ALGORITHMIQUE ET CALCULABLITÉ
FONCTION D'ACKERMANN

Michel Rigo

http://www.discmath.ulg.ac.be/

Année 2007–2008

Rappels

Ensemble des fonctions primitives récursives

- fonctions initiales : 0, $\mathcal{P}_{i,p}$, σ
- composition : $f \in \mathfrak{F}_n$, $g_1, \ldots, g_n \in \mathfrak{F}_p$, $h = f(g_1, \ldots, g_n)(\overline{x})$.
- récursion primitive : $g \in \mathfrak{F}_p$, $h \in \mathfrak{F}_{p+2}$,
  - $f(x_1, \ldots, x_p, 0) = g(x_1, \ldots, x_p)$
  - $f(x_1, \ldots, x_p, n+1) = h(x_1, \ldots, x_p, n, f(x_1, \ldots, x_p, n))$.

Définition

L'ensemble $\mathcal{PR}$ des fonctions *primitives récursives* est la plus petite famille de fonctions contenant les fonctions initiales et stable pour la composition et la récursion primitive.

Fait

Les fonctions $\mathcal{PR}$ sont “calculables” (d’un point de vue naïf). On dispose d’une procédure effective de calcul.

Jusque vers 1900, on pensait que toute fonction “calculable” était primitive récursive.

Question

Existe-t-il des fonctions “calculables” (au sens naïf) mais qui ne sont pas $\mathcal{PR}$?

RÉPONSE

La réponse est oui !

- **Première solution** : argument de “diagonalisation”, réponse facile mais on n’exhibe pas réellement une fonction (construction “existentielle”)
- **Deuxième solution** : la fonction d’Ackermann
  - “calculable” par une procédure effective
  - n’est pas $\mathcal{PR}$
  - on pourrait montrer qu’elle est *récursive* ($\mathcal{PR} \subsetneq \mathcal{R}$)

La fonction d'Ackermann

Définition

La *fonction d'Ackermann* $\mathcal{A} \in \mathfrak{F}_2$ est définie par

$$\begin{cases} \mathcal{A}(0,m) &= m+1, \\ \mathcal{A}(m+1,0) &= \mathcal{A}(m,1), \\ \mathcal{A}(m+1,n+1) &= \mathcal{A}(m,\mathcal{A}(m+1,n)). \end{cases}$$

On pose $\mathcal{A}_m \in \mathfrak{F}_1$ comme

$$\mathcal{A}_m : n \mapsto \mathcal{A}(m,n).$$

$$\mathcal{A}_0(n) = n + 1$$

$$\mathcal{A}_1(0) = \mathcal{A}_0(1) = 2 \quad \text{et} \quad \mathcal{A}_1(n+1) = \mathcal{A}_0(\mathcal{A}_1(n)) = \mathcal{A}_1(n) + 1,$$

