Calculer ou raisonner ?

Pierre Castéran, Université Bordeaux 1 et LaBRI

27 janvier 2010

Programmes et preuves

- Preuves à partir de calculs
- Calculs justifiés par des preuves
- Qu'est-ce qu'une preuve ?

Les infinis

- Infinis potentiel et actuel
- L'infini plus ou moins intuitif
- L'infini en informatique ?
- Un dernier exemple simple

Quelques exemples

- Un compte à rebours infernal
- Hercule contre l'Hydre
- Des preuves illisibles ?
- Conjecture de Kepler

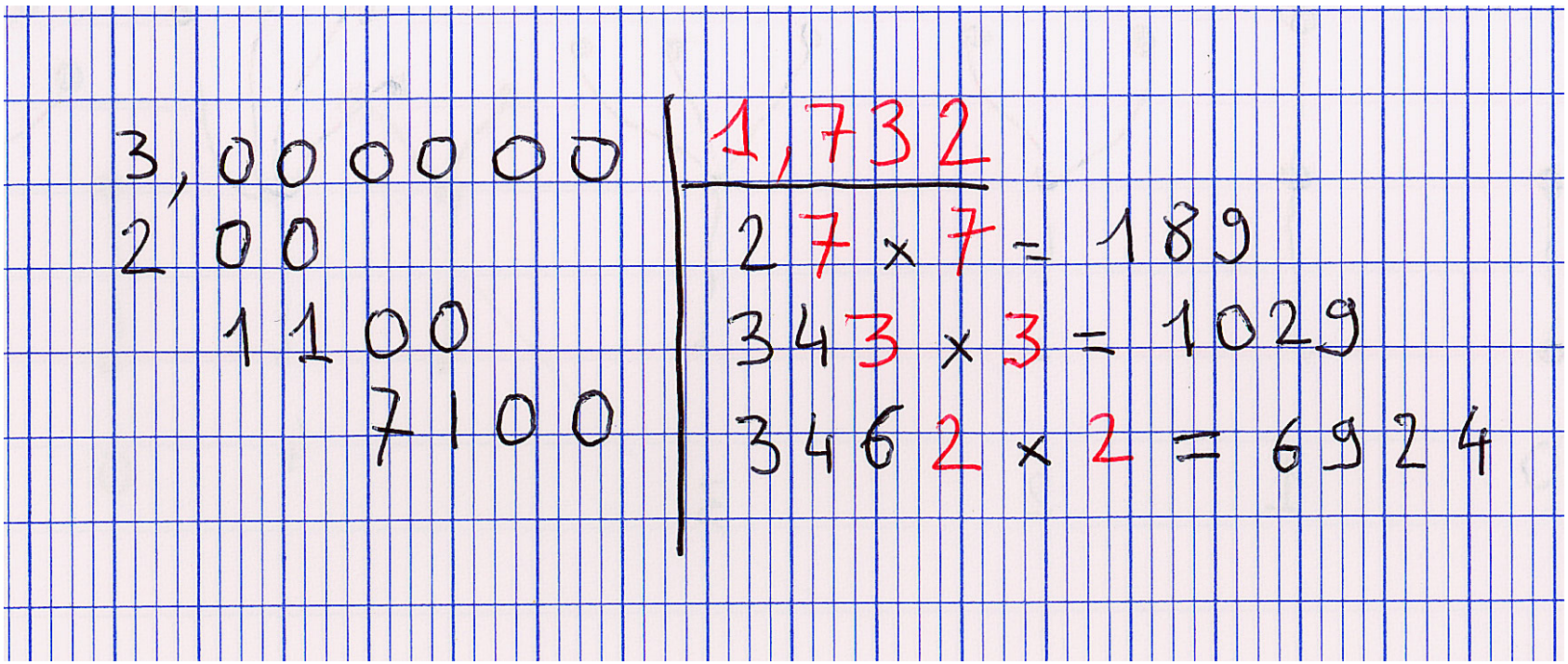
(Logique) Mathématique et Informatique : des univers disjoints ?

Informatique : finitude des machines, des calculs, programmes (algorithmes), automatisme, données concrètes (numériques ou symboliques),

Mathématiques : infinis, abstraction, preuves, impossibilité de tout prouver automatiquement.

Bref, la première impression est celle d'une frontière entre automaticité d'une part et puissance d'expression d'autre part. C'est cette frontière (plutôt perméable) que nous allons un peu explorer.

Prouver des faits à l'aide de calculs



On a prouvé la formule $1,732^2 \leq 3 < 1,7333^2$

```
(sqrt 3)
1.7320508
```

$$1.7320508^2 \leq 3 < 1.7320509^2$$

```
java SumOdd 237
1+3+5+7+9+11+...+469+471+473 = 56169 = 237*237
```

La somme des 237 premiers nombres impairs est égale au carré de 237

Prouver des faits à l'aide de calculs (2)

```
cal 01 2010
     janvier 2010
lu ma me je ve sa di
                 1  2  3
 4  5  6  7  8  9 10
11 12 13 14 15 16 17
18 19 20 21 22 23 24
25 26 27 28 29 30 31
```

On est mercredi

Prouver des faits à l'aide de calculs (3)

```
java Fact 6
10! = 1 * 2 * ... * 6 = 720

java Fact 10
10! = 1 * 2 * ... * 10 = 3628800
```

Prouver des faits à l'aide de calculs (3)

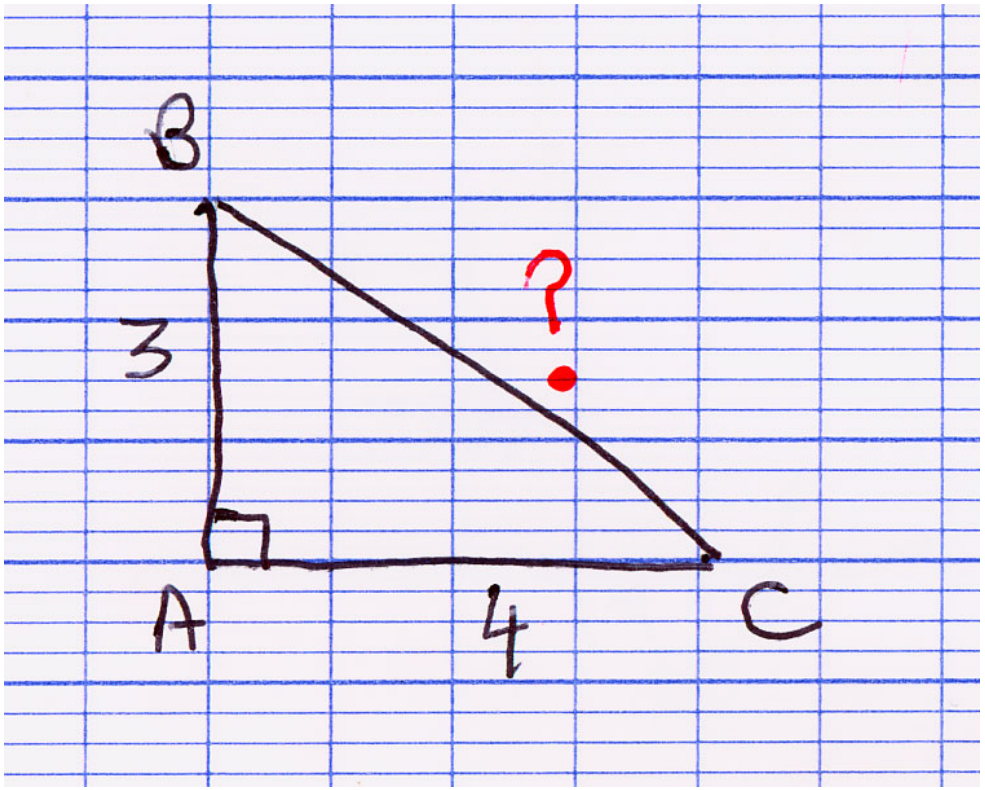
```
java Fact 6
10! = 1 * 2 * ... * 6 = 720

java Fact 10
10! = 1 * 2 * ... * 10 = 3628800

java Fact 60
60! = 1 * 2 * ... * 60 = -8718968878589280256

java Fact 89
89! = 1 * 2 * ... * 89 = 0
```

Réciproque : calculs s'appuyant sur une preuve



$$BC = \sqrt{3^2 + 4^2} = 5 \quad \text{(Théorème de Pythagore)}$$

Exemple de preuve

$$\dfrac{\dfrac{\dfrac{\text{Tout homme est mortel}}{\text{Si Socrate est un homme, alors il est mortel}} \quad \text{Socrate est un homme}}{\text{Socrate est mortel}}}{\text{Il existe au moins un homme mortel}}$$

Une preuve n'est correcte que si l'on applique correctement les *règles d'inférences*, sa portée est conditionnée par l'acceptation des axiomes utilisés. D'un autre côté, un ordinateur peut être utilisé pour *vérifier* une preuve, aussi longue soit-elle.

Attention aux fausses preuves

$$\dfrac{\dfrac{\text{Plus il y a de gruyère, plus il y a de trous} \qquad \text{Plus il y a de trous, moins il y a de gruyère}}{\text{Plus il y a de gruyère, moins il y a de gruyère}}}{\text{Le gruyère n'existe pas}}$$

L(es) infini(s) : un sujet brûlant

Depuis l'antiquité (grecque), l'infini est à la confluence des mathématiques, de la philosophie, et même de la religion.

