Transició de fase i components connexes en grafs aleatoris.

Grup 8:

QUINTANA TORRES, JOSEP

BOYANO IVARS, PAULA

RUBINSTEIN PÉREZ, FEDERICO

MARQUÉS GODÓ, JORDI

Introducció

En aquest projecte, es farà un estudi experimental a partir de diferents grafs generats de manera aleatoria, concretament, models parametritzats de grafs aleatoris. Analitzarem específicament les transicions de fase (dels grafs esmentats) per mesuraments sobre les components connexes.

La transició de fase, si n'hi ha, és aquell valor del paràmetre que determina el graf aleatori on hi ha una important diferencia entre la probabilitat de ser connex entre els valors menors i els majors al donat.

Primerament introduirem els dos tipus de generació de grafs aleatoris que utilitzarem per l'anàlisi: Binomial Random Graph (Erdős–Rényi model) y Random Geometric Graph.

Tot seguit exposem els experiments que hem dut a terme mostrant les gràfiques que reflexen l'evolució i els canvis entre les execucions que hem fet mentre modificavem els paràmetres que ens interessaven i fem una breu valoració dels resultats.

Finalment traiem les conclusions sobre les propietats estudiades, afegim una explicació del que hem après i les referències bibliogràfiques.

Tipus de grafs:

En aquest estudi utilitzem dos models diferents parametritzats de grafs aleatoris.

Binomial Random Graph (Erdős–Rényi model)

Aquest model consisteix en construir un graf G(n, p) connectant nodes de manera aleatoria. Cada aresta s'inclou en el graf amb n nodes amb una probabilitat p.

A mesura que p augmenta (té valors entre 0 i 1), será més probable que al model s'incloguin grafs amb més arestes.

Les aplicacions d'aquest model són molt limitades, atès que poques xarxes reals es comporten tal com es descriu en el model d'Erdös–Rényi. (1) Tot i així, s'utilitza com a base teòrica en la generació d'altres xarxes.

Random Geometric Graph

Aquest model consisteix en col·locar de manera aleatòria n nodes en un espai mètric, segons una distribució de probabilitat específica, y connectar dos nodes amb una aresta només si la distància entre ells està dins d'un rang donat (r), obtenint així el graf G(n,r).

Un espai mètric és un conjunt A, amb una funció de distancia d, entre totes les parelles d'elements que formen el conjunt.

Experiments:

El nostre objectiu és estudiar la transició de fase de diferents paràmetres en diferents tipus de grafs aleatoris. Per això, hem decidit realitzar experiments molt similars però canviant el generador del graf i els paràmetres a estudiar.

Per a cada generador i paràmetre, hem seguit el següent esquema:

- Escollim 10 números diferents de vèrtexs (n), de 100 en 100 fins a 1000.
- Fem probes de diferents valors variables, com ara la probabilitat d'adjacència o el radi del graf.
- Per a cada valor n repetim múltiples vegades el mateix experiment, minimitzant així els resultats inesperats causats per l'aleatorietat.

També ens interessa saber el temps de còmput de les implementacions dels nostres algorismes que permeten comprovar les propietats sota estudi en funció dels diferents valors dels paràmetres dels models de grafs i del hardware utilitzat. Per a això, hem fet servir la llibreria de c++ <time.h>, que ofereix una funció clock() amb la que podem contar els ticks del rellotge intern de l'ordinador i seguidament passar-ho a segons amb CLOCKS_PER_SEC. Hem realitzat aquest procés amb cada una de les nostres implementacions dels algorismes.

El hardware i SO que hem utilitzat per realitzar els experiments es el següent:

- Procesador: Intel® Core™ i5-7300HQ CPU @ 2.50GHz × 4
- Ubuntu 19.04 , 64-bits

Finalment, per a obtenir les dades i representar-les de manera visual, hem utilitzat fitxers amb extensió .csv per a guardar les dades i hem generat els gràfics utilitzant python i les llibreries seaborn, pandas i matplotlib.

