Réduction de graphes markoviens

Julien Collet,

6 juin 2013

Résumé

A la croisée des domaines des probabilités discrètes et de la théorie des graphes, les processus markoviens sont des cas probabilistes que l'on rencontre souvent. C'est dans le cadre de l'analyse linguistique (succession de lettres dans un texte, etc...) que cette branche des probabilités a émergée. Il n'est pas rare de faire apparaitre ces chaînes sous formes de graphes d'états. Or, il apparait parfois que certains états pourtant clairement désolidarisés dans un graphe jouent des rôles similaires. Serait-il alors possible de les grouper ? Et — le cas échéant — sur quels critères ? Existe-t-il une manière algorithmique de procéder et quand serait-elle pertinente ?

1 Des probabilités aux processus Markoviens

Un processus markovien — ou encore, processus sans mémoire — est une séquence de variables aléatoires ayant une propriété fondamentale : la prédiction de leurs états futurs n'est pas rendue plus précise par la connaissance du passé. C'est à dire que la prédiction la plus précise pouvant être faite est conditionnée par l'état présent uniquement :

$$p(X_{n+1} = x | X_0, X_1, \dots, X_n) = p(X_{n+1} = x | X_n) \quad (1)$$

De nombreuses situations réelles peuvent être qualifiées de processus Markoviens : *Marche au hasard*, *Mouvement brownien*, Pile ou Face, etc... Par ailleurs, une epreuve de Bernoulli — 2 issues possibles de probabilités respectives p et $1 - p$ — est un cas particulier de processus Markoviens où la prédiction de l'état suivant est même indépendante de l'état présent !

Une situation probabiliste peut être modélisée sous forme d'un graphe d'état et d'une matrice stochastique. Nous verrons par suite que ces matrices donnent des informations tout à fait pertinentes sur le processus considéré : nombre de coups moyen pour arriver à l'état absorbant depuis un état quelconque, classes d'équivalences, etc... Nous verrons également comment, à partir de telles matrices, il est parfois possible de réduire ces graphes.

2 L'idée

2.1 D'un lancer de dé aux graphes de Markov

Pour contextualiser le problème, prenons l'exemple d'un jeu très simple. On réalise des lancers de pièces jusqu'à obtenir la séquence *Pile-Pile-Face-Face* que l'on notera [0011]. Par exemple : si, à un lancer n, on a [1001], au lancer suivant, on aura soit [0011] (cas gagnant), soit [0010]. Nous n'utiliserons pour cela que des pièces honnêtes, s'assurant ainsi de l'équiprobabilité des évènements *« obtenir l'une ou l'autre des faces »* (tirage de Bernoulli de paramètre 0.5).

2.2 Méthode intuitive

A première vue, il est naturel d'imaginer un graphe markovien contenant $16 + 1$ états (16 états possibles plus l'état initial). On note M, la matrice de Markov associée et G_M, le graphe-candidat (figure 1).

Rappelons ici une méthode de calcul du nombre moyen de coups pour atteindre l'état final — pour gagner donc.

$$M = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 \\
1/16 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix} \tag{2}$$

Soit M, une matrice de Markov telle que :

$$M = \left[\begin{array}{c|c} A & 0 \\ \hline B & I \end{array}\right]$$

I est la matrice identité appropriée. A est une matrice carrée, bloc de la matrice de Markov. On voit également ici que l'on s'est arrangé pour faire apparaitre les états absorbants à la fin de la matrice de Markov. Calculons maintenant M^q avec $q \to \infty$:

$$M^q = \left[\begin{array}{c|c} 0 & 0 \\ \hline B.\sum_{q=0}^{q-1} A^k & I \end{array}\right] \overset{q \to \infty}{\to} \left[\begin{array}{c|c} 0 & 0 \\ \hline B.C & I \end{array}\right]$$

En fait, élever à la puissance q la matrice M revient à trouver la matrice de Markov du système après q itérations. En effet, si l'on pose V_0 (resp. V^n) le vecteur représentant l'état du système initial (resp. après n itérations) :

$$\begin{aligned} V_1 &= M \times V_0 \\ V_2 &= M \times V_1 = M^2 \times V_0 \\ V_n &= M^n \times V_0 \end{aligned}$$

D'après [1], $C = (I-1)^{-1}$. Ce vecteur C nous permet de connaitre le nombre de coups moyen pour arriver au but quelque soit l'état de départ. Etant donné que l'on gère le cas de démarrage, seule la première valeur de ce vecteur nous interessera. Ce vecteur nous sera très interessant par la suite. Passons maintenant à la méthode qui nous intéresse.