> *Aucun sens ne perçoit l'infini. Aucun sens ne permet de conclure qu'il existe. . . .*
> *C'est à l'intelligence qu'il appartient de juger et de rendre compte des choses absentes, que le temps et l'espace éloignent de nous.*

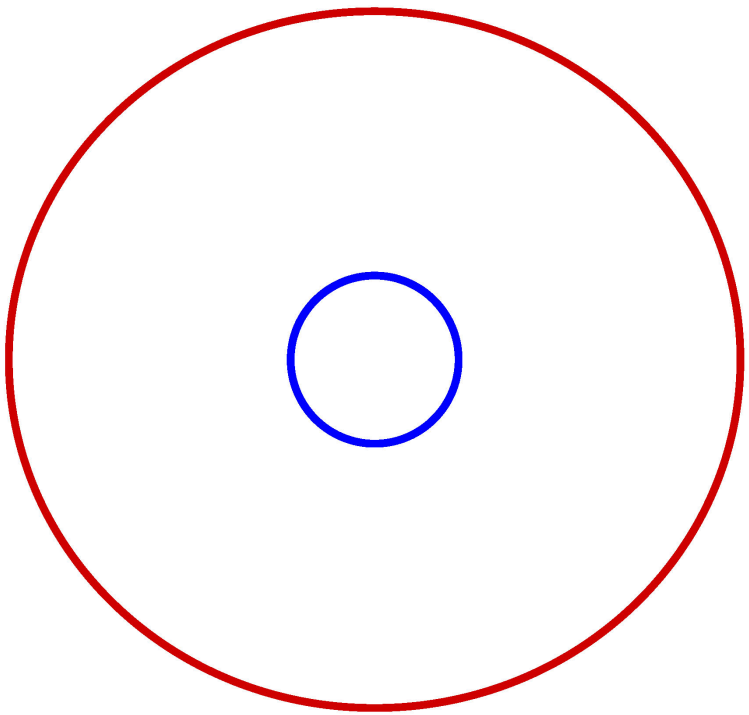
Giordano Bruno : L'infini, l'univers et les mondes

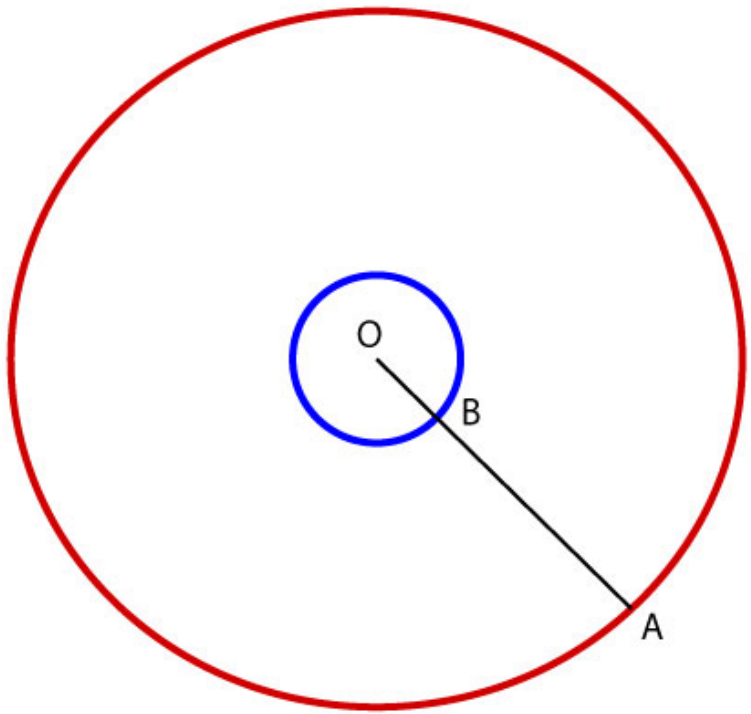
Infinis potentiel et actuel

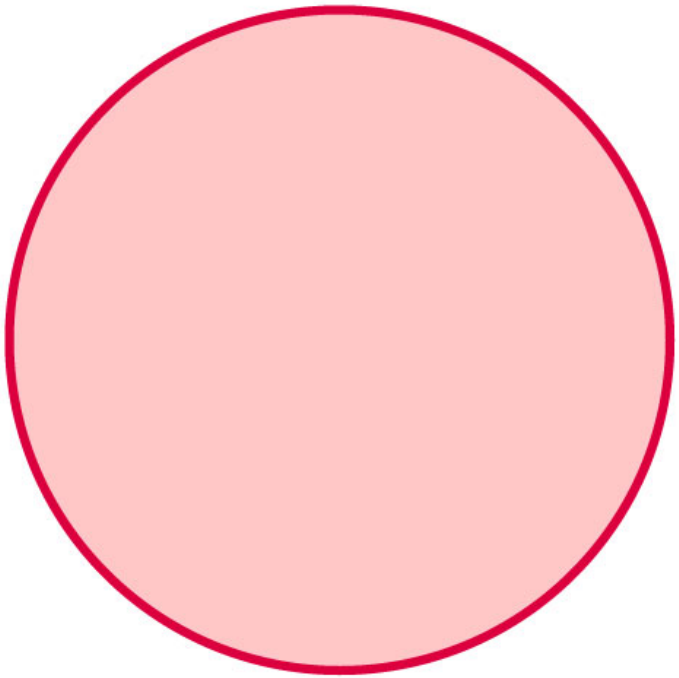
infini potentiel :
- $0, 1, 2, 3, \ldots, n, n+1, \ldots$
- $0, 1, 4, 9, 16, 32, \ldots$
- $2, 3, 5, 7, 11, \ldots$

infini actuel :
- $\mathbb{N} = \{0, 1, 2, 3, \ldots\}$
- $\{2^n | n \in \mathbb{N}\}$
- L'ensemble des nombres premiers

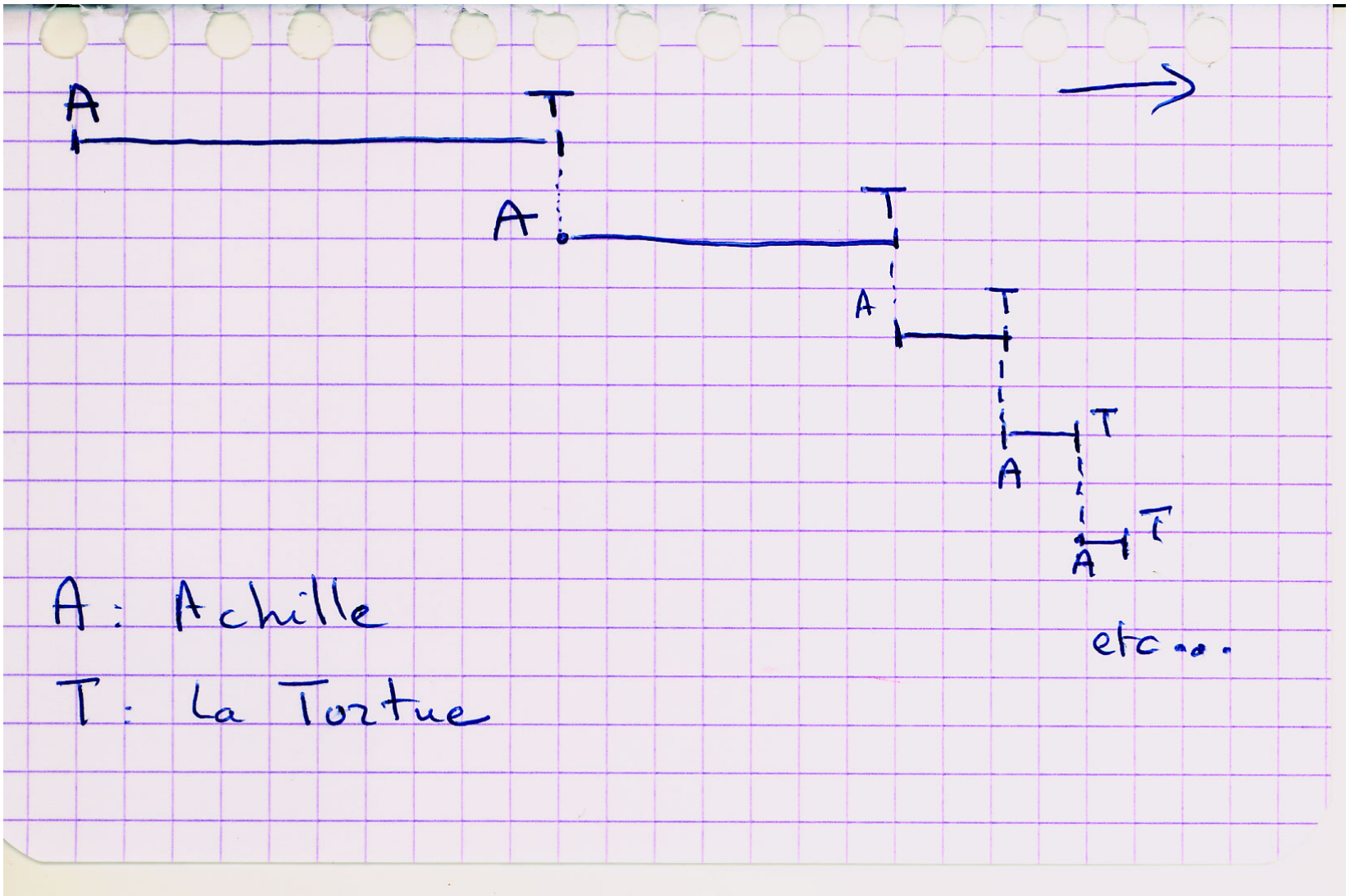
L'infini plus ou moins intuitif




O
B
A



Paradoxe de Zénon : Achille et la tortue



Cantor : *Les* infinis (les *ordinaux*)

