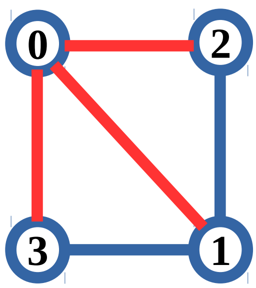
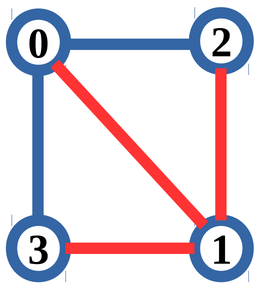
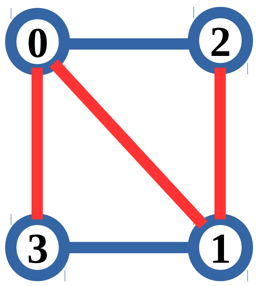
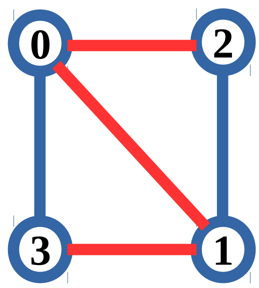
Partie 1

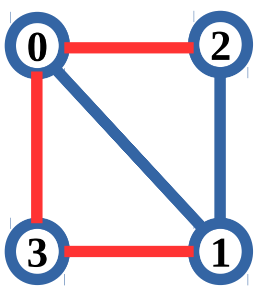
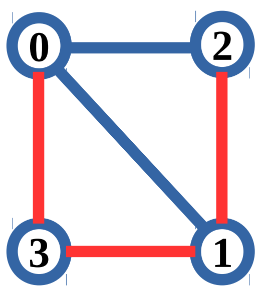
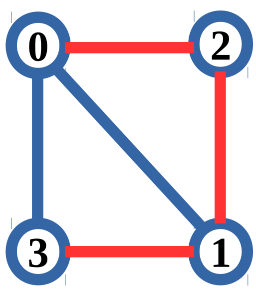
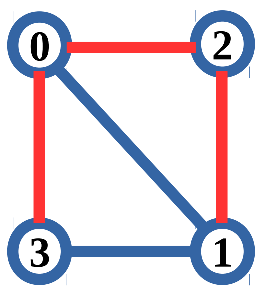
Q1)

pour le graphe G1 on a deux cas figure :

a) **si on prend l’arrête (0,1)** (l’arrête appartient a l’arbre couvrante) on va prendre avec exactement 2 arrête parmi les 4 qui reste ,tell-que ces 2 arrête sont différente a ((0,2) , (2,3)) et ((0,3) , (3,1)) (pour ne fais pas cycle) .

cas possible			
			
(0,1) ,(0,2),(0,3)	(0,1) ,(1,2),(1,3)	(0,1) ,(0,3),(1,2)	(0,1) ,(0,2),(1,3)

b) **si on prend pas l’arrête (0,1)** on va prendre exactement 3 arrête parmi les 4 qui reste pour faire une arbre couvrante .

cas possible			
			
(0,2),(0,3),(1,3)	(0,3),(1,2),(1,3)	(0,2),(1,2),(1,3)	(0,2),(0,3),(1,2)

Q3)

Les résultat de l’exécution de l'algorithme de Kruskal un million de fois sur le graphe G1 (résultat obtenu par notre class Statistique qui lance l ‘Algorithme un million de fois et calcule la probabilité de chaque arbre couvrante) :

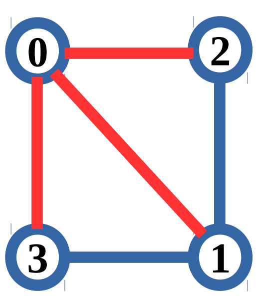
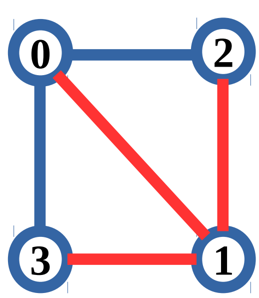
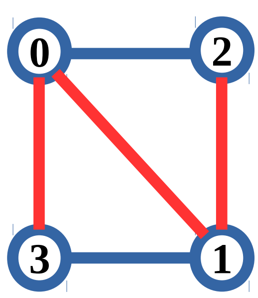
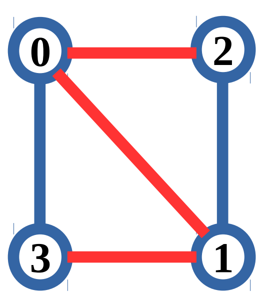
//////////////////////////// Analyse d'Algo de Kruskal ////////////////////////
les résultat de lancement d'algo 1000000 fois sur le meme graphe
temps d'execution moyenne :0.001064000000000000
la probabilité moyenne des arbre :0.125
la variance des probabilité des arbre :0.125
ecart-type des probabilité des arbre :0.3535533905932738

