Ensembles infinis, bijections et fonctions récursives

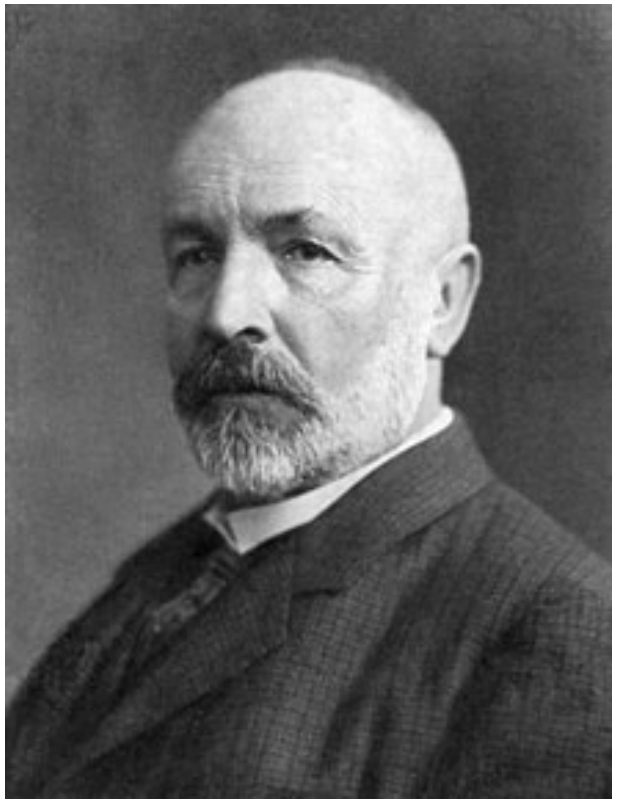
Julien David

S3 348 - julien.david@unicaen.fr

Un peu d'histoire

Georg Cantor, était un mathématicien allemand, né en 1845 et mort en 1918.

- il a crée la théorie des ensembles,
  (cf. cours précédent)
- il a défini les ensembles infinis,
  (cours d'aujourd'hui)
- ses travaux vont montrer l'importance des bijections.
  (cours d'aujourd'hui)

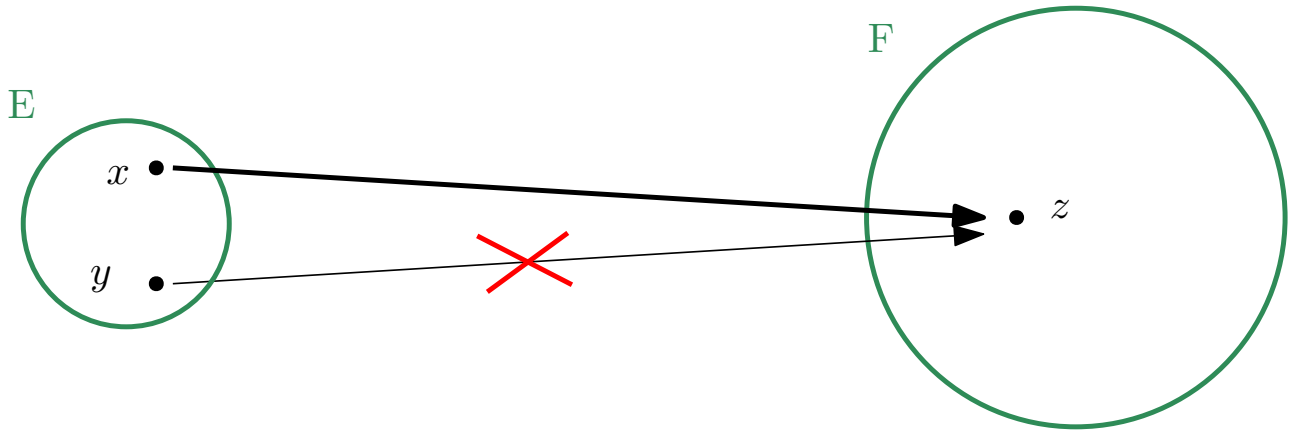


Injections, Surjection, Bijection

Injection

Soient un E et F deux ensembles et $i : E \to F$ une **application** de E vers F. L'application i est une **injection** s'il n'existe pas deux éléments distincts de E qui possèdent la même image par i. Autrement dit,

