Chapitre 8

Raisonnements inductifs

Notions	Commentaires
Ensemble ordonné, prédécesseur et successeur, prédécesseur et successeur immédiat. Élement minimal. Ordre produit, ordre lexicographique. Ordre bien fondé.	On fait le lien avec la notion d'accessibilité dans un graphe orienté acyclique. L'objectif n'est pas d'étudier la théorie abstraite des ensembles ordonnés mais de poser les définitions et la terminologie.
Ensemble inductif, défini comme le plus petit ensemble engendré par un système d'assertions et de règles d'inférence. Ordre induit. Preuve par induction structurelle.	On insiste sur les aspects pratiques : construction de structure de données et filtrage par motif. On présente la preuve par induction comme une généralisation de la preuve par récurrence.

Extrait de la section 2 du programme officiel de MP2I : « Récursivité et induction ».

Sommaire

Chapitre non-encore rédigé.

0 Relations binaires (rappels de maths)

Dans toute cette section, A, B et E sont des ensembles.

Définition 1 (Relation binaire).
Une **relation binaire** $\mathscr{R}$ sur $A \times B$ est une partie de $A \times B$. Intuitivement, c'est la donnée d'un ensemble de paires (a, b) tels que a et b sont « reliés » (selon un certain critère qui dépend de la relation).
Pour tous $(a, b) \in A \times B$, on note $a \mathscr{R} b$ lorsque $(a, b) \in \mathscr{R}$, c'est à dire lorsque a et b sont « reliés » ; et on note $a \not\mathscr{R} b$ sinon.
Si $A = B$, on parle de relation binaire **homogène**.

Exemple.

- Une fonction $f : A \to B$ est une relation binaire : les paires sont les $(x, f(x))$. Autrement dit, la relation est $\{(x, f(x)) \; t.q. \; x \in A\}$.
- $\leq_{\mathbb{R}}$ est une relation binaire homogène sur $\mathbb{R}$.
- $\leq_{\mathbb{N}}$ est une relation binaire homogène sur $\mathbb{N}$.
- $\in$ est une relation binaire sur $E \in \mathscr{P}(E)$.
- $\subseteq$ est une relation binaire homogène sur $\mathscr{P}(E) \times \mathscr{P}(E)$.
- $=$ est une relation binaire homogène sur E.
- $\vdash$, la relation telle que $P \vdash Q$ signifie « la propriété P prouve la propriété Q », est une relation binaire homogène sur les formules logiques.

Dorénavant, $\mathscr{R}$ dénote une relation binaire homogène sur E.

0.0 Relations d'équivalence

Définition 2 (Relation d'équivalence).
On dit que $\mathscr{R}$ est :

- **réflexive** lorsque pour tout $x \in E$, $x \mathscr{R} x$.
- **symétrique** lorsque pour tout $(x, y) \in E^2$, on a $x \mathscr{R} y$ si et seulement si $y \mathscr{R} y$.
- **transitive** lorsque pour tout $(x, y, z) \in E^3$, si $x \mathscr{R} y$ et $y \mathscr{R} z$ alors $x \mathscr{R} z$.

Une relation réflexive symétrique transitive est appelée **relation d'équivalence**.

Exemple.

- $=$ sur E
- Dans $\mathbb{Z}$, pour tout $p \in \mathbb{Z}^*$ fixé, la relation « avoir le même reste modulo p » est une relation d'équivalence.
- « renvoyer les mêmes sorties sur les mêmes entrées » est une relation d'équivalence sur les programmes.
- Dans un graphe non-orienté, l'accessibilité est une relation d'équivalence.

FIGURE 8.1 – Un graphe non-orienté et la relation d'accessibilité

Définition 3 (Classes d'équivalence).
Soit $\mathscr{R}$ une relation d'équivalence sur E.
Pour tout $x \in E$, on définit la **classe d'équivalence** par $\mathscr{R}$ de x, notée $[x]_{\mathscr{R}}$, comme :

$$[x]_{\mathscr{R}} = \{y \in E \, t.q. \, x \mathscr{R} y\}$$

Pour tout $y \in [x]_{\mathscr{R}}$, on dit que y est un **représentant** de la classe d'équivalence $[x]_{\mathscr{R}}$.

Exemple.

- Pour la congruence modulo p, les classes d'équivalence sont les éléments de $\mathbb{Z}/p\mathbb{Z}$. Les représentants canoniquement utilisés sont $0, 1, ..., p-1$:

FIGURE 8.2 – Classes d'équivalence de la congruence modulo p.

- Pour l'équivalence des sorties des programmes, une classe d'équivalence est un ensemble de programmes qui calculent la même chose. Mais ils peuvent faire ces calculs différemment, avec des raisonnements différents voire des complexités différentes. Par exemple, il existe de nombreux algorithmes de tri mais ils fonctionnent très différemment.
- Les classes d'équivalence de l'accessibilité dans un graphe non-orienté sont les composantes connexes (C.C.). Dans un graphe orienté, on peut dire que x et y sont *mutuellement accessibles* s'il y a un chemin de x à y et de y à x. C'est une relation d'équivalence, dont les classes d'équivalence sont les composantes fortement connexes (C.F.C.) :

FIGURE 8.3 – Un graphe orienté et ses C.F.C.

Théorème 4 (Partitionnement par les classes d'équivalence).
Soit $\mathscr{R}$ une relation d'équivalence sur E. Alors :

- Tout élément appartient à sa classe d'équivalence. Formellement, pour tout $x \in E$ on a $x \in [x]_{\mathscr{R}}$.
- Deux éléments sont en relation si et seulement si ils ont la même classe d'équivalence. Formellement, pour tout $(x, y) \in E^2$, on a $x \mathscr{R} y$ si et seulement si $[x]_{\mathscr{R}} = [y]_{\mathscr{R}}$.
- Deux classes d'équivalence sont soit égales soit disjointes. Formellement, pour tout $(x, y) \in E^2$, on a $[x]_{\mathscr{R}} = [y]_{\mathscr{R}}$ ou bien $[x]_{\mathscr{R}} \cap [y]_{\mathscr{R}} = \emptyset$

En particulier, les classes d'équivalences de $\mathscr{R}$ partitionnent E.

Démonstration.

- Par réflexivité de $\mathscr{R}$.
- Soient $(x, y) \in E^2$. Procédons par double implication :
  - $\Leftarrow$) Si $[x]_{\mathscr{R}} = [y]_{\mathscr{R}}$, alors $x \in [x]_{\mathscr{R}} = [y]_{\mathscr{R}}$ donc $x \mathscr{R} y$.
  - $\Rightarrow$) Si $x \mathscr{R} y$, montrons l'égalité des classes. Pour tout $z \in E$, on a :

$$z \in [x]_{\mathscr{R}} \iff z \mathscr{R} x \overset{\mathscr{R} \text{ transitive}}{\iff} z \mathscr{R} y \iff z \in [y]_{\mathscr{R}}$$

- Soient $(x, y) \in E^2$. Supposons que $[x]_{\mathscr{R}} \cap [y]_{\mathscr{R}} \neq \emptyset$ et soit z dans cette intersection. Donc $z \mathscr{R} x$ et $z \mathscr{R} y$. Donc par transitivité $x \mathscr{R} y$, donc d'après le point précédent : $[x]_{\mathscr{R}} = [y]_{\mathscr{R}}$.

□

Exemple. On peut partitionner les programmes en fonction de ce qu'ils calculent (de leurs entrées/sorties).

0.1 Relations d'ordres

$\mathscr{R}$ désigne encore une relation homogène sur E.

Définition 5 (Relation d'ordre).
On dit que $\mathscr{R}$ est :

- **antiréflexive** lorsque pour tout $x \in E$, $x \not\mathscr{R} x$.
- **antisymétrique** lorsque pour tout $(x, y) \in E^2$ avec $x \neq y$, on n'a pas à la fois $x \mathscr{R} y$ et $y \mathscr{R} x$.

Une relation réflexive, antisymétrique et transitive est appelée **relation d'ordre** (aussi appelée **relation d'ordre large**).
Une relation antiréflexive, antisymétrique et transtive est appelé **relation d'ordre stricte**.

Remarque. Une autre définition de l'antisymétrie est « pour tout $(x, y) \in E^2$, $x \mathscr{R} y$ et $y \mathscr{R} x$ implique $x = y$ ».

Exemple.

- $\leq_{\mathbb{Z}}$ est une relation d'ordre large sur $\mathbb{Z}$.
- $<_{\mathbb{Z}}$ est une relation d'ordre strict sur $\mathbb{Z}$.
- $\subseteq$ est une relation d'ordre large sur $\mathscr{P}(E)$.
- $\subsetneq$ est une relation d'ordre stricte sur $\mathscr{P}(E)$.
- $\vdash$ n'est *pas* une relation d'ordre, puisque deux propriétés équivalentes se prouvent mutuellement.

Définition 6 (Ordre partiel, ordre total).
Soit $\mathscr{R}$ une relation d'ordre sur E.
Soient $(x, y) \in E^2$. On dit que x et y sont comparables par $\mathscr{R}$ lorsque $x\mathscr{R}y$ ou $y\mathscr{R}x$.
On dit que $\mathscr{R}$ est **total** si tous éléments x et y sont comparables. Dans le cas contraire, on dit que l'ordre est **partiel**.

