Vérification Probabiliste et Approximation

Richard Lassaigne <lassaign@logique.jussieu.fr>

M2 LMFI (Université Paris-Diderot)

Draft

2e Semestre 2007-2008

Table des matières

Chapitre 1

Algorithmes probabilistes et complexité

1.1 Algorithmes Las Vegas et Monte-Carlo

1.1.1 Algorithmes Las Vegas

La classe des algorithmes Las Vegas définit des algorithmes probabilistes sans erreur.

Un exemple : Quicksort probabiliste.
- *Entrée :* Une liste L d'entiers.
- *Sortie :* Cette liste triée en ordre croissant.

Algorithme :

1. Choisir *aléatoirement de manière uniforme*[1] un entier x de la liste,
2. Déterminer les 2 sous-listes

$$L_1 = \{y \in L \mid y > x\} \text{ et } L_2 = \{y \in L \mid y < x\},$$

3. Appliquer la procédure récursivement à L_1 et L_2,
4. Retourner L_1 triée , x, L_2 triée .

Cet algorithme possède les propriétés suivantes :
- Il donne toujours un résultat correct (sans erreur),
- Le comportement peut varier d'une exécution à l'autre, même sur une seule entrée, mais le résultat est le même,
- Le temps d'exécution devient une variable aléatoire,
- On peut montrer que l'espérance du temps d'exécution est en $\mathcal{O}(n.\log_2 n)$.

Les avantages principaux des algorithmes probabilistes sont :

[1] On suppose que l'on dispose d'un générateur aléatoire uniforme.

– La performance, pour de nombreux problèmes, ils sont plus rapides que les meilleurs algorithmes déterministes connus,
– La simplicité de description et d'implémentation (modulo les générateurs aléatoires).

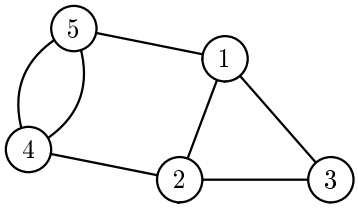
1.1.2 Algorithmes Monte Carlo

A contrario, les algorithmes Monte-Carlo peuvent produire des résultats erronés, mais l'on peut borner la probabilité d'erreur et la rendre exponentiellement petite avec un nombre polynomial d'itérations de l'algorithme.

Un exemple : coupure minimale dans un graphe. Un multigraphe est un graphe dans lequel il peut y avoir plusieurs arêtes entre deux sommets.

Définition 1. Une coupure dans un multigraphe est un ensemble d'arêtes dont la suppression rend le multigraphe non connexe.
Une coupure est minimale si elle est de taille minimum.

Exemple :



L'opération de *contraction* d'une arête consiste à :
– choisir aléatoirement de manière uniforme une arête et la contracter en identifiant les deux extrémités de cette arête,
– supprimer toutes les arêtes entre les deux sommets identifiés,
– conserver les arêtes existant entre les autres sommets.

L'algorithme consiste à itérer le processus de contraction jusqu'à ce qu'il ne reste que deux sommets et à produire l'ensemble des arêtes entre ces deux sommets comme candidat à être une coupure minimale.

Exemple :

Proposition 1. *L'opération de contraction ne diminue pas la taille d'une coupure dans le multigraphe.*

Preuve. Toute coupure dans le multigraphe à une étape intermédiaire de l'algorithme est une coupure dans le multigraphe d'origine.

La question de la correction de l'algorithme probabilite se pose donc en ces termes : « quelle est la probabilité d'erreur ? », c'est-à-dire quelle est la probabilité que la coupure produite ne soit pas minimale.

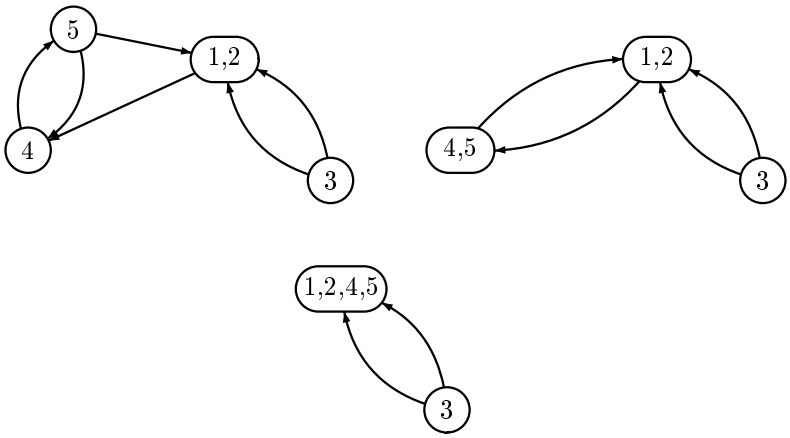


FIG. 1.1 – Contraction de 1 et 2, puis 4 et 5, puis 1,2 et 4,5

Calcul d'une borne de la probabilité d'erreur. Soit C une coupure minimale dans le multigraphe G. On définit l'évènement E_i par « aucune arête de C n'est contractée à la i-ème itération ».

On cherche à borner la probabilité d'erreur, dans le but de la rendre aussi petite que l'on veut.

Si l'on considère les deux évènements E_1 et E_2, qui ne sont pas indépendants : $Prob[E_1 \wedge E_2] = Prob[E_1] \times Prob[E_2 \mid E_1]$ où $Prob[E_1] \times Prob[E_2 \mid E_1]$ désigne la probabilité conditionnelle de E_2 étant donné E_1. Plus généralement, si l'on a k évènements $E_1, E_2, \ldots, E_k$ non indépendants :

$$Prob[\bigwedge_{i=1}^{k} E_i] = Prob[E_1] \times Prob[E_2 \mid E_1] \times \ldots Prob[E_k \mid \bigwedge_{i=1}^{k-1} E_i].$$

Soit k la taille de la coupure minimale C. Le graphe possède *au moins* $k\frac{n}{2}$ arêtes. En effet, s'il existait un sommet de degré[2] strictement inférieur à k, alors l'ensemble des arêtes incidentes à ce sommet constituerait une coupure de taille strictement inférieure à k.

On va borner inférieurement la probabilité qu'aucune arête de C ne soit contractée durant une exécution de l'algorithme. La probabilité que l'arête choisie uniformément à la 1e étape de l'algorithme soit dans C est inférieure à $\frac{k}{k/(n/2)} = \frac{2}{n}$, donc

$$Prob[E_1] \geq 1 - \frac{2}{n} \text{ et } Prob[E_2 \mid E_1] \geq 1 - \frac{2}{n-1}.$$

[2]degré : nombre d'arêtes incidentes à ce sommet.

La probabilité qu'aucune arête de C ne soit contractée à l'issue de la ième étape est telle que :

$$Prob[E_i \mid \bigwedge_{j=1}^{i-1} E_j] \geq 1 - \frac{2}{n-i+1}.$$

Comme il y a $n-2$ itérations, la probabilité que l'algorithme produise la coupure minimale C peut être bornée inférieurement par l'inverse d'un polynôme en n :

$$Prob[\bigwedge_{j=1}^{n-2} E_j] \geq \prod_{i=1}^{n-2}(1 - \frac{2}{n-j+1}) = \frac{2}{n(n-1)} \geq \frac{2}{n^2}.$$

Réduction de la probabilité d'erreur Une méthode classique consiste à itérer l'algorithme $\frac{n^2}{2}$ fois (ou plus généralement, un nombre polynomial de fois) :

$$Prob[\text{Erreur}] \leq (1 - \frac{2}{n^2})^{\frac{n^2}{2}} \simeq \frac{1}{e} < \frac{1}{2}.$$

En fait, si l'on itère l'algorithme précédent $\frac{n^3}{2}$ fois, on rend la probabilité d'erreur exponentiellement petite : $Prob[\text{Erreur}] \leq \frac{1}{2^n}$.

1.2 Classes de complexité probabilistes

On rappelle que la classe NP est celle des problèmes de décision vérifiables sur machine de TURING non déterministe (MTND) en temps polynomial.

On peut toujours supposer la machine *précise*, c'est à dire que tous les calculs sur une entrée X sont de même longueur, polynomiale en $|X|$.

Définition 2 (Classe NP – *Nondeterministic Polynomial-Time*)**.** Soit Σ un alphabet et $\mathcal{L} \subseteq \Sigma^*$.
On dit que L est dans NP si il existe une MTND N d'alphabet Σ et un polynôme à coefficients entiers p bornant le temps de calcul de N, tels que :
pour tout x de Σ^*,
- si $x \in L$, alors il existe $y \in \Sigma^*$ tel que $|y| \leq P(|x|)$ et n accepte (x, y),
- si $x \notin L$, alors pour tout $y \in \Sigma^*$, N rejette (x, y).

1.2.1 Classes RP et RLP

La classe RP est la classe des problèmes de décision pour lesquels il existe un algorithme probabiliste en temps polynomial, avec erreur d'un seul côté : il peut y avoir des réponses négatives fausses.

Définition 3 (Classe RP – *Randomized Polynomial-Time*)**.** La classe RP est la classe des langages L pour lesquels il existe une machine de Turing probabiliste A fonctionnant en temps polynomial telle que :
pour tout x de Σ^*,
- Si $x \in L$ alors $Prob_\Omega[A \text{ accepte } x] \geq \frac{1}{2}$,

– Si $x \notin L$ alors $Prob_\Omega[A \text{ accepte } x] = 0$.

Ω est l'espace probabiliste des tirages de la machine A.

On peut remarquer que :

– la classe RP correspond aux algorithmes Monte-Carlo (en temps polynomial) avec erreur possible lorsque la réponse est négative ($x \in L$ et $Arejettex$)

– la probabilité d'erreur

$$Prob_\Omega[x \in L \wedge A \text{ rejette } x]$$

est strictement inférieure à $\frac{1}{2}$,

– la classe co-RP est définie comme la classe des langages (ou des problèmes) dont le complémentaire est dans RP.

Exemple :

Le problème de primalité est dans la classe co-RP.

Propriété fondamentale de RP La probabilité d'erreur peut être réduite de manière exponentielle à l'aide d'un nombre polynomial d'itérations : si l'on itère k fois l'algorithme A, on obtient un algorithme A' tel que

$$Prob[x \in L \wedge A' \text{ rejette } x] < \frac{1}{2^k}.$$

De plus, pour δ arbitraire, si $k > \lceil \log \frac{1}{\delta} \rceil$, la probabilité d'erreur de A' devient strictement inférieur à δ. Autrement dit :

$$\text{si } x \in L, Prob[A' \text{ accepte } x] \geq 1 - \delta.$$

Classe ZPP Si l'on définit la classe ZPP (*Zero-Error Probabilistic Poly-Time*) comme la classe des problémes pour lesquels il existe un algorithme Las Vegas en temps polynomial, il est possible de montrer que :

$$\text{ZPP} \equiv \text{RP} \cap \text{coRP}.$$

En effet, un problème de RP $\cap$ coRP possède 2 algorithmes Monte-Carlo, l'un sans réponse positive fausse et l'autre sans réponse négative fausse. On peut alors lancer les deux algorithmes en parallèle...

Enfin, il est facile de montrer la relation suivante avec les classes P et NP : P $\subseteq$ RP $\subseteq$ NP. Pour cela, il suffit de reformuler les 2 conditions de la définition de la classe RP :

– si $x \in L$, alors la probabilité qu'un chemin soit acceptant pour x est $\geq \frac{1}{2}$,

– si $x \notin L$, alors il n'existe pas de chemin acceptant x.

Classe RLP On désire maintenant définir la classe des langages admettant un algorithme de Monte-Carlo, avec erreur d'un seul côté, en espace logarithmique. Un premier essai de définition pourrait être le suivant.

Définition 4 (Classe RL' – *Randomized Logarithmic-Space'*)**.** *RL'* est la classe des langages L pour lesquels il existe une machine probabiliste $\mathcal{M}$ fonctionnant en espace logarithmique telle que :
pour tout x de Σ^*,
- Si $x \in \mathcal{L}$ alors $Prob_\Omega[\mathcal{M} \text{ accepte } x] \geq \frac{1}{2}$,
- Si $x \notin \mathcal{L}$ alors $Prob_\Omega[\mathcal{M} \text{ accepte } x] = 0$.

Malheureusement, cette définition permettrait de montrer que NL = RL' :
En fait on définit RLP, souvent appelé RL :

Définition 5 (Classe RLP – *Randomized Logarithmic-Space*)**.** *RLP* est la classe des langages L pour lesquels il existe une machine probabiliste $\mathcal{M}$ fonctionnant en espace logarithmique et en temps polynomial telle que :
pour tout x de Σ^*,
- Si $x \in \mathcal{L}$ alors $Prob_\Omega[\mathcal{M} \text{ accepte } x] \geq \frac{1}{2}$,
- Si $x \notin \mathcal{L}$ alors $Prob_\Omega[\mathcal{M} \text{ accepte } x] = 0$.

Il est facile de voir que $\subseteq$ RLP $\subseteq$ NL.

1.2.2 Classes PP et BPP

Notre objectif dans cette section est de définir une classe de complexité correspondant aux langages décidables en temps polynomial par un algorithme du type Monte-Carlo avec erreurpossible des deux côtés.

Définition 6 (Classe PP – *Probabilistic Polynomial-Time*)**.** *PP* est la classe des langages L pour lesquels il existe une machine probabiliste A fonctionnant en temps polynomial telle que :
pour tout x de Σ^*,
- Si $x \in L$ alors $Prob_\Omega[A \text{ accepte } x] \geq \frac{1}{2}$,
- Si $x \notin L$ alors $Prob_\Omega[A \text{ accepte } x] < \frac{1}{2}$.

Cette classe a une définition assez naturelle pour certains problèmes théoriques, mais n'a pas vraiment de signification "très pratique". En effet, la condition d'acceptation par majorité est une condition trop fragile : si le temps de la machine A est borné par le polynôme p, une entrée x de taille n peut être dans L avec une probabilité d'acceptation

$$Prob_\Omega[A \text{ accepte } x] = \frac{1}{2} + \frac{1}{2^{p(n)}},$$

c'est-à-dire avec seulement 2 chemins acceptant de plus que de chemins rejettant.