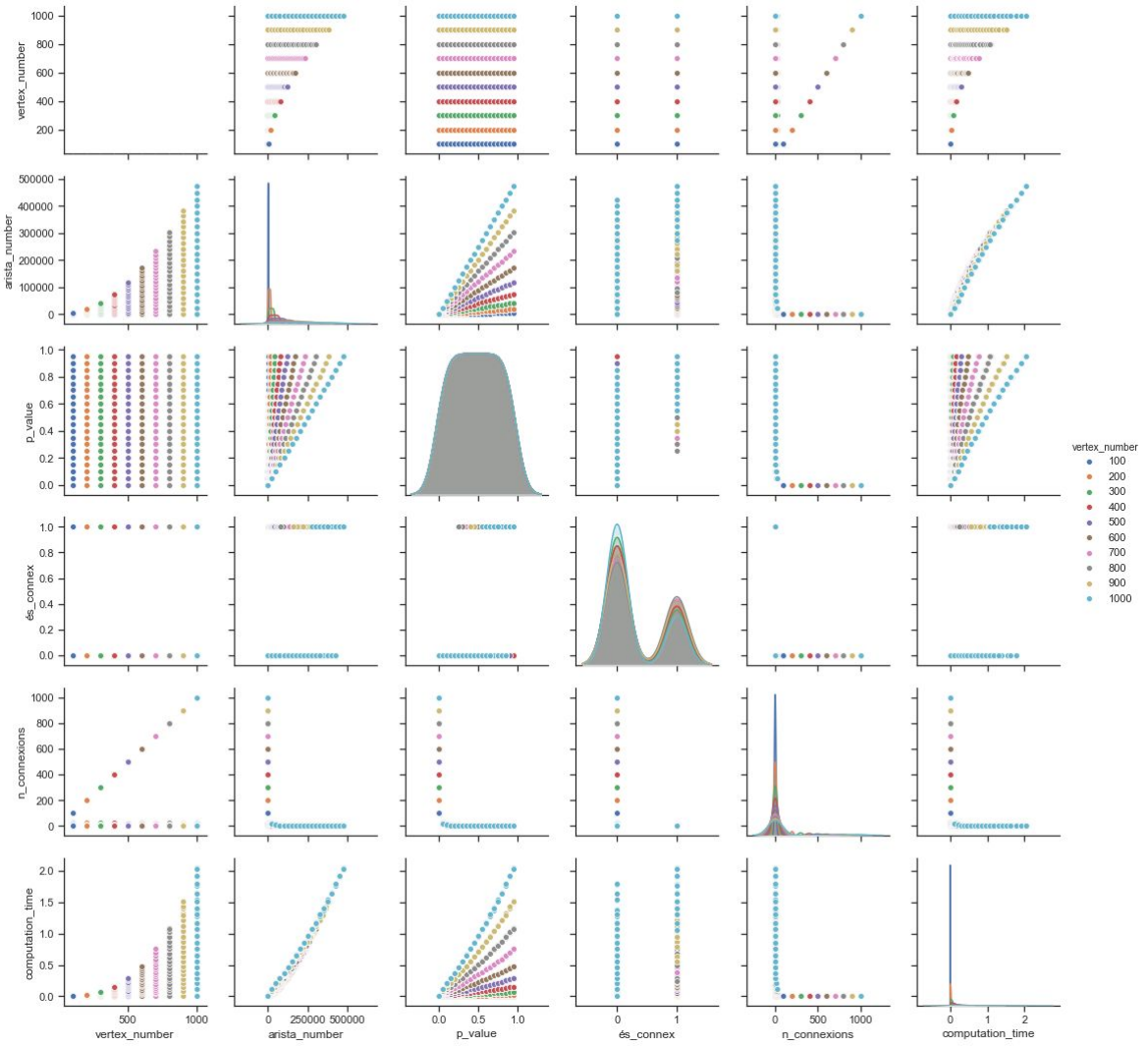
Primer, hem creat tots els gràfics possibles, és a dir, dos gràfics a cada conjunt de variables x, y. Així, podem observar fàcilment quins ens interessen més analitzar i, seguidament, treballar-los millor.

Resultats obtinguts:

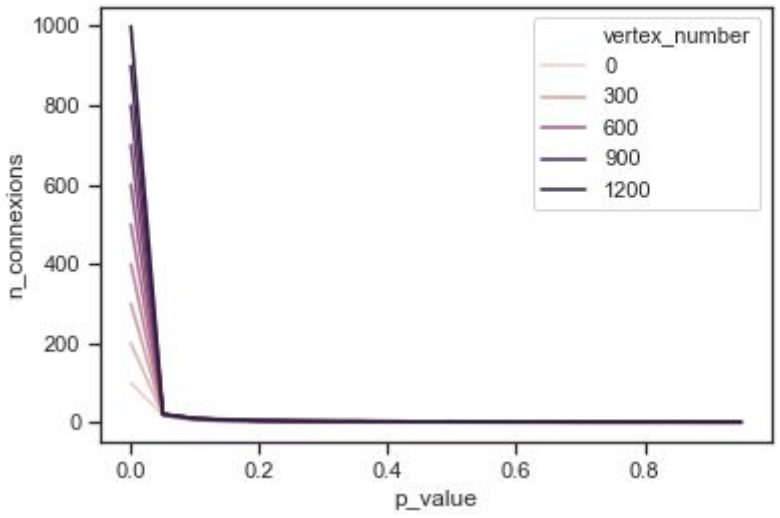
Binomial Random Graph (Erdős–Rény) - Graf(n,p):

Per aquest graf, hem obtingut dades de les següents variables:

- vertex_number: nombre de vèrtexs del graf
- arista_number: nombre d'arestes del graf
- p_value: valor entre 0 i 1 que indica la probabilitat que hi hagi una aresta
- és_connex: si el graf resultant és connex (aquest valor és binari)
- n_connexions: Nombre de connexions que té el graf
- computation_time: el temps que triga la nostra implementació en saber les components connexes, en segons



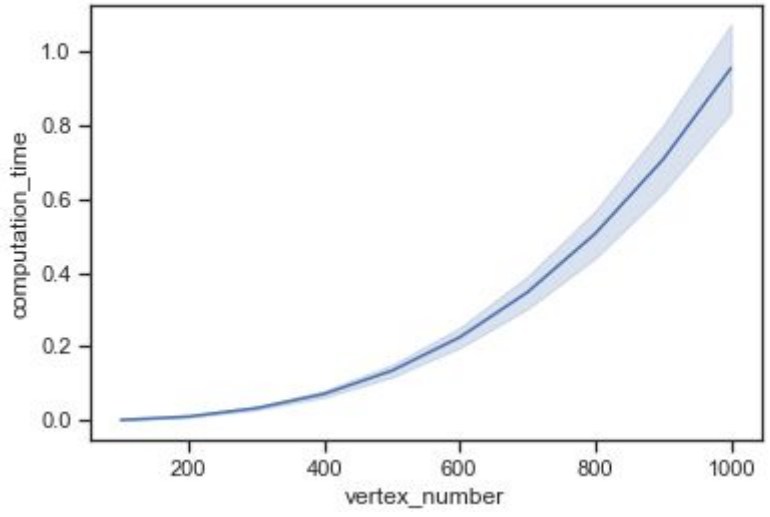
Els gràfics que hem generat amb totes les dades recollides. Alguns no tenen sentit, per això a continuació ens centrarem en els que ens han semblat més interessants.



En aquest gràfic estudiem el nombre de connexions depenent de p.