num	arbre	nombre d'occurence	probabilité
1)	0--2 1--3 1--2	116653	0.116653
2)	0--3 0--2 1--3	116512	0.116512
3)	1--3 0--1 1--2	133055	0.133055
4)	0--3 1--2 0--1	133221	0.133221
5)	0--2 0--3 0--1	133418	0.133418
6)	1--2 1--3 0--3	116874	0.116874
7)	0--1 0--2 1--3	133067	0.133067
8)	0--3 0--2 1--2	117200	0.1172

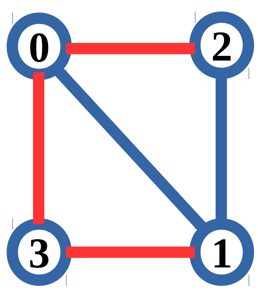
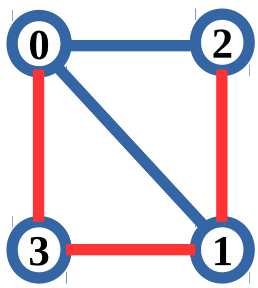
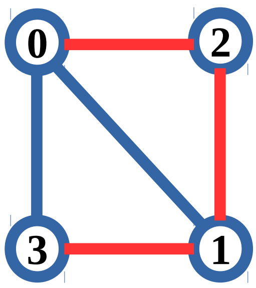
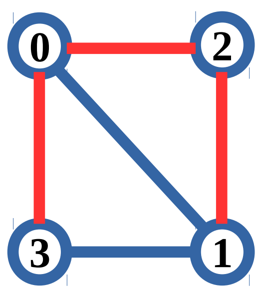
- on remarque que la probabilité d'apparaître pour les arbre qui contient l’arrête (0,1) [l’arbre 3,4,5,7] est plus élevé par rapport à les qui ne contient pas l’arrête (0,1) [l’arbre 1,2,6,8] .
-la valeur de variance des probabilité est élevé (loin de 0) car les probabilité sont loin a sa moyenne.
-alors les arbre ayons des probabilité différente d'apparaître et cette différence est remarquable (importante).

Q4) on va commencer par des définition (notation) :

A1 : ensemble (partition) des arbre qui contient l’arrête (0,1) .

			
(0,1) ,(0,2),(0,3)	(0,1) ,(1,2),(1,3)	(0,1) ,(0,3),(1,2)	(0,1) ,(0,2),(1,3

A2 : ensemble (partition) des arbre qui ne contient pas l’arrête (0,1) .

			
(0,2),(0,3),(1,3)	(0,3),(1,2),(1,3)	(0,2),(1,2),(1,3)	(0,2),(0,3),(1,2)

A : l’ensemble globale (domaine d’étude) qui contient les 8 arbre couvrante, A=A1∪A2 .

P(S):la probabilité que l’arbre couvrante calculer par Algo de Kruskal appartient a l’ensemble S .

Ord(a):
-après l’étape de mélange les arrête ayons un ordre , Ord(a) c’est l’ordre de l’arrête a (l’indice) .
-exemple :
après l’étape de mélange : [(0,3),(0,2),(1,2),(1,3)]
Ord((1,2)) = 3 , Ord(0,3) =1 , Ord ((1,3)) =4

Pr({A1,A2,….,An}):
-le nombre de permutation possible des arrête A1...An .
-exemple :
S={(0,1),(0,2)}
Pr(S)=2 !=2*1=2
on a 2 possibilité [(0,1),(0,2)] , [(0,2) , (0,1)]

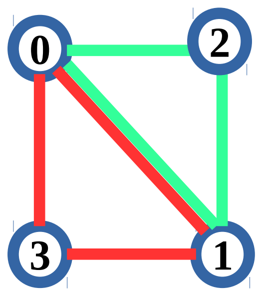
preuve :

pour prouver que les arbre couvrante n'ont pas tous la même probabilité d'apparaître il suffit de prouver que P(A1) ≠ P(A2) , car card(A1) =card(A2) =4 alors si touts les arbre ayons la même probabilité d'apparaître P(A1)=P(A2)=1/2.

