Théorie des types : introduction et utilisation en Lean.

Dorian Guillet

9 avril 2025

Table des matières

1 Théorie des Types

1.1 Introduction

La théorie des types se veut être une alternative à la théorie des ensembles de Zermelo-Fraenkel (ZF). Or les fondations de cette dernière s'appuient sur le système déductif de la logique du premier ordre, et la théorie des ensembles est formulée à l'intérieur de ce système, donc cette théorie est formée de deux couches : la logique du premier ordre, puis la théorie des ensembles, formée de ses axiomes. On a donc deux objets fondamentaux dans cette approche : les propositions (qui se basent sur la logique du premier ordre) et les ensembles de ZF.

Pour éviter cette construction en deux couches, la théorie des types possède son propre système déductif, par conséquent une fois les jugements et les règles du système déductif définies, il est prêt à être utilisé sans axiomes supplémentaires. Ainsi, cette théorie ne possède qu'un objet fondamental : les **types**.

Ayant uniquement des types, pour pouvoir formuler des théorèmes on a besoin de l'équivalent des propositions qui sont ici des types particulier, dont la construction suit des règles qui seront précisées après. Dans cette théorie, prouver un théorème est donc équivalent à construire un objet qui serait un élément du type correspondant au théorème (ici l'objet en question est une preuve).

On peut aussi voir les types d'un point de vue plus proche de la théorie des ensembles, en interprétant le fait qu'un élément a soit de type A comme l'affirmation $a \in A$. Cependant ici l'affirmation $a \in A$ est une proposition alors que dire "a est de type A" (que l'on notera dorénavant $a : A$) est un jugement. En effet, dans la théorie des types un élément possède toujours un type (uniquement) déterminé.

Le système déductif de cette théorie est composé de deux jugements principaux :

1. **Jugement de typage** $a : A$, qui affirme que a est de type A.
2. **Jugement d'égalité** $a \equiv b : A$, qui affirme que a et b sont égaux par définition dans le type A.

A noter que le symbole "$\equiv$" est différent de "$=$". En effet, si $a, b : A$, alors on a le type $a =_A b$ qui correspond à une égalité que l'on peut prouver, c'est l'égalité propositionnelle. Pour l'égalité "par définition" du système déductif, la prouver ou la supposer n'a pas réellement de sens étant donné qu'elle est vrai par définition (ou par construction).

Cette distinction entre proposition et jugement est fondamentalle. Un jugement est une affirmation dans le système déductif de la théorie, qui est donc considéré comme vrai et n'est pas à prouver, alors qu'un proposition est un type, pouvant être non vide et se situe donc dans la théorie elle-même.

1.2 Construction de types

Pour construire un nouveau type à partir d'autres, on doit donner 3 règles :

— **Formation du type**, qui permet de construire un type à partir d'autres types ou famille de types,

— **Introduction**, qui explique comment sont construits les éléments de ce type,
— **Élimination**, décrivant comment utiliser ces éléments.

Partant de ces 3 règles, on peut construire un type et éventuellement ajouter des règles supplémentaires concernant leur comportement par rapport à l'égalité par définition.

Univers. A FAIRE.

Fonctions. Étant donné deux types $A, B : \mathcal{U}$, on peut former le type des fonctions de A dans B noté $A \to B$ grâce à la règle de formation suivante :

$$\frac{\Gamma \vdash A : \mathcal{U} \qquad \Gamma \vdash B : \mathcal{U}}{\Gamma \vdash A \to B : \mathcal{U}} (\to \text{f})$$

Une fonction est donc un élément f de type $A \to B$. Si on se donne $a : A$, on peut évaluer f en a ce qui donne un élément de type B que l'on note $f\ a$ ou bien $f(a)$ qui est appelé valeur de f en a, c'est la règle d'élimination de $\to$:

$$\frac{\Gamma \vdash f : A \to B \qquad \Gamma \vdash a : A}{\Gamma \vdash f\ a : B} (\to \text{e})$$

Pour former une fonction f de A dans B *ie.* un élément de type $A \to B$, la manière canonique de faire est d'utiliser les λ-abstraction et la règle d'introduction suivante :

$$\frac{\Gamma, x : A \vdash \Phi : B}{\Gamma \vdash \lambda(x : A).\Phi : A \to B} (\to \text{i})$$

Dans cette règle Φ correspond à une formule qui peut éventuellement faire intervenir la variable x.

