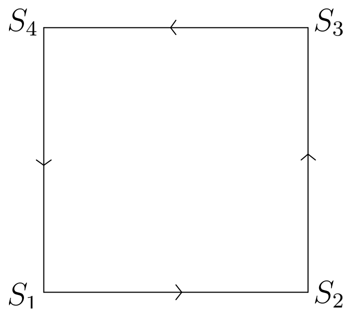
⧫ TP11 – Un exemple de marche aléatoire

I. — Présentation du problème

On considère le jeu suivant. On dispose de deux urnes U_1 et U_2. L'urne U_1 contient 9 boules noires et 1 boule blanche et l'urne U_2 contient 2 boules noires et 3 boules blanches. Initialement, on dispose un pion sur le sommet S_1 d'un carré. On tire une boule dans l'urne U_1. Si cette boule est blanche, on déplace le pion au sommet suivant dans le sens des flèches : il se retrouve donc sur S_2. Sinon, on laisse le pion dans sa position : il reste donc sur S_1. Ensuite, on répète le processus de la manière suivante. Lorsqu'on est sur un sommet S_i, on tire une boule dans l'urne U_1 si $i = 1$ ou $i = 4$ et dans l'urne U_2 si $i = 2$ ou $i = 3$. Si cette boule est blanche, on déplace le pion sur le sommet suivant dans le sens des flèches et, sinon, on laisse le pion sur le sommet S_i.



Cette situation où on étudie le mouvement d'un objet qui évolue de façon aléatoire s'appelle une **marche aléatoire**.

Pour tout entier $n \in \mathbb{N}$, on note A_n l'évènement : « après le $n-$ième lancer, le pion se trouve sur le sommet S_1 ». On définit de même les évènements B_n, C_n et D_n respectivement pour les sommets S_2, S_3 et S_4.

On note, pour tout $n \in \mathbb{N}$, a_n, b_n, c_n et d_n les probabilités respectives des évènements A_n, B_n, C_n et D_n.

Question 1. Déterminer les valeurs de a_0, b_0, c_0 et d_0.

Question 2. Déterminer a_1, b_1, c_1 et d_1 puis utiliser la formule de probabilités totales pour calculer a_2, b_2, c_2 et d_2.

Le but de ce qui suit est de déterminer, lorsque n devient grand, la probabilité de chacun des évènements A_n, B_n, C_n et D_n i.e. de déterminer, si elles existent, les limites des suites (a_n), (b_n), (c_n) et (d_n).

II. — Simulation

Question 3. Écrire une fonction `urne1` sans paramètre qui simule un tirage dans l'urne U_1 et qui renvoie 1 si on tire une boule blanche et 0 sinon. On utilisera la fonction `randint` du module `random`.

Question 4. Écrire une fonction `urne2` sans paramètre qui simule un tirage dans l'urne U_2 et qui renvoie 1 si on tire une boule blanche et 0 sinon.

Question 5. Écrire une fonction `marche` qui prend en argument un entier naturel `n` non nul, qui simule `n` déplacements du pion et qui renvoie le numéro du sommet où se trouve le pion à l'issue des `n` déplacements.

Question 6. Écrire une fonction `simulation` qui prend en arguments un entier naturel `n` non nul et un entier naturel `N` non nul, qui simule `N` marches aléatoires de `n` déplacements à l'aide de la fonction `marche` et qui renvoie, sous forme de liste, les fréquences de réalisation des évènements A_n, B_n et C_n et D_n.

Par exemple, si en appelant `simulation(100,1000)`, la fonction `marche(1000)` a renvoyé 40 fois 1, 6 fois 2, 5 fois 3 et 49 fois 4 alors la fonction `simulation` doit renvoyer la liste

```
[0.4, 0.06, 0.05, 0.49].
```

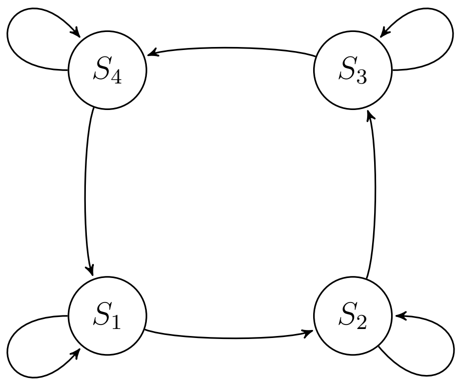
Question 7. En prenant différentes valeurs de `n` et de `N`, conjecturer le comportement asymptotique des suites (a_n), (b_n), (c_n) et (d_n).