$$0, 1, 2, 3, 4, , \ldots$$

Cantor : *Les* infinis (les *ordinaux*)

$$0, 1, 2, 3, 4, , \ldots, n, n+1, \ldots$$

Cantor : *Les* infinis (les *ordinaux*)

$$0, 1, 2, 3, 4, , \ldots, n, n+1, \ldots$$

$$\omega,$$

Cantor : *Les* infinis (les *ordinaux*)

$$0, 1, 2, 3, 4, , \ldots, n, n+1, \ldots$$

$$\omega, \omega + 1,$$

Cantor : *Les* infinis (les *ordinaux*)

$$0, 1, 2, 3, 4, , \ldots, n, n+1, \ldots$$

$$\omega, \omega+1, \omega+2, \ldots$$

Cantor : *Les* infinis (les *ordinaux*)

$$0, 1, 2, 3, 4, , \ldots, n, n+1, \ldots$$

$$\omega, \omega+1, \omega+2, \ldots$$

$$\omega+\omega,$$

Cantor : *Les* infinis (les *ordinaux*)

$$0, 1, 2, 3, 4, , \ldots, n, n+1, \ldots$$
$$\omega, \omega+1, \omega+2, \ldots$$
$$\omega+\omega, \omega+\omega+1, \omega+\omega+2, \ldots$$

Cantor : *Les* infinis (les *ordinaux*)

$$0, 1, 2, 3, 4, , \ldots, n, n+1, \ldots$$

$$\omega, \omega+1, \omega+2, \ldots$$

$$\omega+\omega, \omega+\omega+1, \omega+\omega+2, \ldots$$

$$\ldots$$

$$\omega^{\omega}, \ldots$$

$$\ldots$$

$$\omega^{\omega^{\omega}}, \ldots$$

$$\ldots$$

L'infini en informatique ?

- On rappelle le caractère borné d'une mémoire d'ordinateur (même si la capacité de stockage des machines croît régulièrement).
- Faire entrer des notions infinitaires dans une mémoire finie pose donc de sérieux problèmes ...

L'infini en informatique ?

- On rappelle le caractère borné d'une mémoire d'ordinateur (même si la capacité de stockage des machines croît régulièrement).
- Faire entrer des notions infinitaires dans une mémoire finie pose donc de sérieux problèmes ...
- **Solution :** convertir l'espace en temps, c'est-à-dire de l'infini acteuel en infini potentiel.

Exemple : les nombres premiers. On peut représenter l'ensemble infini des nombres premiers par un programme qui énumère ces nombres.

Exemple : les nombres premiers. On peut représenter l'ensemble infini des nombres premiers par un programme qui énumère ces nombres.

```
java NthPrime 45
197
java NthPrime 57657
714841
```

L'infini en informatique (2)

- En revanche, la certitude de correction (ou toute autre forme de propriété) d'un programme tient plutôt de l'infini mathématique (actuel).
- Une proposition comme « Tel programme est correct » suppose que l'on prend en compte *toutes* les valeurs possibles des paramètres.
- Cette certitude ne peut pas s'acquérir en général par un nombre fini de tests. D'où la nécessité de recourir aux mathématiques et à l'abstraction.

Un dernier exemple simple

$$\begin{aligned} 1 &= 1 \\ 1+3 &= 4 \end{aligned}$$

Un dernier exemple simple

$$\begin{aligned} 1 &= 1 \\ 1+3 &= 4 \\ 1+3+5 &= 9 \end{aligned}$$

Un dernier exemple simple

$$\begin{aligned} 1 &= 1 \\ 1+3 &= 4 \\ 1+3+5 &= 9 = 3^2 \end{aligned}$$

Un dernier exemple simple

$$
\begin{aligned}
1 &= 1 \\
1+3 &= 4 \\
1+3+5 &= 9 = 3^2 \\
1+3+5+7 &= 16 = 4^2
\end{aligned}
$$

Un dernier exemple simple

$$
\begin{aligned}
1 &= 1 \\
1+3 &= 4 \\
1+3+5 &= 9 = 3^2 \\
1+3+5+7 &= 16 = 4^2 \\
1+3+5+7+9 &= 25 = 5^2
\end{aligned}
$$

Un dernier exemple simple

$$
\begin{aligned}
1 &= 1 \\
1+3 &= 4 \\
1+3+5 &= 9 = 3^2 \\
1+3+5+7 &= 16 = 4^2 \\
1+3+5+7+9 &= 25 = 5^2 \\
&\dots \\
1+3+\dots+2n+1 &= (n+1)^2
\end{aligned}
$$

Peut-on vérifier cette propriété à l'aide d'un ordinateur ?

Un dernier exemple simple

$$
\begin{aligned}
1 &= 1 \\
1+3 &= 4 \\
1+3+5 &= 9 = 3^2 \\
1+3+5+7 &= 16 = 4^2 \\
1+3+5+7+9 &= 25 = 5^2 \\
&\dots \\
1+3+\dots+2n+1 &= (n+1)^2
\end{aligned}
$$

Peut-on vérifier cette propriété à l'aide d'un ordinateur ?
Par un simple programme, NON. À l'aide d'outils sophistiqués et une intervention humaine, OUI.

Un exemple de preuve par récurrence

On veut prouver que pour tout entier naturel n, la somme des n premiers nombres impairs est égale à n^2 (notons cette propriété $P(n)$).

Cas de base $1 = 1^2$

Pas d'induction Soit $n \geq 1$ tel que $P(n)$;

1. On a $1 + 3 + ... + 2n - 1 = n^2$;
2. on en déduit
   $1 + 3 + ... + 2n + 1 = n^2 + 2n + 1 = (n + 1)^2$;
3. donc $P(n + 1)$

Un exemple de preuve par récurrence

On veut prouver que pour tout entier naturel n, la somme des n premiers nombres impairs est égale à n^2 (notons cette propriété $P(n)$).

Cas de base $1 = 1^2$

Pas d'induction Soit $n \geq 1$ tel que $P(n)$;

1. On a $1 + 3 + ... + 2n - 1 = n^2$;
2. on en déduit
   $1 + 3 + ... + 2n + 1 = n^2 + 2n + 1 = (n + 1)^2$;
3. donc $P(n + 1)$

CQFD

Notons que l'acceptation du principe de récurrence est un passage de l'infini potentiel (le pas de récurrence) à l'infini actuel (l'énoncé du théorème).

Quelques exemples

Nous allons présenter quelques exemples dans lesquels la relation entre algorithmes et preuves est assez subtile.
Le but est de *convaincre* de certaines propriétés plus ou moins intuitives.

- Un compte à rebours infernal,
- Le combat d'Hercule contre l'Hydre,
- Le théorème des 4 couleurs,
- Les tas d'oranges (ou de boulets de canon)

t	A	B	C
0	2	2	5

t	A	B	C
0	2	2	5
1	2	2	4

t	A	B	C
0	2	2	5
1	2	2	4
2	2	2	3
3	2	2	2
4	2	2	1
5	2	2	0

t	A	B	C
0	2	2	5
1	2	2	4
2	2	2	3
3	2	2	2
4	2	2	1
5	2	2	0
6	2	1	5
7	2	1	4

t	A	B	C
0	2	2	5
1	2	2	4
2	2	2	3
3	2	2	2
4	2	2	1
5	2	2	0
6	2	1	5
7	2	1	4
11	2	1	0
12	1	11	11

t	A	B	C
0	2	2	5
1	2	2	4
2	2	2	3
3	2	2	2
4	2	2	1
5	2	2	0
6	2	1	5
7	2	1	4
11	2	1	0
12	1	11	11

Quand ce compte va-t'il s'arrêter ? D'ailleurs, s'arrêtera-t'il ?

```
java G4Silent
```

```
java G4Silent
```

Pas de réponse

```
java G4Silent
```
Pas de réponse
```
java G4 10000
(t = 10000) 1 15 2287
```

```
java G4Silent
```
Pas de réponse
```
java G4 10000
(t = 10000) 1 15 2287
(t = 10001) 1 15 2286
```

```
java G4Silent
```
Pas de réponse
```
java G4 10000
(t = 10000) 1 15 2287
(t = 10001) 1 15 2286
java G4 1000000
(t = 1000000) 1 8 572863
```

```
java G4Silent
```
Pas de réponse
```
java G4 10000
(t = 10000) 1 15 2287
(t = 10001) 1 15 2286
java G4 1000000
(t = 1000000) 1 8 572863
(t = 1000001) 1 8 572862
```

```
java G4Silent
```
Pas de réponse
```
java G4 10000
(t = 10000) 1 15 2287
(t = 10001) 1 15 2286
java G4 1000000
(t = 1000000) 1 8 572863
(t = 1000001) 1 8 572862
java G4 10000000
(t = 10000000) 1 5 2582911
```

```
java G4Silent
```
Pas de réponse
```
java G4 10000
(t = 10000) 1 15 2287
(t = 10001) 1 15 2286
java G4 1000000
(t = 1000000) 1 8 572863
(t = 1000001) 1 8 572862
java G4 10000000
(t = 10000000) 1 5 2582911
```

Première certitude

La suite va finir par atteindre **0 0 0**.
On utilise le théorème suivant :

> *Toute suite infinie de nombres entiers naturels dont chaque terme est inférieur ou égal au précdent finit par devenir stationnaire.*

$$1256, 1256, 1010, 1009, 1008, 479, 478, 477, 35, 35, 35, \ldots, 35, \ldots$$

Ce résultat (bonne fondation de $\mathbb{N}$) se prouve par récurrence sur le premier item de la suite.