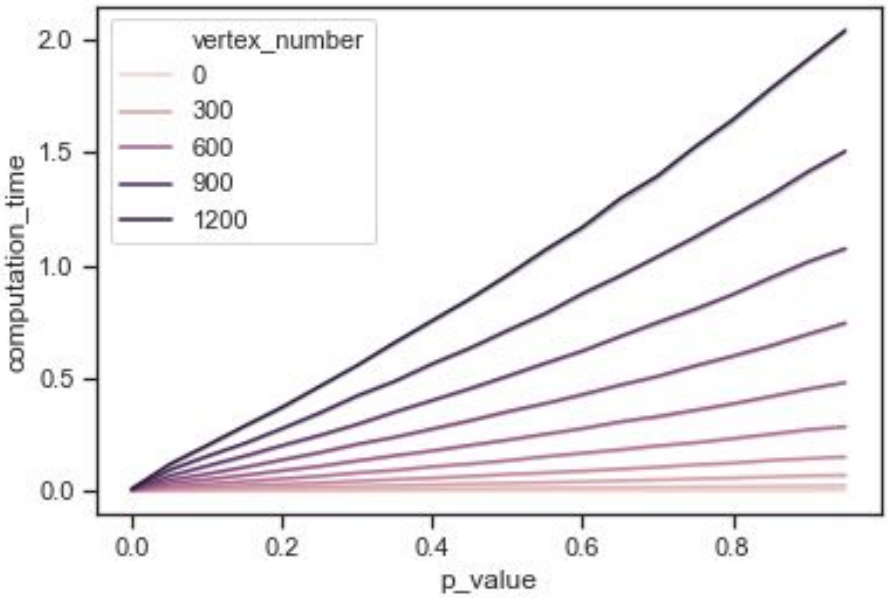
Com era d'esperar, el nombre de connexions s'acosta molt a 1 quant més alt és el valor de p, ja que augmenten les arestes del graf.

També es pot apreciar que les diferents mostres tenen un valor de transició de fase molt similar.

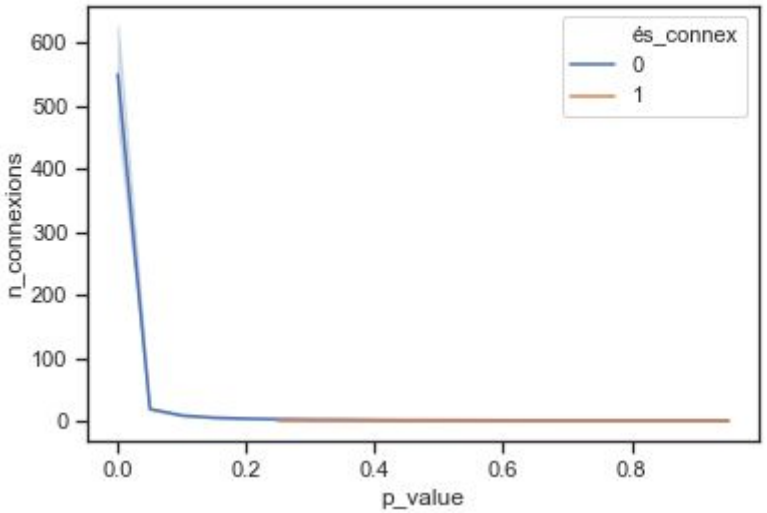


En aquest gràfic veiem la relació entre el nombre de vèrtexs i el temps que triga en computar la nostra funció `getConnectedComponents` que

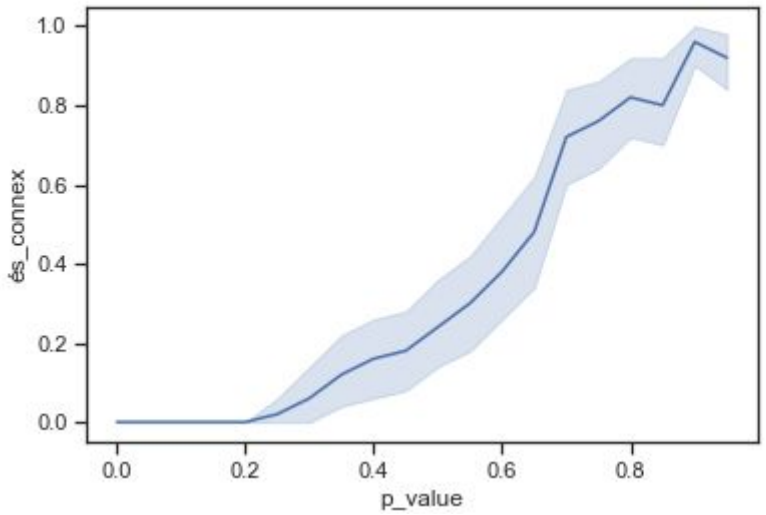
utilitzem per a saber la connectivitat del graf. Es pot apreciar clarament com creix exponencialment.



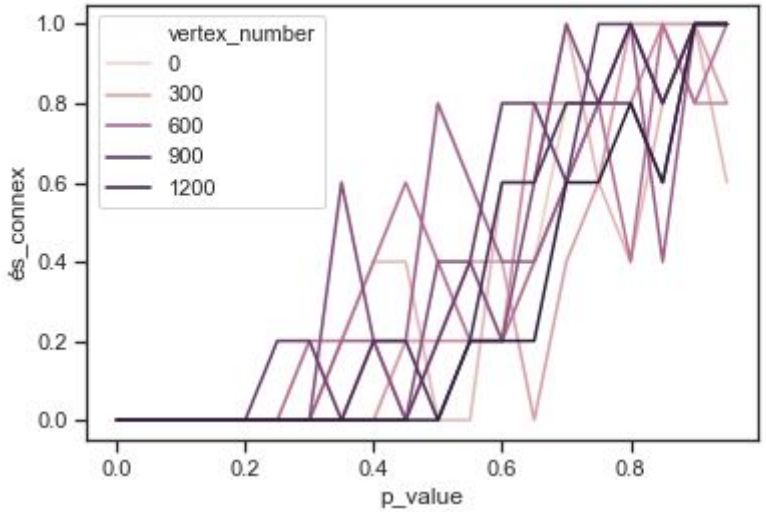
Aquest gràfic de la relació entre p i el temps de còmput ensenya clarament com creix linealment, en contraposició de la relació entre el nombre de vèrtexs i el temps de còmput que creix exponencialment. Això és degut a que el valor de p està directament relacionat amb les arestes del graf, i, per tant, la seva connectivitat. El nostre algorisme creix linealment pel nombre d'arestes.