Enfin, lorsque l'on considère une fonction $f : A \to B$ définie par $f :\equiv \lambda(x : A).\Phi$ où Φ est une formule et un élément $a : A$, il est naturel que la valeur de f en a soit égale à la formule Φ où l'on remplace les occurences de x par des a, ce que l'on note $\Phi[a/x]$. Ce comportement est en fait une règle d'égalité que l'on peut énoncer sous cette forme :

$$\frac{\Gamma, x : A \vdash \Phi : B \qquad \Gamma \vdash a : A}{\Gamma \vdash (\lambda(x : A).\Phi)\ a \equiv \Phi[a/x]} (\beta)$$

Exemple. Considérons un type $A : \mathcal{U}$ et la fonction "identité" de A définie par $I :\equiv \lambda(x : A).x$. Pour prouver que le type de I est bien $A \to A$, on peut utiliser les règles énoncées précédemment :

$$\cfrac{\cfrac{\cfrac{}{A : \mathcal{U} \vdash A : \mathcal{U}}}{A : \mathcal{U}, x : A \vdash x : A}}{A : \mathcal{U} \vdash \lambda(x : A).x : A \to A}$$

On a vu jusqu'ici comment construire des fonctions à une variable, on peut alors généraliser pour obtenir des fonctions à plusieurs variables. Pour ce faire, considérons trois types $A, B, C : \mathcal{U}$. Naïvement, on voudrais construire une fonction de type $A \times B \to C$, mais le type $A \times B$ n'est

pas encore défini. Pour contourner ce problème, on peut voir une fonction f de $A \times B \to C$ comme

$$f(x,y) = (\lambda(a:A).f(a,y))x = ((\lambda(a:A).(\lambda(b:B).f(a,b)))\ x)\ y \quad (1)$$
$$= (\lambda(a:A).\lambda(b:B).f(a,b))\ x\ y. \quad (2)$$

On a ainsi obtenu une fonction de type $A \to B \to C$ qui prend les mêmes valeurs que f grâce à ce processus de *curryfication*.

Exemple. Considérons deux types $A, B : \mathcal{U}$ et la fonction K définie par

$$K :\equiv \lambda(x:A).(\lambda(y:B).x)$$

Grâce à un arbre similaire au précédent, on trouve que le type de K est $A \to B \to A$ (on omet les parenthèses en prenant comme convention que $A \to A \to A$ est égal à $A \to (A \to A)$). De plus, on peut également montrer que si $a : A$ et $b : B$, alors $K\ a\ b \equiv a$. En effet, A FAIRE.

Si on considère deux types $A, B : \mathcal{U}$ et qu'on les voit comme des propositions, alors le type $A \to B$ peut s'interpreter comme l'implication $A \implies B$.

Fonctions dépendantes (Π-type). Une manière de généraliser le type des fonctions est de définir un type de fonctions dépendantes (ou $\prod$-type), qui est un type de fonction dont le codomaine dépend du point d'application de la fonction.

Pour former ce nouveau type, on considère $A : \mathcal{U}$ un type et $B : A \to \mathcal{U}$ une famille de type indexée sur A. On peut alors former le type $\prod_{a:A} B(a)$ grâce à la règle de formation suivante : REVOIR AVEC UNIVERS

$$\frac{\Gamma \vdash A : \mathcal{U} \qquad \Gamma, a : A \vdash B(a) : \mathcal{U}}{\Gamma \vdash \prod_{a:A} B(a) : \mathcal{U}}\ (\Pi\ \text{f})$$