Remarque. En particulier, un ordre total est un ordre large puisque x et x doivent être comparables.

Exemple.

- $\leq_{\mathbb{Z}}$ est un ordre total.
- $<_{\mathbb{Z}}$ est un ordre partiel (mais « presque » total).
- $\subseteq$ est un ordre partiel.
- $=$ sur E est un ordre partiel (c'est même le plus petit ordre large).
- La relation $\emptyset$ (celle qui ne met *aucun* élément en relation) est un ordre strict partiel (c'est même le plus petit). Très très très partiel même !
- Sur $\mathbb{C}$, la relation $\{z_0, z_1 \ t.q. \ Re(z_0) \leq Re(z_1) \ et \ Im(z_0) \leq Im(z_1)\}$ est une relation d'ordre partiel. On verra bientôt qu'il s'agit de l'ordre produit sur $\mathbb{C}$. Visuellement, c'est l'ordre qu'un dit qu'un complexe est inférieur à un ordre s'il est « en dessous à gauche » dans le plan complexe :

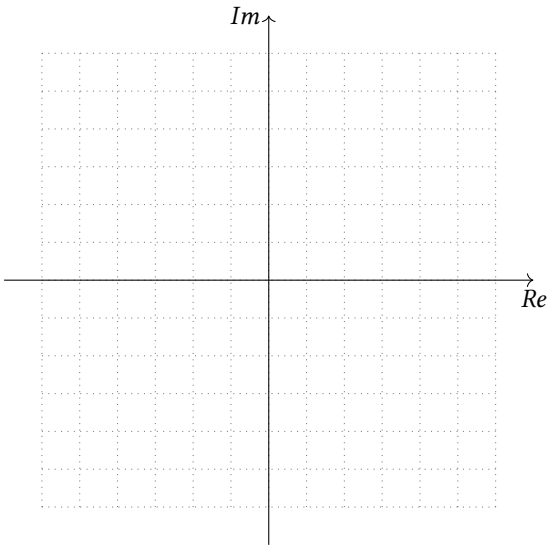

FIGURE 8.4 – Ordre produit sur $\mathbb{C}$

Définition 7 (Restriction et élargissement d'un ordre).
Si $\mathscr{R}$ un ordre strict, posons $\mathscr{T} = \mathscr{R} \cup \{(x, x) \ t.q. \ x \in E\}$ (c'est à dire $\mathscr{R}$ à laquelle on ajoute la réflexivité). Alors $\mathscr{T}$ est le plus petit ordre large (au sens de $\subseteq$) qui contient $\mathscr{R}$. On dit que c'est **l'ordre large associé à** $\mathscr{R}$.
Réciproquement, si $\mathscr{R}$ est un ordre strict, posons $\mathscr{T} = \mathscr{R} \backslash \{(x, x) \ t.q. \ x \in E\}$ (c'est à dire $\mathscr{R}$ à laquelle on enlève la réflexivité). Alors $\mathscr{T}$ est le plus grand ordre strict contenu dans $\mathscr{R}$. On dit que c'est **l'ordre strict associé à** $\mathscr{R}$.

Démonstration. C'est un exercice de maths, mais faisons-le (un peu) quand même.

- Montrons d'abord que $\mathscr{T}$ est un ordre. Il est :
  - Réflexif (par définition).
  - Anti-symétrique car pour tous $(x, y) \in E^2$ avec $x \neq y$, si xTy alors $x\mathscr{R}y$ (car $\mathscr{T} \backslash \mathscr{R} = \{(x, x) \ t.q. \ x \in E\}$). De même si $y\mathscr{R}x$. On conclut par antisymétrie de $\mathscr{R}$.

  - Transitif car on a transitive pour trois x, y, z distincts par transitivité de $\mathscr{R}$, et les cas d'égalité se gèrent très bien.

  Montrons maintenant qu'il s'agit du plus petit ordre large. Soit $\mathscr{T}'$ un ordre large contenat $\mathscr{R}$. Comme il est large, $\mathscr{T}'$ est réflexif et contient donc $\{(x,x)\ t.q.\ x \in E\}$. Il contient également par définition $\mathscr{R}$, et contient donc tout $\mathscr{T}$. D'où le résultat.
- La construction réciproque est similaire.

□

Définition 8 (Prédecesseur, prédecesseur strict, prédecesseur immédiat).
Soit $\mathscr{R}$ un ordre sur E, et $(x,y) \in E^2$. On dit que :

- x est un **prédecesseur** de y si $x \mathscr{R} y$.
  Si de plus $x \neq y$, on dit que c'est un **prédécesseur strict** de y.
  Si en plus de $x \neq y$ il n'existe aucun z tel que $x \mathscr{R} z$ et $z \mathscr{R} y$, on dit que x est un **prédecesseur immédiat** de y.
- y est un **successeur** de x si x précède y.
  Si de plus $y \neq x$, on dit que y est un **successeur strict**.
  Si de plus x précède immédiatement y, on dit que y est un **successeur immédiat** de x.

Ces notions se visualisent très bien sur un schéma. On représente tous les éléments, et on met une flèche de x vers y si x précède immédiatement y.

Exemple. Deux exemples de schémas d'ordres :

(a) $\leq$ sur $\mathbb{N}$ (b) $\subseteq$ sur $\mathscr{P}(\llbracket 0;2 \rrbracket)$

Figure 8.5 – Deux ordres et les schémas associés

Remarque.

- $\subseteq$ est un exemple d'ordre pour lequel il n'y a pas unicité d'un prédecesseur immédiat.
- Même si un élément admet des prédécesseurs, il peut n'admettre aucun prédécesseur immédiat : c'est parce exemple le cas dans les ordres denses, comme $\leq_{\mathscr{R}}$ sur $\mathbb{R}$.

Définition 9 (Ensemble ordonné).
Si $\mathscr{R}$ est un ordre sur E, on dit que $(E, \mathscr{R})$ est un **ensemble ordonné**. Si l'ordre est total, on dit qu'il s'agit d'un **ensemble totalement ordonné**.

Théorème 10.
Tout ensemble peut-être muni d'un ordre total.

Démonstration. Admis. □

Définition 11 (Élément minimal, minimum).
Un élément est dit minimal s'il n'a aucun prédecesseur strict.
Un élément minimal (resp.maximal) est un **minimum** (resp. **maximum**) s'il est de plus comparable à tous les éléments.

Proposition 12 (Unicité du minimum).
Le minimum (resp.maximum) de E, s'il existe, est unique.

Démonstration. Supposons que E admette m et m' deux minimas, et montrons que $m = m'$. Par définition d'un minimum, m et m' sont comparableS. Par définition d'un minimum, on a $m \leq m'$ et $m' \leq m$ donc pas anti-symétrie $m = m'$. □

Définition 13 (Minorant, inf).

- Soit $A \subseteq E$ et $m \in E$. On dit que m est un **minorant** (resp. **majorant**) de A si m est plus petit (resp. plus grand) que tous les éléments de a.
  Formellement, $m \in E$ est un minorant de A si pour tout $a \in A$, on a $m \leq a$ (resp. $a \leq m$).
- S'il existe, on appelle $\inf(A)$ (resp. $\sup(A)$) le plus grand des minorants (resp. le plus petit des majorants).

Exemple. Dans $\left(\mathscr{P}(\llbracket 0;2 \rrbracket), \subseteq\right)$, en posant $A = \{\{0\};\{2\}\}$ on a $\sup(A) = \{0;2\}$.

0.2 Suite monotone

Dans cette partie, $(E, \leq_E)$ et $(F, \leq_F)$ sont deux ensembles ordonnés (pas forcément totalement ordonnés). On note $<_E$ et $<_F$ les ordres stricts associés.

Définition 14 (Fonction croissante).
On dit qu'une fonction $f : E \to F$ est :

- **croissante** si pour tous $(x, y) \in E^2$ on a $x \leq_E y$ qui implique $f(x) \leq_F f(y)$. Si de plus $f(x) <_F f(y)$, on dit que la fonction est **strictement croissante**.
- **décroissante** si pour tous $(x, y) \in E^2$ on a $x \leq_E y$ qui implique $f(y) \leq_F f(x)$. Si de plus $f(y) <_F f(x)$, on dit que la fonction est **strictement décroissante**.

On dit qu'une fonction est monotone (resp. strictement monotone) si elle est croissante ou décroissante (resp. strictement croissante ou strictement décroissante).

Définition 15 (Suite croissante).
Soit (u_n) une suite d'éléments de E, finie ou infinie. On note I l'ensemble des indices de la suite, avec $I \subseteq \mathbb{N}$. On dit que :

- (u_n) est croissante (resp. strictement croissante) si $f : I \to E$ est croissante (resp. strictement croissante). Plus intuitivement, (u_n) est croissante si pour tout i, $u_i \leq_E u_{i+1}$.
- (u_n) est décroissante (resp. strictement décroissante) si $f : I \to E$ est décroissante (resp. strictement décroissante). Plus intuitivement, (u_n) est décroissante si pour tout i, $u_{i+1} \leq_E u_i$.