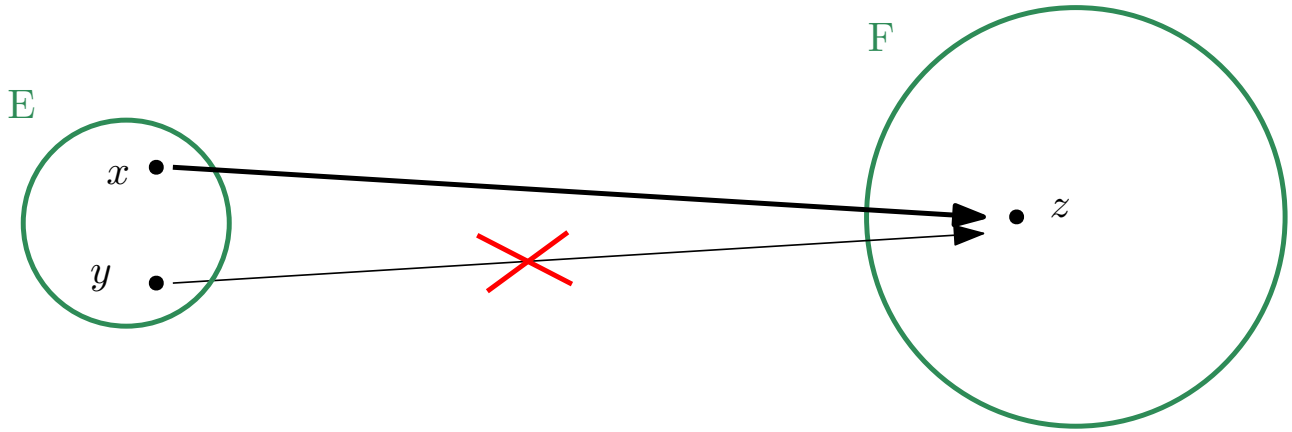
$$\forall (x, y) \in E^2, i(x) = i(y) \implies x = y$$



Injection

Soient un E et F deux ensembles et $i : E \to F$ une **application** de E vers F. L'application i est une **injection** s'il n'existe pas deux éléments distincts de E qui possèdent la même image par i. Autrement dit,

$$\forall (x, y) \in E^2, i(x) = i(y) \implies x = y$$

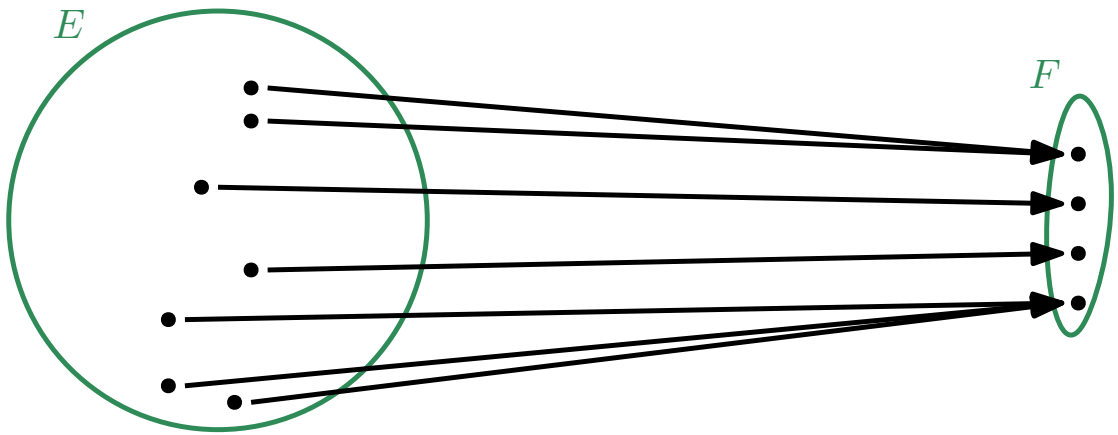


$$|\mathbf{E}| \leq |\mathbf{F}|$$

Surjection

Soient un E et F deux ensembles et $s : E \to F$ une **application** de E vers F. L'application s est une **surjection** si tout élément de F possède **au moins** un antécedent dans E. Autrement dit,