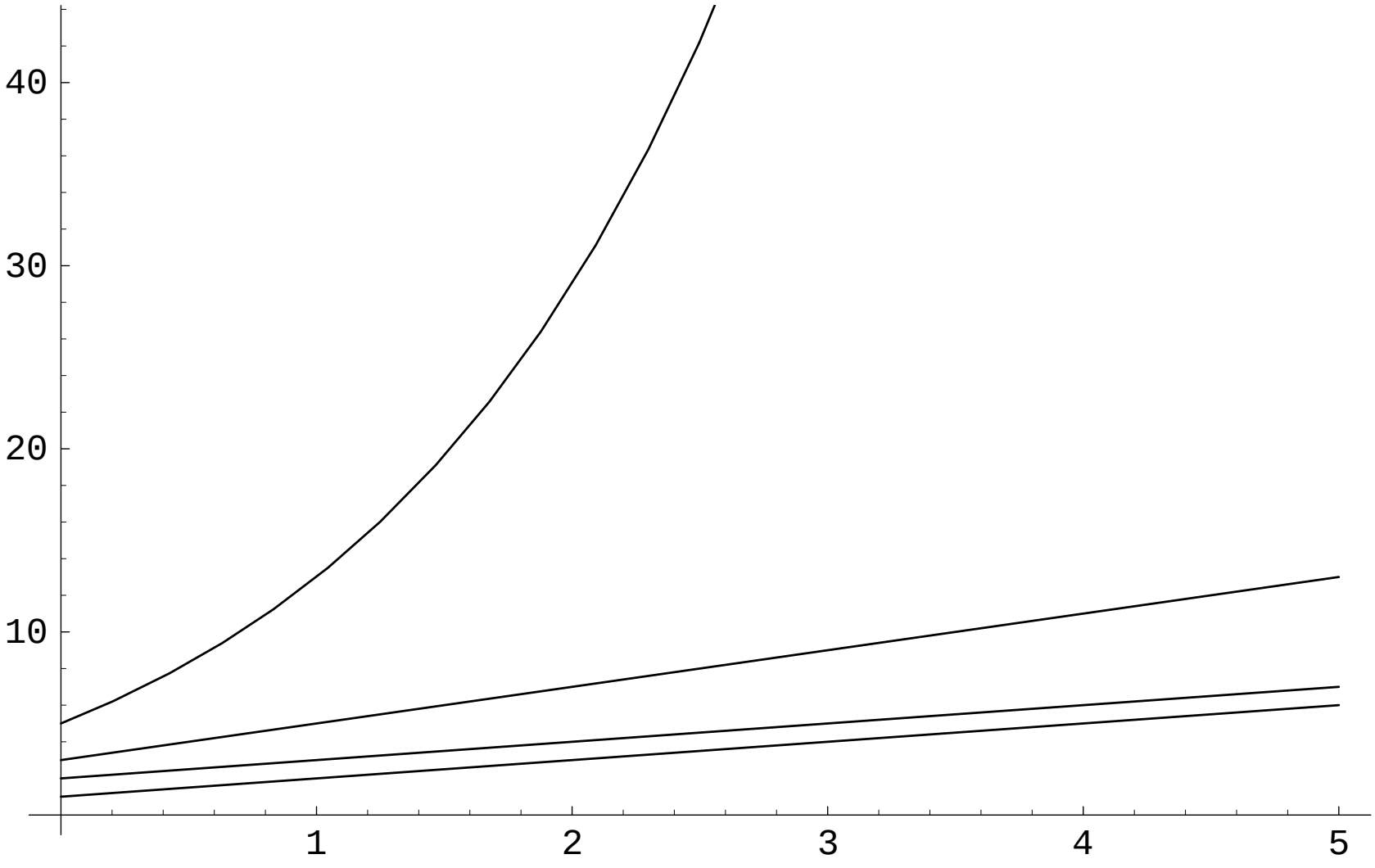
$$\mathcal{A}_1(n) = n + 2.$$

$$\mathcal{A}_2(0) = \mathcal{A}_1(1) = 3 \quad \text{et} \quad \mathcal{A}_2(n+1) = \mathcal{A}_1(\mathcal{A}_2(n)) = \mathcal{A}_2(n) + 2,$$

$$\mathcal{A}_2(n) = 2n + 3.$$

$$\mathcal{A}_3(0) = \mathcal{A}_2(1) = 5 \quad \text{et} \quad \mathcal{A}_3(n+1) = \mathcal{A}_2(\mathcal{A}_3(n)) = 2\mathcal{A}_3(n)+3,$$

$$\mathcal{A}_3(n) = 2^{n+3} - 3.$$


40
30
20
10
1
2
3
4
5

Remarque

Pour tout $m \in \mathbb{N}$, $\mathcal{A}_m \in \mathfrak{F}_1$ est $\mathcal{PR}$.

On procède par récurrence sur m.
cas de base immédiat : $\mathcal{A}_0(n) = n + 1$.

Supposons $\mathcal{A}_m$ de classe $\mathcal{PR}$ et montrons que $\mathcal{A}_{m+1}$ l'est.

On dispose du schéma de récursion primitive suivant

$$\begin{cases} \mathcal{A}_{m+1}(0) = \mathcal{A}_m(1), \\ \mathcal{A}_{m+1}(n+1) = \mathcal{A}_m(\mathcal{A}_{m+1}(n)). \end{cases}$$

$\mathcal{A}_m \in \mathcal{PR}$ n'implique pas $\mathcal{A} \in \mathcal{PR}$.

PROPOSITION

La fonction d'Ackermann $\mathcal{A}$ est calculable par une procédure effective.

On doit être en mesure de calculer des expressions de la forme

$$\mathcal{A}(m_1, \mathcal{A}(m_2, \ldots, \mathcal{A}(m_{k-1}, m_k))) \quad \text{où } k \geq 2$$

codée par le k-uple $\mathsf{S} = (m_1, \ldots, m_k) \in \mathbb{N}^k$
et $|\mathsf{S}|$ en représente la longueur, i.e., $|\mathsf{S}| = k$.