A.on commence par calcule de P(A2) la probabilité que l’arbre appartient a A2 on a 3 cas :

(1) Si Ord((0,1))<3 :

algorithme de Kruskal va prendre forcément l’arrête (0,1) .
car pour que Kruskal ne prend pas (0,1) il faut que (0,1) créer un cycle avec les arête déjà choisie et pour faire un cycle il faut au moins 2 arrête avant (0,1) , soit {(0,2),(2,1)} ou {(0,3),{3,1}} avant (0,1) .alors
si Ord((0,1))<3 on a 0 chance que l’arbre appartient au A2 .



(2) Si Ord((0,1))>3 :

dans ce cas l’arbre appartient forcément a A2 ,car algorithme de Kruskal va prendre les 3 premier arrête et ne prend pas (0,1) .

explication :
pour faire un cycle avec {(0,2) ,(2,1),(1,3),(3,0)} il faut prendre les 4 arrête alors on peut pas faire cycle avec 3 .
si on prend un sous-ensemble {(0,2) ,(2,1),(1,3),(3,0)} de taille 3 alors ce sous-ensemble contient forcément soit {(0,2),(2,1)} ou {(0,3),{3,1}} et ça va crée un cycle avec (0,1) alors il est impossible que (0,1) appartient a l’arbre.

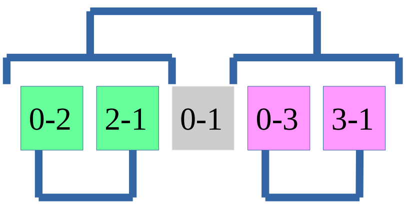
La probabilité que ce cas s’arrive :
ordre possible :
[X1 X2 X3 (0,1) X5]   , ord((0,1))=4 , probabilité =Pr(A-{(0,1)})/Pr(A)=(4 !)/(5!)=1/5
[X1 X2 X3 X5 (0,1)] , ord((0,1))=5 , probabilité =Pr(A-{(0,1)})/Pr(A)=(4 !)/(5!)=1/5
la probabilité totale que ce cas s’arrive = 1/5+1/5=2/5.

(3) Si Ord((0,1))=3 :

dans ce cas l’arbre appartient a A2 ssi les 2 arrête qui sont avant (0,1) crée un cycle avec (0,1) alors l’algorithme ne prend pas (0,1).

explication :
[X1 X2 (0,1) X3 X4]
X1 ∈ {(0,2),(2,1)} et X2∈ {(0,2),(2,1)} l’algorithme prend {(0,2),(2,1)} alors il prend pas (0,1) car il va crée un cycle .
X1 ∈ {(0,3),{3,1}}et X2∈ {(0,3),{3,1}}l’algorithme prend {(0,3),{3,1}} alors il prend pas (0,1) car il va crée un cycle .

Probabilité que l’arbre appartient au A2 dans ce cas :
[X1 X2 (0,1) X3 X4]
la probabilité que X1 ∈ {(0,2),(2,1)} et X2∈ {(0,2),(2,1)} ou X1 ∈ {(0,3),{3,1}} et X2∈ {(0,3),{3,1}}.
on va calculer cette probabilité avec les permutation :



probabilité =(Pr({(0,2),(2,1)})*Pr({(0,3),{3,1}})+Pr({(0,2),(2,1)})*Pr({(0,3),{3,1}})) /Pr(A)
=(2!.2!+2 !.2 !)/5!=8/120 ≃0.066666667.

Alors on déduit que la probabilité que une arbre appartient a A2 est :
P(A2)=2/5+8/120 =56/120.

B.on calcule P(A1) :

$P(A)=P(A1\cup A2)=P(A1)+P(A2)-P(A1\cap A2)$.

$P(A)=P(A1)+P(A2)$ (car $A1\cap A2 = \varnothing$, $P(\varnothing)=0$)
$P(A1)=P(A)-P(A2)=1-P(A2)=64/120$

On a $P(A1) = 64/120$ et $P(A2) = 56/120$, $P(A1) \neq P(A2)$ alors que les arbre couvrante n'ont pas tous la même probabilité d'apparaître_

Q5)

Les résultat de l'exécution de l'algorithme de Aldous-Broder un million de fois sur le graphe G1 (résultat obtenu par notre class Statistique qui lance l 'Algorithme un million de fois et calcule la probabilité de chaque arbre couvrante) :