On construit des éléments de type $\prod_{a:A} B(a)$ de manière analogue aux fonctions en utilisant des λ-abstractions. En effet, $\lambda(a:A).\Phi$ est de type $\prod_{a:A} B(a)$ si lorsque $a : A$, alors $\Phi : B(x)$. Cette règle énoncé plus formellement est la règle d'introduction de ce nouveau type :

$$\frac{\Gamma, a : A \vdash \Phi : B(a)}{\Gamma \vdash \lambda(a:A).\Phi : \prod_{a:A} B(a)}\ (\Pi\ \text{i})$$

Ensuite, la règle d'élimination est similaire à celle des types de fonctions. Si on considère $f : \prod_{a:A} B(a)$ et $a : A$, alors $f\ a$ est de type $B(a)$, ce que l'on peut traduire avec la règle suivante :

$$\frac{\Gamma \vdash f : \prod_{a:A} B(a) \qquad \Gamma \vdash a : A}{\Gamma \vdash f\ a : B(a)}\ (\Pi\ \text{e})$$

Également, la règle β des fonctions se généralise au $\prod$-type :

$$\frac{\Gamma, x : A \vdash \Phi : B(x) \qquad \Gamma \vdash a : A}{\Gamma \vdash (\lambda(x : A).\Phi)\ a \equiv \Phi[a/x]}\,(\beta)$$

L'utilité des types de fonctions dépendantes est notamment de pouvoir définir des fonctions de façon polymorphes, comme le montre les exemples suivants.

Exemple. La fonction identité du paragraphe précédent était défini sur un type $A : \mathcal{U}$ quelconque. On peut alors généraliser sa définition de la manière suivante :

$$I \equiv \lambda(A : \mathcal{U}).\lambda(a : A).\ a : \prod_{A:\mathcal{U}} A \to A.$$

Ainsi définie, la fonction I ne dépend pas d'un type préexistant.

On peut également redéfinir la fonction constante K d'une manière analogue :

$$K \equiv \lambda(A : \mathcal{U}).\lambda(B : \mathcal{U}).\lambda(x : A).\lambda(y : B).\ x : \prod_{A:\mathcal{U}} \prod_{B:\mathcal{U}} A \to B \to A.$$

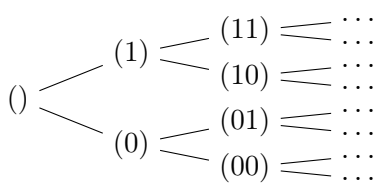
Pour ces types et définition polymorphe, on peut également avoir la notation I_A à la place de $I\ A$ pour plus de clarté.

D'un autre point de vue ce type peut également être vu comme l'équivalent du $\forall$ de la logique du première ordre. En effet, si on considère $A : \mathcal{U}$ un type et $C : A \to \mathcal{U}$ une famille de type, que l'on peut également voir comme un prédicat, alors le type $\prod_{a:A} C(a)$ se comporte comme la proposition $\forall a,\ C(a)$ en logique classique.

Types inductifs : W-types Pour construire le type des entiers $\mathbf{N}$, les constructeurs précédent ne suffisent pas car ils ne permettent pas de décrire le caractère inductif des entiers. Habituellement, on définit le type $\mathbf{N}$ comme étant un élément $0 : \mathbf{N}$ et une fonction successeur $s : \mathbf{N} \to \mathbf{N}$ et un entier est de la forme 0 ou bien $s \circ s \circ \ldots \circ s\ 0$. Une autre manière de le voir est de décrire $\mathbf{N}$ comme un arbre dont 0 est une feuille et chacun des noeuds est d'arité un (*ie.* possède au plus un fils).

$$0 \text{ —— } 1 \text{ —— } 2 \text{ —— } \cdots$$

Un autre exemple de type défini de manière inductive est le type $L(A)$ des listes sur un type $A : \mathcal{U}$. La seule feuille est la liste vide notée $()$ et la fonctions jouant le rôle de constructeur inductif est la fonction $c : A \to L(A) \to L(A)$ qui a un élément $a : A$ et une liste $l : L(A)$ renvoie la liste dont le premier élément est a et le reste est l. Ainsi, toutes les listes sont de la forme $() : L(A)$ ou bien $c\ a_1\ (c\ a_2\ (\ldots (c\ a_n\ ()))) : L(A)$.