1 Ordres bien fondés et induction

Le but de la section est de montrer que la raison pour laquelle les preuves par récurrence fonctionne est que la véracité du cas de base se « transmet » de proche en proche à tous les cas. Autrement dit, on peut faire des récurrences si et seulement si toute suite décroissante finit par tomber sur un cas de base.

(a) Récurrence sur $\mathbb{N}$ (b) Induction sur $\mathscr{P}(\llbracket 0;2 \rrbracket)$

FIGURE 8.6 – Illustration de la récurrence

1.0 Ordres bien fondés

Dans cette sous-section, $(E, \leq)$ est un ensemble ordonné.

Définition 16 (Ordre bien fondé).
On dit que $\leq$ est un **ordre bien fondé** sur E si toute partie non-vide de E admet un élément minimal.

Exemple. Visualisons cela dans quatre cas :

(a) $\mathbb{N}$ muni de l'ordre usuel (b) $\mathbb{N}$ avec l'ordre usuel restreint aux éléments pairs

(c) $\mathbb{N}$ muni de $\geq$ n'est pas bien fondé (d) $\mathscr{P}(\llbracket 0;2 \rrbracket)$ muni de $\subseteq$

FIGURE 8.7 – Ordres bien fondés ou non

Théorème 17.
$(\mathbb{N}, \leq)$ est bien fondé.

Démonstration. Admis. Le prouver demande de savoir ce qu'est $\mathbb{N}$, c'est à dire comment $\mathbb{N}$ est construit : c'est compliqué. □

Théorème 18 (Caractérisation des ordres bien fondés).
$\leq$ est bien fondé sur E si et seulement si il n'existe pas de suite infinie strictement décroissante dans $(E, \leq)$.

Démonstration. Procédons par double inclusion :

⇒) Par contraposition : soit $(u_n)_{n\in\mathbb{N}}$ une suite infinie strictement décroissante, montrons que $\leq$ n'est pas bien fondé. Pour cela, considérons la partie $A = \{u_n \ t.q. \ n \in \mathbb{N}\}$: elle n'admet pas d'élément minimal puisque pour tout $u_i \in A$, on a $u_{i+1} < A$.

⇐) Supposons qu'il n'y a pas de suite infinie strictement décroissanten et procédons par l'absurde : soit $A \subseteq E$ sans élément minimal et $a_0 \in A$. Comme A est sans élément minimal, il existe $a_1 < a_0$. De même, il existe $a_2 < a_1$ et ainsi de suite : par axiome du choix dépendant [1], on obtient une suite (a_n) strictement décroissante.

□

Remarque. C'est généralement le critère le plus simple à manipuler pour montrer qu'un ordre est bien fondé ou non, comme par exemple pour les 4 exemples précédents.

Intuitivement, les ordres bien fondés sont importants car ce sont les ordres qui ont des « cas de bases » (les fameux éléments minimaux demandés dans la définition). Or, les cas de base sont capitaux pour faire des récurrences !

1.1 Variants de boucle/appel

Une première application de cette notion d'ordre bien fondé est la notion de variant !

Définition 19 (Variants (S2)).
Un variant de boucle (resp. d'appels récursifs) est une qantité qui décroit strictement d'une itération à l'autre (resp. d'un appel à l'autre) selon un ordre bien fondé.

Proposition 20.
Une boucle (resp. une suite d'appels récursifs) qui admet un variant itère un nombre fini de fois (resp. contient un nombre fini d'appels).

Démonstration. Si elle itérait un nombre inifi de fois, les valeurs successives du variant donnerait une suite infinie strictement décroissante selon un ordre bien fondé : absurde. □

Exemple.

- Une suite d'entiers minoré strictement décroissant donne un variant.
- Une suite d'entiers majoré strictement croissant donne un variant.
- Une suite d'ensembles finis décroissante pour $\subsetneq$, c'est à dire tels que $E_{i+1} \subsetneq E_i$, donne un variant [2] ! Autrement dit, il n'y a pas besoin de dire que le cardinal de l'ensemble est un variant : l'ensemble lui-même est un variant !
- Plus d'exemples avec l'ordre produit et lexicographique !

1. Et non par récurrence. C'est subtil, on s'en fiche à notre niveau, mais c'est important : on ne fait pas encore de récurrence (on est en train de la construire !).
2. Cf SLOWSELECT vu en DS, ou le retour sur trace de SUBSETSUM.

1.2 Les inductions

1.2.0 Ce qu'il faut savoir

Définition 21 (Preuve par induction).
Soit I une propriété portant sur des éléments de E. Pour prouver I par induction, on doit prouver :

- Initialisation : montrer que tous les éléments minimaux pour $\leq$ vérifient I.
- Hérédité : soit x non-minimal. On doit montrer que si tous les prédecesseurs stricts de x vérifient I, alors x le vérifie aussi.

Sur $(\mathbb{N}, \leq)$, l'induction s'appelle récurrence forte.

Théorème 22 (Induction ssi bien fondé).
Une preuve par induction (basée sur l'ordre $\leq$) prouve qu'une propriété est vraie pour tous les élements de E si et seulement si l'ordre $\leq$ est bien fondé.

Exemple. Un *mobile de Calder* est une succession de « barre » qui chacune portent soit un objet, soit une autre barre :

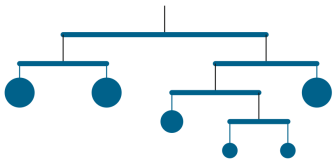

FIGURE 8.8 – Un mobile de Calder (src : Balabonski, Conchon Filliâtre, Nguyen, Sartre)

Autrement dit, un mobile de Calder est :

- Soit un objet
- Soit une barre reliant deux mobiles de Calder.

FIGURE 8.9 – Construction inductive des mobiles de Calder

On veut montrer que dans tout mobile de Calder m, le nombre $obj(m)$ d'objets est supérieur de 1 au nombre $bar(m)$ de barres. On admet que l'ordre « $m_0 \preccurlyeq m_1$ lorsque m_0 apparait dans m_1 » est un ordre bien fondé (on verra plus tard qu'il s'agit de l'ordre structurel). Procédons par induction :

- Initialisation : les cas de base sont les mobiles qui ne contiennent pas d'autres mobiles, autrement dit les mobiles réduits à des objets. Pour un tel mobile, il y a 0 barre et 1 objet : la propriété est bien initialisée.
- Hérédité : soit m un mobile non-réduit à un objet et supposons la propriété vraie pour tout autre mobile $m' \preccurlyeq m$. Comme m n'est pas réduit à un objet, m est une barre reliant deux mobiles m_1 et m_2.
  Donc $obj(m) = obj(m_1) + obj(m_2)$; et $bar(m) = bar(m_1) + bar(m_2) + 1$ (le +1 correspond à la barre reliant m_1 et m_2). En appliquant l'hypothèse d'induction à $obj(m_1)$ et $obj(m_2)$ on obtient :

$$\begin{aligned} obj(m) &= obj(m_1) + obj(m_2) \\ &= bar(m_1) + 1 + bar(m_2) + 1 \\ &= bar(m) + 1 \end{aligned}$$

D'où l'hérédité.

- Conclusion : on a prouvé par induction que pour tout mobile de Calder m, le nombre $obj(m)$ d'objets est supérieur de 1 au nombre $bar(m)$ de barres.

Remarque. Notez que ce n'est pas une récurrence : on n'a pas travaillé sur des entiers ! L'hérédité ne consistait pas à prouver « $H_{n-1} \Rightarrow H_n$ ». En fait, la notion de « passer d'un entier au suivant » ne marche pas sur les mobiles : un mobile a deux prédecesseurs immédiats et non un seul.

1.2.1 Formalisation (hors-programme)

Savoir ce qu'est une preuve par induction n'est pas simple. Il existe en fait une façon ensembliste de formaliser l'induction. Nous allons la détailler sur un exemple ; c'est assez abstrait.

Exemple. Prouvons que pour tout $n \in \mathbb{N}$: $\sum_{k=0}^{n} k = \frac{n(n+1)}{2}$

- **V0 : version basique**
  - Initialisation : Montrons que $P(0)$ est vraie. Si $n = 0$, on a $\sum_{k=0}^{n} k = 0$ et $\frac{n(n+1)}{2} = 0$ .Donc $I(0)$ est vraie.
  - Hérédité : Supposons la propriété vraie pour un rang $n \in \mathbb{N}$, et montrons la vraie au rang $n + 1$. On a :

$$\begin{aligned} \sum_{k=0}^{n+1} k &= (n+1) + \sum_{k=0}^{n} k \\ &= n + 1 + \frac{n(n+1)}{2} \qquad \text{d'après } I(n) \\ &= \frac{n(n+1) + 2(n+1)}{2} = \frac{(n+1)(n+2)}{2} \end{aligned}$$

    D'où l'hérédité.
  - Conclusion : On a donc montré par récurrence que pour tout $n \in \mathbb{N}$, $I(n)$ est vraie.