//////////////////////////////// Analyse d'Algo de Aldous-Broder ///////////////////////////
les résultat de lancement d'algo 1000000 fois sur le meme graphe
temps d'execution moyenne :0.000496000000000000
la probabilité moyenne des arbre :0.12500000093132257
la variance des probabilité des arbre :0.0
ecart-type des probabilité des arbre :0.0

num	arbre	nombre d'occurence	probabilité
1)	1--3 0--1 1--2	124570	0.12457
2)	1--2 1--3 0--3	124426	0.124426
3)	0--3 1--3 0--2	125071	0.125071
4)	1--3 1--2 0--2	125282	0.125282
5)	1--2 0--1 0--3	125213	0.125213
6)	0--1 0--2 1--3	124713	0.124713
7)	0--2 1--2 0--3	125313	0.125313
8)	0--3 0--1 0--2	125412	0.125412

- on remarque que que les 8 arbres couvrants ont presque tous la même probabilité d'apparaître .
-la valeur de variance des probabilité est proche de 0 (approximativement 0) car les probabilité sont proche a sa moyenne.
-alors les arbre ayons la même probabilité d'apparaître.

Q6)

Les résultat de l’exécution de l'algorithme de Wilson un million de fois sur le graphe G1 (résultat obtenu par notre class Statistique qui lance l ‘Algorithme un million de fois et calcule la probabilité de chaque arbre couvrante) :

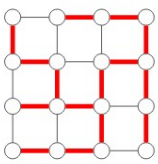
////////////////////////////// Analyse d'Algo de Wilson ////////////////////////
les résultat de lancement d'algo 1000000 fois sur le meme graphe
temps d'execution moyenne :0.000760000000000000
la probabilité moyenne des arbre :0.12499999906867743
la variance des probabilité des arbre :0.0
ecart-type des probabilité des arbre :0.0

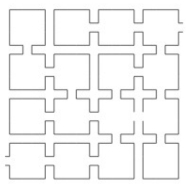
num	arbre	nombre d'occurence	probabilité
1)	0--3 0--2 1--2	126242	0.126242
2)	0--2 1--2 1--3	126105	0.126105
3)	1--2 0--1 0--3	122555	0.122555
4)	1--2 0--1 1--3	125606	0.125606
5)	0--2 1--3 0--1	122612	0.122612
6)	0--3 1--2 1--3	126251	0.126251
7)	0--3 0--2 0--1	124520	0.12452
8)	1--3 0--3 0--2	126109	0.126109

- on remarque que que les 8 arbres couvrants ont presque tous la même probabilité d'apparaître .
-la valeur de variance des probabilité est proche de 0 (approximativement 0) car les probabilité sont proche a sa moyenne.
-alors les arbre ayons la même probabilité d'apparaître.

Q7)

Le labyrinth est une grid où les arretes utilisé dans l’arbre couvrant (used == true) sont des passages et les autres arretes (used == false) sont des murs.





Pour simplier, on a defini une classe Labyrinth qui crée un labyrith a partir d’une dimension donnée et un algorithme qui calcule le passage. L’algorithme consiste seulement en deux étapes :

Laby ← Grid(20) ;
Laby.calculeChemin(Algorithme) ;

Q8)

On a exécuté l’algorithme plusieurs fois, et on a choisi deux exécution pour la comparaison.

/////////////////////////////// Labyrith par l'Algo de Kruskal ///////////////////////////

Nombre d'execution : 1000

Nombre Moyen des Culs de Sac : 119

Distance E/S Moyenne : 56.7

/////////////////////////////// Labyrith par l'Algo de Aldous-Broder ///////////////////////////

Nombre d'execution : 1000

Nombre Moyen des Culs de Sac : 114

Distance E/S Moyenne : 59.782

/////////////////////////////// Labyrith par l'Algo de Kruskal ///////////////////////////

Nombre d'execution : 1000

Nombre Moyen des Culs de Sac : 120

Distance E/S Moyenne : 56.542

/////////////////////////////// Labyrith par l'Algo de Aldous-Broder ///////////////////////////

Nombre d'execution : 1000

Nombre Moyen des Culs de Sac : 114

Distance E/S Moyenne : 59.522

- On observant les resultats fournis par les deux algorithme, on remarque :

- L’algorithme d’Aldous Broder génère moins de culs de sac que l’algorithme de Kruskal.

- Par contre l’algorithme de Kruskal génère des passages plus courts, en moyenne, que ceux générés par l’algorithme d’Aldous Broder