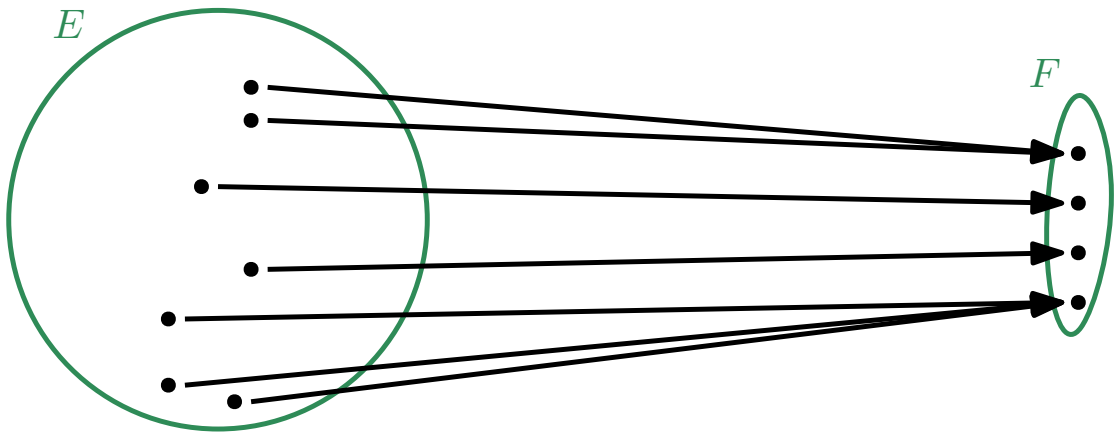
$$\forall y \in F, \exists x \in E, s(x) = y$$



Surjection

Soient un E et F deux ensembles et $s : E \to F$ une **application** de E vers F. L'application s est une **surjection** si tout élément de F possède **au moins** un antécedent dans E. Autrement dit,

$$\forall y \in F, \exists x \in E, s(x) = y$$

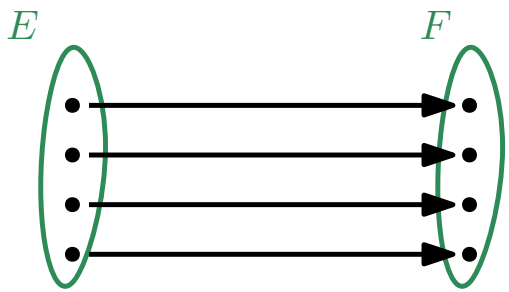


$$\mathbf{|E| \geq |F|}$$

Bijection

Soient un E et F deux ensembles et $b : E \to F$ une **application** de E vers F. L'application b est une **bijection** si elle est à la fois injective et surjective. Autrement dit,

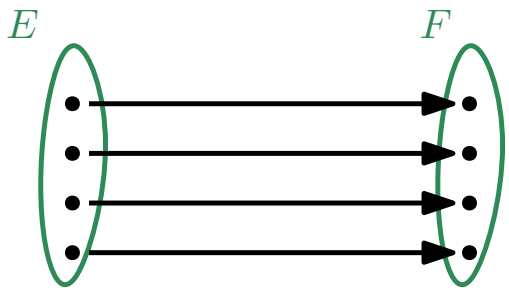
$$\forall y \in F, \exists! x \in E, b(x) = y$$



Bijection

Soient un E et F deux ensembles et $b : E \to F$ une **application** de E vers F. L'application b est une **bijection** si elle est à la fois injective et surjective. Autrement dit,

$$\forall y \in F, \exists! x \in E, b(x) = y$$



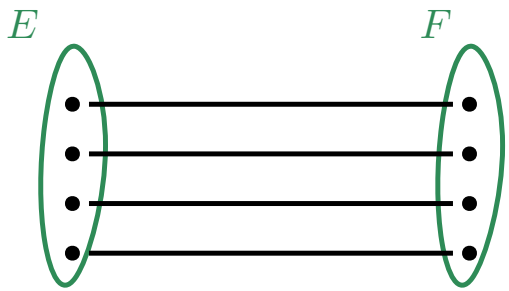
$$|E| = |F|$$

Bijection

Les bijections, également appelée **one to one correspondance** en anglais, permettent d'associer à tout élement de départ un unique élement dans l'ensemble d'arrivée, et inversement.

