**Khai thác cấu trúc mạng xã hội ứng dụng vào lĩnh vực phân tích tâm lý người dùng trên Wiki**

Giảng viên hướng dẫn: Ths. Nguyễn Thị Anh Thư

Sinh viên thực hiện:

Đậu Thị Kim Oanh – 14520650

Phan Nhựt Cường – 14520114

**CHƯƠNG I: TÌM HIỂU VỀ SNAP**

1. **Giới thiệu chung về snap.py**

SNAP – Stanford Network Analysis Project được viết bằng C++ với mục đích là tăng hiệu suất cho việc phân tích và thao tác với các mạng lớn có quy mô lên tới hàng trăm triệu nút và hàng tỷ cạnh. Ngoài ra, snap còn hỗ trợ tính toán kết cấu, tạo đồ thị và hỗ trợ các thuộc tính trên các nút và cạnh. Snap hỗ trợ 2 ngôn ngữ là C++ và Python.

Snap cho C++ có ưu điểm là tốc độ thực thi chương trình nhanh, tuy nhiên, nhược điểm là ngôn ngữ phức tạp hơn so với Python và thực thi theo hướng biên dịch. Ngoài ra, một nhược điểm khác nữa của snap cho C++ là không có giao diện mà hiển thị dưới dạng console.

Snap cho Python là gói snap.py có các ưu điểm như: thực thi chương trình nhanh, ngôn ngữ đơn giản, thực thi theo hướng thông dịch và có giao diện.

Trong đề tài, nhóm sử dụng snap.py là snap giành cho python. Xem thêm về snap tại <http://snap.stanford.edu>.

1. **Hướng dẫn sử dụng snap cho Python**

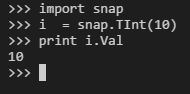
Python phân biệt chữ hoa và chữ thường rất chặt chẽ nên snap.py cũng phân biệt chữ hoa và chữ thường.

***Important background***

Để sử dụng được thư viện snap.py, chúng ta phải import thư viện này bằng câu lệnh sau: Import snap.

***Các kiểu dữ liệu cơ bản trong snap.py và snap***

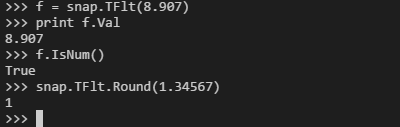
Các kiểu dữ liệu cơ bản của snap đó là: TInt (kiểu int), TFlt (kiểu float), TStr (kiểu string). Đặc điểm rất tiện lợi của snap là tự động chuyển đổi giữa C++ và Python, do đó chúng ta có thể sử dụng cách khai báo kiểu dữ liệu của snap hay python, C++ đều được.



Các function hỗ trợ bởi lớp Tint: Load(SIn) – tải int từ một luồng nhị phân Sln, Save(SOut) – lưu int vào một luồng nhị phân SOut và các function khác như: Abs(val) – phương thức tĩnh trả về giá trị tuyệt đối, GetHexStr(val) – phương thức tĩnh trả về một chuỗi hexidecimal, …

Xem thêm các function hỗ trợ lớp TFloat, TStr tại <http://snap.stanford.edu/snappy/doc/reference/basic.html>

Đây là một ví dụ khác về sử dụng TFlt trong python:



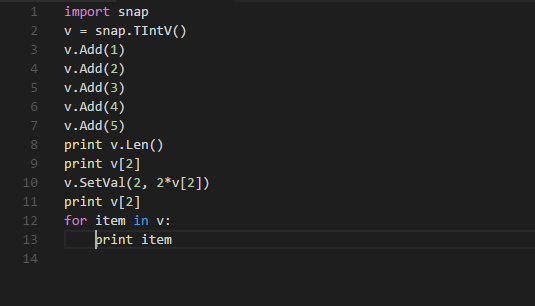
***Các loại vector***

Đặt tên: <tên kiểu dữ liệu>V: TIntV, TFltV, TStrV.

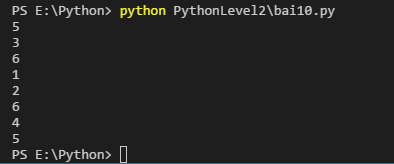
Là tập hợp các dãy giá trị có cùng kiểu dữ liệu. Các giá trị mới có thể được thêm vào cuối. Chúng ta có thể thao tác với các giá trị trong vector.

