Théorème de Rice sur les ensembles récursivement énumerables

Wenjie Fang

22 décembre 2009

Résumé

Dans le domaine des languages récursivement énumérables, il y a des propriétés diverses. Parfois on veut savoir si un language procède une certaine propriété ou pas. Les deux théorèmes de Rice nous donnenet les conditions pour que ce soit possible. Seules les propriétés triviales sont décidables. La condition nécessaire et suffisante pour que une propriété soit récursivement énumérable est beaucoup plus compliquée.

1 Préliminaire

On abordera les deux théorèmes de Rice sur les ensembles récursivement énumerables. C'est bien discuté dans [1], d'où la démonstration est extraite.

Pour simplifier la discussion, on choisit $\Sigma = (0+1)^*$ comme l'alphabet pour les languages récursivement énumérables (désormais noté r.é.). L'universalité de ce choix est justifiée par le fait que tout alphabet fini peut être codé en binaire de façon calculable. On note l'ensemble des languages r.é. $\mathcal{R}$.

Définition 1. Une propriété $\mathcal{P}$ est un sous-ensemble de $\mathcal{R}$. Un language L est dit d'avoir une propriété $\mathcal{P}$ si $L \in \mathcal{P}$.

On impose quelques contraintes sur les machines de Turing pour simplifier la discussion.

1. Toute machine de Turing considérée ici travaille sur un autre alphabet $\Gamma \supset \Sigma$, qui contient un symbole spécial $\#$ pour les cases vides.
2. Toute machine de Turing est codée dans l'alphabet Σ.
3. On confond tout modèle de machine de Turing grâce à l'équivalence entre ces modèles. Toute machine est codée dans sa version avec une seule bande infinie.
4. On suppose que toute machine de Turing est déterministe, car on peut toujours les déterminiser. On supposer aussi que toute machine de Turing accepte une entrée dans un seul état et la rejète dans un autre.
5. Le codage d'une machine de Turing M est noté $\langle M \rangle$. Le codage doit être naturel, donc on peut supposer qu'il existe une machine de Turing universelle MTU qui reçoit $\langle M, w \rangle = \langle \langle M \rangle, w \rangle$, qui simule M sur l'entrée w et, en cas d'arrêt, s'arrête dans l'état acceptant à la tête de la bande en effaçant toute la bande.

On impose la surjection naturelle $\mathcal{L}$ de l'ensemble des codages de machines de Turing à l'ensemble des languages r.é. $\mathcal{R}$ qui associe un codage $\langle M \rangle$ au language reconnu par M, noté $\mathcal{L}(\langle M \rangle)$.

Cette surjection nous permet de discuter les language r.é. dans le cadre de machine de Turing et son codage.

Définition 2. Une propriété $\mathcal{P}$ est dite décidable (resp. r.é.) si $\mathcal{L}^{-1}(\mathcal{P}) \subset \Sigma^*$ est un language décidable (resp. r.é.).

$\mathcal{L}^{-1}(\mathcal{P})$ est en fait l'ensemble de codage dont la machine de Turing associée reconnaît un language dans $\mathcal{P}$.

Après cet encadrement, on peut énoncer les deux théorèmes de Rice.

Théorème 1 (Le Théorème de Rice pour décidabilité)**.** Seules les propriétés triviales sont décidables.

Théorème 2 (Le Théorème de Rice pour décidabilité partielle)**.** Une propriété $\mathcal{P}$ est r.é. si et seulement si elle vérifie les trois conditions suivantes :

1. Le sous-ensemble des languages finis dans $\mathcal{P}$ est r.é..
2. Si un language infini $L \in \mathcal{P}$, il existe un language fini L' contenu dans L qui est dans $\mathcal{P}$.
3. Si un language $L \in \mathcal{P}$ et un autre language $L' \in \mathcal{R}$ contenant L, alors $L' \in \mathcal{P}$.

Dans la démonstration suivante, on manipulera beaucoup les codages de machine. Il sera plus favori de justifier ces manipulations avant de les utiliser.

2 Manipulation sur le codage de machine de Turing

Pour simplifier la justification, on constate facilement que en ajoutant un nouveau symbole $* \in \Gamma$ qui signifie le début de la bande, on peut rendre facilement tout machine de s'arrêter au début de la bande à la fin du calcul.

Définition 3. Une machine est dite régulière si elle s'arrête toujours au début de la bande, c'est-à-dire le symbole $*$ correspondant.