2.3 Méthode succès/échec

Dans ce cas très particulier, il est possible de voir différemment la situation. Plutôt que de réfléchir en termes d'états, voyons les choses sous l'angle de la *distance* à la séquence absorbante. Pour illustrer ceci, détaillons quelques peu les cas. La probabilité de transition de l'état de démarrage[1]. (*Start*) vers l'état x sera notée $p_{s \to x}$. Partons de l'état final :

- **Etat 5** : Séquence valide, état courant : 0011. On ne peut que rester dans cet état (état absorbant). $p_{s \to 5}$: 1/16. Il n'y a qu'une chance sur les 16 états d'obtenir cela au départ.

- **Etat 4** : Séquence $\frac{3}{4}$-valide, état courant X001. Dans cet état, soit on se dirige vers l'état final, soit l'on retourne un autre état. $p_{s \to 4}$: 2/16. Seul deux états correspondent : 0001 et 1001.

- **Etat 3** : Séquence $\frac{2}{4}$-valide, état courant XX00. Ici, soit l'on reboucle sur cet état avec un 0, soit l'on va vers l'état suivant (4). $p_{s \to 3}$: 4/16. Il n'existe que 4 états présentant ce schéma.

- **Etat 2** : Séquence $\frac{1}{4}$-valide, état courant XXX0. Soit l'on continue sur notre lancée avec un 0, soit l'on retourne sur l'état précèdent. $p_{s \to 2}$: 4/16. Sur tous les états restants, 4 se terminent par un 0.

- **Etat 1** : Séquence invalide, état courant quelconque. $p_{s \to 1}$: $1 - \frac{11}{16}$. Ce sont tous les autres états.