- q l'élément m_k le plus à droite de S,
- p son avant-dernier élément m_{k-1} et
- T la suite des $k-2$ premiers éléments (T peut être vide si $k = 2$).

Ainsi,

$$\mathsf{S} = (\underbrace{m_1, \ldots, m_{k-2}}_{\mathsf{T}}, \underbrace{m_{k-1}}_{\mathsf{p}}, \underbrace{m_k}_{\mathsf{q}}) = (\mathsf{T}, \mathsf{p}, \mathsf{q}).$$

L'algorithme suivant calcule $\mathcal{A}(m,n)$:

T$\leftarrow \emptyset$, p$\leftarrow m$, q$\leftarrow n$ (on initialise S à $(\emptyset, m, n)$)
Tant que $|S| \geq 2$
  Si p $= 0$ et q ≥ 0, alors S$\leftarrow$ (T, q $+ 1$)
  Si p > 0 et q $= 0$, alors S$\leftarrow$ (T, p $- 1, 1$)
  Si p > 0 et q > 0, alors S$\leftarrow$ (T, p $- 1$, p, q $- 1$)
Sortir S$=$ (q).

L'idée de cet algorithme est de toujours s'intéresser à la fonction $\mathcal{A}$ la plus imbriquée dans S. Les trois règles de l'algorithme correspondent exactement à la définition de $\mathcal{A}$:

$$(\mathsf{T}, 0, \mathsf{q}) \text{ devient } (\mathsf{T}, \mathsf{q}+1) \text{ car } \mathcal{A}(0,n) = n+1,$$

$$(\mathsf{T}, \mathsf{p}, 0) \text{ devient } (\mathsf{T}, \mathsf{p}-1, 1) \text{ car } \mathcal{A}(m+1,0) = \mathcal{A}(m,1),$$

$$(\mathsf{T}, \mathsf{p}, \mathsf{q}) \text{ devient } (\mathsf{T}, \mathsf{p}-1, \mathsf{p}, \mathsf{q}-1)$$

$$\text{car } \mathcal{A}(m+1,n+1) = \mathcal{A}(m, \mathcal{A}(m+1,n)).$$

Si l'algorithme s'achève, ce sera avec le bon résultat. Il suffit de prouver (par récurrence sur m) qu'il s'achève pour toutes valeurs de m et n.

Pour $m = 0$, S est initialisé à $(0, n)$, on sort de la boucle avec une suite $S = (n + 1)$ de longueur 1.

Si l'algorithme s'achève pour m, montrons qu'il s'achève encore pour $m + 1$. Si S est initialisé à $(m + 1, n)$, alors on a, en appliquant la troisième règle,

$$\begin{aligned} & (m+1, n) \\ \to \quad & (m, m+1, n-1) \\ \to \quad & (m, m, m+1, n-2) \\ & \vdots \\ \to \quad & (\underbrace{m, m, \ldots, m}_{n \text{ fois}}, m+1, 0). \end{aligned}$$

En appliquant la deuxième règle, on trouve

$$\to (m, m, \ldots, m, m, 1)$$

appliquer $n + 1$ fois l'hyp. de réc. pour conclure.

Exemple

Pour calculer $\mathcal{A}(1,2)$, avec les notations précédentes, on trouve

$$(1,2) \to (0,1,1) \to (0,0,1,0) \to (0,0,0,1)$$

$$\to (0,0,2) \to (0,3) \to (4).$$

Pour montrer que $\mathcal{A}$ n'appartient pas à $\mathcal{PR}$, quelques lemmes sont nécessaires.

Lemme 1

Pour tout m et pour tout n, $\mathcal{A}_m(n) > n$.

Par réc. sur m. Pour $m = 0$ et pour tout n, $\mathcal{A}_0(n) = n + 1 > n$.
Supposons que, pour tout n, $\mathcal{A}_m(n) > n$ et vérifions que, pour tout n, $\mathcal{A}_{m+1}(n) > n$.

On procède par réc. sur n. Si $n = 0$, alors par hyp. de réc. sur m, on a bien $\mathcal{A}_{m+1}(0) = \mathcal{A}_m(1) > 1 > 0$.
Supposons à présent que $\mathcal{A}_{m+1}(n) > n$ et vérifions que $\mathcal{A}_{m+1}(n+1) > n + 1$. Il vient en utilisant successivement les hypothèses de récurrence sur m et sur n

$$\mathcal{A}_{m+1}(n+1) = \mathcal{A}_m(\mathcal{A}_{m+1}(n)) > \mathcal{A}_{m+1}(n) > n$$

d'où le résultat annoncé, $\mathcal{A}_{m+1}(n+1) > n + 1$, puisque nous sommes en présence de deux inégalités strictes consécutives.