La classe BPP est la classe des problèmes possédant un algorithmes Monte-Carlo en temps polynomial avec erreur des 2 côtés.

Définition 7 (Classe BPP – *Bounded-Error Probabilistic Polynomial-Time*)**.** *BPP* est la classe des langages L pour lesquels il existe une machine probabiliste A fonctionnant en temps polynomial telle que :
pour tout x de Σ^*,
- Si $x \in L$ alors $Prob_\Omega[A \text{ accepte } x] \geq \frac{3}{4}$,
- Si $x \notin L$ alors $Prob_\Omega[A \text{ accepte } x] \leq \frac{1}{4}$.

On peut montrer que cette classe possède la propriété de réduction exponentielle de la probabilité d'erreur avec un nombre polynomial d'itérations. En fait, les bornes $\frac{3}{4}$ et $\frac{1}{4}$ pourraient être remplacées par $\frac{1}{2} + \frac{1}{p(n)}$ et $\frac{1}{2} - \frac{1}{p(n)}$ pour n'importe quel polynôme p sans affecter la propriété de réduction de la probabilité d'erreur.

Proposition 1. *Les inclusions suivantes se démontrent facilement :*
- $RP \subseteq BPP \subseteq PP$,
- $PP \equiv$ *co-PP et* $BPP \equiv$ *co-BPP.*

1.3 Tests de primalité

Rappels sur les groupes et corps finis Pour tout entier $n \geq 2$, on note $\mathbb{R}_n$ l'ensemble $\mathbb{R}/n\mathbb{R}$ et $\mathbb{R}_n^*$ l'ensemble suivant :

$$\mathbb{R}_n^* = \{a \in \mathbb{R} \mid 1 \leq a \leq n \text{ et } \text{pgcd}(a, n) = 1\}.$$

On considère le groupe additif $(\mathbb{R}_n, +_n)$ et le groupe multiplicatif $(\mathbb{R}_n^*, \cdot_n)$, où $+_n$ et $\cdot_n$ sont l'addition et la multiplication modulo n.

Proposition 2. *Pour tout $n \geq 2$, l'inverse multiplicatif de $z \in \mathbb{R}_n$ peut-être calculé en temps polynomial.*

Preuve. En effet, l'algorithme d'Euclide nous permet de trouver x et y telles que :

$$\text{pgcd}(n, z) = zx + ny.$$

x est donc l'inverse multiplicatif de z.

Théorème 1. *Dans $(Z_n^*, \cdot_n)$, l'exponentiation peut se calculer en temps polynomial.*

Preuve. la stratégie naïve de calcul de a^k nécessite un nombre de multiplications proportionnel à k, donc exponentiel en $\lceil \log_2 k \rceil$.

La méthode des carrés successifs permet un calcul en temps polynomial. Si l'on veut calculer a^k, on considère la représentation binaire de k :

$$k = \sum_{i=0}^{t} b_i 2^i.$$

On remarque que :

$$a^k = \prod_{i=0}^{t} a_i^{b_i}$$

En utilisant les valeurs précalculées de la suite $(a^i)_{0\leq i\leq t}$ pour $t = \lceil \log_2 k \rceil$, le produit a^k peut être calculé en temps $O(\lceil \log_2 k \rceil)$.

On rappelle également, sans démonstration, le théorème (du reste) chinois.

Théorème 2. *Soit $(n_i)_{1\leq i\leq k}$ une suite d'entiers 2 à 2 premiers entre eux et $n = \prod_{i=1}^{k}$ leur produit. Alors pour toute suite $(r_i)_{1\leq i\leq k}$ de "résidus" modulo $n_i (1 \leq i \leq k)$, il existe un entier r tel que $r \equiv r_i$ modulo $n_i (1 \leq i \leq k)$.*

Définition 8 (Fonction Φ d'Euler)**.** La fonction indicatrice Φ d'Euler est définie par $\Phi(n) = |\mathbb{R}_n^*|$.

Exemple :
- Si $n = 11$, $\mathbb{R}_n^* = \{1, 2, \ldots, 10\}$ et $|\mathbb{R}_n^*| = 10$,
- Si $n = 12$, $\mathbb{R}_n^* = \{1, 5, 7, 11\}$ et $|\mathbb{R}_n^*| = 4$,

Théorème 3 (Théorème de Fermat-Euler)**.** *Pour tout $n \in \mathbb{N}$, et $\alpha \in \mathbb{R}_n^*$,*

$$\alpha^{\Phi(n)} \equiv 1 \ [n].$$

Corollaire 4 (Petit théorème de Fermat)**.** *Si p est premier, alors pour tout x entre 1 et $p-1$, $x^{p-1} \equiv 1 \ [p]$.*

La fonction d'Euler possède les propriétés suivantes.
- Si p est premier, alors $\Phi(p) = p - 1$
- Si p est premier et $\alpha > 0$, alors $\Phi(p^\alpha) = p^{\alpha-1}(p-1)$
- Si n, m sont premiers entre eux, alors $\Phi(nm) = \Phi(n)\Phi(m)$

Théorème 5. *si la décomposition de n en facteurs premiers est $n = \prod_{i=1}^{k} p_i^{\alpha_i}$, alors $\Phi(n) = \prod_{i=1}^{k} p_i^{\alpha_i - 1} = n \prod_{i=1}^{k} (1 - \frac{1}{p_i})$*

Définition 9 (Résidu quadratique)**.** Un a dans $\mathbb{R}_n^*$ est un *résidu quadratique* si il existe un x dans $\mathbb{R}_n^*$ tel que $a \equiv x^2 \bmod n$.

Théorème 6 (Critère d'Euler)**.** *Soit p premier et $a \in \mathbb{R}_p^*$. a est résidu quadratique si $a^{\frac{p-1}{2}} \equiv 1$ mod p.*

Il est facile de voir que :
- $a^{p-1} \equiv 1 \bmod p$ et $a^{\frac{p-1}{2}} \in \{1, -1\}$,
- $a^{\frac{p-1}{2}}$ est calculable en temps polynomial.

Définition 10 (Symbole de Jacobi)**.** Soit n impair de décomposition $n = \prod_{i=1}^{k} p_i^{\alpha_i}$ avec les p_i premiers. Pour tout $a \in \mathbb{R}_n^*$, le symbole de Jacobi de a modulo n est noté $\frac{[a}{n]}$ et défini par :

$$[\frac{a}{n}] = \prod_{i=1}^{k} [\frac{a}{p_i}]^{\alpha_i} \quad \text{où } [\frac{a}{p_i}] = a^{\frac{p_i-1}{2}} \text{ est le symbole de Legendre.}$$

Les propriétés suivantes sont utilisées par le test de primalité de Solovay-Strassen :

- La valeur ddu symbole de Jacobi est $[\frac{a}{n}] = 1$ ou -1,
- bien que la définition du symbole de Jacobi utilise la factorisarion de l'entier n, il existe un algorithme en temps polynomial pour calculer $[\frac{a}{n}]$.
- Cet algorithme utilise les règles de simplification suivantes ainsi que la loi de réciprocité quadratique :
  - $[\frac{a.b}{n}] = [\frac{a}{n}].[\frac{b}{n}]$,
  - Si $a \equiv b\ [n], [\frac{a}{n}] = [\frac{b}{n}]$,
  - Si m, n sont premiers entre eux, $[\frac{m}{n}].[\frac{n}{m}] = (-1)^{\frac{n-1}{2}\frac{m-1}{2}}$. C'est ce qu'on appelle la loi de réciprocité quadratique.
  - on se ramène ainsi au calcul de $[\frac{-1}{n}]$ et $[\frac{2}{n}]$, qui sont faciles.

Test de Solovay-Strassen L'idée de cet algorithme est la suivante :

- Si n est premier, alors pour tout $a \in \mathbb{R}_n^*$

$$[\frac{a}{n}] \equiv a^{\frac{n-1}{2}}\ mod n,$$

- Si n est composé, alors il existe une *proportion importante* de $a \in \mathbb{R}_n^*$ tels que

$$[\frac{a}{n}] \not\equiv a^{\frac{n-1}{2}}\ [n].$$

Algorithme de Solovay-Strassen

- *Entrée :* n impair.
- *Sortie :* PREMIER ou COMPOSÉ.

Algorithme :

1. Choisir aléatoirement de manière uniforme un $a \in \mathbb{R}_n \setminus \{0\}$,
2. Calculer le $pgcd(a, n)$,
3. Si $pgcd(a, n) \neq 1$, retourner COMPOSÉ,
4. Calculer $[\frac{a}{n}]$ et $a^{\frac{n-1}{2}} \bmod n$
5. Si $[\frac{a}{n}] \equiv a^{\frac{n-1}{2}}\ [n]$, retourner PREMIER,
6. Retourner COMPOSÉ.

L'algorithme est de type RP car :

- l'algorithme est en temps polynomial,
- lorsque l'algorithme retourne COMPOSÉ, la réponse est correcte ; en effet, c'est qu'il a trouvé un $a \in \mathbb{R}_n \setminus \{0\}$ tel que soit $pgcd(a, n) \neq 1$, soit$[\frac{a}{n}] \not\equiv a^{\frac{n-1}{2}}\ [n]$,
- le lemme suivant montre que la probabilité d'erreur, i.e. celle de retourner PREMIER alors que n est composé, est inférieure à $\frac{1}{2}$.

On considère l'ensemble $J_n = \{a \in \mathbb{R}_n^* \mid [\frac{a}{n}] \equiv a^{\frac{n-1}{2}} \mod n\}$.

Lemme 1. *Si n est composé, $|J_n| \leq \frac{|Z_n^*|}{2}$.*

Preuve (Lemme 1). En remarquant que $(J_n, \cdot_n)$ est un sous-groupe de $(\mathbb{R}_n^*, \cdot_n)$, il suffit donc de montrer que c'est un sous-groupe propre, i.e. $J_n \neq Z_n^*$, car l'ordre d'un sous-groupe divise l'ordre du groupe.

On suppose qu'il existe un entier n composé tel que $J_n = Z_n^*$. La factorisation de n peut s'écrire de la manière suivante :

$$n = \underbrace{p_1^{\alpha_1}}_{q} \cdot \underbrace{\prod_{i=2}^{k} p_i^{\alpha_i}}_{m}.$$

c'est-à-dire $n = q.m$ avec q et m premiers entre eux.

L'application du théorème du reste chinois permet d'affirmer qu'il existe un $a \in \mathbb{R}_n^*$ tel que $a \equiv g \bmod q$ et $a \equiv 1 \bmod m$ où g est un *générateur* du groupe $(\mathbb{R}_q^*, \cdot_q)$. On remarque que : $a \equiv 1 \bmod p_i$ pour tout $i \geq 2$.

On traite séparément les cas $\alpha_1 = 1$ et $\alpha_1 \geq 2$:

– Si $\alpha_1 = 1$, on a $n = p_1.m$:

$$[\frac{a}{n}] = \prod_{i=1}^{k} [\frac{a}{p_i}]^{\alpha_i} = [\frac{g}{p_1}]. \prod_{i=2}^{k} [\frac{a}{p_i}]^{\alpha_i}$$

$$[\frac{a}{n}] = [\frac{g}{p_1}]. \prod_{i=2}^{k} [\frac{1}{p_i}]^{\alpha_i} = [\frac{g}{p_1}] = -1$$

car un générateur de $\mathbb{R}_q^*$ ne peut être résidu quadratique. Par hypothèse $J_n = Z_n^*$ et $a^{\frac{n-1}{2}} \equiv_1 \bmod n$ et donc aussi mod m, ce qui est impossible ($a \equiv 1 \bmod m$).

– Si $\alpha_1 \geq 2$, on a aussi par hypothèse $J_n = Z_n^*$ et $a^{\frac{n-1}{2}} \equiv_1 \bmod n$. On peut en déduire : $a^{n-1} \equiv 1 \bmod n$ et $g^{n-1} \equiv 1 \bmod q$, car q divise n et $a \equiv g \bmod q$. Puisque g est générateur de $\mathbb{R}_q^*$, son ordre est $\Phi(q) = p_1\alpha_1(p_1 - 1)$ Comme $\alpha_1 \geq 2$, p_1 divise $\Phi(q)$. Par ailleurs, l'ordre de g doit diviser $n-1$, et alors p_1 divise $n-1$. Ce qui est impossible car $p_1 | n$. ⊠

Corollaire 7. *Le problème de primalité est dans co-RP.*

Test de Rabin-Miller Le test de Rabin-Miller est un raffinement de celui de Solovay-Srassen qui se fonde sur la remarque suivante : si $a^{\frac{n-1}{2}} \equiv_1 \bmod n$ et si $\frac{n-1}{2}$ est encore pair, alors $a^{\frac{n-1}{4}}$ est une racine carrée de 1 mod n et doit donc être égale à 1 ou -1.

Si l'on examine les racines successives de 1 mod n, $a^{\frac{n-1}{2}}$, $a^{\frac{n-1}{4}}$,...,$a^{\frac{n-1}{2^r}}$ tant que c'est possible, c'est-à-dire jusqu'au plus petit r tel que $\frac{n-1}{2^r}$ soit impair, alors si n est premier, la première racine différente de 1 que l'on obtient doit être -1.

Algorithme de Rabin-Miller

– *Entrée :* n impair.

– *Sortie :* PREMIER ou COMPOSÉ.

Algorithme :

1. Calculer r, s tels que $(n-1) = 2^r s$ avec s impair,
2. Choisir a aléatoirement de manière uniforme dans $\mathbb{R}_n \setminus \{0\}$,
3. Pour $i = 0, \ldots, r$, calculer $b_i = a^{2^i \cdot s}$,
4. Si $a^{n-1} = b_r \not\equiv 1 \bmod n$, retourner COMPOSÉ,
5. Si pour tout $0 \leq i \leq r$, $b_i \equiv 1 \bmod n$, retourner PREMIER,
6. Soit $j = \max\{|i| \mid b_i \not\equiv 1\} \bmod n$,
7. Si $b_j \equiv -1 \bmod n$ retourner COMPOSÉ, sinon PREMIER.

