Partiel - Mardi 31 avril 2009

Dans tout le partiel, une fonction sera dite *récursive* lorsqu'elle est calculable par machine de Turing (qu'elle soit totale ou non). On parlera explicitement de fonction *récursive totale* le cas échéant.
On convient comme tout au long des tds que l'on dispose d'une fonction de codage $\langle \cdot, \cdot \rangle : \mathbb{N}^2 \to \mathbb{N}$ récursive primitive dont les projections réciproques Π_1 et Π_2 sont récursives primitives.
Aucun document n'est autorisé.

Exercice 1. *C'est un peu cours jeune homme*

1. Soit F la suite de Fibonacci :
– $F(0) = F(1) = 1$
– $F(n+2) = F(n) + F(n+1)$
Montrer que F est récursive primitive et en donner une écriture récursive primitive (on pourra utiliser sans justification les fonctions récursives primitives vues en TD).
La récursion primitive permet uniquement de définir $F(n+1)$ en fonction de $F(n)$, et non $F(n-1)$: pour s'en sortir, il faut introduire $F'(n) = (F(n+1), F(n))$.
F' est bien définissable par une récursion primitive : $F'(n) = Rec(g, h)$ avec $g() = (1, 0)$ et $h(n, F'(n)) = ((p_1(F'(n)) + p_2(F'(n))), p_1(F'(n)))$. Et F est la projection de la deuxième coordonnée de $F'(n)$.

2. Les ensembles suivants sont-ils récursifs ? récursivement énumérables ? co-récursivement énumérables (c'est-à-dire de complémentaire récursivement énumérable) ?
Dire directement que Récursif ssi Récursivement énumérable ET co-Récursivement énumérable, pour l'utiliser à chaque fois.

1. $\{i \in \mathbb{N}, \varphi_i(1664)$ n'est pas défini $\}$
   Par Rice, cet ensemble n'est pas récursif (il n'est ni vide ni égal à $\mathbb{N}$). Sur une entrée i, on peut simuler la machine φ_i sur l'entrée 1664, et accepter si ce calcul s'arrête : l'ensemble de définition de la machine ainsi définie est le complémentaire de l'ensemble considéré. L'ensemble est donc co-récursivement énumérable. Donc non récursivement énumérable.
2. $\{x \in \mathbb{N}, \varphi_{1664}(x) = 51\}$
   Le théorème de Rice conserne les ensembles $\{i : P(\varphi_i)\}$ où P est une propriété non triviale des machines de Turing. Ici, l'ensemble n'est pas de cette forme, et Rice ne s'applique pas.
   On peut simuler la machine φ_{1664} sur l'entrée x, accepter si le calcul s'arrête et renvoie 51, boucler sinon : l'ensemble est récursivement énumérable.
   Pour la récursivité , ça dépend de φ_{1664}, et donc du système acceptable de programmation : si φ_{1664} calcule la fonction nulle, alors notre ensemble est vide et récursif. Si φ_{1664} est une machine universelle, décider quand elle s'arrête reviendrait à décider le problème de l'arrêt, et l'ensemble n'est alors pas récursif.
3. $\{\langle i, j \rangle, \varphi_i = \varphi_j\}$
   Si l'ensemble considéré est récursivement énumérable, alors $A = \{i : \varphi_i = \bot\}$, où $\bot$ est la fonction qui boucle tout le temps, est récursivment énumérable.
   Il existe donc une machine M d'ensemble de définition A.

Or il existe une machine qui sur $\langle x, y\rangle$ calcule i le numéro de la machine qui qu'elle que soit l'entréee simule $\varphi_x(y)$.

Si on lance M sur cette entrée i on peut décider quand $varphi_x(y)$ boucle, ce qui est impossible (problème de l'arrêt). L'ensemble n'est pas récursivement énumérable.

De même, si l'ensemble considéré est co-récursivement énumérable, alors $B = \{i : \varphi_i = 1\}$, où 1 est la fonction qui vaut tout le temps 1, est co-récursivment énumérable.

Il existe donc une machine M d'ensemble de définition le complémentaire de B.