Donc on peut supposer que toute machine que l'on va construire est régulière.

On va construire quelques machines que l'on utilisera dans la suite.

Proposition 1. Il existe une machine qui reçoit le codage d'une machine et qui renvoye le codage de la machine régulière correspondante.

Démonstration. Il suffit d'ajouter deux états qui se bougent à gauche jusqu'àu symbole $*$. On branche l'état acceptant à l'un de ces états bouclants et le rend acceptant. On fait le même chose pour l'état rejetant à un autre état bouclant. Toute opération se fait facilement par une machine. □

Proposition 2. Il existe une machine Imp qui reçoit un mot w et qui renvoye le codage d'une machine régulière qui l'imprime sur la bande et s'arrête en l'état acceptant.

Démonstration. Il suffit de construit une machine avec autant états que la longeur du mot qui imprime une lettre du mot à chaque état, avec le dernier état acceptant. C'est facilement réalisé par une machine. Puis on rend facilement cette machine régulière. □

Proposition 3. Il existe une machine *Conc* qui reçoit $\langle M_1 \rangle$ et $\langle M_2 \rangle$, et qui renvoye $\langle M_1 M_2 \rangle$, qui est le codage d'une machine régulière qui fait la chose suivante : elle applique M_1 sur l'entrée et, en cas acceptant, applique M_2 sur ce qui reste sur la bande.

Démonstration. Grâce à la proposition 1, on peut d'abord régulariser ces deux machines, puis les fusionner. Dans cette manipulation, on fait identifier l'état acceptant de M_1 à l'état initial de M_2 et l'état rejetant de M_1 à celui de M_2. L'état acceptant et rejetant de cette nouvelle machine sont respectivement ceux de M_2. Le nouvel état initial est celui de M_1. Toute opération se fait facilement par une machine. □

Proposition 4. Il existe une machine *Cond* qui reçoit $\langle M_1 \rangle$ et $\langle M_2 \rangle$, et qui renvoye le codage d'une machine qui fait marcher M_2 sur la bande si M_1 accepte le mot vide.

Démonstration. D'abord on régularise les deux codages donnés à l'aide de la proposition 1. Puis on ajoute un état initial qui retrouve la fin de l'entrée, qui y ajoute le symbole $*$ pour créer une bande virtuelle et qui passe à l'état initial de M_1. Puis on branche l'état final de M_1 à un nouvel état qui nettoie le symbole $*$ et puis à un nouvel état qui retrouve le début de la bande, en fin on le branche sue l'état initial de M_2. On identifie les états rejetants de ces deux machines. Tout se fait par une machine. □

Proposition 5. Il existe une machine *Para* qui reçoit $\langle \langle M_1 \rangle \langle M_2 \rangle \rangle$ et qui renvoye le codage d'une machine qui fait marcher "parallèlement" ces deux machines sur la même entrée en effectuant à chaque fois une transition pour chaque machine. Cette nouvelle machine accepte un mot si et seulement si les deux l'acceptent.

Démonstration. On utilisera la même technique pour montrer l'équivalence entre machine à une seule bande et machine à plusieures bandes. On fait le produit cartésien des états, on duplique ce produit pour distiguer le début et la fin d'une transition, et on règle les transitions selon ce nouvel ensemble d'états. Puis on ajoute des transitions pour que l'on puisse trouver la bonne position à lire pour faire une transition d'une machine après avoir fait une transition pour l'autre à l'aide d'un marquage spécial. On a aussi besoin une machine pour intercaler les entrée au début du calcul. Tout peut être fait par une machine. □

Car toute machine ici peut être considérée comme régulière, on peut les manipuler comme des fonctions, en copiant parfois des résultats de calcul par une machine très simple.

3 Démonstration du théorme de Rice pour décidablité

On considère le language $L_u = \{\langle M, w\rangle | M \textit{ accepte } w\}$ et son complément $\overline{L_u}$.

Lemme 1. $\overline{L_u}$ n'est pas r.é., L_u n'est pas décidable.

Démonstration. Supposons que $\overline{L_u}$ soit r.é. On sait que MTU accepte $\langle M, w\rangle$ si et seulement si M accepte w.

On pose le language $L = \{\langle M\rangle | \langle MTU, \langle M, M\rangle\rangle \in \overline{L_u}\}$. $\langle M\rangle \in L$ si et seulement si M ne s'arrête pas sur l'entrée $\langle M\rangle$. Il est aussi r.é. car $\overline{L_u}$ est r.é., donc il existe une machine M^* qui accepte ce language L.