- Si la suite démarre en 0, cette suite ne peut être que $0, 0, \ldots, 0, \ldots$ bien !

- Si la suite démarre en 0, cette suite ne peut être que $0, 0, \ldots, 0, \ldots$ bien !
- Soit n un entier naturel, on suppose que notre propriété est vérifiée pour toutes les suites dont le point de départ est inférieur ou égal à n : On considère une suite démarrant en $n+1$.

- Si la suite démarre en 0, cette suite ne peut être que $0, 0, \ldots, 0, \ldots$ bien !
- Soit n un entier naturel, on suppose que notre propriété est vérifiée pour toutes les suites dont le point de départ est inférieur ou égal à n : On considère une suite démarrant en $n+1$.
  - Si c'est la suite $n+1, n+1, \ldots, n+1, \ldots$, c'est gagné !

- Si la suite démarre en 0, cette suite ne peut être que $0, 0, \ldots, 0, \ldots$ bien !
- Soit n un entier naturel, on suppose que notre propriété est vérifiée pour toutes les suites dont le point de départ est inférieur ou égal à n : On considère une suite démarrant en $n+1$.
  - Si c'est la suite $n+1, n+1, \ldots, n+1, \ldots$, c'est gagné !
  - Sinon, son second terme est $\leq n$, et on applique l'hypothèse de récurrence.Bon !

CQFD

Retour vers notre problème

Supposons que le compteur ne s'arrête jamais.

Retour vers notre problème

Supposons que le compteur ne s'arrête jamais.

t	A	B	C
0	2	2	5
1	2	2	4
...			
12	1	11	11
...			

Le compteur `A` prend la valeur 2, puis 1, puis (peut-être 0). Par le théorème précédent, il va stationner en une valeur textcolorblue$a$.

t	A	B	C
$\ldots$	a	y	$\ldots$
$\ldots$	a	y'	$\ldots$
$\ldots$			
$\ldots$	a	b	z
$\ldots$	a	b	z'
$\ldots$			
$\ldots$	a	b	c
$\ldots$	a	b	c
$\ldots$	a	b	c
$\ldots$			

Lorsque `A` devient fixe, B va se mettre à décroître, et atteindre une valeur fixe b. De même pour `C` et c.
Forcément, $a = b = c = 0$, le compteur s'arrête donc.

t	A	B	C
$\ldots$	a	y	$\ldots$
$\ldots$	a	y'	$\ldots$
$\ldots$			
$\ldots$	a	b	z
$\ldots$	a	b	z'
$\ldots$			
$\ldots$	a	b	c
$\ldots$	a	b	c
$\ldots$	a	b	c
$\ldots$			

Lorsque A devient fixe, B va se mettre à décroître, et atteindre une valeur fixe b. De même pour C et c.
Forcément, $a = b = c = 0$, le compteur s'arrête donc.Oui, mais quand ?

```
    java G4zero
(t = 5) 2 2 0
(t = 11) 2 1 0
(t = 23) 2 0 0
(t = 47) 1 23 0
(t = 95) 1 22 0
(t = 191) 1 21 0
```

```
    java G4zero
(t = 5) 2 2 0
(t = 11) 2 1 0
(t = 23) 2 0 0
(t = 47) 1 23 0
(t = 95) 1 22 0
(t = 191) 1 21 0
```

Eureka ! $11 = 3 \times 4 - 1$, $23 = 3 \times 8 - 1$, $47 = 3 \times 16 - 1$,
$95 = 3 \times 32 - 1$,...

On prouve (par récurrence) que le registre de droite vaut 0 lorsque t est de la forme $3 \times 2^i - 1$.
Plus précisément, on a l'évolution suivante :

t	A	B	C
$3 \times 2^n - 1$	a	$y + 1$	0
$\ldots$			
$3 \times 2^{n+1} - 1$	a	y	0

On prouve (par récurrence) que le registre de droite vaut 0 lorsque t est de la forme $3 \times 2^i - 1$.
Plus précisément, on a l'évolution suivante :

t	A	B	C
$3 \times 2^n - 1$	a	$y+1$	0
$\ldots$			
$3 \times 2^{n+1} - 1$	a	y	0
$\ldots$			
$3 \times 2^{n+y+1} - 1 \ldots$	a	0	0

En utilisant ces lemmes et quelques calculs, on obtient :

t	A	B	C
0	2	2	5

En utilisant ces lemmes et quelques calculs, on obtient :

t	A	B	C
0	2	2	5
$47 = 3 \times 2^4 - 1$	1	23	0

En utilisant ces lemmes et quelques calculs, on obtient :

t	A	B	C
0	2	2	5
$47 = 3 \times 2^4 - 1$	1	23	0
$402653183 = 3 \times 2^{4+23} - 1$	1	0	0

En utilisant ces lemmes et quelques calculs, on obtient :

t	A	B	C
0	2	2	5
$47 = 3 \times 2^4 - 1$	1	23	0
$402653183 = 3 \times 2^{4+23} - 1$	1	0	0
3×2^{27}	1	402653183	402653183

En utilisant ces lemmes et quelques calculs, on obtient :

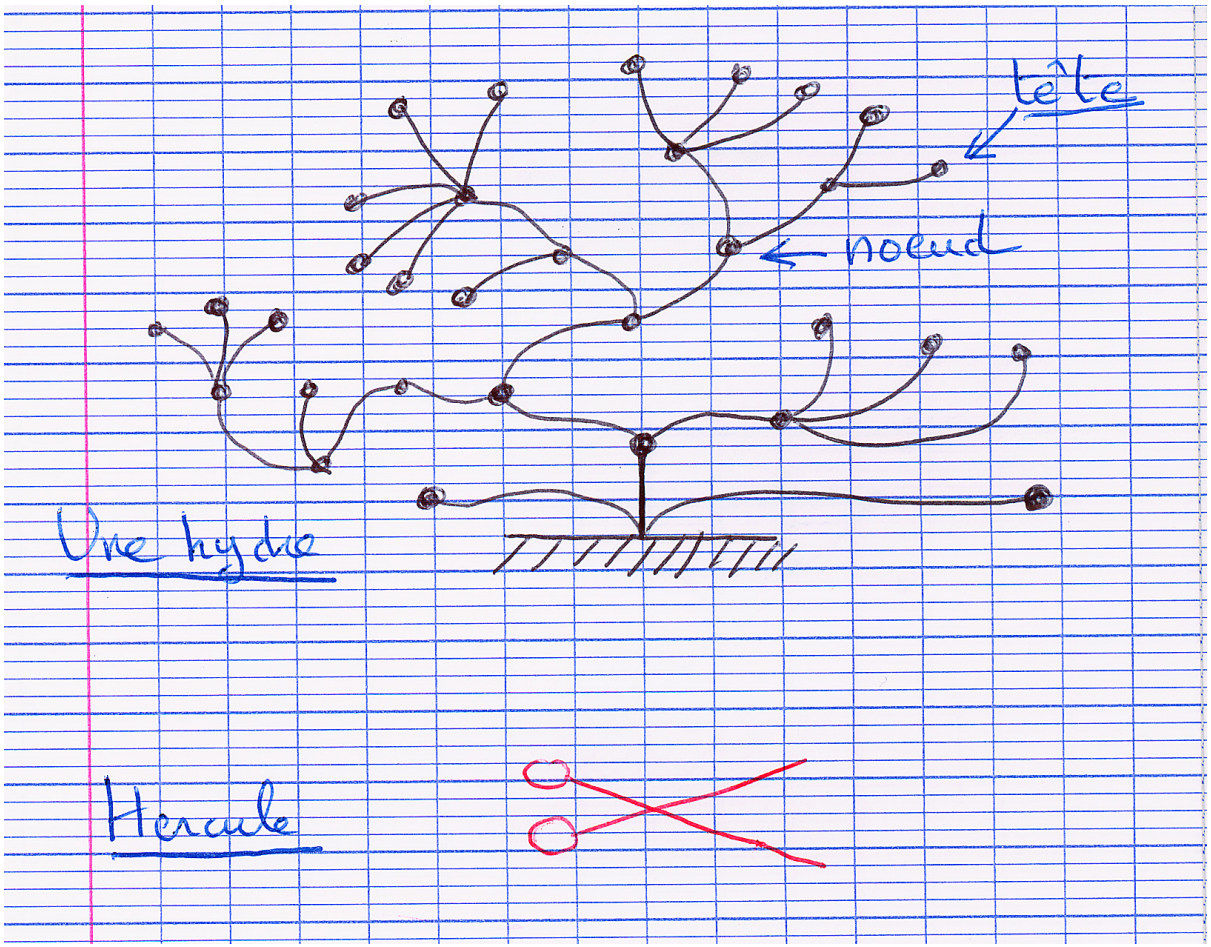
t	A	B	C
0	2	2	5
$47 = 3 \times 2^4 - 1$	1	23	0
$402653183 = 3 \times 2^{4+23} - 1$	1	0	0
3×2^{27}	1	402653183	402653183
$3 \times 2^{28} - 1$	1	402653183	0

En utilisant ces lemmes et quelques calculs, on obtient :