- **V1 : comme V0, mais par récurrence forte**
  Comme Và, mais l'on remplace la ligne d'introduction de l'hérédité par : « Soit $n \in \mathbb{N}^*$ tel que pour tout $m < n$, $I(m)$ soit vrai ; et montrons la vraie au rang $n + 1$. »

- **V2 : on généralise et fusionne les notions d'initialisation et d'hérédité en une seule notion**
  Montrons par récurrence sur $n \in \mathbb{N}$ que $I(n)$: « $\sum_{k=0}^{n} k = \frac{n(n+1)}{2}$ » est vraie. Pour cela, montrons que pour tout $n \in \mathbb{N}$ la propriété suivante est vraie :

$$h_n \ : \ [\forall m < n, I(m)] \implies I(n)$$

  Soit $n \in \mathbb{N}$. On procède par disjonction de cas.
  - Si n est minimal dans $(\mathbb{N}, \leq)$, alors $n = 0$. Le terme gauche de l'implication est donc trivialement vrai [3]. Il faut alors montrer $I(0)$. *Et on refait comme l'initialisation de V0/V1.*

3. Un « Pour tout » qui quantifie sur l'ensemble vide est automatiquement vrai. De même, un « Il Existe » qui quantifie sur l'ensemble vide est automatiquement Faux.

— Sinon, $n \neq 0$ (car 0 est le seul élément minimal de $(\mathbb{N}, \leq)$). Pour montrer l'implication, on suppose que le terme gauche est vrai : $\forall m < n, I(m)$. Il faut alors montrer $I(n)$. *Et on refait comme l'hérédité de V1.*

Conclusion : Par principe de récurrence, on en déduit que pour tout $n \in \mathbb{N}$ $I(n)$ est vrai.

- **V3 : on se ramène à des ensembles**
  Notons pour $n \in \mathbb{N}$, $I(n)$: « $\sum_{k=0}^{n} k = \frac{n(n+1)}{2}$ ». Notons $P = \{n \in \mathbb{N}\ t.q.\ I(n)\}$. On veut montrer que $P = \mathbb{N}$. Pour cela, montrons que pour tout $n \in \mathbb{N}$ la propriété suivante est vraie :

  $$Hered_n : [\forall m < n, m \in P] \implies n \in P$$

  Et on continue comme V2, jusqu'à la conclusion exclue.
  Conclusion : Par principe de récurrence, on en déduit que $P = \mathbb{N}$.

Mais qu'est-ce donc finalement que le principe de récurrence ? De cette V3, on en déduit [4] l'expression ensembliste suivante du principe de récurrence :

Définition 23 (Principe d'induction).
Soit $(E, \leq)$ ensemble ordonné. On appelle principe d'induction sur E la propriété suivante (pas forcément vraie !!) :

Pour toute partie P de E, on a $(\forall x \in E,\ Hered_x) \implies P = E$
où $Hered_x : [\forall y < x, y \in P] \implies x \in P$

Si cette propriété est vraie, on dit que $(E, \leq)$ vérifie le principe d'induction.

Ce que dit cette propriété, c'est que si l'appartenance à un ensemble s'hérite de ses (éventuels) prédecesseurs stricts, alors cet ensemble est l'espace tout entier. Le « cas de base » de la récurrence correspond alors au cas où il n'y a pas de prédecesseur strict, qui appel généralement à une preuve distincte (comme en V2 ci-dessus).

- Plus généralement, si $\Phi(x)$ est la propriété à prouver vraie pour chacun des x de E, alors prouver utiliser le principe d'induction revient à poser $P = \{x \in E \mid \Phi(x)\}$ et utiliser le principe pour prouver $P = E$.
- Vous noterez que cette façon de faire des récurrences correspond à des récurrences fortes. En fait, on peut prouver que tout résultat prouvable par récurrence forte l'est pas récurrence simple, et réciproquement (quitte à transformer un peu le résultat). Ainsi, récurrence forte et simple (et double, et triple, etc) ont exactement le même pouvoir de preuve.
  En mathématiques, il peut être attendu de vous que vous utilisez la « récurrence la plus simple ». En informatique, je ne vous en voudrai jamais de faire une récurrence forte au lieu d'une simple, surtout lorsque vous faites une induction (récurrence hors de $\mathbb{N}$).
  En résumé : faites des inductions fortes en informatique, c'est plus simple (vous avez accès à *tous* les antécédants).
- Le principe d'induction n'est pas toujours vrai ! En fait, pour bien fonctionner il a besoin qu'il existe des « cas de base » : voilà le lien avec les ordres bien fondés. [5]

Théorème 24.
$(E, \leq)$ vérifie le principe d'induction si et seulement si $\leq$ est un ordre bien fondé sur E.

Démonstration. DM/TD étoilé. Procédez par double implication. Le sens direct est le plus simple puisque vous y avez accès à des récurrences ! □

4. Pas au sens de « on prouve », mais au sens de « On comprend que le principe de récurrence est » .
5. On peut formaliser une notion de récurrence sans cas de base, qui est rigoureuse et correcte : on appelle cela de la co-induction. C'est très largement hors de notre portée.

Remarque. Le terme de « récurrence » est souvent reservé à $\mathbb{N}$ et le terme « induction » couvre les « récurrences hors de $\mathbb{N}$. C'est le choix que j'ai fait dans ce cours car j'ai vu des concours faire ce choix ; mais à titre personnel je ne vois pas l'intérêt de distinguer les deux notions. [6] On parle aussi de récurrence noetherienne pour parler de récurrence sur un ordre bien fondé, et d'ordre nothérien pour l'ordre.

2 Construire de nouveaux ordres

Nous avons vu comment faire des indicutions (récurrences hors de $\mathbb{N}$). Mais comme nous venons de le dire, il faut pour cela disposer de relations d'ordre bien fondées pour « descendre » jusqu'à un cas de base : voyons comment en obtenir.

Dans toute cette section, $(E, \leq_E)$ et $(F, \leq_F)$ sont deux ensembles ordonnés (et les ordres partiels associés sont $<_E$ et $<_F$).

2.0 Transporter des ordres

Dans cette sous-section, on veut « transporter » un ordre via une restriction ou une focntion.

Proposition 25 (Restriction d'un ordre bien fondé).
Si $\leq_E$ est bien fondé, et si $A \subseteq E$, alors la restriction de $\leq_E$ à A est un ordre bien fondé sur A.

Démonstration. Une suite strictement décroissante de $(A, \leq_A)$ est une suite strictement décroissante de $(E, \leq_E)$. □

Exemple. $(\llbracket k; +\infty \llbracket, \leq)$ est bien fondé car il s'agit de la restriction à $\llbracket k; +\infty \llbracket$ de l'ordre usuel bien fondé sur $\mathbb{N}$.

Proposition 26.
Si $\leq_E$ est bien fondé, et si $f : E- > F$ est injective, alors on peut munir $f(E)$ d'un ordre bien fondé : il suffit de « transporter » les comparaisons de E vers $f(E)$.

Démonstration. Pour tout $y \in f(E)$, notons $f^{-1}(y)$ son unique antécédant. On définit $\preccurlyeq$ ordre sur F par $y \preccurlyeq y'$ lorsque $f^{-1}(y) \leq_E f^{-1}(y')$. Autrement dit, on « transporte » $\leq_E$ à l'aide de l'injection.

Avec cette construction, toute suite infinie strictement décroissante dans $f(E)$ correspond à une suite infinie strictement décroissante dans E. D'où le résultat. □

Exemple. $(\llbracket -\infty; k \llbracket, \geq)$ est bien fondé car il s'agit de l'image de l'exemple précédent par la fonction injective $f : x \mapsto x$.

Remarque. En pratique, on ne devrait pas attendre de vous que vous puissiez prouver aussi rigoureusement qu'un ordre est bien fondé. Une justification convainquante de pourquoi il n'existe pas de suite infinie strictement décroissante suffira. Toutefois, les règles de l'on construction que l'on donne ici sont bonnes à avoir en tête puisqu'elles permettent de trouver en un coup d'oeil un ordre bien fondé.

Proposition 27.
Tout ensemble dénombrable (c'est à dire en bijection avec $\mathbb{N}$) peut être muni d'un ordre bien fondé.

Démonstration. La définition de dénombrable donne une bijection donc une injection depuis $(\mathbb{N}, \leq)$. □

6. Je soupçonne, sans preuve, que la co-existence de « récurrence » et « induction » en français provient du fait que « induction » est le terme anglais pour « récurrence », et que les inductions bien fondés ont gagné en usage ce dernier demi-siècle, où les articles de recherche sont écrits... en anglais.

Remarque. Cependant, l'ordre bien fondé correspondant n'est pas toujours pratique. Lorsque vous verrez en maths la tête de la bijection entre $\mathbb{Z}$ et $\mathbb{N}$ ou entre $\mathbb{Q}$ et $\mathbb{N}$, vous comprendrez que cet ordre bien fondé n'est pas agréable...

2.1 Ordre produit

Dans cette sous-section, on veut munir $E \times F$ d'un ordre.