Ens ha interessat aquest gràfic perquè es pot apreciar clarament la transició de fase de la probabilitat de que el graf sigui connex dependent del valor de p.



Aquí podem veure la probabilitat de que un graf sigui connex depenent del valor que prengui p. Aquest gràfic és important ja que el valor de p és el paràmetre més decisiu a l'hora de que un gràfic sigui connex. Com es pot apreciar, la probabilitat comença a créixer a mesura a partir del valor de p 0.2, això és perquè, com hem vist al gràfic anterior, a partir de 0.2 és quan el nombre de components comença a ser 1. Creiem que, si haguéssim fet més experiments, aquest gràfic tendiria a ser lineal ja que eliminariem l'aleatorietat.

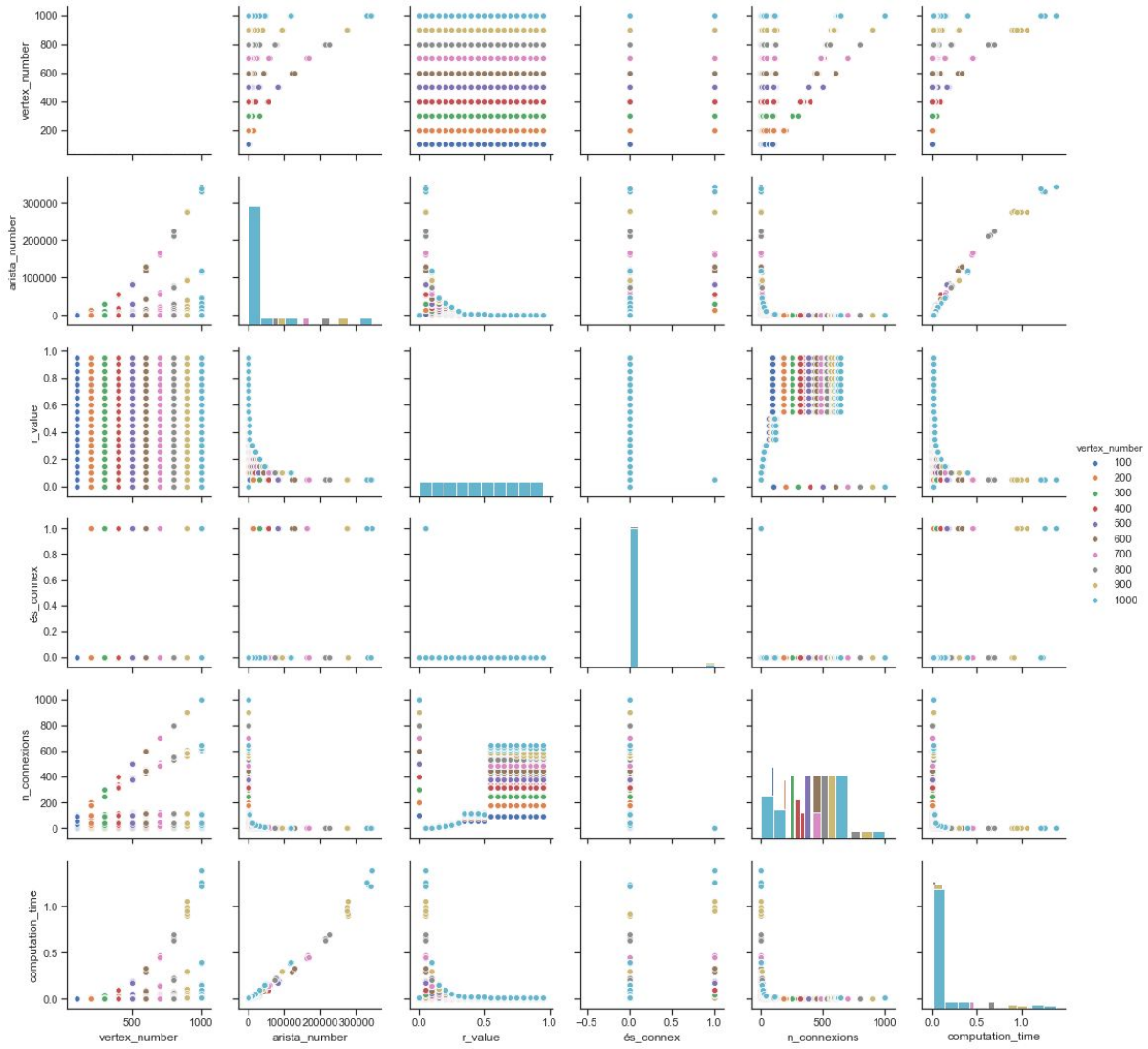


Aquest gràfic també és de la relació entre el valor de p i si és connex o no, però en aquesta ocasió ho hem desglossat pel nombre de vèrtexs del graf.

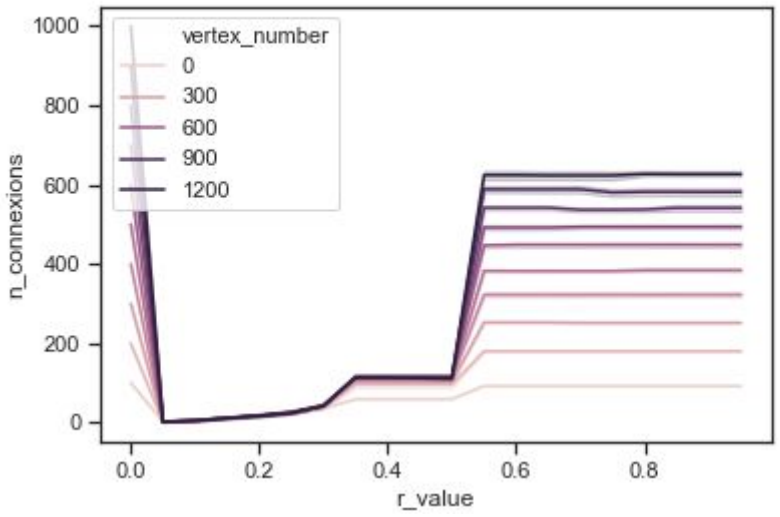
Random Geometric Graph - Graf(n,r):