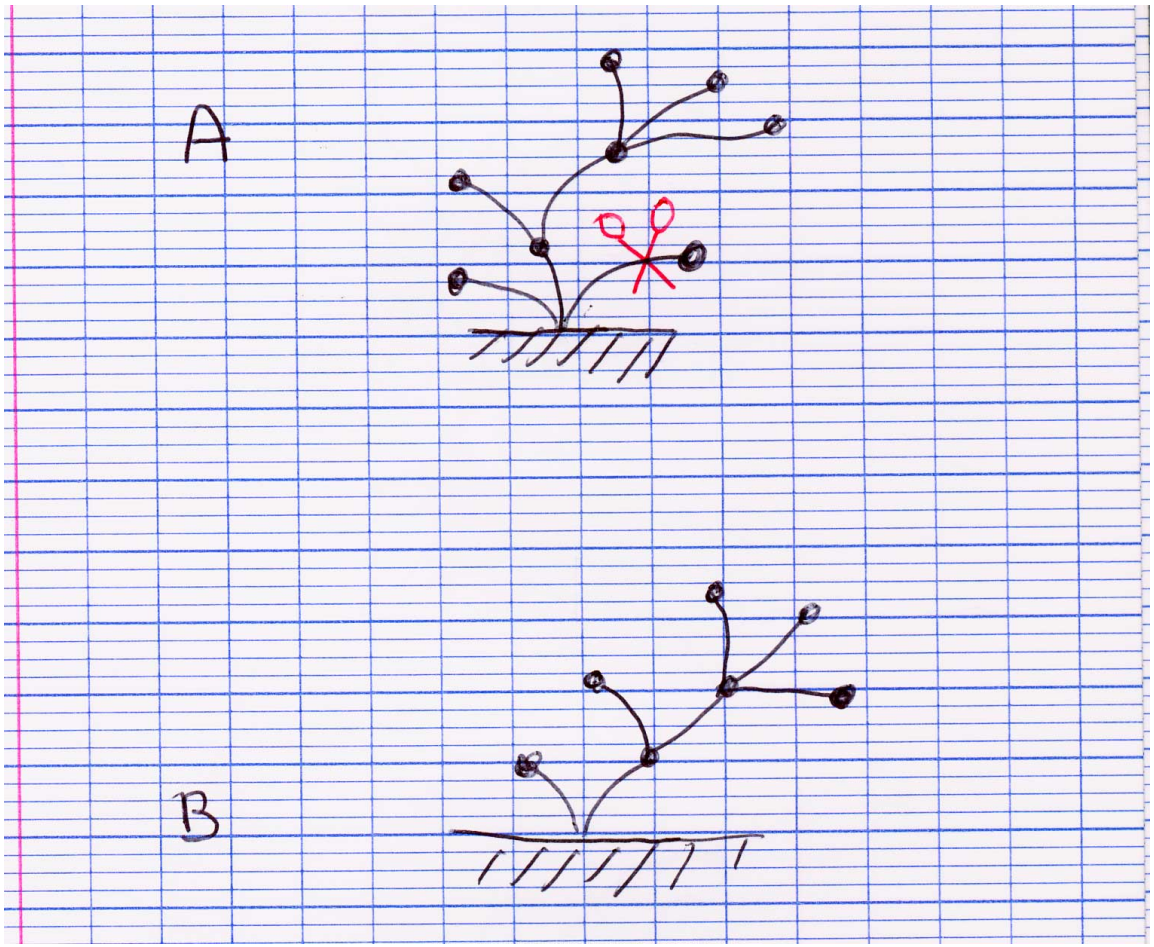
t	A	B	C
0	2	2	5
$47 = 3 \times 2^4 - 1$	1	23	0
$402653183 = 3 \times 2^{4+23} - 1$	1	0	0
3×2^{27}	1	402653183	402653183
$3 \times 2^{28} - 1$	1	402653183	0
$3 \times 2^{28+402653183}$	0	0	0

Le compte à rebours s'arrête après $3 \times 2^{402653211} - 1$ itérations !