Or il existe une machine qui sur $\langle x, y\rangle$ calcule i le numéro de la machine qui qu'elle que soit l'entréee simule $\varphi_x(y)$, et renvoie 1 si ça s'arrête.

Si on lance M sur cette entrée i on peut encore décider quand $varphi_x(y)$ boucle, ce qui est toujours impossible (problème de l'arrêt). L'ensemble n'est pas non plus co-récursivement énumérable.

ATTENTION : il était complètement faux de dire que cet ensemble est égal à $\{\langle n, n\rangle : n \in \mathbb{N}\}$. Plusieurs codes peuvent correspondre à la même fonction ; il existe une infinité de façons de calculer chaque fonction.

4. $\{i \in \mathbb{N}, \varphi_i(x) = 1664$ si $\varphi_x(x)$ s'arrête et 51 sinon$\}$
   Aucune machine φ_i ne peut décider si $\varphi_x(x)$ s'arrête pour tout x : donc cet ensemble est vide. Il est donc récursif.

3. Montrer que si on se donne une machine résolvant le problème de l'arrêt, on peut l'utiliser pour construire une machine résolvant la conjecture de Goldbach. La conjecture de Goldbach affirme que tout entier pair est la somme de deux nombres premiers.

On peut construire une machine M qui qu'elle que soit son entrée énumère tous les entiers pairs i, et pour chacun décide si i est somme de deux nombres premiers (il suffit de vérifier pour les nombres premiers inférieurs à i), puis passe au suivant si i vérifie la propriété, s'arrête et renvoie i si i est un contre-exemple.

Cette machine boucle ssi tous les i vérifient la propriété. Il suffit de donner $(M, 0)$ à notre machine décidant de l'arrêt pour résoudre Goldbach

4. Montrer que si on se donne une machine résolvant le problème de l'arrêt, on peut l'utiliser pour construire une machine résolvant la conjecture de Syracuse. La conjecture de Syracuse affirme que pour tout entier a la suite définie par $u_0 = a$ et

$$u_{n+1} = \begin{cases} u_n/2 & \textbf{si } u_n \textbf{ est pair} \\ 3u_n + 1 & \textbf{sinon} \end{cases}$$

atteint 1 pour un certain n.

Cette fois-ci, il est nécessaire d'utiliser DEUX FOIS la machine de l'arrêt : une fois comme sous-programme pour construire la machine qui décide si sur un entier a donné, la suite u_n atteint 1 ; puis une autre fois pour lancer cette machine sur tous les a entiers en série, et décider si ils conviennent tous ou si on trouve un contre-exemple.

5. Existe-t-il $m \in \mathbb{N}$ tel que φ_m ait comme domaine $\mathbb{N}\backslash\{m\}$? Justifiez votre réponse.

Certains ont remarqué que pour certains systèmes acceptables de programmation la propriété est vérifiée ; par exemple si ϕ_0 boucle sur 0 et vaut 1 partout ailleurs. On peut le montrer en général, quel que soit le SAP.

Soit f la fonction qui sur l'entrée n retourne le numéro d'une machine qui boucle sur n et vaut 1 partout ailleurs. On assurer que f soit récursive.

D'après le théorème de Kleene, comme f est récursive, il existe m tel que $\varphi_m = \varphi_{f(m)}$. Or $\varphi_{f(m)}$ a pour domaine $\mathbb{N}\backslash\{m\}$ par définition de f ; donc φ_m aussi.

ATTENTION : il était absurde de dire : "soit φ_m la fonction qui sur l'entrée m boucle, sinon vaut 1". La définition se mord la queue : φ_m dépend de m, et le numéro m dépend de la définition de la fonction φ_m.

Exercice 2. *Numérologie*

Soit $\mathcal{F}$ un ensemble de fonctions partielles récursives à une variable. On dira que $\mathcal{F}$ est *récursivement énuméré* s'il existe une fonction F partielle récursive à deux variables telle que, si l'on pose $\mathrm{F}_x : y \to \mathrm{F}(x, y)$, l'on a

$$\mathcal{F} = \{\mathrm{F}_x; x \in \mathbb{N}\}$$

1. L'ensemble des fonctions récursives est-il récursivement énuméré ?

2. L'ensemble des fonctions récursives totales est-il récursivement énuméré ?

3. On admet que l'ensemble des fonctions récursives primitives est récursivement énuméré. Montrer que l'ensemble des fonctions récursives primitives strictement croissantes est récursivement énuméré.