On remarque que :

– L'entier n satisfait le test si tous les entiers de la liste $(bi)_{0 \leq i \leq r}$ sont égaux à 1 ou bien si l'un d'entre eux est égal à -1,
– si n est premier, alors la réponse de l'algorithme est correcte.

Proposition 3. *Si n est composé, alors la probabilité que n satisfasse le test de Rabin-Miller est inférieure à $\frac{1}{4}$.*

Chapitre 2

Chaînes de Markov. Analyse d'algorithmes. Bornes de Chernoff

2.1 Variables aléatoires. Espérance. Inégalité de Markov

2.1.1 Espace probabiliste

Tout énoncé de type probabiliste fait référence à un espace probabiliste sous-jacent. Un *espace probabiliste* est défini en termes d'un espace d'*échantillonnage* muni d'une structure algébrique et d'une *mesure de probabilités* sur cet espace. Un espace d'échantillonnage Ω est un ensemble arbitraire, éventuellement infini, dont les éléments sont appelés *évènements élémentaires*. Un sous-ensemble $\mathcal{E} \subseteq \Omega$ est appelé un *évènement*. Cependant toute collection de sous-ensembles de Ω ne permet pas d'obtenir un espace probabiliste bien défini.

Définition 11. Une σ-algèbre $(\Omega, \mathcal{F})$ est la donnée d'un espace d'échantillonnage Ω et d'une collection de sous-ensembles $\mathcal{F}$ satisfaisant les conditions suivantes :
- $\emptyset \in \mathcal{F}$,
- stabilité par passage au complémentaire,
- stabilité par réunion dénombrable.

Définition 12. Etant donné une σ-algèbre $(\Omega, \mathcal{F}$, une mesure de probabilités μ est une fonction : $\mathcal{F} \longrightarrow \mathbb{R}^+$ telle que :
- pour tout $\omega \in \Omega$ $\mu(\{\omega\}) \leq 1$,
- $\mu(\Omega) = 1$,
- pour toute suite d'évènements 2 à 2 disjoints, $(\mathcal{E}_\rangle)_{i \geq 0}$, $\mu(\cap_i \mathcal{E}_\rangle) = \sum_i \mathcal{E}_\rangle$.

Définition 13. Un espace probabiliste $(\Omega, \mathcal{F}, \mu)$ consiste en une σ-algèbre $(\Omega, \mathcal{F}$ munie d'une mesure de probabilités μ.

Dans la suite, pour faire référence à un espace probabiliste, lorsque la σ-algèbre et la mesure de probabilités associées sont évidentes, on utilisera la notation simplifiée Ω.

2.1.2 Variables aléatoires et espérance

Définition 14 (Variable aléatoire)**.** Une variable aléatoire X sur un espace probabiliste Ω est une fonction $X : \Omega \longrightarrow \mathbb{R}$. Une variable aléatoire X est *discrète* si l'ensemble de ses valeurs est fini ou dénombrable.

La notation $X = a$ désigne l'ensemble $\{s \in \Omega \mid X(s) = a\}$.

Définition 15 (Variables aléatoires indépendantes)**.** Deux variables aléatoires X et Y sont *indépendantes* si

$$Prob[X = a \text{ et } Y = b] = Prob[X = a] \times Prob[X = b].$$

Définition 16 (Espérance)**.** L'*espérance* d'une variable aléatoire X discrète, notée $\mathbb{E}[X]$, est définie par :

$$\mathbb{E}[X] = \sum_i i.Prob[X = i].$$

L'espérance est dite *finie* si $\mathbb{E}[X]$ converge. Elle est non bornée sinon.

Proposition 4 (Linéarité)**.** *Pour tout ensemble fini $X_1, \ldots, X_n$ de variables aléatoires discrètes à espérances finies :*

$$\mathbb{E}[\sum_{i=1}^{n} X_i] = \sum_{i=1}^{n} \mathbb{E}[X_i],$$

et

$$\mathbb{E}[c.X] = c.\mathbb{E}[X].$$

Preuve.

$$\begin{aligned}
\mathbb{E}[X+Y] &= \sum_i (\sum_j (i+j) Prob[X = i \wedge Y = j]) \\
&= \sum_i i. \left(\sum_j Prob[X = i \wedge Y = j] + \right. \\
&\qquad \left. \sum_i \sum_j j\, Prob[X = i \wedge Y = j] \right) \\
&= \sum_i i Prob[X = i] + \sum_j j\, Prob[Y = j] \\
&= \mathbb{E}[X] + \mathbb{E}[Y]
\end{aligned}$$

et

$$\begin{aligned}\mathbb{E}[c.X] &= \sum_j j.Prob[c.X = j] \\ &= \sum_j j.Prob[X = \frac{j}{c}] \\ &= c.\mathbb{E}[X]\end{aligned}$$

Proposition 5. *Pour toute variable aléatoire discrète X, $\mathbb{E}[X^2] \geq (\mathbb{E}[X])^2$.*

Preuve. Soit la variable aléatoire $Y = (X - \mathbb{E}[X^2])^2$. Cette variable est positive et son espérance l'est également : $\mathbb{E}[Y] \geq 0$

$$\begin{aligned}\mathbb{E}[Y] &= \mathbb{E}[X^2 - X\mathbb{E}[X] + (\mathbb{E}[X])^2] \\ &= \mathbb{E}[X^2] - \mathbb{E}[X.\mathbb{E}[X]] + (\mathbb{E}[X])^2 \\ &= \mathbb{E}[X^2] - (\mathbb{E}[X])^2\end{aligned}$$

En effet : $\mathbb{E}[X.\mathbb{E}[X]] = (\mathbb{E}[X])^2$

2.1.3 Inégalité de Markov

Théorème 8. *Si X est une variable aléatoire positive, alors pour tout $a > 0$:*

$$\text{Prob}[X \geq a] \leq \frac{\mathbb{E}[X]}{a}.$$

Preuve. Soit I_a la variable (de Bernouilli) définie par

$$I_a = \begin{cases} 0 & \text{si } X \geq A \\ 1 & \text{sinon} \end{cases}$$

Comme I_a est une variable de Bernouilli, c'est-à-dire à valeurs $0, 1$:

$$\mathbb{E}[I_a] = Prob[X \geq a] \leq \mathbb{E}[\frac{X}{a}] = \frac{\mathbb{E}[X]}{a}.$$

On utilise le fait que l'espérance conserve les inégalités pour les variables aléatoires positives : comme $I_a \leq \frac{X}{a}$, $\mathbb{E}[I_a] \leq \frac{\mathbb{E}[X]}{a}$.

2.2 Chaînes de Markov

Dans cette section, on se restreint aux les chaînes de Markov en temps discret avec un ensemble d'états fini ou dénombrable. Dans le cas d'un ensemble d'états fini, il est alors possible de les représenter aussi bien par des graphes de transition probabilistes que par des matrices de Markov.

2.2.1 Définition et représentation

Définition 17 (Processus stochastique)**.** Un *processus stochastique* est un ensemble $\{X(t) \mid t \in T\}$ de variables aléatoires, où t représente en général le temps.

$X(t)$ est l'état du processus à l'instant t, qui sera noté X_t.

Remarques :
- Si pour tout t, X_t prend des valeurs dans un ensemble fini ou dénombrable, le processus est dit en espace discret .
- Le processus est en temps discret si l'ensemble T est dénombrable. On l'appelle alors une DTMC (*Discrete Time Markov Chain*).

Les processus considérés dans la suite seront toujours en espace et en temps discrets.

Définition 18 (Chaîne de Markov)**.** Un processus stochastique en temps discret $(X_t)_{t\in T}$ est une *chaîne de Markov* s'il possède la propriété de Markov d'indépendance par rapport à l'historique :

$$Prob[X_t = \varphi_t | X_0 = a_0, \ldots, X_{t-1} = a_{t-1}] = Prob[X_t = \varphi_t | X_{t-1} = a_{t-1}].$$

Autrement dit, la probabilité d'être dans un certain état X_t à l'instant t ne dépend que de la probabilité d'être dans un autre état X_{t-1} à l'instant précédent, et non pas de l'historique suivant lequel le processus est arrivé dans l'état X_{t-1} .

L'ensemble des états est discret et peut donc être identifié à un segment initial de $\mathbb{N}$.

Représentation matricielle

On suppose le nombre d'états fini, égal à n. On considère la matrice suivante : $P = (p_{i,j})_{1\leq i,j\leq n}$ où $P_{i,j} = Prob[X_t = j | X_{t-1} = i]$, dite de Markov. La propriété caractéristique d'une telle matrice est : pour tout i,

$$\sum_{j=1}^{n} p_{i,j} = 1$$

Si l'on note $\bar{p}(t) = (p_0(t), \ldots, p_n(t))$ la distribution de probabilités à l'instant t, c'est-à-dire $p_i(t)$ est la probabilité d'être dans l'état i à l'instant t, on constate que : $\bar{p}(t) = \bar{p}(t-1).P$

La matrice P^t permet d'obtenir la distribution de probabilités à l'instant t.

Représentation sous forme de système de transition probabiliste

Si l'on se donne un état initial s_0, on peut considérer le système de transition probabiliste $\mathcal{M} = (S, P, s_0)$ où S est l'ensemble des états et P la fonction de transition probabiliste.

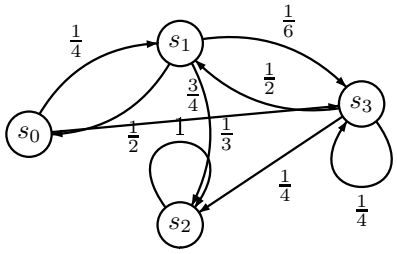


FIG. 2.1 – Système de transition probabiliste

Le graphe de transition (S, R) associé est défini par :

$$(i, j) \in R \text{ si et seulement si } p_{i,j} > 0.$$

Le système de transition probabiliste peut aussi être représenté par la matrice de Markov :

$$P = \begin{pmatrix} 0 & \frac{1}{4} & 0 & \frac{3}{4} \\ \frac{1}{2} & 0 & \frac{1}{3} & \frac{1}{6} \\ 0 & 0 & 1 & 0 \\ 0 & \frac{1}{2} & \frac{1}{4} & \frac{1}{4} \end{pmatrix}$$

Problème du marin saoul[1]

Cet exemple illustre la notion de promenade, ou marche, aléatoire.

Préambule : « Une longue saison de pêche s'achève. Les gobe-mouches des ports s'esbaudissent devant les traits burinés des revenants du domaine Neptunéen. L'un de ces forts gaillards, un marin breton retournant fourbu de tant de lutte, se trouva encore la force de bien vouloir en découdre avec sa sobriété contrainte des derniers temps. Ledit marin, empreint d'une incroyable envie qu'elle et sa lucidité y trépassassent, se bourra la gueule à grand renfort d'alcool (breton, cela va de soi). »

Cela étant, il lui faut maintenant rentrer chez lui, remercier sa femme de son attente. Sa bicoque se situant à un kilomètre du lieu saint d'où le sang du Christ coule à flot, il entreprend l'enivrant trajet. Son état (breton aussi) l'amène à se demander tous les cent mètres si il est dans la bonne direction, et avec probabilité $\frac{1}{2}$, à changer (éventuellement) de direction (continuer ou revenir sur ses pas).

[1] « Marin breton, si vous voulez. »

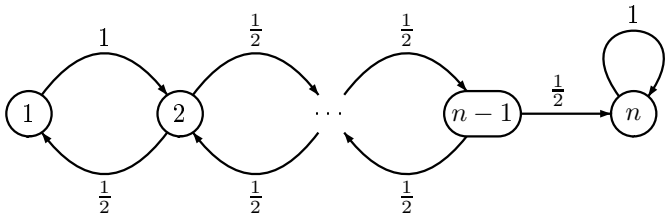


FIG. 2.2 – Représentation du problème

L'état n est absorbant[2].

2.2.2 Algorithme probabiliste pour 2-SAT (C. Papadimitriou)

Problème k-SAT

- *Entrée :* Un ensemble de variables propositionnelles $\{x_1, \ldots, x_n\}$ et une formule F sous forme normale conjonctive : $\bigwedge_j C_j$ où C_j est une clause à exactement deux littéraux.
- *Sortie :* Une valuation satisfaisant la formule ou « Non satisfaisable ».

Le problème 2-SAT peut être résolu par un algorithme déterministe en temps polynomial, mais aussi par un algorithme probabiliste en espace logarithmique, du à Papadimitriou :

Algorithme Rand2-SAT *Algorithme :*

1. Initialiser une valuation v_0 arbitraire,
2. Si v_0 satisfait F, retourner v_0,
3. Répéter $2n^2$ fois :
   (a) Choisir une clause non satisfaite arbitraire,
   (b) Choisir aléatoirement de manière uniforme l'un des deux littéraux de cette clause et changer la valeur de la valuation courante sur cette variable.
4. Si une valuation satisfaisante a été trouvée, retourner cette valuation,
5. Sinon, retourner « Formule non satisfaisable ».

L'algorithme retourne une réponse correcte si la formule n'est pas satisfaisable. Dans le cas contraire, on va montrer que l'algorithme retourne une valuation satisfaisante avec probabilité $\geq \frac{1}{2}$.

Lemme 2. *Si F est une formule 2-SAT satisfaisable, alors l'espérance du nombre d'étapes de l'algorithme est inférieure où égale à n^2.*

[2]absorbant : « rien à voir avec l'état du marin. »

Preuve. On suppose qu'il existe une valuation v satisfaisant F. Soit v_i la valuation obtenue après i étapes de l'algorithme et X_i le nombre de variables pour lesquelles v_i coïncide avec v. Si $X_i = n$, l'algorithme termine avec une valuation satisfaisant F.