Pour toute bijection f, il existe donc une fonction **inverse** f^{-1} telle que

$$\forall e \in E, \exists y \in F, f^{-1}(y) = x \iff f(x) = y$$

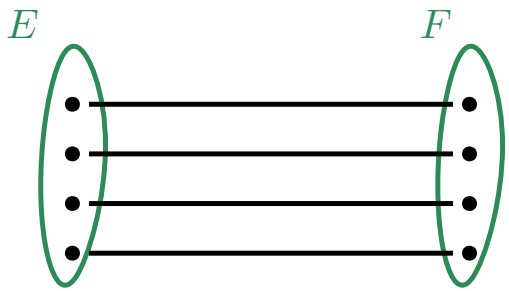


Il existe une bijection entre deux ensembles E et F **si et seulement si** $|E| = |F|$

Bijection

Les bijections, également appelée **one to one correspondance** en anglais, permettent d'associer à tout élement de départ un unique élement dans l'ensemble d'arrivée, et inversement.
Pour toute bijection f, il existe donc une fonction **inverse** f^{-1} telle que

$$\forall e \in E, \exists y \in F, f^{-1}(y) = x \iff f(x) = y$$

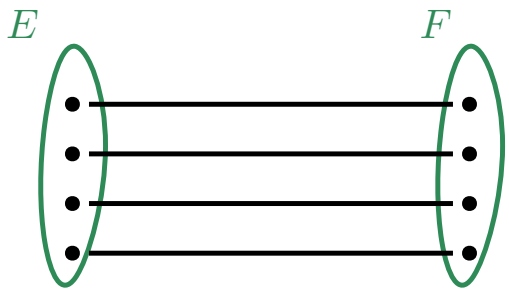


Il existe une bijection entre deux ensembles E et F **si et seulement si** $|E| = |F|$

Bijection

Les bijections, également appelée **one to one correspondance** en anglais, permettent d'associer à tout élement de départ un unique élement dans l'ensemble d'arrivée, et inversement.
Pour toute bijection f, il existe donc une fonction **inverse** f^{-1} telle que

$$\forall e \in E, \exists y \in F, f^{-1}(y) = x \iff f(x) = y$$



Il existe une bijection entre deux ensembles E et F **si et seulement si** $|E| = |F|$

Les bijections chez les informaticiens

Transformer la représentation d'une information

Soit un ensemble fini de lettres, noté Σ, que l'on appelle alphabet.

Exemples

$$\Sigma = \{0, 1\}$$

$$\Sigma = \{a, b, c, d, e, \ldots, z\}$$

$$\Sigma = \{\text{  }, \text{  }, \text{  }, \text{  }\}$$

Transformer la représentation d'une information

On note Σ^* l'ensemble des mots que l'on peut écrire sur l'alphabet Σ.

Exemples

$$0110011 \in \{0,1\}^*$$

$$coucou \in \{a,b,c,d,e,\ldots,z\}^*$$

 $\in \{$  ,  ,  ,  $\}^*$

Transformer la représentation d'une information : Réseaux

On peut transformer une information afin de faciliter sa transmission.

- Codage NRZ, NRZI : $\{0,1\}^* \to \{-,+\}^*$
- Codage Manchester : $\{0,1\}^* \to \{-+,+-\}^*$

Ces transformations sont des bijections.

Transformer la représentation d'une information : Compression

Un algorithme de compression sans perte est une bijection de Σ^* vers un ensemble $C \subset \Sigma^*$ des séquences "compressées".

- il s'agit de transformer un mot de longueur n en un mot de longueur m.
- l'objectif est que m soit inférieur à n
- on veut pouvoir faire la transformation inverse (décompresser)

Transformer la représentation d'une information : Compression

Problème

Prenons l'alphabet le plus simple possible $\Sigma = \{0, 1\}$.

- L'ensemble des mots de longueur n est de cardinal 2^n
- L'ensemble des mots de longueur **inférieur** à n est de cardinal

$$\sum_{i=0}^{n-1} 2^i = 2^n - 1$$

Il est impossible de compresser chaque mot de longueur n en un mot plus petit.

Transformer la représentation d'une information : Compression

Un algorithme de compression sans perte est une bijection de Σ^* vers un ensemble $C \subset \Sigma^*$ des séquences "compressées".