Définition 28 (Ordre produit).
Soient (e_0, f_0) et (e_1, f_1) deux paires de $E \times F$. On pose $(e_0, f_0) \leq_\Pi (e_1, f_1)$ lorque $e_0 \leq_E e_1$ et $f_0 \leq_F f_1$. Cette relation $\leq_\Pi$ est un ordre sur $E \times F$, appelé **ordre produit**.

Démonstration. Pour prouver que $\leq_\Pi$ est un ordre, il faut le montrer réflexif, antisymétrique, transitif. Les trois points s'obtiennent immédiatement via le fait que $\leq_E$ et $\leq_F$ ont les bonnes propriétés. □

Exemple. $\mathbb{C} = \mathbb{R} \times i\mathbb{R}$, on peut donc munir $\mathbb{C}$ de l'ordre produit (cf partie précédente).

L'ordre produit est pratique puisqu'il permet de faire des inductions !

Proposition 29 (Un produit d'ordres bien fondés est bien fondé).
Si $(E, \leq_E)$ et $(F, \leq_F)$ sont bien fondés, alors $(E \times F, \leq_\Pi)$ est bien fondé.
Autrement dit, le produit d'ordres bien fondés est bien fondé.

Démonstration. Il est ici plus confortable de revenir à la définition d'ordre bien fondé (hélas)[7]. Soit $X \subseteq E \times F$ non-vide.

Notons A l'ensemble des premières coordonnées de X : $A = \{e \in E \mid \exists f \in F, (e, f) \in X\}$. Alors A admet un élément minimal puisque c'est une partie non-vide de E et que $\leq_E$ est bien fondé. Nommons le a.

Notons $B(a)$ l'ensemble des secondes coordonnées pouvant être associées à A dans X : $B = \{f \in F \mid (a, f) \in X\}$. De même, $B(a)$ admet un élément minimal b.

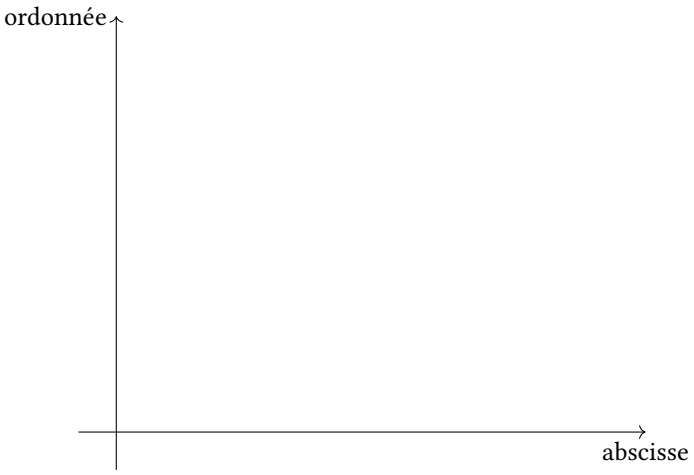

FIGURE 8.10 – Preuve avec l'ordre produit Abscisses × Ordonnées.

Montrons que (a, b) est minimal dans X. Soit (a', b') un prédecesseur de (a, b) par l'ordre produit, et montrons qu'il s'agit en fait de (a, b).

7. Si vous trouvez une version qui utilise la caractérisation et n'est pas un chaos dénotationnel, je suis fort curieux.

$(a',b') \leq_\Pi (a,b)$ implique que $a' \leq_E a$, $b' \leq_F b$ et $(a,b \neq (a',b')$. Par minimalité de a, comme $a' \in A$, $a' \leq_E a$ implique $a = a'$. Mais alors $b' \in B(a)$ et le même raisonnement donne $b = b'$. Donc $(a,b) = (a',b')$.

Donc X admet bien un élément minimal, d'où le résultat. □

Exemple.

- $(\mathbb{N}^2, \leq_\Pi)$ est bien fondé. En particulier, une paire d'entier (x,y) où « parfois x décroit (et y reste constant), parfois y décroit (et x reste constant) » est un variant de boucle !
- Pour tout `len` $\in \mathbb{N}$ fixé, $([\![0;\texttt{len}]\!], \leq)$ et $([\![0;\texttt{len}]\!], \geq)$ sont bien fondés donc l'ensemble produit est bien fondé. En particulier, cela donne une autre façon de prouver la terminaison de la dichotomie !

dicho.c

```c
/** Renvoie true ssi x est présent dans une des len cases de arr
  * Pré-conditions : arr doit être trié par ordre croissant.
  */
bool mem_opt(int x, int arr[], int len) {
  int lo = 0;
  int hi = len-1;

  // Invariant : si x est présent, il est dans arr[lo:hi[
  while (lo < hi) {
    int mid = (lo+hi)/2;
    if (arr[mid] == x)     { return true;}
    else if (arr[mid] < x) { hi = mid; }
    else                   { lo = mid+1; }
  }

  return false;
}
```

Pour prouver la terminaison, on peut prouver que (lo, hi) est un variant ! C'est une preuve plus simple que celle que noius avions fait au S1 (où on a montré que $hi - lo$ est un variant, ce qui était un peu plus dur *mais* avait l'avantage de bien expliquer l'algorithme).

Proposition 30 (Non totalité de l'ordre produit).
Il suffit que $|E| > 1$ et que $|F| > 1$ pour que l'ordre produit $\leq_\Pi$ ne soit pas total.

Démonstration. Supposons que $|E| > 1$ et que $|F| > 1$. Remarquons tout d'abord que s'il existe e_0 et e_1 non comparables dans E, alors pour tout $f \in F$ les paires (e_0, f) et (e_1, f) ne sont pas comparables par $\leq_\Pi$ dans $E \times F$ et donc $\leq_\Pi$ n'est pas total. De même dans F.

Supposons désormais E et F totalement ordonnés, et soient $e_0 <_E e_1$ deux éléments distincts de E et $f <_F f_1$ deux éléments distincts de F. Alors (e_0, f_1) et (e_1, f_0) ne sont pas comparables par $\leq_\Pi$. En effet :

- $e_0 <_E e_1$ implique $(e_1, f_0) \not\leq_\Pi (e_0, f_1)$
- $f_0 <_F f_1$ implique $(e_0, f_1) \not\leq_\Pi (e_1, f_0)$.

□

En d'autres termes, l'ordre produit n'est presque jamais total. Si un ordre total est requis, il ne faut *pas* prendre un ordre partiel.

Exemple. Pour trier un tableau de points[8] de $\mathbb{Z}^2$, il faut un ordre total : l'ordre produit ne peut pas être utilisé. : l'ordre produit n'est pas suffisant pour cela.

8. Cf exercice sur les milieux dans un nuage de points, ou sur les 2 plus proches points dans un nuage.

Définition 31 (Ordre produit sur grands produits).
Soient $E_1, ..., E_n$ n ensembles munis d'ordres $\leq_1, ..., \leq_n$. On définit l'ordre produit $\leq_\Pi$ sur $E_1 \times ... \times E_n$ comme $(e_1, ..., e_n) \leq_\Pi (e'_1, ..., e'_n)$ si et seulement si pour tout $i \in [\![1; n]\!]$, $e_i \leq_0 e'_i$

Démonstration. Comme pour l'ordre produit sur $E \times F$. □

Remarque. C'est en fait l'ordre produit obtenu par récurrence en remarquant que $E_1 \times ... \times E_n = (E_1 \times ... \times E_{n-1} \times) E_n$

2.2 Ordre lexicographique

Dans cette sous-section, on veut munir $E \times F$ d'un ordre qui palie à la non-totalié de l'ordre produit : cers era l'ordre lexicographique. Le nom provient de son origine : c'est l'ordre utilisé pour comparer des mots entre eux. On compare la première lettre, puis la seconde si les premières sont égales, etc.

2.2.0 Sur des éléments de même longueur

Définition 32 (Ordre lexicographique).
Soient (e_0, f_0) et (e_1, f_1) paires de $E \times F$.
On définit la relation d'ordre $\leq_{lex}$ sur $E \times F$ en posant $(e0, f0) \leq_{lex} (e1, f1)$ lorsque $e_0 <_E e_1$ ou que $e_0 = e_1$ et $f_0 \leq_F f_1$.
On appelle $\leq_{lex}$ **ordre lexicographique**, ou produit lexicgraphique de $\leq_E$ et $\leq_F$.

Autrement dit, on compare la première coordonnée pour décider qui est le plus petit. Si elle fait un ex-aequo, on compare la seconde.

Démonstration. Il faut prouver que $\leq_{lex}$ est un ordre. Soient (e_0, f_0), (e_1, f_1) et (e_2, f_2) paires de $E \times F$. On a bien :

- Réflexivité : $e_0 \leq_E e_0$ donc $(e_0, f_0) \leq_{lex} (e_0, f_0)$.
- Anti-symétrie : supposons $(e_0, f_0) \leq_{lex} (e_1, f_1)$ et $(e_1, f_1) \leq_{lex} (e_0, f_0)$.

  Distinguons deux cas. Si $e_0 = e_1$, alors les deux inégalités de $\leq_{lex}$ donnent respectivement $f_0 \leq_F f_1$ et $f_1 \leq_F f_0$. Donc $f_0 = f_1$, donc $(e0, f_0) = (e1, f_1)$.