Pour généraliser les types inductifs, on introduit alors les W-types (*well-founded trees*) de Martin-Löf. Pour ce faire, on utilise un type $A : \mathcal{U}$ qui correspond aux *étiquettes* et un type $B : A \to \mathcal{U}$ qui permet de coder l'arité des noeuds : un noeud dont l'étiquette est $a : A$ aura une arité de $B(a)$. Le type résultant de ces deux types est $W_{a:A}B(a)$ que l'on construit via la règle de formation suivante :

$$\frac{\Gamma \vdash A : \mathcal{U} \qquad \Gamma, a : A \vdash B(a) : \mathcal{U}}{\Gamma \vdash W_{a:A}B(a) : \mathcal{U}}\,\mathrm{W\ f}$$

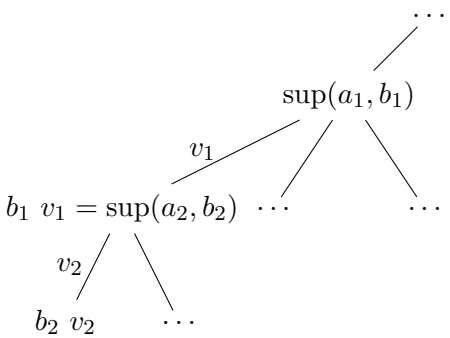
Les éléments d'un type $W_{a:A}B(a)$ sont donc des noeuds d'un arbre auxquels on associe la fonction donnant les noeuds précédents dans l'arbre. On construit ces éléments à l'aide de la règle d'introduction pour les W-types :

$$\frac{\Gamma \vdash a : A \qquad \Gamma, a : A \vdash b : B(a) \to W_{x:A}B(x)}{\Gamma \vdash \sup(a, b) : W_{x:A}B(x)}\,\mathrm{W\ i}$$

La fonction sup utilisé dans cette règle est donc de type

$$\sup : \prod_{a:A} (B(a) \to W_{x:A}B(x)) \to W_{a:A}B(a).$$

Chaque noeud $c :\equiv \sup(a, b) : W_{x:A}B(x)$ est donc relié aux autres par la fonction $b : B(a) \to W_{x:A}B(x)$, en effet si on considère un élément $v : B(a)$ alors $b\ v$ est un prédecesseur du noeud c dans l'arbre $W_{x:A}B(x)$.



Reprenons l'exemple du type $\mathbf{N}$ des entiers. On peut le décrire en terme de W-type en prenant $\mathbf{N} :\equiv W_{x:\mathbf{2}}B(x)$ où $B(0_{\mathbf{2}}) :\equiv \mathbf{0}$ et $B(1_{\mathbf{2}}) :\equiv \mathbf{1}$. En effet, il y a deux manières de construire un entier : ou bien c'est 0 ou bien il est de la forme $s\ k$ avec $k : \mathbf{N}$. Pour 0, son étiquette dans l'arbre est donc $0_{\mathbf{2}}$ tandis que les éléments de la forme $s\ k$ ont l'étiquette $1_{\mathbf{2}}$. On peut donc écrire 0 avec la fonction sup :

$$0 :\equiv \sup(0_{\mathbf{2}}, \mathrm{rec}_{\mathbf{0},\mathbf{N}}).$$

De même la fonction successeur $s : \mathbf{N} \to \mathbf{N}$ peut être définie de la manière suivante :

$$s :\equiv \lambda n.\ \sup(1_{\mathbf{2}}, \lambda(x : \mathbf{1}).\ n) : \mathbf{N} \to \mathbf{N}.$$

Et donc on a $1 :\equiv s\ 0 \equiv \sup(1_{\mathbf{2}}, \lambda x.\ 0)$ etc.

Enfin, pour prouver des propositions sur les W-types, on a besoin de la règle d'élimination qui s'énonce de la manière suivante :

$$\frac{\Gamma \vdash t : W_{x:A}B(x) \quad \Gamma, a : A, b : B(a) \to W_{x:A}B(x), c : \prod_{v:B(a)} E(f\ v) \vdash e(a,b,c) : E(\sup(a,b))}{\Gamma \vdash \mathrm{rec}(E,e,t) : E(t)}\ \mathrm{W\ e}$$

2 Formalisation du théorème de Bowen en Lean

Cette section est un retour d'expérience sur l'utilisation de Lean en tant qu'outil de formalisation pour les mathématiques.