Lemme 2

Pour tout m, la fonction $\mathcal{A}_m$ est strictement croissante, i.e., $\mathcal{A}_m(n+1) > \mathcal{A}_m(n)$.

Si $m = 0$, c'est immédiat puisque $\mathcal{A}_0(n) = n + 1$.
Si $m > 0$, alors en utilisant le lemme précédent, il vient

$$\mathcal{A}_m(n+1) = \mathcal{A}_{m-1}(\mathcal{A}_m(n)) > \mathcal{A}_m(n).$$

Lemme 3

Pour tout m et pour tout n, on a

$$\mathcal{A}_{m+1}(n) \geq \mathcal{A}_m(n).$$

Pour $n = 0$, on a $\mathcal{A}_{m+1}(0) = \mathcal{A}_m(1) > \mathcal{A}_m(0)$ car $\mathcal{A}_m$ est une fonction strictement croissante (cf. lemme 2).
Pour $n > 0$, au vu du lemme 1, $\mathcal{A}_{m+1}(n-1) \geq n$ et puisque $\mathcal{A}_m$ est une fonction strictement croissante, il vient

$$\mathcal{A}_m(\mathcal{A}_{m+1}(n-1)) \geq \mathcal{A}_m(n).$$

En appliquant la définition de la fonction d'Ackermann, on en conclut que

$$\mathcal{A}_{m+1}(n) = \mathcal{A}_m(\mathcal{A}_{m+1}(n-1)) \geq \mathcal{A}_m(n).$$

On note $\mathcal{A}_m^k$ la composition de k copies de $\mathcal{A}_m$, i.e.,

$$\mathcal{A}_m^k = \underbrace{\mathcal{A}_m \circ \cdots \circ \mathcal{A}_m}_{k \text{ fois}}.$$

En particulier, $\mathcal{A}_m^0$ est la fonction identité.

Lemme 4

Les fonctions $\mathcal{A}_m^k$ sont toutes strictement croissantes. De plus, pour tous m, n, k, ℓ, on a

- $\mathcal{A}_m^{k+1}(n) > \mathcal{A}_m^k(n)$,
- $\mathcal{A}_m^k(n) \geq n$,
- si $m \leq p$, alors $\mathcal{A}_m^k(n) \leq \mathcal{A}_p^k(n)$.

La preuve est immédiate et découle principalement du fait que $\mathcal{A}_m$ est un fonction strictement croissante. La dernière assertion découle du lemme 3.

Definition

Soient $f \in \mathfrak{F}_1$ et $g \in \mathfrak{F}_p$. La fonction f *domine* g, ce que l'on notera $g \prec f$, s'il existe une constante C telle que

$$\forall (x_1, \ldots, x_p) \in \mathbb{N}^p : \; g(x_1, \ldots, x_p) \leq f(\sup(x_1, \ldots, x_p, C)).$$

Remarque

Si $f \in \mathfrak{F}_1$ est une fonction strictement croissante, alors $g \prec f$ SSI $g(x_1, \ldots, x_p) \leq f(\sup(x_1, \ldots, x_p))$ sauf pour un nombre fini de points.

$\Rightarrow$ Le nombre de p-uples $(x_1, \ldots, x_p) \in \mathbb{N}^p$ tels que $\sup(x_1, \ldots, x_p) < C$ est majoré par C^p et est donc fini.

$\Leftarrow$ soient $\overline{a_1}, \ldots, \overline{a_t}$ les points (en nombre fini) tels que $g(\overline{a_i}) > f(\sup(\overline{a_i}))$. On pose $C = \sup\{g(\overline{a_1}), \ldots, g(\overline{a_t})\}$. Puisque f est strictement croissante, il existe D tel que pour tout $x \geq D$, $f(x) > C$. Ainsi, $\forall \overline{x} \in \mathbb{N}^p$, $g(\overline{x}) \leq f(\sup(\overline{x}, D)$ et $g \prec f$.