- il s'agit de transformer un mot de longueur n en un mot de longueur m.
- on veut pouvoir faire la transformation inverse (décompresser)
- **dans certains cas on aura $m < n$ et dans d'autres $m \geq n$**

Ensembles infinis

Ensemble infinis

Un ensemble E est dit **infini** si, pour tout entier n, il n'existe pas de bijection entre E et $\{0, 1, \ldots, n-1\}$

Ensemble infinis

Exemple d'ensemble infini :

- $\mathbb{N}$: { ensemble des entiers naturels } $= \{0, 1, 2, \ldots\}$
- $\mathbb{Z}$: { ensemble des entiers relatifs } $= \{\ldots, -3, -2, -1, 0, 1, 2, \ldots\}$
- $\mathbb{R}$: { ensemble des entiers réels }
- $\{0, 1\}^*$: { ensembles des mots binaires }

Bijection sur les ensembles infinis

Soit l'ensemble des entiers pairs

$$2\mathbb{N} = \{n \in \mathbb{N} \mid n \text{ est pair}\} = \{n \in \mathbb{N} \mid \exists m \in \mathbb{N}, n = 2m\}$$

Soit la fonction $f : \mathbb{N} \to 2\mathbb{N}$ telle que

$$\forall m \in \mathbb{N}, f(m) = 2m$$

La fonction f est une bijection [1], donc $|\mathbb{N}| = |2\mathbb{N}|$

1. La fonction inverse $f^{-1}(2m) = m$

Bijection sur les ensembles infinis

Soit l'ensemble des entiers pairs

$$2\mathbb{N} = \{n \in \mathbb{N} \mid n \text{ est pair}\} = \{n \in \mathbb{N} \mid \exists m \in \mathbb{N}, n = 2m\}$$

Soit la fonction $f : \mathbb{N} \to 2\mathbb{N}$ telle que

$$\forall m \in \mathbb{N}, f(m) = 2m$$

La fonction f est une bijection [1], donc $|\mathbb{N}| = |2\mathbb{N}|$

1. La fonction inverse $f^{-1}(2m) = m$

Bijection sur les ensembles infinis

Soit la fonction $f : \mathbb{N} \to \mathbb{Z}$ telle que

$$f(n) = \begin{cases} \frac{n}{2} \text{ si } n \text{ est pair} \\ -\frac{n+1}{2} \text{ si } n \text{ est impair} \end{cases}$$

Les premières valeurs

$n \in \mathbb{N}$:	0	1	2	3	4	5	6	7	8
$f(n) \in \mathbb{Z}$:	0	-1	1	-2	3	-3	4	-4	5

La fonction f est une bijection[2], donc $|\mathbb{N}| = |\mathbb{Z}|$

2. La fonction inverse $f^{-1}(n) = 2n$ si $x \geq 0$ et $f^{-1}(n) = -2n - 1$ si $x < 0$

Bijection sur les ensembles infinis

Soit la fonction $f : \mathbb{N} \to \mathbb{Z}$ telle que

$$f(n) = \begin{cases} \frac{n}{2} \text{ si } n \text{ est pair} \\ -\frac{n+1}{2} \text{ si } n \text{ est impair} \end{cases}$$

Les premières valeurs

$n \in \mathbb{N}$:	0	1	2	3	4	5	6	7	8
$f(n) \in \mathbb{Z}$:	0	-1	1	-2	3	-3	4	-4	5

La fonction f est une bijection [2], donc $|\mathbb{N}| = |\mathbb{Z}|$

2. La fonction inverse $f^{-1}(n) = 2n$ si $x \geq 0$ et $f^{-1}(n) = -2n - 1$ si $x \leqslant 0$

Bijection sur les ensembles infinis

Soit l'ensemble des couples d'entiers :

$$\mathbb{N}^2 = \{(x, y) \in x \in \mathbb{N} \text{ et } y \in \mathbb{N}\}$$

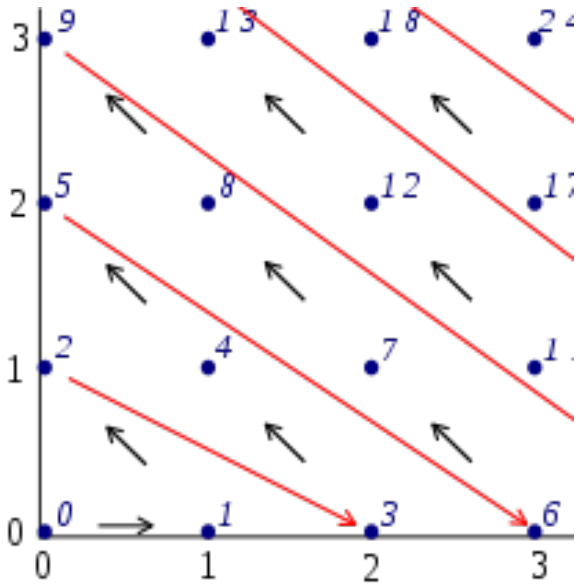
Bijection sur les ensembles infinis

Soit l'ensemble des couples d'entiers :

