get the **total number** of edges in G
  - http://snap.stanford.edu/snappy/doc/reference/graphs.html?highlight=getedges
- Example
  - ```
    G = snap.LoadEdgeList(snap.PNGraph, "wiki-Vote.txt", 0, 1)
    print "G: Nodes %d, Edges %d" % (G.GetNodes(), G.GetEdges())
    ```
    ```

Useful Functions: `CntSelfEdges(G)` & `CntUniqDirEdges(G)`

- Get the **total number** of self edges in G
  - http://snap.stanford.edu/snappy/doc/reference/CntSelfEdges.html
  - Example
    - ```
      Count1 = snap.CntSelfEdges(G)
      print "Count of self edges is G is %d" % Count1
      ```

- Get the **total number** of unique directed edges in G
  - http://snap.stanford.edu/snappy/doc/reference/CntUniqDirEdges.html
  - Example
    - ```
      Count2 = snap.CntUniqDirEdges(G)
      print "Count of unique directed edges is %d" % Count2
      ```

Useful Functions: `CntUniqUndirEdges(G)`

- Get the **total number** of unique undirected edges in G
  - http://snap.stanford.edu/snappy/doc/reference/CntUniqUndirEdges.html
- Example
  - ```
    Count3 = snap.CntUniqUndirEdges(G)
    print "Count of unique undirected edges is %d" % Count3
    ```
    ```

Useful Functions: `GetInDeg(G)` & `GetOutDeg()`

- Get the **in-degree** of a node *n*
  - http://snap.stanford.edu/snappy/doc/reference/graphs.html?highlight=getindeg
  - Example
    - `n.GetInDeg()`
- Get the **out-degree** of a node *n*
  - http://snap.stanford.edu/snappy/doc/reference/graphs.html?highlight=getoutdeg
  - Example
    - `n.GetOutDeg()`

Useful Functions: `GetWccs(G, C)` & `GetMxWcc(G)`

- Get all **weakly connected components** in G
  - http://snap.stanford.edu/snappy/doc/reference/GetWccs.html
  - Example
    - ```
      Components = snap.TCnComV()
      snap.GetWccs(G, Components)
      for CnCom in Components:
        print "Size of component: %d" % CnCom.Len()
      ```
- Get the **largest weakly connected component** in G
  - http://snap.stanford.edu/snappy/doc/reference/GetMxWcc.html
  - Example
    - ```
      MxWcc = snap.GetMxWcc(G)
      for EI in MxWcc.Edges():
        print "edge: (%d, %d)" % (EI.GetSrcNId(), EI.GetDstNId())
      ```

Useful Functions: `GetPageRank(G,P)` & `GetHits(G,H,A)`

- Get the **Pagerank score** of every node in G
  - http://snap.stanford.edu/snappy/doc/reference/GetPageRank.html
  - Example
    - ```
      PRankH = snap.TIntFltH()
      snap.GetPageRank(G, PRankH)
      sorted_PRankH = sorted(PRankH, key = lambda key: PRankH[key], reverse
      = True)
      ```

- Get the **Hubs & Authorities score** of every node in G
  - http://snap.stanford.edu/snappy/doc/reference/GetHits.html?highlight=gethits
  - Example
    - ```
      NIdHubH = snap.TIntFltH()
      NIdAuthH = snap.TIntFltH()
      snap.GetHits(G, NIdHubH, NIdAuthH)
      sortedByAuth = sorted(NIdAuthH, key = lambda key: NIdAuthH[key], reverse = True)
      sortedByHub = sorted(NIdHubH, key = lambda key: NIdHubH[key], reverse = True)
      ```

Thank you!