Si $\langle M^*\rangle \in L$, M^* accepte $\langle M^*\rangle$, donc s'arrête sur cette entrée, qui signifie que $\langle M^*\rangle \notin L$, M^*.

Si $\langle M^*\rangle \notin L$, M^*, M^* ne s'arrête pas sur $\langle M^*\rangle$, donc par définition, $\langle M^*\rangle \in L$, M^*.

On a une contradiction. Donc $\overline{L_u}$ n'est pas r.é.

Si L_u est décidable, son complément $\overline{L_u}$ l'est aussi, c'est absurde. Donc L_u n'est pas décidable. □

D'après ce lemme, on sait que le language L_u n'est pas décidable. Dans la démonstration suivante, on fera une réduction au language L_u pour avoir l'indécidabilité.

Théorème 1 (Le Théorème de Rice pour décidabilité)**.** Seules les propriétés triviales sont décidables.

Démonstration. On fait une démonstration par l'absrude.

Supposons qu'on ait une certaine propriété $\mathcal{P}$ non triviale décidable, donc son complément l'est aussi. Quitte à prendre son complément, on va supposer que $\mathcal{P}$ ne contient pas $\emptyset$. Il existe une machine M qui s'arrête toujours et qui décide le language $\mathcal{L}^{-1}(\mathcal{P})$.

Car $\mathcal{P}$ n'est pas triviale, il existe un language non vide $L \in \mathcal{P}$ qui est reconnu par une machine M_L.

Pour $\langle M', w\rangle$, on considère le codage $Cond(Conc(Imp(w), \langle M'\rangle), \langle M_L\rangle)$. C'est le codage d'une machine, noté M^*, qui fait la chose suivant.

– Faire marcher M' sur l'entrée w.

– Si c'est accepté, faire marcher M_L sur l'entrée réelle, sinon rejète tout.

Si $w \in \mathcal{L}(M')$, M^* accepte le même language que M_L, donc elle accepte L. Sinon, M^* n'accepte rien, donc elle accepte le language vide.

$\langle M^*\rangle$ est bien calculable par une machine à partir de $\langle M', w\rangle$. M accepte ce codage si et seulement si M' accepte w, c'est-à-dire $\langle M', w\rangle \in L_u$. Donc on peut construit une machine qui décide L_u par calculer d'abord le codage $\langle M^*\rangle$ correspondant à l'entrée, puis faire marcher M là-dessus. On obtient bien une contradiction, car L_u n'est pas décidable.

Donc si une propriété est non triviale, elle n'est pas décidable. On décide facilement les deux propriété triviales (Σ^* et $\emptyset$) avec une machine qui accepte tout et une autre qui rejète tout. □

4 Le Théorème de Rice pour décidabilité partielle

Avant de procéder à la démonstration du théorème, on montrera quelques lemmes qui démontrent en fait la nécessité des conditions imposées. D'après le lemme 1, on sait que le language $\overline{L_u}$ n'est pas r.é. Dans la démonstration suivante, on fera une réduction au language $\overline{L_u}$ pour les cas non r.é.

Lemme 2. Dans une propriété $\mathcal{P}$, s'il existe un language $L_1 \in \mathcal{P}$ et un autre language $L_2 \in \mathcal{R}, L_1 \subset L_2, L_2 \notin \mathcal{P}$, alors $\mathcal{P}$ n'est pas r.é.

Démonstration. On fait une démonstration par l'absrude.

Supposons $\mathcal{P}$ soit r.é., alors il existe une machine M qui accepte le language $\mathcal{L}^{-1}(\mathcal{P})$. L_1, L_2 sont r.é., donc il existe deux machine M_1, M_2 qui accepte respectivement ces deux languages.

On considère le codage $Para(\langle M_1\rangle, Cond(Conc(Imp(w), \langle M'\rangle), \langle M_2\rangle))$ pour $\langle M', w\rangle$. C'est le codage d'une machine, noté M^*, qui fait la chose suivant.

- Faire marcher M' sur w
- En cas acceptant, faire marcher M_1, M_2 sur l'entrée, sinon faire marcher que M_1.

M^* accepte L_2 si et seulement si M' accepte w, sinon M^* accepte L_1.