$$\mathbb{N}^2 = \{(x, y) \in x \in \mathbb{N} \text{ et } y \in \mathbb{N}\}$$

La fonction de couplage de Cantor $f : \mathbb{N}^2 : \mathbb{N}$ définit par

$$f(p, q) = q + \sum_{0 \leq k < p+q} k + 1$$



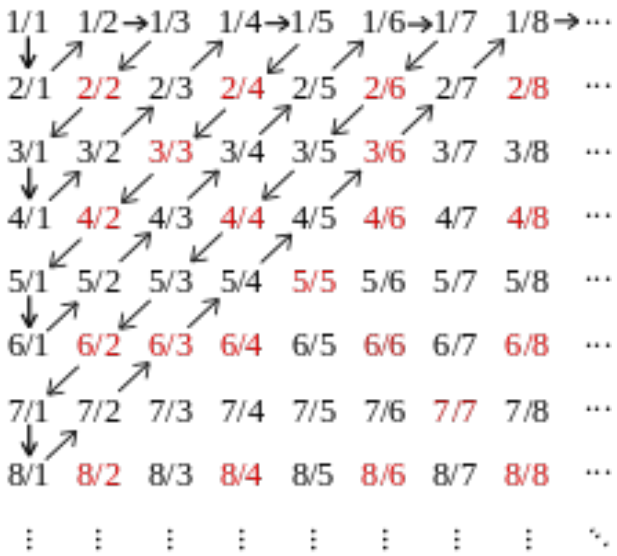
est une bijection.

Bijection sur les ensembles infinis

L'ensemble des nombres rationnels $\mathbb{Q}$ est l'ensemble des nombres qui peut s'exprimer comme le quotient de deux entiers relatifs.

$$\mathbb{Q} = \{\frac{m}{n} \mid (m, n) \in \mathbb{Z} \times (\mathbb{Z} \setminus 0)\}$$

Il existe une bijection entre $\mathbb{Q}$ et $\mathbb{N}$



Bijection sur les ensembles infinis

Question

Tous les ensembles infinis sont-ils en bijection ?

Bijection sur les ensembles infinis

On rappelle que $\mathcal{P}(E)$ est l'ensemble des parties de l'ensemble E.

Soit E un ensemble quelconque (fini ou infini).
Il n'existe pas de bijection entre E et $\mathcal{P}(E)$!

Bijection sur les ensembles infinis

On rappelle que $\mathcal{P}(E)$ est l'ensemble des parties de l'ensemble E.

Soit E un ensemble quelconque (fini ou infini).
Il n'existe pas de bijection entre E et $\mathcal{P}(E)$!

Bijection sur les ensembles infinis

Soit E un ensemble quelconque (fini ou infini).
Supposons qu'il existe une fonction bijective f de E vers $\mathcal{P}(E)$.

- tout élément $x \in E$, doit avoir une image dans $\mathcal{P}(E)$,
- tout élément $y \in \mathcal{F}$, doit avoir exactement un antécédent dans E.

Bijection sur les ensembles infinis

On note D l'ensemble des élements $x \in E$ qui n'appartiennent pas à leur image par f :

$$D = \{x \in E \mid x \notin f(x)\}$$

On a $D \in \mathcal{P}(E)$, or D n'a pas n'antécédent dans E
On le montre par un raisonnment par l'absurde.

Bijection sur les ensembles infinis

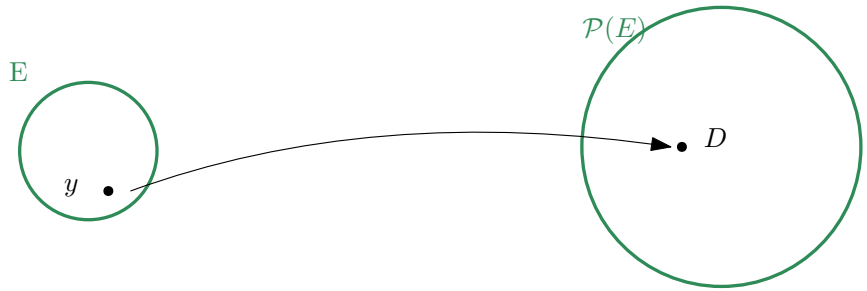
On note D l'ensemble des élements $x \in E$ qui n'appartiennent pas à leur image par f :

$$D = \{x \in E \mid x \notin f(x)\}$$

On a $D \in \mathcal{P}(E)$, or D n'a pas n'antécédent dans E
On le montre par un raisonnment par l'absurde.