1. En fait, 4 lancers sont effectués dans ce lancer afin de se placer directement dans un cas avec 4 résultats en stock

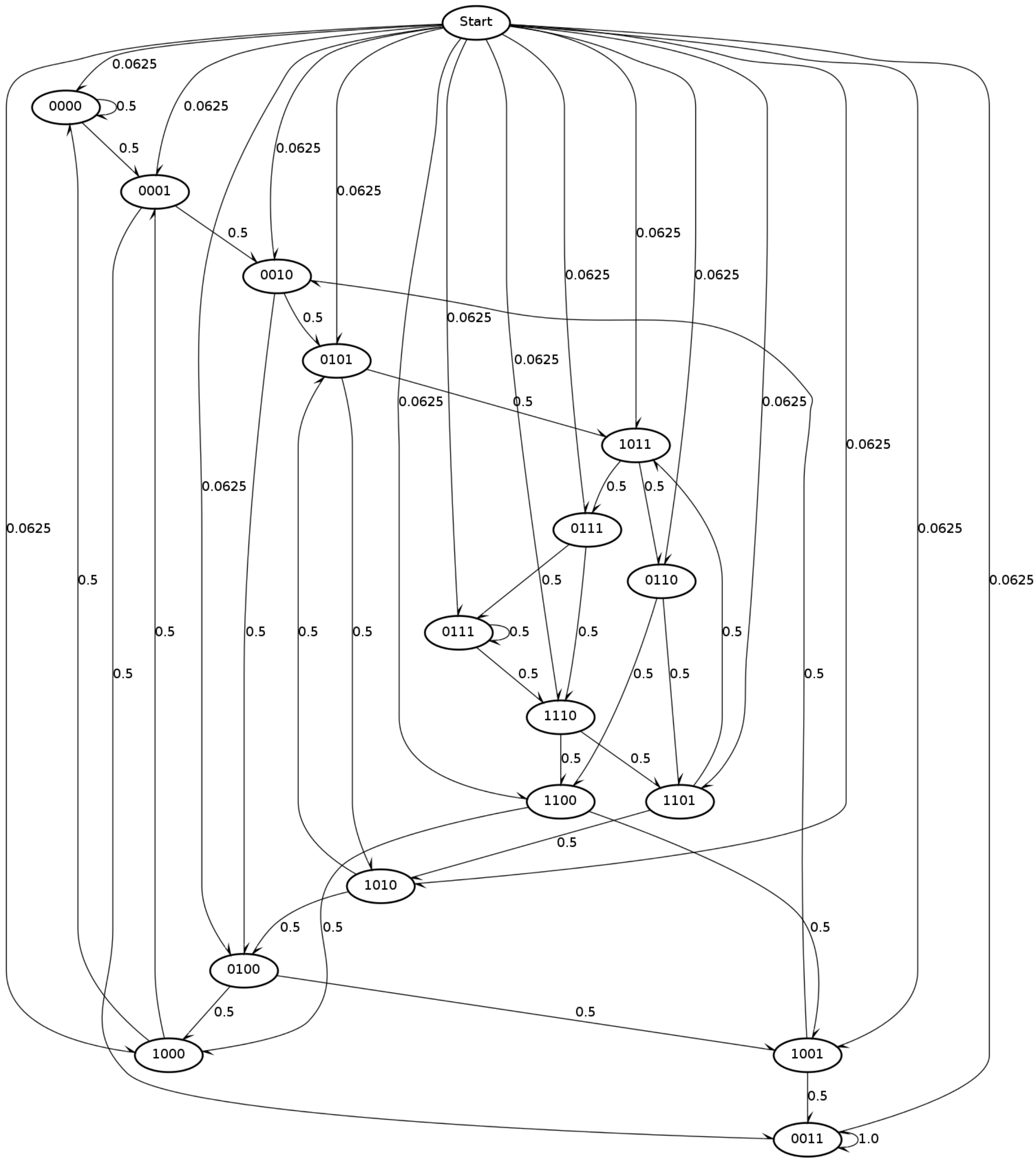

Figure 1 — Graphe trivial

Voici le graphe correspondant à la méthode empirique :

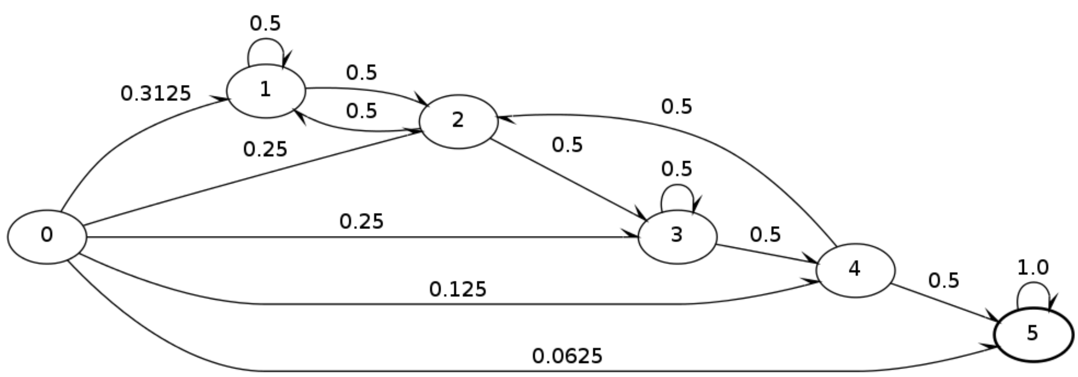

Figure 2 — Graphe Simplifié

On construit la matrice d'adjacence associée :

$$M = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 5/16 & 1/2 & 1/2 & 0 & 0 & 0 \\ 4/16 & 1/2 & 0 & 0 & 1/2 & 0 \\ 4/16 & 0 & 1/2 & 1/2 & 0 & 0 \\ 2/16 & 0 & 0 & 1/2 & 0 & 0 \\ 1/16 & 0 & 0 & 0 & 1/2 & 1 \end{bmatrix} \tag{3}$$

3 Experimentation

3.1 Lien entre les deux graphes ?

On prétend que ces deux graphes sont plus que simplement très proches du même processus : ils en sont les représentations effectives, à divers degrés d'optimalité.