Durant ce TER, j'ai donc utilisé Lean pour formaliser le théorème de Bowen, dont un preuve figure dans mon rapport de stage en annexe (preuve initialement dûe à R. Bowen). Ce théorème établit l'existence et l'unicité d'une certaine mesure de probabilité dite de Gibbs sur un espace métrique particulier, et donc ce théorème fait intervenir notamment de la topologie et de la théorie de la mesure.

Pour formaliser ce théorème, j'ai donc eu recours à des outils extérieurs à Lean comme leanblueprint, qui permet à partir d'un texte mathématiques d'obtenir un graphe décrivant comment le résultat final dépend des lemmes, théorèmes et définitions intermédiaires et d'en donner l'état d'avancement dans le code Lean associé.

3 Topologie des espaces ultramétriques

Pour montrer l'unicité dans le théorème de Bowen, on a besoin d'un résultat de topologie de certains espaces, appelés espaces ultramétriques, qui sont des espaces métriques où l'inégalité triangulaire habituelle est remplacée par une inégalité dite ultramétrique :

$$\forall x, y, z \in E, d(x,z) \leq \max\left(d(x,y), d(y,z)\right).$$

Cette inégalité donne de nombreux résultats topologiques très différents des espaces métriques, notamment sur les triangles et les boules.

Le théorème permettant de prouver l'unicité du théorème de Bowen est le suivant, et nous allons en donner une preuve dans cette section, puis une formalisation possible en Lean.

Théorème 3.1. *Soit (E,d) un espace ultramétrique et $\mathcal{O}$ un ouvert borné de E, alors il existe une partie $R \subseteq \mathcal{O}$ et une application $r\colon R \longrightarrow \mathbf{R}_+^*$ tels que*

$$\mathcal{O} = \bigsqcup_{x \in R} B(x, r(x)).$$

De plus, si E est séparable alors on peut imposer que R soit au plus dénombrable.

Dans la suite on fixe (E,d) un espace ultramétrique Commençons par quelques lemmes décrivant les relations entres les boules dans ces espaces.

Lemme 3.2. *Soit $x, y \in E$ et un réel $r > 0$, si $y \in B(x,r)$, alors $B(x,r) = B(y,r)$*

Démonstration. Supposons que $y \in B(x,r)$. Soit $z \in B(x,r)$, alors

$$d(y,z) \leq \max\left(d(y,x), d(x,z)\right) \leq \max\left(r,r\right) \leq r,$$

car $y, z \in B(x,r)$. Donc on a une inclusion : $B(x,r) \subseteq B(y,r)$.

Réciproquement comme $y \in B(x,r)$ on a $x \in B(y,r)$ par symétrie de la distance et donc avec le même raisonnement on a l'autre inclusion. Finalement, $B(x,r) = B(y,r)$. □

Lemme 3.3. *Soit $x, y \in E$ et deux réels $r, s > 0$, alors il y a trois possibilités :*

1. $B(x,r) \subseteq B(y,s)$,
2. $B(y,s) \subseteq B(x,r)$,
3. $B(x,r) \cap B(y,s) = \emptyset$.

Démonstration. Supposons que $B(x,r) \cap B(y,s) \neq \emptyset$, alors soit $z \in B(x,r) \cap B(y,s)$. On peut alors écrire $B(x,r) = B(z,r)$ et $B(y,s) = B(z,s)$ grâce au lemme 3.2, et ainsi on a bien le résultat voulu en fonction de l'ordre de r et s. □