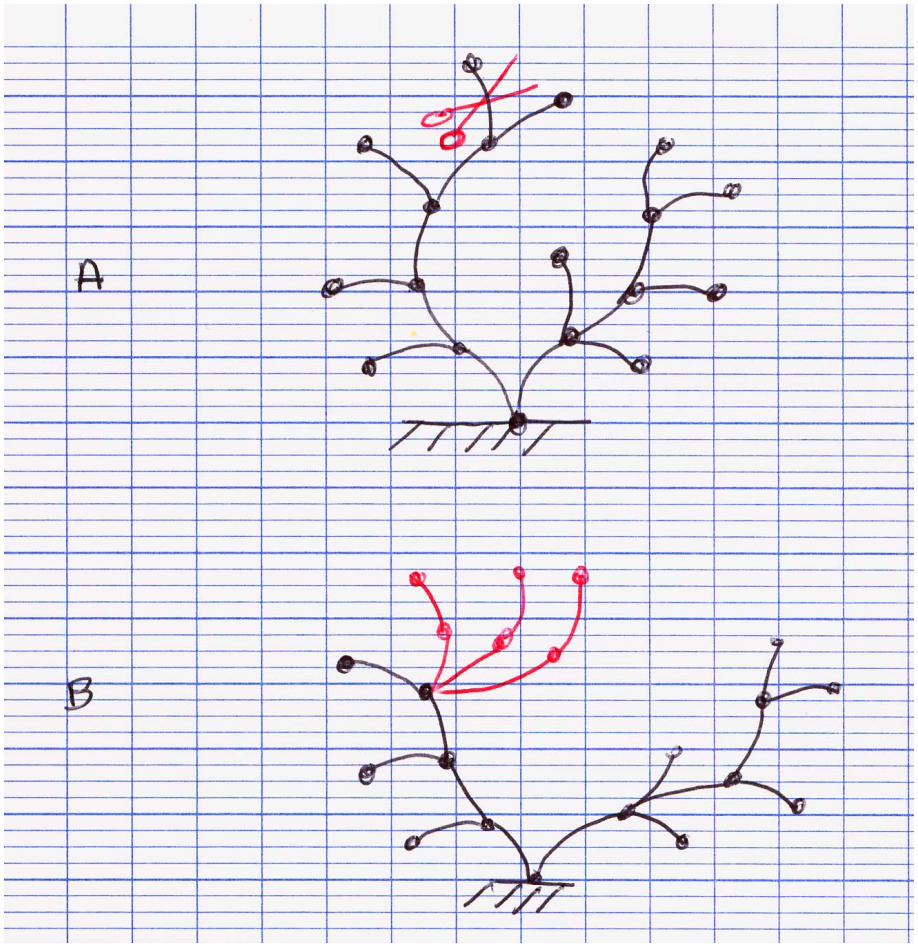
Batailles d'Hydre



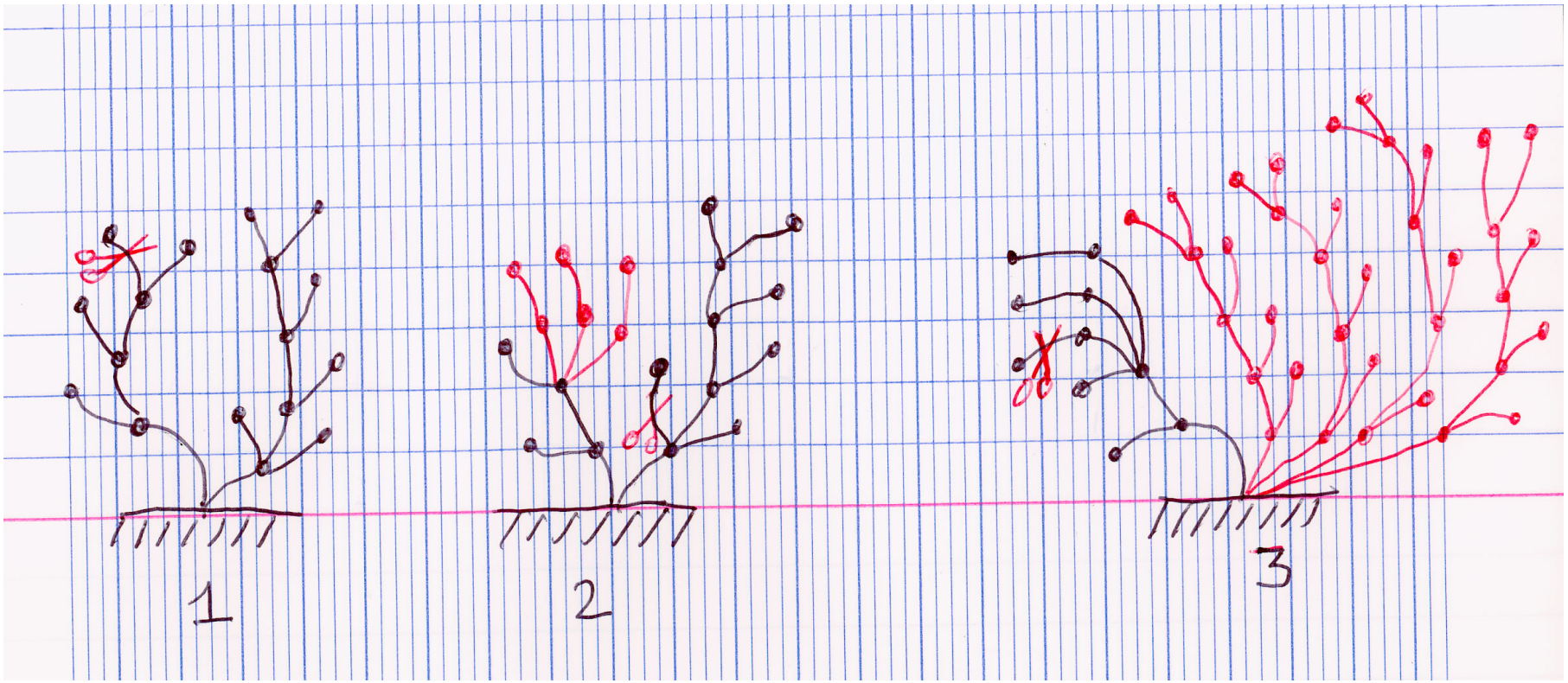
Décapitation : premier cas



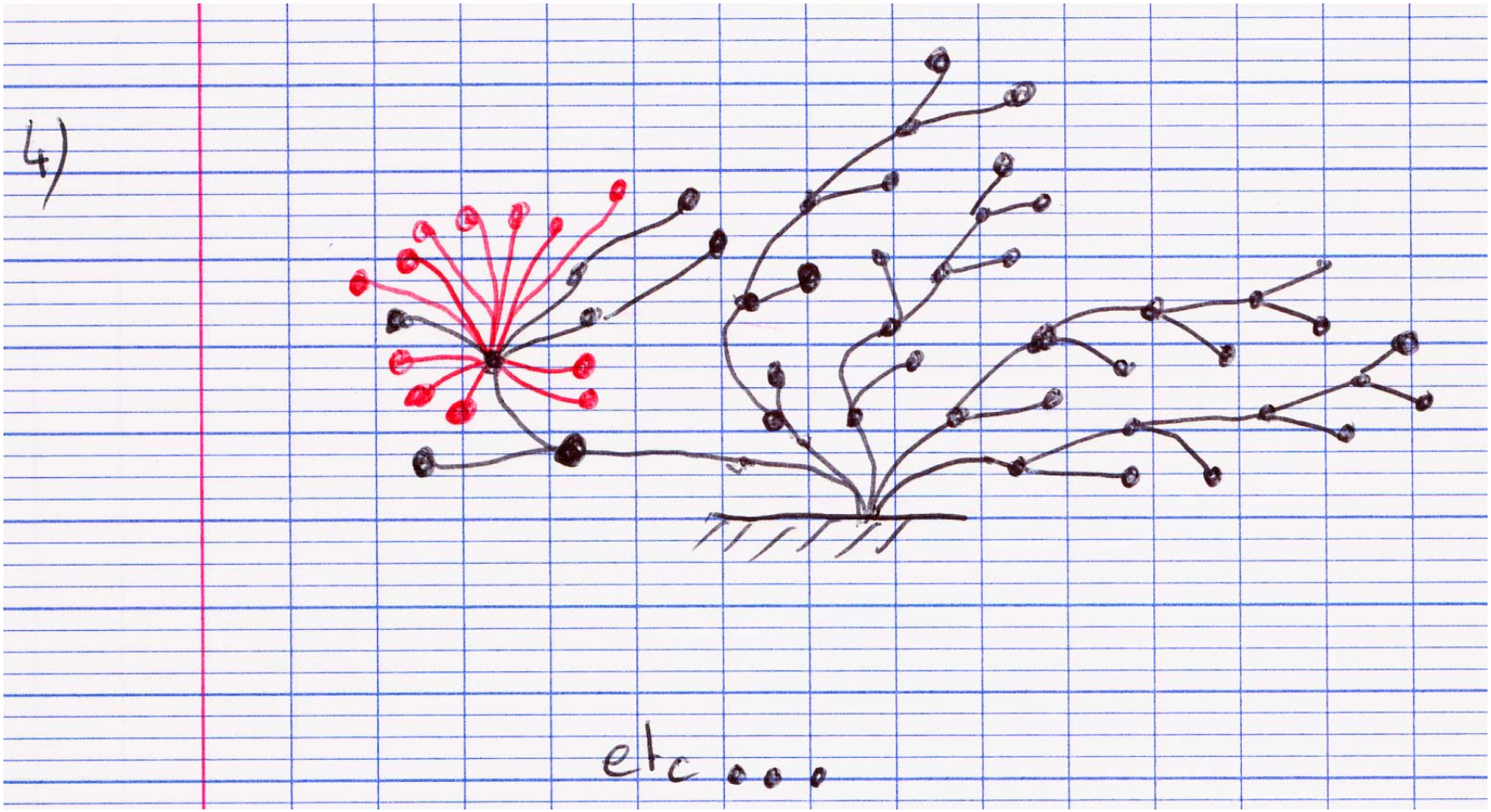
Décapitation : second cas



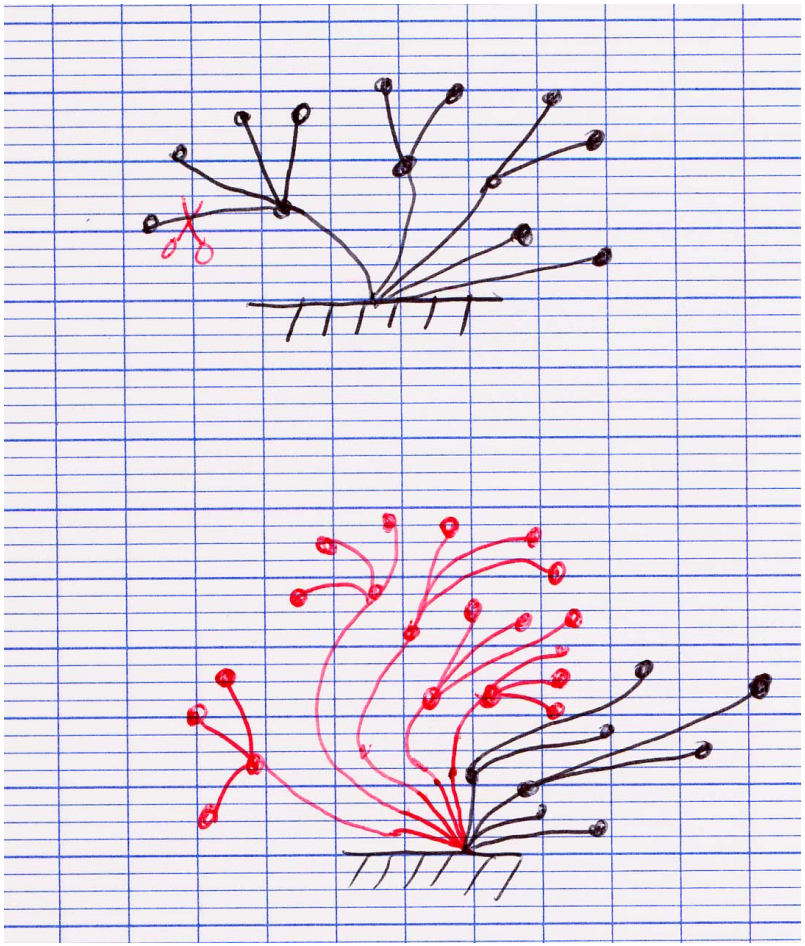
Un combat (début)