Notation

Pour $m \geq 0$, on pose

$$\mathfrak{C}_m = \{g \in \mathfrak{F} \mid \exists k \in \mathbb{N} : g \prec \mathcal{A}_m^k\}$$

comme étant l'ensemble des fonctions dominées par un itéré de la fonction $\mathcal{A}_m$ et

$$\mathfrak{C} = \bigcup_{m \geq 0} \mathfrak{C}_m.$$

Les fonctions primitives récursives de base appartiennet à $\mathfrak{C}_0$.
En outre, il est évident que si $f, g \in \mathfrak{F}_p$, si $g \in \mathfrak{C}_m$ et si

$$\forall (x_1, \ldots, x_p) \in \mathbb{N}^p : f(x_1, \ldots, x_p) \leq g(x_1, \ldots, x_p),$$

alors f appartient aussi à $\mathfrak{C}_m$.

Remarque

Il est aisé de se convaincre que les fonctions

- $\sup : (x_1, \ldots, x_p) \mapsto \sup(x_1, \ldots, x_p)$,
- $\Sigma_2 : (m, n) \mapsto m + n$,
- pour $k \in \mathbb{N}$, $n \mapsto k.n$

appartiennent toutes à $\mathfrak{C}_2$. Cette observation nous sera utile pour la suite.

Notre but est de montrer que $\mathfrak{C}$ contient l'ensemble $\mathcal{PR}$. Puisque $\mathfrak{C}$ contient les fonctions primitives récursives de base, il nous suffit de vérifier que $\mathfrak{C}$ est stable par composition et récursion primitive.

Lemme 5

Pour tout $m \geq 0$, l'ensemble $\mathfrak{C}_m$ est stable par composition. En particulier, $\mathfrak{C}$ l'est aussi.

Soient $f_1, \ldots, f_n$ des fonctions de $\mathfrak{F}_p \cap \mathfrak{C}_m$ et une fonction $g \in \mathfrak{F}_n \cap \mathfrak{C}_m$. Par définition de l'ensemble $\mathfrak{C}_m$, il existe des constantes $k, k_1, \ldots, k_n$ et $C, C_1, \ldots, C_n$ telles que

$$\forall (y_1, \ldots, y_n) \in \mathbb{N}^n : g(y_1, \ldots, y_n) \leq \mathcal{A}_m^k(\sup(y_1, \ldots, y_n, C))$$

et pour tout $i \in \{1, \ldots, n\}$, on a

$$\forall (x_1, \ldots, x_p) \in \mathbb{N}^p : f_i(x_1, \ldots, x_p) \leq \mathcal{A}_m^{k_i}(\sup(x_1, \ldots, x_p, C_i)).$$

Posons $D = \sup(C, C_1, \ldots, C_n)$ et $K = \sup(k_1, \ldots, k_n)$, il vient

$$\begin{array}{rl} & g(f_1(x_1, \ldots, x_p), \ldots, f_n(x_1, \ldots, x_p)) \\ \leq & \mathcal{A}_m^k(\sup(f_1(x_1, \ldots, x_p), \ldots, f_n(x_1, \ldots, x_p), C) \\ \leq & \mathcal{A}_m^k(\mathcal{A}_m^K \sup(x_1, \ldots, x_p, D)) \\ \leq & \mathcal{A}_m^{k+K}(\sup(x_1, \ldots, x_p, D)) \end{array}$$

où, à l'avant-dernière ligne, on a utilisé le lemme 4.

Lemme 6

Pour tous m, n, k, on a

$$\mathcal{A}_m^k(n) \leq \mathcal{A}_{m+1}(n+k).$$

On procède par récurrence sur k. Le cas de base $k = 0$ est immédiat.

Supposons le résultat satisfait pour k et vérifions-le pour $k+1$.

Il vient, en appliquant l'hypothèse de récurrence,

$$\mathcal{A}_m^{k+1}(n) = \mathcal{A}_m(\mathcal{A}_m^k(n)) \leq \mathcal{A}_m(\mathcal{A}_{m+1}(n+k)) = \mathcal{A}_{m+1}(n+k+1),$$

la dernière égalité étant la définition même de la fonction d'Ackermann.

Lemme 7

Si $g \in \mathfrak{F}_p$ et $h \in \mathfrak{F}_{p+2}$ sont deux fonctions de $\mathfrak{C}_m$, alors la fonction f obtenue par récursion primitive à partir de g et de h appartient à $\mathfrak{C}_{m+1}$. En particulier, $\mathfrak{C}$ est stable par récursion primitive.