Question : Partant de $X_i < n$, combien d'étapes faut-il pour que $X_j = n$?

On remarque que si $X_i = 0$, $X_{i+1} = 1$ et $Prob[X_{i+1} = 1 | X_i = 0] = 1$, On suppose maintenant que $1 \leq X_i \leq n-1$. A chaque étape, on choisit une clause C non satisfaite. Comme $v(c) = 1$, v_i et v diffèrent au moins sur une variable de cette clause. Chaque clause ayant exactement 2 littéraux, la probabilité d'augmenter le nombre d'accords entre v_i et v est supérieure à $\frac{1}{2}$:

$$Prob[X_{i+1} = j+1 | X_i = j] \geq \frac{1}{2},$$

$$Prob[X_{i+1} = j-1 | X_i = j] \leq \frac{1}{2}.$$

Le processus stochastique $X_0, X_1, \ldots, X_i, X_{i+1}, \ldots$ n'est pas nécessairement une chaîne de Markov. la probabilité d'augmenter X_i dépend du fait que v_i et v diffèrent sur 1 ou 2 variables, et donc des clauses qui ont été considérées dans le passé.

On considère la chaîne de Markov $Y_0, Y_1, \ldots, y_i, Y_{i+1}, \ldots$ qui est une version "pessimiste" du processus précédent :

$$Y_0 = X_0 \text{ et}$$

$$Prob[Y_{i+1} = j+1 | Y_i = j] = Prob[Y_{i+1} = j-1 | Y_i = j] = \frac{1}{2}.$$

Le temps moyen pour atteindre la valeur n à partir d'un point quelconque est plus grand pour la chaîne de Markov Y que pour le procesus X. cette chaîne de markov modélise une promenade alétoire sur un graphe non orienté, qui est en fait une droite : c'est celle quui correspond au problème du marin de la page 18.

Soit Z_j une variable aléatoire représentant le nombre d'étapes nécessaires pour atteindre n à partir de j et h_j l'espérance de cette variable. Si $1 \leq j \leq n-1$:

- avec probabilité $\frac{1}{2}$, l'état suivant est $j-1$ et $Z_j = 1 + Z_{j-1}$,
- avec probabilité $\frac{1}{2}$, l'état suivant est $j+1$ et $Z_j = 1 + Z_{j+1}$.

On obtient le système d'équations suivant pour h_j :

- $h_n = 0$,
- $h_0 = h_1 + 1$,
- $1 \leq j \leq n-1$:

$$\begin{aligned} h_j &= \mathbb{E}[Z_j] = E[\frac{1}{2}(1 + Z_{j-1}) + \frac{1}{2}(1 + Z_{j+1})] \\ &= \frac{h_{j+1} + 1}{2} + \frac{h_{j-1} + 1}{2} \end{aligned}$$

Il est facile de montrer par récurrence sur j que $h_j = h_{j+1} + 2j + 1$ et donc que :

$$h_0 = \sum_{i=1}^{n-1} (2i+1) = n^2.$$

Théorème 9. *L'algorithme Rand2-SAT retourne un résultat correct si la formule n'est pas satisfaisable. Si la formule est satisfaisable, alors il retourne une valuation la satisfaisant avec probabilité supérieure à* $\frac{1}{2}$.

Preuve. On suppose que la formule F est satisfaisable. Soit Z est le nombre d'étapes nécessaire pour trouver une valuation satisfaisant F :

$$Prob[Z > 2n^2] \leq \frac{\mathbb{E}[Z]}{2n^2} = \frac{n^2}{2n^2} = \frac{1}{2}.$$

On remarque que cet algorithme probabiliste est du type RP et que l'on peut réduire exponentiellement la probabilité d'erreur en l'itérant un nombre polynomial de fois.

2.2.3 Algorithme probabiliste pour 3-SAT (U. Schönig)

La première idée consiste à réutiliser l'algorithme précédent avec un nombre m d'étapes, i.e. de mofification de la valuation courante sur une variable. En reprenant le même type d'analyse que précédemment, on obtient un processus stochastique $(X_i)i \geq 0$ satisfaisant :

$$Prob[X_{i+1} = j+1 | X_i = j] \geq \frac{1}{3},$$

$$Prob[X_{i+1} = j-1 | X_i = j] \leq \frac{2}{3}.$$

La chaîne de Markov associée $(X_i)i \geq 0$ est alors définie par :

$$Y_0 = X_0 \text{ et}$$

$$Prob[Y_{i+1} = j+1 | Y_i = j] = \frac{1}{3}$$

$$Prob[Y_{i+1} = j-1 | Y_i = j] = \frac{2}{3}.$$

La probabilité de « revenir en arrière » est donc de $\frac{2}{3}$ et l'espérance h_j du nombre d'étapes pour atteindre n à partir de j est donnée par :

- $h_n = 0$,
- $h_0 = h_1 + 1$,
- $h_j = \frac{2}{3}h_{j-1} + \frac{1}{3}h_{j+1} + 1 (1 \leq j \leq n-1)$

La solution de ce système d'équations est :

$$h_j = (2^{n+2} - 2^{j+2}) - 3(n-j).$$

Le nombre moyen d'étapes nécessaires serait alors exponentiel ($\Theta(2^n)$).
La remarque fondamentale de U. Schönig est la suivante :

- Après initialisation, le processus a plus de chances d'aller vers 0 que vers n. Par conséquent, il peut être meilleur de redémarrer le processus avec beaucoup de valuations initiales choisies aléatoirement et de le faire tourner chaque fois avec un petit nombre d'étapes, pluôt que de le faire tourener un grand nombre de fois avec la même valuation initiale.
- Si l'on choisit une valuation initiale aléatoirement de manière uniforme, alors le nombre de variables sur lesquelles elle coïncide avec la valuation satisfaisant F admet une distribution binomiale d'espérance $\frac{n}{2}$. Avec une probabilité non négligeable, le processus peut être initialisé avec une valuation coïncidant avec v sur plus de $\frac{n}{2}$ variables.

Algorithme Rand3-SAT

- *Entrée :* Une formule sous forme normale conjonctive.
- *Sortie :* Une valuation satisfaisant la formule ou « Non satisfaisable ».

Algorithme : Répéter m fois :

1. Initialiser une valuation choisie aléatoirement de manière uniforme,
2. Répéter $3n$ fois :
   (a) Choisir une clause non satisfaite arbitraire,
   (b) Choisir aléatoirement de manière uniforme l'un des trois littéraux de cette clause et changer la valeur de la valuation sur la variable correspondante.
3. Si une valuation satisfaisante a été trouvée, retourner cette valuation,
4. Sinon, retourner « Formule non satisfaisable ».

Analyse : On suppose qu'il existe v telle que $v(F) = 1$. Soit q la probabilité que l'algorithme atteigne la valuation v, ou une autre valuation satisfaisante, en $3n$ étapes à partir d'une valuation aléatoire.

On note q_j une borne inférieure sur la probabilité que le processus modifié atteigne la valuation v, ou une autre valuation satisfaisante, en $3n$ étapes à partir d'une valuation initiale différant de v sur exactement j variables. On peut associer à ce processus une marche aléatoire qui représente le déplacement d'une particule sur la droite des entiers avec probabilité $\frac{1}{3}$ de déplacement vers le haut ($\uparrow$) et probabilité $\frac{2}{3}$ de déplacement vers le bas ($\downarrow$). La probabilité que dans une suite de $(j+2k)$ déplacements, il y ait k déplacements vers le bas et $j+k$ déplacements vers le hautest la suivante :

$$\binom{j+2k}{k} \left(\frac{2}{3}\right)^k \left(\frac{1}{3}\right)^{j+k}.$$

On obtient ainsi une borne inférieure sur la probabilité que le processus modifié atteigne la valuation v, ou une autre valuation satisfaisante, en $(j+2k) \leq 3n$ étapes :

$$q_j \geq \max_{0 \leq k \leq j} \binom{j+2k}{k} \left(\frac{2}{3}\right)^k \left(\frac{1}{3}\right)^{j+k}.$$

En particulier, pour $k = j$:

$$q_j \geq \binom{3j}{j} \left(\frac{2}{3}\right)^j \left(\frac{1}{3}\right)^{2j}.$$

Par ailleurs, la formule de Stirling permet d'obtenir une approximation pour les coefficients du binôme :

Lemme 3 (Stirling)**.**

$$\sqrt{2\pi m}\left(\frac{m}{e}\right)^m \leq m! \leq 2\sqrt{2\pi m}\left(\frac{m}{e}\right)^m.$$

Corollaire 10.

$$\binom{3j}{j} = \frac{(3j)!}{(2j)!.j!} \geq \frac{c}{\sqrt{j}}\left(\frac{27}{4}\right)^j \quad \textit{où } c = \frac{\sqrt{3}}{8\sqrt{\pi}}.$$

On obtient ainsi une borne inférieure pour q :

$$\begin{aligned} q &\geq \sum_{j=0}^{n} \textit{Prob}[\text{une valuation aléatoire diffère de } v \text{ sur } j \text{ variables}] \\ &\geq \frac{1}{2^n} + \sum_{j=1}^{n} \binom{n}{j}\left(\frac{1}{2}\right)^n \left(\frac{c}{\sqrt{j}.2^j}\right) \\ &\geq \frac{c}{\sqrt{n}}\left(\frac{1}{2}\right)^n \sum_{j=0}^{n} \binom{n}{j}\left(\frac{1}{2}\right)^j j^n \\ &\geq \frac{c}{\sqrt{n}\left(\frac{1}{2}\right)^n}\left(\frac{3}{2}\right)^n = \frac{c}{\sqrt{n}}\left(\frac{3}{4}\right)^n \end{aligned}$$

Le nombre de valuations essayés par le processus avant d'en trouver une satisfaisante est une variable aléatoire géométrique dec paramètre q et donc d'espérance $\frac{1}{q}$. Le nombre d'étapes de l'algorithme pour chaque valuation initiale est en $\mathcal{O}(n)$. L'espérance du nombre d'étapes pour trouver une solution est donc en $\mathcal{O}(n^{\frac{3}{2}}.\left(\frac{4}{3}\right)^n)$.

L'algorithme probabiliste obtenu est un algorithme de Monte-Carlo dont la complexité en temps est exponentielle, mais dont la complexité en espace est *logarithmique*.

2.2.4 Compléments sur les chaînes de Markov

Classification des états

Définition 19. On note $i \to j$ si $Prob(X_n = j \text{ pour un } n \geq 0 | X_0 = i) > 0$, et $i \leftrightarrow j$ si $i \to j$ et $j \to i$.

$\leftrightarrow$ est alors une relation d'équivalence.

On rappelle qu'une chaîne de Markov sur un ensemble d'états S est définie par :

- une distribution initiale $\lambda = (\lambda_i)_{i \in S}$,
- une matrice de Markov $P = (p_{ij})_{i,j \in S}$.

La matrice $P^n = (p_{ij}^{(n)})$ donne la probabilité $p_{ij}^{(n)}$ de passer de i à j en n étapes.

Proposition 6. *Si $i \neq j$, les conditions suivantes sont équivalentes :*

1. $i \to j$,
2. *Il existe* $i_0 = i, \ldots, i_n = j$ *tels que* $\prod_{i=0}^{n-1} p_{i_k i_{k+1}} > 0$,
3. $p_{ij}^{(n)} > 0$ *pour un certain* $n > 0$.

Preuve.

$$p_{ij}^{(n)} \leq Prob(X_n = j \text{ pour un } n > 0 | X_0 = i) \leq \sum_{n \geq 0} p_{ij}^{(n)}$$

$$p_{ij}^{(n)} = \sum_{i_0, \ldots, i_n} \prod_{k=0}^{n-1} p_{i_k i_{k+1}}$$

Définition 20 (Classe close)**.** Une classe d'états C est close si pour tout $i \in C$ et $j \in S$ tels que $i \to j$ alors $j \in C$.

États récurrents et transients

Définition 21. Soit $(X_n)_{n \geq 0}$ une chaîne de Markov.

- Un état i est un état récurrent si $Prob(X_n = i \text{ pour un } n | X_0 = i) = 1$,
- Un état i est un état transient si $Prob(X_n = i \text{ pour un } n | X_0 = i) < 1$.

Dans une chaîne de Markov, on remarque les deux faits suivants :

- une composante récurrente (i.e. une composante dont tous les états sont récurrents) est close,
- soit on part d'un état récurrent, soit on part d'un étant transient.

Proposition 7. *Si une chaîne de Markov commence dans une composante récurrente C, alors la chaîne visite tout état de C infiniment souvent avec probabilité 1.*

Preuve. Si i est l'état initial, alors la probabilité de l'évènement e_n définit par « atteindre l'état j au moins n fois » est de :

$$Prob[\text{atteindre l'état } j|X_0 = i] \times Prob[\text{atteindre l'état} j|X_0 = j]^{n-1} = 1,$$

car les états sont récurrents.

Or, $\{e_n|n \in \mathbb{N}\}$ est un ensemble dénombrable d'évènements tels que $Prob[e_n] = 1$ pour tout n. Si $e = \cap e_n$, alors $Prob[e] = 1$.

En effet, $1 - Prob[e] = Prob[\mathbb{E} - \cap e_n] = Prob[\cup \bar{e_n}] \leq \sum Prob[\bar{e_n}] = 0$.

Propriété fondamentale des chaînes de Markov

Définition 22 (Chaîne de Markov irréductible)**.** Une chaîne de Markov est *irréductible* si son graphe de transition est fortement connexe.

On note r_{ij}^t la probabilité d'atteindre l'état j, pour la première fois, à partir de l'état i, à l'instant t.

On remarque qu'un état i est *récurrent* si $\sum_{t \geq 1} r_{ii}^t = 1$ et transient si $\sum_{t \geq 1} r_{ii}^t < 1$.

Définition 23. Un état récurrent i est dit *récurrent positif* si l'espérance du temps de retour à l'état i $h_{ii} = \sum_{t \geq 1} t.r_{ii}^t$ est bornée.

