Projeto de Pesquisa
Avaliação comparativa de três pacotes Python para redes complexas

Gonzalo Travieso

2022

Resumo

A linguagem Python tem sido amplamente utilizada em ciência computacional, devido à sua facilidade de uso e à disponibilidade de pacotes na linguagem para as mais diversas atividades científicas. Para o estudo de redes complexas, diversos pacotes Python estão disponíveis aos cientistas, entre eles o NetworkX, o Python igraph e e graph-tool. Surge então, para quem está interessado em começar um projeto na área, a questão de qual desses pacotes utilizar. Para ajudar a fundamentar essa escolha, este projeto propõe uma avaliação em termos de facilidade de uso e desempenho dos três pacotes citados. Os resultados devem ser disponbilizados para a comunidade, preferencialmente em um conjunto de páginas Web.

1 Introdução e Justificativa

Desde a virada do milênio, a ciência de redes [1] tem apresentado uma grande atividade, com aplicações nas mais variadas áreas de conhecimento [2]. Têm se demonstrado características topológicas comuns na estrutura de interação entre os elementos de diversos sistemas e também o impacto dessas características em dinâmicas que ocorrem nessas redes [3].

Na análise de redes reais, após a coleta de dados sobre o sistema, deve ser construído um grafo que representa as interações entre seus elementos. Como os sistemas têm frequentemente um grande número de elementos o grafo resultante tem um grande número de vértices. Para caracterizá-lo, diversas medidas [4] podem precisar ser calculadas. Para possibilitar a análise de redes grandes é pois importante que tanto sua representação no computador quanto a implementação dos algoritmos usados seja eficiente [5].

Outro fenômeno importante é o crescimento do uso da linguagem Python em ciência computacional. Um dos principais fatores para isso é o desenvolvimento dos projetos relacionados NumPy [6], SciPy [7], Matplotlib [8], Jupyter [9], Pandas [10] e scikit-learn [11]. Esses pacotes e ferramentas facilitam o desenvolvimento do software e aumentam a produtividade do cientista computacional. Da mesma forma existem pacotes Python disponíveis para análise de redes. Citaremos aqui em especial o NetworkX [12], o Python-igraph [13] e o graph-tool [14].

A efetividade dessas ferramentas está diretamente ligada à sua facilidade de uso e ao seu desempenho em tempo de execução.

2 Objetivo

O objetivo do trabalho a ser realizado pelo bolsista é avaliar comparativamente o NetworkX, o Python-igraph e o graph-tool em termos qualitativos quanto à sua facilidade de uso e em termos quantitativos quanto ao seu desempenho. Devem ser avaliadas a facilidade de instalação dos pacotes, a facilidade de uso das ferramentas disponibilizada, a abrangência de cobertura e o desempenho comparativo, especialmente para redes grandes. Os resultados devem ser publicados, preferencialmente em páginas Web, para garantir fácil acesso aos pesquisadores da área.

3 Metodologia

Dada a abrangência limitada deste projeto, as análises qualitativas serão realizadas de maneira subjetiva, através da experiência do próprio bolsista na instalação e aprendizado de uso dos pacotes. Uma avaliação com base em testes envolvendo grupos de programadores seria cientificamente mais válida, mas não poderá ser realizada no escopo deste projeto.

As análises quantitativas de desempenho serão feitas selecionando um grupo de cálculos presentes nas três ferramentas e os executando com as mesmas entradas diversas vezes, para realizar um levantamento estatístico dos tempos de execução. Devem ser realizados estudos com diferentes números de vértices e números de arestas, para verificar a escalabilidade das implementações.

Serão realizadas as seguintes atividades:

1. Criar uma máquina virtual Linux com uma instalação Linux padrão, com a versão mais recente de Python disponível na distribuição instalada, mas sem outros pacotes, tanto do sistema quanto do Python. Usaremos essa máquina virtual para verificar o processo de instalação dos pacotes, sem a influência de pacotes que já estejam instalados no sistema computacional a ser usado.

2. Instalar nessa máquina virtual o `pip` e o ambiente `minconda`.

3. Criar um ambiente virtual `python venv` e um `conda` para cada um dos pacotes a serem avaliados. A intenção é manter os pacotes separados.

4. Instalar os três pacotes usando `pip` e usando `conda`, nos distintos ambientes virtuais. Neste processo, anotar o processo de instalação, possíveis dificuldades que ocorram nas instalações, e verificar quanto espaço de disco foi utilizado por cada pacotes depois da instalação. Terminada a análise das instalações, a máquina virtual Linux criada não será mais necessária, e o restante do trabalho pode ser realizado diretamente no computador hospedeiro.

5. Se familiarizar com o uso dos três pacotes. Isto deve ser feito através da leitura de tutoriais e manuais de usuário e da execução de operações simples de análise de rede nos três pacotes. Neste processo anotar a disponibilidade e qualidade do material de estudo para cada pacote e o quanto o uso do pacote é intuitivo para um programador Python.

6. Catalogar toda a funcionalidade fornecida pelos três pacotes e comparar a abrangência de funcionalidades desse pacotes. Notar também possíveis limitações ou dificuldades em certas funcionalidades para algum pacote.

7. Selecionar algumas das funcionalidades para a realização de análise comparativa de desempenho. A seleção deve usar os seguintes critérios:

   - As funcionalidades devem estar disponíveis nos três pacotes.
   - Elas devem abranger tipos distintos de operações, como geração de grafos, análise de medidas de vértices, análise de medidas globais da rede, estrutura de comunidades, etc.
   - Elas devem envolver tanto algoritmos de baixa quanto de alta complexidade computacional.
   - O número de funcionalidades a incluir será limitado à quantidade de tempo disponível, visto que precisam ser realizados os cálculos para diversos tamanhos de rede, e em diversas redes para cada tamanho, para lidar com aspectos estocásticos. Portanto, a decisão sobre o que incluir somente poderá ser feita depois de realizar alguns experimentos preliminares com os pacotes.