Thao tác với vector chúng ta sử dụng: Add – thêm giá trị vào vector, Len – trả về kích thước vector, [] – lấy ra giá trị của một phần tử hiện có, SetVal() – đặt một giá trị của một phần tử hiện có, …

Dưới đây là một ví dụ về vector



Kết quả thực hiện:

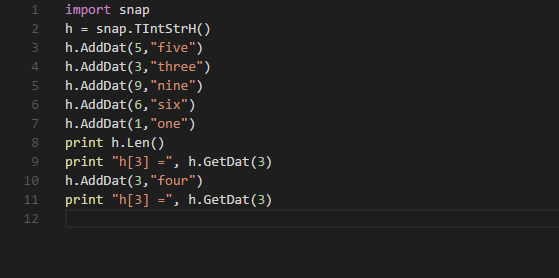


***Các loại Hash table***

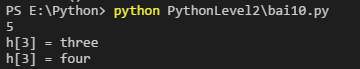
Đặt tên: <type1><type2>H: TIntH, TIntFlt, TStrH.

Mỗi cặp gồm (key, value), trong đó: key phải có cùng kiểu, value phải cùng kiểu dữ liệu (có thể khác kiểu dữ liệu với loại khóa). Chúng ta có thể thêm mới cặp (key, value). Các giá trị hiện tại được truy cập và thay đổi qua key.

Các thao tác: AddDat – thêm giá trị mới hoặc thay đổi giá trị hiện tại, Len – kích thước hash table, GetKey – trả về chìa khóa, GetDat – trả về giá trị, …



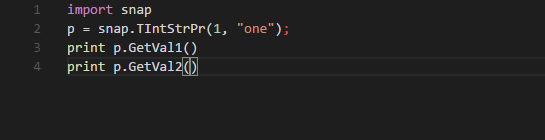
Và đây là kết quả nhận được:

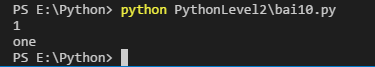


***Composite type***

Đặt tên: <type1, type2>Pr: TIntPr, TFltPr, TStrPr.

Type1 và type2 có thể khác nhau về kiểu dữ liệu.





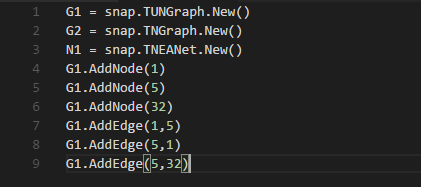
***Mạng và đồ thị***

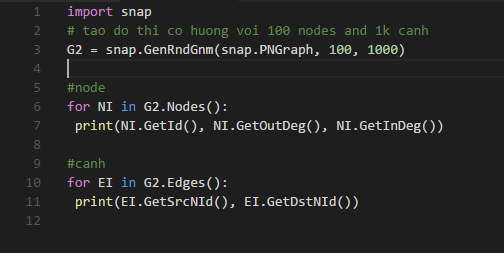
* TUNGraph: đồ thị vô hướng.
* TNGraph: đồ thị có hướng.
* TNEANet: multigraph với các thuộc tính trên các nút và cạnh.

Quy tắc:

* Đồ thị: tên đồ thị bắt đầu bằng “P”
* Function: bắt đầu bằng “T”
* Biến: bắt đầu bằng “P”

*Tạo mới một đồ thị với các nút và cạnh:*





*Save binary:*

FOut = snap.TFOut("test.graph")

G2.Save(FOut)

FOut.Flush()

*Load binary:*

FIn = snap.TFIn("test.graph")

G4 = snap.TNGraph.Load(FIn)

*Save và load từ 1 file:*

snap.SaveEdgeList(G4, "test.txt", “List of edges")

G5 = snap.LoadEdgeList(snap.PNGraph, "test.txt", 0, 1)

*Chuyển đổi đồ thị có hướng sang vô hướng:*

graph G7 = snap.ConvertGraph(snap.PUNGraph, G6)

*Thành phần kết nối yếu:*

WccG = snap.GetMxWcc(G6)

*Tạo 1 mạng sử dụng mô hình Forest Fire*

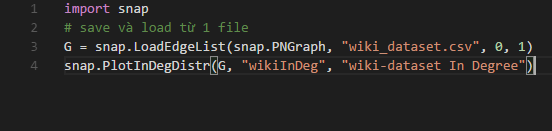
G8 = snap.GenForestFire(1000, 0.35, 0.35)

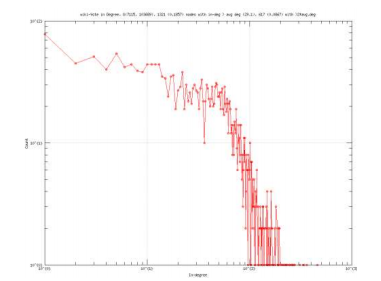
1. **Vẽ đồ thị với snap**

Tùy chọn vẽ đồ thị trong snap:

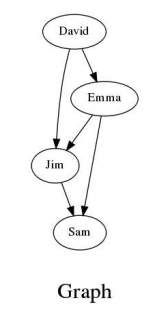
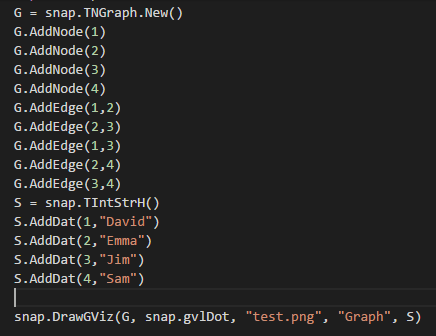
* Thuộc tính đồ thị: gnuplot: <http://www.gnuplot.info>
* Hiển thị đồ thị: graph: <http://www.graphviz.org>
* Tùy chọn khác: <http://www.matplotlib.org>

Ví dụ 1:





Ví dụ 2:



Sau khi thực hiện lệnh trên, 3 tập tin được tạo ra:

* .png hoặc .eps là plot
* .tab là tập tin chứa dữ liệu
* .plt là lệnh vẽ cho gnuplot

**CHƯƠNG II: TÌM HIỂU VIBER**

1. **Predicting Positive and Negative Links in Online Social Networks (Jure Leskovec, Daniel Huttenlocher, Jon Kleinberg – 11 Mar 2010)**

***Tóm tắt:***

Đề tài nghiên cứu về các mạng xã hội trực tuyến như Epinons, Slashdot và Wikipedia dựa trên việc xem xét các mối quan hệ tích cực hoặc tiêu cực. Từ đó, đưa ra dự đoán các liên kết trong các mạng xã hội tiềm ẩn với độ chính xác cao, sử dụng các mô hình tổng quát hóa thông qua sự đa dạng của các mạng xã hội trực tuyến.

Mô hình này nghiên cứu sâu về các nguyên tắc cơ bản thúc đẩy sự hình thành của các liên kết có dấu trong mạng xã hội, làm sáng tỏ lý thuyết về sự cân bằng từ các lời bình luận. Đồng thời, đề tài cũng đề xuất xây dựng ứng dụng có thể ước tính được quan điểm giữa các người dùng với nhau thông qua mối quan hệ của họ với các thành viên khác của mạng xã hội.

***Giới thiệu:***

**Câu hỏi cơ bản:** Cách mà một liên kết nhất định tương tác với các liên kết có dấu khác trong vùng lân cận của nó hay rộng ra là trên toàn mạng lưới. Trả lời các câu hỏi này có thể lý giải về mức độ tiêu cực trong các hệ thống trực tuyến. Đề tài cố gắng suy luận thái độ (không quan sát) giữa các người dùng với nhau bằng cách sử dụng các mối quan hệ tích cực và tiêu cực đã được quan sát được ở vụng phụ cận của người này. Từ đó, đề xuất các mối quan hệ mới với người dùng bằng cách đề xuất sự hình thành liên kết tới các người dùng khác có chung bạn bè, sở thích…

**Thách thức:** 1 người dùng có thể có cả quan điểm tích cực và tiêu cực từ trước với những người khác về những điều mà họ chia sẻ. Do đó, trước khi đưa ra gợi ý như trên thì phải ước tính bằng các bằng chứng hiện có trong mạng.

**Dự đoán dấu cạnh:** Giả sử một mạng xã hội có dấu hiệu trên tất cả các cạnh, tuy nhiên, dấu hiệu trên cạnh từ nút u đến nút v được biểu thị là s(u, v) đã được “ẩn”. Làm thế nào để xác định được dấu hiệu s(u, v) này một cách đáng tin cậy bằng cách sử dụng thông tin được cung cấp bởi phần còn lại của mạng?

Đề tài sử dụng một thử nghiệm khung được khớp nối bởi Guha trong nghiên cứu của họ về tin tưởng và không tin tưởng vào Epinions và mở rộng cách tiếp cận thông qua:

* Machine-learning framework cho phép đánh giá một trong những đặc tính cấu trúc của hầu hết các thông tin cho các nhiệm vụ dự đoán nhằm cải thiện hiệu suất dự đoán.
* Điều tra trên một loạt các tập dữ liệu và xác định nguyên tắc chung cho chúng => so sánh trực tiếp với các lý thuyết về các dấu hiệu liên kết tâm lý xã hội mà cụ thể là lý thuyết về sự cân bằng và trạng thái.