On remarque que si un état i est récurrent, alors une fois que la chaîne a visité cet état, elle y retournera dans le futur *presque surement*, c'est-à-dire avec probabilité 1.

Définition 24. Un état j est *périodique* s'il existe un entier $k > 1$ tel que $Prob[X_{t+s} = j|X_t = j] > 0$ entraine que s est divisible par k, et *apériodique* sinon.
Une chaîne de Markov est *ergodique* ssi tous ses états sont *ergodiques*, c'est-à-dire récurrents positifs et apériodiques.

Lemme 4. *Toute chaîne de Markov finie, irréductible et apériodique est une chaîne ergodique.*

Définition 25. Une distribution de probabilités $\bar{\pi}$d'une chaîne de Markov est dite *stationnaire* si $\bar{\pi} = \bar{\pi}.P$

Théorème 11 (Propriété fondamentale des chaînes de Markov)**.** *Toute chaîne de Markov finie, irréductible et apériodique a les propriétés suivantes :*

- *elle possède une distribution stationnaire unique notée* $\bar{\pi} = (\bar{\pi}_0, \ldots, \bar{\pi}_n)$,
- *pour tout* i, j, $\lim_{t \to \infty} P_{ji}^t$ *existe et est indépendante de* j.
- *pour tout* i, $\bar{\pi}_i = \lim_{t \to \infty} P_{ji}^t = \frac{1}{h_{ii}}$, *où* h_{ii} *est l'espérance du temps de retour à l'état* i.

2.2.5 Promenade aléatoire sur un graphe non orienté

Une promenade, ou marche, aléatoire sur un graphe est un cas particulièrement simple de chaîne de Markov, souvent utile dans l'analyse d'algorithmes. Dans la suite, on montre qu'une promenade aléatoire permet d'obtenir un algorithme probabiliste de complexité en temps polynomial et en espace logarithmique pour le problème de (s,t)-connexité d'un graphe.

Définition 26 (Problème de la (s,t)-connexité d'un graphe)**.**

- *Entrée :* $G = (E,R)$ un graphe non orienté, avec $|E| = n$ et $|R| = m$; ainsi que deux sommets s et t,
- *Question :* existe-t-il un chemin dans G entre s et t ?

Le problème de s-t-connexité ne s'exprime pas en logique du premier ordre, mais en logique du premier ordre avec opérateur de clôture transitive. On peut montrer que c'est un problème NL-complet. De plus, il existe un algorithme déterministe de complexité en temps $\mathcal{O}(n+m)$ et en espace $\mathcal{O}(n)$.

Définition 27. Soit $G = (E,R)$ un graphe fini, non orienté (c'est-à-dire R symétrique). On suppose $E = \{1,\ldots,n\}$ et $|R| = m$. La proemnade aléatoire sur G est défini en prenant comme états les sommets de G et comme probabilités de transition :

$$p_{ij} = \begin{cases} \frac{1}{d(i)} & \text{si } (i,j) \in R \\ 0 & \text{sinon} \end{cases}$$

où $d(i)$ est le degré du sommet i, c'est-à-dire le nombre de sommets adjacents à i.

Lemme 5. *Une promenade aléatoire sur un graphe G est apériodique ssi le graphe G est non biparti.*

Preuve. Dans un graphe non orienté, il existe toujours un chemin de longueur 2 d'un sommet vers lui-même. D'autre part, un graphe est biparti ssi il ne possède pas de cycle de longueur impaire. Si le graphe G est biparti, alors la promenade aléatoiore est périodique de période2. Si le graphe G n'est pas biparti, alors il a un cycle de longueur impaire et en traversant ce cycle, on obtient un chemin de longueur impaire d'un sommet vers lui-même.

Théorème 12. *Toute promenade aléatoire sur un graphe non orienté fini, et non biparti, possède une distribution stationnaire $\bar{\pi}$ telle que $\pi_i = \frac{d(i)}{2|R|}$ $(1 \leq i \leq n)$.*

Preuve. On montred'abord que π est bien une distribution de probabilités sur E : comme $\sum_{i\in E} d(i) = 2|R|$, $\sum_{i\in E} \pi_i = \sum_{i\in E} \frac{d(i)}{2|R|} = 1$.

Soit P la matrice de transition associée à la promenade aléatoire. La relation $\bar{\pi} = \bar{\pi}.P$ est équivalente à $\pi_i = \sum_{j\in V(i)} \frac{1}{d(j)}.\frac{d(j)}{2|R|}$, c'est-à-dire $\pi_i = \frac{d(i)}{2|R|}$. En effet, $V(i)$ est l'esnemble des sommets adjacents à i et $|V(i)| = d(i)$.

Corollaire 13. *Pour tout sommet i de G, l'espérance du temps de retour à i $h_{ii} = \frac{2|R|}{d(i)}$.*

Lemme 6. *Si $(i,j) \in R$, alors $h_{ji} < 2|R|$.*

Preuve. On peut calculer h_{ii} de deux manières diifférentes :

$$\frac{2|R|}{d(i)} = h_{ii} = \frac{1}{d(i)}. \sum_{j \in V(i)} (1 + h_{ji})$$

On obtient ainsi : $2|R| = \sum_{j \in V(i)}(1 + h_{ji})$ et $h_{ji} < 2|R|$.

Définition 28. Le temps de couverture d'un graphe est le maximum sur tous les sommets i de l'espérance du temps de visite de tous les sommets par une promenade aléatoire commençant en i.

Lemme 7. *le temps de couverture d'un graphe $G = (E,R)$ est inférieur à $4|E|.|R|$.*

Preuve. On considère un arbre couvrant G, c'est-à-dire un sous-ensemble d'arêtes qui constitue un graphe sans cycle connectant tous les sommets. Il existe un circuit sur cet arbre couvrant parcourant chaque arête une fois dans chaque direction : un tel circuit peut être trouvé par exemple par un parcours en profondeur d'abord.

Soit $s_0, s_1, \ldots, s_{2|E|-2} = s_0$ l'ensemble des sommets d'un tel circuit. L'espérance du temps de parcours de ces sommets est une borne supérieure du temps de couverture du graphe :

$$\sum_{i=0}^{2|E|-3} h_{i,i+1} < (2|E| - 2).2|R| < 4|E|.|R|$$

L'algorithme probabiliste suivant utilise $O(log_2 n)$ (n est le nombre de sommets du graphe). On suppose que le graphe n'a pas de composantes connexes biparties.

Algorithme Rand(s,t)-Connexité *Algorithme :*

1. Initialiser une promenade aléatoire au sommet s.
2. Si la promenade aléatoire atteint le sommet t en au plus $4n^3$ étapes, retourner "oui", sinon "non".

Remarques :

Théorème 14. *L'algorithme retourne une réponse correcte si il n'existe pas de chemin entre s et t. S'il existe un chemin entre s et t, alors il retourne une réponse correcte avec probabilité $\geq \frac{1}{2}$.*

Preuve. S'il existe un chemin entre s et t, l'algorithme commet une erreur s'il ne le trouve pas en au plus $4n^3$ étapes. L'espérance du temps pour atteindre t à partir de s est bornée par le temps de couverture de leur composante connexe, qui est inférieur à $4n.m \leq 2n^3$, car le nombre d'arêtes $m \leq \frac{n(n-1)}{2}\frac{n^2}{2}$. D'après l'inégalité de Markov, la probabilité de ne pas trouver de chamin en $4n^3$ étapes est inférieure à $\frac{1}{2}$.

Corollaire 15. *L'algorithme Rand(s,t)-Connexité est du type RLP.*

2.3 Bornes de Chernoff-Hoeffding

L'objectif de cette section est de montrer que l'on peut obtenir des bornes plus fines que simplement en utilisant l'inégalité de Markov pour la partie finale ("queue") d'une distribution de probabilité. De plus, ces bornes peuvent être rendues exponentiellement petites.

Théorème 16 (Borne de Chernoff-Hoeffding, 1963)**.** *Soient $X_1, \ldots, X_n$ des variables aléatoires indépendantes prenant les valeurs 1 (respectivement 0) avec probabilité p (resp. $1-p$). Soit $X = \sum_{i=1}^{n} X_i$ et $\mu = np$. Alors pour tout $0 < \varepsilon < 1$:*

$$\text{Prob}[X \geq (1+\varepsilon)\mu] \leq e^{-\frac{\varepsilon^2}{3}\mu}$$

$$\text{Prob}[X \leq (1-\varepsilon)\mu] \leq e^{-\frac{\varepsilon^2}{2}\mu}$$

Dans cet énoncé, $(1+\varepsilon)\mu$ et$(1-\varepsilon)\mu$ représentent les bornes sur l'erreur relative.

Preuve. Pour tout $t > 0$, l'évènement $X \geq (1+\varepsilon)\mu$ est équivalent à l'évènement $e^{tX} \geq k$ avec $k = e^{t.(1+\varepsilon).\mu}$.

D'après l'inégalité de Markov :

$$\begin{aligned} Prob[X \geq k] &\leq \frac{\mathbb{E}[X]}{k} \quad \text{d'où,} \\ Prob[X \geq (1+\varepsilon)\mu] &\leq \frac{\mathbb{E}[e^{tX}]}{e^{t.(1+\varepsilon).\mu}} \end{aligned}$$

La linéarité de l'espérance permet d'écrire :

$$\mathbb{E}[e^{tX}] = \mathbb{E}[e^{t\sum_{i=1}^{n} X_i}] = \mathbb{E}[\prod_{i=1}^{n} e^{tX_i}] = \prod_{i=1}^{n} \mathbb{E}[e^{tX_i}].$$

La variable X_i prenant les valeurs 1 (respectivement 0) avec probabilité p (resp. $1-p$) : $\mathbb{E}[e^{tX_i}] = (pe^t + (1-p))$. On obtient alors la borne suivante sur la

probabilité considérée :

$$\begin{aligned} Prob[X \geq (1+\varepsilon)\mu] &\leq \frac{(1+p(e^t-1))^n}{e^{t(1+\varepsilon)\mu}} \\ &\leq \frac{e^{np(1+\varepsilon)}}{e^{t(1+\varepsilon)\mu}} \text{ puisque } (1+a)^n \leq e^{an} \text{ si } a \geq 0 \\ &\leq [\frac{e^{e^t-1}}{e^{t(1+\varepsilon)}}]^\mu \end{aligned}$$

On étudie la fonction suivante pour la rendre minimum :

$$\begin{aligned} h(t) &= e^{(e^t-1)-t(1+\varepsilon)} \\ h'(t) &= e^t-(1-\varepsilon) \end{aligned}$$

donc $h'(t)=0$ si $t=\ln(1+\varepsilon)$, et $h(t)$ est un minimum de la fonction h.

La borne ainsi obtenue est :

$$Prob[X \geq (1+\varepsilon)\mu] \leq \left(\frac{e^\varepsilon}{(1+\varepsilon)^{1+\varepsilon}}\right)^\mu.$$

On désire montrer que cette borne satisfait :

$$\frac{e^\varepsilon}{(1+\varepsilon)^{1+\varepsilon}} \leq e^{\frac{-\varepsilon^2}{3}}.$$

Il suffit donc de considérer la fonction f et de montrer qu'elle est négative :

$$f(\varepsilon)=\varepsilon-(1+\varepsilon)\ln(1+\varepsilon)+\frac{\varepsilon^2}{3}.$$

$$f'(\varepsilon)=1-\frac{1+\varepsilon}{1+\varepsilon}-\ln(1+\varepsilon)+\frac{2\varepsilon}{3}.$$

ε	0		$\frac{1}{2}$		1
$f''(\varepsilon)$	$-\frac{1}{3}$	$-$	0	$+$	$\frac{1}{6}$
$f'(\varepsilon)$	0	$\searrow$		$\nearrow$	$-\ln(2)+\frac{2}{3}$
$f(\varepsilon)$	0	$-$	$\searrow$	$-$	

La fonction f est bien négative sur l'intervalle $[0,1]$.

La borne obtenue est : $Prob[X \geq (1+\varepsilon)\mu] \leq e^{-\frac{\varepsilon^2}{3}}$.

On peut faire un raisonnement analogue avec $t<0$ et obtenir :

$$Prob[X \leq (1-\varepsilon)\mu] \leq e^{-\frac{\varepsilon^2}{2}}.$$

On conclut que

$$Prob[(1-\varepsilon)\mu \leq X \leq (1+\varepsilon)\mu] \geq 1-2e^{-\frac{\varepsilon^2}{3}\mu}.$$

Si $n \geq \frac{3}{\varepsilon^2 p}\ln(\frac{2}{\delta})$ alors $2e^{-\frac{\varepsilon^2}{3}\mu} \leq \delta$.

On dispose donc de deux paramètres pour quantifier la qualité de l'estimation obtenue pour la valeur de μ :

- un paramètre d'erreur sur l'approximation : ε, par exemple 10^{-2},
- un paramètre de confiance : $(1-\delta)$ tel que $2e^{-\frac{\varepsilon^2}{3}\mu} \leq \delta$ si $n \geq \frac{2}{\varepsilon^3 \mu} \ln\left(\frac{1}{\delta}\right)$ où n est la taille de l'échantillon. Par exemple, on peut prendre $\delta = 10^{-8}$.

Théorème 17 (Borne de Chernoff-Hoeffding avec erreur absolue)**.** *Soient $X_1, \ldots, X_n$ des variables aléatoires indépendantes prenant les valeurs 1 (respectivement 0) avec probabilité p (resp. $1-p$).*

Soit $X = \frac{1}{n}\sum_{i=1}^{n} X_i$. Alors pour tout $0 < \varepsilon < 1-p$:

$$\begin{aligned} \text{Prob}[X \geq p+\varepsilon] &\leq e^{-2n\varepsilon^2} \\ \text{Prob}[X \leq p-\varepsilon] &\leq e^{-2n\varepsilon^2} \end{aligned}$$

Chapitre 3

Vérification de systèmes probabilistes

3.1 Model Checking d'un système probabiliste

3.1.1 Système de transition probabiliste

Un système de transition probabiliste est décrit par un quadruplet

$$\mathcal{M} = (S, P, s_0, L).$$

- S est un ensemble d'états (fini ou dénombrable),
- $P : S^2 \to [0, 1]$ est une fonction de transition probabiliste, c'est-à-dire telle que pour tout $s \in S$, $\sum_{t \in S} P(s, t) = 1$ (propriété de Markov),
- $s_0 \in S$ est un état initial ,
- $L : S \to 2^{AP}$ est une fonction d'étiquetage des états par des ensembles de propriétés atomiques.