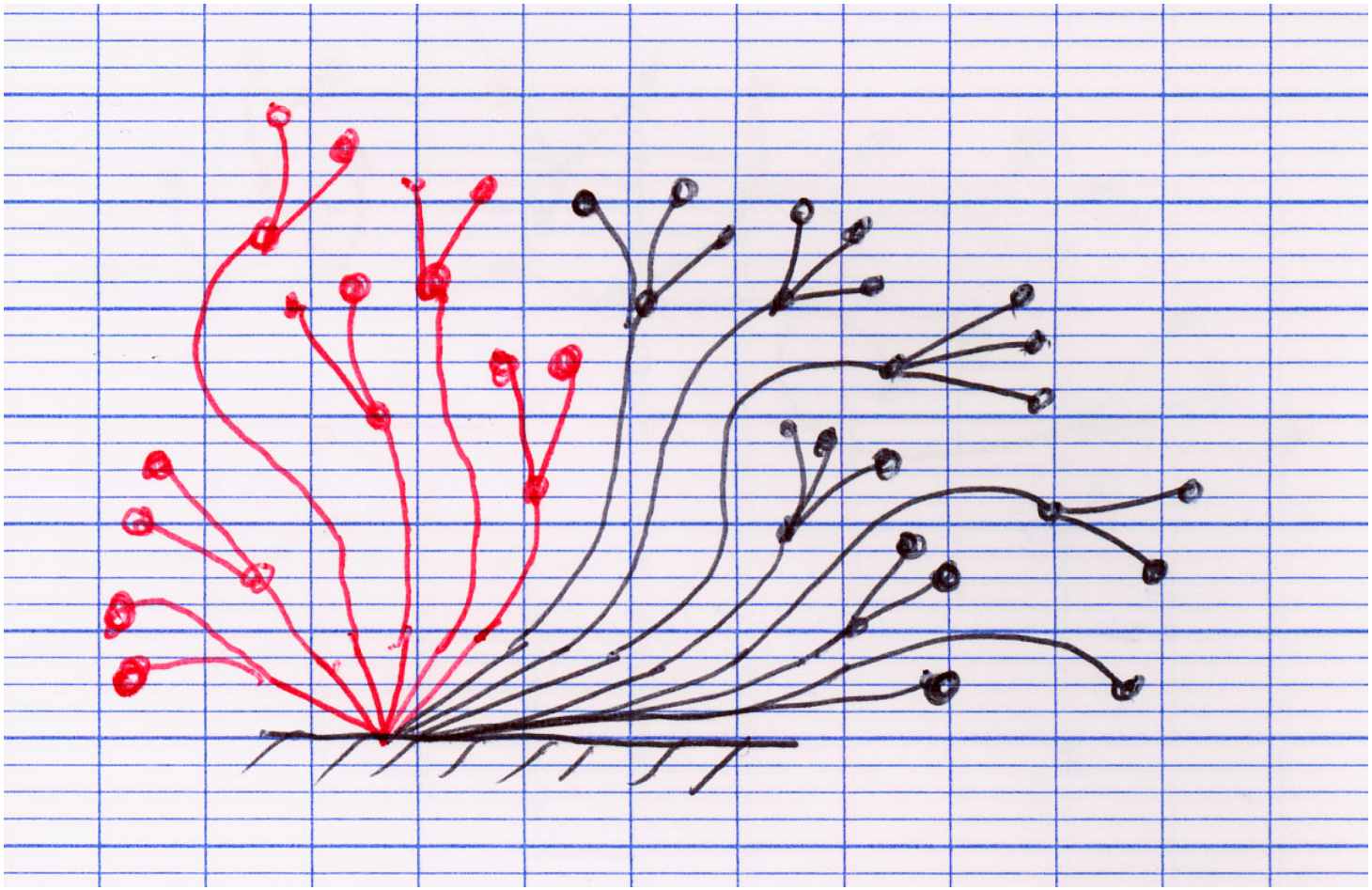


Un combat (suite)

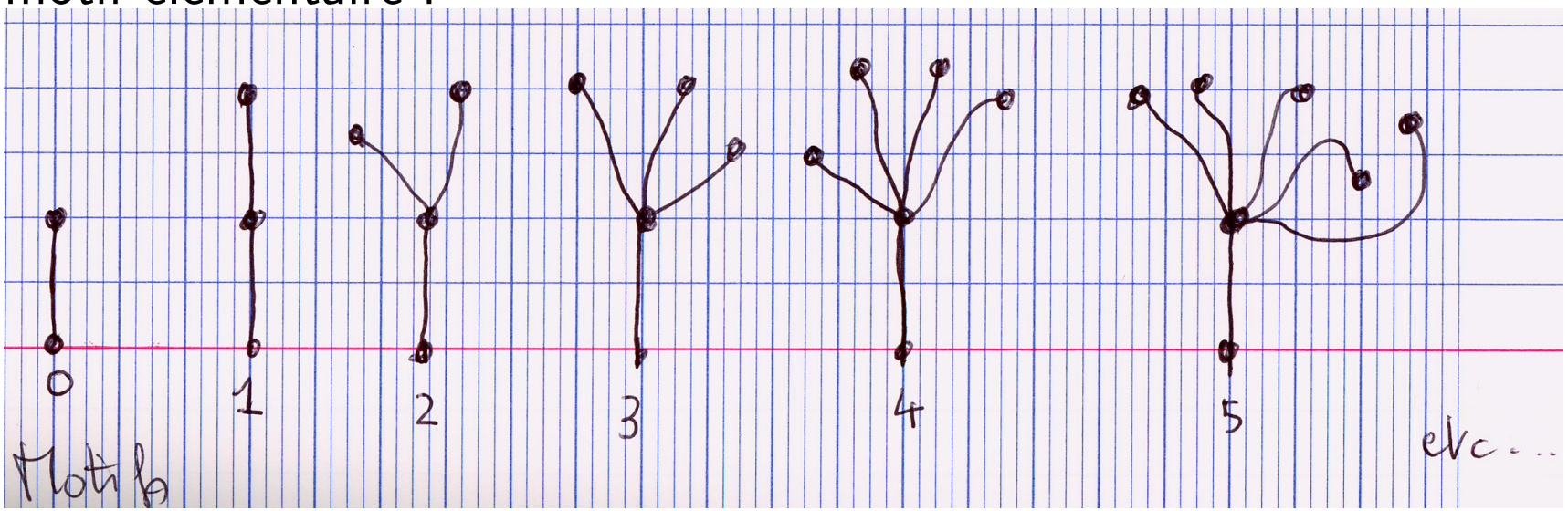


Étude d'une espèce particulière d'hydre (tentacules de longueur ≤ 2)

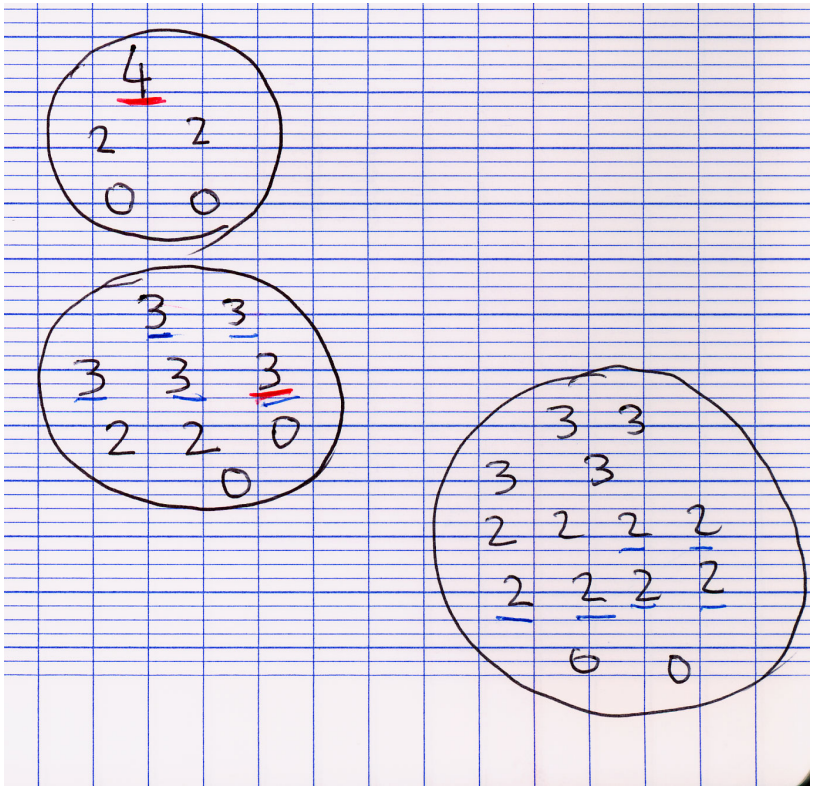




On veut prouver qu'hercule finit toujours par vaincre une telle hydre.
Poure faire on commence par associer un nombre entier à chaque motif élémentaire :

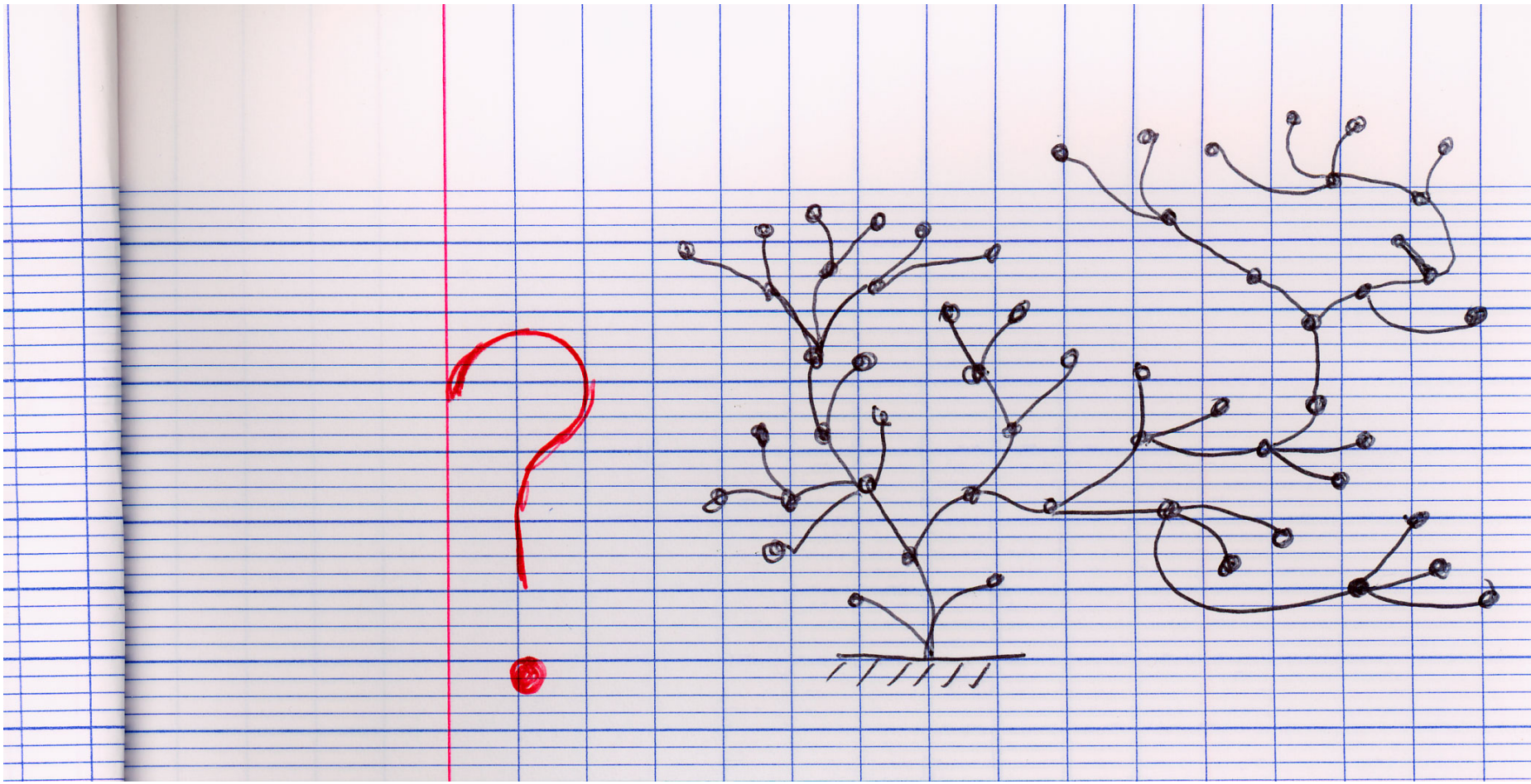


On constitue des « sacs » de nombres :



- L'ensemble des sacs est bien fondé (se prouve en raisonnant sur le plus gros nombre d'un sac et combien de fois il apparaît).
- A chaque tour, le sac associé à l'Hydre diminue.