***Khái quát hóa trên các bộ dữ liệu:*** Thực hiện trên 3 bộ dữ liệu của 3 mạng xã hội có các cấu trúc và tính chất khác nhau: Epinions, Slashdot và Wikipedia để chứng minh phương pháp tiếp cận của tác giả phù hợp cho nhiều mô hình mạng xã hội khác nhau.

***Xem xét các vấn đề khác ngoài vấn đề dự đoán dấu hiệu:*** Đặt câu hỏi liệu thông tin về các liên kết “tiêu cực” có thể hữu ích trong việc giải quyết các câu hỏi liên quan đến các liên kết thuần túy.

***Dataset***

Epinions, Slashdot và Wikipedia

Trong tất cả các mạng, tỷ lệ cạnh “positive” tương ứng gần 80%.

***Dự đoán dấu cạnh***

***Machine-Learing Formulation:*** Đồ thị có dấu G = (V, E) với một dấu hiệu *positive* (ký hiệu “+”) và *negative* (ký hiệu “-”) trên mỗi cạnh.

s(x, y) biểu diễn dấu của cạnh (x, y):

s(x, y) = 1 nếu (x, y) là “+”

s(x, y) = -1 nếu (x, y) là “-”

s(x, y) = 0 nếu không có cạnh có hướng từ x đến y.

Đôi khi ta cũng quan tâm tới dấu hiệu của một cạnh trực tiếp kết nối x và y, bất kể hướng của nó.

Do đó, ta viết:

s¯ (x, y) = 1: khi có cạnh “+” một trong hai hướng (x, y) hoặc (y, x)

s¯ (x, y) = -1: khi có cạnh “-” một trong hai hướng (x, y) hoặc (y, x)

s¯ (x, y) = 0 cho tất cả các trường hợp khác

Giả sử: một cạnh (u, v) dấu hiệu s(u, v) hoặc s¯ (u, v) bị ẩn và chúng ta đang cố gắng để suy luận nó. Chia bộ dữ liệu thành 2 lớp: lớp học đầu tiên dựa trên trên bậc của nút, dựa trên việc ghi lại tổng hợp của mạng local mối quan hệ giữa của 1 nút với phần còn lại của mạng. Lớp thứ 2 dựa trên nguyên tắc tâm lý xã hội mà chúng ta có thể hiểu được mối quan hệ giữa các cá thể u và v thông qua mối quan hệ chung với bên thứ ba w. Với việc dự đoán dấu hiệu từ cạnh u tới v, đề tài xem xét các cạnh đi từ u và các cạnh tới điểm v.

Cụ thể: in(v) biểu thị số lượng cạnh “+” và “-” đến v, out(u) biểu thị số cạnh “+” và “-” đi từ u, C(u, v) để biểu thị tổng số hàng xóm chung của u và v mà không quan tâm tới hướng (tức số nút w sao cho w được liên kết bởi một cạnh ở cả hai hướng với cả u và v). Tất cả lớp thứ 2 được mã hóa thành vectow 16 chiều xác định bộ ba mỗi loại mà (u, v) có liên quan.

***Phương pháp nghiên cứu và kết quả:***

Sử dụng phương pháp hồi quy hậu cần: phân loại để kết hợp các bằng chứng là các đặc tính riêng biệt để dự đoán dấu hiệu cạnh. Hồi quy logistic được sử dụng để đào tạo mô hình học theo công thức:



Trong đó, x = (x1, x2,…,xn) là vector có n chiều và b0, b1, …, bn là hệ số được ước tính dựa trên dữ liệu huấn luyện. Do cạnh “+” có số lượng áp đảo với gần 80%. Đề tài xem xét và đánh giá dựa trên 2 hướng tiếp cận:

* Hướng 1: sử dụng bộ dữ liệu đầy đủ
* Hướng 2: làm theo phương pháp Guha, tạo một tập dữ liệu cân bằng với số lượng tương đương giữa các cạnh “+” và “-” => đoán ngẫu nhiên sẽ cho kết quả đúng 50%. Với độ đo là: độ chính xác phân loại và ROC là vùng dưới đường cong (UAC).

Hình dưới đây là biểu đồ biểu diễn kết quả dự đoán dấu hiệu cạnh (u, v) lần lượt trên 3 mạng xã hội: (A) Epinions, (B) Slashdot, (C) Wikipedia.