Un tel système est en fait simplement une chaîne de Markov, munie d'un état initial et d'un étiquetage des états ; Dans le cas où l'ensemble des états est fini :

- P peut être représentée par une matrice de Markov,
- $\mathcal{M}$ peut être représenté par un graphe de transition probabiliste étiqueté,

Exemple de système de transition probabiliste :

Définition 29 (Chemin d'exécution)**.** Un *chemin d'exécution* d'origine s_0 est une suite infinie d'états $\sigma = (s_0, \ldots, s_i, s_{i+1} \ldots)$ telle que pour tout i : $P(s_i, s_{i+1}) > 0$.

Pour tout $i > 0$, on note σ^i le chemin $(s_i, s_{i+1} \ldots)$.

3.1.2 Mesure de probabilité sur un ensemble de chemins

L'ensemble des chemins d'origine s_0 est noté $Path(s_0)$.

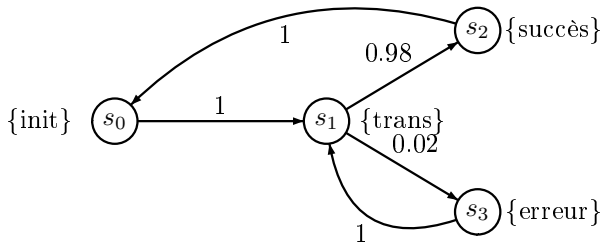


FIG. 3.1 – Système de transition probabiliste

Définition 30 (Probabilité d'un ensemble de chemins)**.** La mesure de probabilité *Prob* est définie d'abord sur les "cônes", c'est-à-dire sur les ensembles de la forme :

$$C(\pi) = \{\sigma \in Path(s_0) \mid \pi \text{ est un préfixe fini } \{s_0, \ldots, s_n\} \text{ de } \sigma\}.$$

$$Prob[\{\sigma \in Path(s_0) \mid \sigma\restriction n = \{s_0, \ldots, s_n\}\}] = \prod_{i=0}^{n-1} P(s_i, s_{i+1}),$$

La mesure de probabilité *Prob* est étendue à la σ-algèbre engendrée par les cônes. Un ensemble de cette σ-algèbre est dit *mesurable*. La mesure de probabilité *Prob* est la fonction définie de manière unique par :

- $Prob[\{\sigma \in Path(s_0)\}] = 1$,
- si $(X_i)_{i \in I}$ est une famille dénombrable d'ensembles mesurables deux à deux disjoints, $Prob[\bigcup_{i \in I} X_i] = \sum_{i \in I} Prob[X_i]$,
- si X est un ensemble mesurable, $Prob[\text{Path}(s_0) \setminus X] = 1 - Prob[X]$.

3.1.3 Logique PCTL (Hansson et Johnson)

L'ensemble des formules de chemin est défini comme en CTL à l'aide des deux opérateurs temporels X (unaire) et U (binaire). L'ensemble des formules d'état est défini comme le plus petit ensemble :

- contenant *true* et l'ensemble AP des variables propositionnelles,
- clos par les connecteurs propositionnels ($\neg$, $\vee$, $\wedge$, $\rightarrow$),
- clos par application de l'opérateur probabiliste $[\Phi]_{\geq p}$ où Φ est une formule de chemin et $p \in [0, 1]$.

La satisfaction des formules est définie comme pour CTL sauf pour l'opérateur probabiliste, qui a remplacé les quantifications existentielle et universelle sur les chemins. Etant donné un système de transition probabiliste $\mathcal{M}$ et un état s_0 :

$$(\mathcal{M}, s_0) \models [\Phi]_{\geq p} \text{ ssi } Prob[\{\sigma \in Path(s_0) \mid (\mathcal{M}, \sigma) \models \Phi\}] \geq p$$

Il est facile de vérifier (par induction sur la formule Phi) que la mesure $Prob[\{\sigma \in Path(s_0) \mid (\mathcal{M}, \sigma) \models \Phi\}]$ est bien définie.

3.1.4 Model Checking de PCTL

L'algorithme consiste à déterminer $Sat(\varphi) = \{s \in S \mid (\mathcal{M}, s) \models \varphi$ par induction sur la formule d'état φ :

$Sat(true) = S$ $Sat(a) = \{s \in S \mid a \in L(s)\}$ si $a \in AP$

$Sat(\neg\varphi) = S - Sat(\phi)$ $Sat(\varphi_1 \vee \varphi_2) = Sat(\varphi_1) \cup Sat(\varphi_2)$...

Etant donné une formule de chemin Φ, le vecteur $x = (x_s)_{s \in S}$ est le vecteur de probabilités défini par : $x_s = Prob[\{\sigma \in Path(s_0) \mid (\mathcal{M}, \sigma) \models \Phi\}]$.

Pour déterminer $Sat([X\varphi]_{\geq p}) = \{s \in S \mid x_s \geq p\}$, il suffit de résoudre le système d'équations linéaires : $x = P.b$ où $b = (b_s)_{s \in S}$ est défini par $b_s = 1$ si $(\mathcal{M}, s) \models \varphi$, 0 sinon, et de comparer ensuite à p.

Dans le cas de $Sat([\varphi_1 U \varphi_2]_{\geq p})$, on détermine d'abord :

$S_0 = Sat(\varphi_2)$, $S^+ = Sat(\exists(\varphi_1 U \varphi_2))$ comme pour le model checking de CTL, et $S^- = S - S^+$.

Soit $S' = S - (S_0 \cup S^-)$. le vecteur $x = (x_s)_{s \in S}$ est alors solution du système :

$$x_s = \begin{cases} 1 & \text{si } s \in S_0 \\ 0 & \text{si } s \in S^- \\ \Sigma_{t \in S} P(s,t).x_t & \text{si } s \in S' \end{cases}$$

Le système d'équations linéaires peut s'écrire :

$$x_s = \Sigma_{t \in S'} P(s,t).x_t + \Sigma_{t \in S_0} P(s,t)$$

$$x = A.x + P.b$$

où A est la matrice P restreinte à l'ensemble S' et $b = (b_s)_{s \in S}$ est défini par $b_s = 1$ si $s \in S_0$, 0 sinon.

La complexité de cet algorithme est en temps $0(poly(|S|).|\varphi|)$ et en espace $0(|S|^2)$. Cette méthode de model checking est utilisée par le model checker probabiliste PRISM (Université de Birmingham).

3.1.5 Vérification probabiliste (Coucourbetis, Yannakakis)

Le problème de la vérification qualitative est celui de la satisfaction d'une formule LTL par un système de transition probabiliste :

étant donnés $\mathcal{M} = (S, P, S_0, L)$ et Φ, existe-t-il un chemin σ d'origine s_0 tel que $(\mathcal{M}, \sigma) \models \Phi$, i.e. $Prob[\Phi] = Prob[\{\sigma \in \text{Path}(s_0) \mid (\mathcal{M}, \sigma) \models \Phi\}] > 0$?

Le problème de la vérification quantitative consiste à calculer $Prob[\Phi]$.

Théorème 18. *[?]*

1. *La satisfaction d'une formule LTL Φ par une chaîne de Markov $\mathcal{M}$ peut être déterminée en temps $\mathcal{O}(|\mathcal{M}|.2^{|\Phi|})$, et en espace polylogarithmique par rapport à $|\mathcal{M}$ / et polynomial par rapport à $|\Phi|$.*

2. *La probabilité de satisfaction d'une formule LTL Φ par une chaîne de Markov $\mathcal{M}$ peut être calculée en temps polynomial par rapport à $|\mathcal{M}|$, exponentiel par rapport à $|\Phi|$, et en espace polylogarithmique par rapport à $|\mathcal{M}|$ et polynomial par rapport à $|\Phi|$.*

Vérification quantitative

L'idée de la méthode conçue par C. Coucourbetis et M. Yannakakis consiste à transformer, étape par étape, la formule Φ et la chaîne de Markov $\mathcal{M}$ en éliminant un par un les opérateurs temporels de Φ et en conservant la probabilité de satisfaction de la formule Φ. Au cours de cette transformation, une propriété fondamentale des chaînes de Markov est utilisée.

Définition 31 (Composante fortement connexe terminale)**.** Une composante fortement connexe C est dite terminale s'il n'existe pas d'arête sortant de C.

Proposition 8 (Propriété fondamentale des chaînes de Markov)**.** *Presque sûrement, c'est-à-dire avec probabilité 1, tout chemin dans une chaîne de Markov atteint finalement une composante fortement connexe terminale C et visite infiniment souvent tous les états de C.*

Transformation de la chaîne de Markov

Il existe 2 sortes de transformation associées respectivement aux opérateurs temporels X et U. On décrit la transformation associée à l'opérateur U. On considère la sous-formule $(\Phi_1 U \Phi_2)$ de Φ "la plus à l'intérieur possible", c'est-à-dire les formules Φ_1, Φ_2 ne sont composées que de propositions atomiques et de connecteurs propositionnels. On peut donc évaluer les formules Φ_1, Φ_2 sur tous les états et donc déterminer $Sat(\Phi_1)$ et $Sat(\Phi_2)$. On va transformer la chaîne de Markov $\mathcal{M}$ en une chaîne $\mathcal{M}'$ et la formule Φ en une formule Φ^a sur $\mathrm{AP}' = \mathrm{AP} \cup \{a\}$ de manière à conserver la probabilité de satisfaction.

Pour cela, on effectue une partition de S en 3 sous-ensembles disjoints $S_0, S^-, S^?$tels que :

- si $s \in S_0$, alors $Prob_s[(\Phi_1 U \Phi_2)] = 1$,
- si $s \in S^-$, alors $Prob_s[(\Phi_1 U \Phi_2)] = 0$,
- si $s \in S^?$, alors $Prob_s[(\Phi_1 U \Phi_2)] > 0$ et $Prob_s[\neg(\Phi_1 U \Phi_2)] > 0$.

La construction de la partition s'effectue en 4 étapes :

1. $S_0 = Sat(\Phi_2), S^- = Sat(\neg\Phi_1 \wedge \neg\Phi_2)$; on se restreint dans la suite au sous-graphe G sur les états qui satisfont $\Phi_1 \wedge \neg\Phi_2$,
2. on met dans S^- tous les $s \in G$ tels qu'il n'existe pas dans G de chemin de s à un état t tel que $R(t,u)$ avec $u \in \mathrm{Sat}(\Phi_2)$, c'est-à-dire à partir de s il n'est pas possible d'atteindre un état $u \in Sat(\Phi_2)$ sans passer par un état satisfaisant $(\neg\Phi_1 \wedge \neg\Phi_2)$,
3. On met dans S_0 tous les $s \in G$ tels qu'il n'existe pas dans G de chemin de s à un état $t \in S^-$ ou qui possède une transition $R(t,u)$ avec $u \in \mathrm{Sat}(\neg\Phi_1 \wedge \neg\Phi_2)$, c'est-à-dire à partir de s il n'est pas possible d'atteindre un état dans S^- sans passer d'abord par un état de S_0,

4. on met dans $S^?$ les états restants, c'est-à-dire $S^? = S - (S_0 \cup S^-)$.

Lemme 8. *Soit* $p_s = \text{Prob}_s[\Phi_1 U \Phi_2]$. *Alors les probabilités* $(p_s)_{s \in S}$ *peuvent être calculées à l'aide du système d'équations linéaires :*

$$p_s = \begin{cases} 1 & si\ s \in S_0 \\ 0 & si\ s \in S^- \\ \sum_{t \in S} p(s,t).p_t & si\ s \in S^? \end{cases}$$

De plus, ce système admet une solution unique.

Preuve. On considère les quatre cas correspondant aux étapes de construction de la partition.

1. Si $s \in S_0$, $(\mathcal{M}, s) \models \Phi_2$ et $(\mathcal{M}, \pi) \models (\Phi_1 U \Phi_2)$ pour tout $\pi \in Path(s)$. Si $s \in S^-$, $(\mathcal{M}, s) \models (\neg\Phi_1 \wedge \neg\Phi_2)$ et $(\mathcal{M}, \pi) \models \neg(\Phi_1 U \Phi_2)$ pour tout $\pi \in Path(s)$.
2. Si $s \in S^-$, on prend $\pi \in Path(s)$:
   (a) soit π reste dans G, ne visite jamais un état satisfaisant Φ_2, et donc $(\mathcal{M}, \pi) \not\models (\Phi_1 U \Phi_2)$,
   (b) soit π sort de G en un état satisfaisant $(\neg\Phi_1 \wedge \neg\Phi_2)$.

   Dans les 2 cas, $(\mathcal{M}, \pi) \models \neg(\Phi_1 U \Phi_2)$.
3. Si $s \in S_0$, on prend $\pi \in \text{Path}(s)$. Presque surement, π atteint finalement une composante fortement connexe terminale C et visite infiniment souvent tous les états de C. On suppose que π ne sort jamais de G et donc $C \subseteq G$. Par conseéquent, tous les états de C ont été classés dans S^- à la 2e étape, ce qui contredit le fait que $s \in S_0$. De fait, π sort de G par une transition $R(t,u)$ vers un état $u \in Sat(\Phi_2)$ et par suite, $(\mathcal{M}, s) \models (\Phi_1 U \Phi_2)$.
4. Si $s \in S^?$, alors $(\mathcal{M}, \pi) \models (\Phi_1 \wedge \neg\Phi_2)$. Soit $\pi \in Path(s)$:
   $(\mathcal{M}, \pi) \models (\Phi_1 U \Phi_2)$ ssi $(\mathcal{M}, \pi^1) \models (\Phi_1 U \Phi_2)$
   Si t est le premier état de π^1, alors soit $p_t = Prob_t[\Phi_1 U \Phi_2]$. Il est facile de voir que : $p_s = \sum_{t \in S} P(s,t).p_t$

On construit ensuite une nouvelle chaîne de Markov $\mathcal{M}' = (S', P', L')$ et on rajoute une proposition atomique a : $\text{AP}' = \text{AP} \cup \{a\}$ pour pouvoir remplacer $(\Phi_1 U \Phi_2)$ par a dans la formule principale Φ.