$\langle M^*\rangle$ est bien calculable par une machine à partir de $\langle M', w\rangle$. M accepte ce codage si et seulement si M' n'accepte pas w, c'est-à-dire $\langle M', w\rangle \in \overline{L_u}$. Donc on peut construit une machine qui accepte $\overline{L_u}$ par calculer d'abord le codage $\langle M^*\rangle$ correspondant à l'entrée, puis faire marcher M là-dessus. On obtient bien une contradiction, car $\overline{L_u}$ n'est pas r.é. □

Lemme 3. Dans une propriété $\mathcal{P}$, s'il existe un language infini $L \in \mathcal{P}$ tel que pour tout $L' \subset L$ fini, $L' \notin \mathcal{P}$, alors $\mathcal{P}$ n'est pas r.é.

Démonstration. On fait une démonstration par l'absrude.

Supposons $\mathcal{P}$ est r.é., il existe une machine M qui accepte le language $\mathcal{L}^{-1}(\mathcal{P})$. $L \in \mathcal{P} \subset \mathcal{R}$, donc il existe une machine M_L qui accepte L. C'est raisonnable de supposer que, à la fin du calcul, M_L laisse l'entrée originale sur la bande, car on peut faire une copie avant de procéder le calcul.

Car le codage de machine est naturel, il existe une machine T qui reçoit $\langle n, M, w\rangle$, avec n un entier, M une machine et w un mot, et qui simule M sur c pour n transitions. Il suffit de simuler une transition de M à chaque fois qu'on diminue n par un.

On peut aussi facilement construire une machine qui reçoit un mot x et qui imprime sur la bande $\langle |x|, M, w\rangle$. C'est simplement une machine qui calcule la longeur de l'entrée concaténée proprement à $Imp(\langle M', w\rangle)$ pour faire un bon codage, avec M' la machine obtenue à partir de M en inversant les rôles de l'état acceptant et l'état rejetant de M. Le codage de cette machine peut bien être construit par une machine, appelée S, à partir de $\langle M, w\rangle$.

Pour $\langle M', w\rangle$, on considère le codage $Conc(\langle M_L\rangle, Conc(\langle S\rangle, \langle T\rangle))$. C'est le codage d'une machine, noté M^*, qui fait la chose suivant.

- Faire marcher M_L sur l'entrée x.
- En cas acceptant, faire marcher M' sur w pour $|x|$ transitions. Si M' n'accepte pas w en $|x|$ transitions, la machine accepte l'entrée x.

Si M' accepte w, on note la longeur de calcul n. Alors le language accepté par cette machine M^* est $L' = L \setminus \{x \big| |x| \geq n\}$, car tout mot de longeur plus grande que n dans L est rejeté par la machine par définition. On constate que L' est fini, donc $L' \notin \mathcal{P}$.

Si M' n'accepte pas w, le language accepté par cette machine M^* est $L \in \mathcal{P}$, car la deuxième étape est toujours acceptant.

Alors on a M accepte $\langle M^* \rangle$ si et seulement si M n'accepte pas w, c'est-à-dire $\langle M', w \rangle \in \overline{L_u}$. Car $\langle M^* \rangle$ peut être calculé par une machine à partir de $\langle M', w \rangle$, on a une machine qui accepte $\overline{L_u}$ en calculant le codage $\langle M^* \rangle$ correspondant à l'entrée, puis en faisant marcher M sur ce codage.

Or, $\overline{L_u}$ n'est pas r.é., donc on a une contradiction. Pour conclusion, $\mathcal{P}$ n'est pas r.é. □

Lemme 4. Pour une propriété $\mathcal{P}$, l'ensemble des codages de languages finis dans $\mathcal{P}$ est r.é.

Démonstration. En utilisant la bijection $(i, j) \mapsto \frac{(i+j)*(i+j+1)}{2} + j$ entre $\mathbb{N}^2$ et $\mathbb{N}$, on peut construire facilement un générateur des paires d'entiers, noté *Paire*.

Car $\mathcal{P}$ est r.é., il existe une machine qui accepte $\mathcal{L}^{-1}(\mathcal{P})$, noté M.

On propose un codage de language fini. On utilise `0` pour représenter séparation de deux éléments, `10` pour 0, `11` pour 1. On peut bien vérifier la validité d'un codage de language fini par une machine.

Pour un certain codage i, on peut construire facilement le codage d'une machine $M^{(i)}$ qui reconnaît le language fini de codage i. C'est essentiellement un automate. Cela peut être fait facilement par une machine, notée C.