3.1.1 Spectre

Premier point troublant, il a été constaté que ces deux matrices ont un spectre identique. Après une rapide implémentation matlab et grâce à l'utilisation de la fonction idoine, voici les spectres des matrices considérées (où `Piece17`, resp. `Piece6` représente la matrice triviale, resp. optimale) :

```
>> eig(Piece17)

ans =
  1.000  0.9196  0.5000  -0.2098+0.3031i...
  ... -0.2098-0.3031i 0 0 0 0 0 0 0 0 0 0 0 0

>> eig(Piece6)

ans =
  1.000 0.9196  0.5000  -0.2098+0.3031i...
  ... -0.2098-0.3031i 0
```

Il est à noter que, les *zéros* sont de *vrais* zéros, et non des approximations. Une hypothèse se profile : est-ce que les zéros du spectre, indiquant une dégénérescence du rang de la matrice triviale, peuvent être un indicateur de la non-optimalité de la représentation ? Autre question : quelle peut être la signification profonde du spectre de telles matrices ? Il doit être fort pour que de telles similitudes apparaissent malgré les différences entre ces deux graphes.

3.1.2 Réflexion sur les groupements/classes d'états

On a vu que certains états pouvaient être groupés. Essayons de construire cette liste de classes :

- Classe 0 : $\{3\}$
- Classe 1 : $\{1, 9\}$
- Classe 2 : $\{2, 6, 10, 14\}$
- Classe 3 : $\{5, 7, 11, 13, 15\}$
- Classe 4 : $\{0, 4, 8, 12\}$
- Classe 5 [2] : $\{Start\}$

Il est aisé de construire ces groupements de manière empirique. Partons de l'état final : 3. Dans notre cas très *binaire*, cela correspond à 0011. Or, il ne peut y avoir que deux antécèdents : 1001 et 0001 soit respectivement 9 et 1. En remontant ainsi, et en ne tenant compte que des états n'appartenant à aucune autre classe, on peut construire ces classes en *remontant* les antécèdents. Essayons de trouver une explication plus rationelle.

3.2 Classes de coups moyens

3.2.1 Experimentations

On prétend maintenant que ces groupements ne sont pas seulement le fruit d'une observation empirique. [1] nous explicite la façon de parvenir — par des biais calculatoires — au nombre de coups moyen pour arriver à l'état absorbant depuis un état quelconque. Effectuons le calcul grâce à notre outil favori — matlab, pour ne pas le citer — sur les deux matrices [3] :

```
>> V = averageStepComputing(Piece17)

V =
  13 10 8 14 10 16 14 16 10 8 14 16 10 16 ...
  ... 14 16

>> V = averageStepComputing(Piece6)

V =
  13 16 14 10 8
```

3.2.2 Commentaires

La i-ème valeur de V contient le nombre de coups moyen pour aller à l'état absorbant, depuis i ($i \in \{Start, 0, 1, \ldots, 15, 3\}$).

Première remarque, la première valeur des deux vecteurs est identique. C'est rassurant : 13 représentant le nombre de coups moyen depuis l'état $Start$ — le nombre de coup moyen tout court donc — il est donc intéressant de retrouver cette même valeur pour les deux modèles.

Deuxièmement, ces vecteurs sont de tailles respectives 16 et 5. En effet, on ne calcule pas le nombre de coups pour aller à l'état absorbant depuis cet état... Ceci amène à une autre observation : on remarque, pour les deux vecteurs, que l'on obtient le même nombre de classes (au sens statistique), c'est à dire 5, à laquelle

2. Ne contenant que l'état initial, la question de sa considération comme un état — et donc une classe à part entière — reste encore à discuter.

3. le script effectuant ce calcul est implémenté dans *averageStepComputing.m*

il conviendra d'ajouter la classe contenant l'état absorbant : On obtient 6 classes au total, ce qui vient en accord avec la proposition concernant les groupements d'états (voir 2.4.2. Réflexion sur les groupements/classes d'états).