Les états de $\mathcal{M}'$ sont définis en prenant :
– pour chaque état $s \in S_0$, un état $(s, a) \in S'$,
– pour chaque état $s \in S^-$, un état $(s, \neg a) \in S'$,
– pour chaque état $s \in S^?$, deux états $(s, a) \in S'$ et $(s, \neg a) \in S'$.

La distribution initiale p'_0 de $\mathcal{M}'$ est définie à partir de la distribution initiale p_0 de $\mathcal{M}$ par :
- si $s \in S_0 \cup S^-$, $p'_0((s, \varepsilon a)) = p_0(s)$,
- si $s \in S^?$, $p'_0((s,a)) = p_0(s).p_s$ et $p'_0((s, \neg a)) = p_0(s)(1 - p_s)$.

Les probabilités de transition dans $\mathcal{M}'$ sont définies par :

- si $s,t \in S_0 \cup S^-$, $P'((s,\varepsilon a),(t,\varepsilon' a)) = P(s,t)$,
- si $s \in S_0 \cup S^-$ et $t \in S^?$,
  $P'((s,\varepsilon a),(t,a)) = P(s,t).p_t$ et $P'((s,\varepsilon a),(t,\neg a)) = P(s,t).(1-p_t)$,
- si $s,t \in S^?$,
  $P'((s,a),(t,a)) = P(s,t).\left(\frac{p_t}{p_s}\right)$ et $P'((s,\neg a),(t,\neg a)) = P(s,t).\left(\frac{1-p_t}{1-p_s}\right)$,
- si $s \in S^?$ et $t \in S_0$, $P'((s,a),(t,a)) = P(s,t)\frac{1}{p_s}$,
- si $s \in S^?$ et $t \in S^-$, $P'((s,\neg a),(t,\neg a)) = P(s,t)\frac{1}{(1-p_s)}$.

La fonction d'étiquetage des états de $\mathcal{M}'$ est définie comme suit :
$L'^{-1}(a) = \{(s,a) \mid s \in S\}$,
$L'^{-1}(b) = \{(s,a),(s,\neg a) \mid s \in L^{-1}(b)\}$ pour tout $b \in AP$.

La nouvelle formule Φ^a est obtenue en remplaçant toutes les occurrences de $(\Phi_1 U \Phi_2)$ par a dans la formule Φ.

Lemme 9. *Tout chemin $\pi' = (s'_0,\ldots,s'_i,\ldots)$ de la nouvelle chaîne $\mathcal{M}'$ satisfait la propriété suivante avec probabilité 1 :*

$$(\forall i \geq 0)\ s'_i \textit{ satisfait } a \Leftrightarrow (\mathcal{M}', \pi'^i) \models \Phi_1 U \Phi_2.$$

Preuve. Il suffit de montrer que :

– Si $s'_0 = (s_0,a)$ alors $Prob[(\mathcal{M}',\pi') \models \Phi_1 U \Phi_2] = 1$,

– Si $s'_0 = (s_0,\neg a)$ alors $Prob[(\mathcal{M}',\pi') \models \neg(\Phi_1 U \Phi_2)] = 1$,

Etant donné un chemin π' dans $\mathcal{M}'$, soit π le chemin dans $\mathcal{M}$ obtenu en projetant les états de π' sur leur première composante.

Premier cas : $s'_0 = (s,\neg a)$ et $s \in S^-$.

Soit G'_1 le sous-graphe de $\mathcal{M}'$ restreint aux états dont la première composante est dans S^- :

- si $(\mathcal{M},s) \models (\neg\Phi_1 \wedge \neg\Phi_2)$ alors $(\mathcal{M}',\pi') \models \neg(\Phi_1 U \Phi_2)$,
- si $(\mathcal{M},s) \models (\Phi_1 \wedge \neg\Phi_2)$ alors :
  – soit π' ne sort pas de G'_1 et reste dans une composante connexe terminale dont tous les états satisfont $(\Phi_1 \wedge \neg\Phi_2)$,
  – soit π' sort de G'_1 en un état qui satisfait $(\neg\Phi_1 \wedge \neg\Phi_2)$.

Deuxième cas : $s'_0 = (s,a)$ et $s \in S_0$.

Soit G'_2 le sous-graphe de $\mathcal{M}'$ restreint aux états dont la première composante est dans S_0 :

- si $(\mathcal{M},s) \models \Phi_2$, alors $(\mathcal{M}',\pi') \models (\Phi_1 U \Phi_2)$,
- si $(\mathcal{M},s) \models \Phi_1 \wedge \neg\Phi_2$, alors il n'existe pas de composante fortement connexe terminale dans G'_2, composée d'états satisfaisants $(\Phi_1 \wedge \neg\Phi_2)$ puisqu'une telle composante serait dans G'_1 ; avec probabilité 1, tout chemin π issu de s'_0 visite un état satisfaisant Φ_2 et satisfait $(\Phi_1 U \Phi_2)$.

Troisième cas : $s'_0 = (s,\varepsilon a)$ avec $s \in S^?$.

Soit G'_3, respectivement G'_4, le sous-graphe induit sur les états (s,a), respectivement $(s,\neg a)$, avec $s \in S^?$. Ces sous-graphes ne contiennent pas de composante fortement connexe terminale de $\mathcal{M}'$ puisqu'une telle composante serait déjà dans G'_1. On remarque que :

- toute transition à l'extérieur de G'_3 apparaît sur un état de G'_2,
- toute transition a l'extérieur de G'_4 apparaît sur un état de G'_1.

Soit π' un chemin partant de G'_3. Avec probabilité 1, ce chemin visite un état de G'_2 avec tous les états précédents dans G'_3. Comme tout état dans G'_3 satisfait $(\Phi_1 \wedge \neg\Phi_2))$ et que presque tous les chemins partant de G'_2 satisfont $(\Phi_1 U \Phi_2)$, le chemin π' satisfait $(\Phi_1 U \Phi_2)$ avec probabilité 1.

Lemme 10. *Les chemins π et π' ont la même distribution de probabilité.*

Preuve. Il suffit de le montrer pour les enembles de base $C(\rho) = \{\pi \in Path(s) | \rho = (s'_0, \ldots, s'_n)$ préfixe fini de $\pi\}$.

On remarque que si $(t, \varepsilon a)$ est un état de $\mathcal{M}'$ et qu'il existe une transition $R(s,t)$, alors $(t, \varepsilon a)$ a exactement un prédécesseur dans $\mathcal{M}'$ de première composante s.

Partant d'un état $s'_n = (s_n, \varepsilon a)$ on procède « en arrière » pour obtenir un unique chemin $s'_0, \ldots, s'_n$ avec projection $s_0, \ldots, s_n$ dans $\mathcal{M}$.

Il est alors facile de vérifier que :

- si $\varepsilon a = a$, $Prob_{\mathcal{M}'}(s'_0, \ldots, s'_n) = Prob_{\mathcal{M}}(s_0, \ldots, s_n).p_{s_n}$,
- si $\varepsilon a = \neg a$, $Prob_{\mathcal{M}'}(s'_0, \ldots, s'_n) = Prob_{\mathcal{M}}(s_0, \ldots, s_n).(1 - p_{s_n})$,

Proposition 9. *La probabilité de satisfaction de la formule Φ dans $\mathcal{M}$ est préservée, c'est-à-dire :*

$$\mathrm{Prob}_{\mathcal{M}}[\{\pi \in Path(s_0) \mid (\mathcal{M}, \pi) \models \Phi\}] = \mathrm{Prob}_{\mathcal{M}'}[\{\pi' \in Path(s'_0) \mid (\mathcal{M}, \pi') \models \Phi^a\}]$$

Preuve. Comme les chemins π et π' ont même distribution de probabilités :

$$Prob_{\mathcal{M}}[\Phi] = Prob_{\mathcal{M}'}[\Phi] = Prob_{\mathcal{M}'}[\Phi^a]$$

La second égalité est une conséquence du fait que la formule $(\Phi_1 U \Phi_2) \equiv a$ sur tout état des

3.1.6 Classe RLP

chemins de $\mathcal{M}'$ avec probabilité 1.

Complexité de l'algorithme

Les différentes étapes de l'algorithme d'élimination de la sous-formule $(\Phi_1 U \Phi_2)$ sont les suivantes.

1. Évaluer Φ_1 et Φ_2 sur tout état de $\mathcal{M}$.
2. Initialiser S_0 à $\mathrm{Sat}(\Phi_2)$ et S^- à $\mathrm{Sat}(\neg\Phi_1 \wedge \neg\Phi_2)$.
3. Soit G le sous graphe induit sur les états satisfaisants $(\Phi_1 \wedge \neg\Phi_2)$. Calculer les composantes fortement connexes de G.
4. Traiter les composantes de façon *bottom-up* : pour toute composante fortement connexe terminale D de G, répartir les états de D dans S_0, S^- et $S^?$ et supprimer D de G :

(a) s'il n'existe pas d'arête de $\mathcal{M}$ sortant de D ou si toute arête sortante va à un état de S^-, mettre D dans S^-,

(b) s'il existe une arête de $\mathcal{M}$ sortant de D et si toute arête sortante va à un état de S_0, mettre D dans S^0,

(c) sinon mettre D dans S^-.

Complexité en temps :

Soit t le nombre de connecteurs booléen de $(\Phi_1 U \Phi_2)$. La taille de la formule Φ^a est $|\Phi^a| = |\Phi| - t$. Alors en reprenant la numérotation des étapes de l'élimination sous-formule $(\Phi_1 U \Phi_2)$:

- l'étape 1 s'effectue en temps $\mathcal{O}(t.|\mathcal{M}|)$,
- l'étape 2 s'effectue en temps $\mathcal{O}(|\mathcal{M}|)$,
- l'étape 3 s'effectue en temps $\mathcal{O}(|\mathcal{M}|)$.

La construction de la nouvelle chaîne $\mathcal{M}'$ se fait en temps $\mathcal{O}(t.|\mathcal{M}|)$ et l'élimination de $\Phi_1 U \Phi_2$ *au plus* double le nombre d'états et d'arêtes.

Le temps de construction de la chaîne finale est donc en $\mathcal{O}(2^{|\Phi|}.|\mathcal{M}|)$.

Complexité en espace :

La construction du graphe de la nouvelle chaîne $\mathcal{M}'$ utilise un espace en $\mathcal{O}(|\Phi| + \log^2(|\mathcal{M}|))$. En effet, le problème d'accessibilité dans un graphe à n sommets utilise un espace $O(\log^2 n)$.

L'algorithme récursif permettant d'engendrer les états et les transitions de la chaîne finale possède les caractéristiques suivantes :

- la profondeur de récursion est $|\Phi|$,
- la taille de la chaîne de Markov (au plus) double à chaque transformation.

L'espace nécessaire est donc :

$$\mathcal{O}(|\Phi|.(|\Phi| + \log^2(2^{|\Phi|}.|\mathcal{M}|))) = \mathcal{O}(|\Phi|^3 + |\Phi|.\log^2(|\mathcal{M}|)).$$

3.2 Méthodes d'approximation pour la vérification probabiliste

L'objectif de cette section est de montrer plusieurs types de résultats :

1. l'un de type négatif qui montre que l'existence d'un algorithme probabiliste d'approximation (relative) en temps polynomial pour le calcul de la probabilité de satisfaction d'une formule LTL entrainerait l'existence d'un algorithme probabiliste pour décider tout problème de la classe NP,
2. l'autre de type positif qui montre l'existence d'un algorithme probabiliste d'approximation (absolue) en espace logarithmique pour le calcul de la probabilité de satisfaction des formules LTL exprimant des classes de propriétés intéressantes (monotones, anti-monotones),
3. cette dernière méthode d'approximation permettant la conception et l'implantation d'un model checker probabiliste (APMC : "Approximate Probabilistic Model Checker")

3.2.1 Problèmes de comptage et approximation

Intuitivement, la classe $\sharp P$ rend compte des problèmes de comptage pour le nombre de solutions des problèmes de la classe NP. Les problèmes de comptage associés à tous les problèmes NP-complets connus sont $\sharp P$-complets.

On introduit la notion de schéma probabiliste d'approximation en temps pleinement polynomial qui est du à R. Karp et Luby [**?**]. On considère un problème de comptage et soit $\#(x)$ le nombre de solutions distinctes pour une instance x de ce problème. La taille de cette instance est notée $|x|$.

Définition 32. Un schéma probabiliste d'approximation (RAS) pour un problème de comptage est un algorithme probabiliste $\mathcal{A}$ qui prend en entrée x, et 2 nombres réels $\varepsilon, \delta > 0$, et produit en sortie une valeur $A(x, \varepsilon, \delta)$ telle que :

$$Pr\big(|A(x,\varepsilon,\delta) - \#(x)| \leq \varepsilon.\#(x)\big) \;\geq\; 1-\delta.$$

Un schéma probabiliste d'approximation est dit pleinement polynomial (FPRAS) si son temps de calcul est borné par un polynôme en $|x|$, $\frac{1}{\varepsilon}$ et $\log(\frac{1}{\delta})$.

La probabilité Pr est définie sur les choix aléatoires de l'algorithme. L'algorithme prend non seulement en entrée la donnée du problème, mais aussi 2 paramètres : ε le *paramètre d'approximation* et $(1-\delta)$ le *paramètre de confiance*. L'approximation est dite relative car ε est un paramètre multiplicatif qui prend en compte la valeur de $\#(x)$.