Bijection sur les ensembles infinis

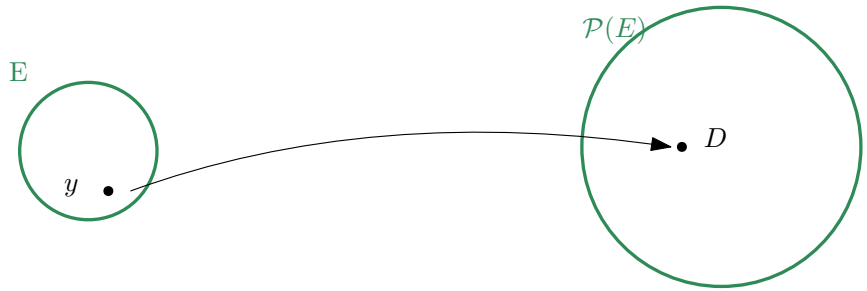
On rappele que $D = \{x \in E \mid x \notin f(x)\}$
Supposons que l'ensemble D ait un antécédent y par f dans E.



Est-ce que y appartient à D ?

Bijection sur les ensembles infinis

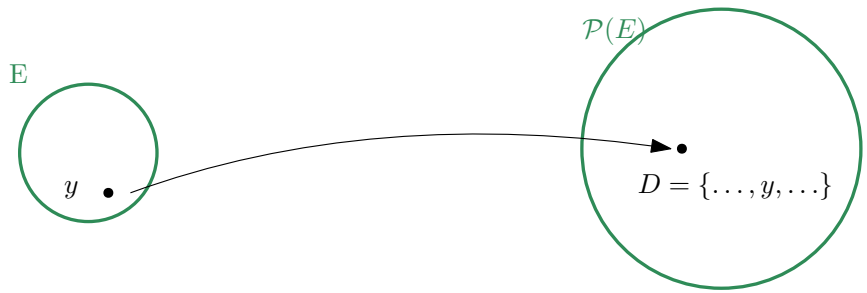
On rappele que $D = \{x \in E \mid x \notin f(x)\}$
Supposons que l'ensemble D ait un antécédent y par f dans E.



Est-ce que y appartient à D ?

Bijection sur les ensembles infinis

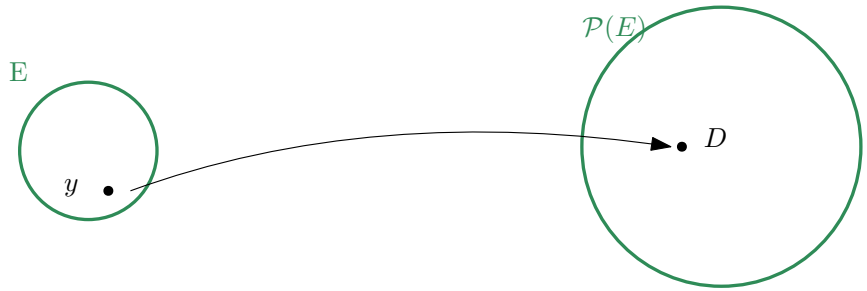
On rappele que $D = \{x \in E \mid x \notin f(x)\}$
Supposons que l'ensemble D ait un antécédent y par f dans E.



Si $y \in D$, alors y n'appartient pas à son image... donc $y \notin D$

Bijection sur les ensembles infinis

On rappele que $D = \{x \in E \mid x \notin f(x)\}$
Supposons que l'ensemble D ait un antécédent y par f dans E.



Si $y \notin D$, alors y appartient à son image... donc $y \in D$

Bijection sur les ensembles infinis

Remarque

Il n'existe pas de bijection entre $\mathbb{N}$ et $\mathcal{P}(\mathbb{N})$. On a :

$$|E| < |P(E)|$$

Il existe donc des ensembles infinis plus grand que d'autres !

Bijection sur les ensembles infinis

Remarque

L'ensemble $\mathcal{P}(\mathbb{N})$ est lui même un ensemble infini, il n'est donc pas en bijection avec $\mathcal{P}(\mathcal{P}(\mathbb{N}))\ldots$ et ainsi de suite.

Il existe une hiérarchie infinie d'ensembles infinis en terme cardinalité

Ensemble infinis dénombrable

Un ensemble E est dit **infini dénombrable** s'il existe une bijection entre $\mathbb{N}$ et E.