  Sinon, les deux inégalités donnent $e_0 <_E e_1$ et $e_1 <_E e_0$. Donc $e_0 = e_1$, contradiction avec le « Sinon ».

  D'où l'antisymétrie.
- Un peu laborieux à cause des disjonctions de cas sur le comparaisons des premières coordonnées. Rien de très profond.

□

Exemple.

- C'est le tri des mots du dictionnaires (ici les deux mots ont deux lettres). Par exemple, ba $\leq$ bb et ba $\leq$ ab.
- C'est l'ordre utilisé en DS pour trier un nuage de points !

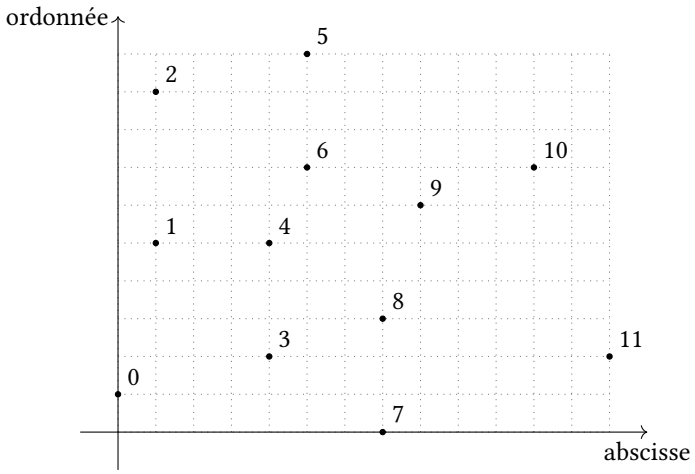

FIGURE 8.11 – Deux suites croissantes dans le plan muni de l'ordre lexicographique « Abscisse × Ordonnée ».

Proposition 33 (Un produit lexicographique d'ordres bien fondés est bien fondé).
Si $(E, \leq_E)$ et $(F, \leq_F)$ sont bien fondés, alors $(E \times F, \leq_{lex})$ est bien fondé.
Autrement dit, le produit lexicographiques d'ordres bien fondés est bien fondé.

Démonstration. On procède de manière similaire à la preuve précédente. Soit $X \subseteq E \times F$ non-vide, montrons que X admet un élément minimal.

On construit A, a, $B(a)$ et b comme dans la preuve précédente.

Montrons que (a, b) est minimal dans X. Soit (a', b') un prédecesseur de (a, b) par l'ordre lexicographique, et montrons qu'il s'agit en fait de (a, b).

Tout d'abord, par minimalité de a, il est impossible que $a' <_E a$. Donc $a = a'$, donc $b' \leq_F b$. Mais par minimalité de b, on a donc $b' = b$.

D'où le résultat. □

Notons que l'exemple d'éléments non comparables dans $\mathbb{C}$ par l'ordre produit était $1 + 0i$ et $0 + i$. Ces deux éléments sont tout à fait comparables par l'ordre lexicographique : $i \leq 1$ (comparaison de la première coordonnée).

Proposition 34.
Si $\leq_E$ et $\leq_F$ sont totaux, alors $\leq_{lex}$ est total.

Démonstration. Supposons $\leq_E$ et $\leq_F$ totaux. Soient (e_0, f_0) et (e_1, f_1) paires de $E \times F$.

Par hypothèse, e_0 et f_0 sont comparables puisque $\leq_E$ est total. On distingue deux cas :

- Si $e_0 = e_1$, alors soit $f_0 \leq_F f_1$ et donc $(e_0, f_0) \leq_{lex} (e_1, f_1)$, soit (par totalité de $\leq_F$) $f_1 \leq_F f_0$ et donc $(e_1, f_1) \leq_{lex} (e_0, f_0)$. Ainsi, (e_0, f_0) et (e_1, f_1) sont comparables.
- Sinon, on a soit $e_0 <_E e_1$ et donc $(e_0, f_0) \leq_{lex} (e_1, f_1)$, soit $e_1 \leq_E e_0$ et donc $(e_1, f_1) \leq_{lex} (e_0, f_0)$. Ainsi, (e_0, f_0) et (e_1, f_1) sont comparables.

□

Ainsi, l'ordre lexicographique permet d'avoir un ordre total et bien fondé !

Exemple. La fonction d'Ackermann [9] [10] [11] est définie par :

9. En réalité de Rozsa Péter, celle d'Ackermann avait trois variables.
10. C'est une fonction « monstrueuse » faite pour croitre très très vite et casser certains espoirs en informatique fondamentale. Ne lui cherchez pas de sens profond.
11. Vous la croiserez peut-être en MPI, où l'on peut mentionner sa réciproque qui croit très très lentement (le logarithme est à supraluminique à coté). Mais je doute que vous fassiez plus qu'en entendre parler.

$$A : (n,x) \in \mathbb{N} \times \mathbb{N}^* \mapsto \begin{cases} x+1 \text{ si n} = 0 \\ A(n-1,1) \text{ si } x = 0 \\ A(n-1, A(n,x-1)) \text{sinon} \end{cases}$$

Prouver par induction dans $(\mathbb{N} \times \mathbb{N}^*, \leq_{lex})$ que pour tous n et x on a $A(n,x) > x$.

- Initialisation : si $n = 0$, $A(n,x) = x + 1 > x$.
- Hérédité : Soit (n,x) avec $n \neq 0$ tel que la propriété soit vraie pour tout $(n',x') <_{lex} (n,x)$, montrons-la vraie pour (n,x). Distinguons deux cas :
  - si $x = 0$, alors $A(n,0) = A(n-1,1) > 1 > 0$.
  - sinon :

$$\begin{aligned} A(n,x) &= A(n-1, A(n,x-1)) \\ &> A(n,x-1) \\ &\geq A(n,x-1) + 1 \\ &> x - 1 + 1 \\ &> x \end{aligned}$$

  Ainsi, dans les deux cas la propriété est héréditaire.
- Conclusion : Pour tout (n,x), $A(n,x) > x$.

Définition 35 (Ordre lexicographique sur grands produits).
Soient $E_1, ..., E_n$ n ensembles munis d'ordres $\leq_1, ..., \leq_n$. On définit l'ordre lexicographique $\leq_{lex}$ sur $E_1 \times ... \times E_n$ comme $(e_1, ..., e_n) \leq_{lex} (e'_1, ..., e'_n)$ si et seulement si il existe $i \leq n$ tel que les i premières coordonnées soient égales et que la $i+1$ème (si elle existe) vérifie $e_{i+1} <_{i+1} e'_{i+1}$.
Formellement, $e = (e_1, ..., e_n) \leq_{lex} (e'_1, ..., e'_n)e'$ si et seulement si $e = e'$ ou il existe $i \in [\![1; n[\![$ tel que pour tout $1 \leq j \leq i$, $e_j = e'_j$ et $e_{i+1} <_{i+1} e'_{i+1}$.

Démonstration. Comme pour l'ordre lexicographique sur $E \times F$. □

Remarque. C'est en fait l'ordre lexicographique obtenu par récurrence en remarquant que $E_1 \times ... \times E_n = (E_1 \times ... \times E_{n-1} \times) E_n$

2.2.1 Sur des éléments de longueurs distinctes

Quand on compare des mots dans des livres, on veut aller plus loin : on veut comparer des mots de longueurs différentes. En pratique, on dit que « si l'on a fini de parcourir un mot et pas l'autre, le plus court est plus petit » ; donc *baba* est plus petit que *babaaurhum*.

Cela se code très bien : voici par exemple un code OCaml qui compare lexicographiquement deux listes (qui peuvent être de longueur différents) :

lex.ml

```
(** Renvoie -1 si e < e', 0 si = et 1 sinon *)
let rec cmp_lex e e' =
  match (e , e') with
  | [] , [] ->  0
  | [] , _  -> -1
  | _  , [] ->  1
  | t::q , t'::q' ->
      if t < t' then -1
      else if t > t' then 1
      else cmp_lex q q'
```

Cependant, autoriser les uplets à être de taille différente et aussi grande que l'on veut brise la bonne fondaison :

Proposition 36.
Un ordre lexicographique sur un ensembles d'uplets de longueur variable et non bornée peut ne pas être bien fondé.

Démonstration. Posons $\Sigma = \{a, b\}$ ordonné par $a < b$. et Σ^+ l'ensemble des n-uplets sur Σ pour tout $n \in \mathbb{N}^*$:

$$\Sigma^+ = \{(a); (b); (a,a); (a,b); (b,a); (b,b); (a,a,a); (a,a,b); ...\}$$

Pour simplifier, je note ab au lieu de (a, b).
La suite suivante est infinie strictement décroissante :

$$b <_{lex} ab <_{lex} aab <_{lex} <_{lex} aaab <_{lex} ...$$

□

2.3 Clotures

2.3.0 Définitions

Dans cette sous-section, on se donne E est un ensemble et $\mathscr{R}$ est une relation binaire homogène sur E. On se demande si on peut déduire un ordre de $\mathscr{R}$.