On adapte la notion de schéma probabiliste d'approximation aux problèmes de calcul de probabilités. Un problème de calcul de probabilités est défini par la donnée d'une representation succincte d'un système probabiliste et d'une propriété x. On désire calculer la measure de probabilités $\mu(x)$ de l'ensemble des chemins d'exécution satisfaisant cette propriété.

Définition 33. Un schéma probabiliste d'approximation (RAS) pour un problème de probabilités est un algorithme probabiliste $\mathcal{A}$ qui prend en entrée x, et 2 nombres réels $\varepsilon, \delta > 0$ et produit en sortie une valeur $A(x, \varepsilon, \delta)$ telle que :

$$Pr\big(|A(x,\varepsilon,\delta) - \mu(x)| < \varepsilon.\mu(x)\big) \;\geq\; 1-\delta.$$

Si le temps de calcul de $\mathcal{A}$ est polynomial en $|x|$, $\frac{1}{\varepsilon}$ et $\log(\frac{1}{\delta})$, $\mathcal{A}$ est dit pleinement polynomial.

Théorème 19. *S'il existait un schéma probabiliste d'approximation pleinement polynomial pour le problème de calcul de la probabilité $Prob[\psi]$ de satisfaction des formules ψ de LTL , alors $NP \subseteq BPP$.*

Preuve. On considère le fragment $L(\mathbf{F})$ de LTL dans lequel $\mathbf{F}$ est le seul opérateur temporel. Le résultat suivant est dû à Clarke et Sistla [**?**] : le problème de décider l'existence d'un chemin satisfaisant une formule de $L(\mathbf{F})$ dans un systéme de transition est NP-complet. Leur preuve utilise une réduction en temps polynomial de SAT au probléme de décider la satisfaction des formules de $L(\mathbf{F})$.

A partir de cette réduction, on peut obtenir une réduction, "solution à solution", du probléme $\sharp SAT$ au problème de comptage des chemins finie, de longueur donnée, dont toutes les extensions infinies satisfont la formule associée $L(\mathbf{F})$. Soit $\bigwedge_{j=1}^{m} c_j$ une formule SAT , sur les variables propositionnelles $\{x_i \ / \ i = 1, \ldots, n\}$. On construit le système de transition probabiliste suivant $\mathcal{M} = (S, P, y_0, L)$:

- $AP = \{c_j \ / \ j = 1, \ldots, m\}$ (ensemble de propositions atomiques),
- $S = \{x_i \ / \ i = 1, \ldots, n\} \cup \{x'_i \ / \ i = 1, \ldots, n\} \cup \{y_i \ / \ i = 0, \ldots, n\}$,
- $P(y_{i-1}, x_i) = 1/2$, $P(y_{i-1}, x'_i) = 1/2$ pour $i = 1, \ldots, n$,
  $P(x_i, y_i) = 1$, $P(x'_i, y_i) = 1$ pour $i = 1, \ldots, n$, $P(y_n, y_n) = 1$,
- $L(x_i) = \{c_j \ / \ x_i \text{ appears in } c_j\}$ pour $i = 1, \ldots, n$,
  $L(x'_i) = \{c_j \ / \ \neg x_i \text{ appears in } c_j\}$ pour $i = 1, \ldots, n$.

Maintenant si l'on considére la formule LTL $\bigwedge_{j=1}^{m} \mathbf{F} c_j$, il est facile de voir que toute valuation satisfaisant $\bigwedge_{j=1}^{m} c_j$ correspond à un chemin fini allant de y_0 à y_n, dont toute extension infinie satisfait la formule LTL $\bigwedge_{j=1}^{m} \mathbf{F} c_j$. Par conséquent, le problème de comter les chemins finis , de longueur $2n$, satisfaisant une formule de $L(\mathbf{F})$ est $\sharp P$-complet.

Une conséquence de ce résultat est le caractère $\sharp P$-dur du calcul des probabilités de satisfaction des formules LTL. On remarque ainsi que s'il existait un schéma probabiliste d'approximation pleinement polynomial pour le calcul de $Prob[\psi]$ (ψ formule LTL), on pourrait approximer de manière efficace $\sharp SAT$.

Un schéma probabiliste d'approximation en temps polynomial pour $\sharp SAT$ pourrait alors être utilisé pour distinguer, étant donné une entrée x, entre le cas $\sharp(x) = 0$ et the cas $\sharp(x) > 0$, ce qui fournirait un algorithme probabiliste en temps polynomial pour le problème de décision SAT.□

D'après une conséquence d'un résultat de Jerrum, Valiant et Vazirani [?] et une remarque de Sinclair [?], l'existence d'un schéma probabiliste d'approximation pleinement polynomial pour $\sharp SAT$ entrainerait $RP = NP$, où RP est la classe des problèmes de décision qui possèdent un algorithme probabiliste en temps polynomial, avec erreur d'un seul côté.

3.2.2 Approximation pour le model checking probabiliste

Dans cette section, on considère certaines restrictions sur la classe des formules LTL, et sur le type d'approximation (absolue), permettant d'obtenir des schémas probabilistes d'approximation efficaces pour le calcul des probabilités de satisfaction.

Pour de nombreuses propriétés intéressantes, comme l'accessibilité, la satisfaction par un chemin d'execution de longueur finie entraîne la satisfaction par toute extension de ce chemin. De telles propriétés sont appellées monotones. Une autre importante classe de propriétés, comme les propriétés de sûreté, peut être exprimée comme négation de propriétés monotones. On peut ramener le calcul de la probabilité de satisfaction d'une propriété de sûreté au problème pour sa négation, qui est une propriété monotone.

Dans le but d'approximer la probabilité de satisfaction $Prob[\psi]$ d'une propriété monotone, on considère d'abord $Prob_k[\psi]$, la mesure de probabilité associée à l'espace probabiliste des chemins d'exécution de longueur finie k. Le caractère monotone de la propriété considérée permet d'obtenir le résultat suivant.

Proposition 10. *Soit ψ une formule LTL exprimant une propriété monotone et $\mathcal{M}$ un système de transition probabiliste. Alors la suite $(Prob_k[\psi])_{k\in\mathbb{N}}$ converge vers $Prob[\psi]$.*

La méthode consiste à engendrer aléatoirement des chemins de longueur k dans l'espace probabilistie associé et à calculer une variable aléatoire Y qui approxime $p = Prob_k[\psi]$. La qualité de l'approximation sera correcte, i.e $|Y-p| < \varepsilon$ (erreur relative), avec confiance $(1-\delta)$, après un échantillonnage de taille polynomiale en $\frac{1}{\epsilon}, \log\frac{1}{\delta}$. Ce résultat est obtenu à l'aide de la borne de Chernoff-Hoeffding sur la partie terminale ("queue") de la distribution de probabilités d'une somme de variables aléatoires independantes.

Le principal avantage de la méthode est qu'elle n'utilise seulement qu'une representation succincte du système de transition dans un language d'entrée, comme par exemple *Reactive modules* (PRISM), éliminant ainsi le problème de complexité en espace, dû au phénomène d'explosion de l'espace des états.

Définition 34. Une représentation succincte, ou diagramme, d'un système de transition probabiliste $\mathcal{M} = (S, P, s_0, L)$ est une représentation qui permet d'engendrer aléatoirement, pour tout état s, un état t tel que $P(s,t) > 0$.

La taille d'une représentation succincte est beaucoup plus petite que celle du système correspondant. Par exemple, une representation succincte peut être fournie par programme représentant le système dans le language d'entrée d'un models checkers probabiliste tel que PRISM or APMC. De manière typique la taille d'une telle représentation succincte est polylogarithmique dans la taille du système.

La fonction suivante **Random Path** utilise une représentation succincte pour engendrer aléatoirement un chemin de longueur k et vérifier la formule ψ sur ce chemin :

Random Path
Input : $diagram_{\mathcal{M}}, k, \psi$
Output : valeur de la formule ψ sur un chemin aléatoire π de longueur k

1. Engendrer aléatoirement un chemin π de longueur k
2. Si ψ est vraie sur π alors retourner 1 sinon 0

On considère maintenant l'algorithme d'échantillonnage $\mathcal{GAA}$ conçu pour l'approximation de $Prob_k[\psi]$:

Generic approximation algorithm $\mathcal{GAA}$
Input : $diagram_{\mathcal{M}}, k, \psi, \epsilon, \delta$
Output : approximation de $Prob_k[\psi]$
$N := \ln(\frac{2}{\delta})/2\varepsilon^2$
$A := 0$
For $i = 1$ to N do
$A := A +$ **Random Path**$(diagram_{\mathcal{M}}, k, \psi)$
Return $Y = A/N$

Théorème 20. *L'algorithme d'approximation générique $\mathcal{GAA}$ est un schéma probabiliste d'approximation pleinement polynomial, avec erreur absolue, pour le calcul de $p = Prob_k[\psi]$ lorsque ψ est une formule LTL exprimant une propriété monotone et $p \in]0,1[$.*

Preuve. On désire estimer $p = Prob_k[\psi]$ par un échantillonnage aléatoire de taille N. La variable aléatoire A est la somme de N variables aléatoires indépendantes suivant chacune une distribution de Bernouilli de probabilité p. On utilise la borne de Chernoff-Hoeffding pour borner la probabilité d'erreur de l'algorithme. Soit $X_1, ..., X_N$ N variables aléatoires indépendantes qui prennent la valeur 1 avec probabilité p et 0 avec probabilité $(1-p)$, et $Y = \sum_{i=1}^{N} X_i/N$. Alors la borne de Chernoff donne :

$Prob\big(Y < (p-\varepsilon)\big) < e^{-2N\cdot\varepsilon^2}$ et $Prob\big(Y > (p+\varepsilon)\big) < e^{-2N\cdot\varepsilon^2}$.

Dans l'algorithme précédent, si $N \geq \frac{\log(\frac{2}{\delta})}{2\varepsilon^2}$, alors :
$Prob\big(p-\varepsilon) \leq A/N \leq (p+\varepsilon)\big) \geq 1-\delta$. □

On peut obtenir un schéma probabiliste d'approximation pour $Prob[\psi]$ en itérant l'algorithme précédent. Le principal avantage de l'approximation ainsi obtenue est que la complexité en espace est logarithmique.

Corollaire 21. *L'algorithme de point fixe obtenu en itérant l'algorithme d'approximation $\mathcal{GAA}$ est un schéma probabiliste d'approximation, dont la complexité en espace est logarithmique, pour le calcul de $p = Prob[\psi]$ lorsque la formule ψ est "monotone" et $p \in]0,1[$.*

Preuve. Pour calculer une ε-approximation de $Prob_k[\psi]$, on doit stocker les $N := \ln(\frac{2}{\delta})/2\varepsilon^2$ résultats booléens de la fonction **Random Path**. L'espace nécessaire est $logN$ et est indépendant d'une borne appropriée sur la longueur k des chemins d'exécution engendrés pour obtenir une bonne approximation de $Prob[\psi]$.

Dans la suite, on rappelle quelques faits de base de la théorie des chaînes de Markov et on donne un critère sur la vitesse de convergence de la méthode dans le cas important des chaînes de Markov irréductibles et aperiodiques. Soit P la matrice de transition d'une chaîne de Markov finie, en temps discret. Une chaîne de Markov est dite irréductible si le graphe de transition sous-jacent est fortement connexe. Une chaîne de Markov est dite primitive si l'une des puissances de la matrice de transition a tous ses éléments positifs. Il est bien connu qu'une chaîne de Markov est primitive ssi elle est irréductible et aperiodique.

Proposition 11. *Soit P la matrice de transition d'une chaîne de Markov primitive et v un vecteur de probabilités arbitraire. Alors $lim_{k\to\infty} vP^k = u$ où u est l'unique vecteur propre gauche associé à la valeur propre $\lambda = 1$.*

Si l'on démarre une chaîne de Markov avec probabilités initiales données par v, alors le vecteur de probabilités $v.P^k$ fournit les probabilités d'être dans les différents états après k étapes. La proposition précédente établit le fait que la probailité d'être alors dans l'état s_j est u_j.

Le théorème suivant, connu sous le nom de théorème de Perron-Frobenius , est un résultat classique de la théorie des matrices.

Théorème 22. *Soit P une matrice primitive non négative d'ordre n. Alors il existe une valeur propre relle λ_1 de multiplicité 1 telle que $\lambda_1 > 0$ et $\lambda_1 > |\lambda_j|$ pour toute autre valeur propre λ_j. Soit $\lambda_2, \lambda_3, \ldots, \lambda_n$ les valeurs propres de P autres que λ_1 ordonnées de manière que $\lambda_1 > |\lambda_2| \geq |\lambda_3| \geq \cdots \geq |\lambda_n|$ et que, si $|\lambda_2| = |\lambda_j|$ pour un $j \geq 3$, alors $m_2 \geq m_j$, où m_j est la multiplicité de λ_j. Alors*

$$P^k = \lambda_1^k vu + O(k^{m_2-1}|\lambda_2|^k)$$

où u, v sont les vecteurs propres gauche et droit associés à λ_1.
Si de plus, P est une matrice de Markov, alors $\lambda_1 = 1$.

Ainsi le taux de convergence est géométrique et le problème est de déterminer k tel que $O(k^{m_2-1}|\lambda_2|^k) \leq \varepsilon$. Si la multiplicité de la seconde valeur propre λ_2 est 1, alors le nombre k d'itérations suffisant pour obtenir une ε-approximation est $O(ln(1/\varepsilon))$. En général, on ne connaît pas de critère de convergence en temps polynomial. Le model checker probabiliste APMC, qui implémente la méthode probabiliste d'approximation précédente, utilise un critère d'arrêt pratique. Le nombre d'itérations peut être exponentiel dans la taille d'une representation succincte du modèle. Cependant, la complexité en espace est logarithmique et la méthode d'approximation peut être utilisée en pratique pour verifier des systèmes de très grande taille.