Per aquest graf, hem obtingut dades de les següents variables:

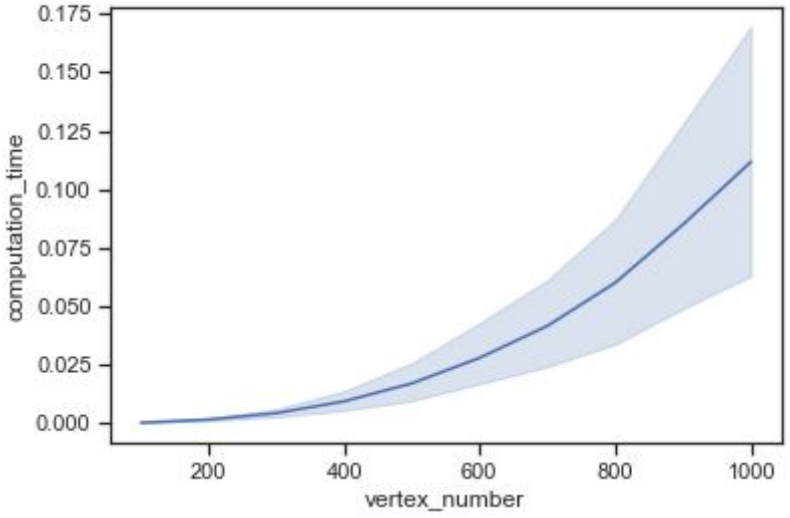
- vertex_number: nombre de vèrtexs del graf
- arista_number: nombre d'arestes del graf
- r_value: valor entre 0 i 1 el qual fa que dos vèrtex p i q ϵ V estiguin connectats únicament si la seva distància és menor que r
- és_connex: si el graf resultant és connex (aquest valor és binari)
- n_connexions: Nombre de connexions que té el graf
- computation_time: el temps que triga la nostra implementació en saber les components connexes, en segons



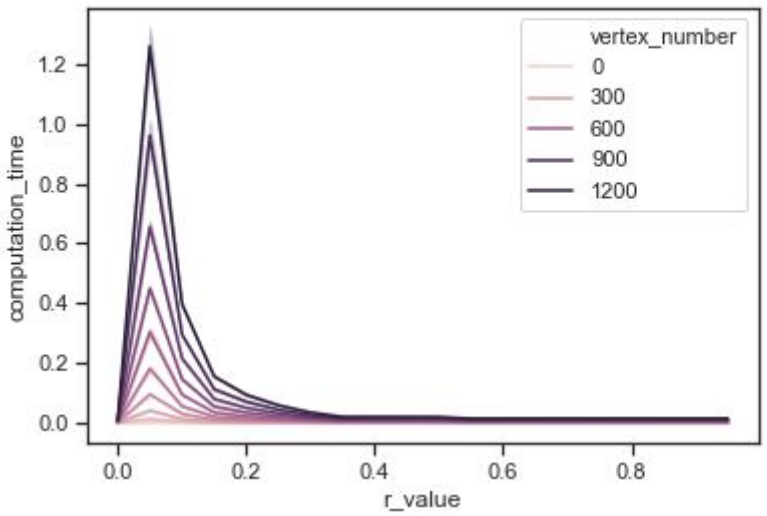
Aquests són tots els gràfics que podem obtindre amb les dades recollides. A continuació, veurem amb més detall els que ens han semblat més interessants d'estudiar.



En aquest cas podem observar la relació que guarda el nombre de connexions i el valor de r. Com era d'esperar, per una r molt propera a 0, el nombre de connexions és pràcticament el mateix que el nombre de vèrtexs del graf. És interessant notar com, a partir del valor r ≈ 0.5, hi ha una transició de fase notable.



En aquest gràfic mostrem el temps de còmput de la nostra funció `getConnectedComponents` dependent del nombre de vèrtexs del graf. Com podem observar, creix exponencialment.



Aquest gràfic és molt interessant ja que mostra com, per una r molt petita, augmenta de manera significativa el cost de la funció `getConnectedComponents`. Creiem que això és degut a la implementació que hem utilitzat per representar els grafs, l'algorisme Union-Find, què per a grafs pràcticament connexos, el temps d'executar la nostra funció és molt petit.

Components Connexes Gegants:

Considerem una component connexa gegant quan compleixi dues condicions:

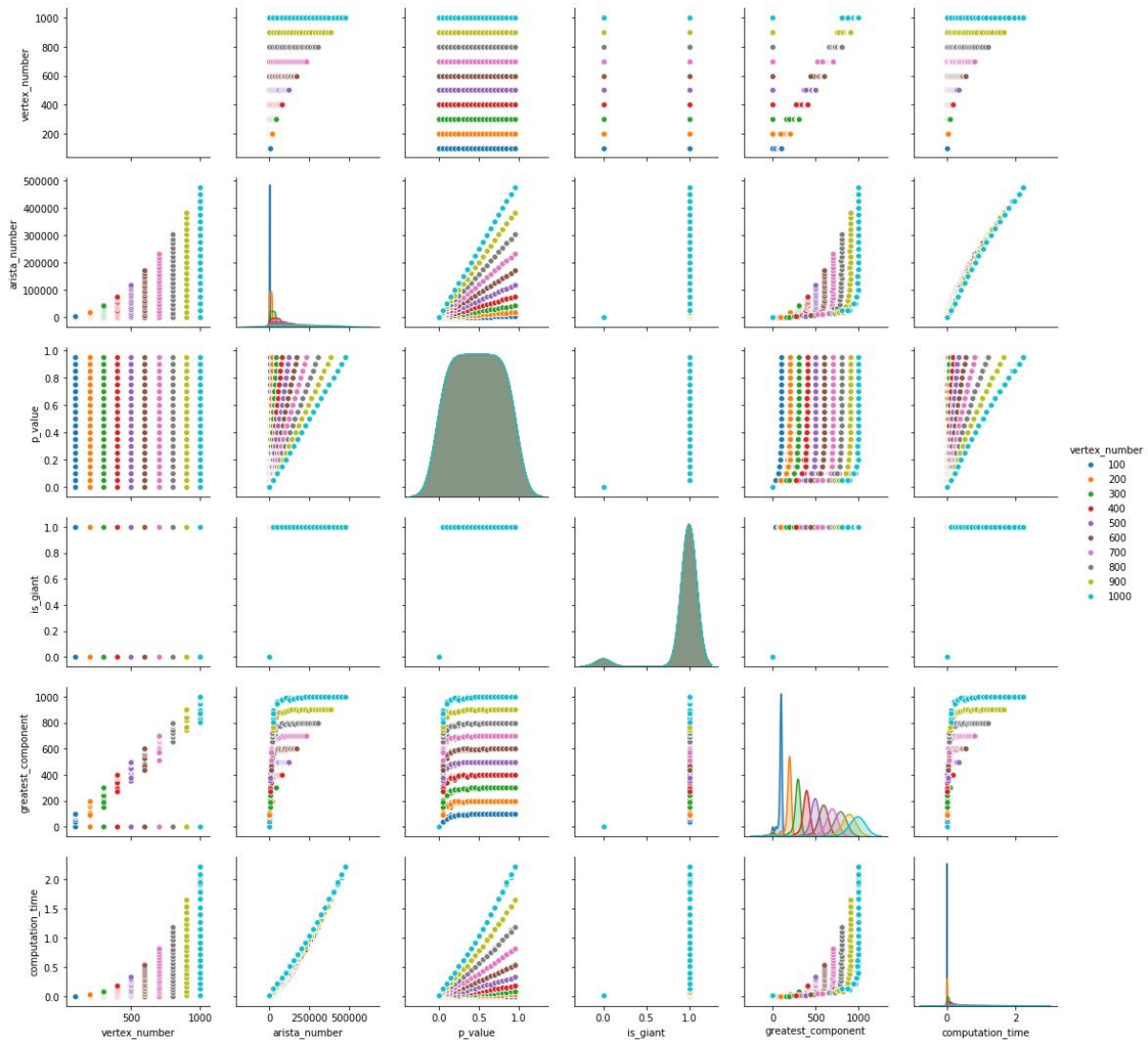
- És la component més gran del graf
- Compleix la fórmula següent:

$$\mathbb{E}[k^2] - 2\mathbb{E}[k] > 0$$

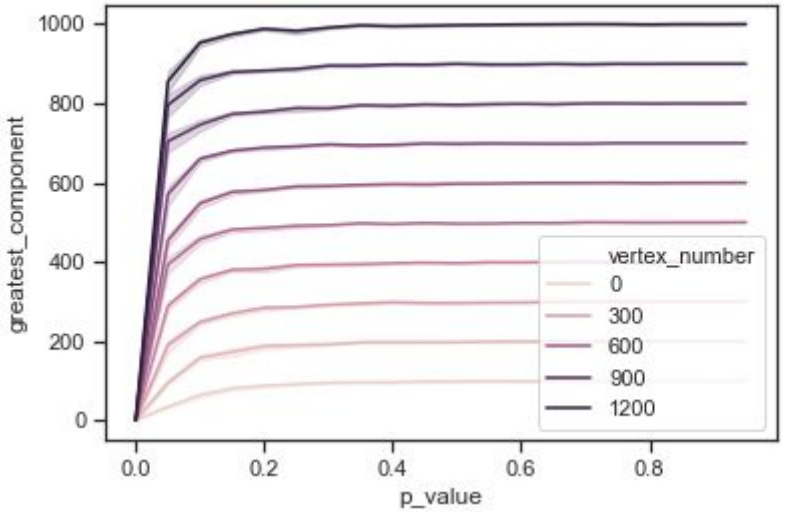
On E es un vèrtex qualsevol de la component connexa, i k es el nombre d'arestes d'aquest node.

Per aquest estudi, hem obtingut dades de les següents variables:

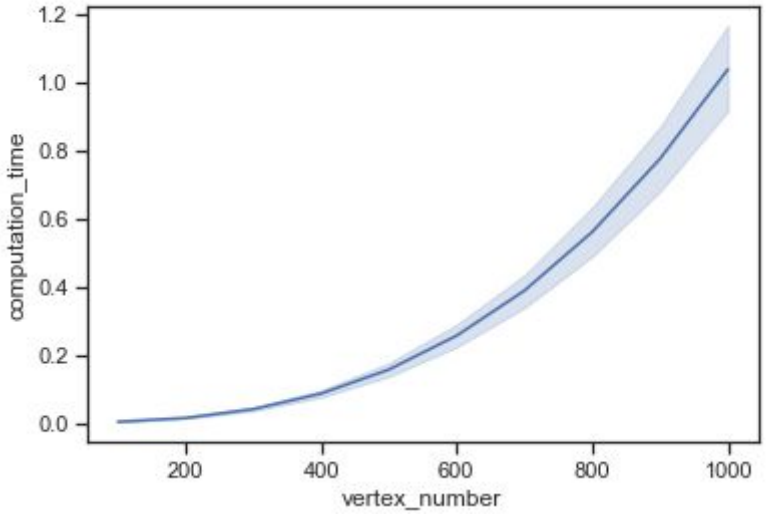
- vertex_number: nombre de vèrtexs del graf
- arista_number: nombre d'arestes del graf
- p_value: valor entre 0 i 1 que indica la probabilitat que hi hagi una aresta
- is_giant: si el graf resultant és gegant (aquest valor és binari)
- greatest_component: Nombre de vèrtexs que té la component connexa gegant
- computation_time: el temps que triga la nostra implementació en saber la component gegant, en segons