Cette dernière observation mérite que l'on s'y attarde. En effet, si l'on groupe les états par classes de nombre de coups moyen, on retombe sur la même classification d'états que celle établie empiriquement.

3.3 Conclusion préliminaire

Les résultats expérimentaux concernant la similarité du spectre de graphes différents mais modélisant les mêmes processus est troublant. Cela l'est d'autant plus que les classes obtenues sont validées par un modèle théorique et que l'on connait — a priori — le critère de regroupement : le nombre de coups moyen. En s'inspirant de la méthode utilisée pour trouver empiriquement les classes d'états, un algorithme a été implémenté pour la réduction d'un graphe non-optimal. L'implémentation sera succinctement décrite et discutée dans la suite.

4 Algorithme, Python et Implémentation

4.1 Présentation de l'algorithme

L'algorithme en lui même n'est pas d'une grande complexité : il s'agit d'identifier le ou les états terminaux et de les grouper si possible. Lorsque l'on parlera de grouper des états, cela signifiera effectuer deux actions : premièrement, on fusionne les colonnes (ce qui revient à faire une moyenne pondérée, termes à termes, des colonnes concernées) et on sommes les lignes correspondantes dans la matrice : on conserve ainsi les propriétés d'une matrice stochastique comme par exemple le fait que la somme par colonne soit égale à 1.

Une fois l'état (où la classe d'états) absorbant identifié, on cherche ses antécédents que l'on groupe ensuite par probabilités de transition. Chaque état identifié dans une classe est marqué. De sorte que l'on ne puisse pas avoir un état dans 2 classes différentes. Une fois que tous les états sont marqués, on considère que l'algorithme a convergé.

Algorithm 1: Markovian-graph optimisation

Data:
Graphe $G(V, S)$
Classe Temporaire T
Etat Courant E

Recherche de la classe absorbante :
$T \leftarrow \varnothing$
$E \leftarrow \varnothing$
for $\forall s \in S$
do
  if $\exists \nu_{s,s}$ $t.q.$ $p_{s\to s} = 1$
  then
    $T \leftarrow T \cup \{s\}$
    $S \leftarrow S \setminus \{s\}$
    $V \leftarrow V \setminus \{\nu_{s,s}\}$
  end
end
$E \leftarrow T$
$V \leftarrow V \cup \{\nu_{T,T}\}$
$p_{T\to T} \leftarrow 1$
$S \leftarrow S \cup T$
$T \leftarrow \varnothing$

Recherche des antécédents de la classe absorbante :
while $S \neq \varnothing$
do
  for $\forall s \in S$
  do
    if $\exists \nu_{s,E}$ $t.q.$ $(p_{s\to E} \neq 0 \wedge s \neq T)$
    then
      $T \leftarrow T \cup \{s\}$
      $S \leftarrow S \setminus \{s\}$
      $V \leftarrow V \setminus \{\nu_{s,E}\}$
    end
  end
  $V \leftarrow V \cup \{\nu_{T\to E}\}$
  $p_{T\leftarrow E} \to \frac{\sum p_{(s\in T)\to E}}{Card(T)}$
  $S \leftarrow S \cup T$
  $E \leftarrow T$
  $T \leftarrow \varnothing$
end

4.2 Sur un exemple

Déroulons l'algorithme sur l'exemple pris précèdemment avec la matrice M : On trouve trois antécédents à l'état final (colonne 16) : 3 ($p = 1$), 1 et 9 ($p = 0.5$). On les regroupe par probabilités de transition (fusion des colonnes puis sommation des lignes) :

Itération 1 (état courant $\{1, 9\}$) :

$$
fusion \rightarrow \quad \begin{array}{c} Start \\ 0 \\ 1 \\ 2 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \\ 10 \\ 11 \\ 12 \\ 13 \\ 14 \\ 15 \\ 3 \end{array} : \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{0.5+0.5}{2} & 0 \\
1/16 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{0.5+0.5}{2} & 1
\end{bmatrix}
$$