Dans tout cette sous-section, lorsque l'on compare des relations (« $\mathscr{R}$ est plus petite/grande que $\mathscr{T}$ ») la comparaison utilisée est l'inclusion.

Définition 37 (sur-relation).
On appelle **sur-relation** de $\mathscr{R}$ une relation $\mathscr{T}$ sur E telle que $\mathscr{R} \subseteq \mathscr{T}$.
Réciproquement, on dit dans ce cas que $\mathscr{R}$ est une **sous-relation** de $\mathscr{T}$.

Remarque. Nous avions implicitement utilisé cette notion lorsque l'on a donné le lien entre ordre large et strict.

Définition 38 (Cloture).
La **cloture réflexive** de $\mathscr{R}$, notée $\mathscr{R}^0$, est la plus petite sur-relation réflexive de $\mathscr{R}$.
On définit de même la **cloture symétrique** $\mathscr{R}^S$ et la **cloture transitive** $\mathscr{R}^+$ comme étant les plus petites sur-relations de $\mathscr{R}$ symétrique et transitive respectivement.
On note $\mathscr{R}^*$ la **cloture réflexive transitive**, i.e. la plus petite sur-relation de $\mathscr{R}^0$ qui est à la fois réflexive et transitive.

Démonstration. Il faudrait démontrer que ce snotions eixstent bel et bien, c'est à dire que la plus petite sur-relation reflexive/symétrique/transitive existe.

Il existe des sur-relations réflexive, symétrique ou transitive car la relation plein (qui met *tous* les éléments en relation) est réflexive, symétrique et transitive. L'existence de la plus petite est pour l'instant admis. □

Remarque.

- On appelle aussi une cloture une *fermeture* (réflexive, transitive, symétrique, etc).
- La cloture antisymétrique n'est pas définissable. En effet, si une relation $\mathscr{R}$ n'est pas antisymétrique, elle contient un couple (x, y) tel que $x\mathscr{R}y$ et $y\mathscr{R}x$, donc toutes ses sur-relations contiennent ce couple (x, y) et aucune d'entre elles ne peut être antisymétrique. Nous verrons plus tard une condition plus générale sur l'antisymétrie.

FIGURE 8.12 – Graphe de la congruence modulo p dans $\mathbb{Z}$

- La cloture réflexive transitive de $\mathscr{R}$ est par définition la plus petite relation réflexive transitive contenant $\mathscr{R}$, mais c'est aussi la cloture transitive de la cloture symétrique de $\mathscr{R}$, ou encore la cloture symétrique de la cloture transitive de $\mathscr{R}$. On peut de même ajouter une cloture réflexive dans n'importe quel ordre. *(admis)*
- En termes de schéma/graphe d'une relation, la cloture réflexive signifie que l'on peut « faire du sur place », symétrique que « si on peut aller de a à b on peut aller de b à a » et transitive que « on peut suivre une suite de flèches ».

Exemple.

- La cloture transtive réflexive de $\{(n, n+1), n \in \mathbb{N}\}$ est l'ordre usuel sur $\mathbb{N}$.
- La cloture réflexive symétrique transitive de la relation vide est l'égalité.

Proposition 39.
La cloture réflexive symétrique transitive de $\mathscr{R}$ est la plus petite relation d'équivalence contenant $\mathscr{R}$.

Démonstration. On admet pour l'instant l'existance de cette plus petite sur-relation d'équivalence.

Soit $\mathscr{R}'$ une relation d'équivalence contenant $\mathscr{R}$. Alors $\mathscr{R}'$ est réflexive symétrique transitive et contient $\mathscr{R}$, donc elle est contenue dans la cloture réflexive symétrique transitive de $\mathscr{R}$ par définition.

Autrement dit, la cloture est minimale parmi les sur-relations d'équivalence. Par unicité d'un plus petit élément, on conclut. □

Proposition 40.
On peut expliciter les relations $\mathscr{R}^0$, $\mathscr{R}^S$ et $\mathscr{R}^*$:

- $\mathscr{R}^0 = \mathscr{R} \cup \{(x, x)\ s.t.\ x \in E\}$
- $\mathscr{R}^S = \mathscr{R} \cup \{(y, x)\ s.t.\ x\mathscr{R}y\}$
- Pour $i \in \mathbb{N}^*$, posons $\mathscr{R}^i = \mathscr{R}^{i-1}\{(x, y)\ t.q.\ \exists c_0, \dots, c_i$ avec $c_0 = x,\ c_i = y$ et $\forall 0 \leqslant k < i, c_k\mathscr{R}c_{k+1}\}$. Alors :
  - $\mathscr{R}^+ = \cup_{i\in\mathbb{N}^*} R^i$
  - $\mathscr{R}^* = \mathscr{R}^0 \cup \mathscr{R}^+$

En particulier, ces constructions prouvent que ces clotures existent.

Démonstration.

- La relation $\mathscr{R}^0$ donnée est une relation réflexive, et elle inclut bien $\mathscr{R}$. De plus, toute sur-relation réflexive de $\mathscr{R}$ contient au moins $\mathscr{R}$ et les paires identiques, c'est à dire au moins $\mathscr{R}^0$. D'où la minimalité.
- Même raisonnement.
- On remarque que $\mathscr{R} = \mathscr{R}^1$. Ainsi, la relation $\mathscr{R}^+$ donnée inclut bien $\mathscr{R}$. Montrons maintenant qu'elle est transitive :
  Si $x\mathscr{R}^+y$ et $y\mathscr{R}^+z$, alors $\exists i, x\mathscr{R}^i y$ et $\exists j, y\mathscr{R}^j z$.

C'est à dire :
$\exists c_0, \dots, c_i$ avec $c_0 = x,\ c_i = y$ et $\forall 0 \leqslant k < i, c_k \mathscr{R} c_{k+1}$
et $\exists c'_0, \dots, c'_j$ avec $c'_0 = y,\ c'_j = z$ et $\forall 0 \leqslant k < j, c'_k \mathscr{R} c'_{k+1}$.
On en déduit que $x \mathscr{R}^{i+j} z$ par la séquence $x = c_0, \dots, c_i = y = c'_0, \dots, c'_j = z$ de longueur $i + j$. Donc $x \mathscr{R}^+ z$, d'où la transitivité.
Pour la minimalité, si $\mathscr{T}$ est une sur-relation transitive de $\mathscr{R}$, on montre par récurrence sur i que tous les $\mathscr{R}^i$ sont inclus dans $\mathscr{T}$.

- Similaire aux deux premiers.

□

Remarque.

- La définition de $\mathscr{R}^+$ se comprend mieux en français : $\mathscr{R}^i$ est l'ensemble des paires peuvent être « reliées » par un chemin de i applications de$\mathscr{R}$. Par exemple, si $(x, y) \in \mathscr{R}^3$, on a un chemin de la forme $x \mathscr{R} u \mathscr{R} v \mathscr{R} y$. Notez que dans cet exemple x et y ne sont pas forcément en relation par $\mathscr{R}$: on affirme juste qu'il y a un chemin de longueur au plus 3 (et donc qu'ils sont en relation dans la cloture transitive).

Figure 8.13 – $\mathscr{P}(\llbracket 0; 4 \rrbracket)$ et les chemins de longueur 3

- Ces constructions s'expriment bien en français : pour rendre une relation réflexive on ajoute les paires identiques, pour la rendre symétrique on symétrise, et pour la rendre transitive on « suit les chemins » (c'est à dire qu'on ajoute la transitivité).

2.3.1 Engendrer un ordre

La question est maintenant : comment construire des relations d'ordre avec cela ?

Définition 41 (Acyclicité d'une relation).
Soit $l \in \mathcal{N}$. Un cycle de longueur l pour $\mathscr{R}$ est une suite finie $(u_n)_{n\in[\![0;\ell[\![}$ d'éléments deux à deux distincts telle que pour tout n, $u_n \mathscr{R} u_{n+1}$ et telle que $u_{\ell-1} \mathscr{R} u_0$.
On dit que $\mathscr{R}$ est sans cycle s'il n'y a pas de cycle de longueur au moins 2 pour $\mathscr{R}$.

Cette notion se comprend très bien sur un schéma :

Figure 8.14 – Un cycle

Exemple. Si on considère $(\mathbb{Z}/3\mathbb{Z}, \leq)$ et que l'on définit une relation d'ordre sur cet ensemble par :
$0 \leqslant 1$, et $1 \leqslant 2$ et $2 \leqslant 0$
alors 0, 1, 2 est un cycle de longueur 3.

Exemple. Notons | la relation de divisibilité, c'est à dire que $a|b$ signifie $a|b$. Si l'on considère que tout entier divise 0, alors $(\mathbb{N}, |)$ contient de nombreux cycles. Sinon, la relation est acyclique car $a|b \implies a \leq a$.

Figure 8.15 – Cycles de la division de 0

Remarque. On pourrait aussi définir un cycle avec la contrainte alternative : pour tout n, $u_{n+1} \mathscr{R} u_n$. C'est une question de conventions. L'important est que ce soit toujours dans le même sens.