On construit directement un énumérateur de l'ensemble des codages de languages finis dans $\mathcal{P}$. D'abord on énumère des paires à l'aide de *Paire*. Pour chaque paire d'entiers (i, j), en cosidérant i comme un codage d'un language fini, on prend le codage $C(i)$ et simule M sur $C(i)$ pour j transitions en utilisant la machine T dans le lemme 3. En cas acceptant, la machine écrit i sur la bande.

C'est bien un énumérateur de l'ensemble des codages de languages finis dans $\mathcal{P}$. Donc c'est r.é. □

Théorème 2 (Le Théorème de Rice pour décidabilité partielle)**.** Une propriété $\mathcal{P}$ est r.é. si et seulement si elle vérifie les trois conditions suivantes :

1. Le sous-ensemble des languages finis dans $\mathcal{P}$ est r.é..
2. Si un language infini $L \in \mathcal{P}$, il existe un language fini L' contenu dans L qui est dans $\mathcal{P}$.
3. Si un language $L \in \mathcal{P}$ et un autre language $L' \in \mathcal{R}$ contenant L, alors $L' \in \mathcal{P}$.

Démonstration. La nécessité de ces conditions est donnée par les trois lemmes. Il suffit de constuire une machine M^* qui accepte $\mathcal{L}^{-1}(\mathcal{P})$ pour $\mathcal{P}$ vérifiant les conditions.

D'après le lemme 4, il existe un énumérateur E des codages des languages finis dans $\mathcal{P}$.

M^* prend en argument le codage d'une machine $\langle M \rangle$. On utilise d'abord le générateur *Paire* de paires dans le lemme 4. Pour chaque paire (i, j), on

simule E pour obtenir le i-ième language fini, noté L_i, puis on simule M pour j transitions sur chaque mot dans L_i en utilisant la machine T proposée dans le lemme 3. Si tout est accepté, M^* accepte $\langle M \rangle$, sinon elle examine une autre paire généré par *Paire*.

Soit M une machine avec $\mathcal{L}(M) \in \mathcal{P}$, d'après la condition 2, il existe un language fini $L' \subset \mathcal{L}(M)$. L' est produit par E par définition de E, on suppose qu'il est le i-ième. L' est fini, donc on peut prendre j la longeur maximale de calcul effectué par M sur un mot dans L'. Quand la paire (i, j) est générée dans M^*, $\langle M \rangle$ sera accepté par M^*, si pas déjà. Donc $\mathcal{L}^{-1}(\mathcal{P}) \subset \mathcal{L}(M^*)$.

Si un codage $\langle M \rangle$ est accepté par M^*, il existe une paire (i, j) tel que M accepte tout mot dans L_i pour au plus j transitions. L_i est généré par E. Par définition de E, $L_i \in \mathcal{P}$. Or, $\mathcal{L}(M) \in \mathcal{R}$, $L_i \subset \mathcal{L}(M)$, donc d'après la condition 3, $\mathcal{L}(M) \in \mathcal{P}$. Donc $\mathcal{L}(M^*) \subset \mathcal{L}^{-1}(\mathcal{P})$.

Par double inclusion, $\mathcal{L}(M^*) = \mathcal{L}^{-1}(\mathcal{P})$. $\mathcal{P}$ est r.é. □

Ce théorème nous porte plein de résultats intéressants et non triviaux du tout. Voici quelques exemples.

Corollaire 1. Les propriétés des languages r.é. suivantes ne sont pas r.é. :

1. L est régulier
2. L est décidable.
3. L n'est pas décidable.
4. L contient exactement k éléments, avec k un entier fixé.
5. L contient au plus k éléments, avec k un entier fixé.
6. $\overline{L}$ contient exactement k éléments, avec k un entier fixé.
7. $\overline{L}$ contient au plus k éléments, avec k un entier fixé.
8. $L \setminus L_u \neq \emptyset$

Démonstration. Les cinq premières ne vérifient pas la troisième condition. Les deux suivante ne vérifient pas la deuxième condition. La dernière ne vérifie pas la première condition, car $\overline{L_u}$ n'est pas r.é. □

Corollaire 2. Les propriétés des languages r.é. suivantes ne sont pas r.é. :

1. L contient au moins k éléments, avec k un entier fixé.
2. Pour un language fini fixé L_0, $L_0 \subset L$
3. $L \cap L_u \neq \emptyset$

Démonstration. On fait une vérification rapide pour les conditions du théorème. □

Références

[1] John E. Hopcroft and Jeffery D. Ullman. *Introduction to Automata Theory, Languages, and Computation.* Addison-Wesley, 1979.