états : $Start \quad 0 \quad 2 \quad 4 \quad 5 \quad 6 \quad 7 \quad 8 \quad 10 \quad 11 \quad 12 \quad 13 \quad 14 \quad 15 \quad \{1, 9\} \quad 3$

$$
sommation \rightarrow \quad \begin{array}{c} Start \\ 0 \\ 2 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 10 \\ 11 \\ 12 \\ 13 \\ 14 \\ 15 \\ \{1, 9\} \\ 3 \end{array} : \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 \\
1/16 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 \\
1/16 & 0.5 & 0 & 0.5 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 1
\end{bmatrix}
$$

états : $Start \quad 0 \quad 2 \quad 4 \quad 5 \quad 6 \quad 7 \quad 8 \quad 10 \quad 11 \quad 12 \quad 13 \quad 14 \quad 15 \quad \{1, 9\} \quad 3$

La classe $\{1, 9\}$ à des antécédents qu'il est possible de grouper de par leur probabilité de transition vers cette classe, on prend cette nouvelle classe comme état courant de référence.

Itération 2 (état courant $\{0, 4, 8, 12\}$) :

$$
fusion \rightarrow \quad \begin{array}{c} Start \\ 0 \\ 2 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 10 \\ 11 \\ 12 \\ 13 \\ 14 \\ 15 \\ \{1, 9\} \\ 3 \end{array} : \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.25 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 \\
1/16 & 0.5 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0.5 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.25 & 0 & 0 \\
1/16 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & & 0.5 & 1
\end{bmatrix}
$$

états : $Start \quad 2 \quad 5 \quad 6 \quad 7 \quad 10 \quad 11 \quad 13 \quad 14 \quad 15 \quad \{0, 4, 8, 12\} \quad \{1, 9\} \quad 3$

$$
sommation \rightarrow \quad \begin{array}{c} Start \\ 2 \\ 5 \\ 6 \\ 7 \\ 10 \\ 11 \\ 13 \\ 14 \\ 15 \\ \{0, 4, 8, 12\} \\ \{1, 9\} \\ 3 \end{array} : \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 \\
1/16 & 0.5 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 \\
1/16 & 0.5 & 0 & 0.5 & 0 & 0.5 & 0 & 0 & 0.5 & 0 & 0.5 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 1
\end{bmatrix}
$$

états : $Start \quad 2 \quad 5 \quad 6 \quad 7 \quad 10 \quad 11 \quad 13 \quad 14 \quad 15 \quad \{0, 4, 8, 12\} \quad \{1, 9\} \quad 3$

Les antécédents de cette classe sont : 2, 6, 10, 14 et la classe elle-même. Les probabilités de transitions sont égales, on peut les grouper dans une nouvelle classe (exception faite de la classe existante).

Itération 3 (état courant $\{2, 6, 10, 14\}$) :

$$
fusion \quad \rightarrow \quad
\begin{array}{c}
Start \\ 2 \\ 5 \\ 6 \\ 7 \\ 10 \\ 11 \\ 13 \\ 14 \\ 15 \\ \{0,4,8,12\} \\ \{1,9\} \\ 3
\end{array}
:
\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0 & 0.25 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0.5 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0.5 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0 & 0.25 & 0 & 0 & 0 \\
1/16 & 0 & 0.5 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0.5 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0.5 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 1
\end{bmatrix}
$$

$$
\text{états :} \quad Start \quad 5 \quad 7 \quad 11 \quad 13 \quad 15 \quad \{2,6,10,14\} \quad \{0,4,8,12\} \quad \{1,9\} \quad 3
$$

$$
sommation \quad \rightarrow \quad
\begin{array}{c}
Start \\ 5 \\ 7 \\ 11 \\ 13 \\ 15 \\ \{2,6,10,14\} \\ \{0,4,8,12\} \\ \{1,9\} \\ 3
\end{array}
:
\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0 & 0.25 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0.5 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0 & 0.25 & 0 & 0 & 0 \\
1/16 & 0 & 0.5 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 \\
1/16 & 0.5 & 0.5 & 0.5 & 0.5 & 0.5 & 0 & 0 & 0.5 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0.5 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 1
\end{bmatrix}
$$

$$
\text{états :} \quad Start \quad 5 \quad 7 \quad 11 \quad 13 \quad 15 \quad \{2,6,10,14\} \quad \{0,4,8,12\} \quad \{1,9\} \quad 3
$$