Proposition 42 (Cloture réflexive transitive d'une relaiton acyclique).
Si $\mathscr{R}$ est sans cycle, alors sa cloture réflexive transitive $\mathscr{R}^*$ est la plus petite relation d'ordre contenant $\mathscr{R}$

Démonstration. Il y a deux points à prouver : que c'est une relation d'ordre, et que c'est la plus petite. Commençons par le second.

Toute relation d'ordre contenant R est en particulier réflexive et transitive. Par définition de la cloture réflexive transitive, $\mathscr{R}^*$ est bien la plus petite.

Montrons maintenant que c'est bien une relation d'ordre. Comme elle est par définition réflexive et transitive, il reste à montrer qu'elle est antisymétrique.

Soient x et y tels que $x\mathscr{R}^*y$ et $y\mathscr{R}^*x$. Par construction de $\mathscr{R}^*$, notons i et j les premiers rangs tels que $x\mathscr{R}^iy$ et $y\mathscr{R}^jx$. Si i ou j sont non-nuls, alors cela fournit un cycle de $\mathscr{R}$: absurde. Donc $i = j = 0$, i.e. $x = y$. La relation est donc bien antisymétrique. □

Définition 43 (Ordre engendré).
Soit $\mathscr{R}$ une relation sans cycle sur E. On appelle **ordre engendré par** $\mathscr{R}$ la cloture réflexive transitive de $\mathscr{R}$.

Exemple. L'ordre engendré par la relation $\{(n, n+1)\ t.q.\ n \in \mathbb{N}\}$ est l'ordre usuel sur $\mathbb{N}$.

Remarque.

- Pour définir un ordre, il suffit parfois de donner les prédecesseurs immédiats de chaque élément (ou ses successeurs immédiats). Nous en verrons les conditions en TD.
- Les deux grosses applications des ordres engendrés sont l'ordre topologique (que nous verrons dans quelques mois) et l'induction structurelle.

3 Induction structurelle

3.0 Construction inductive d'un ensemble : exemple illustratif

On reprend l'exemple des mobiles de Calder. On suppose pour simplifier que les objets suspendus sont des entiers :

calder.ml

```
4 type objet = int
5 type mobile = Obj of objet | Bar of mobile * mobile
```

Ainsi, le mobile ci-dessous correspond à la déclaration OCaml qui l'accompagne :

Figure 8.16 – Un mobile et son code OCaml

On veut savoir quels sont précisément les éléments de type `mobile`. Notons $\mathcal{O}$ l'ensemble des objets, et on veut définir M l'ensemble des mobiles.

Un mobile ets obtenu par des applications successives des règles de construction. Notons f_M la fonction qui « applique une étape de la construction à un ensemble X de mobiles déjà construits» :

$$f_T : X \mapsto \mathcal{O} \cup \{\texttt{Bar}(m_1, m_2) \text{ où } m_1 \in X, m_2 \in X\}$$

Autrement dit, si l'on dispose de X un ensemble de mobiles, les mobiles que l'on peut obtenir à aprtir de X sont :

- Les objets (sans utiliser les mobiles de X donc).

- Deux mobiles de X reliés par une barre.

On peut alors nommer les étapes successives de construction :

- $M_0 = f_M(\emptyset) = \mathcal{O}$
- Pour $n \in \mathbb{N}$, $M_{n+1} = M_n \cup M_T(M_n)$

Ainsi, M_n sont les mobiles que l'on obtient en *au plus* $n+1$ étapes de constructions.

Définition 44.
L'ensemble M des mobiles est :

$$M = \lim_{n \to +\infty} M_n = \cup_{n=0}^{+\infty} M_n$$

C'est à dire l'ensemble des arbres que l'on obtient en partant des cas de bases et en appliquant un nombre fini de fois le constructeur récursif `Bar`.

Remarque. Cette définition exclut les expressions infinies puisque le constructeur n'est appliqué qu'un nombre fini de fois. En conséquence, les listes infinies ne peuvent pas exister puisqu'une liste est une suite d'application du constructeur `::` à partir du cas de base `[]` .[12]

Il existe une autre façon de voir M, qui n'utilise pas cette construction par récurrence :

Démonstration. On procède en deux temps : on montre que M est plus petit que tout point fixe, puis que c'est un point fixe.

- si X est un point fixe de f_M, alors $f_M(X) = X$. En particulier, $M_0 = \mathcal{O} \subseteq X$. Mais alors $f_M(M_0) \subseteq f(X) = X$. Ainsi, par récurrence sur i, pour tout i, $M_i \subseteq X$. Il s'ensuit que $M \subseteq X$
- Pour tout $n \in \mathbb{N}^*$, $f_M(M_n) = M_{n+1}$. Donc $f_M\left(\cup_{n=0}^{+\infty} M_n\right) = \cup_{n=1}^{+\infty} T_n$. D'où $f_M(M) \subseteq M$. Mais $M_0 = \mathcal{O} \subseteq M_1$. Donc $\cup_{n=0}^{+\infty} M_n \subseteq f_M(M)$, donc $f_M(M) = M$.

□

Le type des mobiles était plutot simple en cela qu'il n'y avait qu'un seul cas de base, et un seul cas récursif. Mais on peut tout à fait définir des cas plus compliqué, comme par exemple :

```
type expr =
  | Int of int
  | Var of string
  | Add of expr * expr
  | Mul of expr * expr
  | Modulo of expr * expr
```

Ce type permet de représenter récusivement des expressions arithmétiques pouvant faire apparaitre des entiers, des variables nommées, des additions, soustractions ou modulos. Par exemple, « ((x+1)(x-1)) mod 3 » correspond à :

12. (Hors-Programme) Toutefois, OCaml supporte partiellement les listes infinies... pour peu qu'elles soient un « cycle », c'est à dire un facteur fini qui se répète à l'infini. On peut par exemple faire `let rec l = 0::1::l` . On retombne assez vite sur la notion de co-induction, déjà évoquée mais de niveau M2.

FIGURE 8.17 – Représentation arborescente d'une expression arithmétique

3.1 Induction structurelle

Soit E un ensemble obtenu par une construction inductive[13]. Par exemple, les mobiles de Calder ou les expressions arithmétiques ou les listes OCaml.

On note $\mathscr{R}_X$ la relation homogène sur X définie par : « x $\mathscr{R}_X$ y lorsque y est construit en appliquant un constructeur à (entre autres) x ».

Exemple.

- Dans les mobiles, m_1 $\mathscr{R}$ `Bar`(m_1, m_2)
- Dans les suites OCaml, `[2;0;3]` $\mathscr{R}$ `[1; 2; 0; 3]`. De même, `[2;0;3]` $\mathscr{R}$ `[5; 2; 0; 3]`.

Proposition 45.
$\mathscr{R}_X$ est sans cycle.

Démonstration. Admis. □

Corollaire 46 (ordre structurel).
On peut étendre $\mathscr{R}_X$ en un ordre, appelé « ordre structurel ».
Intuitivement, cet ordre dit que $x \leq y$ si et seulement si x a été utilisé dans la construction récursive de y.

Démonstration. Il s'agit de la fermeture réflexive tranbsitive d'une relation acyclique. □

Théorème 47.
L'ordre structurel est bien fondé.

Démonstration. Idée la preuve sur l'exemple des mobiles : à chaque mobile, on associe le plus petit n tel que le mobile appartienne à M_n. On peut alors montrer qu'une suite infinie strictement décroissante de mobiles correspond à une suite infinie strictement décroissante de $n \in \mathbb{N}$: impossible. □

Et voilà le travail ! Nous pouvons maintenant faire des inductions *sur les membres d'un type inductif eux-mêmes* !

13. ou récursive, c'est synonyme

Convention 48 (Méthodologie des inductions structurelles).
Pour faire une induction structurelle, on doit :

- **Énoncer clairement la propriété** à prouver et le fait qu'on procède par induction structurelle.
- Traiter comme **cas de base** tous les cas de base du type (les éléments produits par des constructeurs non-récursifs).
- Traiter l'hérédité, qui consiste à **prendre un élément qui n'est pas un cas de base, supposer la propriété vraie pour tous ses sous-éléments, et en déduire la propriété vraie pour lui**.
- **Conclure** par principe d'induction structurelle.

Exemple. Nous avons en fait déjà fait un exemple d'induction structurelle sur les mobiles ! Et nous en ferons *beaucoup* sur les arbres.

Remarque. ⚠ Certains rapports de jury sont dubitatifs sur les rédactions des inductions structurelles. Faites-y très attention. Notamment, définissez précisément qui sont les « sous-éléments ».

3.2 Un exercice final

Voici une fonction récursive. Prouver qu'elle termine et qu'elle est totalement correcte :

lex.ml

```
(** Fusionne les deux listes triées l0 et l1
  * en une seule liste triée.
  *)
let rec merge l0 l1 =
  match (l0 , l1) with
  | [] , [] ->  []
  | [] , _  -> l1
  | _  , [] -> l0
  | t0::q0 , t1::q1 ->
      if t0 <= t1 then t0 :: (merge q0 l1)
      else t1 :: (merge l0 q1)
```