Calculabilité (Spoiler M1)

Un ordinateur peut-il répondre à TOUTES les questions que vous lui posez ?

Problème de décision

Un **problème de décision** est un problème dans lequel on a :

- un objet x
- un ensemble L, potentiellement infini.

et l'on souhaite répondre à la question x **appartient-il à** L **?**

Problème de décision - Exemples

- Le nombre n est-il pair ?
- Ce cours est-il intéressant ?
- Manger des hamburgers tous les jours, est-ce une alimentation saine ?

Problème de décision : reformulation

Un **problème de décision** est un problème dans lequel on a :

- un **mot** $x \in \{0,1\}^*$
- un ensemble $L \subseteq \{0,1\}^*$.

et l'on souhaite répondre à la question x **appartient-il à** L**?**

Pour tout ensemble $L \in \mathcal{P}(\{0,1\}^*)$, on peut créer un problème de décision.
L'ensemble des problèmes de décision PB est un surensemble de $\mathcal{P}(\{0,1\}^*)$

Problème de décision : reformulation

Un **problème de décision** est un problème dans lequel on a :

- un **mot** $x \in \{0,1\}^*$
- un ensemble $L \subseteq \{0,1\}^*$.

et l'on souhaite répondre à la question x **appartient-il à** L**?**

Pour tout ensemble $L \in \mathcal{P}(\{0,1\}^*)$, on peut créer un problème de décision.
L'ensemble des problèmes de décision PB est un surensemble de $\mathcal{P}(\{0,1\}^*)$

Problème de décision : reformulation

Un **problème de décision** est un problème dans lequel on a :

- un **mot** $x \in \{0,1\}^*$
- un ensemble $L \subseteq \{0,1\}^*$.

et l'on souhaite répondre à la question x **appartient-il à** L**?**

Pour tout ensemble $L \in \mathcal{P}(\{0,1\}^*)$, on peut créer un problème de décision.
L'ensemble des problèmes de décision PB est un surensemble de $\mathcal{P}(\{0,1\}^*)$

Programme informatique

Tout programme informatique est stocké en binaire sur l'ordinateur.
L'ensemble des programme *PR* est donc un sous-ensemble de $\{0,1\}^*$.

On a donc

$$|PR| \leq \{0,1\}^* < \mathcal{P}(\{0,1\}^*) \leq |PB|$$

Il existe donc plus de problèmes que de programmes ! Il existe donc des questions auxquelles un ordinateur ne peut pas répondre.

Programme informatique

Tout programme informatique est stocké en binaire sur l'ordinateur.
L'ensemble des programme PR est donc un sous-ensemble de $\{0,1\}^*$.

On a donc

$$|PR| \leq \{0,1\}^* < \mathcal{P}(\{0,1\}^*) \leq |PB|$$

Il existe donc plus de problèmes que de programmes ! Il existe donc des questions auxquelles un ordinateur ne peut pas répondre.

Fonctions récursives

Fonction récursive

Une **fonction récursive** est une fonction dont la définition contient un appel à elle-même.

Fibonacci

La fonction de fibonacci est définie par

$$\begin{cases} f_0 = 1 \\ f_1 = 1 \\ f_n = f_{n-1} + f_{n-2} \end{cases}$$

Fonction récursive

Une **fonction récursive** est une fonction dont la définition contient un appel à elle-même.

Fibonacci

```
1 int fibonacci(int n){
2   if(n == 0 || n == 1)
3     return 1;
4   return fibonacci(n-1) + fibonacci(n-2);
5 }
```

Fonction itérative

Une fonction itérative est une fonction qui utilise des boucles **for** ou **while**

Fibonacci

```
1  int fibonacci_iteratif(int n){
2    int prec = 1;
3    int prec_prec = 1;
4    int res = 1;
5    for(int i = 2; i <= n; i++){
6      res = prec + prec_prec;
7      prec_prec = prec;
8      prec = res;
9    }
0    return res;
1  }
```

Langages fonctionnels

Certains langages de programmation, dit **fonctionnels**, n'utilisent pas de boucles, mais uniquement des fonctions récursives.

Lisp (1958)

OCaml (1987)

Haskell (1987)

Scala (2003)

Fonctions récursives vs Fonctions itératives

Il existe une bijection entre les fonctions itératives et les fonctions récursives. Les langages fonctionnels peuvent calculer exactement la même chose que des langages de programmation impérative.