Encore une étape passée. Cette fois-ci, les antécédents de la classe considérée sont : 5, 7, 11, 13, 15 et la classe référencée $\{1, 9\}$. On groupe les états non-marqués :

Itération 4 (état courant $\{5, 7, 11, 13, 15\}$) :

$$
fusion \quad \rightarrow \quad
\begin{array}{c}
Start \\ 5 \\ 7 \\ 11 \\ 13 \\ 15 \\ \{2,6,10,14\} \\ \{0,4,8,12\} \\ \{1,9\} \\ 3
\end{array}
:
\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0.25 & 0 & 0 & 0 \\
1/16 & 0.1 & 0 & 0 & 0 & 0 \\
1/16 & 0.2 & 0 & 0 & 0 & 0 \\
1/16 & 0 & 0.25 & 0 & 0 & 0 \\
1/16 & 0.2 & 0 & 0 & 0 & 0 \\
1/16 & 0.5 & 0 & 0 & 0.5 & 0 \\
1/16 & 0 & 0.5 & 0.5 & 0 & 0 \\
1/16 & 0 & 0 & 0.5 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0.5 & 1
\end{bmatrix}
$$

$$
\text{états :} \quad Start \quad \{5,7,11,13,15\} \quad \{2,6,10,14\} \quad \{0,4,8,12\} \quad \{1,9\} \quad 3
$$

$$
sommation \quad \rightarrow \quad
\begin{array}{c}
Start \\ \{5,7,11,13,5\} \\ \{2,6,10,14\} \\ \{0,4,8,12\} \\ \{1,9\} \\ 3
\end{array}
:
\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 \\
1/16 & 0.5 & 0.5 & 0 & 0 & 0 \\
1/16 & 0.5 & 0 & 0 & 0.5 & 0 \\
1/16 & 0 & 0.5 & 0.5 & 0 & 0 \\
1/16 & 0 & 0 & 0.5 & 0 & 0 \\
1/16 & 0 & 0 & 0 & 0.5 & 1
\end{bmatrix}
$$

$$
\text{états :} \quad Start \quad \{5,7,11,13,15\} \quad \{2,6,10,14\} \quad \{0,4,8,12\} \quad \{1,9\} \quad 3
$$

A la fin de cette itération, on remarque que tous les antécédents possibles de l'état courant sont déjà marqués. L'algorithme a convergé : on retombe sur la matrice obtenue précédemment. On a donc réussi à établir un lien algorithmique entre les deux matrices-graphes.

4.3 Discussion de l'implémentation

Une implémentation réalisée en `Python` a été proposée car ce langage permet de se focaliser sur l'algorithme plutôt que sur des détails de programmation qui seront plutôt traités lorsqu'une implémentation *sérieuse* — dans un langage comme le C par exemple — sera effectuée. Toutefois, le script offre une sortie au format graphViz (langage dot). On peut voir en figure 4, la sortie lorsque l'on passe en paramètre notre matrice exemple : on tombe sur les résultats trouvés précèdemment. Le script décrit notamment les équivalences classes/états avant de terminer son execution, en voici la trace pour notre exemple :

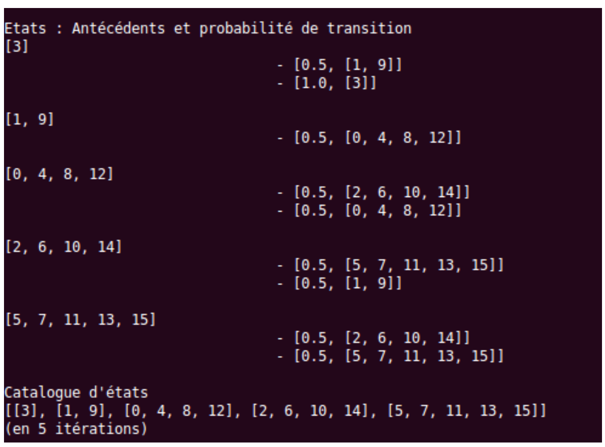

Figure 3 — Capture d'écran

4.4 Critiques

On l'a vu, l'algorithme permet finalement d'obtenir les classes que l'on obtiendrait également par calcul matriciel. Il n'est donc pas exclu de se demander si un tel algorithme est *utile*.