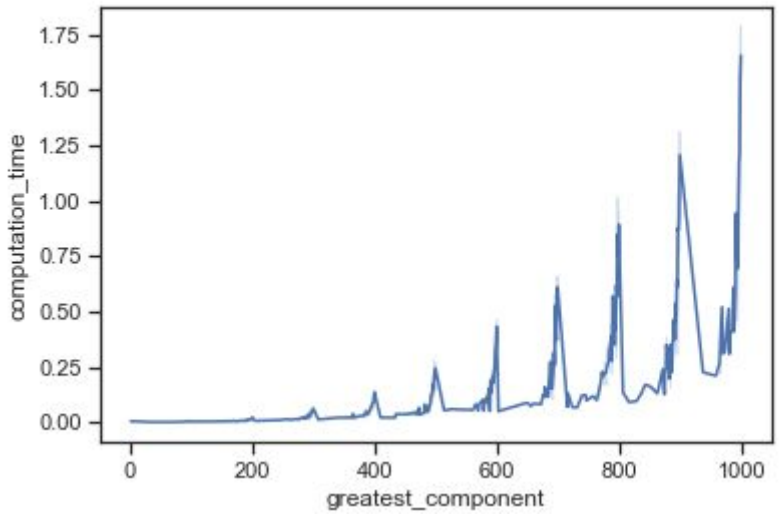
Tornem a representar tots els gràfics possibles per a tenir un punt de partida i poder-nos fixar en els gràfics més interessants.



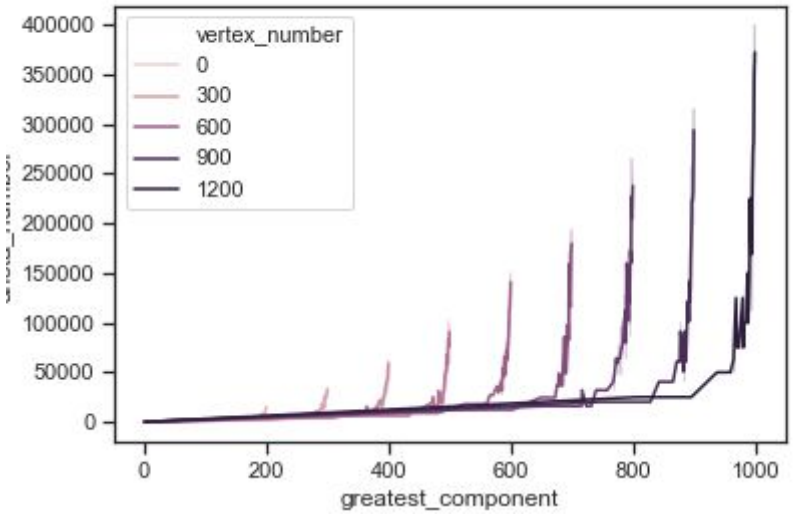
Per a estudiar la transició de fase de la mida esperada de la component connexa gegant hem fet aquest gràfic on podem veure que, com era d'esperar, la component connexa gegant creix ràpidament a mesura q creix el valor de p, però després es manté bastant estable, fins que finalment equival al nombre de vèrtexs (s'ha convertit en un graf connex). Es pot veure clarament la transició de fase.



No ens sorprèn que el temps de còmput de la funció `greatest_components` sigui exponencial respecte el nombre de vèrtexs, ja que realment el cost d'aquesta funció és el cost de la funció `getConnectedComponents` + el número de components connexes i, com hem vist anteriorment, la funció `getConnectedComponents` té un cost exponencial respecte el nombre de vèrtexs.



En aquest altre gràfic sobre el temps de còmput, també podem veure que, per diferents nombres de vèrtexs, el temps de còmput creix exponencialment.



Aquest gràfic mostra la relació entre la component connexa gegant i el nombre d'arestes del graf, desglossat pel nombre de vèrtexs. Ens ha interessat d'aquest gràfic com creixen exponencialment les arestes amb la component connexa i que es veu clarament la transició de fase d'aquesta.