III. — Étude mathématique à l'aide de matrices

Question 8. Soit $n \in \mathbb{N}$. Déterminer la probabilité de A_{n+1} sachant A_n, la probabilité de B_{n+1} sachant A_n, la probabilité de C_{n+1} sachant A_n et la probabilité de D_{n+1} sachant A_n.

Ces probabilités s'appellent les **probabilités de transition** de l'état A_n aux états A_{n+1}, B_{n+1}, C_{n+1} et D_{n+1}. On remarque que ces probabilités ne dépendent pas de n. On les appellera aussi les probabilités de transition d'un sommet à un autre.

Question 9. Sur le graphe ci-contre, indiquer à côté des arcs les probabilités de transition entre les différents sommets. Ce graphe pondéré s'appelle un **graphe probabiliste**.



Question 10. On note M la matrice d'adjacence de ce graphe pondéré. Déterminer M. La matrice M s'appelle la **matrice de transition** associée à la marche aléatoire.

On définit, pour tout $n \in \mathbb{N}$, la matrice ligne $X_n = \begin{pmatrix} a_n & b_n & c_n & d_n \end{pmatrix}$. La matrice X_n s'appelle l'**état probabiliste** après n lancers.

Question 11. Déterminer la matrice X_0 et justifier que, pour tout $n \in \mathbb{N}$, $X_{n+1} = X_n \times M$.

Question 12. Démontrer par récurrence que, pour tout $n \in \mathbb{N}$, $X_n = X_0 M^n$.

IV. — Détermination du comportement asymptotique à l'aide de Python

1) Conjecture à l'aide du calcul matriciel

On a vu dans le TP4 sur le pivot de Gauss qu'on peut représenter une matrice en Python par une liste de listes. Cette représentation, qui se prête bien à la programmation de l'algorithme du pivot, n'est en revanche pas adaptée lorsqu'on veut effectuer des calculs sur les matrices.

Dans la suite, on va utiliser le module `numpy` qui permet d'effectuer simplement des calculs sur les matrices. Pour cela, on commence par importer ce module en le renommant `np` à l'aide de l'instruction

```python
import numpy as np
```

Une fois le module importé, on peut :

- définir une matrice à l'aide de la fonction `array`.
- additionner et soustraire des matrices à l'aide des opérateurs `+` et `-` ;
- multiplier des matrices à l'aide de l'opérateur `@` (attention, l'opérateur `*` ne correspond pas au produit matriciel) ;
- calculer une puissance d'une matrice carrée à l'aide de la méthode `linalg.matrix_power()` ;
- calculer l'inverse d'une matrice carrée inversible à l'aide de la méthode `linalg.inv()` ;

Par exemple, pour définir les matrices $A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$, $B = \begin{pmatrix} 5 \\ 6 \end{pmatrix}$ et $C = \begin{pmatrix} 7 & 8 \end{pmatrix}$ puis calculer $D = AB$, $E = CA$ et $F = A^4$ et $G = A^{-1}$, on écrit

```python
A=np.array([[1,2],[3,4]])
B=np.array([[5],[6]])
C=np.array([[7,8]])
D=A@B
E=C@A
F=np.linalg.matrix_power(A,4)
G=np.linalg.inv(A)
```

Question 13. Avec les notations de la partie III, définir les matrices X_0 (qu'on appellera simplement `X`) et M.

Question 14. Déterminer des valeurs approchées de a_{10}, b_{10}, c_{10} et d_{10}.

Question 15. Calculer a_n, b_n, c_n et d_n pour de « grandes valeurs » de n et conjecturer la limite de chacune des ces suites. Ces conjectures sont-elles en accord avec celles de la question 7 ?

2) Démonstration des conjectures

Le but de cette partie est de démontrer les conjectures précédentes. Pour cela, on va diagonaliser la matrice M en utilisant les possibilités du module `numpy` qui permet de déterminer les valeurs propres d'une matrice, ainsi que des vecteurs propres associés, à l'aide de la méthode `linalg.eig()`. On ajoutera le code suivant pour obtenir les valeurs exactes des coefficients sous forme de fractions.

```python
import fractions
np.set_printoptions(formatter={'all':lambda x:
    str(fractions.Fraction(x).limit_denominator()
    )})
```

Question 16. Déterminer, à l'aide de `numpy`, les valeurs propres de la matrice M et des vecteurs propres associés à chacune de ces valeurs propres. Pourquoi peut-on affirmer que M est diagonalisable ?

Question 17. En déduire une matrice diagonale D et une matrice inversible P telles que $M = PDP^{-1}$.

Question 18. À l'aide de `numpy`, déterminer P^{-1}.

Question 19. Utiliser la question 14 pour calculer, à la main, pour tout $n \in \mathbb{N}$, la matrice X_n.

Question 20. Démontrer les conjectures précédentes.