En pratique, pour des cas comportant peu de sommets, il parait évident que le calcul matriciel est plus simple à mettre en place. Et encore, l'obtention du vecteur des coups moyens requiert une inversion de matrice — au milieu d'autres opérations — ce qui n'est pas anodin en termes de calcul. L'algorithme présente lui l'avantage de ne comporter que des opérations élémentaires ce qui peut le rendre plus performant pour un plus grand nombre d'états.

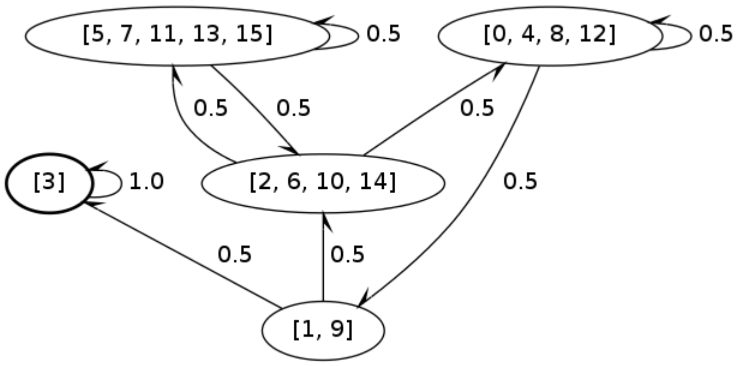

Figure 4 — Graphe de sortie

5 Conclusion et perspectives

L'algorithme demandera des confrontations expérimentales avec la méthode plus directe afin d'en valider les gains. L'outil actuellement implémenté permet en tout cas pour l'heure de générer une sortie *graphique* en langage `dot` qui permet la génération de graphes avec graphViz.

Le sens des valeurs propres de ces matrices markoviennes reste encore à découvrir. C'est probablement l'axe de travail le plus dense. Et intéressant.

Cet algorithme peut trouver des applications dans plusieurs domaines. Tout d'abord, dans la modélisation : si l'on peut modéliser un phénomène markovien en minimisant le nombre d'états, il y a fort à parier que le modèle n'en sera que plus léger, rapide et facile à utiliser.

Dans un tout autre domaine, en synthèse de systèmes numériques, on a vu qu'il est possible de modéliser certains systèmes sous forme de graphes/machines d'états très semblables aux graphes markoviens qui nous rassemblent aujourd'hui. Il n'est pas exclu que cette *optimisation* puisse être appliquée dans certains cas — ce fut d'ailleurs le cas pour un exemple où cela a permis un gain en nombre de bascules utilisées.

De plus, le cas échéant, pouvoir garantir l'optimalité d'une solution peut être quelque chose de remarquable et recherché. Pour ce cas très particulier, il conviendra cependant d'être vigilant, le raccourci probabilité-machine d'état peut ne pas être si trivial qu'il n'y parait. En effet, on a pas directement affaire à des probabilités, mais a des équations de combinatoires. Cependant, il n'est pas exclu — puisque cela a fonctionné pour un exemple — que l'on puisse étendre cette logique de simplification à ce domaine d'application.

De même, cet outil pourrait éventuellement être utilisé dans le cadre de l'analyse linguistique (détermination de redondances, simplification de modèles de langues). On sait également que certaines techniques de décodages utilisent des chaînes de Markov pour déterminer la langue d'origine d'un texte encodé. Cette simplification pourrait peut-être permettre de d'affiner ou d'optimiser certains modèles employés.

Table des matières

Références

[1] **J.-M. Becker**, *Module MSP3 - Support théorique (Chaines de Markov), TP2* - 3ETI, CPE Lyon, 2012