Et pour une hydre quelconque ?



Remarque pour les mathématiciens

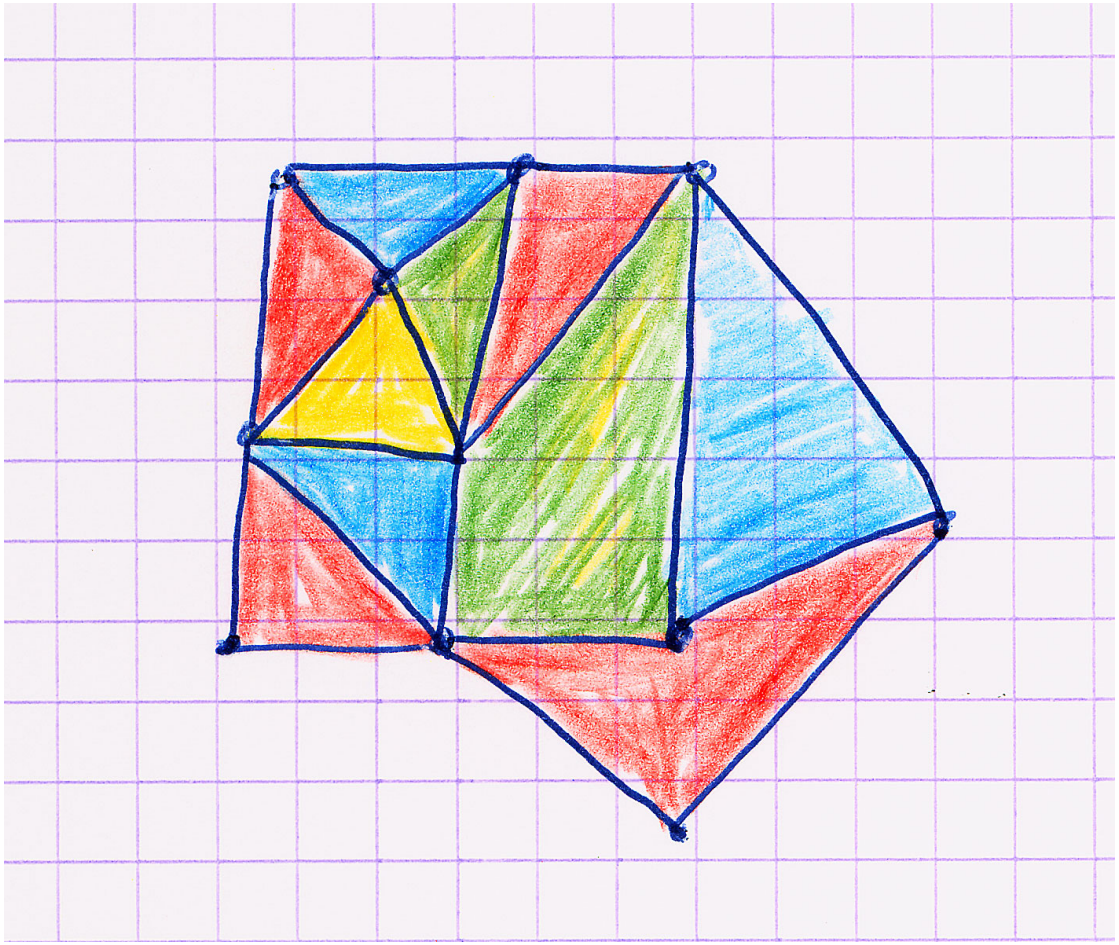
La complexité des trois problèmes de terminaison précédents peut se mesurer à l'aide des ordinaux de Cantor :

Compte à rebours : ω^3

Hydre dont les tentacules sont de longueur ≤ 2 : ω^ω

Hydre quelconque : ϵ_0 (plus grand que ω^ω, ω^{ω^ω}, $\omega^{\omega^{\omega^\omega}}$, $\omega^{\omega^{\omega^{\omega^{\cdots}}}}$)

Des preuves illisibles ?



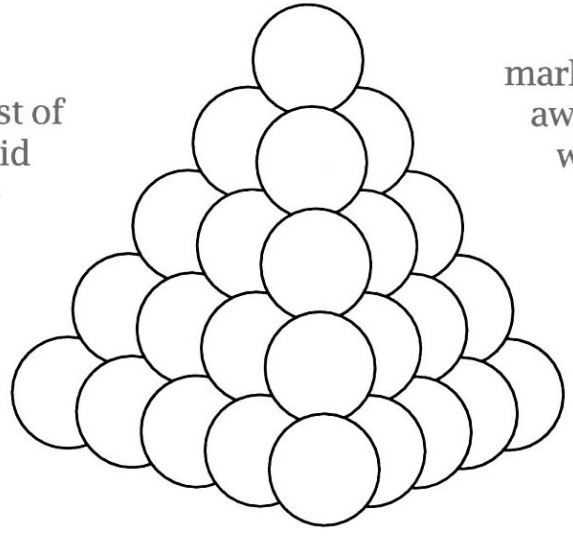
Le théorème des quatre couleurs

- Conjecture proposée en 1852
- Beaucoup de « preuves » fausses, y compris par de grands mathématiciens (de Morgan, Cayley, Hamilton),
- 1976, Appel et Haken :Première preuve sur ordinateur ; des milions de cas traités par programme, justifiés par une analyse manuelle gigantesque (1936 cas, 400 pages),
- 1995, Robertson, Saunders, Seympour et Thomas : simplification en 1995,
- 2005 : Preuve en *Coq* par Gonthier : 60000 lignes de commandes, mais partie critique réduite : 200 lignes de définitions, vérificateur de preuves de *Coq*.

Conjecture de Kepler

- Conjecture de Kepler (1611) : *Il n'existe pas de faÃ§on de ranger des sphères de même diamètre dont la densité soit supérieure à celle du rangement cubique à faces centrées (empilements d'oranges ou de boulets de canon) :* $\frac{\pi}{\sqrt{18}}$
- Contributions de Gauss, Thue (en 2D), Fejes Tóth.

When Hilbert introduced his famous list of 23 problems, he said a test of the perfection of a mathematical problem is whether it can be explained to the first person in the street. Even after a full century, Hilbert's problems have never been thoroughly tested. Who has ever chatted with a telemarketer about the Riemann hypothesis or discussed general reciprocity laws with the family physician?



market. "We need you down here right away. We can stack the oranges, but we're having trouble with the artichokes."

To me as a discrete geometer there is a serious question behind the flippancy. Why is the gulf so large between intuition and proof? Geometry taunts and defies us. For example, what about stacking tin cans? Can anyone doubt that parallel rows of upright cans give the best arrangement? Could some disordered heap of cans

- Preuve par Thomas Hales (1998), dont certaines parties utilisent des calculs sur machine :
  - Énumération de graphes (planaires) pouvant être des contre-exemples à la conjecture. Ces contre-exemples éventuels sont caractérisés par 8 contraintes portant sur les cycles, degrés des sommets, affectation de poids aux faces. Le programme *Java* de Hales énumère 5128 graphes satisfaisant ces contraintes.
  - Programmation linéaire, destinée à vérifier qu'aucun de ces graphes ne constitue un réel contre-exemple.

Le projet Flyspeck

- Vérification de la preuve de Hales par une équipe de 12 arbitres, et conférence consacrée à la preuve : résultat : certitude à 99%,
- Projet Flyspeck : construire une version formelle de la preuve de 1998, principalement à l'aide principalement de *HOL/Light* : mais aussi en *Isabelle/HOL* et *Coq*.

Contribution de Bauer et Nipkow (2005)

- Preuve de la complétude de l'énumération des graphes par le programme de Hales,
- Détection de redondances (2771 graphes au lieu des 5128 énumérés par le programme *Java* de Hales,)
- Détection d'une contrainte absente de la preuve de 98, mais traitée dans le programme *Java*,
- Bilan : la mécanisation de cette preuve a permis de nombreuses simplification dans le calcul de l'énumération des graphes,
- 17000 lignes d'*Isabelle*, 165 minutes de vérification, 2300000 graphes engendrés durant la preuve
- Reste à faire : vérifier que les 2771 graphes ne sont pas des contre-exemples réels à la conjecture de Kepler.