Lemme 3.4. *Soit $(x_i)_{i\in I}$ une famille de point de E et $(r_i)_{i\in I}$ une famille de réels strictement positifs. De plus, supposons qu'il existe $x \in E$ tel que $x \in B(x_i, r_i)$ pour tout $i \in I$ et que $R = \sup_{i\in I} r_i < \infty$. Alors*

$$\bigcup_{i\in I} B(x_i, r_i) = B(x, R).$$

Démonstration. Commençons par réécrire chaque boule en changeant son centre grâce au lemme 3.2 et notons $R = \sup_{i\in I} r_i < \infty$, on a alors l'inclusion

$$\bigcup_{i\in I} B(x_i, r_i) = \bigcup_{i\in I} B(z, r_i) \subseteq B(z, R).$$

Réciproquement si $y \in B(z,R)$, alors il existe $i_0 \in I$ tel que $d(y,z) \leq r_{i_0} \leq R$ et donc $y \in B(z, r_{i_0}) \subseteq \bigcup_{i\in I} B(z, r_i)$. Finalement, l'union de ces boules est encore une boule. □

On notera dans la suite $\mathcal{O} \subseteq E$ un ouvert borné, et donc on a pour chaque $x \in \mathcal{O}$ un réel $r_x > 0$ tel que $B(x, r_x) \subseteq \mathcal{O}$. On a alors

$$\mathcal{O} = \bigcup_{x\in\mathcal{O}} B(x, r_x).$$

Remarque. On peut voir les $(r_x)_{x\in\mathcal{O}}$ comme une une application $r \colon \mathcal{O} \longrightarrow \mathbf{R}_+^*$.

Definition 3.1. On note $B(\mathcal{O})$ l'ensemble des boules incluses dans $\mathcal{O}$, en particulier :

$$B(\mathcal{O}) := \{B(x,r) \mid x \in \mathcal{O} \wedge r > 0 \wedge B(x,r) \subseteq \mathcal{O}\} \subseteq \mathcal{P}(E).$$

Soit $\mathcal{U} \subseteq B(\mathcal{O})$, on définit une relation $\sim_{\mathcal{U}}$ (ou plus simplement $\sim$ s'il n'y a pas d'ambuiguité) sur $\mathcal{U}$ par :

$$\forall u, v \in \mathcal{U}, \quad u \sim_{\mathcal{U}} v \iff \exists w \in \mathcal{U}, u \cup v \subseteq w.$$

Proposition 3.5. *Pour tout $\mathcal{U} \subseteq B(\mathcal{O})$, la relation $\sim_{\mathcal{U}}$ est une relation d'équivalence.*

Démonstration. Soit $u_1, u_2, u_3 \in \mathcal{U}$ trois boules.

Pour la reflexivité, on considère $w = u_1 \in \mathcal{U}$ et on a bien $u_1 \sim u_1$.

Pour la symétrie, la définition de la relation est clairement symétrique.

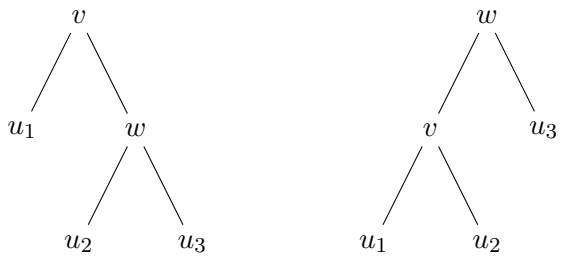
Enfin pour la transitivité, on suppose que $u_1 \sim u_2$ et $u_2 \sim u_3$. On a alors $v, w \in \mathcal{U}$ tels que

$$u_1 \subseteq v, u_2 \subseteq v, u_2 \subseteq w \text{ et } u_3 \subseteq w.$$

Ainsi, $\emptyset \neq u_2 \subseteq v \cap w$, or v et w sont des boules d'un espace ultramétrique. On a alors deux cas :

— $v \subseteq w$ dans ce cas on a $u_1 \overset{v}{\sim} u_3$

— $w \subseteq v$ et alors on a $u_1 \overset{w}{\sim} u_3$



Donc $\sim_{\mathcal{U}}$ est bien une relation d'équivalence sur $\mathcal{U}$. □

Pour la suite de la section on fixe $\mathcal{U} \subseteq B(\mathcal{O})$ un sous-ensemble de l'ensemble des boules incluses dans $\mathcal{O}$.