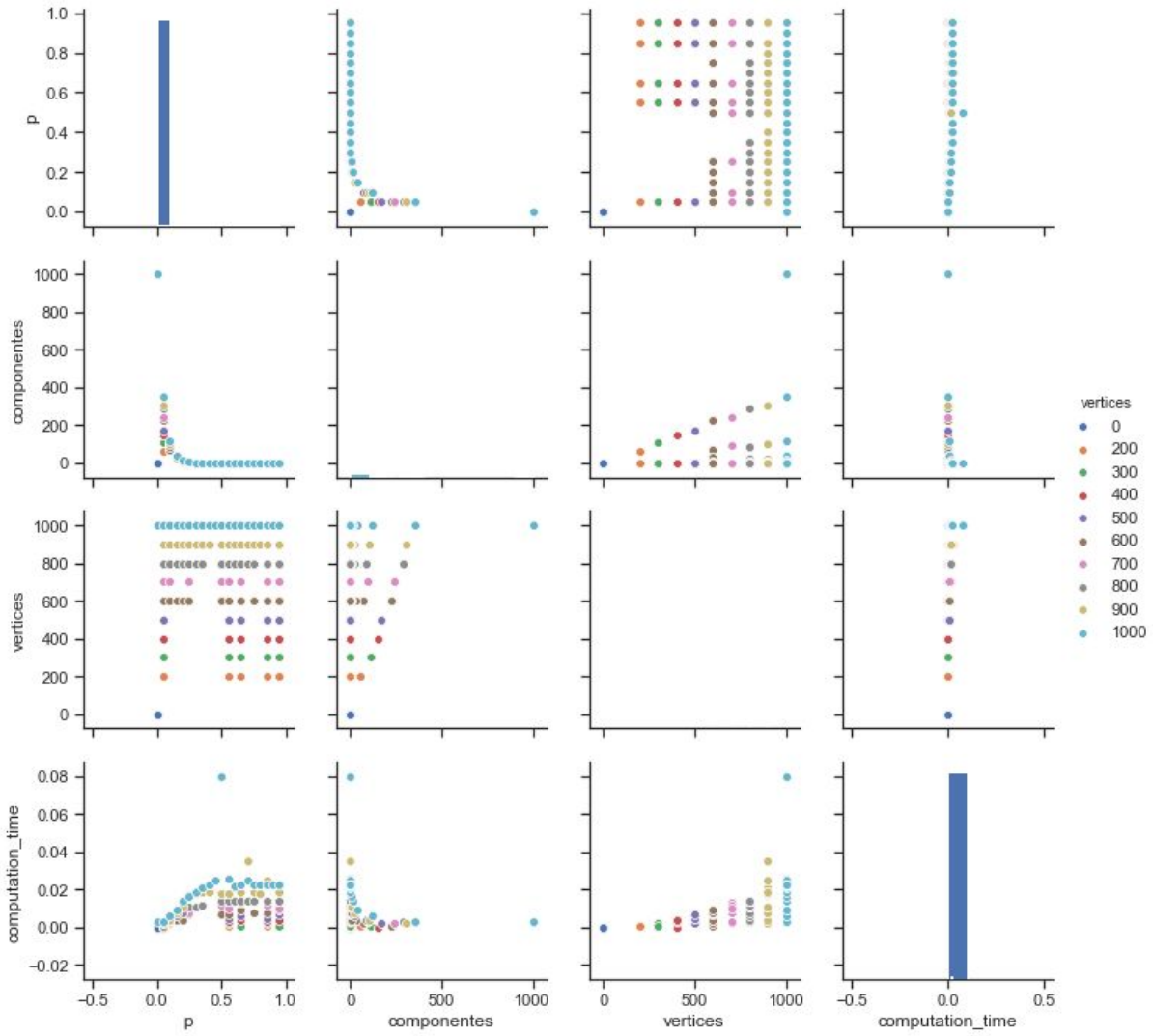
Uniform Random Intersection Graph - Graf(n,m,p):

Com a graf extra, hem escollit el graf G(n,m,p), un graf aleatori que es construeix de la següent manera:

S'etiqueta cada node n amb un set de p colors diferents escollit aleatoriament de un set de tamany m colors. Dos nodes u i w ϵ V estan units per una aresta si i només si un color apareix en els sets dels dos nodes.

Per implementar-ho, hem utilitzat la llibreria NetworkX de python.

Els gràfics obtinguts són els següents:



Ens interessa d'aquests gràfics veure com la transició de fase és molt similar a la que hem estat veient durant tot el treball en els altres models aleatoris (com podem veure en els gràfics de la relació entre p i el nombre de components). Això ens porta a pensar que aquesta serà similar a tots els models de grafs aleatoris. Creiem que per això són útils per a l'estudi d'altres tipus de grafs.

El temps de còmput que hem calculat és el temps que triga en calcular el nombre de components connexes la llibreria NetworkX.

Replicar Experiments:

Per realitzar una replica dels experiments que hem realitzat és necessari contar amb el compilador g++, i en la carpeta src realitzar primer un #make, i després executar el #main.x.

Per obtenir els experiments extra i els gràfics, és necessari python3, pandas, seaborn, matplotlib i networkx.

Per obtenir els gràfics i les dades de l'experiment extra cal la comanda #python3 creaGrafics.py

Conclusions:

Hem pogut concloure que per una part que, tant si un graf és connex com si conté una connexió gegant, semblaria ser, que no depèn del número de vèrtex, si no de la probabilitat de tenir arestes (p_value) i del valor de r.

També podem assumir que el cost dels algorismes utilitzats, són exponencials pel nombre de vèrtex i lineals per el número d'arestes, i per tant, lineal pel valor de la probabilitat p i el valor de r.

Gràcies a això, aquesta informació ens serviria per aplicar-la a casos de simulacions en el món real, podent fer estimacions d'un sistema amb milions de nodes utilitzant grafs més petits per representar-los.

Per una altre banda, no considerem haver pogut experimentar de la manera més òptima possible amb el *Random Geometric Graf,* potser hauríem de provar més experiments diferents, o fixar-se més en certs valors més rellevants del r_value.

Gràcies a aquesta pràctica, hem pogut aprendre el funcionament de l'algorisme UnionFind, i la seva utilitat per a explorar components connexes, a més d'entendre millor l'implementació d'un graf en C++. També hem après a poder practicar amb diferents models de grafs parametritzats aleatoris i la utilitat d'aquests. A més a més, hem pogut practicar amb les diferents llibreries que ofereix python per a crear gràfics, i representar-los.

Existeixen un nombre bastant elevat de tipus de graf en la llibreria networkx, alguns basats en models específics de la realitat, que podrien ser molt interessants d'experimentar-hi amb ells, tot i que molts d'aquests estan pensats per a experimentar propietats de grafs connexos.

Bibliografia

- (1)"Random graph models of social networks", M. E. J. Newman, D. J. Watts, S. H. Strogatz, 2566–2572, PNAS, February 19, 2002, vol. 99, suppl. 1
- Random Geometric Graph: https://en.wikipedia.org/wiki/Random_geometric_graph
- Erdős–Rényi model: https://en.wikipedia.org/wiki/Erd%C5%91s%E2%80%93R%C3%A9nyi_model
- Newman, Mark. E. J.; Strogatz, S. H.; Watts, D. J. (2001). "Random graphs with arbitrary degree distributions and their applications". *Physical Review E*. **64**: 026118. arXiv:cond-mat/0007235. Bibcode:2001PhRvE..64b6118N. doi:10.1103/PhysRevE.64.026118., Eq. (1)
- Penrose, Mathew. (2003). *Random geometric graphs*. Oxford: Oxford University Press. ISBN 0198506260. OCLC 51316118.
- Connectivity of the Uniform Random Intersection Graph - Simon R. Blackburn∗ and Stefanie Gerke Department of Mathematics Royal Holloway, University of London: https://arxiv.org/abs/0805.2814