Soient $g \in \mathfrak{F}_p$ et $h \in \mathfrak{F}_{p+2}$ deux fonctions de $\mathfrak{C}_m$. On considère la fonction f définie par

$$\begin{cases} f(x_1, \ldots, x_p, 0) = g(x_1, \ldots, x_p) \\ f(x_1, \ldots, x_p, n+1) = h(x_1, \ldots, x_p, n, f(x_1, \ldots, x_p, n)). \end{cases}$$

Par définition de l'ensemble $\mathfrak{C}_m$, il existe des constantes C_1, C_2, k_1, k_2 telles que

$$\forall (x_1, \ldots, x_p) \in \mathbb{N}^p : \; g(x_1, \ldots, x_p) \leq \mathcal{A}_m^{k_1}(\sup(x_1, \ldots, x_p, A_1))$$

et $\forall (x_1, \ldots, x_p, x_{p+1}, x_{p+2}) \in \mathbb{N}^{p+2}$:

$$h(x_1, \ldots, x_p, x_{p+1}, x_{p+2}) \leq \mathcal{A}_m^{k_2}(\sup(x_1, \ldots, x_p, x_{p+1}, x_{p+2}, A_2)).$$

On montre tout d'abord par récurrence sur n que

$$f(x_1, \ldots, x_p, n) \leq \mathcal{A}_m^{k_1+nk_2}(\sup(x_1, \ldots, x_p, n, A_1, A_2)). \quad (1)$$

Le résultat est vrai pour $n = 0$. Supposons-le satisfait pour n et vérifions-le pour $n + 1$. Il vient

$$f(x_1, \ldots, x_p, n+1) \leq \mathcal{A}_m^{k_2}(\sup(x_1, \ldots, x_p, n, f(x_1, \ldots, x_p, n), A_2)).$$

En appliquant l'hypothèse de récurrence, on trouve

$$\begin{aligned} f(x_1, \ldots, x_p, n+1) &\leq \mathcal{A}_m^{k_2}(\mathcal{A}_m^{k_1+nk_2}(\sup(x_1, \ldots, x_p, n, A_1, A_2))) \\ &\leq \mathcal{A}_m^{k_1+(n+1)k_2}(\sup(x_1, \ldots, x_p, n, A_1, A_2)) \\ &\leq \mathcal{A}_m^{k_1+(n+1)k_2}(\sup(x_1, \ldots, x_p, n+1, A_1, A_2)) \end{aligned}$$

Nous pouvons à présent conclure. Par le lemme précédent appliqué à (1),

$$f(x_1, \ldots, x_p, n+1) \leq \mathcal{A}_{m+1}(\sup(x_1, \ldots, x_p, n, A_1, A_2)+k_1+(n+1)k_2)$$

$$f(x_1, \ldots, x_p, n+1) \leq \mathcal{A}_{m+1}(\sup(x_1, \ldots, x_p, n, A_1, A_2)+k_1+(n+1)k_2)$$

membre de droite : composée d'une fonction appartenant à $\mathfrak{C}_{m+1}$ (à savoir $\mathcal{A}_{m+1}$) avec une fonction de $\mathfrak{C}_2$. Pour $m \geq 1$, elle appartient donc à $\mathfrak{C}_{m+1}$ au vu du lemme 5. Pour $m = 0, 1$, elle appartient à $\mathfrak{C}_2$.

d'où la même conclusion pour f.

PROPOSITION

On a $\mathcal{PR} \subseteq \mathfrak{C}$.

Théorème

La fonction d'Ackermann n'est pas primitive récursive.

On procède par l'absurde. Supposons $\mathcal{A}$ de classe $\mathcal{PR}$. La fonction $f : \mathbb{N} \to \mathbb{N} : n \mapsto \mathcal{A}(n, 2n)$ est alors elle aussi de classe $\mathcal{PR}$. Au vu de la proposition précédente ($f \in \mathfrak{C}$), il existe des constantes C, m et k telles que pour tout $n > C$, $\mathcal{A}(n, 2n) \leq \mathcal{A}_m^k(n)$. Ainsi, en appliquant le lemme 6, pour tout $n > C$,

$$\mathcal{A}(n, 2n) \leq \mathcal{A}_m^k(n) \leq \mathcal{A}_{m+1}(n + k).$$

D'autre part, si $n > \sup(C, k, m + 1)$, alors

$$\mathcal{A}_{m+1}(n + k) < \mathcal{A}_{m+1}(2n) < A_n(2n) = \mathcal{A}(n, 2n)$$

d'où une contradiction.

Nous avons donc trouvé en la fonction d'Ackermann une fonction calculable (au sens naïf du terme, i.e., pour laquelle on dispose d'une procédure de calcul) qui n'est pas primitive récursive. Une méthode de *diagonalisation* permet, elle aussi, d'assurer l'existence d'une fonction calculable non primitive récursive.