   Escolhidas as funcionalidades, realizar os cálculos e temporização de sua execução para diversos modelos de rede, diversos tamanhos de rede e para diversas redes de cada modelo e tamanho. Analisar comparativamente os tempos de execução e a escalabilidade com o tamanho da rede.

8. Escrever, usando os três pacotes, códigos para algumas análises específicas de redes e comparar qualitativamente os código resultantes e quantitativamente seu desempenho. Inicialmente, estamos pensando em escrever código para:

   - Reproduzir o resultado de percolação em redes de Erdős e Rényi [15].
   - Reproduzir o resultado sobre o efeito de mundo pequeno de Watts e Strogatz [16].
   - Reproduzir o resultado de resiliência a ataques aleatórios e direcionados de redes com distribuição de grau livre de escala de Albert, Jeong e Barabási [17].

4 Análise dos resultados

Os tabalhos listados na seção 3 permitirão uma avaliação comparativa dos três pacotes em questão. Serão analisados os seguintes pontos:

- Comparação da dificuldade de instalação dos pacotes supondo um sistema sem outros pacotes Python pré-instalados, além dos que são distribuídos por *default*, tanto para usuários de `pip` como de `conda`.
- Comparação da dificuldade de aprendizado de uso dos pacotes, tanto em termos da disponibilidade de boa documentação *on-line* quanto da própria organização dos pacotes.
- Comparação da abrangência dos pacotes, em termos das funcionalidades prontas fornecidas.
- Comparação do desempenho dos pacotes, verificando se existe uma vantagem geral para algum dos pacotes ou se o desempenho relativo varia de uma funcionalidade para outra. Aqui deve ser verificado estatisticamente se existe diferença significativa entre as médias de tempo de execução. Por outro lado, ao tirar conclusões nesse ponto deve-se levar em consideração que os códigos serão executados em apenas um computador, e portanto os resultados podem ser diferentes em outros computadores, especialmente se as diferenças de tempo forem pequenas.

5 Cronograma de Atividades

As atividades listadas na seção 3 podem ser realizadas sequencialmente na ordem apresentada. Uma avaliação preliminar de sua duração segue:

- Itens 1 a 4: 2 meses.
- Item 5: 2 meses.
- Item 6: 4 meses.
- Item 7: 3 meses.
- Item 8: 1 mês.

Referências

[1] M. Newman. *Networks*. Oxford University Press, 2nd edition, 2018.

[2] L. da F. Costa, O. N. Oliveira Jr, G. Travieso, F. A. Rodrigues, P. R. Villas Boas, L. Antiqueira, M. P. Viana, and L. E. Correa Rocha. Analyzing and modeling real-world phenomena with complex networks: a survey of applications. *Advances in Physics*, 60(3):329–412, 2011.

[3] S. Boccaletti, V. Latora, Y. Moreno, M. Chavez, and D.-U. Hwang. Complex networks: Structure and dynamics. *Physics Reports*, 424(4):175–308, 2006.

[4] L. da F. Costa, F. A. Rodrigues, G. Travieso, and P. R. Villas Boas. Characterization of complex networks: A survey of measurements. *Advances in Physics*, 56(1):167–242, 2007.

[5] C. A. Ruggiero, O. M. Bruno, G. Travieso, and L. da F. Costa. On the efficiency of data representation on the modeling and characterization of complex networks. *Physica A-Statistical Mechanics and Its Applications*, 390(11):2172–2180, 2011.

[6] C. R. Harris, K. Jarrod Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. Fernández del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020.

[7] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. Jarrod Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Yu Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020.

[8] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007.

[9] T. Kluyver, B. Ragan-Kelley, F. Pérez, B. Granger, M. Bussonnier, J. Frederic, K. Kelley, J. Hamrick, J. Grout, S. Corlay, P. Ivanov, D. Avila, S. Abdalla, and C. Willing. Jupyter notebooks – a publishing format for reproducible computational workflows. In F. Loizides and B. Schmidt, editors, *Positioning and Power in Academic Publishing: Players, Agents and Agendas*, pages 87–90. IOS Press, 2016.

[10] W. McKinney. Data Structures for Statistical Computing in Python. In S. van der Walt and J. Millman, editors, *Proceedings of the 9th Python in Science Conference*, pages 56–61, 2010.

[11] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[12] A. A. Hagberg, D. A. Schult, and P. J. Swart. Exploring network structure, dynamics, and function using networkx. In G. Varoquaux, T. Vaught, and J. Millman, editors, *Proceedings of the 7th Python in Science Conference*, pages 11–15, Pasadena, CA USA, 2008.

[13] G. Csardi and T. Nepusz. The igraph software package for complex network research. *InterJournal*, Complex Systems:1695, 2006.

[14] T. P. Peixoto. The graph-tool python library. *figshare*, 2014.

[15] P. Erdős and A. Rényi. On the strength of connectedness of a random graph. *Acta Math. Sci. Hungary*, 12:261–267, 1961.

[16] D. J. Watts and S. H. Strogatz. Collective dynamics of "small-world" networks. *Nature*, 393:440–443, 1998.

[17] R. Albert, H. Jeong, and A.-L. Barabási. Error and attack tolerance of complex networks. *Nature*, 406:378–382, 2000.