Definition 3.2. Soit $u \in \mathcal{U}$, la classe d'équivalence de u pour la relation $\sim$ est noté $\bar{u}$. On introduit également $\mathcal{B}_u \in B(\mathcal{O})$ l'union des éléments de la classe d'équivalence de u :

$$\mathcal{B}_u := \bigcup_{u \sim u'} u' \subseteq \mathcal{O}.$$

Remarquons que $\mathcal{B}_u$ est une boule dès lors que cette union est bornée (d'après le lemme 3.4), or cette $\mathcal{B}_u \subseteq \mathcal{O}$ et $\mathcal{O}$ est borné donc $\mathcal{B}_u$ est toujours une boule.

Definition 3.3. Soit R un système complet de représentant pour la relation $\sim$. On pose alors

$$\mathcal{U}' := \mathcal{U} \cup \{\mathcal{B}_u \mid u \in R\} \subseteq B(\mathcal{O}).$$

car $\mathcal{B}_u \in B(\mathcal{O})$ pour chaque $u \in R$.

On considère désormais $\sim' = \sim_{\mathcal{U}'}$ qui est encore une relation d'équivalence, et soit R' un système complet de représentant pour la relation $\sim'$.

Théorème 3.6. *Soit $\mathcal{O}$ un ouvert borné de E. Avec les notations introduites précédemment on a*

$$\mathcal{O} = \bigsqcup_{u \in R'} \mathcal{B}_u.$$

Démonstration. On commence par montrer que cette union est bien égale à $\mathcal{O}$. En considérant $\mathcal{U} = \{B(x, r_x) \mid x \in \mathcal{O}\}$,

$$\mathcal{O} = \bigcup_{u \in \mathcal{U}} u = \bigcup_{u \in \mathcal{U}'} u = \bigcup_{u \in R'} \left(\bigcup_{v \sim' u} v \right) = \bigcup_{u \in R'} \mathcal{B}_u$$

Il reste encore à montrer que cette union est disjointe. Supposons que $\mathcal{B}_u \cap \mathcal{B}_v \neq \emptyset$. Alors on a $\mathcal{B}_u \subseteq \mathcal{B}_v$ ou $\mathcal{B}_v \subseteq \mathcal{B}_u$ par le lemme 3.3. Dans les deux cas on a $u \sim' v$ avec $w = \mathcal{B}_v \in \mathcal{U}'$ dans le premier cas et $w = \mathcal{B}_u \in \mathcal{U}'$ dans le second. Or R est un système complet de représentant pour $\sim'$ et donc nécessairement $u = v$. Finalement les $(\mathcal{B}_u)_{u \in R'}$ sont deux à deux disjoints et leur union est égale à $\mathcal{O}$, ce qui conclut la preuve du théorème 3.1. □

Avec la même preuve on obtient en changeant une hypothèse la même conclusion.

Corollaire 3.6.1. *Soit E un espace ultramétrique tel qu'il existe $(x_i)_{i \in I} \in E^I$ et $(r_i)_{i \in I}$ des réels strictement positifs vérifiant la propriété suivante :*

$$E = \bigcup_{i \in I} B(x_i, r_i) \qquad \text{et} \qquad \sup_{i \in I} r_i < \infty.$$

Alors pour tout ouvert $\mathcal{O}$ de E, il existe $R \subseteq E$ et $r \colon R \longrightarrow \mathbf{R}_+^*$ tel que

$$\mathcal{O} = \bigsqcup_{x \in R} B(x, r(x)).$$

Démonstration. On réécrit $\mathcal{O}$ comme l'union suivante :

$$\mathcal{O} = \bigsqcup_{i \in I} B(x_i, r_i) \cap \mathcal{O}.$$

Cette union est disjointe car les boules des espaces ultramétriques sont disjointe ou bien incluses l'une dans l'autre, donc on peut se ramener à ce cas (A VÉRIFIER). Ainsi, pour tout $i \in I$, $B(x_i, r_i) \cap \mathcal{O}$ est un ouvert borné et donc peut être écrit comme une union disjointe de boule. De cette manière on obtient le résultat voulu. □