4. Nous allons montrer que l'ensemble des fonctions récursives strictement croissantes n'est pas récursivement énuméré. Pour cela, supposons que cet ensemble est récursivement énuméré ; soit F la fonction qui énumère ces fonctions. Construire une fonction récursive totale strictement croissante qui n'est égale à aucun F_x. Conclure.

Exercice 3. *Arrêt de production*

On note, pour n entier, W_n le domaine de définition de φ_n. Un ensemble A est dit *productif* s'il existe une fonction récursive totale unaire g telle que

$$\forall n \in \mathbb{N}\ (W_n \subseteq A \Rightarrow g(n) \in A \setminus W_n)$$

1. Un ensemble productif peut-il être récursivement énumérable ?

2. Soit A et B deux ensembles de naturels. Montrer que si A se réduit[1] à B et si A est productif, alors B est productif.

3. Montrer que $\{n \in \mathbb{N}/n \notin W_n\}$ est productif. En déduire là comme ça, paf, que le problème de l'arrêt est indécidable, c'est-à-dire qu'il n'existe pas d'algorithme décidant si la machine de Turing numéro n s'arrête sur l'entrée x.

4. Pour la route, montrer que $\{n \in \mathbb{N}/W_n$ est fini$\}$ et $\{n \in \mathbb{N}/\varphi_n$ est injective$\}$ sont productifs.

5. Montrer que pour toute fonction récursive totale unaire g, il existe une fonction récursive totale k telle que $\forall n \in \mathbb{N}\ W_{k(n)} = W_n \cup \{g(n)\}$. En déduire que tout ensemble productif contient un ensemble récursivement énumérable infini.

1. On dit que A se réduit à B quand il existe une fonction récursive totale f telle que $\forall n \in \mathbb{N}\ (n \in A \Leftrightarrow f(n) \in B)$.

Exercice 4. *Coloriage*

Soit k un entier non nul. Un graphe[2] est dit k-coloriable si il existe une fonction de l'ensemble des sommets dans $\{1, \ldots, k\}$ telle que deux sommets adjacents n'ont pas la même image par cette fonction.

✎ **Montrer qu'un graphe infini est k-coloriable si et seulement si tout sous-graphe fini est k-coloriable.**

Stratégie : il y a un sens évident (infini implique fini), un sens non trivial. Si vous n'avez pas d'idée pour le sens difficile, vous pouvez dire clairement que vous démontrez uniquement un sens ; si par contre vous juxtaposez une preuve correcte du sens facile et une preuve fausse du sens difficile, la démonstration du premier s'en trouve décrédibilisée.

On démontre ce résultat par théorème de compacité du calcul propositionnel. Il est possible de construire un ensemble de formules booléennes correspondant à un graphe donné, telle que l'ensemble de ces formules sera satisfiable si et seulement si le graphe est k-coloriable. On remarque aisément que toute partie finie de ces formules sera satisfiable si tout sous-graphe fini est k-coloriable. L'application du théorème de compacité finit le boulot.
Si vous avez le temps, écrivez ces formules et justifiez ; sinon, une explication comme ce qui précède est bien sûr valorisée.

Il n'est pas complètement impossible de s'en tirer avec le théorème de compacité du calcul des prédicats, mais c'est beaucoup plus délicat, puisque celui-ci nous assure l'existence d'un "modèle" satisfaisant nos formules ; assurer que le modèle correspond bien forcément un coloriage d'un sous graphe est compliqué. Donc utiliser pour ce genre de questions le théorème du calcul propositionnel, ce qui implique de le dire clairement, de n'utiliser que des formules booléennes, et de ne mettre aucun quantificateur dans les formules.

2. Un graphe $G = (V, E)$ est défini par un ensemble de sommets V et un ensemble d'arrêtes $E \subseteq V \times V$ reliant ces sommets. Deux sommets u et v sont dits adjacents si et seulement si $(u, v) \in E$.